REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE 08 MAI 1945 - GUELMA-
FACULTE DES MATHEMATIQUES, INFORMATIQUE ET SCIENCES DE LA MATIERE
DEPARTEMENT D'INFORMATIQUE

Polycopié de cours

Master 1

Sciences et Technologies de l'Information et de la Communication (STIC)

Dr. Brahim FAROU

- 2022-

Préface

L'informatisation de notre société a considérablement amélioré nos capacités pour générer et collecter des données à partir de diverses sources. Une énorme quantité de données a inondé presque tous les aspects de notre vie. Cette croissance explosive des données stockés ou transitoires a généré un besoin urgent de nouvelles techniques et d'outils automatisés qui peuvent nous aider intelligemment à transformer les vastes quantités de données en informations utiles et connaissances. Cela a conduit à la génération d'une frontière prometteuse et florissante en informatique appelée data mining, et ses diverses applications.

L'exploration de données, aussi communément appelée découverte de connaissances à partir de données (KDD), est la méthode d'extraction de modèles représentant des connaissances implicitement stockées ou capturées dans de grandes bases de données, des entrepôts de données, le Web, d'autres référentiels d'informations massifs, ou flux de données.

Ce manuscrit explore les concepts et les techniques de la découverte des connaissances et de l'exploration de données.

En tant que domaine pluridisciplinaire, le datamining s'appuie sur des travaux dont les statistiques, l'apprentissage automatique, la reconnaissance de formes, les technologies de base de données, la recherche d'informations, les réseaux, les systèmes à base de connaissances et l'intelligence artificielle.

Nous nous concentrons sur les questions relatives à la faisabilité, l'utilité, l'efficacité et évolutivité des techniques pour la découverte de modèles cachés dans de grands ensembles de données.

Ce manuscrit est plutôt une introduction complète à l'exploration de données. Il est utile pour les étudiants en informatique, les développeurs d'applications et les entreprises professionnels, ainsi que les chercheurs impliqués dans l'une des disciplines énumérées précédemment.

Motivation

Actuellement, les données sont collectées quotidiennement avec une masse tellement grande qu'il est devenu difficile de les Analyser avec les anciennes méthodes. Pour répondre à ce besoin, le Data Mining offre des outils sophistiqués qui permettent de découvrir les connaissances à partir des données.

De la donnée Vers l'information

L'informatisation de notre société et le développement rapide des outils de collecte et de stockage de données a fortement contribué dans la croissance exponentielle du volume de données issue principalement des réseaux informatiques, du Web et des périphériques de stockage de données provenant des entreprises, de la société, de la science, de la médecine, etc.

Si on prend par exemple l'entreprise, on observe que presque toutes les entreprises du monde génèrent de gigantesques ensembles de données telles que : les transactions de vente, les descriptions de produits, les promotions des ventes, les profils de l'entreprise, et les commentaires des clients.

Les pratiques scientifiques génèrent également des données d'ordres élevés de manière continue, à travers la télédétection, la mesure des procédés, les expériences scientifiques, les observations techniques et surveillance de l'environnement.

Les réseaux de télécommunications mondiaux ont une part dans cette explosion. Ils transportent des dizaines de pétaoctets de trafic de données tous les jours. L'industrie médicale et de la santé génère d'énormes quantités de données à partir des dossiers médicaux, le suivi des patients et l'imagerie médicale. Des milliards de recherches sur le Web pris en charge par les moteurs de recherche, traitent quotidiennement des dizaines de pétaoctets de données. Les réseaux sociaux sont devenus de plus en plus le générateur numéro un des sources de données par le biais des images, des vidéos et des blogs.

La nécessité de découvrir des informations précieuses à partir de cette masse de données en croissance et de pouvoir les transformer en connaissances organisées a conduit à la naissance du data mining.

Évolution des technologies de l'information

Le Data mining est considéré dans la littérature comme étant le résultat de l'évolution naturelle des technologies de l'information. L'industrie des bases de données et de la gestion des données a évolué avec le développement de plusieurs fonctionnalités critiques, à savoir : la collecte de données, les SGBD ainsi que la gestion et l'analyse de données. En effet, le développement de mécanismes de collecte de données et de création de bases de données a servi de préalable pour le développement ultérieur de mécanismes efficaces pour le stockage, la récupération des données, et le traitement des requêtes et des transactions.

L'histoire du développement des bases de données et des technologies de l'information remonte aux année 70 à partir des systèmes de traitement de fichiers primitifs. La recherche et le développement de systèmes de bases de données sont passés de systèmes hiérarchiques et réseau aux systèmes relationnelles. De plus, les utilisateurs sont devenus expérimentés et accède désormais aux données via les langages de requête et les interfaces utilisateur.

L'analyse avancée des données est apparue à partir de la fin des années 80. Les avancées technologique et les progrès fulgurants de la technologie du matériel informatique ont banalisée les ordinateurs et les équipements de collecte de données et supports de stockage. Cela a donné un coup de fouet aux bases de données et aux informations industrielles, et il a engendré une certaine disponibilité de ces outils pour la gestion des transactions, la récupération d'informations et l'analyse des données.

Les données peuvent désormais être stockés dans de nombreux types de bases de données et de référentiels d'informations. L'entrepôt de données est une conséquence logique de cette diversité ou il s'agit d'un référentiel de plusieurs sources de données hétérogènes organisées sous un schéma, sur un site unique, pour faciliter la prise de décision de gestion. Dans les Entrepôts de données on trouve principalement le nettoyage des données, l'intégration des données et le traitement analytique en ligne (OLAP), c'est-à-dire des techniques d'analyse avec des fonctionnalités telles que la synthèse, la consolidation, et l'agrégation, ainsi que la possibilité d'afficher des informations provenant de différents angles. Bien que les outils OLAP prennent en charge l'analyse multidimensionnelle et la prise de décision, des outils d'analyse de données

supplémentaires sont nécessaires pour une analyse approfondie tels que l'exploration, la classification, le regroupement, la détection des valeurs aberrantes/anomalies et la caractérisation des changements de données dans le temps.

Au-delà des bases et d'entrepôt, on trouve le Web qui contient divers types de bases de données interconnectées et hétérogènes sous forme d'XML.

Tables des matières

Chapitre I : Un Aperçu Générale sur le Processus de Data Mining

Chapitre II : Prétraitement des données

Chapitre III : Exploration de Modèles Fréquents, d'Associations et de Corrélations

Chapitre IV : Classification

Chapitre V : Clustering

Liste des figures

Liste des Tableaux

Chapitre I : Un Aperçu Générale sur le Processus de Data Mining

1. Définition

Data Mining ou l'exploration de données peut être définit de différentes manières à cause de sa nature interdisciplinaire. Le terme exploration de données ne présente pas vraiment tous les principaux composants de l'image. Pour faire référence à l'extraction de minéraux à partir de roches ou de sable, nous disons extractions minérales au lieu d'extraction de roches ou de sable. C'est pour cette raison qu'on dit souvent « extraction de connaissances à partir de données » au lieu de data Mining pour exprimer ce processus en français.

Le data mining désigne le processus d'analyse de volumes massifs de données et du Big Data sous différents angles afin d'identifier des relations entre les data et de les transformer en informations exploitables. Ce dispositif rentre dans le cadre de la Business Intelligence et a pour but d'aider les entreprises à résoudre des problèmes, à atténuer des risques et à identifier et saisir de nouvelles opportunités business.

En outre, de nombreux autres termes ont une signification similaire à l'exploration de données, tels que : l'extraction de connaissances, fouille de données, forage de données, l'analyse de données/de modèles, archéologie et dragage de données.

De nombreuses personnes considèrent l'exploration de données comme un synonyme d'un autre terme couramment utilisé, la découverte de connaissances à partir de données, ou KDD, tandis que d'autres considèrent l'exploration de données comme une simple étape essentielle du processus de découverte des connaissances. Le processus de découverte des connaissances est illustré à la Figure 1.1 comme une séquence itérative des étapes suivantes :

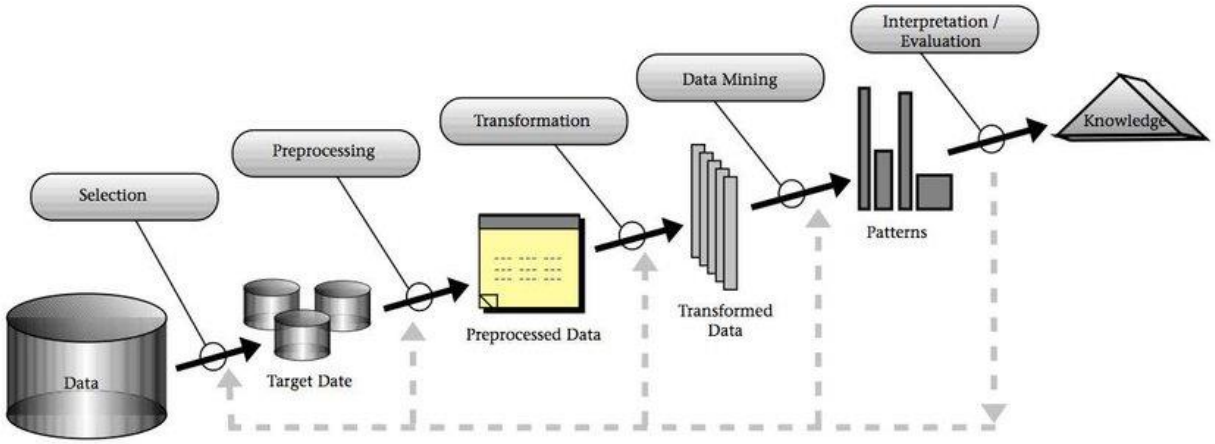

Figure 1.1 Processus de découverte des connaissances [1]

1. Nettoyage des données : supprimer le bruit et les données incohérentes
2. Intégration des données : utiliser lorsque plusieurs sources de données peuvent être combinées
3. Sélection des données : extraction des données pertinentes de la base de données pour la tâche d'analyse.
4. Transformation des données : transformer les données et les consolidées dans des formes appropriées pour l'exploration par le bais des opérations de synthèse ou d'agrégation
5. Data mining : appliquer des méthodes intelligentes pour extraire des modèles de données.
6. Évaluation des modèles : identifier les modèles intéressants qui permettent de représenter les connaissances en se basant sur des mesures d'intérêt.
7. Présentation des connaissances : présenter les connaissances extraites aux utilisateurs par des techniques de visualisation et de représentation des connaissances.

Les étapes de 1 à 4 sont considérées comme des prétraitements de données, où les données sont préparées pour l'exploration. L'étape d'exploration de données peut interagir avec l'utilisateur ou avec une base de connaissances. Les modèles intéressants sont présentés à l'utilisateur et peuvent être stockés en tant que nouvelles connaissances dans la base de connaissances.

La vue précédente montre le Data Mining comme une étape dans le processus de découverte des connaissances. C'est une étape indispensable car elle permet de détecter des modèles cachés

[1] Mathern, Benoît & Mille, Alain & Bellet, Thierry. (2012). Rendre interactive la découverte d'automates à partir de traces d'activités.

pour l'évaluation. Cependant, dans l'industrie, dans les médias et dans le milieu de la recherche, le terme Data Mining est souvent utilisé pour désigner l'ensemble du processus de découverte de connaissances. Les sources de données peuvent inclure des bases de données, des entrepôts de données, le Web, d'autres référentiels d'informations ou des données diffusées dynamiquement dans le système.

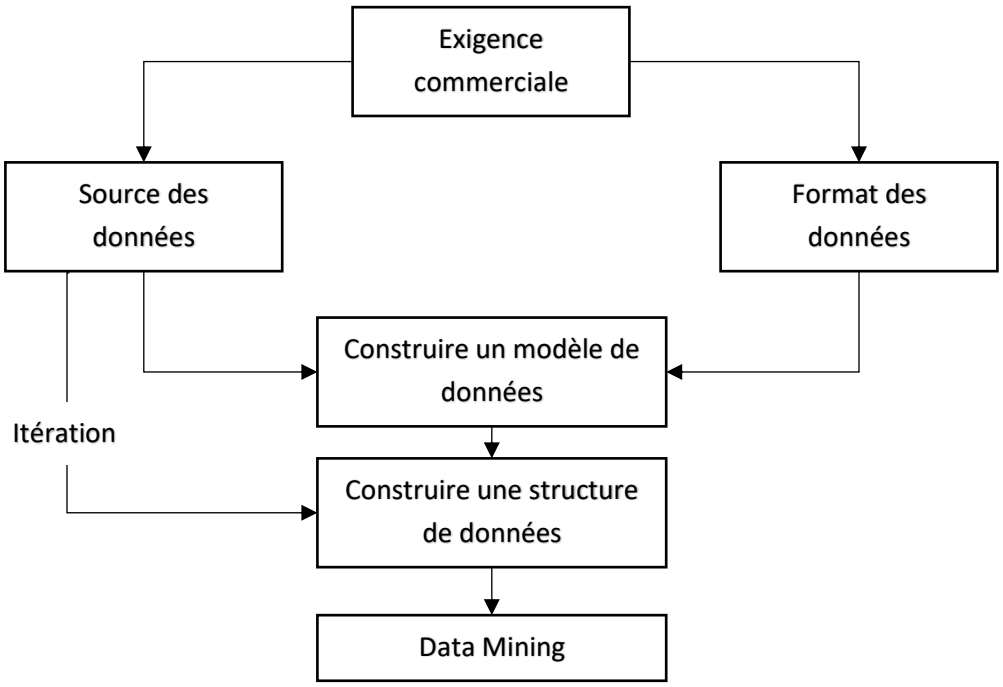

Figure 1.2 Processus du Data Mining

2. Types de données

En tant que technologie générale, l'exploration de données peut être appliquée à tout type de données tant que les données sont significatives pour une application cible. Les formes de données les plus élémentaires sont les données issues des bases de données, les entrepôts de données, et les données transactionnelles.

L'exploration de données peut également être appliquée à d'autres formes de données (par exemple, les flux de données, les données ordonnées/séquentielles, les données graphiques ou en réseau, les données spatiales, les données textuelles, les données multimédias et le WWW).

2.1. Bases de données

Un système de base de données, également appelé système de gestion de base de données (SGBD), consiste en une collection de données interdépendantes, appelée base de données, et un ensemble de programmes logiciels pour gérer et accéder aux données. Les logiciels fournissent des mécanismes pour :

- Définir la structure de la base de données et le stockage de données ;
- Spécifier et gérer l'accès simultané, partagé ou distribué aux données ;
- Assurer la cohérence et la sécurité des informations stockées malgré les pannes du système ou les tentatives d'accès non autorisé.

Une base de données relationnelle est un ensemble de tables, chacune d'entre elles se voyant attribuer un nom unique. Chaque table se compose d'un ensemble d'attributs (colonnes ou champs) et stocke généralement un grand ensemble de tuples (enregistrements ou lignes). Chaque tuple d'une table relationnelle représente un objet identifié par une clé unique et décrit par un ensemble de valeurs d'attribut. Une sémantique modèle de données, tel qu'un modèle de données entité-relation (ER), est souvent construit pour les bases de données relationnelles. Un modèle de données ER représente la base de données comme un ensemble d'entités et leurs relations.

Les données relationnelles sont accessibles par des requêtes de base de données écrites dans un langage de requête relationnel (par exemple, SQL) ou à l'aide d'interfaces utilisateur graphiques. Une requête donnée est transformée en un ensemble d'opérations relationnelles, telles que la jointure, la sélection et la projection, puis optimisée pour un traitement efficace.

Une requête permet de récupérer des sous-ensembles spécifiés de données. Les langages relationnels utilisent également des fonctions d'agrégation telles que sum, avg (moyenne), count, max (maximum) et min (minimum).

Lors de l'exploration de bases de données relationnelles, nous pouvons aller plus loin en recherchant des tendances ou des modèles de données. Par exemple, les systèmes d'exploration de données peuvent analyser les données des clients pour prédire le risque de crédit des

nouveaux clients en fonction de leurs revenus, de leur âge et des informations de crédit précédentes. Les systèmes d'exploration de données peuvent également détecter des écarts, c'est-à-dire des articles avec des ventes qui sont loin de ceux attendus par rapport à l'année précédente. De telles déviations peuvent ensuite faire l'objet d'une enquête plus approfondie. Par exemple, l'exploration de données peut découvrir qu'il y a eu un changement dans l'emballage d'un article ou une augmentation significative du prix.

Les bases de données relationnelles sont l'un des référentiels d'informations les plus couramment disponibles et les plus riches, et constituent donc une forme de données majeure dans l'étude de l'exploration de données.

2.2. Entrepôt de données (DataWarehouses)

La plupart des entreprises de renommé internationale prospère avec des filiales dans le monde entier. Chaque branche possède son propre ensemble de bases de données. Si on veut par exemple fournir une analyse des ventes de l'entreprise par type d'article et par filiale pour le troisième quart il faut consulter toutes les données pertinentes qui sont réparties sur plusieurs bases de données physiquement implantées sur de nombreux sites. Pour simplifier une telle tâche, il est préférable d’utiliser la notion d’entrepôt de données que base de données.

Un entrepôt de données est un référentiel d'informations collectées à partir de plusieurs sources, stockées sous un schéma unifié et résidant généralement sur un seul site. Les entrepôts de données sont construits à travers plusieurs processus (figure 1.3) :

- Nettoyage des données,
- Intégration des données
- Transformation des données
- Chargement des données
- Actualisation périodique des données.

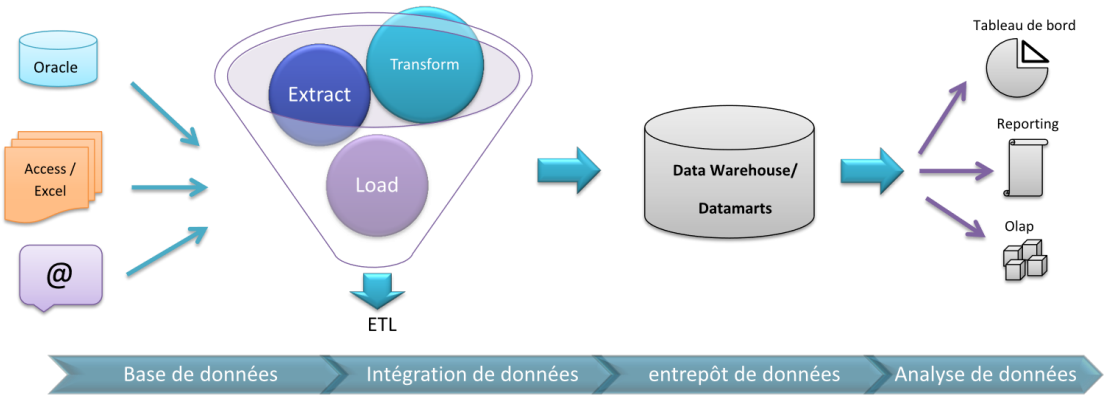

Figure 1.3 Schémas d'intégration de données [2]

Pour faciliter la prise de décision, les données d'un entrepôt de données sont organisées autour de sujets majeurs (par exemple, client, article, fournisseur et activité). Les données sont stockées pour fournir des informations d'un point de vue historique (par exemple au cours des 3, 6 ou 12 derniers mois), et sont généralement résumées. Effectivement, au lieu de stocker les détails de chaque transaction de vente, l'entrepôt de données peut stocker un résumé des transactions par type d'article pour chaque magasin ou, résumé à un niveau supérieur, pour chaque région de vente.

Un entrepôt de données est généralement modélisé par une structure de données multidimensionnelle, appelée cube de données, dans laquelle chaque dimension correspond à un attribut ou à un ensemble d'attributs dans le schéma, et chaque cellule stocke la valeur d'une mesure agrégée (Count, Sum, Avg).

Un cube de données fournit une vue multidimensionnelle des données et permet le précalcule et l'accès rapide aux données résumées.

[2] https://fr.wikiversity.org/wiki/Datawarehouse

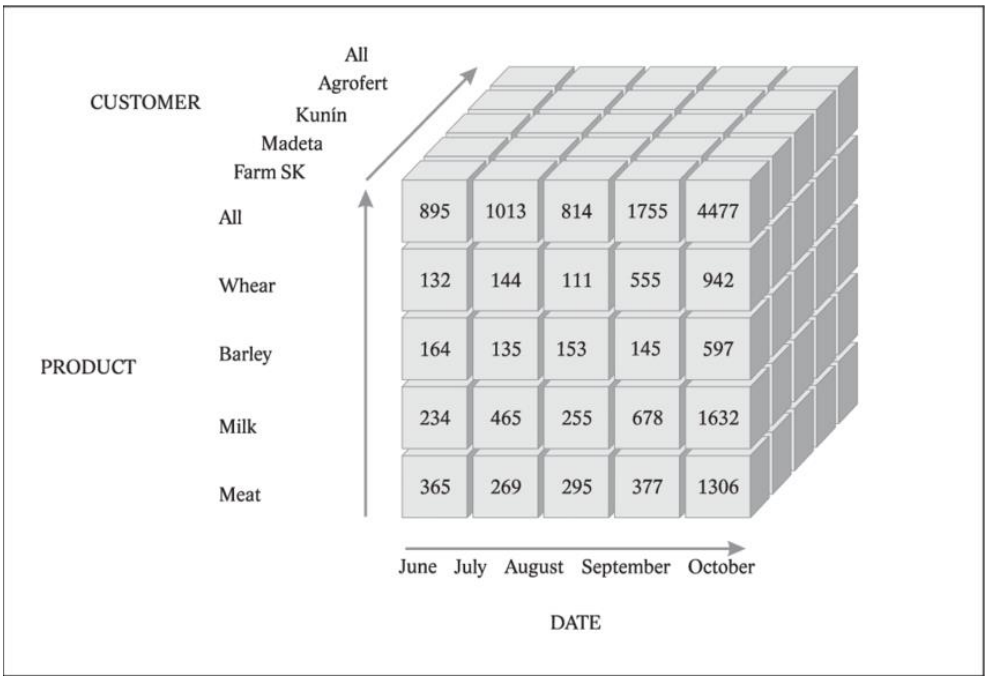

Figure 1.4 Cube de données [3]

En fournissant des vues de données multidimensionnelles et le précalcule des données résumées, les systèmes d'entrepôt de données peuvent fournir une prise en charge inhérente d'OLAP. Les opérations de traitement analytique en ligne s'appuient sur des connaissances préalables concernant le domaine des données étudiées pour permettre la présentation des données à différents niveaux d'abstraction.

De telles opérations s'adaptent aux différents points de vue des utilisateurs. Des exemples d'opérations OLAP incluent l'exploration vers le bas et le cumul, qui permettent à l'utilisateur de visualiser les données à différents degrés de synthèse, comme illustré à la figure 1.4. Par exemple, nous pouvons explorer les données de ventes résumées par trimestre pour voir les données résumées par mois.

Bien que les outils d'entrepôt de données aident à prendre en charge l'analyse de données, des outils supplémentaires pour l'exploration de données sont souvent nécessaires pour une analyse approfondie. L'exploration de données multidimensionnelle permet l'exploration de multiples combinaisons de dimensions à différents niveaux de granularité dans l'exploration de données,

[3] https://www.researchgate.net/figure/Example-of-data-cube_fig5_336473865

et a donc un plus grand potentiel pour découvrir des modèles intéressants représentant les connaissances.

2.3. Données transactionnelles

En général, chaque enregistrement d'une base de données transactionnelle capture une transaction, comme l'achat d'un client, une réservation de vol ou les clics d'un utilisateur sur une page Web. Une transaction comprend généralement un numéro d'identification de transaction unique (trans ID) et une liste des éléments constituant la transaction.

Une base de données transactionnelle peut avoir des tables supplémentaires, qui contiennent d'autres informations liées aux transactions (description de l'article, des informations sur le vendeur, etc.).

Exemple : "Quels sont les articles qui se sont bien vendus ensemble ?"

Ce type d'analyse des données du panier de consommation permet d'augmenter les ventes. Par exemple, étant donné que les imprimantes sont couramment achetées avec les ordinateurs, on peut suggérer d'offrir certaines imprimantes avec une forte remise aux clients qui achètent des ordinateurs sélectionnés, dans l'espoir de vendre plus d'ordinateurs.

Un système de base de données traditionnel n'est pas en mesure d'effectuer une analyse des données du panier de consommation. Cependant, l'exploration de données sur les données transactionnelles peut le faire en explorant des ensembles d'articles fréquents, c'est-à-dire des ensembles d'articles qui sont fréquemment vendus ensemble (extraction des modèles fréquents).

2.4. Autres types de données

De nombreux autres types de données peuvent avoir des formes et des structures polyvalentes et des significations sémantiques assez différentes. Ces types de données peuvent être observés dans de nombreuses applications :

- Données temporelles ou séquentielles : enregistrements historiques, données boursières, séries chronologiques et données de séquence biologique)

- Flux de données : vidéosurveillance et données de capteurs qui sont transmises en continu
- Données spatiales : des cartes
- Données de conception technique : la conception de bâtiments, de composants de système ou de circuits intégrés
- Données hypertextes et multimédias : données textuelles, images, vidéo et audio
- Les graphiques et les données en réseau : les réseaux sociaux et d'information
- Web : un énorme référentiel d'informations largement distribué mis à disposition par Internet.

Ces applications apportent de nouveaux défis, comme la façon de gérer des données portant des structures spéciales (par exemple, des séquences, des arbres, des graphiques et des réseaux) et une sémantique spécifique (telle que l'ordre, l'image, le contenu audio et vidéo, et la connectivité), et comment exploiter les motifs qui portent des structures riches et sémantique. Différents types de connaissances peuvent être extraits de ces types de données.

Exemples :

- Planification des caissiers de banque en fonction du volume de trafic client en exploitant les données temporelles pour les tendances changeantes
- Les données boursières peuvent être extraites pour découvrir des tendances qui pourraient aider à planifier des stratégies d'investissement
- Détecter les intrusions basées sur l'anomalie des flux de messages, qui peuvent être découvertes par le regroupement, la construction dynamique de modèles de flux ou en comparant les modèles fréquents actuels avec ceux d'un moment antérieur.
- Rechercher des modèles qui décrivent les changements dans les taux de pauvreté métropolitaine en fonction des distances entre les villes et les principales autoroutes par le biais données spatiales
- Identifier l'évolution des sujets d'actualité dans le domaine en explorant des données textuelles, telles que la littérature sur l'exploration de données des dix dernières années.

- Evaluer les sentiments des clients et comprendre dans quelle mesure un produit est adopté par un marché en explorant les commentaires des utilisateurs sur les produits
- Identifier des objets et les classer en leur attribuant des étiquettes sémantiques ou des balises à partir de données multimédias
- Détecter des séquences vidéo correspondant à des buts en extrayant les données vidéo d'un match de foot Ball.
- L'exploration Web peut nous aider à en savoir plus sur la distribution d'informations sur le WWW en général, caractériser et classer les pages Web et découvrir la dynamique du Web et l'association et d'autres relations entre les différentes pages Web, les utilisateurs, les communautés et les activités Web.

Il est important de garder à l'esprit que, dans de nombreuses applications, plusieurs types de données sont présents.

Exemple :

- Dans l'exploration Web, il existe souvent des données textuelles et des données multimédias sur des pages Web, des données graphiques telles que des graphiques Web et des données cartographiques sur certains sites Web.
- En bioinformatique, les séquences génomiques, les réseaux biologiques et les structures spatiales 3D des génomes peuvent coexister pour certains objets biologiques.

L'exploration de plusieurs sources de données complexes conduit souvent à des découvertes fructueuses en raison de l'amélioration et de la consolidation mutuelles de ces multiples sources. D'autre part, c'est également difficile en raison des difficultés de nettoyage et d'intégration des données, ainsi que des interactions complexes entre les multiples sources de ces données.

Bien que ces données nécessitent des installations sophistiquées pour un stockage, une récupération et une mise à jour efficaces, elles offrent également un terrain fertile et soulèvent des problèmes de recherche et de mise en œuvre difficiles pour l'exploration de données. L'exploration de données sur de telles données est un sujet avancé.

3. Les types de modèles

En plus des différents types de référentiels de données et d'informations sur lesquels l'exploitation peut être effectuée, les types de modèles peuvent être également exploités.

Il existe un certain nombre de fonctionnalités d'exploration de données :

- La caractérisation et discrimination
- L'exploration de modèles fréquents, d'associations et de corrélations
- Classification et régression
- Analyse de regroupement
- Analyse des valeurs aberrantes

Les fonctionnalités d'exploration de données sont utilisées pour spécifier les types de modèles à trouver dans les tâches d'exploration de données. En général, de tels tâches peuvent être classées en deux catégories : descriptives et prédictives.

- Exploration descriptive : les tâches caractérisent les propriétés des données dans un ensemble de données cible.
- Exploration prédictive : Effectuer une induction sur les données courantes afin de faire des prédictions.

Les fonctionnalités d'exploration de données et les types de modèles qu'elles peuvent découvrir sont décrits au-dessous

3.1. Description du concept : Caractérisation et discrimination

Les entrées de données peuvent être associées à des classes ou à des concepts. Il peut être utile de décrire des classes individuelles et des concepts en termes résumés, concis et pourtant précis. De telles descriptions d'une classe ou d'un concept sont appelés des descriptions de classe/concept. Ces descriptions peuvent être dérivées en utilisant :

- La caractérisation des données : en résumant les données de la classe étudiée appelée souvent la classe cible en termes généraux

- La discrimination des données : par comparaison de la classe cible avec une ou un ensemble de classes comparatives appelées souvent classes
- Hybride : à la fois la caractérisation et la discrimination des données. La caractérisation des données est un résumé des caractéristiques ou caractéristiques générales d'une classe cible de données. Les données correspondant à la classe spécifiée par l'utilisateur sont généralement collectées par une requête.

Il existe plusieurs méthodes pour résumer et caractériser efficacement les données. L'opération de cumul OLAP basée sur le cube de données peut être utilisée pour effectuer une synthèse des données contrôlée par l'utilisateur selon une dimension spécifiée. La technique d'induction orientée attribut peut être utilisée pour effectuer la généralisation des données et caractérisation sans interaction étape par étape de l'utilisateur. Le résultat de la caractérisation des données peut être présenté sous diverses formes (graphiques à secteurs, graphiques à barres, des courbes, des cubes de données multidimensionnels, tableaux, tableaux croisés, etc.). Les descriptions qui en résultent peuvent également être présentées comme relations généralisées ou sous forme de règles (appelées règles caractéristiques).

La discrimination des données est une comparaison des caractéristiques générales des données d'objets de la classe cible par rapport aux caractéristiques générales des objets d'une ou de plusieurs classes sectionnées.

Les classes cibles et sectionnées peuvent être spécifiées par un utilisateur, et les données d'objets peuvent être récupérés via des requêtes de la base de données.

Exemple :

Comparer les caractéristiques générales des produits avec des ventes qui ont augmenté de 5 % la dernière année par rapport à ceux dont les ventes ont diminué d'au moins de 15 % au cours de la même période.

Les méthodes utilisées pour la discrimination des données sont similaires à celles utilisées pour la caractérisation des données. Les formes de présentation des sorties sont similaires à celles des descriptions de caractéristiques, bien que les descriptions de discrimination devraient inclure des

mesures comparatives qui aident à faire la distinction entre la cible et des classes contrastées. Les descriptions des discriminations exprimées sous forme de règles sont appelées règles discriminantes.

3.2. Extraction de modèles fréquents, d'associations et de corrélations

Les modèles fréquents, comme leur nom l'indique, sont des modèles qui se produisent fréquemment dans les données. Il existe de nombreux types de modèles fréquents :

- Des ensembles d'éléments fréquents : fait généralement référence à un ensemble d'éléments qui apparaissent souvent ensemble dans un ensemble de données transactionnelles, par exemple, le lait et le pain, qui sont fréquemment achetés ensemble dans les épiceries par de nombreux clients.
- Des sous-séquences fréquentes : également appelées modèles séquentiels telle que le modèle selon lequel les clients ont tendance à acheter d'abord un ordinateur portable, suivi d'un appareil photo numérique, puis d'une carte mémoire, est un modèle séquentiel (fréquent)
- Des sous-structures fréquentes : peut faire référence à différentes formes structurelles (par exemple, des graphiques, des arbres ou des treillis) qui peuvent être combinées avec des ensembles d'éléments ou des sous-séquences. Si une sous-structure se produit fréquemment, on l'appelle un modèle structuré (fréquent).

L'exploration de modèles fréquents conduit à la découverte d'associations et de corrélations intéressantes au sein des données.

3.3. Classification et régression pour l'analyse prédictive

La classification est le processus de recherche d'un modèle (ou d'une fonction) qui décrit et distingue des classes de données ou des concepts. Le modèle est dérivé sur la base de l'analyse d'un ensemble de données d'apprentissage (c'est-à-dire des objets de données pour lesquels les étiquettes de classe sont connues). Le modèle est utilisé pour prédire l'étiquette de classe des objets pour lesquels l'étiquette de classe est inconnue.

Le modèle dérivé peut être représenté sous diverses formes, telles que des règles de classification (c'est-à-dire des règles SI-ALORS), des arbres de décision, des formules

mathématiques ou des réseaux neuronaux (Figure 1.5). Un test sur une valeur d'attribut de chaque branche représente un résultat du test et les feuilles de l'arbre représentent des classes ou des distributions de classes.

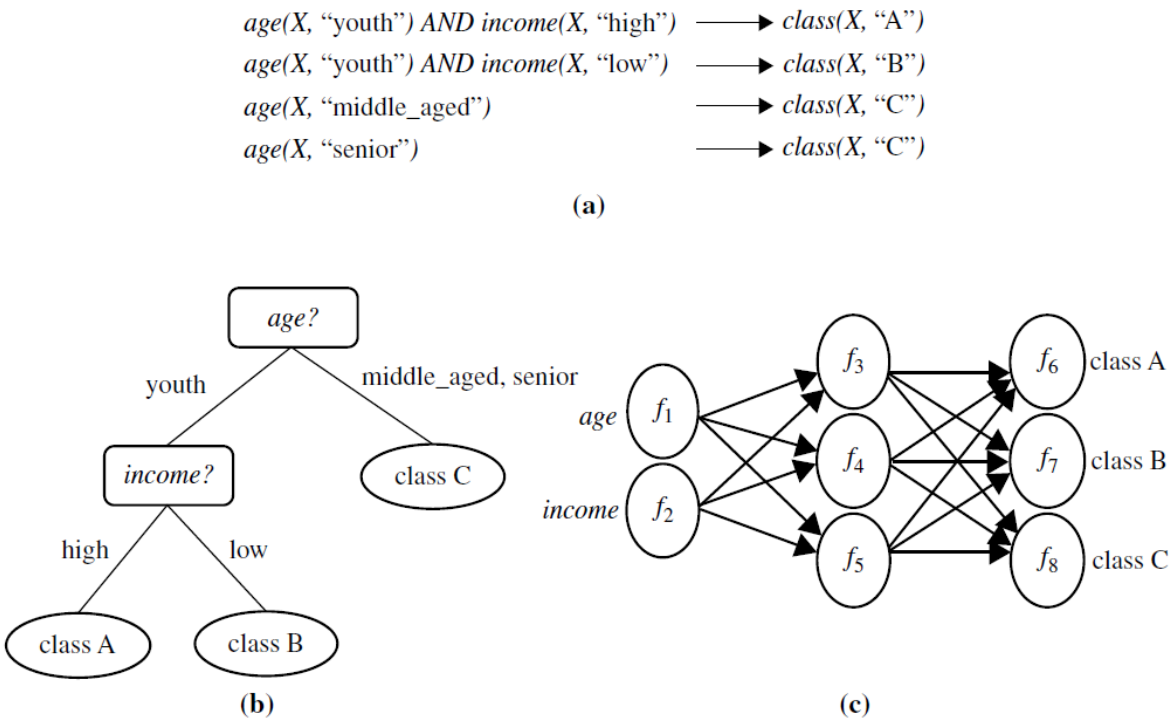

Figure 1.5 Modèles de classification : (a) Des règles SI-ALORS, (b) Un arbre de décision ou (c) Un réseau de neurones.[4]

Les arbres de décision peuvent facilement être converti en règles de classification. Un réseau neuronal, lorsqu'il est utilisé pour la classification, est généralement un ensemble d'unités de traitement de type neurone avec des connexions pondérées entre les unités. Il existe de nombreuses autres méthodes pour construire des modèles de classification, telles que : la classification bayésienne naïve, machines à vecteurs support et classification par les k plus proches voisins.

Alors que la classification prédit des étiquettes catégorielles (discrètes, non ordonnées), la régression modélise des fonctions à valeurs continues. Autrement dit, la régression est utilisée

[4] https://www.brainkart.com/article/Rule-Based-Classification_8323/

pour prédire les valeurs de données numériques manquantes ou indisponibles plutôt que les étiquettes de classe (discrètes). Le terme prédiction fait référence à la fois à la prédiction numérique et à la prédiction d'étiquette de classe. L'analyse de régression est une méthodologie statistique qui est le plus souvent utilisée pour la prédiction numérique, bien que d'autres méthodes existent également. La régression englobe également l'identification des tendances de distribution sur les bases des données disponibles.

La classification et la régression peuvent devoir être précédées d'une analyse de pertinence, qui tente d'identifier les attributs qui sont significativement pertinents pour le processus de classification et de régression. Ces attributs seront sélectionnés pour le processus de classification et de régression. D'autres attributs, qui ne sont pas pertinents, peuvent alors être exclus de l'examen.

3.4. Analyse de cluster

Contrairement à la classification et à la régression, qui analysent des ensembles de données (d'apprentissage) étiquetés par classe, le clustering analyse les objets de données sans consulter les étiquettes de classe. Dans de nombreux cas, les données étiquetées par classe peuvent tout simplement ne pas exister au début. Le clustering peut être utilisé pour générer des étiquettes de classe pour un groupe de données. Les objets sont regroupés selon le principe de maximisation de la similarité intraclasse et de minimisation de la similarité interclasse. C'est-à-dire que des groupes d'objets sont formés de sorte que les objets au sein d'un groupe présentent une grande similitude les uns par rapport aux autres, mais sont plutôt différents des objets dans d'autres groupes. Chaque cluster ainsi formé peut être considéré comme une classe d'objets, à partir de laquelle des règles peuvent être dérivées.

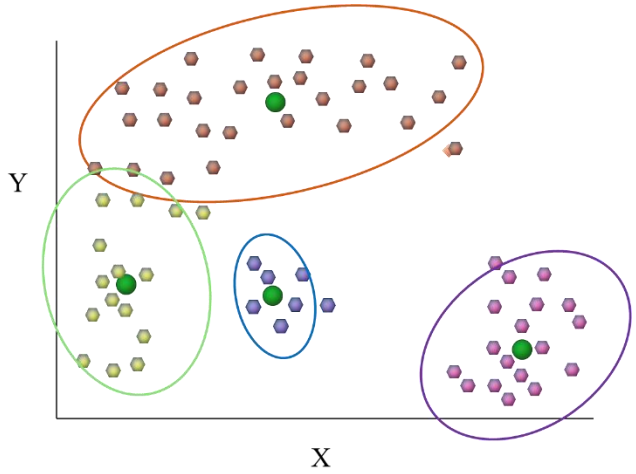

Figure1.6 : Un exemple d'analyse de clustering avec k-means [5]

Le regroupement peut également faciliter la formation de la taxonomie, c'est-à-dire l'organisation des observations dans une hiérarchie de classes qui regroupent des événements similaires.

3.5. Analyse des valeurs aberrantes

Un ensemble de données peut contenir des objets qui ne sont pas conformes au comportement général ou au modèle des données. Ces objets de données sont des valeurs aberrantes. De nombreuses méthodes d'exploration de données rejettent les valeurs aberrantes comme du bruit ou des exceptions. Cependant, dans certaines applications (par exemple, la détection de fraude), les événements rares peuvent être plus intéressants que ceux qui se produisent plus régulièrement. L'analyse des données aberrantes est appelée analyse des valeurs aberrantes ou extraction d'anomalies.

Les valeurs aberrantes peuvent être détectées à l'aide de tests statistiques qui supposent un modèle de distribution ou de probabilité pour les données, ou à l'aide de mesures de distance où les objets éloignés de tout autre cluster sont considérés comme des valeurs aberrantes. Plutôt que d'utiliser des mesures statistiques ou de distance, les méthodes basées sur la densité peuvent identifier les valeurs aberrantes dans une région locale, bien qu'elles semblent normales d'un point de vue de la distribution statistique mondiale.

[5] https://www.freepng.fr/png-6jl8vu/

3.6. Les modèles d'intérêt

Un système d'exploration de données a le potentiel de générer des milliers, voire des millions de modèles ou de règles. Cependant, tous les modèles ne sont pas forcément intéressants. Effectivement, seule une petite fraction des modèles potentiellement générés intéresserait réellement un utilisateur donné.

Cela soulève de sérieuses questions pour l'exploration de données :

3.6.1. Qu'est-ce qui rend un motif intéressant ?

Pour répondre à cette question, un modèle est intéressant s'il est :

- Facilement compris par les humains
- Valide sur des données nouvelles ou de test avec un certain degré de certitude
- Potentiellement utile
- Relativement nouveau.
- S'il valide une hypothèse que l'utilisateur a cherché à confirmer.
- Un modèle intéressant représente la connaissance.

Il existe plusieurs mesures objectives de l'intérêt des modèles. Celles-ci sont basées sur la structure des modèles découverts et les statistiques qui les sous-tendent. Une mesure objective des règles d'association de la forme X =>Y est la prise en charge des règles, représentant le pourcentage de transactions d'une base de données de transactions que la règle donnée satisfait. Il s'agit de la probabilité P(X U Y), où X U Y indique qu'une transaction contient à la fois X et Y, c'est-à-dire l'union des ensembles d'éléments X et Y. Une autre mesure objective des règles d'association est la confiance, qui évalue le degré de certitude de l'association détectée. Ceci est considéré comme la probabilité conditionnelle P(Y/X), c'est-à-dire la probabilité qu'une transaction contenant X contienne également Y. Plus formellement, le support et la confiance sont définis comme

$$support\ (X \Rightarrow Y) = P(X \cup Y)$$
$$Confidence\ (X \Rightarrow Y) = P(Y/X)$$

En général, chaque mesure d'intérêt est associée à un seuil, qui peut être contrôlé par l'utilisateur. Par exemple, les règles qui ne satisfont pas un seuil de confiance de, disons, 50 % peuvent être considérées comme inintéressantes. Les règles inférieures au seuil reflètent probablement du bruit, des exceptions ou des cas minoritaires et ont probablement moins de valeur.

D'autres mesures objectives d'intérêt incluent la précision et la couverture des règles de classification (IF-THEN). En d'autres termes, la précision nous indique le pourcentage de données correctement classées par une règle. La couverture est similaire à la prise en charge, en ce sens qu'elle nous indique le pourcentage de données auquel une règle s'applique. En ce qui concerne la compréhensibilité, nous pouvons utiliser des mesures objectives simples qui évaluent la complexité ou la longueur en bits des modèles extraits.

Bien que les mesures objectives aident à identifier des modèles intéressants, elles sont souvent insuffisantes à moins d'être combinées avec des mesures subjectives qui reflètent les besoins et les intérêts d'un utilisateur particulier.

Exemple :

Les modèles décrivant les caractéristiques des clients qui achètent fréquemment devraient être intéressants pour le responsable marketing, mais peuvent être de peu d'intérêt pour les autres analystes qui étudient la même base de données pour les modèles sur les performances des employés. De plus, de nombreux modèles qui sont intéressants selon des normes objectives peuvent représenter le bon sens et, par conséquent, sont en fait inintéressants.

Les mesures d'intérêt subjectif sont basées sur les croyances des utilisateurs dans les données. Ces mesures trouvent des modèles intéressants si les modèles sont inattendus (contredisant la croyance d'un utilisateur) ou offrent des informations stratégiques sur lesquelles l'utilisateur peut agir. Dans ce dernier cas, ces modèles sont qualifiés d'actionnables.

Exemple :

Des schémas tels que "un grand tremblement de terre suit souvent un groupe de petits tremblements de terre" peuvent être très exploitables si les utilisateurs peuvent agir sur les

informations pour sauver des vies. Les modèles attendus peuvent être intéressants s'ils confirment une hypothèse que l'utilisateur souhaite valider ou s'ils ressemblent à une intuition de l'utilisateur.

3.6.2. Un système d'exploration de données peut-il générer tous les modèles intéressants ?

Cette fait référence à l'exhaustivité d'un algorithme d'exploration de données. Il est souvent irréaliste et inefficace pour les systèmes d'exploration de données de générer tous les modèles possibles. Au lieu de cela, les contraintes fournies par l'utilisateur et les mesures d'intérêt doivent être utilisées pour cibler la recherche.

Pour certaines tâches de minage, telles que l'association, cela est souvent suffisant pour garantir l'exhaustivité de l'algorithme. L'exploration de règles d'association est un exemple où l'utilisation de contraintes et de mesures d'intérêt peut garantir l'exhaustivité de l'exploration.

3.6.3. Le système peut-il générer uniquement des modèles intéressants ?

Cette question est relative à un problème d'optimisation du data mining. Il est hautement souhaitable que les systèmes d'exploration de données ne génèrent que des modèles intéressants. Cela serait efficace pour les utilisateurs et les systèmes d'exploration de données, car aucun des deux n'aurait à rechercher dans les modèles générés pour identifier ceux qui sont vraiment intéressants. Des progrès ont été réalisés dans cette direction ; Cependant, une telle optimisation reste un problème difficile dans l'exploration de données.

Les mesures de l'intérêt des modèles sont essentielles pour la découverte efficace des modèles par les utilisateurs cibles. De telles mesures peuvent être utilisées après l'étape d'exploration de données pour classer les modèles découverts en fonction de leur intérêt, en filtrant ceux qui ne sont pas intéressants.

Plus important encore, de telles mesures peuvent être utilisées pour guider et contraindre le processus de découverte, améliorant l'efficacité de la recherche en supprimant les sous-ensembles de l'espace de modèles qui ne satisfont pas les contraintes d'intérêt prédéfinies.

4. Les technologies utilisées

En tant que domaine fortement axé sur les applications, l'exploration de données a intégré de nombreuses techniques d'autres domaines tels que les statistiques, l'apprentissage automatique, la reconnaissance de formes, les systèmes de bases de données et d'entrepôts de données, la recherche d'informations, la visualisation, les algorithmes, le calcul haute performance et de nombreux domaines d'application (Figure 1.7).

La nature interdisciplinaire de la recherche et du développement en matière d'exploration de données contribue de manière significative au succès de l'exploration de données et de ses nombreuses applications. Dans cette section, nous donnons des exemples de plusieurs disciplines qui influencent fortement le développement des méthodes de fouille de données.

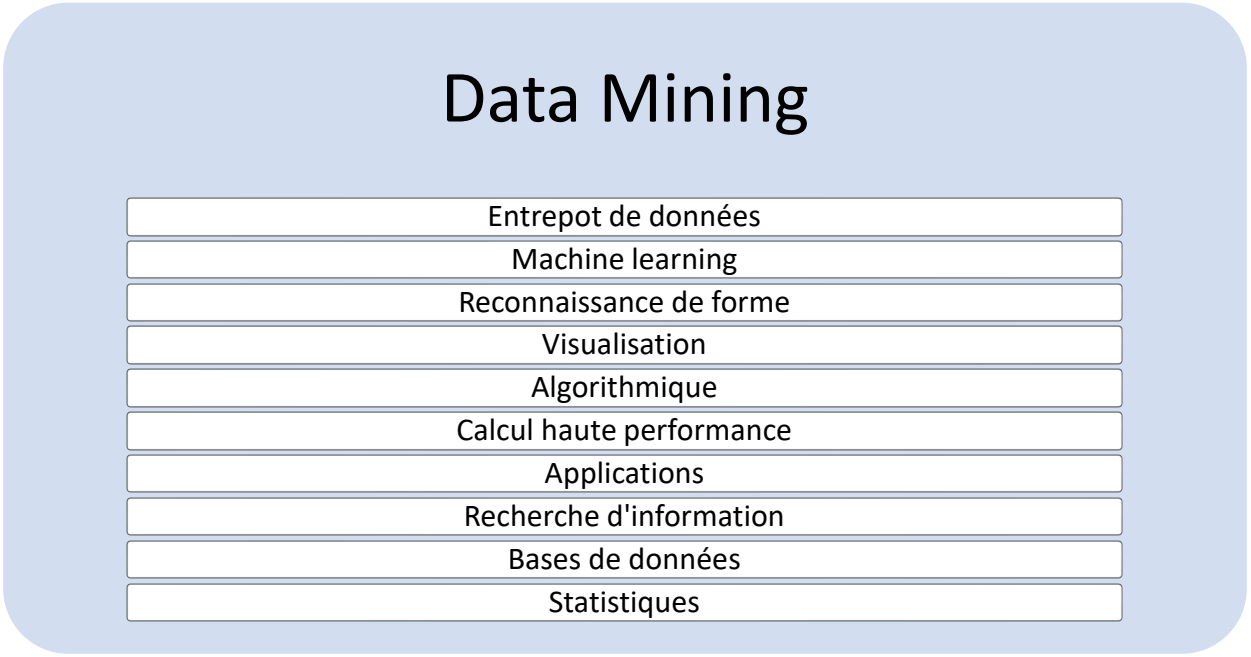

Figure 1.7 les techniques adoptées dans le Data Mining

4.1. Statistiques

La statistique étudie la collecte, l'analyse, l'interprétation ou l'explication et la présentation des données. L'exploration de données à un lien inhérent avec les statistiques.

Un modèle statistique est un ensemble de fonctions mathématiques qui décrivent le comportement des objets d'une classe cible en termes de variables aléatoires et leurs distributions de probabilité associées. Les modèles statistiques sont largement utilisés pour modéliser des données et des classes de données.

Dans les tâches d'exploration de données telles que la caractérisation et la classification des données, des modèles de classes cibles peuvent être construits. En d'autres termes, de tels modèles statistiques peuvent être le résultat d'une tâche d'exploration de données. Alternativement, les tâches d'exploration de données peuvent être construites sur des modèles statistiques. En effet, on peut utiliser des statistiques pour modéliser le bruit et les valeurs de données manquantes. Ensuite, lors de l'exploration de modèles dans un grand ensemble de données, le processus d'exploration de données peut utiliser le modèle pour identifier et gérer les valeurs bruyantes ou manquantes dans les données.

La recherche statistique développe des outils de prédiction et de prévision à l'aide de données et de modèles statistiques. Les méthodes statistiques peuvent être utilisées pour résumer ou décrire une collection de données. Les statistiques sont utiles pour extraire divers modèles à partir de données ainsi que pour comprendre les mécanismes sous-jacents qui génèrent et affectent les modèles. Les statistiques prédictives modélisent les données d'une manière qui tient compte du caractère aléatoire et de l'incertitude dans les observations et sont utilisées pour tirer des conclusions sur le processus ou la population étudiée.

Des méthodes statistiques peuvent également être utilisées pour vérifier les résultats de l'exploration de données. En effet, après l'exploration d'un modèle de classification ou de prédiction, le modèle doit être vérifié par des tests d'hypothèses statistiques. Un test d'hypothèse statistique, appelé également analyse de données de confirmation, prend des décisions statistiques à l'aide de données expérimentales. Un résultat est dit statistiquement significatif s'il est peu probable qu'il se soit produit par hasard. Si le modèle de classification ou de prédiction est vrai, alors les statistiques descriptives du modèle augmentent la solidité du modèle.

L'application de méthodes statistiques à l'exploration de données est loin d'être triviale. Souvent, un défi sérieux est de savoir comment mettre à l'échelle une méthode statistique sur un grand ensemble de données. De nombreuses méthodes statistiques ont une complexité de calcul élevée. Lorsque de telles méthodes sont appliquées sur de grands ensembles de données qui sont également distribués sur plusieurs sites logiques ou physiques, les algorithmes doivent être soigneusement conçus et réglés pour réduire le coût de calcul. Ce défi devient encore plus difficile pour les applications en ligne, telles que les suggestions de requêtes en ligne dans les moteurs de recherche, où l'exploration de données est nécessaire pour gérer en continu des flux de données rapides et en temps réel.

4.2. Apprentissage automatique

L'apprentissage automatique étudie comment les ordinateurs peuvent apprendre ou améliorer leurs performances sur la base de données. C'est un domaine de recherche principal dans lequel les programmes informatiques apprennent automatiquement à reconnaître des modèles complexes et à prendre des décisions intelligentes basées sur des données.

Exemple :

Un problème typique d'apprentissage automatique consiste à programmer un ordinateur afin qu'il puisse reconnaître automatiquement les codes postaux manuscrits sur le courrier après avoir appris à partir d'un ensemble d'exemples. L'apprentissage automatique est une discipline en plein essor. Ici, nous illustrons les problèmes classiques de l'apprentissage automatique qui sont étroitement liés à l'exploration de données.

4.2.1. L'apprentissage supervisé

C'est essentiellement synonyme de classification. La supervision dans l'apprentissage provient des exemples étiquetés dans l'ensemble de données d'apprentissage. Par exemple, dans le problème de reconnaissance du code postal, un ensemble d'images de code postal manuscrites et leurs traductions lisibles par machine correspondantes sont utilisées comme exemples d'apprentissage, qui supervisent l'apprentissage du modèle de classification.

4.2.2. L'apprentissage non supervisé

C’est essentiellement synonyme de clustering. Le processus d'apprentissage n'est pas supervisé car les exemples d'entrée ne sont pas étiquetés par classe. En règle générale, on peut utiliser le clustering pour découvrir des classes dans les données. Cependant, comme les données d'apprentissage ne sont pas étiquetées, le modèle appris ne peut pas nous dire la signification sémantique des clusters trouvés.

4.2.3. L'apprentissage semi-supervisé

C’est une classe de techniques d'apprentissage automatique qui utilisent des exemples étiquetés et non étiquetés lors de l'apprentissage d'un modèle. Les exemples étiquetés sont utilisés pour apprendre des modèles de classe et les exemples non étiquetés sont utilisés pour affiner les frontières entre les classes. Pour un problème à deux classes, on peut considérer l'ensemble des exemples appartenant à une classe comme les exemples positifs et ceux appartenant à l'autre classe comme les exemples négatifs. Dans la figure 1.8, si l'on ne considère pas les exemples non étiquetés, la ligne pointillée est la frontière de décision qui sépare le mieux les exemples positifs des exemples négatifs. En utilisant les exemples non étiquetés, nous pouvons affiner la limite de décision à la ligne continue. De plus, nous pouvons détecter que les deux exemples positifs dans le coin supérieur droit, bien qu'étiquetés, sont probablement du bruit ou des valeurs aberrantes.

4.2.4. L'apprentissage actif

C’est une approche d'apprentissage automatique qui permet aux utilisateurs de jouer un rôle actif dans le processus d'apprentissage. Une approche d'apprentissage actif peut demander à un utilisateur (par exemple, un expert du domaine) d'étiqueter un exemple, qui peut provenir d'un ensemble d'exemples non étiquetés ou synthétisé par le programme d'apprentissage. L'objectif est d'optimiser la qualité du modèle en acquérant activement des connaissances auprès des utilisateurs humains, compte tenu d'une contrainte sur le nombre d'exemples qu'ils peuvent être invités à étiqueter.

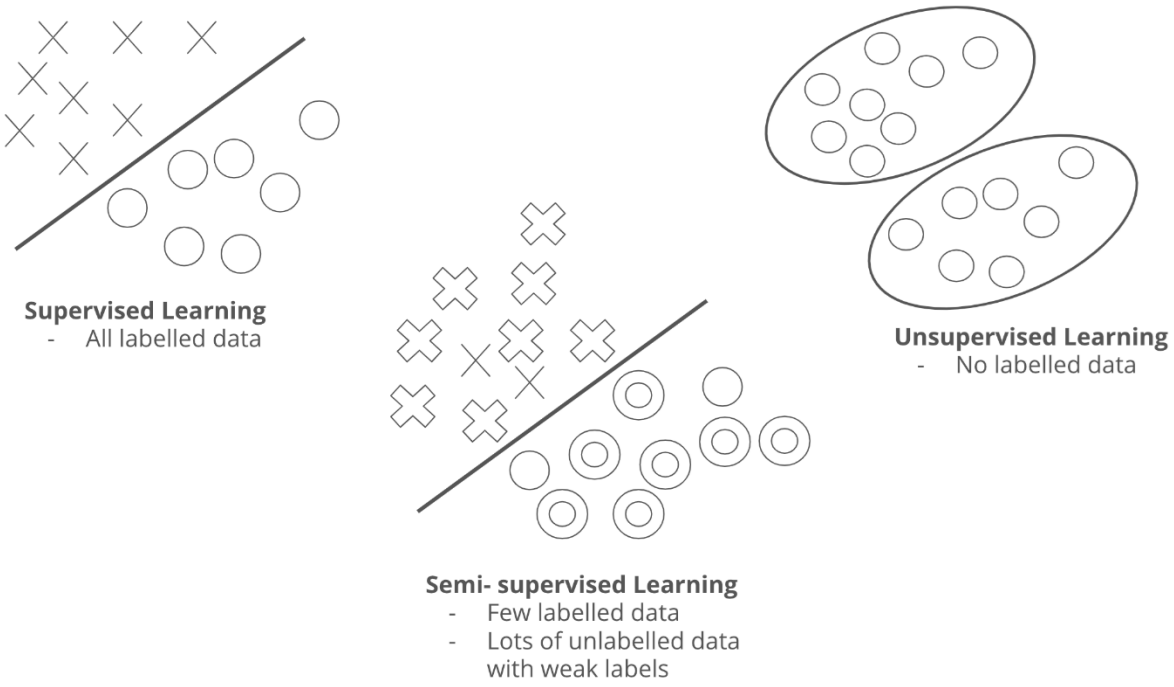

Figure 1.8 Les types d'apprentissage[6]

Il est clair que 'il existe un lien étroit entre le Data Mining et l'apprentissage automatique.

Pour les tâches de classification et de regroupement, la recherche en apprentissage automatique se concentre souvent sur la précision du modèle. En plus de la précision, la recherche sur l'exploration de données met fortement l'accent sur l'efficacité et l'évolutivité des méthodes d'exploration sur de grands ensembles de données, ainsi que sur les moyens de gérer des types de données complexes et d'explorer de nouvelles méthodes alternatives.

4.3. Bases de données et entrepôts de données

La recherche sur les systèmes de bases de données se concentre sur la création, la maintenance et l'utilisation de bases de données pour les organisations et les utilisateurs finaux. En particulier, les chercheurs en systèmes de bases de données ont établi des principes hautement reconnus dans :

- Les modèles de données

[6] https://www.gemeic.top/ProductDetail.aspx?iid=102892086&pr=45.88

- Les langages de requête
- Les méthodes de traitement et d'optimisation des requêtes
- Le stockage des données et les méthodes d'indexation et d'accès.

Les systèmes de bases de données sont souvent bien connus pour leur grande évolutivité dans le traitement de très grands ensembles de données relativement structurés.

De nombreuses tâches d'exploration de données doivent gérer de grands ensembles de données ou même des données de diffusion rapide en temps réel. Par conséquent, l'exploration de données peut faire bon usage des technologies de bases de données évolutives pour atteindre une efficacité et une évolutivité élevées sur de grands ensembles de données. De plus, les tâches d'exploration de données peuvent être utilisées pour étendre la capacité des systèmes de bases de données existants afin de répondre aux exigences d'analyse de données sophistiquées des utilisateurs avancés.

Les systèmes de base de données récents ont construit des capacités d'analyse de données systématiques sur les bases de données en utilisant des installations d'entreposage de données et d'exploration de données. Un entrepôt de données intègre des données provenant de plusieurs sources et de différentes échéances. Il consolide les données dans un espace multidimensionnel pour former des cubes de données partiellement matérialisés. Le modèle de cube de données facilite non seulement l'OLAP dans les bases de données multidimensionnelles, mais favorise également l'exploration de données multidimensionnelle.

4.4. La recherche d'information

La recherche d'informations (RI) est la science de la recherche de documents ou d'informations dans des documents. Les documents peuvent être du texte ou du multimédia et peuvent résider sur le Web. La différence entre les systèmes traditionnels de recherche d'informations et les systèmes de bases de données réside dans le fait que la recherche d'informations suppose que les données recherchées ne sont pas structurées et que les requêtes sont formées principalement par des mots clés, qui n'ont pas de structures complexes (contrairement aux requêtes SQL dans les systèmes de bases de données).

Les approches typiques en recherche d'information adoptent des modèles probabilistes. Par exemple, un document texte peut être considéré comme un sac de mots, c'est-à-dire un multiensemble de mots apparaissant dans le document. Le modèle du document est la fonction de densité de probabilité qui génère le sac de mots dans le document. La similarité entre deux documents peut être mesurée par la similarité entre leurs modèles linguistiques correspondants.

De plus, un sujet dans un ensemble de documents texte peut être modélisé comme une distribution de probabilité sur le vocabulaire, ce que l'on appelle un modèle de sujet. Un document texte, qui peut impliquer un ou plusieurs sujets, peut être considéré comme un mélange de plusieurs modèles de sujets.

En intégrant des modèles de recherche d'informations et des techniques d'exploration de données, nous pouvons trouver les grands thèmes d'une collection de documents et, pour chaque document de la collection, les grands thèmes concernés.

De plus en plus de données textuelles et multimédias ont été accumulées et rendues disponibles en ligne en raison de la croissance rapide du Web et des applications telles que :

- Les bibliothèques numériques
- Les gouvernements numériques
- Les systèmes d'information sur les soins de santé.

Leur recherche et leur analyse efficaces ont soulevé de nombreux problèmes difficiles dans l'exploration de données. Par conséquent, l'exploration de texte et l'exploration de données multimédia, intégrées aux méthodes de recherche d'informations, sont devenues de plus en plus importantes.

4.5. La reconnaissance de formes

En termes simples, la reconnaissance de formes donne à une entreprise un avantage stratégique, ce qui la rend capable d'évoluer et de s'améliorer régulièrement sur un marché en constante évolution.

C'est le processus de segmentation et de distinction des données selon des éléments communs ou des critères définis, qui est effectué par des algorithmes spéciaux. La reconnaissance de

formes permet des améliorations supplémentaires, ce qui en fait une partie intégrante de la technologie d'apprentissage automatique. Il identifie les modèles de données qui racontent les histoires de données à travers des pics et des lignes plates, des flux et des reflux. Les données peuvent provenir de :

- Sentiments
- Des sons
- Images
- Textes et autres

Les trois principaux modèles de reconnaissance de formes sont :

Template Matching : pour faire correspondre les caractéristiques de l'objet à l'aide d'un modèle prédéfini et analyser l'objet par procuration. L'un des exemples de correspondance de modèles est la « vérification du plagiat ».

Analyse Syntaxique/Structurelle : pour interpréter une relation complexe entre des composants, tels que les parties d'un discours.

Statistique : pour déterminer l'appartenances des pièces particulières. Par exemple, pour identifier si l'objet est un gâteau ou non.

Le processus impliqué dans la reconnaissance des formes comprend :

- Les données sont recueillies à partir de leurs sources
- Les données sont extraites du bruit à l'aide d'outils spéciaux
- Les informations récupérées sont examinées à la recherche d'éléments communs ou de caractéristiques pertinentes
- Ces éléments ou caractéristiques sont regroupés dans des catégories particulières
- Ces catégories sont examinées pour obtenir un aperçu des ensembles de données
- Les informations dérivées sont exécutées dans l'opération commerciale

Les principaux cas d'utilisation de la reconnaissance de formes sont :

Analyse des données : recherche d'audience et prévisions boursières - pour l'étude comparative des bourses et la prévision des résultats possibles, ainsi que l'analyse des données des utilisateurs

Traitement du langage naturel : chatbots, traduction de texte, analyse de texte et génération de texte - pour la découverte de sujets, la détection de plagiat, la recherche de la signification du texte et la recréation du message dans d'autres langues

Reconnaissance optique de caractères : vérification de signature et classification de documents, le traitement d'échantillons d'écriture manuscrite, la transcription de texte, traitement approfondi du document.

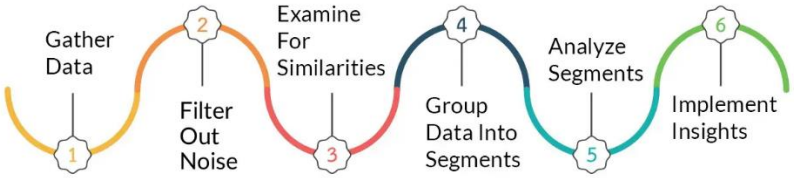

Figure 1.9 Les étapes du processus de reconnaissance [7]

4.6. La visualisation

La visualisation des données est le processus d'affichage d'informations visuelles à partir des données complexes existantes pour tirer une conclusion particulière en un coup d'œil sans avoir besoin d'étudier les résultats théoriques. Les applications comprennent des informations sur les données satellitaires, des informations sur les résultats de la recherche, des données scientifiquement étudiées, etc.

La visualisation des données fournit un mécanisme puissant pour aider l'utilisateur pendant le prétraitement des données et l'exploration des données proprement dite. Grâce à la visualisation des données d'origine, l'utilisateur peut naviguer pour avoir une sensation sur les propriétés de ces données.

[7] https://www.analytixlabs.co.in/blog/difference-between-machine-learning-pattern-recognition-and-data-mining/

La visualisation peut être aussi utilisée pour la détection des valeurs aberrantes, qui met en évidence les surprises dans les données, c'est-à-dire les instances de données qui ne sont pas conformes au comportement général ou au modèle des données. De plus, l'utilisateur est aidé à sélectionner les données appropriées via une interface visuelle. La transformation des données est une étape importante du prétraitement des données. Lors de la transformation des données, la visualisation des données peut aider l'utilisateur à garantir l'exactitude de la transformation. C'est-à-dire que l'utilisateur peut déterminer si les deux vues (originale ou transformée) des données sont équivalentes. La visualisation peut également être utilisée pour aider les utilisateurs lors de l'intégration des sources de données, en les aidant à voir les relations au sein des différents formats.

Les techniques de visualisation sont classées selon trois aspects : leur orientation (symbolique / géométrique), leur stimulus (2D versus 3D), et enfin, leur affichage (statique ou dynamique). En outre, les données d'un référentiel de données peuvent être considérées comme différents niveaux de granularité ou d'abstraction, ou comme différentes combinaisons d'attributs ou de dimensions. Les données peuvent être présentées dans divers formats visuels, y compris les diagrammes en boîte, les diagrammes de dispersion, les cubes 3D, les diagrammes de distribution des données, les courbes, la visualisation du volume, les surfaces ou les graphiques de liens, entre autres.

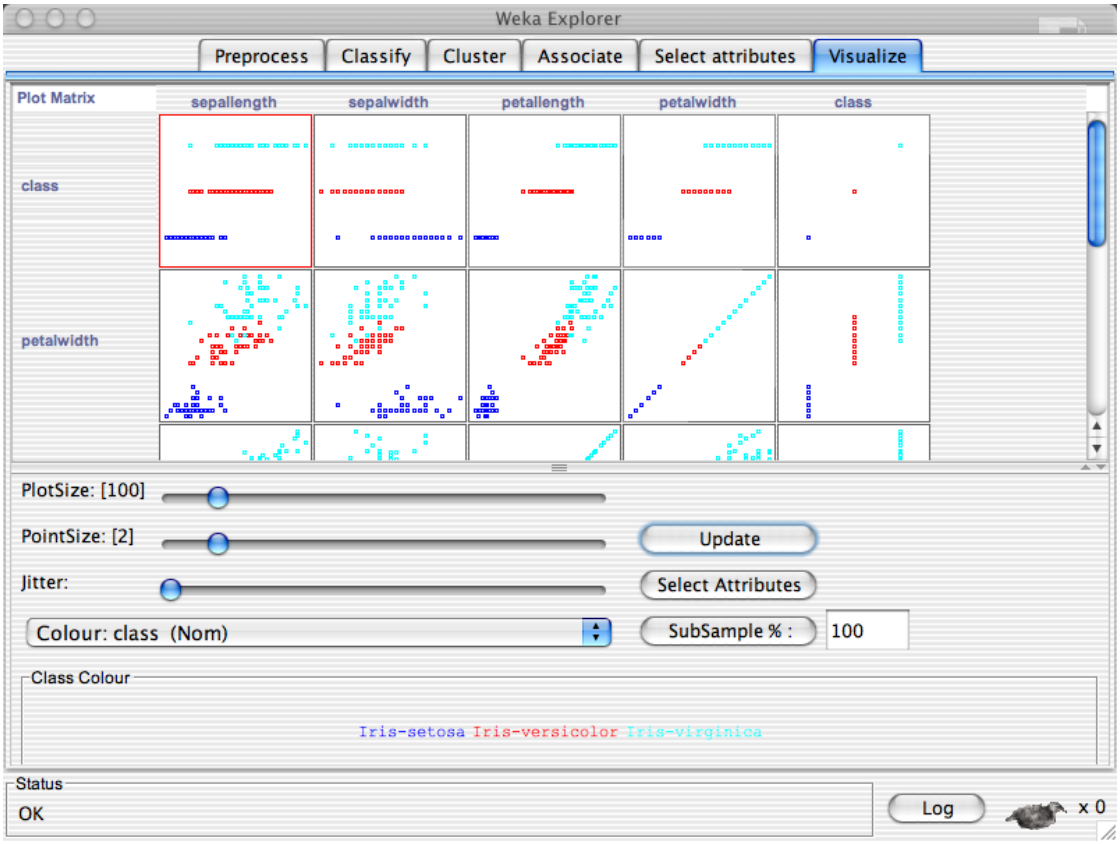

Figure 1.10 Visualisation des données avec Weka [8]

5. Les domaines d'application

En tant que discipline hautement axée sur les applications, l'exploration de données a connu de grands succès dans de nombreuses applications. Il est impossible d'énumérer toutes les applications où l'exploration de données joue un rôle critique. Les présentations de l'exploration de données dans des domaines d'application à forte intensité de connaissances, tels que la bioinformatique et le génie logiciel, nécessitent un traitement plus approfondi. Pour démontrer l'importance des applications en tant que dimension majeure dans la recherche et le développement de l'exploration de données, nous discutons brièvement de deux exemples d'applications très réussies et populaires de l'exploration de données : l'informatique décisionnelle et les moteurs de recherche.

[8] https://www.cs.waikato.ac.nz/~ml/weka/gui_explorer.html

5.1. Informatique décisionnelle (Business Intelligence)

Il est essentiel pour les entreprises d'acquérir une meilleure compréhension du contexte commercial de leur organisation, tels que leurs clients, le marché, l'offre et les ressources, et les concurrents. Les technologies de Business Intelligence (BI) fournissent des vues historiques, actuelles et prédictives des opérations commerciales. Les exemples incluent le reporting, le traitement analytique en ligne, la gestion des performances commerciales, la veille concurrentielle, l'analyse comparative et l'analyse prédictive.

Sans l'exploration de données, de nombreuses entreprises peuvent ne pas être en mesure d'effectuer une analyse efficace du marché, de comparer les commentaires des clients sur des produits similaires, de découvrir les forces et les faiblesses de leurs concurrents, de fidéliser des clients très précieux et de prendre des décisions commerciales intelligentes.

Les outils de traitement analytique en ligne en intelligence d'affaires s'appuient sur l'entreposage de données et l'exploration de données multidimensionnelles. Les techniques de classification et de prédiction sont au cœur de l'analyse prédictive en intelligence d'affaires, pour laquelle il existe de nombreuses applications dans l'analyse des marchés, des approvisionnements et des ventes. De plus, le clustering joue un rôle central dans la gestion de la relation client, qui regroupe les clients en fonction de leurs similitudes. En utilisant des techniques d'exploration de caractérisation, nous pouvons mieux comprendre les caractéristiques de chaque groupe de clients et développer des programmes de récompenses personnalisés.

Figure 1.11 Les étapes de traitement des données dans l'Informatique décisionnelle [9]

5.2. Moteurs de recherche Web

Un moteur de recherche Web est un serveur informatique spécialisé qui recherche des informations sur le Web. Les résultats de la recherche d'une requête utilisateur sont souvent renvoyés sous forme de liste (parfois appelés résultats). Les hits peuvent être constitués de pages Web, d'images et d'autres types de fichiers. Certains moteurs de recherche recherchent et renvoient également des données disponibles dans des bases de données publiques ou des répertoires ouverts.

Les moteurs de recherche diffèrent des annuaires Web, les annuaires Web sont gérés par des éditeurs humains, tandis que les moteurs de recherche fonctionnent de manière algorithmique ou par un mélange algorithmiques/humaines.

Les moteurs de recherche Web sont essentiellement de très grandes applications d'exploration de données. Diverses techniques d'exploration de données sont utilisées dans tous les aspects des moteurs de recherche :

[9] https://ostaraward.com/bi/

- L'exploration : décider quelles pages doivent être explorées et les fréquences d'exploration
- L'indexation : sélectionner les pages à indexer et décider dans quelle mesure l'index doit être construit
- La recherche : décider comment les pages doivent être classées, quelles publicités doivent être ajoutées et comment les résultats de la recherche peuvent être personnalisés ou rendus sensibles au contexte

Les moteurs de recherche posent de grands défis à l'exploration de données :

- La gestion d'une quantité énorme et sans cesse croissante de données. En règle générale, ces données ne peuvent pas être traitées à l'aide d'une ou de quelques machines. Au lieu de cela, les moteurs de recherche doivent souvent utiliser des nuages informatiques, qui se composent de milliers, voire de centaines de milliers d'ordinateurs qui exploitent en collaboration l'énorme quantité de données. La mise à l'échelle des méthodes d'exploration de données sur le cloud et les grands ensembles de données distribuées est un domaine de recherche plus approfondie.
- Les moteurs de recherche Web doivent souvent traiter des données en ligne. Un moteur de recherche peut se permettre de construire un modèle hors ligne sur d'énormes ensembles de données. Pour ce faire, il peut construire un classificateur de requête qui attribue une requête de recherche à des catégories prédéfinies en fonction du sujet de la requête. Pour construire un modèle hors ligne, l'application du modèle en ligne doit être suffisamment rapide pour répondre aux requêtes des utilisateurs en temps réel.
- Maintenir et à mettre à jour progressivement un modèle sur des flux de données à croissance rapide. En effet, un classificateur de requêtes peut devoir être maintenu de manière incrémentielle en continu car de nouvelles requêtes continuent d'émerger et de catégories prédéfinies et la distribution des données peut changer. La plupart des méthodes de formation de modèles existantes sont hors ligne et statiques et ne peuvent donc pas être utilisées dans un tel scénario.

- Traiter des requêtes qui ne sont posées qu'un très petit nombre de fois. Supposons qu'un moteur de recherche souhaite fournir des recommandations de requête contextuelles. Autrement dit, lorsqu'un utilisateur pose une requête, le moteur de recherche tente de déduire le contexte de la requête à l'aide du profil de l'utilisateur et de son historique de requêtes afin de renvoyer des réponses plus personnalisées en une petite fraction de seconde. Cependant, bien que le nombre total de requêtes posées puisse être énorme, la plupart des requêtes ne peuvent être posées qu'une ou plusieurs fois. Des données aussi fortement asymétriques représentent un défi pour de nombreuses méthodes d'exploration de données et d'apprentissage automatique.

6. Les problèmes rencontrés par le data Mining

L'exploration de données est un domaine dynamique et en pleine expansion. Dans cette section, nous décrivons brièvement les principaux problèmes de la recherche en data mining, en les répartissant en cinq groupes : méthodologie de minage, interaction avec l'utilisateur, efficacité et évolutivité, diversité des types de données, et data mining et société. Bon nombre de ces problèmes ont été abordés dans une certaine mesure dans la recherche et le développement récents en matière d'exploration de données et sont désormais considérés comme des exigences en matière d'exploration de données ; d'autres sont encore au stade de la recherche. Les problèmes continuent de stimuler les recherches et l'amélioration de l'exploration de données.

6.1. Méthodologie

Les chercheurs ont vigoureusement développé de nouvelles méthodologies d'exploration de données. Cela implique l'investigation de nouveaux types de connaissances, l'exploration de l'espace multidimensionnel, l'intégration de méthodes d'autres disciplines et la prise en compte des liens sémantiques entre les objets de données. En outre, les méthodologies d'exploration doivent tenir compte des problèmes tels que l'incertitude des données, le bruit et l'incomplétude. Certaines méthodes d'exploration explorent comment des mesures spécifiées par l'utilisateur peuvent être utilisées pour évaluer l'intérêt des modèles découverts ainsi que pour guider le processus de découverte.

6.1.1. Exploitation de nouveaux types de connaissances

L'exploration de données couvre un large éventail de tâches d'analyse de données et de découverte de connaissances, de la caractérisation et de la discrimination des données, l'analyse d'association et de corrélation, la classification, la régression, le regroupement, l'analyse des valeurs aberrantes, l'analyse de séquence et l'analyse des tendances et de l'évolution. Ces tâches peuvent utiliser la même base de données de différentes manières et nécessitent le développement de nombreuses techniques d'exploration de données. En raison de la diversité des applications, de nouvelles tâches de minage continuent d'émerger, faisant du data mining un domaine dynamique et en pleine croissance. Par exemple, pour une découverte efficace des connaissances dans les réseaux d'information, le regroupement et le classement intégrés peuvent conduire à la découverte de clusters et de classements d'objets de haute qualité dans de grands réseaux.

6.1.2. Exploration de connaissances dans un espace multidimensionnel

Lors de la recherche de connaissances dans de grands ensembles de données, on peut explorer les données dans un espace multidimensionnel. Autrement dit, on peut rechercher des modèles intéressants parmi des combinaisons de dimensions (attributs) à différents niveaux d'abstraction. Une telle extraction est connue sous le nom d'exploration de données multidimensionnelle. Dans de nombreux cas, les données peuvent être agrégées ou visualisées sous la forme d'un cube de données multidimensionnel. L'exploration de connaissances dans l'espace cubique peut considérablement améliorer la puissance et la flexibilité de l'exploration de données.

6.1.3. Exploration de données interdisciplinaire

La puissance de l'exploration de données peut être considérablement améliorée en intégrant de nouvelles méthodes issues de plusieurs disciplines. Pour extraire des données avec du texte en langage naturel, il est logique de fusionner des méthodes d'exploration de données avec des méthodes de recherche d'informations et de traitement du langage naturel. L'extraction de bogues logiciels dans de grands programmes est une forme de minage, appelée minage de bugs, qui bénéficie de l'incorporation du génie logiciel dans le processus d'exploration de données.

6.1.4. Accroître la puissance de la découverte dans un environnement en réseau

La plupart des objets de données résident dans un environnement lié ou interconnecté, qu'il s'agisse du Web, de relations de bases de données, de fichiers ou de documents. Les liens sémantiques entre plusieurs objets de données peuvent être utilisés de manière avantageuse dans l'exploration de données. Les connaissances dérivées d'un ensemble d'objets peuvent être utilisées pour stimuler la découverte de connaissances dans un ensemble d'objets "connexes" ou sémantiquement liés.

6.1.5. Gestion de l'incertitude, du bruit et de l'incomplétude des données

Les données contiennent souvent du bruit, des erreurs, des exceptions, des incertitudes, ou ils sont tout simplement incomplètes. Les erreurs et le bruit peuvent perturber le processus d'exploration de données, conduisant à la dérivation de modèles erronés. Le nettoyage des données, le prétraitement des données, la détection et la suppression des valeurs aberrantes et le raisonnement sur l'incertitude sont des exemples de techniques qui doivent être intégrées au processus d'exploration de données.

6.1.6. Évaluation de modèles et exploration guidée basée modèles

Tous les modèles générés par les processus d'exploration de données ne sont pas intéressants. Ce qui rend un modèle intéressant peut varier d'un utilisateur à l'autre. Par conséquent, des techniques sont nécessaires pour évaluer l'intérêt des modèles découverts sur la base de mesures subjectives. Ceux-ci estiment la valeur des modèles par rapport à une classe d'utilisateurs donnée, en fonction des croyances ou des attentes des utilisateurs.

De plus, en utilisant des mesures d'intérêt ou des contraintes spécifiées par l'utilisateur pour guider le processus de découverte, nous pouvons générer des modèles plus intéressants et réduire l'espace de recherche.

6.2. Interaction de l'utilisateur

L'utilisateur joue un rôle important dans le processus d'exploration de données. Les domaines de recherche intéressants incluent comment :

- Interagir avec un système d'exploration de données

- Intégrer les connaissances de base d'un utilisateur dans l'exploration de données
- Visualiser et comprendre les résultats de l'exploration de données.

6.2.1. Exploration interactive

Le processus d'exploration de données doit être hautement interactif. Ainsi, il est important de construire des interfaces utilisateur flexibles et un environnement de minage exploratoire, facilitant l'interaction de l'utilisateur avec le système. Un utilisateur peut souhaiter d'abord échantillonner un ensemble de données, explorer les caractéristiques générales des données et estimer les résultats d'exploration potentiels. L'exploration interactive devrait permettre aux utilisateurs de modifier dynamiquement l'orientation d'une recherche, d'affiner les requêtes d'exploration en fonction des résultats renvoyés, et d'explorer, de découper et de pivoter dans l'espace de données et de connaissances de manière interactive, en explorant dynamiquement l'espace cubique pendant l'exploration.

6.2.2. Intégration des connaissances de base

Les connaissances de base, les contraintes, les règles et d'autres informations concernant le domaine à l'étude doivent être intégrées au processus de découverte des connaissances. Ces connaissances peuvent être utilisées pour l'évaluation de modèles ainsi que pour guider la recherche vers des modèles intéressants.

6.2.3. Exploration de données ad hoc et langages de requête

Les langages de requête ont joué un rôle important dans la recherche flexible, car ils permettent aux utilisateurs de poser des requêtes ad hoc. De même, les langages de requête d'exploration de données de haut niveau ou d'autres interfaces utilisateur flexibles de haut niveau donneront aux utilisateurs la liberté de définir des tâches d'exploration de données ad hoc.

Cela devrait faciliter la spécification des ensembles de données pertinents pour l'analyse, la connaissance du domaine, les types de connaissances à exploiter et les conditions et contraintes à appliquer aux modèles découverts. L'optimisation du traitement de ces requêtes minières flexibles est un autre domaine d'étude prometteur.

6.2.4. Présentation et visualisation des résultats

Il s'agit de prévoir des mécanismes afin qu'un système d'exploration de données puisse présenter les résultats de l'exploration de données de manière vivante et flexible. En effet, il s'agit de facilité la compréhension et l'utilisabilité des connaissances découvertes par les humains. Ceci est particulièrement crucial si le processus d'exploration de données est interactif. Pour cela, le système nécessite d'adopter des représentations expressives des connaissances, des interfaces conviviales et des techniques de visualisation.

6.3. Efficacité et Evolutivité

L'efficacité et l'évolutivité sont toujours prises en compte lors de la comparaison d'algorithmes d'exploration de données. Alors que les quantités de données continuent de se multiplier, ces deux facteurs sont particulièrement critiques.

6.3.1. Efficacité et évolutivité des algorithmes d'exploration de données

Les algorithmes d'exploration de données doivent être efficaces et évolutifs afin d'extraire efficacement des informations à partir d'énormes quantités de données dans de nombreux référentiels de données ou dans des flux de données dynamiques. En d'autres termes, le temps d'exécution d'un algorithme d'exploration de données doit être prévisible, court et acceptable par les applications. L'efficacité, l'évolutivité, les performances, l'optimisation et la capacité d'exécution en temps réel sont des critères clés qui conduisent au développement de nombreux nouveaux algorithmes d'exploration de données.

6.3.2. Algorithmes d'exploration de données parallèles, distribués et incrémentiels

La taille gigantesque de nombreux ensembles de données, la large distribution des données et la complexité de calcul de certaines méthodes d'exploration de données sont des facteurs qui motivent le développement d'algorithmes d'exploration de données parallèles et distribués. De tels algorithmes divisent d'abord les données en "morceaux". Chaque pièce est traitée, en parallèle, par la recherche de motifs. Les processus parallèles peuvent interagir les uns avec les autres. Les modèles de chaque partition sont finalement fusionnés.

6.3.3. Le cloud computing et le cluster computing

Ces techniques sont considérées comme des thèmes de recherche actifs dans l'exploration de données parallèle. Ils utilisent des ordinateurs de manière distribuée et collaborative pour s'attaquer à des tâches de calcul à très grande échelle. De plus, le coût élevé de certains processus d'exploration de données et la nature incrémentielle des entrées favorisent l'exploration de données incrémentielle, qui intègre de nouvelles mises à jour de données sans avoir à extraire l'intégralité des données. De telles méthodes effectuent une modification des connaissances de manière incrémentielle pour modifier et renforcer ce qui a été découvert précédemment.

6.4. Diversité des types de bases de données

La grande diversité des types de bases de données pose des défis à l'exploration de données.

6.4.1. Gestion de types de données complexes

Diverses applications génèrent un large éventail de nouveaux types de données :

- Des données structurées : les données relationnelles et d'entrepôt de données aux données semi-structurées et non structurées ;
- Des référentiels de données stables aux flux de données dynamiques
- Des objets de données simples aux données temporelles : séquences biologiques, données de capteurs, données spatiales, données hypertextes, données multimédias, code de programme logiciel, données Web et données de réseaux sociaux.

Il n'est pas réaliste de s'attendre à ce qu'un seul système d'exploration de données exploite toutes sortes de données, étant donné la diversité des types de données et les différents objectifs de l'exploration de données.

Des systèmes d'exploration de données dédiés à un domaine ou à une application sont en cours de construction pour l'exploration en profondeur de types de données spécifiques. La construction d'outils d'exploration de données efficaces et efficients pour diverses applications reste un domaine de recherche stimulant et actif.

6.4.2. Extraction de référentiels de données dynamiques, en réseau et mondiaux

De multiples sources de données sont connectées par Internet et divers types de réseaux, formant des systèmes et des réseaux d'information mondiaux gigantesques, distribués et hétérogènes. La découverte de connaissances provenant de différentes sources de données structurées, semi-structurées ou non structurées mais interconnectées avec une sémantique de données diversifiée pose de grands défis à l'exploration de données. L'exploitation de ces gigantesques réseaux d'informations interconnectés peut aider à révéler beaucoup plus de modèles et de connaissances dans des ensembles de données hétérogènes que ce qui peut être découvert à partir d'un petit ensemble de référentiels de données isolés. L'exploration de données Web, l'exploration de données multisources et l'exploration de réseaux d'information sont devenues des domaines d'exploration de données difficiles et en évolution rapide.

6.4.3. Les enjeux de société

A. Impacts sociaux de l'exploration de données

Il est important d'étudier l'impact de l'exploration de données sur la société car celle-ci a pénétré notre vie quotidienne. Il s'agit de trouver la façon adéquate d'utilisation de la technologie d'exploration de données au profit de la société. Il également important aussi de garantir une utilisation non abusive. En effet, la divulgation ou l'utilisation inappropriée des données et la violation potentielle des droits individuels à la vie privée et à la protection des données sont des sujets de préoccupation qui doivent être traités.

B. Exploration de données préservant la confidentialité

L'exploration de données contribuera à la découverte scientifique, à la gestion d'entreprise, à la reprise économique et à la protection de la sécurité. Cependant, cela présente le risque de divulguer les informations personnelles d'un individu. Des études sur la publication et l'exploration de données préservant la confidentialité sont en cours. La philosophie est d'observer la sensibilité des données et de préserver la vie privée des personnes tout en effectuant une exploration de données réussie.

C. Exploration de données invisible

Il est clair que la maitrise des techniques d'exploration de données n'est pas à la portée de tous les membres de la société. De plus en plus de systèmes devraient intégrer des fonctions d'exploration de données afin que les utilisateurs puissent effectuer l'exploration de données ou utiliser les résultats de l'exploration de données simplement en cliquant sur la souris, sans aucune connaissance des algorithmes d'exploration de données. Les moteurs de recherche intelligents et les magasins basés sur Internet effectuent une telle exploration de données invisibles en incorporant l'exploration de données dans leurs composants pour améliorer leurs fonctionnalités et leurs performances. Cela se fait souvent à l'insu de l'utilisateur. Par exemple, lors de l'achat d'articles en ligne, les utilisateurs peuvent ignorer que le magasin collecte probablement des données sur les habitudes d'achat de ses clients, qui peuvent être utilisées pour recommander d'autres articles à acheter à l'avenir.

D. Les Objets et les types d'attributs

Les ensembles de données sont constitués d'objets de données. Un objet de données représente une entité. Dans une base de données de ventes, les objets peuvent être des clients, des articles de magasin et des ventes ; dans une base de données médicale, les objets peuvent être des patients ; dans une base de données universitaire, les objets peuvent être des étudiants, des professeurs et des cours. Les objets de données sont généralement décrits par des attributs. Les objets de données peuvent également être appelés échantillons, exemples, instances, points de données ou objets. Si les objets de données sont stockés dans une base de données, ce sont des tuples de données. Autrement dit, les lignes d'une base de données correspondent aux objets de données et les colonnes correspondent aux attributs.

i. Définition

Un attribut est un champ de données, représentant une caractéristique ou un élément d'un objet de données. Les noms attribut, dimension, caractéristique et variable sont souvent utilisés de manière interchangeable dans la littérature. Le terme dimension est couramment utilisé dans l'entreposage de données. La littérature sur l'apprentissage automatique a tendance à utiliser le terme fonctionnalité, tandis que les statisticiens préfèrent le terme variable. Les professionnels

de l'exploration de données et des bases de données utilisent couramment le terme attribut. Les valeurs observées pour un attribut donné sont appelées observations. Un ensemble d'attributs utilisés pour décrire un objet donné est appelé un vecteur d'attributs (ou vecteur de caractéristiques). La distribution des données impliquant un attribut (ou variable) est appelée univariée. Une distribution bivariée implique deux attributs, et ainsi de suite.

ii. Attributs nominaux

Les valeurs d'un attribut nominal sont des symboles ou des noms de choses. Chaque valeur représente une sorte de catégorie, de code ou d'état, et donc les attributs nominaux sont également appelés catégoriques. Les valeurs n'ont pas d'ordre significatif. En informatique, les valeurs sont également appelées énumérations. Bien que les valeurs d'un attribut nominal soient des symboles ou des noms de choses, il est possible de représenter ces symboles ou noms par des nombres.

iii. Attributs binaires

Un attribut binaire est un attribut nominal avec seulement deux catégories ou états : 0 ou 1, où 0 signifie généralement que l'attribut est absent et 1 signifie qu'il est présent. Les attributs binaires sont appelés booléens si les deux états correspondent à vrai et faux. Un attribut binaire est symétrique si ses deux états ont la même valeur et ont le même poids ; c'est-à-dire qu'il n'y a pas de préférence quant au résultat qui doit être codé 0 ou 1.

Un attribut binaire est asymétrique si les résultats des états ne sont pas également importants, comme les résultats positifs et négatifs d'un test médical pour le COVID.

iv. Attributs ordinaux

Un attribut ordinal est un attribut avec des valeurs possibles qui ont un ordre significatif ou rang parmi eux, mais l'amplitude entre les valeurs successives n'est pas connue.

Les attributs ordinaux peuvent également être obtenus à partir de la discrétisation des quantités numériques en divisant la plage de valeurs en un nombre fini de catégories ordonnées.

La tendance centrale d'un attribut ordinal peut être représentée par son mode et sa médiane, mais la moyenne ne peut pas être définie.

Les attributs nominaux, binaires et ordinaux sont qualitatifs. Autrement dit, ils décrivent une caractéristique d'un objet sans donner de taille ou de quantité réelle. Les valeurs de ces attributs qualitatifs sont généralement des mots représentant des catégories. Si des nombres entiers sont utilisés, ils représentent des codes informatiques pour les catégories, par opposition à des quantités mesurables (par exemple, 0 petite taille, 1 taille moyenne et 2 pour une grande taille).

v. Attributs numériques

Un attribut numérique est quantitatif ; c'est-à-dire qu'il s'agit d'une quantité mesurable, représentée en valeurs entières ou réelles. Les attributs numériques peuvent être mis à l'échelle par intervalle ou par rapport.

- **Attributs à échelle d'intervalle :** Les attributs à échelle d'intervalle sont mesurés sur une échelle d'unités de taille égale. Les valeurs des attributs mis à l'échelle par intervalle sont ordonnées et peuvent être positives, 0 ou négatives. Ainsi, en plus de fournir un classement des valeurs, ces attributs nous permettent de comparer et de quantifier la différence entre les valeurs.
- **Attributs proportionnels :** Un attribut proportionnel est un attribut numérique avec un point zéro inhérent. Autrement dit, si une mesure est mise à l'échelle par rapport, nous pouvons parler d'une valeur comme étant un multiple (ou rapport) d'une autre valeur. De plus, les valeurs sont ordonnées et nous pouvons également calculer la différence entre les valeurs, ainsi que la moyenne, la médiane et le mode.

vi. Attributs discrets et continus

Il existe de nombreuses façons d'organiser les types d'attributs. Les types ne sont pas mutuellement exclusifs. Les algorithmes de classification développés à partir du domaine de l'apprentissage automatique parlent souvent d'attributs comme étant discrets ou continus. Chaque type peut être traité différemment.

- **Attribut discret :** Un attribut discret a un ensemble fini ou dénombrable infini de valeurs, qui peuvent ou non être représentées sous forme de nombres entiers. Les attributs couleur des cheveux, fumeur, test médical et taille de la boisson ont chacun un nombre fini de valeurs et sont donc discrets. Un attribut est dénombrable infini si l'ensemble des

valeurs possibles est infini mais les valeurs peuvent être mises dans une correspondance un à un avec des nombres naturels.

- **Attribut continu :** Les termes attribut numérique et attribut continu sont souvent utilisés de manière interchangeable dans la littérature. Cela peut prêter à confusion car, au sens classique, les valeurs continues sont des nombres réels, alors que les valeurs numériques peuvent être des nombres entiers ou des nombres réels. En pratique, les valeurs réelles sont représentées à l'aide d'un nombre fini de chiffres. Les attributs continus sont généralement représentés sous forme de variables à virgule flottante.

7. Descriptions statistiques des bases des données

Pour que le prétraitement des données soit réussi, il est essentiel d'avoir une vue d'ensemble sur les données. Des descriptions statistiques de base peuvent être utilisées pour identifier les propriétés des données et mettre en évidence les valeurs de données qui doivent être traitées comme le bruit et les valeurs aberrantes.

7.1. Mesure de la tendance centrale

Les statistiques décrivant la position de la distribution comprennent la Moyenne, la Médiane, le Mode et la Somme de toutes les valeurs.

Mode : Le mode est la valeur la plus fréquente dans un échantillon.

Médiane : la médiane est un nombre qui divise en 2 parties la population telle que chaque partie contient le même nombre de valeurs. Dans la même logique, il y a les quartiles, déciles et centiles, qui divisent respectivement en 4, 10 et 100 la population.

Somme : total des valeurs pour toutes les observations n'ayant pas de valeur manquante.

Moyenne : La moyenne arithmétique est la somme des valeurs de la variable divisée par le nombre d’individus.

7.2. Mesurer la dispersion des données

Les statistiques mesurant la variance ou la dispersion dans les données, comprennent l'écart type, la variance, la plage, le minimum, le maximum et l'erreur standard (ES) de la moyenne.

7.2.1. Ecart type

Mesure de la dispersion des valeurs autour de la moyenne. Dans le cas d'une distribution normale, 68 % des observations se situent à l'intérieur d'un écart type de la moyenne et 95 % se situent à l'intérieur de deux écarts types. Par exemple, si la moyenne d'âge est de 45 avec un écart type égal à 10, une distribution normale verra 95 % des observations se situer entre 25 et 65.

7.2.2. Variance

Mesure de la dispersion des valeurs autour de la moyenne, égale à la somme des carrés des écarts par rapport à la moyenne, divisée par le nombre d'observations moins un. La variance se mesure en unités, qui sont égales au carré des unités de la variable.

7.2.3. Plage

Différence entre la valeur maximale et la valeur minimale d'une variable numérique (maximum - minimum).

7.2.4. Minimum

Plus petite valeur d'une variable numérique.

7.2.5. Maximum

Plus grande valeur d'une variable numérique.

7.2.6. Erreur standard de la moyenne

Mesure du taux de variation de la valeur de la moyenne sur des échantillons provenant de la même distribution. Cette mesure permet de comparer approximativement la moyenne observée avec une valeur hypothétique (autrement dit, vous pouvez conclure que ces deux valeurs sont différentes si le rapport de la différence avec l'erreur standard est inférieur à -2 ou supérieur à +2).

7.3. Distribution de données

L'asymétrie et kurtosis sont des statistiques qui décrivent la forme et la symétrie de la distribution. Ces statistiques sont présentées avec leurs erreurs standard.

7.3.1. Asymétrie

Mesure de l'asymétrie d'une distribution. La distribution normale est symétrique et possède une valeur d'asymétrie égale à 0. Une distribution dont la valeur d'asymétrie est positive présente une extrémité droite allongée. Une distribution caractérisée par une importante asymétrie négative présente une extrémité gauche plus allongée. Pour simplifier, une valeur d'asymétrie deux fois supérieure à l'erreur standard correspond à une absence de symétrie.

7.3.2. Kurtosis

Mesure de l'étendue de la présence de valeurs extrêmes. Dans le cas d'une distribution normale, la valeur de la statistique kurtosis est égale à zéro. Une valeur positive indique que les données comportent plus de valeurs extrêmes qu'une distribution normale. Une valeur négative indique que les données comportent moins de valeurs extrêmes qu'une distribution normale.

8. Visualisation de données

Il s’agit de trouver le bon mécanisme pour transmettre efficacement les données aux utilisateurs. La visualisation de données vise à communiquer les données clairement et efficacement grâce à une représentation graphique. La visualisation de données a été largement utilisée dans de nombreuses applications : la création de rapports, la gestion des opérations commerciales et le suivi de l'avancement des tâches. Plus communément, les techniques de visualisation permettent de découvrir des relations de données qui ne sont autrement pas facilement observables en regardant les données brutes. De nos jours, les gens utilisent également la visualisation de données pour créer des graphiques amusants et intéressants.

8.1. Visualisation orientée pixel

Un moyen simple de visualiser la valeur d'une dimension consiste à utiliser un pixel où la couleur du pixel reflète la valeur de la dimension. Pour un ensemble de données de m dimensions, les techniques orientées pixel créent m fenêtres à l'écran, une pour chaque dimension. Les valeurs de dimension m d'un enregistrement sont mappées en pixels aux positions correspondantes dans les fenêtres. Les couleurs des pixels reflètent les valeurs correspondantes.

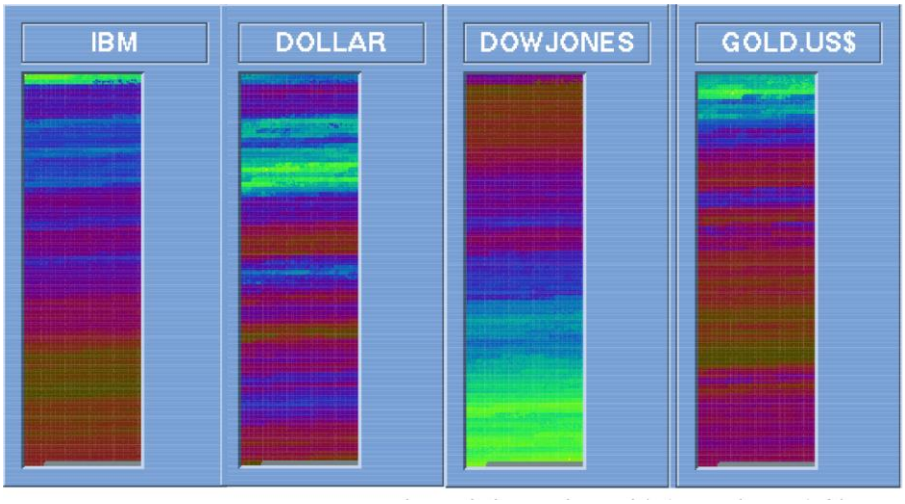

Figure1.12 Techniques de visualisation orientée pixel[10]

À l'intérieur d'une fenêtre, les valeurs de données sont disposées dans un ordre global partagé par toutes les fenêtres. L'ordre global peut être obtenu en triant tous les enregistrements de données d'une manière significative pour la tâche à accomplir.

Dans les techniques orientées pixel, les enregistrements de données peuvent également être triés en fonction de la requête. Par exemple, étant donné une requête ponctuelle, on peut trier tous les enregistrements par ordre décroissant de similarité avec la requête ponctuelle.

Remplir une fenêtre en disposant les enregistrements de données de manière linéaire peut ne pas fonctionner correctement pour une fenêtre large. Le premier pixel d'une rangée est éloigné du dernier pixel de la rangée précédente, bien qu'ils soient côte à côte dans l'ordre global. De plus, un pixel est à côté de celui au-dessus de lui dans la fenêtre, même si les deux ne sont pas à côté l'un de l'autre dans l'ordre global. Pour résoudre ce problème, on peut disposer les enregistrements de données dans une courbe de remplissage d'espace pour remplir les fenêtres. Une courbe de remplissage d'espace est une courbe dont la plage couvre l'ensemble de l'hypercube unitaire à n dimensions. Comme les fenêtres de visualisation sont en 2D, on peut utiliser n'importe quelle courbe de remplissage d'espace en 2D. La figure 1.13 montre quelques courbes de remplissage d'espace 2D fréquemment utilisées.

[10] Keim, Daniel A.. “Pixel-Oriented Visualization Techniques for Exploring Very Large Data Bases.” Journal of Computational and Graphical Statistics 5 (1996): 58-77.

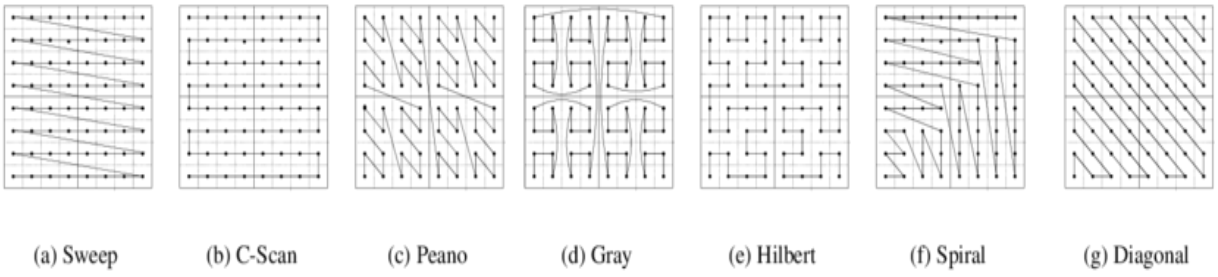

Figure 1.13 Courbes de remplissage d'espace bidimensionnelles[11]

Notez que les fenêtres ne doivent pas nécessairement être rectangulaires. Par exemple, la technique des segments de cercle utilise des fenêtres en forme de segments de cercle, comme illustré à la figure 1.14.

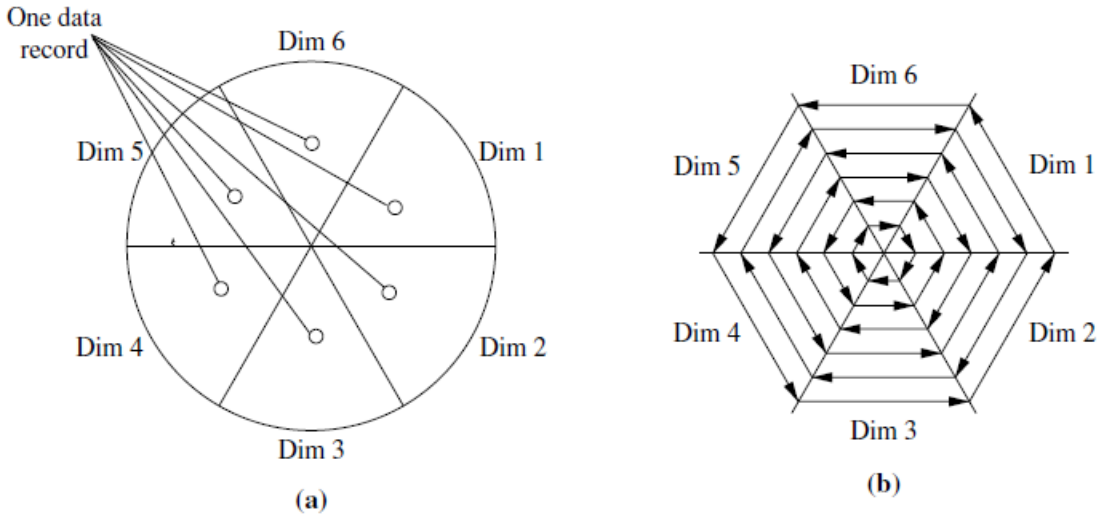

Figure 1.14 La technique du segment de cercle. (a) Représentation d'un enregistrement de données en segments de cercle. (b) Disposition des pixels en segments de cercle. [12]

Cette technique peut faciliter la comparaison des dimensions car les fenêtres de dimension sont situées côte à côte et forment un cercle.

La visualisation orientée pixel présente l'inconvenant d'être incapable de faire comprendre la distribution des données lorsqu'il s'agit d'un espace multidimensionnel. En effet, cette

[11] Mokbel, M.F. & Aref, Walid & Elbassioni, Khaled & Kamel, I.. (2004). Scalable multimedia disk scheduling. Proceedings - International Conference on Data Engineering. 20. 498 - 509. 10.1109/ICDE.2004.1320022.
[12] https://trenovision.com/data-visualization/

représentation ne permet pas de montrer s'il existe une zone dense dans un sous-espace multidimensionnel.

8.2. Visualisation par projection géométrique

Les techniques de projection géométrique permettent de pallier les inconvenants de la visualisation orientée pixel en aident les utilisateurs à trouver des projections intéressantes de données multidimensionnelles.

8.2.1. Nuage de points 2D

Le défi central que les techniques de projection géométrique tentent de relever est de savoir comment visualiser un espace de grande dimension sur un écran 2D. Un nuage de points affiche des points de données 2D à l'aide de coordonnées cartésiennes. Une troisième dimension peut être ajoutée en utilisant différentes couleurs ou formes pour représenter différents points de données (figure 1.15).

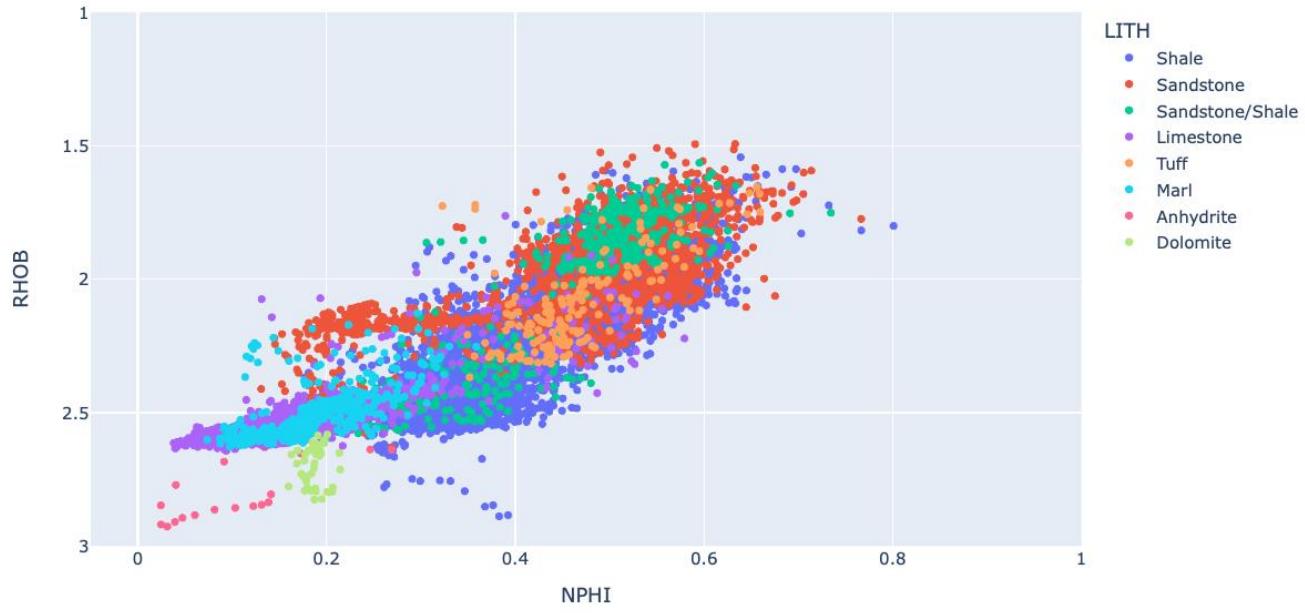

Figure 1.15 Visualisation d'un ensemble de données 2D à l'aide d'un nuage de points[13]

[13] https://towardsdatascience.com/using-plotly-express-to-create-interactive-scatter-plots-3699d8279b9e

8.2.2. Nuage de points 3D/4D

Un nuage de points 3D utilise trois axes dans un système de coordonnées cartésiennes. S'il utilise également la couleur, il peut afficher jusqu'à des points de données 4D (Figure 1.16).

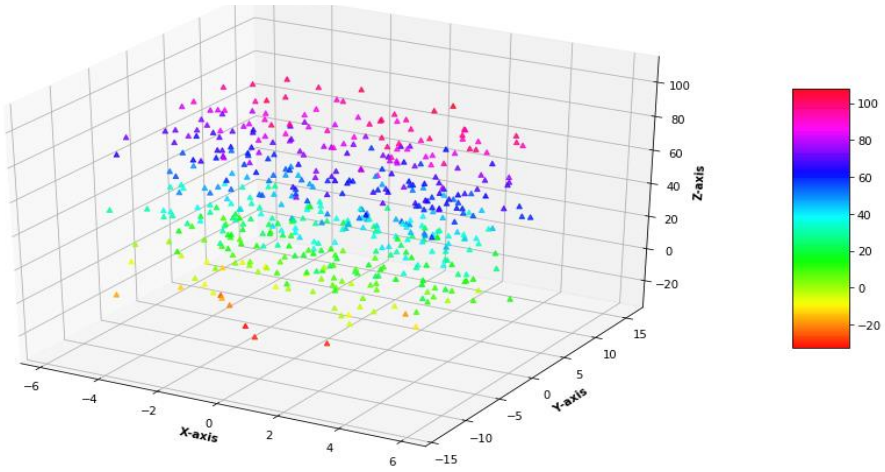

Figure 1.16 Un nuage de points 4D[14]

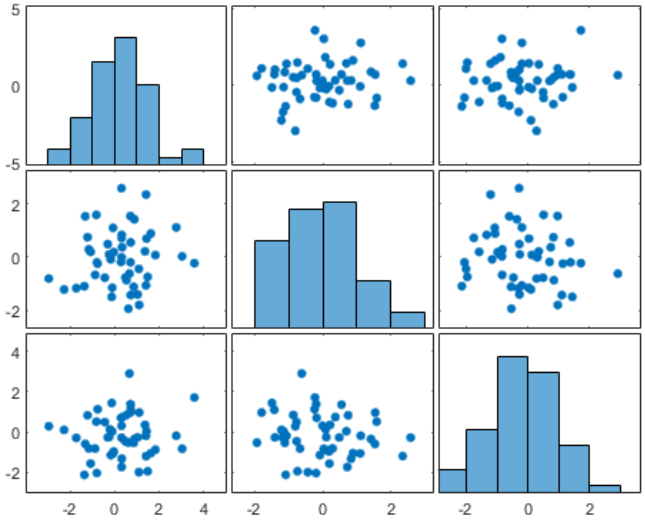

Figure 1.17 Matrice de nuages de points[15]

[14] https://www.geeksforgeeks.org/3d-scatter-plotting-in-python-using-matplotlib/
[15] https://la.mathworks.com/help/matlab/ref/plotmatrix.html

8.2.3. Matrice de nuages de points

Pour les ensembles de données à plus de quatre dimensions, les nuages de points sont généralement inefficaces.

La technique de la matrice de nuages de points est une extension utile du nuage de points. Pour un ensemble de données à n dimensions, une matrice de nuages de points est une grille n x n de nuages de points 2D qui fournit une visualisation de chaque dimension avec toutes les autres dimensions. La matrice de nuages de points devient moins efficace à mesure que la dimensionnalité augmente.

8.2.4. Coordonnées parallèles

Une autre technique populaire, appelée coordonnées parallèles, peut gérer une dimensionnalité plus élevée. Pour visualiser des points de données à n dimensions, la technique des coordonnées parallèles dessine n axes équidistants, un pour chaque dimension, parallèlement à l'un des axes d'affichage.

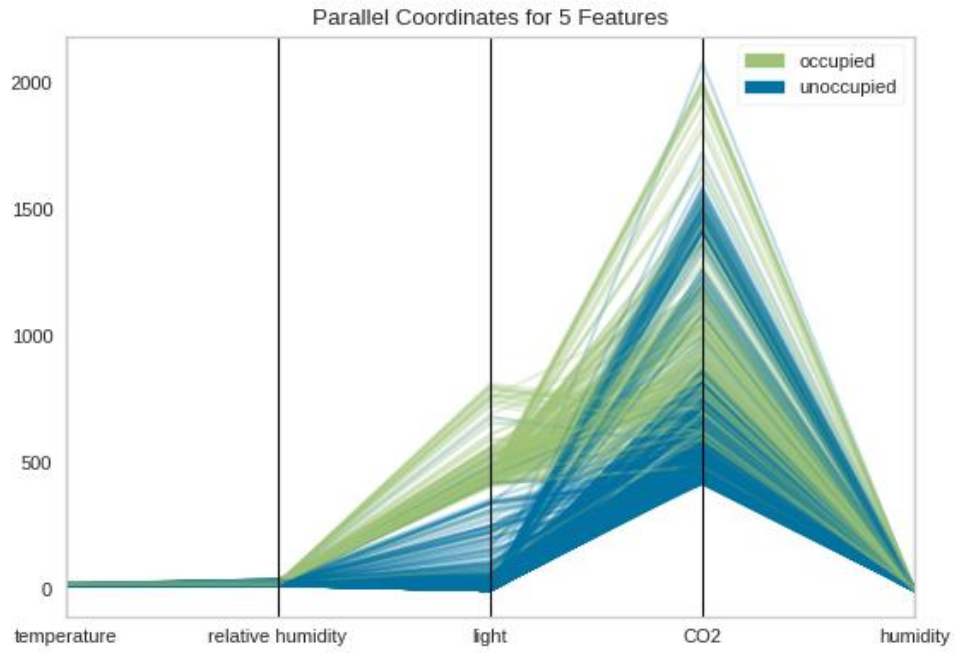

Figure 1.18 Visualisation basée sur des coordonnées parallèles [16]

[16] https://www.scikit-yb.org/en/latest/api/features/pcoords.html

Un enregistrement de données est représenté par une ligne polygonale qui coupe chaque axe au point correspondant à la valeur de dimension associée (Figure 1.18).

Une limitation majeure de la technique des coordonnées parallèles est qu'elle ne peut pas afficher efficacement un ensemble de données de nombreux enregistrements. Même pour un ensemble de données de plusieurs milliers d'enregistrements, l'encombrement visuel et le chevauchement réduisent souvent la lisibilité de la visualisation et rendent les modèles difficiles à trouver.

8.3. Visualisation basées icônes

Les techniques de visualisation basées sur des icônes utilisent de petites icônes pour représenter des valeurs de données multidimensionnelles. Parmi les plus populaires basées sur des icônes : les visages de Chernoff et les Stickw figures.

8.3.1. Les visages de Chernoff

Les visages de Chernoff ont été introduits en 1973 par le statisticien Herman Chernoff. Ils affichent des données multidimensionnelles jusqu'à 18 variables (ou dimensions) en tant que des dessins de visage humains (Figure 1.19).

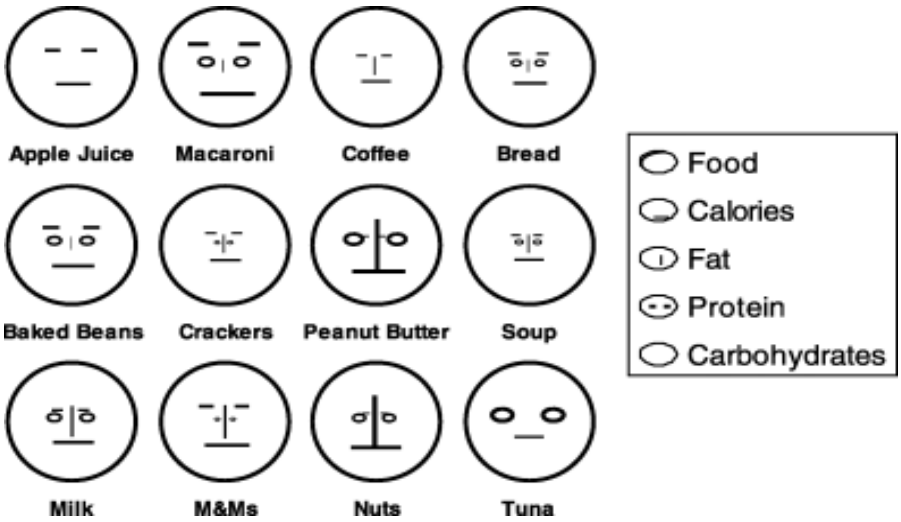

Figure 1.19 Les visages de Chernoff [17]

[17] Hunt, N. (2004), Chernoff Faces in Microsoft Excel. Teaching Statistics, 26: 75-77. https://doi.org/10.1111/j.1467-9639.2004.00173.x

Les visages de Chernoff aident à révéler les tendances dans les données. Les composants du visage, tels que les yeux, les oreilles, la bouche et le nez, représentent les valeurs des dimensions par leur forme, leur taille, leur emplacement et leur orientation. Par exemple, les dimensions peuvent être associées aux caractéristiques faciales suivantes :

- Taille des yeux, espacement des yeux, excentricité des yeux
- Longueur du nez, largeur du nez
- Courbure de la bouche, largeur de la bouche, ouverture de la bouche
- Taille de la pupille, inclinaison des sourcils, l'excentricité de la tête, etc.

Les visages de Chernoff utilisent la capacité de l'esprit humain à reconnaître de petites différences dans les caractéristiques faciales et à assimiler de nombreuses caractéristiques faciales à la fois.

L'affichage de grandes tables de données peut être fastidieux. En condensant les données, les visages de Chernoff rendent les données plus faciles à digérer pour les utilisateurs. De cette manière, ils facilitent la visualisation des régularités et des irrégularités présentes dans les données, bien que leur pouvoir de mise en relation multiples soit limité. Une autre limitation est que les valeurs de données spécifiques ne sont pas affichées.

De plus, les traits du visage varient en importance perçue. Cela signifie que la similarité de deux visages, représentant deux points de données multidimensionnels, peut varier en fonction de l'ordre dans lequel les dimensions sont attribuées aux caractéristiques faciales. Par conséquent, cette cartographie doit être choisie avec soin. La taille des yeux et l'inclinaison des sourcils se sont avérées importantes.

Des faces de Chernoff asymétriques ont été proposées comme une extension de la technique originale. Puisqu'un visage a une symétrie verticale (le long de l'axe y), les côtés gauche et droit d'un visage sont identiques, ce qui gaspille de l'espace. Les visages de Chernoff asymétriques doublent le nombre de caractéristiques faciales, permettant ainsi d'afficher jusqu'à 36 dimensions.

8.3.2. Stick figures

La technique des sticks figures mappe les données multidimensionnelles sur des bonshommes en cinq parties, où chaque personnage a quatre membres et un corps. Deux dimensions sont mappées sur les axes d'affichage (x et y) et les dimensions restantes sont mappées sur l'angle et/ou la longueur des membres (figure 1.20)

Si les éléments de données sont relativement denses par rapport aux deux dimensions d'affichage, la visualisation résultante montre des motifs de texture, reflétant la tendance des données.

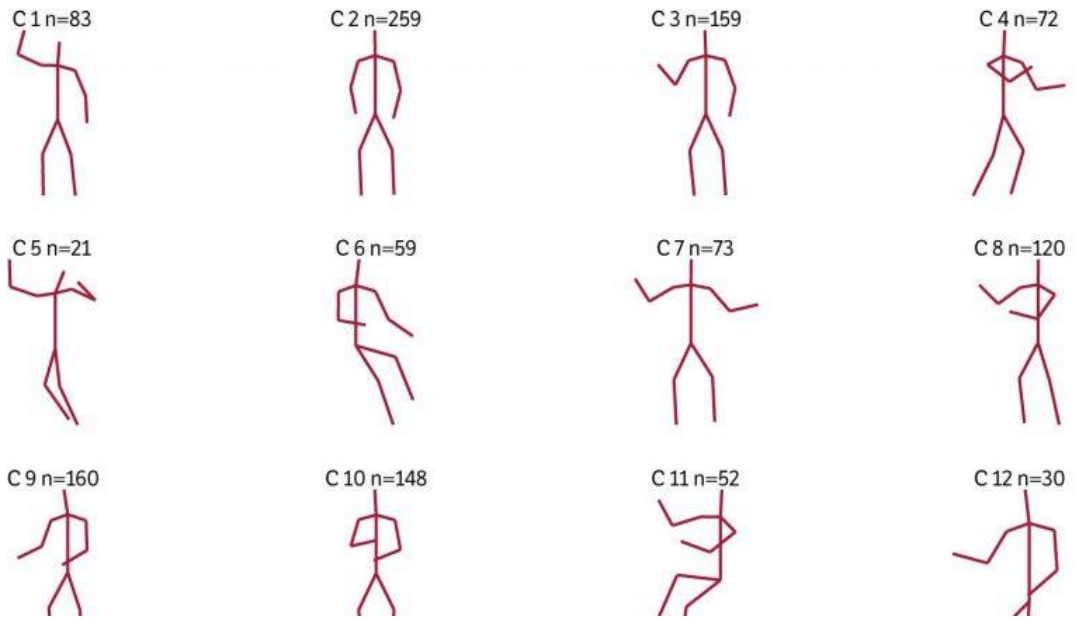

Figure 1.20 Sticks figures [18]

8.4. Visualisation hiérarchique

Les techniques de visualisation discutées jusqu'à présent se concentrent sur la visualisation simultanée de plusieurs dimensions. Cependant, pour un grand ensemble de données de haute dimensionnalité, il serait difficile de visualiser toutes les dimensions en même temps. Les

[18] https://www.epfl.ch/schools/cdh/tag/visualization/

techniques de visualisation hiérarchique partitionnent toutes les dimensions en sous-ensembles. Les sous-espaces sont visualisés de manière hiérarchique.

8.4.1. n-Vision (Worlds-within-Worlds)

C'est une méthode de visualisation hiérarchique représentative. L'utilisateur visualise alors les changements résultants du monde intérieur. De plus, un utilisateur peut faire varier les dimensions utilisées dans le monde intérieur et le monde extérieur. Étant donné plus de dimensions, plus de niveaux de mondes peuvent être utilisés, c'est pourquoi la méthode est appelée " Worlds-within-Worlds".

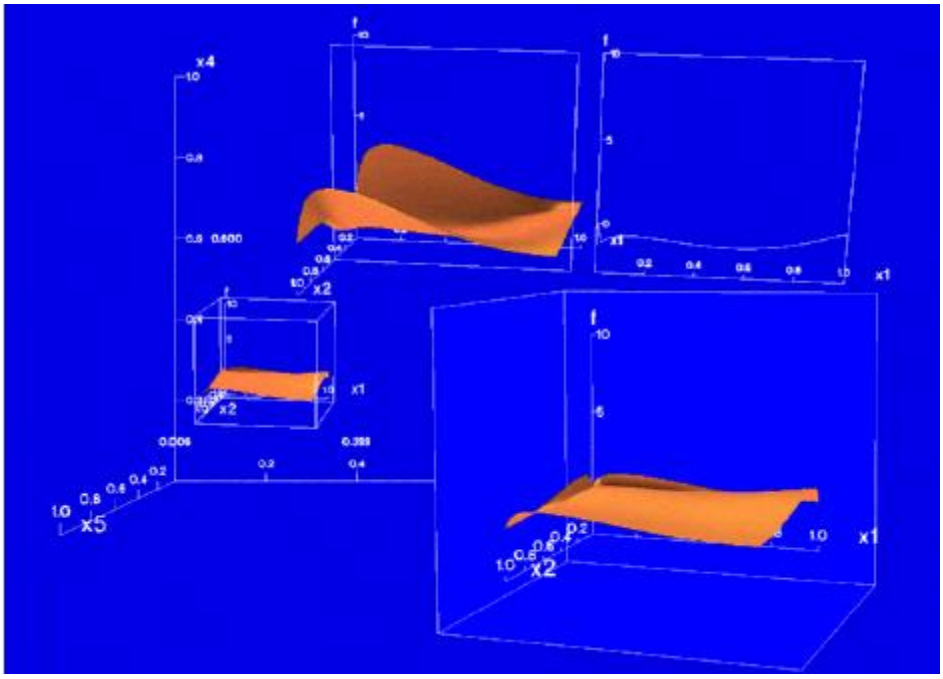

Figure 1.21 La technique de visualization Worlds-within-Worlds [19]

8.4.2. Les cartes arborescentes

Les cartes arborescentes est un autre exemple de méthodes de visualisation hiérarchique. Elles permettent d'afficher des données hiérarchiques sous la forme d'un ensemble de rectangles imbriqués (figure 1.22)

[19] https://www.researchgate.net/figure/World-within-worlds12_fig3_3723289

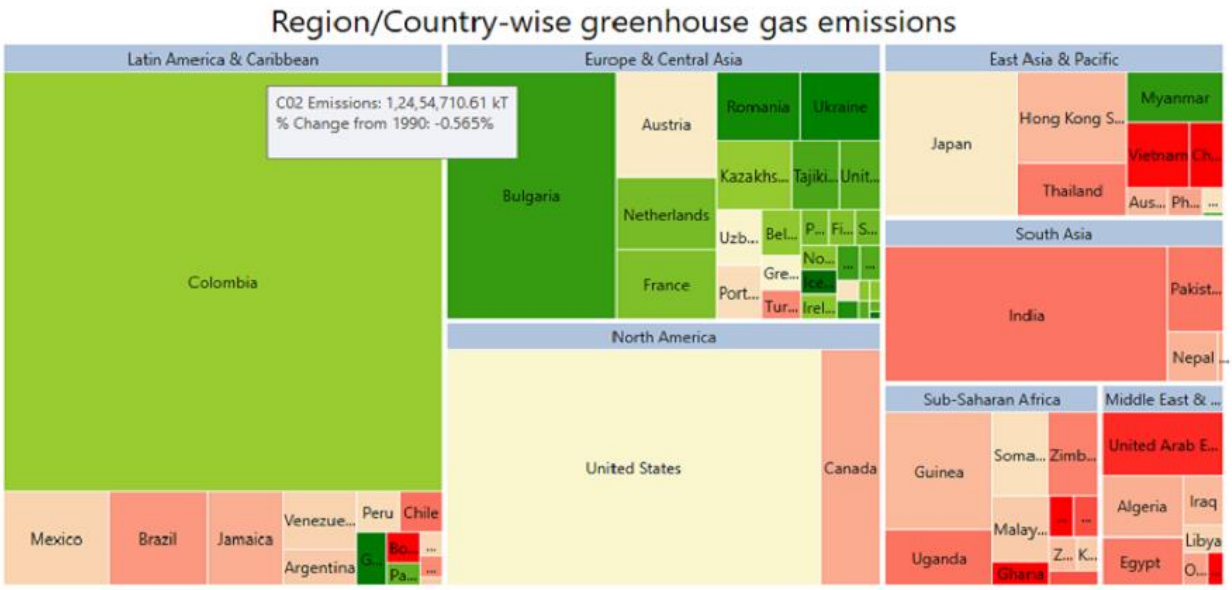

Figure 1.22 Les cartes arborescentes [20]

8.5. La visualisation de données dans les relations complexes

Au début, les techniques de visualisation étaient principalement destinées aux données numériques. Récemment, de plus en plus de données non numériques, telles que le texte et les réseaux sociaux, sont devenues disponibles. La visualisation et l'analyse de telles données suscitent beaucoup d'intérêt.

Il existe de nombreuses nouvelles techniques de visualisation dédiées à ce type de données. Par exemple, de nombreuses personnes sur le Web étiquette divers objets tels que des images, des entrées de blog et des critiques de produits.

8.5.1. Nuage de Tag

Un nuage de tags est une visualisation des statistiques des tags générés par l'utilisateur. Souvent, dans un nuage de tags, les tags sont répertoriés par ordre alphabétique ou dans un ordre préféré par l'utilisateur. L'importance d'un tag est indiquée par la taille ou la couleur de la police. La figure 1.23 montre un ensemble de résultats de recherche sous la forme d'une combinaison de nuages de tag.

[20] https://www.grapecity.com/blogs/analyzing-bivariate-data-with-treemap-charts

Figure 1.23 Visualisation par combinaison de nuages de tag

Les nuages de tags sont souvent utilisés de deux manières :

- Dans un nuage de tags pour un seul élément, on peut utiliser la taille d'un tag pour représenter le nombre de fois que le tag est appliqué à cet élément par différents utilisateurs.
- Lors de la visualisation des statistiques de balise sur plusieurs éléments, on peut utiliser la taille d'un tag pour représenter le nombre d'éléments auxquels la balise a été appliquée, c'est-à-dire la popularité de la balise.

8.5.2. Les graphes d'influence

Outre les données complexes, les relations complexes entre les entrées de données posent également des problèmes de visualisation. Un graphique d'influence pour visualiser les

corrélations entre les données utilise les nœuds ou la taille de chaque nœud est proportionnelle à la prévalence de la donnée correspondante. Deux nœuds sont reliés par une arête si les données correspondantes ont une forte corrélation. La largeur d'un bord est proportionnelle à la force du modèle de corrélation des deux données correspondantes (figure 1.24).

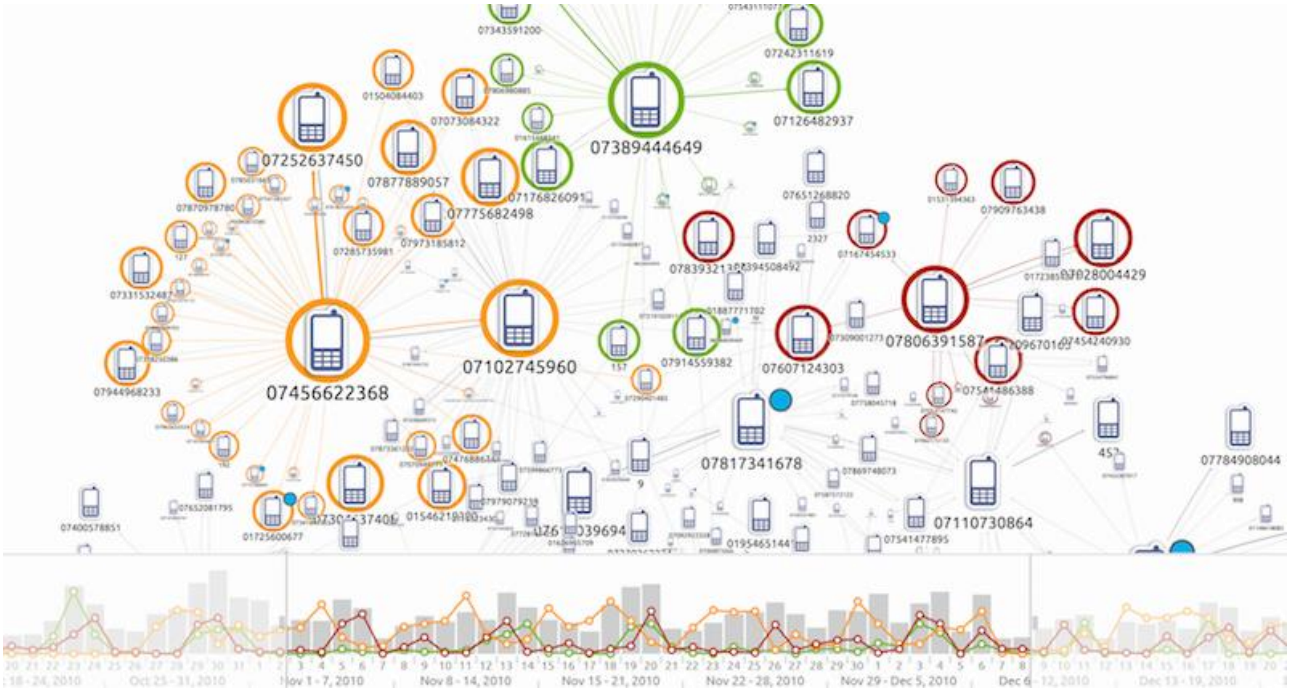

Figure 1.24 Graphe d'influence [21]

9. Mesures de similarité des données

Dans les applications du Data mining, telles que le clustering, l'analyse des valeurs aberrantes et la classification du plus proche voisin, on a besoin de moyens pour évaluer à quel point les objets sont similaires ou non les uns par rapport aux autres. Par exemple, un magasin peut souhaiter rechercher des groupes d'objets client, ce qui entraîne des groupes de clients présentant des caractéristiques similaires (revenus, zone de résidence et âge similaires). Ces informations peuvent ensuite être utilisées à des fins de marketing. Un cluster est une collection d'objets de données tels que les objets au sein d'un cluster sont similaires les uns aux autres et différents des objets dans d'autres clusters. L'analyse des valeurs aberrantes utilise également des techniques basées sur le regroupement pour identifier les valeurs aberrantes potentielles comme des objets

[21] https://www.pinterest.com/pin/821062575786429032/

très différents des autres. La connaissance des similitudes d'objets peut également être utilisée dans les schémas de classification du plus proche voisin où un objet donné se voit attribuer une étiquette de classe en fonction de sa similitude avec d'autres objets du modèle.

Les mesures de similarité et de dissimilarité, appelées mesures de proximité sont liées. En effet, une mesure de similarité pour deux objets, i et j, va renvoyer généralement la valeur 0 si les objets ne se ressemblent pas. Plus la valeur de similarité est élevée, plus la similarité entre les objets est grande.

Généralement, une valeur de 1 indique une similarité complète, c'est-à-dire que les objets sont identiques. Une mesure de dissimilarité fonctionne dans le sens opposé. Il renvoie une valeur de 0 si les objets sont identiques. Plus la valeur de dissimilarité est élevée, plus les deux objets sont dissemblables.

9.1. Matrice de données et matrice de dissimilarité

Lorsqu'il s'agissait d'objets unidimensionnels, c'est-à-dire décrits par un seul attribut, les moyens utilisés étaient la tendance centrale, la dispersion et la propagation des valeurs observées pour un attribut X. Dans cette section, nous allons aborder les objets décrits par plusieurs attributs. Par conséquent, on a besoin d'un changement de notation.

Soit n objets décrits par p attributs. Les objets sont :

$$X_1 = \left(x_{11}, x_{12}, x_{13}, \dots, x_{1p}\right),$$

$$X_2 = (x_{21}, x_{22}, x_{23}, \dots, x_{2p}$$

où x_{ij} est la valeur de l'objet x_i du j^{ème} attribut.

Les objets peuvent être des tuples dans une base de données relationnelle et sont également appelés échantillons de données ou vecteurs de caractéristiques. Le clustering basé sur la mémoire principale et les algorithmes du plus proche voisin fonctionnent généralement sur l'une des deux structures de données suivantes :

9.1.1. Matrice de données

Cette structure stocke les n objets de données sous la forme d'une table relationnelle, ou matrice n x p. Chaque ligne correspond à un objet.

$$\begin{bmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{bmatrix}$$

9.1.2. Matrice de dissimilarité

Cette structure stocke une collection de proximités qui sont disponibles pour toutes les paires de n objets. Il est souvent représenté par un tableau n x n.

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & \ldots & \cdots & 0 \end{bmatrix}$$

où $d(i,j)$ est la dissimilarité mesurée entre les objets i et j. En général $d(i,j)$ est un nombre non négatif qui est proche de 0 lorsque les objets i et j sont très similaires ou "proches" l'un de l'autre, et devient plus grand plus ils diffèrent. Les mesures de similarité peuvent souvent être exprimées en fonction de mesures de dissimilarité.

Une matrice de données est composée de deux entités ou "choses", à savoir des lignes (pour les objets) et des colonnes (pour les attributs). Par conséquent, la matrice de données est souvent appelée matrice à deux modes. La matrice de dissimilarité contient un type d'entité et est donc appelée une matrice à un mode. De nombreux algorithmes de clustering et de plus proche voisin fonctionnent sur une matrice de dissimilarité. Des données sous la forme d'une matrice de données peuvent être transformées en une matrice de dissimilarité avant d'appliquer de tels algorithmes.

9.2. Calcul des distances pour les attributs

9.2.1. Les attributs nominaux

Un attribut nominal peut prendre deux ou plusieurs états. Soit M le nombre d'états d'un attribut nominal. Les états peuvent être désignés par des lettres, des symboles ou un ensemble d'entiers, tels que 1,2, …, M.

Notez que ces entiers sont utilisés uniquement pour le traitement des données et ne représentent aucun ordre spécifique. La dissimilarité entre deux objets i et j peut être calculée sur la base du rapport de la non-concordances :

$$d(i,j) = \frac{p-m}{p}$$

où m est le nombre de correspondances (c'est-à-dire le nombre d'attributs pour lesquels i et j sont dans le même état), et p est le nombre total d'attributs décrivant les objets. Des poids peuvent être attribués pour augmenter l'effet de m ou pour attribuer un poids plus important aux correspondances dans les attributs ayant un plus grand nombre d'états.

Alternativement, la similarité peut être calculée comme

$$s(i,j) = 1 - d(i,j) = \frac{m}{p}$$

La proximité entre les objets décrits par des attributs nominaux peut être calculée à l'aide d'un schéma de codage alternatif. Les attributs nominaux peuvent être codés à l'aide d'attributs binaires asymétriques en créant un nouvel attribut binaire pour chacun des états M. Pour un objet avec une valeur d'état donnée, l'attribut binaire représentant cet état est défini par 1, tandis que les attributs binaires restants sont définis par 0.

9.2.2. Attributs binaires

Examinons les mesures de dissimilarité et de similarité pour les objets décrits par des attributs binaires symétriques ou asymétriques.

Rappelons qu'un attribut binaire n'a qu'un seul des deux états : 0 et 1, où 0 signifie que l'attribut est absent, et 1 signifie qu'il est présent. Traiter les attributs binaires comme s'ils étaient

numériques peut être trompeur. Par conséquent, des méthodes spécifiques aux données binaires sont nécessaires pour calculer la dissimilarité.

Une approche consiste à calculer une matrice de dissimilarité à partir des données binaires. Si tous les attributs binaires sont considérés comme ayant le même poids, nous avons la table de contingence suivante :

	Objet i			
Objet j		1	0	sum
	1	q	r	q+r
	0	s	t	s+t
	sum	q+s	r+t	p

Tableau 1.1 Table de contingence

où q est le nombre d'attributs égaux à 1 pour les objets i et j, r est le nombre d'attributs égaux à 1 pour l'objet i mais égaux à 0 pour l'objet j, s est le nombre d'attributs égaux à 0 pour l'objet i mais égal à 1 pour l'objet j, et t est le nombre d'attributs égaux à 0 pour les objets i et j. P est le nombre total d'attributs, où p = q+r+s+t .

Rappelez-vous que pour les attributs binaires symétriques, chaque état a la même valeur. La dissemblance basée sur des attributs binaires symétriques est appelée dissimilarité binaire symétrique. Si les objets i et j sont décrits par des attributs binaires symétriques, alors la dissimilarité entre i et j est

$$d(i,j) = \frac{r+s}{q+r+s+t}$$

Pour les attributs binaires asymétriques, les deux états n'ont pas la même importance, comme les résultats positifs (1) et négatifs (0) d'un test de maladie. Étant donné deux attributs binaires asymétriques, la concordance de deux 1 (une correspondance positive) est alors considérée comme plus significative que celle de deux 0 (une correspondance négative). Par conséquent, de tels attributs binaires sont souvent considérés comme monétaires (ayant un état). La dissimilarité basée sur ces attributs est appelée dissimilarité binaire asymétrique, où le nombre de

correspondances négatives, t , est considéré comme sans importance et est donc ignoré dans le calcul suivant :

$$d(i,j) = \frac{r+s}{q+r+s}$$

De manière complémentaire, on peut mesurer la différence entre deux attributs binaires en se basant sur la notion de similarité au lieu de la dissimilarité. Par exemple, la similarité binaire asymétrique entre les objets i et j peut être calculée comme

$$s(i,j) = \frac{q}{q+r+s} = 1 - d(i,j)$$

Le coefficient s(i,j) est appelé le coefficient de Jaccard et est communément référencé dans la littérature.

9.2.3. Attributs numériques

Soit deux points de E, (x1, x2, …,xn) et (y1, y2, …,yn), on exprime les différentes distances ainsi :

- **Distance euclidienne :** $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
- **Distance de Manhattan :** $\sum_{i=1}^{n}|x_i - y_i|$
- **Distance de Minkowski :** $\sqrt[p]{\sum_{i=1}^{n}|x_i - y_i|^p}$
- **Distance de Tchebychev :** $\lim_{p\to\infty} \sqrt[p]{\sum_{i=1}^{n}|x_i - y_i|^p} = sup_{1<i<n}|x_i - y_i$

9.2.4. Attributs ordinaux

Les valeurs d'un attribut ordinal ont un ordre ou un classement significatif à leur sujet, mais l'amplitude entre les valeurs successives est inconnue. Des attributs ordinaux peuvent également être obtenus à partir de la discrétisation d'attributs numériques en divisant la plage de valeurs en un nombre fini de catégories. Ces catégories sont organisées en rangs. C'est-à-dire que la plage d'un attribut numérique peut être mappée sur un attribut ordinal t ayant des états M_t.

Exemple :

La plage de température de l'attribut à échelle d'intervalle peut être organisée selon les états suivants : -30 à -10, -10 à 10, 10 à 30, représentant les catégories de température froide, modérée et chaude.

Soit M le nombre d'états possibles que peut avoir un attribut ordinal. Ces états ordonnés définissent le classement 1, … , M_t . Le traitement des attributs ordinaux est assez similaire à celui des attributs numériques lors du calcul de la dissimilarité entre objets. Les valeurs de similarité pour les attributs ordinaux peuvent être interprétées à partir de la dissimilarité.

9.2.5. Attributs de types mixtes

Dans de nombreuses bases de données réelles, les objets sont décrits par un mélange de types d'attributs. En général, une base de données peut contenir tous types d'attributs. Le défi est de trouver comment calculer la dissimilarité entre des objets avec des attributs de types mixtes.

- Une approche consiste à regrouper chaque type d'attribut, en effectuant une analyse d'exploration de données distincte pour chaque type. Cela est faisable si ces analyses aboutissent à des résultats compatibles. Cependant, dans des applications réelles, il est peu probable qu'une analyse séparée par type d'attribut génère des résultats compatibles.
- Une approche plus préférable consiste à traiter tous les types d'attributs ensemble, en effectuant une seule analyse. Une de ces techniques combine les différents attributs en une seule matrice de dissimilarité, amenant tous les attributs significatifs sur une échelle commune de l'intervalle [0, 1].

Supposons que l'ensemble de données contient p attributs de type mixte. La dissimilarité entre les objets i et j est définie comme

$$d(i,j) = \frac{\sum_{t=1}^{p} \delta_{ij}^{(t)} d_{ij}^{(t)}}{\sum_{t=1}^{p} \delta_{ij}^{(t)}}$$

où l'indicateur $\delta_{ij}^{(t)} = 0$ si

- x_{it} ou x_{jt} est manquant (c'est-à-dire qu'il n'y a pas de mesure de l'attribut t pour l'objet i ou l'objet j)
- $x_{it}= x_{jt}= 0$ et l'attribut et t est un binaire asymétrique

La contribution de l'attribut t à la dissimilarité entre i et est calculé en fonction de son type.

Ces étapes sont identiques pour chaque type d'attributs. La seule différence concerne les attributs numériques, la normalisation est effectuée afin que les valeurs correspondent à l'intervalle [0, 1]. Ainsi, la dissimilarité entre objets peut être calculée même lorsque les attributs décrivant les objets sont de types différents.

9.3. Similarité basée cosinus

Un document peut être représenté par des milliers d'attributs, chacun enregistrant la fréquence d'un mot particulier ou d'une phrase dans le document. Ainsi, chaque document est un objet représenté par ce qu'on appelle un vecteur terme-fréquence. Les vecteurs terme-fréquence sont généralement très longs et ont de nombreuses valeurs 0.

Les applications utilisant de telles structures comprennent la recherche d'informations, le regroupement de documents texte, la taxonomie biologique et la cartographie des caractéristiques génétiques. Les mesures de distance traditionnelles ne fonctionnent pas bien pour des données numériques aussi éparpillées. Par exemple, deux vecteurs terme-fréquence peuvent avoir de nombreuses valeurs 0 en commun, ce qui signifie que les documents correspondants ne partagent pas beaucoup de mots, mais cela ne les rend pas similaires. Nous avons besoin d'une mesure qui se concentrera sur les mots que les deux documents ont en commun et sur la fréquence d'occurrence de ces mots. En d'autres termes, nous avons besoin d'une mesure pour les données numériques qui ignore les correspondances par zéro.

La similarité cosinus est une mesure de similarité qui peut être utilisée pour comparer des documents ou, par exemple, donner un classement de documents par rapport à un vecteur donné de mots de requête.

Soient x et y deux vecteurs de comparaison. En utilisant la mesure du cosinus comme fonction de similarité, nous avons :

$$S(x, y) = \frac{x.y}{\|x\|.\|y\|}$$

où $\|x\|$ est la norme euclidienne du vecteur x et $\|y\|$ est la norme euclidienne du vecteur y. La mesure calcule le cosinus de l'angle entre les vecteurs x et y. Une valeur de cosinus de 0 signifie que les deux vecteurs sont à 90 degrés l'un de l'autre (orthogonal) et ne correspondent pas. Plus la valeur du cosinus est proche de 1, plus l'angle est petit et plus la correspondance entre les vecteurs est grande. Notez que parce que la mesure de similarité cosinus n'obéit pas à toutes les propriétés définissant les mesures métriques, elle est appelée mesure non métrique.

Lorsque les attributs sont à valeur binaire, la fonction de similarité cosinus peut être interprétée en termes de caractéristiques ou d'attributs partagés. Supposons qu'un objet x possède le $i^{ème}$ attribut si xi = 1. Alors $x^t.y$ est le nombre d'attributs possédés par x et y, et $|x|.|y|$est la moyenne géométrique du nombre d'attributs possédés par x et le nombre possédé par y. Ainsi, s(x,y) est une mesure de possession relative d'attributs communs. Dans ce cas, la similarité cosinus s'écrit comme :

$$s(x, y) = \frac{x.y}{x.x + y.y - x.y}$$

qui est le rapport du nombre d'attributs partagés par x et y au nombre d'attributs possédés par x ou y. Cette fonction, connue sous le nom de coefficient de Tanimoto ou distance de Tanimoto, est fréquemment utilisée dans la recherche d'informations et la taxonomie biologique.

Chapitre II : Prétraitement des données

1 Introduction

Aujourd'hui, les bases de données sont très sensibles aux données bruyantes, manquantes et incohérentes en raison de leur taille et de leur origine probable à partir de sources multiples et hétérogènes. Des données de mauvaise qualité conduiront à des résultats de mauvaise qualité. Pour cela, Il existe plusieurs techniques de prétraitement des données.

Le nettoyage des données peut être appliqué pour supprimer le bruit et corriger les incohérences dans les données. L'intégration de données fusionne les données provenant de plusieurs sources dans un magasin de données cohérent tel qu'un entrepôt de données. La réduction des données peut réduire la taille des données en éliminant les fonctionnalités redondantes ou en les regroupant. Des transformations de données peuvent être appliquées, où les données sont mises à l'échelle pour tomber dans une plage plus petite. Cela peut améliorer la précision et l'efficacité des algorithmes de data Mining impliquant des mesures de distance. Ces techniques ne sont pas mutuellement exclusives ; ils peuvent travailler ensemble.

2. Qualité des données

Les données sont de qualité si elles satisfont aux exigences de l'utilisation prévue. Il existe de nombreux facteurs qui composent la qualité des données, notamment l'exactitude, l'exhaustivité, la cohérence, l'actualité, la crédibilité et l'interprétabilité.

Les données inexactes, incomplètes et incohérentes sont des propriétés courantes des grandes bases de données et des entrepôts de données du monde réel. Il existe de nombreuses raisons possibles pour des données inexactes :

- Les instruments de collecte de données utilisés peuvent être défectueux.
- Des erreurs humaines ou informatiques lors de la saisie des données.

- Les utilisateurs peuvent volontairement soumettre des valeurs de données incorrectes pour les champs obligatoires lorsqu'ils ne souhaitent pas soumettre d'informations personnelles.
- Des erreurs de transmission de données peuvent également se produire.
- Des limitations technologiques telles qu'une taille de mémoire tampon limitée pour coordonner le transfert et la consommation de données synchronisées.
- Incohérences dans les conventions de dénomination ou les codes de données, ou de formats incohérents pour les champs d'entrée. Les tuples en double nécessitent également un nettoyage des données.

Des données incomplètes peuvent se produire pour un certain nombre de raisons :

- Les attributs intéressants peuvent ne pas toujours être disponibles
- D'autres données peuvent ne pas être incluses simplement parce qu'elles n'étaient pas considérées comme importantes au moment de leur saisie.
- Les données pertinentes peuvent ne pas être enregistrées en raison d'un malentendu ou en raison de dysfonctionnements de l'équipement.
- Les données qui étaient incompatibles avec d'autres données enregistrées peuvent avoir été supprimées.
- L'enregistrement de l'historique des données ou des modifications peut avoir été négligé.

Les données manquantes, en particulier pour les tuples avec des valeurs manquantes pour certains attributs, peuvent devoir être déduites. Généralement, la qualité des données dépend de :

- L’utilisation prévue des données. En effet, deux utilisateurs différents peuvent avoir des évaluations très différentes de la qualité d'une base de données.
- L'actualité qui affecte également la qualité des données.
- La crédibilité qui reflète le degré de confiance des utilisateurs dans les données
- L'interprétabilité qui reflète la facilité avec laquelle les données sont comprises.

3. Prétraitement des données

Dans cette section, nous allons aborder les principales étapes du prétraitement des données, à savoir le nettoyage des données, l'intégration des données, la réduction des données et la transformation des données.

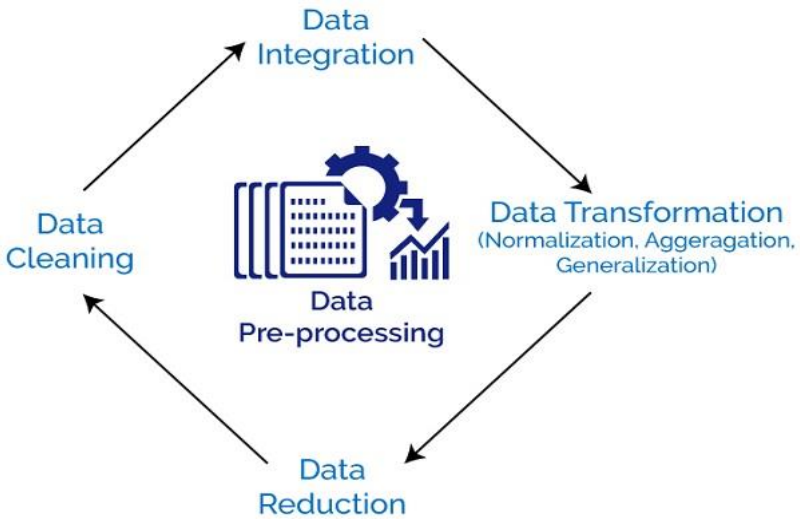

Figure 2.1 Prétraitement des données[22]

3.1. Nettoyage de données

Le processus de nettoyage de données a pour objectif de nettoyer les données en remplissant les valeurs manquantes, en lissant les données bruyantes, en identifiant ou en supprimant les valeurs aberrantes et en résolvant les incohérences. Les données brutes peuvent semer la confusion dans la procédure d'extraction, ce qui entraîne une sortie peu fiable. Bien que la plupart des routines d'exploration de données aient des procédures pour traiter les données incomplètes ou bruyantes, elles ne sont pas toujours robustes. Au lieu de cela, ils peuvent se concentrer sur le fait d'éviter de sur-ajuster les données à la fonction modélisée. Par conséquent, une étape de prétraitement utile consiste à exécuter via certaines routines de nettoyage des données.

[22] https://www.electronicsmedia.info/2017/12/20/what-is-data-preprocessing/

3.1.1. Valeurs manquantes

Les valeurs manquantes sont des tuples n'ayant aucune valeur enregistrée pour un ou plusieurs attributs. Il existe plusieurs manières pour remplir les valeurs manquantes :

- **Ignorer le tuple :** cela se fait généralement lorsque l'étiquette de classe est manquante. Cette méthode n'est pas très efficace, sauf si le tuple contient plusieurs attributs avec des valeurs manquantes. Il est particulièrement mauvais lorsque le pourcentage de valeurs manquantes par attribut varie considérablement. En ignorant le tuple, les valeurs des attributs restants dans le tuple sont ignorées.
- **Remplissage manuel :** en général, cette approche prend du temps et peut ne pas être réalisable compte tenu d'un grand ensemble de données avec de nombreuses valeurs manquantes.
- **Utilisez une constante globale :** remplacez toutes les valeurs d'attribut manquantes par la même constante. Par conséquent, bien que cette méthode soit simple, elle n'est pas infaillible.
- **Utilisez une mesure de tendance centrale pour l'attribut** : (par exemple, la moyenne ou la médiane) pour remplir la valeur manquante.
- **Utilisez l'attribut moyenne ou médiane pour tous les échantillons appartenant à la même classe que le tuple donné**
- **Utilisez la valeur la plus probable :** cela peut être déterminé avec une régression, des outils basés sur l'inférence utilisant un formalisme bayésien ou une induction d'arbre de décision.

Il est important de noter que, dans certains cas, une valeur manquante peut ne pas impliquer une erreur dans les données.

Exemple :

Lors d'une demande de carte de crédit, les candidats peuvent être invités à fournir leur numéro de permis de conduire. Les candidats qui n'ont pas de permis de conduire peuvent naturellement laisser ce champ vide. Les formulaires doivent permettre aux répondants de spécifier des valeurs telles que « sans objet ». Des routines logicielles peuvent également être utilisées pour découvrir

d'autres valeurs nulles (par exemple, "ne sait pas", "?" ou "aucune"). Idéalement, chaque attribut devrait avoir une ou plusieurs règles concernant la condition nulle. Les règles peuvent spécifier si les valeurs nulles sont autorisées ou non et/ou comment ces valeurs doivent être traitées ou transformées. Les champs peuvent également être intentionnellement laissés vides s'ils doivent être fournis à une étape ultérieure du processus métier. Par conséquent, bien que nous puissions faire de notre mieux pour nettoyer les données après leur saisie, une bonne conception de la base de données et de la procédure de saisie des données devrait aider à minimiser le nombre de valeurs manquantes ou d'erreurs en premier lieu.

3.1.2. Les Bruits

Le bruit est une erreur aléatoire ou une variance dans une variable mesurée. Certaines techniques de description statistique de base et les méthodes de visualisation de données peuvent être utilisées pour identifier les valeurs aberrantes, qui peuvent représenter du bruit. Parmi les techniques de lissage des données :

- **Regroupement :** les méthodes de regroupement lissent une valeur de données triée en consultant son "voisinage", c'est-à-dire les valeurs qui l'entourent. Les valeurs triées sont distribuées dans un certain nombre de « buckets » ou bacs. Étant donné que les méthodes de regroupement consultent le voisinage des valeurs, elles effectuent un lissage local. En général, plus la largeur est grande, plus l'effet du lissage est important. Alternativement, les groupes peuvent être de largeur égale, où la plage d'intervalle de valeurs dans chaque groupe est constante.
- **Régression :** le lissage de données peut également être effectué par régression, une technique qui conforme les valeurs des données à une fonction. La régression linéaire consiste à trouver la meilleure ligne pour ajuster deux attributs afin qu'un attribut puisse être utilisé pour prédire l'autre. La régression linéaire multiple est une extension de la régression linéaire, où plus de deux attributs sont impliqués et les données sont ajustées à une surface multidimensionnelle.
- **Analyse des valeurs aberrantes :** les valeurs aberrantes peuvent être détectées par regroupement, par exemple, lorsque des valeurs similaires sont organisées en groupes ou

"clusters". Intuitivement, les valeurs qui ne font pas partie de l'ensemble de clusters peuvent être considérées comme des valeurs aberrantes.

- **L'induction d'arbre de décision :** il s'agit comme une forme de réduction de données pour les méthodes d'exploration de données basées sur la logique qui effectue à plusieurs reprises des comparaisons de valeurs sur des données triées. Les hiérarchies de concepts sont une forme de discrétisation de données qui peut également être utilisée pour le lissage des données.

3.1.3. Processus de nettoyage

Les valeurs manquantes, le bruit et les incohérences contribuent à des données inexactes. La première étape du nettoyage des données en tant que processus est la détection des écarts. Les écarts peuvent être causés par plusieurs facteurs :

- Des formulaires de saisie de données mal conçus qui comportent de nombreux champs facultatifs
- Une erreur humaine dans la saisie des données
- Des erreurs délibérées (par exemple, les répondants ne veulent pas divulguer des informations sur eux-mêmes)
- La dégradation des données (par exemple, des adresses obsolètes) .
- Représentations de données incohérentes
- Une utilisation incohérente des codes.
- Les erreurs dans les dispositifs d'instrumentation qui enregistrent les données
- Les erreurs système.

Des erreurs peuvent également se produire lorsque les données sont (inadéquatement) utilisées à des fins autres que celles initialement prévues. Il peut également y avoir des incohérences dues à l'intégration des données (par exemple, lorsqu'un attribut donné peut avoir des noms différents dans différentes bases de données).

Pour détecter les écarts, il faut d'abord utiliser toutes les connaissances possédées déjà concernant les propriétés des données. Ces connaissances sont appelées métadonnées.

La surcharge de champs est une autre source d'erreur qui se produit généralement lorsque les développeurs compriment de nouvelles définitions d'attributs dans des portions inutilisées (bit) d'attributs déjà définis.

Les données doivent également être examinées en ce qui concerne les règles uniques, les règles consécutives et les règles nulles. Une règle unique indique que chaque valeur de l'attribut donné doit être différente de toutes les autres valeurs de cet attribut. Une règle consécutive indique qu'il ne peut y avoir aucune valeur manquante entre les valeurs les plus basses et les plus élevées pour l'attribut, et que toutes les valeurs doivent également être uniques. Une règle nulle spécifie l'utilisation d'espaces, de points d'interrogation, de caractères spéciaux ou d'autres chaînes qui peuvent indiquer la condition nulle et comment ces valeurs doivent être gérées.

Il existe un certain nombre d'outils commerciaux différents qui peuvent aider à l'étape de détection des écarts. Les outils de nettoyage des données utilisent une simple connaissance du domaine (par exemple, la connaissance des adresses postales et la vérification orthographique) pour détecter les erreurs et apporter des corrections aux données. Ces outils s'appuient sur des techniques d'analyse et de correspondance floue lors du nettoyage de données provenant de plusieurs sources. Les outils d'audit des données détectent les écarts en analysant les données pour découvrir les règles et les relations, et en détectant les données qui violent ces conditions.

Certaines incohérences de données peuvent être corrigées manuellement à l'aide de références externes. Par exemple, les erreurs commises lors de la saisie des données peuvent être corrigées en effectuant un relevé papier. Cependant, la plupart des erreurs nécessiteront des transformations de données. Autrement dit, une fois les écarts trouvés, il faut définir et appliquer une série de transformations pour les corriger.

Des outils commerciaux peuvent aider à l'étape de transformation des données. Les outils de migration de données permettent de spécifier des transformations simples comme remplacer la chaîne « gender » par « sex ». Les outils ETL (extraction/transformation/chargement) permettent aux utilisateurs de spécifier des transformations via une interface utilisateur graphique (GUI). Ces outils ne prennent généralement en charge qu'un ensemble restreint de transformations.

Le processus en deux étapes de détection des écarts et de transformation des données (pour corriger les écarts) se répète. Cependant, ce processus est source d'erreurs et prend du temps. Certaines transformations peuvent introduire davantage de divergences. Certaines divergences imbriquées ne peuvent être détectées qu'après que d'autres ont été corrigées.

Les transformations sont souvent effectuées par lots pendant que l'utilisateur attend sans retour.

Ce n'est qu'une fois la transformation terminée que l'utilisateur peut revenir en arrière et vérifier qu'aucune nouvelle anomalie n'a été créée par erreur. En règle générale, de nombreuses itérations sont nécessaires avant que l'utilisateur ne soit satisfait. Tous les tuples qui ne peuvent pas être automatiquement gérés par une transformation donnée sont généralement écrits dans un fichier sans aucune explication concernant le raisonnement derrière leur échec. Par conséquent, l'ensemble du processus de nettoyage des données souffre également d'un manque d'interactivité.

Une autre approche pour augmenter l'interactivité dans le nettoyage des données est le développement de langages déclaratifs pour la spécification des opérateurs de transformation de données. Ces travaux se concentrent sur la définition d'extensions puissantes de SQL et d'algorithmes permettant aux utilisateurs d'exprimer efficacement les spécifications de nettoyage des données.

3.2. Intégration de données

L'intégration de données est une étape nécessaire dans le Data Mining. Une intégration soignée peut aider à réduire et à éviter les redondances et les incohérences dans l'ensemble de données résultant. Cela peut aider à améliorer la précision et la vitesse du processus d'exploration de données ultérieur. L'hétérogénéité sémantique et la structure des données posent de grands défis à l'intégration des données.

3.2.1. Identification d'entité

Il est probable que la tâche d'analyse de données impliquera l'intégration de données, qui combine des données provenant de plusieurs sources dans un magasin de données cohérent,

comme dans l'entreposage de données. Ces sources peuvent inclure plusieurs bases de données, cubes de données ou fichiers plats.

Un certain nombre de problèmes doivent être pris en compte lors de l'intégration des données. L'intégration de schémas et la mise en correspondance d'objets peuvent être délicates. Le problème se pose lorsqu'on veut faire correspondre des entités équivalentes du monde réel à partir de plusieurs sources de données.

Les métadonnées peuvent être utilisées pour aider à éviter les erreurs dans l'intégration du schéma. Les métadonnées peuvent également être utilisées pour aider à transformer les données.

Lors de la mise en correspondance d'attributs d'une base de données à une autre lors de l'intégration, une attention particulière doit être portée à la structure des données. Cela permet de s'assurer que toutes les dépendances fonctionnelles d'attributs et les contraintes référentielles du système source correspondent à celles du système cible.

3.2.2. Analyse de redondance et de corrélation

La redondance est un autre problème important dans l'intégration des données. Un attribut (tel que le revenu annuel, par exemple) peut être redondant s'il peut être « dérivé » d'un autre attribut ou ensemble d'attributs. Les incohérences dans la dénomination des attributs ou des dimensions peuvent également entraîner des redondances dans l'ensemble de données résultant.

Certaines redondances peuvent être détectées par analyse de corrélation. Étant donné deux attributs, une telle analyse peut mesurer à quel point un attribut implique l'autre, sur la base des données disponibles. Pour les données nominales, on utilise le test X^2 (chi deux). Pour les attributs numériques, on peut utiliser le coefficient de corrélation et la covariance, qui permettent tous deux d'accéder à la manière dont les valeurs d'un attribut varient par rapport à celles d'un autre.

3.2.3. Duplication de tuple

En plus de détecter les redondances entre les attributs, la duplication doit également être détectée au niveau du tuple. L'utilisation de tables dénormalisées est une autre source de redondance des données. Des incohérences surviennent souvent entre divers doublons, en raison d'une saisie de données inexacte ou de la mise à jour de certaines occurrences de données, mais pas de toutes.

3.2.4. Détection et résolution des conflits

L'intégration de données implique également la détection et la résolution des conflits de valeurs de données. Par exemple, pour la même entité du monde réel, les valeurs d'attribut de différentes sources peuvent différer.

Cela peut être dû à des différences de représentation, d'échelle ou d'encodage. Les attributs peuvent également différer au niveau de l'abstraction, où un attribut dans un système est enregistré à, disons, un niveau d'abstraction inférieur au « même » attribut dans un autre.

3.3. Réduction de données

La réduction de données permet d'obtenir une représentation réduite de l'ensemble de données qui est beaucoup plus petit en volume, mais produit les mêmes résultats analytiques. Les stratégies de réduction de données comprennent la réduction de la dimensionnalité et la réduction de la numéroté.

3.3.1. Réduction de la dimensionnalité

Dans la réduction de la dimensionnalité, des schémas de codage de données sont appliqués de manière à obtenir une représentation réduite ou compressée des données d'origine incluent :

- Les techniques de compression de données (par exemple, les transformations en ondelettes et l'analyse des composantes principales),
- La sélection de sous-ensembles d'attributs (par exemple, la suppression des attributs non pertinents)
- La construction d'attributs (par exemple, lorsqu'un petit ensemble d'attributs plus utiles est dérivé de l'ensemble d'origine).

A. Transformées en ondelettes

La transformée discrète en ondelettes (DWT) est une technique de traitement de signal linéaire qui, lorsqu'elle est appliquée à un vecteur de données X, le transforme en un vecteur numériquement différent, X', de coefficients d'ondelettes. Les deux vecteurs sont de même longueur. Lors de l'application de cette technique à la réduction de données, nous considérons chaque tuple comme un vecteur de données à n dimensions, c'est-à-dire $X = (x_1, x_2, \ldots, x_n)$, représentant n mesures effectuées sur le tuple à partir de n attributs de base de données.

L'utilité de cette technique réside dans le fait que les données transformées en ondelettes peuvent être tronquées. Une approximation compressée des données peut être conservée en ne stockant qu'une petite fraction du plus fort des coefficients d'ondelettes.

La représentation des données résultante est donc très éparpillée, de sorte que les opérations qui peuvent tirer parti de la rareté des données sont très rapides en termes de calcul si elles sont effectuées dans l'espace des ondelettes. La technique fonctionne également pour supprimer le bruit sans lisser les principales caractéristiques des données, ce qui la rend également efficace pour le nettoyage des données. Étant donné un ensemble de coefficients, une approximation des données d'origine peut être construite en appliquant l'inverse du DWT utilisé.

La DWT est étroitement liée à la transformée de Fourier discrète (DFT), une technique de traitement du signal impliquant des sinus et des cosinus. En général, le DWT réalise une meilleure compression avec perte. Autrement dit, si le même nombre de coefficients est retenu pour un DWT et un DFT d'un vecteur de données donné, la version DWT fournira une approximation plus précise des données d'origine. Ainsi, pour une approximation équivalente, la DWT nécessite moins d'espace que la DFT. Contrairement à la DFT, les ondelettes sont assez localisées dans l'espace, contribuant à la conservation des détails locaux.

La procédure générale d'application d'une transformée en ondelettes discrètes utilise un algorithme de pyramide hiérarchique qui divise par deux les données à chaque itération, ce qui entraîne une vitesse de calcul rapide. La méthode est la suivante :

1. La longueur, L, du vecteur de données d'entrée doit être une puissance entière de 2. Cette condition peut être satisfaite en complétant le vecteur de données avec des zéros si nécessaire (L≥ n).
2. Chaque transformation implique l'application de deux fonctions. La première applique un certain lissage des données, comme une somme ou une moyenne pondérée. La seconde effectue une différence pondérée, qui agit pour faire ressortir les caractéristiques détaillées des données.
3. Les deux fonctions sont appliquées à des paires de points de données dans X, c'est-à-dire à toutes les paires de mesures (x_{2i} ,x_{2i+1}). Il en résulte deux ensembles de données de longueur L/2. En général, ceux-ci représentent une version lissée ou basse fréquence des données d'entrée et leur contenu haute fréquence, respectivement.
4. Les deux fonctions sont appliquées récursivement aux jeux de données obtenus dans la boucle précédente, jusqu'à ce que les jeux de données résultants obtenus soient de longueur 2.
5. Les valeurs sélectionnées à partir des ensembles de données obtenus dans les itérations précédentes sont appelées les coefficients d'ondelettes des données transformées.

De manière équivalente, une multiplication matricielle peut être appliquée aux données d'entrée afin d'obtenir les coefficients d'ondelettes, la matrice utilisée dépendant de la DWT donnée. La matrice doit être orthonormée, ce qui signifie que les colonnes sont des vecteurs unitaires et sont mutuellement orthogonales, de sorte que l'inverse de la matrice n'est que sa transposée.

Les transformées en ondelettes peuvent être appliquées à des données multidimensionnelles telles qu'un cube de données. Cela se fait en appliquant d'abord la transformation à la première dimension, puis à la seconde, et ainsi de suite. La complexité de calcul impliquée est linéaire par rapport au nombre de cellules dans le cube. Les transformées en ondelettes donnent de bons résultats sur des données éparpillées ou asymétriques et sur des données avec des attributs ordonnés. La compression avec perte par ondelettes serait meilleure que la compression JPEG, la norme commerciale actuelle. Les transformées en ondelettes ont de nombreuses applications

dans le monde réel, notamment la compression d'images d'empreintes digitales, la vision par ordinateur, l'analyse de données de séries chronologiques et le nettoyage des données.

B. Analyse en composantes principales

L'analyse en composantes principales (ACP ; également appelée méthode de Karhunen-Loeve ou K-L) recherche les vecteurs orthogonaux k n-dimensionnels qui peuvent être utilisés au mieux pour représenter les données, où k ≤ n. Les données d'origine sont ainsi projetées sur un espace beaucoup plus petit, ce qui entraîne une réduction de la dimensionnalité. L'ACP combine l'essence des attributs en créant un ensemble de variables alternatif plus petit. Les données initiales peuvent alors être projetées sur cet ensemble réduit. L'ACP révèle souvent des relations qui n'étaient pas soupçonnées auparavant et permet ainsi des interprétations qui n'en résulteraient pas normalement. La procédure de base est la suivante :

1. Les données d'entrée sont normalisées, de sorte que chaque attribut se situe dans la même plage. Cette étape permet de s'assurer que les attributs avec de grands domaines ne domineront pas les attributs avec des domaines plus petits.
2. Calculer les k vecteurs orthonormés qui fournissent une base pour les données d'entrée normalisées. Ce sont des vecteurs unitaires qui pointent chacun dans une direction perpendiculaire aux autres. Ces vecteurs sont appelés les composantes principales. Les données d'entrée sont une combinaison linéaire des composantes principales.
3. Les principales composantes sont triées par ordre décroissant d'importance ou de force. Les composantes principales servent essentiellement de nouvel ensemble d'axes pour les données, fournissant des informations importantes sur la variance. C'est-à-dire que les axes triés sont tels que le premier axe montre la plus grande variance parmi les données, le deuxième axe montre la prochaine variance la plus élevée, et ainsi de suite.
4. Étant donné que les composantes sont triées par ordre décroissant de, la taille des données peut être réduite en éliminant les composantes les plus faibles, c'est-à-dire celles dont la variance est faible.

En utilisant les composantes principales les plus fortes, il devrait être possible de reconstruire une bonne approximation des données originales.

L'ACP peut être appliquée à des attributs ordonnés et non ordonnés et peut gérer des données éparses et des données asymétriques. Les données multidimensionnelles de plus de deux dimensions peuvent être traitées en réduisant le problème à deux dimensions. Les composantes principales peuvent être utilisées comme données d'entrée pour la régression multiple et l'analyse par clusters. Par rapport aux transformées en ondelettes, l'ACP a tendance à mieux gérer les données éparses, tandis que les transformées en ondelettes conviennent mieux aux données de haute dimensionnalité.

C. Sélection de sous-ensemble d'attributs

Les ensembles de données à analyser peuvent contenir des centaines d'attributs, dont beaucoup peuvent être sans rapport avec la tâche d'exploration ou redondants. Bien qu'il puisse être possible pour un expert du domaine de sélectionner certains des attributs utiles, cela peut être une tâche difficile et longue, en particulier lorsque le comportement des données n'est pas bien connu. Cela peut entraîner la découverte de modèles de mauvaise qualité. De plus, le volume supplémentaire d'attributs non pertinents ou redondants peut ralentir le processus de minage.

La sélection de sous-ensembles d'attributs réduit la taille de l'ensemble de données en supprimant les attributs ou dimensions non pertinents ou redondants. L'objectif de la sélection de sous-ensembles d'attributs est de trouver un ensemble minimum d'attributs tel que la distribution de probabilité résultante des classes de données soit aussi proche que possible de la distribution d'origine obtenue à l'aide de tous les attributs.

L'exploration d'un ensemble réduit d'attributs présente un avantage supplémentaire : elle réduit le nombre d'attributs apparaissant dans les modèles découverts, ce qui facilite la compréhension des modèles.

Pour trouver le bon sous-ensemble des attributs d'origine pour n attributs, il existe 2^n sous-ensembles possibles. Une recherche exhaustive du sous-ensemble optimal d'attributs peut être d'un coût prohibitif, d'autant plus que n et le nombre de classes de données augmentent. Par

conséquent, les méthodes heuristiques qui explorent un espace de recherche réduit sont couramment utilisées pour la sélection de sous-ensembles d'attributs. Ces méthodes sont généralement gourmandes en ce sens que, lors de la recherche dans l'espace d'attributs, elles font toujours ce qui semble être le meilleur choix à un moment donné. Leur stratégie est de faire un choix localement optimal dans l'espoir que cela conduira à une solution globalement optimale. Ces méthodes gourmandes sont efficaces en pratique et peuvent se rapprocher de l'estimation d'une solution optimale.

Les meilleurs attributs sont généralement déterminés à l'aide de tests de signification statistique, qui supposent que les attributs sont indépendants les uns des autres. De nombreuses autres mesures d'évaluation d'attributs peuvent être utilisées, telles que la mesure de gain d'informations utilisée dans la construction d'arbres de décision pour la classification

Les méthodes heuristiques de base de la sélection de sous-ensembles d'attributs incluent les techniques qui suivent :

1. **Stepwise forward selection:** la procédure commence avec un ensemble vide d'attributs comme ensemble réduit. Le meilleur des attributs d'origine est déterminé et ajouté à l'ensemble réduit. À chaque itération ou étape suivante, le meilleur des attributs d'origine restants est ajouté à l'ensemble.
2. **Stepwise backward elimination:** la procédure commence avec l'ensemble complet d'attributs. A chaque étape, il supprime le pire attribut restant dans l'ensemble.
3. **Combinaison :** Les méthodes de sélection en avant pas à pas et d'élimination en arrière peuvent être combinées de sorte qu'à chaque étape, la procédure sélectionne le meilleur attribut et supprime le pire parmi les attributs restants.
4. **Arbre de décision :** les algorithmes d'arbre de décision (par exemple, ID3, C4.5 et CART) étaient initialement destinés à la classification. L'induction d'arbre de décision construit une structure de type organigramme où chaque nœud interne désigne un test sur un attribut, chaque branche correspond à un résultat du test et chaque nœud externe désigne une prédiction de classe. À chaque nœud, l'algorithme choisit le "meilleur" attribut pour partitionner les données en classes individuelles. Lorsque l'induction d'arbre

de décision est utilisée pour la sélection de sous-ensembles d'attributs, un arbre est construit à partir des données. Tous les attributs qui n'apparaissent pas dans l'arborescence sont supposés non pertinents. L'ensemble d'attributs apparaissant dans l'arbre forme le sous-ensemble réduit d'attributs.

Les critères d'arrêt des méthodes peuvent varier. La procédure peut employer un seuil sur la mesure utilisée pour déterminer quand arrêter le processus de sélection d'attribut.

Dans certains cas, nous pouvons vouloir créer de nouveaux attributs basés sur d'autres. Une telle construction d'attributs peut aider à améliorer la précision et la compréhension de la structure des données de grande dimension.

3.3.2. Réduction du nombre

Dans la réduction du nombre, les données sont remplacées par des représentations alternatives plus petites utilisant des modèles paramétriques tels que les modèles de régression et log-linéaires ou des modèles non paramétriques tels que les histogrammes, les clusters, un échantillonnage ou une agrégation de données.

A. Modèles de régression et log-linéaires

Des modèles de régression et log-linéaires peuvent être utilisés pour approximer les données. Dans la régression linéaire (simple), les données sont modélisées pour s'adapter à une ligne droite. Une variable aléatoire, y (appelée variable de réponse), peut être modélisée comme une fonction linéaire d'une autre variable aléatoire, x (appelée variable prédictive), avec l'équation

$$Y = wx + b$$

où la variance de y est supposée constante. Dans le contexte de l'exploration de données, x et y sont des attributs de base de données numériques. Les coefficients, w et b (appelés coefficients de régression), spécifient respectivement la pente de la droite et l'ordonnée à l'origine. Ces coefficients peuvent être résolus par la méthode des moindres carrés, qui minimise l'erreur entre la ligne réelle séparant les données et l'estimation de la ligne. La régression linéaire multiple est une extension de la régression linéaire (simple), qui permet de modéliser une variable de réponse, y, comme une fonction linéaire de deux variables prédictives ou plus.

Les modèles log-linéaires se rapprochent des distributions de probabilités multidimensionnelles discrètes. Étant donné un ensemble de tuples à n dimensions, on peut considérer chaque tuple comme un point dans un espace à n dimensions. Les modèles log-linéaires peuvent être utilisés pour estimer la probabilité de chaque point dans un espace multidimensionnel pour un ensemble d'attributs discrétisés, sur la base d'un sous-ensemble plus petit de combinaisons dimensionnelles. Cela permet de construire un espace de données de dimension supérieure à partir d'espaces de dimension inférieure.

Les modèles log-linéaires sont donc également utiles pour :

- La réduction de la dimensionnalité puisque les points de dimension inférieure occupent ensemble moins d'espace que les points de données d'origine
- Le lissage des données puisque les estimations agrégées dans l'espace de dimension inférieure sont moins sujettes aux variations d'échantillonnage que les estimations dans l'espace de dimension supérieure.

Les modèles de régression et log-linéaire peuvent tous deux être utilisés sur des données éparpillées, bien que leur application puisse être limitée. Alors que les deux méthodes peuvent gérer des données asymétriques, la régression fonctionne exceptionnellement bien. La régression peut être gourmande en calcul lorsqu'elle est appliquée à des données de grande dimension, tandis que les modèles log-linéaires présentent une bonne évolutivité jusqu'à environ 10 dimensions.

B. Histogrammes

Les histogrammes utilisent le regroupement pour approximer les distributions de données et constituent une forme populaire de réduction des données. Un histogramme pour un attribut, A, partitionne la distribution des données de A en sous-ensembles disjoints, appelés compartiments ou bacs. Si chaque compartiment ne représente qu'une seule paire attribut-valeur/fréquence, les compartiments sont appelés compartiments singleton. Souvent, les compartiments représentent à la place des plages continues pour l'attribut donné.

Les histogrammes sont très efficaces pour approximer les données éparpillées et denses, ainsi que les données fortement asymétriques et uniformes. Les histogrammes décrits précédemment pour des attributs uniques peuvent être étendus pour plusieurs attributs. Les histogrammes multidimensionnels peuvent capturer les dépendances entre les attributs. Ces histogrammes se sont avérés efficaces pour approximer les données avec jusqu'à cinq attributs. D'autres études sont nécessaires concernant l'efficacité des histogrammes multidimensionnels pour les hautes dimensionnalités. Les compartiments singleton sont utiles pour stocker les valeurs aberrantes à haute fréquence.

C. Clustering

Les techniques de clustering considèrent les tuples de données comme des objets. Ils partitionnent les objets en groupes de sorte que les objets d'un cluster soient similaires les uns aux autres et différents des objets des autres clusters. La similarité est généralement définie en termes de proximité des objets dans l'espace, sur la base d'une fonction de distance. La qualité d'un cluster peut être représentée par son diamètre, la distance maximale entre deux objets quelconques dans le cluster. La distance centroïde est une mesure alternative de la qualité du cluster et est définie comme la distance moyenne de chaque objet du cluster par rapport au centroïde du cluster

Dans la réduction des données, les représentations en cluster des données sont utilisées pour remplacer les données réelles. L'efficacité de cette technique dépend de la nature des données. Il est beaucoup plus efficace pour les données qui peuvent être organisées en clusters distincts que pour les données étalées.

D. Échantillonnage

L'échantillonnage peut être utilisé comme technique de réduction des données car il permet de représenter un grand ensemble de données par un sous-ensemble de données aléatoires beaucoup plus petit. Supposons qu'un grand ensemble de données, D, contienne N tuples. Les façons les plus courantes pour échantillonner D pour la réduction des données sont :

- **Échantillon aléatoire simple sans remise de taille s :** il est créé en tirant s des N tuples de D (s < N), où la probabilité de tirer n'importe quel tuple dans D est 1/N, c'est-à-dire que tous les tuples sont également susceptibles d'être échantillonnés.
- **Échantillon aléatoire simple avec remplacement de taille s :** Ceci est similaire au premier, sauf que chaque fois qu'un tuple est tiré de D, il est enregistré puis remplacé. C'est-à-dire qu'après qu'un tuple est dessiné, il est replacé dans D afin qu'il puisse être dessiné à nouveau.
- **Échantillon de cluster :** si les tuples de D sont regroupés en M clusters mutuellement disjoints, alors un échantillon aléatoire simple de s clusters peut être obtenu, où s < M.
- **Échantillon stratifié :** si D est divisé en parties mutuellement disjointes appelées strates, un échantillon stratifié(accumulé) de D est généré en obtenant un échantillon aléatoire simple à chaque strate. Cela permet de garantir un échantillon représentatif, en particulier lorsque les données sont biaisées.

Un avantage de l'échantillonnage pour la réduction des données est que le coût d'obtention d'un échantillon est proportionnel à la taille de l'échantillon s par opposition à N la taille de l'ensemble de données. Par conséquent, la complexité de l'échantillonnage est potentiellement sous-linéaire à la taille des données. D'autres techniques de réduction de données peuvent nécessiter au moins un passage complet à travers D. Pour une taille d'échantillon fixe, la complexité d'échantillonnage n'augmente que de manière linéaire à mesure que le nombre de dimensions de données n augmente, tandis que les techniques utilisant des histogrammes, par exemple, augmentent de manière exponentielle en n .

Lorsqu'il est appliqué à la réduction des données, l'échantillonnage est le plus souvent utilisé pour estimer la réponse à une requête agrégée. Il est possible de déterminer une taille d'échantillon suffisante pour estimer une fonction donnée avec un degré d'erreur spécifié. Cette taille d'échantillon s peut être extrêmement petite par rapport à N. L'échantillonnage est un choix naturel pour l'affinement progressif d'un ensemble de données réduit. Un tel ensemble peut être encore affiné en augmentant simplement la taille de l'échantillon.

E. Agrégation de cubes de données

Les cubes de données stockent des informations agrégées multidimensionnelles. Chaque cellule contient une valeur de données agrégée, correspondant au point de données dans l'espace multidimensionnel. Des hiérarchies de concepts peuvent exister pour chaque attribut, permettant l'analyse des données à plusieurs niveaux d'abstraction. Les cubes de données offrent un accès rapide à des données résumées précalculées, bénéficiant ainsi du traitement analytique en ligne ainsi que de l'exploration de données.

Le cube créé au niveau d'abstraction le plus bas est appelé cuboïde de base. Le cuboïde de base doit correspondre à une entité individuelle d'intérêt telle que les ventes ou le client dans la gestion des magasins.

En d'autres termes, le niveau le plus bas doit être utilisable ou utile pour l'analyse. Un cube au plus haut niveau d'abstraction est le cuboïde apex. Les cubes de données créés pour différents niveaux d'abstraction sont souvent appelés cuboïdes, de sorte qu'un cube de données peut à la place faire référence à un réseau de cuboïdes. Chaque niveau d'abstraction supérieur réduit encore la taille des données résultantes. Lors de la réponse aux demandes d'exploration de données, le plus petit cuboïde disponible correspondant à la tâche donnée doit être utilisé.

3.3.3. Transformation de données

Dans cette étape de prétraitement, les données sont transformées ou consolidées afin que le processus d'exploration résultant soit plus efficace et que les modèles trouvés soient plus faciles à comprendre. La discrétisation des données, une forme de transformation des données, est également abordée.

A. Présentation des stratégies de transformation des données

Les stratégies de transformation des données incluent les éléments suivants :

- **Le lissage :** il permet de supprimer les bruits des données. Les techniques incluent le binning, la régression et le clustering.
- **Construction d'attributs :** nouveaux attributs sont construits et ajoutés à partir de l'ensemble d'attributs donné pour faciliter le processus d'exploration.

- **Agrégation :** des opérations de synthèse ou d'agrégation sont appliquées aux données. Cette étape est généralement utilisée dans la construction d'un cube de données pour l'analyse de données à plusieurs niveaux d'abstraction.
- **Normalisation :** les données d'attribut sont mises à l'échelle de manière à se situer dans une plage plus petite.
- **Discrétisation :** les valeurs brutes d'un attribut numérique sont remplacées par des étiquettes d'intervalle ou des étiquettes conceptuelles. Les étiquettes, à leur tour, peuvent être organisées de manière récursive en concepts de niveau supérieur, résultant en une hiérarchie de concepts pour l'attribut numérique.
- **Génération de la hiérarchie des concepts :** les attributs tels que la rue peuvent être généralisés à des concepts de niveau supérieur, comme la ville ou le pays. De nombreuses hiérarchies d'attributs nominaux sont implicites dans le schéma de la base de données et peuvent être définies automatiquement au niveau de la définition du schéma.

Les techniques de discrétisation peuvent être classées en fonction de la manière dont la discrétisation est effectuée, par exemple si elle utilise des informations de classe ou dans quelle direction elle procède (c'est-à-dire, de haut en bas ou de bas en haut). Si le processus de discrétisation utilise des informations de classe, alors on dit qu'il s'agit d'une discrétisation supervisée. Sinon, il est non supervisé. Si le processus commence par trouver d'abord un ou quelques points (appelés points de fractionnement ou points de coupure) pour diviser toute la plage d'attributs, puis le répète de manière récursive sur les intervalles résultants, cela s'appelle une discrétisation ou une division descendante. Cela contraste avec la discrétisation ou la fusion ascendante, qui commence par considérer toutes les valeurs continues comme des points de partage potentiels, en supprime certaines en fusionnant des valeurs de voisinage pour former des intervalles, puis applique de manière récursive ce processus aux intervalles résultants.

B. Transformation des données par normalisation

L'unité de mesure utilisée peut affecter l'analyse des données. Par exemple, changer les unités de mesure de mètres en pouces pour la taille, ou de kilogrammes en livres pour le poids, peut conduire à des résultats très différents. En général, l'expression d'un attribut dans des unités plus

petites conduira à une plage plus large pour cet attribut, et aura donc tendance à donner à un tel attribut un plus grand effet ou "poids". Pour éviter de dépendre du choix des unités de mesure, les données doivent être normalisées ou standardisées. Cela implique de transformer les données pour qu'elles se situent dans une plage plus petite ou commune telle que [0, 1]. Les termes standardiser et normaliser sont utilisés de manière interchangeable dans le prétraitement des données, bien qu'en statistique, ce dernier terme ait également d'autres connotations.

La normalisation des données tente de donner à tous les attributs un poids égal. La normalisation est particulièrement utile pour les algorithmes de classification impliquant des réseaux de neurones ou des mesures de distance telles que la classification et le clustering du plus proche voisin.

Il existe de nombreuses méthodes de normalisation des données. Les plus utilisées sont : min-max, z-score et par mise à l'échelle décimale. Soit A un attribut numérique avec n valeurs observées, $v_1, v_2, \dots, v_n$.

- **La normalisation min-max :** Effectue une transformation linéaire sur les données d'origine. Supposons que min_A et max_A soient les valeurs minimale et maximale d'un attribut, A. La normalisation min-max mappe une valeur v_i de A à v'_i dans la plage [nouveau min_A, nouveau max_A] en calculant

$$v'_i = \frac{v_i - min_A}{max_A - min_A} \ (nouveau\ min_A, nouveau\ max_A) + nouveau\ min_A$$

  La normalisation min-max préserve les relations entre les valeurs de données d'origine. Le problème avec cette normalisation est qu'une erreur "hors limites" peut se produire si un futur cas d'entrée pour la normalisation tombe en dehors de la plage de données d'origine pour A.
- **La normalisation z-score :** les valeurs d'un attribut A sont normalisées en fonction de la moyenne et de l'écart type de A. Une valeur v_i de A est normalisée à v'_i en calculant

$$v'_i = \frac{v_i - \mu}{\sigma}$$

où μ et σ sont respectivement la moyenne et l'écart type de l'attribut A. Cette méthode de normalisation est utile lorsque le minimum et le maximum réels de l'attribut A sont inconnus, ou lorsqu'il existe des valeurs aberrantes qui dominent la normalisation min-max. Une variation de cette normalisation du score z remplace l'écart type par l'écart absolu moyen de A. L'écart absolu moyen de A, noté s_A, est

$$s_A = \frac{1}{n}(|v_1 - \mu| + |v_2 - \mu| + \cdots + |v_n - \mu|)$$

Ainsi, la normalisation du score z à l'aide de l'écart absolu moyen est

$$v'_i = \frac{v_i - \mu}{s_A}$$

L'écart absolu moyen s_A est plus robuste aux valeurs aberrantes que l'écart type σ. Lors du calcul de l'écart absolu moyen, les écarts par rapport à la moyenne ne sont pas quadrillés ; par conséquent, l'effet des valeurs aberrantes est quelque peu réduit.

- **La normalisation par mise à l'échelle décimale** : permet de normaliser en déplaçant la virgule décimale des valeurs de l'attribut A. Le nombre de virgules décimales déplacées dépend de la valeur absolue maximale de A. Une valeur v_i de A est normalisée à v'_i en calculant

$$v'_i = \frac{v_i - \mu}{10^j}$$

Ou j est l'entier le plus petit tel que $\max(|v'_i| < 1$

Notez que la normalisation peut modifier considérablement les données d'origine, en particulier lors de l'utilisation de la normalisation du z-score ou de la mise à l'échelle décimale. Il est également nécessaire d'enregistrer les paramètres de normalisation afin que les données futures puissent être normalisées de manière uniforme.

C. Discrétisation par Binning

Le regroupement est une technique de fractionnement descendante basée sur un nombre spécifié de groupes. Les méthodes de lissage de données sont également utilisées pour discrétisation, la réduction de données et la génération de hiérarchies de concepts. Par exemple,

les valeurs d'attribut peuvent être discrétisées en appliquant un regroupement à largeur égale ou à fréquence égale, puis en remplaçant chaque valeur de compartiment par la moyenne ou la médiane de compartiment, comme dans le lissage par les moyens de compartiment ou le lissage par les médianes de compartiment, respectivement. Ces techniques peuvent être appliquées de manière récursive aux partitions résultantes pour générer des hiérarchies de concepts. Le binning n'utilise pas d'informations de classe et est donc une technique de discrétisation non supervisée. Il est sensible au nombre de bins spécifié par l'utilisateur, ainsi qu'à la présence de valeurs aberrantes.

D. Discrétisation par analyse d'histogramme

Comme le binning, l'analyse d'histogramme est une technique de discrétisation non supervisée car elle n'utilise pas d'informations de classe. Un histogramme partitionne les valeurs d'un attribut en plages disjointes appelées buckets ou bins. Diverses règles de partitionnement peuvent être utilisées pour définir les histogrammes. Dans un histogramme de largeur égale les valeurs sont partitionnées en plages de taille égale. Avec un histogramme à fréquence égale, les valeurs sont partitionnées de sorte que chaque partition contienne le même nombre de tuples de données. L'algorithme d'analyse d'histogramme peut être appliqué de manière récursive à chaque partition afin de générer automatiquement une hiérarchie de concepts à plusieurs niveaux. La procédure se terminant une fois qu'un nombre prédéfini de niveaux de concepts a été atteint. Une taille d'intervalle minimale peut également être utilisée par niveau pour contrôler la procédure récursive. Cela spécifie la largeur minimale d'une partition ou le nombre minimal de valeurs pour chaque partition à chaque niveau. Les histogrammes peuvent également être partitionnés en fonction de l'analyse par cluster de la distribution de données.

E. Discrétisation par cluster

L'analyse de cluster est une méthode populaire de discrétisation de données. Un algorithme de clustering peut être appliqué pour discrétiser un attribut numérique A en partitionnant les valeurs de A en clusters ou groupes. Le clustering prend en considération la distribution de A, ainsi que la proximité des points de données, et est donc capable de produire des résultats de discrétisation de haute qualité. Le clustering peut être utilisé pour générer une hiérarchie de

concepts pour A en suivant soit une stratégie de fractionnement descendante, soit une stratégie de fusion ascendante, où chaque cluster forme un nœud de la hiérarchie de concepts. Dans le premier cas, chaque cluster ou partition initial peut être décomposé en plusieurs sous-clusters, formant un niveau inférieur de la hiérarchie. Dans ce dernier cas, les clusters sont formés en regroupant à plusieurs reprises des clusters voisins afin de former des concepts de niveau supérieur.

F. Discrétisation par arbre de décision

Les techniques de génération d'arbres de décision pour la classification peuvent être appliquées à la discrétisation. Ces techniques utilisent une approche de fractionnement descendante. Contrairement aux autres méthodes mentionnées jusqu'à présent, les approches de discrétisation par arbre de décision sont supervisées, c'est-à-dire qu'elles utilisent des informations d'étiquette de classe. L'idée principale est de sélectionner des points de partage afin qu'une partition résultante donnée contienne autant de tuples de la même classe que possible. L'entropie est la mesure la plus couramment utilisée à cette fin. Pour discrétiser un attribut numérique A, le procédé sélectionne la valeur de A qui a l'entropie minimale comme point de partage, et partitionne de manière récursive les intervalles résultants pour arriver à une discrétisation hiérarchique. Une telle discrétisation forme une hiérarchie de concepts pour A. Étant donné que la discrétisation basée sur l'arbre de décision utilise des informations de classe, il est plus probable que les limites d'intervalle (points de division) soient définies pour se produire à des endroits qui peuvent aider à améliorer la précision de la classification.

G. Discrétisation par analyses de corrélation

Des mesures de corrélation peuvent être utilisées pour la discrétisation. ChiMerge est une méthode de discrétisation basée sur X^2. Les méthodes de discrétisation que nous avons étudiées jusqu'à présent ont toutes utilisé une stratégie de découpage descendante. Cela contraste avec ChiMerge, qui utilise une approche ascendante en trouvant les meilleurs intervalles voisins, puis en les fusionnant pour former des intervalles plus grands, de manière récursive. Comme pour l'analyse d'arbre de décision, ChiMerge est supervisé en ce sens qu'il utilise des informations de classe. La notion de base est que pour une discrétisation précise, les fréquences de classe

relatives doivent être assez cohérentes dans un intervalle. Par conséquent, si deux intervalles adjacents ont une distribution de classes très similaire, alors les intervalles peuvent être fusionnés. Sinon, ils doivent rester séparés.

ChiMerge procède comme suit. Initialement, chaque valeur distincte d'un attribut numérique A est considérée comme un intervalle. Les tests X^2 sont effectués pour chaque paire d'intervalles adjacents. Les intervalles adjacents avec les valeurs les plus faibles de X^2 sont fusionnés, car les valeurs faibles de X^2 pour une paire indiquent des distributions de classe similaires. Ce processus de fusion se déroule de manière récursive jusqu'à ce qu'un critère d'arrêt prédéfini soit satisfait.

4. Génération de la hiérarchie des concepts pour les données nominales

Les attributs nominaux ont un nombre fini de valeurs distinctes, sans ordre parmi les valeurs. La définition manuelle des hiérarchies de concepts peut être une tâche fastidieuse pour un utilisateur ou un expert du domaine. Heureusement, de nombreuses hiérarchies sont implicites dans le schéma de la base de données et peuvent être définies automatiquement au niveau de la définition du schéma. Les hiérarchies de concepts peuvent être utilisées pour transformer les données en plusieurs niveaux de granularité.

En générale, il existe quatre méthodes pour la génération de hiérarchies de concepts pour des données nominales :

- **Spécification d'un classement partiel des attributs** : Les hiérarchies conceptuelles pour les attributs nominaux ou les dimensions impliquent généralement un groupe d'attributs. Un utilisateur ou un expert peut facilement définir une hiérarchie de concepts en spécifiant un ordre partiel ou total des attributs au niveau du schéma.
- **Spécification d'une partie d'une hiérarchie par regroupement explicite de données :** Il s'agit essentiellement de la définition manuelle d'une partie d'une hiérarchie de concepts. Dans une grande base de données, il n'est pas réaliste de définir une hiérarchie de concepts entière par une énumération explicite des valeurs. Au contraire, on peut

facilement spécifier des regroupements explicites pour une petite partie des données de niveau intermédiaire.

- **Spécification d'un ensemble d'attributs sans ordre partiel** : Un utilisateur peut spécifier un ensemble d'attributs formant une hiérarchie de concepts, mais omettre d'énoncer explicitement leur ordre partiel. Le système peut alors essayer de générer automatiquement l'ordre des attributs afin de construire une hiérarchie de concepts significative.
- **Spécification d'un ensemble partiel d'attributs** : parfois, un utilisateur peut être négligent lors de la définition d'une hiérarchie ou n'avoir qu'une vague idée de ce qui doit être inclus dans une hiérarchie. Par conséquent, l'utilisateur peut n'avoir inclus qu'un petit sous-ensemble des attributs pertinents dans la spécification de la hiérarchie. Pour gérer ces hiérarchies partiellement spécifiées, il est important d'intégrer la sémantique des données dans le schéma de la base de données afin que les attributs avec des connexions sémantiques étroites puissent être épinglés. De cette manière, la spécification d'un attribut peut déclencher l'insertion de tout un groupe d'attributs sémantiquement étroitement liés pour former une hiérarchie complète. Les utilisateurs doivent cependant avoir la possibilité de remplacer cette fonctionnalité, si nécessaire.

Chapitre III : Exploration de Modèles Fréquents, d'Associations et de Corrélations

1 Définition

Les modèles fréquents sont des modèles qui apparaissent fréquemment dans un ensemble de données :

- Un ensemble d'éléments : tels que le lait et le pain, qui apparaissent fréquemment ensemble dans un ensemble de données de transaction est un ensemble d'éléments fréquents.
- Une sous-séquence : telle que l'achat d'abord d'un PC, puis d'un appareil photo numérique, puis d'une carte mémoire, si elle se produit fréquemment dans une base de données d'historique d'achats, est un schéma séquentiel (fréquent).
- Une sous-structure : peut faire référence à différentes formes structurelles, telles que des sous-graphes, des sous-arbres ou des sous-réseaux, qui peuvent être combinés avec des ensembles d'éléments ou de sous-séquences. Si une sous-structure se produit fréquemment, on l'appelle un modèle structuré (fréquent).

La recherche de modèles fréquents joue un rôle essentiel dans les associations minières, les corrélations et de nombreuses autres relations intéressantes entre les données. De plus, il aide à la classification des données, au clustering et à d'autres tâches d'exploration de données. Ainsi, l'exploration fréquente de modèles est devenue une tâche importante d'exploration de données et un thème central de la recherche en exploration de données.

2. Motivation

L'exploration fréquente d'ensembles d'éléments conduit à la découverte d'associations et de corrélations entre des éléments dans de grands ensembles de données transactionnelles ou relationnelles. Avec des quantités massives de données continuellement collectées et stockées, de nombreuses industries s'intéressent à l'extraction de ces modèles à partir de leurs bases de

données. La découverte de relations de corrélation intéressantes entre d'énormes quantités d'enregistrements de transactions commerciales peut aider dans de nombreux processus décisionnels commerciaux tels que la conception de catalogues, le marketing croisé et l'analyse du comportement d'achat des clients.

Un exemple typique d'extraction fréquente d'itemsets est l'analyse du panier de consommation. Ce processus analyse les habitudes d'achat des clients en trouvant des associations entre les différents articles que les clients placent dans leurs « paniers ». La découverte de ces associations peut aider les détaillants à développer des stratégies de marketing en obtenant un aperçu des articles fréquemment achetés ensemble par les clients.

Si on considère l'univers comme l'ensemble des articles disponibles dans un magasin, alors chaque article a une variable booléenne représentant la présence ou l'absence de cet article. Chaque panier peut alors être représenté par un vecteur booléen de valeurs affectées à ces variables. Les vecteurs booléens peuvent être analysés pour les modèles d'achat qui reflètent des articles qui sont fréquemment associés ou achetés ensemble. Ces motifs peuvent être représentés sous la forme de règles d'association. Par exemple, l'information selon laquelle les clients qui achètent des ordinateurs ont également tendance à acheter des logiciels antivirus en même temps est représentée dans la règle d'association suivante :

Ordinateur ➔logiciel antivirus [support = 2 %, confiance = 60 %]

Le support et la confiance des règles sont deux mesures de l'intérêt des règles. Ils reflètent respectivement l'utilité et la certitude des règles découvertes. Un support de 2% signifie que 2% de toutes les transactions analysées montrent que les logiciels informatiques et antivirus sont achetés ensemble. Une confiance de 60 % signifie que 60 % des clients qui ont acheté un ordinateur ont également acheté le logiciel. Typiquement, les règles d'association sont considérées comme intéressantes si elles satisfont à la fois un seuil de support minimum et un seuil de confiance minimum. Ces seuils peuvent être définis par des utilisateurs ou des experts du domaine. Une analyse supplémentaire peut être effectuée pour découvrir des corrélations statistiques intéressantes entre les éléments associés.

3. Ensembles d'éléments fréquents, ensembles d'éléments fermés et règles d'association

Soit $\mathrm{I} = \{I_1, I_2, \ldots, I_m\}$ un itemset. Soit D, les données pertinentes pour la tâche, un ensemble de transactions de base de données où chaque transaction T est un ensemble d'éléments non vide tel que T ⊂ I. Chaque transaction est associée à un identifiant, appelé TID. Soit A un ensemble d'items. Une transaction T est dite contenir A si A ⊆ T. Une règle d'association est une implication de la forme A ⇒ B, où A ⊂ I, B ⊂ I, A ≠∅ ;, B ≠∅;, et A ∩ B =∅ . La règle A ⇒ B est valable dans l'ensemble de transactions D avec le support s, où s est le pourcentage de transactions dans D qui contiennent A ∪ B. Il s'agit de la probabilité, P(A ∪ B). La règle A ⇒ B a une confiance c dans l'ensemble de transactions D, où c est le pourcentage de transactions dans D contenant A qui contiennent également B. Il s'agit de la probabilité conditionnelle, P(B/A). C'est-à-dire :

$$support(A \Longrightarrow B) = P(A \cup B)$$

$$Confiance(A \Longrightarrow B) = P(B/A)$$

Les règles qui satisfont à la fois un seuil de support minimum et un seuil de confiance minimum sont dites fortes. Par convention, nous écrivons les valeurs de support et de confiance de manière à se situer entre 0% et 100%, plutôt qu'entre 0 et 1.

Un ensemble d'éléments est appelé itemset. Un ensemble d'éléments contenant k éléments est un k-itemset. La fréquence d'occurrence d'un itemset est le nombre de transactions qui contiennent l'itemset. Ceci est également connu comme la fréquence, le nombre de supports ou le nombre de l'ensemble d'éléments. Il est à noter que le support d'itemset est parfois appelé support relatif, tandis que la fréquence d'occurrence est appelée support absolu. Si le support relatif d'un itemset I satisfait un seuil de support minimum prédéfini, alors I est un itemset fréquent. L'ensemble des k-itemsets fréquents est communément noté par L_k tel que :

$$Confiance(A \Longrightarrow B) == P(B/A) = \frac{support\ (A \cup B)}{support\ (A)}$$

Cette éuation montre que la confiance de la règle A $\Rightarrow$ B peut être facilement déduite du comptage du support de A et A $\cup$ B. Autrement dit, une fois que le comptage du support de A, B et A $\cup$ B sont trouvés, il est simple de dériver les règles d'association correspondantes A $\Rightarrow$ B et B $\Rightarrow$ A et de vérifier si elles sont fortes. Ainsi, le problème de minage des règles d'association peut être réduit à celui de minage d'itemsets fréquents.

En général, l'extraction de règles d'association peut être considérée comme un processus en deux étapes :

1. Trouver tous les ensembles itemsets fréquents : par définition, chacun de ces ensembles d'éléments se produira au moins aussi fréquemment qu'un nombre de support minimum prédéterminé.

2. Générer des règles d'association fortes à partir des itemsets fréquents : Par définition, ces règles doivent satisfaire un minimum de support et un minimum de confiance.

Des mesures d'intérêt supplémentaires peuvent être appliquées pour la découverte de relations de corrélation entre des éléments associés. Étant donné que la deuxième étape est beaucoup moins coûteuse que la première, la performance globale des règles d'association minière est déterminée par la première étape.

Un défi majeur dans l'extraction d'itemsets fréquents à partir d'un grand ensemble de données est le fait qu'une telle extraction génère souvent un grand nombre d'itemsets qui satisfait le seuil de support minimum, en particulier lorsqu'il est réglé bas. En effet, si un itemset est fréquent, chacun de ses sous-ensembles est également fréquent. Un ensemble d'éléments long contiendra un nombre combinatoire de sous-ensembles d'éléments plus courts et fréquents.

Un itemset X est fermé dans un ensemble de données D s'il n'existe pas de super-itemset approprié Y tel que Y ait le même nombre de supports que X dans D. Un itemset X est un itemset fermé fréquent dans l'ensemble D si X est à la fois fermé et fréquent dans D. Un itemset X est un itemset fréquent maximal (ou max-itemset) dans un ensemble de données D si X est fréquent, et il n'existe pas de super-itemset Y tel que X $\subset$ Y et Y est fréquent dans D.

Soit C l'ensemble des itemsets fréquents fermés pour un ensemble de données D satisfaisant un seuil de support minimum. Soit M l'ensemble des itemsets fréquents maximaux pour D satisfaisant le support minimum. Supposons que nous ayons le nombre de supports de chaque ensemble d'éléments dans C et M. L'ensemble C et ses informations de comptage peuvent être utilisées pour dériver l'ensemble des ensembles d'éléments fréquents.

Ainsi, nous disons que C contient des informations complètes concernant ses itemsets fréquents correspondants. D'autre part, M enregistre uniquement le support des itemsets maximaux. Il ne contient généralement pas les informations de support complètes concernant ses itemsets fréquents correspondants.

4. Méthodes d'exploration des itemsets fréquents

4.1. Recherche d'itemsets fréquents par génération de candidats confinés (l'Algorithme Apriori)

L'algorithme Apriori a été proposé par R. Agrawal et R. Srikant en 1994 pour extraire des itemsets fréquents pour les règles d'association booléennes. Le nom de l'algorithme est basé sur le fait que l'algorithme utilise une connaissance préalable des propriétés fréquentes des itemsets. Apriori utilise une approche itérative connue sous le nom de recherche par niveau, où k-itemsets sont utilisés pour explorer (k +1)-itemsets. Tout d'abord, l'ensemble des fréquents 1-itemsets est trouvé en parcourant la base de données pour accumuler le nombre de chaque élément et en collectant les éléments qui satisfont le support minimum. L'ensemble résultant est noté L1. Ensuite, L1 est utilisé pour trouver L2, l'ensemble des 2-itemsets fréquents, qui est utilisé pour trouver L3, et ainsi de suite, jusqu'à ce qu'il n'y ait plus de k-itemsets fréquents. La découverte de chaque Lk nécessite une analyse complète de la base de données.

Pour améliorer l'efficacité de la génération par niveau d'itemsets fréquents, une propriété importante appelée la propriété Apriori est utilisée pour réduire l'espace de recherche.

Propriété a priori : tous les sous-ensembles non vides d'un ensemble d'éléments fréquents doivent également être fréquents. La propriété Apriori est basée sur l'observation que si un itemset I ne satisfait pas le seuil de support minimum, alors I n'est pas fréquent. Si un élément A

est ajouté à l'itemset I, alors l'itemset résultant (c'est-à-dire I ∪ A) ne peut pas apparaître plus fréquemment que I. Par conséquent, I ∪ A n'est pas fréquent non plus.

Cette propriété appartient à une catégorie spéciale de propriétés appelée antimonotonicité en ce sens que si un ensemble ne peut pas réussir un test, tous ses sur-ensembles échoueront également au même test. On l'appelle antimonotonicité car la propriété est monotone dans le contexte de l'échec d'un test. La propriété Apriori est utilisée dans l'algorithme en se basant sur la facon d'utilisé L_{k-1} pour trouverL_k ,pour k ≥ 2. Pour cela, un processus à deux étapes est utilisé consistant en des actions de jointure et d'élagage.

- **L'étape de jointure :** Pour trouver L_k, un ensemble de k-itemsets candidats est généré en joignant L_{k-1} à lui-même. Cet ensemble de candidats est noté C_k. Soit I_1 et I_2 des itemsets dans L_{k-1}. La notation $I_i[j]$ fait référence au jème élément de I_i. Pour une implémentation efficace, Apriori suppose que les éléments d'une transaction ou d'un ensemble d'éléments sont triés dans l'ordre lexicographique. La jointure, $L_{k-1} \bowtie L_{k-1}$, est effectuée, où les membres de L_{k-1} sont joignables si leurs premiers éléments (k-2) sont en commun. C'est-à-dire que les membresI_1 et I_2 de L_{k-1} sont joints si $(I_1[1] = I_2[1]) \wedge (I_1[2] = I_2[2]) \wedge \ldots \wedge (I_1[k-2] = I_2[k-2]) \wedge (I_1[k-1] < I_2[k-1])$. La condition $I_1[k-1] < I_2[k-1]$ garantit simplement qu'aucun doublon n'est généré. L'itemset résultant formé en joignant I_1 et I_2 est $\{I_1[1], I_1[2], \ldots, I_1[k-2], I_1[k-1], I_2[k-1]\}$
- **L'étape d'élagage :** C_k est un sur-ensemble de L_k, c'est-à-dire que ses membres peuvent être fréquents ou non, mais tous les k-itemsets fréquents sont inclus dans C_k. Une analyse de la base de données pour déterminer le nombre de chaque candidat dans C_k entraînerait la détermination de L_k, (c'est-à-dire que tous les candidats ayant un nombre non inférieur au nombre de support minimum sont fréquents par définition et appartiennent donc à L_k,). C_k peut être énorme, et cela pourrait donc impliquer des calculs lourds. Pour réduire la taille de C_k, la propriété Apriori est utilisée comme suit. Tout (k-1)-itemset qui n'est pas fréquent ne peut pas être un sous-ensemble d'un k-itemset fréquent. Par conséquent, si un (k -1)-sous-ensemble d'un k-itemset candidat

n'est pas dans L_{k-1}, alors le candidat ne peut pas non plus être fréquent et peut donc être retiré de C_k. Ce test de sous-ensemble peut être effectué rapidement en maintenant un arbre de hachage de tous les ensembles d'éléments fréquents.

4.1.1. Génération de règles d'association à partir d'itemsets fréquents

Une fois que les itemsets fréquents des transactions dans une base de données D ont été trouvés, il est simple de générer des règles d'association fortes à partir d'eux (où les règles d'association fortes satisfont à la fois au support minimum et à la confiance minimum). Cela peut être fait en utilisant l'équation suivante complet :

$$Confiance(A \Longrightarrow B) == P(B/A) = \frac{Compteur\_support\ (A \cup B)}{Compteur\_support\ (A)}$$

La probabilité conditionnelle est exprimée en termes de nombre d'éléments pris en charge, où $Compteur\_support\ (A \cup B)$ est le nombre de transactions contenant les itemsets A ∪ B, et $Compteur\_support\ (A)$ est le nombre de transactions contenant l'itemset A. Basé sur cette équation, les règles d'association peuvent être générées comme suit :

- Pour chaque itemset fréquent l, générer tous les sous-ensembles non vides de l.
- Pour chaque sous-ensemble non vide s de l, afficher la règle $s \Longrightarrow (l - s)$ si $\frac{Compteur\_support\ (l)}{Compteur\_support\ (s)} \geq \min\ \_conf$, où $\min\ \_conf$ est le seuil de confiance minimum.

Étant donné que les règles sont générées à partir itemsets fréquents, chacune satisfait automatiquement la prise en charge minimale. Les itemsets fréquents peuvent être stockés à l'avance dans des tables de hachage avec leur nombre afin qu'ils puissent être consultés rapidement.

4.1.2. Les variantes de l'algorithme Apriori

De nombreuses variantes de l'algorithme Apriori ont été proposées pour l'amélioration de l'efficacité de l'algorithme d'origine.

A. Technique basée sur le hachage

Une technique basée sur le hachage peut être utilisée pour réduire la taille des k-ensembles d'éléments candidats, C_k, pour k > 1. Par exemple, lors de l'analyse de chaque transaction dans la base de données pour générer les 1-itemsets fréquents, L1, nous pouvons générer tous les 2-itemsets pour chaque transaction, les hacher dans les différents compartiments d'une structure de table de hachage, et augmentez le nombre de seaux correspondants. Un ensemble de 2 éléments avec un nombre de compartiments correspondant dans la table de hachage qui est inférieur au seuil de prise en charge ne peut pas être fréquent et doit donc être supprimé de l'ensemble candidat. Une telle technique basée sur le hachage peut réduire considérablement le nombre de k-itemsets candidats examinés.

B. Réduction des transactions

Une transaction qui ne contient pas de k-itemsets fréquents ne peut pas contenir de (k+1)-itemsets fréquents. Par conséquent, une telle transaction peut être marquée ou retirée de toute considération ultérieure car les analyses de base de données ultérieures pour les j-itemsets, où j > k, n'auront pas besoin de prendre en compte une telle transaction.

C. Partitionnement

Une technique de partitionnement peut être utilisée en faisant uniquement deux analyses de base de données pour extraire les itemsets fréquents. Dans la phase I, l'algorithme divise les transactions de D en n partitions non superposées. Si le seuil de prise en charge minimum pour les transactions dans D est $\min\_sup$, alors le nombre de prise en charge minimum pour une partition est $\min\_\sup \times$ *le nombre de transactions dans cette partition*. Pour chaque partition, tous les itemsets fréquents locaux sont trouvés. Un itemset local fréquent peut ou non être fréquent par rapport à l'ensemble de la base de données D. Cependant, tout itemset potentiellement fréquent par rapport à D doit apparaître comme un itemset fréquent dans au moins une des partitions. Les itemsets fréquents sont des itemsets candidats par rapport à D. La collection d'itemets fréquents de toutes les partitions forme les itemsets candidats globaux par rapport à D. Dans la phase II, une deuxième analyse de D est effectuée dans laquelle le soutien réel de chaque candidat est évalué pour déterminer les ensembles d'items fréquents globaux. La

taille et le nombre de partitions sont définis de manière à ce que chaque partition puisse tenir dans la mémoire principale et donc être lue une seule fois dans chaque phase.

D. Échantillonnage

L'idée de base de l'approche d'échantillonnage est de choisir un échantillon aléatoire S de la base D, puis de rechercher des itemsets fréquents dans S au lieu de D. La taille de l'échantillon S est telle que la recherche d'itemsets fréquents dans S peut être effectuée dans la mémoire principale, et donc un seul balayage des transactions dans S est requis globalement. Parce que nous recherchons des itemsets fréquents dans S plutôt que dans D, il est possible que certains des itemsets fréquents globaux sont manqués. Pour réduire cette possibilité, un seuil de support inférieur au support minimum est utilisé pour trouver les itemsets fréquents locaux à S (notés L^S). Le reste de la base de données est ensuite utilisé pour calculer les fréquences réelles de chaque itemset dans L^S. Un mécanisme est utilisé pour déterminer si tous les itemsets fréquents globaux sont inclus dans L^S. Si L^S contient réellement tous les ensembles d'éléments fréquents dans D, alors un seul balayage de D est requis. Sinon, une deuxième passe peut être effectuée pour trouver les itemsets fréquents qui ont été manqués lors de la première passe. L'approche d'échantillonnage est particulièrement bénéfique lorsque l'efficacité est de la plus haute importance, comme dans les applications à forte intensité de calcul qui doivent être exécutées fréquemment.

E. Comptage dynamique d'itemsets

une technique de comptage dynamique d'itemsets a été proposée dans laquelle la base de données est partitionnée en blocs marqués par des points de départ. Dans cette variante, de nouveaux jeux d'éléments candidats peuvent être ajoutés à n'importe quel point de départ, contrairement à Apriori, qui détermine les nouveaux jeux d'éléments candidats juste avant chaque analyse complète de la base de données. La technique utilise le décompte jusqu'à présent comme limite inférieure du décompte réel. Si le décompte dépasse la prise en charge minimale, l'ensemble d'éléments est ajouté à la collection d'éléments fréquents et peut être utilisé pour générer des candidats plus longs. Cela conduit à moins d'analyses de base de données qu'avec Apriori pour trouver tous les ensembles d'éléments fréquents.

4.2. Exploration des itemsets fréquents par Croissance de modèle

La méthode de génération et de test des candidats Apriori réduit considérablement la taille des ensembles candidats, ce qui entraîne un bon gain de performances. Cependant, il peut souffrir de deux coûts non négligeables :

- Il peut encore avoir besoin de générer un grand nombre d'ensembles candidats.
- Il peut être nécessaire d'analyser à plusieurs reprises l'ensemble de la base de données et de vérifier un grand nombre de candidats par correspondance de modèles. Il est coûteux de passer en revue chaque transaction dans la base de données pour déterminer la prise en charge des itemsets candidats.

Une méthode intéressante dans cette tentative est appelée croissance de modèle fréquent, ou simplement croissance FP, qui adopte une stratégie de division pour régner comme suit. Tout d'abord, il comprime la base de données représentant les éléments fréquents dans un arbre de modèles fréquents, ou arbre FP, qui conserve les informations d'association d'éléments. Il divise ensuite la base de données compressée en un ensemble de bases de données conditionnelles chacune associée à un élément fréquent ou « fragment de modèle », et exploite chaque base de données séparément. Pour chaque « fragment de modèle », seuls ses ensembles de données associés doivent être examinés. Par conséquent, cette approche peut réduire considérablement la taille des ensembles de données à rechercher, ainsi que la "croissance" des modèles examinés.

La méthode de croissance FP transforme le problème de la recherche de modèles fréquents longs en recherche de modèles plus courts dans des bases de données conditionnelles beaucoup plus petites de manière récursive, puis en concaténant le suffixe. Il utilise les items les moins fréquents comme suffixe, offrant une bonne sélectivité. Le procédé réduit considérablement les coûts de recherche.

Lorsque la base de données est volumineuse, il est parfois irréaliste de construire un arbre FP basé sur la mémoire principale. Une alternative intéressante consiste à partitionner d'abord la base de données en un ensemble de bases de données projetées, puis à construire un arbre FP et à l'exploiter dans chaque base de données projetée. Ce processus peut être appliqué de

manière récursive à n'importe quelle base de données projetée si son arbre FP ne peut toujours pas tenir dans la mémoire principale.

Une étude des performances de la méthode de croissance FP montre qu'elle est efficace et évolutive pour extraire des modèles fréquents longs et courts, et qu'elle est d'environ un ordre de grandeur plus rapide que l'algorithme Apriori.

4.3. Extraction d'itemsets fréquents à l'aide du format de données vertical

Les méthodes Apriori et croissance FP exploitent des modèles fréquents à partir d'un ensemble de transactions au format TID-itemset, où TID est un ID de transaction et itemset est l'ensemble des articles achetés dans la transaction TID. C'est ce qu'on appelle le format de données horizontal. Alternativement, les données peuvent être présentées au format item-TID-set où item est le nom d'article, et TID-set est l'ensemble d'identificateurs de transaction contenant l'article. C'est ce qu'on appelle le format de données vertical.

Le processus d'extraction d'itemsets fréquents en explorant le format de données vertical peut être effectuer comme suit : Tout d'abord, nous transformons les données formatées horizontalement en format vertical en scannant l'ensemble de données une fois. Le nombre de support d'un ensemble d'éléments est simplement la longueur de l'ensemble TID de l'ensemble d'éléments. À partir de k =1, les k-itemsets fréquents peuvent être utilisés pour construire les (k+1)-itemsets candidats basés sur la propriété Apriori.

Le calcul est fait par intersection des ensembles TID des k-itemsets fréquents pour calculer les ensembles TID des (k+1)-itemsets correspondants. Ce processus se répète, avec k incrémenté de 1 à chaque fois, jusqu'à ce qu'aucun itemsets fréquents ou candidats ne puisse être trouvé.

En plus de tirer parti de la propriété Apriori dans la génération de (k+1)-itemset candidats à partir de k-itemsets fréquents, un autre mérite de cette méthode est qu'il n'est pas nécessaire de parcourir la base de données pour trouver le support de (k+1)-itemsets pour k≥ 1.

Cela est dû au fait que l'ensemble TID de chaque k-itemset contient les informations complètes requises pour compter un tel support. Cependant, les ensembles TID peuvent être assez longs,

prenant un espace mémoire substantiel ainsi qu'un temps de calcul pour croiser les ensembles longs.

Pour réduire davantage le coût d'enregistrement de longs ensembles TID, ainsi que les coûts ultérieurs des intersections, on peut utiliser une technique appelée diffset, qui ne garde trace que des différences entre les ensembles TID d'un (k+1)-itemset et un correspondant k-itemset. Les expériences montrent que dans certaines situations, comme lorsque l'ensemble de données contient de nombreux modèles denses et longs, cette technique peut réduire considérablement le coût total de l'extraction de format vertical d'ensembles d'éléments fréquents.

4.4. Exploitation des modèles fermés et maximum

L'exploitation d'itemsets fréquents peut générer un grand nombre d'itemsets fréquents, en particulier lorsque le seuil min_sup est bas ou lorsqu'il existe de longs modèles dans l'ensemble de données. Les itemsets fréquents fermés peuvent réduire considérablement le nombre de motifs générés dans l'exploration d'itemsets fréquents tout en préservant l'information complète concernant l'ensemble d'itemsets fréquents. Autrement dit, à partir de l'ensemble d'itemsets fréquents fermés, nous pouvons facilement dériver l'ensemble d'itemsets fréquents et leur support. Ainsi, en pratique, il est plus souhaitable d'exploiter l'ensemble des itemsets fréquents fermés plutôt que l'ensemble de tous les itemsets fréquents dans la plupart des cas.

L'exploration itemsets fréquents fermés peut se faire par une approche naïve en effectuant une extraction de l'ensemble complet d'itemsets fréquents, puis de supprimer chaque itemset fréquent qui est un sous-ensemble approprié d'un itemset fréquent existant et qui porte le même support. Cependant, cela est assez coûteux.

Une méthodologie recommandée consiste à rechercher des itemsets fréquents fermés directement pendant le processus d'exploration. Cela nous oblige à élaguer l'espace de recherche dès que nous pouvons identifier le cas d'itemsets fermés lors de l'exploration. Les stratégies d'élagage comprennent :

a) **Fusion d'éléments :** si chaque transaction contenant un ensemble d'éléments fréquents X contient également un ensemble d'éléments Y mais pas de surensemble approprié de

Y, alors X ∪ Y forme un ensemble d'éléments fermé fréquent et il n'est pas nécessaire de rechercher un ensemble d'éléments contenant X mais pas de Y.

b) **Élagage des sous-ensembles d'éléments :** si un ensemble d'éléments fréquents X est un sous-ensemble approprié d'un ensemble d'éléments fermés fréquents déjà trouvé Y et nombre de support de X est égale au nombre de support de Y, alors X et tous les descendants de X dans l'arbre d'énumération de l'ensemble ne peuvent pas être itemsets fermés fréquemment et peuvent donc être élagués.

c) **Saut d'items :** lors de l'exploration en profondeur d'itemsets fermés, à chaque niveau, il y aura un itemset de préfixe X associé à une table d'en-tête et à une base de données projetée. Si un élément fréquent local p a le même support dans plusieurs tables d'en-tête à différents niveaux, nous pouvons en toute sécurité élaguer p des tables d'en-tête à des niveaux supérieurs.

Outre l'élagage de l'espace de recherche dans le processus d'extraction d'éléments fermés, une autre optimisation importante consiste à effectuer une vérification efficace de chaque ensemble d'éléments fréquents nouvellement dérivé pour voir s'il est fermé. En effet, le processus d'exploration de données ne peut pas garantir que chaque ensemble d'éléments fréquents généré est fermé.

Lorsqu'un nouvel ensemble d'éléments fréquents est dérivé, il est nécessaire d'effectuer deux types de vérification de fermeture :

- **Vérification du sur-ensemble** : vérifie si ce nouvel ensemble d'éléments fréquents est un sur-ensemble de certains ensembles d'éléments fermés déjà trouvés avec le même support
- **Vérification du sous-ensemble :** qui vérifie si l'ensemble d'éléments nouvellement trouvé est un sous-ensemble d'un ensemble d'éléments fermé déjà trouvé avec le même support.

Si on adopte la méthode d'élagage par fusion d'éléments dans un cadre de division pour mieux régner, la vérification du sur-ensemble est en fait intégrée et il n'est pas nécessaire d'effectuer explicitement la vérification du sur-ensemble. En effet, si un ensemble d'éléments fréquent

$X \cup Y$ est trouvé plus tard que l'ensemble d'éléments X et porte le même support que X, il doit se trouver dans la base de données projetée de X et doit avoir été généré lors de la fusion de l'ensemble d'éléments.

Pour aider à la vérification des sous-ensembles, un arbre de modèles compressé peut être construit pour maintenir l'ensemble des ensembles d'éléments fermés exploités jusqu'à présent. Le pattern-tree est similaire dans sa structure au FP-tree sauf que tous les itemsets fermés trouvés sont stockés explicitement dans les branches correspondantes de l'arbre. Pour une vérification efficace des sous-ensembles, nous pouvons utiliser la propriété suivante : si l'itemset courant S_c peut être subsumé par un autre itemset fermé déjà trouvé S_a, alors

- S_c et S_a ont le même support
- La longueur de S_c est inférieure à celui de S_a
- Tous les éléments de S_c sont contenus dans S_a.

Sur la base de cette propriété, une structure d'index de hachage à deux niveaux peut être construite pour un accès rapide à l'arbre de modèle : le premier niveau utilise l'identifiant du dernier élément dans S_c comme clé de, et le deuxième niveau utilise le support de S_c comme clé de hachage. Cela accélérera considérablement le processus de vérification des sous-ensembles.

De nombreuses techniques d'optimisation peuvent être étendues à l'extraction d'itemets fréquents maximaux puisque les itemsets fréquents maximaux partagent de nombreuses similitudes avec les itemsets fréquents fermés.

5. Méthodes d'évaluation des Modèles

La plupart des algorithmes d'extraction de règles d'association utilisent un cadre de soutien-confiance.

Bien que les seuils de support et de confiance minimaux aident à éliminer ou à exclure l'exploration d'un bon nombre de règles inintéressantes, de nombreuses règles générées ne sont toujours pas intéressantes pour les utilisateurs. Malheureusement, cela est particulièrement vrai

lors de l'extraction à des seuils de support bas ou de l'extraction de modèles longs. Cela a été une brèche d'étranglement majeur pour l'application réussie de l'extraction de règles d'association.

5.1. Apport des règles strictes

Les règles strictes ne sont pas nécessairement intéressantes. L'intérêt ou non d'une règle peut être évalué de manière subjective ou objective. En définitive, seul l'utilisateur peut juger si une règle donnée est intéressante, et ce jugement, étant subjectif, peut différer d'un utilisateur à l'autre. Cependant, des mesures objectives de l'intérêt, basées sur les statistiques, peuvent être utilisées comme une étape vers l'objectif d'éliminer les règles inintéressantes qui seraient autrement présentées à l'utilisateur.

5.2. De l'analyse d'association à l'analyse de corrélation

Comme nous l'avons vu jusqu'à présent, les mesures de soutien et de confiance sont insuffisantes pour filtrer les règles d'association inintéressantes. Pour remédier à cette faiblesse, une mesure de corrélation peut être utilisée pour augmenter le cadre de soutien-confiance pour les règles d'association. Cela conduit à des règles de corrélation de la forme

$$A \Rightarrow B[\text{support}, \text{confiance}, \text{corrélation}]$$

Autrement dit, une règle de corrélation est mesurée non seulement par son support et sa confiance, mais également par la corrélation entre les ensembles d'éléments A et B. Il existe de nombreuses mesures de corrélation différentes parmi lesquelles choisir :

- **LIFT** : c'est une mesure de corrélation simple qui est donnée comme suit. L'occurrence de l'itemset A est indépendant de l'occurrence de l'itemset B si $P(A \cup B) = P(A) \times P(B)$; sinon, les ensembles d'éléments A et B sont dépendants et corrélés en tant qu'événements. Cette définition peut facilement être étendue à plus de deux itemsets. L'ascenseur entre l'occurrence de A et B peut être mesuré en calculant

$$LIFT\ (A, B) = \frac{P(A \cup B)}{P(A).P(B)}$$

  Si la valeur résultante de l'équation est inférieure à 1, alors l'occurrence de A est négativement corrélée à l'occurrence de B, ce qui signifie que l'occurrence de l'un conduit

probablement à l'absence de l'autre. Si la valeur résultante est supérieure à 1, alors A et B sont positivement corrélés, ce qui signifie que l'occurrence de l'un implique l'occurrence de l'autre. Si la valeur résultante est égale à 1, alors A et B sont indépendants et il n'y a pas de corrélation entre eux. L'équation de LIFT peut être écrite d'une autre manière :

$$LIFT\ (A,B) = \frac{P(B/A)}{P(B)} = \frac{conf\,(A \Rightarrow B)}{\sup\,(B)}$$

En d'autres termes, il évalue dans quelle mesure l'occurrence de l'un « soulève » l'occurrence de l'autre.

Exemple :

Si A correspond à la vente de jeux informatiques et B correspond à la vente de vidéos, alors compte tenu des conditions actuelles du marché, on dit que la vente de jeux augmente ou « élève » la probabilité de vente de vidéos d'un facteur de la valeur retournée par LIFT.

- **X^2 :** Pour calculer la valeur X^2, nous prenons la différence au carré entre la valeur observée et la valeur attendue pour un créneau (paire A et B) dans le tableau de contingence, divisée par la valeur attendue. Ce montant est additionné pour tous les créneaux du tableau de contingence.

Chapitre IV: Classification

1 Introduction

La classification est une forme d'analyse de données qui extrait des modèles décrivant des classes de données importantes. De tels modèles, appelés classificateurs, prédisent des étiquettes de classe catégorielles (discrètes, non ordonnées).

Exemple :

On peut créer un modèle de classification pour catégoriser les demandes de prêt bancaire comme sûres ou risquées. Une telle analyse peut nous aider à mieux comprendre les données dans leur ensemble.

De nombreuses méthodes de classification ont été proposées par des chercheurs en apprentissage automatique, en reconnaissance de formes et en statistiques. La plupart des algorithmes résident en mémoire, supposant généralement une petite taille de données. La recherche récente sur l'exploration de données s'est appuyée sur ces travaux, développant des techniques de classification et de prédiction évolutives capables de gérer de grandes quantités de données résidant sur le disque. La classification a de nombreuses applications, notamment la détection des fraudes, le marketing ciblé, la prédiction des performances, la fabrication et le diagnostic médical.

2. Fonctionnement

La classification des données est un processus en deux étapes, composé d'une étape d'apprentissage où un modèle de classification est construit et d'une étape de classification où le modèle est utilisé pour prédire les étiquettes de la classe qui correspond à une donnée.

Dans la première étape, un classificateur est construit décrivant un ensemble prédéterminé de classes ou de concepts de données. Il s'agit de l'étape d'apprentissage, où un algorithme de classification construit le classifieur en apprenant à partir d'un ensemble d'apprentissage composé de tuples de base de données et de leurs étiquettes de classe associées. Un tuple X est

représenté par un vecteur d'attributs à n dimensions, $X = (x_1, x_2, \dots, x_n)$ représentant n mesures effectuées sur le tuple à partir de n attributs de base de données. Chaque tuple X est supposé appartenir à une classe prédéfinie telle que déterminée par un autre attribut de la base de données appelé l'attribut d'étiquette de classe. L'attribut d'étiquette de classe est à valeur discrète et non ordonné. Il est catégorique en ce que chaque valeur sert de catégorie ou de classe. Les tuples individuels constituant l'ensemble d'apprentissage sont appelés tuples d'apprentissage et sont échantillonnés de manière aléatoire à partir de la base de données en cours d'analyse. Dans le contexte de la classification, les tuples de données peuvent être appelés échantillons, exemples, instances, points de données ou objets. Étant donné que l'étiquette de classe de chaque tuple d'apprentissage est fournie, cette étape est également connue sous le nom d'apprentissage supervisé. Cela contraste avec l'apprentissage non supervisé (ou clustering), dans lequel l'étiquette de classe de chaque tuple d'apprentissage n'est pas connue, et le nombre ou l'ensemble de classes à apprendre peut ne pas être connu à l'avance.

Cette première étape du processus de classification peut également être considérée comme l'apprentissage d'un mappage ou d'une fonction $y = f(x)$ qui peut prédire l'étiquette de classe associée y d'un tuple X donné.

Dans cette vue, on souhaite apprendre un mappage ou une fonction qui sépare les classes de données. Généralement, cette cartographie est représentée sous la forme de règles de classification, d'arbres de décision ou de formules mathématiques. Les règles peuvent être utilisées pour catégoriser les futurs tuples de données, ainsi que pour fournir un aperçu plus approfondi du contenu des données. Ils fournissent également une représentation des données compressées.

Dans la deuxième étape, le modèle est utilisé pour la classification. Tout d'abord, la précision prédictive du classifieur est estimée. Si on devait utiliser l'ensemble d'apprentissage pour mesurer la précision du classificateur, cette estimation serait probablement optimiste, car le classificateur a tendance à surajuster les données. C'est-à-dire que pendant l'apprentissage, il peut incorporer certaines anomalies particulières des données d'apprentissage qui ne sont pas présentes dans l'ensemble de données générales. Par conséquent, un jeu de test est utilisé,

composé de tuples de test et de leurs étiquettes de classe associées. Ils sont indépendants des tuples d'apprentissage, ce qui signifie qu'ils n'ont pas été utilisés pour construire le classifieur.

La précision d'un classificateur sur un jeu de test donné est le pourcentage de tuples du jeu de test qui sont correctement classés par le classificateur. L'étiquette de la classe associée pour chaque tuple de test est comparée à la prédiction de la classe du classificateur appris pour ce tuple. Si la précision du classificateur est considérée comme acceptable, le classificateur peut être utilisé pour classer les futurs tuples de données pour lesquels l'étiquette de classe n'est pas connue.

3. Les arbres de décision

Un arbre de décision est une structure arborescente de type organigramme, où chaque nœud interne désigne un test sur un attribut, chaque branche représente un résultat du test et chaque nœud feuille contient une étiquette de classe. Le nœud le plus haut dans un arbre est le nœud racine. Un arbre de décision typique est illustré à la Figure 1. Certains algorithmes d'arbre de décision ne produisent que des arbres binaires, tandis que d'autres peuvent produire des arbres non binaires.

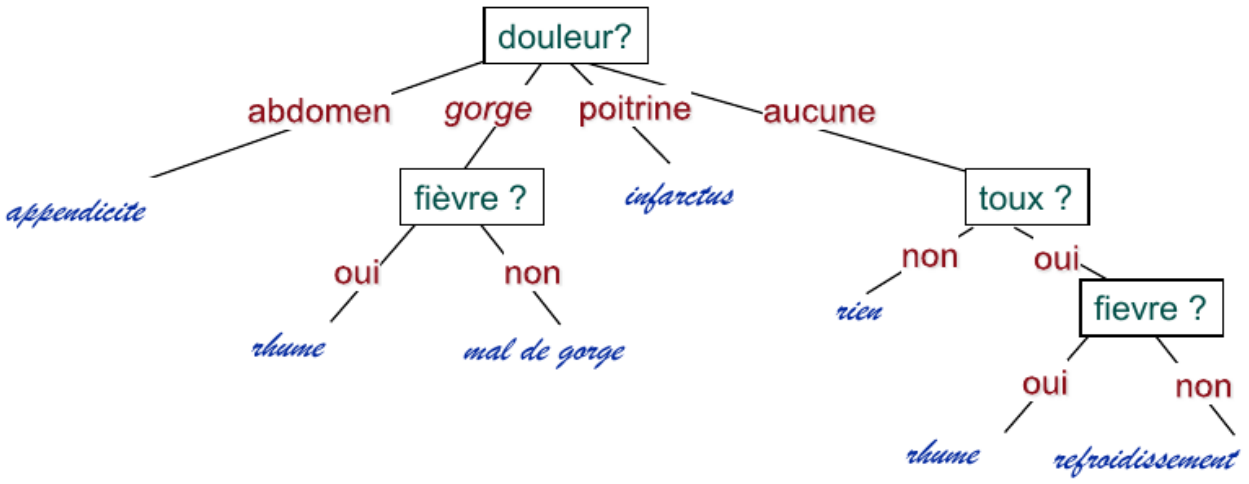

Figure 4.1 Arbre de décision pour le diagnostic d'une douleur[23]

[23] http://cedric.cnam.fr/vertigo/cours/ml2/coursArbresDecision.html

Étant donné un tuple X pour lequel l'étiquette de classe associée est inconnue, les valeurs d'attribut du tuple sont testées par rapport à l'arbre de décision. Un chemin est tracé de la racine à un nœud feuille, qui contient la prédiction de la classe pour ce tuple. Les arbres de décision peuvent facilement être convertis en règles de classification.

La construction de classificateurs d'arbres de décision ne nécessite aucune connaissance du domaine ou réglage de paramètres, et est donc appropriée pour la découverte de connaissances exploratoires. Les arbres de décision peuvent gérer des données multidimensionnelles. Leur représentation des connaissances acquises sous forme d'arbre est intuitive et généralement facile à assimiler par l'homme. Les étapes d'apprentissage et de classification de l'induction de l'arbre de décision sont simples et rapides. En général, les classificateurs d'arbres de décision ont une bonne précision. Cependant, une utilisation réussie peut dépendre des données disponibles. Les algorithmes d'induction d'arbre de décision ont été utilisés pour la classification dans de nombreux domaines d'application tels que la médecine, la fabrication et la production, l'analyse financière, l'astronomie et la biologie moléculaire. Les arbres de décision sont à la base de plusieurs systèmes commerciaux d'induction de règles.

3.1. Induction

À la fin des années 1970 et au début des années 1980, J. Ross Quinlan, chercheur en apprentissage automatique, a développé un algorithme d'arbre de décision appelé ID3 (Iterative Dichotomiser). Ce travail a élargi les travaux antérieurs sur les systèmes d'apprentissage de concepts, décrits par E. B. Hunt, J. Marin et P. T. Stone. Quinlan a ensuite présenté C4.5 (un successeur d'ID3), qui est devenu une référence à laquelle les nouveaux algorithmes d'apprentissage supervisé sont souvent comparés. En 1984, un groupe de statisticiens L. Breiman et al. a publié le livre Classification and Regression Trees (CART), qui décrivait la génération d'arbres de décision binaires. ID3 et CART ont été inventés indépendamment l'un de l'autre à peu près au même moment, mais suivent une approche similaire pour apprendre les arbres de décision à partir de tuples d'entraînement. Ces deux algorithmes fondamentaux ont engendré une vague de travaux sur l'induction d'arbres de décision.

ID3, C4.5 et CART adoptent une approche sans retour en arrière dans laquelle les arbres de décision sont construits de manière descendante et récursive de type diviser pour régner. La plupart des algorithmes d'induction d'arbre de décision suivent également une approche descendante, qui commence par un ensemble d'apprentissage de tuples et leurs étiquettes de classe associées. L'ensemble d'apprentissage est partitionné de manière récursive en sous-ensembles plus petits au fur et à mesure de la construction de l'arbre.

3.2. Stratégie

L'algorithme est appelé avec trois paramètres :

- D : une partition de données,
- Liste d'attributs : une liste d'attributs de paramètre décrivant les tuples
- Méthode de sélection d'attribut : spécifie une procédure heuristique pour sélectionner l'attribut qui a la meilleure discrimination des tuples donnés en fonction de la classe.

Initialement, il s'agit de l'ensemble complet des tuples d'entraînement et de leurs étiquettes de classe associées. Cette procédure utilise une mesure de sélection d'attribut telle que le gain d'information ou l'indice de Gini. Le fait que l'arbre soit strictement binaire est généralement déterminé par la mesure de sélection d'attribut. Certaines mesures de sélection d'attributs, telles que l'indice de Gini, obligent l'arbre résultant à être binaire. D'autres, comme le gain d'informations, ne le font pas, ce qui permet des divisions multivoies, c'est-à-dire que deux branches ou plus se développent à partir d'un nœud.

L'arbre commence par un nœud unique N représentant les tuples d'apprentissage dans D. Si les tuples de D sont tous de la même classe, alors le nœud N devient une feuille et il est étiqueté avec cette classe. Sinon, l'algorithme appelle la méthode de sélection d'attribut pour déterminer le critère de fractionnement. Le critère de division nous indique quel attribut tester au nœud N en déterminant la « meilleure » façon de séparer ou de partitionner les tuples de D en classes individuelles. Le critère de division nous indique également quelles branches doivent se développer à partir du nœud N par rapport aux résultats du test choisi. Plus précisément, le critère de division indique l'attribut de division et peut également indiquer soit un point de division, soit un sous-ensemble de division. Le critère de découpage est déterminé pour que,

idéalement, les partitions résultantes à chaque branche soient les plus « pures » possibles. Une partition est pure si tous les tuples qu'elle contient appartiennent à la même classe. En d'autres termes, si on découpe les tuples dans D selon les résultats mutuellement exclusifs du critère de découpage, on espère que les partitions résultantes seront aussi pures que possible.

Le nœud N est étiqueté avec le critère de découpage, qui sert de test au nœud. Une branche est développée à partir du nœud N pour chacun des résultats du critère de division. Les tuples dans D sont partitionnés en conséquence. Soit A l'attribut de découpage. A à v valeurs distinctes, $\{a_1, a_2, \dots, a_v\}$, basées sur les données d'apprentissage. Il existe trois scénarios possibles :

- A est à valeurs discrètes : dans ce cas, les résultats du test au nœud N correspondent directement aux valeurs connues de A. Une branche est créée pour chaque valeur connue a_j de A et étiquetée avec cette valeur. La partition D_j est le sous-ensemble de tuples étiquetés par classe dans D ayant la valeur a_j de A. Étant donné que tous les tuples d'une partition donnée ont la même valeur pour A, A n'a pas besoin d'être pris en compte dans un futur partitionnement des tuples. Par conséquent, il est supprimé de la liste des attributs.
- A est à valeur continue : dans ce cas, le test au nœud N a deux résultats possibles correspondant aux conditions A ≤ point_de_partage et A > point_de_partage, respectivement, où point de partage est le point de partage renvoyé par la méthode de sélection d'attribut dans le cadre du critère de fractionnement. En pratique, le point de partage est souvent considéré comme le point médian de deux valeurs adjacentes connues de A et peut donc ne pas être une valeur préexistante de A à partir des données d'apprentissage. Deux branches sont développées à partir de N et étiquetées selon les résultats précédents. Les tuples sont partitionnés de telle sorte que D_1 contient le sous-ensemble de tuples étiquetés par classe dans D pour lequel A ≤ point_de_partage, tandis que D_2 contient le reste.
- A est à valeurs discrètes et un arbre binaire doit être produit : Le test au nœud N est de la forme $A \in S_A$, où S_A est le sous-ensemble de fractionnement pour A, renvoyé par la méthode de sélection d'attribut dans le cadre du critère de fractionnement. C'est un sous-

ensemble des valeurs connues de A. Si un tuple donné a la valeur a_j de A et si $a_j \in S_A$, alors le test au nœud N est satisfait. Deux branches sont issues de N. Par convention, la branche gauche de N est étiquetée oui afin que D_1 corresponde au sous-ensemble de tuples étiquetés par classe dans D qui satisfont au test. La branche droite de N est étiquetée no afin que D_2 corresponde au sous-ensemble de tuples étiquetés par classe de D qui ne satisfont pas le test.

L'algorithme utilise le même processus de manière récursive pour former un arbre de décision pour les tuples à chaque partition résultante D_j de D.

Le partitionnement récursif s'arrête uniquement lorsque l'une des conditions de fin suivantes est vraie :

- Tous les tuples de la partition D appartiennent à la même classe.
- Il n'y a plus d'attributs sur lesquels les tuples peuvent être encore partitionnés. Dans ce cas, le vote majoritaire est utilisé. Cela implique de convertir le nœud N en une feuille et de l'étiqueter avec la classe la plus courante dans D. La distribution de classe des tuples de nœud peut être stockée.
- Il n'y a pas de tuples pour une branche donnée, c'est-à-dire qu'une partition D_j est vide. Dans ce cas, une feuille est créée avec la classe majoritaire en D.

L'arbre de décision résultant est retourné. La complexité de calcul de l'algorithme étant donné l'ensemble d'apprentissage D est O(n x |D| x log(D)), où n est le nombre d'attributs décrivant les tuples dans D et |D| est le nombre de tuples d'apprentissage dans D.

Des versions incrémentales de l'induction d'arbre de décision ont également été proposées. Lorsqu'on leur donne de nouvelles données d'entraînement, celles-ci restructurent l'arbre de décision acquis à partir de l'apprentissage sur les données d'entraînement précédentes, plutôt que de réapprendre un nouvel arbre à partir de zéro.

Les différences dans les algorithmes d'arbre de décision incluent la manière dont les attributs sont sélectionnés lors de la création de l'arbre et les mécanismes utilisés pour l'élagage.

L'algorithme de base décrit précédemment nécessite un passage sur les tuples d'apprentissage dans D pour chaque niveau de l'arbre. Cela peut entraîner de longs temps de formation et un manque de mémoire disponible lorsqu'il s'agit de grandes bases de données.

3.3. Mesures de sélection d'attributs

Une mesure de sélection d'attribut est une heuristique permettant de sélectionner le critère de fractionnement qui sépare le mieux une partition de données donnée, D, de tuples d'apprentissage étiquetés par classe en classes individuelles. La division de D en partitions plus petites selon les résultats du critère de division génère idéalement des partitions pure (c'est-à-dire que tous les tuples qui tombent dans une partition donnée appartiendraient à la même classe). Conceptuellement, le meilleur critère de fractionnement est celui qui se rapproche le plus d'un tel scénario. Les mesures de sélection d'attributs sont également appelées règles de fractionnement car elles déterminent comment les tuples d'un nœud donné doivent être fractionnés.

La mesure de sélection d'attribut fournit un classement pour chaque attribut décrivant les tuples d'apprentissage donnés. L'attribut ayant le meilleur score pour la mesure est choisi comme attribut de séparation pour les tuples donnés. Si l'attribut de fractionnement est à valeur continue ou si nous sommes limités aux arbres binaires, alors, respectivement, un point de fractionnement ou un sous-ensemble de fractionnement doit également être déterminé dans le cadre du critère de fractionnement. Le nœud d'arbre créé pour la partition D est étiqueté avec le critère de fractionnement, les branches sont développées pour chaque résultat du critère et les tuples sont partitionnés en conséquence. Trois mesures de sélection d'attributs courantes seront abordées : le gain d'informations, le rapport de gain et l'indice de Gini.

Soit D un ensemble d'entraînement de tuples étiquetés par classe. Supposons que l'attribut d'étiquette de classe ait m valeurs distinctes définissant m classes distinctes $C_i, i \in \{1 \dots m\}$. Soit $C_{i,D}$ l'ensemble des tuples de classe C_i dans D. Soit $|D|$ et $\left|C_{i,D}\right|$ le nombre de tuples dans D et $C_{i,D}$, respectivement.

3.3.1. Gain d'informations

ID3 utilise le gain d'information comme mesure de sélection d'attribut. Cette mesure s'appuie sur les travaux pionniers de Claude Shannon sur la théorie de l'information, qui a étudié la valeur ou le « contenu informationnel » des messages. On suppose que le nœud N content les tuples de la partition D. L'attribut avec le gain d'informations le plus élevé est choisi comme attribut de fractionnement pour le nœud N. Cet attribut minimise les informations nécessaires pour classer les tuples dans les partitions résultantes et reflète le moins d'aléatoire dans ces cloisons.

Une telle approche minimise le nombre attendu de tests nécessaires pour classer un tuple donné et garantit qu'un arbre simple est trouvé. L'information attendue pour classer un tuple dans D est donnée par :

$$Info(D) = -\sum_{i=1}^{m} P_i \, log_2(P_i)$$

où P_i est la probabilité non nulle qu'un tuple arbitraire dans D appartenant à la classe C_i et est estimée par $\left|C_{i,D}\right| \, / |D|$. Une fonction log à la base 2 est utilisée, car les informations sont codées en bits. Info(D) est juste la quantité moyenne d'informations nécessaires pour identifier l'étiquette de classe d'un tuple dans D. Notez qu'à ce stade, les informations dont nous disposons sont uniquement basées sur les proportions de tuples de chaque classe. Info(D) est également connu comme l'entropie de D.

Maintenant, on désire partitionner les tuples dans D sur un attribut A ayant v valeurs distinctes, $\{a_1, a_2, \dots, a_v\}$ comme observé à partir des données d'apprentissage. Si A est à valeurs discrètes, ces valeurs correspondent directement aux v résultats d'un test sur A. L'attribut A peut être utilisé pour diviser D en v partitions ou sous-ensembles, $\{D_1, D_2, \dots, D_v\}$, où Dj contient ces tuples dans D qui ont pour résultat a_j de A. Ces partitions correspondraient aux branches issues du nœud N. Malheureusement, ce partitionnement produise une classification impure (par exemple, lorsqu'une partition peut contenir une collection de tuples de différentes classes plutôt que d'une seule classe). Pour arriver à une classification exacte on doit maximiser l'information sur A avec l'équation suivante :

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

Le terme $\frac{|D_j|}{|D|}$ agit comme le poids de la j[ième] partition. $Info_A(D)$ est l'information attendue requise pour classer un tuple de D en fonction du partitionnement par A. Plus l'information attendue est petite, plus la pureté des partitions est grande.

Le gain d'information est défini comme la différence entre l'exigence d'information d'origine (c'est-à-dire basée uniquement sur la proportion de classes) et la nouvelle exigence (c'est-à-dire obtenue après partitionnement sur A). C'est-à-dire,

$$Gain(A) = Info(D) - Info_A(D)$$

En d'autres termes, Gain(A) nous indique combien serait gagné en se branchant sur A. C'est la réduction attendue de l'exigence d'information causée par la connaissance de la valeur de A. L'attribut A avec le gain d'information le plus élevé est choisi comme attribut de fractionnement au nœud N.

Pour calculer le gain d'information d'un attribut à valeur continue, on doit déterminer le "meilleur" point de partage pour A, où le point de partage est un seuil sur A. Pour cela, les valeurs de A sont triées par ordre croissant. En règle générale, le point médian entre chaque paire de valeurs adjacentes est considéré comme un point de partage possible. Par conséquent, étant donné v valeurs de A, alors v-1 scissions possibles sont évaluées. Par exemple, le point médian entre les valeurs $a_i$et a_{i+1} de A est

$$\frac{a_i + a_{i+1}}{2}$$

Si les valeurs de A sont triées à l'avance, la détermination de la meilleure répartition pour A ne nécessite qu'un seul passage parmi les valeurs. Pour chaque point de division possible pour A, on évalue nous évaluons $Info_A(D)$, où le nombre de partitions est deux, c'est-à-dire v = 2.

Le point avec l'exigence minimale d'informations attendues pour A est sélectionné comme point de partage pour A. D_1 est l'ensemble de tuples dans D satisfaisant A point de partage, et D_2 est l'ensemble de tuples dans D satisfaisant A > point de partage.

3.3.2. Rapport de Gain

La mesure du gain d'information est biaisée vers des tests avec de nombreux résultats. C'est-à-dire qu'il préfère sélectionner des attributs ayant un grand nombre de valeurs. Par exemple, considérez un attribut qui agit comme un identifiant unique tel qu'un ID de produit. Une division sur l'ID de produit entraînerait un grand nombre de partitions (autant qu'il y a de valeurs), chacune contenant un seul tuple. Étant donné que chaque partition est pure, les informations requises pour classer l'ensemble de données D en fonction de ce partitionnement seraient $info_{product_{id(D)}=0}$. Par conséquent, les informations obtenues en partitionnant sur cet attribut sont maximales. Il est clair qu'un tel partitionnement est inutile pour la classification. C4.5, un successeur de ID3, utilise une extension du gain d'information connue sous le nom de rapport de gain, qui tente de surmonter ce biais. Il applique une sorte de normalisation au gain d'information à l'aide d'une valeur de "split information" définie de manière analogue à $info(D)$ comme

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2\left(\frac{|D_j|}{|D|}\right)$$

Cette valeur représente les informations potentielles générées en divisant l'ensemble de données d'apprentissage D en v partitions, correspondant aux v résultats d'un test sur l'attribut A. Notez que, pour chaque résultat, il considère le nombre de tuples ayant ce résultat par rapport au nombre total de tuples dans D. Il diffère du gain d'informations, qui mesure les informations relatives à la classification acquises sur la base du même partitionnement. Le rapport de gain est défini comme

$$GainRatio(A) = \frac{Gain\ (A)}{SplitInfo_A(D)}$$

L'attribut avec le rapport de gain maximal est sélectionné comme attribut de division. Cependant, que lorsque l'information fractionnée approche de 0, le rapport devient instable. Une contrainte est ajoutée pour éviter cela, selon laquelle le gain d'information du test sélectionné doit être important, au moins aussi grand que le gain moyen sur tous les tests examinés.

3.3.3. Indice de Gini

L'indice de Gini mesure l'impureté de D, une partition de données ou un ensemble de tuples d'apprentissage, telle que

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

où p_i est la probabilité qu'un tuple de D appartienne à la classe C_i et est estimée par $C_{i,D}/|D|$. La somme est calculée sur m classes.

L'indice de Gini est utilisé dans CART, il considère une division binaire pour chaque attribut. Considérons d'abord le cas où A est un attribut à valeurs discrètes ayant v valeurs distinctes, $a_1, a_2, \dots, a_v$ apparaissant dans D. Pour déterminer le meilleur découpage binaire sur A, nous examinons tous les sous-ensembles possibles qui peuvent être formé en utilisant des valeurs connues de A. Chaque sous-ensemble, S_A, peut être considéré comme un test binaire pour l'attribut A de la forme "A $\in S_A$". Étant donné un tuple, ce test est satisfait si la valeur de A pour le tuple fait partie des valeurs répertoriées dans S_A. Si A a v valeurs possibles, alors il y a 2^v sous-ensembles possibles.

Lorsqu'on considère une division binaire, on calcule une somme pondérée de l'impureté de chaque partition résultante. Par exemple, si une division binaire sur A partitionne D en D1 et D2, l'indice de Gini de D étant donné que le partitionnement est

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$

Pour chaque attribut, chacune des divisions binaires possibles est considérée. Pour un attribut à valeur discrète, le sous-ensemble qui donne l'indice de Gini minimum pour cet attribut est sélectionné comme son sous-ensemble de fractionnement.

Pour les attributs à valeur continue, chaque point de partage possible doit être pris en compte. La stratégie est similaire à celle décrite précédemment pour le gain d'informations, où le point médian entre chaque paire de valeurs adjacentes (triées) est pris comme point de partage possible. Le point donnant l'indice de Gini minimum pour un attribut donné (à valeur continue) est pris comme point de partage de cet attribut. Rappelons que pour un possible point de partage de A, D_1 est l'ensemble des tuples de D satisfaisant A ≤ point de partage, et D2 est l'ensemble des tuples de D satisfaisant A > point de partage.

La réduction de l'impureté qui serait encourue par une division binaire sur un attribut à valeur discrète ou continue A est

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

L'attribut qui maximise la réduction des impuretés (ou, de manière équivalente, qui a l'indice de Gini minimum) est sélectionné comme attribut de séparation. Cet attribut et son sous-ensemble de fractionnement (pour un attribut de fractionnement à valeur discrète) ou son point de fractionnement (pour un attribut de fractionnement à valeur continue) forment ensemble le critère de fractionnement.

3.4. Élagage des arbres

Lorsqu'un arbre de décision est construit, de nombreuses branches reflèteront des anomalies dans les données d'apprentissage dues au bruit ou aux valeurs aberrantes. Les méthodes d'élagage des arbres résolvent ce problème de surajustement des données. Ces méthodes utilisent généralement des mesures statistiques pour supprimer les branches les moins fiables. Les arbres taillés ont tendance à être plus petits et moins complexes et donc plus faciles à comprendre. Ils sont généralement plus rapides et plus efficaces pour classer correctement les données de test indépendantes que les arbres non élagués.

Il existe deux approches courantes de l'élagage des arbres : le pré-élagage et le post-élagage.

3.4.1. Pré-élagage

Dans l'approche de pré-élagage, un arbre est "élagué" en arrêtant sa construction plus tôt (par exemple, en décidant de ne pas diviser ou partitionner davantage le sous-ensemble de tuples

d'apprentissage à un nœud donné). Lors de l'arrêt, le nœud devient une feuille. La feuille peut contenir la classe la plus fréquente parmi les tuples du sous-ensemble ou la distribution de probabilité de ces tuples. Lors de la construction d'un arbre, des mesures telles que la signification statistique, le gain d'information, l'indice de Gini, etc., peut être utilisé pour évaluer la qualité d'une scission. Si le partitionnement des tuples au niveau d'un nœud entraîne une division qui tombe en dessous d'un seuil prédéfini, alors le partitionnement supplémentaire du sous-ensemble donné est arrêté. Il est cependant difficile de choisir un seuil approprié. Des seuils élevés pourraient entraîner des arbres trop simplifiés, tandis que des seuils bas pourraient entraîner très peu de simplification.

3.4.2. Post-élagage

La deuxième approche, la plus courante, est le post-élagage, qui supprime les sous-arbres d'un arbre entièrement développé. Un sous-arbre à un nœud donné est élagué en supprimant ses branches et en le remplaçant par une feuille. La feuille est étiquetée avec la classe la plus fréquente parmi le sous-arbre à remplacer.

L'algorithme d'élagage de la complexité des coûts utilisé dans CART est un exemple de l'approche de post-élagage. Cette approche considère que la complexité du coût d'un arbre est fonction du nombre de feuilles dans l'arbre et du taux d'erreur de l'arbre. Il part du bas de l'arbre. Pour chaque nœud interne N, il calcule la complexité de coût du sous-arbre en N, et la complexité de coût du sous-arbre en N s'il devait être élagué. Les deux valeurs sont comparées. Si l'élagage du sous-arbre au nœud N entraîne une moindre complexité de coût, alors le sous-arbre est élagué. Sinon, il est conservé.

Un ensemble d'élagage de tuples étiquetés par classe est utilisé pour estimer la complexité des coûts. Cet ensemble est indépendant de l'ensemble d'apprentissage utilisé pour construire l'arbre non élagué et de tout ensemble de test utilisé pour l'estimation de la précision. L'algorithme génère un ensemble d'arbres progressivement élagués. En général, le plus petit arbre de décision qui minimise la complexité des coûts est préféré.

C4.5 utilise une méthode appelée élagage pessimiste, qui est similaire à la méthode de la complexité des coûts en ce qu'elle utilise également des estimations du taux d'erreur pour

prendre des décisions concernant l'élagage des sous-arbres. La taille pessimiste ne nécessite pas l'utilisation d'un ensemble de pruneaux. Au lieu de cela, il utilise l'ensemble d'apprentissage pour estimer les taux d'erreur. Une estimation de la précision ou de l'erreur basée sur l'ensemble d'apprentissage est trop optimiste et, par conséquent, fortement biaisée. La méthode d'élagage pessimiste ajuste donc les taux d'erreur obtenus à partir de l'ensemble d'apprentissage en ajoutant une pénalité, de manière à contrer le biais encouru.

Plutôt que d'élaguer les arbres en fonction des taux d'erreur estimés, on peut élaguer les arbres en fonction du nombre de bits nécessaires pour les coder. Le meilleur arbre élagué est celui qui minimise le nombre de bits de codage.

Contrairement à l'élagage de la complexité des coûts, il ne nécessite pas un ensemble indépendant de tuples. Alternativement, le pré- élagage et le post- élagage peuvent être entrelacées pour une approche combinée. Le post-élagage nécessite plus de calculs que le pré-élagage, mais conduit généralement à un arbre plus fiable. Aucune méthode d'élagage ne s'est avérée supérieure à toutes les autres. Bien que certaines méthodes d'élagage dépendent de la disponibilité de données supplémentaires pour l'élagage, ce n'est généralement pas un problème lorsqu'il s'agit de grandes bases de données.

Bien que les arbres élagués aient tendance à être plus compacts que leurs homologues non élagués, ils peuvent encore être assez grands et complexes. Les arbres de décision peuvent souffrir de répétitions et de réplications, ce qui les rend accablants à interpréter. La répétition se produit lorsqu'un attribut est testé à plusieurs reprises le long d'une branche donnée de l'arbre

Dans la réplication, des sous-arborescences en double existent dans l'arborescence. Ces situations peuvent entraver l'exactitude et la compréhensibilité d'un arbre de décision.

L'utilisation de fractionnements multivariés peut prévenir ces problèmes. Une autre approche consiste à utiliser une forme différente de représentation des connaissances, telle que des règles, au lieu d'arbres de décision.

3.5. Évolutivité et induction de l'arbre de décision

L'efficacité des algorithmes d'arbre de décision existants, tels que ID3, C4.5 et CART, a été bien établie pour des ensembles de données relativement petits. L'efficacité devient un problème lorsque ces algorithmes sont appliqués à l'exploration de très grandes bases de données du monde réel. Les algorithmes d'arbre de décision pionniers ont la restriction que les tuples d'apprentissage doivent résider en mémoire.

Dans les applications d'exploration de données, de très grands ensembles d'apprentissage de millions de tuples sont courants.

Le plus souvent, les données d'entraînement ne tiennent pas en mémoire ! Par conséquent, la construction de l'arbre de décision devient inefficace en raison de l'échange des tuples d'apprentissage dans et hors de la mémoire principale et de la mémoire cache. Des approches plus évolutives, capables de gérer des données de formation trop volumineuses pour tenir en mémoire, sont nécessaires. Les stratégies antérieures pour économiser de l'espace comprenaient la discrétisation des attributs à valeur continue et l'échantillonnage des données à chaque nœud. Ces techniques supposent toujours que l'ensemble d'apprentissage peut tenir dans la mémoire.

Plusieurs méthodes d'induction d'arbres de décision évolutives ont été introduites dans des études récentes.

3.5.1. RainForest

Il s'adapte à la quantité de mémoire principale disponible et s'applique à tout algorithme d'induction d'arbre de décision. La méthode maintient un ensemble AVC (où "AVC" signifie "Attribute-Value, Classlabel") pour chaque attribut, à chaque nœud d'arbre, décrivant les tuples d'apprentissage au niveau du nœud. L'ensemble AVC d'un attribut A au nœud N donne les comptes d'étiquettes de classe pour chaque valeur de A pour les tuples à N. L'ensemble de tous les ensembles AVC à un nœud N est le groupe AVC de N. La taille d'un ensemble AVC pour l'attribut A au nœud N dépend uniquement du nombre de valeurs distinctes de A et du nombre de classes dans l'ensemble N. En règle générale, cette taille doit tenir en mémoire, même pour les données du monde réel. Cependant, RainForest dispose également de techniques pour gérer

le cas où le groupe AVC ne rentre pas dans la mémoire. Par conséquent, la méthode a une grande évolutivité pour l'induction d'arbre de décision dans de très grands ensembles de données.

3.5.2. BOAT (Bootstrapped Optimistic Algorithm for Tree construction)

C'est un algorithme d'arbre de décision qui adopte une approche complètement différente de l'évolutivité. Il n'est pas basé sur l'utilisation de structures de données spéciales. Au lieu de cela, il utilise une technique statistique connue sous le nom de "bootstrapping" pour créer plusieurs échantillons plus petits des données d'apprentissage données, dont chacun tient en mémoire. Chaque sous-ensemble est utilisé pour construire un arbre, résultant en plusieurs arbres. Les arbres sont examinés et utilisés pour construire un nouvel arbre, T', qui s'avère être très proche de l'arbre qui aurait été généré si toutes les données d'apprentissage d'origine avaient tenu en mémoire.

BOAT peut utiliser n'importe quelle mesure de sélection d'attribut qui sélectionne des divisions binaires et qui est basée sur la notion de pureté des partitions telles que l'indice de Gini. BOAT utilise une limite inférieure sur la mesure de sélection d'attributs pour détecter si cet arbre très bon T', est différent du "vrai" arbre T, qui aurait été généré en utilisant toutes les données. Il raffine T'pour arriver à T.

BOAT ne nécessite généralement que deux balayages de D. C'est une nette amélioration, même par rapport aux algorithmes d'arbre de décision traditionnels qui nécessitent un balayage par niveau d'arbre. BOAT s'est avéré deux à trois fois plus rapide que RainForest, tout en construisant exactement le même arbre. Un avantage supplémentaire de BOAT est qu'il peut être utilisé pour des mises à jour incrémentielles. Autrement dit, BOAT peut prendre de nouvelles insertions et suppressions pour les données d'apprentissage et mettre à jour l'arbre de décision pour refléter ces changements, sans avoir à reconstruire l'arbre à partir de zéro.

4. Méthodes de classification de Bayes

Les classificateurs bayésiens sont des classificateurs statistiques. Ils peuvent prédire les probabilités d'appartenance à une classe, telles que la probabilité qu'un tuple donné appartienne à une classe particulière. La classification bayésienne est basée sur le théorème de Bayes. Des

études comparant les algorithmes de classification ont trouvé qu'un classificateur bayésien simple connu sous le nom de classificateur bayésien natif était comparable en termes de performances avec l'arbre de décision et les classificateurs de réseau neuronal sélectionnés. Les classificateurs bayésiens ont également fait preuve d'une précision et d'une vitesse élevées lorsqu'ils sont appliqués à de grandes bases de données. Les classificateurs bayésiens natifs supposent que l'effet d'une valeur d'attribut sur une classe donnée est indépendant des valeurs des autres attributs. Cette hypothèse est appelée indépendance conditionnelle de classe. Il est fait pour simplifier les calculs impliqués et, en ce sens, est considéré comme "naïf".

4.1. Théorème de Bayes

Le théorème de Bayes porte le nom de Thomas Bayes, un pasteur anglais non conformiste qui a fait ses premiers travaux sur la théorie des probabilités et de la décision au XVIIIe siècle. Soit X un tuple de données. En termes bayésiens, X est considéré comme une preuve. Comme d'habitude, il est décrit par des mesures effectuées sur un ensemble de n attributs. Soit H une hypothèse telle que le tuple de données X appartient à une classe C spécifiée. Pour les problèmes de classification, nous voulons déterminer P(H\X), la probabilité que l'hypothèse H soit vérifiée compte tenu de la preuve ou du tuple de données observé X. En d'autres termes, nous recherchons la probabilité que le tuple X appartienne à la classe C, étant donné que nous connaissons la description de l'attribut de X. P(H\X) est appelée la probabilité a posteriori de H conditionnée sur X tandis que P(H) est la probabilité a priori de H. P(H), P(X\H) et P(X) peuvent être estimés à partir des données fournies. Le théorème de Bayes est utile car il fournit un moyen de calculer la probabilité a posteriori, P(H\X), à partir de P(H), P(X\H) et P(X). Le théorème de Bayes est

$$P(H\backslash X) = \frac{P(X\backslash H).P(H)}{P(X)}$$

4.2. Classification Bayésienne Naïve

Le classificateur bayésien naïve, ou classificateur bayésien simple, fonctionne comme suit :

- Soit D un ensemble d'entraînement de tuples et leurs étiquettes de classe associées. Comme d'habitude, chaque tuple est représenté par un vecteur d'attributs à n dimensions

$X = (x_1, x_2, \dots, x_n)$ représentant n mesures effectuées sur le tuple à partir de n attributs, respectivement $A_1, A_2, \dots, A_n$.

- Supposons qu'il existe m classes $C_1, C_2, \dots, C_m$. Étant donné un tuple X le classificateur prédira que X appartient à la classe ayant la probabilité a posteriori la plus élevée, conditionnée sur X. Autrement dit, le classificateur bayésien naïve prédit que le tuple X appartient à la classe C_i si et seulement si $P(C_i\backslash X) > P(C_j\backslash \text{X})$ pour 1 ≤ j ≤m, j ≠ i. Ainsi, on maximise $P(C_i\backslash X)$. La classe C_i pour laquelle $P(C_i\backslash X)$ est maximisée est appelée hypothèse du maximum a posteriori. Par le théorème de Bayes

$$P(C_i\backslash X) = \frac{P(X\backslash C_i).P(C_i)}{P(X)}$$

- Comme P(X) est constant pour toutes les classes, seul $P(X\backslash C_i).P(C_i)$ doit être maximisé. Si les probabilités a priori de la classe ne sont pas connues, alors on suppose généralement que les classes sont équiprobable , c'est-à-dire $P(C_1) = P(C_2) = \cdots = P(C_m)$et nous maximiserons donc $P(X\backslash C_i)$. Sinon, on maximise $P(X\backslash C_i).P(C_i)$. Notez que les probabilités a priori de la classe peuvent être estimées par $P(C_i) = \left|C_{i,D}\right|/|D|$ où $\left|C_{i,D}\right|$est le nombre de tuples d'apprentissage de la classe C_i dans D.
- Compte tenu des ensembles de données avec de nombreux attributs, il serait extrêmement coûteux en calcul de calculer $P(X\backslash C_i)$. Pour réduire le calcul lors de l'évaluation de $P(X\backslash C_i)$, l'hypothèse naïve d'indépendance conditionnelle de classe est faite. Cela suppose que les valeurs des attributs sont conditionnellement indépendantes les unes des autres, compte tenu de l'étiquette de classe du tuple (c'est-à-dire qu'il n'y a pas de relations de dépendance entre les attributs). Ainsi

$$P(X\backslash C_i) = \prod_{k=1}^{n} P(x_k \backslash C_i)$$

On peut facilement estimer les probabilités $P(x_k\backslash C_i)$ à partir des tuples d'apprentissage. Rappelons qu'ici x_k fait référence à la valeur de l'attribut A_k pour le tuple X. Pour chaque attribut, nous examinons si l'attribut est catégoriel ou à valeur continue. Par exemple, pour calculer $P(X\backslash C_i)$, on considère ce qui suit :

- Si A_k est catégorique, alors $P(x_k \backslash C_i)$ est le nombre de tuples de classe C_i dans D ayant la valeur x_k pour A_k, divisé par $|C_{i,D}|$, le nombre de tuples de classe C_i dans D.
- Si A_k est à valeur continue, alors $P(x_k \backslash C_i) = g(x_k, \mu_{c_i}, \sigma_{c_i})$. Un attribut à valeur continue est généralement supposé avoir une distribution gaussienne avec une moyenne et un écart type.

- Pour prédire l'étiquette de classe de X, $P(X \backslash C_i).P(C_i)$ est évalué pour chaque classe C_i . Le classifieur prédit que le label de classe du tuple X est la classe C_i si et seulement si $P(X \backslash C_i).P(C_i) > P(X \backslash C_j).P(C_j)$ pour 1 ≤ j ≤m, j ≠ i. En d'autres termes, l'étiquette de classe prédite est la classe C_i pour laquelle $P(X \backslash C_i).P(C_i)$ est le maximale.

Diverses études empiriques de ce classificateur par rapport aux classificateurs d'arbre de décision et de réseau neuronal ont trouvé qu'il était comparable dans certains domaines. En théorie, les classificateurs bayésiens ont le taux d'erreur minimum par rapport à tous les autres classificateurs. Cependant, dans la pratique, ce n'est pas toujours le cas, en raison d'inexactitudes dans les hypothèses formulées pour son utilisation, telles que l'indépendance conditionnelle de classe, et du manque de données de probabilité disponibles.

Les classificateurs bayésiens sont également utiles en ce sens qu'ils fournissent une justification théorique pour d'autres classificateurs qui n'utilisent pas explicitement le théorème de Bayes.

4.3. Classification basée sur des règles

Dans la classification basée sur des règles, le modèle appris est représenté sous la forme d'un ensemble de règles SI-ALORS. Ces derniers peuvent être générées, soit à partir d'un arbre de décision, soit directement à partir des données d'apprentissage à l'aide d'un algorithme de recouvrement séquentiel.

4.4. Utilisation des règles SI-ALORS pour la classification

Les règles sont un bon moyen de représenter des informations ou des éléments de connaissance. Un classificateur basé sur des règles utilise un ensemble de règles SI-ALORS pour la classification. Une règle SI-ALORS est une expression de la forme SI condition ALORS conclusion.

La partie "SI" (ou côté gauche) d'une règle est connue sous le nom d'antécédent de règle ou de précondition. La partie "ALORS" (ou côté droit) est la règle conséquente. Dans l'antécédent de règle, la condition consiste en un ou plusieurs tests d'attribut. Le conséquent de la règle contient une prédiction de classe. Si la condition dans un antécédent de règle est vraie pour un tuple donné, nous disons que l'antécédent de règle est satisfait et que la règle couvre le tuple.

Une règle R peut être évaluée par sa couverture et sa précision. Étant donné un tuple, X, à partir d'un ensemble de données étiqueté par classe, D, soit $n_{couvert}$ le nombre de tuples couverts par R ; $n_{correct}$ soit le nombre de tuples correctement classés par R ; et |D| le nombre de tuples dans D. Nous pouvons définir la couverture et la précision de R comme

$$Couverture\ (R) = \frac{n_{couvert}}{|D|}, Précision = \frac{n_{correct}}{n_{couvert}}$$

Autrement dit, la couverture d'une règle est le pourcentage de tuples qui sont couverts par la règle. Pour la précision d'une règle, nous examinons les tuples qu'elle couvre et voyons quel pourcentage d'entre eux la règle peut correctement classer. Si une règle est satisfaite par X, la règle est dite déclenchée. Notez que le déclenchement ne signifie pas toujours le déclenchement car il peut y avoir plus d'une règle qui est satisfaite ! Si plus d'une règle est déclenchée, nous avons un problème potentiel.

Si plusieurs règles sont déclenchées, nous avons besoin d'une stratégie de résolution de conflit pour déterminer quelle règle se déclenche et attribuer sa prédiction de classe à X. Il existe de nombreuses stratégies possibles les plus utilisées sont : l'ordre des tailles et l'ordre des règles.

4.4.1. Résolution de conflit par l'ordre des tailles

Le schéma de classement par taille attribue la priorité la plus élevée à la règle de déclenchement qui a les exigences les plus dures, où la robustesse est mesurée par la taille de l'antécédent de la règle. C'est-à-dire que la règle de déclenchement avec le plus de tests d'attributs est déclenchée.

Le schéma de classement des règles hiérarchise les règles au préalable. Le classement peut être basé sur des classes ou sur des règles. Avec le classement basé sur les classes, les classes sont triées par ordre décroissant d'importance. C'est-à-dire que toutes les règles de la classe la plus fréquente viennent en premier, les règles de la classe la plus répandue suivante viennent ensuite,

et ainsi de suite. Alternativement, ils peuvent être triés en fonction du coût de mauvaise classification par classe. Au sein de chaque classe, les règles ne sont pas ordonnées - elles n'ont pas à l'être car elles prédisent toutes la même classe.

4.4.2. Résolution de conflit par l'ordre des règles

Avec le classement basé sur des règles, les règles sont organisées en une longue liste de priorités, selon une certaine mesure de la qualité des règles, telle que la précision, la couverture ou la taille, ou sur la base des conseils d'experts du domaine. Lorsque l'ordre des règles est utilisé, l'ensemble de règles est appelé liste de décision. Avec l'ordre des règles, la règle de déclenchement qui apparaît en premier dans la liste a la priorité la plus élevée et déclenche donc sa prédiction de classe. Toute autre règle qui satisfait X est ignorée. La plupart des systèmes de classification basés sur des règles utilisent une stratégie de classement des règles basée sur les classes.

Dans la première stratégie, globalement les règles ne sont pas ordonnées. Ils peuvent être appliqués dans n'importe quel ordre lors de la classification d'un tuple. C'est-à-dire qu'une disjonction est implicite entre chacune des règles. Chaque règle représente un élément de connaissance autonome. Cela contraste avec le schéma d'ordonnancement des règles pour lequel les règles doivent être appliquées dans l'ordre prescrit afin d'éviter les conflits. Chaque règle dans une liste de décision implique la négation des règles qui la précèdent dans la liste. Par conséquent, les règles d'une liste de décision sont plus difficiles à interpréter.

Lorsqu'il n'y a pas de règle satisfaite par X, une règle de secours ou par défaut peut être configurée pour spécifier une classe par défaut, basée sur un ensemble de formation. Il peut s'agir de la classe majoritaire ou de la classe majoritaire des tuples qui n'étaient couverts par aucune règle. La règle par défaut est évaluée à la fin, si et seulement si aucune autre règle ne couvre X. La condition dans la règle par défaut est vide. De cette manière, la règle se déclenche lorsqu'aucune autre règle n'est satisfaite.

4.4.3. Extraction de règles à partir d'un arbre de décision

Pour extraire des règles d'un arbre de décision, une règle est créée pour chaque chemin de la racine à un nœud feuille. Chaque critère de découpage le long d'un chemin donné est

logiquement associé à un ET pour former l'antécédent de la règle (partie "SI"). Le nœud feuille contient la prédiction de classe, formant la règle conséquente (partie "ALORS").

Une disjonction (OU logique) est implicite entre chacune des règles extraites. Parce que les règles sont extraites directement de l'arbre, elles sont mutuellement exclusives et exhaustives. Mutuellement exclusif signifie que nous ne pouvons pas avoir de conflits de règles ici car aucune règle ne sera déclenchée pour le même tuple. (Nous avons une règle par feuille, et tout tuple ne peut être mappé qu'à une seule feuille.) Exhaustif signifie qu'il existe une règle pour chaque combinaison attribut-valeur possible, de sorte que cet ensemble de règles ne nécessite pas de règle par défaut. Par conséquent, l'ordre des règles n'a pas d'importance, elles ne sont pas ordonnées.

Puisqu'on se retrouve avec une règle par feuille, l'ensemble des règles extraites n'est pas beaucoup plus simple que l'arbre de décision correspondant ! Les règles extraites peuvent être encore plus difficiles à interpréter que les arbres d'origine dans certains cas. L'ensemble de règles résultant extrait peut être volumineux et difficile à suivre, car certains des tests d'attributs peuvent être non pertinents ou redondants. Bien qu'il soit facile d'extraire des règles d'un arbre de décision, nous devrons peut-être faire un peu plus de travail en élaguant l'ensemble de règles résultant.

Pour un antécédent de règle donné, toute condition qui n'améliore pas la précision estimée de la règle peut être élaguée (c'est-à-dire supprimée), généralisant ainsi la règle. C4.5 extrait les règles d'un arbre non élagué, puis élague les règles en utilisant une approche pessimiste similaire à sa méthode d'élagage d'arbre. Les tuples d'apprentissage et leurs étiquettes de classe associées sont utilisés pour estimer la précision des règles. Cependant, comme cela se traduirait par une estimation optimiste, alternativement, l'estimation est ajustée pour compenser le biais, ce qui donne une estimation pessimiste. En outre, toute règle qui ne contribue pas à la précision globale de l'ensemble de règles peut également être élaguée.

Cependant, d'autres problèmes surviennent lors de l'élagage des règles, car les règles ne seront plus mutuellement exclusives et exhaustives. Pour la résolution des conflits, C4.5 adopte un schéma de classement basé sur les classes. Il regroupe toutes les règles d'une même classe, puis

détermine un classement de ces ensembles de règles de classe. Dans un jeu de règles, les règles ne sont pas ordonnées. C4.5 ordonne les ensembles de règles de classe de manière à minimiser le nombre d'erreurs de faux positifs. La règle de classe définie avec le moins de faux positifs est examinée en premier. Une fois l'élagage terminé, une dernière vérification est effectuée pour supprimer les doublons. Lors du choix d'une classe par défaut, C4.5 ne choisit pas la classe majoritaire, car cette classe aura probablement de nombreuses règles pour ses tuples. Au lieu de cela, il sélectionne la classe qui contient le plus de tuples d'apprentissage qui n'étaient couverts par aucune règle.

4.4.4. Induction de règles à l'aide d'un algorithme de recouvrement séquentiel

Les règles SI-ALORS peuvent être extraites directement des données d'apprentissage à l'aide d'un algorithme de recouvrement séquentiel. Le nom vient de la notion que les règles sont apprises séquentiellement, où chaque règle pour une classe donnée couvrira idéalement plusieurs des tuples de la classe. Les algorithmes de couverture séquentielle sont l'approche la plus largement utilisée pour extraire des ensembles disjonctifs de règles de classification.

Il existe de nombreux algorithmes de recouvrement séquentiel. Les variantes populaires incluent AQ, CN2 et le plus récent RIPPER. Les règles sont apprises une par une. Chaque fois qu'une règle est apprise, les tuples couverts par la règle sont supprimés et le processus se répète sur les tuples restants. Cet apprentissage séquentiel de règles s'oppose à l'induction par arbre de décision. Étant donné que le chemin vers chaque feuille d'un arbre de décision correspond à une règle, on peut considérer l'induction de l'arbre de décision comme l'apprentissage simultané d'un ensemble de règles.

Idéalement, lors de l'apprentissage d'une règle pour une classe C, on essaye que la règle couvre tous les tuples d'apprentissage de la classe C et aucun des tuples des autres classes. De cette façon, les règles apprises doivent être d'une grande précision. Les règles ne doivent pas nécessairement avoir une couverture élevée. En effet, nous pouvons avoir plusieurs règles pour une classe, de sorte que différentes règles peuvent couvrir différents tuples au sein de la même classe. Le processus se poursuit jusqu'à ce que la condition de fin soit remplie, par exemple lorsqu'il n'y a plus de tuples d'apprentissage ou que la qualité d'une règle renvoyée est inférieure

à un seuil spécifié par l'utilisateur. La procédure Learn One Rule trouve la meilleure règle pour la classe actuelle, compte tenu de l'ensemble actuel de tuples d'apprentissage.

Généralement, les règles sont développées de manière générale à spécifique. On peut considérer cela comme une recherche de faisceau, où il s'agit de commencer avec une règle vide, puis continuer progressivement à y ajouter des tests d'attributs. L'attribut test est ajouté comme un conjoint logique à la condition existante de l'antécédent de la règle.

Ensuite chaque test d'attribut possible qui peut être ajouté à la règle est considéré. Ceux-ci peuvent être dérivés du paramètre Att_vals, qui contient une liste d'attributs avec leurs valeurs associées. Par exemple, pour une paire attribut-valeur, on peut envisager des tests d'attribut tels que att = val, att ≤ val, att > val, etc. En règle générale, les données d'apprentissage contiendront de nombreux attributs, chacun pouvant avoir plusieurs valeurs possibles. Trouver un ensemble de règles optimal devient explosif en termes de calcul. Au lieu de cela, Learn One Rule adopte une stratégie gourmande en profondeur. Chaque fois qu'il est confronté à l'ajout d'un nouveau test d'attribut à la règle actuelle, il sélectionne celui qui améliore le plus la qualité de la règle, en fonction des échantillons d'apprentissage.

La recherche gourmande ne permet pas de revenir en arrière. A chaque étape, on ajoute heuristiquement ce qui semble être le meilleur choix du moment. Pour réduire le risque de faire le mauvais choix, au lieu de sélectionner le meilleur test d'attribut à ajouter à la règle actuelle, on peut sélectionner les meilleurs tests d'attribut k. De cette manière, une recherche de faisceau de largeur k est effectuée, dans laquelle les k meilleurs candidats globalement à chaque étape sont maintenus, plutôt qu'un seul meilleur candidat.

Chapitre V : Clustering

1 Définition

Le clustering est le processus de partitionnement d'un ensemble d'objets de données ou d'observations en sous-ensembles. Chaque sous-ensemble est un cluster, de sorte que les objets d'un cluster sont similaires les uns aux autres, mais différents des objets des autres clusters. L'ensemble de clusters résultant d'une analyse de cluster peut être appelé clustering. Dans ce contexte, différentes méthodes de clustering peuvent générer différents clusterings sur le même ensemble de données. Le partitionnement n'est pas effectué par des humains, mais par l'algorithme de clustering. Par conséquent, le regroupement est utile dans la mesure où il peut conduire à la découverte de groupes jusque-là inconnus dans les données.

2. Domaine d'application

L'analyse de cluster a été largement utilisée dans de nombreuses applications telles que l'informatique décisionnelle, la reconnaissance de formes d'images, la recherche sur le Web, la biologie et la sécurité. En intelligence d'affaires, le clustering peut être utilisé pour organiser un grand nombre de clients en groupes, où les clients au sein d'un groupe partagent de fortes caractéristiques similaires. Cela facilite le développement de stratégies commerciales pour une meilleure gestion de la relation client.

De plus, pour améliorer la gestion de projet, le regroupement peut être appliqué pour diviser les projets en catégories basées sur la similarité afin que l'audit et le diagnostic de projet puissent être menés efficacement. Dans la reconnaissance d'images, le regroupement peut être utilisé pour découvrir des groupes ou des « sous-classes » dans les systèmes de reconnaissance de caractères manuscrits. Le clustering a également trouvé de nombreuses applications dans la recherche Web. Par exemple, une recherche par mot-clé peut souvent renvoyer un très grand nombre de résultats en raison du très grand nombre de pages Web. Le regroupement peut être utilisé pour organiser les résultats de la recherche en groupes et présenter les résultats de manière concise et facilement accessible.

De plus, des techniques de regroupement ont été développées pour regrouper les documents en sujets, qui sont couramment utilisés dans la pratique de la recherche d'informations.

En tant que fonction d'exploration de données, l'analyse de clusters peut être utilisée comme un outil autonome pour mieux comprendre la distribution des données, observer les caractéristiques de chaque cluster et se concentrer sur un ensemble particulier de clusters pour une analyse plus approfondie. Alternativement, il peut servir d'étape de prétraitement pour d'autres algorithmes, tels que la caractérisation, la sélection de sous-ensembles d'attributs et la classification, qui opéreraient alors sur les clusters détectés et les attributs ou caractéristiques sélectionnés.

Étant donné qu'un cluster est une collection d'objets de données similaires les uns aux autres au sein du cluster et différents des objets d'autres clusters, un cluster d'objets de données peut être traité comme une classe implicite. En ce sens, le regroupement est parfois appelé classification automatique. Le clustering peut automatiquement trouver les groupements. Il s'agit d'un avantage distinct de l'analyse par clusters.

Le clustering est également appelé segmentation des données dans certaines applications, car le clustering partitionne de grands ensembles de données en groupes en fonction de leur similarité. Le clustering peut également être utilisé pour la détection des valeurs aberrantes, où les valeurs aberrantes peuvent être plus intéressantes que les cas courants. Les applications de la détection des valeurs aberrantes comprennent la détection de la fraude par carte de crédit et la surveillance des activités criminelles dans le commerce électronique.

Le regroupement de données est en plein développement. Les domaines de recherche contributifs comprennent l'exploration de données, les statistiques, l'apprentissage automatique, la technologie des bases de données spatiales, la recherche d'informations, la recherche sur le Web, la biologie, le marketing et de nombreux autres domaines d'application. En raison des énormes quantités de données collectées dans les bases de données, l'analyse par clusters est récemment devenue un sujet très actif dans la recherche sur l'exploration de données.

En tant que branche des statistiques, l'analyse typologique a été largement étudiée, l'accent étant mis principalement sur l'analyse typologique basée sur la distance. Des outils d'analyse de cluster basés sur k-means, k-medoids et plusieurs autres méthodes ont également été intégrés à de nombreux progiciels ou systèmes d'analyse statistique, tels que S-Plus, SPSS et SAS.

3. Clustering vs classification

Dans l'apprentissage automatique, la classification est connue sous le nom d'apprentissage supervisé parce que les informations d'étiquette de classe sont données, c'est-à-dire que l'algorithme d'apprentissage est supervisé en ce sens qu'il est informé de l'appartenance à la classe de chaque tuple d'apprentissage. Le clustering est connu sous le nom d'apprentissage non supervisé car les informations d'étiquette de classe ne sont pas présentes. Pour cette raison, le regroupement est une forme d'apprentissage par l'observation, plutôt que d'apprentissage par des exemples. Dans le domaine de l'exploration de données, les efforts se sont concentrés sur la recherche de méthodes pour une analyse par clusters efficace et efficiente dans de grandes bases de données. Les thèmes de recherche actifs portent sur l'évolutivité des méthodes de regroupement, l'efficacité des méthodes de regroupement de formes complexes et de types de données, les techniques de regroupement de grande dimension et des méthodes pour regrouper des données numériques et nominales mixtes dans de grandes bases de données.

4. Les contraintes

Le clustering est un domaine de recherche exigeant. Parmi les exigences typiques du clustering dans le Data mining :

- **Évolutivité :** de nombreux algorithmes de clustering fonctionnent bien sur de petits ensembles de données contenant moins de plusieurs centaines d'objets de données ; cependant, une grande base de données peut contenir des millions, voire des milliards d'objets, en particulier dans les scénarios de recherche sur le Web. Le regroupement sur un seul échantillon d'un grand ensemble de données donné peut conduire à des résultats biaisés. Par conséquent, des algorithmes de clustering hautement évolutifs sont nécessaires.

- **Capacité à traiter différents types d'attributs :** de nombreux algorithmes sont conçus pour regrouper des données numériques. Cependant, les applications peuvent nécessiter le regroupement d'autres types de données, tels que des données binaires, nominales et ordinales, ou des mélanges de ces types de données. Récemment, de plus en plus d'applications ont besoin de techniques de clustering pour des types de données complexes tels que des graphiques, des séquences, des images et des documents.
- **Découverte de clusters de forme arbitraire :** De nombreux algorithmes de clustering déterminent des clusters basés sur des mesures de distance euclidienne ou Manhattan. Les algorithmes basés sur de telles mesures de distance ont tendance à trouver des clusters sphériques de taille et de densité similaires. Cependant, un cluster peut avoir n'importe quelle forme.
- **Déterminer les paramètres d'entrée :** de nombreux algorithmes de clustering exigent que les utilisateurs fournissent une connaissance du domaine sous la forme de paramètres d'entrée tels que le nombre souhaité de clusters. Par conséquent, les résultats de regroupement peuvent être sensibles à ces paramètres. Les paramètres sont souvent difficiles à déterminer, en particulier pour les ensembles de données à haute dimensionnalité et lorsque les utilisateurs n'ont pas encore acquis une compréhension approfondie de leurs données. Exiger la spécification de la connaissance du domaine non seulement pèse sur les utilisateurs, mais rend également la qualité du clustering difficile à contrôler.
- **Capacité à traiter des données bruyantes :** la plupart des ensembles de données du monde réel contiennent des valeurs aberrantes et/ou des données manquantes, inconnues ou erronées. Les lectures des capteurs, par exemple, sont souvent bruyantes - certaines lectures peuvent être inexactes en raison des mécanismes de détection, et certaines lectures peuvent être erronées en raison d'interférences provenant d'objets transitoires environnants. Les algorithmes de clustering peuvent être sensibles à ce bruit et peuvent produire des clusters de mauvaise qualité. Par conséquent, nous avons besoin de méthodes de clustering robustes au bruit.

- **Regroupement incrémentiel :** dans de nombreuses applications, des mises à jour incrémentielles (représentant des données plus récentes) peuvent arriver à tout moment. Certains algorithmes de clustering ne peuvent pas incorporer des mises à jour incrémentielles dans les structures de clustering existantes et, à la place, doivent recalculer un nouveau clustering à partir de zéro. Les algorithmes de clustering peuvent également être sensibles à l'ordre des données d'entrée. Autrement dit, étant donné un ensemble d'objets de données, les algorithmes de regroupement peuvent renvoyer des regroupements radicalement différents selon l'ordre dans lequel les objets sont présentés. Des algorithmes de clustering incrémentiels et des algorithmes insensibles à l'ordre d'entrée sont nécessaires.
- **Capacité de regrouper des données à haute dimensionnalité :** un ensemble de données peut contenir de nombreuses dimensions ou attributs. Lors du regroupement de documents, par exemple, chaque mot-clé peut être considéré comme une dimension, et il existe souvent des milliers de mots-clés. La plupart des algorithmes de clustering sont bons pour gérer des données de faible dimension telles que des ensembles de données impliquant seulement deux ou trois dimensions. Trouver des clusters d'objets de données dans un espace de grande dimension est difficile, surtout si l'on considère que ces données peuvent être très rares et très asymétriques.
- **Clustering basé sur les contraintes :** les applications du monde réel peuvent avoir besoin d'effectuer un clustering sous différents types de contraintes. Il consiste à trouver des groupes de données avec un bon comportement de regroupement qui satisfont aux contraintes spécifiées.
- **Interprétabilité et convivialité :** les utilisateurs veulent que les résultats de clustering soient interprétables, compréhensibles et utilisables. Autrement dit, le regroupement peut devoir être lié à des interprétations et applications sémantiques spécifiques. Il est important d'étudier comment un objectif d'application peut influencer la sélection des méthodes et caractéristiques de clustering

5. Taxonomie

Voici les aspects avec lesquels les méthodes de clustering peuvent être comparées :

- **Les critères de partitionnement :** Dans certaines méthodes, tous les objets sont partitionnés de sorte qu'aucune hiérarchie n'existe entre les clusters. Autrement dit, tous les clusters sont conceptuellement au même niveau. Une telle méthode est utile, par exemple, pour partitionner les clients en groupes afin que chaque groupe ait son propre gestionnaire. Alternativement, d'autres méthodes partitionnent les objets de données de manière hiérarchique, où des clusters peuvent être formés à différents niveaux sémantiques.
- **Séparation des clusters :** certaines méthodes partitionnent les objets de données en clusters mutuellement exclusifs. Dans certaines autres situations, les clusters peuvent ne pas être exclusifs, c'est-à-dire qu'un objet de données peut appartenir à plusieurs clusters.
- **Mesure de similarité :** Certaines méthodes déterminent la similarité entre deux objets par la distance qui les sépare. Une telle distance peut être définie sur un espace euclidien, un réseau routier, un espace vectoriel, ou tout autre espace. Dans d'autres méthodes, la similarité peut être définie par la connectivité basée sur la densité ou la contiguïté, et peut ne pas reposer sur la distance absolue entre deux objets. Les mesures de similarité jouent un rôle fondamental dans la conception des méthodes de clustering. Alors que les méthodes basées sur la distance peuvent souvent tirer parti des techniques d'optimisation, les méthodes basées sur la densité et la continuité peuvent souvent trouver des clusters de forme arbitraire.
- **Espace de clustering :** de nombreuses méthodes de clustering recherchent des clusters dans l'ensemble de l'espace de données. Ces méthodes sont utiles pour les ensembles de données de faible dimensionnalité. Cependant, avec des données de grande dimension, il peut y avoir de nombreux attributs non pertinents, ce qui peut rendre les mesures de similarité peu fiables. Par conséquent, les clusters trouvés dans l'espace complet sont souvent dénués de sens. Il est souvent préférable de rechercher plutôt des clusters dans

différents sous-espaces du même ensemble de données. Le clustering de sous-espaces découvre des clusters et des sous-espaces qui manifestent une similarité d'objet.

6. Méthodes de clustering

Il existe de nombreux algorithmes de clustering dans la littérature. Il est difficile de fournir une catégorisation précise des méthodes de regroupement car ces catégories peuvent se chevaucher de sorte qu'une méthode peut avoir des caractéristiques de plusieurs catégories. Néanmoins, il est utile de présenter une image relativement organisée des méthodes de clustering. Ces méthodes sont brièvement résumées dans le tableau suivant.

Méthode	Caractéristiques
Partitionnement	• Trouver des clusters mutuellement exclusifs de forme sphérique • Basé sur la distance • Peut utiliser la moyenne ou la moyenne pour représenter le centre du cluster • Efficace pour les ensembles de données de petite à moyenne taille
Hiérarchiques	• Impossible de corriger les fusions ou les scissions erronées • Peut incorporer d'autres techniques comme le microclustering ou considérer les "liens" d'objets
Basé sur la Densité	• Peut trouver des clusters de forme arbitraire • Les clusters sont des régions denses d'objets dans l'espace qui sont séparés par des régions à faible densité • Densité de cluster : Chaque point doit avoir un nombre minimum de points dans son "voisinage" • Peut filtrer les valeurs aberrantes
Basé sur la grille	• Utiliser une structure de données de grille multirésolution • Temps de traitement rapide (généralement indépendant du nombre d'objets de données, mais dépendant de la taille de la grille)

Tableau 5.1. Caractéristiques de base des méthodes de clustering

Certains algorithmes de clustering intègrent les idées de plusieurs méthodes de clustering, de sorte qu'il est parfois difficile de classer un algorithme donné comme appartenant uniquement à une seule catégorie de méthode de clustering. De plus, certaines applications peuvent avoir des critères de clustering qui nécessitent l'intégration de plusieurs techniques de clustering. En général, la notation utilisée est la suivante : Soit « D » un ensemble de données de n objets à

regrouper. Un objet est décrit par d variables, où chaque variable est également appelée un attribut ou une dimension, et peut donc également être appelée un point dans un espace objet à d dimensions.

6.1. Méthodes de partitionnement

Étant donné un ensemble de n objets, une méthode de partitionnement construit k partitions de données, où chaque partition représente un cluster. Autrement dit, il divise les données en k groupes de sorte que chaque groupe doit contenir au moins un objet.

Les méthodes de partitionnement de base adoptent généralement une séparation exclusive des clusters. Autrement dit, chaque objet doit appartenir à exactement un groupe. Cette exigence peut être assouplie, par exemple, dans les techniques de partition floue.

La plupart des méthodes de partitionnement sont basées sur la distance. Étant donné k le nombre de partitions à construire, une méthode de partitionnement crée un partitionnement initial. Il utilise ensuite une technique de relocalisation itérative qui tente d'améliorer le partitionnement en déplaçant des objets d'un groupe à un autre. Le critère général d'un bon partitionnement est que les objets d'un même cluster sont « proches » ou liés les uns aux autres, alors que les objets de clusters différents sont « éloignés » ou très différents. Il existe différents types de critères pour juger de la qualité des partitions. Les méthodes de partitionnement traditionnelles peuvent être étendues pour le clustering de sous-espace, plutôt que de rechercher l'espace de données complet. Ceci est utile lorsqu'il existe de nombreux attributs et que les données sont rares.

Atteindre l'optimalité globale dans le clustering basé sur le partitionnement est souvent prohibitif en termes de calcul, nécessitant potentiellement une énumération exhaustive de toutes les partitions possibles. Au lieu de cela, la plupart des applications adoptent des méthodes heuristiques populaires, telles que des approches gourmandes comme les algorithmes k-means et k-medoids, qui améliorent progressivement la qualité du clustering et se rapprochent d'un optimum local. Ces méthodes de clustering heuristiques fonctionnent bien pour trouver des clusters de forme sphérique dans des bases de données de petite à moyenne taille. Pour trouver

des clusters aux formes complexes et pour de très grands ensembles de données, les méthodes basées sur le partitionnement doivent être étendues.

6.1.1. k-Means

Supposons qu'un ensemble de données, D, contienne n objets dans l'espace euclidien. Les méthodes de partitionnement distribuent les objets de D en k clusters, $C_1, C_2, \dots, C_k$, c'est-à-dire $C_i \subset D \; et \; C_i \cap C_j = \emptyset$ pour (1 ≤ i, j ≤ k). Une fonction objective est utilisée pour évaluer la qualité du partitionnement afin que les objets d'un cluster soient similaires les uns aux autres mais différents des objets d'autres clusters. C'est-à-dire que la fonction objective vise une forte similarité intraclasse et une faible similarité interclasse.

Une technique de partitionnement basée sur le centroïde utilise le centroïde d'un cluster C_i pour représenter ce cluster. Conceptuellement, le centroïde d'un cluster est son point central. Le centre de gravité peut être défini de différentes manières, par exemple par la moyenne ou la médoïde des objets (ou points) affectés au cluster. Le médoïde d'un cluster est le point du cluster le plus proche de tous les autres. La différence entre un objet $p \in C_i$ et c_i, le représentant du cluster, est mesurée par dist(p, c_i), où dist(x,y) est la distance euclidienne entre deux points x et y. La qualité du cluster C_i peut être mesurée par la variation intercluster, qui est la somme de l'erreur quadratique entre tous les objets de C_i et le centroïde c_i, défini comme

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, c_i)^2$$

où E est la somme de l'erreur quadratique pour tous les objets de l'ensemble de données ; p est le point de l'espace représentant un objet donné ; et c_i est le centre de gravité du cluster C_i (p et c_i sont multidimensionnels). En d'autres termes, pour chaque objet de chaque cluster, la distance entre l'objet et son centre de cluster est élevée au carré et les distances sont additionnées. Cette fonction objective essaie de rendre les k clusters résultants aussi compacts et aussi séparés que possible.

L'optimisation de la variation intra-cluster est un défi de calcul. Dans le pire des cas, il faut énumérer un certain nombre de partitionnements possibles qui sont exponentiels au nombre de

clusters et vérifier les valeurs de variation intra-cluster. Il a été montré que le problème est NP-difficile dans l'espace euclidien général même pour deux clusters. De plus, le problème est NP-difficile pour un nombre général de clusters k même dans l'espace euclidien 2-D. Si le nombre de clusters k et la dimensionnalité de l'espace d sont fixés, le problème peut être résolu en temps $O(n^{dk+1} \log n)$, où n est le nombre d'objets. Pour surmonter le coût de calcul prohibitif de la solution exacte, des approches gourmandes sont souvent utilisées dans la pratique.

L'algorithme k-means définit le centre de gravité d'un cluster comme la valeur moyenne des points au sein du cluster. Il procède comme suit :

- Tout d'abord, il sélectionne au hasard k objets dans D, dont chacun représente initialement une moyenne ou un centre de cluster.
- Pour chacun des objets restants, un objet est affecté au cluster auquel il est le plus similaire, en fonction de la distance euclidienne entre l'objet et la moyenne du cluster. L'algorithme k-means améliore ensuite itérativement la variation intra-cluster.
- Pour chaque cluster, il calcule la nouvelle moyenne à l'aide des objets affectés au cluster lors de l'itération précédente. Tous les objets sont ensuite réaffectés à l'aide des moyens mis à jour en tant que nouveaux centres de cluster.
- Les itérations se poursuivent jusqu'à ce que l'affectation soit stable, c'est-à-dire que les clusters formés au tour en cours soient les mêmes que ceux formés au tour précédent.

La méthode k-means n'est pas garantie de converger vers l'optimum global et se termine souvent à un optimum local. Les résultats peuvent dépendre de la sélection aléatoire initiale des centres de cluster. Pour obtenir de bons résultats en pratique, il est courant d'exécuter plusieurs fois l'algorithme k-means avec différents centres de cluster initiaux.

La complexité temporelle de l'algorithme des k-moyennes est O(n.k.t), où n est le nombre total d'objets, k est le nombre de clusters et t est le nombre d'itérations. Normalement, k<<n et t<<n. Par conséquent, la méthode est relativement évolutive et efficace dans le traitement de grands ensembles de données.

Il existe plusieurs variantes de la méthode k-means. Celles-ci peuvent différer dans la sélection des k-moyennes initiales, le calcul de la dissimilarité et les stratégies de calcul des moyennes des clusters.

La méthode k-means ne peut être appliquée que lorsque la moyenne d'un ensemble d'objets est définie. Cela peut ne pas être le cas dans certaines applications, par exemple lorsque des données avec des attributs nominaux sont impliquées. La méthode k-modes est une variante de k-means, qui étend le paradigme k-means pour regrouper les données nominales en remplaçant les moyennes des clusters par des modes. Il utilise de nouvelles mesures de dissemblance pour traiter les objets nominaux et une méthode basée sur la fréquence pour mettre à jour les modes des clusters. Les méthodes k-means et k-modes peuvent être intégrées pour regrouper des données avec des valeurs numériques et nominales mixtes.

La nécessité pour les utilisateurs de spécifier le nombre de clusters à l'avance peut être considérée comme un inconvénient. Cependant, des études ont été menées sur la manière de surmonter cette difficulté, par exemple en fournissant une plage approximative de valeurs de k, puis en utilisant une technique analytique pour déterminer le meilleur k en comparant les résultats de regroupement obtenus pour les différentes valeurs de k. La méthode k-means n'est pas adaptée à la découverte de clusters de formes non convexes ou de clusters de tailles très différentes. De plus, elle est sensible au bruit et aux points de données aberrants car un petit nombre de ces données peut influencer considérablement la valeur moyenne.

Une approche pour rendre la méthode k-means plus efficace sur de grands ensembles de données consiste à utiliser un ensemble d'échantillons de bonne taille en clustering. Une autre consiste à utiliser une approche de filtrage qui utilise un index de données hiérarchique spatial pour réduire les coûts lors du calcul des moyens. Une troisième approche explore l'idée de microclustering, qui regroupe d'abord les objets proches en «microclusters», puis effectue un clustering k-means sur les microclusters.

6.1.2. k-médoïdes

L'algorithme k-means est sensible aux valeurs aberrantes car ces objets sont éloignés de la majorité des données et, par conséquent, lorsqu'ils sont affectés à un cluster, ils peuvent

considérablement fausser la valeur moyenne du cluster. Cela affecte par inadvertance l'affectation d'autres objets aux clusters. Cet effet est particulièrement exacerbé en raison de l'utilisation de la fonction d'erreur au carré.

Pour diminuer une telle sensibilité aux valeurs aberrantes, au lieu de prendre la valeur moyenne des objets dans un cluster comme point de référence, on peut choisir des objets réels pour représenter les clusters, en utilisant un objet représentatif par cluster.

Chaque objet restant est affecté au cluster dont l'objet représentatif est le plus similaire. La méthode de partitionnement est alors réalisée sur le principe de la minimisation de la somme des dissemblances entre chaque objet p et son objet représentatif correspondant. Autrement dit, un critère d'erreur absolue est utilisé, défini comme

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, o_i)$$

où E est la somme de l'erreur absolue pour tous les objets p dans l'ensemble de données, et o_i est l'objet représentatif de C_i . C'est la base de la méthode des k-médoïdes, qui regroupe n objets en k clusters en minimisant l'erreur absolue.

Lorsque k = 1, on peut trouver la médiane exacte en $O(n^2)$ temps. Cependant, lorsque k est un nombre général positif, le problème k-médioïde est NP-difficile.

L'algorithme de partitionnement autour des médoïdes (PAM) est une réalisation populaire du clustering des k-médoïdes. Il aborde le problème de manière itérative et gourmande. Comme l'algorithme k-means, les objets représentatifs initiaux (appelés graines) sont choisis arbitrairement. On examine ensuite si le remplacement d'un objet représentatif par un objet non représentatif améliorerait la qualité du clustering. Tous les remplacements possibles sont essayés. Le processus itératif de remplacement d'objets représentatifs par d'autres objets se poursuit jusqu'à ce que la qualité du regroupement résultant ne puisse être améliorée par aucun remplacement.

Cette qualité est mesurée par une fonction de coût de la dissimilarité moyenne entre un objet et l'objet représentatif de son cluster.

Plus précisément, soit $o_1, o_2, \ldots, o_k$ l'ensemble actuel d'objets représentatifs (c'est-à-dire les médoïdes). Pour déterminer si un objet non représentatif, noté $o_{aléatoire}$, est un bon remplacement pour un médoïde actuel o_j 1≤j ≤k/, on calcule la distance de chaque objet p à l'objet le plus proche dans l'ensemble $\{o_1, \ldots, o_{j-1}, o_{aléatoire}, o_{j+1} \ldots, o_k\}$, et utilisez la distance pour mettre à jour la fonction de coût. Les réaffectations d'objets à l'ensemble précèdent sont simples.

Chaque fois qu'une réaffectation se produit, une différence d'erreur absolue, E, est ajoutée à la fonction de coût. Par conséquent, la fonction de coût calcule la différence de valeur d'erreur absolue si un objet représentatif actuel est remplacé par un objet non représentatif. Le coût total de l'échange est la somme des coûts encourus par tous les objets non représentatifs. Si le coût total est négatif, alors o_j est remplacé ou échangé par $o_{aléatoire}$ car l'erreur absolue réelle E est réduite. Si le coût total est positif, l'objet représentatif courant o_j est considéré comme acceptable, et rien n'est changé dans l'itération.

6.1.3. k-means vs k-médoïdes

La méthode des k-médoïdes est plus robuste que **k-means** en présence de bruit et de valeurs aberrantes, car une médoïde est moins influencée par les valeurs aberrantes ou d'autres valeurs extrêmes qu'une moyenne. Cependant, la complexité de chaque itération dans l'algorithme **k-médoïdes** est $O(k(n-k)^2$. Pour de grandes valeurs de n et k, un tel calcul devient très coûteux, et beaucoup plus coûteux que la méthode des **k-means**. Les deux méthodes exigent que l'utilisateur spécifie le nombre de clusters « k ».

6.1.4. CLARA (Clustering LARge Applications)

Un algorithme de partitionnement k-medoids typique comme PAM fonctionne efficacement pour les petits ensembles de données, mais ne s'adapte pas bien aux grands ensembles de données. Pour traiter des ensembles de données plus importants, une méthode basée sur l'échantillonnage appelée CLARA (Clustering LARge Applications) peut être utilisée. Au lieu de prendre en compte l'ensemble des données, CLARA utilise un échantillon aléatoire de l'ensemble de données. L'algorithme PAM est ensuite appliqué pour calculer les meilleurs médioïdes à partir de l'échantillon. Idéalement, l'échantillon devrait représenter étroitement l'ensemble de

données d'origine. Dans de nombreux cas, un grand échantillon fonctionne bien s'il est créé de manière à ce que chaque objet ait la même probabilité d'être sélectionné dans l'échantillon. Les objets représentatifs (médioïdes) choisis seront probablement similaires à ceux qui auraient été choisis dans l'ensemble de données. CLARA construit des regroupements à partir de plusieurs échantillons aléatoires et renvoie le meilleur regroupement en sortie. La complexité du calcul des médoïdes sur un échantillon aléatoire est $O(ks^2+ k(n−k)$, où s est la taille de l'échantillon, k est le nombre de clusters et n est le nombre total d'objets. CLARA peut traiter des ensembles de données plus volumineux que PAM.

L'efficacité de CLARA dépend de la taille de l'échantillon. Notez que PAM recherche les meilleurs k-médoïdes parmi un ensemble de données, tandis que CLARA recherche les meilleurs k-médoïdes parmi l'échantillon sélectionné de l'ensemble de données. CLARA ne peut pas trouver un bon regroupement si l'un des meilleurs médoïdes échantillonnés est loin des meilleurs k-médoïdes. Si un objet est l'un des meilleurs k-médoïdes mais n'est pas sélectionné lors de l'échantillonnage, CLARA ne trouvera jamais le meilleur regroupement.

6.1.5. CLARANS (Clustering Large Applications based upon RANdomized Search)

C’est un algorithme randomisé qui présente un compromis entre le coût et l'efficacité de l'utilisation d'échantillons pour obtenir le clustering. En effet, lors de la recherche de meilleurs médoïdes, PAM examine chaque objet de l'ensemble de données par rapport à chaque médoïde actuel, tandis que CLARA limite les médoïdes candidats à un échantillon aléatoire de l'ensemble de données.

Tout d'abord, **CLARANS** sélectionne au hasard k objets dans l'ensemble de données comme médoïdes actuels. Il sélectionne alors aléatoirement un médoïde courant x et un objet y qui n'est pas l'un des médoïdes courants. Si le remplacement de x par y améliore le critère d'erreur absolue, le remplacement est effectué. CLARANS effectue une telle recherche aléatoire l fois. L'ensemble des médoïdes courants après les l étapes est considéré comme un optimum local. CLARANS répète ce processus randomisé m fois et renvoie le meilleur optimal local comme résultat final.

6.2. Méthodes hiérarchiques

Une méthode hiérarchique crée une décomposition hiérarchique d'un ensemble d'objets. Il existe plusieurs façons orthogonales de catégoriser les méthodes de clustering hiérarchique. Par exemple, elles peuvent être classées en méthodes algorithmiques, méthodes probabilistes et méthodes bayésiennes. Les méthodes agglomératives, séparatives et multiphases sont algorithmiques, ce qui signifie qu'ils considèrent les objets de données comme déterministes et calculent les clusters en fonction des distances déterministes entre les objets. Les méthodes probabilistes utilisent des modèles probabilistes pour capturer les clusters et mesurer la qualité des clusters par l'adéquation des modèles. Les méthodes bayésiennes calculent une distribution des regroupements possibles. Autrement dit, au lieu de produire un seul clustering déterministe sur un ensemble de données, ils renvoient un groupe de structures de clustering et leurs probabilités, en fonction des données. Une méthode hiérarchique peut être classée comme étant soit agglomérative, soit divisionnaire, en fonction de la manière dont la décomposition hiérarchique est formée.

6.2.1. Hiérarchique agglomératif

Une méthode de clustering hiérarchique agglomératif utilise une stratégie ascendante. Il commence généralement en laissant chaque objet former son propre cluster et fusionne de manière itérative les clusters en clusters de plus en plus grands, jusqu'à ce que tous les objets soient dans un seul cluster ou que certaines conditions de terminaison soient satisfaites. Le cluster unique devient la racine de la hiérarchie. Pour l'étape de fusion, il trouve les deux clusters les plus proches l'un de l'autre selon une mesure de similarité et combine les deux pour former un cluster. Étant donné que deux clusters sont fusionnés par itération, où chaque cluster contient au moins un objet, une méthode agglomérative nécessite au plus n itérations.

6.2.2. Hiérarchique divisionnaire

L'approche par division, également appelée approche descendante, commence par placer tous les objets dans un cluster, qui est la racine de la hiérarchie. Il divise ensuite le cluster racine en plusieurs sous-clusters plus petits et partitionne récursivement ces clusters en plus petits. Le processus de partitionnement se poursuit jusqu'à ce que chaque cluster au niveau le plus bas soit

suffisamment cohérent, soit qu'il ne contienne qu'un seul objet, soit que les objets d'un cluster soient suffisamment similaires les uns aux autres.

Un défi avec les méthodes de division est de savoir comment partitionner un grand cluster en plusieurs plus petits. Par exemple, il existe 2^{n-1} -1 façons possibles de partitionner un ensemble de n objets en deux sous-ensembles exclusifs, où n est le nombre d'objets. Lorsque n est grand, il est prohibitif en termes de calcul d'examiner toutes les possibilités. Par conséquent, une méthode de division utilise généralement des heuristiques dans le partitionnement, ce qui peut conduire à des résultats inexacts. Dans un souci d'efficacité, les méthodes de division ne reviennent généralement pas sur les décisions de partitionnement qui ont été prises. Une fois qu'un cluster est partitionné, tout partitionnement alternatif de ce cluster ne sera plus considéré. En raison des défis posés par les méthodes de division, il existe beaucoup plus de méthodes agglomératives que de méthodes de division.

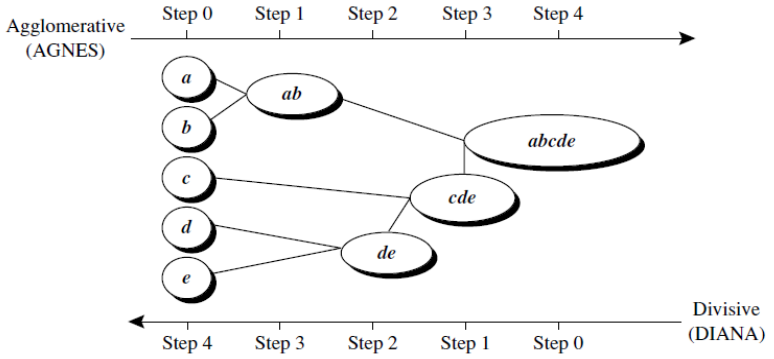

Figure 5.1 Regroupement hiérarchique agglomératif et divisionnaire[24]

6.2.3. Limites

Les méthodes hiérarchiques souffrent du fait qu'une fois qu'une étape (fusion ou division) est effectuée, elle ne peut jamais être annulée. Cette rigidité est utile en ce qu'elle conduit à des coûts de calcul plus faibles en n'ayant pas à se soucier d'un nombre combinatoire de choix différents. De telles techniques ne peuvent pas corriger des décisions erronées ; cependant, plusieurs méthodes pour améliorer la qualité du regroupement hiérarchique ont été proposées dans la littérature.

[24] http://primo.ai/index.php?title=Hierarchical_Clustering;_Agglomerative_(HAC)_%26_Divisive_(HDC)

6.2.4. Dendrogramme

Une structure arborescente appelée dendrogramme est couramment utilisée pour représenter le processus de regroupement hiérarchique. Il montre comment les objets sont regroupés (dans une méthode d'agglomération) ou partitionnés (dans une méthode de division) étape par étape. Nous pouvons également utiliser un axe vertical pour montrer l'échelle de similarité entre les clusters.

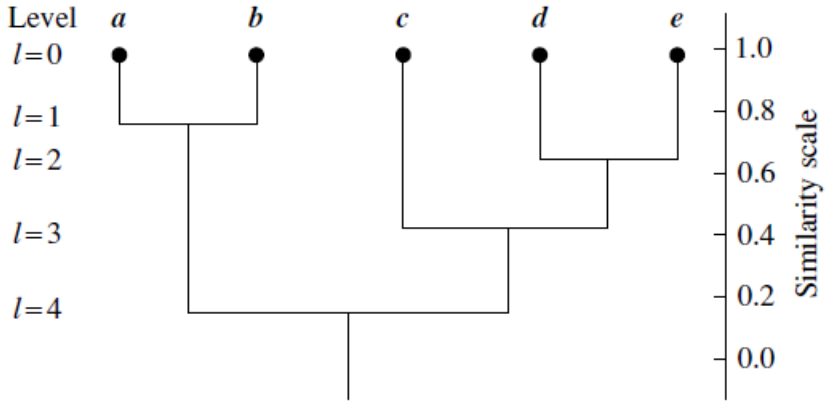

Figure 5.2 Dendrogramme pour le regroupement hiérarchique d'objets[25]

6.2.5. Mesures de distance entre clusters

Que l'on utilise une méthode d'agglomération ou une méthode de division, un besoin essentiel est de mesurer la distance entre deux clusters, où chaque cluster est généralement un ensemble d'objets.

Quatre mesures largement utilisées pour la distance entre les clusters sont les suivantes, où $|O - \acute{O}|$ est la distance entre deux objets ou points, O et $\acute{O}$; m_i est la moyenne du cluster C_i ; et n_i est le nombre d'objets dans C_i . Elles sont également connues sous le nom de mesures de liaison.

- Distance entre les centres des clusters
- Distance minimale : $dist_{min}(C_i, C_j) = min_{O \in C_i, \acute{O} \in C_j}\{|O - \acute{O}|\}$
- Distance maximale : $dist_{max}(C_i, C_j) = max_{O \in C_i, \acute{O} \in C_j}\{|O - \acute{O}|\}$
- Distance moyenne : $dist_{mean}(C_i, C_j) = |m_i - m_j|\}$

[25] Selvi, H. & Caglar, Burak. (2018). Using cluster analysis methods for multivariate mapping of traffic accidents. Open Geosciences. 10. 772-781. 10.1515/geo-2018-0060.

- Distance moyenne normalisé : $dist_{avg}(C_i, C_j) = \frac{1}{n_i.n_j}\sum_{O\in C_i, Ó\in C_j}|O - Ó|$

Lorsqu'un algorithme utilise la distance minimale $dist_{min}(C_i, C_j)$ pour mesurer la distance entre les clusters, il est parfois appelé un algorithme de clustering du plus proche voisin. De plus, si le processus de clustering est terminé lorsque la distance entre les clusters les plus proches dépasse un seuil défini, on l'appelle un algorithme à liaison unique (single-linkage algorithm). Si on considère les points de données comme des nœuds d'un graphe, avec des arêtes formant un chemin entre les nœuds d'un cluster, alors la fusion de deux clusters C_i, C_j correspond à l'ajout d'une arête entre la paire de nœuds la plus proche dans C_i et C_j . Étant donné que les arêtes reliant les clusters vont toujours entre des clusters distincts, le graphe résultant générera un arbre. Ainsi, un algorithme de clustering hiérarchique agglomératif qui utilise la mesure de distance minimale est également appelé un algorithme d'arbre couverture minimale (minimal spanning tree algorithm), où un arbre couvrant d'un graphe est un arbre qui relie tous les sommets, et un arbre couverture minimale est celui avec la plus petite somme de poids de bord.

Lorsqu'un algorithme utilise la distance maximale $dist_{max}(C_i, C_j)$ pour mesurer la distance entre les clusters, il est parfois appelé algorithme de clustering le plus éloigné (farthest-neighbor clustering algorithm). Si le processus de clustering est terminé lorsque la distance maximale entre les clusters les plus proches dépasse un seuil défini par l'utilisateur, il s'agit d'un algorithme de liaison complète (complete-linkage algorithm). En visualisant les points de données comme des nœuds d'un graphe, avec des arêtes reliant les nœuds, nous pouvons considérer chaque cluster comme un sous-graphe complet, c'est-à-dire avec des arêtes reliant tous les nœuds des clusters. La distance entre deux clusters est déterminée par les nœuds les plus éloignés des deux clusters.

Les algorithmes du plus lointain voisin ont tendance à minimiser l'augmentation du diamètre des clusters à chaque itération. Si les vrais clusters sont plutôt compacts et de taille approximativement égale, la méthode produira des clusters de haute qualité. Sinon, les clusters produits peuvent être dénués de sens.

Les mesures minimales et maximales précédentes représentent deux extrêmes dans la mesure de la distance entre les clusters. Ils ont tendance à être trop sensibles aux valeurs aberrantes ou

aux données bruyantes. L'utilisation de la distance moyenne ou moyenne normalisée est un compromis entre les distances minimale et maximale et surmonte le problème de sensibilité aux valeurs aberrantes. Alors que la distance moyenne est la plus simple à calculer, la distance moyenne normalisée est avantageuse en ce qu'elle peut gérer des données catégorielles ainsi que des données numériques. Le calcul du vecteur moyen pour les données catégoriques peut être difficile ou impossible à définir.

6.2.6. Clustering hiérarchique multiphase statique

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) est une méthode conçue pour regrouper une grande quantité de données numériques en intégrant le clustering hiérarchique et d'autres méthodes de clustering telles que le partitionnement itératif. Il surmonte les deux difficultés des méthodes de clustering agglomératif, à savoir : l'évolutivité et l'impossibilité d'annuler ce qui a été fait à l'étape précédente.

BIRCH utilise les notions de fonctionnalité de clustering pour résumer un cluster et les caractèristiques de clustering de l'arbre (CF-tree) pour représenter une hiérarchie de clusters. Ces structures aident la méthode de clustering à atteindre une bonne vitesse et une bonne évolutivité dans les bases de données volumineuses ou même en continu, et la rendent également efficace pour le clustering incrémentiel et dynamique des objets entrants.

Considérons un groupe d'objets de données ou de points de dimension n. La fonction de regroupement (CF) du cluster est un vecteur 3D résumant les informations sur les clusters d'objets. Il est défini comme

$$CF = \langle n, LS, SS \rangle$$

où LS est la somme linéaire des n points ($\sum_{i=1}^{n} x_i$), et SS est la somme des carrés des points de données ($\sum_{i=1}^{n} {x_i}^2$). Une caractéristique de clustering est essentiellement un résumé des statistiques pour le cluster donné. En utilisant une caractéristique de clustering, nous pouvons facilement dériver de nombreuses statistiques utiles d'un cluster. Par exemple, le centroïde du cluster, x_0, le rayon, R et le diamètre, D, sont

$$x_0 = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{LS}{n}$$

$$R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x_0)^2}{n}} = \sqrt{\frac{nSS - 2LS^2 + nLS}{n^2}}$$

$$D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i - x_0)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}$$

Ici, R est la distance moyenne entre les objets membres et le centroïde, et D est la distance moyenne par paire au sein d'un cluster. R et D reflètent tous deux l'étroitesse du cluster autour du centroïde.

Résumer un cluster à l'aide de la caractéristique de clustering peut éviter de stocker les informations détaillées sur les objets ou les points individuels. Au lieu de cela, on a besoin que d'une taille d'espace constante pour stocker la caractéristique de clustering. C'est la clé de l'efficacité de BIRCH dans l'espace.

De plus, les caractéristiques de regroupement sont additives. Autrement dit, pour deux clusters disjoints, C_1 et C_2, avec les caractéristiques de clustering $CF_1 = \langle n_1, LS_1, SS_1 \rangle$ et $CF_2 = \langle n_2, LS_2, SS_2 \rangle$, respectivement, la caractéristique de clustering pour le cluster formé en fusionnant C_1 et C_2 est simplement

$$CF_1 + CF_2 = \langle n_1 + n_2, LS_1 + LS_2, SS_1 + SS_2 \rangle$$

Un CF-tree est un arbre équilibré en hauteur qui stocke les caractéristiques de clustering pour un clustering hiérarchique. Par définition, un nœud non-feuille dans un arbre a des descendants ou « enfants ». Les nœuds non feuilles stockent les sommes des CF de leurs enfants et résument ainsi les informations de clustering sur leurs enfants. Un arbre CF a deux paramètres : le facteur de branchement, B, et le seuil, T. Le facteur de branchement spécifie le nombre maximum d'enfants par nœud non-feuille. Le paramètre de seuil spécifie le diamètre maximal des sous-clusters stockés aux nœuds feuilles de l'arbre. Ces deux paramètres contrôlent implicitement la taille de l'arbre résultant.

Étant donné une quantité limitée de mémoire principale, une considération importante dans BIRCH est de minimiser le temps requis pour les entrées/sorties (E/S). BIRCH applique une

technique de clustering multiphase : une seule analyse de l'ensemble de données donne un bon clustering de base, et les phases primaires sont :

- **Phase 1 :** BIRCH analyse la base de données pour créer un arbre CF initial en mémoire, qui peut être considéré comme une compression à plusieurs niveaux des données qui tente de préserver la structure de clustering inhérente aux données.
- **Phase 2 :** BIRCH applique un algorithme de clustering pour regrouper les nœuds feuilles de l'arbre CF, qui supprime les clusters clairsemés en tant que valeurs aberrantes et regroupe les clusters denses en plus grands.

Pour la phase 1, l'arborescence CF est construite dynamiquement au fur et à mesure que des objets sont insérés. Ainsi, la méthode est incrémentale. Un objet est inséré dans l'entrée feuille la plus proche (sous-cluster). Si le diamètre du sous-cluster stocké dans le nœud feuille après l'insertion est supérieur à la valeur de seuil, alors le nœud feuille et éventuellement d'autres nœuds sont divisés. Après l'insertion du nouvel objet, les informations sur l'objet sont transmises à la racine de l'arbre. La taille de l'arborescence CF peut être changée en modifiant le seuil. Si la taille de la mémoire qui est nécessaire pour stocker l'arborescence CF est supérieure à la taille de la mémoire principale, alors une valeur de seuil plus grande peut être spécifiée et l'arborescence CF est reconstruite.

Le processus de reconstruction est effectué en construisant un nouvel arbre à partir des nœuds feuilles de l'ancien arbre. Ainsi, le processus de reconstruction de l'arbre se fait sans qu'il soit nécessaire de relire tous les objets ou points. Ceci est similaire à l'insertion et à la division des nœuds dans la construction des arbres BC. Par conséquent, pour construire l'arbre, les données doivent être lues une seule fois. Certaines heuristiques et méthodes ont été introduites pour traiter les valeurs aberrantes et améliorer la qualité des arbres CF par des analyses supplémentaires des données. Une fois l'arbre CF construit, tout algorithme de clustering, tel qu'un algorithme de partitionnement typique, peut être utilisé avec l'arbre CF dans la phase 2.

La complexité temporelle de l'algorithme est O(n), où n est le nombre d'objets à regrouper. Des expérimentations ont montré la scalabilité linéaire de l'algorithme par rapport au nombre d'objets, et une bonne qualité de regroupement des données. Cependant, étant donné que

chaque nœud d'un arbre CF ne peut contenir qu'un nombre limité d'entrées en raison de sa taille, un nœud d'arbre CF ne correspond pas toujours à ce qu'un utilisateur peut considérer comme un cluster naturel. De plus, si les clusters ne sont pas de forme sphérique, BIRCH ne fonctionne pas bien car il utilise la notion de rayon ou de diamètre pour contrôler la frontière d'un cluster.

Les idées de regroupement d'entités et d'arbres CF ont été appliquées au-delà de BIRCH. Les idées ont été empruntées par beaucoup d'autres pour résoudre les problèmes de clustering de flux et de données dynamiques.

6.2.7. Clustering hiérarchique multiphase dynamique

Chameleon est un algorithme de clustering hiérarchique qui utilise la modélisation dynamique pour déterminer la similarité entre des paires de clusters. Dans Chameleon, la similarité des clusters est évaluée en fonction de:

- La qualité des objets connectés au sein d'un cluster
- La proximité des clusters.

Autrement dit, deux clusters sont fusionnés si leur interconnectivité est élevée et s'ils sont proches l'un de l'autre. Ainsi, Chameleon ne dépend pas d'un modèle statique fourni par l'utilisateur et peut s'adapter automatiquement aux caractéristiques internes des clusters fusionnés. Le processus de fusion facilite la découverte de clusters naturels et homogènes et s'applique à tous les types de données tant qu'une fonction de similarité peut être spécifiée.

Chameleon utilise une approche de graphe k-plus proche voisin pour construire un graphe creux, où chaque sommet du graphe représente un objet de données, et il existe une arête entre deux sommets (objets) si un objet est parmi les k objets les plus similaires à L'autre. Les bords sont pondérés pour refléter la similitude entre les objets. Chameleon utilise un algorithme de partitionnement de graphe pour partitionner le graphe k-plus proche voisin en un grand nombre de sous-clusters relativement petits de sorte qu'il minimise la coupe des bords. Autrement dit, un cluster C est partitionné en sous-clusters C_i et C_j de manière à minimiser le poids des arêtes qui seraient coupées si C était divisé en C_i et C_j . Il évalue l'interconnectivité absolue entre les clusters C_i et C_j.

Chameleon utilise ensuite un algorithme de clustering hiérarchique agglomératif qui fusionne de manière itérative les sous-clusters en fonction de leur similarité. Pour déterminer les paires de sous-clusters les plus similaires, il prend en compte à la fois l'interconnectivité et la proximité des clusters.

Plus précisément, Chameleon détermine la similarité entre chaque paire de clusters C_i et C_j en fonction de leur interconnectivité relative, $RI(C_i , C_j)$ et de leur proximité relative, $RC(C_i , C_j)$.

- L'interconnectivité relative $RI(C_i, C_j)$ entre deux clusters C_i et C_j est définie comme l'interconnectivité absolue entre C_i et C_j normalisée par rapport à l'interconnectivité interne des deux clusters, C_i et C_j. C'est-à-dire,

$$RI(C_i, C_j) = \frac{|EC_{\{C_i,C_j\}}|}{\frac{1}{2}\left(|EC_{C_i}| + |EC_j|\right)}$$

où $EC_{\{C_i,C_j\}}$ est la coupe d'arête telle qu'a été définie précédemment pour un cluster contenant à la fois C_i et C_j. De même, $EC_{\{C_i\}}$ est la somme minimale des arêtes coupées qui partitionnent C_i en deux parties à peu près égales.

- La proximité relative $RC(C_i, C_j)$ entre une paire de clusters C_i et C_j est la proximité absolue entre C_i et C_j normalisée par rapport à la proximité interne des deux clusters C_i et C_j . Il est défini comme

$$RC(C_i, C_j) = \frac{\bar{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i| + |C_j|}\,\bar{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i| + |C_j|}\,\bar{S}_{EC_{C_j}}}$$

où $\bar{S}_{EC_{\{C_i,C_j\}}}$ est le poids moyen des arêtes qui relient les sommets de C_i aux sommets de C_j , et $\bar{S}_{EC_{C_i}}$ est le poids moyen des arêtes qui appartiennent à la bissectrice min_cut du cluster C_i.

Il a été démontré que Chameleon a une plus grande capacité à découvrir des clusters de haute qualité de forme arbitraire que plusieurs algorithmes bien connus tels que BIRCH et DBSCAN basé sur la densité. Cependant, le coût de traitement des données de grande dimension peut nécessiter $O(n^2)$ temps pour n objets dans le pire des cas.

6.2.8. Clustering hiérarchique probabiliste

Les méthodes de regroupement hiérarchique utilisant des mesures de liaison ont tendance à être faciles à comprendre et sont souvent efficaces dans le regroupement. Ils sont couramment utilisés dans de nombreuses applications d'analyse de clustering. Cependant, les méthodes de clustering hiérarchique peuvent souffrir de plusieurs inconvénients. Premièrement, choisir une bonne mesure de distance pour le clustering hiérarchique est souvent loin d'être trivial. Deuxièmement, pour appliquer une méthode, les objets ne peuvent pas avoir de valeurs d'attribut manquantes. Dans le cas de données partiellement observées, il n'est pas facile d'appliquer une méthode de regroupement hiérarchique car le calcul de la distance ne peut pas être effectué. Troisièmement, la plupart des méthodes de regroupement hiérarchique sont heuristiques et recherchent localement à chaque étape une bonne décision de fusion/séparation. Par conséquent, l'objectif d'optimisation de la hiérarchie de cluster résultante peut ne pas être clair.

Le clustering hiérarchique probabiliste vise à surmonter certains de ces inconvénients en utilisant des modèles probabilistes pour mesurer les distances entre les clusters.

Une façon d'aborder le problème de regroupement consiste à considérer l'ensemble d'objets de données à regrouper comme un échantillon du mécanisme de génération de données sous-jacent à analyser ou, formellement, le modèle génératif.

En pratique, on peut supposer que les modèles générateurs de données adoptent des fonctions de distribution communes, telles que la distribution gaussienne ou la distribution de Bernoulli, qui sont régies par des paramètres. La tâche d'apprentissage d'un modèle génératif est alors réduite à trouver les valeurs de paramètre pour lesquelles le modèle correspond le mieux à l'ensemble de données observé.

Étant donné un ensemble d'objets, la qualité d'un cluster formé par tous les objets peut être mesurée par le maximum de vraisemblance. Pour un ensemble d'objets partitionné en m clusters $C_1, C_2, \dots, C_m$ la qualité peut être mesurée par

$$Q(\{C_1, C_2, \dots, C_m\}) = \prod_{i=1}^{m} P(C_i)$$

où P est le maximum de vraisemblance. Si on fusionne deux clusters, C_{j1} et C_{j2} , en un cluster C_F tel que $C_F = \{C_{j1} \cup C_{j2}\}$, alors, le changement de qualité du clustering global $\acute{Q}$ est

$$\acute{Q} = Q\big(\{C_1, C_2, \dots, C_m\} - \{C_{j1}, C_{j2}\} \cup C_F\big) - Q(\{C_1, C_2, \dots, C_m\})$$

$$= \frac{\prod_{i=1}^{m} P(C_i).P(C_F)}{P(C_{j1}).P(C_{j2})} - \prod_{i=1}^{m} P(C_i)$$

Lorsque vous choisissez de fusionner deux clusters dans un clustering hiérarchique,

$\prod_{i=1}^{m} P(C_i)$ est constant pour toute paire de clusters. Par conséquent, étant donné les clusters $C_1 \; et \; C_2$, la distance entre eux peut être mesurée par

$$dist(C_1, C_2) = -log \frac{P(C_1 \cup C_2)}{P(C_1).P(C_2)}$$

Une méthode de clustering hiérarchique probabiliste peut adopter le cadre de clustering agglomératif, mais utiliser des modèles probabilistes pour mesurer la distance entre les clusters.

Les méthodes de regroupement hiérarchique probabiliste sont faciles à comprendre et ont généralement la même efficacité que les méthodes de regroupement hiérarchique agglomératif. Les modèles probabilistes sont plus interprétables, mais parfois moins flexibles que les métriques de distance. Les modèles probabilistes peuvent traiter des données partiellement observées. Par exemple, étant donné un ensemble de données multidimensionnel où certains objets ont des valeurs manquantes sur certaines dimensions, nous pouvons apprendre un modèle gaussien sur chaque dimension indépendamment en utilisant les valeurs observées sur la dimension. La hiérarchie de cluster résultante atteint l'objectif d'optimisation consistant à ajuster les données aux modèles probabilistes sélectionnés.

Un inconvénient de l'utilisation du clustering hiérarchique probabiliste est qu'il ne produit qu'une seule hiérarchie par rapport à un modèle probabiliste choisi. Il ne peut pas gérer l'incertitude des hiérarchies de clusters. Étant donné un ensemble de données, il peut exister plusieurs hiérarchies qui correspondent aux données observées. Ni les approches algorithmiques ni les approches probabilistes ne peuvent trouver la distribution de telles hiérarchies. Récemment, des modèles arborescents bayésiens ont été développés pour traiter de tels problèmes.

6.3. Méthodes basées sur la densité

La plupart des méthodes de partitionnement regroupent les objets en fonction de la distance entre les objets. De telles méthodes ne peuvent trouver que des clusters de forme sphérique et rencontrent des difficultés pour découvrir des clusters de formes arbitraires. D'autres méthodes de clustering ont été développées sur la base de la notion de densité. Leur idée générale est de continuer à développer un cluster donné tant que la densité (nombre d'objets ou de points de données) dans le « voisinage » dépasse un certain seuil. Par exemple, pour chaque point de données dans un cluster donné, le voisinage d'un rayon donné doit contenir au moins un nombre minimum de points. Une telle méthode peut être utilisée pour filtrer le bruit ou les valeurs aberrantes et découvrir des clusters de forme arbitraire.

Les méthodes basées sur la densité peuvent diviser un ensemble d'objets en plusieurs clusters exclusifs ou en une hiérarchie de clusters. En règle générale, les méthodes basées sur la densité ne prennent en compte que les clusters exclusifs et ne prennent pas en compte les clusters flous. De plus, les méthodes basées sur la densité peuvent être étendues du clustering de l'espace complet au sous-espace.

6.3.1. DBSCAN

La densité d'un objet « o » peut être mesurée par le nombre d'objets proches de « o ». DBSCAN (Density-Based Spatial Clustering of Applications with Noise) trouve les objets principaux, c'est-à-dire les objets qui ont des voisinages denses. Il relie les objets centraux et leurs voisinages pour former des régions denses sous forme de clusters.

DBSCAN quantifie le voisinage d'un objet en se basant sur un paramètre spécifié par l'utilisateur $\varepsilon > 0$. Le paramètre ε est utilisé pour spécifier le rayon d'un voisinage pour chaque objet.

En raison de la taille de voisinage fixe paramétrée par ε, la densité d'un voisinage peut être mesurée simplement par le nombre d'objets dans le voisinage. Pour déterminer si un voisinage est dense ou non, DBSCAN utilise un autre paramètre spécifié par l'utilisateur, MinPts, qui spécifie le seuil de densité des régions denses. Un objet est un objet central si le ε-voisinage de l'objet contient au moins un nombre d'objets supérieure ou égale à MinPts. Les objets centraux sont les piliers des régions denses.

Étant donné un ensemble D d'objets, on peut identifier tous les objets de base par rapport aux paramètres ε et MinPts. La tâche de regroupement est alors réduite à l'utilisation d'objets centraux et de leurs voisinages pour former des régions denses, où les régions denses sont des clusters.

Pour un objet central q et un objet p, on dit que p est directement accessible en densité à partir de q (par rapport à et MinPts) si p est dans le ε-voisinage de q. Clairement, un objet p est directement accessible en densité depuis un autre objet q si et seulement si q est un objet central et p est dans le ε-voisinage de q. En utilisant la relation directement accessible à la densité, un objet central peut "amener" tous les objets de son ε-voisinage dans une région dense.

Dans DBSCAN, p est accessible en densité à partir de q (par rapport à et MinPts dans D) s'il existe une chaîne d'objets $p_1, p_2, \dots, p_n$ telle que $p_1 = q, p_n = p$, et p_{i+1} est directement accessible par densité à partir de p_i par rapport à ε et MinPts, pour $1 \leq i \leq n, p_i \in D$. Notons que l'accessibilité par densité n'est pas une relation d'équivalence car elle n'est pas symétrique. Si o_1 et o_2 sont des objets centraux et que o_1 est accessible par densité depuis o_2, alors o_2 est accessible par densité depuis o_1. Cependant, si o_2 est un objet central mais que o_1 ne l'est pas, alors o_1 peut être accessible par densité à partir de o_2, mais pas l'inverse.

Pour connecter les objets centraux ainsi que leurs voisins dans une région dense, DBSCAN utilise la notion de connectivité par densité. Deux objets $p_1, p_2 \in D$ sont connectés par densité par rapport à ε et MinPts s'il existe un objet $q \in D$ tel que $p_1 \; et \; p_2$ sont accessibles en densité à partir de q par rapport à ε et MinPts. Contrairement à l'accessibilité par densité, la connectivité par densité est une relation d'équivalence.

Nous pouvons utiliser la fermeture de la densité-connexité pour trouver des régions denses connectées sous forme de clusters. Chaque ensemble fermé est un cluster basé sur la densité. Un sous-ensemble $C \in D$ est un cluster si :

- Pour deux objets quelconques $o_1, o_2 \in C$, o_1 et o_2 sont connectés par densité
- Il n'existe pas d'objet $o_2 \in C$ et un autre objet $\acute{o} \in (D - C)$ tels que o et $\acute{o}$ soient connectés par densité.

Pour trouver les clusters, d'abord tous les objets d'un ensemble de données D donné sont marqués comme "non visités". DBSCAN sélectionne au hasard un objet p non visité, marque p comme "visité" et vérifie si le ε-voisinage de p contient au moins des objets MinPts.

Si la condition n'est pas vérifiée, p est marqué comme un point de bruit. Sinon, un nouveau cluster C est créé pour p, et tous les objets du ε-voisinage de p sont ajoutés à un ensemble candidat N. DBSCAN ajoute itérativement à C les objets de N qui n'appartiennent à aucun cluster. Dans ce processus, pour un objet $\acute{p}$ dans N qui porte l'étiquette "non visité", DBSCAN le marque comme "visité" et vérifie son ε-voisinage. Si le ε-voisinage de $\acute{p}$ a au moins des objets MinPts, ces objets dans le ε-voisinage de $\acute{p}$ sont ajoutés à N. DBSCAN continue d'ajouter des objets à C jusqu'à ce que C ne puisse plus être étendu, c'est-à-dire que N soit vide. À ce moment, le cluster C est terminé et est donc sorti.

Pour trouver le cluster suivant, DBSCAN sélectionne au hasard un objet non visité parmi ceux qui restent. Le processus de regroupement se poursuit jusqu'à ce que tous les objets soient visités.

Si un index spatial est utilisé, la complexité de calcul de DBSCAN est O.(n log n), où n est le nombre d'objets de la base de données. Sinon, la complexité est $O(n^2)$. Avec des réglages appropriés des paramètres ε et MinPts, l'algorithme est efficace pour trouver des clusters de forme arbitraire.

6.3.2. OPTICS

Bien que DBSCAN puisse regrouper des objets en fonction de paramètres d'entrée tels que le rayon maximal d'un voisinage et le nombre minimal de points requis dans le voisinage d'un objet central, il encombre les utilisateurs de la responsabilité de sélectionner les valeurs de paramètres qui conduiront à la découverte de clusters acceptables. Il s'agit d'un problème associé à de

nombreux autres algorithmes de clustering. Ces réglages de paramètres sont généralement définis de manière empirique et difficiles à déterminer, en particulier pour les ensembles de données de grande dimension du monde réel. La plupart des algorithmes sont sensibles à ces valeurs de paramètres : des paramètres légèrement différents peuvent conduire à des regroupements très différents de données. De plus, les ensembles de données de grande dimension du monde réel ont souvent des distributions très asymétriques, de sorte que leur structure de regroupement intrinsèque peut ne pas être bien caractérisée par un ensemble unique de paramètres de densité globale.

Notez que les clusters basés sur la densité sont monotones par rapport au seuil de voisinage. Autrement dit, dans DBSCAN, pour une valeur MinPts fixe et deux seuils de voisinage,

$\varepsilon_1 < \varepsilon_2$, un cluster C par rapport à ε_1 et MinPts doit être un sous-ensemble d'un cluster $\acute{C}$ par rapport à $\varepsilon_2$2 et MinPts. Cela signifie que si deux objets se trouvent dans un cluster basé sur la densité, ils doivent également se trouver dans un cluster avec une exigence de densité inférieure.

Pour surmonter la difficulté d'utiliser un ensemble de paramètres globaux dans l'analyse de clustering, une méthode d'analyse de cluster appelée OPTICS (Ordering Points to Identify the Clustering Structure) a été proposée. OPTICS ne produit pas explicitement un clustering d'ensembles de données. Au lieu de cela, il génère un ordre de cluster. Il s'agit d'une liste linéaire de tous les objets en cours d'analyse et représente la structure de regroupement basée sur la densité des données. Les objets d'un cluster plus dense sont répertoriés plus près les uns des autres dans l'ordre des clusters.

Cet ordre équivaut à un regroupement basé sur la densité obtenu à partir d'un large éventail de réglages de paramètres. Ainsi, OPTICS ne demande pas à l'utilisateur de fournir un seuil de densité spécifique. L'ordre de cluster peut être utilisé pour extraire des informations de clustering de base, dériver la structure de clustering intrinsèque, ainsi que fournir une visualisation du clustering.

Pour construire simultanément les différents clusterings, les objets sont traités dans un ordre précis. Cette commande sélectionne un objet accessible à la densité par rapport à la valeur la

plus basse afin que les clusters avec une densité plus élevée (lower) soient terminés en premier. Sur la base de cette idée, OPTICS a besoin de deux informations importantes par objet :

- La distance centrale d'un objet p : c'est la plus petite valeur $\acute{\varepsilon}$ telle que le $\acute{\varepsilon}$-voisinage de p a au moins des objets MinPts. Autrement dit, $\acute{\varepsilon}$ est le seuil de distance minimum qui fait de p un objet central. Si p n'est pas un objet central par rapport à ε et MinPts, la distance centrale de p est indéfinie.
- La distance d'accessibilité à l'objet p à partir de q : c'est la valeur de rayon minimale qui rend la densité p accessible à partir de q. Selon la définition de l'accessibilité par densité, q doit être un objet central et p doit être au voisinage de q. Par conséquent, la distance d'accessibilité de q à p est $\max\{distance\ centrale(q), dist(p,q)\}$. Si q n'est pas un objet central par rapport à ε et MinPts, la distance d'accessibilité à p de q est indéfinie.

Un objet p peut être directement accessible à partir de plusieurs objets centraux. Par conséquent, p peut avoir plusieurs distances d'accessibilité par rapport à différents objets centraux. La plus petite distance d'accessibilité de p est particulièrement intéressante car elle donne le chemin le plus court pour lequel p est connecté à un cluster dense.

OPTICS calcule un ordre de tous les objets dans une base de données et pour chaque objet de la base de données, stocke la distance centrale et une distance d'accessibilité appropriée. OPTICS maintient une liste appelée OrderSeeds pour générer l'ordre de sortie. Les objets dans Order-Seeds sont triés par la distance d'accessibilité de leurs objets principaux respectifs les plus proches, c'est-à-dire par la plus petite distance d'accessibilité de chaque objet.

OPTICS commence par un objet arbitraire de la base de données d'entrée comme objet courant p. Il récupère le ε-voisinage de p, détermine la distance centrale et met la distance d'accessibilité sur undefined. L'objet courant p est ensuite écrit dans la sortie.

Si p n'est pas un objet principal, OPTICS passe simplement à l'objet suivant dans la liste OrderSeeds (ou la base de données d'entrée si OrderSeeds est vide). Si p est un objet principal, alors pour chaque objet q dans le ε-voisinage de p, OPTICS met à jour sa distance d'accessibilité

à partir de p et insère q dans OrderSeeds si q n'a pas encore été traité. L'itération continue jusqu'à ce que l'entrée soit entièrement consommée et que OrderSeeds soit vide.

L'ordre des clusters d'un ensemble de données peut être représenté graphiquement, ce qui permet de visualiser et de comprendre la structure de clustering dans un ensemble de données grâce au diagramme d'accessibilité.

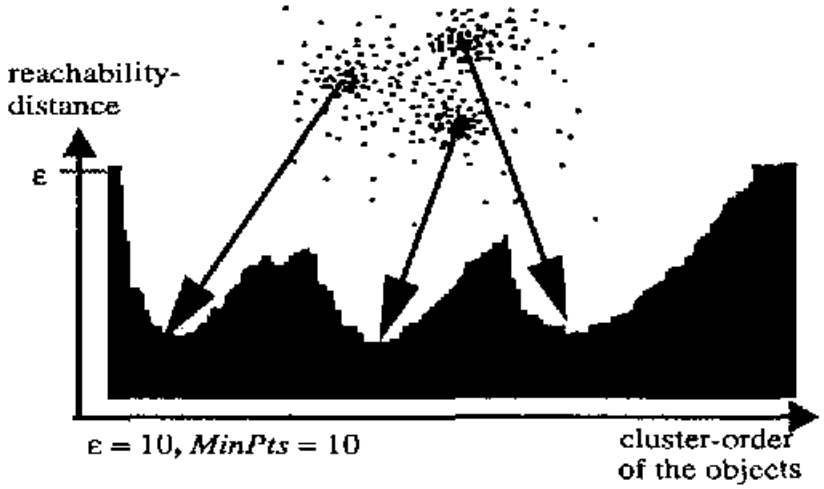

Figure 5.3 Ordre des clusters dans OPTICS [26]

Les objets de données sont tracés dans l'ordre de regroupement (axe horizontal) avec leurs distances d'accessibilité respectives (axe vertical). Les trois « bosses » gaussiennes dans le graphique reflètent trois clusters dans l'ensemble de données. Des méthodes ont également été développées pour visualiser les structures de clustering de données de grande dimension à différents niveaux de détail.

La structure de l'algorithme OPTICS est très similaire à celle de DBSCAN. Par conséquent, les deux algorithmes ont la même complexité temporelle. La complexité est O(n log n) si un index spatial est utilisé, et $O(n^2)$ sinon, où n est le nombre d'objets.

[26] https://www.semanticscholar.org/paper/OPTICS%3A-ordering-points-to-identify-the-clustering-Ankerst-Breunig/80c983b2f36e3db461e35a5e8836d4b20b485d4f

6.3.3. DENCLUE

L'estimation de la densité est un problème central dans les méthodes de regroupement basées sur la densité. DENCLUE (DENsity-based CLUstEring) est une méthode de clustering basée sur un ensemble de fonctions de distribution de densité.

En probabilité et en statistique, l'estimation de la densité est l'estimation d'une fonction de densité de probabilité sous-jacente non observable basée sur un ensemble de données observées. Dans le contexte du regroupement basé sur la densité, la fonction de densité de probabilité sous-jacente non observable est la vraie distribution de la population de tous les objets possibles à analyser. L'ensemble de données observé est considéré comme un échantillon aléatoire de cette population.

Dans DBSCAN et OPTICS, la densité est calculée en comptant le nombre d'objets dans un voisinage défini en utilisant le rayon. De telles estimations de densité peuvent être très sensibles à la valeur de rayon utilisée. Pour surmonter ce problème, l'estimation de densité par noyau peut être utilisée, qui est une approche d'estimation de densité non paramétrique à partir de statistiques. L'idée générale derrière l'estimation de la densité du noyau est simple. Nous traitons un objet observé comme un indicateur de densité à haute probabilité dans la région environnante. La densité de probabilité en un point dépend des distances de ce point aux objets observés.

Formellement, soit $x_1, x_2, \dots, x_n$un échantillon indépendant et identiquement distribué d'une variable aléatoire f . L'approximation de la densité du noyau de la fonction de densité de probabilité est

$$\widehat{f_n}(x) = \frac{1}{n.h}\sum_{i=1}^{n} K(\frac{x - x_i}{h})$$

où K est un noyau et h est la bande passante servant de paramètre de lissage. Un noyau peut être considéré comme une fonction qui modélise l'influence d'un point d'échantillonnage dans son voisinage. Techniquement, un noyau K est une fonction intégrable à valeur réelle non

négative qui doit satisfaire deux exigences : $\int_{-\infty}^{+\infty} K(u)du = 1 \; et \; K(-u) = K(u)$ pour toutes les valeurs de u. le noyau le fréquemment utilisé est une fonction gaussienne standard avec une moyenne de 0 et une variance de 1 :

$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-x_i)^2}{2.h^2}}$$

DENCLUE utilise un noyau gaussien pour estimer la densité en fonction de l'ensemble d'objets à regrouper. Un point x^*est appelé attracteur de densité s'il s'agit d'un maximum local de la fonction de densité estimée. Pour éviter les points maximaux locaux triviaux, DENCLUE utilise un seuil de bruit ε et ne considère que les attracteurs de densité x^* tels que $\hat{f}(x^*) \geq \varepsilon$. Ces attracteurs de densité non triviaux sont les centres des clusters.

Les objets analysés sont affectés à des clusters via des attracteurs de densité en utilisant une procédure d'escalade progressive. Pour un objet, x, la procédure d'escalade commence à partir de x et est guidée par le gradient de la fonction de densité estimée. Autrement dit, l'attracteur de densité pour x est calculé comme

$$x^0 = x$$

$$x^{j+1} = x^j + \delta \frac{\nabla \hat{f}(x^j)}{|\nabla \hat{f}(x^j)|}$$

où δ est un paramètre pour contrôler la vitesse de convergence, et

$$\nabla \hat{f}(x) = \frac{1}{h^{d+2}.n \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)(x - x_i)}$$

La procédure d'escalade s'arrête à l'étape k > 0 si $\hat{f}(x^{k+1}) < \hat{f}(x^k)$, et affecte x à l'attracteur de densité $x^* = x^k$. Un objet x est une valeur aberrante ou un bruit s'il converge dans la procédure d'escalade vers un maximum local x^* avec $\hat{f}(x^*) < \varepsilon$.

Un cluster dans DENCLUE est un ensemble d'attracteurs de densité X et un ensemble d'objets d'entrée C tels que chaque objet de C est affecté à un attracteur de densité dans X, et il existe un chemin entre chaque paire d'attracteurs de densité où la densité est supérieure ε. En utilisant

plusieurs attracteurs de densité connectés par des chemins, DENCLUE peut trouver des clusters de forme arbitraire. DENCLUE présente plusieurs avantages. Il peut être considéré comme une généralisation de plusieurs méthodes de clustering bien connues telles que les approches à liaison unique et DBSCAN. De plus, DENCLUE est invariant contre le bruit. L'estimation de la densité du noyau peut réduire efficacement l'influence du bruit en répartissant uniformément le bruit dans les données d'entrée.

6.4. Méthodes basées sur la grille

Les méthodes basées sur la grille quantifient l'espace objet en un nombre fini de cellules qui forment une structure de grille. Toutes les opérations de regroupement sont effectuées sur la structure de grille (c'est-à-dire sur l'espace quantifié). Le principal avantage de cette approche est son temps de traitement rapide, qui est généralement indépendant du nombre d'objets de données et dépend uniquement du nombre de cellules dans chaque dimension de l'espace quantifié.

L'utilisation de grilles est souvent une approche efficace pour de nombreux problèmes d'exploration de données spatiales, y compris le clustering. Par conséquent, les méthodes basées sur la grille peuvent être intégrées à d'autres méthodes de regroupement telles que les méthodes basées sur la densité et les méthodes hiérarchiques.

6.4.1. STING

STING (STatistical INformation Grid) est une technique de clustering multirésolution basée sur une grille dans laquelle la zone spatiale d'intégration des objets d'entrée est divisée en cellules rectangulaires. L'espace peut être divisé de manière hiérarchique et récursive. Plusieurs niveaux de telles cellules rectangulaires correspondent à différents niveaux de résolution et forment une structure hiérarchique : Chaque cellule à un niveau haut est partitionnée pour former un certain nombre de cellules au niveau immédiatement inférieur. Des informations statistiques concernant les attributs dans chaque cellule de grille, telles que les valeurs moyennes, maximales et minimales, sont précalculées et stockées en tant que paramètres statistiques. Ces paramètres statistiques sont utiles pour le traitement des requêtes et pour d'autres tâches d'analyse de données.

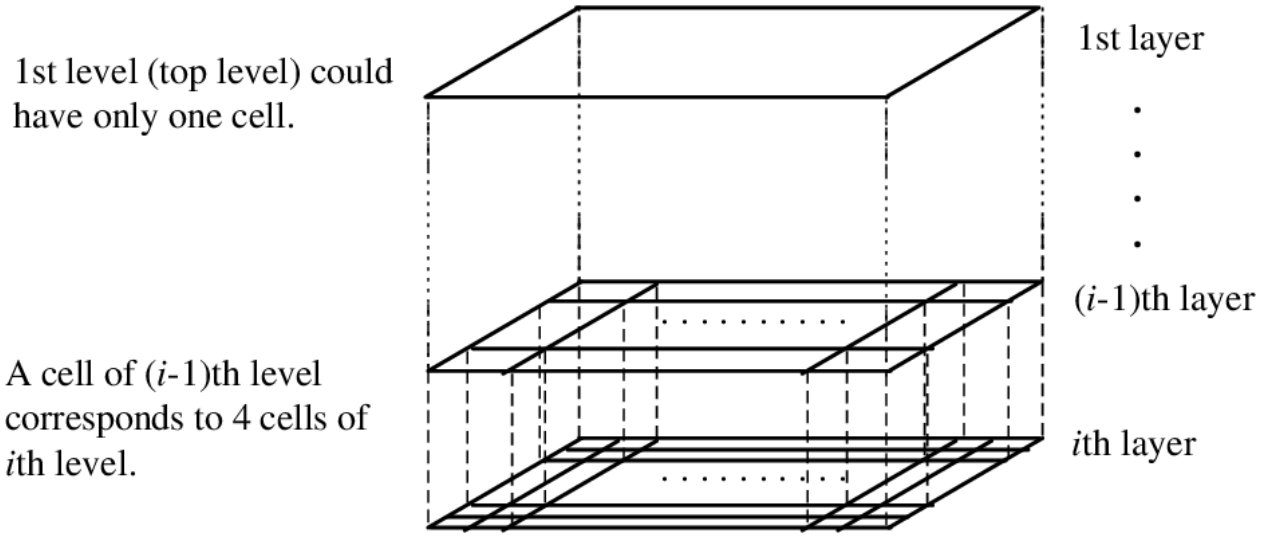

Figure 5.4 Structure hiérarchique par la méthode STING[27]

Les paramètres statistiques des cellules de niveau supérieur peuvent facilement être calculés à partir des paramètres des cellules de niveau inférieur. Ces paramètres comprennent les éléments suivants : le paramètre indépendant de l'attribut, count ; et les paramètres dépendants de l'attribut, moyenne, stdev (écart type), min (minimum), max (maximum) et le type de distribution de l'attribut suivant dans la cellule, comme normal, uniforme, exponentiel ou aucun (si la répartition est inconnue). Lorsque les données sont chargées dans la base de données, les paramètres count, mean, stdev, min et max des cellules de niveau inférieur sont calculés directement. La valeur de la distribution peut être soit attribuée par l'utilisateur si le type de distribution est connu à l'avance, soit obtenue par des tests d'hypothèses tels que le test X^2. Le type de distribution d'une cellule de niveau supérieur peut être calculé sur la base de la majorité des types de distribution de ses cellules de niveau inférieur correspondantes conjointement avec un processus de filtrage de seuil. Si les distributions des cellules de niveau inférieur ne concordent pas et échouent au test de seuil, le type de distribution de la cellule de niveau supérieur est défini sur aucun.

[27] https://www.semanticscholar.org/paper/STING%3A-A-Statistical-Information-Grid-Approach-to-Wang-Yang/cf304776b89bbfeb109fdc2bccc6ab8d1da144cf

Les paramètres statistiques peuvent être utilisés de manière descendante, basée sur une grille, comme suit. Tout d'abord, une couche à l'intérieur de la structure hiérarchique est déterminée à partir de laquelle le processus de requête-réponse doit commencer.

Cette couche contient généralement un petit nombre de cellules. Pour chaque cellule de la couche actuelle, on calcule l'intervalle de confiance reflétant la pertinence de la cellule par rapport à la requête donnée. Les cellules non pertinentes sont supprimées de toute considération ultérieure.

Le traitement du niveau inférieur suivant n'examine que les cellules pertinentes restantes. Ce processus est répété jusqu'à ce que la couche inférieure soit atteinte. À ce stade, si la spécification de la requête est satisfaite, les régions des cellules pertinentes qui satisfont la requête sont renvoyées. Sinon, les données qui tombent dans les cellules pertinentes sont récupérées et traitées jusqu'à ce qu'elles répondent aux exigences de la requête.

Une propriété intéressante de STING est qu'il se rapproche du résultat de regroupement de DBSCAN si la granularité se rapproche de 0 (c'est-à-dire vers des données de très bas niveau). En d'autres termes, en utilisant les informations sur le nombre et la taille des cellules, les clusters denses peuvent être identifiées approximativement à l'aide de STING. Par conséquent, STING peut également être considéré comme une méthode de regroupement basée sur la densité.

STING offre plusieurs avantages :

- Le calcul basé sur la grille est indépendant de la requête car les informations statistiques stockées dans chaque cellule représentent les informations récapitulatives des données dans la cellule de la grille, indépendamment de la requête ;
- La structure en grille facilite le traitement parallèle et la mise à jour incrémentielle ;
- L'efficacité de la méthode est un avantage majeur :

STING parcourt la base de données une fois pour calculer les paramètres statistiques des cellules, et donc la complexité temporelle de la génération des clusters est O(n), où n est le nombre total d'objets. Après génération de la structure hiérarchique, le temps de traitement de la requête est

O(g) où g est le nombre total de cellules de la grille au niveau le plus bas, qui est généralement beaucoup plus petit que n.

Étant donné que STING utilise une approche multirésolution pour l'analyse de clusters, la qualité du clustering STING dépend de la granularité du niveau le plus bas de la structure de la grille. Si la granularité est très fine, le coût de traitement augmentera sensiblement ; cependant, si le niveau inférieur de la structure de la grille est trop grossier, cela peut réduire la qualité de l'analyse de cluster. De plus, STING ne prend pas en compte la relation spatiale entre les enfants et leurs cellules voisines pour la construction d'une cellule parent. En conséquence, les formes des clusters résultants sont isothétiques, c'est-à-dire que toutes les limites des clusters sont horizontales ou verticales, et aucune limite diagonale n'est détectée. Cela peut réduire la qualité et la précision des clusters malgré le temps de traitement rapide de la technique.

6.4.2. CLIQUE

Un objet de données a souvent des dizaines d'attributs, dont beaucoup peuvent ne pas être pertinents. Les valeurs des attributs peuvent varier considérablement. Ces facteurs peuvent compliquer la localisation des clusters couvrant l'ensemble de l'espace de données. Il peut être plus significatif de rechercher des clusters dans différents sous-espaces des données.

CLIQUE (CLustering In QUEst) est une méthode simple qui utilise une grille pour trouver des clusters basés sur la densité dans des sous-espaces. CLIQUE partitionne chaque dimension en intervalles sans chevauchement, partitionnant ainsi tout l'espace d'intégration des objets de données en cellules.

Il utilise un seuil de densité pour identifier les cellules denses et clairsemées. Une cellule est dense si le nombre d'objets qui lui sont mappés dépasse le seuil de densité.

La principale stratégie derrière CLIQUE pour identifier un espace de recherche candidat utilise la monotonie des cellules denses par rapport à la dimensionnalité. Ceci est basé sur la propriété Apriori utilisée dans la recherche fréquente de modèles et de règles d'association. Dans le contexte des clusters dans les sous-espaces, la monotonie propose ce qui suit. Une cellule « c » k-dimensionnelle (k > 1) peut avoir au moins l points seulement si chaque (k-1) projection dimensionnelle de c est une cellule dans un sous-espace (k-1) dimensionnel, a au moins l points.

CLIQUE effectue le clustering en deux étapes :

- **Etape 1 :** CLIQUE partitionne l'espace de données d-dimensionnel en unités rectangulaires non superposées, identifiant les unités denses parmi celles-ci. CLIQUE trouve des cellules denses dans tous les sous-espaces. Pour ce faire, CLIQUE partitionne chaque dimension en intervalles et identifie les intervalles contenant au moins l points, où l est le seuil de densité. CLIQUE joint alors itérativement deux mailles denses k-dimensionnelles c_1 et c_2 dans des sous-espaces $(D_{i1}, D_{i2}, \dots, D_{ik})$ et $(D_{j1}, D_{j2}, \dots, D_{jk})$, respectivement, si $D_{i1} = D_{j1}, D_{i2} = D_{j2}, \dots, D_{ik-1} = D_{jk-1}$, et c_1 et c_2 partagent les mêmes intervalles dans ces dimensions. L'opération de jointure génère une nouvelle cellule candidate c (k+1)-dimensionnelle dans l'espace $(D_{i1}, D_{i2}, \dots, D_{ik-1}, D_{jk})$. CLIQUE vérifie si le nombre de points dans c dépasse le seuil de densité. L'itération se termine lorsqu'aucun candidat ne peut être généré ou qu'aucune cellule candidate n'est dense.
- **Etape 2 :** CLIQUE utilise les cellules denses de chaque sous-espace pour assembler des clusters, qui peuvent être de forme arbitraire. L'idée est d'appliquer la longueur de description minimale (MDL) pour utiliser les régions maximales pour couvrir les cellules denses connectées, où une région maximale est un hyperrectangle où chaque cellule tombant dans cette région est dense, et la région ne peut pas être étendue davantage dans aucune dimension du sous-espace.

Trouver la meilleure description d'un cluster en général est NP-Hard. Ainsi, CLIQUE adopte une démarche gourmande simple. Il commence par une cellule dense arbitraire, trouve une région maximale couvrant la cellule, puis travaille sur les cellules denses restantes qui n'ont pas encore été couvertes. La méthode se termine lorsque toutes les cellules denses sont couvertes.

CLIQUE trouve automatiquement les sous-espaces de dimensionnalité la plus élevée de sorte que des clusters à haute densité existent dans ces sous-espaces. Il est insensible à l'ordre des objets d'entrée et ne présume aucune distribution canonique des données. Il évolue de manière linéaire avec la taille de l'entrée et présente une bonne évolutivité lorsque le nombre de dimensions dans les données augmente. Cependant, l'obtention d'un regroupement significatif dépend du réglage approprié de la taille de la grille et du seuil de densité. Cela peut être difficile en pratique car la

taille de la grille et le seuil de densité sont utilisés dans toutes les combinaisons de dimensions de l'ensemble de données. Ainsi, la précision des résultats de regroupement peut être dégradée au détriment de la simplicité de la méthode. De plus, pour une région dense, toutes les projections de la région sur des sous-espaces de dimensionnalité inférieure seront également denses. Cela peut entraîner un chevauchement important entre les régions denses signalées. Outre, il est difficile de trouver des clusters de densités assez différentes dans des sous-espaces dimensionnels différents.

Plusieurs extensions de cette approche suivent une philosophie similaire. Par exemple, on peut considérer une grille comme un ensemble de bacs fixes. Au lieu d'utiliser des bacs fixes pour chacune des dimensions, nous pouvons utiliser une stratégie adaptative basée sur les données pour déterminer dynamiquement les bacs pour chaque dimension en fonction des statistiques de distribution des données. Alternativement, au lieu d'utiliser un seuil de densité, nous pouvons utiliser l'entropie comme mesure de la qualité des clusters du sous-espace.

Bibliographie

- C.C. Aggarwal. Data Mining: The Textbook. Springer International Publishing, 2015.
- P. Bhatia. Data Mining and Data Warehousing: Principles and Practical Techniques. Cambridge University Press, 2019.
- M.J.A. Berry and G.S. Linoff. Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. Wiley, 2004.
- M. Bramer. Principles of Data Mining. Undergraduate Topics in Computer Science. Springer London, 2007.
- M.S. Brown. Data Mining For Dummies. –For dummies. Wiley, 2014.
- D.M. Dziuda. Data Mining for Genomics and Proteomics: Analysis of Gene and Protein Expression Data. Wiley Series on Methods and Applications in Data Mining. Wiley, 2010.
- J. Han, J. Pei, and M. Kamber. Data Mining, Southeast Asia Edition. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2006.
- K. Jamsa. Introduction to Data Mining and Analytics. Jones & Bartlett Learning, 2020.
- Kononenko and M. Kukar. Machine Learning and Data Mining. Elsevier Science, 2007.
- R. Layton. Learning Data Mining with Python. Packt Publishing, 2017.
- B. Liu. Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Data-Centric Systems and Applications. Springer Berlin Heidelberg, 2007.
- Maheshwari. Business Intelligence and Data Mining. Big data and business analytics collection. Business Expert Press, 2014.
- D.L. Olson and D. Delen. Advanced Data Mining Techniques. Springer Berlin Heidelberg, 2008.
- F. Provost and T. Fawcett. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking. O'Reilly Media, 2013.
- Ratner. Statistical and Machine-Learning Data Mining: Techniques for Better Predictive Modeling and Analysis of Big Data, Third Edition. CRC Press, 2017.
- Rajaraman and J.D. Ullman. Mining of Massive Datasets. Cambridge University Press, 2012.
- G. Shmueli, P.C. Bruce, and N.R. Patel. Data Mining for Business Analytics: Concepts, Techniques, and Applications with XLMiner. Wiley, 2016.
- G. Shmueli, P.C. Bruce, I. Yahav, N.R. Patel, and K.C. Lichtendahl. Data Mining for Business Analytics: Concepts, Techniques, and Applications in R. Wiley, 2017.
- [TF11] L. Torgo and Taylor & Francis. Data Mining with R: Learning with Case Studies. Chapman & Hall/CRC data mining and knowledge discovery series. Chapman and Hall/CRC, 2011.
- P.N. Tan, M. Steinbach, and V. Kumar. Introduction to Data Mining. Pearson India, 2016.
- P.N. Tan, M. Steinbach, V. Kumar, and A. Karpatne. Introduction to Data Mining eBook: Global Edition. Pearson Education, 2019.
- I.H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques, Second Edition. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2005.
- I.H. Witten, E. Frank, F.E. coaut, C.D.D. ed, and J. Gray. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Database Management Systems Series. Elsevier Science, 2000.

- I.H. Witten, E. Frank, M.A. Hall, and C. Pal. Data Mining: Practical Machine Learning Tools and Techniques. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2016.
- N. Ye. The Handbook of Data Mining. Human factors and ergonomics. CRC Press, 2003.
- Y. Zhao. R and Data Mining: Examples and Case Studies. Elsevier Science, 2012.
- M.J. Zaki and W. Meira. Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, 2014.
- M.J. Zaki and W. Meira. Data Mining and Machine Learning: Fundamental Concepts and Algorithms. Cambridge University Press, 2020.
- C. Zong, R. Xia, and J. Zhang. Text Data Mining. Springer Singapore, 2021.