جامعة مولاي إسماعيل
UNIVERSITÉ MOULAY ISMAÏL

كلية العلوم والتقنيات
FACULTÉ DES SCIENCES ET TECHNIQUES

Master : SIDI-TA

MODULE :DATA MINING

PROF : MOHAMMED SABIRI

Analyse de Sentiment et Clustering dans les avis D'utilisateurs : Une Approche Data Mining

Réalisé par : FATHI SALMA

Année universitaire :2023/2024

Sommaire

Introduction :

Le Data Mining, également appelé fouille de données, désigne le processus d'exploration et d'analyse de grandes quantités de données à la recherche de motifs, de tendances, d'associations ou de structures significatives.

L'objectif est d'extraire des connaissances précieuses, souvent cachées, pour prendre des décisions éclairées, identifier des relations complexes, ou prédire des comportements futurs. Cette discipline fait appel à des techniques statistiques, d'apprentissage automatique et de traitement de l'information pour transformer des données brutes en informations exploitables.

À l'ère du numérique, où chaque clic, commentaire et évaluation sont enregistrés, l'explosion des données d'avis d'utilisateurs offre une mine d'informations précieuses pour les entreprises. L'union entre l'Analyse de Sentiment et le Clustering, sous l'égide du Data Mining, représente une approche stratégique pour extraire des connaissances exploitables de ce riche gisement de données.

Analyse de Sentiment :

L'analyse de sentiment, également appelée **mining d'opinions** ou **analyse d'opinions**, est une technique de traitement du langage naturel (NLP) qui vise à déterminer et extraire les sentiments, les émotions ou les opinions exprimés dans un texte. Cela peut inclure des documents, des critiques, des commentaires sur les médias sociaux, des articles de blog, etc.

Les principales tâches de l'analyse de sentiment comprennent :

1. **Classification des sentiments :** Attribuer une étiquette ou une catégorie (par exemple, positif, négatif, neutre) à un texte en fonction du ton ou de l'émotion exprimée.
2. **Extraction d'entités :** Identifier les entités (mots, expressions) spécifiques pour lesquelles les sentiments sont exprimés.
3. **Classification d'émotions :** Identifier des émotions spécifiques telles que la joie, la colère, la tristesse, etc.
4. **Évaluation de la polarité :** Déterminer le degré de positivité ou de négativité du texte.

L'analyse de sentiment est utilisée dans divers domaines, y compris le marketing pour évaluer la réception des produits, la veille sociale pour suivre l'opinion publique, et même dans le domaine financier pour évaluer le sentiment du marché. Elle repose souvent sur des techniques de machine Learning, telles que les modèles de classification, pour automatiser le processus d'analyse des textes à grande échelle.

Le clustering K-means

Le clustering K-means est un algorithme de regroupement (clustering) largement utilisé en apprentissage automatique et en analyse de données. L'objectif principal du clustering est de diviser un ensemble de données en groupes homogènes ou clusters, où les éléments d'un même groupe sont plus similaires entre eux qu'avec ceux des autres groupes. L'algorithme K-means est particulièrement populaire en raison de sa simplicité et de son efficacité.

Voici comment fonctionne l'algorithme K-means :

1. **Initialisation** : Sélectionnez k centres de cluster initiaux de manière aléatoire dans l'espace des données, où k est le nombre prédéfini de clusters.
2. **Attribution** : Attribuez chaque point de données au cluster dont le centre est le plus proche en termes de distance euclidienne.
3. **Mise à jour du centre** : Recalculer les centres des clusters en prenant la moyenne de tous les points appartenant à chaque cluster.
4. **Répétez** : Répétez les étapes 2 et 3 jusqu'à ce que les centres des clusters ne changent plus de manière significative ou qu'un nombre prédéfini d'itérations soit atteint.

L'algorithme converge généralement vers une solution où les points de données sont regroupés de manière optimale en k clusters. Cependant, la qualité des clusters dépend fortement du choix initial des centres et du nombre k.

Le K-means est largement utilisé dans des domaines tels que la segmentation de marché, l'analyse d'image, la compression d'image, la biologie computationnelle, et d'autres applications où la division de données en groupes similaires est nécessaire.

Haut du formulaire

Application

L'application "Analyse de Sentiment et Clustering (Data Mining)" offre une approche complète pour explorer les avis d'utilisateurs à partir de fichiers CSV. Cette application, développée avec l'interface graphique Tkinter, combine l'analyse de sentiment à l'aide de TextBlob et le clustering avec l'algorithme K-means.

Description de l'Application :

L'application permet aux utilisateurs de charger un fichier CSV contenant des avis d'utilisateurs. Elle effectue ensuite une analyse de sentiment sur ces avis en utilisant la bibliothèque TextBlob, attribuant un score de polarité à chaque avis. En parallèle, elle applique l'algorithme K-means pour regrouper les avis en clusters, permettant une exploration des similitudes entre les avis.

Utilisations Potentielles :

- **Exploration d'Avis Produit/Service :** Pour comprendre les sentiments des utilisateurs et découvrir des groupes d'opinions similaires.
- **Catégorisation d'Avis :** Pour classifier automatiquement les avis en fonction de leur polarité et de leur regroupement.

Objectifs :

Cette application a pour objectif d'effectuer une analyse de sentiment et un clustering sur les données textuelles d'un fichier CSV. Voici ses objectifs principaux :

1. **Analyse de Sentiment :**
    - L'application utilise la bibliothèque TextBlob pour effectuer une analyse de sentiment sur la colonne "Review" du DataFrame extrait du fichier CSV.
    - La polarité du sentiment est calculée et stockée dans une nouvelle colonne "Sentiment" du DataFrame.
    - Une moyenne du sentiment est calculée, et des informations textuelles sont générées en fonction de cette moyenne.
2. **Clustering avec K-means :**
    - Utilisation de la bibliothèque scikit-learn pour effectuer le clustering K-means sur les données textuelles.
    - La représentation TF-IDF des avis textuels est utilisée comme fonctionnalités d'entrée pour le clustering.
    - Les dimensions sont réduites avec TruncatedSVD avant l'application de l'algorithme K-means.
    - Les résultats du clustering sont stockés dans une nouvelle colonne "Cluster" du DataFrame.
3. **Affichage des Résultats :**
    - Les résultats de l'analyse de sentiment et du clustering sont affichés dans la fenêtre principale de l'application.
    - Les informations de la base de données, la moyenne du sentiment, et les résultats du clustering sont inclus dans le rapport PDF généré.

4.Affichage des Informations en Temps Réel :

- Les informations sur le chargement du fichier, l'analyse de sentiment, et les résultats du clustering sont affichées en temps réel dans la fenêtre de l'application

Outils :

Cette application est développée ***en python*** dans

Visual studio code :

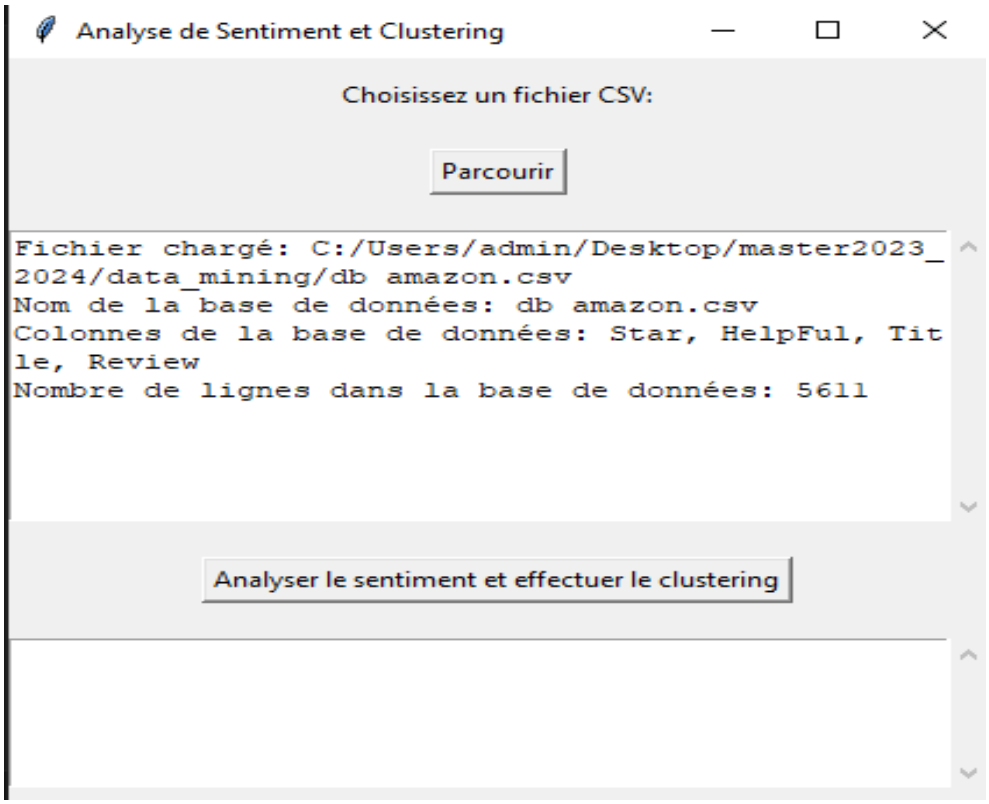
Éditeur de code source autonome qui s'exécute sur Windows, MacOs et Linux. Le meilleur choix pour JavaScript et les développeurs web, avec des extensions pour prendre en charge à peu près n'importe quel langage de programmation

.

Anaconda Navigator est l'interface de navigation d'Anaconda, elle permet de lancer les différentes API disponibles et de gérer les différents packages et environnements du logiciel. Cette interface permet de naviguer simplement dans le logiciel, sans devoir connaître toutes les lignes de code de Conda

Interface :

#fenêtre principale : **Chargement de Fichier CSV**



Rapport d'analyse de sentiment

Moyenne du sentiment: 0.38

Description de la base de données:

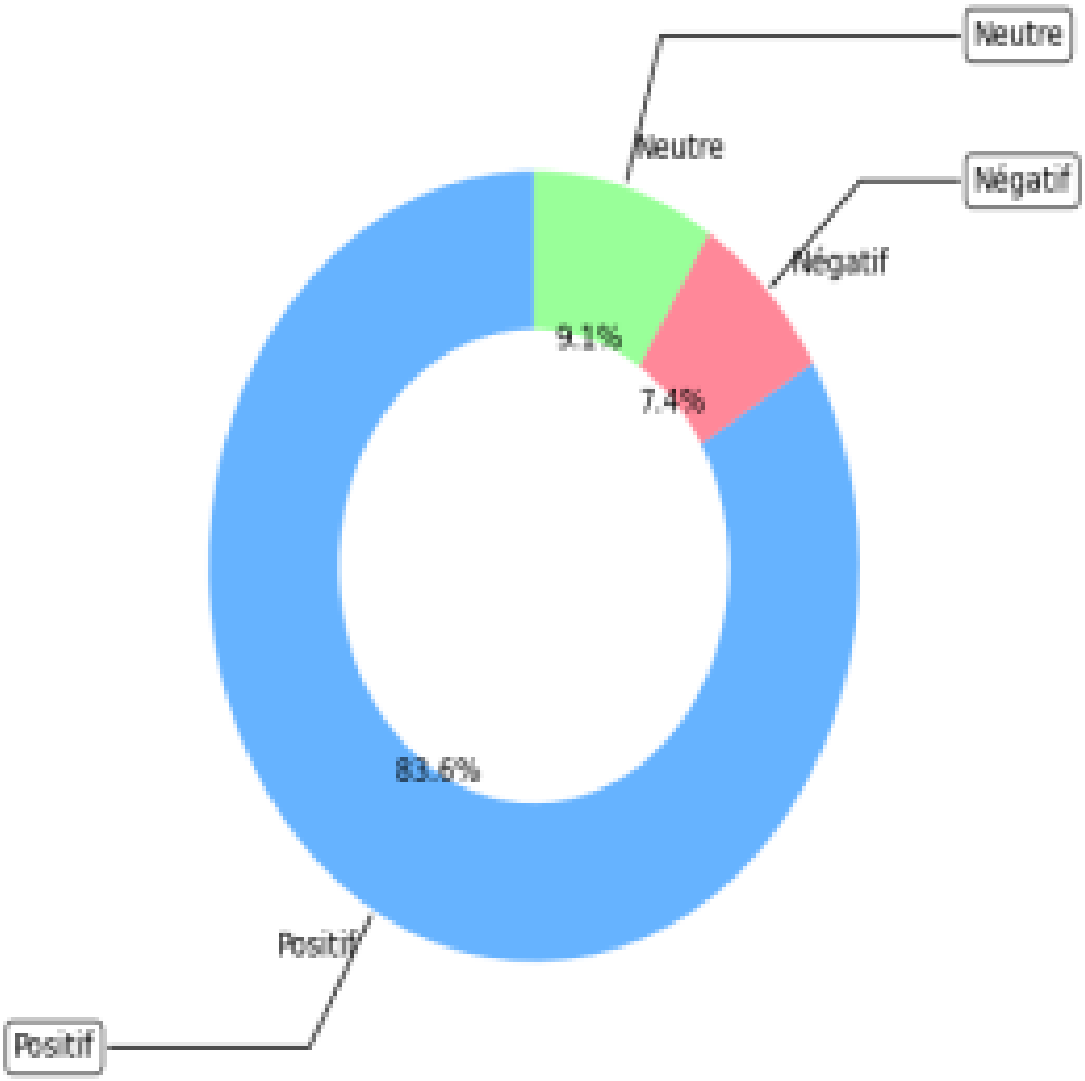
Nom de la base de données: db amazon.csv

Colonnes de la base de données: Star, HelpFul, Title, Review, Sentiment

Nombre de lignes dans la base de données: 5611

Analyse: Les avis indiquent généralement un sentiment positif envers le produit.

Répartition des sentiments (Diagramme circulaire) :

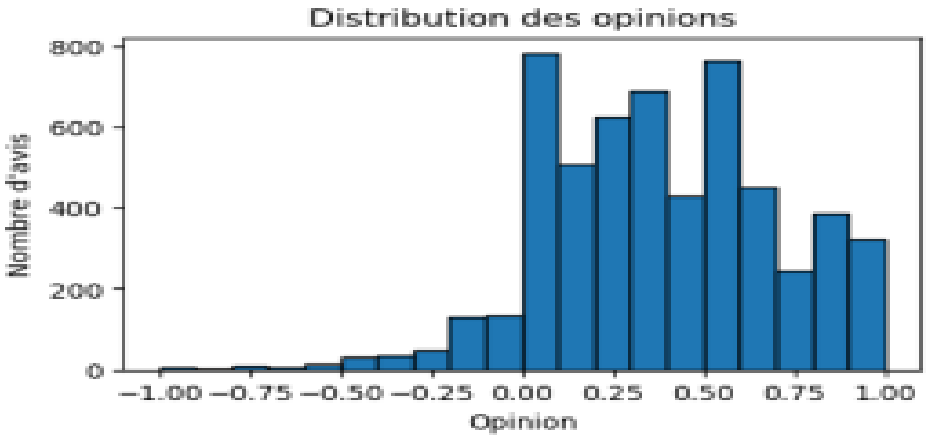


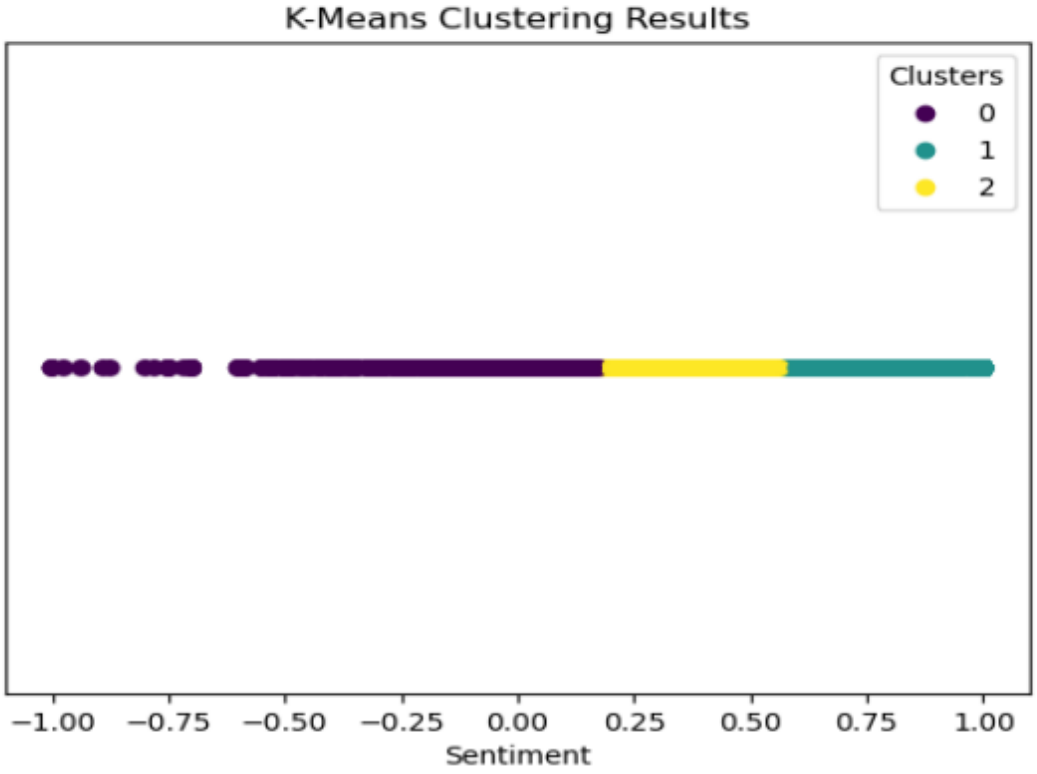
Nombre d'avis par type:

Type d'avis	Nombre
Positif	4690
Négatif	413

Neutre	508

Distribution des opinions:





Pour une explication détaillée du graphique du clustering k-means, veuillez vous référer au fichier texte : C:\Users\admin\Desktop\master2023_2024\data_mining\description_k_means.txt

Cluster	Nombre de points de données
0	1510
1	1571
2	2530

Fonctionnalités Principales :

1. **Chargement de Fichier CSV :** Les utilisateurs peuvent sélectionner un fichier CSV contenant les avis à analyser et à regrouper.
2. **Analyse de Sentiment :** L'application utilise TextBlob pour évaluer la polarité des avis, indiquant s'ils sont positifs, négatifs ou neutres.
3. **Clustering avec K-means :** Les avis sont regroupés en clusters à l'aide de l'algorithme K-means, permettant d'identifier des tendances ou des similarités entre les avis.
4. **Affichage des Résultats :** Les résultats de l'analyse de sentiment et du clustering sont affichés de manière claire, mettant en évidence les clusters formés et présentant des exemples d'avis au sein de chaque cluster.

Voici quelques captures de Code source

//chargement de base de données

```
def load_csv(self):
    file_path = filedialog.askopenfilename(filetypes=[("CSV files", "*.csv")])
    if file_path:
        self.df = pd.read_csv(file_path)
        self.df.name = os.path.basename(file_path)  # Utilisez le nom du fichier comme nom de base de données
        self.text.insert(tk.END, f"Fichier chargé: {file_path}\n")
        self.display_database_info()  # Affichez les informations de la base de données
        self.generate_pdf_report(0, "")  #générer le rapport PDF avec les informations de la base de données
```

/ /analyse de sentiment

```
def analyze_sentiment(self):
    if hasattr(self, 'df'):
        self.df['Sentiment'] = self.df['Review'].apply(lambda x: TextBlob(str(x)).sentiment.polarity)
        self.text.insert(tk.END, "Analyse de sentiment terminée.\n")

        average_sentiment = self.df['Sentiment'].mean()
        sentiment_result = "Moyenne du sentiment: {:.2f}".format(average_sentiment)
        self.result_text.delete(1.0, tk.END)
        self.result_text.insert(tk.END, sentiment_result)
        # CREER LE DIAGRAMME
        opinions_chart_path = os.path.join("C:\\Users\\admin\\Desktop\\master2023_2024\\data_mining", "opinions_chart.png")
        self.figure_opinions, self.ax_opinions = plt.subplots(figsize=(5, 3), dpi=100)
        self.ax_opinions.hist(self.df['Sentiment'], bins=20, edgecolor='black')
        self.ax_opinions.set_title('Distribution des opinions')
        self.ax_opinions.set_xlabel('Opinion')
        self.ax_opinions.set_ylabel("Nombre d'avis")
        self.figure_opinions.savefig(opinions_chart_path, bbox_inches='tight')
        plt.close()
        self.text.insert(tk.END, f"Graphe des opinions généré: {opinions_chart_path}\n")
        # Texte analytique
        self.result_text.insert(tk.END, "\n\nAnalyse:\n")
        if average_sentiment > 0.2:
            self.result_text.insert(tk.END, "Les avis indiquent généralement un sentiment positif envers le produit/le service.\n")
        elif average_sentiment < -0.2:
            self.result_text.insert(tk.END, "Les avis indiquent généralement un sentiment négatif envers le produit/le service.\n")
            # Appel de la nouvelle méthode pour inclure les informations dans le rapport PDF
            pdf_path = os.path.join("C:\\Users\\admin\\Desktop\\master2023_2024\\data_mining", "rapport_sentiment.pdf")
            self.generate_pdf_report(average_sentiment, pdf_path)
        else:
            self.result_text.insert(tk.END, "Les avis sont plutôt neutres envers le produit.\n")

        # Générer le rapport PDF
        pdf_path = os.path.join("C:\\Users\\admin\\Desktop\\master2023_2024\\data_mining", "rapport_sentiment.pdf")
```

```
def generate_pdf_report(self, average_sentiment, pdf_path):
    # Vérifier si le répertoire existe, sinon le créer
    directory = os.path.dirname(pdf_path)
    if not directory:
        print("Error: The directory path is not specified.")
    else:
        try:
    # Attempt to create the directory
            os.makedirs(directory)
            print(f"Directory '{directory}' created successfully.")
        except FileExistsError:
            print(f"Directory '{directory}' already exists.")
        except Exception as e:
            print(f"Error creating directory '{directory}': {e}")
    doc = SimpleDocTemplate(pdf_path, pagesize=letter)
    styles = getSampleStyleSheet()

    # Ajout de la moyenne du sentiment au rapport
    content = [Paragraph("Rapport d'analyse de sentiment", styles['Title']),
               Paragraph("Moyenne du sentiment: {:.2f}".format(average_sentiment), styles['BodyText'])]

    # Ajout du texte analytique au rapport
    analysis_text = "Analyse:\n"
    # Ajout des informations de la base de données au rapport PDF
    content.append(Paragraph("Description de la base de données:", styles['BodyText']))
    if hasattr(self, 'df'):
        content.append(Paragraph(f"Nom de la base de données: {self.df.name}", styles['BodyText']))
        content.append(Paragraph(f"Colonnes de la base de données: {', '.join(self.df.columns)}", styles['BodyText']))
    else:
        content.append(Paragraph("Aucune base de données chargée", styles['BodyText']))
    content.append(Paragraph(f"Nombre de lignes dans la base de données: {len(self.df)}", styles['BodyText']))
```

Conclusion :

Cette application fournit un moyen efficace et convivial d'explorer et d'analyser les sentiments exprimés dans un ensemble de données textuelles, tout en offrant des fonctionnalités avancées telles que le clustering pour une compréhension plus approfondie des tendances au sein des avis.

Perspectives :

1. **Optimisation des Paramètres du Clustering :**
    - Permettre à l'utilisateur de spécifier le nombre de clusters lors de l'application de l'algorithme K-means pour une personnalisation accrue.
    - Explorer d'autres algorithmes de clustering pour comparer les performances et les résultats.
2. **Traitement de Données Manquantes :**
    - Mettre en place une stratégie de gestion des données manquantes plus élaborée, comme l'imputation ou l'exclusion contrôlée, pour améliorer la robustesse de l'application.