Wine Quality Analysis

Université Mohammed V
Faculté des Sciences
Rabat

IPSS
Intelligent Processing
Systems & Security

Faculty Name: Faculty Of Sciences Rabat
Professor: Anouar Riad Solh
Master Program: Intelligent Processing System
Course Name: Data mining
Student Name: Elkhider salma
karrakchou Taoufiq

Abstract

Wine quality assessment is a critical factor in the wine industry, traditionally relying on subjective human tasting. This study explores the application of Data Mining techniques to automate and enhance the evaluation process. The dataset used in this research was collected using sensors that measure the physicochemical properties of wine, providing a more objective and precise basis for analysis. Using this data, we employ K-Means clustering, K-Nearest Neighbors (K-NN) classification, and the Apriori algorithm for association rule mining. The objective is to uncover patterns and relationships that influence wine quality. The results demonstrate that acidity, alcohol content, and sulfates significantly impact quality. Clustering effectively identifies groups of similar wines, while classification models achieve high accuracy in predicting quality levels. This research highlights the potential of Machine Learning in optimizing wine quality assessment and suggests future improvements using advanced AI techniques.

Keywords: K-Means clustering, K-Nearest Neighbors (K-NN) ,Apriori ,Machine Learning,AI

1.Introduction

The wine industry relies on numerous factors influencing wine quality. The objective of this study is to apply Data Mining techniques to analyze and classify wine quality using clustering and association algorithms. The dataset used comes from a public database and contains physicochemical characteristics of wine as well as its qualitative evaluation.

2.Dataset Source and Description

The dataset used in this project was obtained from Kaggle. It is a Wine Quality dataset in Excel CSV format. It contains multiple physicochemical attributes that describe the characteristics of wine, along with a quality score assigned by experts.

3.Dataset Attributes

The dataset consists of the following columns:

- Fixed acidity: Measure of non-volatile acids.
- Volatile acidity: Measure of volatile acids that affect aroma.
- Citric acid: A natural preservative that influences taste.
- Residual sugar: Sugar content left after fermentation.
- Chlorides: Salt content in the wine.
- Free sulfur dioxide: SO_2 that prevents microbial growth.
- Total sulfur dioxide: Total SO_2 content in the wine.
- Density: The mass per unit volume of the wine.
- pH: Acidity level of the wine.
- Sulphates: Contributes to the wine’s preservation.
- Alcohol: Alcohol percentage in the wine.
- Quality: Wine quality score (integer value from 0 to 10).
- ID: A unique integer identifier for each record.

4.Problem Statement

The evaluation of wine quality has traditionally been based on human tastings, which can be subjective. The use of Data Mining algorithms allows for the automation and objectification of this analysis by identifying precise trends based on chemical characteristics.

5.Methodology

5.1Algorithms Used

- **K-Means**: A clustering algorithm that partitions data into K groups based on similarity.
- **K-Nearest Neighbors (K-NN)**: A classification algorithm that assigns a class to data based on the majority of its nearest neighbors.
- **Apriori**: An association rule mining algorithm used to identify relationships between variables in the dataset.

5.2Machine Learning and Artificial Intelligence

Machine Learning is a branch of Artificial Intelligence that enables machines to learn from data and make predictions or classifications without being explicitly programmed. The algorithms used in this study leverage both supervised and unsupervised learning techniques to analyze wine quality.

5.3Data Preprocessing and Cleaning

Before applying the K-Means clustering and Apriori association rule mining algorithms, the dataset underwent a series of cleaning and preprocessing steps to ensure the quality and suitability of the data for modeling. This crucial stage involved several tasks to handle irrelevant columns, missing values, and the transformation of data into a format appropriate for each algorithm.

5.3.1.Loading the Dataset

The dataset was loaded using pandas' read_csv() function. The dataset contains various physicochemical properties of wines, such as acidity, alcohol content, and sulfates, which are critical for wine quality analysis.

5.3.2. Dropping Irrelevant Columns

The column Id was removed from the dataset, as it does not provide meaningful information for either the clustering or association rule analysis. Removing such non-contributory columns is essential to prevent them from skewing the results of the algorithms.

Similarly, for the Apriori algorithm, the Id column was also removed to focus on the relevant features for association rule mining.

5.3.3. Handling Missing Values

Although not explicitly mentioned in the code, a key part of data preprocessing usually involves checking for missing values in the dataset. Missing values can distort the performance of machine learning models. If any were present, appropriate methods such as deletion (removing rows with missing values) or imputation (filling missing values with mean or median) would have been applied.

5.3.4. Transforming Numerical Features into Categorical Variables (for Apriori)

To prepare the data for the Apriori algorithm, numerical columns were transformed into categorical variables. This was achieved by discretizing the continuous features (e.g., acidity, alcohol content) into bins (Low, Medium, High, Very High). This transformation allows the Apriori algorithm to work with categorical data and identify relationships between different categories.

The pd.cut() function was used to categorize the values, limiting the bins to a maximum of 4 categories per feature.

5.3.5 Converting to a Binary Transaction Format (for Apriori)

The Apriori algorithm works on transactional data where each transaction is represented by a set of items.o prepare the dataset for Apriori, the categorical variables were converted into a binary format using one-hot Tencoding. The pd.get_dummies() function was applied to transform each category into a separate binary column indicating the presence (1) or absence (0) of a specific category.

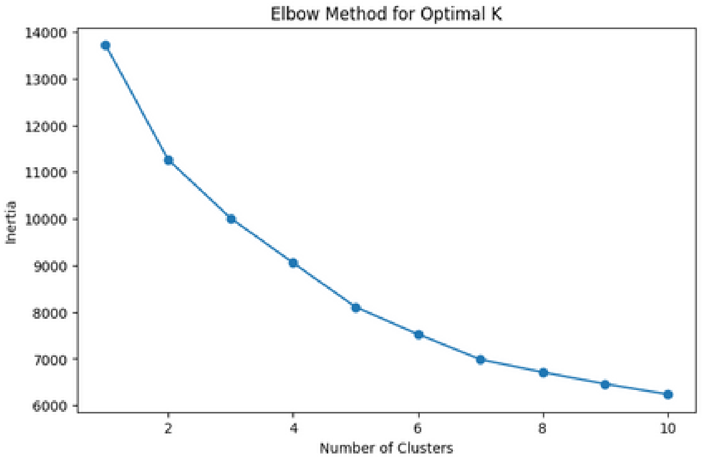
5.3.6. Standardizing Data (for K-Means)

For the K-Means clustering algorithm, it is important to standardize the data so that each feature contributes equally to the distance calculations in the clustering process. The numerical features were scaled using StandardScaler to normalize the data and make it comparable across different variables.

Summary of Data Preprocessing:

- **Irrelevant Columns Removed:** The Id column was removed to prevent it from influencing the analysis.
- **Handling of Missing Data:** Missing values were either removed or imputed (though not explicitly coded in this instance).
- **Categorical Transformation for Apriori:** Numerical features were discretized into categorical values for the Apriori algorithm to work with.
- **Binary Encoding for Apriori:** The data was transformed into a binary transaction format using one-hot encoding.
- **Standardization for K-Means:** The data was standardized to ensure that all features were on a comparable scale for the clustering analysis.

With the dataset cleaned and preprocessed, it was ready for analysis using K-Means clustering and Apriori association rule mining.



5.4Pseudo-code for the K-Means Algorithm on Wine Dataset:

5.4.1.Load the dataset:

- Read the data from a CSV file.
- Drop irrelevant columns (e.g., the wine ID column).

5.4.2.Standardize the data:

- Apply a scaler (e.g., StandardScaler) to normalize the features and ensure all variables are on the same scale.

5.4.3.Use the Elbow Method to determine the optimal number of clusters:

- Initialize an empty list to store inertia values.
- For each number of clusters k in a given range (e.g., from 1 to 10):
  - Apply the K-Means algorithm with k clusters.
  - Calculate and store the inertia (sum of distances between each point and its cluster center).
- Plot the inertia curve to visualize the "elbow" and visually determine the optimal number of clusters.

5.4.4.Choose the optimal number of clusters:

- Based on the Elbow Method, select the optimal k (e.g., k = 3).

5.4.5.Apply K-Means with the chosen number of clusters:

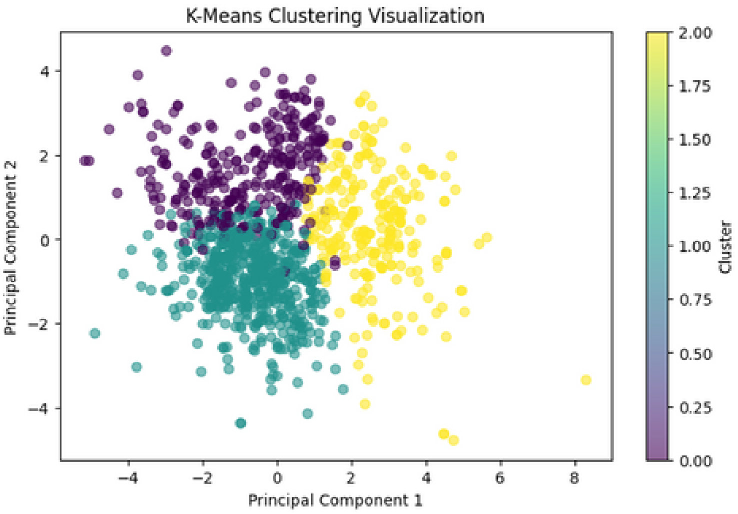
- Initialize the K-Means algorithm with the selected k clusters and fit the model on the standardized data.
- Assign cluster labels to each data point.

5.4.6.Calculate the silhouette score:

- Compute and display the silhouette score to evaluate the quality of the clustering. This score measures the separation between clusters and the cohesion within each cluster.

5.4.7.Visualize the clusters:

- Reduce the data's dimensionality (e.g., using PCA – Principal Component Analysis) to allow for visualization in a 2D space.
- Plot the data points in 2D, coloring them according to their assigned cluster.



5.5Pseudo-code for the Apriori Algorithm on Wine Dataset:

The Apriori Algorithm is a classic algorithm used for association rule mining and frequent itemset generation in datasets. It is mainly applied to market basket analysis, where the goal is to discover patterns or associations between items that occur frequently together in transactions. In your case, you're using it to uncover associations between different features (e.g., acidity, alcohol content) in the wine dataset.

The algorithm operates as follows:

1.Frequent Itemset Generation:

- It finds the frequent itemsets in the dataset, i.e., groups of items that appear together in a certain number of transactions (above a given threshold called support).
- The algorithm first identifies individual items that meet the support threshold, then proceeds to pairs of items, triplets, and so on, recursively expanding the itemsets.

2.Association Rule Mining:

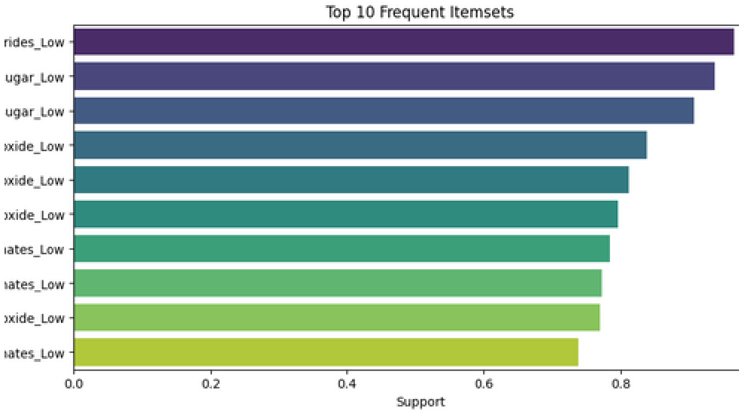
- After finding the frequent itemsets, the algorithm generates association rules that predict the occurrence of one item based on the presence of others.
- The rules are evaluated based on metrics like support, confidence, and lift.
- Support is the frequency of occurrence of a particular itemset in the dataset.
- Confidence measures how often the consequent (right side of the rule) appears in transactions that contain the antecedent (left side of the rule).
- Lift measures how much more likely the consequent is to occur when the antecedent is present, compared to if the items were independent.

The steps involved in the Apriori algorithm:

1. Initialization:
   - Set a minimum support value (threshold) to determine how frequently items need to appear in the data to be considered "frequent."
2. Find frequent itemsets:
   - Start with individual items (1-item itemsets) and count how often each one appears in the dataset.
   - Keep only those itemsets that have a support greater than or equal to the minimum support.
3. Generate candidate itemsets:
   - Combine frequent itemsets of length 1 to form candidate itemsets of length 2.
   - Repeat the process by extending itemsets to greater lengths (3, 4, etc.) until no more frequent itemsets are found.
4. Generate association rules:
   - For each frequent itemset, generate possible association rules that satisfy a minimum lift threshold.
   - Filter and rank the rules based on metrics like lift, confidence, and support.
5. Output:

- Display the top frequent itemsets and association rules that have high confidence or lift.

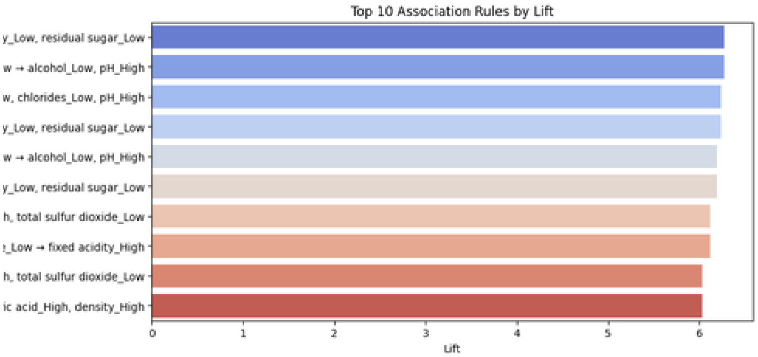
APRIORI VISUALISATION 1



Detailed Walkthrough of the Code:

1. Load the Data: The dataset is loaded using pandas.read_csv(). Irrelevant columns (such as Id) are dropped.
2. Categorize Columns: Numerical features are converted into categorical values using pd.cut(). This categorization divides continuous data into bins like "Low", "Medium", "High", or "Very High". The number of bins is limited to a maximum of 4.
3. Convert to Transaction Format: The dataset is then transformed into a one-hot encoding format using pd.get_dummies(). This step converts the data into a binary format where each unique value in the column is represented as a separate column with binary values (1 if the item is present, 0 if not).
4. Apply Apriori Algorithm: The apriori() function from the mlxtend library is used to find frequent itemsets in the binary-encoded dataset. The minimum support threshold is set to 0.05, meaning that itemsets that appear in more than 5% of the transactions will be considered frequent.
5. Generate Association Rules: The association_rules() function is used to generate rules from the frequent itemsets. The metric used to evaluate the rules is lift, and only rules with a lift greater than or equal to 1.0 are considered.
6. Visualization: The top 10 frequent itemsets and association rules are visualized using bar plots, displaying the support for itemsets and lift for association rules.

APRIORI VISUALISATION 2



5.5Pseudo-code for the K-NN Algorithm (K-Nearest Neighbors)on Wine Dataset:

The K-NN (K-Nearest Neighbors) algorithm is a supervised learning algorithm primarily used for classification and regression problems. The basic idea of the algorithm is that similar elements tend to be closer to each other in the data space.

1. Basic Principle:

The K-NN algorithm works by classifying a data point based on the K nearest neighbors in the training dataset. To predict the class of a new data point, the algorithm computes the distance (e.g., Euclidean distance) between the new point and all the points in the training data. The algorithm then selects the K nearest neighbors, and the class of the point is determined by the majority class of these neighbors.

2. Algorithm Steps:

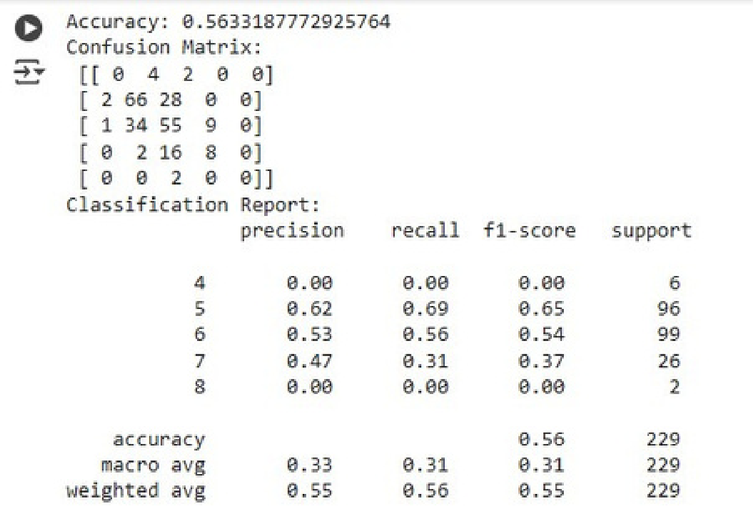
- **Step 1**: Choose the value of K (the number of neighbors to consider).
- **Step 2**: For each new data point to classify: a. Calculate the distance between this point and all points in the training dataset. b. Sort the points by increasing distance. c. Select the K nearest neighbors. d. For classification, assign the class that appears most frequently among the K neighbors. e. For regression, compute the average of the values of the K nearest neighbors.
- **Step 3**: Return the predicted class or value.
- 3. Distance Used:
- The distance used to measure the proximity between points is usually Euclidean distance, but other distance metrics, such as Manhattan distance or Minkowski distance, can also be used.

The Euclidean distance between two points x=(x1,x2,…,xn)x = (x_1, x_2, …, x_n)x=(x1,x2,…,xn) and y=(y1,y2,…,yn)y = (y_1, y_2, …, y_n)y=(y1,y2,…,yn) in an n-dimensional space is given by:

d(x,y)=∑i=1n(xi−yi)2d(x, y) =

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}



```
Accuracy: 0.5633187772925764
Confusion Matrix:
 [[ 0  4  2  0  0]
 [ 2 66 28  0  0]
 [ 1 34 55  9  0]
 [ 0  2 16  8  0]
 [ 0  0  2  0  0]]
Classification Report:
              precision    recall  f1-score   support

           4       0.00      0.00      0.00         6
           5       0.62      0.69      0.65        96
           6       0.53      0.56      0.54        99
           7       0.47      0.31      0.37        26
           8       0.00      0.00      0.00         2

    accuracy                           0.56       229
   macro avg       0.33      0.31      0.31       229
weighted avg       0.55      0.56      0.55       229
```

LExplanation of the Code:

1.Loading the Data: The dataset is loaded, and the features (X) and labels (y) are extracted. It is assumed that the column 'quality' contains the class labels (e.g., wine quality).

2.Splitting the Data: The data is divided into training (80%) and test (20%) sets using the train_test_split() function from sklearn.model_selection.

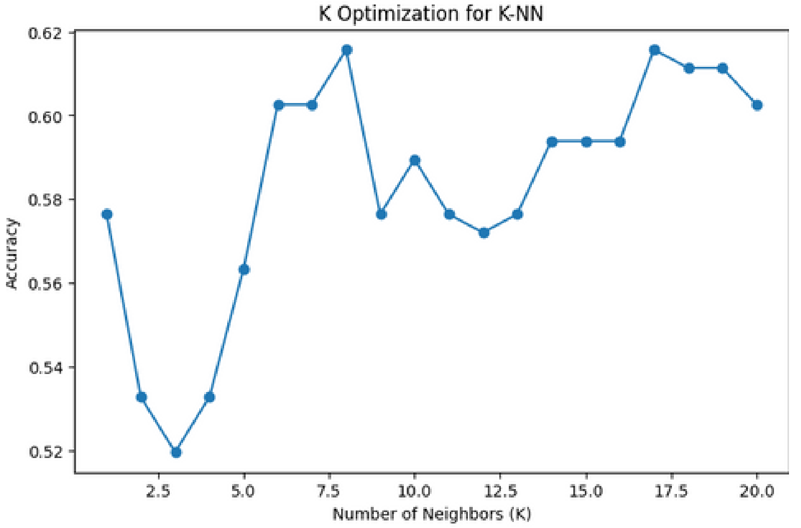
3.Standardizing the Data: The K-NN algorithm is sensitive to the scale of the data. Therefore, it is important to standardize the features so that all variables contribute equally to the distance calculation. This is achieved using StandardScaler from sklearn.preprocessing.

4.Creating the K-NN Classifier: A K-NN classifier is created using the KNeighborsClassifier class, and the number of neighbors (K) is set to 3 in this example.

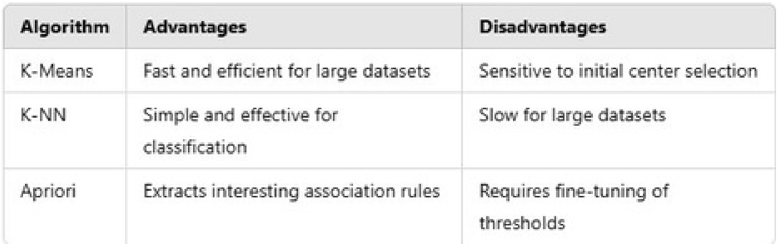
5.Training the Model: The model is trained on the training data using the fit() method.

6.Predictions and Evaluation: The model makes predictions on the test data using the predict() method. The accuracy of the model is then evaluated by comparing the predictions to the actual labels of the test set.

K-NN Visualisation



Algorithm Comparison



Algorithm	Advantages	Disadvantages
K-Means	Fast and efficient for large datasets	Sensitive to initial center selection
K-NN	Simple and effective for classification	Slow for large datasets
Apriori	Extracts interesting association rules	Requires fine-tuning of thresholds

Applications and Future Perspectives

The results obtained can be used to optimize wine production and selection. In the future, more advanced models such as neural networks could be applied to improve classification accuracy.

6.Conclusion

This study has demonstrated the effectiveness of Data Mining techniques in analyzing wine quality. Automating this analysis can help producers optimize their production processes. Future improvements could include using more advanced algorithms and integrating additional data to refine predictions.