**DOCUMENTATION**

**Data Loading and Initial Exploration**

* **Purpose:**
* The code begins by importing necessary libraries and loading the dataset train.csv into a panda DataFrame.
* Then, it displays the first few rows and provides information about the dataset to understand its structure and identify any initial issues.
* **Explanation:**
* pd. read\_csv('train.csv') loads the dataset into a DataFrame named Beer\_Data.
* . head() displays the first few rows of the DataFrame to get an overview of the data.
* .info() provides information about the dataset, including the data types and number of non-null values in each column.

**Data Cleaning and Preprocessing**

* **Purpose:**
* This section aims to clean the data and prepare it for further analysis and modeling.
* **Explanation:**
* Columns like 'user/profileName', 'review/timeStruct', and 'beer/beerId' are dropped as they are identifiers and not relevant for analysis.
* Columns are renamed for better readability using .rename() method.
* Gender values are analyzed using .value\_counts() to understand the distribution.
* The relationship between gender and alcohol content is visualized using a bar plot.

**Gender Analysis by Beer Name**

* **Purpose:**
* Analyze the distribution of gender reviews for each beer.
* Fill missing gender values in the dataset based on predominant gender for each beer.
* **Explanation:**
* Data is split into subsets based on gender.
* Each subset is grouped by BeerName to count the occurrences of each beer.
* The counts of female and male reviews for each beer are merged into a single DataFrame.
* Missing gender values in the original dataset are filled based on the predominant gender for each beer.
* A function compare\_columns() is defined to determine the dominant gender for each BeerName.
* The function is applied to create a new column 'gender' in the merged DataFrame indicating the dominant gender.
* Missing gender values in the original dataset are filled based on BeerName.

**Gender Imputation**

* **Purpose:**
* Fill missing gender values in the dataset with a default value ('Male').
* **Explanation:**
* This part of the code iterates over each row of the DataFrame.
* If the gender value is missing (NaN), it is replaced with 'Male'.

**Gender Analysis Visualization**

* **Purpose:**
* Visualize the distribution of genders in the dataset.
* **Explanation:**
* sns.countplot() is used to create a count plot of gender distribution in the dataset.
* value\_counts() is used to display the count of each gender category.

**Dropping Unnecessary Columns**

* **Purpose:**
* Remove columns that are not relevant for analysis or modeling.
* **Explanation**:
* 66666666Columns like 'index', 'user/ageInSeconds', 'user/birthdayRaw', 'user/birthdayUnix', 'review/timeUnix' are dropped using .drop() method.

**Sentiment Analysis**

* **Purpose:**
* Assign sentiment-based ratings to beer reviews.
* **Explanation:**
* Sentiment analysis is performed using VADER (Valence Aware Dictionary and sEntiment Reasoner).
* The sentiment score is converted into a rating scale ranging from 1 to 5 based on predefined thresholds.

**Dummy Formation**

* **Purpose:**
* Create dummy variables for the gender column.
* **Explanation:**
* Dummy variables are created using pd.get\_dummies() to represent the gender categories as binary indicators.
* These dummy variables are concatenated with the original DataFrame using pd.concat().

**Dropping Textual and Non-relevant Columns**

* **Purpose:**
* Remove columns that are either textual or not useful for modeling.
* **Explanation:**
* Columns like 'BeerName', 'review/text', 'gender', 'beer/style' are dropped using .drop() method.

**Categorical Rating Assignment**

* **Purpose:**
* Convert the numerical overall ratings into categorical ratings.
* **Explanation:**
* A function assign\_rating() is defined to map numerical ratings to categorical labels.
* The function is applied to create a new column 'Overall\_rating' containing categorical ratings.

**1) Decision Tree Classification**

* **Purpose:**
* Train a Decision Tree classifier to predict the overall ratings.
* **Explanation:**
* DecisionTreeClassifier is initialized and trained on the training data.
* Predictions are made on the testing data and accuracy is evaluated using accuracy\_score().
* Classification report and confusion matrix are generated to assess model performance.

**2) Logistic Regression Classification**

* **Purpose:**
* Train a Logistic Regression classifier to predict the overall ratings.
* **Explanation:**
* LogisticRegression is imported from scikit-learn.
* The dataset is split into features (X) and target (Y).
* Then, it is further split into training and testing sets using train\_test\_split().
* Logistic Regression model is initialized and trained on the training data.
* Predictions are made on the testing data and accuracy is evaluated using accuracy\_score().
* Classification report is generated to provide a detailed summary of model performance including precision, recall, F1-score, and support for each class.

**3) Random Forest Classification**

* **Purpose:**
* Train a Random Forest classifier to predict the overall ratings.
* **Explanation:**
* RandomForestClassifier is imported from scikit-learn.
* A RandomForestClassifier object is initialized with specified parameters such as the number of estimators (trees) and random state for reproducibility.
* The classifier is trained on the training data using fit() method.
* Predictions are made on the testing data using predict() method.
* Accuracy is calculated to evaluate the model's performance using accuracy\_score().
* Classification report is generated to provide a detailed summary of model performance including precision, recall, F1-score, and support for each class.

**REGRESSION TECHNIQUE**

1. **Decision Tree Regression**

* **Purpose:**
* Train a Decision Tree regressor to predict the overall ratings.
* **Explanation:**
* The dataset is split into features (X) and target (y).
* Then, it is further split into training and testing sets using train\_test\_split().
* DecisionTreeRegressor is imported from scikit-learn.
* A DecisionTreeRegressor object is initialized with a specified random state for reproducibility.
* The regressor is trained on the training data using fit() method.

1. **Linear Regression Training and Evaluation**

* **Purpose:**
* Train a Linear Regression model to predict the overall ratings.
* Evaluate the performance of the trained model.
* **Explanation:**
* The dataset is split into features (X) and target (y).
* Then, it is further split into training and testing sets using train\_test\_split().
* LinearRegression is imported from scikit-learn.
* A LinearRegression object is initialized.
* The model is trained on the training data using fit() method.
* Predictions are made on the testing data using predict() method.
* Mean Squared Error (MSE) and R-squared (R^2) scores are calculated to evaluate the model's performance.

1. **Random Forest Regression Training and Evaluation**

* **Purpose:**
* Train a Random Forest Regression model to predict overall ratings.
* Evaluate the model's performance.
* **Explanation:**
* The dataset is split into features (X) and target (y) using train\_test\_split().
* A RandomForestRegressor is initialized with 100 estimators and a random state of 42.
* The model is trained on the training data using fit() method.
* Predictions are made on the testing data using predict() method.
* Mean Squared Error (MSE) and R-squared (R^2) scores are calculated.
* Residual analysis is performed by plotting residuals against predicted values to assess model assumptions.

**FEATURE SELECTIVE TRAINING**

For increasing accuracy we again trained the model by dropping certain columns (the columns kept were based on Correlation Matrix Analysis), the remaining features were used for training the model include alcohol content, appearance, aroma, palate, taste, and gender. These features were selected to improve the accuracy of the model's predictions.