Data Pre-processing and Cleaning of Red Wine Quality Dataset

Data Science II

COSC 4337

Submitted to

DR. Ricardo Vilalta

Submitted by

Dylan Flores (1872416)

Introduction

For this assignment I have chosen to use the wine quality dataset, provided by the UCI Machine Learning Repository, which includes two separate datasets, a white wine sample and a red wine sample dataset. This dataset provides an opportunity for individuals to grasp just how in-depth the world of wine is. The data includes physicochemical and sensory data from samples of red and white variants of the Portuguese "Vinho Verde" wine. In this instance I have decided to utilize the red wine sample firstly because I like red wine and secondly, they both contained a good selection of examples and features that would be an acceptable choice. Inside the datasets CSV file there is a total of 1600 examples to analyze and 13 different features that includes acidity, sugar content, alcohol level, and more, alongside a quality rating assigned by wine experts.

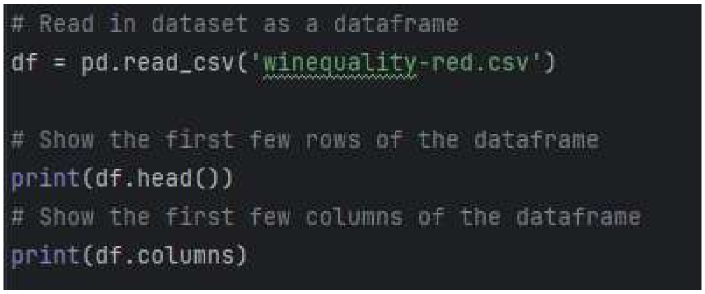
Data Description

- Data source link - https://archive.ics.uci.edu/dataset/186/wine+quality
- Column and data type –
    - **Fixed acidity** - concentration of non-volatile acids in the wine, such as tartaric, malic, citric, and succinic acids, which contribute to the wine's acidity (float).
    - **Volatile acidity** - Measures the amount of acetic acid in the wine (float).
    - **Citric acid** – Concentration of Citric acid in the wine (float).
    - Residual sugar - amount of sugar remaining after fermentation has stopped, either because it was not wholly converted to alcohol or was added post-fermentation (float).
    - **Chlorides** - The amount of salt in the wine (float).
    - **Free sulfur dioxide** - The portion of SO2 that is not bound to other molecules and is free in the wine (int).
    - **Total sulfur dioxide** - The total amount of SO2 in the wine, including both free and bound forms (int).

- **Density** – total density of the wine (float).
- **pH** - acidity or basicity of the wine on a scale from 0 (very acidic) to 14 (very basic) (int).
- **Sulphates** – Number of sulphates or additives present in the wine (float).
- **Alcohol** - The percentage of alcohol by volume in the wine (float).
- **Quality (score between 0 and 10)** - serves as the target for modeling efforts, where the goal is to predict wine quality based on its chemical makeup (int).

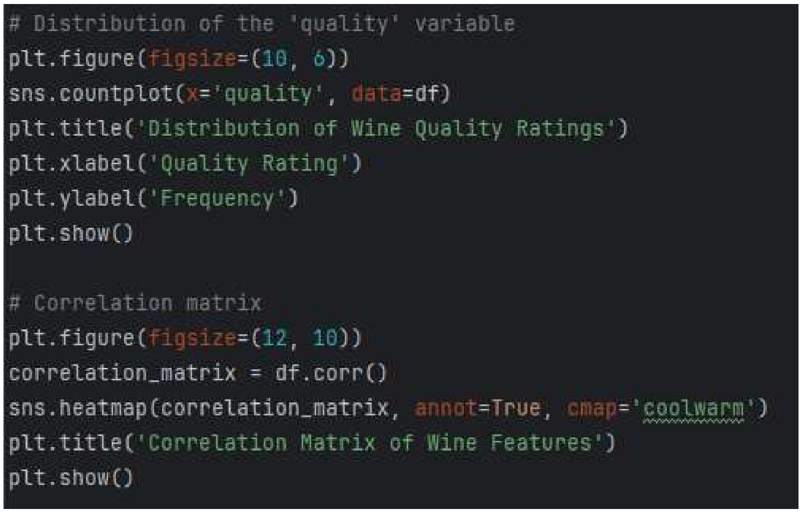
Exploratory Data Analysis

1. To start the exploratory data analysis, I first needed to load and preview the dataset to make sure the data was loaded correctly into python.



```
# Read in dataset as a dataframe
df = pd.read_csv('winequality-red.csv')

# Show the first few rows of the dataframe
print(df.head())
# Show the first few columns of the dataframe
print(df.columns)
```

2. Next the exploratory data analysis was conducted by creating a distribution chart of the 'quality' feature as well as a correlation matrix.

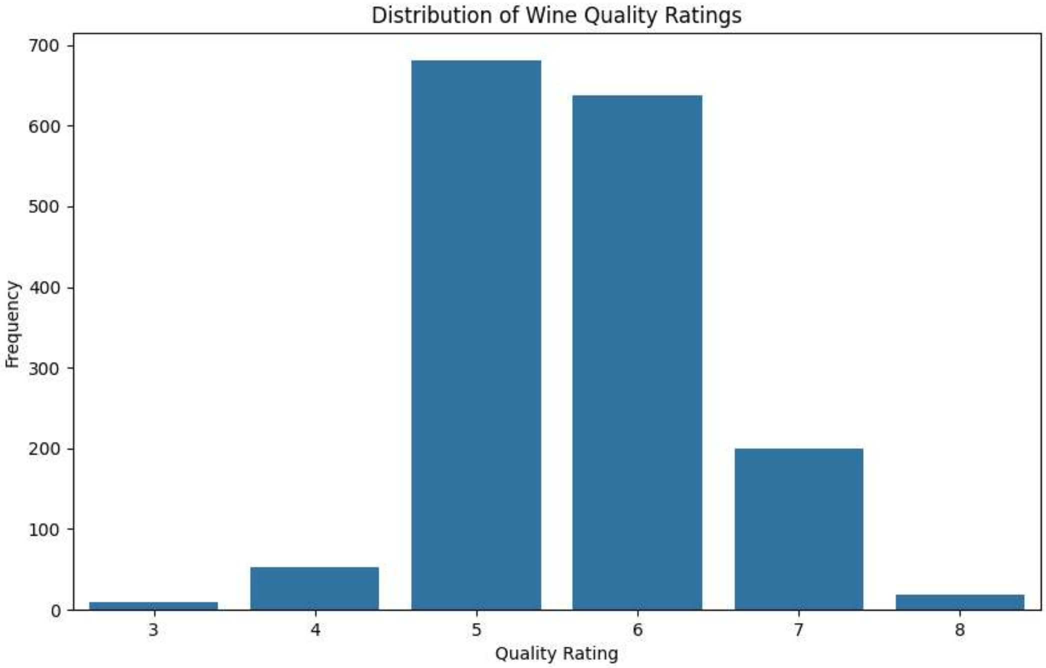


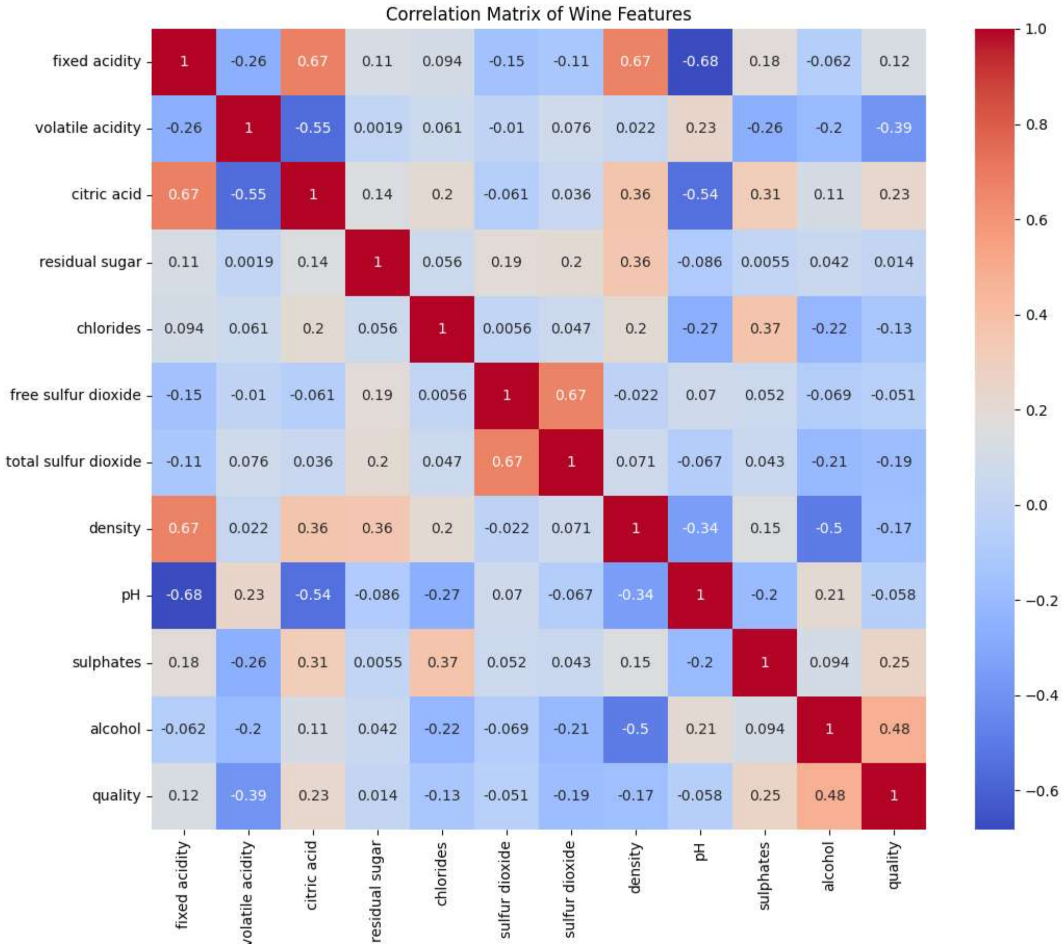
```
# Distribution of the 'quality' variable
plt.figure(figsize=(10, 6))
sns.countplot(x='quality', data=df)
plt.title('Distribution of Wine Quality Ratings')
plt.xlabel('Quality Rating')
plt.ylabel('Frequency')
plt.show()

# Correlation matrix
plt.figure(figsize=(12, 10))
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix of Wine Features')
plt.show()
```

a. EDA is crucial for understanding the structure of the dataset, identifying patterns, outliers, and relationships between variables.

   i. **Distribution of 'quality'**: This visualization helps understand how the quality ratings are distributed, which is essential for determining if the dataset is imbalanced or if certain quality ratings are underrepresented.

   ii. **Correlation Matrix:** By plotting a heatmap of the correlation matrix, I can identify how different features correlate with each other and with the variable (quality). This step is vital for feature selection and for understanding which variables might have more influence on the quality rating.





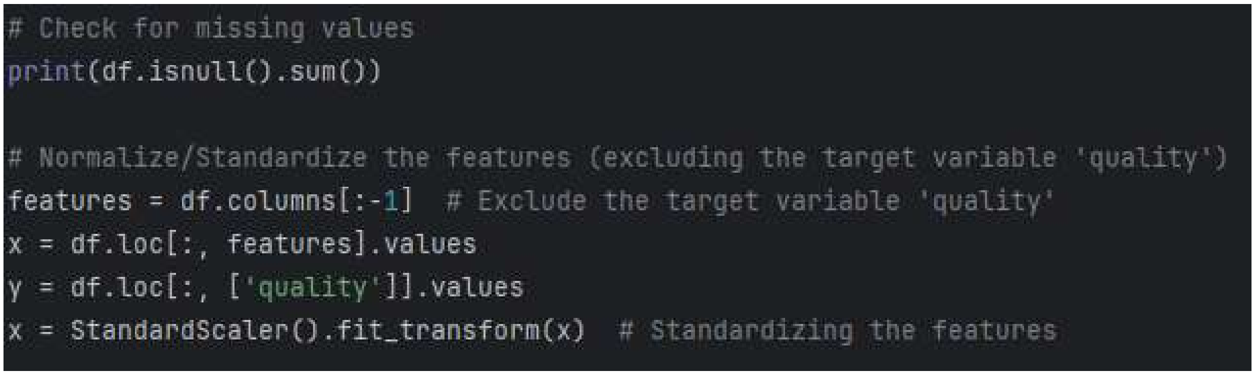
Correlation Matrix of Wine Features

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	sulfur dioxide	sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1	-0.26	0.67	0.11	0.094	-0.15	-0.11	0.67	-0.68	0.18	-0.062	0.12
volatile acidity	-0.26	1	-0.55	0.0019	0.061	-0.01	0.076	0.022	0.23	-0.26	-0.2	-0.39
citric acid	0.67	-0.55	1	0.14	0.2	-0.061	0.036	0.36	-0.54	0.31	0.11	0.23
residual sugar	0.11	0.0019	0.14	1	0.056	0.19	0.2	0.36	-0.086	0.0055	0.042	0.014
chlorides	0.094	0.061	0.2	0.056	1	0.0056	0.047	0.2	-0.27	0.37	-0.22	-0.13
free sulfur dioxide	-0.15	-0.01	-0.061	0.19	0.0056	1	0.67	-0.022	0.07	0.052	-0.069	-0.051
total sulfur dioxide	-0.11	0.076	0.036	0.2	0.047	0.67	1	0.071	-0.067	0.043	-0.21	-0.19
density	0.67	0.022	0.36	0.36	0.2	-0.022	0.071	1	-0.34	0.15	-0.5	-0.17
pH	-0.68	0.23	-0.54	-0.086	-0.27	0.07	-0.067	-0.34	1	-0.2	0.21	-0.058
sulphates	0.18	-0.26	0.31	0.0055	0.37	0.052	0.043	0.15	-0.2	1	0.094	0.25
alcohol	-0.062	-0.2	0.11	0.042	-0.22	-0.069	-0.21	-0.5	0.21	0.094	1	0.48
quality	0.12	-0.39	0.23	0.014	-0.13	-0.051	-0.19	-0.17	-0.058	0.25	0.48	1

Data Cleaning and Dimensionality

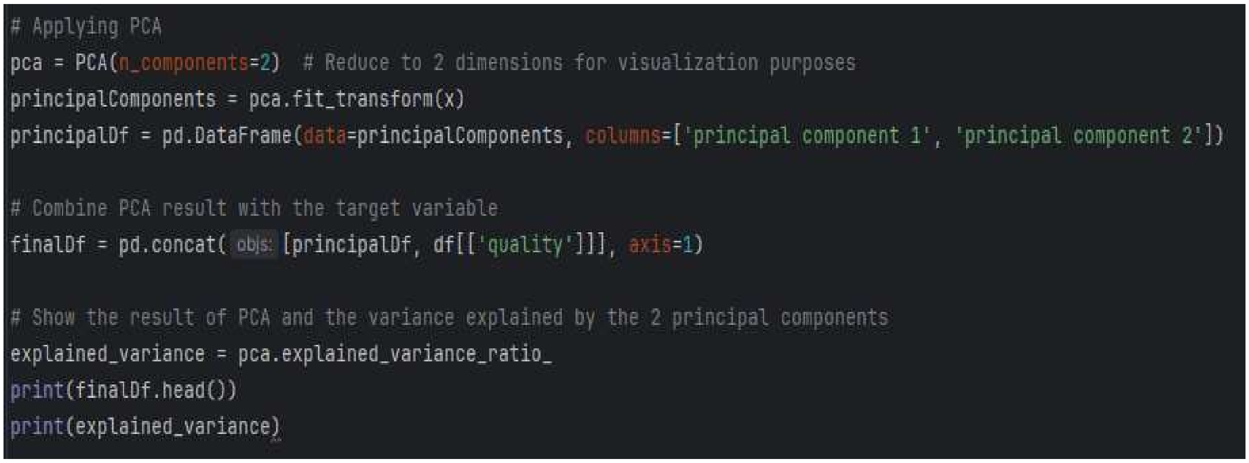
1. **Data Cleaning:**
   a. To start cleaning the data I started by checking for any missing values so that I know when I train the data it isn’t incomplete which would lead to skewed results. Luckily this dataset did not have any missing data. However, if it did, I would possibly have to remove certain rows or columns.
   b. Next, I needed to normalize the data and to do this I decided to use ‘StandardScaler’ which standardizes features by removing the mean and scaling to unit variance. This step is very important before moving onto

dimensionality reduction since the method I used is sensitive to variances of the initial variables.



```
# Check for missing values
print(df.isnull().sum())

# Normalize/Standardize the features (excluding the target variable 'quality')
features = df.columns[:-1]  # Exclude the target variable 'quality'
x = df.loc[:, features].values
y = df.loc[:, ['quality']].values
x = StandardScaler().fit_transform(x)  # Standardizing the features
```

2. **Dimensionality reduction:**
   a. To perform the dimensionality reduction PCA or Principal component reduction was used to transform the data into a lower-dimensional space while retaining most of the variance. I decided to reduce the data to 2 dimensions (‘n_components=2’) for the sake of simplicity and visualization which will allow me to see how much of the variance can be captured in the 2 dimensions.

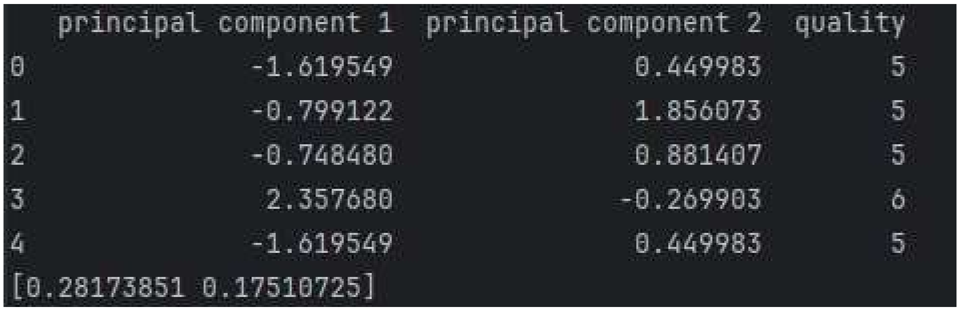


```
# Applying PCA
pca = PCA(n_components=2)  # Reduce to 2 dimensions for visualization purposes
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data=principalComponents, columns=['principal component 1', 'principal component 2'])

# Combine PCA result with the target variable
finalDf = pd.concat( objs: [principalDf, df[['quality']]], axis=1)

# Show the result of PCA and the variance explained by the 2 principal components
explained_variance = pca.explained_variance_ratio_
print(finalDf.head())
print(explained_variance)
```

   b. The ‘explained_variance_ratio’ indicates the proportion of the datasets variance that lies along each principal component which in turn helps to understand the reduction.

c. I’ve also created a new data frame, ‘principalDF’, that contains the principal components and combined it with the quality column which allows me to see the reduced features in conjunction with the quality variable.

The results of the dimensionality reduction for the first 4 rows are as follows:



```
   principal component 1  principal component 2  quality
0              -1.619549               0.449983        5
1              -0.799122               1.856073        5
2              -0.748480               0.881407        5
3               2.357680              -0.269903        6
4              -1.619549               0.449983        5
[0.28173851 0.17510725]
```

The PCA transformation indicates that the first two principal components capture a significant portion of the variance in the wine quality dataset. This suggests that these components encapsulate the essential patterns and relationships among the features that influence wine quality.

Steps for running the code attached

1. Import both “WineQualityCleaning.py” and “winequality-red.csv” into an open PyCharm project.
2. Ensure there are no errors with the libraries at the top of the file being imported and if there are errors make sure that you install each library before trying to run the code.
3. Once there are no errors and both files are in the directory of the PyCharm project simply click run and all the results listed should appear.