exploratory-data-analysis-1

August 13, 2023

[ ]:

```
# Q1. What are the key features of the wine quality data set? Discuss the importance of each feature in predicting the quality of wine.
```

Here are some key features often found in the "Wine Quality" dataset and their importance:

1 Fixed Acidity: Fixed acidity is the total amount of acids present in the wine. It contributes to the overall taste and structure of the wine. Wines with balanced acidity are generally considered of higher quality.

2 Volatile Acidity: Volatile acidity refers to the presence of volatile acids in the wine, which can lead to off-flavors and unpleasant aromas. Low levels of volatile acidity are desirable in quality wines.

3 Citric Acid: Citric acid is a natural acid found in citrus fruits. It can contribute to the freshness and flavor balance of the wine. Proper levels of citric acid can enhance the wine's quality.

4 Residual Sugar: Residual sugar is the amount of sugar left after fermentation. It affects the wine's sweetness and can balance the overall taste. The optimal level of residual sugar depends on the wine type.

5 Chlorides: Chlorides contribute to the saltiness of the wine. A balanced chloride level is crucial for maintaining the wine's overall flavor profile.

6 Free Sulfur Dioxide: Free sulfur dioxide is used as a preservative and antioxidant in wine. Its presence helps prevent spoilage and oxidation, ensuring the wine's stability and quality.

7 Total Sulfur Dioxide: Total sulfur dioxide includes both free and bound sulfur dioxide. It's an important factor in preserving wine quality and preventing microbial growth.

8 Density: The density of wine can indicate its level of alcohol and sugar content. It's an essential parameter for winemakers to monitor and control during production.

9 pH: pH influences the wine's taste, texture, and overall quality. Wines with appropriate pH levels are more stable and enjoyable to drink.

10 Sulphates: Sulphates, or sulfites, are added to wines as a preservative. They can affect the wine's flavor, texture, and aging potential.

11 Alcohol: Alcohol content influences the wine's body, flavor,

[ ]:

```
# Q2. How did you handle missing data in the wine quality data set during the feature engineering process?

# Discuss the advantages and disadvantages of different imputation techniques.
```

Handling missing data is a critical step in the feature engineering process to ensure the quality and reliability of the analysis and modeling. In the context of the wine quality dataset, missing data might arise due to various reasons such as measurement errors or data collection issues. Different imputation techniques can be employed to address missing values.

13 Mean/Median Imputation:

Advantages: Simple to implement, doesn't distort the distribution of the feature significantly. Disadvantages: Ignores relationships between variables, can lead to biased estimates if data is not missing at random.

14 Mode Imputation:

Advantages: Suitable for categorical variables, preserves the mode of the distribution. Disadvantages: Similar to mean/median imputation, doesn't consider relationships between variables.

15 K-Nearest Neighbors (KNN) Imputation:

Advantages: Considers relationships between variables, can provide more accurate imputations. Disadvantages: Computationally intensive, sensitive to the choice of the number of neighbors.

[ ]:

```
# Q3. What are the key factors that affect students' performance in exams?

# How would you go about analyzing these factors using statistical techniques.
```

Study Time: The amount of time a student spends studying can have a significant impact on exam performance.

Attendance: Regular attendance in classes and lectures can contribute to a better understanding of the material.

Prior Academic Performance: Students with a strong academic background are likely to perform better in exams.

Test Anxiety: Anxiety or stress related to exams can negatively affect performance.

Learning Style: Different students have different learning styles, and aligning teaching methods with their preferences can impact understanding.

Socioeconomic Background: Economic and social factors can influence access to resources, like tutoring, study materials, and a conducive learning environment.

Parental Involvement: Support from parents or guardians can positively affect students' motivation and preparation.

Health and Well-being: Physical and mental health issues can impact concentration and cognitive performance.

16 To analyze these factors using statistical techniques:

Data Collection: Gather data on each of the identified factors, such as study time, attendance, previous exam scores, etc.

Descriptive Analysis: Start with descriptive statistics to understand the central tendencies, distributions, and variations in the data.

Correlation Analysis: Calculate correlation coefficients to assess the strength and direction of relationships between variables. For example, you can check if there's a correlation between study time and exam scores.

Regression Analysis: Conduct regression analysis to quantify the impact of different factors on exam performance. For instance, you can build a regression model with exam scores as the dependent variable and study time, attendance, prior scores, etc., as independent variables.

Hypothesis Testing: Use hypothesis testing to determine if certain factors have a statistically significant impact on exam performance. For example, you can perform a t-test to see if students with high attendance have significantly different exam scores than those with low attendance.

Data Visualization: Create visualizations like scatter plots, bar charts, and box plots to visually represent relationships between variables.

Multivariate Analysis: If multiple factors interact to influence performance, consider multivariate techniques like factor analysis to identify underlying dimensions that contribute to student success.

Segmentation Analysis: Group students based on characteristics (e.g., socioeconomic background) and compare performance within and across segments.

Machine Learning: If the dataset is large and complex, consider using machine learning algorithms to predict exam scores based on various factors.

[ ]:

```
# Q4. Describe the process of feature engineering in the context of the student performance data set.

# How did you select and transform the variables for your model.
```

17 1. Data Understanding:

Begin by understanding the dataset, its variables, and their meanings. Identify the target variable, which in this case might be "Exam Score." # 2. Feature Selection:

Start by selecting relevant variables that might have a direct impact on students' exam performance. Variables like "Study Time," "Attendance," "Prior Exam Scores," and "Parental Involvement" could be considered as potential features. # 3. Handling Missing Data:

Check for missing values in the selected variables and decide on an appropriate strategy for handling them (imputation, removal, etc.). # 4. Categorical Variable Encoding:

If there are categorical variables like "Learning Style," encode them into numerical format using techniques like label encoding or one-hot encoding. # 5. Feature Creation:

Create new features that might capture meaningful relationships or interactions. For example, you could create a "Study Efficiency" feature by dividing "Study Time" by "Attendance." Compute aggregated features like the average of prior exam scores to capture overall academic history. # 6. Data Normalization/Scaling:

Normalize or scale numerical features to ensure that they're on a similar scale. This is especially important for algorithms that are sensitive to feature scales, such as distance-based algorithms. # 7. Feature Transformation:

Apply transformations to features to make the relationships more linear or to correct skewed distributions. Techniques like logarithmic transformation can help in these cases. # 8. Dimensionality Reduction:

If the dataset has high dimensionality, consider techniques like Principal Component Analysis (PCA) to reduce the number of features while retaining important information. # 9. Data Splitting:

Split the dataset into training and testing sets to assess model performance. # 10. Model Building:

Use the engineered features to build a predictive model. You can start with simple models like linear regression and then explore more complex models. # 11. Model Evaluation:

Evaluate the model's performance using appropriate metrics (e.g., Mean Absolute Error, Root Mean Squared Error) on the test set. # 12. Iteration and Refinement:

Iterate through steps 4 to 11, refining feature selection, engineering, and modeling to improve performance. # 13. Interpretability:

Analyze the model's coefficients (for linear models) or feature importances (for tree-based models) to understand which features are most influential.

[ ]:

```
# Q5. Load the wine quality data set and perform exploratory data analysis␣
 ↪(EDA) to identify the distribution of each feature.



# Which feature(s) exhibit non-normality, and what transformations could be␣
 ↪applied to these features to improve normality?
```

[3]:

```
import pandas as pd

# Define column names for the dataset
column_names = ['Class', 'Alcohol', 'Malic Acid', 'Ash', 'Alcalinity of Ash',␣
 ↪'Magnesium', 'Total Phenols', 'Flavanoids', 'Nonflavanoid Phenols',␣
 ↪'Proanthocyanins', 'Color Intensity', 'Hue', 'OD280/OD315 of Diluted Wines',␣
 ↪'Proline']

# Load the wine data from the downloaded file
with open('wine.data', 'r') as file:
    lines = file.readlines()
```

```
# Create a list to hold the data
data = []

# Split and process each line of data
for line in lines:
    values = line.strip().split(',')
    data.append(values)

# Create a pandas DataFrame
wine_df = pd.DataFrame(data, columns=column_names)

# Save the DataFrame as a CSV file
wine_df.to_csv('wine_quality.csv', index=False)

print("CSV file saved as 'wine_quality.csv'")
```

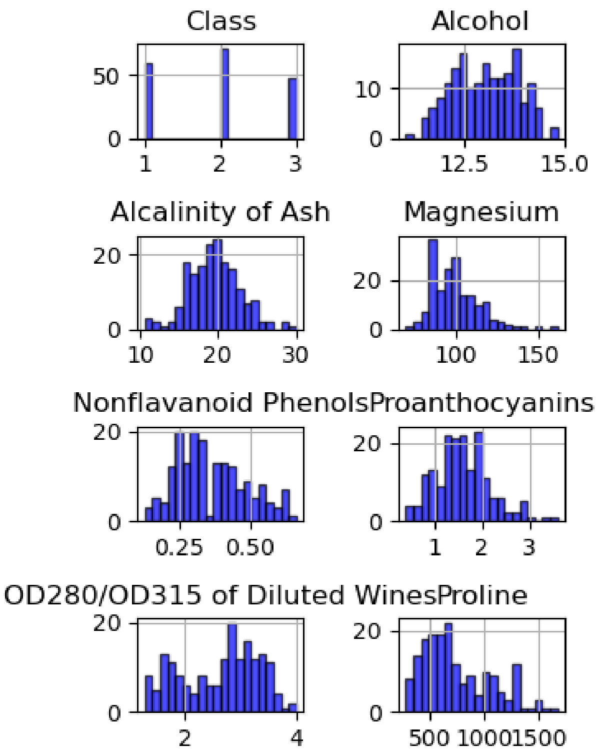
```
CSV file saved as 'wine_quality.csv'
```

[5]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats

# Load the wine quality dataset
wine_data = pd.read_csv('wine_quality.csv')

# Display summary statistics
print(wine_data.describe())

# Visualize distribution of each feature
plt.figure(figsize=(12, 8))
wine_data.hist(bins=20, color='blue', edgecolor='black', alpha=0.7)
plt.tight_layout()
plt.show()

# Check for skewness
skewness = wine_data.skew()
print("Skewness:")
print(skewness)

# Apply transformations to improve normality (e.g., logarithmic transformation)
skewed_features = skewness[skewness > 1].index
for feature in skewed_features:
    wine_data[feature] = wine_data[feature].apply(lambda x: np.log1p(x))

# Visualize transformed distribution of each skewed feature
```

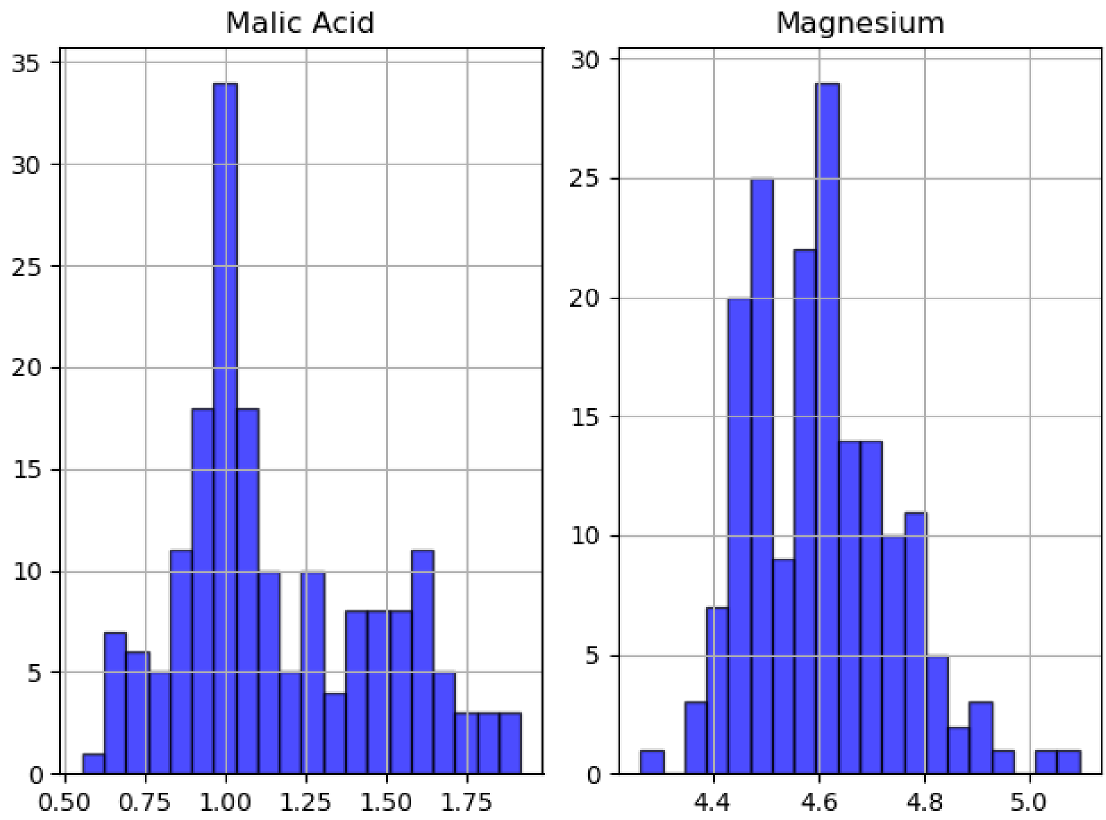
```python
plt.figure(figsize=(12, 8))
wine_data[skewed_features].hist(bins=20, color='blue', edgecolor='black',␣
 ↪alpha=0.7)
plt.tight_layout()
plt.show()
```

```
            Class     Alcohol  Malic Acid         Ash  Alcalinity of Ash  \
count  178.000000  178.000000  178.000000  178.000000         178.000000
mean     1.938202   13.000618    2.336348    2.366517          19.494944
std      0.775035    0.811827    1.117146    0.274344           3.339564
min      1.000000   11.030000    0.740000    1.360000          10.600000
25%      1.000000   12.362500    1.602500    2.210000          17.200000
50%      2.000000   13.050000    1.865000    2.360000          19.500000
75%      3.000000   13.677500    3.082500    2.557500          21.500000
max      3.000000   14.830000    5.800000    3.230000          30.000000

        Magnesium  Total Phenols  Flavanoids  Nonflavanoid Phenols  \
count  178.000000     178.000000  178.000000            178.000000
mean    99.741573       2.295112    2.029270              0.361854
std     14.282484       0.625851    0.998859              0.124453
min     70.000000       0.980000    0.340000              0.130000
25%     88.000000       1.742500    1.205000              0.270000
50%     98.000000       2.355000    2.135000              0.340000
75%    107.000000       2.800000    2.875000              0.437500
max    162.000000       3.880000    5.080000              0.660000

       Proanthocyanins  Color Intensity         Hue  \
count       178.000000       178.000000  178.000000
mean          1.590899         5.058090    0.957449
std           0.572359         2.318286    0.228572
min           0.410000         1.280000    0.480000
25%           1.250000         3.220000    0.782500
50%           1.555000         4.690000    0.965000
75%           1.950000         6.200000    1.120000
max           3.580000        13.000000    1.710000

       OD280/OD315 of Diluted Wines      Proline
count                    178.000000   178.000000
mean                       2.611685   746.893258
std                        0.709990   314.907474
min                        1.270000   278.000000
25%                        1.937500   500.500000
50%                        2.780000   673.500000
75%                        3.170000   985.000000
max                        4.000000  1680.000000

<Figure size 1200x800 with 0 Axes>
```



```
Skewness:
Class                           0.107431
Alcohol                        -0.051482
Malic Acid                      1.039651
Ash                            -0.176699
Alcalinity of Ash               0.213047
Magnesium                       1.098191
Total Phenols                   0.086639
Flavanoids                      0.025344
Nonflavanoid Phenols            0.450151
Proanthocyanins                 0.517137
Color Intensity                 0.868585
Hue                             0.021091
OD280/OD315 of Diluted Wines   -0.307285
Proline                         0.767822
dtype: float64
```

```
<Figure size 1200x800 with 0 Axes>
```



In this code:

We load the wine quality dataset using pd.read_csv(). We display summary statistics using describe(). We visualize the distribution of each feature using histograms. We calculate the skewness of each feature using skew() to identify non-normality. We identify features with skewness greater than 1 (a common threshold for identifying skewed distributions). We apply a logarithmic transformation (np.log1p()) to improve normality of skewed features. We visualize the distribution of the transformed skewed features.

[ ]:

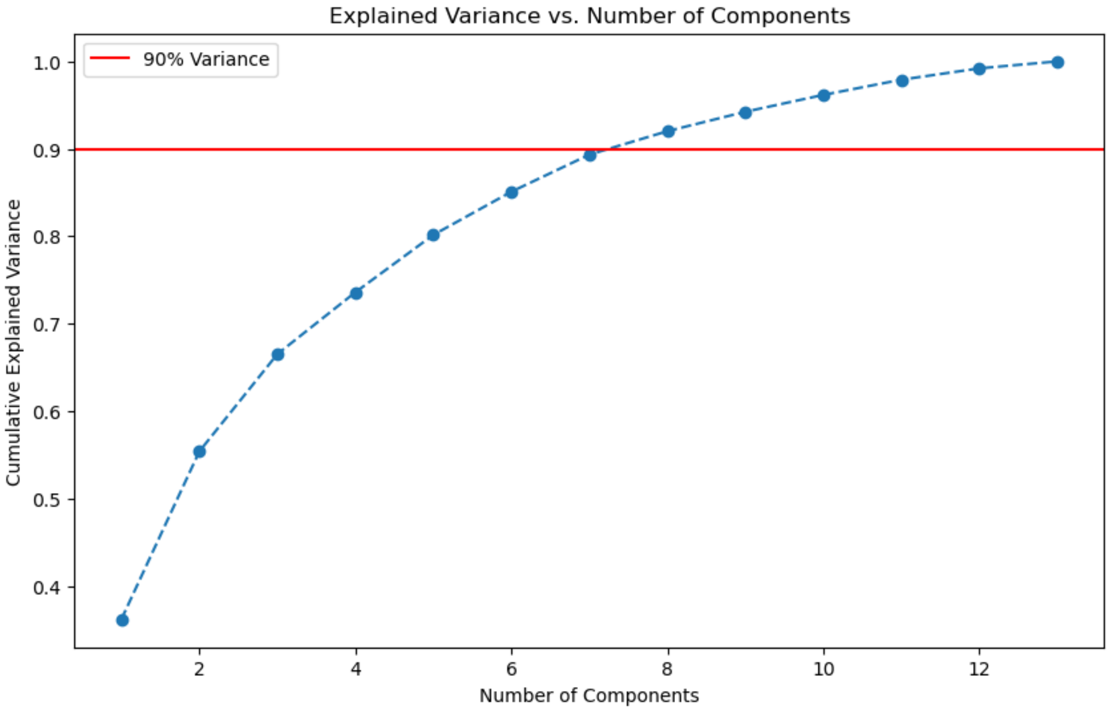
```
# Q6. Using the wine quality data set, perform principal component analysis (PCA) to reduce the number of features.

# What is the minimum number of principal components required to explain 90% of the variance in the data?
```

[6]:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Load the wine quality dataset
wine_data = pd.read_csv('wine_quality.csv')
```

```python
# Separate features and target variable
X = wine_data.drop(columns=['Class'])
y = wine_data['Class']

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Perform PCA
pca = PCA()
X_pca = pca.fit_transform(X_scaled)

# Calculate explained variance ratio
explained_variance_ratio = pca.explained_variance_ratio_

# Calculate cumulative explained variance ratio
cumulative_explained_variance = np.cumsum(explained_variance_ratio)

# Find the minimum number of components to explain 90% of the variance
min_components = np.argmax(cumulative_explained_variance >= 0.9) + 1

print("Explained Variance Ratio:", explained_variance_ratio)
print("Cumulative Explained Variance:", cumulative_explained_variance)
print("Minimum Number of Components for 90% Variance:", min_components)

# Plot explained variance ratio
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(explained_variance_ratio) + 1),␣
  ↪cumulative_explained_variance, marker='o', linestyle='--')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('Explained Variance vs. Number of Components')
plt.axhline(y=0.9, color='r', linestyle='-', label='90% Variance')
plt.legend()
plt.show()
```

```
Explained Variance Ratio: [0.36198848 0.1920749  0.11123631 0.0706903
0.06563294 0.04935823
 0.04238679 0.02680749 0.02222153 0.01930019 0.01736836 0.01298233
 0.00795215]
Cumulative Explained Variance: [0.36198848 0.55406338 0.66529969 0.73598999
0.80162293 0.85098116
 0.89336795 0.92017544 0.94239698 0.96169717 0.97906553 0.99204785
 1.        ]
Minimum Number of Components for 90% Variance: 8
```



In this code:

We load the wine quality dataset and separate the features from the target variable. We standardize the features using StandardScaler. We perform PCA on the standardized features. We calculate the explained variance ratio for each principal component and the cumulative explained variance. We find the minimum number of components needed to explain 90% of the variance.

```
[ ]:
```