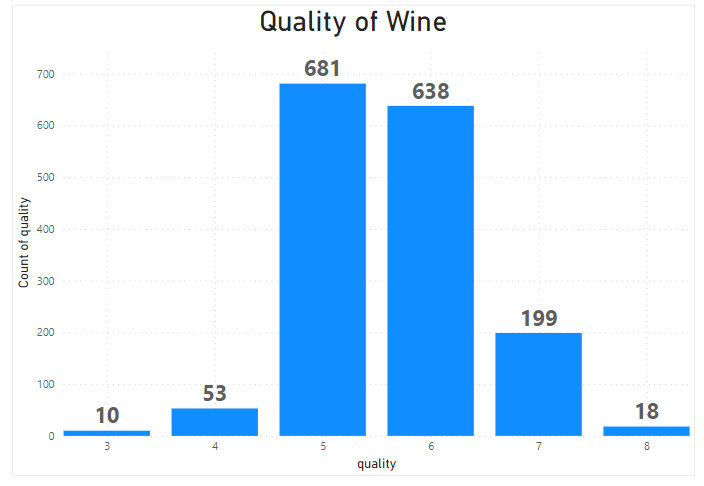
**Answers**

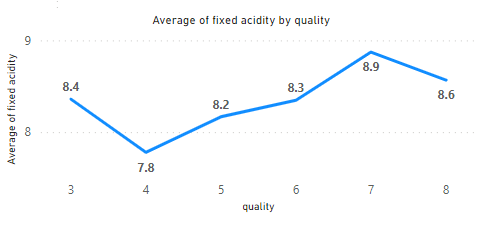
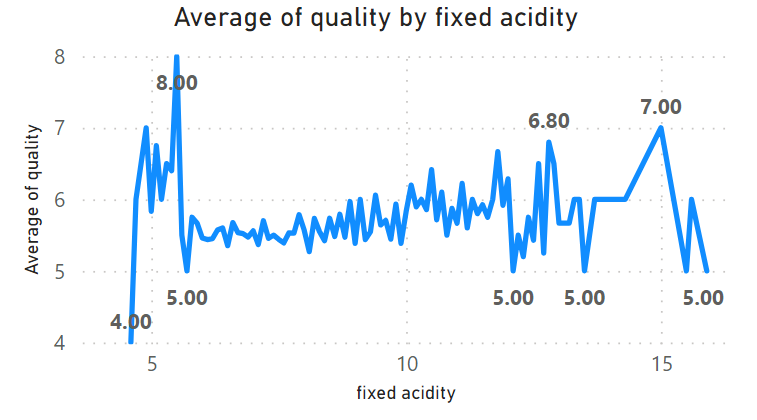
1. **What is the most frequently occurring wine quality? What is the highest number in and the lowest number in the quantity column?**

The most frequently occurring wine quality is 5 

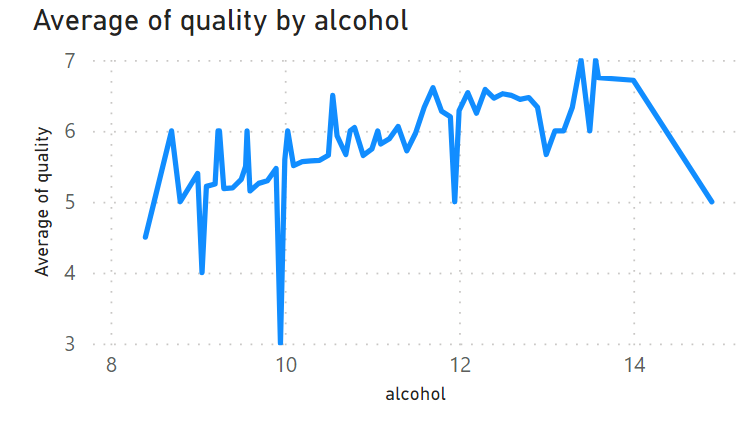
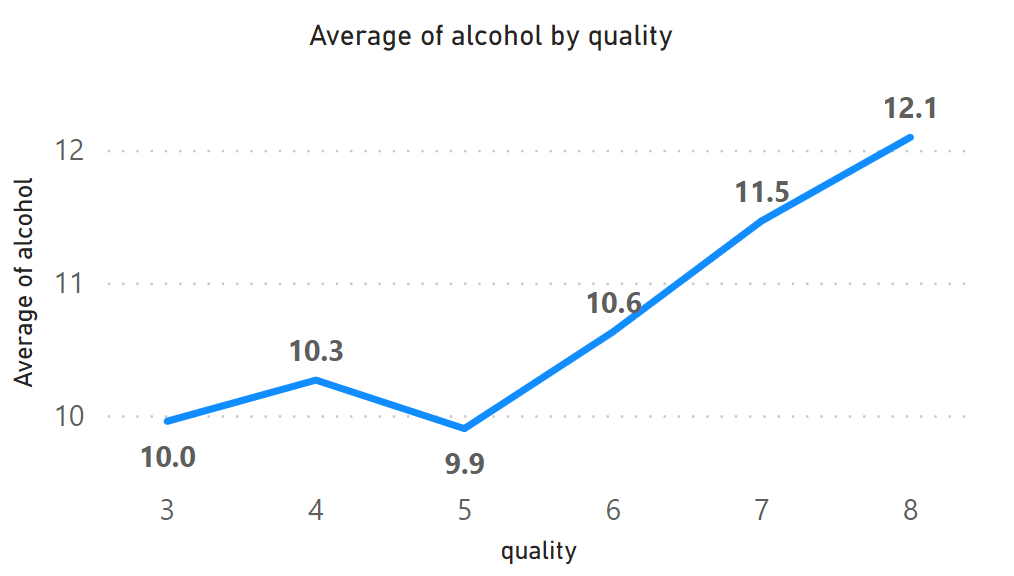
Highest and lowest number of wine quality being 8 and 3 respectively

1. **How is `fixed acidity` correlated to the quality of the wine? How does the alcohol content affect the quality? How is the `free Sulphur dioxide` content correlated to the quality of the wine?**

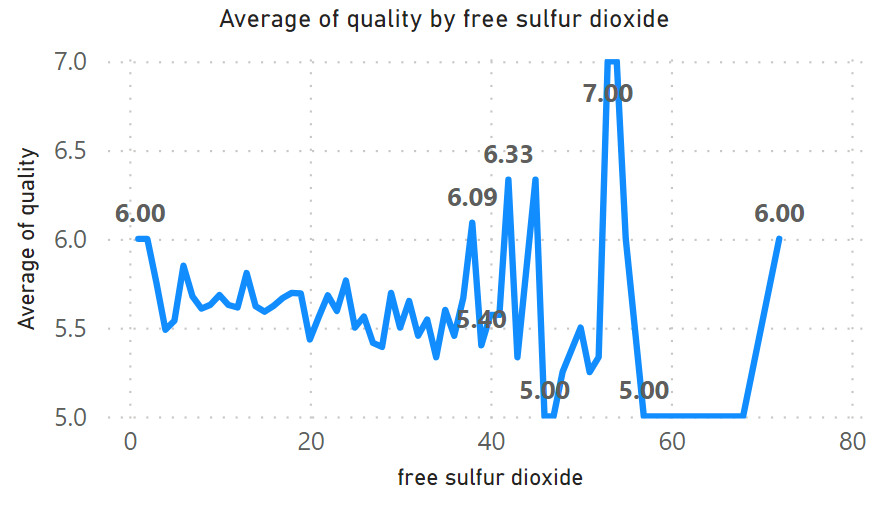
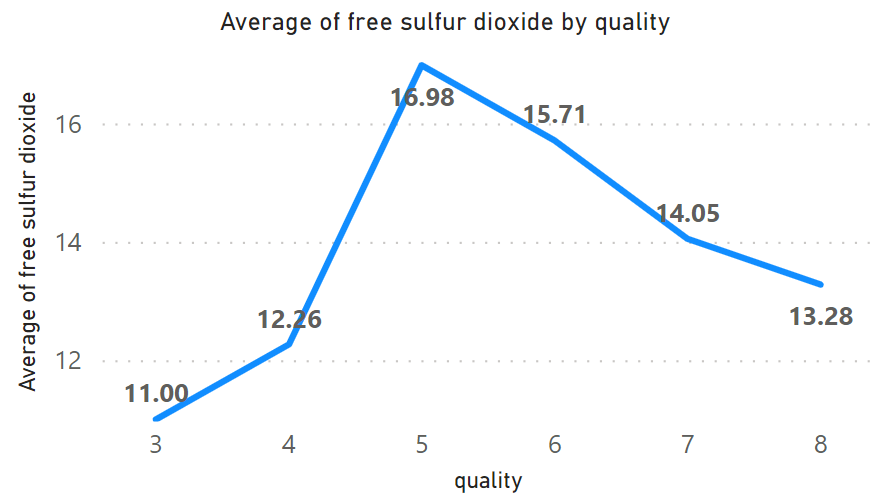
For a fixed acidity of 5.5, we have the highest average quality of wine and although there is no strong correlation, as a general trend, wine quality increases with increase in fixed acidity from 5.7 to 11.8. However one can say that the average fixed acidity varies as follows for the following wine quality



Quality of wine increases in general with increase in alcohol content except for when alcohol content is more than 14. However on an average, higher quality wine has greater alcohol

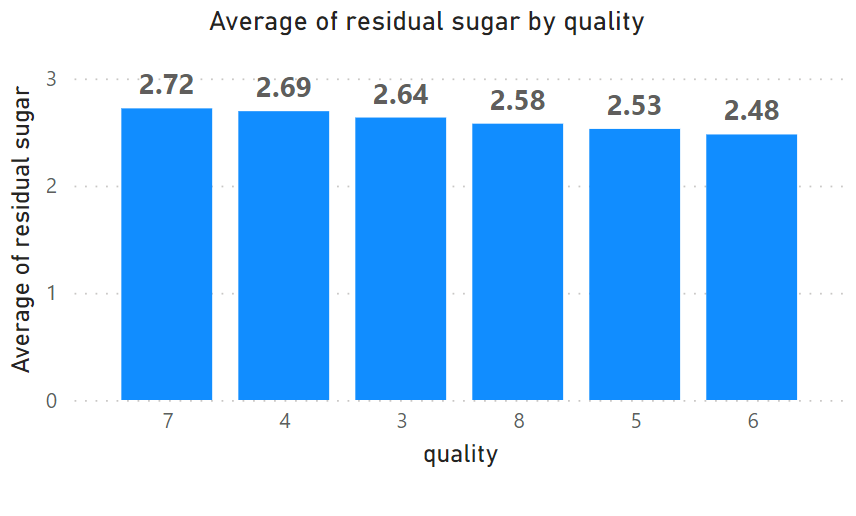
 

Not much correlation can be concluded for free sulphur dioxide and wine quality however one can say that the average free sulphur dioxide is highest for wine quality of 5

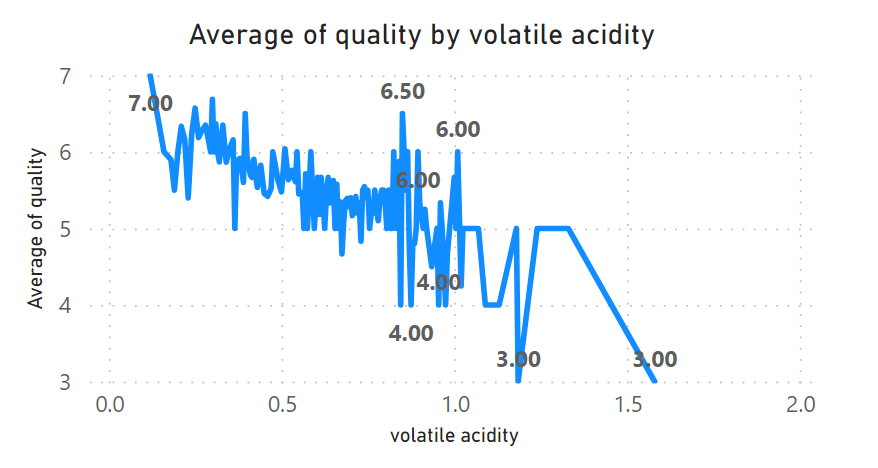
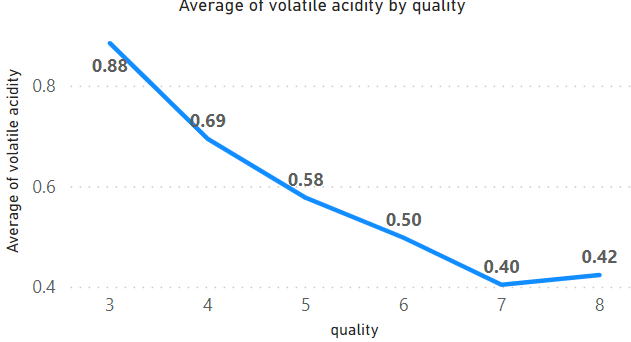
1. **What is the average `residual sugar` for the best quality wine and the lowest quality wine in the dataset?**

2.64 for lowest quality and 2.58 for highest quality wine



1. **Does `volatile acidity` has an effect over the quality of the wine samples in the dataset?**

In general the lower the volatility, the higher the quality of wine which can be represented from this graph. Additionally, for the wine quality of 7, the avg volatility is the lowest and greatest for wine quality of 3

1. **Train a Decision Tree model and Random Forest Model separately to predict the Quality of the given samples of wine. Compare the Accuracy scores for both models.**

**Decision Tree**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import classification\_report, confusion\_matrix

# Load the dataset

file\_path = r'C:\Users\ishita\Downloads\COEP EXTRACURRICLUARS\FINLATICS ML\wine\_data.csv' # Replace with your file path

wine\_data = pd.read\_csv(file\_path)

# Separate the features and the target variable

X = wine\_data.drop('quality', axis=1)

y = wine\_data['quality']

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize and train the Decision Tree model

dt\_model = DecisionTreeClassifier(random\_state=42)

dt\_model.fit(X\_train, y\_train)

# Predict the quality on the test set

y\_pred = dt\_model.predict(X\_test)

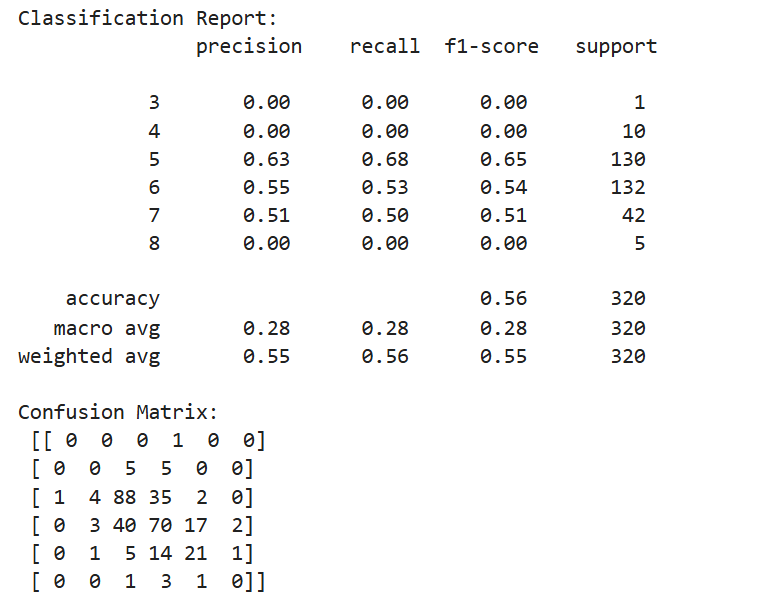
# Evaluate the model

classification\_rep = classification\_report(y\_test, y\_pred)

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print("Classification Report:\n", classification\_rep)

print("Confusion Matrix:\n", conf\_matrix)



**Random Forest Tree**

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

# Separate the features and the target variable

X = wine\_data.drop(columns=['quality'])

y = wine\_data['quality']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Standardize the features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Display the shapes of the training and testing sets

(X\_train\_scaled.shape, X\_test\_scaled.shape, y\_train.shape, y\_test.shape)

# Separate the features and the target variable

X = wine\_data.drop('quality', axis=1)

y = wine\_data['quality']

# Split the data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize and train the Random Forest model

rf\_model = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_model.fit(X\_train, y\_train)

# Predict the quality on the test set

y\_pred = rf\_model.predict(X\_test)

# Evaluate the model

classification\_rep = classification\_report(y\_test, y\_pred)

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print("Classification Report:\n", classification\_rep)

print("Confusion Matrix:\n", conf\_matrix)

