NAME: JAYANTH RAGAVAN M

COURSE: DATA SCIENCE (SELF PLACED)

BATCH: JULY-AUG

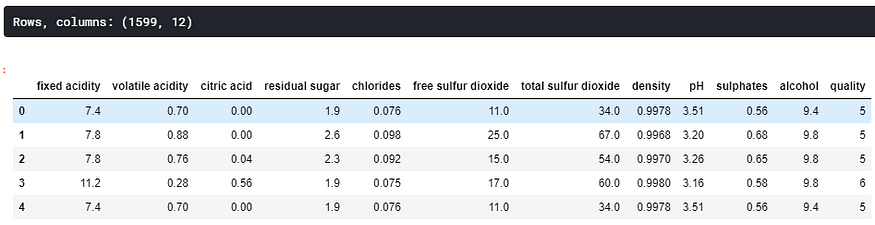
**Importing Libraries**

import numpy as np  
import pandas as pd  
import matplotlib as plt  
import seaborn as sns  
import plotly.express as px

df = pd.read\_csv("../input/red-wine-quality-cortez-et-al-2009/winequality-red.csv")

**Understanding Data**

# See the number of rows and columns  
print("Rows, columns: " + str(df.shape))# See the first five rows of the dataset  
df.head()



There are a total of 1599 rows and 12 columns. The data looks very clean by looking at the first five rows, but I still wanted to make sure that there were no missing values.

**Missing Values**

# Missing Values  
print(df.isna().sum())

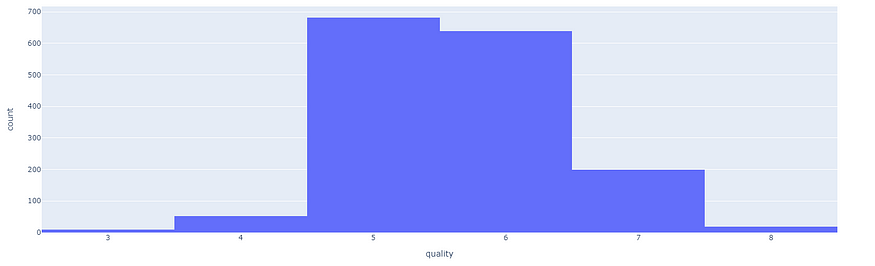


This is a very beginner-friendly dataset. I did not have to deal with any missing values, and there isn’t much flexibility to conduct some feature engineering given these variables. Next, I wanted to explore my data a little bit more.

**Exploring Variables**

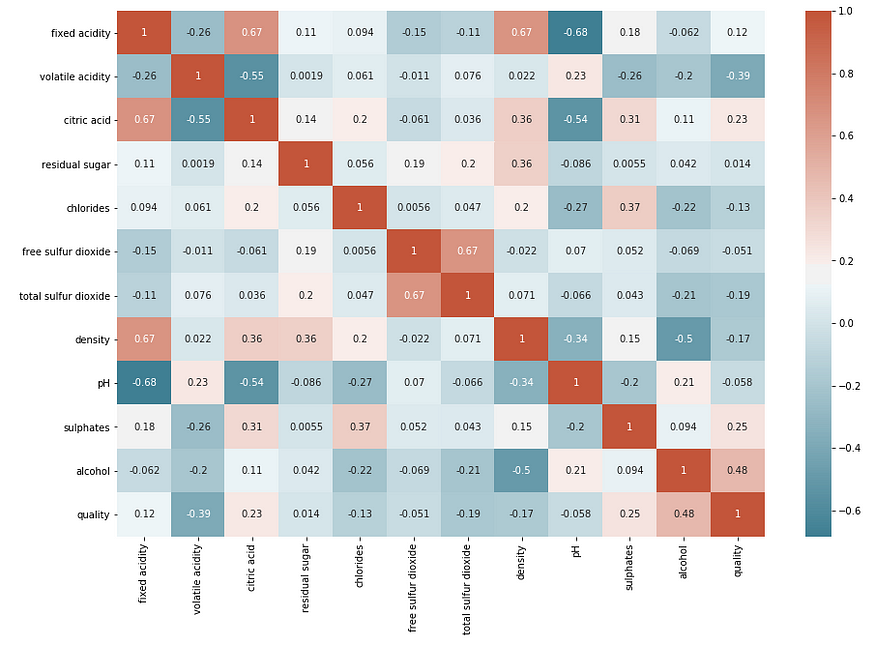
**Histogram of ‘quality’ variable**

fig = px.histogram(df,x='quality')  
fig.show()



**Correlation Matrix**

corr = df.corr()  
matplotlib.pyplot.subplots(figsize=(15,10))  
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, annot=True, cmap=sns.diverging\_palette(220, 20, as\_cmap=True))



**Convert to a Classification Problem**

# Create Classification version of target variable  
df['goodquality'] = [1 if x >= 7 else 0 for x in df['quality']]# Separate feature variables and target variable  
X = df.drop(['quality','goodquality'], axis = 1)  
y = df['goodquality']

**Proportion of Good vs Bad Wines**

# See proportion of good vs bad wines  
df['goodquality'].value\_counts()



**Preparing Data for Modelling**

**Standardizing Feature Variables**

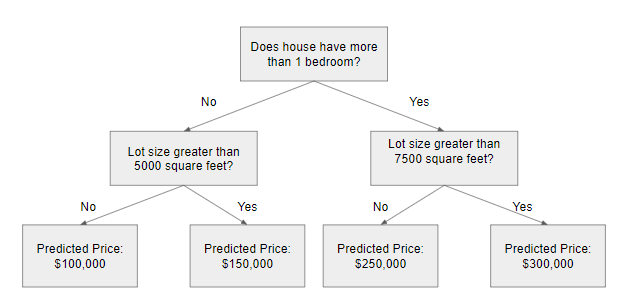
# Normalize feature variables  
from sklearn.preprocessing import StandardScaler  
X\_features = X  
X = StandardScaler().fit\_transform(X)

**Split data**

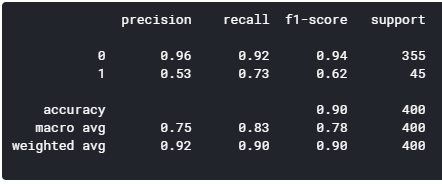
# Splitting the data  
from sklearn.model\_selection import train\_test\_split  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=.25, random\_state=0)

**Modelling**

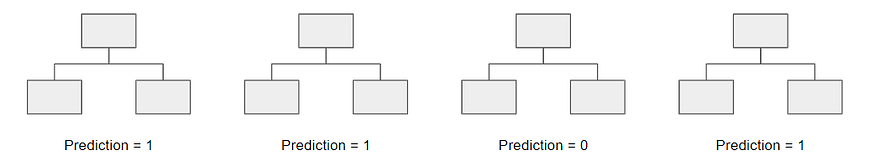
**Model 1: Decision Tree**



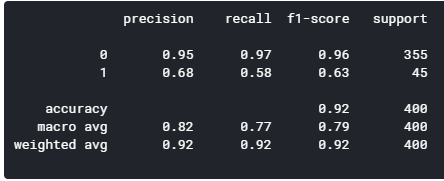
from sklearn.metrics import classification\_report  
from sklearn.tree import DecisionTreeClassifiermodel1 = DecisionTreeClassifier(random\_state=1)  
model1.fit(X\_train, y\_train)  
y\_pred1 = model1.predict(X\_test)print(classification\_report(y\_test, y\_pred1))



**Model 2: Random Forest**

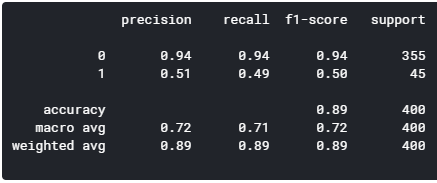


model2 = RandomForestClassifier(random\_state=1)  
model2.fit(X\_train, y\_train)  
y\_pred2 = model2.predict(X\_test)print(classification\_report(y\_test, y\_pred2))



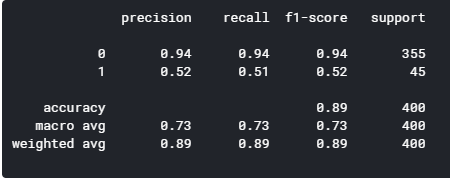
**Model 3: AdaBoost**

from sklearn.ensemble import AdaBoostClassifier  
model3 = AdaBoostClassifier(random\_state=1)  
model3.fit(X\_train, y\_train)  
y\_pred3 = model3.predict(X\_test)print(classification\_report(y\_test, y\_pred3))



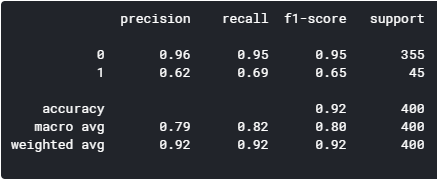
**Model 4: Gradient Boosting**

from sklearn.ensemble import GradientBoostingClassifier  
model4 = GradientBoostingClassifier(random\_state=1)  
model4.fit(X\_train, y\_train)  
y\_pred4 = model4.predict(X\_test)print(classification\_report(y\_test, y\_pred4))



**Model 5: XGBoost**

import xgboost as xgb  
model5 = xgb.XGBClassifier(random\_state=1)  
model5.fit(X\_train, y\_train)  
y\_pred5 = model5.predict(X\_test)print(classification\_report(y\_test, y\_pred5))



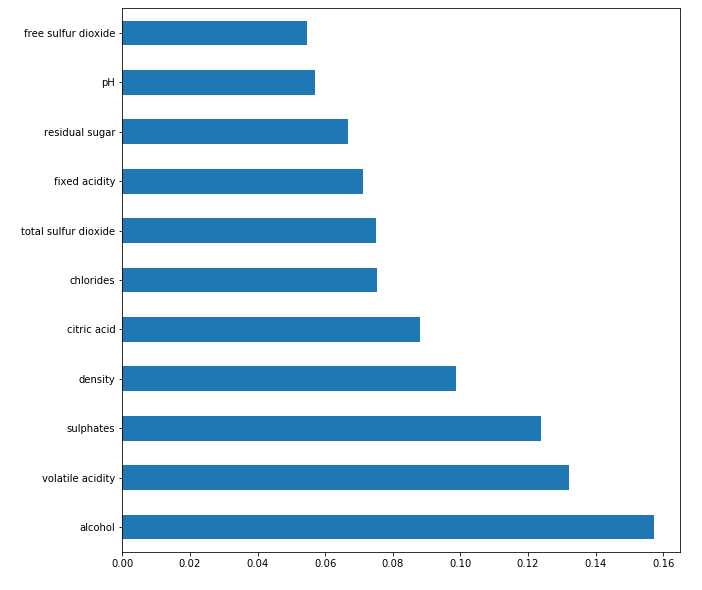
By comparing the five models, the random forest and XGBoost seems to yield the highest level of accuracy. However, since XGBoost has a better f1-score for predicting good quality wines (1), I’m concluding that the XGBoost is the winner of the five models.

**Feature Importance**

Below, I graphed the feature importance based on the Random Forest model and the XGBoost model. While they slightly vary, the top 3 features are the same: alcohol, volatile acidity, and sulphates. If you look below the graphs, I split the dataset into good quality and bad quality to compare these variables in more detail.

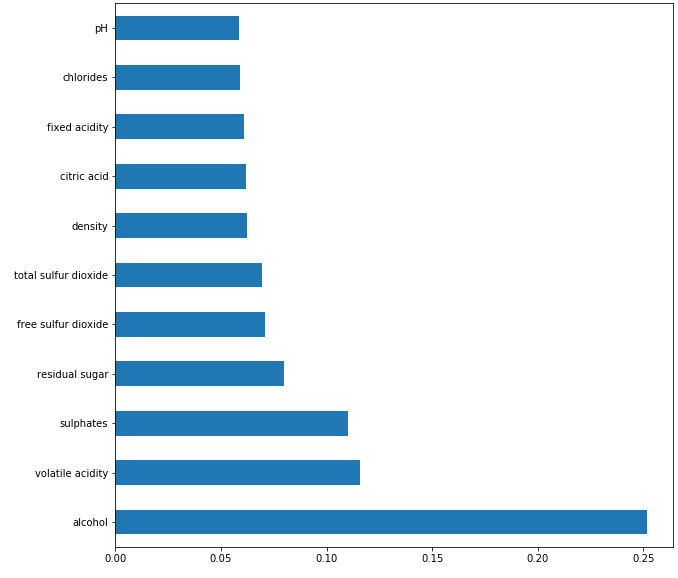
**via Random Forest**

feat\_importances = pd.Series(model2.feature\_importances\_, index=X\_features.columns)  
feat\_importances.nlargest(25).plot(kind='barh',figsize=(10,10))



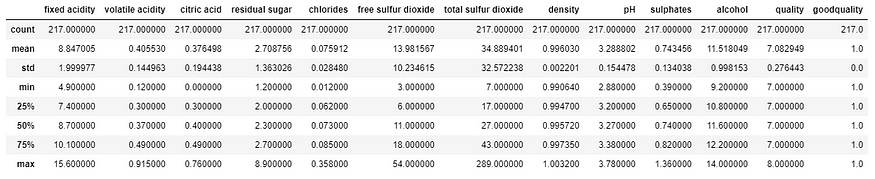
**via XGBoost**

feat\_importances = pd.Series(model5.feature\_importances\_, index=X\_features.columns)  
feat\_importances.nlargest(25).plot(kind='barh',figsize=(10,10))

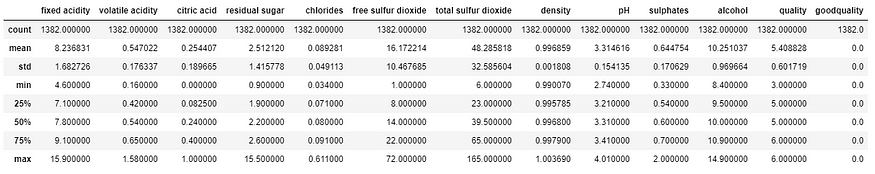


**Comparing the Top 4 Features**

# Filtering df for only good quality  
df\_temp = df[df['goodquality']==1]  
df\_temp.describe()# Filtering df for only bad quality  
df\_temp2 = df[df['goodquality']==0]  
df\_temp2.describe()



Good quality



Bad Quality

**By looking into the details, we can see that good quality wines have higher levels of alcohol on average, have a lower volatile acidity on average, higher levels of sulphates on average, and higher levels of residual sugar on average.**