CS249, Winter 2022

Project Report

Red and White Wine

By Team: Gohar Artenyan and
Lusine Artenyan

❖INTRODUCTION

Can the quality of wine be determined without using wine testing techniques? Can we use the predictive methods of machine learning to quantify the wine quality with scale and lesser expertise?
We will present the study of Red and White wine datasets: variants of the Portuguese Vinho Verde wine. It is Portugal's largest wine region and is the third biggest in Europe.

❖ *What is the submission format?*

1. Imported all important libraries and CSV files of Red and White wines. The data set was obtained from: https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/
2. Prepared datasets: at first, the names of all features were changed to capital letters in both Red and White wines CSV files, then checked if any null value existed, and reviewed the missing data for the Red and White datasets.
3. Checked outliers in datasets, the correlation between features.
4. Split the dataset into train and test data sets, scale it. Also, split into train-validation-test datasets for more analysis.
5. Completed some analysis, used algorithms Logistic Regression, Decision tree, kNN, and Ridge Regression, compared Logistic Regression, K-Nearest Neighbors, Random Forest Classifier, and Decision Tree performances.

So we will use cross-validation, and we will deal with many Machine Learning models. The problem will be treated with the Classification algorithm, where the output Quality variable has a discrete value.

The datasets are based on Red and White wine samples, in which there are 11 input attributes and one output attribute. The data attribute shows the physicochemical composition and the corresponding quality of each variety rated on a 10 point scale by wine tasters (10 = Excellent Quality, 0 = Bad Quality). We will look at each wine's input factors and predict wine ratings based on given attributes and features responsible for affecting the grade of the wine. We didn't compare a red wine to white wine since they are

different, not comparable (it cannot be said a white wine is better than red wine, or vice versa). And the last goal is to create Machine Learning models, find out what factors contribute to a good quality of Red and White wines, and make an analysis.

- *Deliverables*
- *Introduction to Dataset*

Both datasets have the same 12 features. The name and description of the 12 features are as follows:

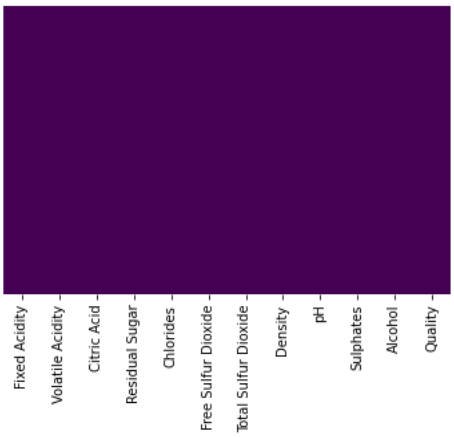
1. ***Fixed Acidity*** - Amount of Tartaric Acid in wine, measured in g/dm^3
2. ***Volatile Acidity*** - Amount of Acidic Acid in wine, which leads to unpleasant vinegar taste, measured in g/dm^3
3. ***Citric Acid*** - The amount of citric acid in wine in g/dm^3 contributes to the crispness of the wine, acts as a preservative to increase acidity.
4. ***Residual Sugar*** - the amount of sugar left in the wine after fermentation. Measured in g/dm3
5. ***Chlorides*** - the amount of Sodium Chloride(salt) in wine. Measured in g/dm^3
6. ***Free Sulfur Dioxide*** - Amount of SO_2 in free form. Prevents microbial growth and the oxidation of the wine. Measured in mg/dm^3
7. ***Total Sulfur Dioxide*** - Total amount of SO_2. Too much SO_2 can lead to a pungent smell. SO_2 acts as an antioxidant and antimicrobial agent.
8. ***Density*** - Density Wine in g/dm^3, sweeter wines have a higher density.
9. ***pH*** - pH of wine on a scale of 0 - 14. 0 means highly Acidic, while 14 means highly basic.
10. ***Sulphates*** - the amount of Potassium Sulphate in wine, measured in g/dm^3, contributes to the formation of SO_2. Acts as an antioxidant and antimicrobial
11. ***Alcohol*** - alcohol content in wine (in terms of % volume)
12. ***Quality*** - Wine Quality is graded on a scale of 1- 10 (Higher is better)

❖ *Dataset Preparation*

Red and White wine datasets, we have 1599 and 4898 data points, respectively. We checked if there are any missing values in the Red Wine dataset. The result is shown below, that there is no missing data:

```
1 sns.heatmap(red_df.isnull(),yticklabels=False, cbar=False, cmap='viridis')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1be0fbb3430>
```



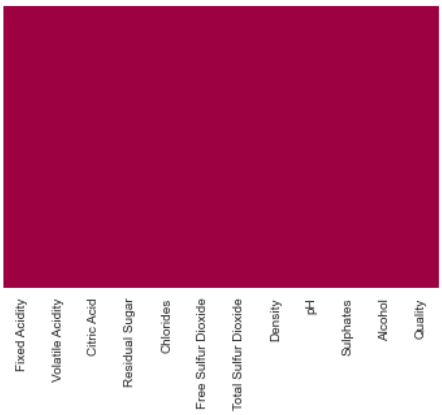

```
1 red_df.isnull().sum()
```

```
Fixed Acidity           0
Volatile Acidity        0
Citric Acid             0
Residual Sugar          0
Chlorides               0
Free Sulfur Dioxide     0
Total Sulfur Dioxide    0
Density                 0
pH                      0
Sulphates               0
Alcohol                 0
Quality                 0
dtype: int64
```

In White datasets as well, there is no missing data:

```
1 sns.heatmap(white_df.isnull(),yticklabels=False, cbar=False, cmap='Spectral')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1be0fe9d400>
```

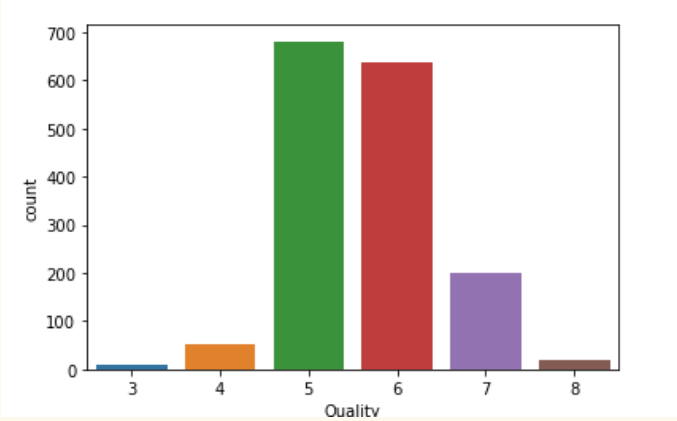



```
1 white_df.isnull().sum()
```

```
Fixed Acidity           0
Volatile Acidity        0
Citric Acid             0
Residual Sugar          0
Chlorides               0
Free Sulfur Dioxide     0
Total Sulfur Dioxide    0
Density                 0
pH                      0
Sulphates               0
Alcohol                 0
Quality                 0
dtype: int64
```

Based on a bar chart, most Red Wines are under rank 5 and 6, a few in 8 and 3, some in 4. We are processing descriptive statistics (see Appendix A), we can say that for Red Wine, there are no 0,1,2, and 9, 10 quality values. For Red wine, the maximum quality is 8.

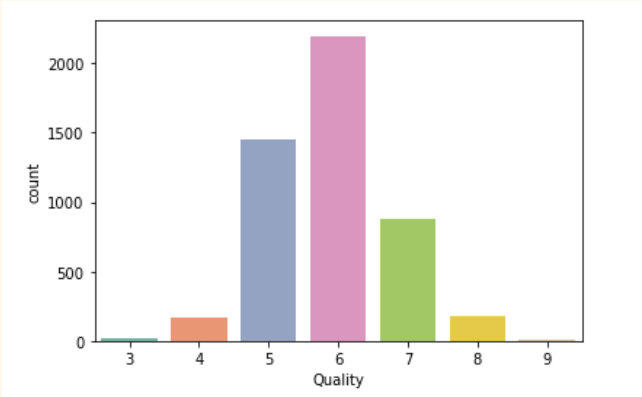
Red Wine Quality Counts




```
For Red WIne there are 10 different wines in Quality 3 wine
For Red WIne there are 53 different wines in Quality 4 wine
For Red WIne there are 681 different wines in Quality 5 wine
For Red WIne there are 638 different wines in Quality 6 wine
For Red WIne there are 199 different wines in Quality 7 wine
For Red WIne there are 18 different wines in Quality 8 wine
```

Almost similar results are for White Wines. There are more wines in rank 7 compared to Red wines.

White Wine Quality Count




```
For White Wines there are 20 different wines in Quality 3 wine
For White Wines there are 163 different wines in Quality 4 wine
For White Wines there are 1457 different wines in Quality 5 wine
For White Wines there are 2198 different wines in Quality 6 wine
For White Wines there are 880 different wines in Quality 7 wine
For White Wines there are 175 different wines in Quality 8 wine
For White Wines there are 5 different wines in Quality 9 wine
```

White wine has a maximum ranking of 9 and also has no 0,1,2. Based on the bar chart, we see that White Wine has a lot of data in 6, then comes 5, then 7. After checking outliers and correlation for Red and White wines, we have to convert all bad wines to 0s (we decided to call bad wines with qualities with 0 to 6) and convert all good wines to 1 (from 7 to 9). So, a column with 0 and 1 values for each Red and White dataset was added, and the Quality column was deleted (see Appendix A).

❖ *Outliers*

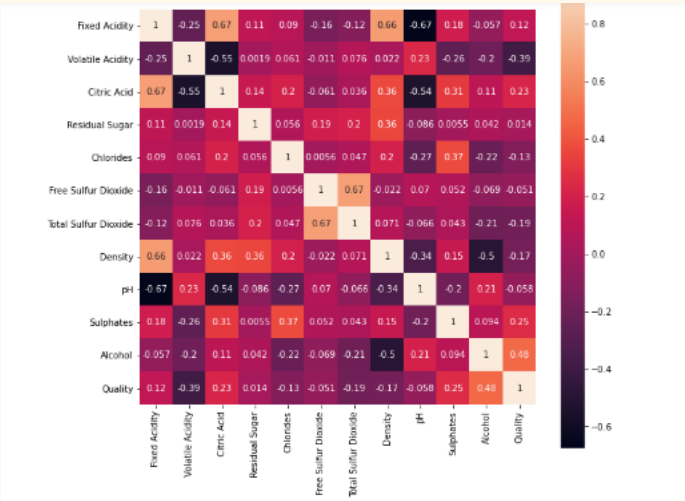
Outlier Data instances may change the outcome of our algorithms, so we want to delete such data, if possible. (See bar chart, boxplot analysis for outliers in Appendix A). Based on the descriptive representations of Red and White wines results, we see that high-quality wine may have outliers from other average quality wines. We do not want to delete too much data, and if a feature has a large number of outliers, we prefer not to use that feature. Based on *Descriptive Statistics (Appendix A)* the mean for the Red wine

is 5.63, and for White wine is 5.57. (Comparing these mean across the range in descriptive we see there's quite a variation for them. We have to scale them to make the standard deviation within a range). After checking the outfitters, there were no deleted features in the Red and White datasets, so we can not remove or modify outlier values in our datasets.

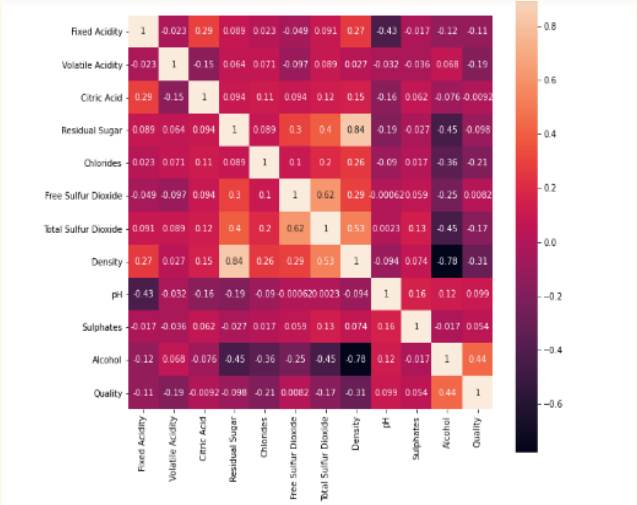
❖ *Correlation between features.*

We preferred to create a correlation matrix and visualize it among different methods for finding the correlation. We use Seaborn graphs to show the relationship between various features in the dataset.

When we checked the correlation between columns, we saw that some features strongly correlate with quality while others do not. The negative values get darker, for example, for



Correlation of different features of Red Wine dataset with quality:
Fixed Acidity : 0.1208
Volatile Acidity : -0.3906
Citric Acid : 0.2264
Residual Sugar : 0.0137
Chlorides : -0.1289
Free Sulfur Dioxide : -0.0507
Total Sulfur Dioxide : -0.1851
Density : -0.1749
pH : -0.0577
Sulphates : 0.2514
Alcohol : 0.4762
Quality : 1.0000



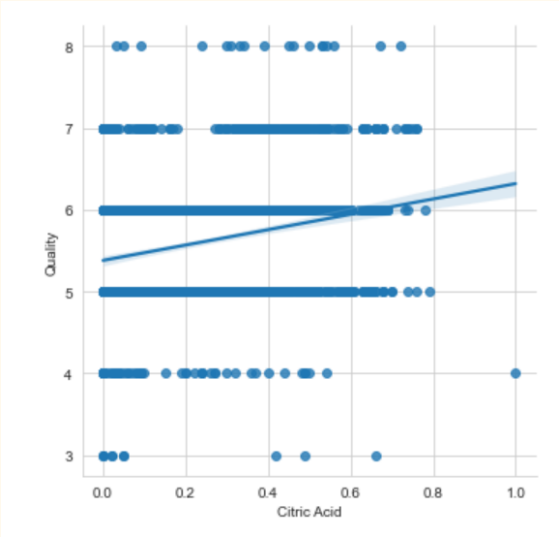
Correlation of different features of White Wine dataset with quality:
Fixed Acidity : -0.1137
Volatile Acidity : -0.1947
Citric Acid : -0.0092
Residual Sugar : -0.0976
Chlorides : -0.2099
Free Sulfur Dioxide : 0.0082
Total Sulfur Dioxide : -0.1747
Density : -0.3071
pH : 0.0994
Sulphates : 0.0537
Alcohol : 0.4356
Quality : 1.0000

Red Wine Volatile Acidity and Quality, the relationship is negatively correlated, and for Alcohol the correlation is positive.
From the above plots and values, we observe

- The Quality is correlated with Chlorides, Volatile Acidity, Residual Sugar, and Total Sulfur Dioxide. The Quality improves with a decrease in these values, and an increase in Alcohol, pH, Density. Based on our plots higher-quality Red Wines had

higher Sulphates, but the White Wines almost all ranks had an equal level of Sulphate, only levels 8 and 9 had less quantity of Sulphate.

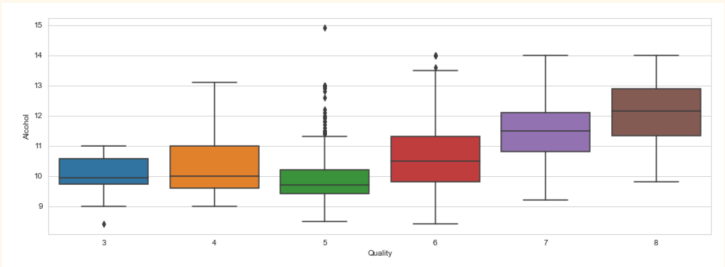
- Quality is strongly correlated with Alcohol, Density, pH, Citric Acid which are positively correlated.
- Total Sulfur Dioxide and Free Sulfur Dioxide are highly correlated to each other.
- Features with <0.5 correlated to Quality can be removed from the dataset.

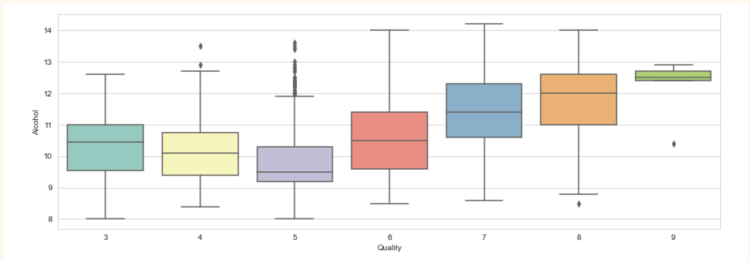


- As we can see in the plot shown on the left, the quality of wine increases along with the amount of Citric Acid, which shows it is an important feature for grading the wine. (Please, also see Appendix E for more correlations between the features.)

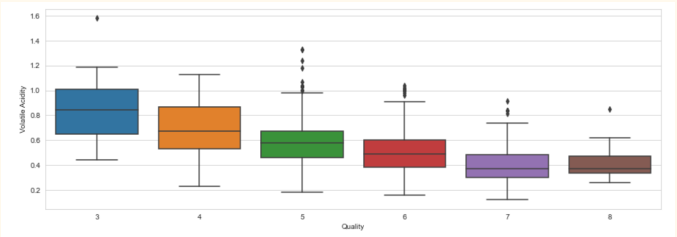
High-quality wine may have a very rare composition from other average-quality wines. We need to consider every feature while predicting the Quality of wine, so we choose not to remove any feature from the dataset.

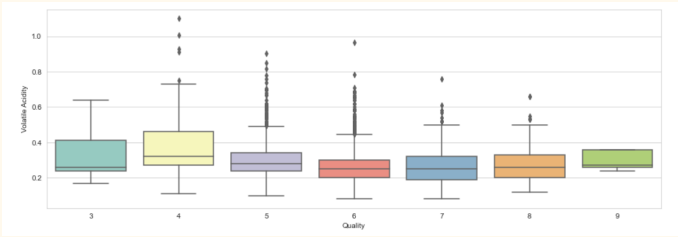
If we look at bar charts to check the Alcohol and Quality, Red and White Wines respectively, higher quality wines the median alcohol is higher.





In the same way, we can see the negative relationship between Quality and Volatile Acidity:





For more Features Correlation visual analysis representation see Appendix A.

❖ Data Processing

After converting the Quality column to 0s and 1s for both Red and White datasets, features and label separation into two different variables (for both Red and White datasets). After that, a Cross-validation simple split was made by dividing each Red and

```
Red Wine X: 12 features x 1599 entires
Red Wine y: 1599 data entries
```

```
Red Wine training: 1119, Red Wine testing: 480
```

White dataset into train/test using a 70 / 30 split.

```
White Wine X: 12 features x 4898 entires
White Wine y: 4898 data entries
```

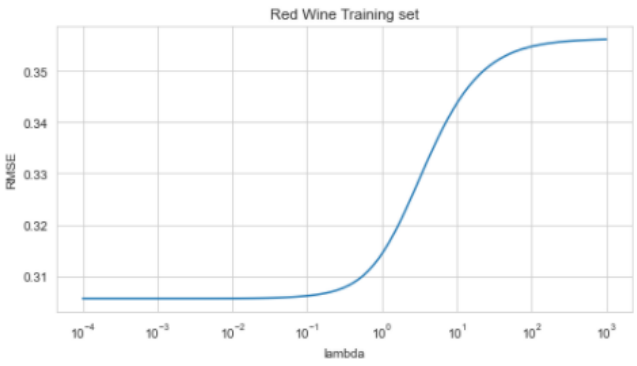
```
White Wine training: 3428, White Wine testing: 1470
```

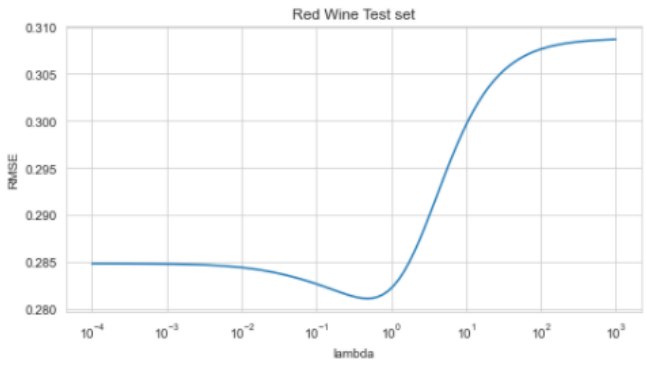
❖ Ridge Regression

After this processing, we fitted Ridge Regression to the train data for the different values of the regularization hyperparameters. We computed the RMSE of the fitted model in the training and test data. Then draw train and test RMSE for different values of the regularization hyperparameter. The Red and White wines plots were drowned (as 1. Training set/ RMSE, and 2. Test set / RMSE).

For Red Wine in case 1., when lambda is very small, there is no regularization(training data set error is small, the fitting to training data is well performed), but in case 2, lambda is larger. When we increase the regularization parameter, training error will increase, and as we are reducing overfitting, we are improving generalization error. So test error will decrease, but some point forward will grow again.

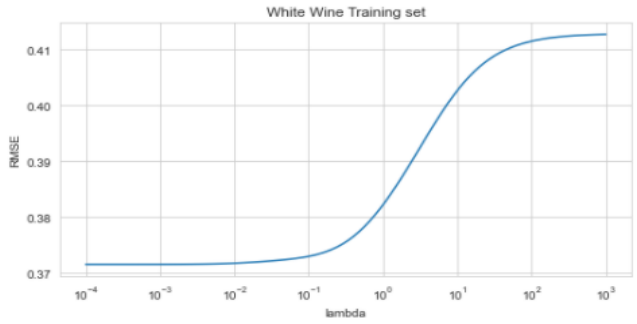
```
(4.466835921509635e-05,
 2238.72113856834,
 0.27969763974392103,
 0.3100447296384591)
```

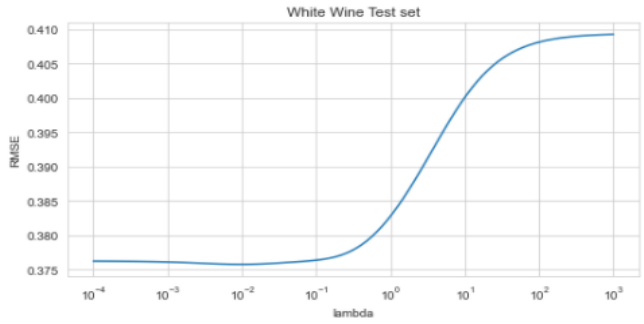




For White wine, Test and Training representation looks similar.

```
(4.466835921509635e-05,
 2238.72113856834,
 0.3740823653048619,
 0.4109507418898767)
```





❖ *Logistic Regression*

Then we wanted to fit the Logistic Regression model to Red Wine data and White Wine data. So Logistic Regression has a linear boundary, and to show that we used ML Algorithm through sklearn libraries. So we defined Logistic Regression model, and fit it into our training data. For the 70/30 simple split there was computed Red and White wines Logistic Regression accuracies.

```
Red Wine Logisitic Regression accuracy: 86.67%
```

```
White Wine Logisitic Regression accuracy: 80.75%
```

When 70/15/15 split was made, we came back to Logistic Regression and made some more analisis. For each Red and White wines we generated 0-1 predictions in the train dataset and validation dataset, also predicted the probabilities by using predict_proba method. Predict_proba generates the list of probabilities (we were chosen to have second column of the raws which correspond to probability of a positive sample).

```
For Red Wine first four 0-1 predictions for the validation set:
[0 0 0 0]
For Red Wine first four predicted probabilities for positive label in the validation set:
[0.03320907 0.37504025 0.21119591 0.0115025 ]
For Red Wine Logistic Reression accuracy:92.5%
```

So first four 0-1 predictions for Red wine validation set all of them are 0, the first four predicted probabilities for positive label in the validation set values all of them are less than 0.5 (that was a threshold to generate these score).

Red Wine Logistic Regression accuracy is 92.5% ,which is bigger than its 70/30 split accuracy.

```
For White Wine first four 0-1 predictions for the validation set:
[0 0 0 0]
For White Wine first four predicted probabilities for positive label in the validation set:
[0.36142962 0.58451514 0.09761169 0.24007144]
For White Wine Logistic Reression accuracy for White Wine:78.23%
```

First four 0-1 predictions for the White wine validation set are 0, and the first four predicted probabilities for the positive label in the validation set there is 0.58 number, which is more than 0.5. White Wine Logistic Regression accuracy is 78.23%, which is less than its 70/30 split accuracy(this can be explained that White wine dataset is bigger than red Wine dataset, that's why in not simple split accuracy the Red Wine performed better result, than White wine).

Please, see *Appendix B*.

❖ Decision Tree

For Decision Tree we used sklearn, there were used criterion = entropy, and maximum depth 3, and minimum sample split was chosen 100. We fitted a model to training data. After there was generated 0-1 prediction,and

```
For Red Wine first four 0-1 predictions for the validation set:
[0 1 0 0]
For Red Wine first four predicted probabilities for positive label in the validation set:
[0.0374415  0.50243902 0.14652015 0.0374415 ]
```

```
For White Wine first four 0-1 predictions for the validation set:
[0 0 0 0]
For White Wine first four predicted probabilities for positive label in the validation set:
[0.3954608  0.3954608  0.04189189 0.3954608 ]
```

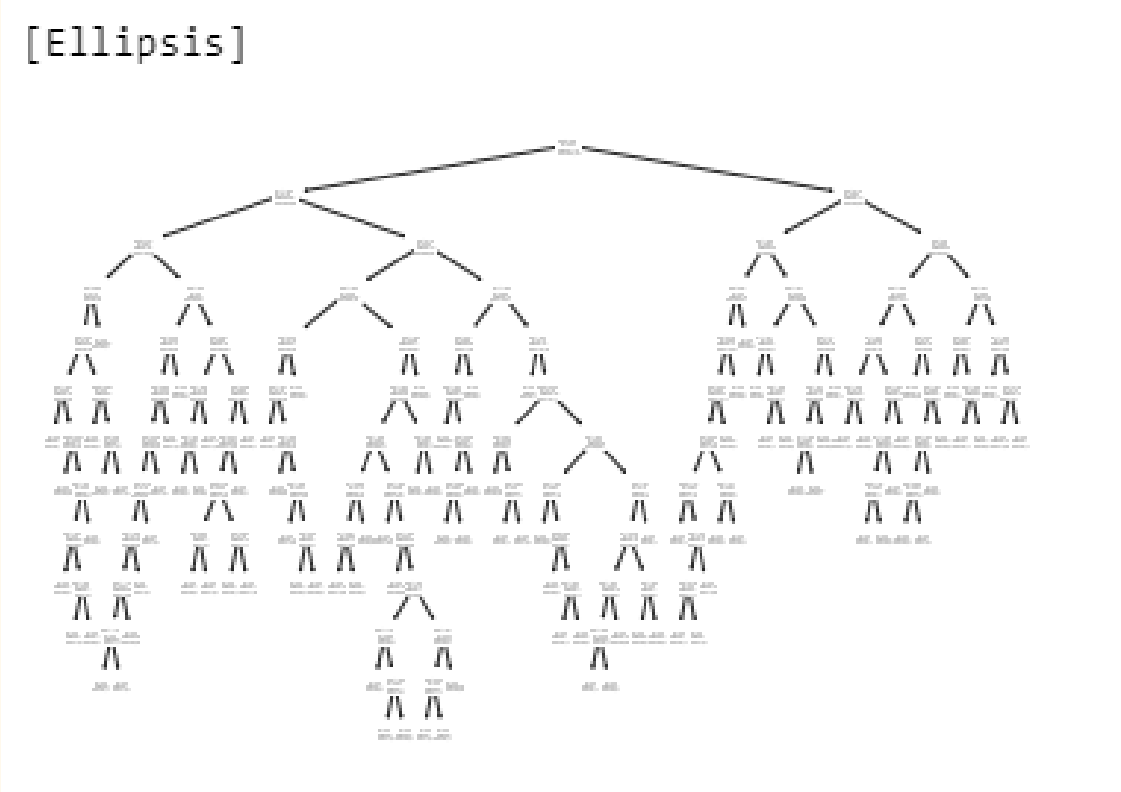
probabilities. The results for Red and White wines are below.

Also, based on 70/30 split there were calculated Red and Wite wines decision tree classification score, and generated their decision trees (please, see below)

```
Red Wine decision tree classifier score is:

0.8854166666666666
```

```
White Wine decision tree classifier score is:

0.8770833333333333
```

Red Wine Decision Tree



White Wine Decision Tree

Please, see Appendix C.

❖ *K-Nearest Neighbors (KNN)*

In KNN, an observation is classified based on the class of its K-nearest neighbors. It's a valuable model when the decision boundary is non-linear, but it will not tell us which essential predictors are. There were made some metrics calculations for Red and White wine models, see below.

For Red Wine k = 3, percent accuracy 88.54%

	precision	recall	f1-score	support
0	0.95	0.92	0.94	430
1	0.46	0.56	0.50	50
accuracy			0.89	480
macro avg	0.70	0.74	0.72	480
weighted avg	0.90	0.89	0.89	480

```
Red Wine Train RMSE of the linear regression model is:  0.2623193450446023
Red Wine Test RMSE of the linear regression model is:  0.31622776601683794
```

For the Red wine train, RMSE is less than test RMSE, but since the gap between these RMSE values is not too big, then the model fitted to training data somehow generalizes to test data.

```
For Red Wine nearest neighbor 1, quality: 100.0%
For Red Wine nearest neighbor 2, quality: 93.66%
For Red Wine nearest neighbor 3, quality: 93.12%
For Red Wine nearest neighbor 4, quality: 90.17%
For Red Wine nearest neighbor 5, quality: 89.81%
For Red Wine kNN with 1 nearest neighbor percent accuracy: 100.0%
```

For White Wine k = 3, percent accuracy 82.38%

	precision	recall	f1-score	support
0	0.87	0.91	0.89	1157
1	0.60	0.51	0.55	313
accuracy			0.82	1470
macro avg	0.74	0.71	0.72	1470
weighted avg	0.82	0.82	0.82	1470

RMSE tells us about the model's performance here for White wine RMSE in train data less than RMSE in test data. But since the gap between them is not too big, the model fitted to training data somehow generalizes to test data. But model fits much better to trained data compared to test data.

```
White Wine Train RMSE of the linear regression model is:  0.29335304555243524
White Wine Test RMSE of the linear regression model is:  0.43799170619302547
```

```
For White Wine nearest neighbor 1, quality: 100.0%
For White Wine nearest neighbor 2, quality: 92.56%
For White Wine nearest neighbor 3, quality: 91.39%
For White Wine nearest neighbor 4, quality: 89.61%
For White Wine nearest neighbor 5, quality: 88.89%
For White Wine kNN with 1 nearest neighbor percent accuracy: 100.0%
```

For the 70/15/15 split

```
For Red Wine first four 0-1 predictions for the validation set:
[0 0 0 0]
For Red Wine first four predicted probabilities for positive label in the validation set:
[0.03320907 0.37504025 0.21119591 0.0115025 ]
For Red Wine Logistic Reression accuracy:92.5%
```

```
For White Wine first four 0-1 predictions for the validation set:
[0 0 0 0]
For White Wine first four predicted probabilities for positive label in the validation set:
[0.36142962 0.58451514 0.09761169 0.24007144]
For White Wine Logistic Reression accuracy for White Wine:78.23%
```

We run a total comparison for algorithms we used, so For Red Wine.
For each classification model, we analyzed how the results vary. The study includes the analysis of classifiers on both red and white wine data sets. Different classifiers like Logistic Regression, K-Nearest-Neighbors, Random Forests, and Decision Tree. Results from the below chart lead us to conclude that for

- Red Wine Random Forests Algorithm performs better than the other presented classification tasks.

```
Red Wine models selection based on the model names in an array
```

	Algorithm	Accuracy
0	Logistic Regression	0.877083
1	K-Nearest Neighbors	0.875000
2	Random Forest Classifier	0.904167
3	Decision Tree	0.875000

- White Wine Logistic regression performs better compared to other presented classification tasks.

White Wine models selection based on the model names in an array

	Algorithm	Accuracy
0	Logistic Regression	0.721088
1	K-Nearest Neighbors	0.681633
2	Random Forest Classifier	0.666667
3	Decision Tree	0.656463

Please, see Appendix F.

❖Conclusion

From the Machine Learning and Engineering Classification approaches, we can conclude:

As we were looking for accuracy, not for approximation (high-quality wine may have a very rare composition from other average quality wines), we chose not to remove outliers and extract more relevant features from both datasets.

For more comparison of the Quality and Feature correlations, please see Appendix E. We might or will get different results if we remove the outliers and consider a feature extraction. More data we can use more accurate models would be. In this case, even though we had a lot of data, 1599 Red Wine, and 4898 data points for White Wine, which was relatively a good amount of data, the scale of it is quite small compared to what we might consider big data.

Models like Logistic Regression, KNN, Decision Tree, and Ridge regression were used to predict the Quality of Red and White wines. We have also calculated their performance metrics-based classifiers, and it was found that these classifiers performed well. Logistic Regression performed good accuracy results for Red wine in 70/15/15 split, although in a comparison of Classification models (for 70/30 split), Random Forrest Classification model gave the best accuracy result for the Red Wine dataset and was considered as a good model for predicting the Red Wine quality problem, and in the same way, Logistic Regression was considered as a good model for predicting the White Wine quality problem.

Appendix A.

Descriptive Statistics

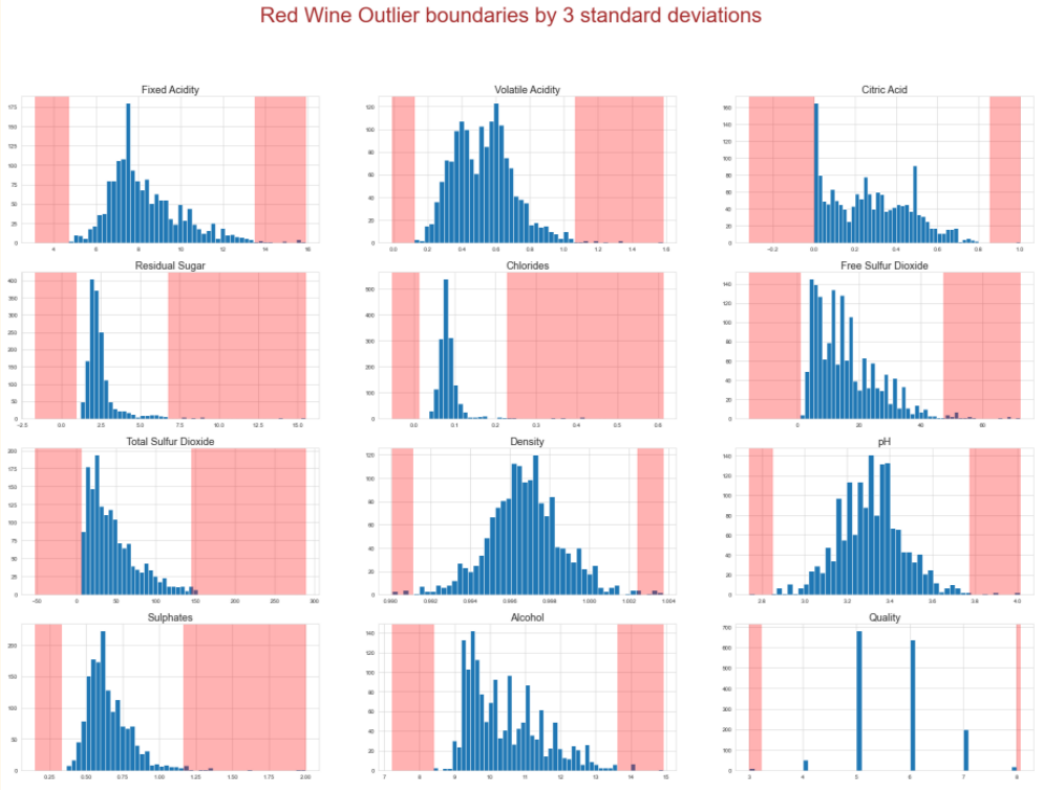
```
#For White Wine descriptive statistics for all he variables
white_df.describe()
```

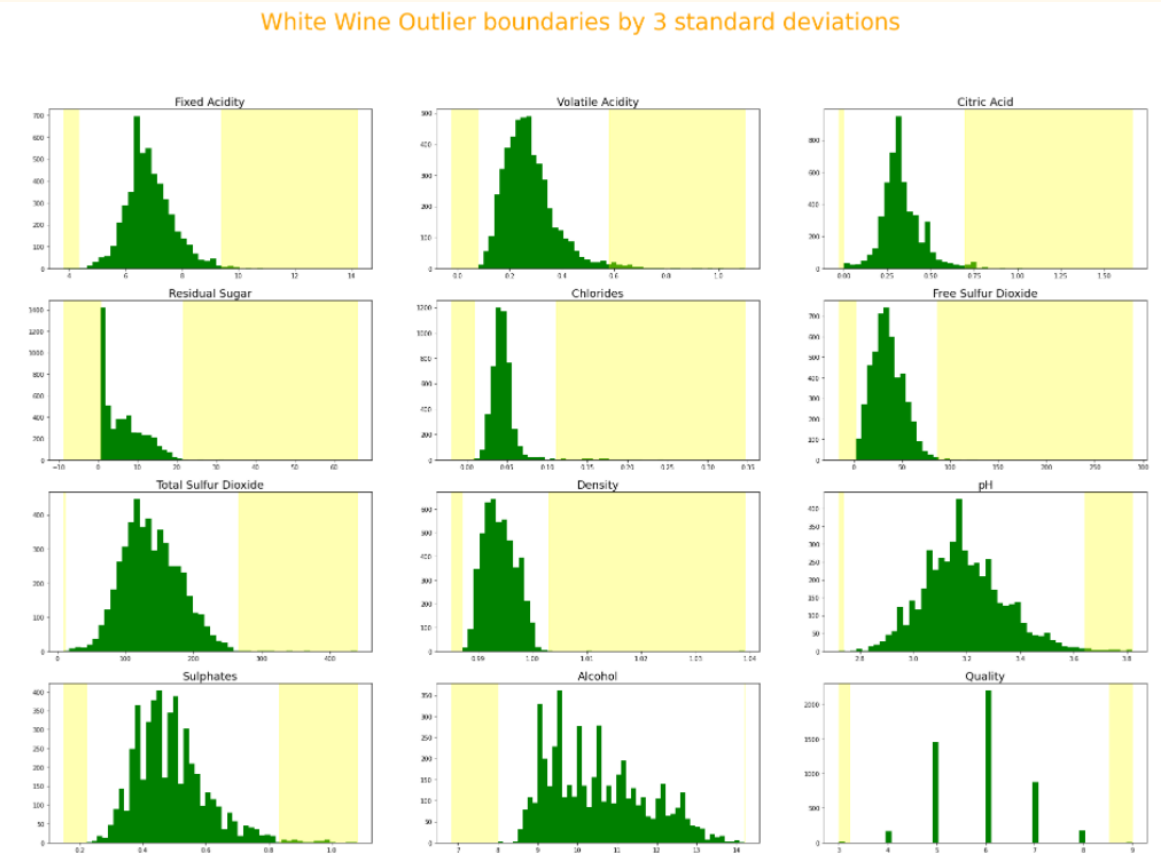
	Fixed Acidity	Volatile Acidity	Citric Acid	Residual Sugar	Chlorides	Free Sulfur Dioxide	Total Sulfur Dioxide	Density	pH	Sulphates	Alcohol	C
count	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.0
mean	6.854788	0.278241	0.334192	6.391415	0.045772	35.308085	138.360657	0.994027	3.188267	0.489847	10.514267	5.8
std	0.843868	0.100795	0.121020	5.072058	0.021848	17.007137	42.498065	0.002991	0.151001	0.114126	1.230620	0.8
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9.000000	0.987110	2.720000	0.220000	8.000000	3.0
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	108.000000	0.991723	3.090000	0.410000	9.500000	5.0
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	134.000000	0.993740	3.180000	0.470000	10.400000	6.0
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	167.000000	0.996100	3.280000	0.550000	11.400000	6.0
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	440.000000	1.038980	3.820000	1.080000	14.200000	9.0

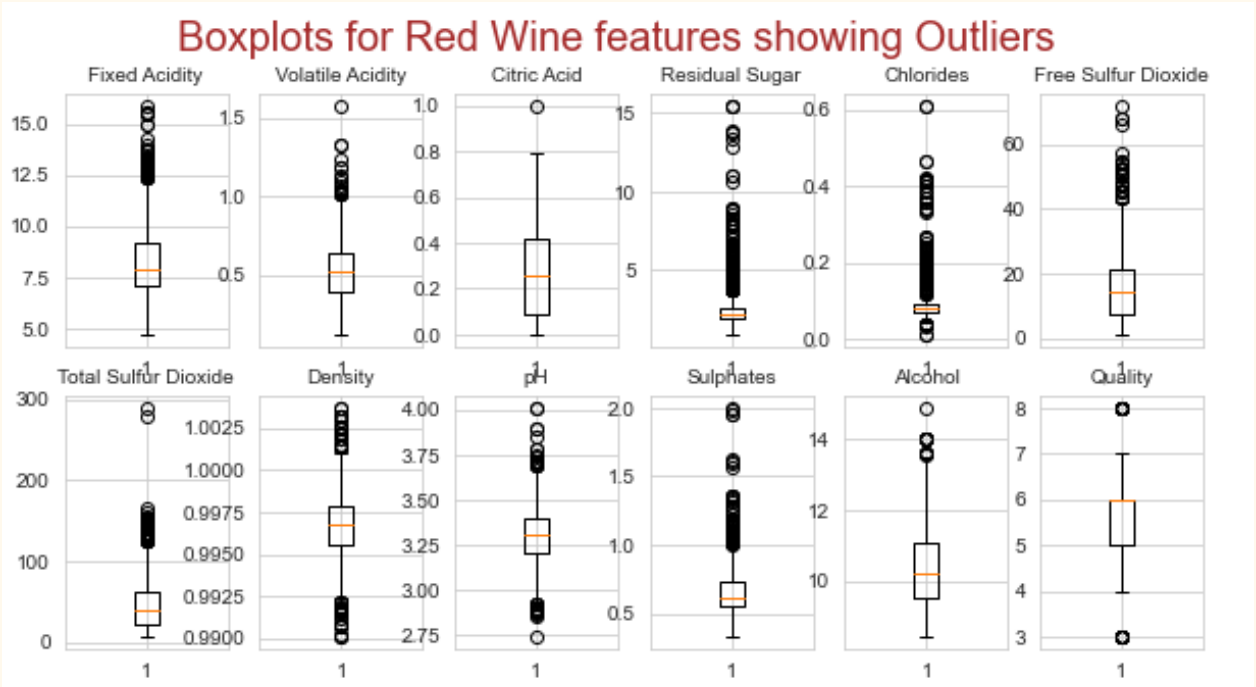
```
#For Red Wine descriptive statistics for all he variables
red_df.describe()
```

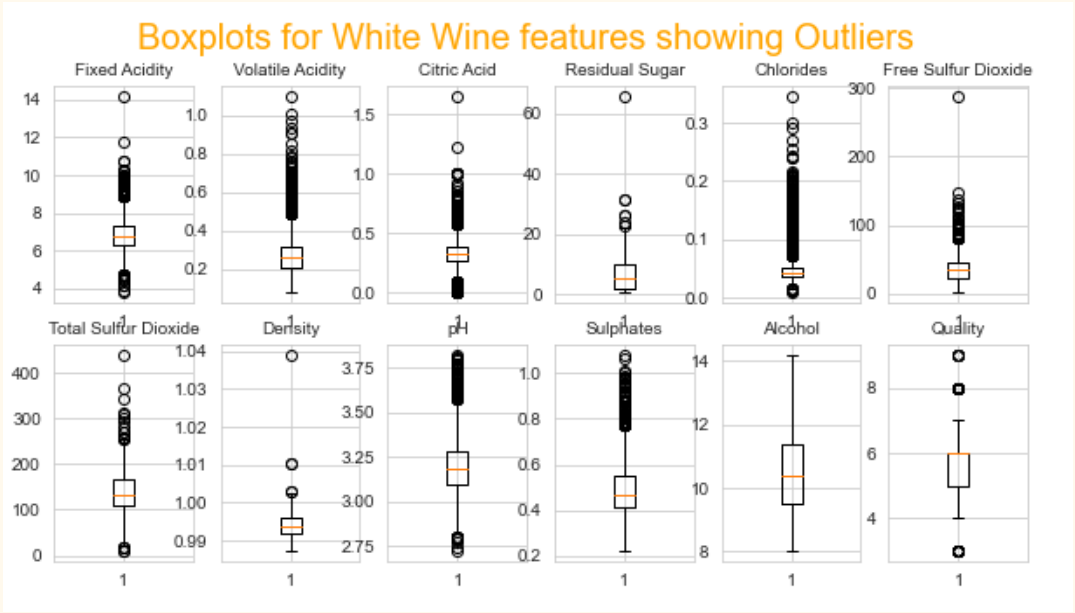
	Fixed Acidity	Volatile Acidity	Citric Acid	Residual Sugar	Chlorides	Free Sulfur Dioxide	Total Sulfur Dioxide	Density	pH	Sulphates	Alcohol	C
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.0
mean	8.315260	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113	0.658149	10.422983	5.6
std	1.733973	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386	0.169507	1.065668	0.8
min	4.700000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.0
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000	0.550000	9.500000	5.0
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000	0.620000	10.200000	6.0
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000	0.730000	11.100000	6.0
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.0

Dataset Preparation Outlier




White Wine Outlier boundaries by 3 standard deviations
Fixed Acidity
Volatile Acidity
Citric Acid
Residual Sugar
Chlorides
Free Sulfur Dioxide
Total Sulfur Dioxide
Density
pH
Sulphates
Alcohol
Quality


Boxplots for Red Wine features showing Outliers
Fixed Acidity
Volatile Acidity
Citric Acid
Residual Sugar
Chlorides
Free Sulfur Dioxide
Total Sulfur Dioxide
Density
pH
Sulphates
Alcohol
Quality




```
#For Red Wine
# Is the Red Wine acceptable? 1 = True, 0 = False

red_df['Score'] = red_df['Quality'].map( { 0:0,\
                                           1:0,\
                                           2:0,\
                                           3:0,\
                                           4:0,\
                                           5:0,\
                                           6:0,\
                                           7:1,\
                                           8:1,\
                                           9:1,\
                                           10:1 \
                                         } )
```

```
#For Red Wine change data into correct form

df_mod = red_df.drop(['Quality', 'Score'], axis=1)
X = df_mod.values
y = red_df['Score'].to_numpy()

print(f"Red Wine X: {len(X[1])} features x {len(X)} entires")
print(f"Red Wine y: {len(y)} data entries")
```

```
Red Wine X: 11 features x 1599 entires
Red Wine y: 1599 data entries
```

```
#For White Wine
# Is the White Wine acceptable? 1 = True, 0 = False

white_df['Score1'] = white_df['Quality'].map( { 0:0,\
                                                1:0,\
                                                2:0,\
                                                3:0,\
                                                4:0,\
                                                5:0,\
                                                6:0,\
                                                7:1,\
                                                8:1,\
                                                9:1,\
                                                10:1 \
                                              } )
```

```
#For White Wine change data into correct form

df_mod1 = white_df.drop(['Quality', 'Score1'], axis=1)
X1 = df_mod1.values
y1 = white_df['Score1'].to_numpy()

print(f"White Wine X: {len(X1[1])} features x {len(X1)} entires")
print(f"White Wine y: {len(y1)} data entries")
```

```
White Wine X: 11 features x 4898 entires
White Wine y: 4898 data entries
```

Appendix B.

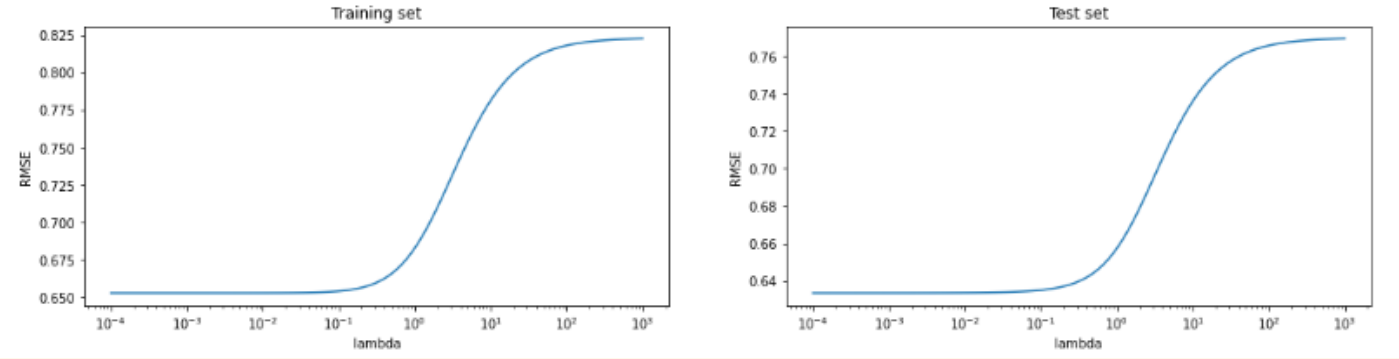
Ridge Regression

```
#Apply Ridge
from sklearn.linear_model import Ridge
```

```
#Ridge Regression for Red Wine
```

```
rmses_train=[]
rmses_test =[]
lambdas = np.logspace(-4,3,100)

for l in lambdas:
    model = Ridge(normalize=True)
    model.set_params(alpha=l)
    model.fit(Xr_train, yr_train)

    preds_train =model.predict(Xr_train)
    rmse_train = np.sqrt(np.mean((yr_train -preds_train)**2))

    preds_test =model.predict(Xr_test)
    rmse_test = np.sqrt(np.mean((yr_test -preds_test)**2))

    rmses_train.append(rmse_train)
    rmses_test.append(rmse_test)
```

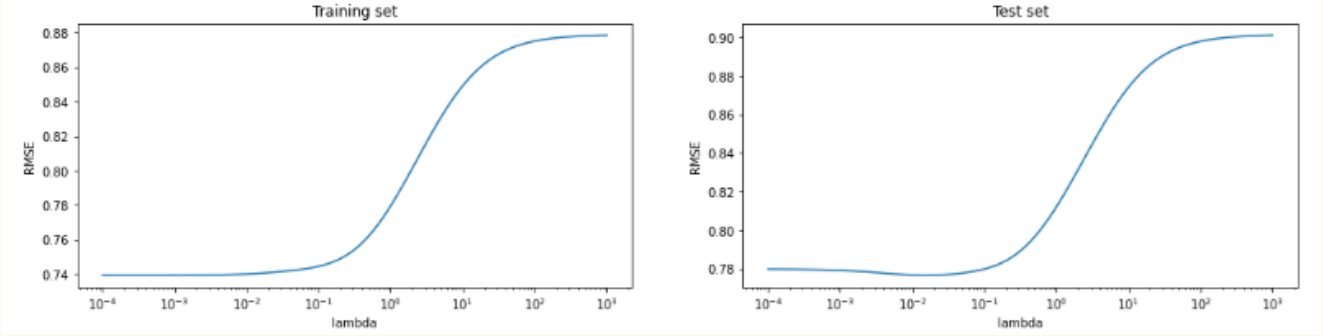
```
plt.figure(figsize=(18, 4))

plt.subplot(121)
ax = plt.gca()
ax.plot(lambdas, rmses_train)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Training set")
plt.axis("tight")

plt.subplot(122)
ax = plt.gca()
ax.plot(lambdas, rmses_test)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Test set")
plt.axis("tight")

plt.show()
```



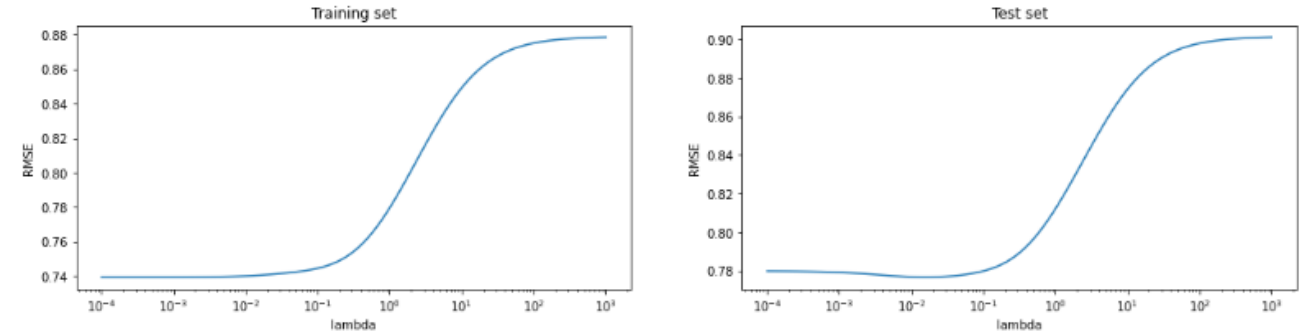

```
#Ridge Regression for White Wine
```

```
rmses_trainw=[]
rmses_testw =[]
lambdas = np.logspace(-4,3,100)

for l in lambdas:
    model_w = Ridge(normalize=True)
    model_w.set_params(alpha=l)
    model_w.fit(Xw_train, yw_train)

    preds_trainw =model_w.predict(Xw_train)
    rmse_trainw = np.sqrt(np.mean((yw_train -preds_trainw)**2))

    preds_testw =model_w.predict(Xw_test)
    rmse_testw = np.sqrt(np.mean((yw_test -preds_testw)**2))

    rmses_trainw.append(rmse_trainw)
    rmses_testw.append(rmse_testw)
```

```
plt.figure(figsize=(18, 4))

plt.subplot(121)
ax = plt.gca()
ax.plot(lambdas, rmses_trainw)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Training set")
plt.axis("tight")

plt.subplot(122)
ax = plt.gca()
ax.plot(lambdas, rmses_testw)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Test set")
plt.axis("tight")

plt.show()
```



```
plt.figure(figsize=(18, 4))

plt.subplot(121)
ax = plt.gca()
ax.plot(lambdas, rmses_trainw)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Training set")
plt.axis("tight")

plt.subplot(122)
ax = plt.gca()
ax.plot(lambdas, rmses_testw)
ax.set_xscale("log")
plt.xlabel("lambda")
plt.ylabel("RMSE")
plt.title("Test set")
plt.axis("tight")

plt.show()
```



Split Red and White datasets to 70 / 15 / 15 as for train - validation - test

```
#Training and Predicting
#Logistic Regression for Red Wine
```

```
model_lr=LogisticRegression(
    solver='liblinear',
    penalty= 'l2',
    C=0.5
)
model_lr.fit(Xr_train, yr_train)

preds_train = model_lr.predict(Xr_train)
preds_val = model_lr.predict(Xr_val)
print("0-1 predictions for valid set:")
print(preds_val[0:4])

probs_train = model_lr.predict_proba(Xrtest)[:,1]
probs_val = model_lr.predict_proba(Xr_val)[:,1]
print("predicted probabilities for positive label in valid set:")
print(probs[0:4])

percent = round(100* model_lr.score(Xr_test,yr_test), 2)
print(f"Logisitic Regression accuracy for Red wine: {percent}%")
```

```
0-1 predictions for valid set:
[6 6 6 5]
predicted probabilities for positive label in valid set:
[[0.00667411 0.10529574 0.40100689 0.38348819 0.08719595 0.01155088
  0.00478823]
 [0.00678644 0.0403942  0.47117427 0.42306459 0.04706717 0.00934565
  0.00216768]
 [0.00264061 0.01298607 0.20602844 0.57175915 0.16797437 0.0314148
  0.00719656]
 [0.00262226 0.00096845 0.03427627 0.43164621 0.35086478 0.1738159
  0.00580613]]
Logisitic Regression accuracy for Red wine: 62.29%
```

```
#Split Red Wine to 70 / 15 / 15 as train - validation - test datasets
```

```
Xr_val, Xrtest, yr_val, yrtest = train_test_split(
   Xr_test, yr_test, test_size =0.5,
   shuffle=True,
   random_state=0
)
print("shape of X_train is: ", Xr_train.shape)
print("shape of X_val is: ", Xr_val.shape)
print("shape of X_test is: ", Xrtest.shape)
```

```
shape of X_train is:  (1119, 11)
shape of X_val is:  (240, 11)
shape of X_test is:  (240, 11)
```

```
#Training and Predicting
#Logistic Regression for White Wine
```

```
model_l=LogisticRegression(
    solver='liblinear',
    penalty= 'l2',
    C=0.5
)
model_l.fit(Xw_train, yw_train)

preds_train = model_l.predict(Xw_train)
preds_val = model_l.predict(Xw_val)
print("0-1 predictions for valid set:")
print(preds_val[0:4])

probs_train = model_l.predict_proba(Xwtest)[:,1]
probs_val = model_l.predict_proba(Xw_val)[:,1]
print("predicted probabilities for positive label in valid set:")
print(probs[0:4])

percent1 = round(100* model_l.score(Xwtest,ywtest), 2)
print(f"Logisitic Regression accuracy for White wine: {percent1}%")
```

```
0-1 predictions for valid set:
[6 7 5 6]
predicted probabilities for positive label in valid set:
[[0.00667411 0.10529574 0.40100689 0.38348819 0.08719595 0.01155088
  0.00478823]
 [0.00678644 0.0403942  0.47117427 0.42306459 0.04706717 0.00934565
  0.00216768]
 [0.00264061 0.01298607 0.20602844 0.57175915 0.16797437 0.0314148
  0.00719656]
 [0.00262226 0.00096845 0.03427627 0.43164621 0.35086478 0.1738159
  0.00580613]]
Logisitic Regression accuracy for White wine: 50.61%
```

Appendix C.

Decision Tree

```
#Split White Wine to 70 / 15 / 15 as train - validation - test datasets
```

```
Xw_val, Xwtest, yw_val, ywtest = train_test_split(
   Xw_test, yw_test, test_size =0.5,
   shuffle=True,
   random_state=0
)
print("shape of X_train is: ", Xw_train.shape)
print("shape of X_val is: ", Xw_val.shape)
```

```
shape of X_train is:  (3428, 11)
shape of X_val is:  (735, 11)
```

```
#Decsion Free for Red Wine
```

```
model_dt = tree.DecisionTreeClassifier(
    criterion="entropy",
    max_depth=3,
    min_samples_split=100,
    random_state=0
)
model_dt.fit(Xr_train, yr_train)

preds_train = model_dt.predict(Xr_train)
preds_val = model_dt.predict(Xr_val)
print("First four 0-1 predictions for the validation set")
print(preds_val[0:4])

probs_train = model_dt.predict_proba(Xr_train)[:, 1]
probs_val = model_dt.predict_proba(Xr_val)[:, 1]
print("First four predicted probabilities for positive labele in the validation set")
print(probs_val[0:4])
```

```
First four 0-1 predictions for the validation set
[6 6 6 6]
First four predicted probabilities for positive labele in the validation set
[0.0215311  0.         0.02739726 0.0215311 ]
```

```
#Decsion Free for White Wine
```

```
model_d = tree.DecisionTreeClassifier(
    criterion="entropy",
    max_depth=3,
    min_samples_split=100,
    random_state=0
)
model_d.fit(Xw_train, yw_train)

pred_train = model_d.predict(Xw_train)
pred_val = model_d.predict(Xw_val)
print("First four 0-1 predictions for the validation set")
print(preds_val[0:4])

prob_train = model_d.predict_proba(Xw_train)[:, 1]
prob_val = model_d.predict_proba(Xw_val)[:, 1]
print("First four predicted probabilities for positive labele in the validation set")
print(probs_val[0:4])
```

```
First four 0-1 predictions for the validation set
[6 7 5 6]
First four predicted probabilities for positive labele in the validation set
[0.01853758 0.00498025 0.01493682 0.02876232]
```

Appendix D.

K-Nearest Neighbors (KNN)

```
#K-Nearest Neighbors, RMSE for k=3
from sklearn.neighbors import KNeighborsRegressor
```

```
#K-Nearest Neighbors, RMSE for Red Wine
```

```
Red_model = KNeighborsRegressor(n_neighbors=3)
Red_model.fit(Xr_train, yr_train)
Red_preds_train = Red_model.predict(Xr_train)
Red_rmse_train = np.sqrt(np.mean((yr_train - Red_preds_train)**2))

Red_preds_test = Red_model.predict(Xr_test)
Red_rmse_test = np.sqrt(np.mean((yr_test - Red_preds_test)**2))

print("Train RMSE of the KNN for Red Wine model is: ", Red_rmse_train)
print("Test RMSE of the KNN regression model for Red Wine is: ", Red_rmse_test)
```

```
Train RMSE of the KNN for Red Wine model is:  0.4631168446474912
Test RMSE of the KNN regression model for Red Wine is:  0.7155676327483228
```

```
#K-Nearest Neighbors, RMSE for White Wine
```

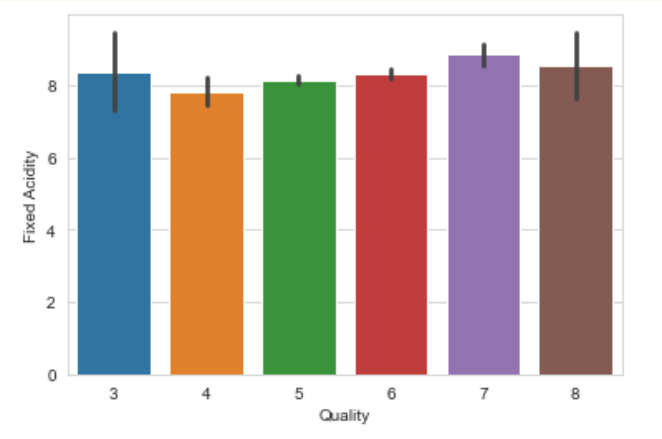
```
White_model = KNeighborsRegressor(n_neighbors=3)
White_model.fit(Xw_train, yw_train)
White_preds_train = White_model.predict(Xw_train)
White_rmse_train = np.sqrt(np.mean((yw_train - White_preds_train)**2))

White_preds_test = White_model.predict(Xw_test)
White_rmse_test = np.sqrt(np.mean((yw_test - White_preds_test)**2))

print("Train RMSE of the KNN for White Wine model is: ", White_rmse_train)
print("Test RMSE of the KNN regression model for White Wine is: ", White_rmse_test)
```

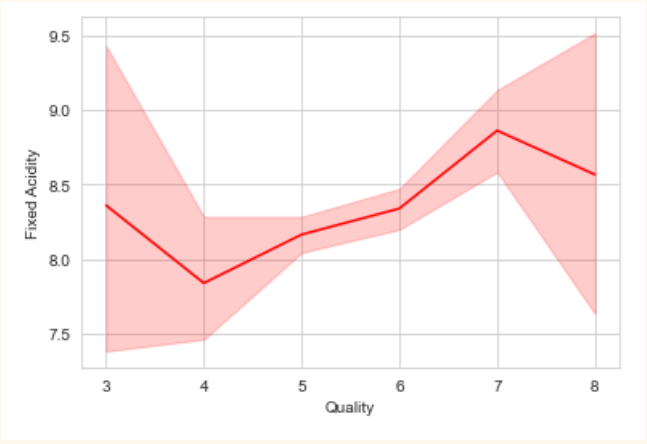
```
Train RMSE of the KNN for White Wine model is:  0.49143447569325716
Test RMSE of the KNN regression model for White Wine is:  0.7538262188687572
```

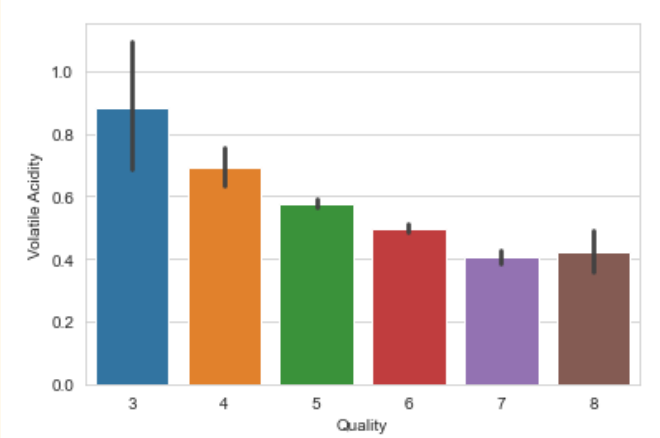
Appendix E.

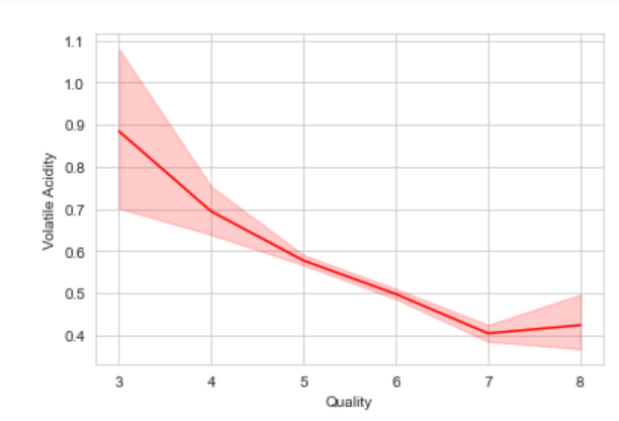
Wine Quality vs Features

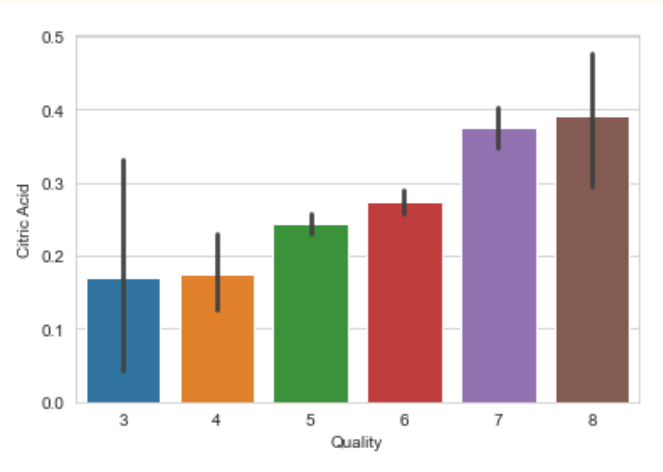
Red Wine

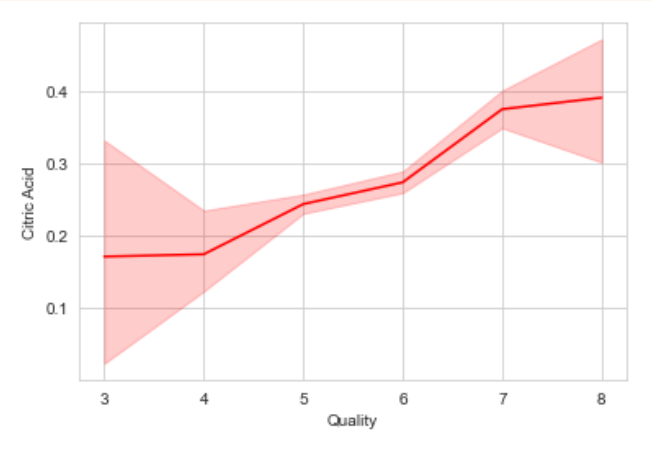


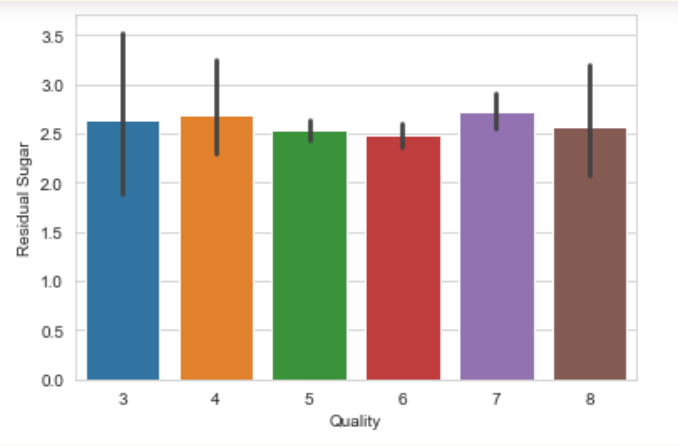


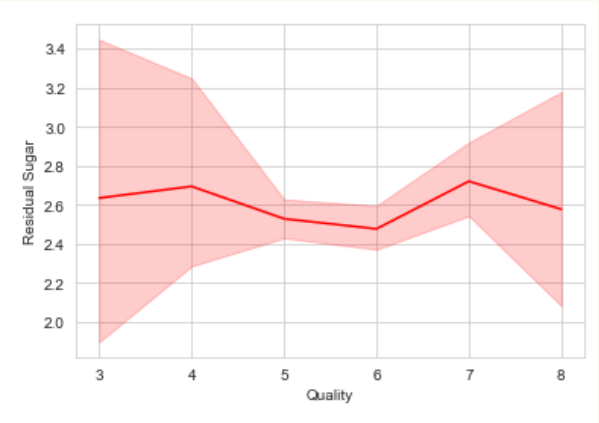

Volatile Acidity
1.0
0.8
0.6
0.4
0.2
0.0
3
4
5
6
7
8
Quality

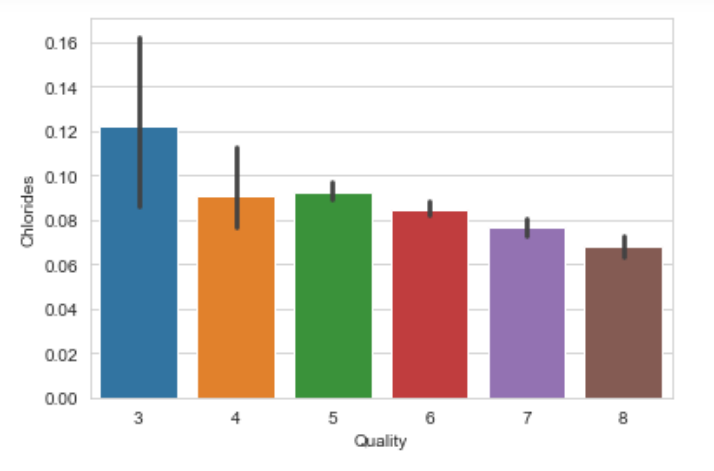

Volatile Acidity
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
3
4
5
6
7
8
Quality

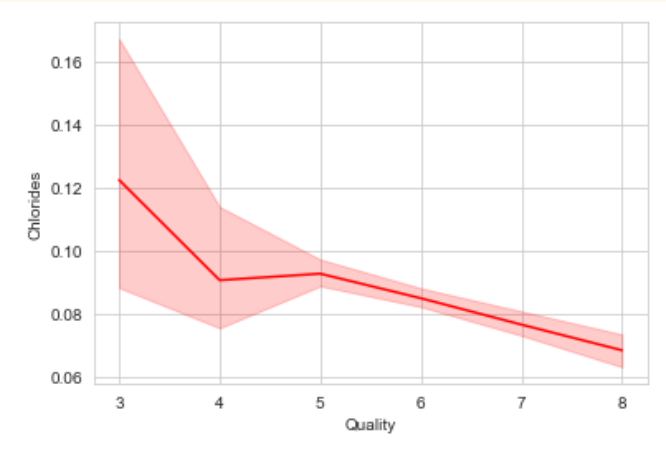

Citric Acid
0.5
0.4
0.3
0.2
0.1
0.0
3
4
5
6
7
8
Quality

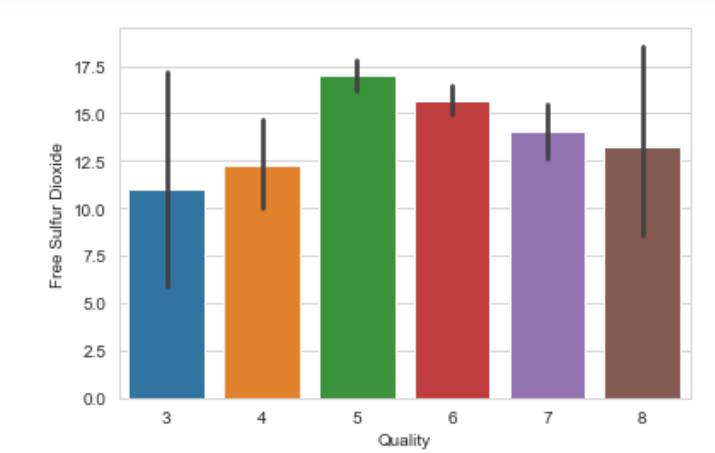

Citric Acid
0.4
0.3
0.2
0.1
3
4
5
6
7
8
Quality

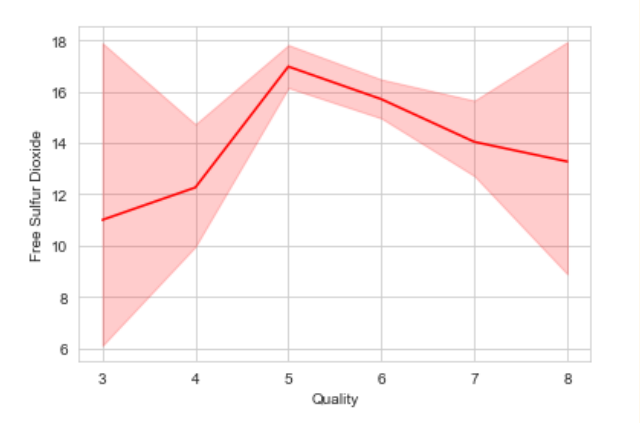

Residual Sugar
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
3
4
5
6
7
8
Quality

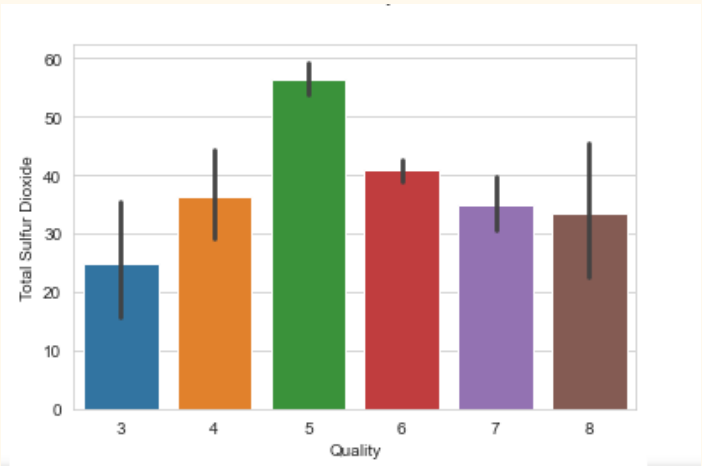

Residual Sugar
3.4
3.2
3.0
2.8
2.6
2.4
2.2
2.0
3
4
5
6
7
8
Quality

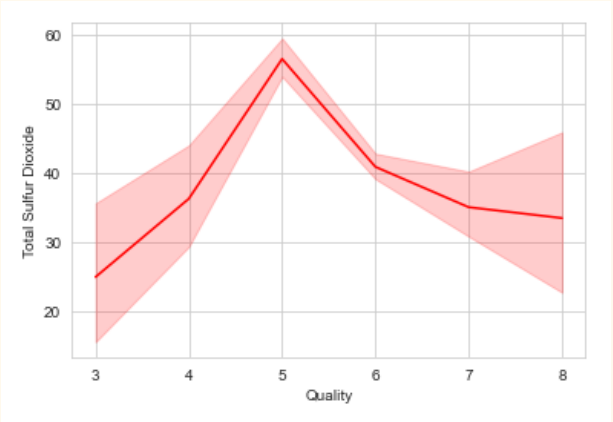

Chlorides
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
3
4
5
6
7
8
Quality

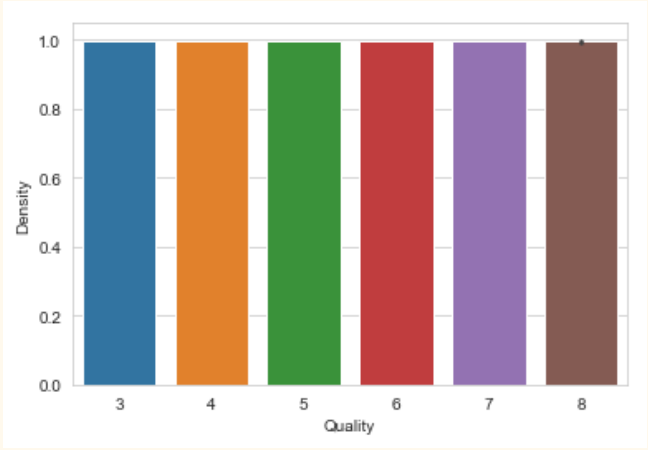

Chlorides
0.16
0.14
0.12
0.10
0.08
0.06
3
4
5
6
7
8
Quality

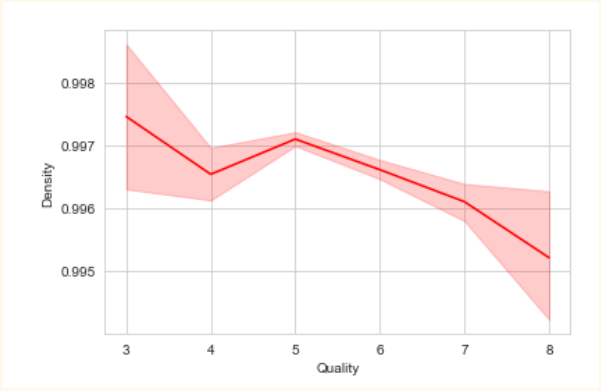

Free Sulfur Dioxide
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
3
4
5
6
7
8
Quality

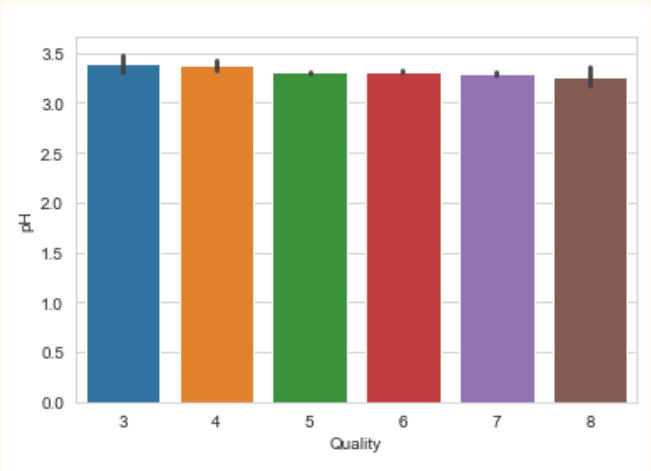

Free Sulfur Dioxide
18
16
14
12
10
8
6
3
4
5
6
7
8
Quality

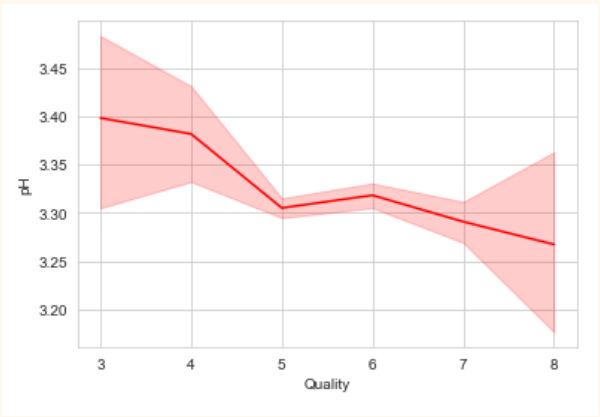

Total Sulfur Dioxide
60
50
40
30
20
10
0
3
4
5
6
7
8
Quality

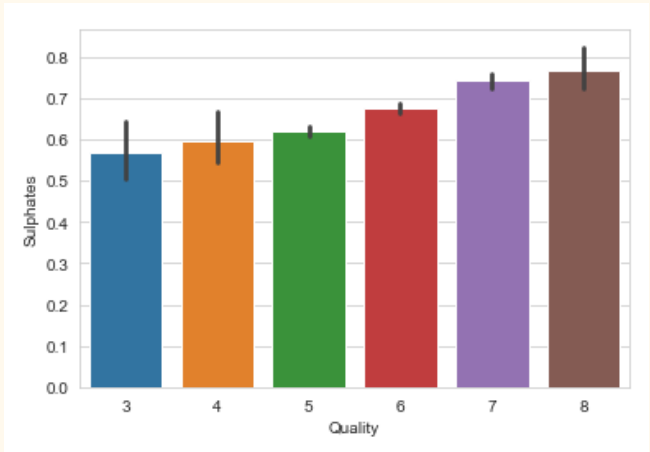

Total Sulfur Dioxide
60
50
40
30
20
3
4
5
6
7
8
Quality

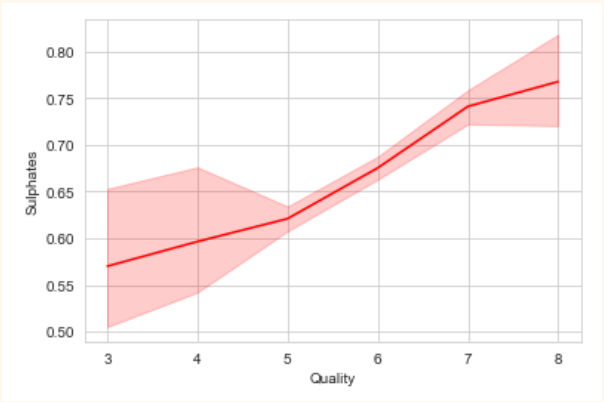

Density
Quality
1.0
0.8
0.6
0.4
0.2
0.0
3
4
5
6
7
8

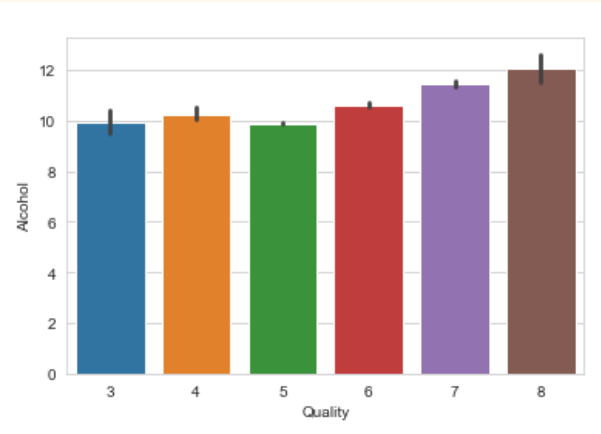

Density
Quality
0.998
0.997
0.996
0.995
3
4
5
6
7
8

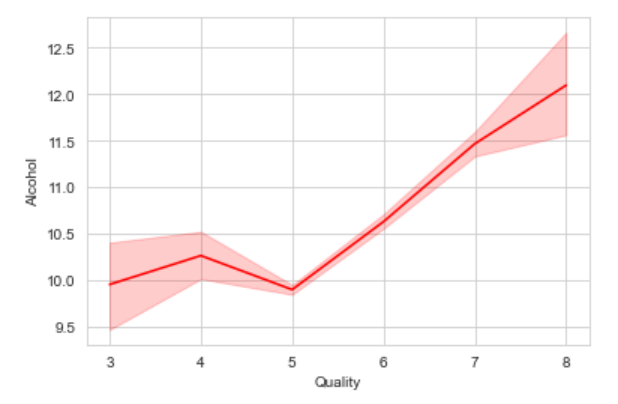

pH
Quality
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
3
4
5
6
7
8

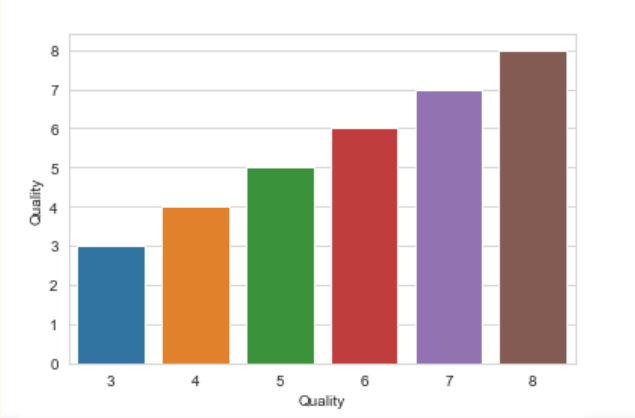

pH
Quality
3.45
3.40
3.35
3.30
3.25
3.20
3
4
5
6
7
8

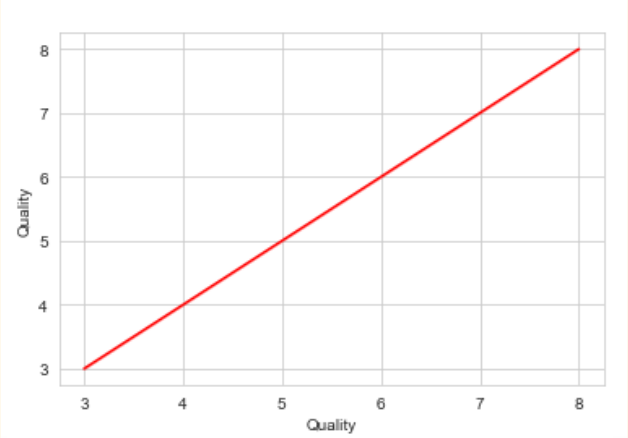

Sulphates
Quality
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
3
4
5
6
7
8


Sulphates
Quality
0.80
0.75
0.70
0.65
0.60
0.55
0.50
3
4
5
6
7
8

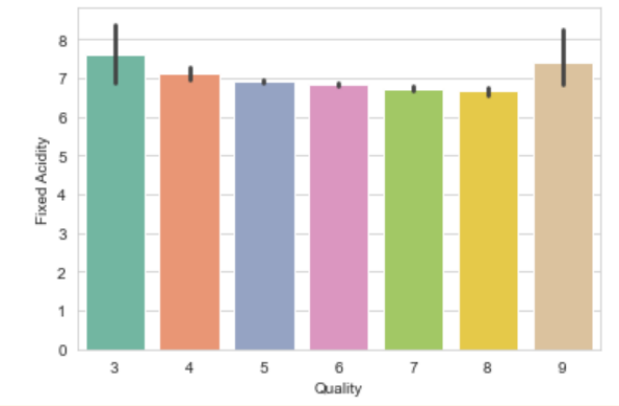

Alcohol
Quality
0
2
4
6
8
10
12
3
4
5
6
7
8

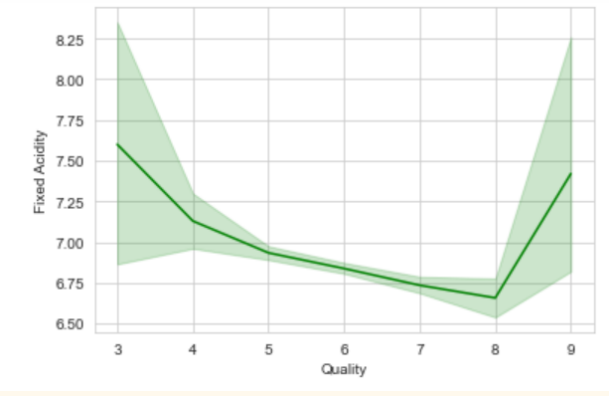

Alcohol
Quality
9.5
10.0
10.5
11.0
11.5
12.0
12.5
3
4
5
6
7
8

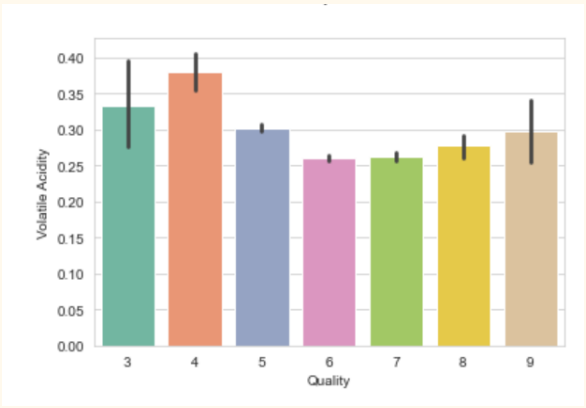

Quality
Quality
0
1
2
3
4
5
6
7
8
3
4
5
6
7
8

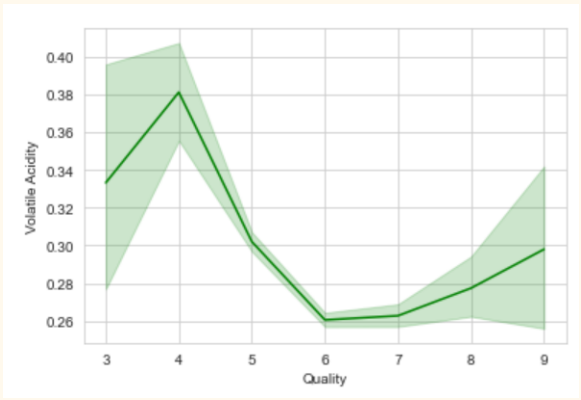

Quality
Quality
3
4
5
6
7
8
3
4
5
6
7
8

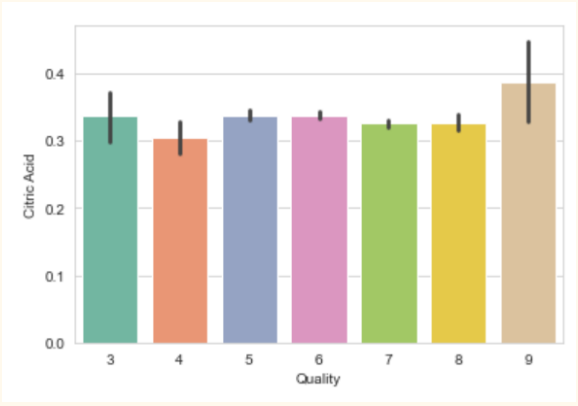
White Wine

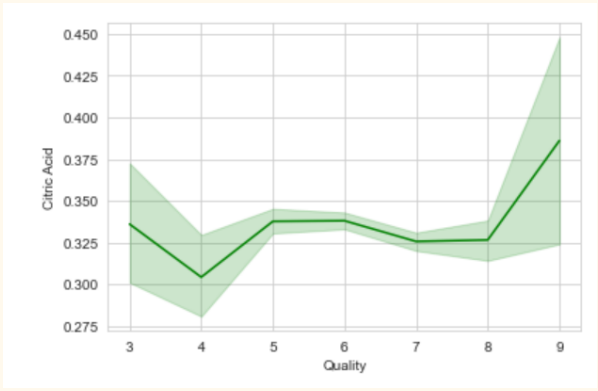

Fixed Acidity
Quality
0
1
2
3
4
5
6
7
8
3
4
5
6
7
8
9

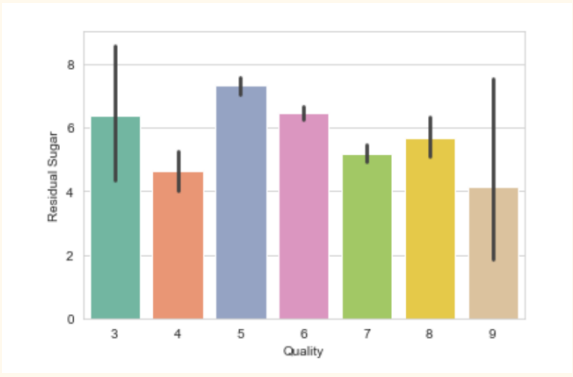

Fixed Acidity
Quality
6.50
6.75
7.00
7.25
7.50
7.75
8.00
8.25
3
4
5
6
7
8
9

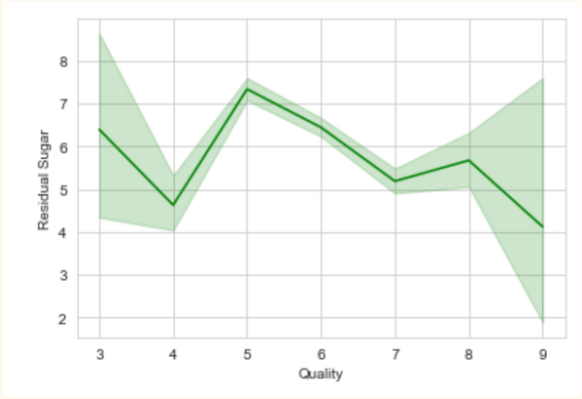

Volatile Acidity
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
3 4 5 6 7 8 9
Quality

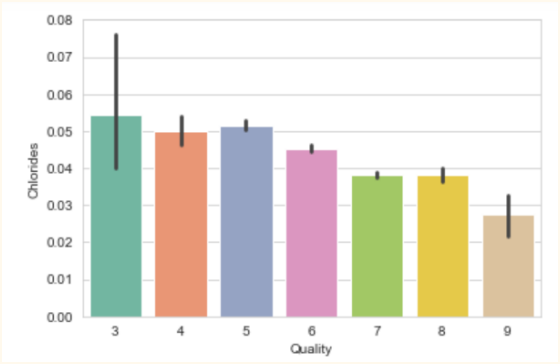

Volatile Acidity
0.40
0.38
0.36
0.34
0.32
0.30
0.28
0.26
3 4 5 6 7 8 9
Quality

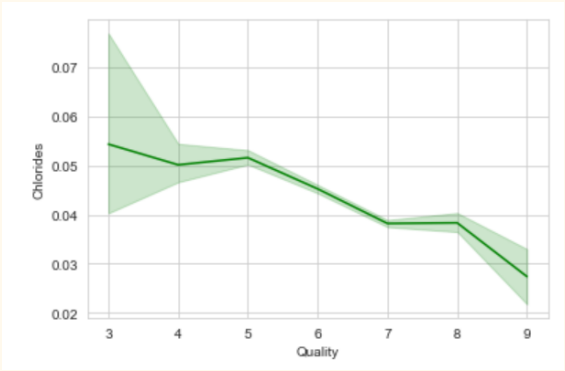

Citric Acid
0.4
0.3
0.2
0.1
0.0
3 4 5 6 7 8 9
Quality

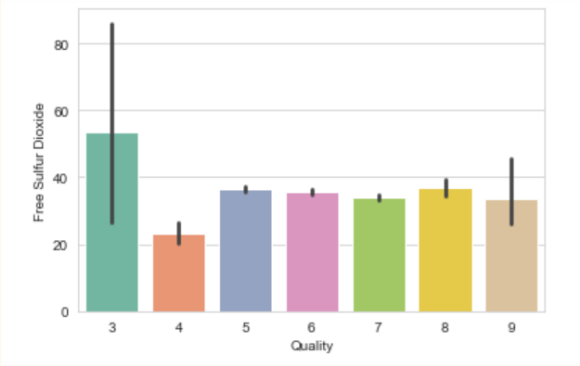

Citric Acid
0.450
0.425
0.400
0.375
0.350
0.325
0.300
0.275
3 4 5 6 7 8 9
Quality

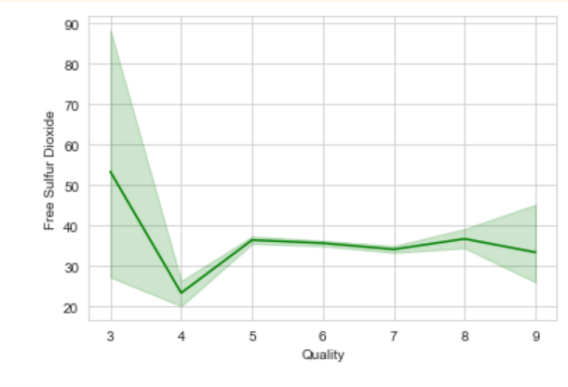

Residual Sugar
8
6
4
2
0
3 4 5 6 7 8 9
Quality

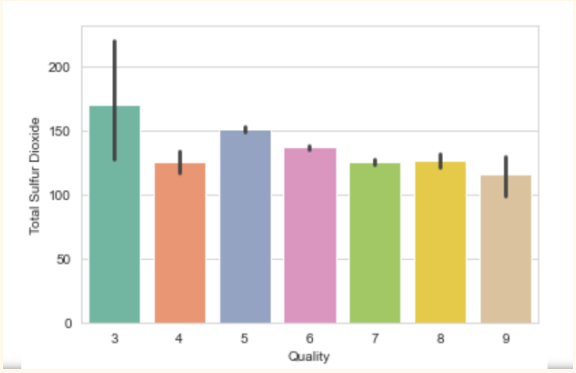

Residual Sugar
8
7
6
5
4
3
2
3 4 5 6 7 8 9
Quality

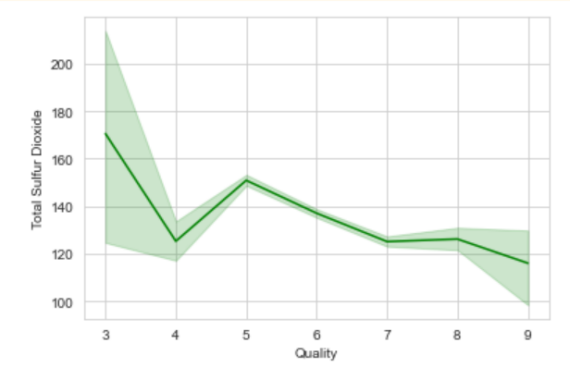

Chlorides
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
3 4 5 6 7 8 9
Quality

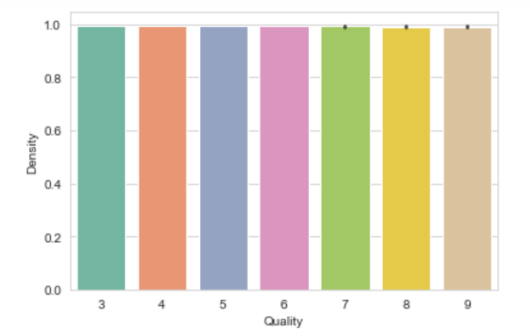

Chlorides
0.07
0.06
0.05
0.04
0.03
0.02
3 4 5 6 7 8 9
Quality

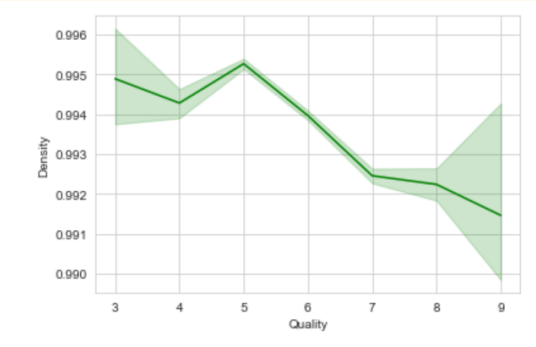

Free Sulfur Dioxide
Quality
0
20
40
60
80
3
4
5
6
7
8
9

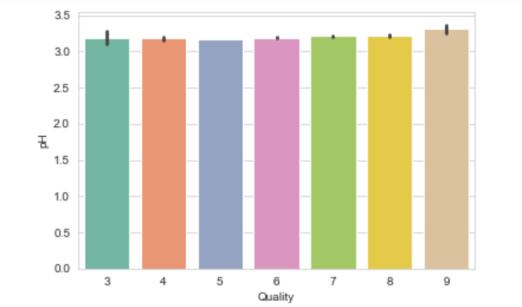

Free Sulfur Dioxide
Quality
20
30
40
50
60
70
80
90
3
4
5
6
7
8
9

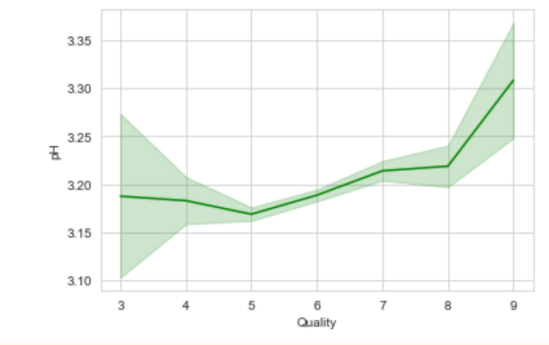

Total Sulfur Dioxide
Quality
0
50
100
150
200
3
4
5
6
7
8
9

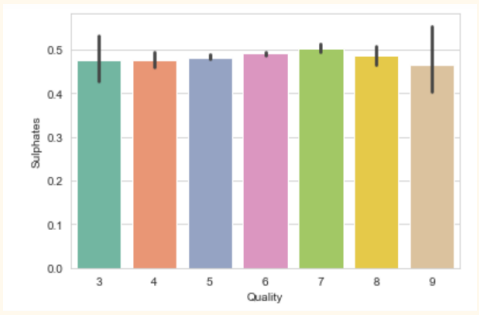

Total Sulfur Dioxide
Quality
100
120
140
160
180
200
3
4
5
6
7
8
9

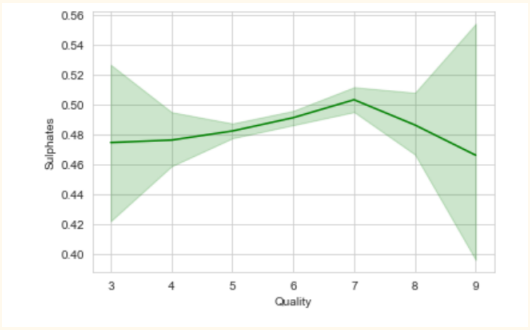

Density
Quality
0.0
0.2
0.4
0.6
0.8
1.0
3
4
5
6
7
8
9

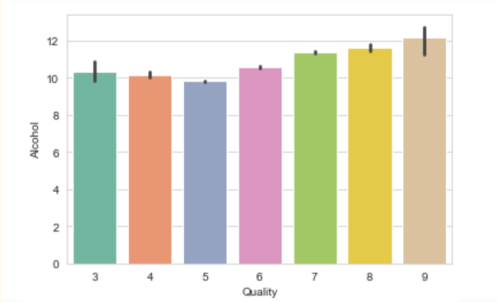

Density
Quality
0.990
0.991
0.992
0.993
0.994
0.995
0.996
3
4
5
6
7
8
9

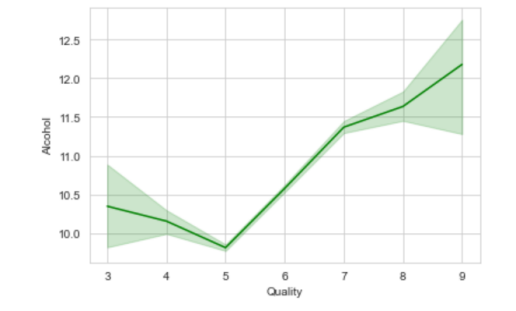

pH
Quality
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
3
4
5
6
7
8
9

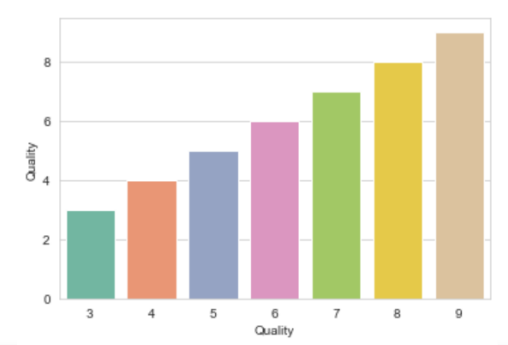

pH
Quality
3.10
3.15
3.20
3.25
3.30
3.35
3
4
5
6
7
8
9

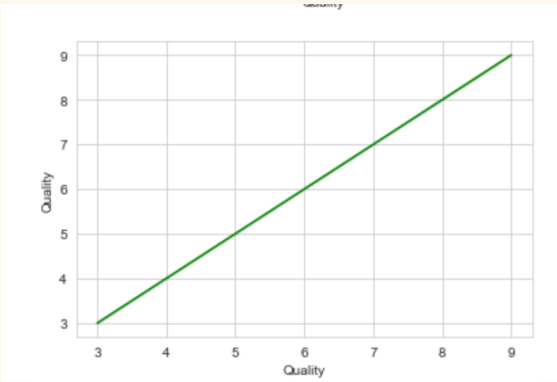

Sulphates
Quality
0.0
0.1
0.2
0.3
0.4
0.5
3
4
5
6
7
8
9


Sulphates
Quality
0.40
0.42
0.44
0.46
0.48
0.50
0.52
0.54
0.56
3
4
5
6
7
8
9


Alcohol
Quality
0
2
4
6
8
10
12
3
4
5
6
7
8
9


Alcohol
Quality
10.0
10.5
11.0
11.5
12.0
12.5
3
4
5
6
7
8
9


Quality
Quality
0
2
4
6
8
3
4
5
6
7
8
9


Quality
Quality
3
4
5
6
7
8
9
3
4
5
6
7
8
9

Appendix F.

Decision Tree

```
#For Red Wine select the best models, based on the model names in an array
models=[LogisticRegression(),
        KNeighborsClassifier(),
        RandomForestClassifier(),
        DecisionTreeClassifier()]
model_names=['Logistic Regression',
             'K-Nearest Neighbors',
             'Random Forest Classifier',
             'Decision Tree']


# creating an accuracy array and a matrix to join the accuracy of the models
acc=[]
m={}

# get the accuracy for each
for model in range(len(models)):
     clf=models[model]
     clf.fit(X_train,y_train)
     pred=clf.predict(X_test)
     acc.append(accuracy_score(pred,y_test))

m={'Algorithm':model_names,'Accuracy':acc}

print("Red Wine models selection based on the model names in an array")
# putt into a data frame and list out the results
acc_frame=pd.DataFrame(m)
acc_frame
```

```
Red Wine models selection based on the model names in an array
```

	Algorithm	Accuracy
0	Logistic Regression	0.877083
1	K-Nearest Neighbors	0.875000
2	Random Forest Classifier	0.910417
3	Decision Tree	0.887500

```
#For White Wine select the best models, based on the model names in an array
models1=[LogisticRegression(),
        KNeighborsClassifier(),
        RandomForestClassifier(),
        DecisionTreeClassifier()]
model_names=['Logistic Regression',
             'K-Nearest Neighbors',
             'Random Forest Classifier',
             'Decision Tree']


# creating an accuracy array and a matrix to join the accuracy of the models
acc=[]
m={}

# get the accuracy for each
for model in range(len(models1)):
     clf=models1[model]
     clf.fit(X_train1,y_train1)
     pred1=clf.predict(X_test1)
     acc.append(accuracy_score(pred1,y_test1))

m={'Algorithm':model_names,'Accuracy':acc}

print("White Wine models selection based on the model names in an array")
# putt into a data frame and list out the results
acc_frame=pd.DataFrame(m)
acc_frame
```

```
White Wine models selection based on the model names in an array
```

	Algorithm	Accuracy
0	Logistic Regression	0.721088
1	K-Nearest Neighbors	0.681633
2	Random Forest Classifier	0.666667
3	Decision Tree	0.656463

Thank you for your time!

Best regards,

Gohar Artenyan

Lusine Artenyan