Quality of Wines Exported from Portugal

CSIT456_01

By: William Bautista and Teresa Luz Miller

Introduction

Can we determine the quality of wine based on a certain chemical makeup?

Let’s see if it is possible to do so through machine learning

Description of the data set

Through the UCI ML Repository,* we selected a dataset detailing red vinho (wine) samples from the north of Portugal.

The datasets contained 12 attributes based on physicochemical tests ranging from:

- fixed acidity
- volatile acidity
- citric acid
- residual sugar
- chlorides
- free sulfur dioxide
- total sulfur dioxide
- density
- pH
- Sulphates
- alcohol level
- quality

https://archive.ics.uci.edu/ml/datasets/wine+quality

Data Preparation

In [30]:
```
red = pd.read_csv('winequality-red.csv',sep=";")
red
```

Out[30]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows × 12 columns

Data Normalization

Although our dataset was cleaned up prior to loading it, we realized that the data contained numerous outliers.

We used sklearn preprocessing module to normalize our data in order to have a more defined accuracy in our predictions.

In [67]:
```
from sklearn import preprocessing
X = preprocessing.StandardScaler().fit(X).transform(X)
```

Data Processing

We began processing our data by setting our target variable , Y, to “quality” and then split the dataset into training and test sets (80% and 20% respectively)

```
In [69]: X = np.asarray(red.iloc[:,:-1])
         y = np.asarray(red["quality"])
```

Train and test set

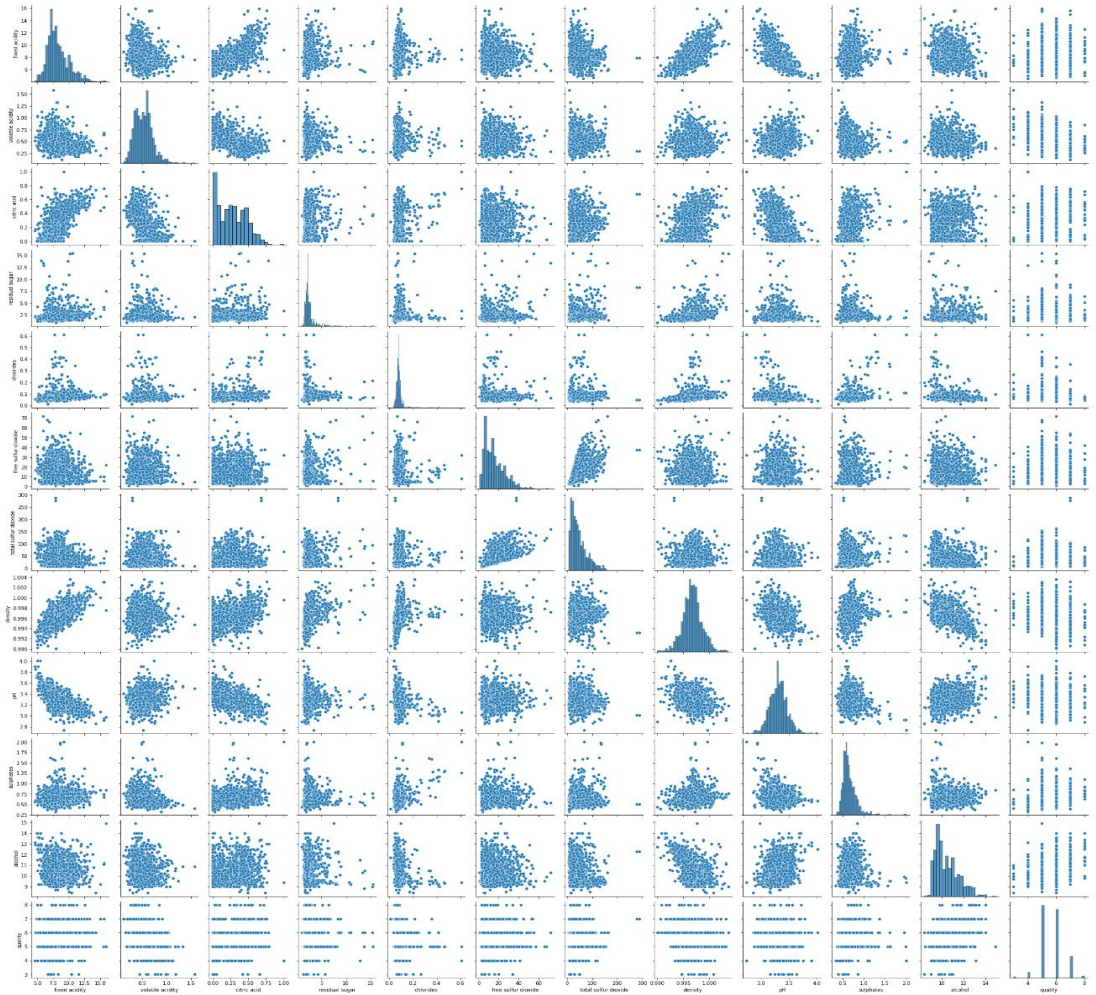
```
In [70]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=0)
         print ("Train set:", X_train.shape, y_train.shape)
         print ("Test set:", X_test.shape, y_test.shape)

         Train set: (1279, 11) (1279,)
         Test set: (320, 11) (320,)
```

Data Visualization

```
In [6]: sns.pairplot(red)
```

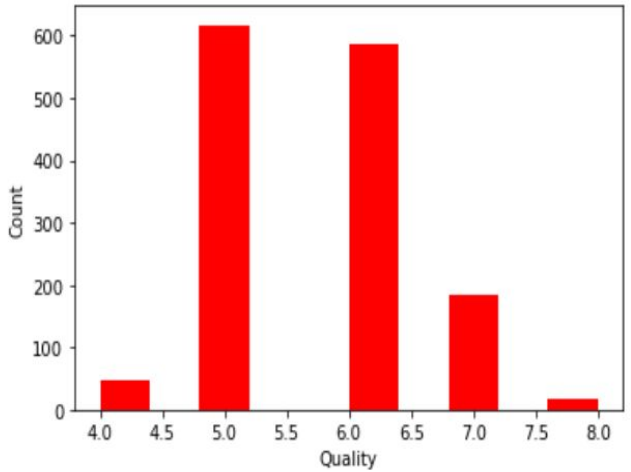
```
Out[6]: <seaborn.axisgrid.PairGrid at 0x7fcbebc66340>
```

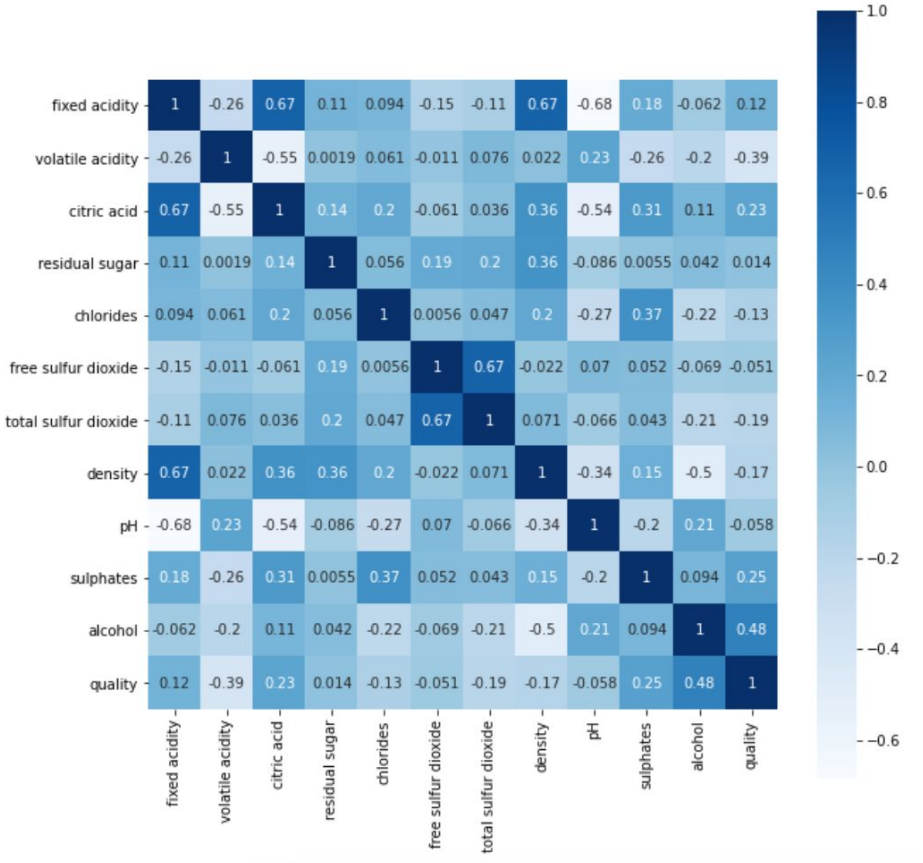


Data Visualization

```
In [7]: correlation = red.corr()
        fig = plt.subplots(figsize=(10,10))
        sns.heatmap(correlation,vmax=1,square=True,annot=True,cmap='Blues')
Out[7]: <AxesSubplot:>
```

```
In [55]: plt.hist(red['quality'],color='red',bins=10)
         plt.xlabel('Quality')
         plt.ylabel('Count')
Out[55]: Text(0, 0.5, 'Count')
```





Model Selection

Logistic Regression

+

K-Nearest Neighbors

Model Implementation for Logistic Regression

In [71]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import roc_auc_score

logreg = LogisticRegression(multi_class="multinomial",solver ="newton-cg")
logreg.fit(X_train, y_train)

y_pred = logreg.predict(X_test)

print(metrics.classification_report(y_test, y_pred, digits=3, zero_division = 1))
print("accuracy", accuracy_score(y_test, y_pred))


accuracy = cross_val_score(logreg, X, y, scoring = "roc_auc_ovr",cv=10)

print("cross validation score with roc_auc", accuracy.mean())
print("roc_auc_score",roc_auc_score(y_test,logreg.predict_proba(X_test), multi_class="ovr"))
```

```
              precision    recall  f1-score   support

           3      1.000     0.000     0.000         2
           4      1.000     0.000     0.000        11
           5      0.658     0.756     0.703       135
           6      0.622     0.627     0.625       142
           7      0.409     0.333     0.367        27
           8      1.000     0.000     0.000         3

    accuracy                          0.625       320
   macro avg      0.782     0.286     0.283       320
weighted avg      0.638     0.625     0.605       320

accuracy 0.625
cross validation score with roc_auc 0.7910407311982046
roc_auc_score 0.7312427759625839
```

Model Implementation for K-Nearest Neighbors

K-Nearest Neighbors

In [17]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train,y_train)
```

Out[17]:
```
KNeighborsClassifier(n_neighbors=3)
```

In [18]:
```
knn.score(X_test,y_test)
```

Out[18]:
```
0.498281786941580756
```

Tuning Hyper Parameter

In [19]:
```
from sklearn.model_selection import StratifiedKFold
skfold = StratifiedKFold(n_splits=5,shuffle=True)
```

In [20]:
```
from sklearn.model_selection import cross_val_score

neighbors = np.arange(1, 16, 2)
cross_val_scores = []

for i in neighbors:
    knn = KNeighborsClassifier(n_neighbors=i)
    scores = cross_val_score (knn, X_train, y_train, cv = skfold)
    cross_val_scores.append(np.mean(scores))

print("Best cross-validation score: {:.3f}".format(np.max(cross_val_scores)))
best = neighbors[np.argmax(cross_val_scores)]
print("Best neighbors: {}".format(best))
```

```
Best cross-validation score: 0.625
Best neighbors: 1
```

In [21]:
```
knn = KNeighborsClassifier(n_neighbors=best)
knn.fit(X_train,y_train)
knn.score(X_test,y_test)
```

Out[21]:
```
0.6116838487972509
```

Conclusion:

We applied classification to answer how several variables are related — specifically how the 11 categorized features of our data set impacts quality of wine.

Comparing the two models, we have found that they are more or less equally accurate in predicting new data although we would consider Logistic Regression as the better model due to the higher AOC score it produces.

Thank you!