PREDICTING WINE QUALITY

Group Members :

Khushi Chavan – 60004190061

Mayav Antani – 60004190066

Problem :

Predicting the wine quality through Logistic Regression and K-Nearest Neighbors

Methodology :

Logistic Regression :

Logistic Regression acts somewhat very similar to linear regression. It also calculates the linear output, followed by a stashing function over the regression output. Sigmoid function is the frequently used logistic function. You can see below clearly, that the z value is same as that of the linear regression output in Eqn(1).

$$z = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots.$$

$$h(\theta) = g(z)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

The h(θ) value here corresponds to P(y=1|x), ie, probability of output to be binary 1, given input x. P(y=0|x) will be equal to 1-h(θ) when value of z is 0, g(z) will be 0.5. Whenever z is positive, h(θ) will be greater than 0.5 and output will be binary 1. Likewise, whenever z is negative, value of y will be 0. As we use a linear equation to find the classifier, the output model also will be a linear one, that means it splits the input dimension into two spaces with all points in one space corresponds to same label.

K-Nearest Neighbors :

K-nearest neighbors is a non-parametric method used for classification and regression. It is one of the most easy ML technique used. It is a lazy learning model, with local approximation.

The basic logic behind KNN is to explore your neighborhood, assume the test datapoint to be similar to them and derive the output. In KNN, we look for k neighbors and come up with the prediction.

In case of KNN classification, a majority voting is applied over the k nearest datapoints whereas, in KNN regression, mean of k nearest datapoints is calculated as the output. As a rule of thumb, we selects odd numbers as k. KNN is a lazy learning model where the computations happens only runtime.

Logistic Regression vs KNN :

1. KNN is a non-parametric model, where LR is a parametric model.
2. KNN is comparatively slower than Logistic Regression.
3. KNN supports non-linear solutions where LR supports only linear solutions.
4. LR can derive confidence level (about its prediction), whereas KNN can only output the labels.

Algorithm : Logistic Regression, K-Nearest Neighbors.

Code & Output :

In [45]:
```
#import statements
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.neighbors import KNeighborsClassifier
```

In [46]:
```
#Reading the data
wine = pd.read_csv("winequality.csv")
```

In [47]:
```
#Understanding the data
wine.head()
```

Out[47]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

In [48]:
```
wine.tail()
```

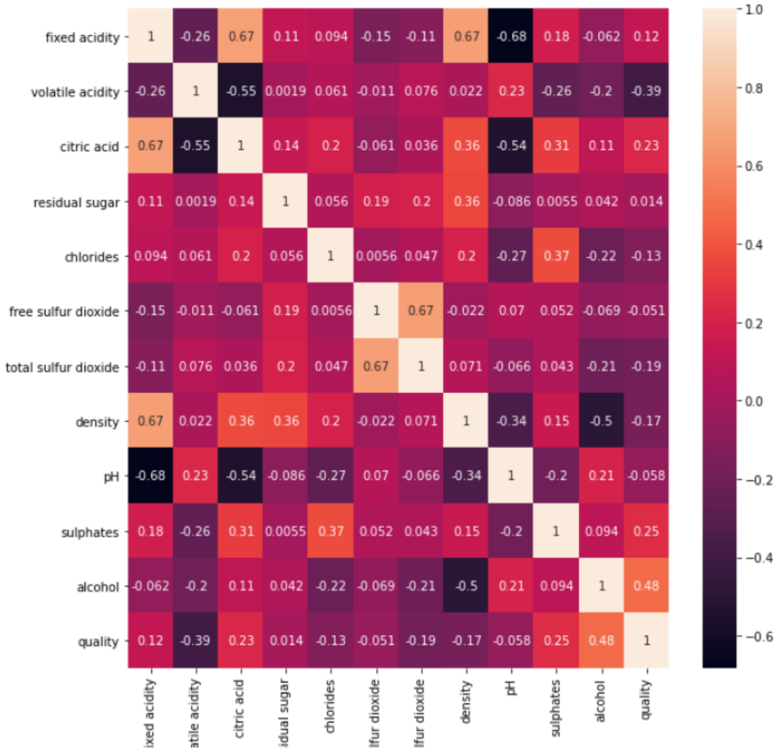
Out[48]:

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

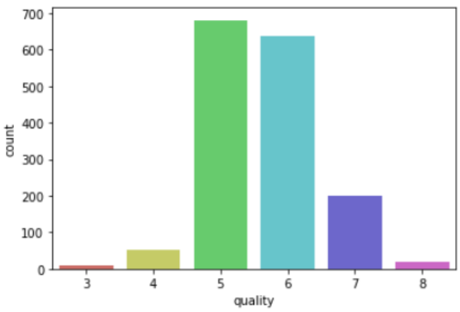
In [49]:
```
#Missing Values
wine.isnull().sum()
```

Out[49]:
```
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 [50]:
```python
#Correlation
plt.figure(figsize=(10,10))
corr=wine.corr()
ax=sns.heatmap(corr, annot=True)
plt.show()
```



In [51]:
```python
#Bar Plot for quality variable
wine.quality.value_counts()
sns.countplot(x="quality",data=wine,palette = 'hls')
plt.show()
```



In [53]:
```python
x=wine[['fixed acidity', 'volatile acidity', 'citric acid',
        'residual sugar', 'chlorides', 'free sulfur dioxide',
        'total sulfur dioxide', 'density', 'pH', 'sulphates', 'alcohol']]
y=wine["quality"]
```

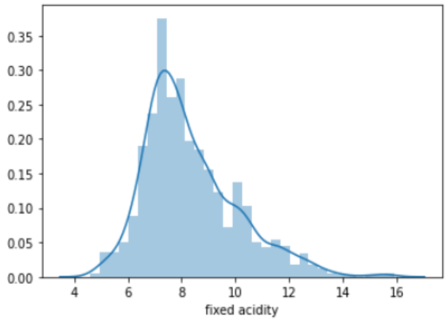
In [54]:
```python
#standardization
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(x)
scaled_x = scaler.transform(x)
```

In [56]:
```python
#spliting the data
x_train, x_test, y_train, y_test = train_test_split(scaled_x, y , test_size = 0.2, random_state  = 365)
```

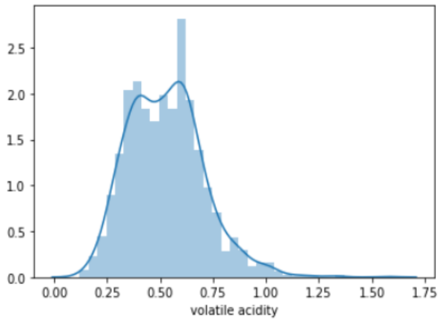
In [58]:
```
#Logistic Regression
logreg = LogisticRegression()
logreg.fit(x_train,y_train)
y_pred = logreg.predict(x_test)
acc = metrics.accuracy_score(y_pred,y_test)
acc
```
Out[58]: 0.621875

In [59]:
```
#KNN
knn = KNeighborsClassifier()
knn.fit(x_train,y_train)
y_predknn = knn.predict(x_test)
acc_knn = metrics.accuracy_score(y_predknn,y_test)
acc_knn
```
Out[59]: 0.590625

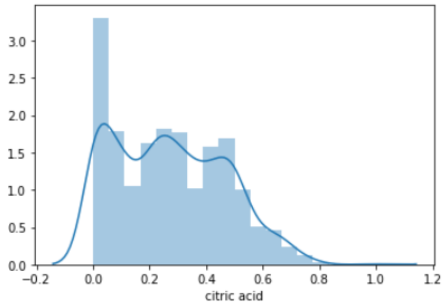
In [61]:
```
sns.distplot(wine['fixed acidity'])
```
Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x19ed97a4310>



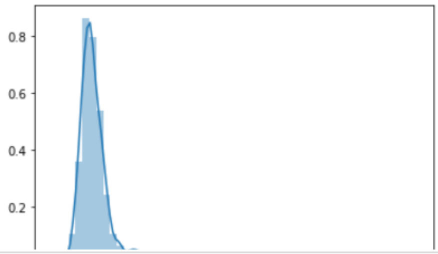
In [62]:
```
sns.distplot(wine['volatile acidity'])
```
Out[62]: <matplotlib.axes._subplots.AxesSubplot at 0x19ed9858070>



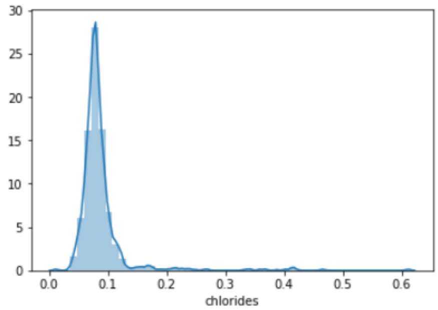
In [63]:
```
sns.distplot(wine['citric acid'])
```
Out[63]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9910be0>
```



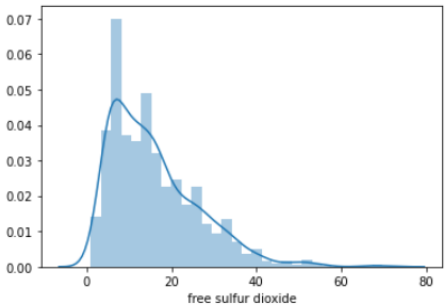
In [64]:
```
sns.distplot(wine['residual sugar'])
```
Out[64]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9983ac0>
```

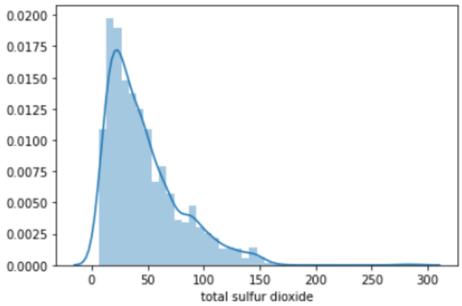


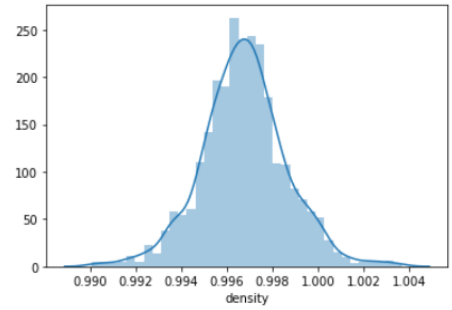
In [65]:
```
sns.distplot(wine['chlorides'])
```
Out[65]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9a6f940>
```

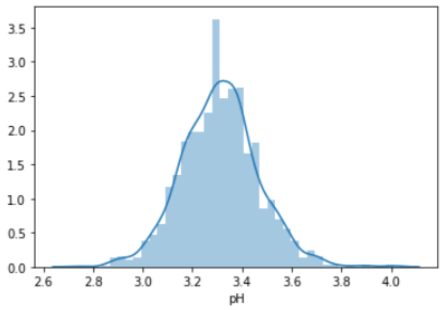


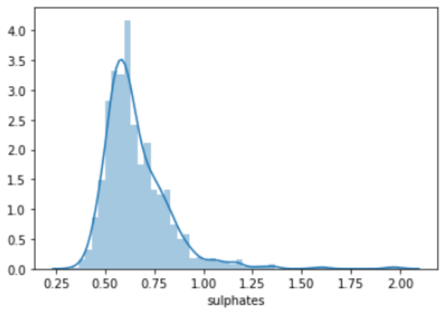
In [66]:
```
sns.distplot(wine['free sulfur dioxide'])
```
Out[66]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9b4e970>
```



In [67]:
```
sns.distplot(wine['total sulfur dioxide'])
```
Out[67]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9b867c0>
```


In [68]:
```
import seaborn as sns
sns.distplot(wine['density'])
```
Out[68]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9cabcd0>
```


In [69]:
```
sns.distplot(wine['pH'])
```
Out[69]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9ca6b20>
```


In [70]:
```
sns.distplot(wine['sulphates'])
```
Out[70]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ed9e190a0>
```


In [71]:
```
sns.distplot(wine['alcohol'])
```

Out[71]: <matplotlib.axes._subplots.AxesSubplot at 0x19ed9dfc820>