In [39]:
```
import pandas as pd
data=pd.read_csv("https://raw.githubusercontent.com/aniruddhachoudhury/Red-Wine-Quality/maste
```

In [40]:
```
data.head()
```

Out[40]:

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

In [41]:
```
data.quality.unique() # there are 6 catergory
```

Out[41]:
```
array([5, 6, 7, 4, 8, 3], dtype=int64)
```

In [42]:
```
data['quality'].value_counts()
```

Out[42]:
```
5    681
6    638
7    199
4     53
8     18
3     10
Name: quality, dtype: int64
```

In [43]:
```
data.columns
```

Out[43]:
```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

In [44]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [ ]:
```
1
```

In [58]:
```
data.describe().T
```

Out[58]:

	count	mean	std	min	25%	50%	75%	max
fixed acidity	1599.0	8.319637	1.741096	4.60000	7.1000	7.90000	9.200000	15.90000
volatile acidity	1599.0	0.527821	0.179060	0.12000	0.3900	0.52000	0.640000	1.58000
citric acid	1599.0	0.270976	0.194801	0.00000	0.0900	0.26000	0.420000	1.00000
residual sugar	1599.0	2.538806	1.409928	0.90000	1.9000	2.20000	2.600000	15.50000
chlorides	1599.0	0.087467	0.047065	0.01200	0.0700	0.07900	0.090000	0.61100
free sulfur dioxide	1599.0	15.874922	10.460157	1.00000	7.0000	14.00000	21.000000	72.00000
total sulfur dioxide	1599.0	46.467792	32.895324	6.00000	22.0000	38.00000	62.000000	289.00000
density	1599.0	0.996747	0.001887	0.99007	0.9956	0.99675	0.997835	1.00369
pH	1599.0	3.311113	0.154386	2.74000	3.2100	3.31000	3.400000	4.01000
sulphates	1599.0	0.658149	0.169507	0.33000	0.5500	0.62000	0.730000	2.00000
alcohol	1599.0	10.422983	1.065668	8.40000	9.5000	10.20000	11.100000	14.90000
quality	1599.0	5.636023	0.807569	3.00000	5.0000	6.00000	6.000000	8.00000

In [59]:
```
# minimum value and max for particular featue is high so scaling is required
```

In [60]:
```
X=data.drop("quality",axis=1)
```

In [61]:
```
X.head()
```

Out[61]:

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

In [62]:
```
y=data["quality"]
```

In [63]:
```
y.head()
```

Out[63]:
```
0    5
1    5
2    5
3    6
4    5
Name: quality, dtype: int64
```

In [64]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

In [65]:
```
X_test.head()
```

Out[65]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcol
803	7.7	0.56	0.08	2.50	0.114	14.0	46.0	0.9971	3.24	0.66	
124	7.8	0.50	0.17	1.60	0.082	21.0	102.0	0.9960	3.39	0.48	
350	10.7	0.67	0.22	2.70	0.107	17.0	34.0	1.0004	3.28	0.98	
682	8.5	0.46	0.31	2.25	0.078	32.0	58.0	0.9980	3.33	0.54	
1326	6.7	0.46	0.24	1.70	0.077	18.0	34.0	0.9948	3.39	0.60	1(

In [66]:
```
from sklearn.preprocessing import StandardScaler
```

In [67]:
```
scaler = StandardScaler() #
```

In [68]:
```
scaler.fit(X_train)##calculate the mean and std dev
```

Out[68]:
```
StandardScaler()
```

In [69]:
```
print(scaler.mean_) # these are the mean for each feature
```

```
[ 8.30345472  0.53246499  0.26933707  2.54691877  0.08772736 15.91223156
 46.76330532  0.99677933  3.31453782  0.65881419 10.41521942]
```

In [70]:
```
X_train_tf=scaler.transform(X_train)
```

In [71]:
```
X_train_tf
```

Out[71]:
```
array([[ 2.40069523, -1.03103722,  1.12742595, ..., -1.26096312,
         0.52726134, -0.01431863],
       [-0.93967131,  1.22920403, -1.32502245, ...,  1.52622836,
        -0.28225704,  2.24363201],
       [-0.99827424,  0.55113165, -1.37611513, ..., -0.74241587,
        -1.20742091, -0.86105011],
       ...,
       [-0.6466567 ,  0.49462562, -1.06955908, ...,  1.26695473,
        -0.68701624, -0.86105011],
       [-0.23643625, -1.87862768,  0.4121285 , ...,  0.03540501,
         0.81637505,  1.39690052],
       [-1.46709761, -1.3700734 , -0.04770558, ...,  0.48913386,
        -0.68701624,  2.90220094]])
```

In [72]:
```
y
```

Out[72]:
```
0       5
1       5
2       5
3       6
4       5
       ..
1594    5
1595    6
1596    6
1597    5
1598    6
Name: quality, Length: 1599, dtype: int64
```

In [73]:
```
from sklearn.svm import SVC
model=SVC()
```

In [74]:
```
model.fit(X_train_tf,y_train)
```

Out[74]:
```
SVC()
```

In [75]:
```
model.score(X_train_tf,y_train)
```

Out[75]:
```
0.6778711484593838
```

In [76]:
```
X_test_tf=scaler.transform(X_test)
```

In [77]:
```
y_predict=model.predict(X_test_tf)
```

In [78]:
```
y_test
```

Out[78]:
```
803    6
124    5
350    6
682    5
1326    6
      ..
813    4
377    7
898    7
126    5
819    5
Name: quality, Length: 528, dtype: int64
```

In [79]:
```
from sklearn.metrics import accuracy_score
```

In [80]:
```
accuracy_score(y_test,y_predict)
```

Out[80]:
```
0.5984848484848485
```

In [81]:
```
from sklearn.linear_model import LogisticRegression
```

In [82]:
```
model2=LogisticRegression()
```

In [83]:
```
model2.fit(X_train_tf,y_train)
```

Out[83]:
```
LogisticRegression()
```

In [84]:
```
y_predict2=model2.predict(X_test_tf)
```

In [85]:
```
accuracy_score(y_test,y_predict2)
```

Out[85]:
```
0.571969696969697
```

In [86]:
```
X_test_tf[0]
```

Out[86]:
```
array([-0.3536421 ,  0.15558944, -0.96737373, -0.03334372,  0.55556956,
       -0.18596079, -0.02314512,  0.1740298 , -0.48314224,  0.00685666,
       -0.76696884])
```

In [93]:
```
model.predict([[-0.3536421 ,  0.15558944, -0.96737373, -0.03334372,  0.55556956,
       -0.18596079, -0.02314512,  0.1740298 , -0.48314224,  0.00685666,
       -0.76696884]])
```

Out[93]:
```
array([5], dtype=int64)
```

In [36]:
```
#gridsearch CV
```

In [37]:
```
pd.read_csv("https://raw.githubusercontent.com/srinivasav22/Graduate-Admission-Prediction/maste
```

Out[37]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65
...	...	...	...	...	...	...	...	...	...
495	496	332	108	5	4.5	4.0	9.02	1	0.87
496	497	337	117	5	5.0	5.0	9.87	1	0.96
497	498	330	120	5	4.5	5.0	9.56	1	0.93
498	499	312	103	4	4.0	5.0	8.43	0	0.73
499	500	327	113	4	4.5	4.5	9.04	0	0.84

500 rows × 9 columns

#TASK

1. YOU HAVE TO INCREASE THE ACCURACY OF THE SVC MODEL(WINEQUALITY DATASET)
2. HYPERPARAMETER TUNING(GRIDSEARCH cv https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html))
3. YOU HVAE TO IMPLEMENT SVR(ADDIMISSOIN_PREDICTION) https://raw.githubusercontent.com/srinivasav22/Graduate-Admission-Prediction/master/Admission_Predict_Ver1.1.csv (https://raw.githubusercontent.com/srinivasav22/Graduate-Admission-Prediction/master/Admission_Predict_Ver1.1.csv)

In [ ]:
```
1
```