In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df = pd.read_csv('Wine.csv')
df.head()
```

Out[2]:

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid
0	14.23	1.71	2.43	15.6	127	2.80	3.06	
1	13.20	1.78	2.14	11.2	100	2.65	2.76	
2	13.16	2.36	2.67	18.6	101	2.80	3.24	
3	14.37	1.95	2.50	16.8	113	3.85	3.49	
4	13.24	2.59	2.87	21.0	118	2.80	2.69	

In [3]:
```
df["Quality"].value_counts()
```

Out[3]:
```
2    71
1    59
3    48
Name: Quality, dtype: int64
```

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Alcohol               178 non-null    float64
 1   Malic_Acid            178 non-null    float64
 2   Ash                   178 non-null    float64
 3   Ash_Alcanity          178 non-null    float64
 4   Magnesium             178 non-null    int64
 5   Total_Phenols         178 non-null    float64
 6   Flavanoids            178 non-null    float64
 7   Nonflavanoid_Phenols  178 non-null    float64
 8   Proanthocyanins       178 non-null    float64
 9   Color_Intensity       178 non-null    float64
 10  Hue                   178 non-null    float64
 11  OD280                 178 non-null    float64
 12  Proline               178 non-null    int64
 13  Quality               178 non-null    int64
dtypes: float64(11), int64(3)
memory usage: 19.6 KB
```

In [5]:
```
X = df.iloc[:, :13]
y = df.iloc[:, 13]
```

In [6]:
```
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, ran
dom_state = 0)
```

Without PCA

In [7]:
```
#Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
base_model = LogisticRegression()
base_model.fit(X_train, y_train)

#Predicting the Test set results
y_pred = base_model.predict(X_test)

#Evaluation
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

```
C:\Users\rrr90\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion
  n_iter_i = _check_optimize_result(
```

Out[7]:
```
0.9166666666666666
```

Applying PCA

In [8]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
```

In [9]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, ran
dom_state = 0)
```

In [10]:
```
from sklearn.decomposition import PCA
pca_model = PCA(n_components=0.95)

X_train_pca = pca_model.fit_transform(X_train)
X_test_pca = pca_model.transform(X_test)
```

In [11]:
```
X_train_pca.shape
```

Out[11]:
```
(142, 10)
```

In [12]:
```
pca_model.explained_variance_ratio_
```

Out[12]:
```
array([0.36722576, 0.19231879, 0.10830194, 0.07414597, 0.06288414,
       0.05059778, 0.0419487 , 0.02518069, 0.02222384, 0.01858596])
```

In [13]:
```
## Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X_train_pca, y_train)

## Predicting the Test set results
y_pred = classifier.predict(X_test_pca)

#Evaluation
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[13]:
```
1.0
```