21MIS1152 Rajeev Sekar

```
# Import necessary libraries
from sklearn import datasets  # to retrieve the iris Dataset
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import seaborn as sns
```

Loading the dataset

```
#Load the Dataset
iris = datasets.load_iris()
#convert the dataset into a pandas data frame
df = pd.DataFrame(iris['data'], columns = iris['feature_names'])
y=iris['target']
df.head()

   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width 
(cm)
0                5.1               3.5                1.4
0.2
1                4.9               3.0                1.4
0.2
2                4.7               3.2                1.3
0.2
3                4.6               3.1                1.5
0.2
4                5.0               3.6                1.4
0.2
```

Scaling the data

```
scalar = StandardScaler()
scaled_data = pd.DataFrame(scalar.fit_transform(df)) #scaling the data
scaled_data

            0         1         2         3
0   -0.900681  1.019004 -1.340227 -1.315444
1   -1.143017 -0.131979 -1.340227 -1.315444
2   -1.385353  0.328414 -1.397064 -1.315444
3   -1.506521  0.098217 -1.283389 -1.315444
4   -1.021849  1.249201 -1.340227 -1.315444
..        ...       ...       ...       ...
145  1.038005 -0.131979  0.819596  1.448832
146  0.553333 -1.282963  0.705921  0.922303
147  0.795669 -0.131979  0.819596  1.053935
148  0.432165  0.788808  0.933271  1.448832
149  0.068662 -0.131979  0.762758  0.790671
```

```
[150 rows x 4 columns]
```

Training model before and after applying PCA to check the performance, KNN algorithm is used for this task

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(scaled_data, y,
test_size=0.3)

clf = KNeighborsClassifier(n_neighbors = 5)
clf.fit(X_train, y_train)

test_score = clf.score(X_test, y_test)

test_score
```

```
0.9333333333333333
```

PCA is applied to reduce the no of dimensions from 4 to 3

```
#Taking no. of Principal Components as 3
pca = PCA(n_components = 3)
pca.fit(scaled_data)
data_pca = pca.transform(scaled_data)
data_pca = pd.DataFrame(data_pca,columns=['PC1','PC2','PC3'])
data_pca.head()
```

```
        PC1       PC2       PC3
0 -2.264703  0.480027 -0.127706
1 -2.080961 -0.674134 -0.234609
2 -2.364229 -0.341908  0.044201
3 -2.299384 -0.597395  0.091290
4 -2.389842  0.646835  0.015738
```

```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(data_pca, y,
test_size=0.2)

clf = KNeighborsClassifier(n_neighbors = 5)
clf.fit(X_train, y_train)

test_score = clf.score(X_test, y_test)

test_score
```

```
0.9666666666666667
```

```
The performance
```