Lab 07
Smruti Dawale
B21BB007

Que 1)
Pre-Processing of data

By preprocessing data , we make it easier to interpret and use .This process eliminates inconsistencies or duplicates in data, which can otherwise negatively affect a model's accuracy.
I have used DropNa to drop null values , and have performed MinMax scaler to scale the data into the range 0 to 1.

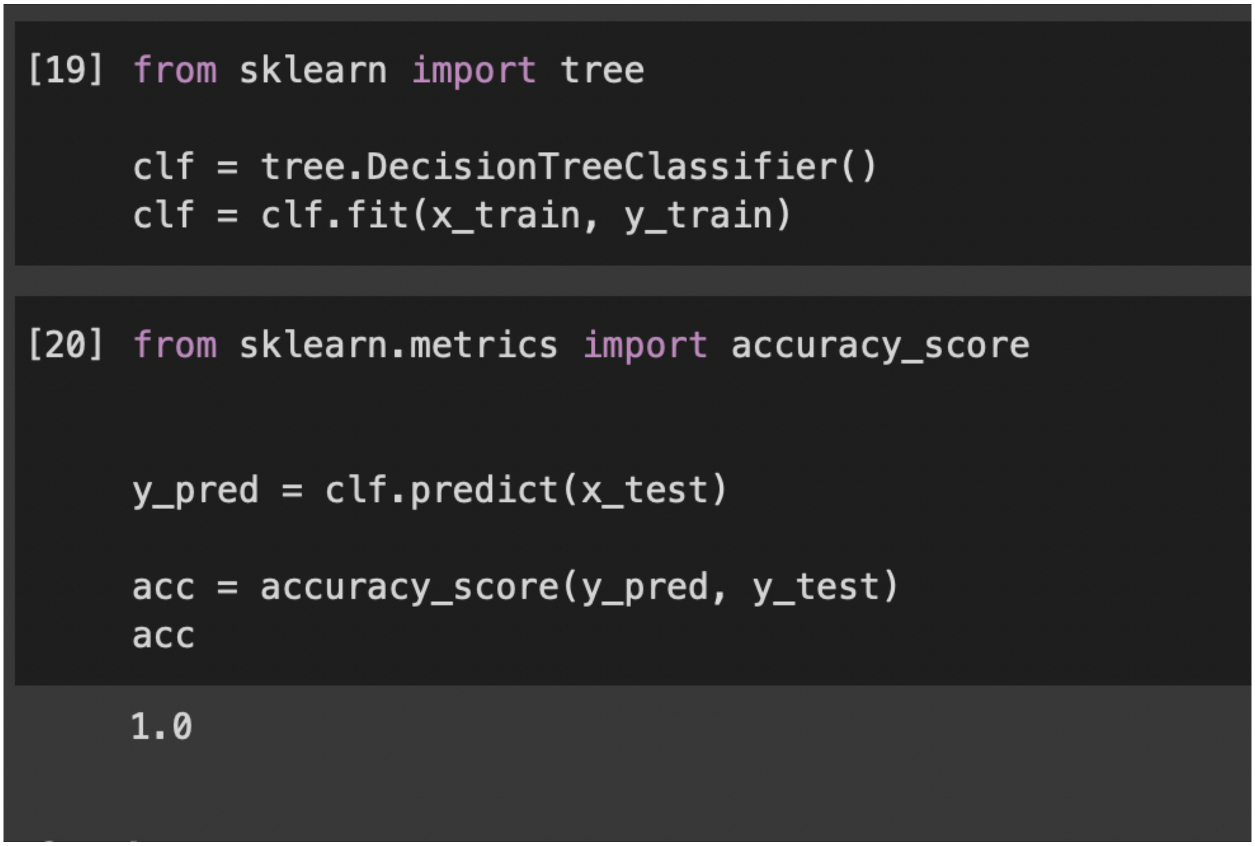
Que 2)
Decision Tree

Given data is a type of classification problem , I have used Decision Tree classifier .
Accuracy - 1.0

Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems.

Advantages of Decision Tree

- It is simple to understand as it follows the same process which a human follows while making any decision in real-life.
- It can be very useful for solving decision-related problems.(Here we have to decide to which class the flower would belong depending upon petal)
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.



```
[19] from sklearn import tree

     clf = tree.DecisionTreeClassifier()
     clf = clf.fit(x_train, y_train)
```

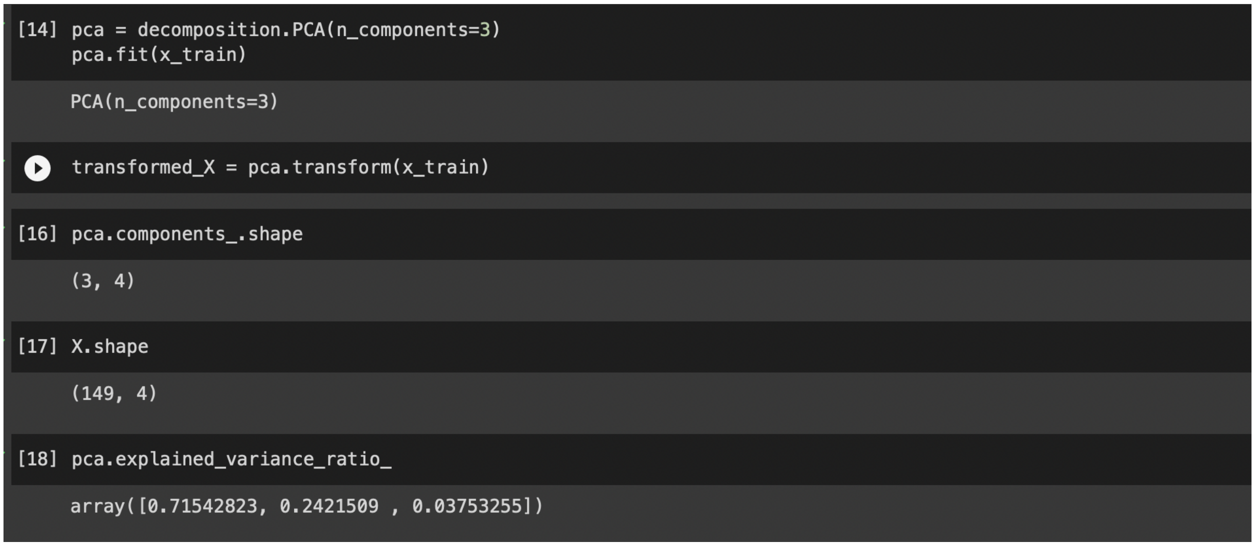
```
[20] from sklearn.metrics import accuracy_score


     y_pred = clf.predict(x_test)

     acc = accuracy_score(y_pred, y_test)
     acc
```

```
     1.0
```

Que 3)

PCA

Principal Component Analysis (PCA) is a statistical technique used for data reduction without losing its properties.It reduces the dimensionality of the data without losing the essence of data , and hence solves the problem of overfitting.



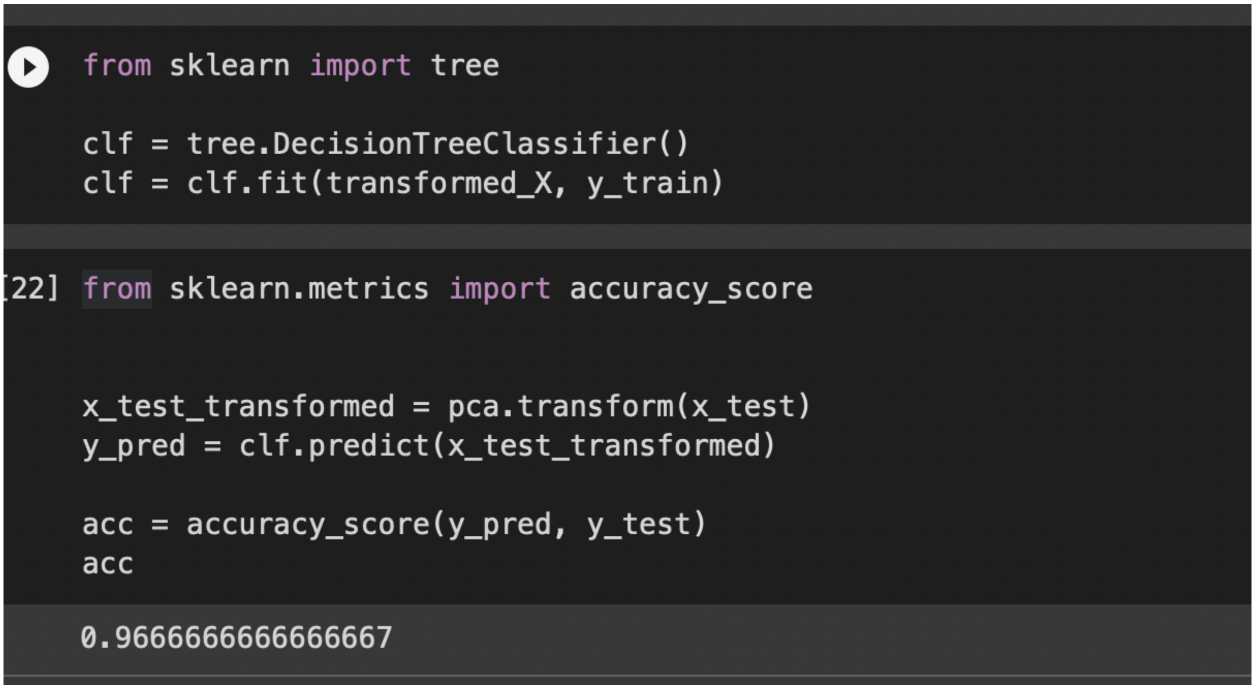
```
[14] pca = decomposition.PCA(n_components=3)
     pca.fit(x_train)

     PCA(n_components=3)

     transformed_X = pca.transform(x_train)

[16] pca.components_.shape

     (3, 4)

[17] X.shape

     (149, 4)

[18] pca.explained_variance_ratio_

     array([0.71542823, 0.2421509 , 0.03753255])
```

Que 4)

Here we have used the data after using pca on it , you can see there is not much difference in the accuracy of data .

Accuracy before - 1.0

Accuracy after - `0.9666666666666667`



```
from sklearn import tree

clf = tree.DecisionTreeClassifier()
clf = clf.fit(transformed_X, y_train)
```

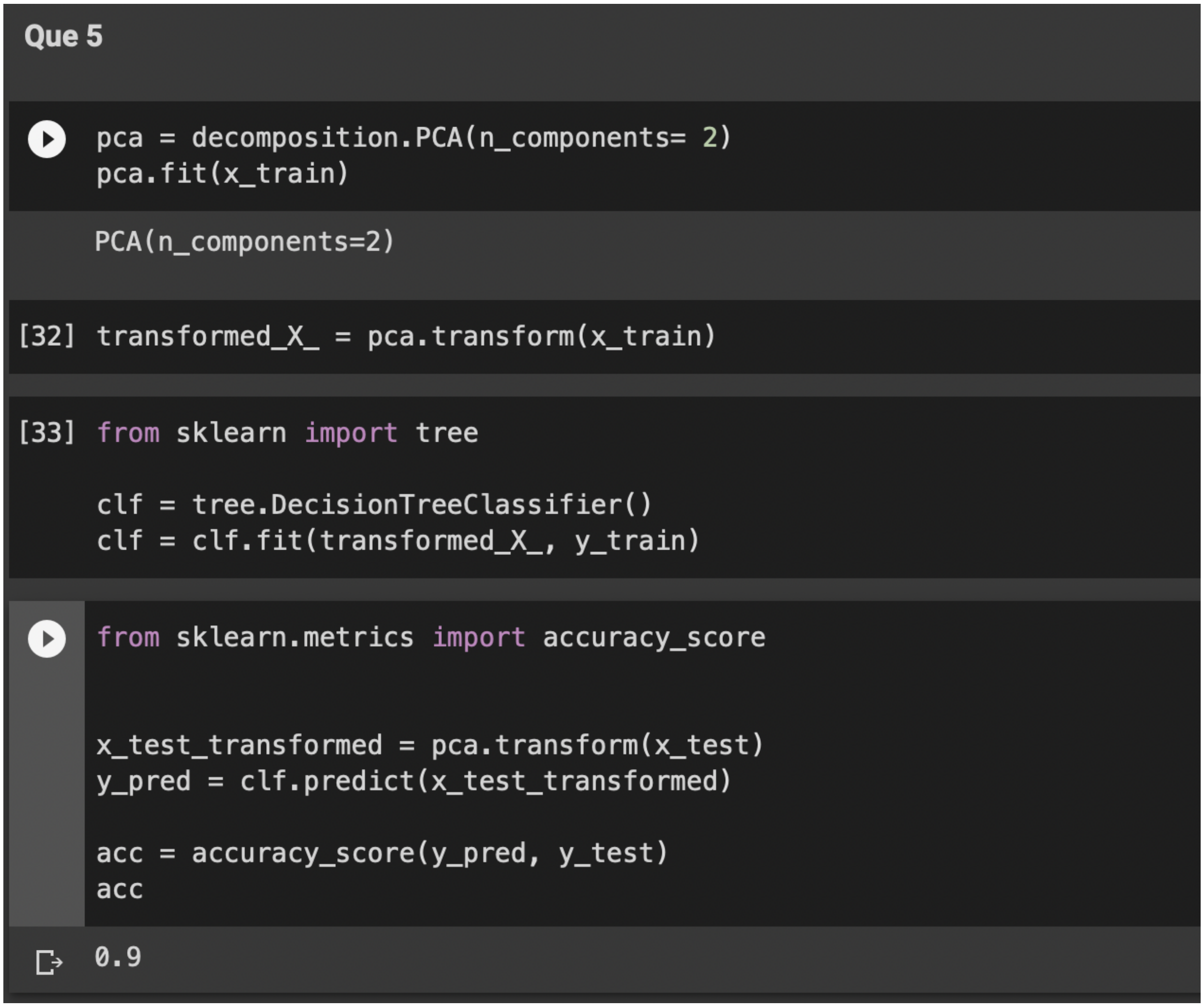
```
from sklearn.metrics import accuracy_score


x_test_transformed = pca.transform(x_test)
y_pred = clf.predict(x_test_transformed)

acc = accuracy_score(y_pred, y_test)
acc
```

```
0.9666666666666667
```

Que 5)

Accuracy when n_component = 3 >>>> 0.9666666666

Accuracy when n_component = 2 >>>> 0.9333



Que 5

```
pca = decomposition.PCA(n_components= 2)
pca.fit(x_train)
```

```
PCA(n_components=2)
```

```
[32] transformed_X_ = pca.transform(x_train)
```

```
[33] from sklearn import tree

     clf = tree.DecisionTreeClassifier()
     clf = clf.fit(transformed_X_, y_train)
```

```
from sklearn.metrics import accuracy_score


x_test_transformed = pca.transform(x_test)
y_pred = clf.predict(x_test_transformed)

acc = accuracy_score(y_pred, y_test)
acc
```

```
0.9
```