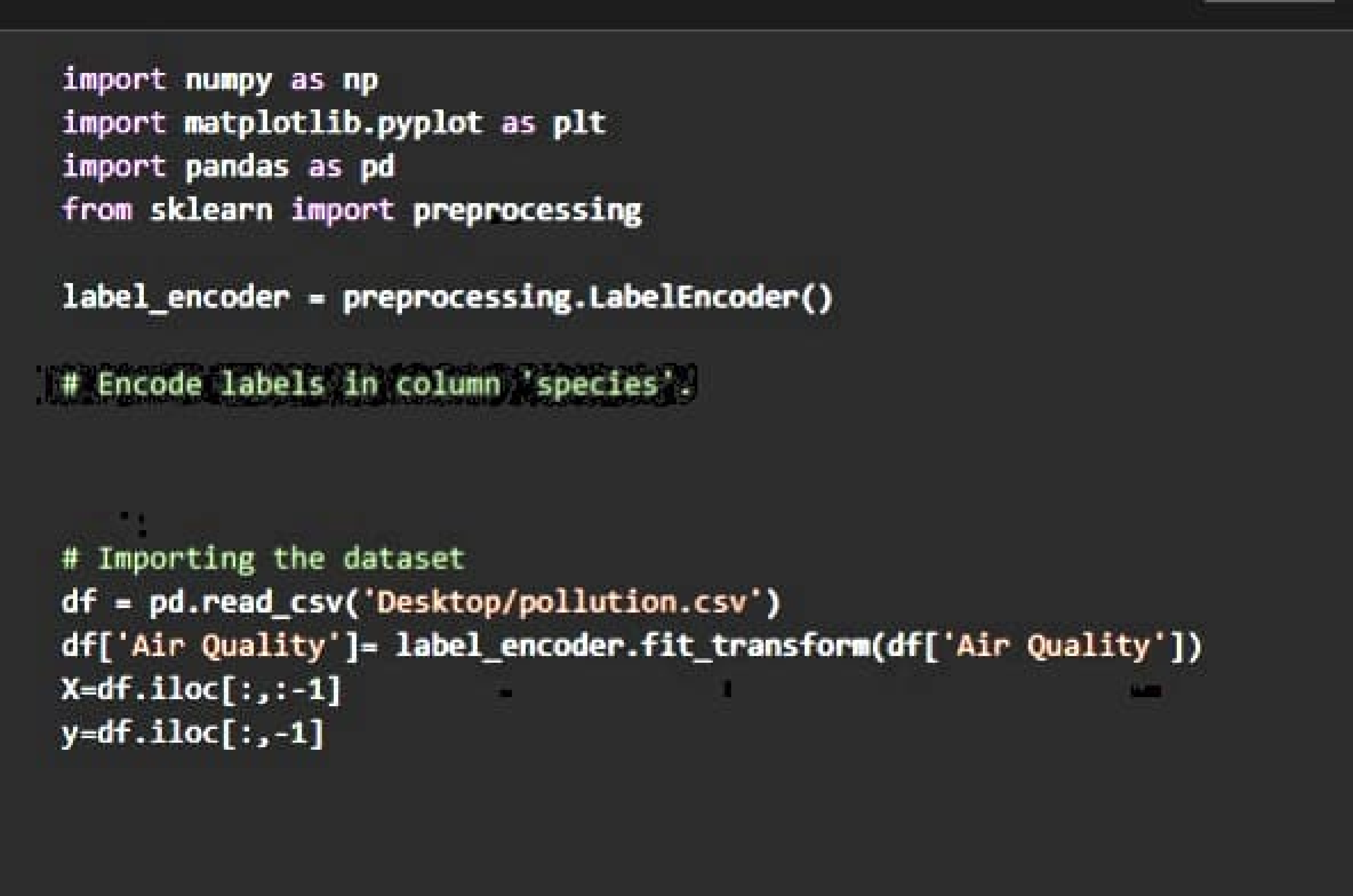


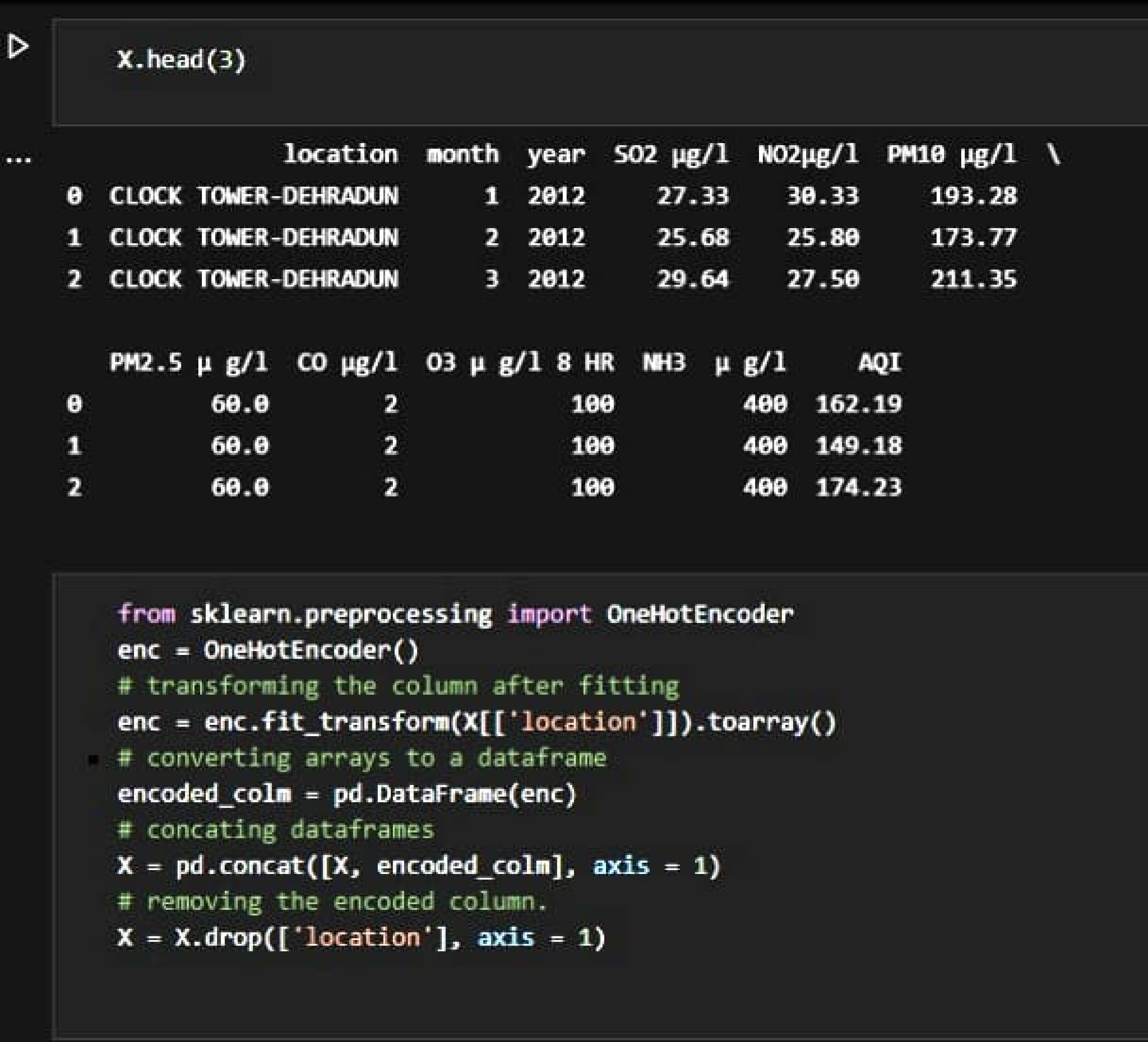
```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import preprocessing

label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.



# Importing the dataset
df = pd.read_csv('Desktop/pollution.csv')
df['Air Quality']= label_encoder.fit_transform(df['Air Quality'])
X=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

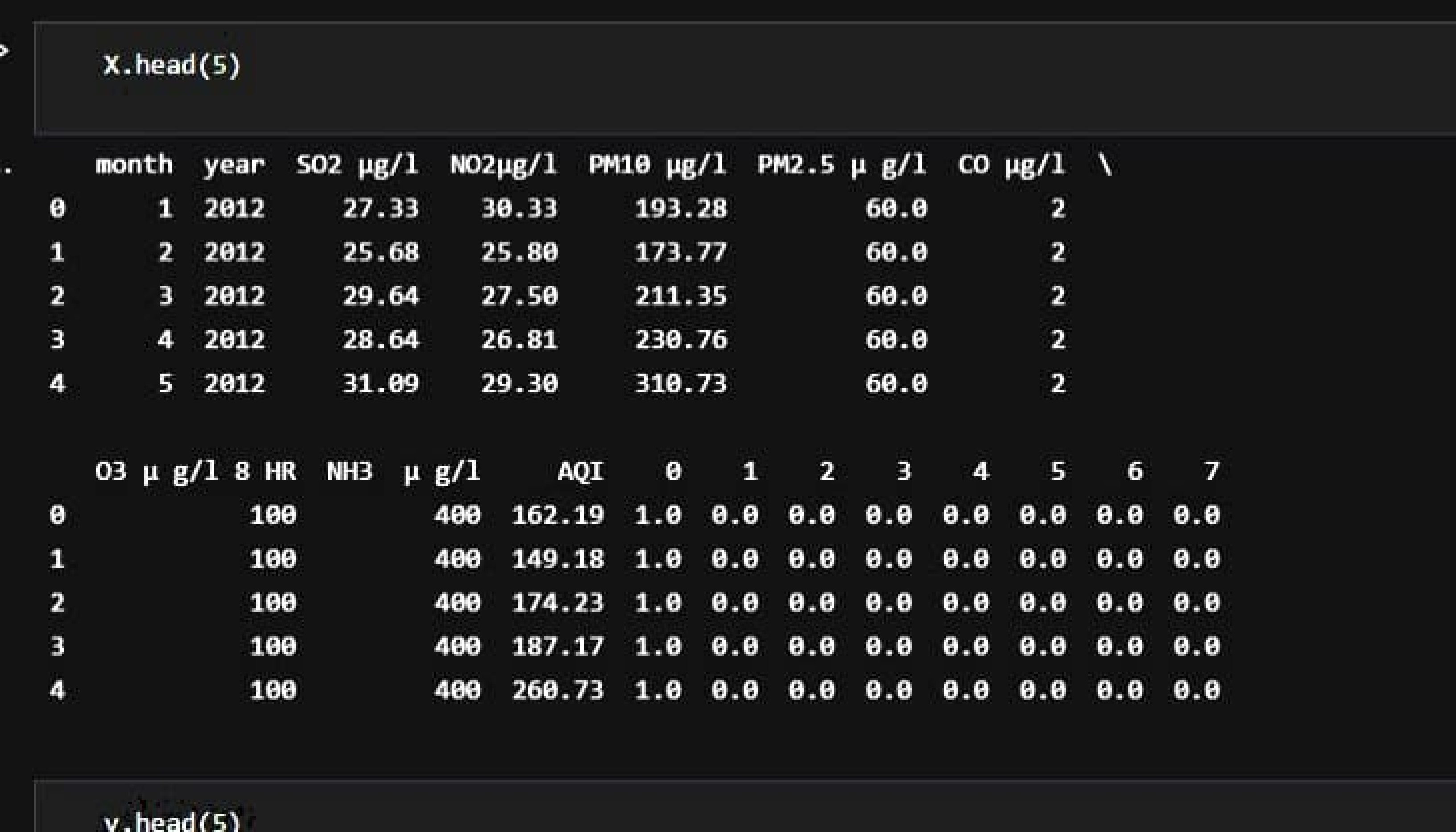


```
X.head(3)
```

```
                 location  month  year  SO2 µg/l  NO2µg/l  PM10 µg/l  \
0  CLOCK TOWER-DEHRADUN      1  2012     27.33    30.33     193.28   
1  CLOCK TOWER-DEHRADUN      2  2012     25.68    25.80     173.77   
2  CLOCK TOWER-DEHRADUN      3  2012     29.64    27.50     211.35   

   PM2.5 µ g/l  CO µg/l  O3 µ g/l 8 HR  NH3  µ g/l     AQI  
0         60.0        2            100          400  162.19  
1         60.0        2            100          400  149.18  
2         60.0        2            100          400  174.23  
```

```
from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder()
# transforming the column after fitting
enc = enc.fit_transform(X[['location']]).toarray()
# converting arrays to a dataframe
encoded_colm = pd.DataFrame(enc)
# concating dataframes
X = pd.concat([X, encoded_colm], axis = 1)
# removing the encoded column.
X = X.drop(['location'], axis = 1)
```

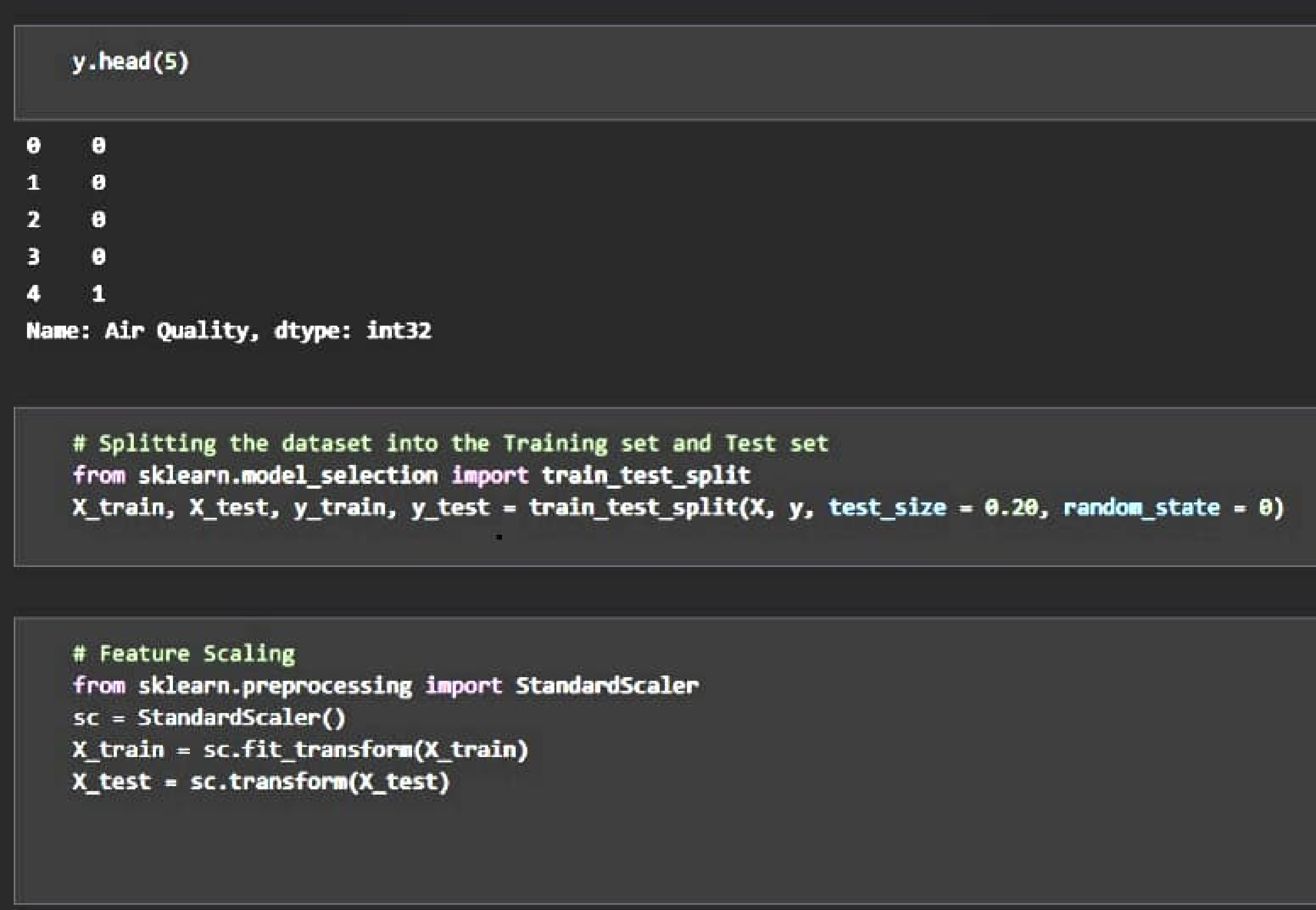


```
X.head(5)
```

```
   month  year  SO2 µg/l  NO2µg/l  PM10 µg/l  PM2.5 µ g/l  CO µg/l  \
0      1  2012     27.33    30.33     193.28         60.0        2   
1      2  2012     25.68    25.80     173.77         60.0        2   
2      3  2012     29.64    27.50     211.35         60.0        2   
3      4  2012     28.64    26.81     230.76         60.0        2   
4      5  2012     31.09    29.30     310.73         60.0        2   

   O3 µ g/l 8 HR  NH3  µ g/l     AQI    0    1    2    3    4    5    6    7  
0            100         400  162.19  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  
1            100         400  149.18  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  
2            100         400  174.23  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  
3            100         400  187.17  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  
4            100         400  260.73  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  
```

```
y.head(5)
```

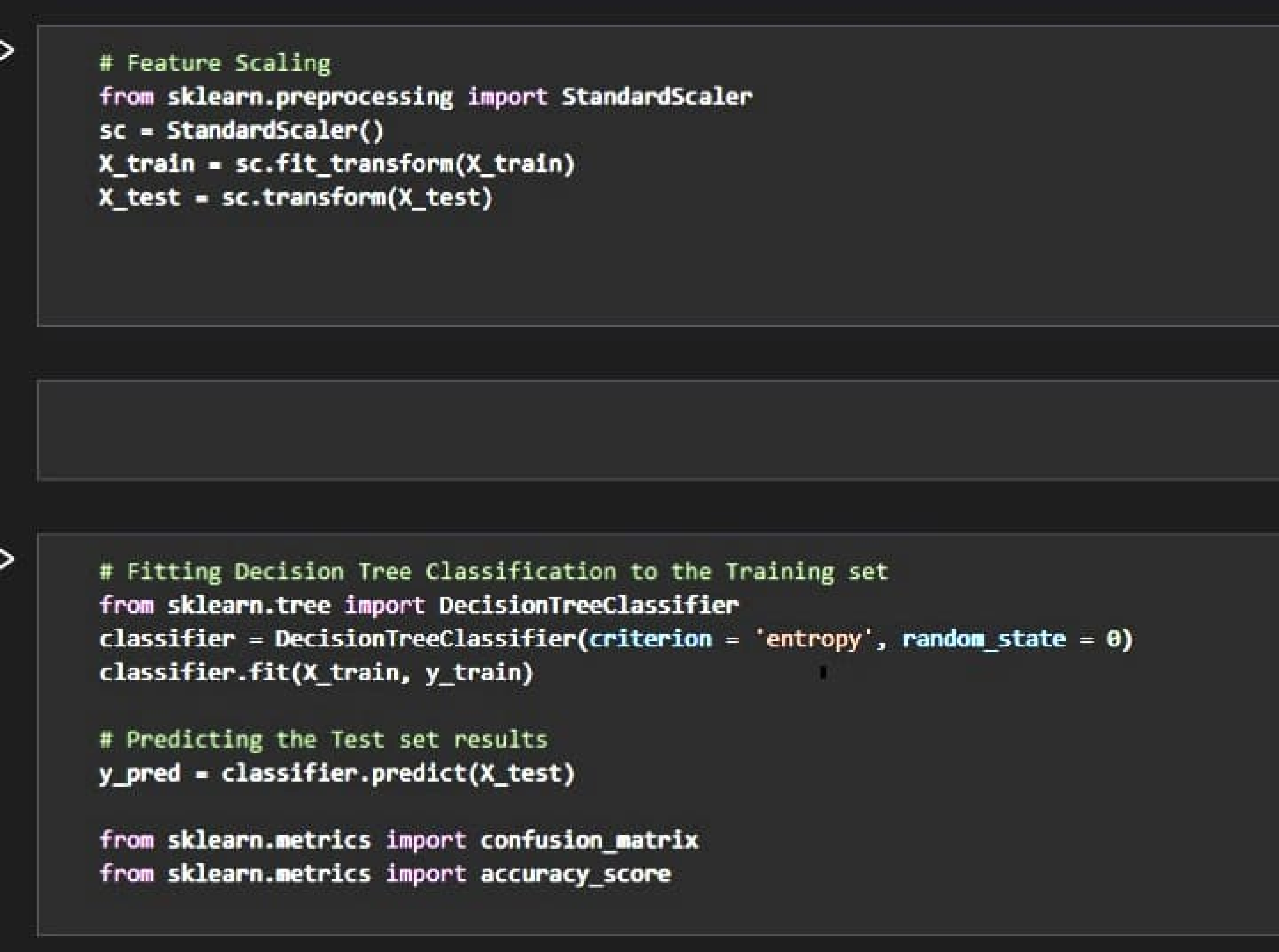


```
y.head(5)
```

```
0    0
1    0
2    0
3    0
4    1
Name: Air Quality, dtype: int32
```

```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
```

```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

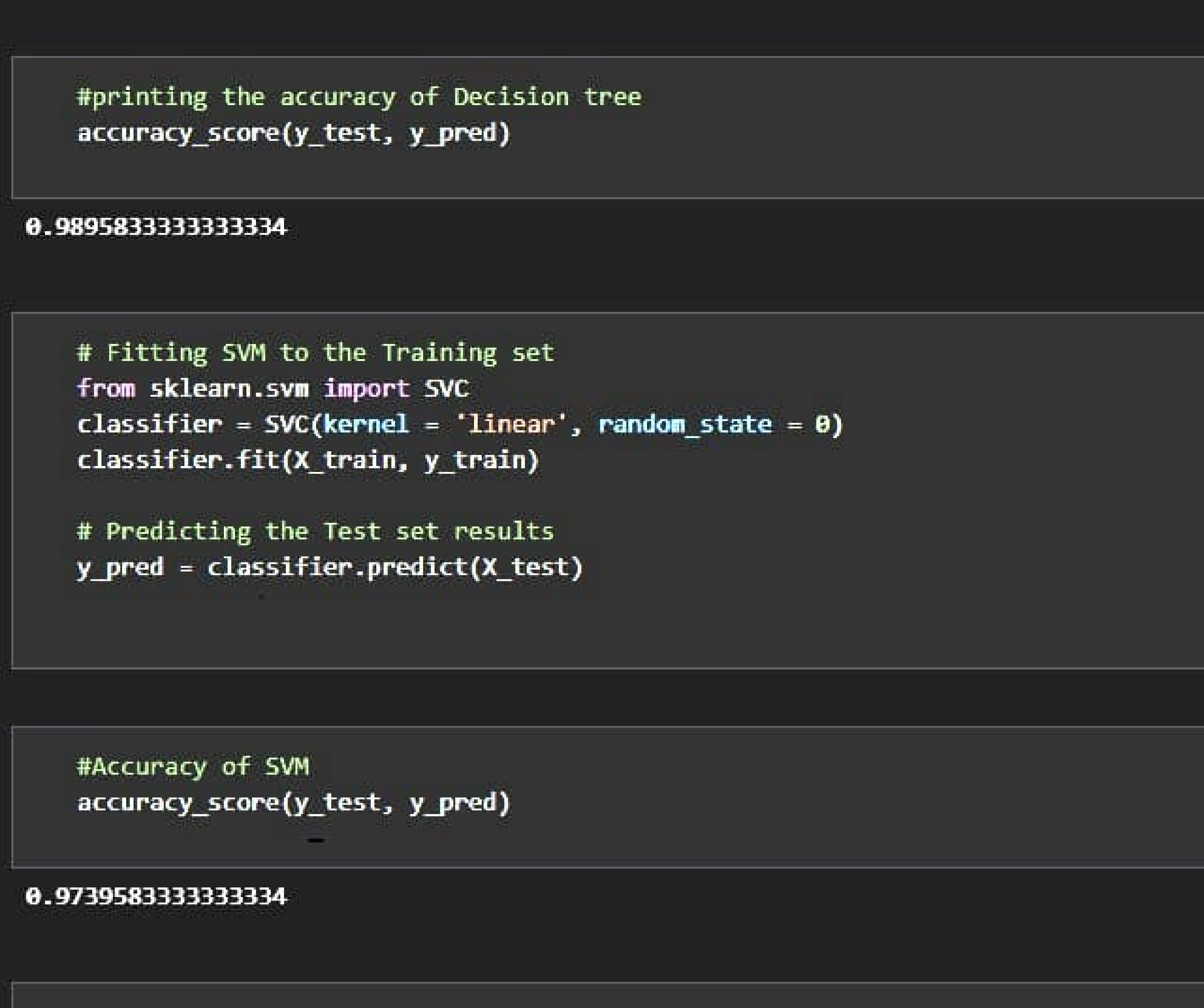


```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
# Fitting Decision Tree Classification to the Training set
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)

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

from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```



```python
#printing the accuracy of Decision tree
accuracy_score(y_test, y_pred)
```

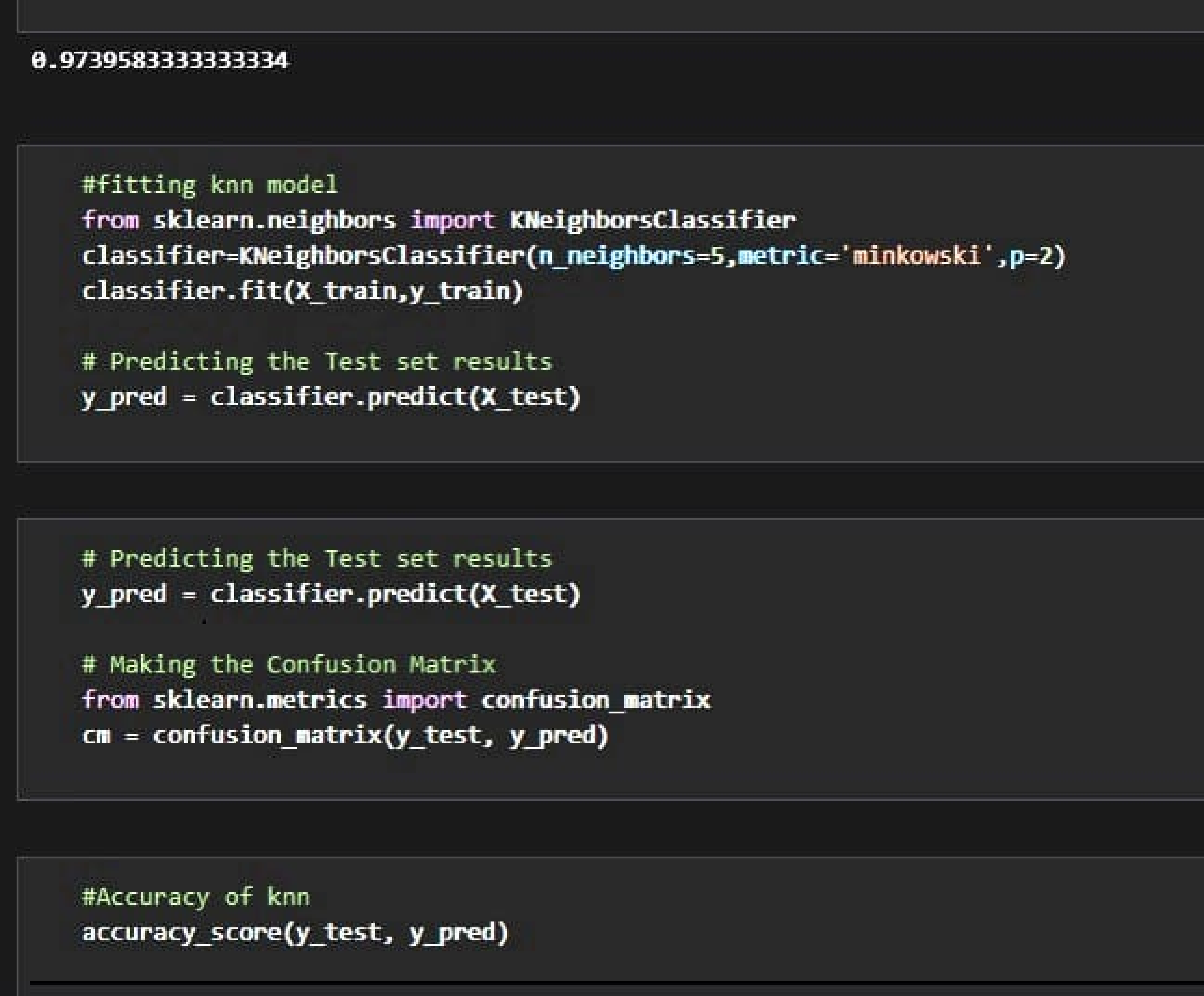
```
0.9895833333333334
```

```python
# Fitting SVM to the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)

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

```python
#Accuracy of SVM
accuracy_score(y_test, y_pred)
```

```
0.9739583333333334
```



```
0.9739583333333334
```

```python
#fitting knn model
from sklearn.neighbors import KNeighborsClassifier
classifier=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
classifier.fit(X_train,y_train)

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

```python
# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

```python
#Accuracy of knn
accuracy_score(y_test, y_pred)
```