Data Science with Python Module 10

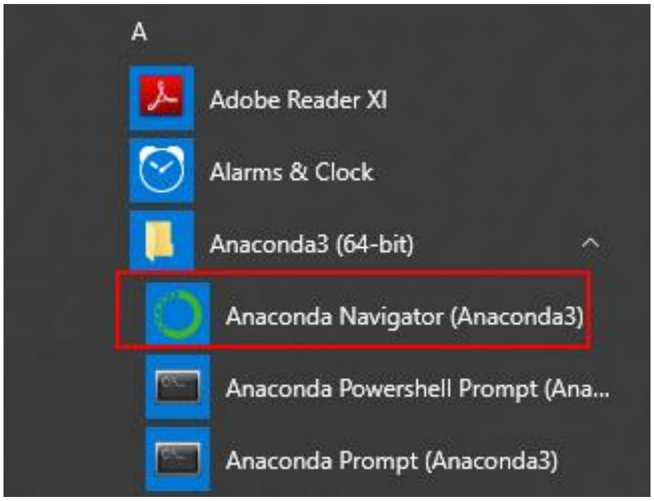
Hands On - 3

support@intellipaat.com
+91-7022374614
US: 1-800-216-8930(Toll Free)

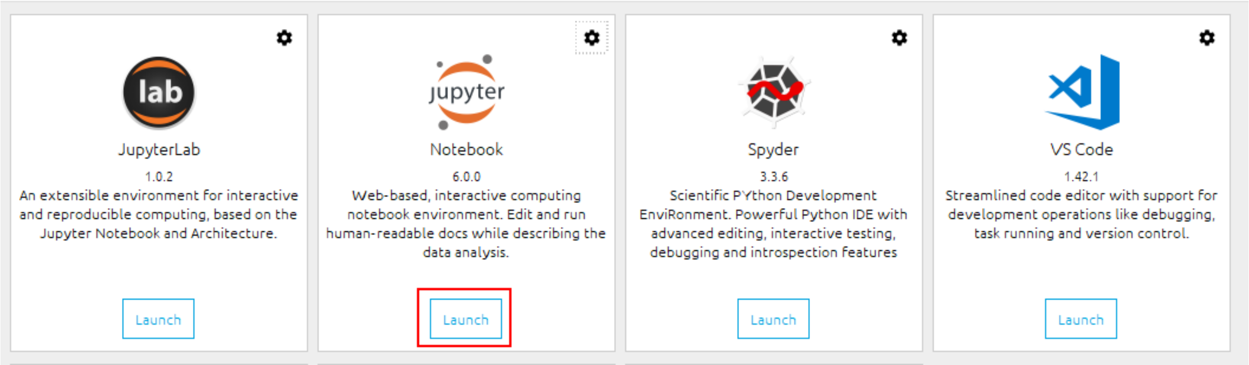
Data Science with Python Module 10: Hands-on: 3

Linear Discriminant Analysis

Step 1: Open Anaconda Navigator


A
Adobe Reader XI
Alarms & Clock
Anaconda3 (64-bit)
Anaconda Navigator (Anaconda3)
Anaconda Powershell Prompt (Ana...
Anaconda Prompt (Anaconda3)

Step 2: Click on Launch button under jupyter notebooks.



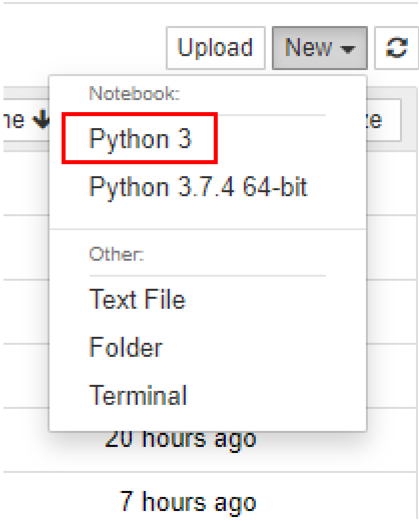
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

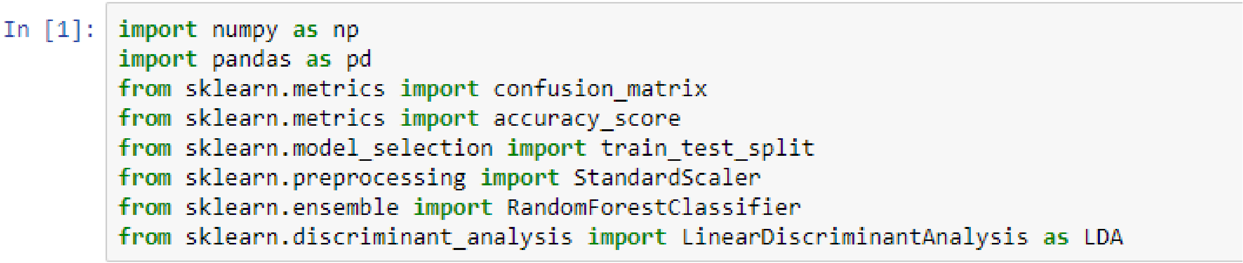
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Streamlined code editor with support for development operations like debugging, task running and version control.

Step 3: After the notebook opens click on new and Python 3.

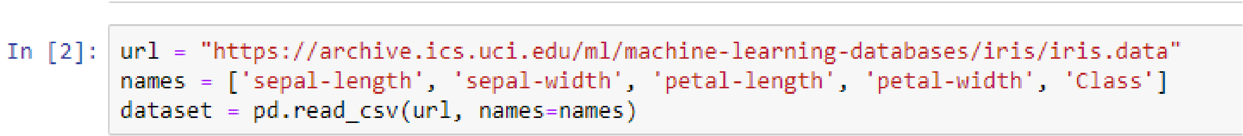


Step 4: Import all the required modules by typing the following code in the notebook and run it by pressing shift + enter



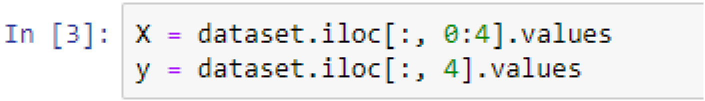
```
In [1]: import numpy as np
        import pandas as pd
        from sklearn.metrics import confusion_matrix
        from sklearn.metrics import accuracy_score
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
```

Step 5: Load the iris dataset.



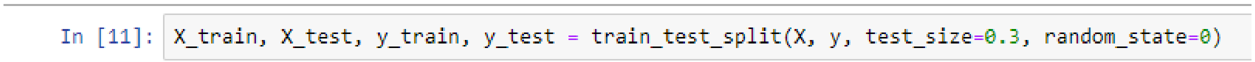
```
In [2]: url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
        names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
        dataset = pd.read_csv(url, names=names)
```

Step 6: Extract X and Y variables out of the dataset.



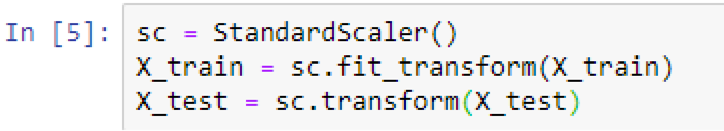
```
In [3]: X = dataset.iloc[:, 0:4].values
        y = dataset.iloc[:, 4].values
```

Step 7: Split the data into 70 percent for training and 30 percent testing.



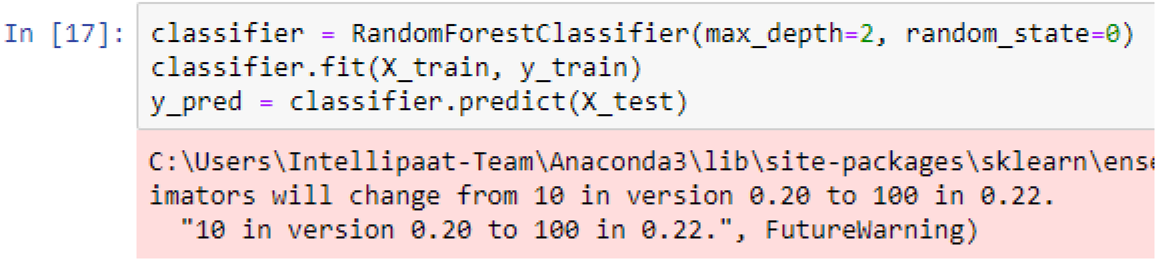
```
In [11]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

Step 8: Scale the data.

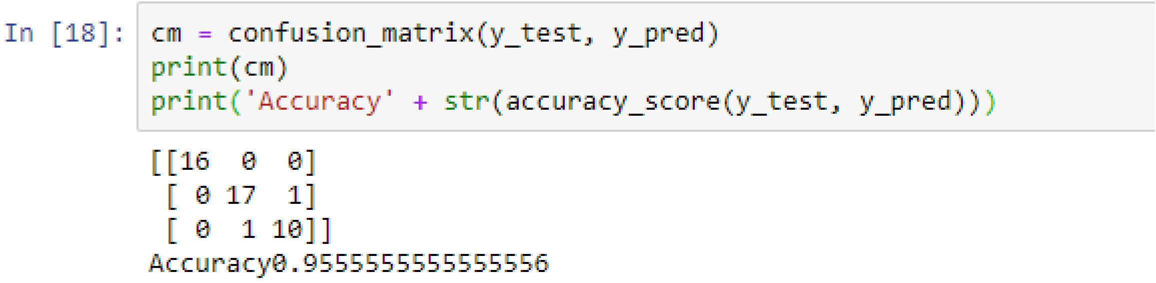


```
In [5]: sc = StandardScaler()
        X_train = sc.fit_transform(X_train)
        X_test = sc.transform(X_test)
```

Step 9: Create a RandomForestCLassifier train it on scaled data and print its accuracy score and confusion matrix.

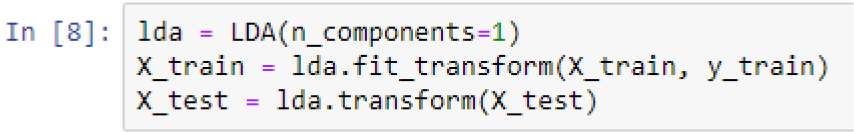


```
In [17]: classifier = RandomForestClassifier(max_depth=2, random_state=0)
         classifier.fit(X_train, y_train)
         y_pred = classifier.predict(X_test)

         C:\Users\Intellipaat-Team\Anaconda3\lib\site-packages\sklearn\ense
         imators will change from 10 in version 0.20 to 100 in 0.22.
           "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```



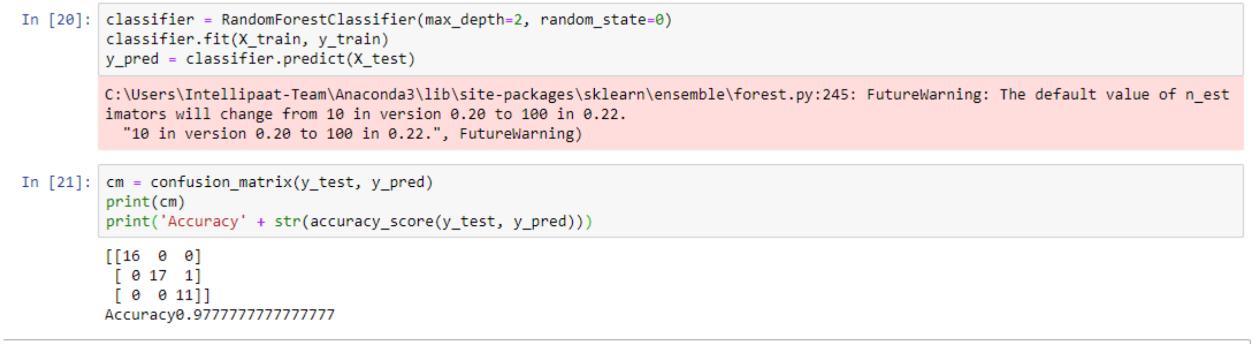
```
In [18]: cm = confusion_matrix(y_test, y_pred)
         print(cm)
         print('Accuracy' + str(accuracy_score(y_test, y_pred)))

         [[16  0  0]
          [ 0 17  1]
          [ 0  1 10]]
         Accuracy0.9555555555555556
```

Step 9: Create n LDA instance and transform x_train and x_test.



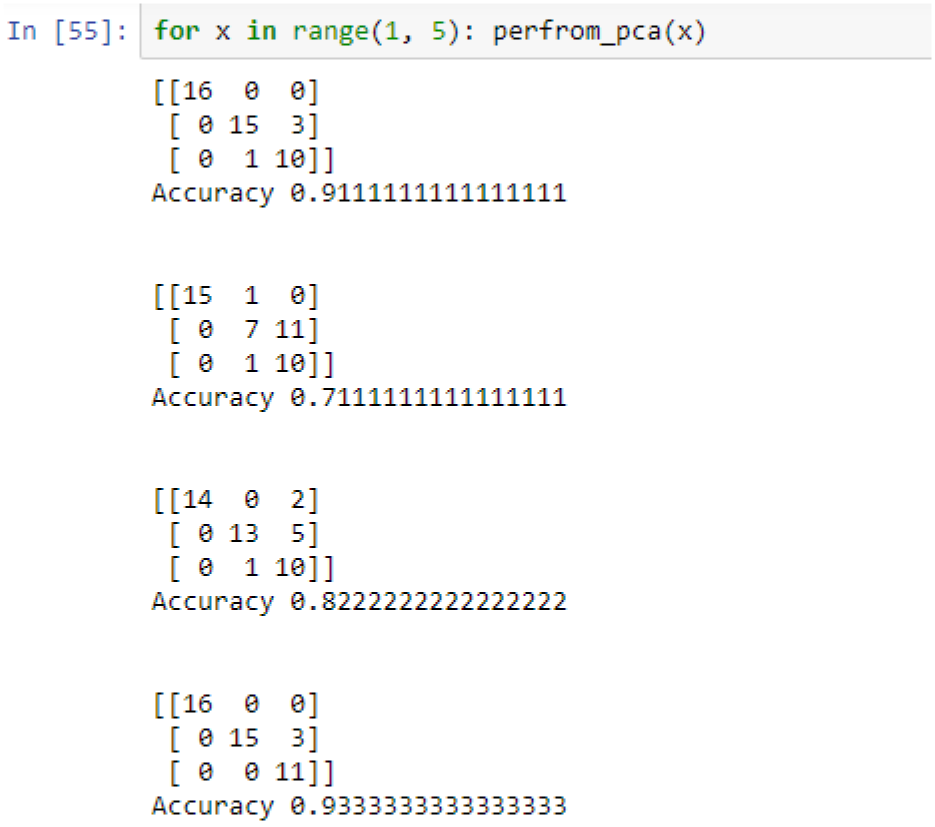
```
In [8]: lda = LDA(n_components=1)
        X_train = lda.fit_transform(X_train, y_train)
        X_test = lda.transform(X_test)
```

Step 10: Create a RandomForestCLassifier train it on scaled and transformed data and print its accuracy score and confusion matrix..



```
In [20]: classifier = RandomForestClassifier(max_depth=2, random_state=0)
         classifier.fit(X_train, y_train)
         y_pred = classifier.predict(X_test)

C:\Users\Intellipaat-Team\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_est
imators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

In [21]: cm = confusion_matrix(y_test, y_pred)
         print(cm)
         print('Accuracy' + str(accuracy_score(y_test, y_pred)))

[[16  0  0]
 [ 0 17  1]
 [ 0  0 11]]
Accuracy0.9777777777777777
```

Step 12: Call the perform_pca method with n_components set to a number from 1 to 4 and print their confusion matrix and accuracy scores.



```
In [55]: for x in range(1, 5): perfrom_pca(x)

         [[16  0  0]
          [ 0 15  3]
          [ 0  1 10]]
         Accuracy 0.9111111111111111


         [[15  1  0]
          [ 0  7 11]
          [ 0  1 10]]
         Accuracy 0.7111111111111111


         [[14  0  2]
          [ 0 13  5]
          [ 0  1 10]]
         Accuracy 0.8222222222222222


         [[16  0  0]
          [ 0 15  3]
          [ 0  0 11]]
         Accuracy 0.9333333333333333
```