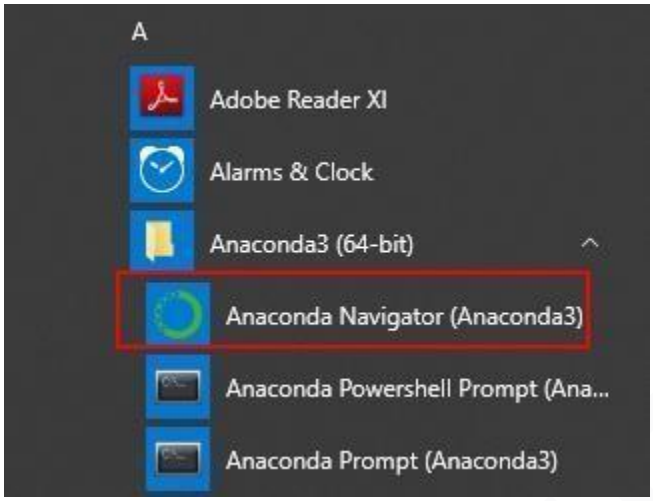
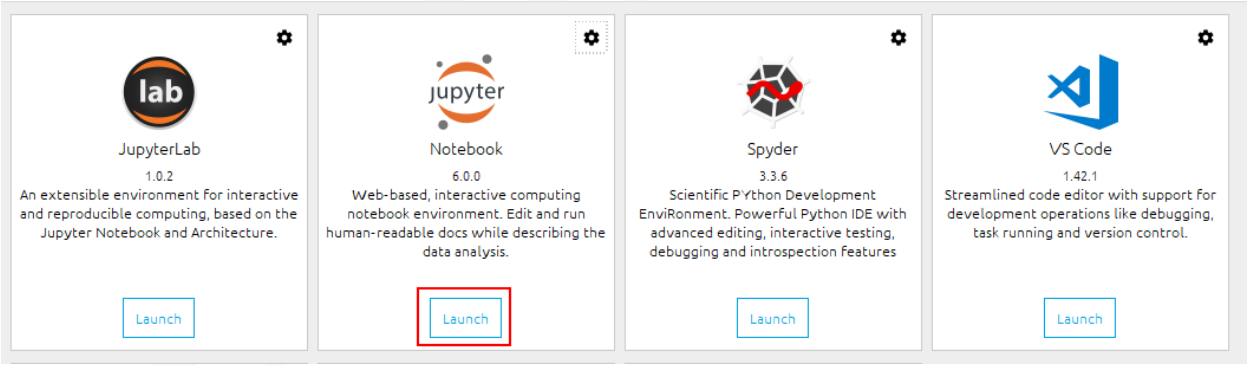
Module 7: Hands-On: 7

Creating and testing model:

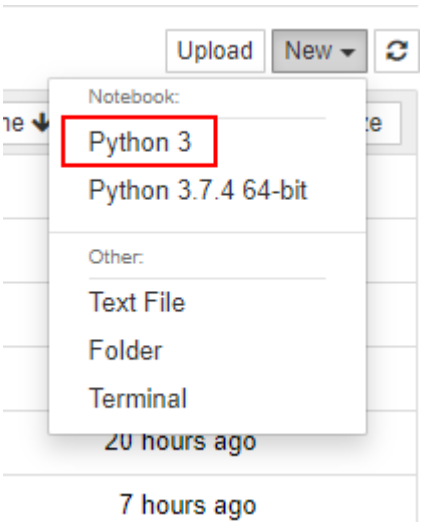
Step 1: Open Anaconda Navigator



Step 2: Click on Launch button under Jupyter Notebook



Step 3: After the notebook opens click on New and Python 3



Step 4: Import the required packages and read data from iris.csv in a DataFrame

```
In [16]: import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LogisticRegression
         from sklearn.metrics import accuracy_score, confusion_matrix
         import seaborn as sns

In [2]:  data = pd.read_csv('iris.csv')

In [3]:  data.head()
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Step 5: Analyze the shape of data

```
In [4]: data.shape
Out[4]: (150, 5)
```

Step 6: Separate data into X and Y variables and split them into training and testing set with 70/30 ratio

```
In [5]: X, Y = data.iloc[:, :-1], data.iloc[:, -1]
```

```
In [9]: x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=.3)
```

Step 7: Instantiate and train a logistic regression model

```
In [11]: clf = LogisticRegression()
```

```
In [12]: clf.fit(x_train, y_train)
C:\Users\Admin\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be changed
to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\Admin\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:469: FutureWarning: Default multi_class will be cha
nged to 'auto' in 0.22. Specify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
Out[12]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

Step 8: Make predictions, check the confusion matrix and count incorrect classifications and check its accuracy score

```
In [13]: y_pred = clf.predict(x_test)

In [18]: confusion_matrix(y_test, y_pred)

Out[18]: array([[16,  0,  0],
                [ 0, 13,  4],
                [ 0,  0, 12]], dtype=int64)

In [21]: accuracy_score(y_pred, y_test)

Out[21]: 0.9111111111111111
```