

# Deployment on Flask

Name, Batch code: LISUM10: 30

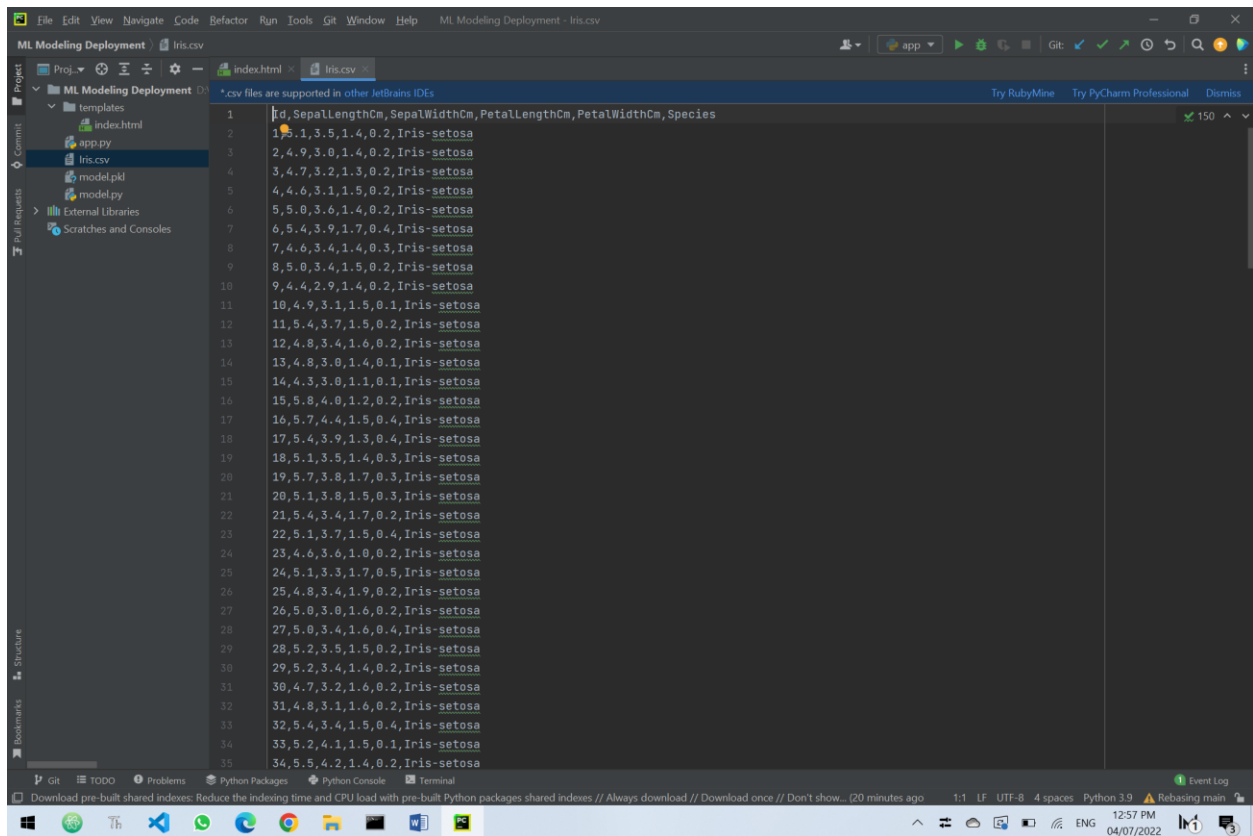
Submission date: 04/07/2022

Submitted to : Data Glacier

Done by: Shaimaa Al-khawlani

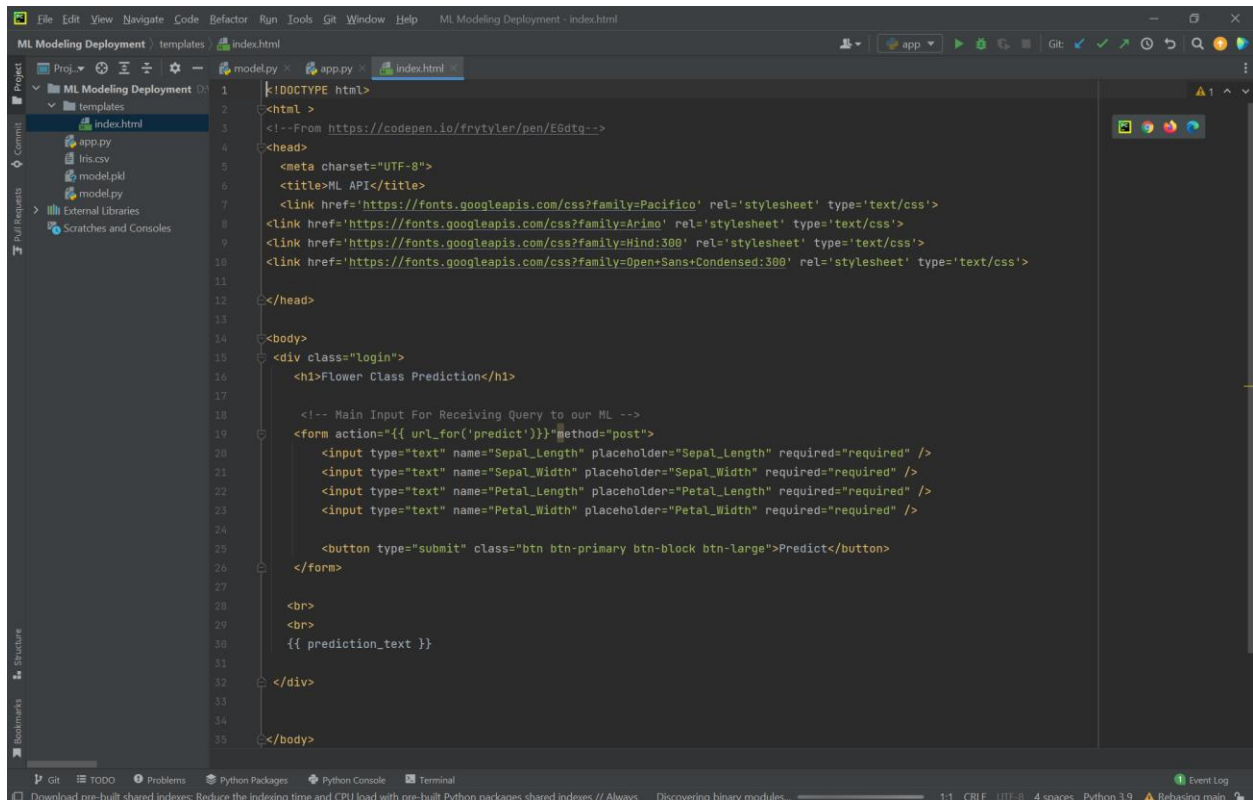
## Snapshot of each step of deployment:

### 1. Download toy data.



```
1  Id, SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm, Species
2  1, 5.1, 3.5, 1.4, 0.2, Iris-setosa
3  2, 4.9, 3.0, 1.4, 0.2, Iris-setosa
4  3, 4.7, 3.2, 1.3, 0.2, Iris-setosa
5  4, 4.6, 3.1, 1.5, 0.2, Iris-setosa
6  5, 5.0, 3.6, 1.4, 0.2, Iris-setosa
7  6, 5.4, 3.9, 1.7, 0.4, Iris-setosa
8  7, 4.6, 3.4, 1.4, 0.3, Iris-setosa
9  8, 5.0, 3.4, 1.5, 0.2, Iris-setosa
10 9, 4.4, 2.9, 1.4, 0.2, Iris-setosa
11 10, 4.9, 3.1, 1.5, 0.1, Iris-setosa
12 11, 5.4, 3.7, 1.5, 0.2, Iris-setosa
13 12, 4.8, 3.4, 1.6, 0.2, Iris-setosa
14 13, 4.8, 3.0, 1.4, 0.1, Iris-setosa
15 14, 4.3, 3.0, 1.1, 0.1, Iris-setosa
16 15, 5.8, 4.0, 1.2, 0.2, Iris-setosa
17 16, 5.7, 4.4, 1.5, 0.4, Iris-setosa
18 17, 5.4, 3.9, 1.3, 0.4, Iris-setosa
19 18, 5.1, 3.5, 1.4, 0.3, Iris-setosa
20 19, 5.7, 3.8, 1.7, 0.3, Iris-setosa
21 20, 5.1, 3.8, 1.5, 0.3, Iris-setosa
22 21, 5.4, 3.4, 1.7, 0.2, Iris-setosa
23 22, 5.1, 3.7, 1.5, 0.4, Iris-setosa
24 23, 4.6, 3.6, 1.0, 0.2, Iris-setosa
25 24, 5.1, 3.3, 1.7, 0.5, Iris-setosa
26 25, 4.8, 3.4, 1.9, 0.2, Iris-setosa
27 26, 5.0, 3.0, 1.6, 0.2, Iris-setosa
28 27, 5.0, 3.4, 1.6, 0.4, Iris-setosa
29 28, 5.2, 3.5, 1.5, 0.2, Iris-setosa
30 29, 5.2, 3.4, 1.4, 0.2, Iris-setosa
31 30, 4.7, 3.2, 1.6, 0.2, Iris-setosa
32 31, 4.8, 3.1, 1.6, 0.2, Iris-setosa
33 32, 5.4, 3.4, 1.5, 0.4, Iris-setosa
34 33, 5.2, 4.1, 1.5, 0.1, Iris-setosa
35 34, 5.5, 4.2, 1.4, 0.2, Iris-setosa
```

2. Create HTML index page which contains the input texts of each feature in the toy data.



The screenshot shows a code editor with a dark theme. The file explorer on the left shows a project named 'ML Modeling Deployment' with a subdirectory 'templates' containing 'index.html'. The main editor area displays the content of 'index.html'. The code is an HTML document with a head section including a meta charset, a title 'ML API', and four Google Fonts links (Pacifico, Arimo, Hind:300, and Open+Sans+Condensed:300). The body section has a 'login' class and a heading 'Flower Class Prediction'. Below the heading is a form for receiving queries to the ML model. The form has four text inputs: 'Sepal\_Length', 'Sepal\_Width', 'Petal\_Length', and 'Petal\_Width', each with a placeholder and 'required' attribute. A 'submit' button labeled 'Predict' is at the bottom of the form. Below the form are two blank lines and a placeholder for 'prediction\_text'.

```
1 <!DOCTYPE html>
2 <html>
3   <!-- From https://codepen.io/frtytyler/pen/E6dtg-->
4   <head>
5     <meta charset="UTF-8">
6     <title>ML API</title>
7     <link href="https://fonts.googleapis.com/css?family=Pacifico" rel="stylesheet" type="text/css">
8     <link href="https://fonts.googleapis.com/css?family=Arimo" rel="stylesheet" type="text/css">
9     <link href="https://fonts.googleapis.com/css?family=Hind:300" rel="stylesheet" type="text/css">
10    <link href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" rel="stylesheet" type="text/css">
11  </head>
12  <body>
13    <div class="login">
14      <h1>Flower Class Prediction</h1>
15
16      <!-- Main Input For Receiving Query to our ML -->
17      <form action="{{ url_for('predict')}}" method="post">
18        <input type="text" name="Sepal_Length" placeholder="Sepal_Length" required="required" />
19        <input type="text" name="Sepal_Width" placeholder="Sepal_Width" required="required" />
20        <input type="text" name="Petal_Length" placeholder="Petal_Length" required="required" />
21        <input type="text" name="Petal_Width" placeholder="Petal_Width" required="required" />
22
23        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
24      </form>
25
26      <br>
27      <br>
28      {{ prediction_text }}
29    </div>
30  </body>
```

3. Create the model.py

- Read CSV file.
- Select independent and dependent variables.
- Split dataset into train and test
- Feature scaling.
- Instantiate the model.
- Fit the model
- Create pickle file.

The screenshot displays a Jupyter Notebook environment with a Python script for Iris dataset classification. The script is as follows:

```

from sklearn.model_selection import train_test_split
import pickle

# read the csv file
iris_df = pd.read_csv("iris.csv")
print(iris_df.head())

# select independent and dependent variable
x = iris_df[["SepalLengthCm", "SepalWidthCm", "PetalLengthCm", "PetalWidthCm"]]
y = iris_df["Species"]

# split the dataset into train and test
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=58)

# feature scaling
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

# instantiate the model
classifier = RandomForestClassifier()

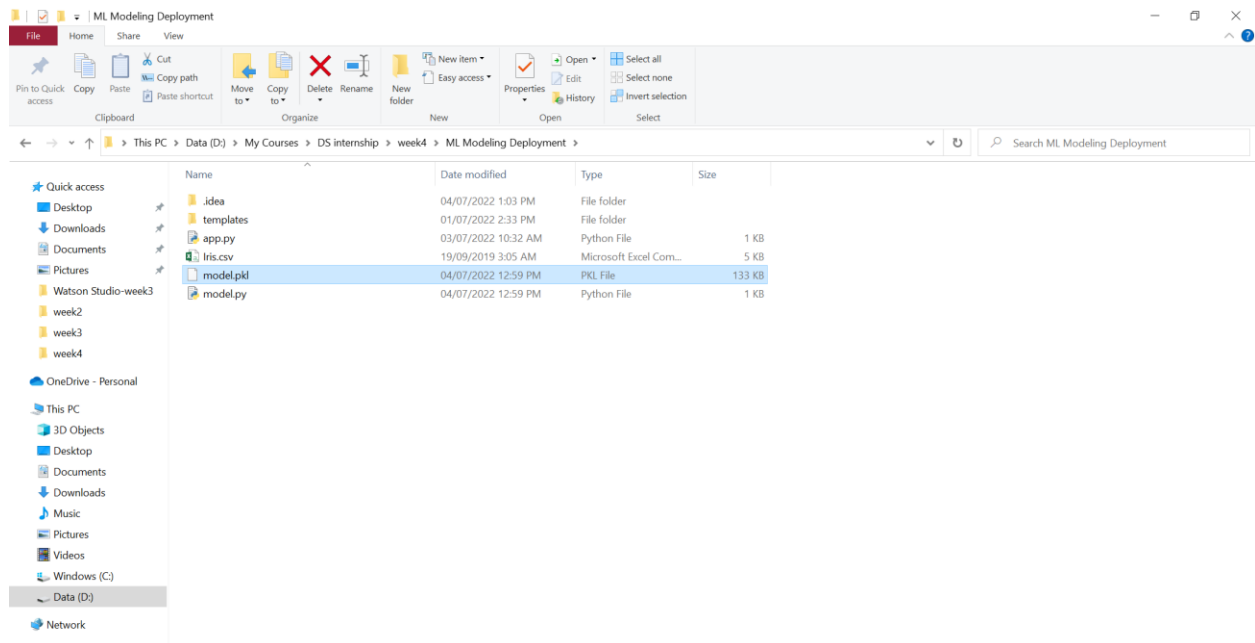
# fit the model
classifier.fit(x_train, y_train)

# make pickle file of our model
pickle.dump(classifier, open("model.pkl", "wb"))

```

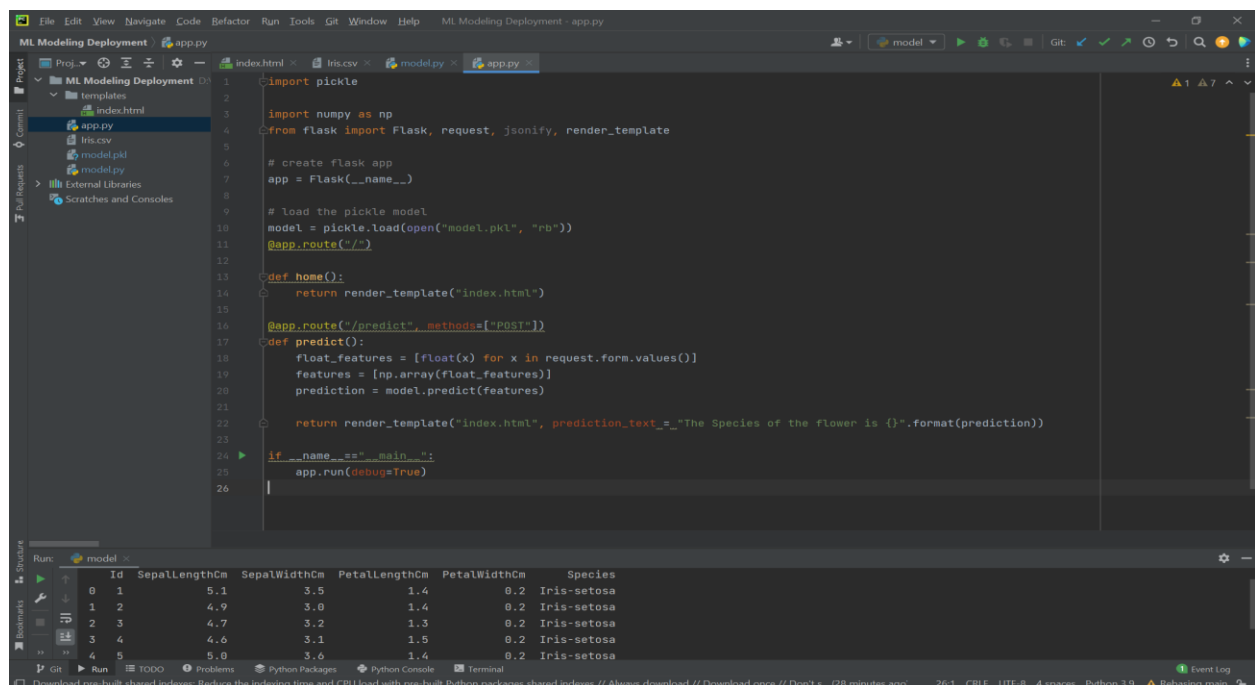
The output of the script shows the first five rows of the Iris dataset:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa



#### 4. Create app.py

- Create flask app
- Load pickle model





## Flower Class Prediction

Sepal_Length
Sepal_Width
Petal_Length
Petal_Width
Predict

The Species of the flower is ['Iris-virginica']