



Week 4 Task: Deployment on Flask

Name: Harshad Chormare

Batch code: LISUM19

Submission date: 28/03/2023

Submitted to: Data Glacier

Dataset used: iris flower data set

ML Model: RandomForest Classifier

A) Building a Model

```
In [2]: 1 import pandas as pd
        2 from sklearn.preprocessing import StandardScaler
        3 from sklearn.ensemble import RandomForestClassifier
        4 from sklearn.model_selection import train_test_split
        5 import pickle

In [7]: 1 # load the csv file
        2 df = pd.read_csv("C://ML Model Deployment//IRIS.csv")

In [8]: 1 print(df.head())
        2
        3     sepal_length  sepal_width  petal_length  petal_width  class
        4     0         5.1           3.5           1.4           0.2  Setosa
        5     1         4.9           3.0           1.4           0.2  Setosa
        6     2         4.7           3.2           1.3           0.2  Setosa
        7     3         4.6           3.1           1.5           0.2  Setosa
        8     4         5.0           3.6           1.4           0.2  Setosa

In [9]: 1 # select independent and dependent variable
        2 X = df[["sepal_length", "sepal_width", "petal_length", "petal_width"]]
        3 Y = df["class"]
        4

In [10]: 1 # Split the dataset into train and test
        2 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=50)
        3

In [11]: 1 # feature selection
        2 sc = StandardScaler()
        3 X_train = sc.fit_transform(X_train)
        4 X_test = sc.fit_transform(X_test)
        5
```

```

In [15]: 1 # Instantiate the model
          2 classifier = RandomForestClassifier()
          3

In [18]: 1 # fit the model
          2 classifier.fit(X_train, Y_train)

Out[18]: RandomForestClassifier()

In [19]: 1 # Make pickle file of the model
          2 pickle.dump(classifier, open("model.pkl", "wb"))
          3

In [ ]: 1

```

B) Model to Web Application

App.py

```

import numpy as np
from flask import Flask, request, render_template
from flask import Flask, jsonify
import pickle

# create flask app
app = Flask(__name__)

# load the pickle model
model = pickle.load(open("model.pkl", "rb"))

@app.route("/")
def home():
    return render_template("index.html")

@app.route("/predict", methods=["POST"])
def predict():
    float_features = [float(x) for x in request.form.values()]
    features = [np.array(float_features)]
    prediction = model.predict(features)

    return render_template("index.html", prediction_text="The flower species is {}".format(prediction))

if __name__ == '__main__':
    app.run(debug=True)

```

Index.html

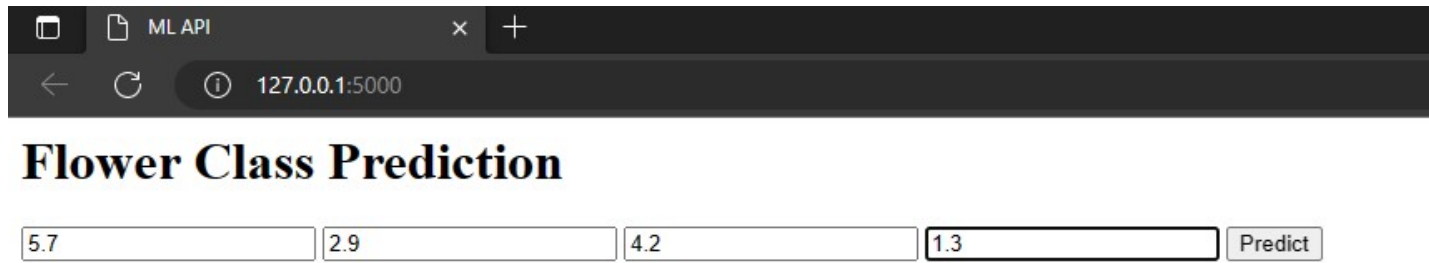
```
<html>
<!--from https://codepen.io/frytyler/pen/EGdtq-->
<head>
  <meta charset="UTF-8">
  <title> ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
</head>
<body>
<div class="login">
  <h1>Flower Class Prediction</h1>

  <!-- main Input For Receiving Query to our ML -->
  <form action="{{ url_for('predict')}}" method="post">
    <input type="text" name="Sepal_Length" placeholder="Sepal_Length" required="required" />
    <input type="text" name="Sepal_Width" placeholder="Sepal_Width" required="required" />
    <input type="text" name="Petal_Length" placeholder="Petal_Length" required="required" />
    <input type="text" name="Petal_Width" placeholder="Petal_Width" required="required" />
    <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
  </form>
  <br>
  <br>
</div>
</body>
</html>
```

C) Deployment Procedure

```
app x
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
* Debugger is active!
* Debugger PIN: 308-885-937
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

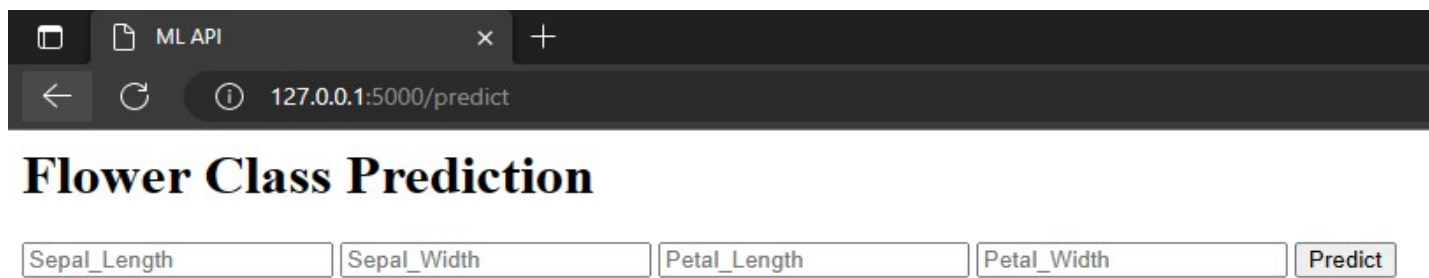
D) Web App Result



ML API

127.0.0.1:5000

Flower Class Prediction



ML API

127.0.0.1:5000/predict

Flower Class Prediction

The flower species is ['Virginica']
