Data Glacier Virtual Internship

Week 5: Cloud API Development

Name: Yusuf Yuhan

Batch Code: LISUM19

Date: 05 April 2023

Submitted to: Data Glacier

Introduction

in this project we are creating flower class prediction mechile learning app with python and flask and publish that to a Heroku could application platform.

Data information

Iris.csv

Csv file with the size of 4 kb which contains 5 attributes
Sepal_Length,Sepal_Width,Petal_Length,Petal_Width,Class
contains 151 rows of excel data

Building model model.py

Importing libraries

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import pickle
```

Load csv file

```
# Load the csv file
df = pd.read_csv("iris.csv")

print(df.head())
```

Building model

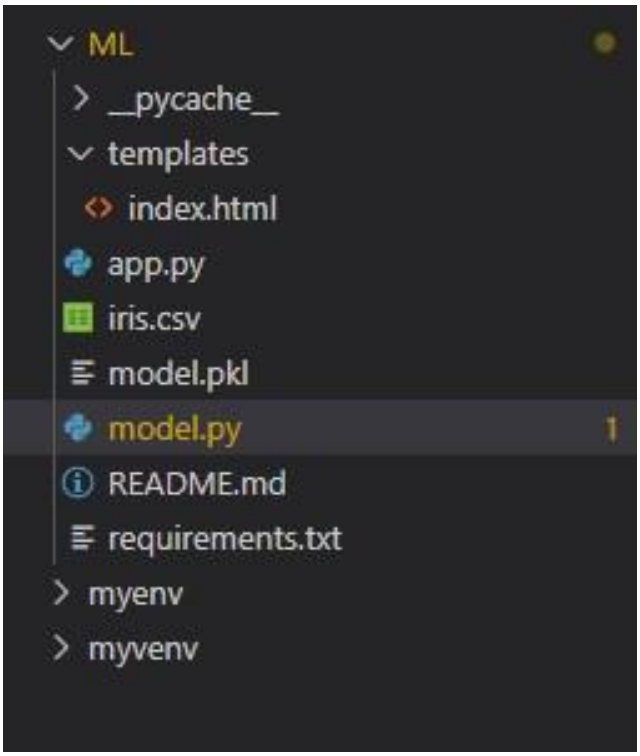
```
# Select independent and dependent variable
X = df[["Sepal_Length", "Sepal_Width", "Petal_Length", "Petal_Width"]]
y = df["Class"]

# 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=50)

# 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)
```

Save model

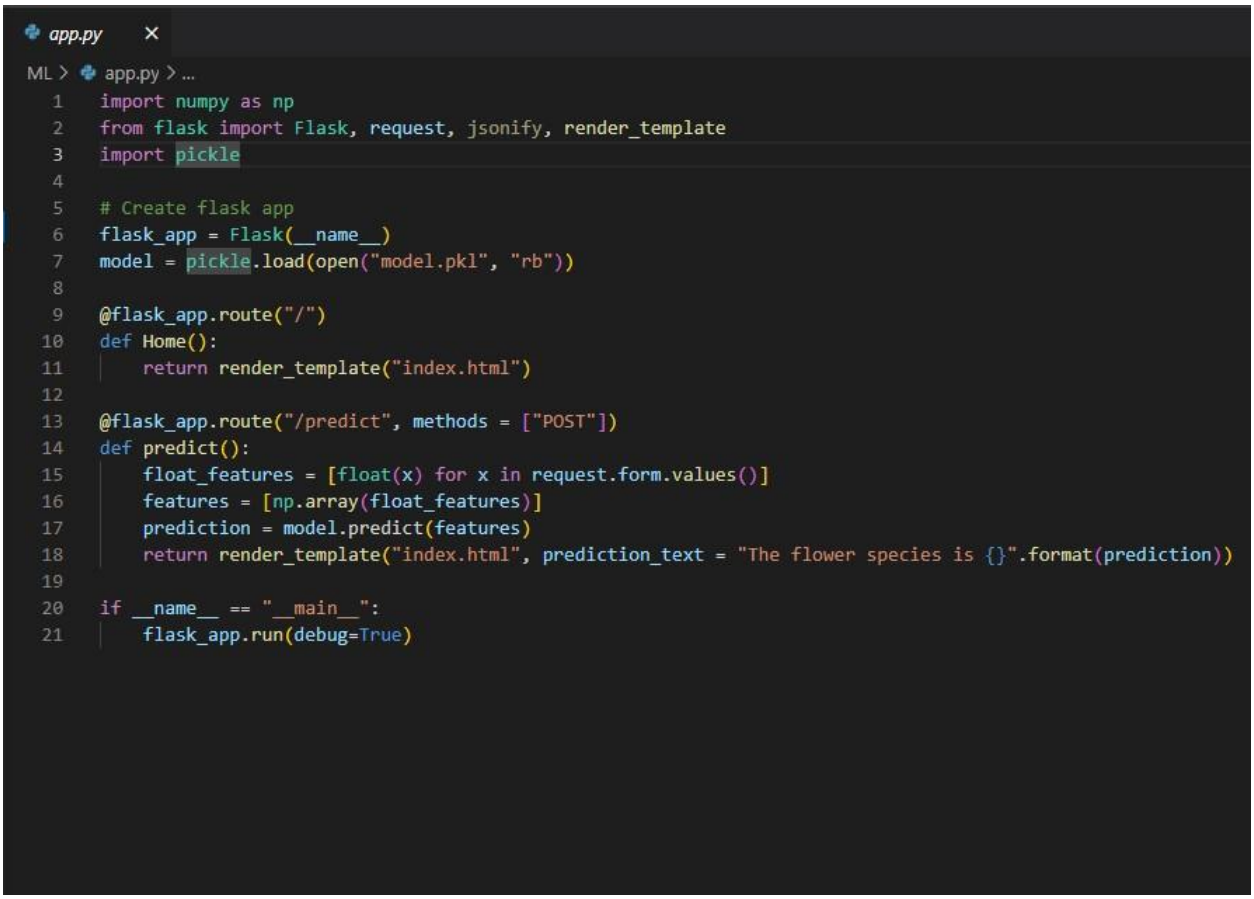
```
# Instantiate the model
classifier = RandomForestClassifier()

# Fit the model
classifier.fit(X_train, y_train)

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

Loading...

Turning model into a web application



App.py



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

# Create flask app
flask_app = Flask(__name__)
model = pickle.load(open("model.pkl", "rb"))

@flask_app.route("/")
def Home():
    return render_template("index.html")

@flask_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__":
    flask_app.run(debug=True)
```

Import necessary libraries: The code imports numpy, Flask, request, jsonify, render_template, and pickle libraries.

Load the trained model: The code loads the trained Random Forest Classifier model from the file "model.pkl" using the pickle library.

Create Flask app and routes: The code creates a Flask app and defines two routes:

The "/" route displays the homepage of the web application using an HTML template.

The "/predict" route takes the input data from a form submitted by the user, converts it to a numpy array, and uses the trained model to make a prediction for the species of iris flower. The prediction is then displayed on the homepage using another HTML template.

Define the HTML templates: The code defines two HTML templates:

"index.html" is the homepage template that displays the input form and the prediction text.

"result.html" is the result template that displays the prediction text.

Run the Flask app: The code runs the Flask app in debug mode using the "run" method.

Index.html

```
ML > templates > <> index.html > ...
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href Follow link (ctrl + click) eapis.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>
   {{ prediction_text }}

 </div>
```

Running web application in web browser using flask

```
Nabeel@mlyyuhan MINGW64 ~/downloads/ML
$ source myenv/Scripts/activate
(myenv)
Nabeel@mlyyuhan MINGW64 ~/downloads/ML
$ cd ML
(myenv)
Nabeel@mlyyuhan MINGW64 ~/downloads/ML/ML
$ pip freeze > requirements.txt
(myenv)
Nabeel@mlyyuhan MINGW64 ~/downloads/ML/ML
$ export FLASK_APP=app.py
(myenv)
Nabeel@mlyyuhan MINGW64 ~/downloads/ML/ML
$ export FLASK_ENV=development
(myenv)
Nabeel@mlyyuhan MINGW64 ~/downloads/ML/ML
$ flask run
```

Opening web browser in http://127.0.0.1:5000

```
  warnings.warn(
C:\Users\Nabeel\downloads\ML\myenv\lib\site-packages\sklearn\base.py:318: UserWarning: Trying to unpickle estimator Rand
omForestClassifier from version 0.23.1 when using version 1.2.2. This might lead to breaking code or invalid results. Us
e at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
'FLASK_ENV' is deprecated and will not be used in Flask 2.3. Use 'FLASK_DEBUG' instead.
'FLASK_ENV' is deprecated and will not be used in Flask 2.3. Use 'FLASK_DEBUG' instead.
 * Serving Flask app 'app.py'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
'FLASK_ENV' is deprecated and will not be used in Flask 2.3. Use 'FLASK_DEBUG' instead.
C:\Users\Nabeel\downloads\ML\myenv\lib\site-packages\sklearn\base.py:318: UserWarning: Trying to unpickle estimator Deci
sionTreeClassifier from version 0.23.1 when using version 1.2.2. This might lead to breaking code or invalid results. Us
e at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
C:\Users\Nabeel\downloads\ML\myenv\lib\site-packages\sklearn\base.py:318: UserWarning: Trying to unpickle estimator Rand
omForestClassifier from version 0.23.1 when using version 1.2.2. This might lead to breaking code or invalid results. Us
e at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
'FLASK_ENV' is deprecated and will not be used in Flask 2.3. Use 'FLASK_DEBUG' instead.
'FLASK_ENV' is deprecated and will not be used in Flask 2.3. Use 'FLASK_DEBUG' instead.
 * Debugger is active!
 * Debugger PIN: 724-722-346
```

Flower Class Prediction

Sepal_Length | Sepal_Width | Petal_Length | Petal_Width | Predict

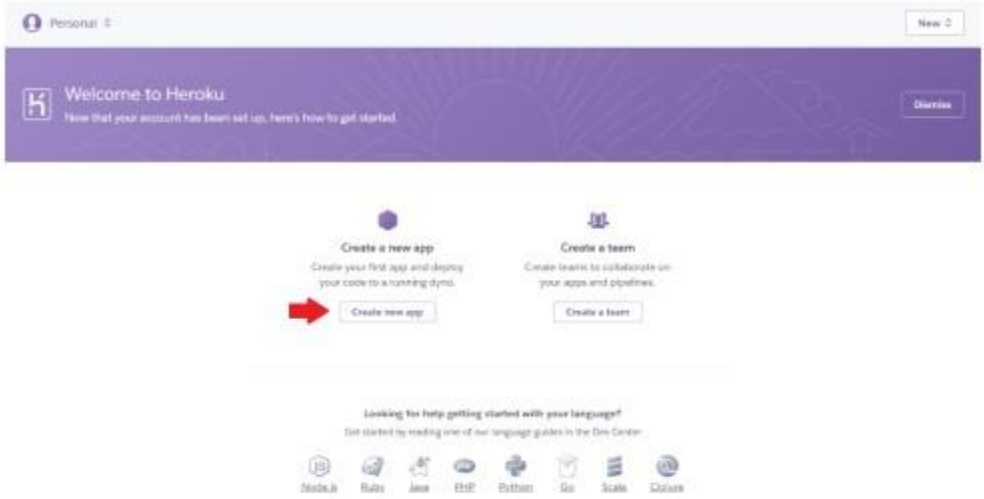
The flower species is ['Virginica']

Flower Class Prediction

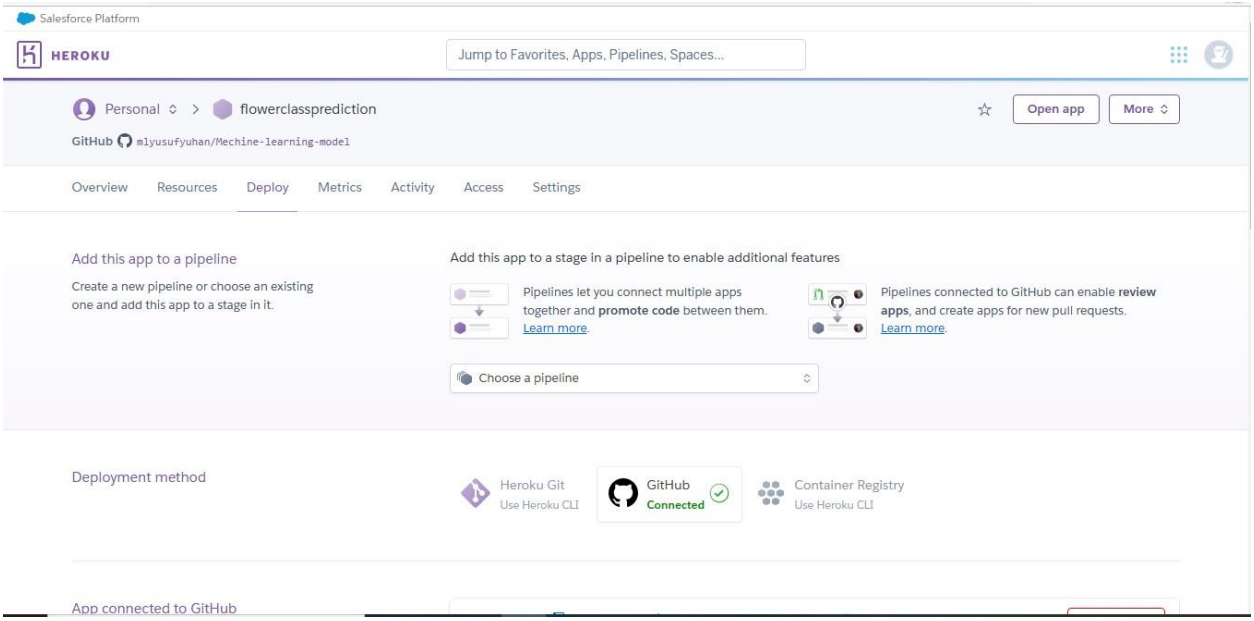
5 | 2 | 3.5 | 1 | Predict

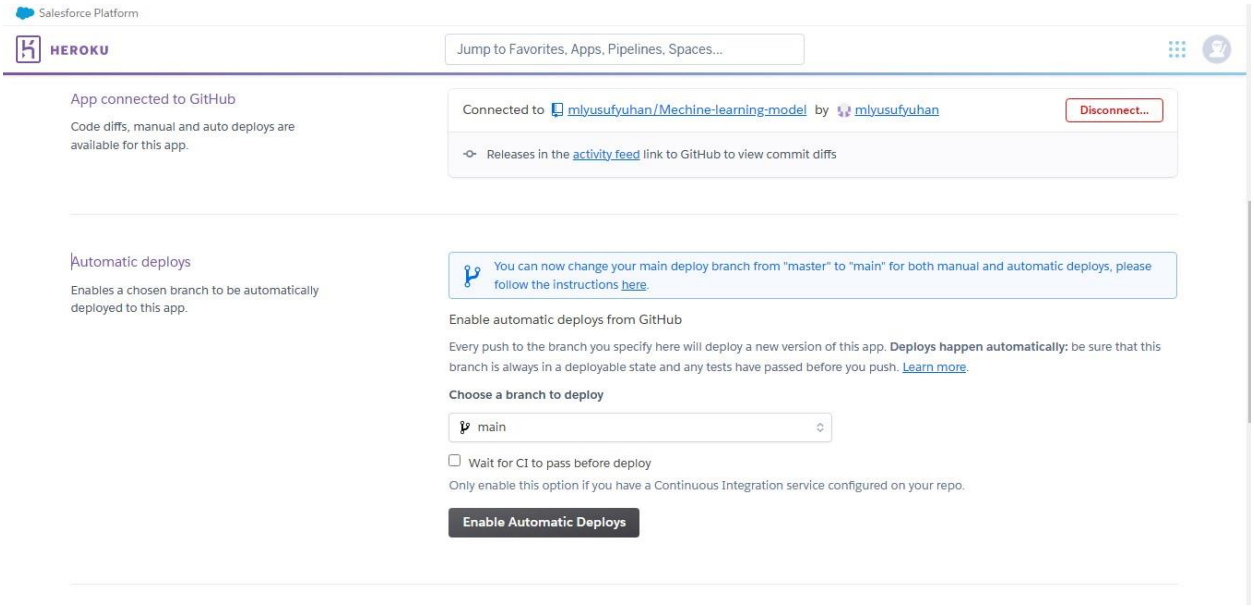
The flower species is ['Virginica']

Deploy the mechine learning model to Heroku



Connect app to git respostary





Successfully deployed app