Data Intake Report

Name: Predict iris species
Report date: 30/Jun/2022
Internship Batch: LISUM10: 30
Version:<1.0>
Data intake by: Moath Bin Musallam
Data intake reviewer:
Data storage location: local storage

Tabular data details:

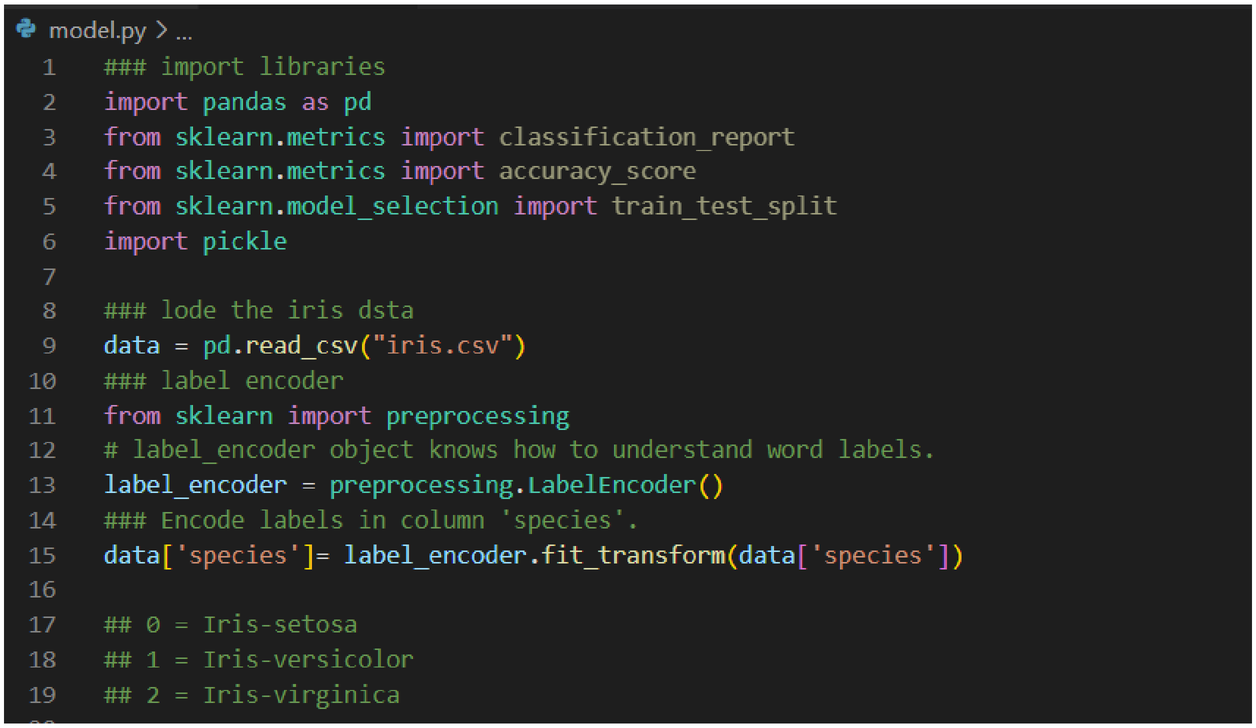
Total number of observations	150
Total number of files	1
Total number of features	5
Base format of the file	.csv
Size of the data	4,49 KB

Note: Replicate same table with file name if you have more than one file.

Proposed Approach:

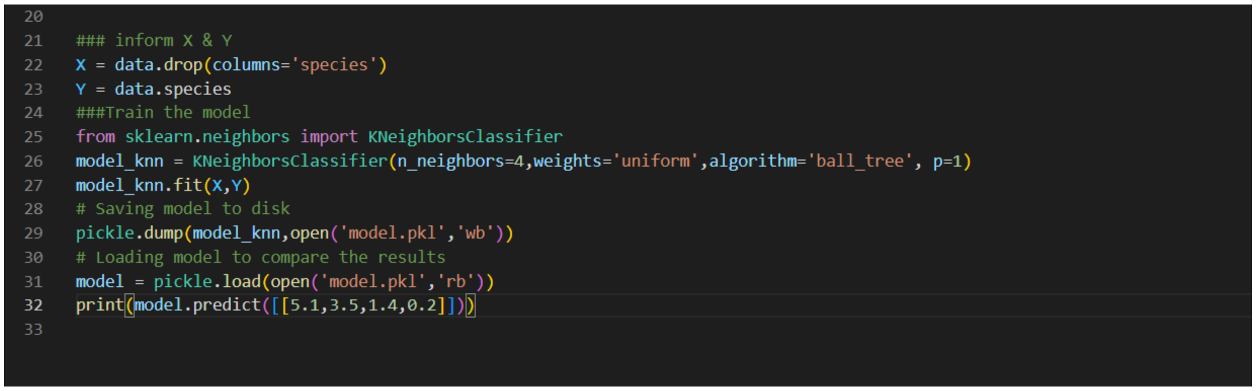
- Use the Python programming language to build the model

Build the classification model to predict iris flower:



model.py > ...

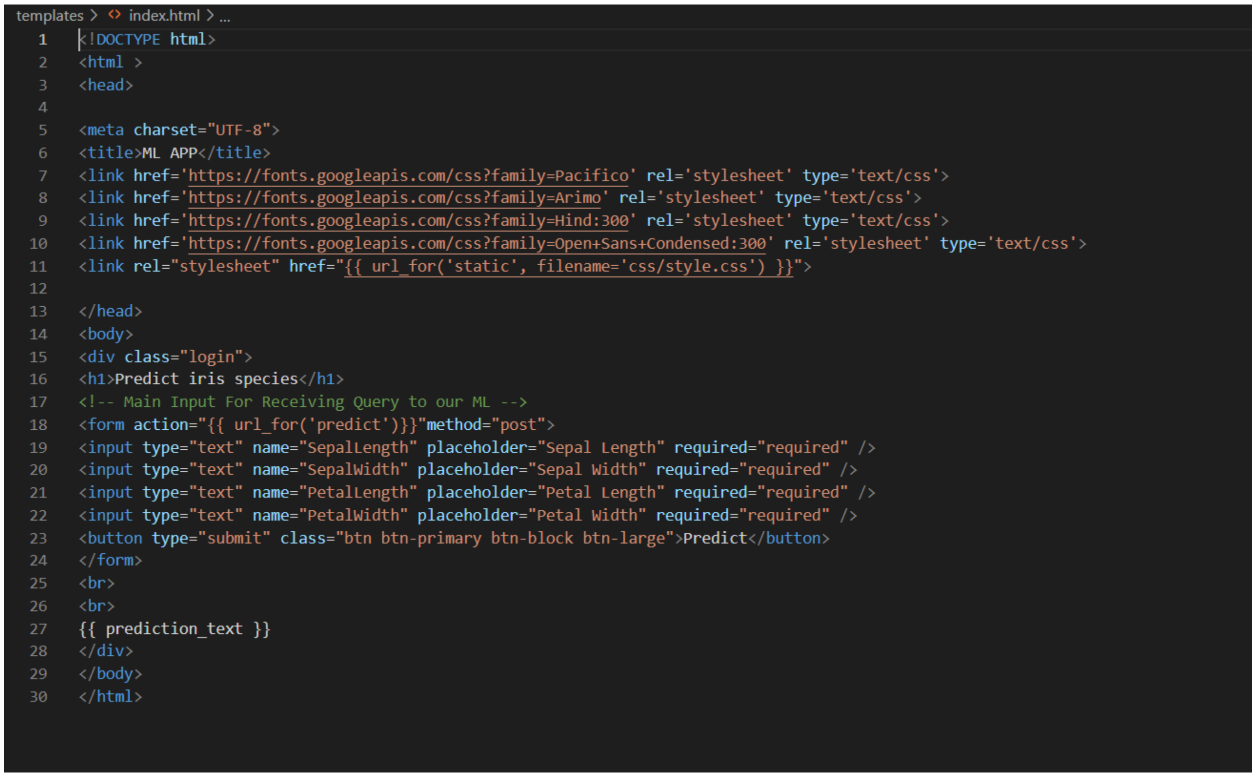
```
### import libraries
import pandas as pd
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
import pickle

### lode the iris dsta
data = pd.read_csv("iris.csv")
### label encoder
from sklearn import preprocessing
# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
### Encode labels in column 'species'.
data['species']= label_encoder.fit_transform(data['species'])

## 0 = Iris-setosa
## 1 = Iris-versicolor
## 2 = Iris-virginica
```



```python

### inform X & Y
X = data.drop(columns='species')
Y = data.species
###Train the model
from sklearn.neighbors import KNeighborsClassifier
model_knn = KNeighborsClassifier(n_neighbors=4,weights='uniform',algorithm='ball_tree', p=1)
model_knn.fit(X,Y)
# Saving model to disk
pickle.dump(model_knn,open('model.pkl','wb'))
# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[5.1,3.5,1.4,0.2]]))

```

HTML page:

Create an html template to deploy flask app. It has a text input to get the body mass and a button to send the data the user wants to predict.

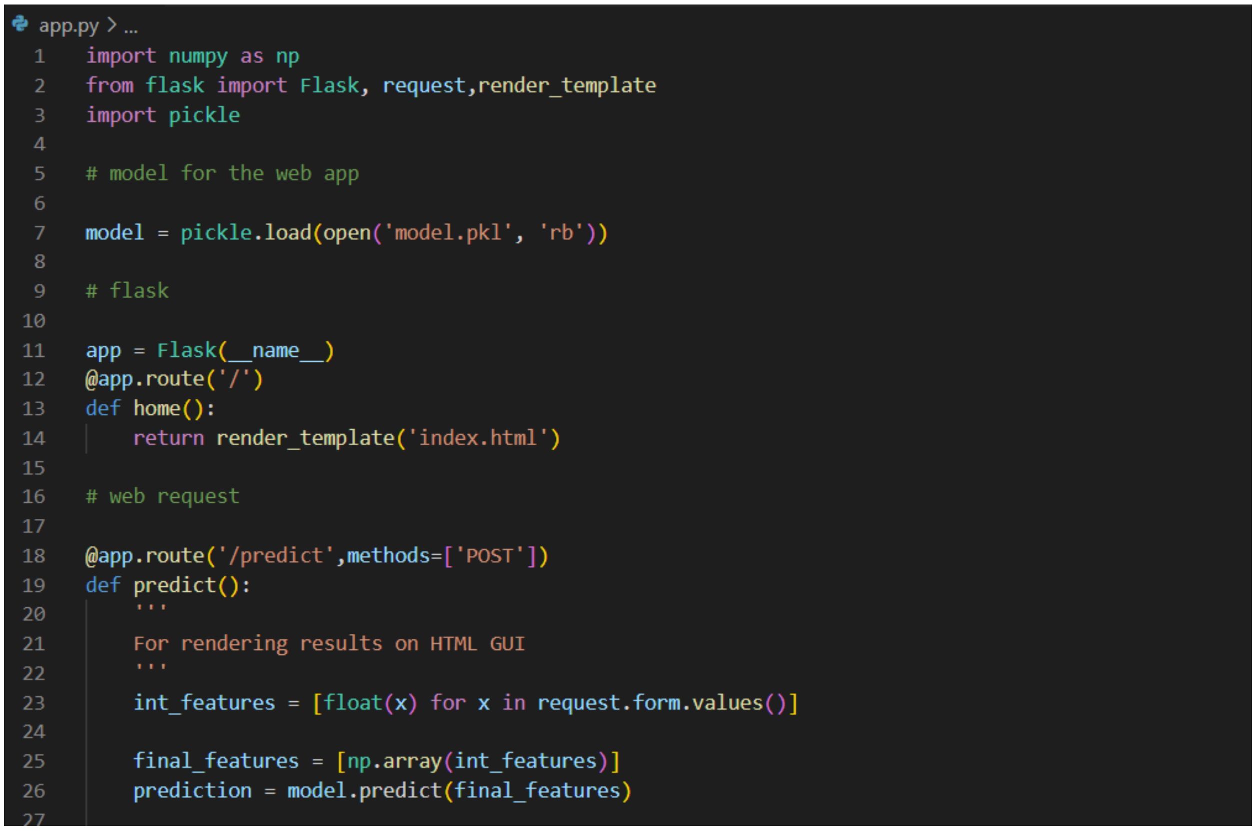


templates > index.html > ...

```html
<!DOCTYPE html>
<html >
<head>

<meta charset="UTF-8">
<title>ML APP</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'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>
<body>
<div class="login">
<h1>Predict iris species</h1>
<!-- Main Input For Receiving Query to our ML -->
<form action="{{ url_for('predict')}}"method="post">
<input type="text" name="SepalLength" placeholder="Sepal Length" required="required" />
<input type="text" name="SepalWidth" placeholder="Sepal Width" required="required" />
<input type="text" name="PetalLength" placeholder="Petal Length" required="required" />
<input type="text" name="PetalWidth" placeholder="Petal Width" required="required" />
<button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
</form>
<br>
<br>
{{ prediction_text }}
</div>
</body>
</html>
```

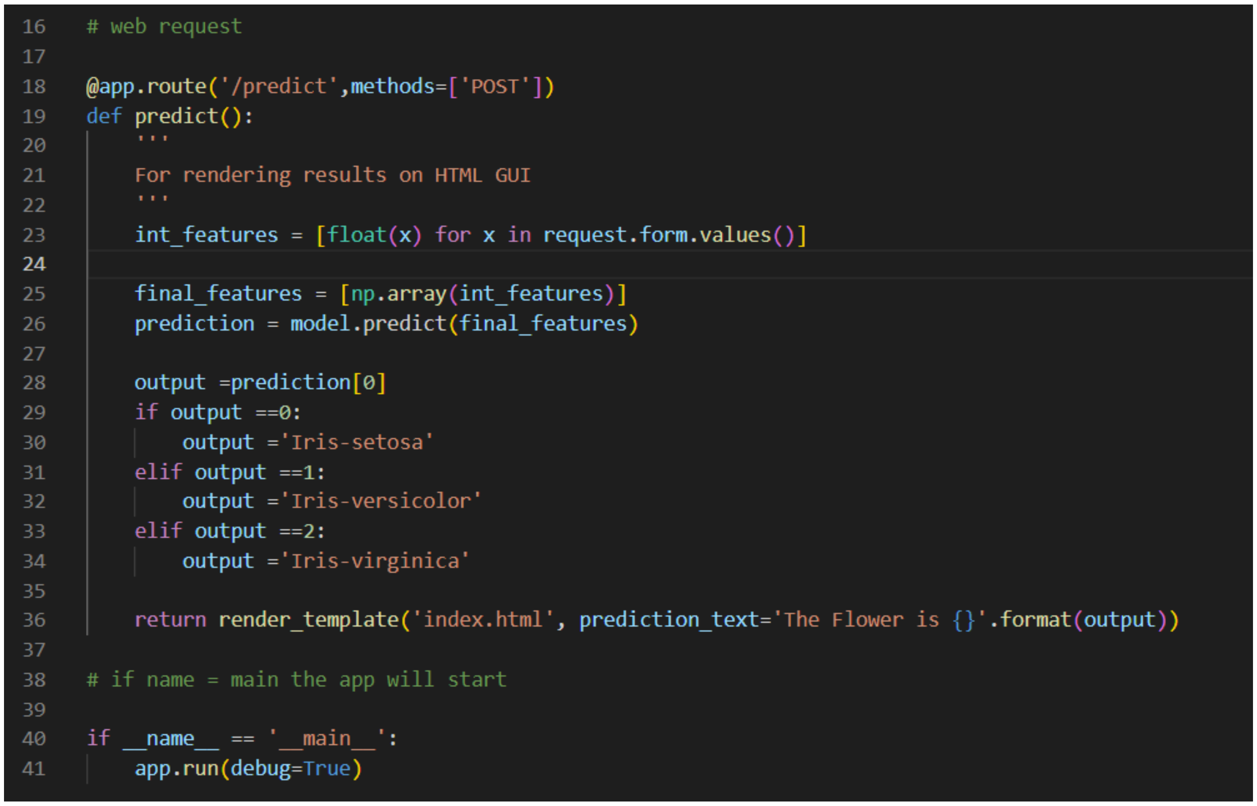
build Flask app.py:

he flask app.py file has ‘predict’ function that get the form values from the html file and predict the output from the model we saved earlier



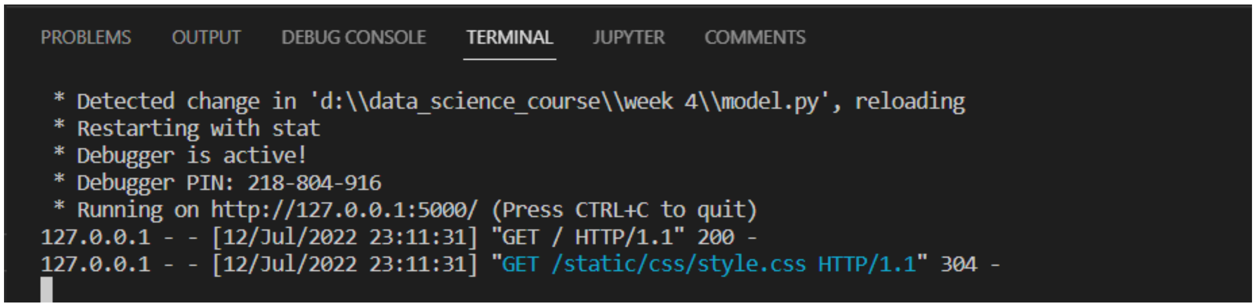
app.py > ...

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

# model for the web app

model = pickle.load(open('model.pkl', 'rb'))

# flask

app = Flask(__name__)
@app.route('/')
def home():
    return render_template('index.html')

# web request

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [float(x) for x in request.form.values()]

    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

```



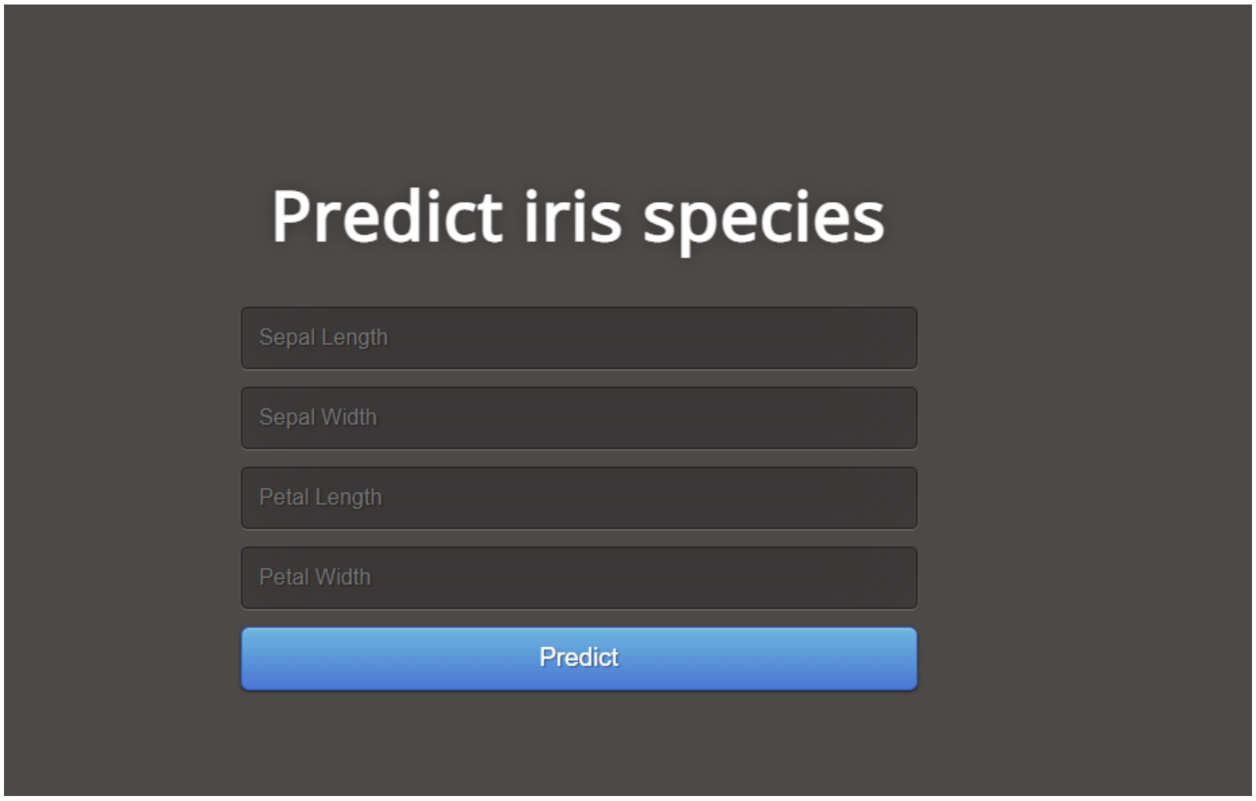
```python
# web request

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [float(x) for x in request.form.values()]

    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output =prediction[0]
    if output ==0:
        output ='Iris-setosa'
    elif output ==1:
        output ='Iris-versicolor'
    elif output ==2:
        output ='Iris-virginica'

    return render_template('index.html', prediction_text='The Flower is {}'.format(output))

# if name = main the app will start

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

Run the project:

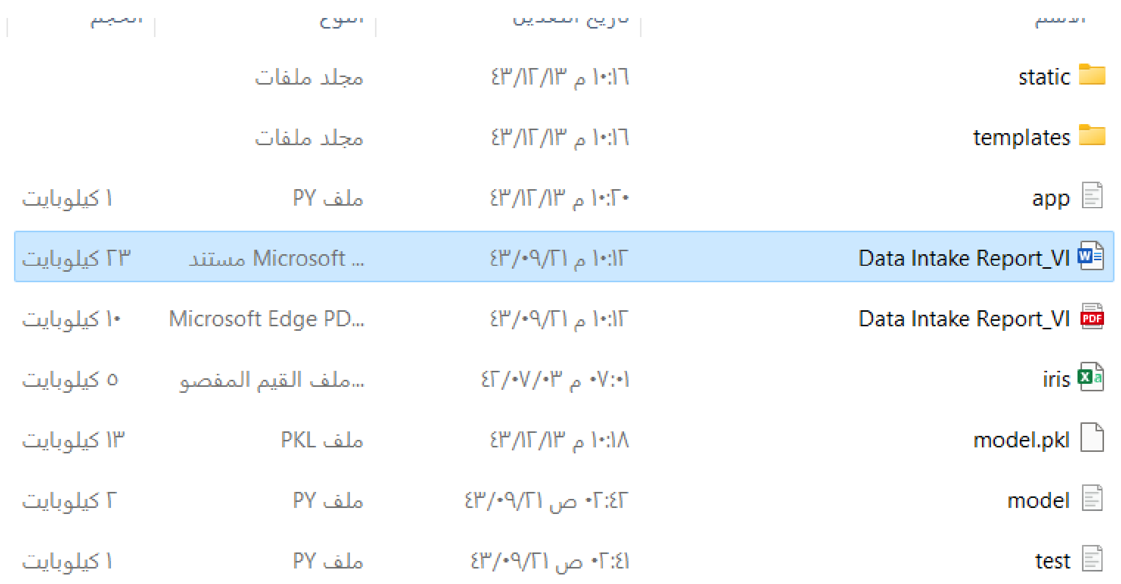


```
 * Detected change in 'd:\\data_science_course\\week 4\\model.py', reloading
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 218-804-916
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [12/Jul/2022 23:11:31] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [12/Jul/2022 23:11:31] "GET /static/css/style.css HTTP/1.1" 304 -
```

• Open the http://127.0.0.1:5000/ url and enter a value then press the predict button to get the result.



Project file:



الاسم	تاريخ التعديل	النوع	الحجم
static	١٠:١٦ م ٤٣/١٢/١٣	مجلد ملفات	
templates	١٠:١٦ م ٤٣/١٢/١٣	مجلد ملفات	
app	١٠:٢٠ م ٤٣/١٢/١٣	ملف PY	١ كيلوبايت
Data Intake Report_VI	١٠:١٢ م ٤٣/٠٩/٢١	Microsoft مستند ...	٢٣ كيلوبايت
Data Intake Report_VI	١٠:١٢ م ٤٣/٠٩/٢١	Microsoft Edge PD...	١٠ كيلوبايت
iris	٠٧:٠١ م ٤٢/٠٧/٠٣	ملف القيم المفصو...	٥ كيلوبايت
model.pkl	١٠:١٨ م ٤٣/١٢/١٣	ملف PKL	١٣ كيلوبايت
model	٠٢:٤٢ ص ٤٣/٠٩/٢١	ملف PY	٢ كيلوبايت
test	٠٢:٤١ ص ٤٣/٠٩/٢١	ملف PY	١ كيلوبايت