

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

Machine learning model:

Using knn because it is better model.

```
#split to test the model
# from sklearn.model_selection import train_test_split

#wit stratification to balance the output y
# X_train,X_test, y_train, y_test= train_test_split (X,Y, test_size=0.3,random_state=1,stratify=Y)

###Train the model
from sklearn.neighbors import KNeighborsClassifier

model_knn = KNeighborsClassifier(n_neighbors=4,weights='uniform',algorithm='ball_tree', p=1)

# model.fit(X_train, y_train) #Training the model
# #Test the model
# predictions = model.predict(X_test)
# print( classification_report(y_test, predictions) )
# print( accuracy_score(y_test, predictions) )
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]]))
```

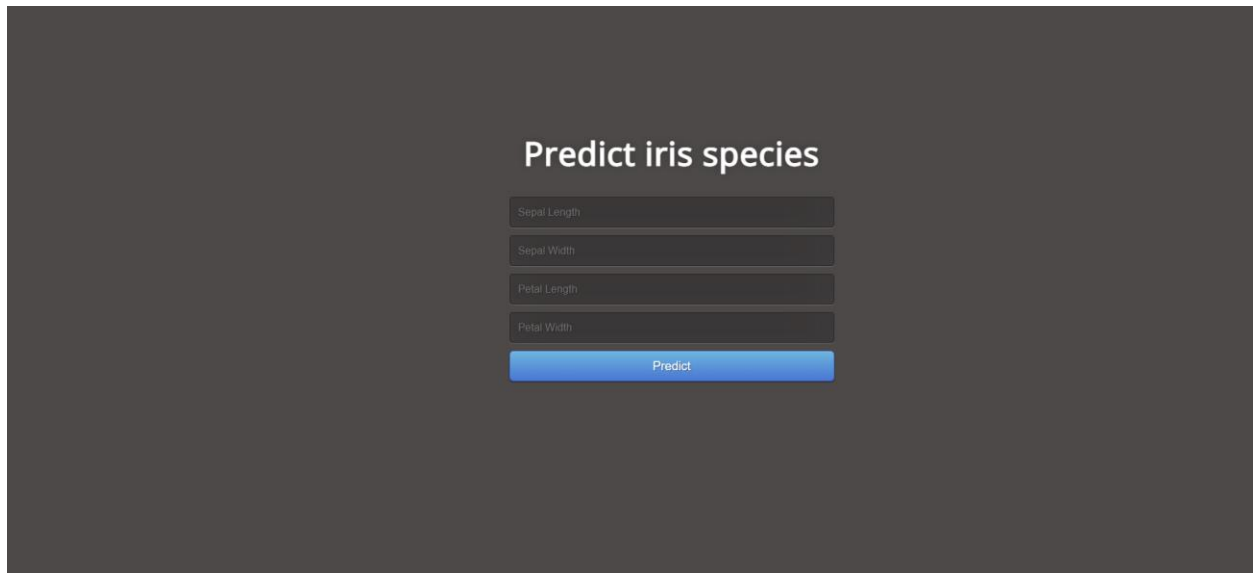
Flask app:

```
app.py > home
4
5  model = pickle.load(open('model.pkl', 'rb'))
6
7
8  app = Flask(__name__)
9  @app.route('/')
10 def home():
11     return render_template('index.html')
12
13 @app.route('/predict',methods=['POST'])
14 def predict():
15     '''
16     For rendering results on HTML GUI
17     '''
18     int_features = [float(x) for x in request.form.values()]
19
20     final_features = [np.array(int_features)]
21     prediction = model.predict(final_features)
22
23     output =prediction[0]
24     if output ==0:
25         output = 'Iris-setosa'
26     elif output ==1:
27         output = 'Iris-versicolor'
28     elif output ==2:
29         output = 'Iris-virginica'
30
31     return render_template('index.html', prediction_text='The Flower is {}'.format(output))
```

HTML:

```
app.py  requirements.txt  index.html X  Procfile  # style.css  model.py  test.py
templates > index.html > html > body
1  <!DOCTYPE html>
2  <html >
3  <head>
4
5  <meta charset="UTF-8">
6  <title>ML APP</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 <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
12
13 </head>
14 <body>
15 <div class="login">
16 <h1>Predict iris species</h1>
17 <!-- Main Input For Receiving Query to our ML -->
18 <form action="{{ url_for('predict')}}"method="post">
19 <input type="text" name="SepalLength" placeholder="Sepal Length" required="required" />
20 <input type="text" name="SepalWidth" placeholder="Sepal Width" required="required" />
21 <input type="text" name="PetalLength" placeholder="Petal Length" required="required" />
22 <input type="text" name="PetalWidth" placeholder="Petal Width" required="required" />
23 <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
24 </form>
25 <br>
26 <br>
```

final product:
after deploying the project to Heroku



The image shows a web application interface for predicting iris species. It features a dark gray background with a white title "Predict iris species" centered at the top. Below the title, there are four input fields stacked vertically, each with a light gray border and a small label inside: "Sepal Length", "Sepal Width", "Petal Length", and "Petal Width". At the bottom of the input fields is a blue button with the word "Predict" in white text.

Create API method

```
@app.route('/predict_API')
def predict_API():
    ...

    For rendering results on postman
    ...

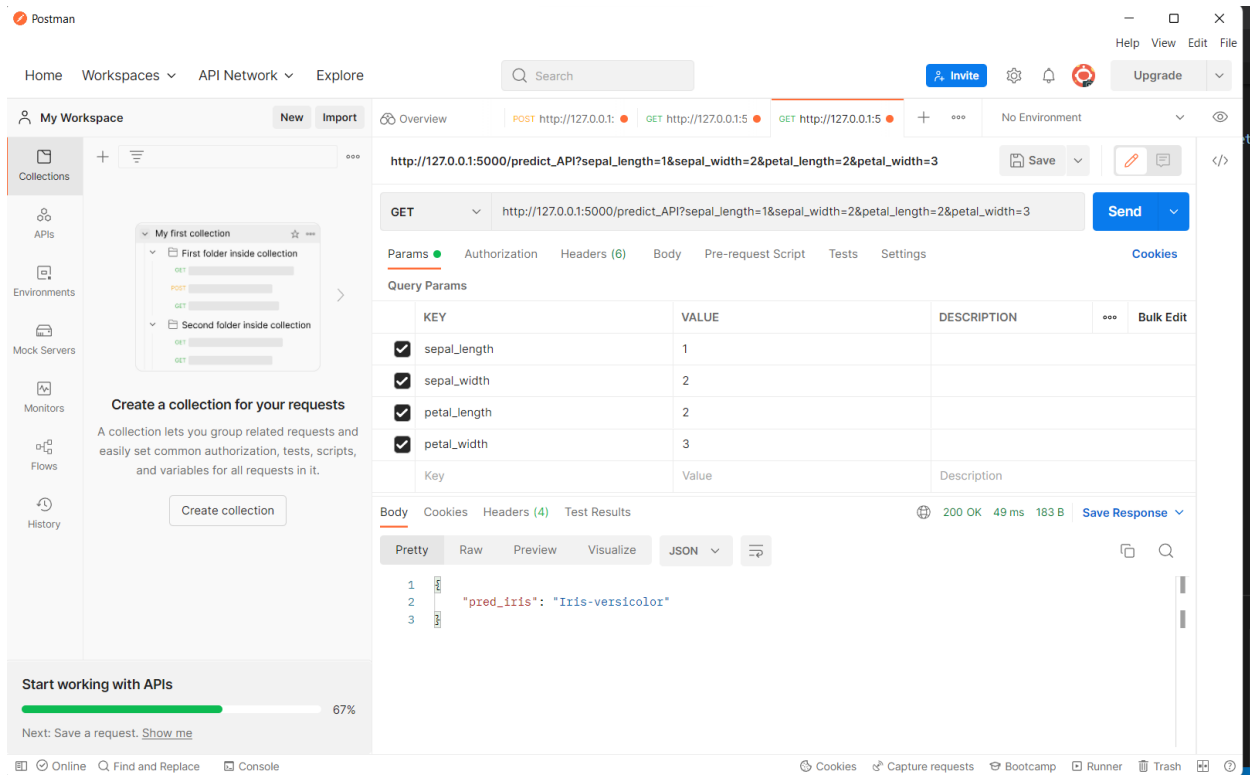
    model = pickle.load(open('model.pkl', 'rb'))
    sepal_length = request.args.get('sepal_length')
    sepal_width = request.args.get('sepal_width')
    petal_length = request.args.get('petal_length')
    petal_width = request.args.get('petal_width')
    test_df = pd.DataFrame({'sepal_length':[sepal_length],
    'sepal_width':[sepal_width], 'petal_length':[petal_length],
    'petal_width':[petal_width]})

    pred_iris = model.predict(test_df).tolist()
    output =pred_iris[0]
    if output ==0:
        output = 'Iris-setosa'
    elif output ==1:
        output = 'Iris-versicolor'
    elif output ==2:
        output = 'Iris-virginica'
    return jsonify({'pred_iris': output})

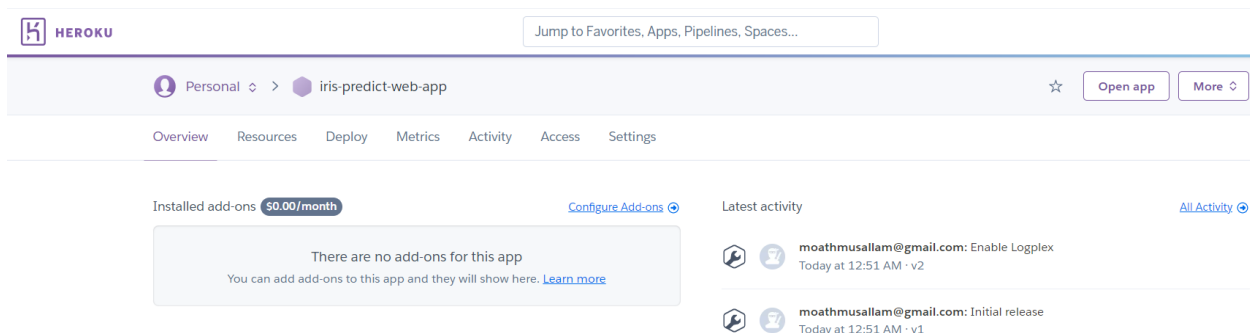
# if name = main the app will start

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

Try the API method with Postman, and its work.





Create new project in Heroku called iris-predict-web-app:




Deploy from GitHub:

Deployment method

 Heroku Git
Use Heroku CLI


 GitHub
Connect to GitHub

 Container Registry
Use Heroku CLI

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.


Search for a repository to connect to

 Moath-Musallam

Flask-Deployment-Heroku

Search

Missing a GitHub organization? [Ensure Heroku Dashboard has team access.](#)

 main

Deploy Branch

Receive code from GitHub

Build **main** 7455b866

```
3.1.0
----> Discovering process types
Procfile declares types -> web
----> Compressing...
Done: 192.3M
----> Launching...
Released v9
https://penguins-heroku-demo.herokuapp.com/ deployed to Heroku
```

☒ Autoscroll with output [View build log](#)

Release phase

Deploy to Heroku


Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

 main

Deploy Branch


Receive code from GitHub

Build **main** 7455b866













Release phase

Deploy to Heroku

Your app was successfully deployed.

 View

project file:

الاسم	^	تاريخ التعديل	النوع	الحجم
static 		١١:٣٦ م ٤٣/١٢/١٣	مجلد ملفات	
templates 		١١:٣٦ م ٤٣/١٢/١٣	مجلد ملفات	
app 		١٢:٤٠ ص ٤٣/١٢/١٤	ملف PY	٢ كيلوبايت
Heroku 		٠٦:١٦ ص ٤٣/٠٩/٢٢	Microsoft ... مستند	٤٣٠ كيلوبايت
Heroku 		٠٦:١٦ ص ٤٣/٠٩/٢٢	Microsoft Edge PD...	١٨٥ 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 		١٢:٥٠ ص ٤٣/١٢/١٤	ملف PY	٢ كيلوبايت
Procfile 		٠٥:٢٧ ص ٤٣/٠٩/٢٢	ملف	١ كيلوبايت
README 		١٢:٣٣ ص ٤٣/١٢/١٤	ملف MD	١ كيلوبايت
requirements 		١٢:٤٦ ص ٤٣/١٢/١٤	مستند نصي	٥ كيلوبايت
test 		٠٢:٤١ ص ٤٣/٠٩/٢١	ملف PY	١ كيلوبايت