

Name: Week 4: Data Intake on API  
Report date: 1/9/2022  
Internship Batch: LISUM16  
Version: 1.0  
Data intake by: Olivia Foster  
Data intake reviewer: Data Glacier  
Data storage location: <https://github.com/LiviaNFoster/Week4-5>

## Data:

The data was pulled from a toy data set from the famous Iris dataset.

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	3,975 bytes

## Model.py:

```
1 import pandas as pd
2 import numpy as np
3 import pickle
4 from sklearn.model_selection import train_test_split
5 from sklearn.linear_model import LinearRegression
6
7 iris = pd.read_csv("/Users/Olivia/Desktop/Week4-5project/iris.csv")
8
9 df = pd.DataFrame(iris)
10
11 df['variety'] = df['variety'].replace(['Setosa'], 0, regex=True) # replace 'Setosa' with 0 for regression
12 df['variety'] = df['variety'].replace('Versicolor', 1, regex=True) # replace 'Versicolor' with 1 for regression
13 df['variety'] = df['variety'].replace('Virginica', 2, regex=True) # replace 'Virginica' with 2 for regression
14
15 print(df.shape)
16 print(df)
17
18 x = df.iloc[:, :4]
19 y = df['variety']
20
21 test = np.array([4.9, 3, 1.3, 0.2])
22 test = test.reshape(1, -1)
23
24 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, train_size=0.7, random_state=100)
25 lm = LinearRegression()
26 lm.fit(x.values, y.values)
27
28 pickle.dump(lm, open('model.pickle', 'wb'))
29 model = pickle.load(open('model.pickle', 'rb'))
30
31 predictor = round(model.predict(test)[0])
32
33 print(predictor)
```

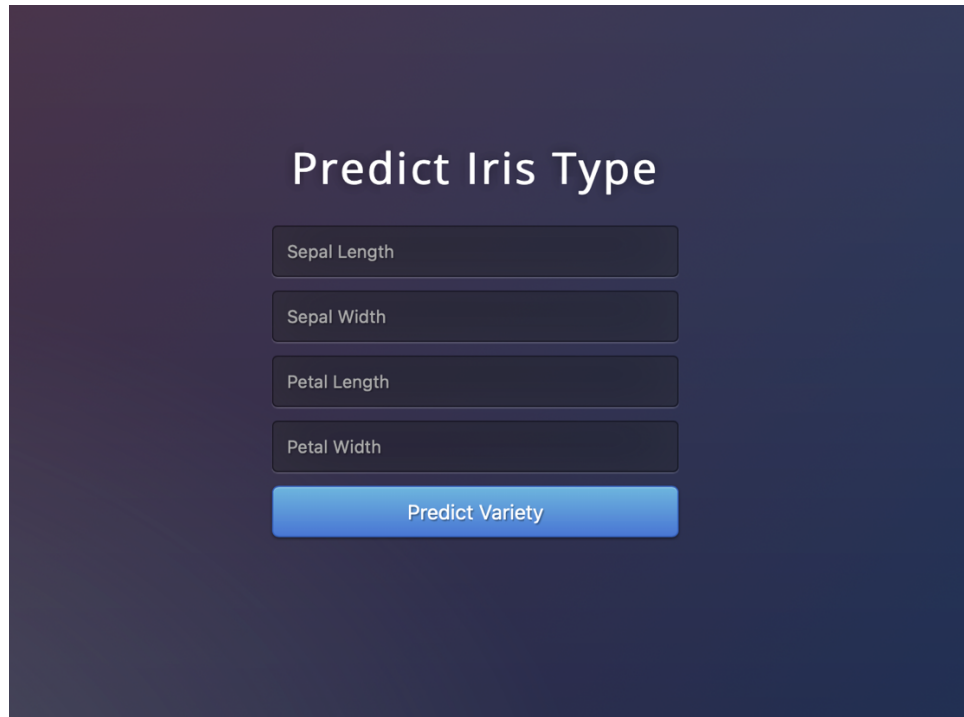
## app.py:

```
1 import numpy as np
2 import pickle
3 from flask import Flask, request, render_template
4
5 app = Flask(__name__, template_folder='template')
6 model = pickle.load(open('model.pickle', 'rb'))
7
8 @app.route('/', methods=['GET'])
9 def home():
10     return render_template('iris.html')
11
12 @app.route('/predict', methods=['POST'])
13 def predict():
14     """
15     For rendering results on HTML GUI
16     """
17     int_features = np.array([float(x) for x in request.form.values()])
18     final_features = [np.array(int_features)]
19     prediction = round(model.predict(final_features)[0])
20
21     print(prediction)
22
23     if prediction < 0.5:
24         return render_template("iris.html", prediction_text='Iris Variety should be Setosa'.format(prediction))
25     elif (prediction >= 0.5) and (prediction < 1.5):
26         return render_template("iris.html", prediction_text='Iris Variety should be Versicolor'.format(prediction))
27     else:
28         return render_template("iris.html", prediction_text='Iris Variety should be Virginica'.format(prediction))
29
30 if __name__ == "__main__":
31     app.run(port=5000, debug=True)
```

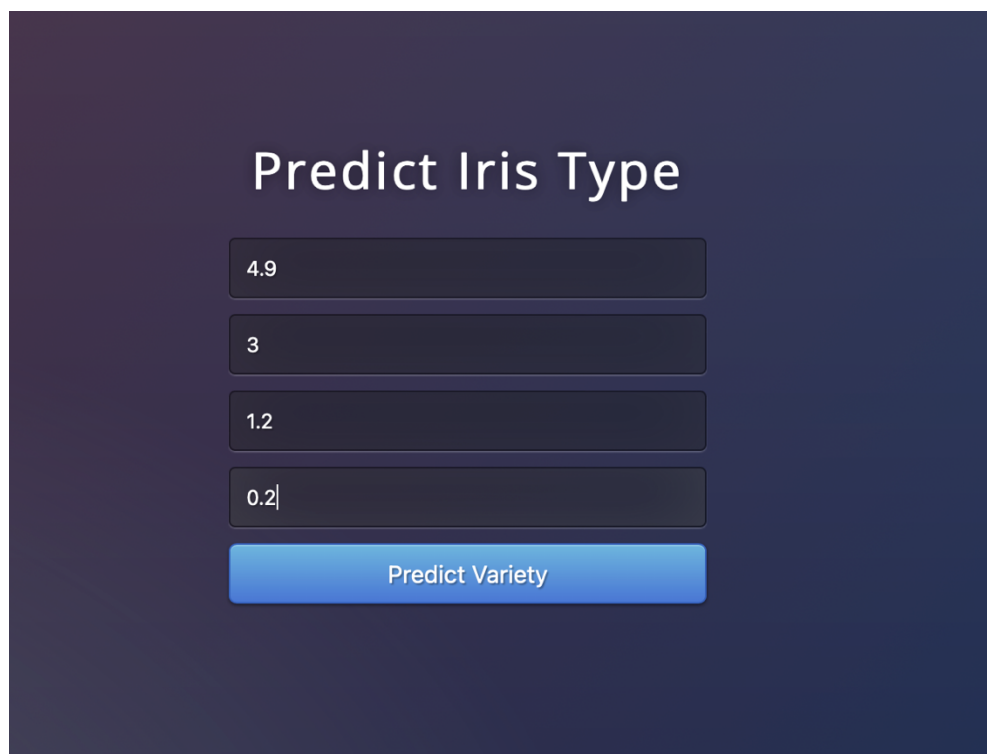
iris.html:

```
1 <!DOCTYPE html>
2 <html >
3 <!-- From https://codepen.io/frytyler/pen/EGdtg-->
4 <head>
5   <meta charset="UTF-8">
6   <title>ML API</title>
7   <!-- <link rel="stylesheet" type="text/css" href="../../static/css/styles.css"> -->
8   <link href = 'https://fonts.googleapis.com/css?family=Pacifico' rel = 'stylesheet' type = 'text/css'>
9   <link href = 'https://fonts.googleapis.com/css?family=Arimo' rel = 'stylesheet' type = 'text/css'>
10  <link href = 'https://fonts.googleapis.com/css?family=Hind:300' rel = 'stylesheet' type = 'text/css'>
11  <link rel = "stylesheet" type = "text/css" href = "{{ url_for('static', filename = 'css/scratch.css') }}">
12 </head>
13 <body>
14 <div class = "login">
15   <h1> Predict Iris Type </h1>
16   <form action="{{ url_for('predict')}}" method="post">
17     <input type="text" name="sepal.length" placeholder="Sepal Length" required="required"/>
18     <input type="text" name="sepal.width" placeholder="Sepal Width" required="required"/>
19     <input type="text" name="petal.length" placeholder="Petal Length" required="required"/>
20     <input type="text" name="petal.width" placeholder="Petal Width" required="required"/>
21     <button type="submit" class="btn btn-primary btn-block btn-large">Predict Variety</button>
22   </form>
23   <br>
24   <br>
25
26   {{ prediction_text }}
27
28 </div>
29
30 </body>
31 </html>
```

Deployment:



A screenshot of a web application interface with a dark blue gradient background. The title "Predict Iris Type" is centered at the top in white. Below the title are four stacked input fields, each with a light gray border and placeholder text: "Sepal Length", "Sepal Width", "Petal Length", and "Petal Width". At the bottom of the input stack is a blue button with white text that says "Predict Variety".



A screenshot of the same web application interface, but with numerical values entered into the input fields. The values are "4.9" for Sepal Length, "3" for Sepal Width, "1.2" for Petal Length, and "0.2" for Petal Width. The "Predict Variety" button remains at the bottom.

# Predict Iris Type

Sepal Length

Sepal Width


Petal Length

Petal Width

Predict Variety

Iris Variety should be Setosa

## API deployment on CircleCi:

**LiviaNFoster**  
Olivia Foster

Dashboard

Projects

Insights

Organization Settings

Plan

Security alert

Rotate all secrets stored on CircleCI. [Read more](#)

Getting Started

Notifications 1

Status OPERATIONAL

Docs

Dashboard

Project

All Pipelines > Week4-5

Week4-5

Add team members

Edit Config

Trigger Pipeline

Project Settings

Security alert. Rotate all secrets stored on CircleCI. [Read more](#)

Filters




Everyone's Pipelines

Week4-5

All Branches

All days

Auto-expand

Pipeline	Status	Workflow	Branch / Commit	Start	Duration	Actions
Week4-5 1	Success	say-hello-workflow	circleci-project-setup 5b07542	10s ago	9s	   ...