

Name: Sebastián J. Castro
Batch code: LISP01
Submission date: 21/03/2021
Submitted to: DataGlacier

Snapshots:

1. Toy data set modeling and pickle serialization:

```
app.py x model.py x index.html x
1 from sklearn.datasets import load_iris
2 from sklearn.model_selection import train_test_split
3 from sklearn.ensemble import RandomForestClassifier
4 import pickle
5
6 X, y = load_iris(return_X_y=True)
7 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y)
8
9 X, y = load_iris(return_X_y=True)
10 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y)
11 clf = RandomForestClassifier()
12 print(clf.fit(X_train, y_train).score(X_test, y_test))
13
14 filename = "model.pkl"
15 pickle.dump(clf, open(filename, "wb"))
```

2. Flask app.py:

```
app.py x model.py x index.html x
1 # importing libraries
2 import ...
3
4
5
6 app = Flask(__name__)
7 model = pickle.load(open('model.pkl', 'rb'))
8
9 @app.route('/')
10 def index():
11     return render_template('index.html')
12
13 def ValuePredictor(to_predict_list):
14     to_predict = np.array(to_predict_list).reshape(1, 4)
15     loaded_model = pickle.load(open('model.pkl', 'rb'))
16     result = loaded_model.predict(to_predict)
17     return result[0]
18
19 @app.route('/predict', methods=['POST'])
20 def predict():
21
22     if request.method == 'POST':
23         to_predict_list = request.form.to_dict()
24         to_predict_list = list(to_predict_list.values())
25
26         try:
27             to_predict_list = list(map(float, to_predict_list))
28             result = ValuePredictor(to_predict_list)
29             if int(result) == 0:
30
31
32 if __name__ == "__main__"
```

```
app.py x model.py x index.html x
19 @app.route('/predict', methods=['POST'])
20 def predict():
21
22     if request.method == 'POST':
23         to_predict_list = request.form.to_dict()
24         to_predict_list = list(to_predict_list.values())
25
26         try:
27             to_predict_list = list(map(float, to_predict_list))
28             result = ValuePredictor(to_predict_list)
29             if int(result) == 0:
30                 prediction: str = 'Iris setosa'
31             elif int(result) == 1:
32                 prediction = 'Iris virginica'
33             elif int(result) == 2:
34                 prediction = 'Iris versicolor'
35             else:
36                 prediction = f'{int(result)} Not defined'
37         except ValueError:
38             prediction = 'Value Error try again'
39
40     return render_template('index.html', prediction_text='The flower is {}'.format(prediction))
41
42
43 if __name__ == "__main__":
44     app.run(debug=True)
45
46 if __name__ == "__main__"
```

3. html file

```
app.py x model.py x index.html x
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Iris Species Classifier</title>
6     <link href="https://fonts.googleapis.com/css?family=Pacifico" rel="stylesheet" type="text/css">
7     <link href="https://fonts.googleapis.com/css?family=Arimo" rel="stylesheet" type="text/css">
8     <link href="https://fonts.googleapis.com/css?family=Hind:300" rel="stylesheet" type="text/css">
9     <link href="https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300" rel="stylesheet" type="text/css">
10    <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
11
12
13
14 </head>
15 <body>
16     <div class="login">
17         <h3>Iris Species Classifier</h3>
18
19
20         <form action="{{ url_for('predict') }}" method="POST">
21             <label for="seplen">Sepal Length</label>
22             <input type="number" step="0.01" id="seplen" name="seplen">
23             <br>
24             <label for="sepwid">Sepal Width</label>
25             <input type="number" step="0.01" id="sepwid" name="sepwid">
26             <br>
27             <label for="petlen">Petal Length</label>
28             <input type="number" step="0.01" id="petlen" name="petlen">
29             <br>
30             <label for="petwid">Petal Width</label>
31             <input type="number" step="0.01" id="petwid" name="petwid">
32             <br>
33             <input type="submit" value="Predict">
34         </form>
35     </div>
36 </body>
37 </html>
```

```
app.py x model.py x index.html x
20 <form action="{{ url_for('predict')}}" method="POST">
21     <label for="seplen">Sepal Length</label>
22     <input type="number" step="0.01" id="seplen" name="seplen">
23     <br>
24     <label for="sepwid">Sepal Width</label>
25     <input type="number" step="0.01" id="sepwid" name="sepwid">
26     <br>
27     <label for="petlen">Petal Length</label>
28     <input type="number" step="0.01" id="petlen" name="petlen">
29     <br>
30     <label for="petwid">Petal Width</label>
31     <input type="number" step="0.01" id="petwid" name="petwid">
32     <br>
33     <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
34
35
36 </form>
37
38 <br>
39 <br>
40 {{ prediction_text }}
41 </div>
42
43 </body>
44 </html>
```

html > body > div.login

4. css file

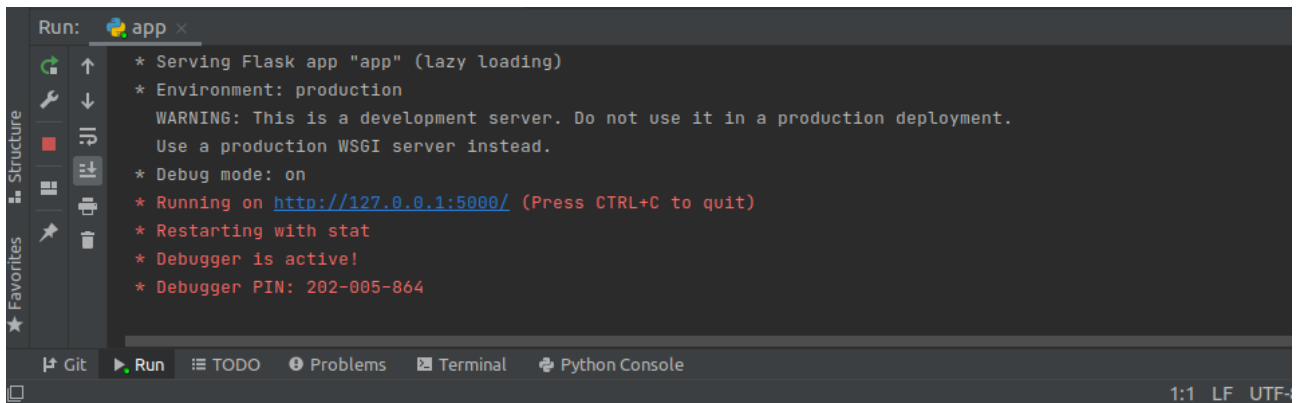
```
app.py x model.py x index.html x style.css x
1 @import url(https://fonts.googleapis.com/css?family=Open+Sans);
2 .btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; }
3 .btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
4 .btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; }
5 .btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: }
6 .btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }
7 .btn-primary.active { color: rgba(255, 255, 255, 0.75); }
8 .btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4);
9 .btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] {
10 .btn-block { width: 100%; display: block; }
11
12 * { -webkit-box-sizing: border-box; -moz-box-sizing: border-box; -ms-box-sizing: border-box; -o-box-sizing: border-box; }
13
14 html { width: 100%; height: 100%; overflow: hidden; }
15
16 body {
17     width: 100%;
18     height: 100%;
19     font-family: 'Helvetica';
20     background: #000;
21     color: #fff;
22     font-size: 24px;
23     text-align: center;
24     letter-spacing: 1.4px;
25
26 }
```

body

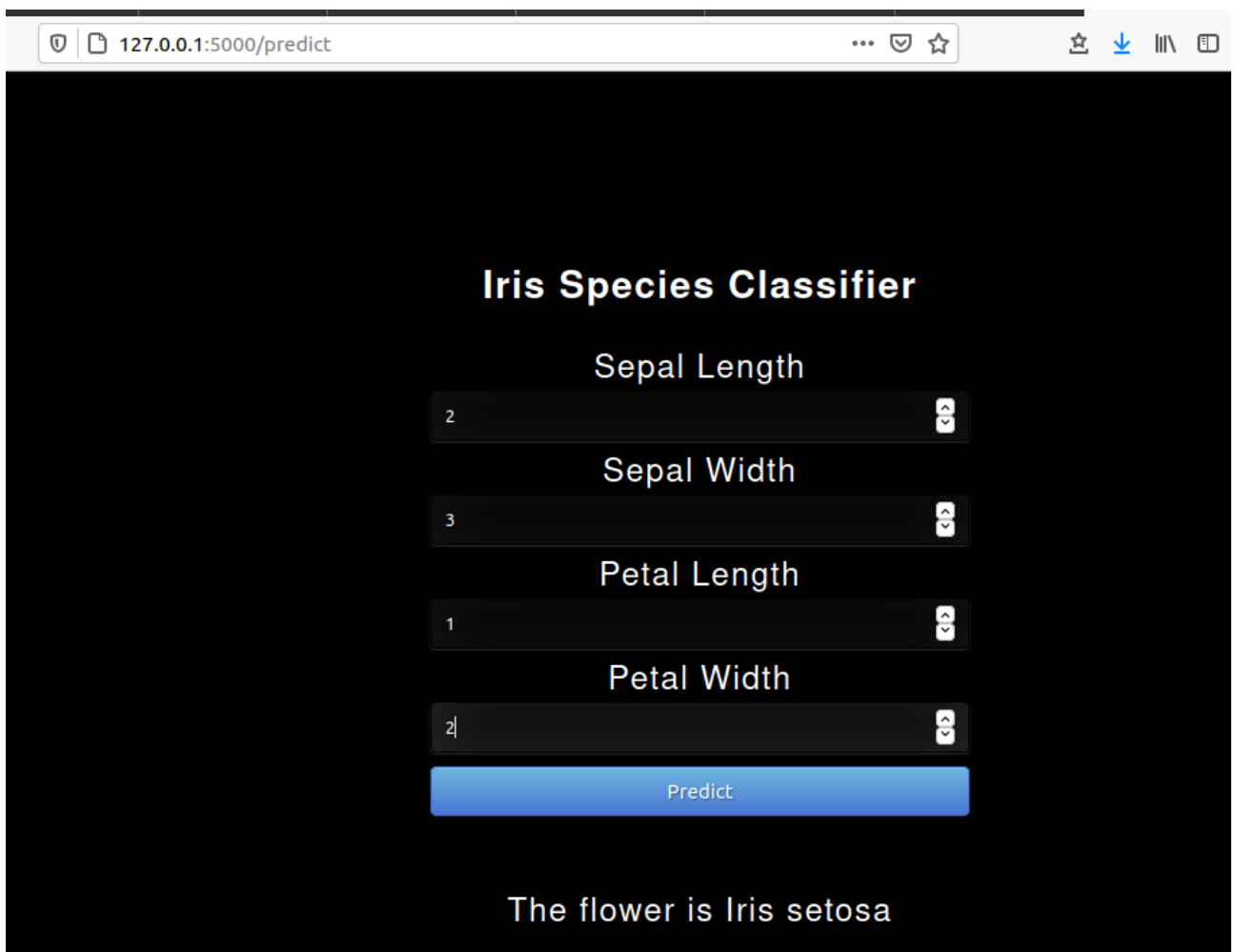
```
app.py x model.py x index.html x style.css x
27 .login {
28     position: absolute;
29     top: 40%;
30     left: 50%;
31     margin: -150px 0 0 -150px;
32     width: 400px;
33     height: 400px;
34 }
35
36 .login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing: 1px; text-align: center; }
37
38 input {
39     width: 100%;
40     margin-bottom: 10px;
41     background: rgba(0,0,0,0.3);
42     border: none;
43     outline: none;
44     padding: 10px;
45     font-size: 13px;
46     color: #fff;
47     text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
48     border: 1px solid rgba(0,0,0,0.3);
49     border-radius: 4px;
50     box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
51     -webkit-transition: box-shadow .5s ease;
52     -moz-transition: box-shadow .5s ease;
53 }
body
```

```
app.py x model.py x index.html x style.css x
37
38 input {
39     width: 100%;
40     margin-bottom: 10px;
41     background: rgba(0,0,0,0.3);
42     border: none;
43     outline: none;
44     padding: 10px;
45     font-size: 13px;
46     color: #fff;
47     text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
48     border: 1px solid rgba(0,0,0,0.3);
49     border-radius: 4px;
50     box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
51     -webkit-transition: box-shadow .5s ease;
52     -moz-transition: box-shadow .5s ease;
53     -o-transition: box-shadow .5s ease;
54     -ms-transition: box-shadow .5s ease;
55     transition: box-shadow .5s ease;
56 }
57 input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2); }
body
```

Flask app:



```
Run: 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
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 202-005-864
```



127.0.0.1:5000/predict

Iris Species Classifier

Sepal Length

Sepal Width

Petal Length

Petal Width

Predict

The flower is Iris setosa