



Data Glacier

Your Deep Learning Partner

Model Deployment with Flask

LISUM02

10th August 2021

**By,
Joseph Antony**

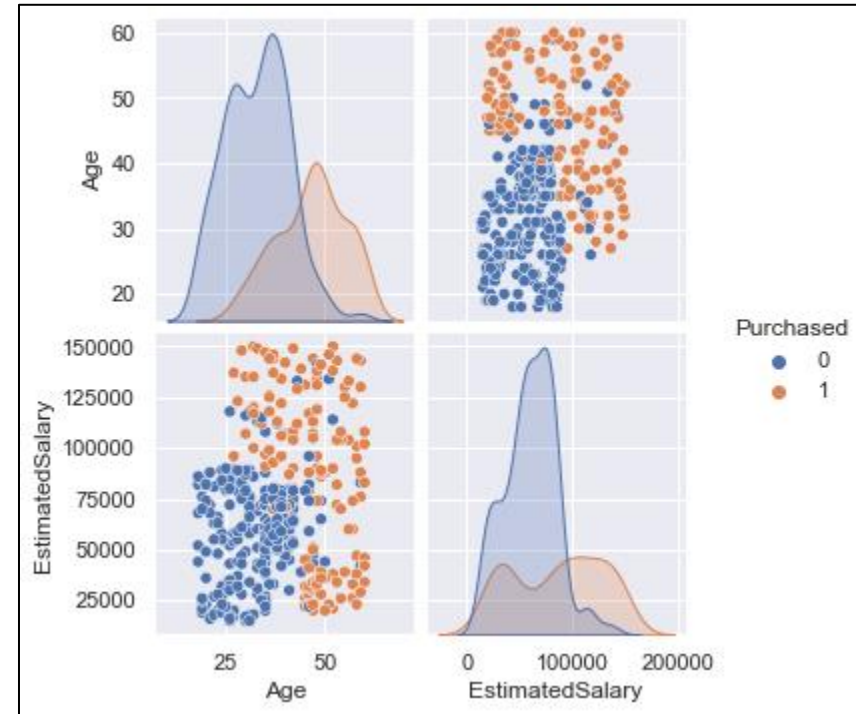
Tasks Involved

- The data used for this task contains market information such as prospective buyers' **Age, Estimated Salary**, as well as whether the customer purchased the product (**Purchased**).
- Trained a **Logistic Regression model** to predict whether a customer will purchase a product based on prospective buyers' Age and Salary.
- Finally, deployed Logistic Regression model to a web-app using **Flask API**.
- This way, we can predict potential customers using the web-app.

Age	EstimatedSalary	Purchased
19	19000	0
35	20000	0
26	43000	0
27	57000	0
19	76000	0
27	58000	0
27	84000	0
32	150000	1
25	33000	0
35	65000	0
26	80000	0
26	52000	0
20	86000	0
32	18000	0
18	82000	0

Model Validation

- Customers who are earn **higher than \$100,000 and older than 40 years** are more likely to purchase a product.
- After splitting the data into train and test set, Logistic Regression model was used for predicting on test set.
- The model achieved a high **accuracy of almost 92.5%.**



```
In [5]: lr_pipe = Pipeline(steps = [  
        ('scaler', StandardScaler()),  
        ('logistic', LogisticRegression(random_state = seed))  
    ])  
  
    lr_pipe.fit(X_train, y_train)  
    y_pred = lr_pipe.predict(X_test)  
  
    print(f'Model Accuracy: {metrics.accuracy_score(y_test, y_pred) * 100:.3f} %')
```

Model Accuracy: 92.500 %

Saving Model & Creating Web-App using Flask API

- The model was then trained on the whole dataset before saving the model to **pickle format**.
- Pickling is done to **convert python object to character object**.
- Next, created a python file to create the web app using **Flask API** module.

```
Saving model

In [6]: lr_pipe.fit(X,y)

        pickle.dump(lr_pipe, open('lr_model.pkl','wb'))
```

```
app.py* x  index.html x  style.css x

1  import numpy as np
2  import pickle
3  from flask import Flask, request, render_template
4
5  app = Flask(__name__)
6  model = pickle.load(open('lr_model.pkl', 'rb'))
7
8  @app.route('/')
9  def home():
10     return render_template('index.html')
11
12  @app.route('/predict',methods=['POST'])
13  def predict():
14     '''
15     For rendering results on HTML GUI
16     '''
17     int_features = [int(x) for x in request.form.values()]
18     final_features = [np.array(int_features)]
19     prediction = model.predict(final_features)
20
21     if prediction == 0:
22         output = "Customer will not purchase"
23     else:
24         output = "Customer will Purchase"
25
26     return render_template('index.html', prediction_text = output)
27
28
29 if __name__ == "__main__":
30     app.run(debug=True)
31
```

HTML

```
<!DOCTYPE html>
<html>
<!-- From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Potential Customer API</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>
This web-app will predict whether a customer will purchase a product based on the customer's
age and salary.

  <div class="login">
    <h1>Enter Customer Age & Salary</h1>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict') }}" method="post">
      <input type="text" name="Age" placeholder="Age" required="required" />
      <input type="text" name="EstimatedSalary" placeholder="Salary" required="required" />

      <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

    <br>
    <br>
    {{ prediction_text }}
  </div>
```

CSS

```
@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-size: 13px; line
.btn:hover, .btn.active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5p
.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-tra
.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
.btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-imag
.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: no
.btn-block { width: 100%; display: block; }

* { -webkit-box-sizing: border-box; -moz-box-sizing: border-box; -ms-box-sizing: border-box; -o-box-sizing: border-box; box-

html { width: 100%; height: 100%; overflow: hidden; }

body {
  width: 100%;
  height: 100%;
  font-family: 'Open Sans', sans-serif;
  background: #092756;
  color: #fff;
  font-size: 18px;
  text-align: center;
  letter-spacing: 1.2px;
  background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linea
background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webki
background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-g
background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear
background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E106D', endColorstr='#092756', GradientType=1 );
}

.login {
  position: absolute;
  top: 40%;
```

Model Deployment

```
Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
C:\Users\joean\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn
e.py:310: UserWarning: Trying to unpickle estimator StandardScaler from version
0.2 when using version 0.24.2. This might lead to breaking code or invalid resu
Use at your own risk.
  warnings.warn(
C:\Users\joean\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn
e.py:310: UserWarning: Trying to unpickle estimator LogisticRegression from ver
0.23.2 when using version 0.24.2. This might lead to breaking code or invalid
ts. Use at your own risk.
  warnings.warn(
C:\Users\joean\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn
e.py:310: UserWarning: Trying to unpickle estimator Pipeline from version 0.23.
en using version 0.24.2. This might lead to breaking code or invalid results. U
: your own risk.
  warnings.warn(
* Debugger is active!
* Debugger PIN: 132-629-808
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

b-app will predict whether a customer will purchase a product based on the customer's age and salary.

Enter Customer Age & Salary

Predict

Enter Customer Age & Salary

Predict

Customer will not purchase

Enter Customer Age & Salary

Predict

Customer will Purchase

- Finally, created the web-app and deployed the model into the web-app.
- Based on the model, we can now use the web application to predict potential customers based on Age and Salary

The End