Name: Lauro Cesar Ribeiro

Batch Code: LISP01

Submission date: 16/03/2021

Submitted to: Data Glacier

Website link: https://lauro-salarypredictor.herokuapp.com/

---------- **Local Deployment** ----------

I will break down this task from a local deployment to the cloud; I will also consider that you are using Visual Studio Code, in case you are not, no problem at all, the first thing is to create a virtual environment in the folder containing your code. Install virtualenv:

pip install virtualenv

In the folder of your project, if not:

cd my-project

Activate the Virtual Environment

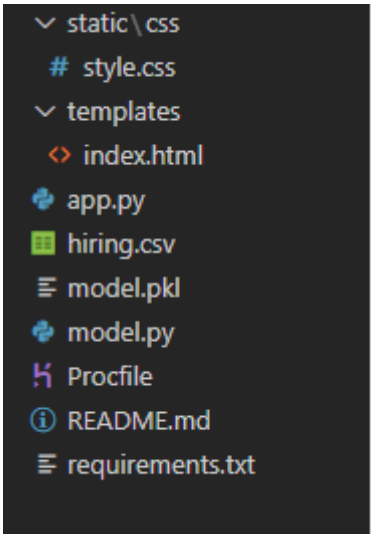
virtualenv --python C:\Path\To\Python\python.exe venv

Then, activate the Virtual Environment:

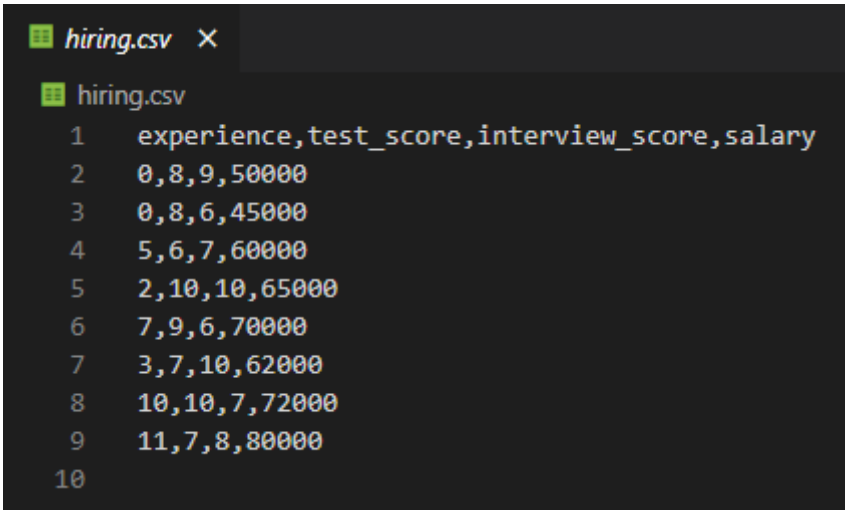
.\venv\Scripts\activate

If you need to install packages from a determined project, type: **pip install -r requirements.txt**

All set to start the development, the final folder structure should look like this:



This application is not meant to be a robust web app; for this reason, I will be working with a small piece of data.



It is time to start working on our machine learning models; here is the following code for **model.py** file:

```python
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle

dataset = pd.read_csv('hiring.csv')

x = dataset.iloc[:, :3]
y = dataset.iloc[:, -1]

#Splitting Training and Test Set
#Since we have a very small dataset, we will train our model with all availabe data.

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()

#Fitting model with trainig data
regressor.fit(x, y)

# Saving model to disk
pickle.dump(regressor, open('model.pkl','wb'))

'''
# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[2, 9, 6]]))
'''
```

Ok, after the model is trained you should save a pickle file of it, then we will work on the app.py file:

```python
app.py > ...
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

app = Flask(__name__) #Initialize the flask App
model = pickle.load(open('model.pkl', 'rb'))

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

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

    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='Employee Salary should be $ {}'.format(output))

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

All right, go to the template folder and open a new file named index.html; here is the following code:

```html
templates > index.html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML 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>
 <div class="login">
  <h1>Predict Salary Analysis</h1>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
      <input type="text" name="experience" placeholder="Experience" required="required" />
        <input type="text" name="test_score" placeholder="Test Score" required="required" />
    <input type="text" name="interview_score" placeholder="Interview Score" required="required" />

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

   <br>
   <br>
   {{ prediction_text }}

 </div>
</body>
</html>
```

To style the website I used Bootstrap to speed up my development.

```
static > css > # style.css > .btn
@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-height: 18px; color: #333333; te
.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: 5px; border-radius: 5px; }
.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.
.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-image: -ms-linear-gradient(top, |
.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }
.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-sizing:border-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-linear-gradient(top,  rgba(57,173,
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top,  rgba(
    background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top,  rgba(57,173,21
    background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top,  rgba(57,173,2
    background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to bottom,  rgba(57
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );

}
```

Create a Procfile to host into Heroku; It is one of their requirements to do it so

```
Procfile
web: gunicorn app:app
```

Type the following commands to generate a list of packages to install when you finish the deployment in the terminal: **pip freeze > requirements.txt**

```
requirements.txt
Flask
itsdangerous
Jinja2
jsonpickle
jsonschema
matplotlib
numpy
pandas
pickleshare
scikit-learn
sklearn
gunicorn
```

At this point, we can run the code to see what displays in the terminal

```
C:\Users\Lauro Ribeiro\Documents\Data Glacier- Virtual Internshi
o/Documents/Data Glacier- Virtual Internship/Week-4-DeploymentOn
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a pro
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 514-416-188
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Go to http://127.0.0.1:5000/ in your browser, you see the website, once you do some request, the terminal should change to show the **API request**, you can notice the methods **POST** and **GET** and the **http** response **200** for successful anything different, it points an error.

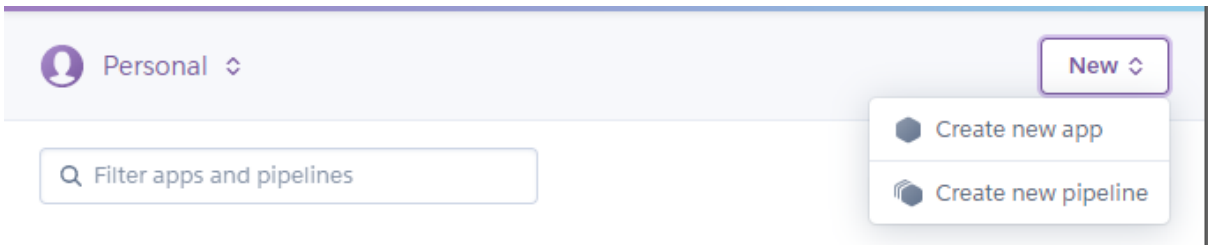
```
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 514-416-188
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [29/Mar/2021 19:49:15] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [29/Mar/2021 19:49:15] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [29/Mar/2021 19:49:15] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [29/Mar/2021 19:49:28] "POST /predict HTTP/1.1" 200 -
```

Finally, after you run all codes, you need to deactivate the virtual environment; here is the command:

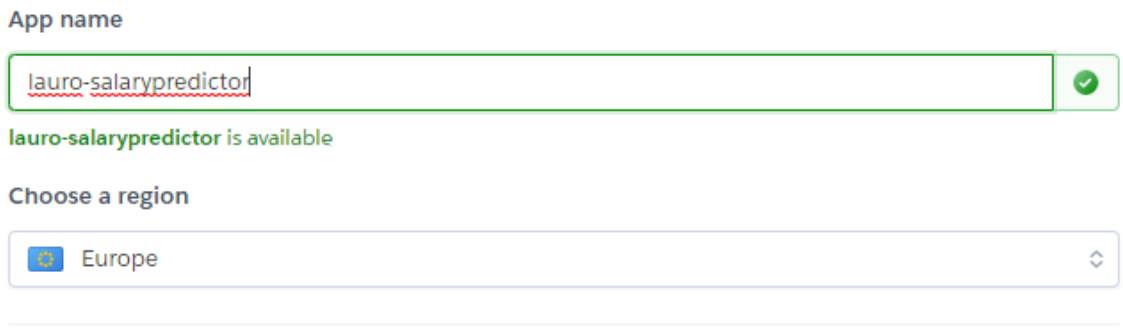
deactivate

---------- **Heroku Deployment** ----------

First thing you should do is register a Heroku account and you can register at https://signup.heroku.com/ .Then, the next step will be a new app creation.



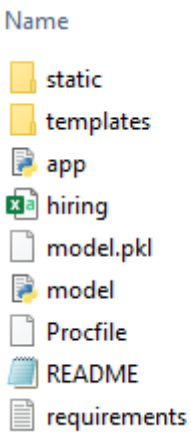
Click the “Create new app” button



My app is named “lauro-salarypredictior” hosted in Europe.

- The next important step is to download the Heroku CLI at https://devcenter.heroku.com/articles/heroku-cli

Attention: What made the difference to me was to create two important files into the folder. Here is my folder structure:



Pay attention to requirements.txt and Procfile

requirements - Notepad

File Edit Format View Help

```
Flask
itsdangerous
Jinja2
jsonpickle
jsonschema
matplotlib
numpy
pandas
pickleshare
scikit-learn
sklearn
gunicorn
```

requirements.txt

Procfile - Notepad

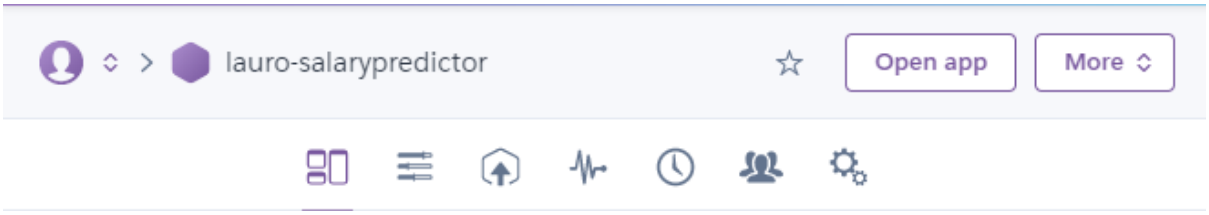
File Edit Format View Help

```
web: gunicorn app:app
```

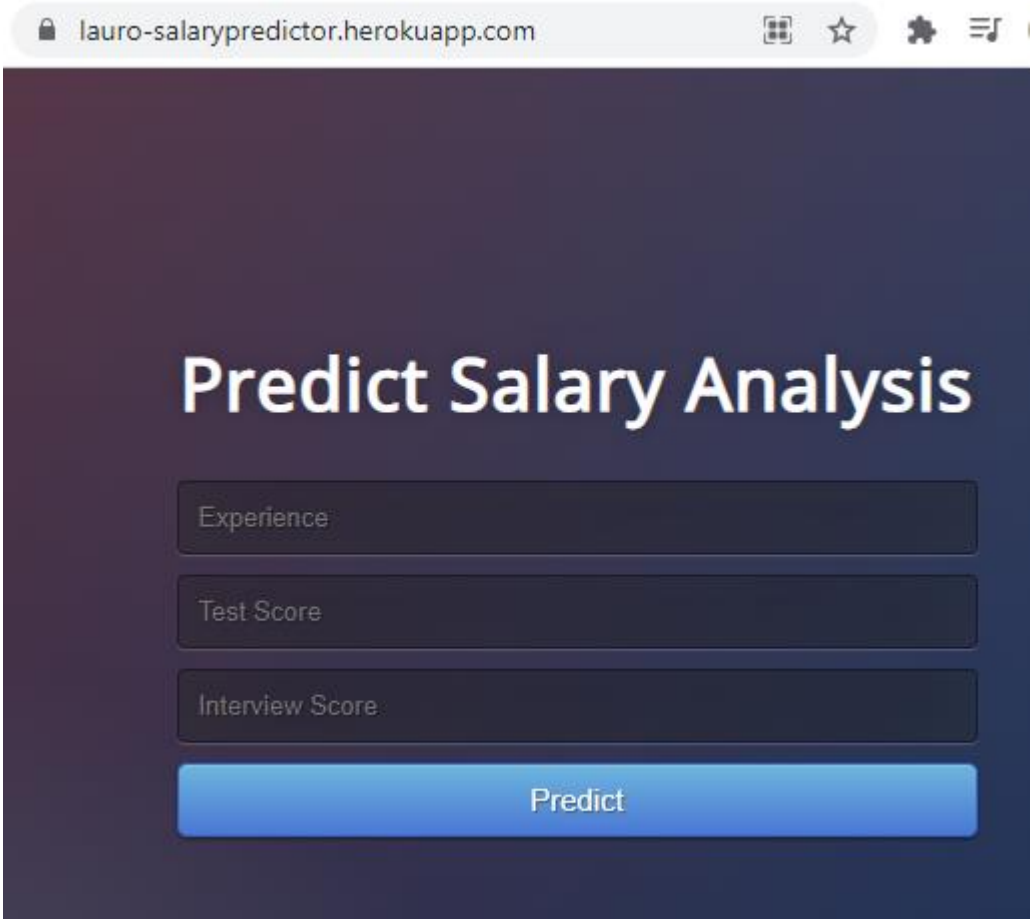
Procfile

Open the command line in the project folder, and type these commands:

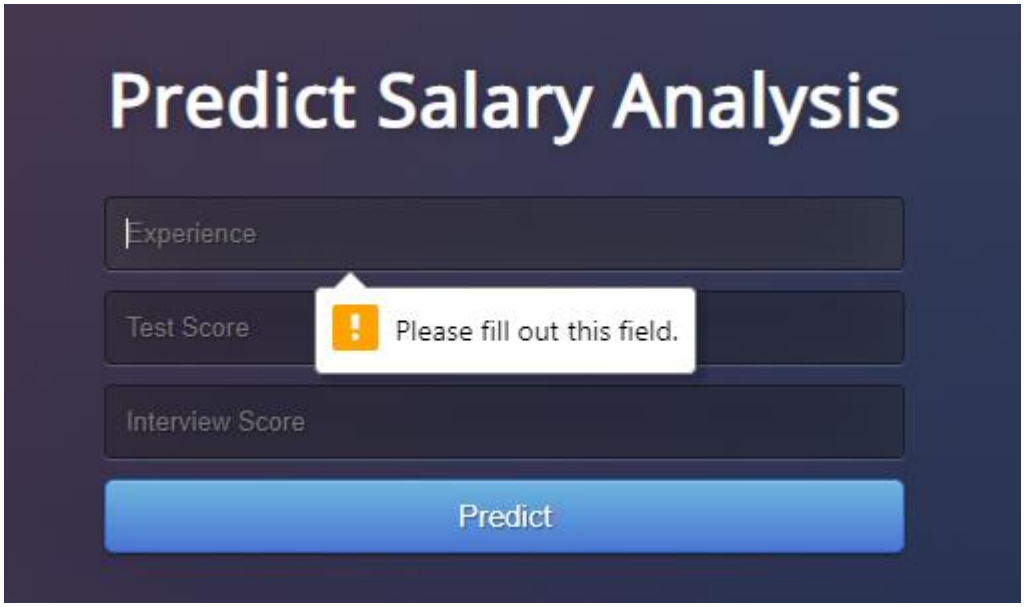
- heroku login - A pop-up window will appear for you to click on the button to give access.
- git init – Initialize the git into the folder.
- heroku git:remote -a lauro-salarypredictor – Point to the Heroku`s app directory.
- git add . – Add all the files from the folder to get them ready for uploading.
- git commit -am "Initial Commit" – Indicate the first change in the folder.
- git push heroku master – Upload everything to the Heroku Server.



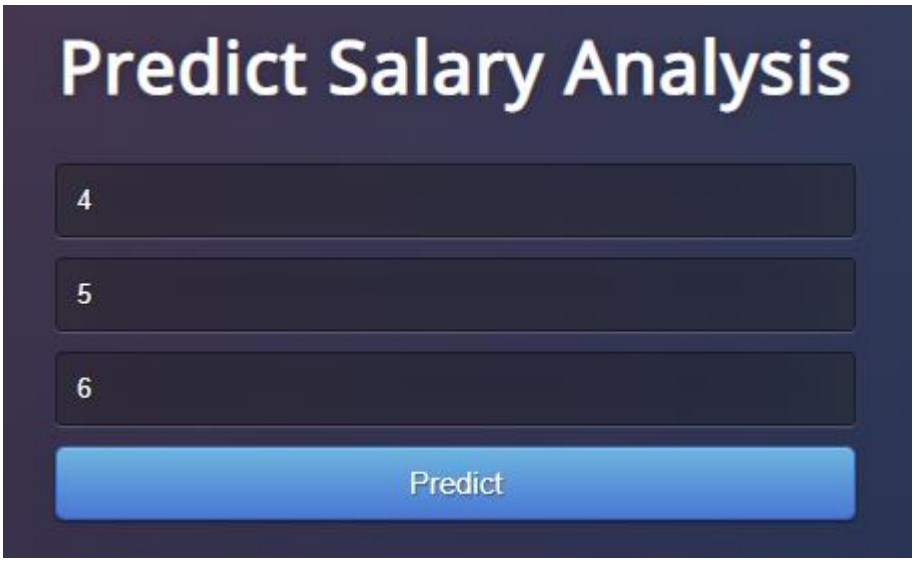
Click the “Open app” button and a pop-up window will come up.



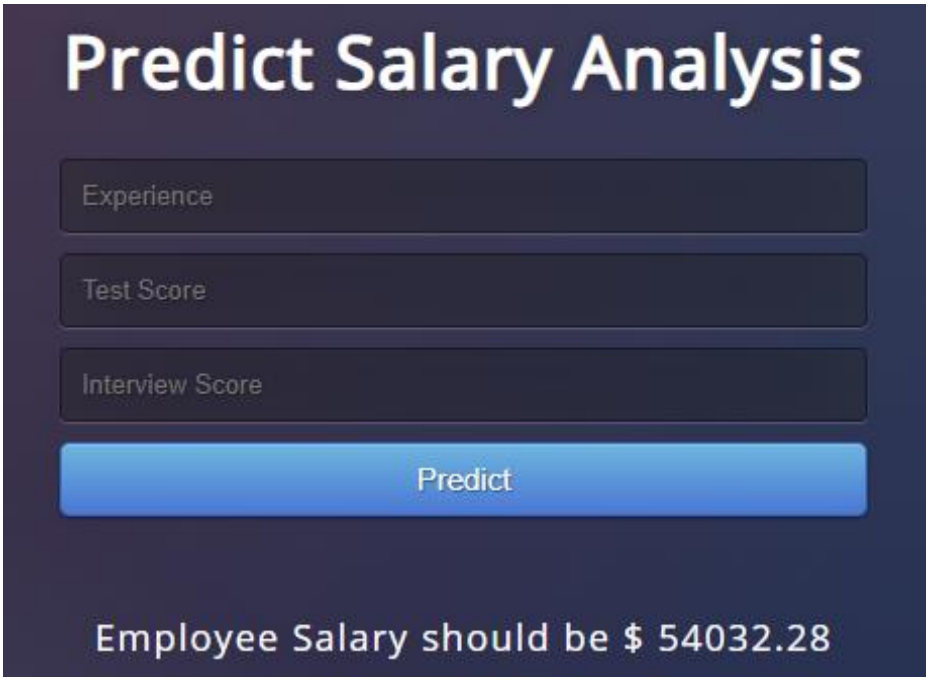
This is the Index page that you must see, your website hosted on the Cloud. Insert integer number values for Experience, Test score and Interview Score to predict employee`s salary.



Do not try to press predict without filling the experience blank space, that will fire a warning.



Once you fill up the gaps, the result will display at the bottom.



Here is the predicted salary for a person with four years of experience, who scored five and six for test and interview score.

I hope you like it 😊