

# Data Science Internship

## Week 4 – Deployment on Flask

**Name:** Sai Vishal Arram

**Batch:** LISUM24

**Submission Date:** 26/08/2023

**Submitted to:** Data Glacier Canvas Dashboard

### Contents

1. Introduction.....	1
2. Metadata.....	1
3. Model Building .....	2
4. Deployment.....	2
4.1 app.py .....	2
4.2 index.html .....	4
4.3 style.css .....	5
5. Result .....	6

### 1. Introduction

This project will demonstrate how a machine learning model can be deployed on a web app using the Flask framework. In this project, we will build a linear regression machine-learning model and deploy it on a web app where we can use its predictive capabilities through HTML requests.

### 2. Metadata

The data set was obtained through Kaggle, it contains 2 features; one highlights the advertising budget of TV, and the other highlights the sales generated through that advertising. Below is the head of the dataset:

TV	Sales
230.1	22.1
44.5	10.4
17.2	9.3

151.5	18.5
-------	------

Feature information:

TV	in thousands
Sales	in millions

### 3. Model Building

The linear regression model was built using the below code:

```

model.py X
C:\Users\> visha > Desktop > Data Glacier Internship > Data-Glacier-Internship > week4 > model.py > ...
1  #importing required modules and libraries
2  import pandas as pd
3  import numpy as np
4  import pickle
5  from sklearn.linear_model import LinearRegression
6  from sklearn.model_selection import train_test_split
7
8  #loading the dataset
9  tvmarketing = pd.read_csv("tvmarketing.csv")
10
11 #preapring for modelling
12 X = tvmarketing["TV"]
13 y = tvmarketing["Sales"]
14
15 X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=101)
16
17 X_train = X_train.values.reshape(-1, 1)
18 X_test = X_test.values.reshape(-1, 1)
19
20
21 lm = LinearRegression()
22 lm.fit(X_train,y_train)
23
24 pickle.dump(lm,open("model.pickle", "wb"))

```

### 4. Deployment

Once we run the above model.py file, the model will be built and saved in model.pickle file which we can use it to create a port and deploy our web app.

#### 4.1 app.py

We will first create app.py where our built model can be incorporated. The two main things required for this are HTML which we can take form requests from the user and port. We used the default port of '8080'. App.py is located in the week4 folder

app.py X

C:\Users\> visha > Desktop > Data Glacier Internship > Data-Glacier-Internship > week4 > app.py > ...

```
1  import numpy as np
2  from flask import Flask, request, render_template
3  import pickle
4
5  app = Flask(__name__)
6  model = pickle.load(open('model.pickle','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     int_features = [int(x) for x in request.form.values()]
15     final_features = [np.array(int_features)]
16     prediction = model.predict(final_features)
17
18     output = prediction
19
20     return render_template('index.html', prediction_text='Expected sale of TV is {}'.format(output))
21
22  if __name__ == "__main__":
23     app.run(port=8080, debug=True)
```

## 4.2 index.html

The index.html has the HTML which is used to take the user input on our web app. It is located in week4 > templates folder

```
index.html M X # style.css 3
C:\Users\> visha\Desktop> Data Glacier Internship> Data-Glacier-Internship> week4> templates> index.html > html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>ML API</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 </head>
13
14 <body>
15     <div class="login">
16         <h1>Predict TV Sale</h1>
17
18
19         <form action="{{ url_for('predict') }}" method="post">
20             <input type="text" name="ad_budget" placeholder="Enter ad budget" required="required" />
21
22             <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
23         </form>
24
25         <br>
26         <br>
27         {{ prediction_text }}
28
29     </div>
30     
31
32 </body>
33 </html>
```

### 4.3 style.css

This contains the css code used to design our web app, it will be used in our index.html for required web app design. It is located in week4 > static > css folder

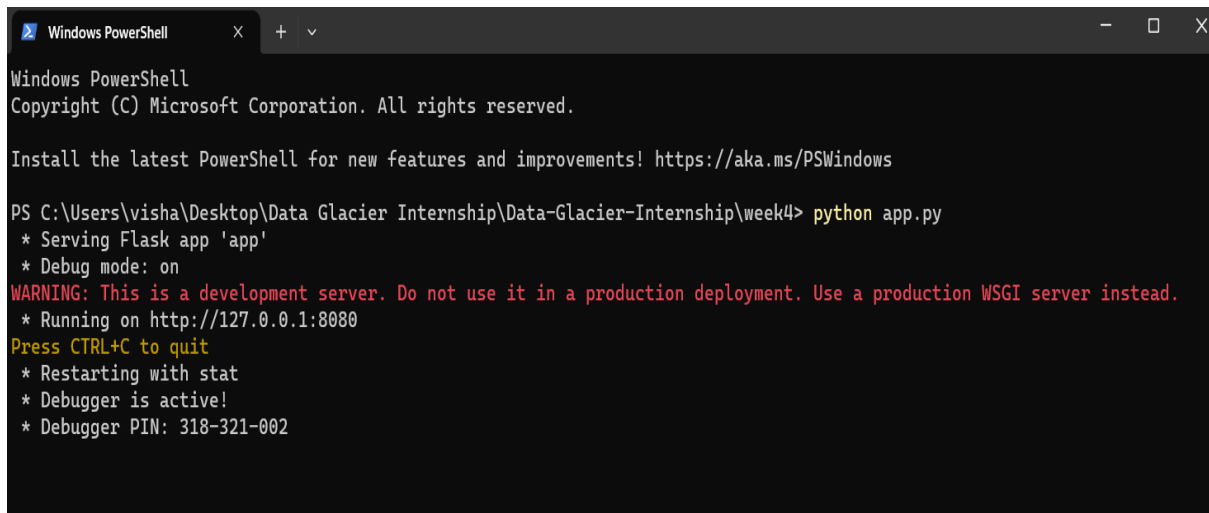
```
0 index.html style.css X
C:\Users\wisha\Desktop>Data-Clacier-Internship>week4>static>css>style.css>_

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; font-size: 13px; line-height: 18px; color: #333333; text-align: center; text-shadow: 0 1px 1px #888; vertical-align: middle; background-color: #f5f5f5; background-image: -moz-linear
3 .btn:hover, .btn.active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
4 .btn-large { padding: 8px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; }
5 .btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear; }
6 .btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 #888; color: #fff; }
7 .btn-primary.active { color: #888; }
8 .btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6e6e6e, #4a77d4); background-image: -ms-linear-gradient(top, #6e6e6e, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6e6e6e), to(#4a77d4)); background-image: -webkit-linear-gradi
9 .btn-primary:hover, .btn-primary.active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }
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; 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: 'Open Sans', sans-serif;
20     color: #fff;
21     font-size: 18px;
22     text-align: center;
23     letter-spacing: 1.2px;
24     background: #333333 !important;
25     filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#333333', endColorstr='#000000', gradientType=1 );
26 }
27
28 .login {
29     position: absolute;
30     top: 40%;
31     left: 50%;
32     margin: -150px 0 0 -150px;
33     width: 400px;
34     height: 400px;
35 }
36
37 .login h1 { color: #fff; text-shadow: 0 0 10px #888; letter-spacing: 1px; text-align: center; }
38
39 input {
40     width: 100%;
41     margin-bottom: 10px;
42     background: #888;
43     border: none;
44     outline: none;
45     padding: 10px;
46     font-size: 13px;
47     color: #fff;
48     text-shadow: 1px 1px #888;
49     border: 1px solid #888;
50     border-radius: 4px;
51     box-shadow: inset 0 -5px 4px #888, 0 1px 1px #888;
52     -webkit-transition: box-shadow .5s ease;
53     -moz-transition: box-shadow .5s ease;
54     -o-transition: box-shadow .5s ease;
55     -ms-transition: box-shadow .5s ease;
56     transition: box-shadow .5s ease;
57 }
58 input:focus { box-shadow: inset 0 -5px 4px #888, 0 1px 1px #888; }
```

We also have the image folder in static which is the logo used on our web app.

## 5. Result

We use the terminal on the folder path and run our app.py file using 'python app.py' command.

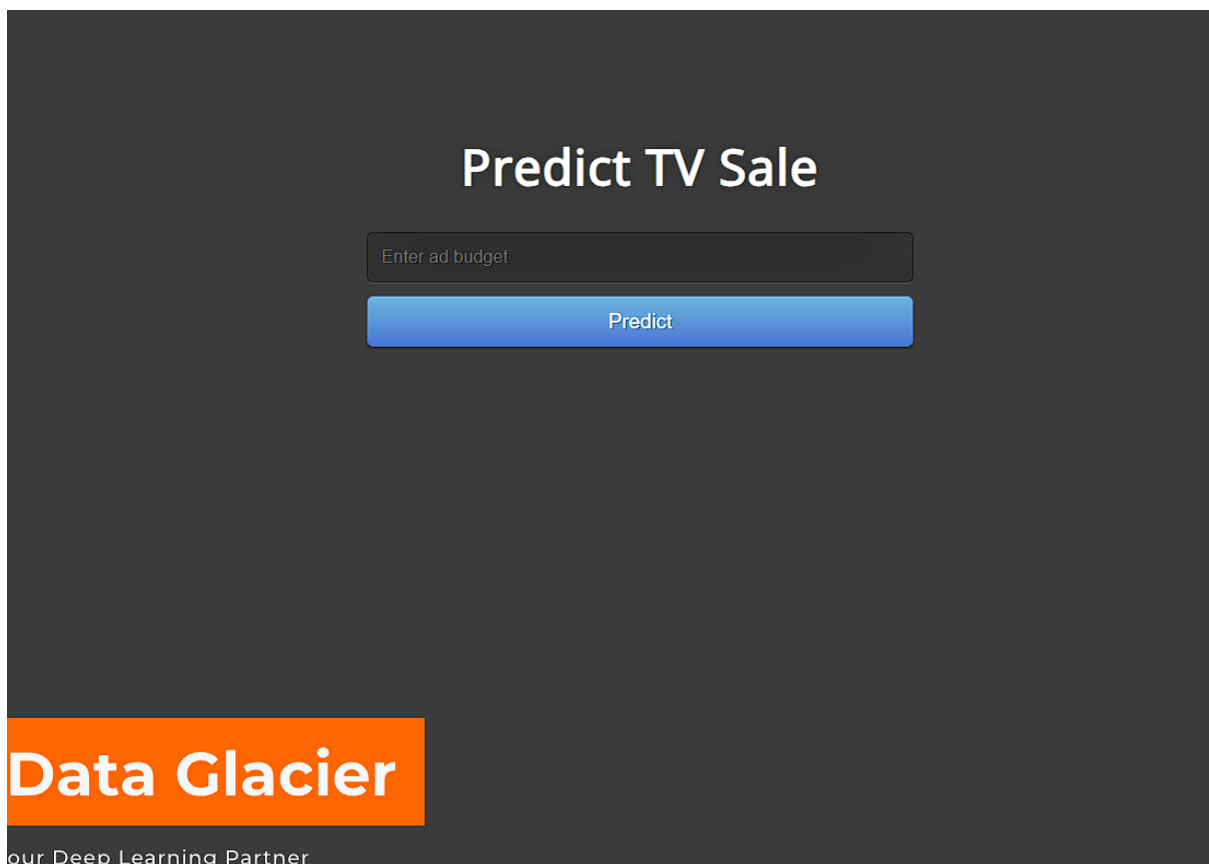


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\visha\Desktop\Data Glacier Internship\Data-Glacier-Internship\week4> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:8080
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 318-321-002
```

After running, a port will be opened as shown in the above image and we can copy and paste the link <http://127.0.0.1:8080> on any web browser to access our web app.



Once an ad budget is entered, the user will click on predict and our web app will display the sale amount.

## Predict TV Sale

Predict

Expected sale of TV is [18.46705464]

**Data Glacier**  
Your Deep Learning Partner