Nutan Follow

Dec 11, 2020 · 6 min read · Listen

Deploy Machine Learning Model in Google Cloud Platform Using Flask — Part 2

Load data, train model and save model — part 1

If you have not trained model, please follow below link:

Deploy Machine Learning Model in Google Cloud Platform Using Flask — Part 1

Create Flask application

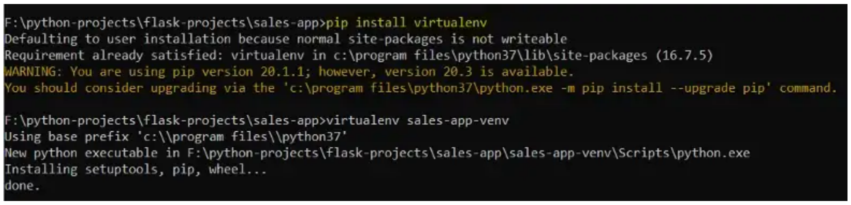
You can choose any directory to create new Flask application. Here i am using below directory to create new Flask project.

Open command prompt and go to where you want to create a flask application. In this example sales-app is my project name.

```
F:
cd F:\python-projects\flask-projects
mkdir sales-app
cd sales-app
```

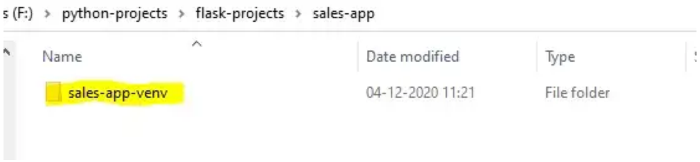
Install virtual environment inside project folder

```
#install
pip install virtualenv

#create virtual environment
virtualenv sales-app-venv #sales-app-venv is my virtual environment
name
```



```
F:\python-projects\flask-projects\sales-app>pip install virtualenv
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: virtualenv in c:\program files\python37\lib\site-packages (16.7.5)
WARNING: You are using pip version 20.1.1; however, version 20.3 is available.
You should consider upgrading via the 'c:\program files\python37\python.exe -m pip install --upgrade pip' command.

F:\python-projects\flask-projects\sales-app>virtualenv sales-app-venv
Using base prefix 'c:\\program files\\python37'
New python executable in F:\python-projects\flask-projects\sales-app\sales-app-venv\Scripts\python.exe
Installing setuptools, pip, wheel...
done.
```

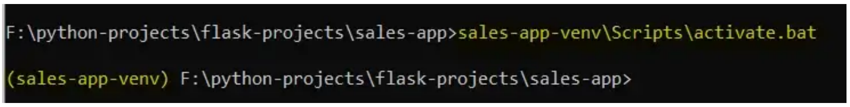
In file explorer, we can see virtual environment created successfully.



Name	Date modified	Type
sales-app-venv	04-12-2020 11:21	File folder

Activate virtual environment

```
sales-app-venv\Scripts\activate.bat
```



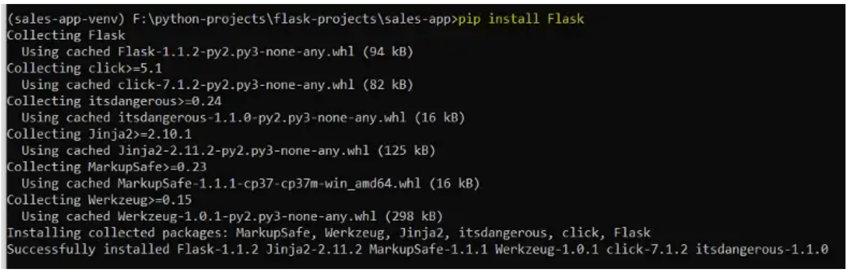
```
F:\python-projects\flask-projects\sales-app>sales-app-venv\Scripts\activate.bat

(sales-app-venv) F:\python-projects\flask-projects\sales-app>
```

Left side we can see(sales-app-venv) before file directory, that means virtual environment is active.

Now virtual environment is activated, we will install required library for the project.

Install Flask

```
pip install Flask
```



```
(sales-app-venv) F:\python-projects\flask-projects\sales-app>pip install Flask
Collecting Flask
  Using cached Flask-1.1.2-py2.py3-none-any.whl (94 kB)
Collecting click>=5.1
  Using cached click-7.1.2-py2.py3-none-any.whl (82 kB)
Collecting itsdangerous>=0.24
  Using cached itsdangerous-1.1.0-py2.py3-none-any.whl (16 kB)
Collecting Jinja2>=2.10.1
  Using cached Jinja2-2.11.2-py2.py3-none-any.whl (125 kB)
Collecting MarkupSafe>=0.23
  Using cached MarkupSafe-1.1.1-cp37-cp37m-win_amd64.whl (16 kB)
Collecting Werkzeug>=0.15
  Using cached Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
Installing collected packages: MarkupSafe, Werkzeug, Jinja2, itsdangerous, click, Flask
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 itsdangerous-1.1.0
```

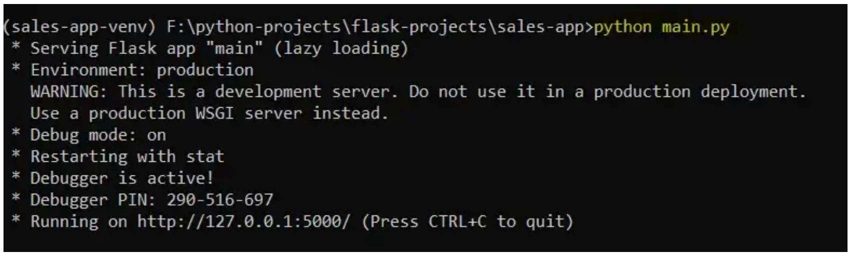
Create main.py file in root project folder

Write the following code in main.py file

```
#import Flask
from flask import Flask

#create an instance of Flask
app = Flask(__name__)

@app.route('/')
def home():
    return "Hello World"

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

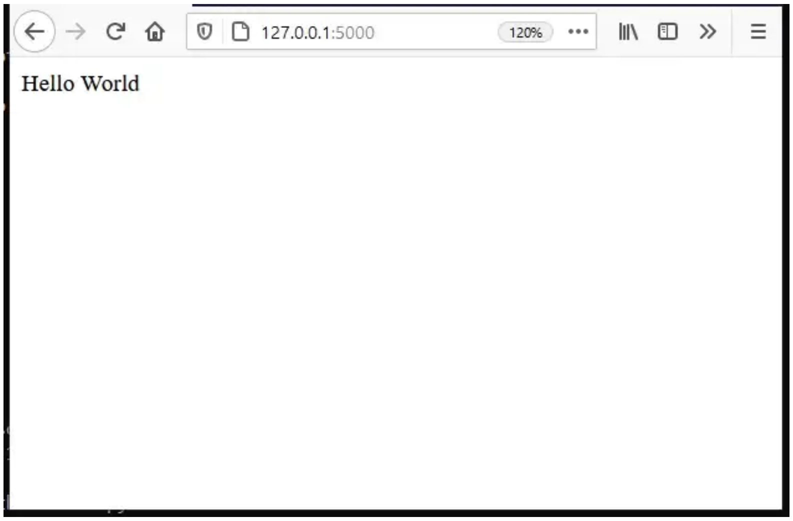
Run the flask application

```
python main.py
```



```
(sales-app-venv) F:\python-projects\flask-projects\sales-app>python main.py
 * Serving Flask app "main" (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
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 290-516-697
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

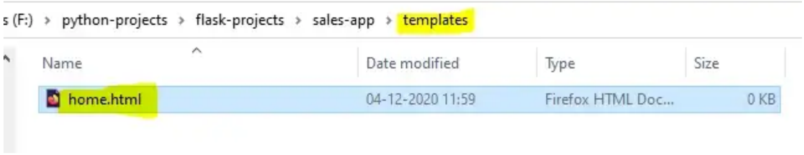
Server is running. Open url http://127.0.0.1:5000/ in browser.



Hello World

Add template to homepage

Create templates folder in project root folder and inside that home.html



s (F:) › python-projects › flask-projects › sales-app › templates

Name	Date modified	Type	Size
home.html	04-12-2020 11:59	Firefox HTML Doc...	0 KB

Write following code in home.html

I am using bootstrap for styling. Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography.

If you don't want bootstrap, you can write your own style sheet.

```
<!doctype html>
<html>
  <head>
    <title> Predict Sales </title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.m
in.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js
"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min
.js"></script>

  </head>
  <body>
      <div class="container pt-3">
        <div id="row">
            <h1>Hello World</h1>
        </div>
      </div>
  </body>
</html>
```

Modify main.py file

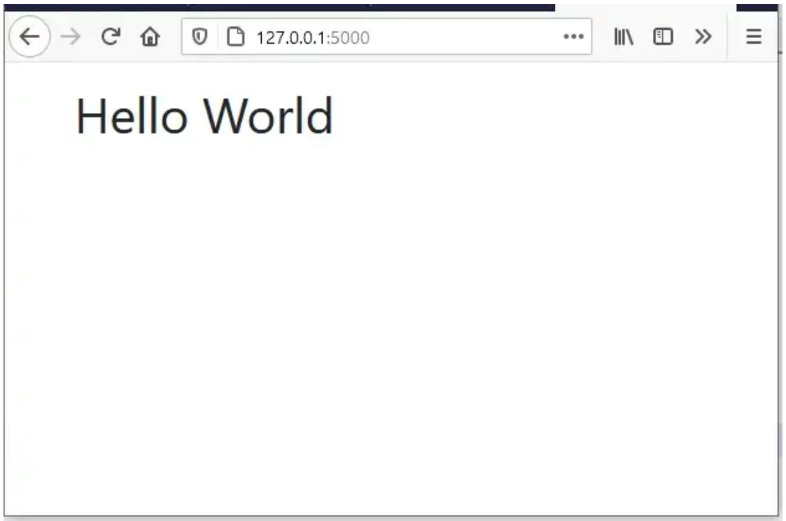
Import render_template from flask and return home.html template on home route url.

```
#import Flask
from flask import Flask, render_template

#create an instance of Flask
app = Flask(__name__)

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

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

Run the application

```
python main.py
```

Browser url: http://127.0.0.1:5000/



Hello World

We can see now homepage is coming from template.

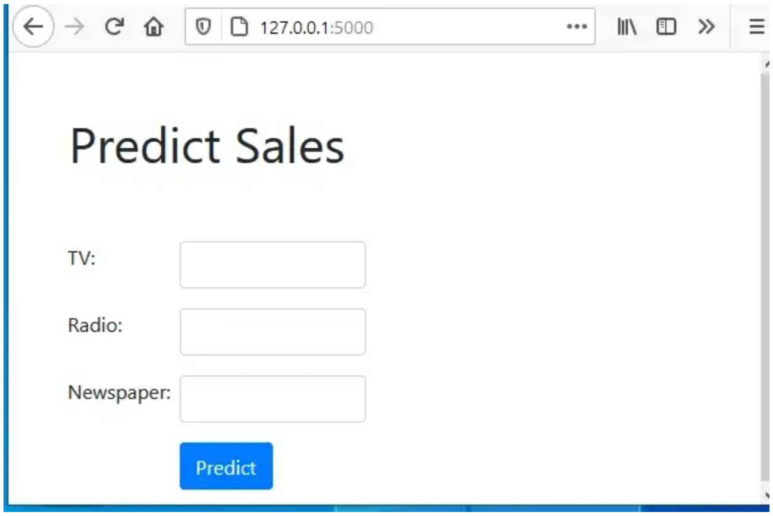
Let us add form in homepage which will take input of media advertising budget

Write following code in template/home.html

```
<!doctype html>
<html>
  <head>
    <title> Predict Sales </title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.m
in.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js
"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min
.js"></script>

  </head>
  <body>
      <div class="container">
        <div class="row my-5 pl-3">
            <h1>Predict Sales</h1>
        </div>

        <!-- Starts form section -->
        <div class="form-container ">
            <form class="form-horizontal" action = "/predict/"
method="post">

                <div class="form-group row">
                  <label class="control-label col-sm-2" for="tv">TV:
</label>
                  <div class="col-sm-4">
                    <input type="text" class="form-control" id="tv"
name="tv">
                  </div>
                </div>
        <div class="form-group row">
         <label class="control-label col-sm-2"
for="radio">Radio:</label>
         <div class="col-sm-4">
          <input type="text" class="form-control"
id="radio" name="radio">
         </div>
        </div>

        <div class="form-group row">
         <label class="control-label col-sm-2"
for="newspaper">Newspaper:</label>
         <div class="col-sm-4">
          <input type="text" class="form-control"
id="newspaper" name="newspaper">
         </div>
        </div>

        <div class="form-group row">
        <label class="control-label col-sm-2" for="">&nbsp;
</label>
         <div class="col-sm-offset-2 col-sm-4">
          <button type="submit" class="btn btn-
primary">Predict</button>
         </div>
        </div>
       </form>
      <!-- Ends form section -->
    </div>
  </div>
 </body>
</html>
```

Refresh the homepage

Already our server is running, just you need to refresh the page.



Create page for prediction

When we submit the form from home page, it will go to “/predict/” url.

Create predict method in app.py

We have to retrieve POST form data, so we are using request.form.get(‘key’).

```
//file: main.py
from flask import Flask, render_template, request


@app.route('/predict/', methods=['GET','POST'])
```

```
def predict():

    if request.method == "POST":

        #get form data
        tv = request.form.get('tv')
        radio = request.form.get('radio')
        newspaper = request.form.get('newspaper')

        return render_template('predict.html')
    pass
```

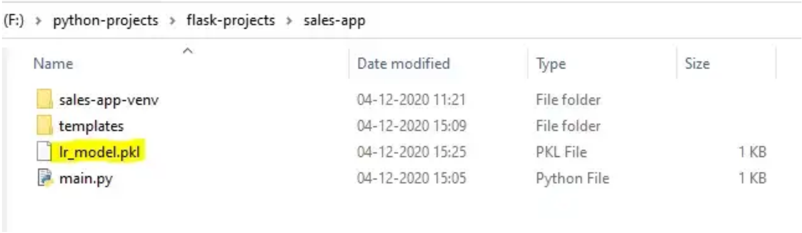
Create predict.html file inside templates folder

Write following code in that.

```
<!doctype html>
<html>
  <head>
    <title> Prediction </title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.m
in.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js
"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min
.js"></script>

  </head>
  <body>
      <div class="container">
        <div class="row my-5 pl-3">
            <h1>Prediction</h1>
        </div>

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

Keep saved model in project root folder

We have saved our trained model in part 1. We need to copy and paste the saved model in project root folder.



(F:) › python-projects › flask-projects › sales-app

Name	Date modified	Type	Size
sales-app-venv	04-12-2020 11:21	File folder	
templates	04-12-2020 15:09	File folder	
lr_model.pkl	04-12-2020 15:25	PKL File	1 KB
main.py	04-12-2020 15:05	Python File	1 KB

Pass form data to model and predict

Create new method called **preprocessDataAndPredict()** and pass all inputs.

```
//file: main.py

def preprocessDataAndPredict(tv, radio, newspaper):
    #put all inputs in array
    test_data = [tv, radio, newspaper]
    print(test_data)

    #convert value data into numpy array
    test_data = np.array(test_data).astype(np.float)

    #reshape array
    test_data = test_data.reshape(1,-1)
    print(test_data)

    #open file
    file = open("lr_model.pkl","rb")

    #load trained model
    trained_model = joblib.load(file)

    #predict
    prediction = trained_model.predict(test_data)

    return prediction

    pass
```

Call above method inside predict() method.

```
#call preprocessDataAndPredict and pass inputs
    try:
        prediction = preprocessDataAndPredict(tv, radio, newspaper)

        #pass prediction to template
        return render_template('predict.html', prediction =
prediction)

    except ValueError:
        return "Please Enter valid values"

    pass
```

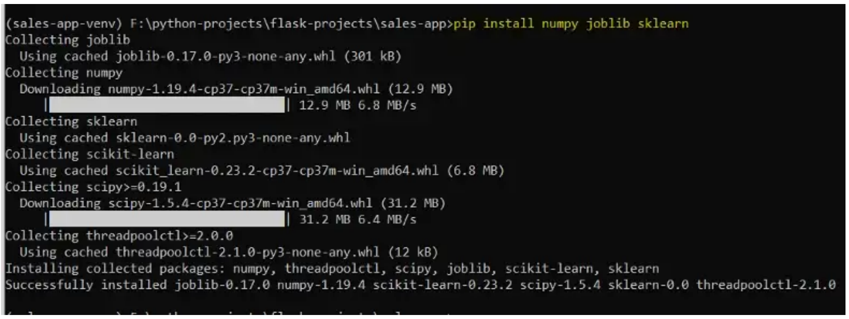
Modify predict.html file and print predicted result

```
<h1>Prediction is {{ prediction }}</h1>
```

Install required library for prediction

Stop the server, and install below library:

```
pip install numpy
pip install joblib
pip install sklearn
```



```
(sales-app-venv) F:\python-projects\flask-projects\sales-app>pip install numpy joblib sklearn
Collecting joblib
  Using cached joblib-0.17.0-py3-none-any.whl (301 kB)
Collecting numpy
  Downloading numpy-1.19.4-cp37-cp37m-win_amd64.whl (12.9 MB)
     |████████████████████████████████| 12.9 MB 6.8 MB/s
Collecting sklearn
  Using cached sklearn-0.0-py2.py3-none-any.whl
Collecting scikit-learn
  Using cached scikit_learn-0.23.2-cp37-cp37m-win_amd64.whl (6.8 MB)
Collecting scipy>=0.19.1
  Downloading scipy-1.5.4-cp37-cp37m-win_amd64.whl (31.2 MB)
     |████████████████████████████████| 31.2 MB 6.4 MB/s
Collecting threadpoolctl>=2.0.0
  Using cached threadpoolctl-2.1.0-py3-none-any.whl (12 kB)
Installing collected packages: numpy, threadpoolctl, scipy, joblib, scikit-learn, sklearn
Successfully installed joblib-0.17.0 numpy-1.19.4 scikit-learn-0.23.2 scipy-1.5.4 sklearn-0.0 threadpoolctl-2.1.0
```

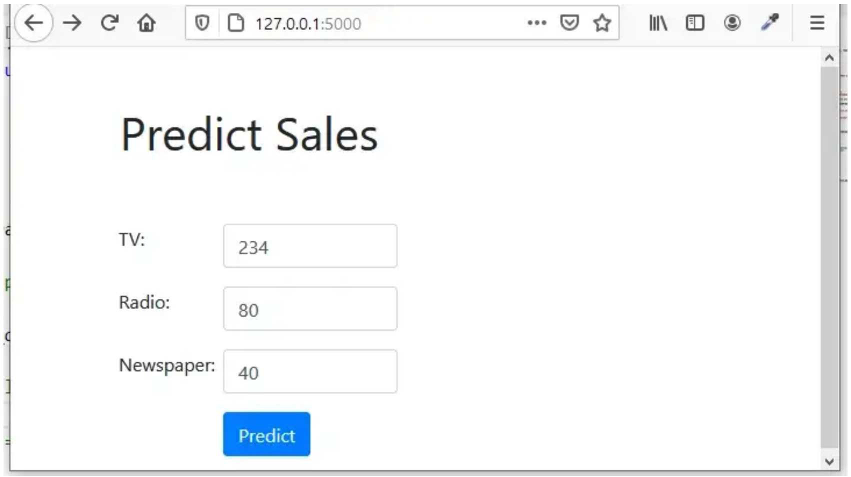
Updated full main.py file

```
#import Flask
from flask import Flask, render_template, request

#create an instance of Flask
app = Flask(__name__)

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

@app.route('/predict/', methods=['GET','POST'])
def predict():

    if request.method == "POST":

        #get form data
        tv = request.form.get('tv')
        radio = request.form.get('radio')
        newspaper = request.form.get('newspaper')

        #call preprocessDataAndPredict and pass inputs
        try:
            prediction = preprocessDataAndPredict(tv, radio, 
newspaper)

            #pass prediction to template
            return render_template('predict.html', prediction = 
prediction)

        except ValueError:
            return "Please Enter valid values"

        pass

    pass

def preprocessDataAndPredict(tv, radio, newspaper):

    #put all inputs in array
    test_data = [tv, radio, newspaper]
    print(test_data)

    #convert value data into numpy array and type float
    test_data = np.array(test_data).astype(np.float)

    #reshape array
    test_data = test_data.reshape(1,-1)
    print(test_data)

    #open file
    file = open("lr_model.pkl","rb")

    #load trained model
    trained_model = joblib.load(file)

    #predict
    prediction = trained_model.predict(test_data)

    return prediction

    pass

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

Run the application

```
python main.py
```

Browse url http://127.0.0.1:5000/, fill the form and click on Predict.



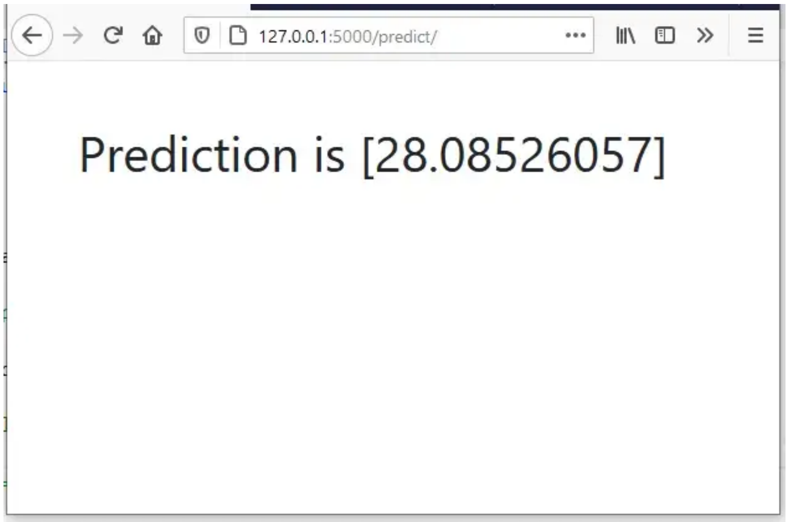
Predict Sales

TV: 234

Radio: 80

Newspaper: 40

Predicted result:



Prediction is [28.08526057]

Ml Model Deployment

Model Deployment Flask

Ml Deployment On Gcp

Google Cloud Platform

Ml Deployment Flask