## Deployment on Flask

Name: Yihsuan Sun

Email: yihsuan.sun88@gmail.com

Internship Batch: LISUM09 Submission date: 05/28/2022

Submitted to: Data Glacier Internship

#### Data:

toy\_datset.csv

#### Goal:

- Deploy the model on flask
- To predict a person is ill or not

#### Step 1: Data preprocessing

- Load the dataset
- Replace city name with number
- Replace gender with number

```
Desktop > FlaskPratice > 💠 app.py > ..
  1 from flask import Flask, render_template, request, jsonify
  2 import pandas as pd
     from sklearn.ensemble import RandomForestClassifier
  4 from sklearn.model_selection import train_test_split
  5 from sklearn.preprocessing import StandardScaler
  6 import pickle
     import numpy as np
 df = pd.read_csv("toy_dataset.csv")
 12 # read first few rows
 13 print(df.head())
     # replace
 16  df = df.replace(
        to_replace=['Austin', "Boston", "Dallas", "Los Angeles", "Mountain View", "New York City", "San Diego", "Washington D.C."],
 17
 18
         value=[1,2,3,4,5,6,7,8,])
 #print(df.head())
20 #print(df.tail())
      to_replace=['Male', "Female"],
value=[0,1])
```

#### Step 2:

- Select independent and dependent variables, and then split the data into training and testing
- Feature scaling
- Fit model
- Pickle

```
# Select independent and dependent vairables\
27  X = df[["City", "Gender", "Age", "Income"]]
y = df["Illness"]
30 # Splting the data into traing and testing
31 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
33 # Feature scaling
34
    sc = StandardScaler()
35
    X_train = sc.fit_transform(X_train)
36  X_test = sc.transform(X_test)
38
    # model
    classifier = RandomForestClassifier()
41 # fit model
42 classifier.fit(X_train, y_train)
45 pickle.dump(classifier, open("model.pkl", "wb"))
```

#### Step 3:

• Deploy the model on flask

```
49  app = Flask(_name__)
50
51  model = pickle.load(open("model.pkl", "rb"))
52
53  @app.route("/")
54  def home():
55  | return render_template("index.html")
56
57  @app.route("/predict", methods = ['POST'])
58  def predict():
59  | float_features = [float(x) for x in request.form.values()]
60  features = [np.array(float_features)]
61  prediction = model.predict(features)
62
63  return render_template("index.html", prediction_text = "Is the preson Ill? (Yes or No) {}".format(prediction))
64
65
66  if __name__ == '__main__':
67  | app.run(port = 3000, debug=True)
```

#### Step 4:

• Create documents designed to be displayed in a web browser (html)

### **Results:**

Enter the city, gender, age, and income to get the prediction result, the result will show whether a person is ill or not. (Yes or No)

This is an example.

Before adding:

## **Illness prediction**

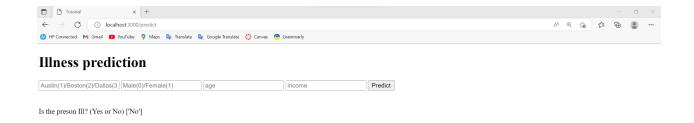
Austin(1)/Boston(2)/Dallas(3 Mal	ale(0)/Female(1)	age	income	Predict
----------------------------------	------------------	-----	--------	---------

Enter the city, gender, age, and income

### **Illness prediction**



Get the prediction



# **Conclusion:**

This web page uses the model to display the predicted results