Name: Yuchi Chen

Batch code: LISUM14

Submission date: 10/19/2022

Submitted to: https://github.com/ycchen00/DataGlacier-W4-Flask □

Model train and save (model.py)

```
# Importing the libraries
import pickle
import numpy as np
import pandas as pd
from sklearn import preprocessing
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
bank = pd.read_csv("bank_data_sk.csv")
nbank = bank.drop('CustomerID', axis=1)
nbank = nbank.dropna()
# one_hot_encoded=pd.get_dummies(nbank.State)
# nbank=pd.concat([nbank,one_hot_encoded],axis=1)
train_set, test_set = train_test_split(nbank, train_size=0.8)
cols = ['Age', 'Balance', 'IsActiveMember', 'CheckingAcct']
x_cols = ['Age', 'Balance', 'IsActiveMember']
y_col = 'CheckingAcct'
train_set = train_set[cols]
test_set = test_set[cols]
# nprmalization
ncolumns = ['Age', 'Balance']
scaler = preprocessing.MinMaxScaler()
nbank[ncolumns] = scaler.fit_transform(nbank[ncolumns])
X_train, X_test, y_train, y_test = train_test_split(nbank[x_cols],
nbank.CheckingAcct, train_size=0.8)
lr = LogisticRegression().fit(X_train, y_train)
```

```
# Saving model to disk
pickle.dump(lr, open('model.pkl', 'wb'))
pickle.dump(scaler, open('scaler.pkl', 'wb'))
py
```

Deploy the model on flask (app.py)

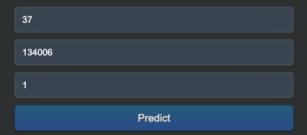
```
import numpy as np
import pandas as pd
from flask import Flask, request, render_template
import pickle
app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))
scaler = pickle.load(open('scaler.pkl', 'rb'))
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict',methods=['POST'])
def predict():
    1.1.1
    For rendering results on HTML GUI
    features = [float(x) for x in request.form.values()]
    test_df = pd.DataFrame([features], columns=['Age', 'Balance',
'IsActiveMember'])
    # final_features = [np.array(int_features)]
    # age = int(request.args.get("Age"))
    # balance = float(request.args.get("Balance"))
    # isActiveMember = int(request.args.get("IsActiveMember"))
    # test_df = pd.DataFrame({'Age': [age], 'Balance': [balance],
'IsActiveMember': [isActiveMember]})
    ncolumns = ['Age', 'Balance']
    test_df[ncolumns] = scaler.fit_transform(test_df[ncolumns])
    prediction = model.predict(test_df)
    prediction_text = "This person will sign up for a checking account" if
prediction == 1 else "This person will not sign up for a checking account"
    return render_template('index.html', prediction_text=prediction_text)
if __name__ == "__main__":
    app.run(debug=True)
                                                                            ру
```

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
       <link href='https://fonts.googleapis.com/css?family=Pacifico'</pre>
                      rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'</pre>
                               type='text/css'>
      <link href='https://fonts.googleapis.com/css?family=Hind:300'</pre>
                      rel='stylesheet' type='text/css'>
                <link href='https://fonts.googleapis.com/css?</pre>
     family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css')}</pre>
                                     }}">
</head>
<body>
  <div class="login">
    <h1>Predict a customer will sign up a checking account</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="Balance" placeholder="Balance" required="required"</pre>
<input type="text" name="IsActiveMember" placeholder="Is active member or</pre>
                      not (1/0)" required="required" />
      <button type="submit" class="btn btn-primary btn-block btn-large">
Predict</button>
    </form>
    <br>
    <br>
    {{ prediction_text }}
  </div>
```

Run

```
W4 - Flask model — python ∢ python app.py — 80×24
(base) ycchen@Yuchis-MacBook-Air W4 - Flask model % python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
* Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with watchdog (fsevents)
* Debugger is active!
* Debugger PIN: 505-457-970
127.0.0.1 - - [18/Oct/2022 22:59:03] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [18/Oct/2022 22:59:09] "POST /predict HTTP/1.1" 200 -
* Detected change in '/Users/ycchen/Library/CloudStorage/OneDrive-UniversityofR
ochester/Projects/000 Glacier/W4 - Flask model/app.py', reloading
* Restarting with watchdog (fsevents)
* Debugger is active!
* Debugger PIN: 505-457-970
* Detected change in '/Users/ycchen/Library/CloudStorage/OneDrive-UniversityofR
ochester/Projects/000 Glacier/W4 - Flask model/model.py', reloading
 * Detected change in '/Users/ycchen/Library/CloudStorage/OneDrive-UniversityofR
ochester/Projects/000 Glacier/W4 - Flask model/model.py', reloading
* Restarting with watchdog (fsevents)
 * Debugger is active!
```

Predict a customer will sign up a checking account



This person will not sign up for a checking account



Your Deep Learning Partner