Name: *Sreevatsa Nukala*
Batch: *LISUM15*
Submission Date: *11/27/2022*
Submitted to: https://github.com/Sreevatsa03/flask-tutorial

Flask Deployment

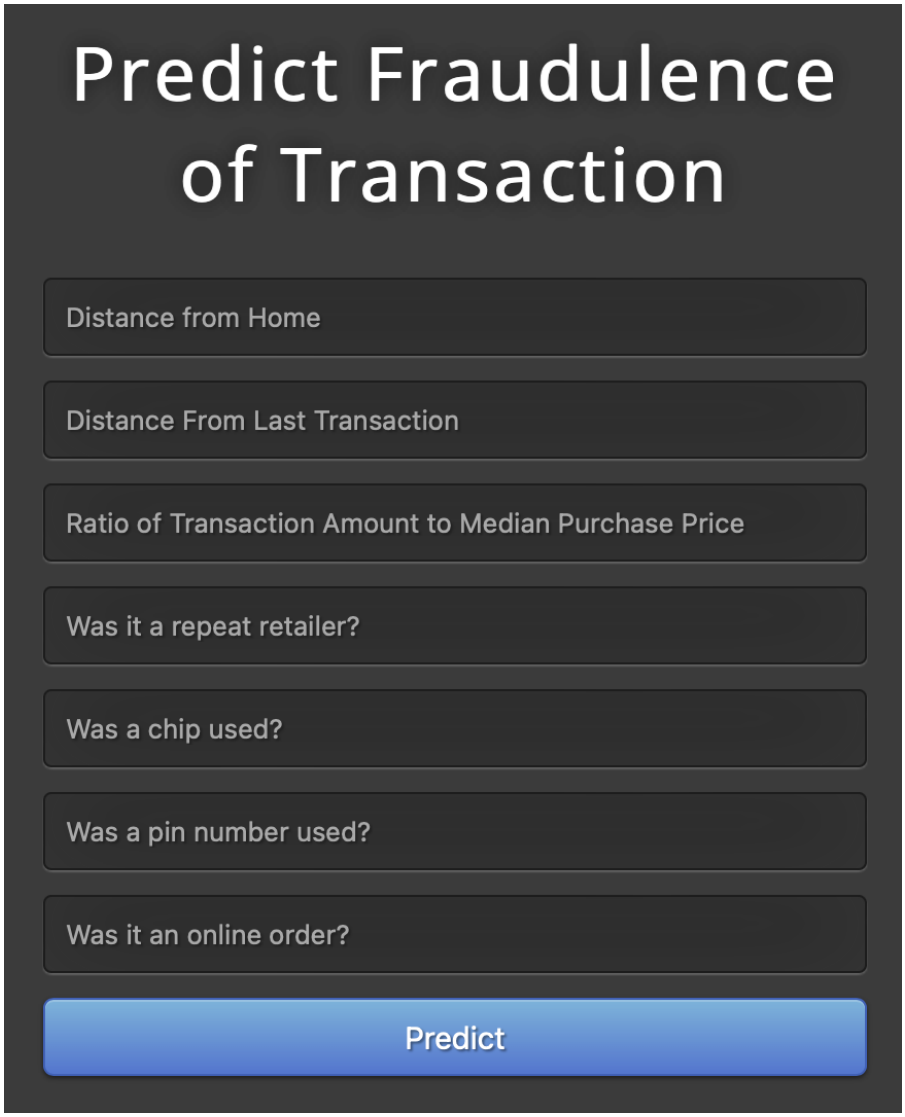
1.Model Creation

```
# imports
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report
import pandas as pd
import pickle

# read the data
card_df = pd.read_csv('card_transdata.csv')
card_df = card_df.astype({'repeat_retailer': int, 'used_chip': int,
                          'used_pin_number': int, 'online_order': int, 'fraud': int})

# downsample card_df
not_fraud = card_df[card_df.fraud == 0]
fraud = card_df[card_df.fraud == 1]

not_fraud = not_fraud.sample(n=5000, random_state=1)
fraud = fraud.sample(n=5000, random_state=1)

card_df_resampled = pd.concat([not_fraud, fraud])

# split X and y
X = card_df_resampled[card_df_resampled.columns.difference(['fraud'])]
y = pd.DataFrame(card_df_resampled['fraud'])

# split training and testing data
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.3,
                                                    random_state=7,
                                                    stratify=y
                                                    )

# classifier
classifier = SVC(C=1000, kernel='rbf', random_state=15)

# fit the model
classifier.fit(X_train, y_train)
```

2. Flask App Deployment

```
import numpy as np
from flask import Flask, request,render_template
import pickle

app = Flask(__name__)
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 = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = prediction[0]

    if output == 0:
        return render_template('index.html', prediction_text='Not Fraud')
    else:
        return render_template('index.html', prediction_text='Fraud')

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

3. Usage


Predict Fraudulence of Transaction
Distance from Home
Distance From Last Transaction
Ratio of Transaction Amount to Median Purchase Price
Was it a repeat retailer?
Was a chip used?
Was a pin number used?
Was it an online order?
Predict