Project: Flask Deployment

Name: Litvinov Sergey
Batch code: LISUM19
Submission date: 16.04.23
Submitted to: Github - https://github.com/LtvnSergey/Internship_Week_4_Flask_Deployment

Steps of deployment:

1. **Selecting data**

Diabetes dataset from Kaggle has been chosen as a data - https://www.kaggle.com/datasets/mathchi/diabetes-data-set

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes.

2. **Saving the model**

- The csv file with dataset was read as Pandas Dataframe
- The DataFrame was split into to features (X) and target (y)
- As for the regression model for predictions ExtraTreeRegressor has been chosen and trained on features
- After training the model was serialized using pickle and saved as 'model.pkt' file

```
# Importing the libraries
import pandas as pd
import pickle
from sklearn.ensemble import ExtraTreesRegressor


dataset = pd.read_csv('diabetes.csv', sep='\t')

X = dataset.iloc[:, :10]

y = dataset.iloc[:, -1]

regressor = ExtraTreesRegressor(n_estimators=100, random_state=0)

#Fitting model with trainig data
regressor.fit(X, y)


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

3. **Deploying the model on flask (web app)**

- Flask app has been initialized (line 5)
- Serialized model was read using pickle module (line 6)
- Home page with a path ‘/’ was set using html template from index.htlm file (lines 8 - 10)
- Page for prediction with path ‘/prediction’ was set with method ‘POST’, which means that the server will take data from the form and after that predict function will use it to perform an output

```
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.pkl', '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     '''
15     For rendering results on HTML GUI
16     '''
17     int_features = [[x for x in request.form.values()]]
18     prediction = model.predict(int_features)
19
20     output = round(prediction[0], 2)
21
22     return render_template('index.html', prediction_text='Quantitative measure of \
23                                                           disease progression one year \
24                                                           after baseline: {:.2f}'.format(output))
25
26 if __name__ == "__main__":
27     app.run(debug=True)
28
```

- Home page with a form for setting values for different features:

- After setting all the parameters and hitting ‘Predict’ button a model result with appear