AIM: DEPLOYMENT OF MACHINE LEARNING MODELS

THEORY:

ML-Model-Flask-Deployment

This is a mini project to elaborate how Machine Learn Models are deployed on production using Flask API

Prerequisites

You must have Numpy, Scikit Learn, Pandas (for Machine Learning Model) and Flask (for API) installed.

This project has four major parts :

model.py - This contains code for our Machine Learning model to predict employee salaries based on training data in 'hiring.csv' file.

app.py - This contains Flask APIs that receives employee details through GUI, computes the predicted value based on our model and returns it.

request.py - This uses a requests module to call APIs already defined in app.py and displays the returned value.

templates - This folder contains the HTML template to allow users to enter employee detail and displays the predicted employee salary.

Running the project

Create the machine learning model by running below command -

python model.py

This would create a serialized version of our model into a file

model.pkl Run app.py using below command to start Flask API *python*

app.py

By default, flask will run on port 5000.

Navigate to URL http://localhost:5000

You should be able to view the homepage.

Enter valid numerical values in all 3 input boxes and hit Predict.

You will be able to see the predicted salary value on the HTML page.

I) CREATING MODEL:

A) IMPORTING LIBRARIES:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

B) READING DATASET:

```
df = pd.read_csv('parkinsons.data')
```

```
df.head()
```

	name	MDVP:Fo(Hz)	MDVP:Fhi(Hz)	MDVP:Flo(Hz)	MDVP:Jitter(%)	MDVP:Jitter(Abs)	MDVP:RAP	MDVP:PPQ	Jitter:DDP	MDVP:Shimmer	MDVP:Shimmer(dB)	Shimmer:APQ3	Shimmer:APQ5	MDVP:APQ	Shimmer:DDA
0	phon_R01_S01_1	119.992	157.302	74.997	0.00784	0.00007	0.00370	0.00554	0.01109	0.04374	0.426	0.02182	0.03130	0.02971	0.06545
1	phon_R01_S01_2	122.400	148.650	113.819	0.00968	0.00008	0.00465	0.00696	0.01394	0.06134	0.626	0.03134	0.04518	0.04368	0.09403
2	phon_R01_S01_3	116.682	131.111	111.555	0.01050	0.00009	0.00544	0.00781	0.01633	0.05233	0.482	0.02757	0.03858	0.03590	0.08270
3	phon_R01_S01_4	116.676	137.871	111.366	0.00997	0.00009	0.00502	0.00698	0.01505	0.05492	0.517	0.02924	0.04005	0.03772	0.08771
4	phon_R01_S01_5	116.014	141.781	110.655	0.01284	0.00011	0.00655	0.00908	0.01966	0.06425	0.584	0.03490	0.04825	0.04465	0.10470

C) LISITING ALL COLUMNS:

```
df.columns
```

```
Index(['name', 'MDVP:Fo(Hz)', 'MDVP:Fhi(Hz)', 'MDVP:Flo(Hz)', 'MDVP:Jitter(%)',
       'MDVP:Jitter(Abs)', 'MDVP:RAP', 'MDVP:PPQ', 'Jitter:DDP',
       'MDVP:Shimmer', 'MDVP:Shimmer(dB)', 'Shimmer:APQ3', 'Shimmer:APQ5',
       'MDVP:APQ', 'Shimmer:DDA', 'NHR', 'HNR', 'status', 'RPDE', 'DFA',
       'spread1', 'spread2', 'D2', 'PPE'],
      dtype='object')
```

D) DEPTH INFO OF DATASET:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 24 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   name              195 non-null    object 
 1   MDVP:Fo(Hz)       195 non-null    float64
 2   MDVP:Fhi(Hz)      195 non-null    float64
 3   MDVP:Flo(Hz)      195 non-null    float64
 4   MDVP:Jitter(%)    195 non-null    float64
 5   MDVP:Jitter(Abs)  195 non-null    float64
 6   MDVP:RAP          195 non-null    float64
 7   MDVP:PPQ          195 non-null    float64
 8   Jitter:DDP        195 non-null    float64
 9   MDVP:Shimmer      195 non-null    float64
 10  MDVP:Shimmer(dB)  195 non-null    float64
 11  Shimmer:APQ3      195 non-null    float64
 12  Shimmer:APQ5      195 non-null    float64
 13  MDVP:APQ          195 non-null    float64
 14  Shimmer:DDA       195 non-null    float64
 15  NHR               195 non-null    float64
 16  HNR               195 non-null    float64
 17  status            195 non-null    int64  
 18  RPDE              195 non-null    float64
 19  DFA               195 non-null    float64
 20  spread1           195 non-null    float64
 21  spread2           195 non-null    float64
 22  D2                195 non-null    float64
 23  PPE               195 non-null    float64
dtypes: float64(22), int64(1), object(1)
memory usage: 36.7+ KB
```

E) <u>COUNTING NULL VALUES:</u>

```
df.isnull().sum()
```

```
name              0
MDVP:Fo(Hz)       0
MDVP:Fhi(Hz)      0
MDVP:Flo(Hz)      0
MDVP:Jitter(%)    0
MDVP:Jitter(Abs)  0
MDVP:RAP          0
MDVP:PPQ          0
Jitter:DDP        0
MDVP:Shimmer      0
MDVP:Shimmer(dB)  0
Shimmer:APQ3      0
Shimmer:APQ5      0
MDVP:APQ          0
Shimmer:DDA       0
NHR               0
HNR               0
status            0
RPDE              0
DFA               0
spread1           0
spread2           0
D2                0
PPE               0
dtype: int64
```

F) DIMENSIONS OF DATASET:

```
df.shape
```

```
(195, 24)
```

G) SPLITTING DATASET:

```
X = df.drop(['name'], 1)
X = X.drop(['status'], 1)
y = df['status']
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

H) NORMALIZATION:

```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0,1))
```

```
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

I) BUILDING XGBOOST MODEL:

```
from xgboost import XGBClassifier
model = XGBClassifier().fit(X_train, y_train)
```

```
predictions = model.predict(X_test)
```

J) CHECKING ACCURACY OF THE MODEL:

```
from sklearn.metrics import accuracy_score, f1_score
```

```
accuracy_score(y_test, predictions)
```

```
0.8974358974358975
```

```
f1_score(y_test, predictions)
```

```
0.9354838709677419
```

K) EXPORTING MODEL:

```
import pickle
# Writing different model files to file
with open( 'modelForPrediction.sav', 'wb') as f:
    pickle.dump(model,f)

with open('standardScalar.sav', 'wb') as f:
    pickle.dump(sc,f)
```

II) DEPLOYMENT OF MODEL:

FRONTEND:
HTML:
Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Parkinson's Diseases Prediction</title>
<link
href="https://fonts.googleapis.com/css2?family=Quicksand:wght@500&display=swap"
rel="stylesheet" />
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet"
  integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
  integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
  crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
  integrity="sha384-
7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB"
  crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js"
  integrity="sha384-
QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
  crossorigin="anonymous"></script>
</head>
<body>
    <header>
      <div class="container">
```

```
                <h1>
                  <p style="text-align: center">Parkinson's Diseases Prediction</p>
                </h1>
              </div>
            </header>

          <div class="container">

              <form action="/predict" method="POST">
                <div class="mb-3">
                <input type="float" name="mdvp_fo"  placeholder="MDVP:Fo(Hz)
range(88,260)" class="form-control" required/><br/>
                <input type="float" name="mdvp_fhi"  placeholder="MDVP:Fhi(Hz)
range(102,592)" class="form-control" required><br/>
                <input type="float" name="mdvp_flo" id="mdvp_flo"
placeholder="MDVP:Flo(Hz) range(65,240)" class="form-control" required><br />
                <input type="float" name="mdvp_jitper" id="mdvp_jitper"
placeholder="MDVP:Jitter(%) range(0.001, 0.033)" class="form-control" required><br />
                <input type="float" name="mdvp_jitabs" id="mdvp_jitabs"
placeholder="MDVP:Jitter(Abs) range(0.00002, 0.0002)" class="form-control"
required><br />
                <input type="float" name="mdvp_jitabs" id="mdvp_jitabs"
placeholder="MDVP:Jitter(Abs) range(0.00002, 0.0002)" class="form-control"
required><br />
                <input type="float" name="mdvp_rap" id="mdvp_rap"
placeholder="MDVP:RAP range(0.0006, 0.02)" class="form-control" required><br />
                <input type="float" name="mdvp_ppq" id="mdvp_ppq"
placeholder="MDVP:PPQ range(0.0009, 0.02)" class="form-control" required><br />
                <input type="float" name="jitter_ddp" id="jitter_ddp" placeholder="Jitter:DDP
range(0.002, 0.065)" class="form-control" required><br />
                <input type="float" name="mdvp_shim" id="mdvp_shim"
placeholder="MDVP:Shimmer range(0.009, 0.12)" class="form-control" required><br />
                <input type="float" name="mdvp_shim_db" id="mdvp_shim_db"
placeholder="MDVP:Shimmer(dB) range(0.085, 1.302)" class="form-control"
required><br />
                <input type="float" name="shimm_apq3" id="shimm_apq3"
placeholder="Shimmer:APQ3 range(0.004, 0.056)" class="form-control" required><br />
                <input type="float" name="shimm_apq5" id="shimm_apq5"
placeholder="Shimmer:APQ5 range(0.005, 0.08)" class="form-control" required><br />
                <input type="float" name="mdvp_apq" id="mdvp_apq"
placeholder="MDVP:APQ range(0.007, 0.14)" class="form-control" required><br />
                <input type="float" name="shimm_dda" id="shimm_dda"
placeholder="Shimmer:DDA range(0.013, 0.17)" class="form-control" required><br />
                <input type="float" name="nhr" id="nhr" placeholder="NHR range(0.0006,
0.31)" class="form-control" required><br />
                <input type="float" name="hnr" id="hnr" placeholder="HNR range(8, 33)"
class="form-control" required><br />
                <input type="float" name="rpde" id="rpde" placeholder="RPDE range(0.25,
0.68)" class="form-control" required><br />
                <input type="float" name="dfa" id="dfa" placeholder="DFA range(0.57, 0.82)"
class="form-control" required><br />
```

```
            <input type="float" name="spread1" id="spread1" placeholder="Spread1
  range(-7, -2)" class="form-control" required><br />
            <input type="float" name="spread2" id="spread2" placeholder="Spread2
  range(0.006, 0.45)" class="form-control" required><br />
            <input type="float" name="d2" id="d2" placeholder="D2 range(1.42, 3.67)"
  class="form-control" required><br />
            <input type="float" name="ppe" id="ppe" placeholder="PPE range(0.04, 0.5)"
  class="form-control" required><br />

            <button type="submit" class="btn btn-primary"
  value="Predict">Predict</button>
          </div>
        </form>
      </div>
  </body>
  </html>
```

Result.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Predicted Result</title>
  <link
href="https://fonts.googleapis.com/css2?family=Quicksand:wght@500&display=swap"
rel="stylesheet" />
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet"
    integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
    crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
    integrity="sha384-
7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB"
    crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js"
    integrity="sha384-
QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
    crossorigin="anonymous"></script>
</head>
<body>
  <div class="container">
    <h2>Predicted Result</h2>
    <p>{{prediction}}</p>
  </div>
```

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

BACKEND:

PYTHON:

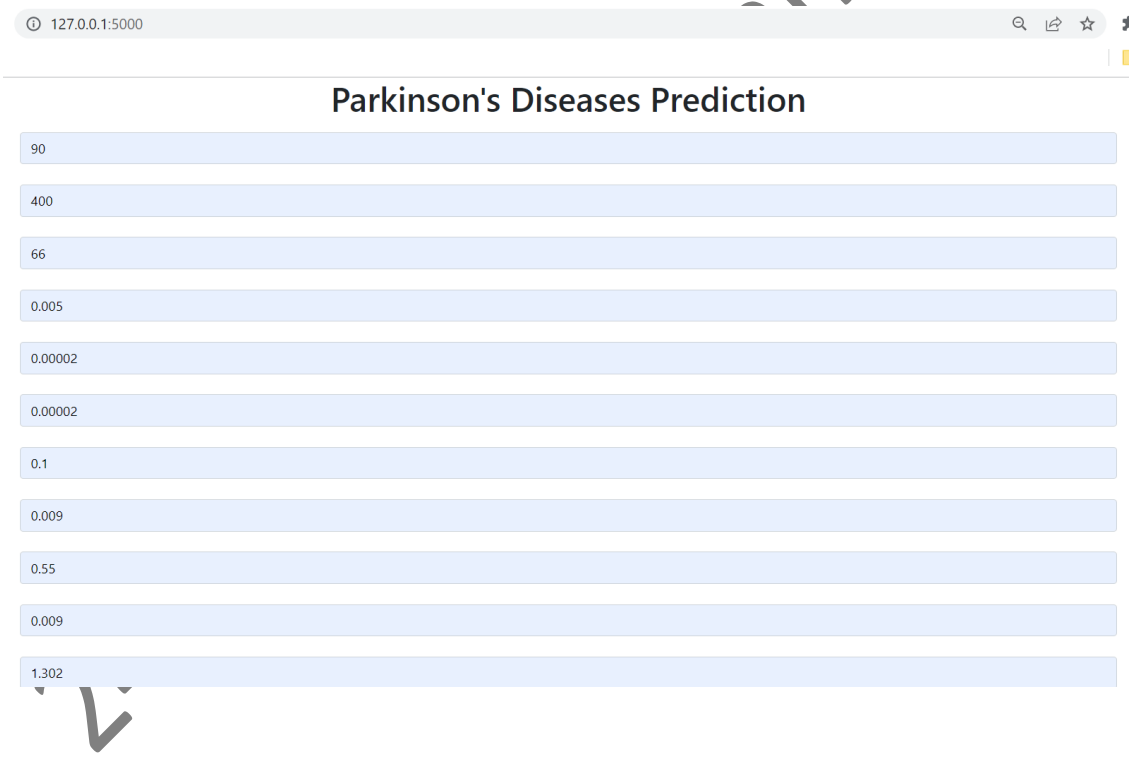
app.py:

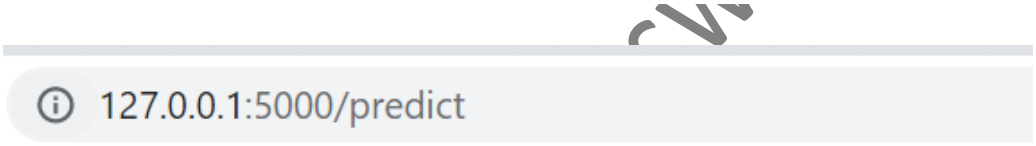
```
# importing the necessary dependencies
from flask import Flask, render_template, request
from flask_cors import CORS,cross_origin
import pickle

app = Flask(__name__) # initializing a flask app

@app.route('/',methods=['GET'])  # route to display the home page
@cross_origin()
def homePage():
    return render_template("index.html")

@app.route('/predict',methods=['POST','GET']) # route to show the predictions in a web
UI
@cross_origin()
def index():
    if request.method == 'POST':
        #  reading the inputs given by the user
        mdvp_fo=float(request.form['mdvp_fo'])
        mdvp_fhi=float(request.form['mdvp_fhi'])
        mdvp_flo=float(request.form['mdvp_flo'])
        mdvp_jitper=float(request.form['mdvp_jitper'])
        mdvp_jitabs=float(request.form['mdvp_jitabs'])
        mdvp_rap=float(request.form['mdvp_rap'])
        mdvp_ppq=float(request.form['mdvp_ppq'])
        jitter_ddp=float(request.form['jitter_ddp'])
        mdvp_shim=float(request.form['mdvp_shim'])
        mdvp_shim_db=float(request.form['mdvp_shim_db'])
        shimm_apq3=float(request.form['shimm_apq3'])
        shimm_apq5=float(request.form['shimm_apq5'])
        mdvp_apq=float(request.form['mdvp_apq'])
        shimm_dda=float(request.form['shimm_dda'])
        nhr=float(request.form['nhr'])
        hnr=float(request.form['hnr'])
        rpde=float(request.form['rpde'])
        dfa=float(request.form['dfa'])
        spread1=float(request.form['spread1'])
        spread2=float(request.form['spread2'])
        d2=float(request.form['d2'])
        ppe=float(request.form['ppe'])

        filename = 'modelForPrediction.sav'
        loaded_model = pickle.load(open(filename, 'rb')) # loading the model file from the
storage
```

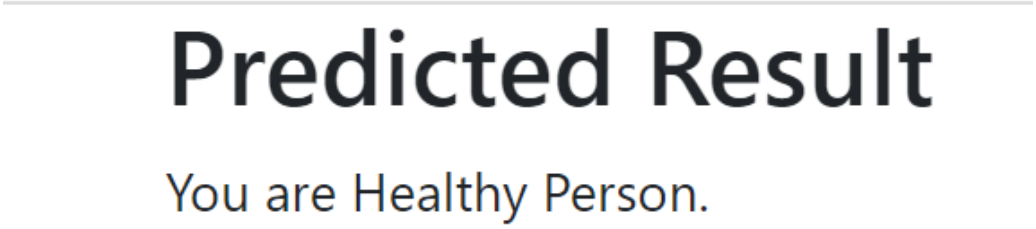
```
        # predictions using the loaded model file
    scaler = pickle.load(open('standardScalar.sav', 'rb'))
    prediction=loaded_model.predict(scaler.transform([[mdvp_fo,mdvp_fhi,mdvp_flo,
mdvp_jitper, mdvp_jitabs,
          mdvp_rap,mdvp_ppq, jitter_ddp, mdvp_shim,
mdvp_shim_db,shimm_apq3,shimm_apq5,mdvp_apq,shimm_dda,nhr,hnr,rpde,dfa,spr
ead1,spread2,d2,ppe]]))
    print('prediction is', prediction)
    if prediction == 1:
      pred = "You have Parkinson's Disease. Please consult a specialist."
      return render_template('results.html', prediction=pred)
    else:
      pred = "You are Healthy Person."
      # showing the prediction results in a UI
      return render_template('results.html',prediction=pred)
  else:
    return render_template('index.html')


if __name__ == "__main__":
  #app.run(host='127.0.0.1', port=8001, debug=True)
  app.run(debug=False) # running the app
```

OUTPUT:


MDVP:APQ range(0.007, 0.14)
Shimmer:DDA range(0.013, 0.17)
NHR range(0.0006, 0.31)
HNR range(8, 33)
RPDE range(0.25, 0.68)
DFA range(0.57, 0.82)
Spread1 range(-7, -2)
Spread2 range(0.006, 0.45)
D2 range(1.42, 3.67)
PPE range(0.04, 0.5)
Predict


127.0.0.1:5000
Parkinson's Diseases Prediction
90
400
66
0.005
0.00002
0.00002
0.1
0.009
0.55
0.009
1.302





CONCLUSION:

From this practical, I have learned how to deploy machine learning applications using flask python framework.