**SOURCE CODE**

import os

from flask import Flask, flash, redirect, render\_template, request, session, abort

from models.keras\_first\_go import KerasFirstGoModel

from clear\_bash import clear\_bash

import numpy as np

from flask import Flask, request, jsonify, render\_template

from flask import Flask, render\_template, flash, url\_for, request, session, redirect, jsonify

import os

import secrets

from PIL import Image

import tensorflow as tf

import keras

from keras.models import load\_model

from tensorflow.python.keras.backend import set\_session

from tensorflow.python.keras.models import load\_model

from tensorflow.keras.optimizers import Adam

import pickle as pickle

import joblib

model1 = joblib.load("models/model.sav")

scalerX = pickle.load(open("models/scalerX", "rb"))

app = Flask(\_\_name\_\_)

cleaner=clear\_bash()

model = load\_model('models/model1.h5')

graph = tf.compat.v1.get\_default\_graph()

def train\_model():

global first\_go\_model

print("Train the model")

first\_go\_model = KerasFirstGoModel()

@app.route("/")

def home():

return render\_template('index.html')

@app.route('/index1')

def index1():

return render\_template('index1.html')

@app.route('/index2')

def index2():

return render\_template('index2.html')

@app.route('/index3')

def index3():

return render\_template('index3.html')

@app.route("/main")

def main():

return render\_template('main.html')

def save\_picture(form\_picture):

random\_hex = secrets.token\_hex(8)

\_, f\_ext = os.path.splitext(form\_picture.filename)

picture\_fn = random\_hex + f\_ext

picture\_path = os.path.join(app.root\_path, 'static/profile\_pics', picture\_fn)

output\_size = (125,125)

i = Image.open(form\_picture)

i.thumbnail(output\_size)

i.save(picture\_path)

return picture\_fn

@app.route("/predict4", methods = ['POST'])

def predict4():

int\_features= [int(x) for x in request.form.values()]

print(int\_features,len(int\_features))

final4=[np.array(int\_features)]

prediction4 = model1.predict(scalerX.transform([int\_features]))

output4=round(prediction4[0],2)

print(output4)

if (int(output4)==0):

prediction = "Either anyone of this job position: Database Developer, Techinal Support, Business Intelligence Analyst, Business Systems Analyst, Portal Administrator, Data Architect "

elif (int(output4)==1):

prediction = "Either anyone of this job position: Systems Security Administrator, CRM Technical Developer,Software Systems Engineer,Mobile Applications Developer,UX Designer,Quality Assurance Associate "

elif (int(output4)==2):

prediction = "Either anyone of this job position: Web Developer,Information Security Analyst, CRM Business Analyst, Project Manager,Information Technology Manager,Programmer Analyst"

elif (int(output4)==3):

prediction = "Either anyone of this job position: Design & UX, Solutions Architect, Systems Analyst, Network Security Administrator,Data Architect,Software Developer"

elif (int(output4)==4):

prediction = "Either anyone of this job position: E-Commerce Analyst, Technical Services/Help Desk/Tech Support, Information Technology Auditor, Database Manager, Applications Developer,Database Administrator "

elif (int(output4)==5):

prediction = "Either anyone of this job position: Network Engineer, Software Engineer, Technical Engineer,Network Security Engineer, Software Quality Assurance (QA) / Testing "

else:

prediction = "invaild!"

return (render\_template('index3.html', prediction\_text = prediction))

@app.route('/prediction',methods=['POST'])

def prediction():

os = request.form["os"]

aoa = request.form["aoa"]

pc = request.form["pc"]

se = request.form["se"]

cn = request.form["cn"]

ma = request.form["ma"]

cs = request.form["cs"]

hac = request.form["hac"]

interest = request.form["interest"]

cert = request.form["cert"]

personality = request.form["personality"]

mantech = request.form["mantech"]

leadership = request.form["leadership"]

team = request.form["team"]

selfab = request.form["selfab"]

myu = [77.00318789848731, 76.99831228903614, 77.07569696212026, 77.11301412676585, 76.9541817727216, 77.0150018752344, 77.060320040005, 5.002687835979497]

sig = [10.071578660726848, 10.098653693844197, 10.137528173238477, 10.088164425588161, 10.018397202418788, 10.18533143324003, 10.095941558583263, 2.582645138598079]

arr = [os,aoa,pc,se,cn,ma,cs,hac]

for i in range(8):

arr[i] = float(arr[i])

arr[i] = (arr[i]- myu[i])/sig[i]

inti = [0,0,0,0,0,0,0,0,0,0,0,0,0]

certi = [0,0,0,0,0,0,0]

if interest == "analyst":

inti[0] = 1

elif interest == "hadoop":

inti[1] = 2

elif interest == "cloud":

inti[2] = 3

elif interest == "data":

inti[3] = 4

elif interest == "hacking":

inti[4] = 5

elif interest == "management":

inti[5] = 6

elif interest == "networks":

inti[6] = 7

elif interest == "programming":

inti[7] = 8

elif interest == "security":

inti[8] = 9

elif interest == "software":

inti[9] = 10

elif interest == "system":

inti[10] = 11

elif interest == "testing":

inti[11] = 12

elif interest == "web":

inti[12] = 13

if cert == "app":

certi[0] = 1

elif cert == "full":

certi[1] = 2

elif cert == "hadoop":

certi[2] = 3

elif cert == "security":

certi[3] = 4

elif cert == "machine":

certi[4] = 5

elif cert == "python":

certi[5] = 6

elif cert == "shell":

certi[6] = 7

for i in certi:

arr.append(i)

for i in inti:

arr.append(i)

if leadership == "yesl":

arr.append(0)

arr.append(1)

else:

arr.append(1)

arr.append(0)

if team == "yest":

arr.append(0)

arr.append(1)

else:

arr.append(1)

arr.append(0)

if personality == "extrovert":

arr.append(1)

arr.append(0)

else:

arr.append(0)

arr.append(1)

if selfab == "nos":

arr.append(1)

arr.append(0)

else:

arr.append(0)

arr.append(1)

if mantech == "man":

arr.append(1)

arr.append(0)

else:

arr.append(0)

arr.append(1)

print ('arr ',arr)

y = model.predict(np.array( [arr,]))

result = np.where(y == np.amax(y))

print(y)

print(result)

if result[0]==[0]:

return render\_template('index1.html', prediction\_text='Business Intelligence')

print('Business Intelligence Analyst')

elif result[0]==[1]:

return render\_template('index1.html', prediction\_text='Database Administrator')

print('Database Administrator')

elif result[0]==[2]:

return render\_template('index1.html', prediction\_text='Project Manager')

print('Project Manager')

elif result[0]==[3]:

return render\_template('index1.html', prediction\_text='Security Administrator')

print('Security Administrator')

elif result[0]==[4]:

return render\_template('index1.html', prediction\_text='Software Developer')

print('Software Developer')

else:

return render\_template('index1.html', prediction\_text='Technical Support')

print('Technical Support')

print("done2")

@app.route('/result',methods = ['POST', 'GET'])

def result():

if request.method == 'POST':

result = request.form.getlist('Job')

train\_model()

processed\_text = first\_go\_model.prediction(result[0])

result = {'Job': processed\_text}

return render\_template("result.html",result = result)

def clear\_bash():

os.system('cls' if os.name == 'nt' else 'clear')

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)