INTEGRATE FLASK WITH SCORING END POINT

|  |  |
| --- | --- |
| **Team ID** | **PNT2022TMID42057** |
| **Project Name** | **Car Resale value Prediction** |

**INTEGRATE FLASK WITH SCORING END POINT**

import pandas as pd import numpy as np

from flask import Flask,render\_template,Response,request import pickle

from sklearn.preprocessing import LabelEncoder import pickle

import requests import json

# NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account.

API\_KEY = "hEAn\_mcoP3u\_-ZjagjeqlxDayqUiETpYVYWdR1OLKAby"

token\_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API\_KEY, "grant\_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token\_response.json()["access\_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

app=Flask( name ,template\_folder='templates/') @app.route('/')

def index():

return render\_template('index.html')

@app.route('/resaleintro.html') def p():

return render\_template('resaleintro.html')

@app.route('/predict') def predict():

return render\_template('resalepredict.html')

@app.route('/y\_predict',methods=['GET','POST']) def y\_predict():

regyear = int(request.form['regyear']) powerps = float(request.form['powerps']) kms = float(request.form['kms'])

regmonth = int(request.form.get('regmonth')) gearbox = request.form['gearbox']

damage = request.form['dam']

model = request.form.get('model\_type') brand = request.form.get('brand')

fuelType = request.form.get('fuel') vehicletype= request.form.get('vehicletype') new\_row =

{'yearOfRegistration':regyear,'powerPS':powerps,'kilometer':kms,'monthOfRegistration':regm onth,'gearbox':gearbox,'notRepairedDamage':damage,'model':model,'brand':brand,'fuelType':f uelType,'vehicleType':vehicletype}

print(new\_row) new\_df =

pd.DataFrame(columns=['vehicleType','yearOfRegistration','gearbox','powerPS','model','kilo meter','monthOfRegistration','fuelType','brand','notRepairedDamage'])

new\_df = new\_df.append(new\_row,ignore\_index=True)

labels = ['gearbox','notRepairedDamage','model','brand','fuelType','vehicleType'] mapper = {}

for i in labels:

mapper[i] = LabelEncoder()

mapper[i].classes\_ = np.load(str('classes'+i+'.npy'),allow\_pickle=True) tr = mapper[i].fit\_transform(new\_df[i])

new\_df.loc[:,i+'\_Labels'] = pd.Series(tr,index=new\_df.index)

labeled = new\_df[ ['yearOfRegistration','powerPS','kilometer','monthOfRegistration'] + [x+"\_Labels" for x in labels]]

X = labeled.values print(X)

# return render\_template('resalepredict.html',ypred="{:.2f}".format(y\_prediction[0]))

payload\_scoring = {"input\_data": [{"field": [['vehicleType','yearOfRegistration','gearbox','powerPS','model','kilometer','monthOfRegis tration','fuelType','brand','notRepairedDamage']], "values": X.tolist()}]}

response\_scoring = requests.post('https://us- south.ml.cloud.ibm.com/ml/v4/deployments/816483ac-44ed-4be2-b780- 7f63d68fc7ce/predictions?version=2022-11-17', json=payload\_scoring,

headers={'Authorization': 'Bearer ' + mltoken}) print("Scoring response")

predictions = response\_scoring.json() print(predictions['predictions'][0]['values'][0][0]) return

render\_template('resalepredict.html',ypred="{:.2f}".format(predictions['predictions'][0][' values'][0][0]))

if name == ' main ': app.run(host='Localhost',debug=True,threaded=False)

