```
Import Libraries

Import pandas as pd

Import numpy as np

From flask import Flask, render_template, Response, request

Import pickle

From sklearn.preprocessing import LabelEncoder

Import requests


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

API_KEY = “Qo9j8ni7qMJ8j1C8VFDRFHbuGRAhYWcTlkVqnYg1AGkE”

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__)#initiate flask app


Def load_model(file=’../Result/resale_model.sav’):#load the saved model

        Return pickle.load(open(file, ‘rb’))


@app.route(‘/’)

Def index():#main page

        Return render_template(‘car.html’)


@app.route(‘/predict_page’)

Def predict_page():#predicting page

        Return render_template(‘value.html’)
```

```
@app.route(‘/predict’, methods=[‘GET’,’POST’])

Def predict():

    Reg_year = int(request.args.get(‘regyear’))

    Powerps = float(request.args.get(‘powerps’))

    Kms= float(request.args.get(‘kms’))

    Reg_month = int(request.args.get(‘regmonth’))


    Gearbox = request.args.get(‘geartype’)

    Damage = request.args.get(‘damage’)

    Model = request.args.get(‘model’)

    Brand = request.args.get(‘brand’)

    Fuel_type = request.args.get(‘fuelType’)

    Veh_type = request.args.get(‘vehicletype’)


    New_row = {‘yearOfReg’:reg_year, ‘powerPS’:powerps, ‘kilometer’:kms,

                ‘monthOfRegistration’:reg_month, ‘gearbox’:gearbox,

                ‘notRepairedDamage’:damage,

                ‘model’:model, ‘brand’:brand, ‘fuelType’:fuel_type,

                ‘vehicletype’:veh_type}


    Print(new_row)


    New_df = pd.DataFrame(columns=[‘vehicletype’,’yearOfReg’,’gearbox’,

        ‘powerPS’,’model’,’kilometer’,’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(‘../Result/’+str(‘classes’+i+’.npy’), allow_pickle=True)
        Transform = mapper[i].fit_transform(new_df[i])
        New_df.loc[:,i+’_labels’] = pd.Series(transform, index=new_df.index)

    Labeled = new_df[[‘yearOfReg’,’powerPS’,’kilometer’,’monthOfRegistration’] + [x+’_labels’ for x in labels]]

    X = labeled.values.tolist()
    Print(‘\n\n’, X)
    #predict = reg_model.predict(X)

    # NOTE: manually define and pass the array(s) of values to be scored in the next line
    Payload_scoring = {“input_data”: [{“fields”: [[‘yearOfReg’, ‘powerPS’, ‘kilometer’, ‘monthOfRegistration’,’gearbox_labels’, ‘notRepairedDamage_labels’, ‘model_labels’,’brand_labels’, ‘fuelType_labels’, ‘vehicletype_labels’]], “values”: X}]}

    Response_scoring = requests.post(‘https://us-south.ml.cloud.ibm.com/ml/v4/deployments/7f67cbed-6222-413b-9901-b2a72807ac82/predictions?version=2022-10-30’, json=payload_scoring, headers={‘Authorization’: ‘Bearer ‘ + mltoken})
    Predictions = response_scoring.json()
    Print(response_scoring.json())
    Predict = predictions[‘predictions’][0][‘values’][0][0]
    Print(“Final prediction :”,predict)

    Return render_template(‘predict.html’,predict=predict)

If __name__==’__main__’:
    Reg_model = load_model()#load the saved model

    App.run(host=’localhost’, debug=True, threaded=False)
```

Footer