

# Car Resale Value Prediction

## Build The Python Flask App

```
# 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

app = Flask(__name__)#initiate flask app

def load_model(file='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(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)

    #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(debug=True)

```