Batch F-3

Lalit Kumar Sharma

Sanjana Chitti Reddy

Roshan Mathew

Bajanthri Vyshnavi

Title : Analysis of Used Car Sales and Prediction of Selling Price

Best ML Model : Random Forest Regression

We have implemented many algorithms like Linear Regression, Decision Tree Regressor, Adaboost Regressor, Random Forest Regressor etc..,. We got the Random Forest Regressor as best model with least Mean Absolute Error and MSE. So we finalized it as best model and we deployed it into flask.

I) Download Pickle File

II) Create HTML file to read input values through template to find selling price of car

III) Create Flask Application and Deploy ML pickle file in it.

App.py File

HTML File:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>

    <div style="color:rgb(16, 233, 45)">
        <form action="{{ url_for('predict')}}" method="post">
            <h1>Prediction of Price of Used Cars</h1>
            <h2>What is the car model(year)</h2>
            <input id="first" name="year" type="number ">
            <h2>How many kms does car has to be driven?</h2><input id="second"
name="km_driven" required="required">
            <h2>What is average mileage needed(kmpl)?</h2><input id="third"
name="AvgMileage" required="required">
            <h2>Kind of Engine CC of car Required?</h2><input id="fourth"
name="EngineCC" required="required">
            <h2>Max power of Car</h2><input id="fifth" name="Maxpower"
required="required">
            <h2>What is the Fuel type required?</h2><br><select
name="Fuel_Petrol" id="Fuel" required="required">
                <option value="Petrol">Petrol</option>
                <option value="Diesel">Diesel</option>
                <option value="CNG">CNG</option>
                <option value="LPG">LPG</option>
            </select>
            <h2>What kind of seller are you looking for?</h2><br><select
name="SellerType_Dealer" id="sellertype" required="required">
                <option value="Dealer">Dealer</option>
                <option value="Individual">Individual</option>
                <option value="TrustmarkDealer">TrustmarkDealer</option>
            </select>
            <h2>Which type are you comfortable with?</h2><br><select
name="Transmission_Manual" id="Transmission" required="required">
```

```
                <option value="Manual">Manual Car</option>
                <option value="Automatic">Automatic Car</option>
            </select>
            <h2>Owner type requirement</h2><br><select name="Owner_First Owner"
id="owner" required="required">
                <option value="first">one Owner</option>
                <option value="second">two owners</option>
                <option value="third">three owners</option>
                <option value="fourth">four and above</option>
                <option value="testdrive">Test drive car</option>
            </select>
            <h3>What is requirement of seating capacity?</h3><br><select
name="Seat_high" id="seats" required="required">
                <option value="normal">five seats</option>
                <option value="medium">seven seater</option>
                <option value="high">greater than seven</option>
            </select>

            <br><br><button id="sub" type="submit ">Calculate the Price</button>
            <br>




        </form>



        <br><br><h3>{{ prediction_text }}<h3>
    </div>




    <style>
        body {
            background-color: rgb(1, 11, 15);
            text-align: center;
            padding: 0px;
        }

        #box {
            border-radius: 60px;
            border-color: 45px;
            border-style: solid;
```

```
        font-family: cursive;
        text-align: center;
        background-color: rgb(56, 4, 124);
        font-size: medium;
        position: absolute;
        width: 700px;
        bottom: 9%;
        height: 850px;
        right: 30%;
        padding: 0px;
        margin: 0px;
        font-size: 14px;
    }

    #fuel {
        width: 99px;
        height: 43px;
        text-align: center;
        border-radius: 14px;
        font-size: 18px;
    }

    #fuel:hover {
        background-color: rgb(24, 139, 233);
    }

    #sellertype {
        width: 99px;
        height: 43px;
        text-align: center;
        border-radius: 14px;
        font-size: 18px;
    }

    #sellertype:hover {
        background-color: rgb(240, 17, 84);
    }

    #Transmission {
        width: 99px;
        height: 43px;
        text-align: center;
        border-radius: 14px;
        font-size: 18px;
    }
```

```css
#Transmission:hover {
    background-color: coral;
}

#owner {
    width: 99px;
    height: 43px;
    text-align: center;
    border-radius: 14px;
    font-size: 18px;
}

#owner:hover {
    background-color: coral;
}

#seats {
    width: 99px;
    height: 43px;
    text-align: center;
    border-radius: 14px;
    font-size: 18px;
}

#seats:hover {
    background-color: rgb(27, 240, 8);
}
#sub {
    width: 120px;
    height: 43px;
    text-align: center;
    border-radius: 14px;
    font-size: 18px;
}

#sub:hover {
    background-color: darkcyan;
}

#first {
    border-radius: 14px;
    height: 25px;
    font-size: 20px;
    text-align: center;
```

```
        }

        #second {
            border-radius: 14px;
            height: 25px;
            font-size: 20px;
            text-align: center;
        }

        #third {
            border-radius: 14px;
            height: 25px;
            font-size: 20px;
            text-align: center;
        }

        #fourth {
            border-radius: 14px;
            height: 25px;
            font-size: 20px;
            text-align: center;
        }

        #fifth {
            border-radius: 14px;
            height: 25px;
            font-size: 20px;
            text-align: center;
    </style>
</body>

</html>
```

Flask(app.py):

```
from flask import Flask, render_template, request
import jsonify
import requests
import pickle
import numpy as np
import sklearn
from sklearn.preprocessing import StandardScaler
app = Flask(__name__)
model = pickle.load(open('random_forest_regression_model.pkl', 'rb'))
@app.route('/',methods=['GET'])
def Home():
    return render_template('index.html')
```

```


standard_to = StandardScaler()
@app.route("/predict", methods=['POST'])
def predict():
    if request.method == 'POST':
        year = int(request.form['year'])

        km_driven=float(request.form['km_driven'])

        AvgMileage=float(request.form['AvgMileage'])

        EngineCC=int(request.form['EngineCC'])

        MaxPower=float(request.form['Maxpower'])

        Fuel_Petrol=request.form['Fuel_Petrol']
        if(Fuel_Petrol=='Petrol'):
                Fuel_CNG=0
                Fuel_Diesel=0
                Fuel_LPG=0
                Fuel_Petrol=1
        elif(Fuel_Petrol=='Diesel'):
                Fuel_CNG=0
                Fuel_Diesel=1
                Fuel_LPG=0
                Fuel_Petrol=0
        elif(Fuel_Petrol=='CNG'):
                Fuel_CNG=1
                Fuel_Diesel=0
                Fuel_LPG=0
```

```
            Fuel_Petrol=0
    else:
            Fuel_CNG=0
            Fuel_Diesel=0
            Fuel_LPG=1
            Fuel_Petrol=0

    SellerType_Dealer=request.form['SellerType_Dealer']
    if(SellerType_Dealer=='Dealer'):
            SellerType_Dealer=1
            SellerType_Individual=0
            SellerType_TrustmarkDealer=0
    elif(SellerType_Dealer=='Individual'):
            SellerType_Dealer=0
            SellerType_Individual=1
            SellerType_TrustmarkDealer=0
    else:
             SellerType_Dealer=0
             SellerType_Individual=0
             SellerType_TrustmarkDealer=1

    Transmission_Manual=request.form['Transmission_Manual']
    if(Transmission_Manual=='Manual'):
        Transmission_Automatic=0
        Transmission_Manual=1
    else:
        Transmission_Automatic=1
        Transmission_Manual=0

    Owner_FirstOwner=request.form['Owner_First Owner']
    if(Owner_FirstOwner=='first'):
           Owner_FirstOwner=1
           Owner_FourthAboveOwner=0
           Owner_SecondOwner=0
           Owner_TestDriveCar=0
           Owner_ThirdOwner=0
    elif(Owner_FirstOwner=='second'):
           Owner_FirstOwner=0
           Owner_FourthAboveOwner=0
           Owner_SecondOwner=1
           Owner_TestDriveCar=0
           Owner_ThirdOwner=0
    elif(Owner_FirstOwner=='third'):
           Owner_FirstOwner=0
           Owner_FourthAboveOwner=0
```

```
                Owner_SecondOwner=0
                Owner_TestDriveCar=0
                Owner_ThirdOwner=1
        elif(Owner_FirstOwner=='forth'):
                Owner_FirstOwner=0
                Owner_FourthAboveOwner=1
                Owner_SecondOwner=0
                Owner_TestDriveCar=0
                Owner_ThirdOwner=0
        else:
                Owner_FirstOwner=0
                Owner_FourthAboveOwner=0
                Owner_SecondOwner=0
                Owner_TestDriveCar=1
                Owner_ThirdOwner=0
      Seat_high=request.form['Seat_high']
        if(Seat_high=='high'):
            Seat_high=1
            Seat_middle=0
            Seat_normal=0
        elif(Seat_high=='medium'):
            Seat_high=0
            Seat_middle=1
            Seat_normal=0
        else:
            Seat_high=0
            Seat_middle=0
            Seat_normal=1
        prediction=model.predict([[year,km_driven,AvgMileage,EngineCC,MaxPower,Fu
el_CNG,Fuel_Diesel,Fuel_LPG,Fuel_Petrol,SellerType_Dealer,SellerType_Individual,S
ellerType_TrustmarkDealer,Transmission_Automatic,Transmission_Manual,Owner_FirstO
wner,Owner_FourthAboveOwner,Owner_SecondOwner,Owner_TestDriveCar,Owner_ThirdOwner
,Seat_high,Seat_middle,Seat_normal]])
        output=round(prediction[0],2)
        if output<0:
            return render_template('index.html',prediction_texts="Sorry there are
no cars with the required features")
        else:
            return render_template('index.html',prediction_text="You Selling
price of car is: {}".format(output))
    else:
        return render_template('index.html')
if __name__=="__main__":
    app.run(debug=True)
```

Final Output: