SPRINT 3

Team ID	PNT2022TMID22229
Project Title	Car Resale Value Prediction
Team Members	Harish.K Karanraj Faseel Ahamed syed Afzal Rahuman

Step1: The Mapper & LabelEncoder concepts are used here for splitting the data.

jupyter Car resale value prediction1 Last Checkpoint: 11/09/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [17]:
```
mapper = {}
for i in labels:
    mapper[i] = LabelEncoder()
    mapper[i].fit(df[i])
    tr = mapper[i].transform(df[i])
    np.save(str('classes'+i+'.npy'), mapper[i].classes_)
    print(i,":",mapper[i])
    df.loc[:,i + '_labels'] = pd.Series(tr, index=df.index)
```

```
gearbox : LabelEncoder()
notRepairedDamage : LabelEncoder()
model : LabelEncoder()
brand : LabelEncoder()
fuelType : LabelEncoder()
vehicleType : LabelEncoder()
```

In [19]:
```
labeled = df[['price'
              ,'yearOfRegistration'
              ,'powerPS'
              ,'kilometer'
              ,'monthOfRegistration'
              ]
             +[x+"_labels" for x in labels]]
```

In [20]:
```
print(labeled.columns)
```

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

Step2: Model Building – After the splitting of data using datascience variants, the model is developed and gets saved in a .sav format.

jupyter Car resale value prediction1 Last Checkpoint: 11/09/2022 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'model_labels', 'brand_labels', 'fuelType_labels',
       'vehicleType_labels'],
      dtype='object')
```

In [27]:
```
Y = labeled.iloc[:,0].values
X = labeled.iloc[:,1:].values
```

In [22]:
```
Y = Y.reshape(-1,1)
```

In [28]:
```
from sklearn.model_selection import cross_val_score, train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state = 3)
```

In [29]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
regressor = RandomForestRegressor(n_estimators=1000,max_depth=10,random_state=34)
regressor.fit(X_train, np.ravel(Y_train,order='C'))
```

Out[29]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000, random_state=34)
```

In [30]:
```
y_pred = regressor.predict(X_test)
print(r2_score(Y_test,y_pred))
```

```
0.834527626497731
```

In [34]:
```
#saving the model for future use.
filename = 'resale_model.sav'
pickle.dump(regressor, open(filename, 'wb'))
```

Step3: Flask app is being developed here.

Spyder (Python 3.9)

C:\Users\hk2k0\Car Resale Value Prediction\carapp.py

```
#Load the model and initialize Flask app
app = Flask(__name__)
filename = 'resale_model.sav'
model_rand = pickle.load(open(filename,'rb'))

@app.route('/')
def index():
    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('modeltype')
    brand = request.form.get('brand')
    fuelType = request.form.get('fuel')
    vehicletype = request.form.get('vehicletype')
    new_row = {'yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer':kms,
               'monthOfRegistration':regmonth, 'gearbox':gearbox, 'notRepairedDamage':damage,
               'model':model, 'brand':brand, 'fuelType':fuelType,
               'vehicleType':vehicletype}
    print(new_row)
    new_df =pd.DataFrame(columns = ['vehicleType', 'yearOfRegistration', 'gearbox',
                                    'powerPS', 'model', 'kilometer', 'monthOfRegistration', 'fuelType',
                                    'brand','notRepairedDamage'])
    new_df = new_df.append(new_row,ignore_index = True)
    labels = ['gearbox', 'notRepairedDamage', 'model', 'fuelType', 'vehicleType']
    mapper = {}
    for i in labels:
        mapper[i] = LabelEncoder()
        mapper[i].classes_ = np.load(str('classes' +i+'.npy'))
```

Console 1/A

```
carapp.py', wdir='C:/Users/hk2k0/
Car Resale Value Prediction')

In [3]: runfile('C:/Users/hk2k0/
Car Resale Value Prediction/
carapp.py', wdir='C:/Users/hk2k0/
Car Resale Value Prediction')

In [4]: runcell(0, 'C:/Users/
hk2k0/Car Resale Value Prediction/
carapp.py')

In [5]: runfile('C:/Users/hk2k0/
Car Resale Value Prediction/
carapp.py', wdir='C:/Users/hk2k0/
Car Resale Value Prediction')

In [6]:
```