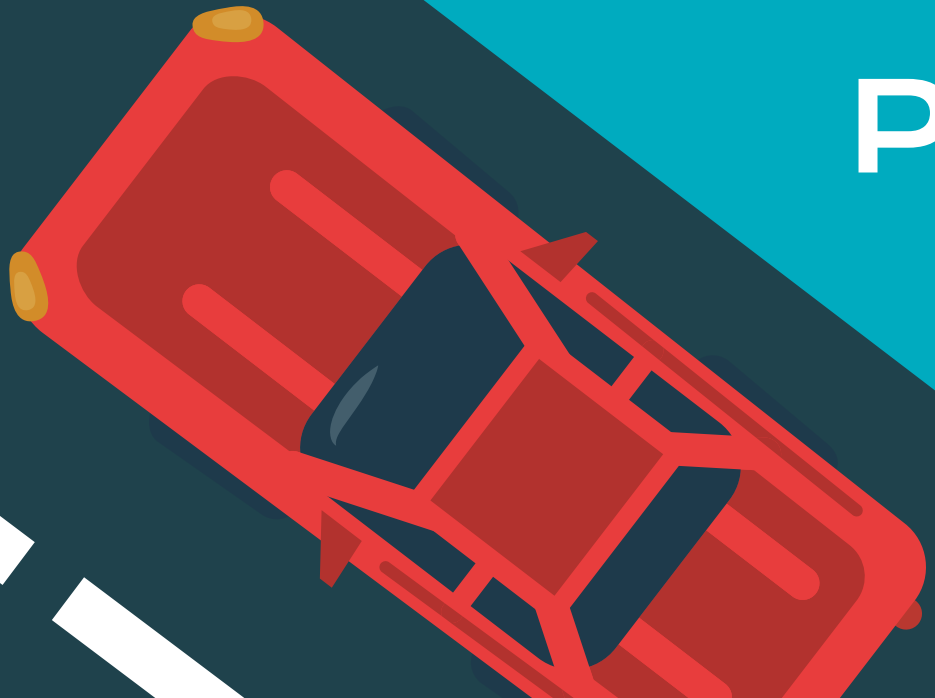
SC1015 Mini Project



Z136 Team 4
Chong Choy Jun
Chew Di Heng
Choh Lit Han Owen

Introduction

Problem Definition

- As COE in Singapore have been rising rapidly, how can one choose the most appropriate car model that will be value for money ?
- Are we able to predict car price through a subset of the car features alone? Does the engine size affects the price more than model?

Dataset

- Ford car price dataset from kaggle

Data Cleaning and Preparation

Before

Setup : Import the Dataset

```
In [9]:
cardata = pd.read_csv('ford.csv')
cardata
```

Out[9]:

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	Fiesta	2017	12000	Automatic	15944	Petrol	150	57.7	1.0
1	Focus	2018	14000	Manual	9083	Petrol	150	57.7	1.0
2	Focus	2017	13000	Manual	12456	Petrol	150	57.7	1.0
3	Fiesta	2019	17500	Manual	10460	Petrol	145	40.3	1.5
4	Fiesta	2019	16500	Automatic	1482	Petrol	145	48.7	1.0
...	...	...	...	...	...	...	...	...	...
17961	B-MAX	2017	8999	Manual	16700	Petrol	150	47.1	1.4
17962	B-MAX	2014	7499	Manual	40700	Petrol	30	57.7	1.0
17963	Focus	2015	9999	Manual	7010	Diesel	20	67.3	1.6
17964	KA	2018	8299	Manual	5007	Petrol	145	57.7	1.2
17965	Focus	2015	8299	Manual	5007	Petrol	22	57.7	1.0

17966 rows × 9 columns

Drop

Tax

Miles Per Gallon

After

Out[15]:

	model	year	price	transmission	mileage	fuelType	engineSize
0	Fiesta	2017	12000	Automatic	15944	Petrol	1.0
1	Focus	2018	14000	Manual	9083	Petrol	1.0
2	Focus	2017	13000	Manual	12456	Petrol	1.0
3	Fiesta	2019	17500	Manual	10460	Petrol	1.5
4	Fiesta	2019	16500	Automatic	1482	Petrol	1.0
...	...	...	...	...	...	...	...
17961	B-MAX	2017	8999	Manual	16700	Petrol	1.4
17962	B-MAX	2014	7499	Manual	40700	Petrol	1.0
17963	Focus	2015	9999	Manual	7010	Diesel	1.6
17964	KA	2018	8299	Manual	5007	Petrol	1.2
17965	Focus	2015	8299	Manual	5007	Petrol	1.0

17966 rows × 7 columns

```
In [23]:
cardata_subset['year'].astype('object')
cardata_subset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17966 entries, 0 to 17965
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   model         17966 non-null  object
 1   year          17966 non-null  object
 2   price         17966 non-null  int64
 3   transmission  17966 non-null  object
 4   mileage       17966 non-null  int64
 5   fuelType      17966 non-null  object
 6   engineSize    17966 non-null  float64
dtypes: float64(1), int64(2), object(4)
memory usage: 982.6+ KB
```

We change “Year” to object type

```
car_numeric_data = cardata_subset.select_dtypes('number')
car_cat_data = cardata_subset.select_dtypes('object')
print(car_numeric_data.columns)
print(car_cat_data.columns)
```

```
Index(['price', 'mileage', 'engineSize'], dtype='object')
Index(['model', 'year', 'transmission', 'fuelType'], dtype='object')
```

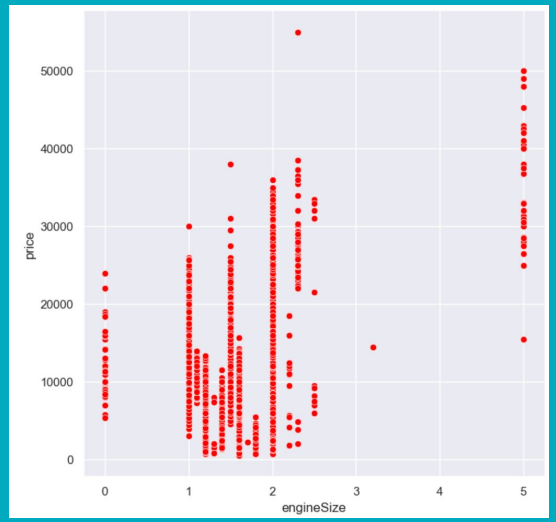
Splitting data set into numerical data and categorical data

Exploratory Data Analysis - Numerical data

Correlation Matrix

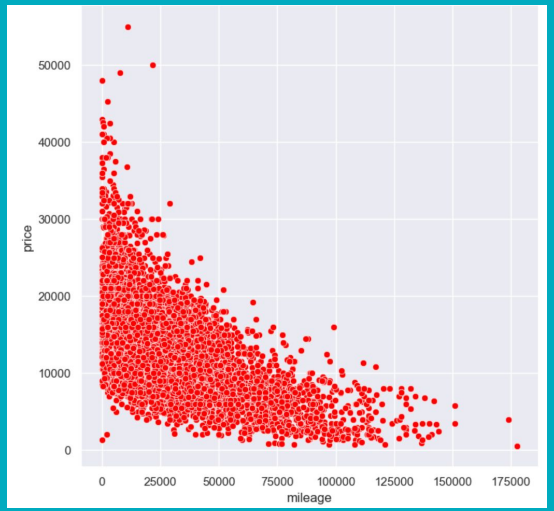
Mileage has a stronger correlation with price

Engine Size



Size “5” > 20000

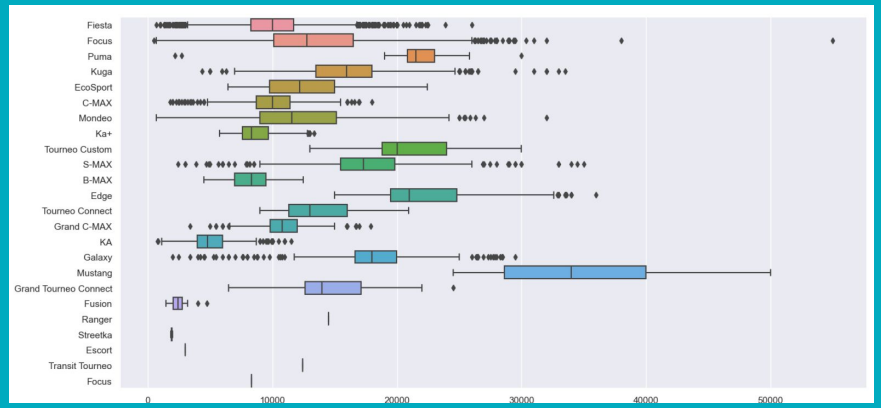
Mileage



Mileage < 25000 attracts high price

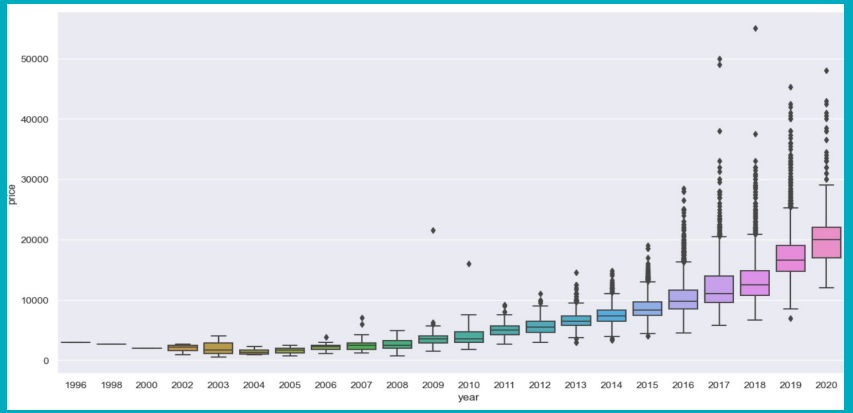
Exploratory Data Analysis - Categorical Data

Model



Model “Focus” have outliers reaching > 50000 in price

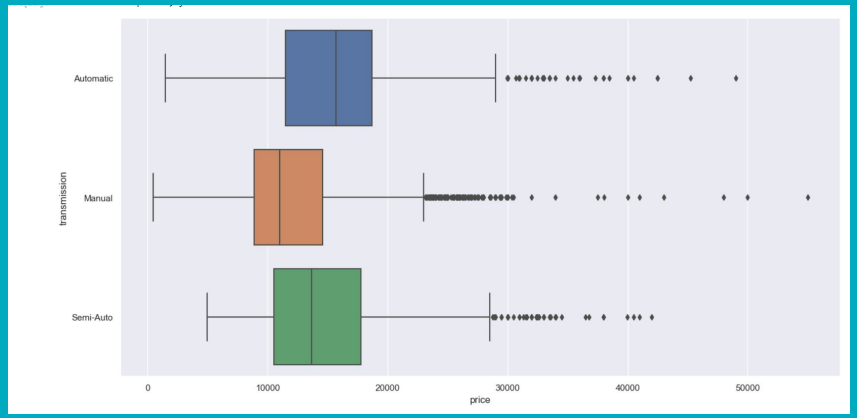
Year



Newer car model in general have higher selling price

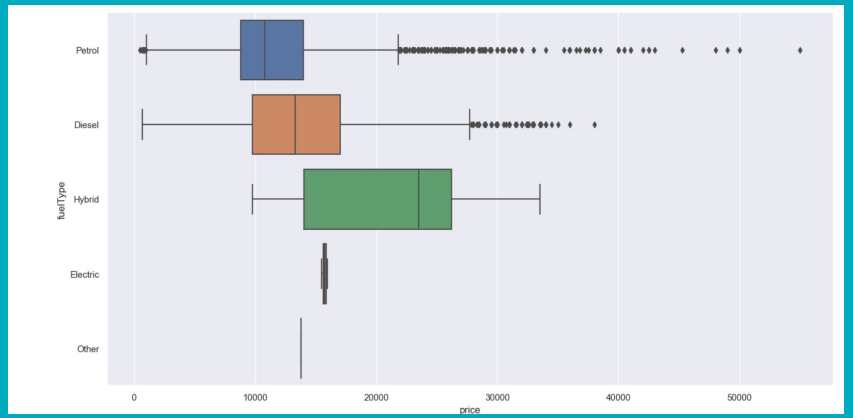
Exploratory Data Analysis - Categorical Data part 2

Transmission



Manual car attracts higher price

Fuel Type



Petrol car attract higher price

Why Random Forest

Random Forest

Machine learning technique

Components

Consists of

Multiple decision trees

Function

Perform regression

Advantage

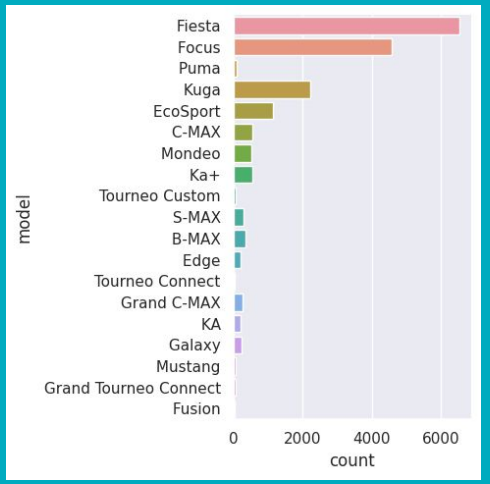
Reduces

Risk of bias and overfitting

Maintain

Accuracy

Model: Random Forest



How we applied it

01 — **OHE**

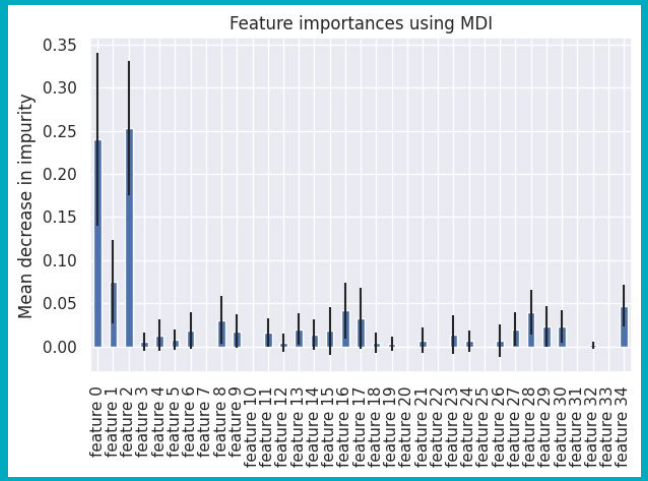
One Hot Encoded (OHE) categorical variables

02 — **Build Model**

Fit the model on the dataset

03 — **SMOTE**

Used SMOTE to rebalance the dataset



Results of the Model

04 — **All features**

R^2 score of 0.719

05 — **Omit features**

R^2 score of 0.681

06 — **After SMOTE**

R^2 score of 0.485

Key Learning Points: Random Forest

1 **Flexible Model**

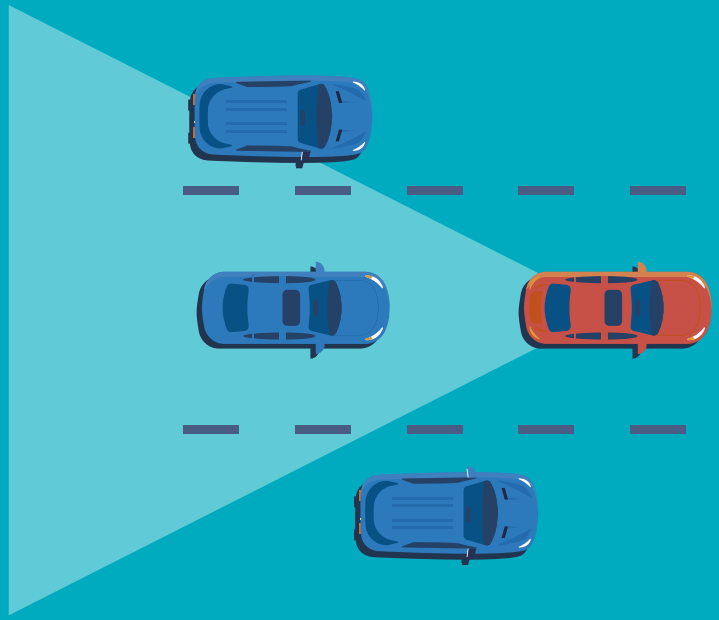
Applicable to both regression and classification problems

2 **Advantage**

Avoids overfitting

3 **SMOTE**

Good for increasing minority class but may decrease performance of model



Model: Decision Tree Classifier

Easy to interpret and visualize

Interpretable

Versatile

Handles both categorical and numerical data

Can be used for feature selection

Feature Selection

Compatible with others

Can be combined with other algorithms (K-Fold)

Why K-Fold

Provides a more accurate estimate of the model's performance

Enables more efficient use of data

Reduces the risk of overfitting

Help with model selection

How we applied it?

01

OHE

One Hot Encoded (OHE) categorical variables

02

K-Fold

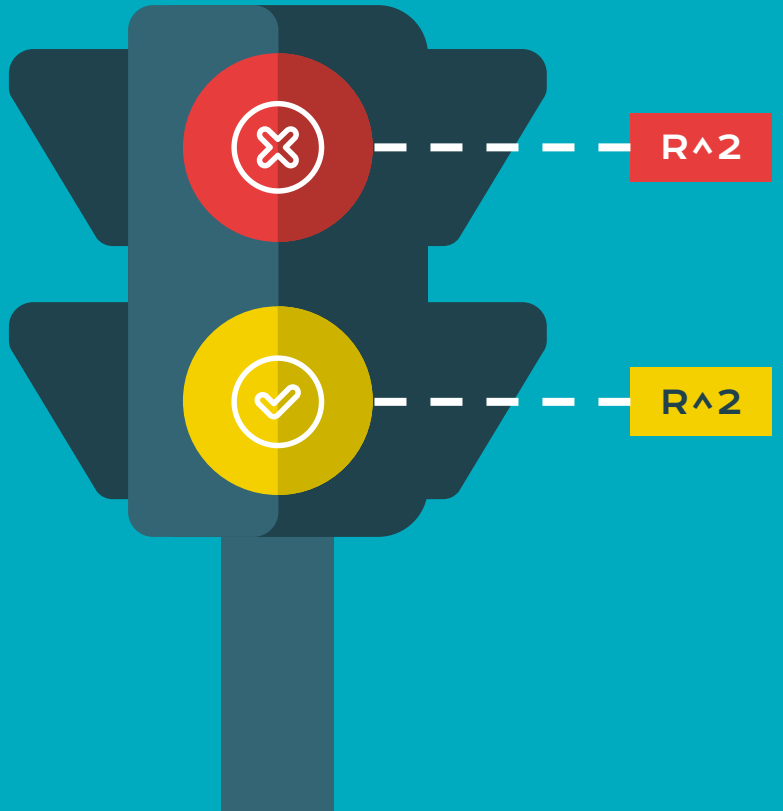
Split the dataset into 10 equally sized folds

03

Build Model

Fit model on to dataset

Results of the model



Decision Tree

R^2 score of 0.636

DT + K-Fold

R^2 score of 0.607

Key Learning Points: Decision Tree Classifier

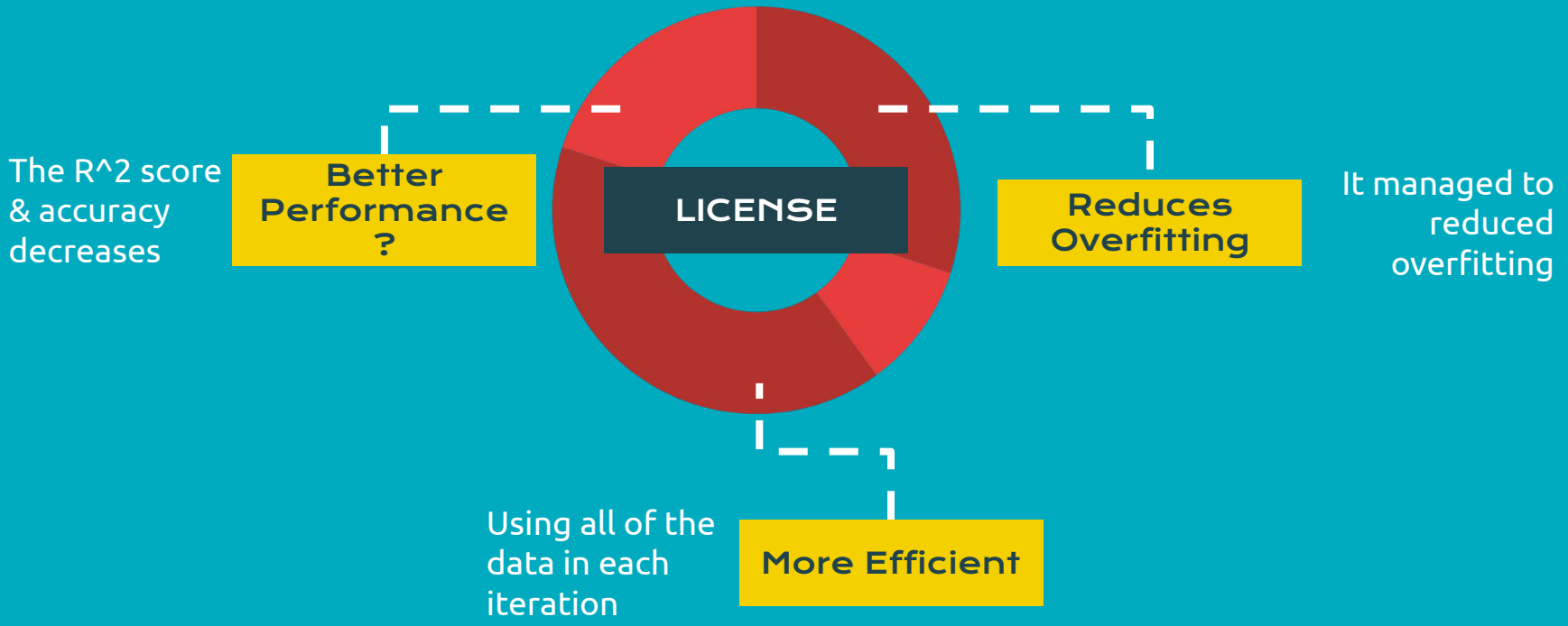
Pros

- Popular and powerful
- Tree-like model
- Versatile

Cons

- Prone to overfitting
- Need to reduces overfitting
- Is not suitable for our dataset

Key Learning Points: K-Fold

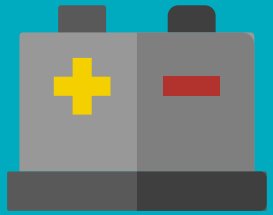


Model: Gradient Boosting Regressor

01

Flexibility

Can be used for a variety of regression tasks



02

High accuracy

Known for producing highly accurate predictions

How did we applied the model?

1 **OHE**

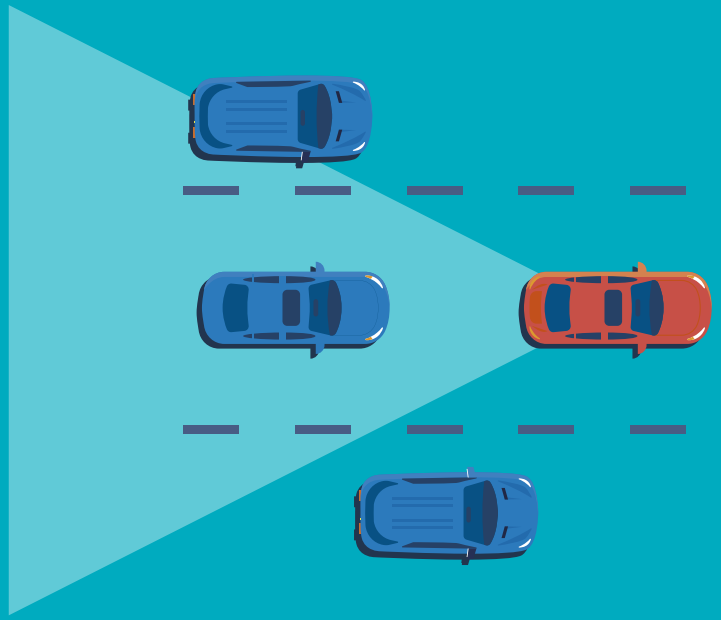
One Hot Encoded (OHE) categorical variables

2 **Split Dataset**

Split into train and test

3 **Build Model**

Fit model on to dataset



Model: Gradient Boosting Regressor

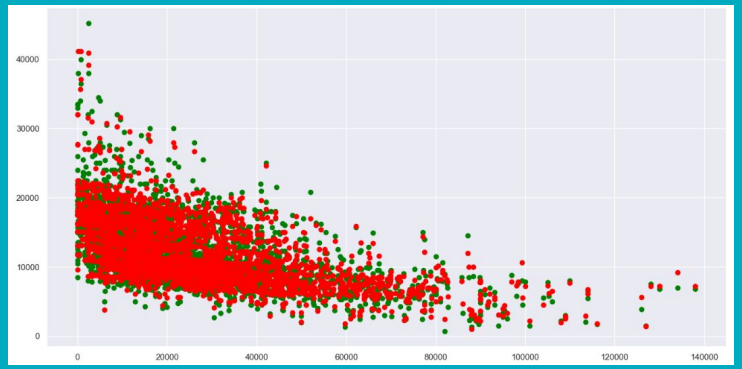
Train Dataset

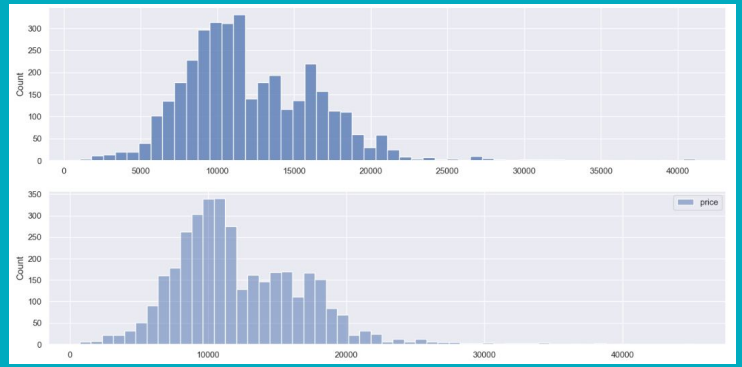
- Accuracy of 0.91838
- R^2 score of 0.91838

Results of the model

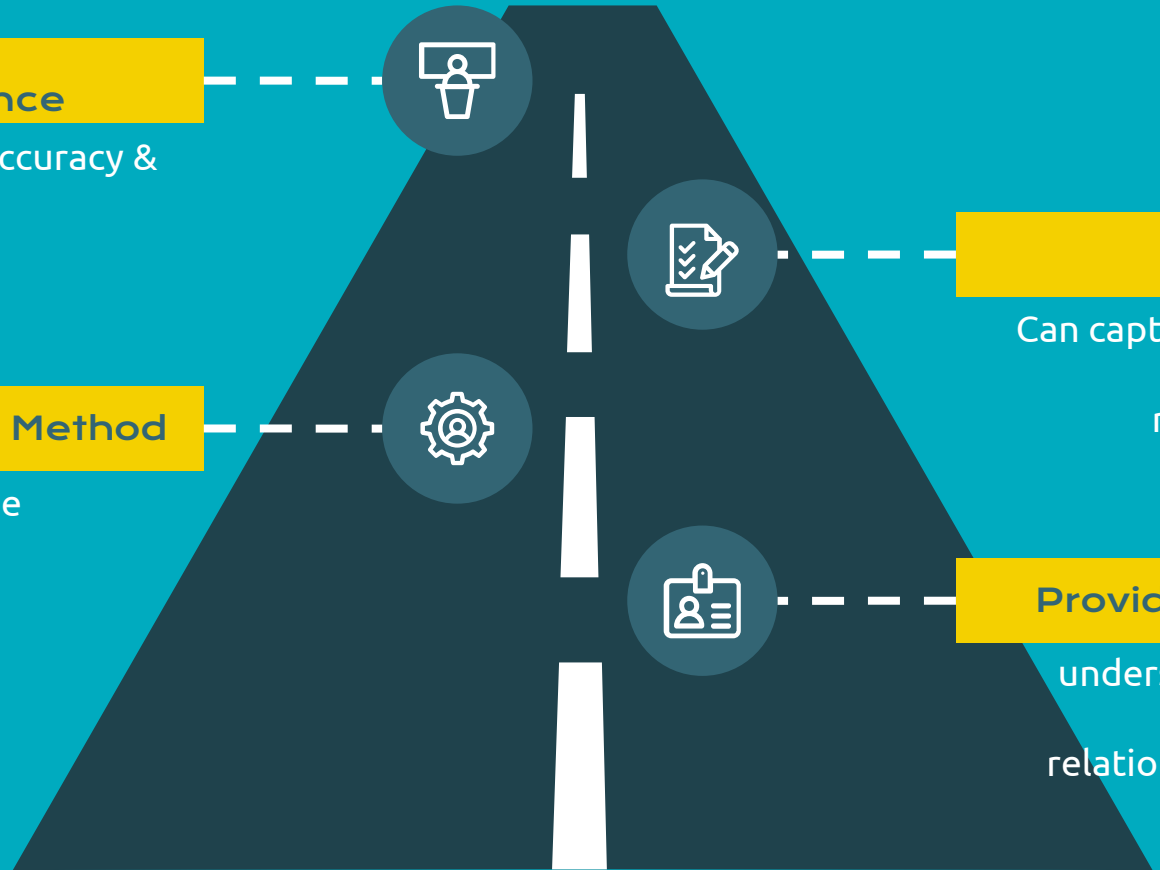
Test Dataset

- Accuracy of 0.92068
- R^2 score of 0.92068





Key Learning Points: Gradient Boosting Regressor



Better Performance

Increase in accuracy & R^2 score

Ensemble Method

Combines the predictions

Flexible

Can capture complex non-linear relationships

Provide Insights

understanding the underlying relationships in the data.

Gradient Boosting Regressor: Importance chart

1

Accuracy for Engine Size

```
Goodness of Fit of Model          Train Dataset
Accuracy for Engine Size          : 0.4116587761551632

Goodness of Fit of Model          Test Dataset
Accuracy for Engine Size          : 0.40256876108154593
```

2

Accuracy for Mileage

```
Goodness of Fit of Model          Train Dataset
Accuracy for Mileage      : 0.3468246383032435

Goodness of Fit of Model          Test Dataset
Accuracy for Mileage      : 0.3454200921503532
```

3

Accuracy for Year

```
Goodness of Fit of Model          Train Dataset
Accuracy for year         : 0.47408399711409654

Goodness of Fit of Model          Test Dataset
Accuracy for year         : 0.4895054746401831
```

4

Accuracy for all variables

```
Goodness of Fit of Model          Train Dataset
Accuracy          : 0.9183832541250874

Goodness of Fit of Model          Test Dataset
Accuracy          : 0.9206849808935842
```

Conclusion

Outcome of our project

01

Outcome

Gradient boosting regressor is the best performing model

02

Insights

Models can provide information about the features affecting price

03

Recommendation

“Out of sample testing”