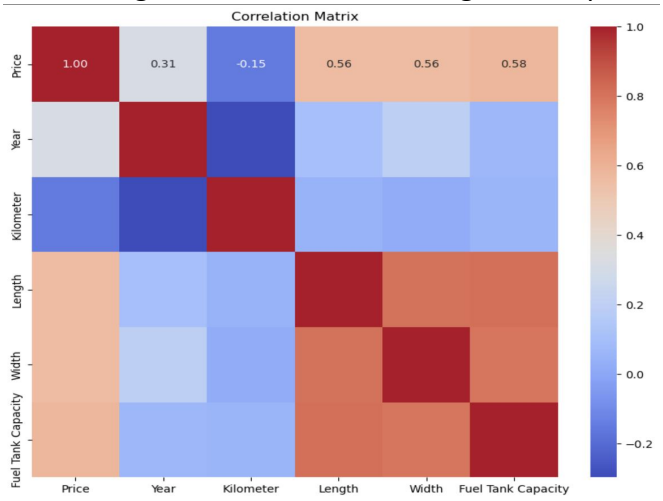
Car Price Prediction Report

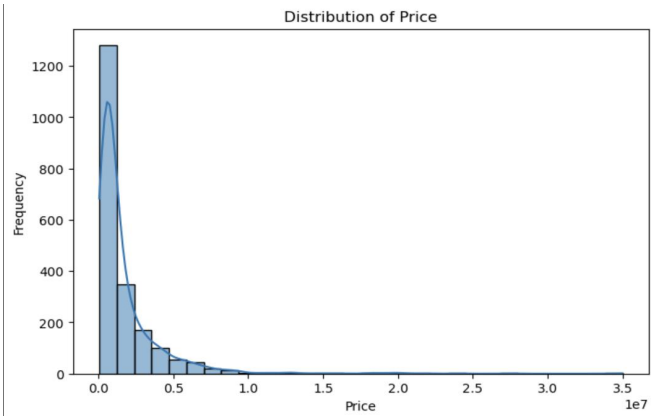
1. Motivation

The goal of this project is to predict second-hand car prices using machine learning techniques. Unlike a new car, whose prices are pretty much determined, used car prices are affected by a lot of factors such as car age, kilometers that have been drove, engine power, number of previous users, etc. Therefore, have a accurate price prediction model is beneficial to both the used car sellers and buyers, eliminate information asymmetry and contribute to market efficiency.

2. EDA and Data Cleaning

The raw data-set include numerical and categorical features: manufacturing year, kilometer, Fuel Type, Transmission, Engine, Max Power, Max Torque, Length, Width, Fuel tank capacity.There are 2000 rows of data which satisfy for machine learning data requirement. Half of all the features like Engine, Max power have almost 100 missing values and outliers. Based on the correlation analysis, most of the features have strong correlation with the target value:price, which is good for prediction.





However most of the variables such as price have highly skewed distribution, which need to be log transformed in data cleaning to make it normal.

For data cleaning, I did the following:

- Handling missing values
- Remove duplicate values
- Handling outliers
- Log transform high skew features

For the feature “Engine”, I extract the numerical value out, eg. From “1278 cc” to “1278”, I did the same cleaning for the features “Max Power” and “Max Torque”.

3. Feature Engineering

In terms of feature selection, I remove highly irrelevant features such as seller type, car location and seller type, since they only add noise to the model and such irrelevant features will hurt the performance of the model. I also remove car age because it’s highly relevant to car manufacture year will lead to multicollinearity issue. In terms of engineering, I first did standardization to numerical features since a lot of models such as Elastic-net are highly sensitive to scale. With standardization, i normalize numerical features to ensure large value features such as prices and kilometer does not dominate small value feature. I also did feature encoding. I used One-Hot Encoding to all categorical features because the categorical features in my model do not have order. So I applied encoding to my categorical values to convert them from labels to numerical values for machine learning models to process. I also did some imputation work including filling missing numerical values with mean, filling missing categorical values with the most frequent categorical.

4. Modelling

In this part, I used prepossessed data to evaluate two regression models: GLM and LGBM models, aim to compare their prediction power. For GLM model, it takes the training data, learn and figuring out the best possible formula to predict the price, and then use that formula on test data. LGBM is another story. Gradient boosting is like a bunch of decision trees working together, they learn the data incrementally branch by branch, and correct the error. LGBM takes my train data and build these decision trees and eventually combine all the trees to derive a final prediction.

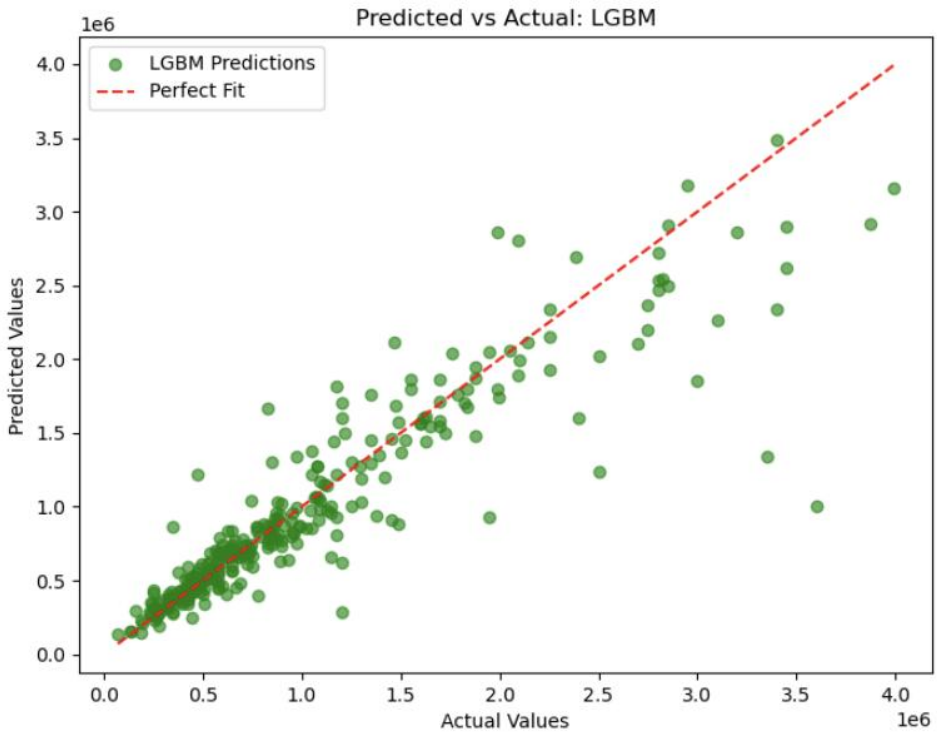
Hyper parameter tuning is the next step after modelling. It adjust all the settings in both models trying to find the optimal hyper parameters including alpha, l1 ratio, learning rate, n estimators which helps to make the most accurate prediction. With hyper parameter tuning, we get out best prediction models.

5. Modelling

The performance of my models is decent.

```
GLM Evaluation:
Mean Squared Error (MSE): 193527210790.20114
Mean Absolute Error (MAE): 235646.1180878465
Mean Absolute Percentage Error (MAPE): 21.79%
R-squared (R2): 0.67820366575838 99

LGBM Evaluation:
Mean Squared Error (MSE): 104901617598.3689
Mean Absolute Error (MAE): 169757.1399699657
Mean Absolute Percentage Error (MAPE): 16.50%
R-squared (R2): 0.8255699761220373
```



The R-squared of GLM model is roughly 70% and 83% for LGBM, which indicate both models explain the variation pretty well while LGBM did a much better job. This might due to the data is more complex than those linear regression can capture. Even though the MAE is quite huge for both model, it's acceptable consider the scale of the target variable(price) is original huge in scale. The MAPE which rules out the scaling issue indicate the prediction models has good performance since the value is around 20% for both models, 16.5% for LGBM Model. From all evaluation metrics we can conclude that LGBM did a significantly better prediction job for the project, this

might because the raw data set is quite complex and there is no much linear relationship GLM can capture, LGBM model is more suitable for prediction model using this complex data set.

There is still room for improvement. The performance will be even better if more relevant features are included in the raw data, also more steps can be included in feature engineering such as creating more interaction terms between categorical features. More advanced algorithm can be used to capture more complex relationships such as Random Forest or Neural Networks.