Used Car Price Prediction

Cars4U

‘Femi Bolarinwa
f3bolarinwa@yahoo.com
June 2023

Background

- Used cars sales is ***trending upwards*** and outpacing new car sales in India
- In 2018 - 19, **4million** used cars sold vs **3.6million** new cars
- ***Cars4U*** is a **growing** tech start-up seeking to established itself in this market

https://global-uploads.webflow.com/6292b2d3cd8d72652bda25a3/6293b5e4fa42ae93c710ac2b_growth-chart2-small.jpeg

Goal

- Several ***factors*** affect price of used car, e.g. mileage, brand, model, age, location, etc

- ***Cars4U*** needs an effective pricing model to **automate** the process of price-setting

https://www.ahsmp.com/wp-content/uploads/2019/10/goal-setting-feature-image.jpg

Key Steps

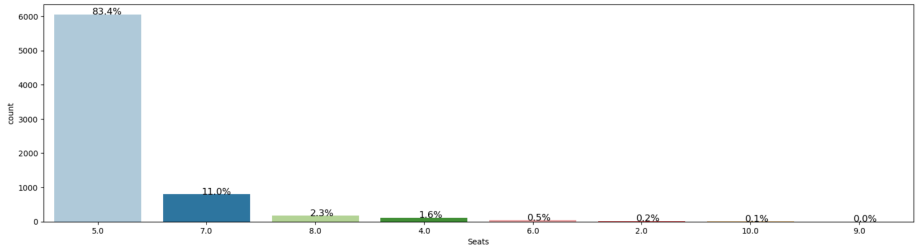
- Exploratory Data Analysis
- Build/train several machine learning models for price prediction
- Evaluate/test performance of models
- Select best model
- Consider pros and cons of best model
- Derive insights and business recommendations

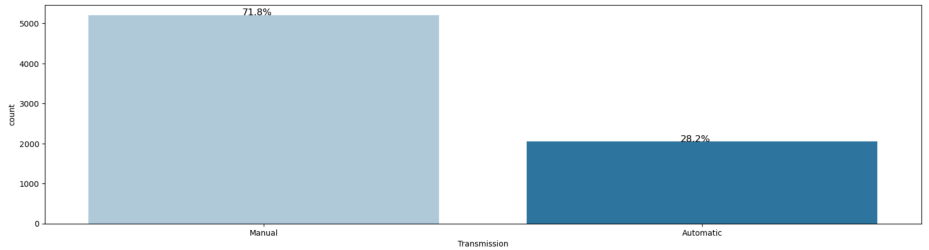
https://cdn.shopify.com/s/files/1/0070/7032/files/AMgoal-setting_HEADER.jpg?v=1579623952

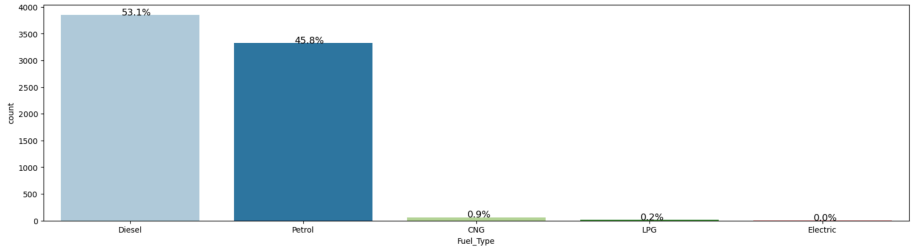
Exploratory Data Analysis

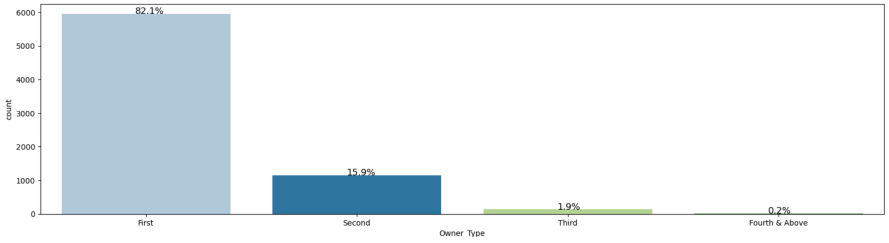
Typical Used Car In Demand

- 5 seats
- Manual transmission
- Diesel or petrol engine
- First ownership



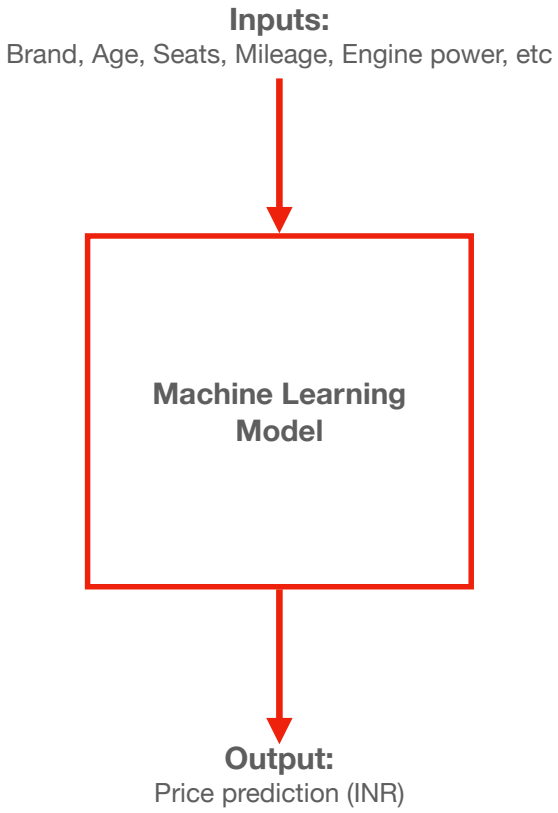






Problem Definition

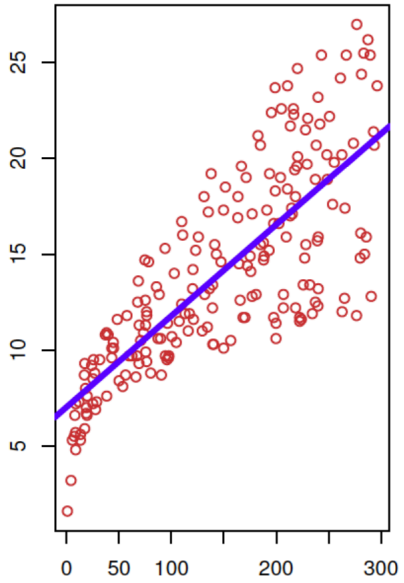
- Target variable (price) is a ***numerical continuous*** variable
- Our pricing model will be a ***regression model***
- Model ***Inputs***: factors and features of car
  - Brand, Age, Seats
  - Mileage, Engine power and size
  - Location, Fuel type, etc.
- Model ***Output:*** Price prediction (INR, 100000)

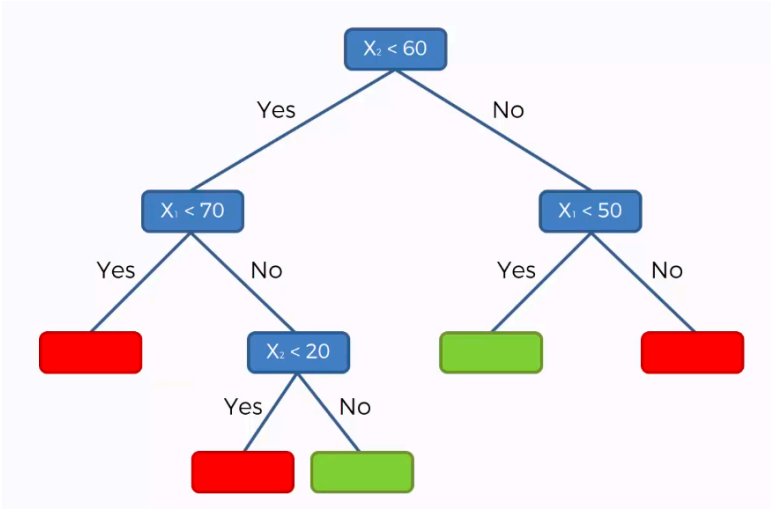


Solution Design

Models

- ***Linear*** Regression:
  - Ridge & Lasso Regularization
- **XGBoost** Regression:
- ***Tree-based*** methods:
  - Decision Tree & Random Forest

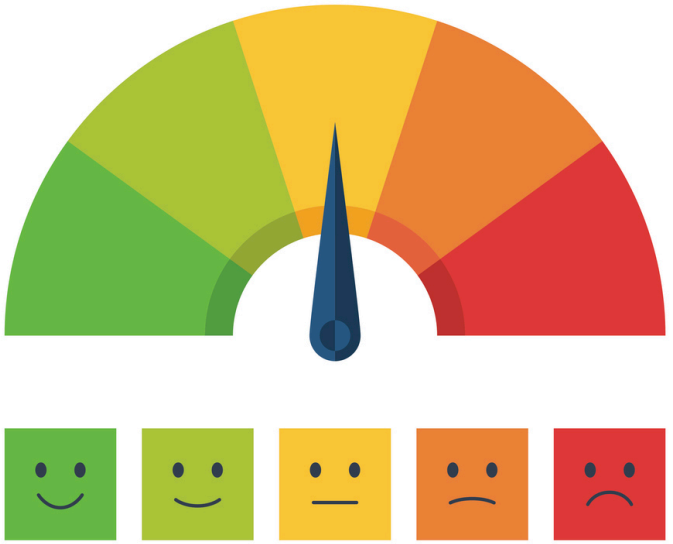




Solution Design

Performance Metrics

- ***Metrics:***
  - r2-score
  - RMSE
  - MAE
- ***Interpretation***:
  - Example: **r2** = *90%*. **RMSE** = *100,000*. **MAE** = *80,000*
  - You can expect the model 90% of the time to correctly predict price with RMSE of 100,000INR and MAE of 80,000INR on ***average***



https://www.apsfulfillment.com/wp-content/uploads/2017/05/APS-Fulfillment-Inc2.jpg

Solution Design

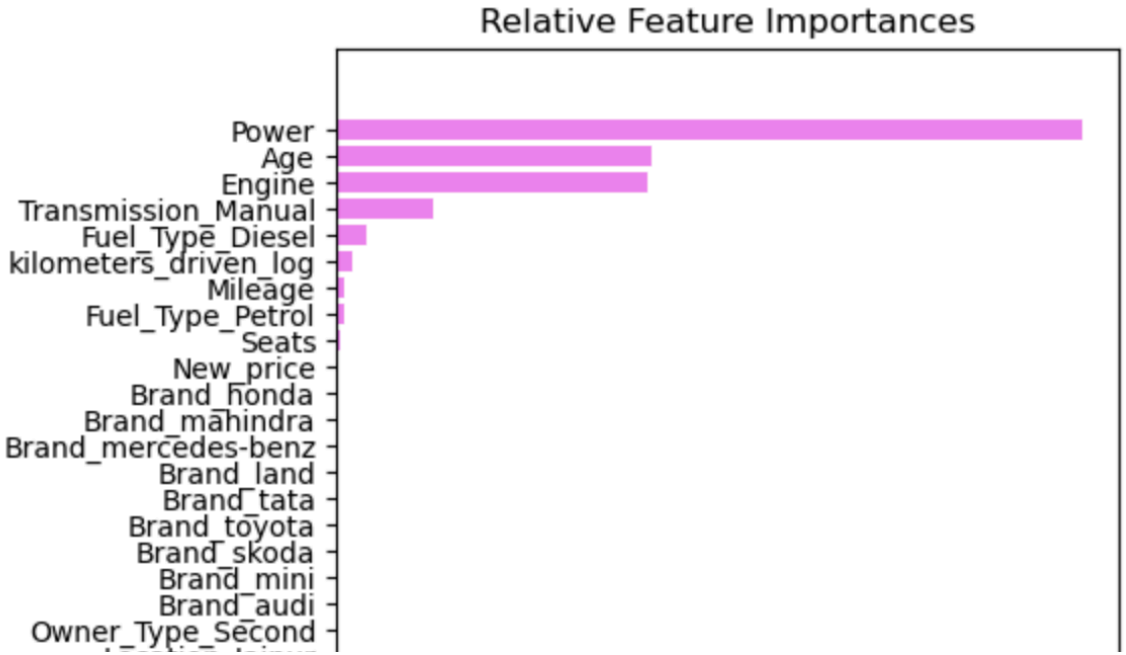
Model Performances on unseen test data

	RMSE	MAE	R-squared	Adj. R-squared
Linear Regressor(OLS)	126387.706546	119762.714162	0.928993	0.926761
Ridge Regressor (Default)	126454.400493	119779.039182	0.928673	0.926473
Lasso Regressor (Default)	158273.841566	142749.712672	0.727034	0.718616
Lasso Regressor (Tuned)	126391.779174	119761.198952	0.928973	0.926783
Decision Tree Regressor (Default)	135726.171149	122863.343147	0.879185	0.875459
Decision Tree Regressor (Tuned)	147100.024235	134344.377786	0.807144	0.801197
Random Forest Regressor (Default)	123408.968062	115913.847907	0.942720	0.940953
Random Forest Regressor (Tuned)	139081.869903	128960.096735	0.859093	0.854748
XGBoost Regressor (Default)	120140.312338	113911.296265	0.956407	0.955063

Solution Design

Model Selection

- Best model: **XGBoost/Random Forest**
- Pros
  - Great performance
- Cons
  - Interpretability



Insight and Recommendation for Implementation

- Overwhelming **majority** of cars have 5 seats, automatic transmission, first ownership, diesel or petrol engine and are manufactured after 2010. This is a useful insight for the business into the typical car in demand.
- Different ML models appear to suggest and agree that the most **relevant features** for pricing prediction are Engine Power, Age, Engine size, Transmission type, fuel type, kilometres driven.
- To **implement** this model for price prediction, 'kilometer_drive' feature has to be log-**transformed**, the categorical variables have to be **hot-encoded.** The calculated prediction would be log(price). The price prediction in INR (100,000) can be obtained by taking the exponent of log(price)
- Greater model performance can be achieved by extensive hyper-parameter **tuning** of models, and by trying **other ML** techniques (e.g. AdaBoost, SVM, KNN, etc).

Executive Summary

- Used cars sales is ***trending upwards***
- In 2018 - 19, **4million** used cars sold vs **3.6million** new cars
- ***Cars4U*** is a growing tech start-up seeking to established itself in this market
- An effective pricing model would help **automate** the process of price-setting
- A **XGBoost/Random Forest** ML model gives a >**94%** chance of correctly predicting price within 114,000INR error

https://www.google.com/url?sa=i&url=https%3A%2F%2Fcanadianautodealer.ca%2F2022%2F09%2Fanother-price-decline-week-in-the-used-car-market-supply-improving%2F&psig=AOvVaw3GJP7PF2KyfW673IFHPG_c&ust=1686938125201000&source=images&cd=vfe&ved=0CBEQjRxqFwoTCOjV5K_sxf8CFQAAAAAdAAAAABAE

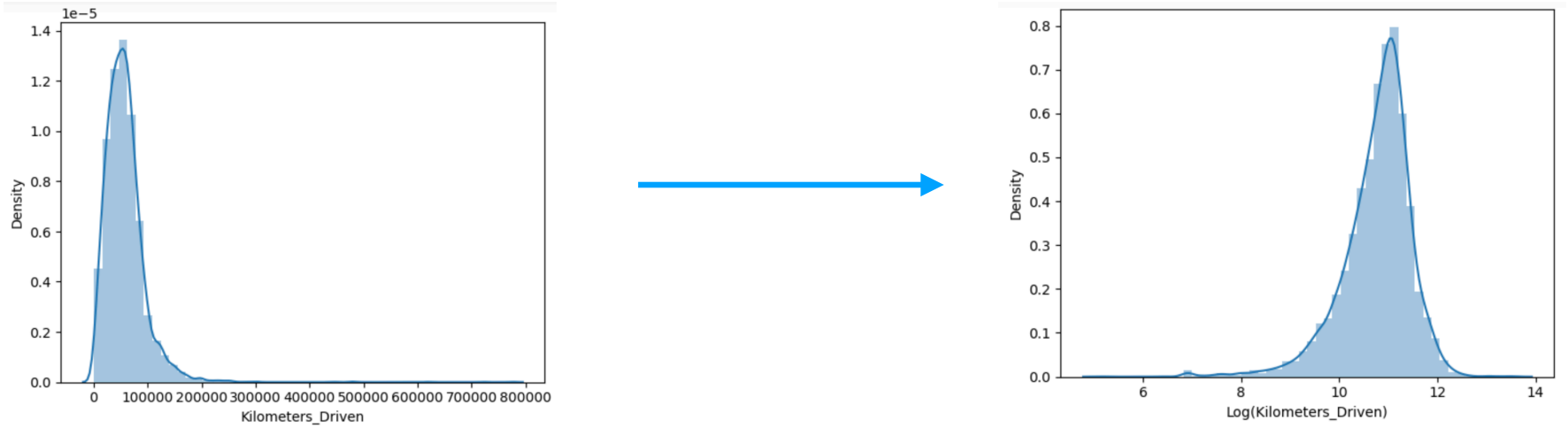
End

Appendix

Python Libraries

```
#Libraries for data visualization
import matplotlib.pyplot as plt
import seaborn as sns

#To suppress warning
import warnings
warnings.filterwarnings("ignore")

#libraries for data pre-processing and tuning
from sklearn.preprocessing import MinMaxScaler, StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score, KFold, GridSearchCV

#Libraries for linear modeling
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet

#Libraries for tree-based and ensemble modeling
from sklearn import tree
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,BaggingRegressor, AdaBoostRegressor, GradientBoostingRegressor

#Library for Extreme Gradient Boost modeling
from xgboost import XGBRegressor

# Libraries for model performance evaluation
from sklearn.metrics import make_scorer,mean_squared_error, r2_score, mean_absolute_error

#StatsModel Libraries for modeling
import statsmodels.api as sm
import statsmodels.stats.api as sms
from statsmodels.tools.tools import add_constant
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.sm_exceptions import ConvergenceWarning
warnings.simplefilter("ignore", ConvergenceWarning)

#To perform goldfeldquandt test for homoscedasticity/heteroscedasticity
from statsmodels.stats.diagnostic import het_white
from statsmodels.compat import lzip
```

Transformation of ‘Kilometers_Driven’



Performance of Selected XGBoost Model

K-Fold Cross-validation of XGBoost Regressor Model

```
cv_Score11 = cross_val_score(xgb, X_train_original, y_train_original["Price_log"], cv = 10)
cv_Score12 = cross_val_score(xgb, X_train_original, y_train_original["Price_log"], cv = 10,
                             scoring = 'neg_mean_squared_error')
print("RSquared: %0.3f (+/- %0.3f)" % (cv_Score11.mean(), cv_Score11.std() * 2))
print("Mean Squared Error: %0.3f (+/- %0.3f)" % (-1*cv_Score12.mean(), cv_Score12.std() * 2))
```

```
RSquared: 0.943 (+/- 0.023)
Mean Squared Error: 0.044 (+/- 0.022)
```

Model performance on training data

```
xgb_reg_perf = model_performance_regression(xgb, X_train_original, y_train_original["Price_log"])
xgb_reg_perf
```

	RMSE	MAE	R-squared	Adj. R-squared
0	109169.598	106745.799	0.990	0.990

Model performance on unseen test data

```
xgb_reg_perf_test = model_performance_regression(xgb, X_test_original, y_test_original["Price_log"])
xgb_reg_perf_test
```

	RMSE	MAE	R-squared	Adj. R-squared
0	120140.312	113911.296	0.956	0.955

Performance of Selected Random Forest Model

Model performance on training data

```
rf_regressor_perf = model_performance_regression(rf_regressor,X_train_original, y_train_original["Price_log"])
rf_regressor_perf
```

	RMSE	MAE	R-squared	Adj. R-squared
0	108881.426	105769.767	0.990	0.990

Model performance on unseen test data

```
rf_regressor_perf_test = model_performance_regression(rf_regressor,X_test_original, y_test_original["Price_log"])
rf_regressor_perf_test
```

	RMSE	MAE	R-squared	Adj. R-squared
0	123408.968	115913.848	0.943	0.941