CAR PRICE PREDICTION

Team Presentation

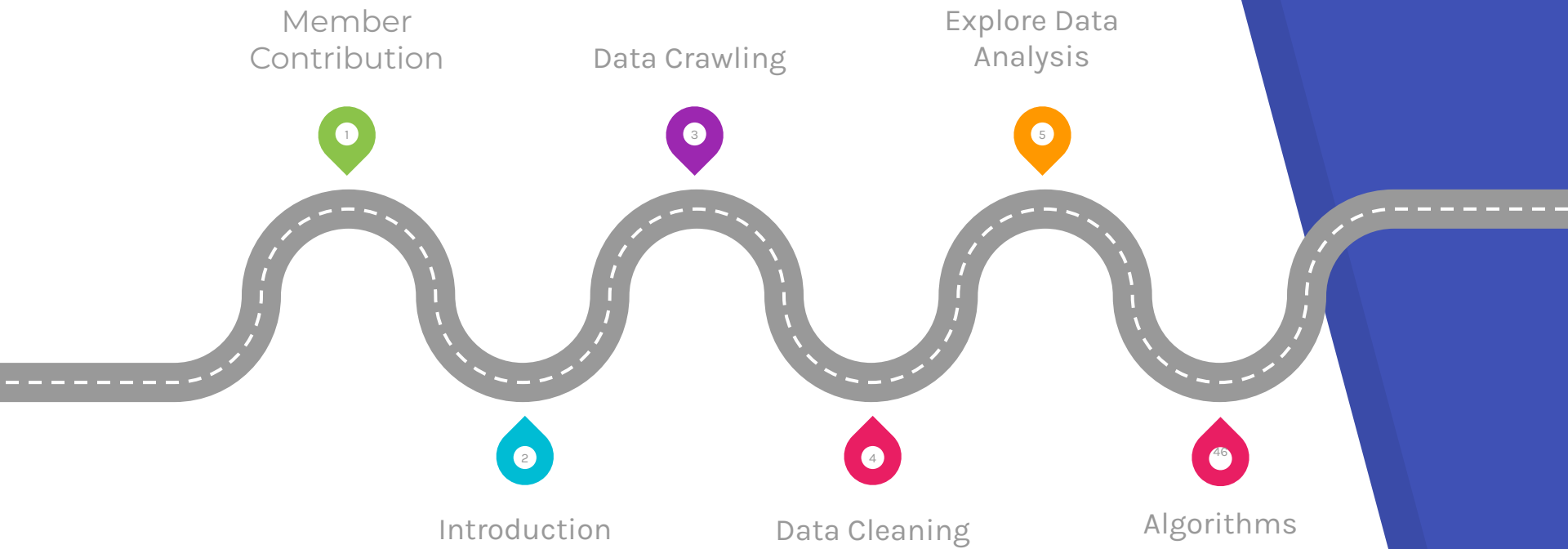
Nguyen Hoang Vu
20190100

Dao Duc Manh
20194794

Dang Quang Minh
20194796

Vu Hoang Nam
20194809

Content


Member Contribution
1
Introduction
2
Data Crawling
3
Data Cleaning
4
Explore Data Analysis
5
Algorithms
46

1.

MEMBER CONTRIBUTION

Member contribution

Member name	Task	Review by
Đào Đức Mạnh	❑ Clean data ❑ Explore data ❑ Report, slide	All member
Đặng Quang Minh	❑ Crawl data ❑ Algorithms ❑ Report, slide	All member
Nguyễn Hoàng Vũ	❑ Crawl data ❑ Algorithms ❑ Report, slide	All member
Vũ Hoàng Nam	❑ Clean data ❑ Explore data ❑ Report, slide	All member

2.
INTRODUCTION

INTRODUCTION

Approximately 40 million used vehicles are sold each year. Effective pricing strategies can help any company or individual to efficiently sell its products in a competitive market and make a profit, that is why we have chosen this topic for the data science project.

In this report, we will illustrate our process of doing the research, beginning from crawling car data to predicting car prices using some Machine Learning algorithm.

3.

DATA CRAWLING

CRAWLING TOOLS

Extract data from sites with multiple levels of navigation

An open-source and collaborative framework for extracting the data

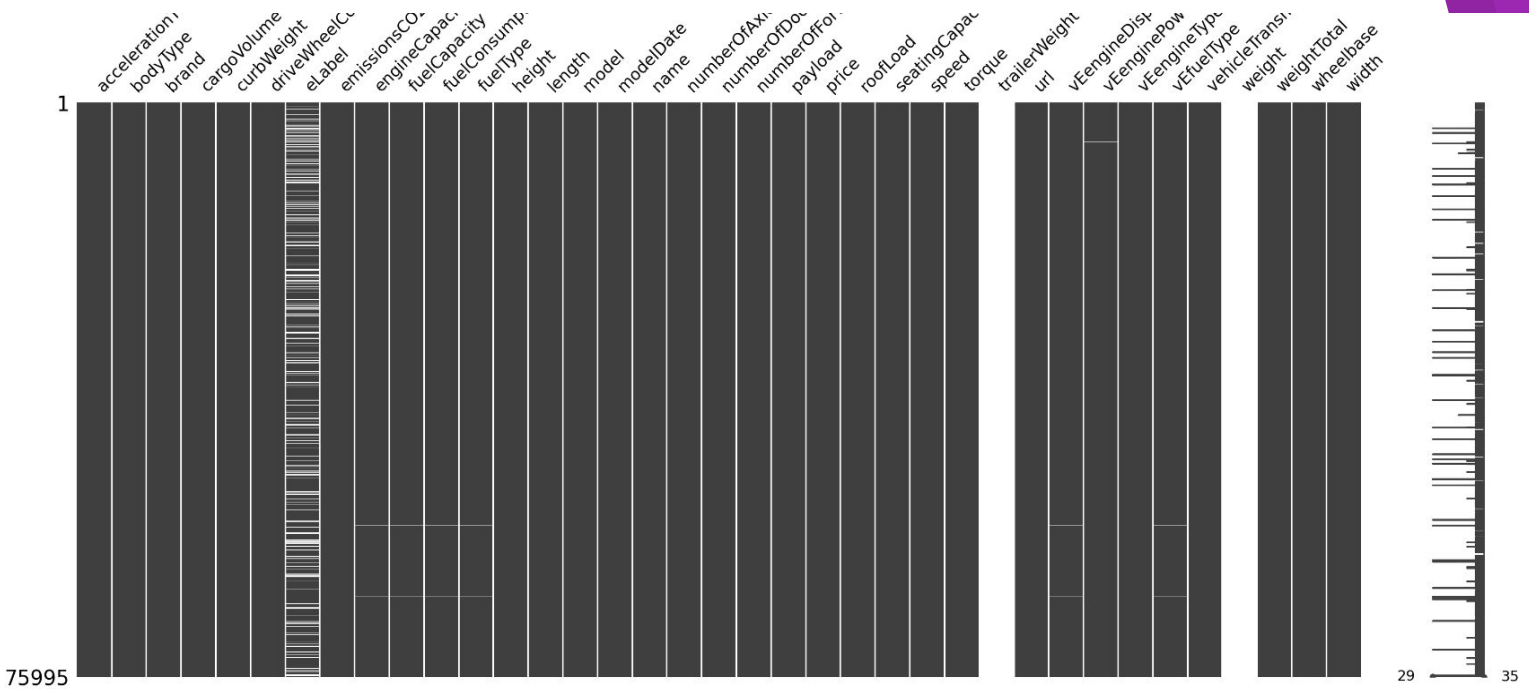
CRAWLING PROCESS

- Flow
  - Using Web Scraper, crawls all car links
    - 84, 000 links available
  - Using scrapy, extracts features from links
    - 37 features + more than 75, 000 data lines
- Difficulty
  - Number of lines of data is large
    - Takes nearly a day for crawling
  - Old links and old website

4.

Data Cleaning

Null value


bodyType
brand
cargoVolume
curbWeight
driveWheelC
eLabel
engineCapac
fuelCapacity
fuelType
height
length
model
modelDate
name
payload
price
roofLoad
seatingCapac
speed
torque
trailerWeight
url
vEngineDisp
vEnginePow
vEngineTyp
vEfuelType
vehicleTransi
weight
weightTotal
wheelbase
width
1
75995
29
35

METHOD HANDLING

Deleting rows

- Pros
  - Complete remove
  - Delete row/column not high value
- Cons
  - Loss info
  - High missing percentage

Mean/Median/Mode

- Pros
  - Prevent data loss
- Cons
  - Imputing variance and bias

Fill nall With KNN

- Pros
  - Correlation is ignored
- Cons
  - Time-consuming
  - Choice distance not robust result

Cleaning Process

- Remove two null columns (trailerWeight, weight)
- Null data after cleaning takes > 30% of data
  - NO deleting rows
- Fill the NaN value by the most frequent value
- Use KNN in order to fill our missing data
  - 30 features and 75824 lines of data

5.

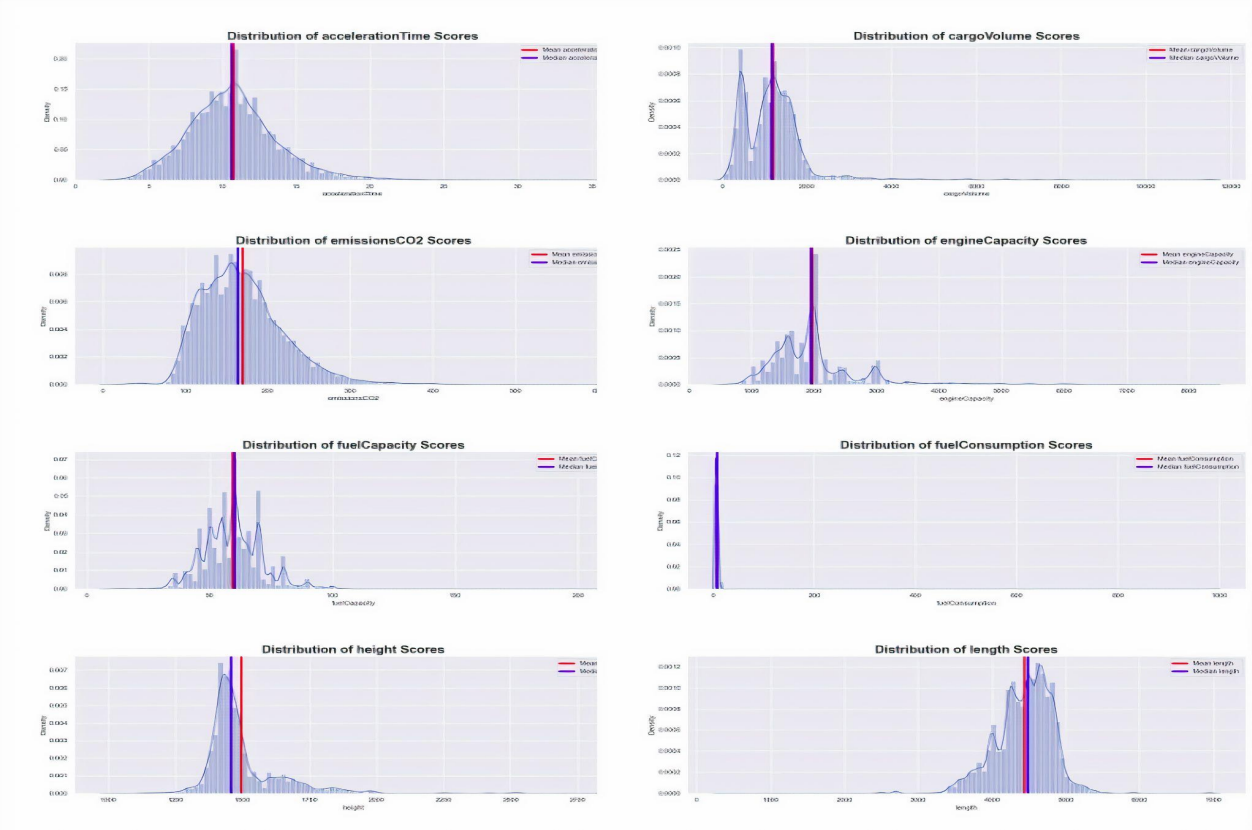
DATA EXPLORING

General Analysis

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75995 entries, 0 to 75994
Data columns (total 37 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   accelerationTime        73763 non-null  float64
 1   bodyType                75995 non-null  object
 2   brand                   75995 non-null  object
 3   cargoVolume             74946 non-null  float64
 4   curbWeight              75769 non-null  float64
 5   driveWheelConfiguration 75995 non-null  object
 6   eLabel                  61841 non-null  object
 7   emissionsCO2            61705 non-null  float64
 8   engineCapacity          75818 non-null  float64
 9   fuelCapacity            75815 non-null  float64
 10  fuelConsumption         74862 non-null  float64
 11  fuelType                75858 non-null  object
 12  height                  75925 non-null  float64
 13  length                  75974 non-null  float64
 14  model                   75995 non-null  object
 15  modelDate               75995 non-null  int64
 16  name                    75995 non-null  object
 17  numberOfAxles           75995 non-null  int64
 18  numberOfDoors           75995 non-null  int64
 19  numberOfForwardGears    74026 non-null  float64
 20  payload                 74976 non-null  float64
 21  price                   75824 non-null  float64
 22  roofLoad                64773 non-null  float64
 23  seatingCapacity         74026 non-null  float64
 24  speed                   75215 non-null  float64
 25  torque                  75888 non-null  float64
 26  trailerWeight           0 non-null      float64
 27  url                     75995 non-null  object
 28  vEengineDisplacement    75818 non-null  float64
 29  vEenginePower           75946 non-null  float64
 30  vEengineType            75995 non-null  object
 31  vEfuelType              75858 non-null  object
 32  vehicleTransmission     75995 non-null  object
 33  weight                  0 non-null      float64
 34  weightTotal             74983 non-null  float64
 35  wheelbase               75969 non-null  float64
 36  width                   75970 non-null  float64
dtypes: float64(23), int64(3), object(11)
memory usage: 21.5+ MB
```

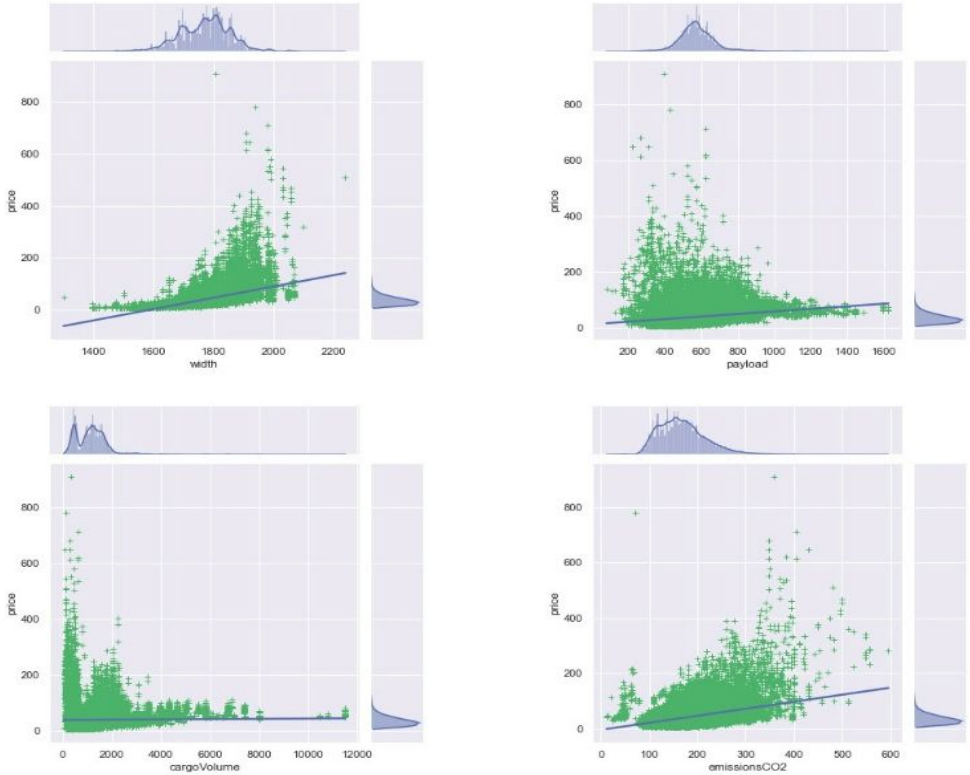
Numerical Data

- Explore Skewness



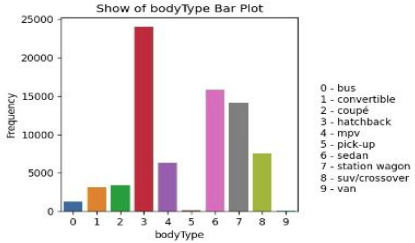
Numerical Data

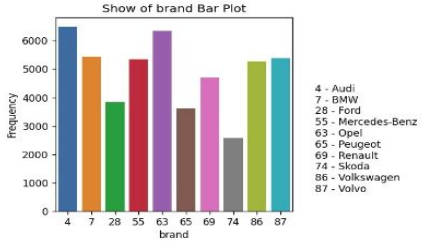
- Relationship with Target

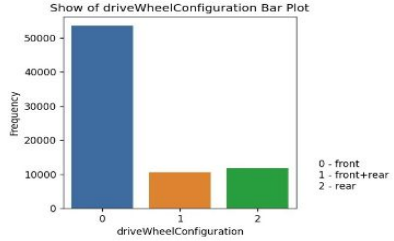


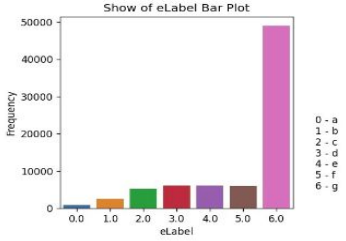
Categorical Data

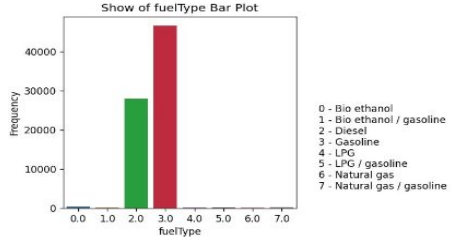
- Barplot

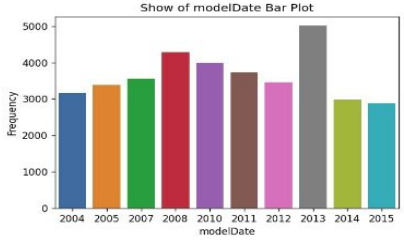






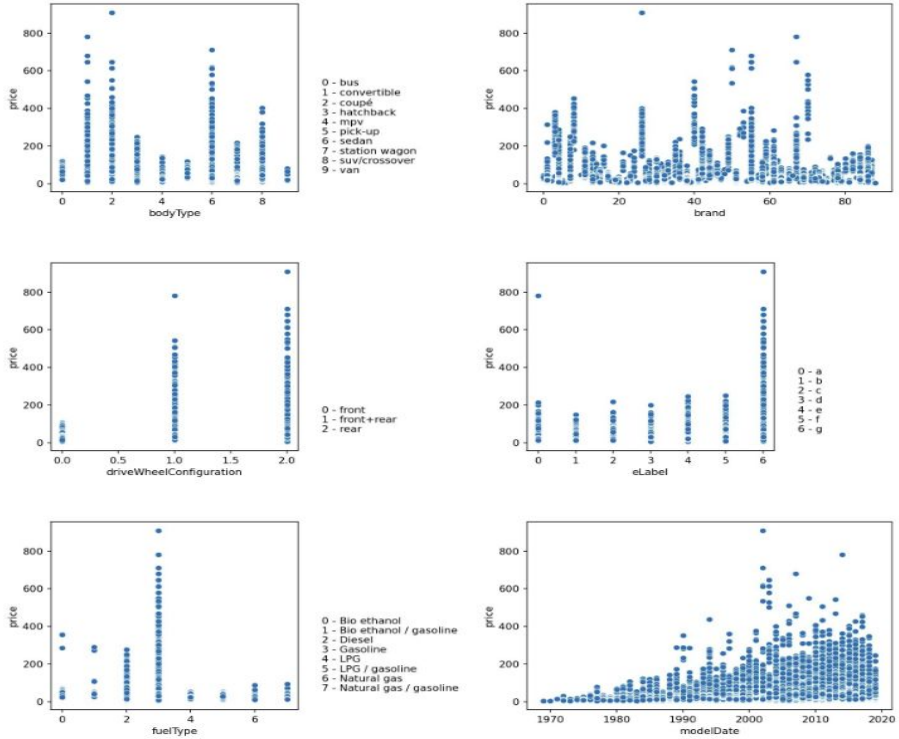




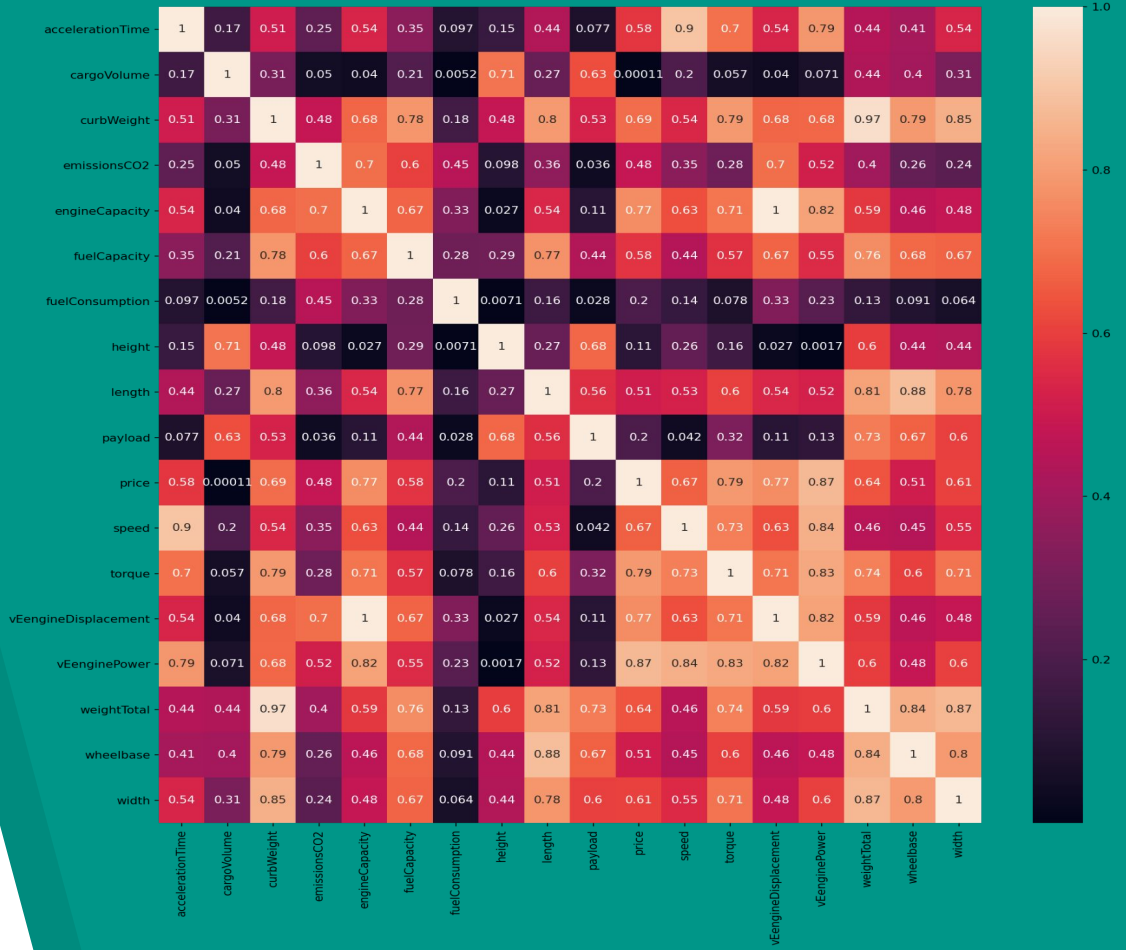


Categorical Data

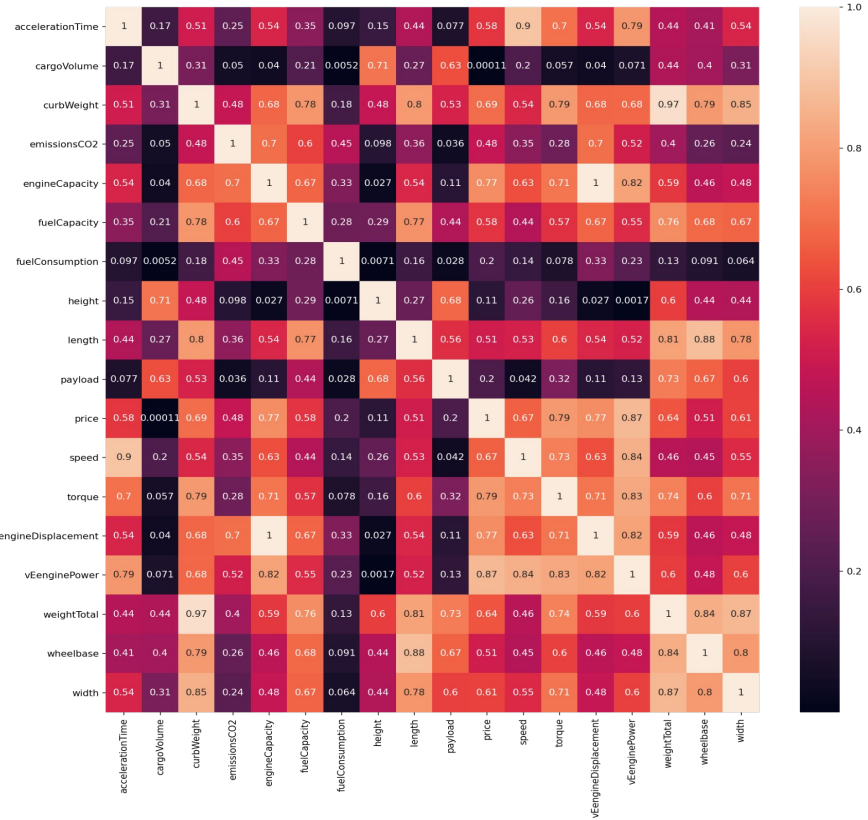
- Scatter plot



Heatmap


accelerationTime 1 0.17 0.51 0.25 0.54 0.35 0.097 0.15 0.44 0.077 0.58 0.9 0.7 0.54 0.79 0.44 0.41 0.54
cargoVolume 0.17 1 0.31 0.05 0.04 0.21 0.0052 0.71 0.27 0.63 0.00011 0.2 0.057 0.04 0.071 0.44 0.4 0.31
curbWeight 0.51 0.31 1 0.48 0.68 0.78 0.18 0.48 0.8 0.53 0.69 0.54 0.79 0.68 0.68 0.97 0.79 0.85
emissionsCO2 0.25 0.05 0.48 1 0.7 0.6 0.45 0.098 0.36 0.036 0.48 0.35 0.28 0.7 0.52 0.4 0.26 0.24
engineCapacity 0.54 0.04 0.68 0.7 1 0.67 0.33 0.027 0.54 0.11 0.77 0.63 0.71 1 0.82 0.59 0.46 0.48
fuelCapacity 0.35 0.21 0.78 0.6 0.67 1 0.28 0.29 0.77 0.44 0.58 0.44 0.57 0.67 0.55 0.76 0.68 0.67
fuelConsumption 0.097 0.0052 0.18 0.45 0.33 0.28 1 0.0071 0.16 0.028 0.2 0.14 0.078 0.33 0.23 0.13 0.091 0.064
height 0.15 0.71 0.48 0.098 0.027 0.29 0.0071 1 0.27 0.68 0.11 0.26 0.16 0.027 0.0017 0.6 0.44 0.44
length 0.44 0.27 0.8 0.36 0.54 0.77 0.16 0.27 1 0.56 0.51 0.53 0.6 0.54 0.52 0.81 0.88 0.78
payload 0.077 0.63 0.53 0.036 0.11 0.44 0.028 0.68 0.56 1 0.2 0.042 0.32 0.11 0.13 0.73 0.67 0.6
price 0.58 0.00011 0.69 0.48 0.77 0.58 0.2 0.11 0.51 0.2 1 0.67 0.79 0.77 0.87 0.64 0.51 0.61
speed 0.9 0.2 0.54 0.35 0.63 0.44 0.14 0.26 0.53 0.042 0.67 1 0.73 0.63 0.84 0.46 0.45 0.55
torque 0.7 0.057 0.79 0.28 0.71 0.57 0.078 0.16 0.6 0.32 0.79 0.73 1 0.71 0.83 0.74 0.6 0.71
vEengineDisplacement 0.54 0.04 0.68 0.7 1 0.67 0.33 0.027 0.54 0.11 0.77 0.63 0.71 1 0.82 0.59 0.46 0.48
vEenginePower 0.79 0.071 0.68 0.52 0.82 0.55 0.23 0.0017 0.52 0.13 0.87 0.84 0.83 0.82 1 0.6 0.48 0.6
weightTotal 0.44 0.44 0.97 0.4 0.59 0.76 0.13 0.6 0.81 0.73 0.64 0.46 0.74 0.59 0.6 1 0.84 0.87
wheelbase 0.41 0.4 0.79 0.26 0.46 0.68 0.091 0.44 0.88 0.67 0.51 0.45 0.6 0.46 0.48 0.84 1 0.8
width 0.54 0.31 0.85 0.24 0.48 0.67 0.064 0.44 0.78 0.6 0.61 0.55 0.71 0.48 0.6 0.87 0.8 1
accelerationTime
cargoVolume
curbWeight
emissionsCO2
engineCapacity
fuelCapacity
fuelConsumption
height
length
payload
price
speed
torque
vEengineDisplacement
vEenginePower
weightTotal
wheelbase
width
1.0
0.8
0.6
0.4
0.2

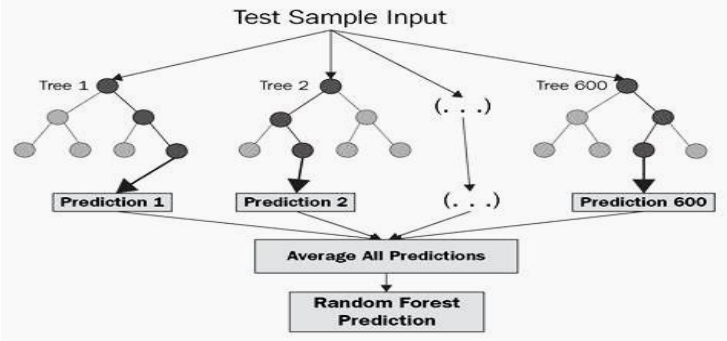
High correlation filter



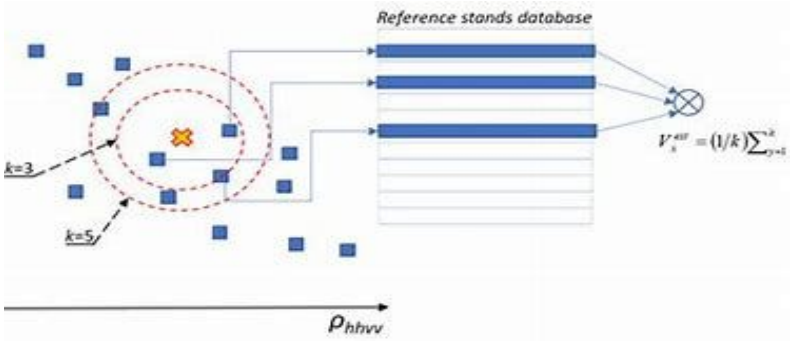
- If the correlation values between some features are big, we can consider them as the same in the dataset.
- Those similar features may make the dataset harder for the model to learn.

6.
ALGORITHMS

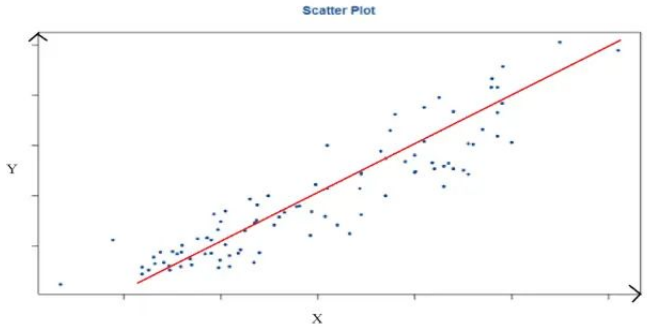
Algorithms



Random Forest



KNN



Linear Regression

Algorithms

- Besides Random Forest, KNN, Linear Regression, we also try some Boosting model:
  - Cat Boosting
  - XGB Boosting
  - LGBM Boosting

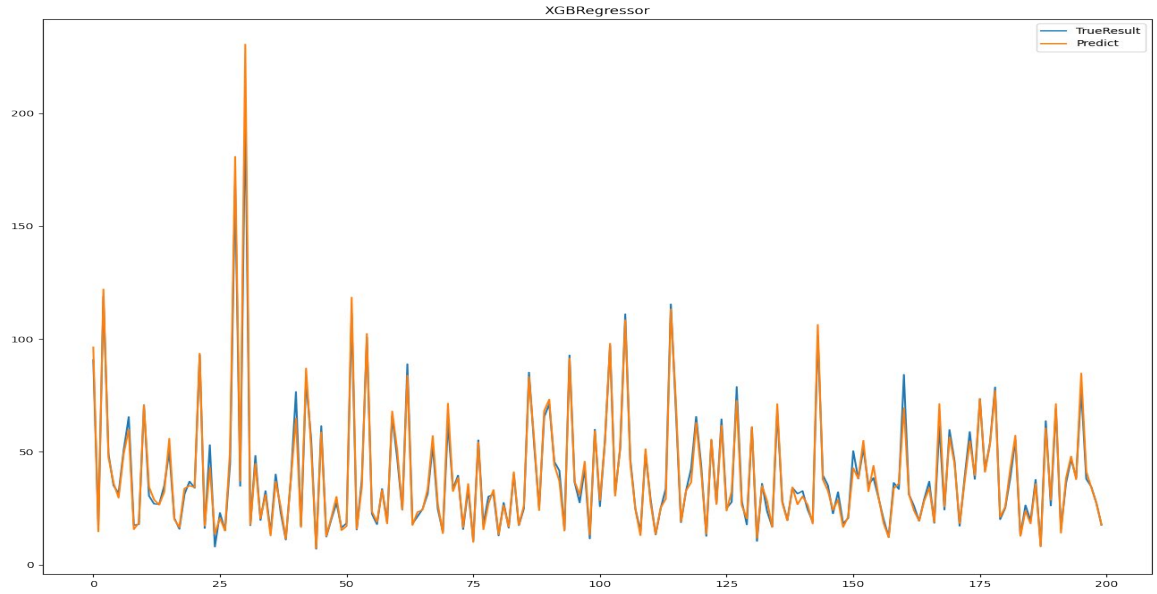
- To our knowledge, the basic idea of the Boosting algorithms is to use the combination of multiple weak models. Then, training weak learners sequentially, each will try to correct its previous model.

Results

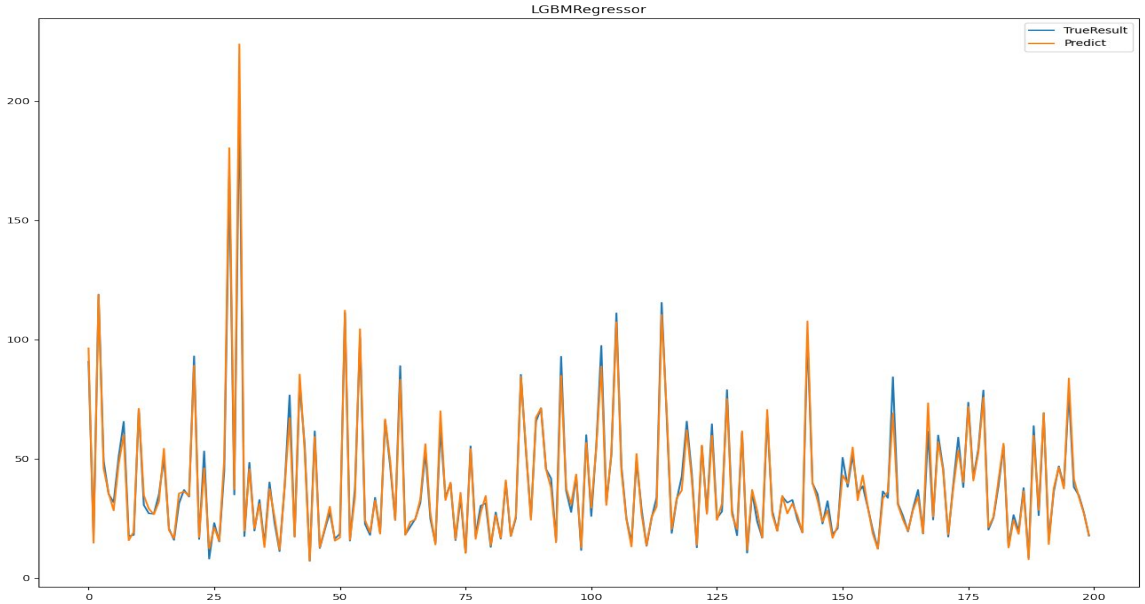
Algorithm	RMSE loss
LinearRegression	11.31
KNN Regressor	7.90
CatBoostRegressor	7.19
RandomForestRegressor	6.19
XGBRegressor	6.05
LGBMRegressor	6.01

- We split the dataset into 2 parts:
  - Train set 70%
  - Test set 30%
- We use StandardScaler for normalizing data.

Two best models



Two best models



THANKS FOR LISTENING!