SMARTLIST

Used Car Price Range Prediction

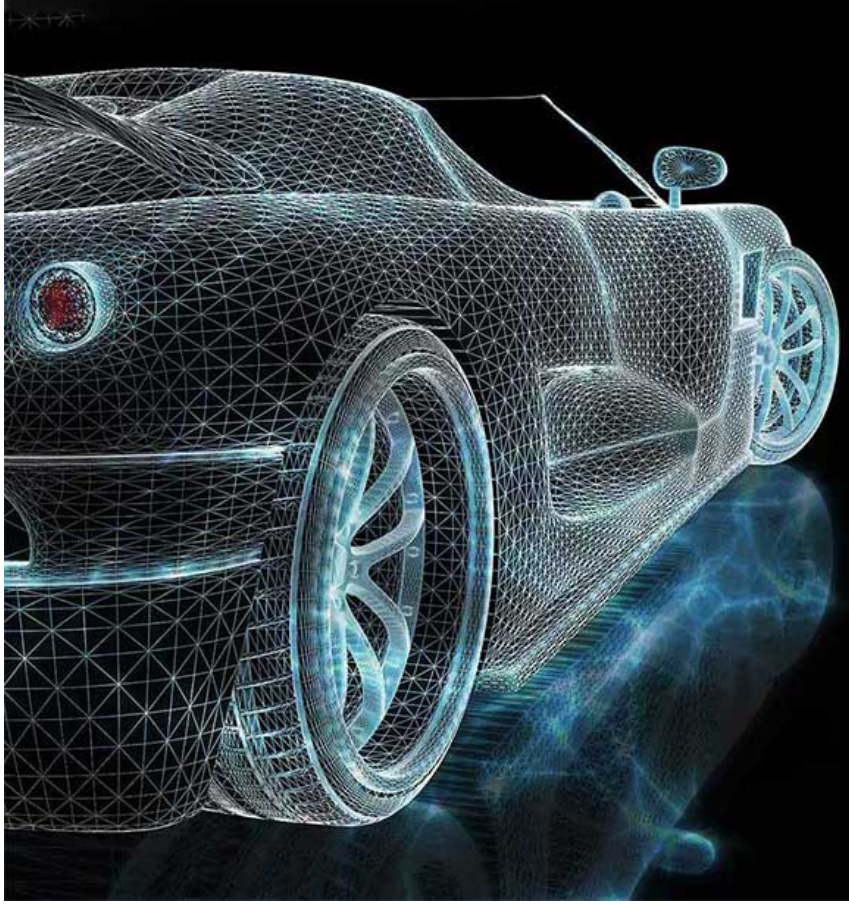
By Anthony Kwok

27 Nov 2023

in/anthony-kwok01 @ kwokanthony073

[Source Code]

Agenda



SMARTLIST

01 **Problem Statement**
What Problem | Who Cares | Why We Need

02 **Data Science Solution**
Our Solution | Potential Impact | Dataset

03 **Preprocessing & Insights**
Data Preprocessing | Key Insights

04 **Result & Conclusion**
Model Evaluation | Demo | Roadmap

Problem Statement

It takes 4 weeks to sell a used car.

How can we speed up the buying/selling process? To help both buyer and seller to achieve their goal?

Who Cares?

SMARTLIST

Buyer

- Price Uncertainty
- Worry about Overpayment

Dealer

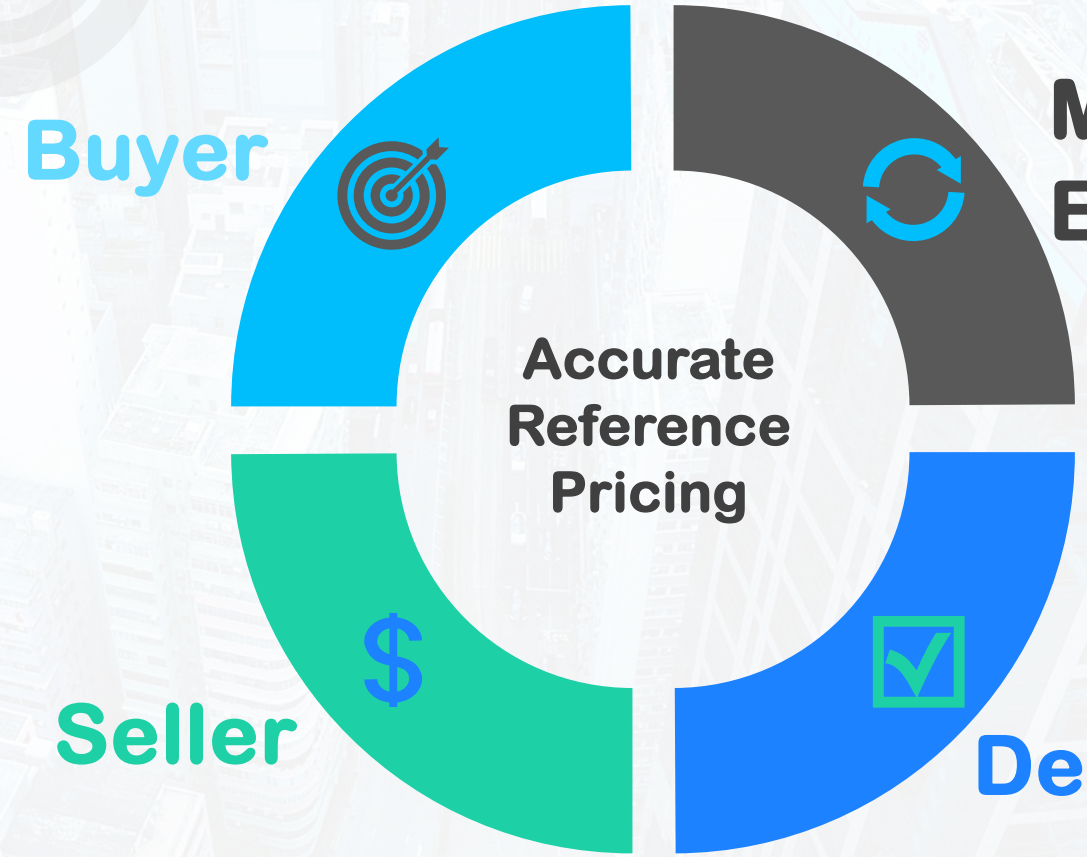
- Hard to manage expectation on both sides
- Long trading time

Seller

- Lack of Knowledge about Vehicle Pricing
- Missed Opportunity

What we need is TODO

SMARTLIST




SMARTLIST
Data-Driven
Solution

Proposed Solution

SMARTLIST

Specification

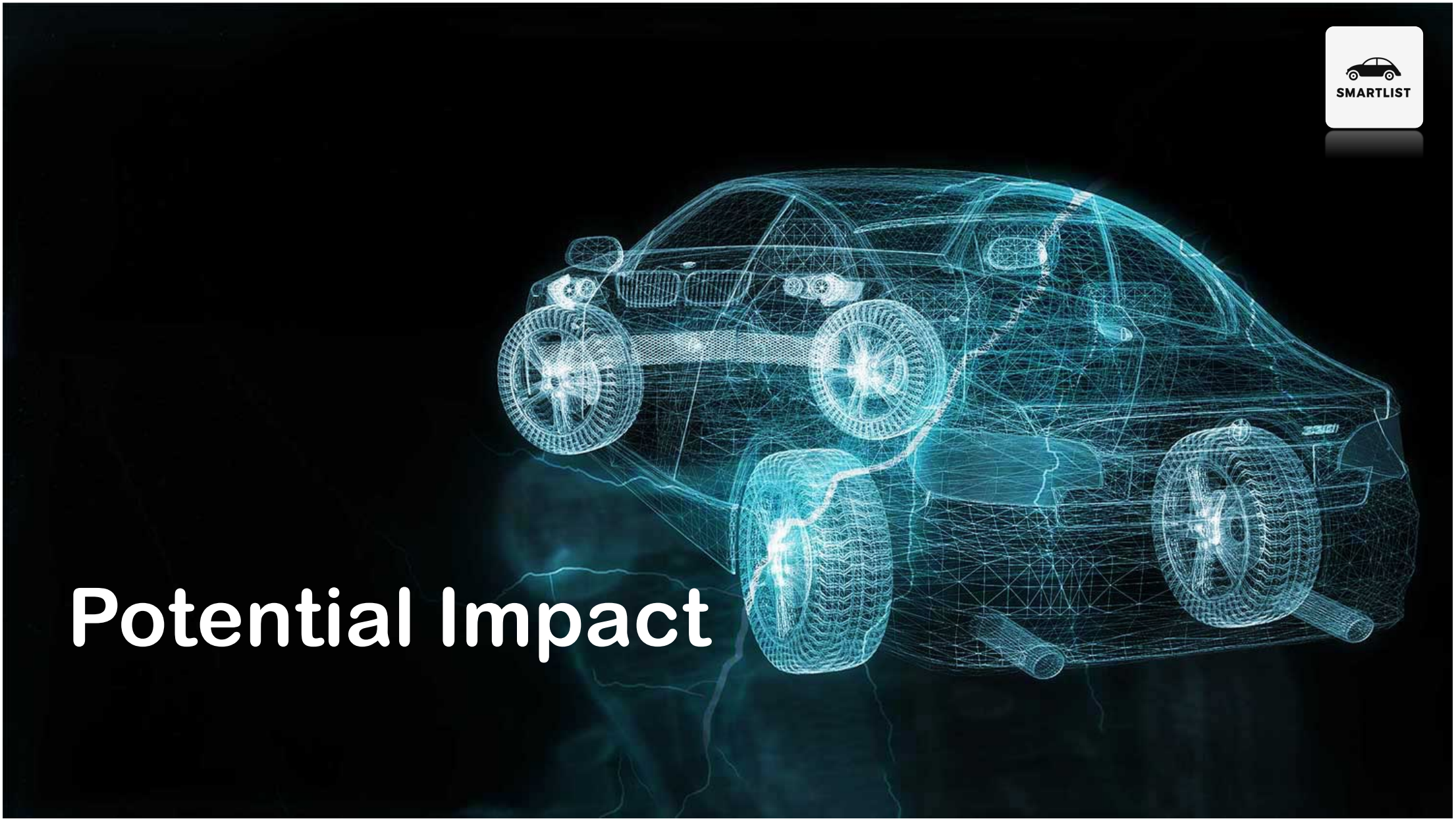
Machine Learning

Price Range

History

AI Solution

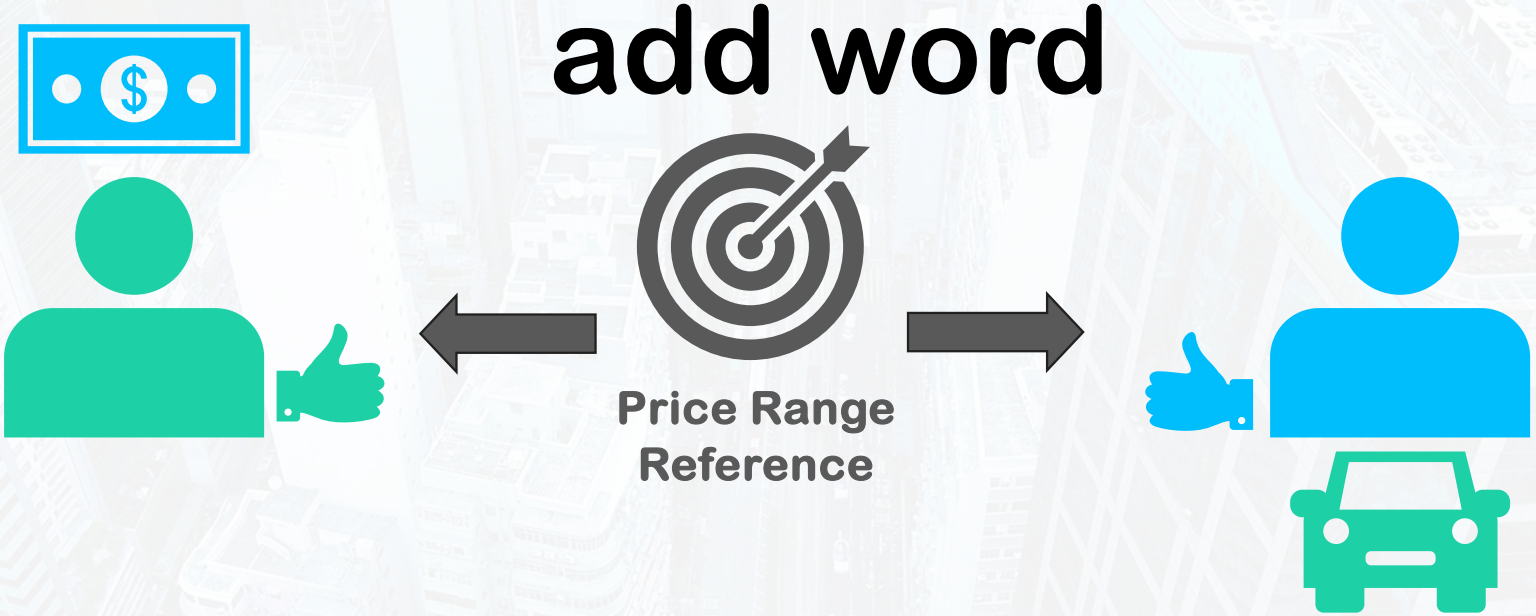
Speed Up


SMARTLIST
Potential Impact

Potential Impact TODO

SMARTLIST

add word



Save
> 1 weeks

10% in
Revenue


SMARTLIST
Dataset

Dataset

SMARTLIST

Geographical

- Canada
- USA

Temporal

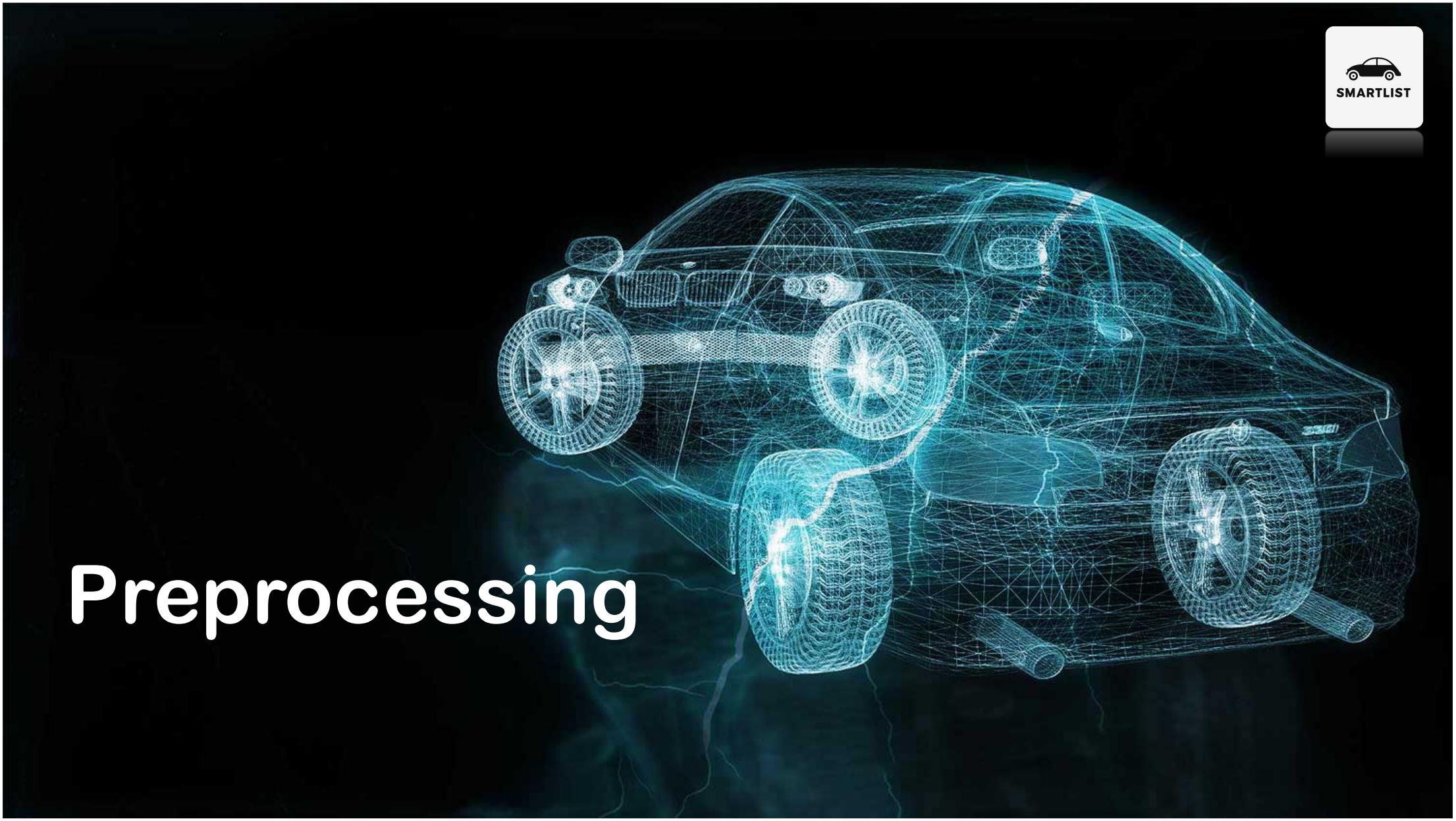
- April 2015
- June 2021

Source

- From Kaggle
- Source Link

Size

- 1.35 GB
- 7.5M Rows
  21 Columns


SMARTLIST
Preprocessing

Data Preprocessing

SMARTLIST

1 **Label Encoding**
Transform exact price to price range.

2 **Missing Value Handling**
Fill in missing value based on the occurrence probability.

3 **Feature Flattening**
Some features are in list format which need to be transformed as vector representation.

4 **Feature Transformation**
Log transformation | One-hot Encoding | Target Encoding


SMARTLIST
Key Insights

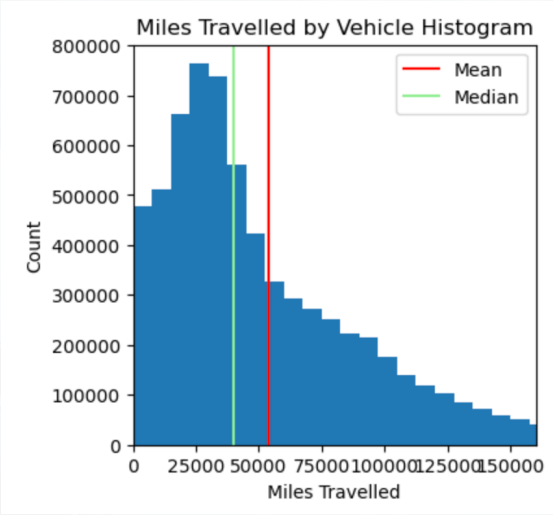
Key Insights - 1

SMARTLIST

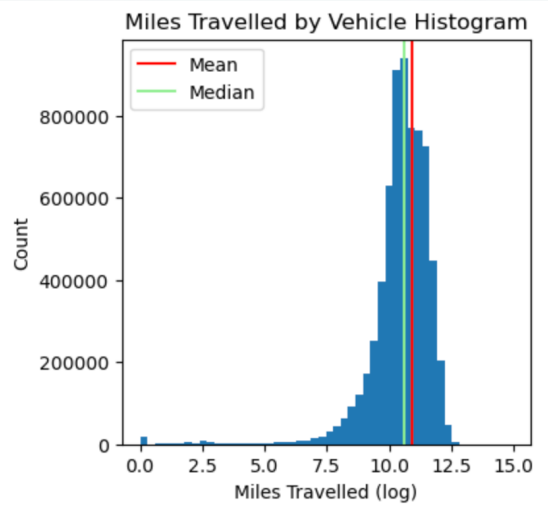
Log-Normal Distribution

- Transform the data to become “Normal Distribution”
- Many machine learning models perform better with normal distribution data / features.

Before



After

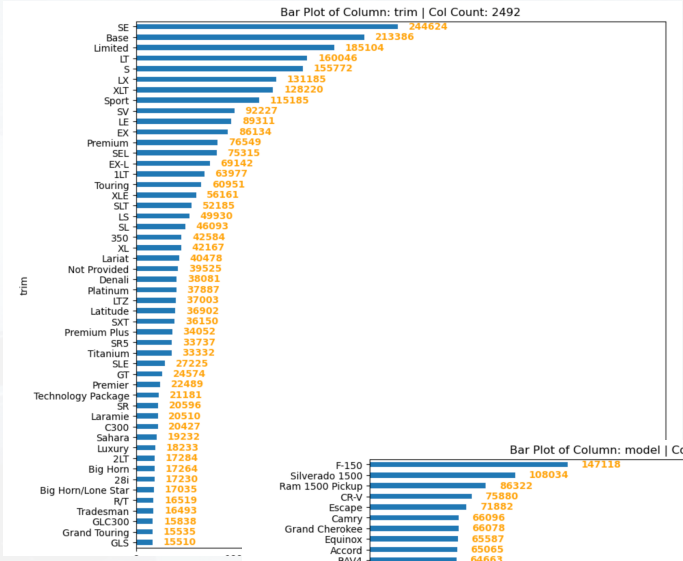


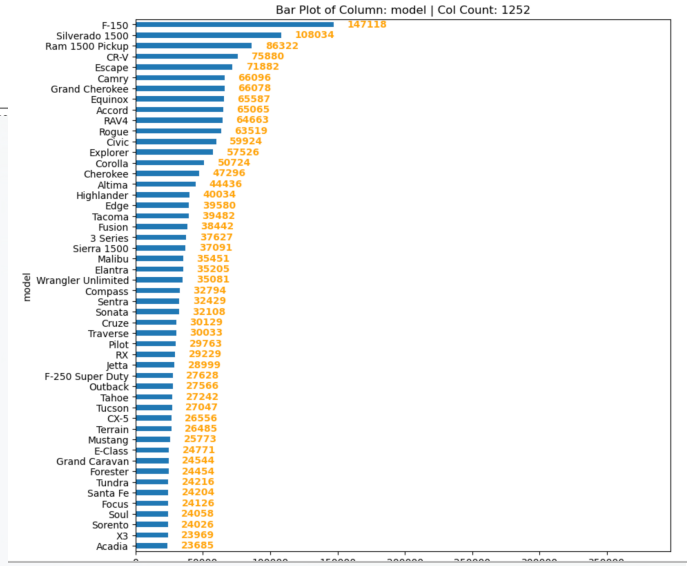
Key Insights - 2

SMARTLIST

Columns with too many values

- One-hot encoding
  - ➔ Increased Dimensions
- Count encoding

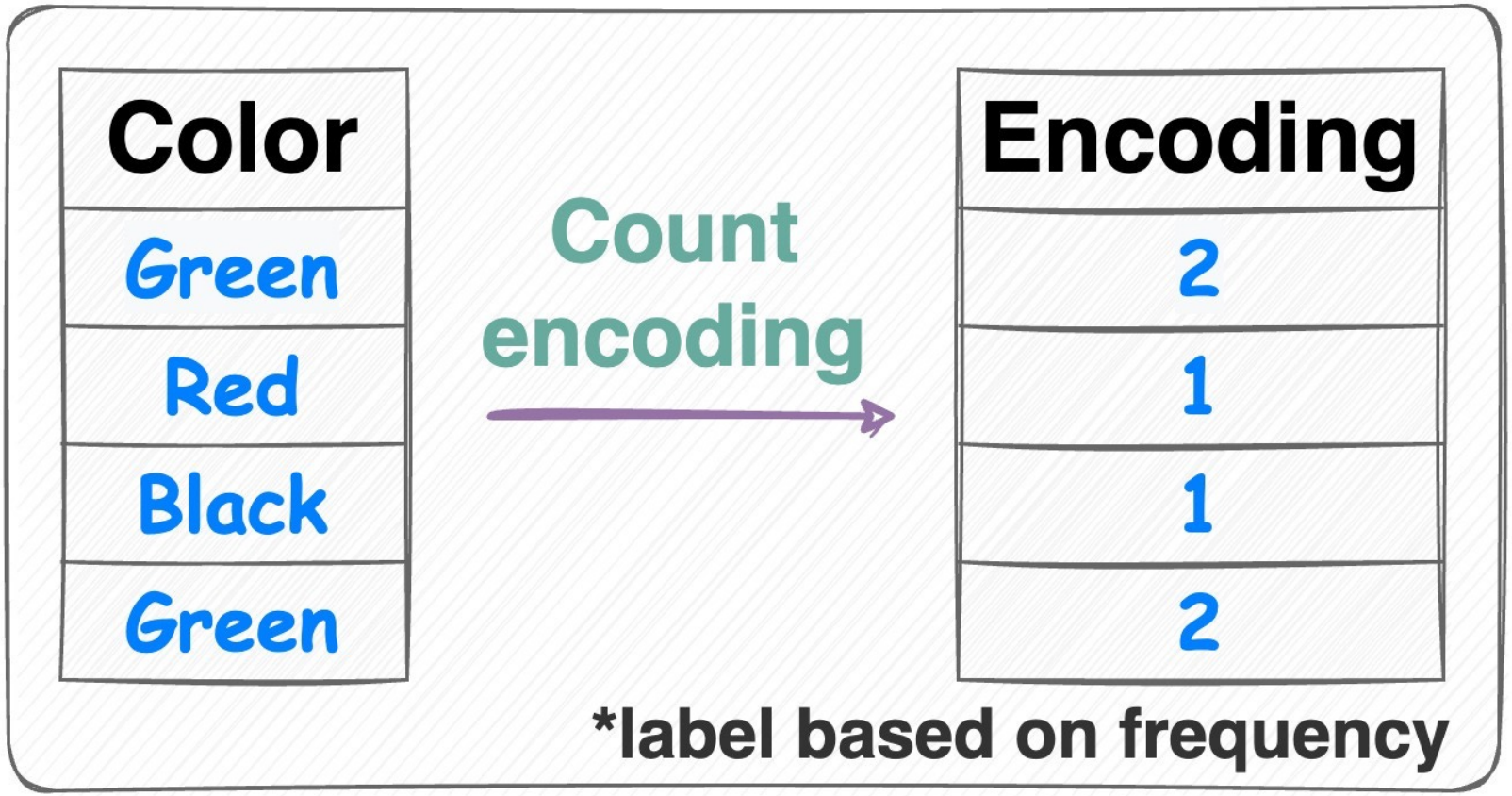





```
Column: make has 63 distinct values
Column: model has 1252 distinct values
Column: trim has 2492 distinct values
Column: city has 6095 distinct values
Column: state has 68 distinct values
```

Count Encoding

SMARTLIST

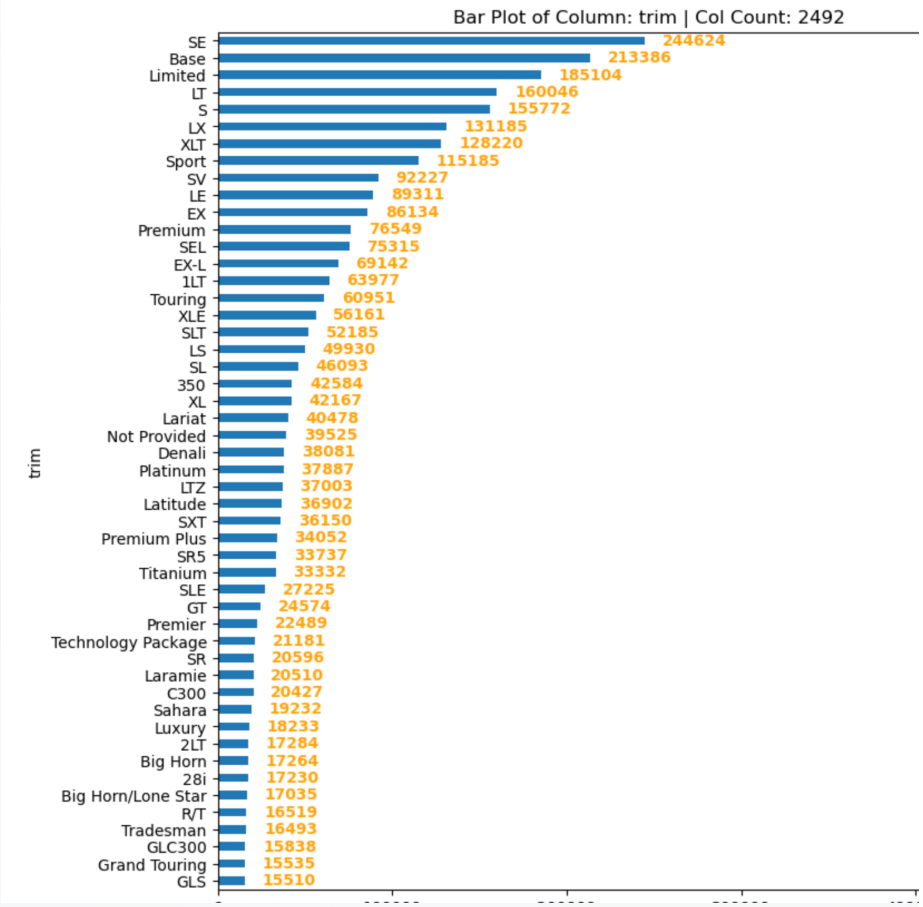


Key Insights - 2

SMARTLIST

Columns with too many values

- One-hot encoding
  - ➔ Increased Dimensions
- Count encoding
  - ➔ Extreme Value
  - ➔ Affecting the scaling steps
- Target encoding



Target Encoding

SMARTLIST

Target Encoding

workclass	target
State-gov	0
Self-emp-not-inc	1
Private	0
Private	0
Private	1

workclass	target mean
State-gov	0
Self-emp-not-inc	1
Private	1/3

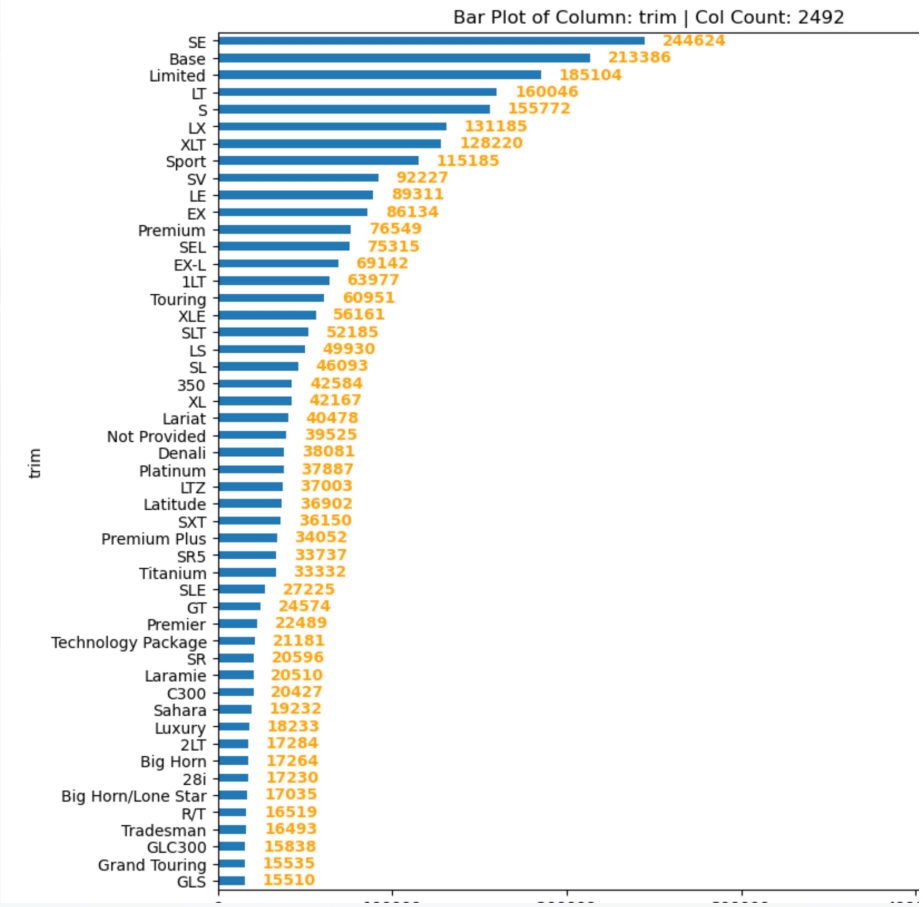
workclass
0
1
1/3
1/3
1/3

Key Insights - 2

SMARTLIST

Columns with too many values

- One-hot encoding
  - ➔ Increased Dimensions
- Count encoding
  - ➔ Extreme Value
  - ➔ Affecting the scaling steps
- Target encoding
  - ➔ Does not increase dimensions
  - ➔ Encoded values between 0 ~ 1




SMARTLIST
Model Evaluation

Model Evaluation

SMARTLIST

Model Performance (10% Data)

Model	Best Params / Params	Training F1 Score	Testing F1 Score	
Logistic Regression (Baseline)	{'C': 1, 'class_weight': 'balanced','max_iter': 10000,'penalty': 'l2'}	69.74%	69.82%	
Logistic Regression	{'C': 1000, 'class_weight': None, 'max_iter': 10000, 'penalty': 'l2'}	71.77%	71.81%	
Decision Tree	{'max_depth': 25, 'min_samples_split': 50}	82.15%	82.94%	
Adaptive Boosting (AdaBoost)	{'learning_rate': 1, 'n_estimators': 100}	85.34%	85.89%	✅
eXtreme Gradient Boosting (XGBoost)	{'max_depth': 15, 'n_estimators': 30}	85.42%	85.85%	✅
Random Forest (RF)	{'max_depth': 50, 'min_samples_split': 25, 'n_estimators': 150}	85.39%	85.82%	✅
Naïve Bayes	*Default*	45.75%	43.18%	
Neural Network	3 Hidden Layers (50,25,10) + ReLU Activation Layers + Epochs = 3	32.06%	32.38%	

Model Evaluation

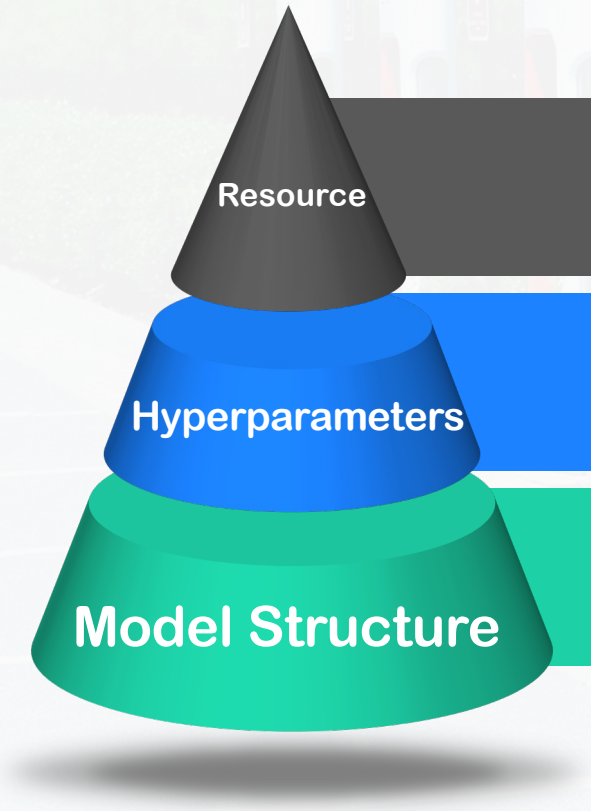
SMARTLIST

Model Performance (Full Set)

Model	Best Params / Params	Training Time	Training F1 Score	Testing F1 Score
Adaptive Boosting (AdaBoost)	{'learning_rate': 1, 'n_estimators': 100}	3h 14m 20s	86.95%	87.06%
eXtreme Gradient Boosting (XGBoost)	{'max_depth': 15, 'n_estimators': 30}	34m 43s	87.32%	87.46%
Random Forest (RF)	{'max_depth': 50, 'min_samples_split': 25, 'n_estimators': 150}	20m 47s	87.30%	87.47%

Model Evaluation Criteria

SMARTLIST



Balance between performance and resources.

Tune the hyperparameters with selected models.

Different model structures give different results.

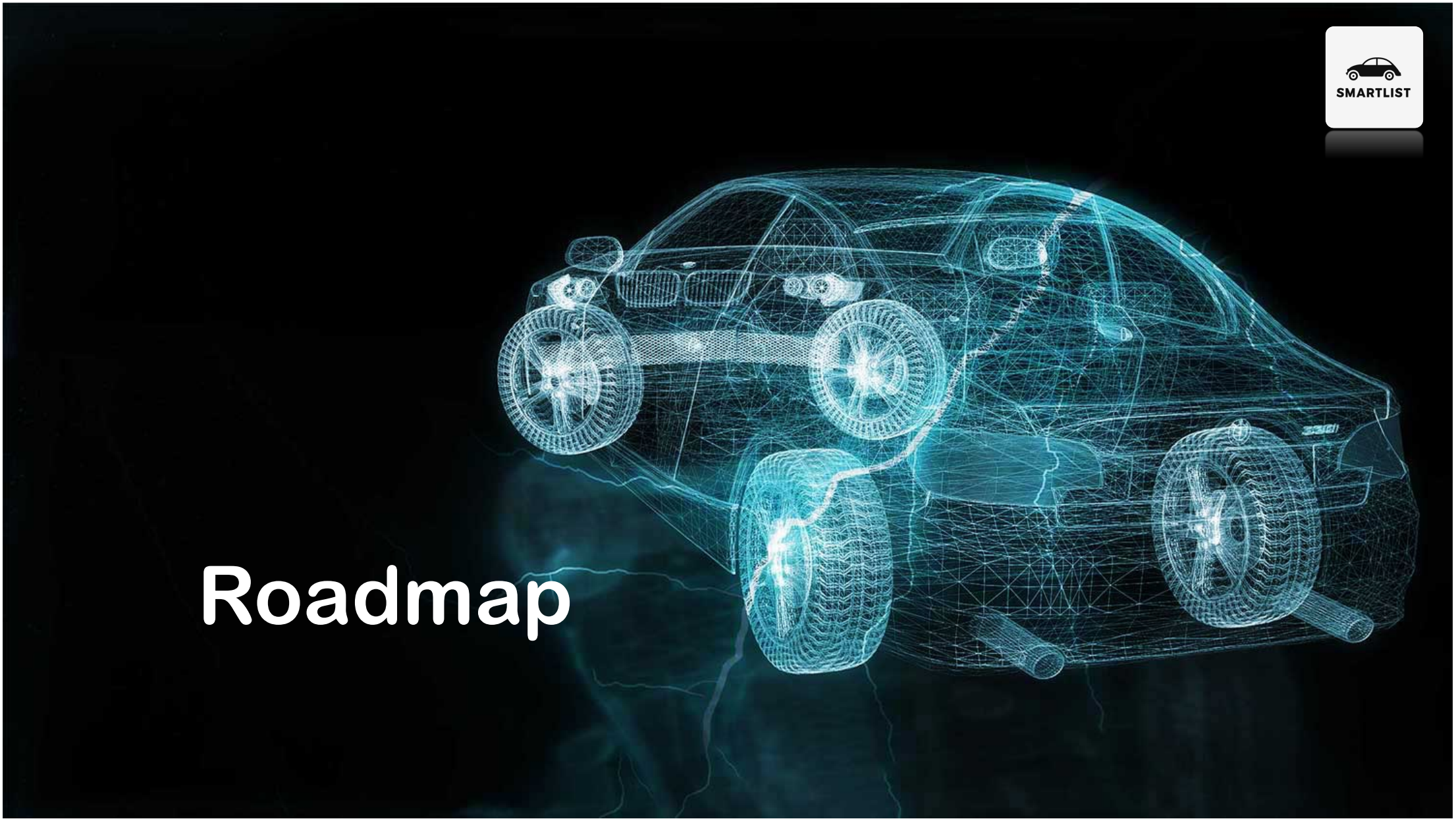
Model Evaluation

SMARTLIST

Model Performance (Full Set)

Model	Best Params / Params	Training Time	Training F1 Score	Testing F1 Score
Adaptive Boosting (AdaBoost)	{'learning_rate': 1, 'n_estimators': 100}	3h 14m 20s	86.95%	87.06%
eXtreme Gradient Boosting (XGBoost)	{'max_depth': 15, 'n_estimators': 30}	34m 43s	87.32%	87.46%
Random Forest (RF)	{'max_depth': 50, 'min_samples_split': 25, 'n_estimators': 150}	**20m 47s**	87.30%	**87.47%**

Demo Time


SMARTLIST
Roadmap

Roadmap

SMARTLIST

Sprint 0

- Problem Space Definition
- Scope Definition
- Dataset Research

Sprint 1

- Data Cleaning
- Exploratory Data Analysis
- Statistical Analysis

Sprint 2

- Data Preprocessing
- Baseline Model Training

Future

- Extend to Regression Task
- i.e. Predicting Exact Price
- Re-train model with more data
- Continually collecting new feedback
- Challenge Ourself ➔ Beat our RF model

Sprint 3

- Advance Modelling
- Web UI Integration
- Product Delivery


SMARTLIST
THANK YOU