

---

# Used Car Price Prediction

**CISC 520 Final Project**

Siyuan Feng

Xinyuan He

Yingyu Cao





# **Content**

**01**

**Introduction**

**02**

**Problem Statement**

**03**

**Methodology**

**04**

**Modeling**

**05**

**Result**

**06**

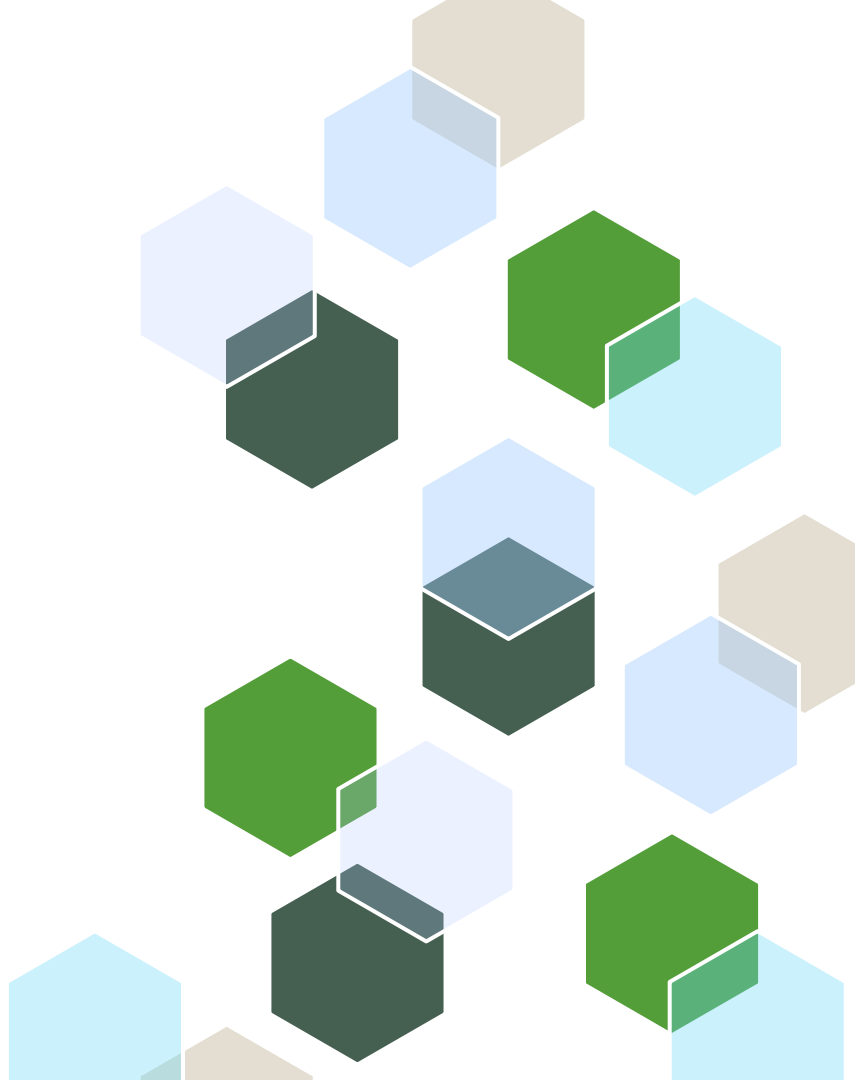
**Discussion**



---

01

# Introduction



# Introduction

The used car market has experienced significant growth in recent years, necessitating accurate pricing mechanisms to facilitate fair transactions between buyers and sellers. Traditional pricing methods, often reliant on manual assessments, fail to capture the complex interactions among various factors influencing car prices. This project aims to develop a predictive model for second-hand car prices using advanced machine learning techniques.

Our approach involves comprehensive data visualization, preprocessing, feature engineering, and model evaluation to ensure the robustness of the predictions. The detailed steps include utilizing permutation importance and auto feature selection with Random Forest to identify the most significant features; implementing imputation, encoding, and grid search on Random Forest and Gradient Boosting models to optimize performance. The final model, a Gradient Boosting Regressor, achieved a test accuracy of 79%.

---

02

# Problem Statement



# Problem Statement

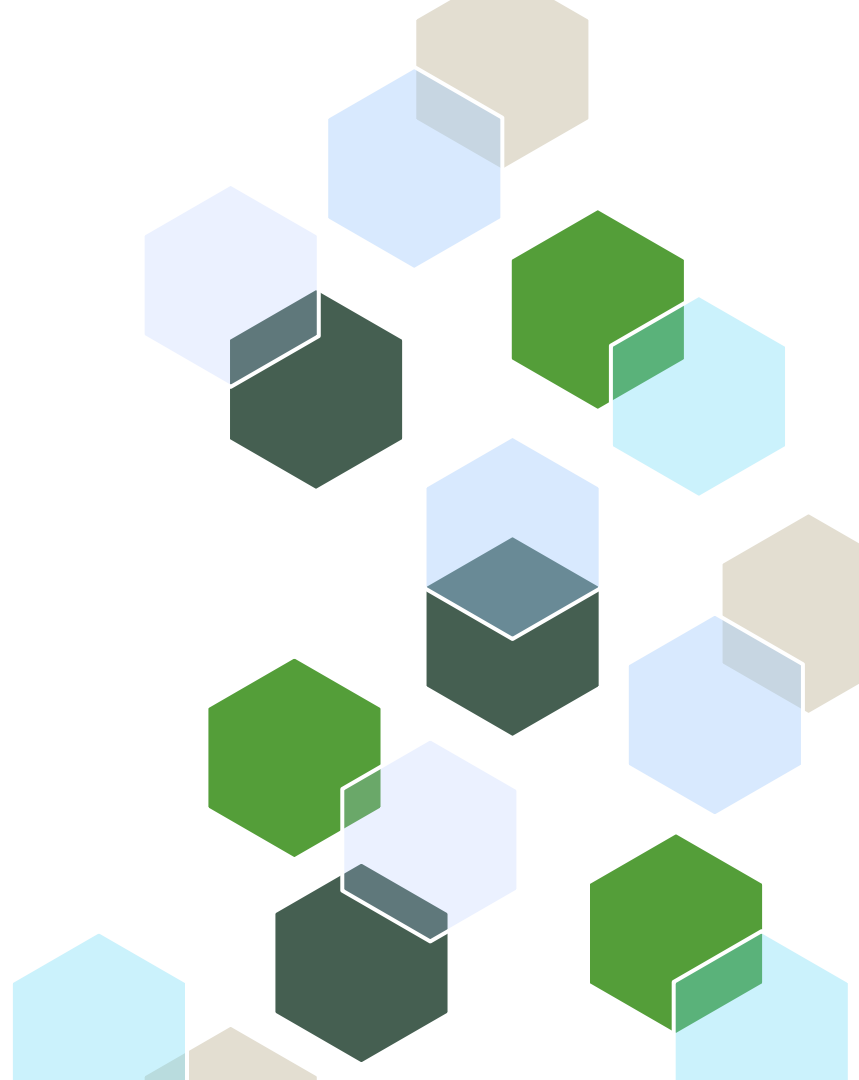
Determining the fair price of a used car is a multifaceted problem involving numerous variables. Traditional methods of pricing, while useful, are limited in their ability to capture non-linear relationships and interactions between these variables. This project seeks to address this limitation by developing a machine learning-based predictive model for second-hand car prices using Craigslist data.

By leveraging historical data and advanced algorithms, we aim to create a tool that provides accurate and data-driven price estimates. This model will assist stakeholders in the used car market by enhancing transparency and efficiency, ultimately leading to fairer transactions.

---

**03**

# Methodology



# Methodology



## Linear Regression

Base model. Easy to implement.



## Random Forest

Non-linear relationship.



## Gradient Boosting

Train on previous errors. Better performance.



## Decision Tree

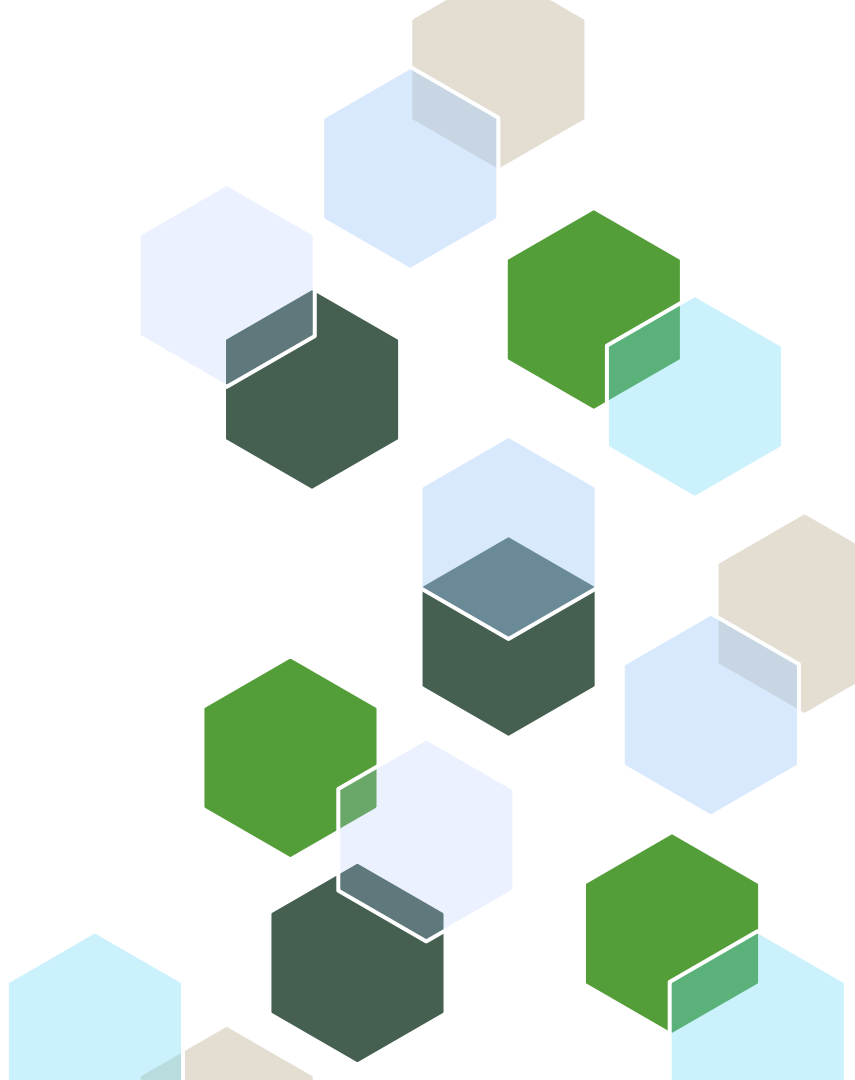
Explainable.



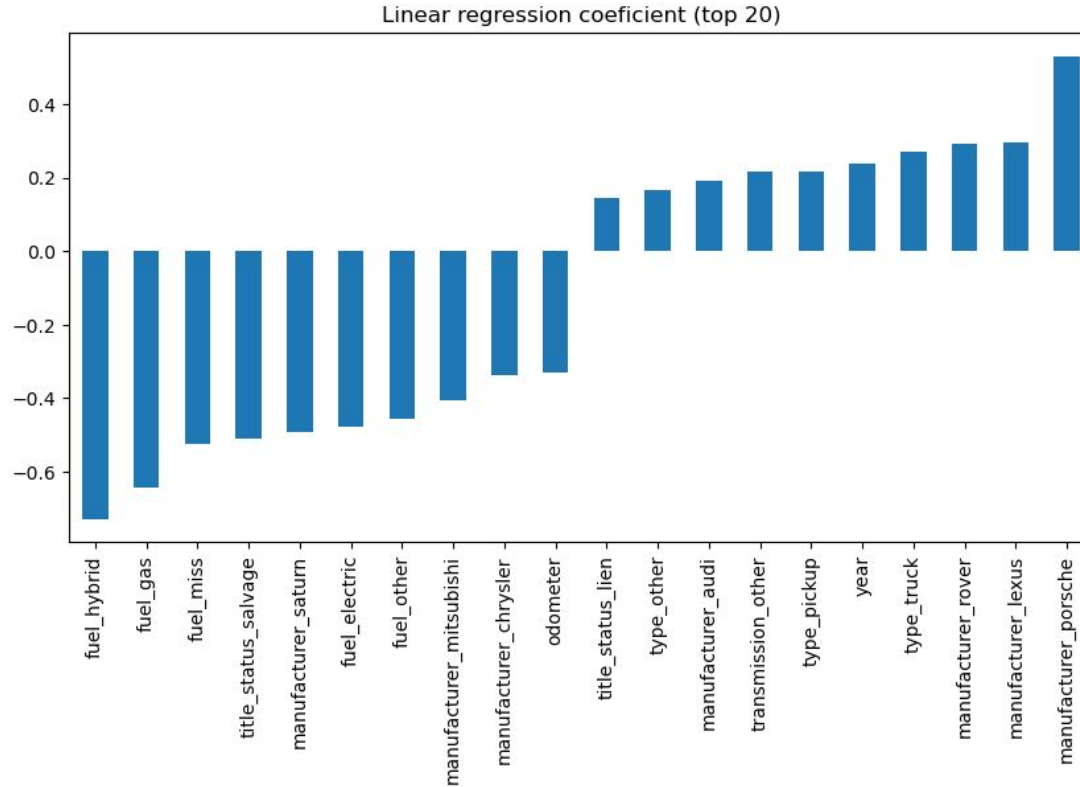
---

**04**

# Modeling



# Linear Regression



# Grid Search

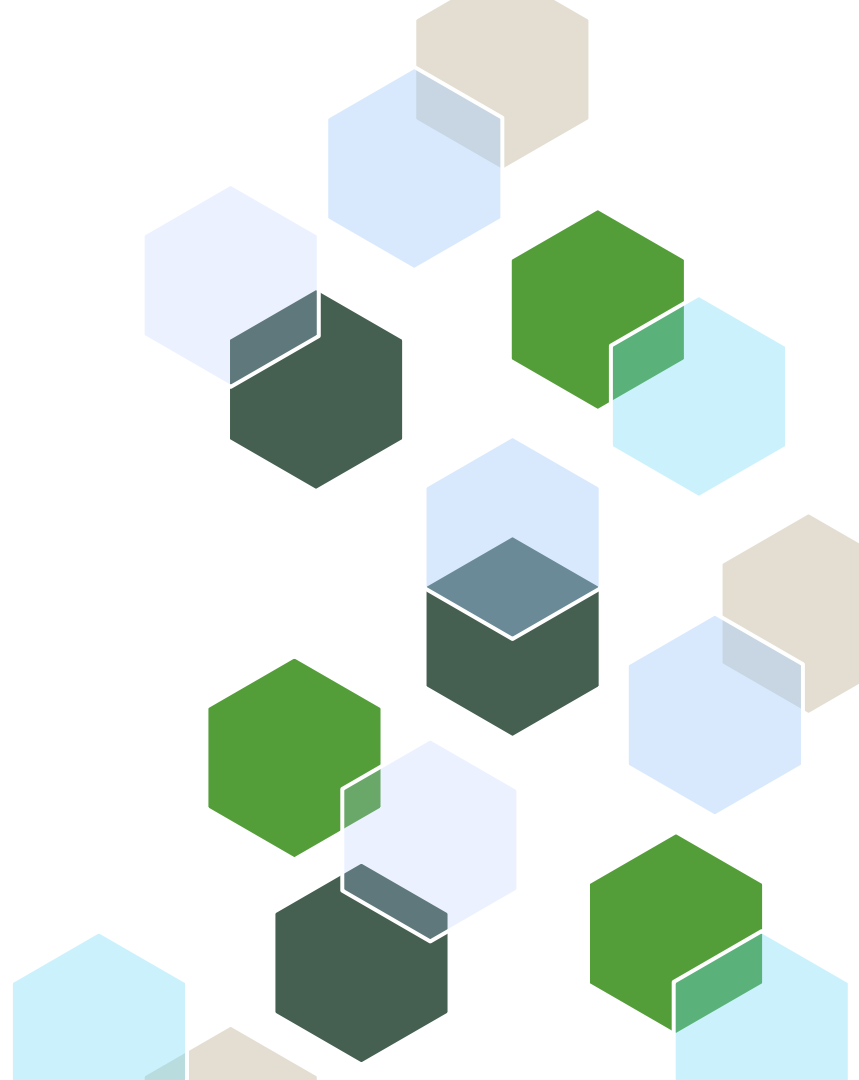
- Evaluated on R-square

Model	Hyper-Parameters	Values
Random Forest	Number of Estimators	[100, 150, 200]
	Max Depth	[7, 9, 11]
Gradient Boosting	Loss Function	['squared_loss', 'absolute_error', 'huber']
	Learning Rate	[0.001, 0.01, 0.1, 0.5]
	Max Depth	[7, 9, 11]
Decision Tree	Max Features	[3, 4, 5, ..., 21]
	Max Depth	[1, 2, ..., 6]

---

# 05

## Results



# Experiment Set-up



## Cleaning

Dropped the **outliers**; imputed *mean* for missing values.



## Transformation

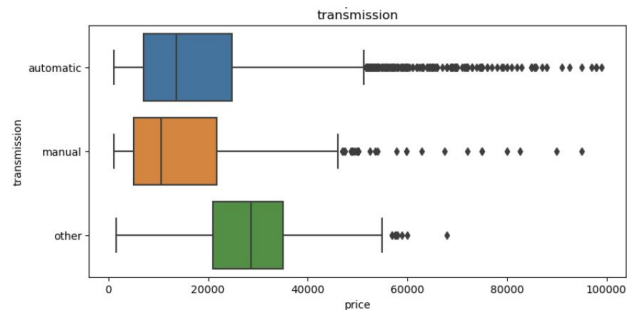
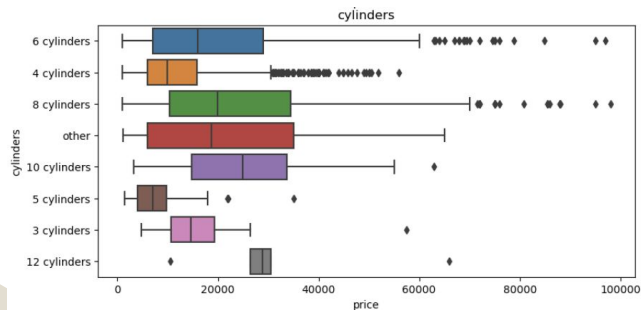
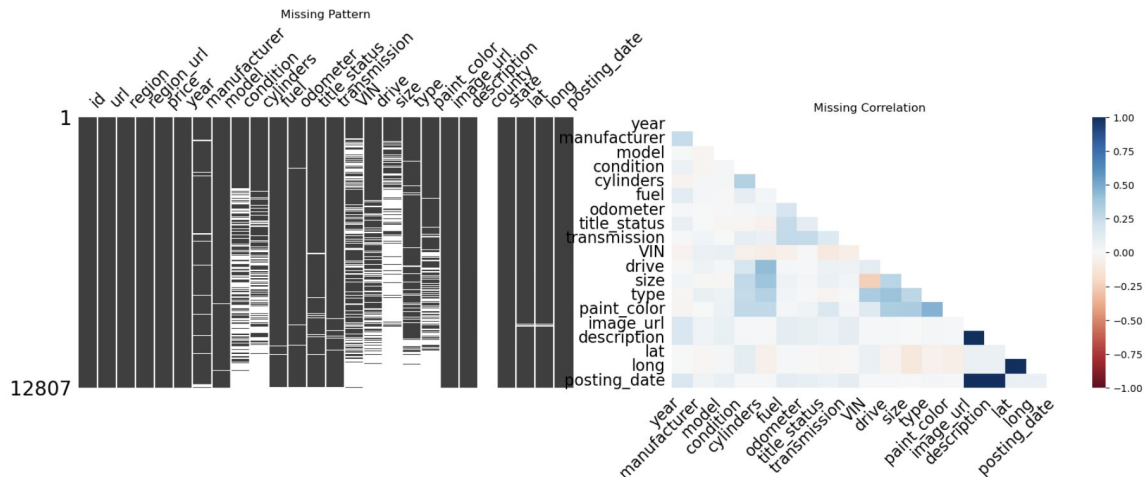
*standardized* numerical variables;  
*one-hot encoded* categorical variables;  
Log-transformed the target variable



## Feature Selection

removed unnecessary features and only kept 12 features: year, manufacturer, condition, cylinders, fuel, odometer, transmission, drive, size, type, paint color, state, and title\_status.

# Experiment Set-up



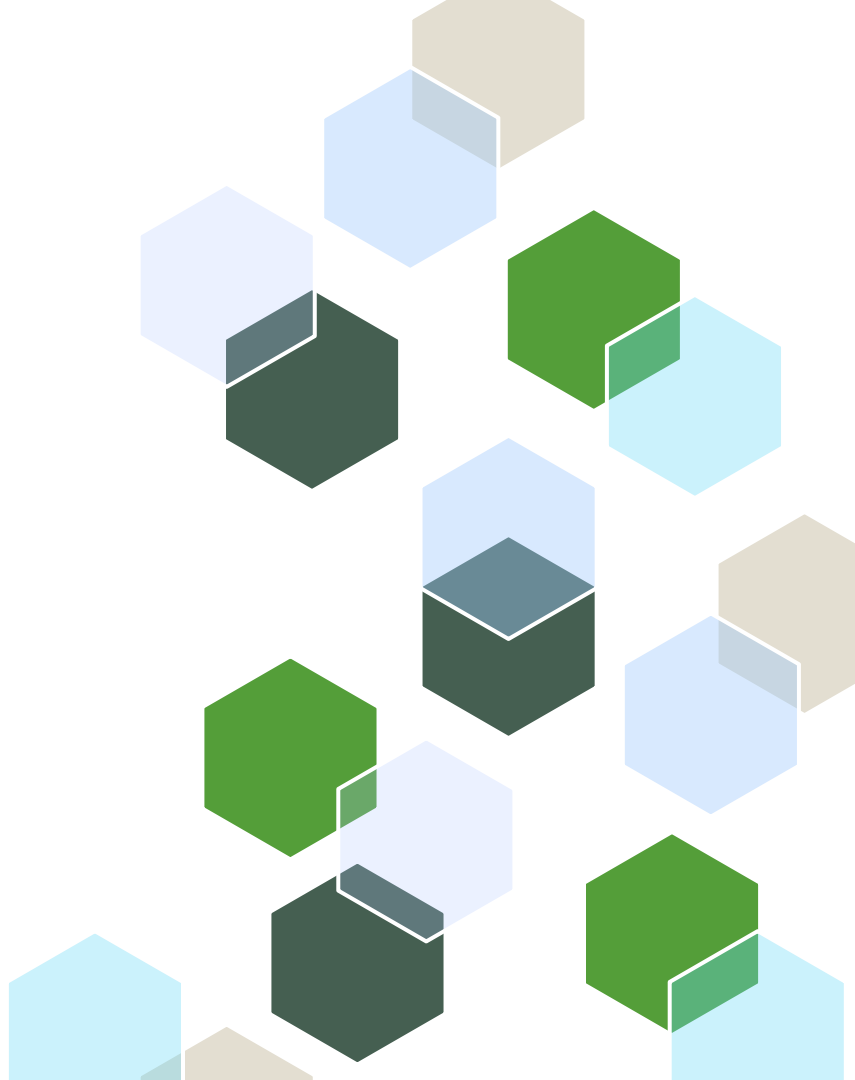
# Model Performance

Model	Hyper-Parameters	R-square
Linear Regression	-	0.57
Decision Tree	Max_depth: 5	0.58
Random Forest	Number of Estimators:100 Max_depth: 9	0.74
Gradient Boosting	Learning rate: 0.1, Loss function: huber, max_depth: 9	0.79

---

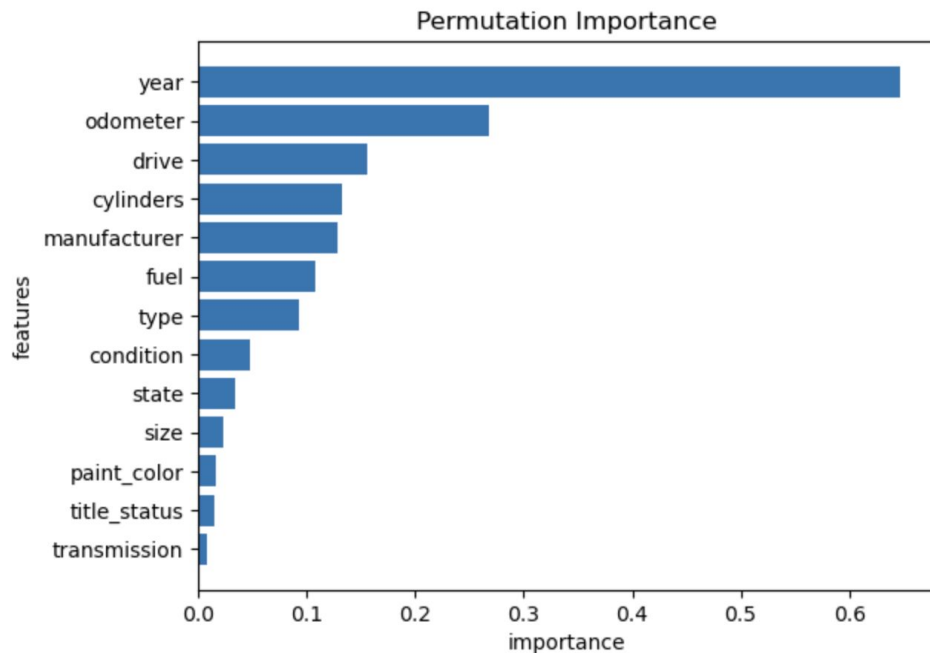
06

# Discussion





# Trade-off between Accuracy and Interpretability



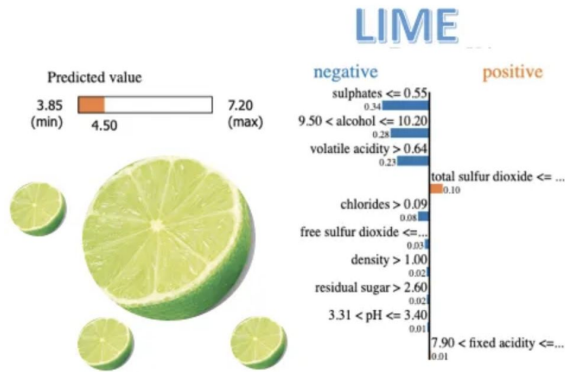
Feature Importance from Gradient-Boosting

```

graph TD
    Root["year <= 114.5  
absolute_error = 11013.502  
samples = 8552  
value = 15995.0"]
    Root --> L1["year <= 110.5  
absolute_error = 7070.546  
samples = 5002  
value = 9500.0"]
    Root --> R1["drive_fed <= 0.5  
absolute_error = 10308.688  
samples = 3550  
value = 27590.0"]
    
    L1 --> L2["cylinders <= 3.5  
absolute_error = 5620.062  
samples = 2684  
value = 6995.0"]
    L1 --> L3["cylinders <= 3.5  
absolute_error = 7518.488  
samples = 2318  
value = 13999.0"]
    
    R1 --> R2["year <= 118.5  
absolute_error = 9980.278  
samples = 2038  
value = 30990.0"]
    R1 --> R3["odometer <= 0.738  
absolute_error = 6088.286  
samples = 881  
value = 17984.0"]
    
    L2 --> L2L["fuel_gas <= 0.5  
absolute_error = 4712.647  
samples = 2178  
value = 5999.0"]
    L2 --> L2R["odometer <= -0.261  
absolute_error = 7066.365  
samples = 786  
value = 10995.0"]
    
    L3 --> L3L["odometer <= -0.191  
absolute_error = 6484.291  
samples = 1685  
value = 12500.0"]
    L3 --> L3R["odometer <= -0.059  
absolute_error = 8691.482  
samples = 353  
value = 23990.0"]
    
    R2 --> R2L["cylinders <= 3.5  
absolute_error = 8539.3  
samples = 1976  
value = 28990.0"]
    R2 --> R2R["cylinders <= 3.5  
absolute_error = 10224.102  
samples = 693  
value = 37990.0"]
    
    R3 --> R3L["cylinders <= 2.0  
absolute_error = 6868.963  
samples = 455  
value = 21989.0"]
    R3 --> R3R["odometer <= -0.316  
absolute_error = 3935.941  
samples = 426  
value = 14987.0"]
    
    L2L --> L2LL["drive_fed <= 0.5  
absolute_error = 3480  
samples = 199  
value = 15500.0"]
    L2L --> L2LR["odometer <= 0.622  
absolute_error = 3954  
samples = 1979  
value = 5950.0"]
    
    L2R --> L2RL["year <= 105.5  
absolute_error = 9549  
samples = 181  
value = 18500.0"]
    L2R --> L2RR["year <= 106.5  
absolute_error = 5588  
samples = 605  
value = 9995.0"]
    
    L3L --> L3LL["drive_fed <= 0.5  
absolute_error = 7144  
samples = 484  
value = 16995.0"]
    L3L --> L3LR["drive_fed <= 0.5  
absolute_error = 5583  
samples = 1201  
value = 10995.0"]
    
    L3R --> L3RL["cylinders <= 6.0  
absolute_error = 7833  
samples = 148  
value = 29990.0"]
    L3R --> L3RR["fuel_gas <= 0.5  
absolute_error = 6799  
samples = 205  
value = 18988.0"]
    
    R2L --> R2LL["odometer <= -0.598  
absolute_error = 8539  
samples = 1636  
value = 27258.5"]
    R2L --> R2LR["odometer <= 0.083  
absolute_error = 8233  
samples = 340  
value = 36545.0"]
    
    R2R --> R2RL["odometer <= -0.874  
absolute_error = 10217  
samples = 567  
value = 36590.0"]
    R2R --> R2RR["fuel_gas <= 0.5  
absolute_error = 9509  
samples = 126  
value = 40635.0"]
    
    R3L --> R3LL["year <= 119.5  
absolute_error = 5073  
samples = 392  
value = 20738.5"]
    R3L --> R3LR["cylinders <= 3.5  
absolute_error = 8988  
samples = 63  
value = 31590.0"]
    
    R3R --> R3RL["cylinders <= 2.0  
absolute_error = 8564  
samples = 214  
value = 16590.0"]
    R3R --> R3RR["year <= 117.5  
absolute_error = 3525.231  
samples = 212  
value = 12772.5"]
    
    L2LL --> L2LLL["absolute <= 0.5  
value = 15500.0"]
    L2LL --> L2LLR["absolute <= 0.5  
value = 15500.0"]
    L2LR --> L2LRL["absolute <= 0.5  
value = 5950.0"]
    L2LR --> L2LRR["absolute <= 0.5  
value = 5950.0"]
    L2RL --> L2RLR["absolute <= 0.5  
value = 18500.0"]
    L2RL --> L2RLR["absolute <= 0.5  
value = 18500.0"]
    L2RR --> L2RRR["absolute <= 0.5  
value = 9995.0"]
    L2RR --> L2RRR["absolute <= 0.5  
value = 9995.0"]
    L3LL --> L3LLR["absolute <= 0.5  
value = 16995.0"]
    L3LL --> L3LLR["absolute <= 0.5  
value = 16995.0"]
    L3LR --> L3LRL["absolute <= 0.5  
value = 10995.0"]
    L3LR --> L3LRR["absolute <= 0.5  
value = 10995.0"]
    L3RL --> L3RLR["absolute <= 0.5  
value = 29990.0"]
    L3RL --> L3RLR["absolute <= 0.5  
value = 29990.0"]
    L3RR --> L3RRR["absolute <= 0.5  
value = 18988.0"]
    L3RR --> L3RRR["absolute <= 0.5  
value = 18988.0"]
    R2LL --> R2LLR["absolute <= 0.5  
value = 27258.5"]
    R2LL --> R2LLR["absolute <= 0.5  
value = 27258.5"]
    R2LR --> R2LRL["absolute <= 0.5  
value = 36545.0"]
    R2LR --> R2LRR["absolute <= 0.5  
value = 36545.0"]
    R2RL --> R2RLR["absolute <= 0.5  
value = 36590.0"]
    R2RL --> R2RLR["absolute <= 0.5  
value = 36590.0"]
    R2RR --> R2RRR["absolute <= 0.5  
value = 40635.0"]
    R2RR --> R2RRR["absolute <= 0.5  
value = 40635.0"]
    R3LL --> R3LLR["absolute <= 0.5  
value = 20738.5"]
    R3LL --> R3LLR["absolute <= 0.5  
value = 20738.5"]
    R3LR --> R3LRL["absolute <= 0.5  
value = 31590.0"]
    R3LR --> R3LRR["absolute <= 0.5  
value = 31590.0"]
    R3RL --> R3RLR["absolute <= 0.5  
value = 16590.0"]
    R3RL --> R3RLR["absolute <= 0.5  
value = 16590.0"]
    R3RR --> R3RRR["absolute <= 0.5  
value = 12772.5"]
    R3RR --> R3RRR["absolute <= 0.5  
value = 12772.5"]
  
```

## Decision-making process from Decision Tree

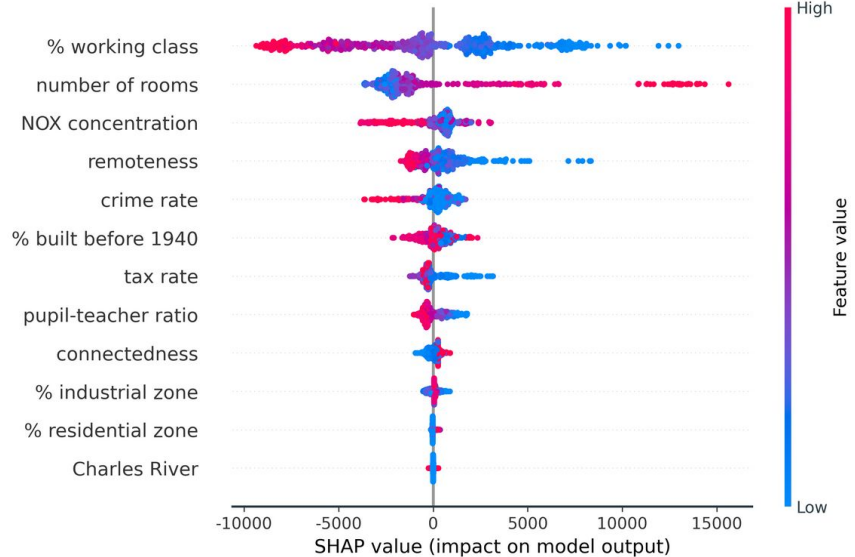
# Trade-off between Accuracy and Interpretability



LIME

(Local Interpretable Model-agnostic Explanations)

Feature	Value
sulphates	0.53
alcohol	9.60
volatile acidity	0.82
total sulfur dioxide	14.00
chlorides	0.10
free sulfur dioxide	5.00
density	1.00
residual sugar	4.10
pH	3.36
fixed acidity	8.10



SHAP

(SHapley Additive Explanations)

# References

- Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.
- Tin Kam Ho. 1995. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.
- Hung Truong Thanh Nguyen, Hung Quoc Cao, Khang Vo Thanh Nguyen, and Nguyen Dinh Khoi Pham. 2021. Evaluation of explainable artificial intelligence: Shap, lime, and cam. In *Proceedings of the FPT AI Conference*, pages 1–6.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011.
- Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.
- Austin Reese. 2021. Used cars dataset. <https://www.kaggle.com/datasets/austinreese/craigslist-carstrucks-data>. Online; accessed 15 June 2024.

# Thanks!

