COMS 4995 - Applied Machine Learning
Used Cars Price Prediction (Team 18)

Jackie Yuan sy2938, Scott Soifer sas2412, Arunit Maity am5689, Prerit Jain pj2383, Yiran Shu ys3373

Abstract – **The used car market is large nowadays, and the used quotes given by car sellers and buyers are manually provided or given with their hidden algorithm. It is not easy for the average person to get an accurate and transparent used car estimation without going to the dealership. This project tries to find an algorithm that can estimate used car prices without human involvement. Additionally, we can discover the essential attributes that can affect used car price listing with interpretable machine learning models.**

I. OBJECTIVE/DATA

This project aims to utilize machine learning techniques with different feature engineering techniques to predict used car listing prices (regression task).

The data in this project is from Used Car Prices, which scraped historical used car information from craigslist websites. The used car listing price datasets contain transmission information, mileage, fuel type, road tax, miles per gallon (mpg), engine size, and car brand.

II. DATA DESCRIPTION

The dataset is readily available on Kaggle and is the collection of prices of the used cars listed on Craiglist.org. It is a collection of .csv files that has size of nearly 1.45 GB. Basic statistics of the dataset is as follows:

- Number of rows: 426,880
- Number of Columns: 27
    - Continuous: 5
    - Categorical: 20
    - Text: 1

The columns' id' and VIN are not required because they are the unique identifiers for the dataset. Similarly, columns url, region URL, and image_url denote the url of the respective columns, so these were dropped from the analysis. Finally, the county column has all values as missing to this was also dropped.

TABLE I
COLUMN DESCRIPTION

Column	Description	Missing Rows
price	Entry Price	0
Region	Listing region	0
year	Entry Year	1,205
manufacturer	Manufacturer of Vehicle	17,646
model	Model of Vehicle	5,277
condition	Condition Of Vehicle	174,104
cylinders	Number of Cylinders	177,678
fuel	Fuel type	3,013
odometer	Miles traveled by vehicle	4,400
title_status	Miles traveled by vehicle	8,242
transmission	Transmission of vehicle	2,556
drive	Type of drive	130,567
size	Size of vehicle	306,361
type	Generic type of vehicle	92,858
paint_color	Color of vehicle	130,203
Id	Unique ID	0
URL	Listing URL	0
Region_url	Region URL	0
VIN	Vehicle Number	161,876
Image_url	URL of image	0
Posting date	Date of Posting	68
description	Listed description of vehicle	70
state	State of listing	0
lat	Latitude	6,549
long	Longitude	6,549

III. EXPLORATORY DATA ANALYSIS / DATA PRE-PROCESSING

Exploratory data analysis is the most crucial step in the model building phase. We have used various techniques such as outlier detection, missing value analysis, univariate, and bivariate analysis to discover the trends and hidden patterns in the dataset. These analyses also helped us transform some variables and strategies for

combining some categories for categorical variables and missing value imputation.

A. *Target Variable*

Price is the target variable in out dataset. The Summary statistics can be found in Fig.1 below.

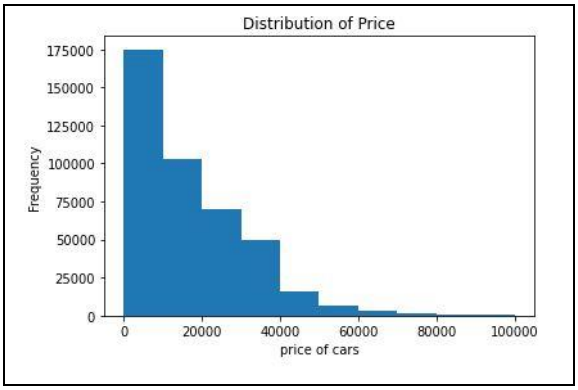

Fig. 1 Histogram of Price

Since some of the used car price listings are not reasonable: some used car prices over millions, we cap to the dataset with the price. We put our price cap to 100,000 since most outliers are outside of 100,000. Furthermore, we found the price distribution of the data is highly right skewed. Thus, we have compared the results using some transformations like log transform to pre-transform the target values and the original target values.

B. *Numerical Variables*

Year – Summary statistics of the Year column are in Fig 2.

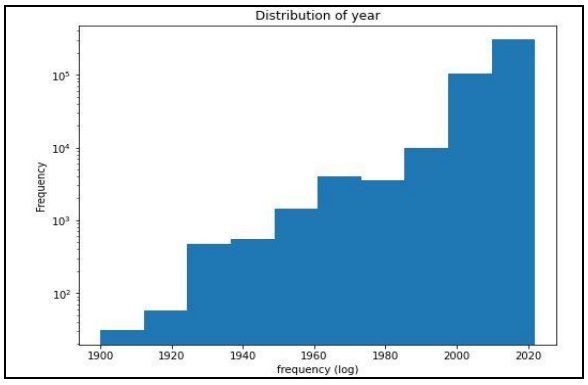

Fig. 2 Histogram of Year

From the distribution of year plot above, we observed that the distribution of year is left-skewed. We observed that most of the cars made year are distributed from 2008 to 2017. We have capped the year in a particular range to ensure the data is close to our real-life scenario. For instance, the max of cars made year is 2022, which is not valid. We have capped our year between 2000 and 2020 since the dataset was updated six months ago.

Odometer – Summary statistics of the Odometer column is in Fig 3.

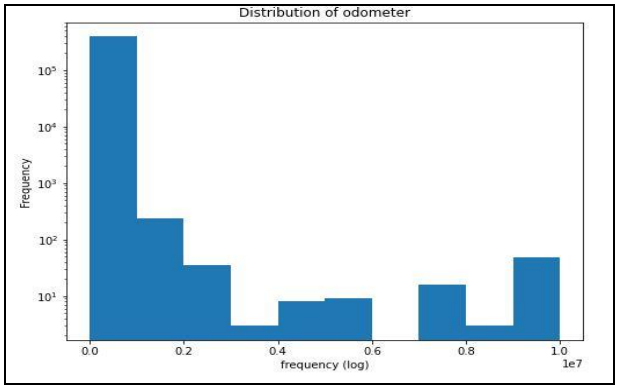

Fig. 3 Histogram of Odometer Reading

There are some outliers in the dataset, and we have capped them. We have also imputed missing values with mean and used standard scaling to the feature before fitting them into the model.

C. *Categorical Variables*

TABLE III
CATEGORICAL VARIABLES STRATEGIES FOR PRE-PROCESSING

Region	404 unique values, combined region, latitude, and longitude into a variable ‘State’ using the US Census Bureau data.
Manufacturer	41 unique values, 4.13% missing values, mode imputation.
Model	XYZ unique values, dropped this column altogether.
Condition	Transform categories from salvage, fair, good, excellent, like new, new, nan into bad, good, new, unknown.
Cylinders	Ordinal Encoding
Fuel	use as it is.
Title_status	use as it is with replacing missing values with ‘unknown’
Transmission	use as it is with replacing missing values with ‘unknown’.
Drive	use as it is with replacing missing values with ‘unknown’.
Size	use as it is with replacing missing values with ‘unknown’.
Type	use as it is.
Paint_color	use as it is.

IV. Model Building

We have the following approaches to the model building phase:

Baseline Model:

Linear Regression is the baseline model; Linear Regression is a linear model with coefficients w to minimize the residual sum of squares between the observed targets in the dataset and the targets predicted by the linear approximation.

Random Forest

Random Forest fits several classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive performance and control over-fitting.

Adaboost

Adaboost begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.

Gradient-boosting Models

Gradient-boosting is an additive model in a forward stage-wise fashion; it optimizes arbitrary differentiable loss functions. In each stage, a regression tree is fit on the negative gradient of the given loss function.

Multi-layer Perceptron

A multi-layer perceptron trains using backpropagation with no activation function in the output layer. In our case, we used mean squared error as our loss function for our multi-layer perceptron.

V. Results

The performance of each technique is depicted in Table III below.

TABLE IV
PERFORMANCE COMPARISON

Technique	**Train R^2 Score**	**Test R^2 Score**
Vanilla Linear Regression	0.4761	0.4663
Decision Tree Regressor	0.5316	0.5342
Random Forest Regressor	0.7228	0.6314
Adaboost Regressor	0.4315	0.4123
Hist Gradient Boosting Regressor	0.8600	0.7755
XGBoost Regressor	0.9406	0.8074
CatBoost Regressor	0.9147	0.7454
MLP Regressor	0.8046	0.6102

VI. Interpretations and Conclusions

We tried out eight different techniques and optimized the model using GridSearchCV and Regularization wherever applicable. The Linear Regression model has the lowest complexity; however, it had a sub-par performance in our scenario compared to models with higher complexity. This might be because we apply various pre-processing strategies such as Ordinal and One-Hot encoding and aggregation of variable values. This shows that the target variable does not have a direct linear relationship with all the columns, and thus we require a more complex model to map the inputs to the target variable. Even the optimized decision tree regressor could only attain a sub-par R^2 score since a single tree algorithm overfits the data. As expected, higher complexity tree-based models such as Random Forest Regressor, Histogram Gradient Boosted Regressor, XGBoost Regressor, and CatBoost Regressor tend to perform much better than linear models. In our case, the best performance was achieved using the XGBoost Regressor with a training R^2 score of 0.9406 and a testing R^2 score of 0.8074. To make the model deployable in a real-world scenario, we decided to choose an XGBoost Regressor with a lesser number of estimators (500) compared to our highest performing regressor with 2000 estimators since the R^2 score difference was marginal (0.03). This was done to reduce the overall model complexity, thereby deeming it appropriate for deployment.