Predictive Modeling on Used Car Prices

University of California San Diego

Justin Phen, Sungjin Choi, Lucas Papaioannou

Introduction

Predictive modeling has gained significant attention in recent years, as it allows us to analyze vast amounts of data and make informed predictions about future events. In the automotive industry, predictive modeling has become increasingly popular for predicting used car prices. With the help of machine learning algorithms, predictive modeling can effectively analyze various factors that affect the price of a used car, such as its make and model, mileage, age, and condition. This paper explores the application of predictive modeling in the used car market, analyzing the factors that influence car prices and the accuracy of various machine learning models in predicting those prices.

The code that this write up is based on can be found here.

1 Dataset

1.1 Dataset Selection

The US Used Car dataset is created by Kaggle user ANANAYMITAL using Cargurus inventory in September 2020. The dataset compreises 3 million real world used car details, with 66 columns as its features. Every row in the dataset represents a car object, and each column contains specific information related to the corresponding vehicle.

1.2 EDA

Basic Statistics

This used cars dataset contains the listings of 3 million used cars gathered from the website CarGurus.com. The columns cover multiple characteristics of the cars. These characteristics include the legroom in the rear seat, the body type of the vehicle, the fuel economy in the city and the highway, how long the listing has been on the market, the height and length of the vehicle, and many others. Essentially every technical aspect of the cars as described in the listing have been recorded.

Data Cleaning

Some columns of the dataset will be excluded immediately as they contain an extremely high percent of missing values. These columns were bed, bed_height, cabin, combined_fuel_economy, is_certified, and vehicle_damage_category.

Some other columns were also excluded immediately as they will not be useful in future predictive tasks. These columns were vin, description, listing_id, and main_picture_url.

In its current form, the dataset has columns that are technically numerical. However, since they are formatted such that they have the units after the numbers, they are unable to be used numerically in a model. These columns include back_legroom, fuel_tank_volume, wheelbase, height, and length among others. In order to fix this, we applied a function to these columns that extracted the numerical values and ignored the units that were in the string. After this cleaning, the columns were now able to be used in a numerical way for models.

EDA Results

Numerical features:

As can be seen below in the correlation heat map in Figure 1, there are many numerical features in this dataset that have a decently strong positive or negative correlation with price. For example, horsepower has a correlation of 0.67 with price, and mileage has a correlation of -0.52 with price. Features that describe the size of the vehicle like height_in and length_in also have a positive correlation with price. Generally, this heatmap lines up with what most people would intuitively think would cause a higher or lower price in a car. However, one thing that was surprising was the correlations of city_fuel_economy and highway_fuel_economy with price being -0.28 and -0.36 respectively. This is surprising as one would initially think that more efficient cars would be more expensive. However, when you think a little further, it becomes apparent that fuel efficiency decreases when an engine has a high number of cylinders and cars with engines with more cylinders are usually more expensive.

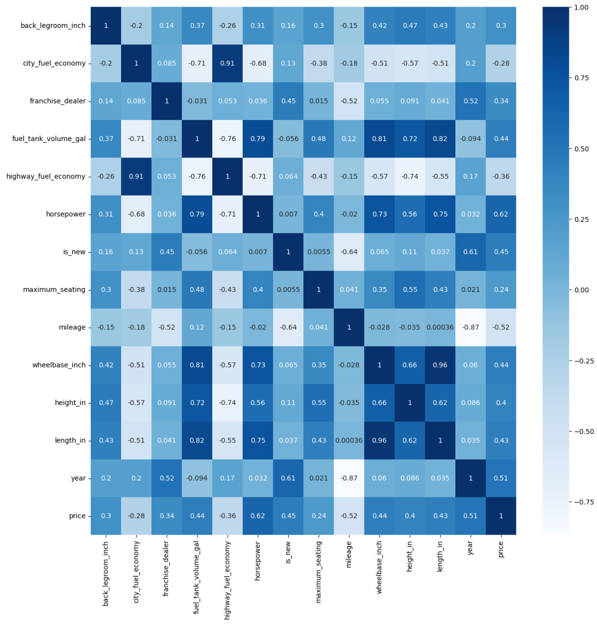

Figure 1: Correlation heat-map of numerical features

body_type: This feature describes the type of car in the listing. Possible values include Sedan, Van, and Convertible. As can be seen below in Figure 2, there are major differences between the mean prices of the body types with Pickup Truck having the highest mean price and Hatchback having the lowest mean price.

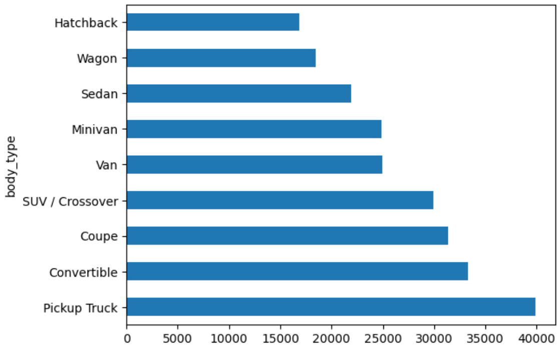

Figure 2: Average price for each body type

engine_type: This feature describes the type of engine the car in the listing has. Possible values include V8, V10, and W12. As can be seen below in Figure 3, there are drastic differences in the average prices for each type of engine. A car with a V12 engine has an average price above 160 000 which is the highest average price while a car with a V8 Propane engine has an average price of around 10 000 which is the lowest average price.

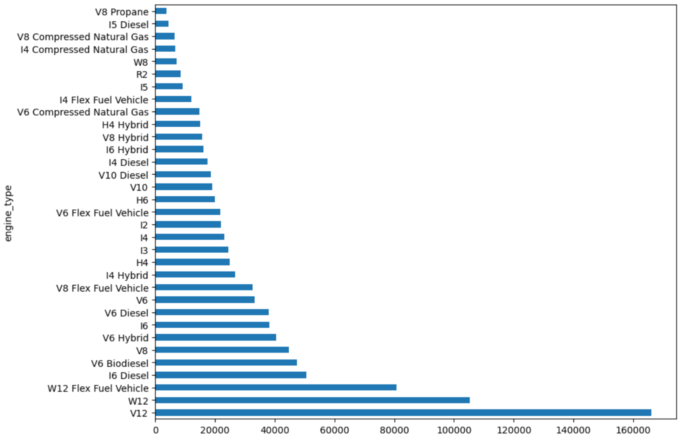

Figure 3: Average price for each engine type

make_name: This feature describes the brand of the car in the listing. Possible values include Jeep, Ford, and Jaguar. As can be seen below in Figure 4, there are drastic differences in the mean prices between brands. Cars that are Rolls-Royce has the highest average price being which is 200 000 and cars that are Daewoo have the lowest average price being which is 2 000.

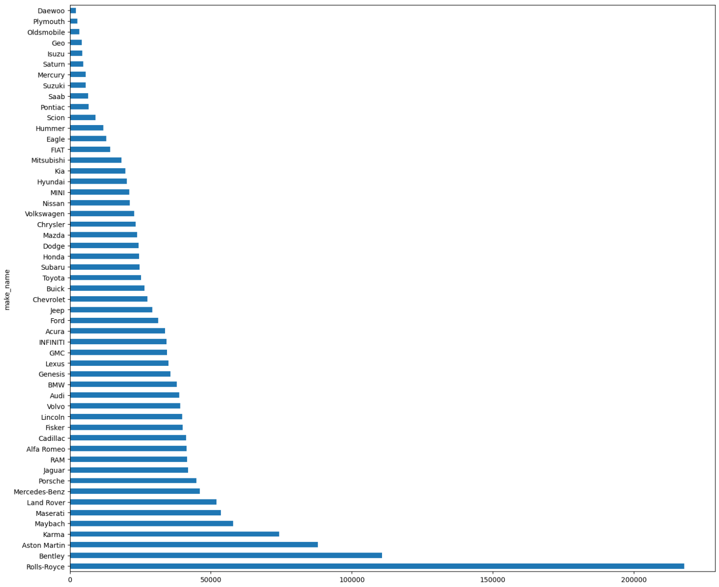

Figure 4: Average price for each engine type

transmission: This feature describes the type of transmission of the car in the listing. Possible values are M, CVT, Dual Clutch, and A. As can be seen below in Figure 5, while the differences of the average prices are not as major as some previous features, there are still some differences. Cars with an M transmission have the lowest average price which is around 20 000 and cars with an A transmission have the highest average price which is just below 30 000.

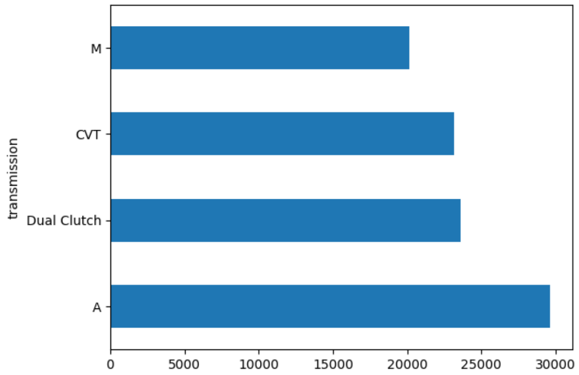

Figure 5: Average price for each engine type

wheel_system: This feature describes the wheel system of the car in the listing. Possible values are FWD, AWD, RWD, 4X2, and 4WD. As can be seen below in Figure 6, there is a big difference in the average price between cars with FWD and cars with 4WD with the average price of cars with FWD being around 20 000 and the average price of cars with 4WD being around 38 000.

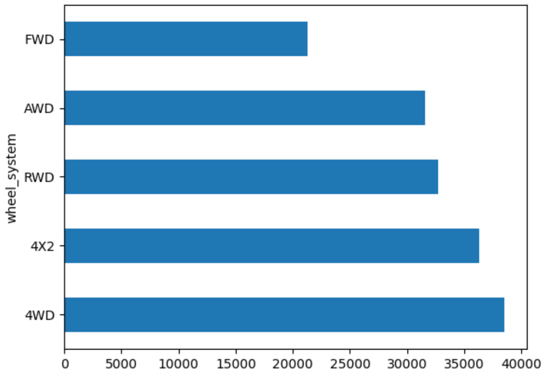

Figure 6: Average price for each wheel system

is_new: This feature describes if the car in the listing has been launched within the past 2 years. Possible values are True and False. As can be seen below in Figure 7, there is a big difference in average price between new and old cars with the average price of new cars being around 36 000 and the average price of old cars being around 22 000.

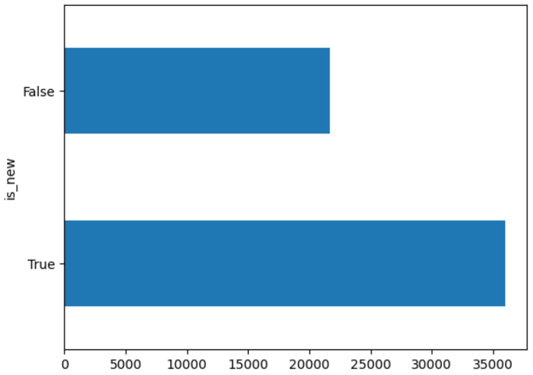

Figure 7: Average price for new and old cars

franchise_dealer: This feature describes if the car in the listing is being sold by franchise dealer or not. Possible values are True and False. As can be seen below in Figure 8, there is a big difference in average price between cars being sold by franchised dealers and cars not being sold by franchised dealers. The average price of cars being sold by franchised dealers is around 31 000 and the average price of not being sold by franchised dealers is around 17 000.

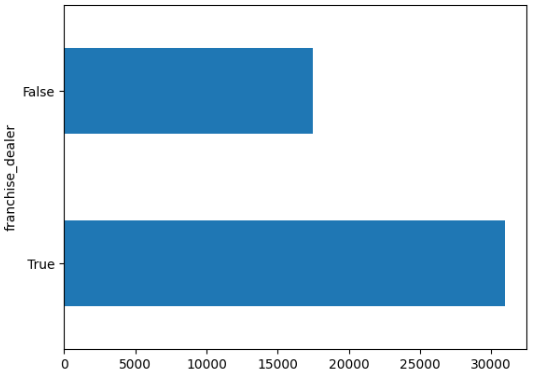

Figure 8: Average price of cars being sold by franchise dealers and non-franchise dealers

2 Predictive Task

2.1 Predictive Task

The predictive task we have decided to tackle is given the characteristics of a used car, and predict the price of the given car. This is a prediction problem that is solved by regression.

2.2 Evaluation

The models trained to solve this predictive task will be evaluated with Root Mean Squared Error (RMSE) and R2 score. RMSE is a commonly used evaluation metric for problems such as this one. Just like Mean Squared Error (MSE), RMSE penalizes larger errors compared to smaller errors as the error is squared. However, in RMSE, the final error value is calculated by taking the square root of the mean of the squared error. This final calculation is essential as it increases the interpretability of the error values as the error is now back in the units of the values being predicted.

In addition to RMSE, R2 score is another useful evaluation metric that will be considered. R2 score, also known as the coefficient of determination, is a statistical measure that evaluates how well the model fits the actual data points. R2 score ranges from 0 to 1, with 1 indicating a perfect fit and 0 indicating that the model's predictions are no better than random.

R2 score is a useful evaluation metric as it provides information on the proportion of variance in the target variable that can be explained by the model. A high R2 score indicates that the model is able to explain a large percentage of the variability in the target variable. However, it is important to note that R2 score should not be used in isolation as it has limitations. For example, R2 score does not indicate whether the model's predictions are biased or unbiased, or whether the model is overfitting or underfitting the data.

Therefore, while evaluating the models, both RMSE and R2 score will be considered to get a more comprehensive understanding of the model's performance. The model with the lowest test RMSE and highest R2 score will be considered the best of the trained models.

2.3 Baseline

The features that will be used in the baseline models will be horsepower, mileage, and fuel_tank_volume_gal (the cleaned up version of the fuel_tank_volume column). These are features that most people would intuitively associate with a higher price if horsepower and fuel_tank_volume_gal' were high and a lower price if mileage was high. Models with these 3 features will be able to be quickly trained and since they are relatively simple, they will make good baselines when compared to more complicated models. The two models that will be used with the above features to make a baseline are a Linear Regression model and a Random Forest Regressor model.

The **Linear Regression model** will be a good baseline as it is resistant to overfitting. However, it only captures linear relationships, so it might result in a big RMSE if it has a nonlinear relationship with price.

The **Random Forest Regression model** will be a good baseline as it is able to capture nonlinear relationships between the features and the price. However, it is susceptible to overfitting if the hyperparameters are tuned incorrectly. To avoid this issue in the baseline model none of the parameters will be changed and will be left as their default value.

The baseline models were trained on the same training set that will be used in training more complicated models. They were also evaluated using RMSE after predicting off the same test set that the more complicated models will be tested on.

2.4 Baseline Results

Model	RMSE	R2
Linear Regression	9121.16	0.484
Random Forest Regressor	5300.36	0.877

Table 1: Results from the baseline models

From the table above, we can see that Random Forest Regressor outperforms the Linear Regression model by 52.99% in RMSE and 57.75% in R-Squared tests.

3 Model

3.1 Feature Selection/Engineering

Given that the dataset comprises 66 columns (features), we realized that the size would pose a significant challenge for model training, resulting in lengthy runtimes. As a result, we employed several feature selection methods. Firstly, we gathered the number of non-null values for each column using the command df.info(verbose=True). We eliminated the 9 columns that had less than 10,000 non-null values in total. The rationale behind this decision was that the missing data in these columns accounted for a substantial portion of our dataset, rendering any attempt to impute or drop the missing values unproductive in terms of deriving useful insights from these columns.

For the second step in our feature selection process, we generated a correlation plot for our current dataframe. Our threshold for column removal was set at less than +/- 0.1 correlation with the price column, which we deemed insignificant for the model training process. However, given that our dataset also includes categorical columns, we were unable to determine their correlation with price. To address this limitation, we selected those columns and conducted a separate test to measure their correlation.

To accurately determine the correlation between categorical and continuous columns, we decided to use OneHotEncoder to transform our categorical columns. We individually tested each categorical column against price, calculating the average correlation of the values for each specific feature. We then applied the same threshold of +/- 0.1 to determine whether to keep or drop the feature. Ultimately, we retained 19 features that we believe will provide adequate information and significant impact in the model training process.

3.2 Model Selection

Similar to our baseline model, we will be mainly testing between 3 models using default parameters: Linear Regression, XGB Regressor, and Random Forest Regressor.

Linear Regression
The linear regression model in sklearn is a popular machine learning algorithm used to establish a linear relationship between a dependent variable and one or more independent variables. The model fits a linear equation to the training data, allowing predictions to be made for new data points. We will be using this model as our baseline model because it is simple to understand, quick to train, and provides a clear benchmark for comparison with more complex models. We believe that the Linear Regression model can serve as a solid foundation for more complex modeling techniques, such as XGB Regressor and Random Forest Regressor.

XGB Regressor
The XGBoost Regressor is a powerful machine learning algorithm that has gained popularity in recent years due to its superior performance in many domains. XGBoost stands for eXtreme Gradient Boosting, and it is an ensemble method that combines several weak prediction models to form a more accurate and robust one. The XGBoost Regressor builds trees iteratively, allowing for the incorporation of new features and the control of overfitting. We will be using the XGBoost Regressor as one of our advanced models due to its ability to handle both linear and nonlinear relationships between the dependent and independent variables. Compared to the simple Linear Regression, given its complexity and nature, we believe that the XGBoost Regressor will significantly enhance our predictive modeling performance and lead to more accurate predictions.

Random Forest Regressor
The Random Forest Regressor is another popular machine learning algorithm that is widely used for predictive modeling tasks. It is an ensemble method that combines multiple decision trees to form a more robust and accurate model. The Random Forest Regressor is particularly useful when dealing with high-dimensional data with complex nonlinear relationships between the dependent and independent variables. It also provides a measure of feature importance, allowing us to identify the most influential features in predicting the outcome variable. We will be using the Random Forest Regressor as another advanced model in our project, as we believe that it will provide a significant improvement over the Linear Regression baseline model. The Random Forest Regressor's ability to handle complex relationships, control overfitting, and feature importance analysis make it a powerful tool for our predictive modeling task.

3.3 Model Testing

For the initial phase of our testing, we will be employing the default parameters for each of the models to compare their performance on our dataset. Prior to training our data, we utilized the train-test-split package within sklearn, specifying a test size of 0.3 and a random state of 42 to split our current dataset into 70 percent training data and 30 percent testing data.
Following the evaluation of results from the three models, we will select the best performing model for our dataset based on our evaluation metric (RMSE and R2). Subsequently, we will perform hyperparameter tuning to further enhance our model's accuracy.

3.4 Evaluation Metric

For each machine learning model on our prediction project, we employed an identical set of categorical and numerical features. Both the Random Forest and XGB regressors perform noticeably better than linear regression. Even though we merely used default values for the hyperparameters for both models, the RMSE and R2 score of the linear regression is 6929.85 and 0.8048, compared to 4266.87 and 0.926 for the XGB regressor and 3951.700 and 0.936 for the Random Forest regressor. We decided to propose a Random Forest regressor and modify hyperparameters for a more effective prediction model as a result of the significant differences in RMSE and R2.

As a result, to predict the price of used cars, our final prediction model employs a Random Forest regressor with hyperparameters (max depth: 25, min samples leaf: 10, min samples split: 2, n estimators: 100) using features (numerical features: 'back legroom inch', 'city fuel economy', 'franchise dealer', 'fuel tank volume gal', 'highway fuel economy', 'horsepower', 'is new', 'maximum seating', 'mileage', 'wheelbase inch', 'height in', 'length in', 'year' and One Hot Encoded categorical features: 'body type', 'engine type', 'make name', 'transmission', 'wheel system').

4 Literature

"Used Car Price Prediction using Machine Learning" by Panwar Abhash Anil
https://towardsdatascience.com/used-car-price-prediction-using-machine-learning-e3be02d977b2

In Anil's project, the price of used cars was predicted using a number of factors. He generated the data for this project from Kaggle, which consists of 26 features and 435849 rows. In order to anticipate the price of used cars, he sought out several significant features. His project's scaled price feature with log transformation is a fundamental component. He noticed that the distribution of price feature is skewed right. A graph resembling the normal distribution is generated by applying the log transformation to the price feature. He utilized RMSLE and R2 score to evaluate the model's accuracy in light of the log transformation.
In order to achieve a better result or fewer errors with high precision, he employed many machine learning models for his prediction project while comparing the essential features for each model. He discovered the essential features in each machine learning model, such as Linear regression, Ridge regression, Lasso regression, KNN, Random Forest, bagging regressor, Adaboost regressor, and XGBoost regressor, are different. He came to the conclusion that the XGBoost regressor exhibits the greatest performance for used vehicle price prediction.
We used his work on the target feature and his method for selecting the optimal machine learning model in our project. We may infer that different machine learning models have different important aspects and that it is required to adjust hyperparameters for each model's most crucial feature, and we can select the machine learning model to be employed based on his project. Also, in contrast to his research, we wished to carry out our prediction project with the use of a greater variety and quantity of features in order for the machine learning model to demonstrate better accuracy.

"Car Predictor USA" by Valcho Valev
https://www.kaggle.com/code/valchovalev/car-predictor-usa/notebook

Our project used the same data set from the "Car Predictor USA" by Valcho Valev. Valev utilizes a dataset from user "AnanayMital" on the Kaggle website with 3 million rows and 65 features. This user built a crawler and used it to browse every used automobile on CarGurus.
Valev was primarily concerned with establishing a connection between pricing and the other 63 features (except price and vin which is index). Valev chose just six numerical features (year, mileage, owner count, highway fuel economy, and latitude) out of 63 columns to focus on. Valev created a prediction model to discover the expected price by selecting one particular selected column for each. Valev filled the NAN values with average values for these columns. Valev applied linear regression for all selected columns vs price, and Valev figured out that year, mileage, owner count, and highway fuel economy are the essential factors to predict the price of used cars.
Following Valev's approach, we looked for a relationship between price and category variables in addition to numerical features. We utilized our significant categorical features on our project with one hot encoding, meanwhile Valev omitted categorical features from his project because they do not appear on the correlation matrix.
We primarily concentrated on both feature engineering and the choice of machine learning models, unlike Valev. Several categorical features were converted to numerical features with dropped units throughout the feature engineering process. Also, we discarded rows that included missing values rather than substituting average values for the missing values, which might result in noisy data. But, since we still had 2.5 million rows of data after removing rows, we decided that dropping missing values was acceptable. In order to better precisely anticipate the price of used cars, we tested multiple machine learning algorithms using the linear regression model as a baseline.
Our findings allow us to draw the conclusion that completing feature engineering, data cleaning, machine learning model selection, and using a larger dataset may increase the accuracy of used car price prediction and develop knowledge of the valuable pattern in the database.

5 Results

5.1 Initial Results

Model	RMSE	R2
Linear Regression	6929.85	0.805
XGB Regressor	4266.87	0.926
Random Forest Regressor	3927.57	0.937

Table 2: Initial results of the 3 models using default parameters

From the results above, we see that the Random Forest Regressor model outperforms both Linear Regression and XGB Regressor in both RMSE and R2. Compared to the original baseline model we had with 3 features, the Random Forest Regressor model outperforms the baseline Linear Regression model by 79.60% in RMSE and 64.33% in R-Squared test, and also outperforms the baseline Random Forest Regressor model by 29.75% in RMSE and 7.25%. Since the Random Forest Regressor model performs the best among our selection of models, we will perform hyperparameter tuning on this model to test how much our model can improve on accuracy.

5.2 Hyperparameter Tuning

To perform hyperparameter tuning on our Random Forest Regressor model, we decided on 4 parameters that we will be experimenting with: max_depth, min_samples_leaf, min_samples_split, and n_estimators. Here, we will perform a grid search using the GridSearchCV package from sklearn. The following are the values we experimented on for the 4 parameters we mentioned earlier.

- max_depth: [5,15,25]
- min_samples_leaf: [1,5,10]
- min_samples_split: [2,5,10]
- n_estimators: [50,100,250]

After performing grid search, we found the best parameters among the values we tested to be {'max_depth':25, 'min_samples_leaf':10, 'min_samples_split':2, 'n_estimators':100}. With these parameters, we get the following result.

Model	RMSE	R2
(Default) RF Regressor	3927.57	0.937
(Optimized) RF Regressor	3755.58	0.943

Table 3: Comparison of model after optimization

After optimization, we find that our model did improve slightly by around 4.56% in RMSE and 0.638% in R-Squared test. Since it was not a significant improvement, we decided to remain with these parameters as it will take way too long to retest a whole new set of parameters, given the size of our dataset.

5.3 Conclusion

Overall, we are very satisfied with our results, with our best performing model achieving a RMSE of 3755.58 while the average of the used car prices in our dataset is around 28328.23. We are able to continously improve our model's accuracy through feature selection and engineering, model selection and optimization with hyperparameter tuning, achieving an improvement of 30% and 80% in RMSE from our baseline models.

References

1. https://towardsdatascience.com/used-car-price-prediction-using-machine-learning-e3be02d977b2
2. https://www.kaggle.com/code/valchovalev/car-predictor-usa/notebook
3. https://www.kaggle.com/datasets/ananaymital/us-used-cars-dataset