CS 229 Project Report: Predicting Used Car Prices

Kshitij Kumbar <kshitjk@stanford.edu>
Pranav Gadre <pgadre@stanford.edu>
Varun Nayak <vunayak@stanford.edu>

Abstract

Determining whether the listed price of a used car is a challenging task, due to the many factors that drive a used vehicle's price on the market. The focus of this project is developing machine learning models that can accurately predict the price of a used car based on its features, in order to make informed purchases. We implement and evaluate various learning methods on a dataset consisting of the sale prices of different makes and models across cities in the United States. Our results show that Random Forest model and K-Means clustering with linear regression yield the best results, but are compute heavy. Conventional linear regression also yielded satisfactory results, with the advantage of a significantly lower training time in comparison to the aforementioned methods.

Motivation

Deciding whether a used car is worth the posted price when you see listings online can be difficult. Several factors, including mileage, make, model, year, etc. can influence the actual worth of a car. From the perspective of a seller, it is also a dilemma to price a used car appropriately[2-3]. Based on existing data, the aim is to use machine learning algorithms to develop models for predicting used car prices.

Dataset

For this project, we are using the dataset on used car sales from all over the United States, available on Kaggle [1]. The features available in this dataset are Mileage, VIN, Make, Model, Year, State and City.

Pre-processing

In order to get a better understanding of the data, we plotted a histogram of the data. We noticed that the dataset had many outliers, primarily due to large price sensitivity of used cars. Typically, models that are the latest year and have low mileage sell for a premium, however, there were many data points that did not conform to this. This is because accident history and condition can have a significant effect on the car's price. Since we did not have access to vehicle history and condition, we pruned our dataset to three standard deviations around the mean in order to remove outliers.

We converted the Make, Model and State into one-hot vectors. Since we had over 2000 unique cities in the dataset, we replaced the string representing the city with a boolean which was set if the population of the city was above a certain threshold i.e. a major city.

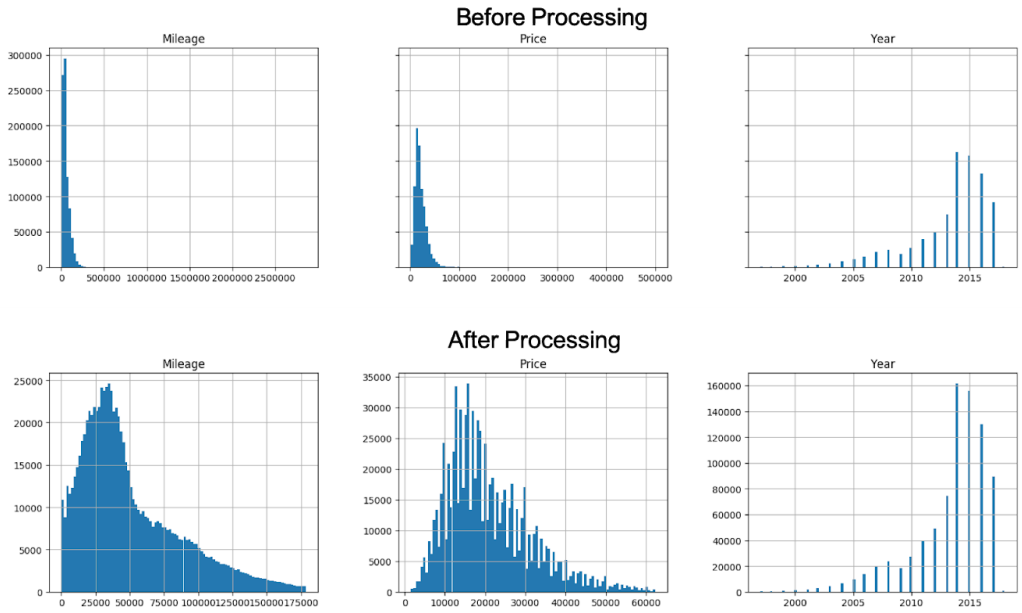

Fig 1: (Top) Histogram for raw data. (Bottom) Histogram after pruning.

Certain features such as VIN numbers dropped during training as these were unique to each vehicle, thus adding no value to training process. Apart from this, while implementing certain Machine Learning frameworks, we realised that certain categorical features had common values which causes problems while using frameworks such as XGBoost which require unique feature names (across all features). For eg: having the string “genesis” in both Make and Model features is not allowed. These common feature names were hence filtered out and renamed to work with these frameworks.

Analysing Linearity in Dataset

To analyze the degree to which our features are linearly related to price, we plotted the Price against Mileage and Year for a particular Make and Model. There seemed to be a fair degree of linearity for these two features.

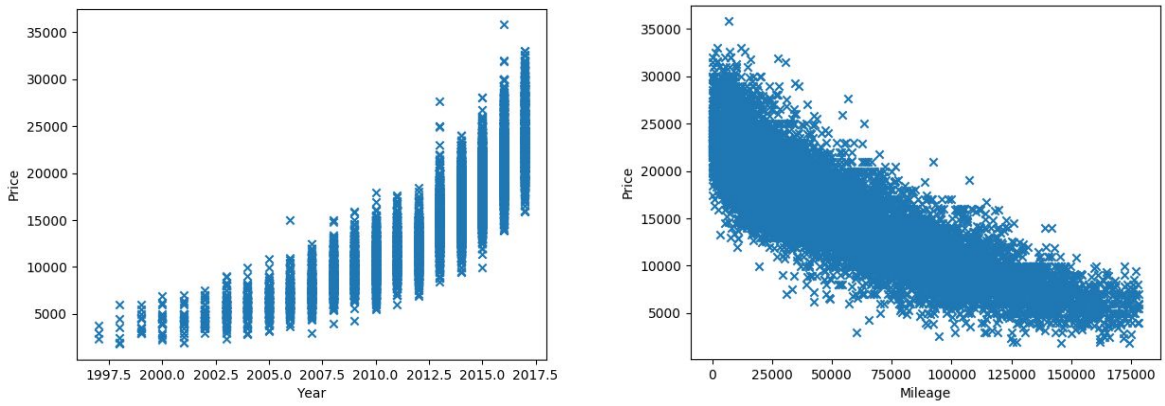

Fig 2: (Left) Price vs Year scatter plot. (Right) Price vs Mileage scatter plot.

Methodology

We utilized several classic and state-of-the-art methods, including ensemble learning techniques, with a 90% - 10% split for the training and test data. To reduce the time required for training, we used 500 thousand examples from our dataset. Linear Regression, Random Forest and Gradient Boost were our baseline methods. For most of the model implementations, the open-source Scikit-Learn package [7] was used.

1. Linear Regression

   Linear Regression was chosen as the first model due to its simplicity and comparatively small training time. The features, without any feature mapping, were used directly as the feature vectors. No regularization was used since the results clearly showed low variance.

2. Random Forest

   Random Forest is an ensemble learning based regression model. It uses a model called decision tree, specifically as the name suggests, multiple decision trees to generate the ensemble model which collectively produces a prediction. The benefit of this model is that the trees are produced in parallel and are relatively uncorrelated, thus producing good results as each tree is not prone to individual errors of other trees. This uncorrelated behavior is partly ensured by the use of Bootstrap Aggregation or bagging providing the randomness required to produce robust and uncorrelated trees. This model was hence chosen to account for the large number of features in the dataset and compare a bagging technique with the following gradient boosting methods.

3. Gradient Boost

   Gradient Boosting is another decision tree based method that is generally described as "a method of transforming weak learners into strong learners". This means that like a typical boosting method, observations are assigned different weights and based on certain metrics, the weights of difficult to predict observations are increased and then fed into another tree to be trained. In this case the metric is the gradient of the loss function. This model was chosen to account for non-linear relationships between the features and predicted price, by splitting the data into 100 regions.

4. XGBoost

   Extreme Gradient Boosting or XGBoost [4] is one of the most popular machine learning models in current times. XGBoost is quite similar at the core to the original gradient boosting algorithm but features many additive features that significantly improve its performance such as built in support for regularization, parallel processing as well as giving additional hyperparameters to tune such as tree pruning, sub sampling and number of decision trees. A maximum depth of 16 was used and the algorithm was run on all cores in parallel.

5. LightGBM

   Light GBM [5] is another gradient boosting based framework which is gaining popularity due it higher speed and accuracy compared to XGBoost or the original gradient boosting method. Similar to

XGBoost, this LightGBM has a leaf-wise tree growth instead of a level-wise approach resulting in higher loss reduction. This framework can also handle categorical features [6], thus eliminating the need to one hot vectorize them and in turn, reducing memory usage. Make, Model and State and cities were declared as categorical features. The algorithm was run at tree depths in multiples of 12 and was run on all cores in parallel.

6. KMeans + Linear Regression

In order to capitalize on the linear regression results and the apparent categorical linearity in the data as indicated in Fig. 2, an ensemble method which used KMeans clustering of the features and linear regression on each cluster was used. Due to large training time, a three-cluster model was used. Then, the dataset was classified into these three clusters and passed through a linear regressor trained on each of the three training sets.

7. Deep Neural Network (MLP Regressor)

To introduce mode complexities in the model, the MLP regressor [7], which uses a deep neural net perceptron regressor model, was used. This model optimizes the squared-loss using LBFGS or stochastic gradient descent. Two hidden layers of width 200 and 20 were used. The learning rate was set at 0.001 and batch size at 200. ReLu was used as the activation function.

Results

The results of our tests were quantified in terms of the R^2 score of our predictions. R^2 score is a statistical measure of how close the data are to the fitted regression line.

Learning Algorithm	R^2 **Score on Test Data**	R^2 **Score on Training Data**	**Training Time**
Linear Regression	0.87	0.87	15 minutes
Gradient Boost	0.64	0.64	130 minutes
Random Forest	0.88	0.98	75 minutes
Light GBM	0.81	0.82	104 seconds
XGBoost	0.78	0.81	180 minutes
KMeans + LinReg	0.88	0.89	70 minutes
Deep Neural Network	0.85	0.85	10 hours

Compared to Linear Regression, most Decision-Tree based methods did not perform comparably well. This can be attributed to the apparent linearity of the dataset. We believe that It can also be attributed to the difficulty in tuning the hyperparameters for most gradient boost methods. The exception to this is the

Random Forest method which marginally outperforms Linear Regression. However Random Forests tend to overfit the dataset due to the tendency of growing longer trees. This was worked upon by restricting the depth of trees to different values and it was observed that beyond limiting depth to 36 resulted in negligible improvement in prediction performance but progressively increased overfitting. As expected lightGBM performed marginally better than XGBoost but had a significantly faster training time.

Building up from the relatively good performance of Linear Regression, the KMeans + Linear Regression Ensemble Learning Method (with K = 3) produced the best R^2 score on test data without high variance as it fits linear relationships categorically. The deep neural network was converging to local minima due to small batch-sizes.

Future Work

For better performance, we plan to judiciously design deep learning network structures, use adaptive learning rates and train on clusters of data rather than the whole dataset. To correct for overfitting in Random Forest, different selections of features and number of trees will be tested to check for change in performance.

Contributions

All team members contributed equally towards this project.
Link to project repository : https://github.com/kshitijkumbar/CS229-Project

References

1. https://www.kaggle.com/jpayne/852k-used-car-listings
2. N. Monburinon, P. Chertchom, T. Kaewkiriya, S. Rungpheung, S. Buya and P. Boonpou, "Prediction of prices for used car by using regression models," *2018 5th International Conference on Business and Industrial Research (ICBIR)*, Bangkok, 2018, pp. 115-119.
3. Listiani M. 2009. Support Vector Regression Analysis for Price Prediction in a Car Leasing Application. Master Thesis. Hamburg University of Technology
4. Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 2016.
5. Ke, Guolin, et al. "Lightgbm: A highly efficient gradient boosting decision tree." *Advances in Neural Information Processing Systems*. 2017.
6. Fisher, Walter D. "On grouping for maximum homogeneity." *Journal of the American statistical Association* 53.284 (1958): 789-798.
7. https://scikit-learn.org/stable/modules/classes.html: Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.