CS230

Predicting used car prices with deep learning

Enoch Li
Department of Computer Science
Stanford University
enochjli@stanford.edu

Bradford Lin
Department of Computer Science
Stanford University
blin1201@stanford.edu

Abstract

Used car pricing today is often varied, based off a relatively small feature set, and is some variant of linear regression. Here, we try various deep learning architectures that incorporate more features in order to better predict used car prices than other models in existing literature today. Specifically, we compare performance between simple linear regression, MLP1, and MLP2 models, both of which are feed-forward artificial neural networks with different setups and hyperparameters. Overall, we find that deeper neural networks with more hidden layers and fewer neurons per layer outperform shallower ones with more neurons per layer, as well as linear regression, based on Mean Squared Log Error.

1 Introduction

When people buy cars new, they often pay at or around manufacturer suggested retail price (MSRP), depending on the dealership. Buying cars new means the pricing strategy is often more transparent and easier for consumers to follow. However, there is an active secondary market for cars, with nearly 70% of Americans saying they would consider a used car as their next auto purchase (PR Newswire, n.d.). Not to mention, people may also choose to auction off their cars through private transactions or on unregulated marketplaces, where pricing is far less transparent. As a result, there is great significance around pricing transparency and accuracy for the used car market.

Unfortunately, used car pricing is not as straightforward as buying a car new, with a number of factors likely playing into its pricing structure, including miles driven, previous accidents, etc. Normally, one can get a used car pricing through an appraisal, but it is important to make sure that the seller is getting a fair price if he/she is selling his/her car to a car dealership or some other party. Likewise, it is equally as important for buyers to be informed of their purchase to ensure they are getting a fair price. Although there currently exist websites that will estimate car prices for users, such as Kelley Blue Book or Edmunds, they often make predictions based on a relatively small feature set, and experts still advise for people to get multiple estimates on their car. Based on our literature review, it also appears that most existing algorithms for car price prediction are some variant of linear regression. Therefore, we see a need to develop a deep-learning algorithm to accurately predict used car prices using standard features such as make, model, mileage, etc.

2 Related Work

Pach (2018) scraped Kelley Blue Book, a website for determining used car prices. Specifically, Pach cleaned the data, ran some exploratory data analysis, and ultimately built a multilinear regression model with ridge regularization, otherwise known as a ridge regression model (Pach, 2018). The accuracy of the Pach (2018) ridge regression model after 10-fold cross validation was 0.262, with his

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

	condition	year	brand	body	color	consumer_review	mileage	city_mpg	highway_mpg	gas_combined_mpg	price	doors	engine
0	1	2015	Kia	Van	Berry	7.9	441.0	18.0	25.0	21.0	28971.0	4	6
1	0	2007	Volkswagen	Convertible	Campanella White	6.9	47771.0	19.0	28.0	22.0	9975.0	2	5
2	0	2014	Cadillac	Sedan	Majestic Plum Metallic	9.2	32066.0	20.0	30.0	23.0	19833.0	4	4
3	0	1993	Chevrolet	Coupe	Green	9.0	67488.0	15.0	22.0	18.0	5928.0	2	8
4	0	2006	Mazda	Van	Grey	7.9	105194.0	19.0	24.0	21.0	4999.0	4	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...
17646	0	2011	GMC	Truck	Charcoal	8.9	52264.0	15.0	21.0	17.0	24995.0	4	8
17647	0	2016	Dodge	Sedan	Charcoal	9.2	15007.0	19.0	31.0	23.0	22339.0	4	6
17648	0	2010	Jeep	Sport Utility	Dark Charcoal Pearl	8.7	55399.0	15.0	19.0	17.0	22999.0	2	6
17649	1	2015	Chevrolet	Sport Utility	Charcoal	7.7	28618.0	22.0	32.0	26.0	22063.0	4	4
17650	0	2010	Jeep	Sport Utility	Dark Charcoal Pearl	8.7	61777.0	15.0	19.0	17.0	21980.0	4	6

16102 rows × 13 columns

Figure 1: Sample data

root mean square log error on the test set being 0.263. For our project, we are using the same data set scraped by Pach (2018), though we hope to achieve better performance metrics through a deep learning framework, rather than using ridge regression.

Additionally, Anil (2021) run a similar framework as Pach (2018), except on a different data set. Anil (2020) conducts more analyses than Pach (2018), including additional machine learning models such as linear regression, lasso regression, K-nearest neighbors, random forest, bagging regressor, adaboost regressor, and XGBoost (Anil, 2021). Anil (2021) finds that XGBoost performs the best, with an accuracy of 89%, far outpacing the 59% achieved via ridge regression. Already, the literature suggests that there are methods and models more powerful for predicting used car prices than the ridge regression developed by Pach (2018), although we should once again note that Anil (2021) did not use any deep learning frameworks.

Lastly, Nawale (2021) also produces his own model for used car pricing predictions, again on a different data set and using random forest as his method of choice. Of course, Nawale (2021) uses different feature selection than both Pach (2018) and Anil (2021). From our literature review, we conclude that there are a number of ways to go about building a price prediction model, even using the same machine learning techniques. Moreover, most existing literature have centered around non-deep learning models for used car price prediction. Where we aim to plug the hole in the literature is to build a deep neural network regression model for used car price prediction and test whether our model performance outshines that of the other regression models currently in the literature.

3 Dataset

We used the dataset prepared by Tai Pach, who scraped the Kelley Blue Book website for 17,000 data points on used car prices (Pach, 2018). We then cleaned the data by converting strings to integers where possible and dropping some features that were likely to have little predictive value, including many with missing values. This left us with a total of 16,102 data points. A snippet of our cleaned dataset can be seen below:

For the categorical variables, such as brand, body, and color, we later encoded them into one-hot vectors to feed into our models. We used sklearn's train_test_split function to create a 90/10 split between train and test sets. After some testing, we ultimately dropped color as a variable as well because it did not add any predictive value (it actually detracted from model performance) due to the fact that every brand names their colors differently (e.g., "arctic racing blue" vs. "denim blue metallic") and keeping the one-hot encoded version of color added an additional 2,000+ columns to our input matrix that did not really add any value.

	mean	std	min	25%	50%	75%	max
condition	0.1	0.3	0.0	0.0	0.0	0.0	1.0
year	2011.7	4.7	1992.0	2008.0	2013.0	2015.0	2018.0
consumer_review	8.5	0.6	1.0	8.1	8.6	8.9	10.0
mileage	55485.0	42764.7	1.0	23080.8	41779.5	82250.8	343153.0
city_mpg	20.7	4.9	10.0	17.0	20.0	25.0	51.0
highway_mpg	28.6	6.3	14.0	24.0	28.0	34.0	48.0
gas_combined_mpg	23.6	5.3	11.0	19.0	23.0	28.0	50.0
price	16614.8	13389.2	500.0	9991.0	13274.5	19991.0	482000.0
doors	3.7	0.7	2.0	4.0	4.0	4.0	5.0
engine	5.2	1.4	1.0	4.0	4.0	6.0	8.0

Figure 2: Statistical distribution

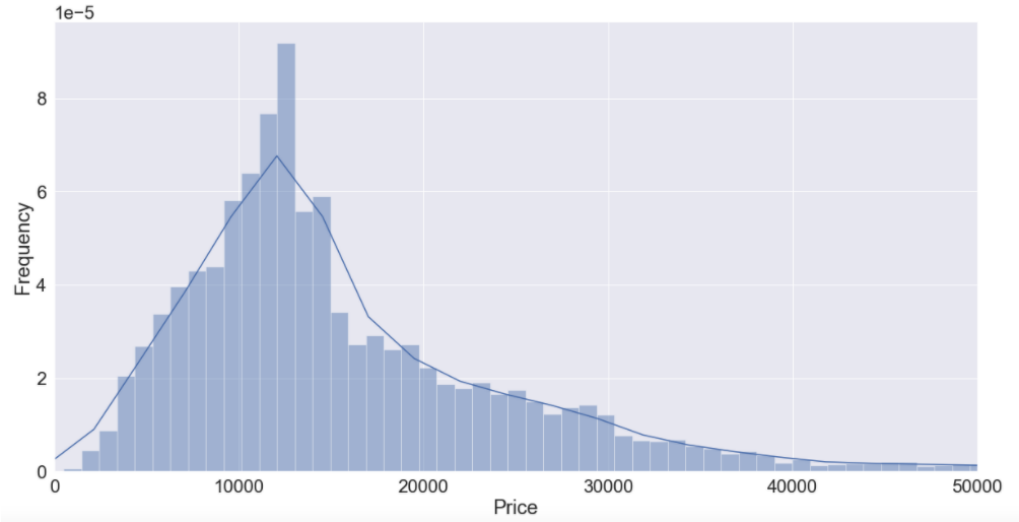

Figure 3: Price distribution

The statistical breakdown of our numerical variables can be referenced in Figure 2, while the histogram of prices, our target variable, can be seen below in Figure 3.

4 Methods

Pach (2018), who was the original author of the data set, ran ridge regression as a means of prediction of used car prices from the Kelley Blue Book data. For our project, we plan to predict used car prices using neural networks, and we plan to compare our results against our own regression models as a baseline. Since we are using the same data set as Pach (2018), we also compare our results against Pach's ridge regression to see how performance is affected by model choice.

In terms of models, we build, train, and run three models: linear regression and two multilayer perceptrons, one turnkey solution from sklearn (which we refer to as MLP1 going forward) and one customized using Tensorflow's Keras framework (which we refer to as MLP2 going forward).

4.1 Linear Regression

For our linear regression, we used sklearn's linear regression model with a 90/10 train/test split on our dataset. We did not change any of the default variables for the model, but forced all the predictions to

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 20)                1340
_________________________________________________________________
dense_1 (Dense)              (None, 16)                336
_________________________________________________________________
dense_2 (Dense)              (None, 8)                 136
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 9
=================================================================
Total params: 1,821
Trainable params: 1,821
Non-trainable params: 0
_________________________________________________________________
```

Figure 4: MLP2 model summary

be at least zero in order to be able to calculate the mean squared log error, since this metric requires that all predictions are positive.

4.2 MLP1

For our first neural network regression, we used sklearn's MLPRegressor model with a 90/10 train/test split on our dataset. This model consisted of a single hidden layer with 100 neurons and a ReLU activation, which forces non-negative predictions as well. We optimized using Adam for squared error loss, which was the default loss function of the model that we could not change.

4.3 MLP2

For our customized neural network, we use Keras' architecture with a 90/10 train/test split on our dataset. Specifically, after sufficient experimentation and hyperparameter tuning, we settled on a model with three layers and forty five neurons. We optimize using Adam for mean squared log error (MSLE) since that is the metric we are using for our project. We plan on using Mean Squared Log Error (MSLE) to evaluate the accuracy of our predicted car values, since this measures the percentage difference rather than the absolute difference, which Mean Squared Error (MSE) would measure. Percentage difference is a better metric, because $1,000 error can mean a lot more on a lower MSRP car than a very high MSRP car (i.e. a Toyota Camry vs. a Lamborghini Aventador).

A summary table of our model is displayed in Figure 4.

5 Results and Discussion

Model	Train MSLE	Test MSLE
Linear Regression	2.368	2.605
MLP1	0.159	0.160
MLP2	0.079	0.080
Pach		0.069

For each model, we report training and test error on our data, using a 90/10 train/test split. We see that linear regression performs the worst, by far, followed by MLP1 and then MLP2. All three of our models perform worse than the ridge regression put forth by Pach (2018), though our MLP2 performance approaches that of Pach (2018) and can be considered comparable. For reference, we show the plot of our training loss for our best-performing model, MLP2 (Figure 5).

Looking at why our different models perform the way they did, we can first see that linear regression performed the worst of all our models. We hypothesize that the problem of predicting used car prices is not a linear problem, and so a simple linear regression without any modifications or variable transformations would easily perform the worst.

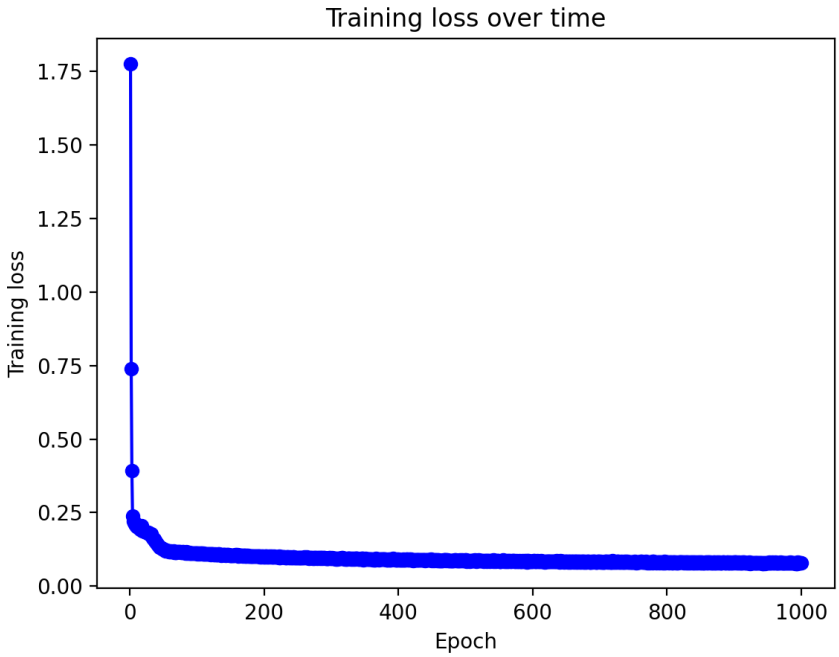

Figure 5: MLP2 training loss over time

Although MLP1 already performs quite well and represents a marked improvement over simple linear regression, we hypothesize that MLP1 under-performs MLP2 for a few main reasons. First, MLP1 optimizes for mean squared error as opposed to mean squared log error, so it thereby follows that MLP2 should perform better than MLP1 when measuring by mean squared log error. Furthermore, MLP1 only has one hidden layer, albeit with 100 neurons. MLP2 has fewer total neurons but more hidden layers. Given that MLP2 has a greater number of hidden layers than MLP1, it can better learn non-linear features within the dataset. Given that we believe used car price prediction to be a non-linear problem, we hypothesize the depth of MLP2 is another reason why it outperforms MLP1.

As for why our MLP2 model approached the performance of the ridge regression run by Pach (2018) but did not beat its performance, we hypothesize that Pach (2018) optimized for root mean squared log error, as opposed for mean squared log error like we did. Even though square root is a monotonic transformation, ridge regression includes penalties for regularization, so the optimization solution may not be the same. Moreover, Pach (2018) removed outliers from his data and tested on the trimmed dataset, so it is quite possible that our model would have performed better than his on the full dataset that we trained on.

6 Conclusion

After building, training, and tuning a linear regression and two deep learning models, we achieve comparable performance to that of existing literature when it comes to predicting used car prices. We found better performance with deeper artificial neural networks with more layers and fewer neurons per layer, a finding that future studies can consider.

In the future, we would expand upon our current research via further hyperparameter tuning, and some other machine learning techniques beyond deep learning, including random forest regression and some of the others used in the literature. We might also introduce regularization in our loss function, similar to what Pach (2018) considered.

7 Contributions

Both EL and BL contributed to the project equally. EL conducted the data cleaning, and built the linear regression and MLP1 models. BL created, tuned, and ran the MLP2 model, as well as conducted the literature review and produced the figures for the report. Both EL and BL contributed to the final report and final video presentation.

References

[1] Majority of americans would consider buying used vehicles. (n.d.). https://www.prnewswire.com/news-releases/majority-of-americans-would-consider-buying-used-vehicles-300967731.html

[2] Pach, T. (2018). Kelley blue book project. GitHub. Retrieved October 2, 2021, from https://tai-pach.github.io/kbb/

[3] Anil, P. A. (2021, April 26). Used car price prediction using machine learning. Medium. https://towardsdatascience.com/used-car-price-prediction-using-machine-learning-e3be02d977b2

[4] Nawale, T. (2021, April 7). Used carprice prediction -complete machine learning project. Geek Culture. https://medium.com/geekculture/used-carprice-prediction-complete-machine-learning-project-d25559cf2d2a

[5] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2.

[6] Developers, TensorFlow. TensorFlow. Zenodo, 2021. Zenodo, https://doi.org/10.5281/zenodo.5637331.

[7] Keras: The Python Deep Learning API. https://keras.io/. Accessed 4 Nov. 2021.

[8] Scikit-Learn: Machine Learning in Python — Scikit-Learn 1.0.1 Documentation. https://scikit-learn.org/stable/index.html. Accessed 4 Nov. 2021.