Udacity Capstone Project: "Becoming a machine learning engineer"

Topic: Can we predict a used-car price with one variable?

1.) Definition

Domain background:

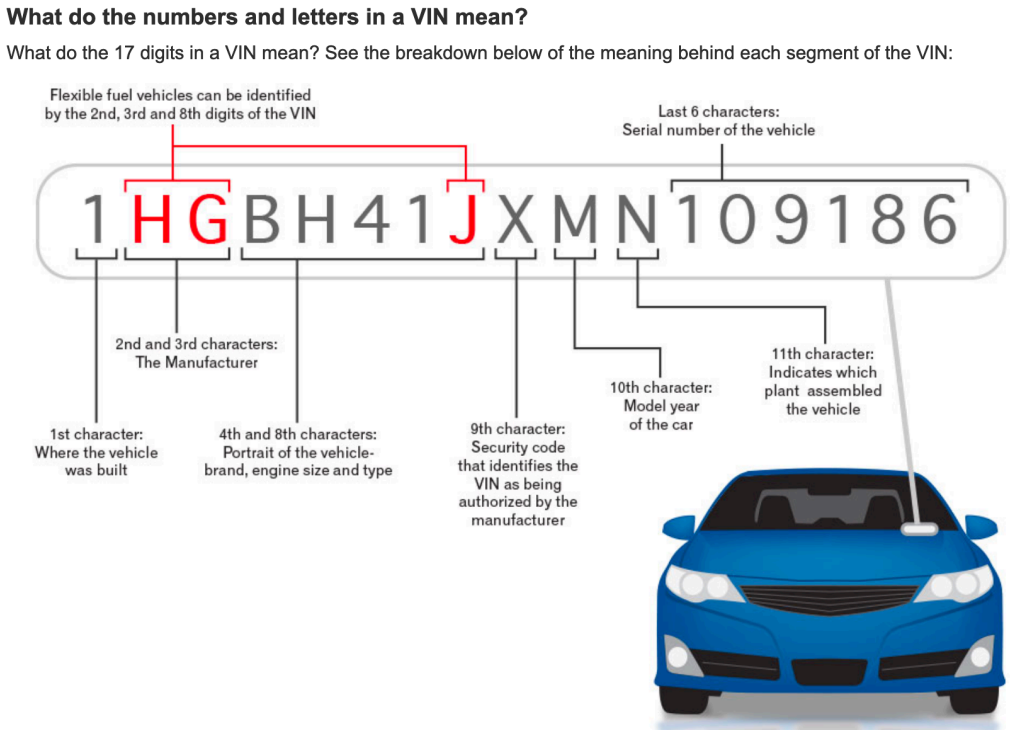
Assessing the price of an asset is a typical machine learning use case. Based on economic theory, a (market) price is determined by the relationship between the supply and the demand for that asset. I would like to dig deeper into one of the industries where price prediction is especially complex: the used-car market. In the used-car market, price prediction is inherently complex given that each used car is unique (you don't find two identical used cars).

Problem statement:

If you browse through Kaggle there are a number of machine learning approaches that aim to predict the price of a used car.

So how does my approach differ?

I am focusing especially on one feature where I think it has a lot of explanatory power, but was not used in prior approaches that aimed to predict the price of a used car. So which feature can it be? **THE VIN. The unique Vehicle Identification Number.** You might wonder how an identification number could be an important feature for price prediction. The reason is the following:



Source: https://www.autocheck.com/vehiclehistory/vin-basics

In fact, the 17-digit VIN could be a powerful feature that reveals the model year, brand, engine size, type, etc. – all features that are assumed to considerably affect the price of a car.

I will thus, address the following question: How much of the used-car price variation can be explained solely by the 17-digit VIN.

Dataset and inputs:

The dataset includes approximately 800,000 used-car prices scraped from www.truecar.com. It is available on Kaggle (https://www.kaggle.com/jpayne/852k-used-car-listings).

Problem statement:

I will use the listed price of the car as the target variable. I will derive features from the VIN variable. For modelling the relationship between the VIN and the listing price, I will use a Neural Network.

One of the most important parts in this respect is the encoding of the VIN variable. Given that the VIN contains so many different information, one-hot encoding might not be a good strategy (keep in mind that the VIN contains 17 digits, each digit could take on numerical or alphabetical values, meaning we would have 17 * 35 categories). Instead I will apply embedding, meaning that I will include an embedding layer which learns categories from the VIN categories.

Benchmark model:

A comparable ML approach for predicting used-car listing prices (https://www.kaggle.com/vbmokin/used-cars-price-prediction-by-15-models) achieved maximum R-squared values of up to 80% on the test set. RMSEs for this kind of prediction problem on Kaggle are somewhere in the area of 8,000 € (meaning that for a 30,000€ car, you would expect an average prediction in the [22000 €; 38,000€] interval).

Given that we have substantially less information available, we would be happy with somewhat lower R-squared / RMSE values.

Evaluation metrics:

The main metric I would like to focus on is the Mean Absolute Error (MAE). Given that my dataset will contain a lot of outliers (customers can list their car at whatever price they one) that should not be penalized too strongly, I will not focus on on MSE / RMSE.

As a secondary metric, I will take a look at the R-squared of my predictions, which takes into account the MSE of our predictions.

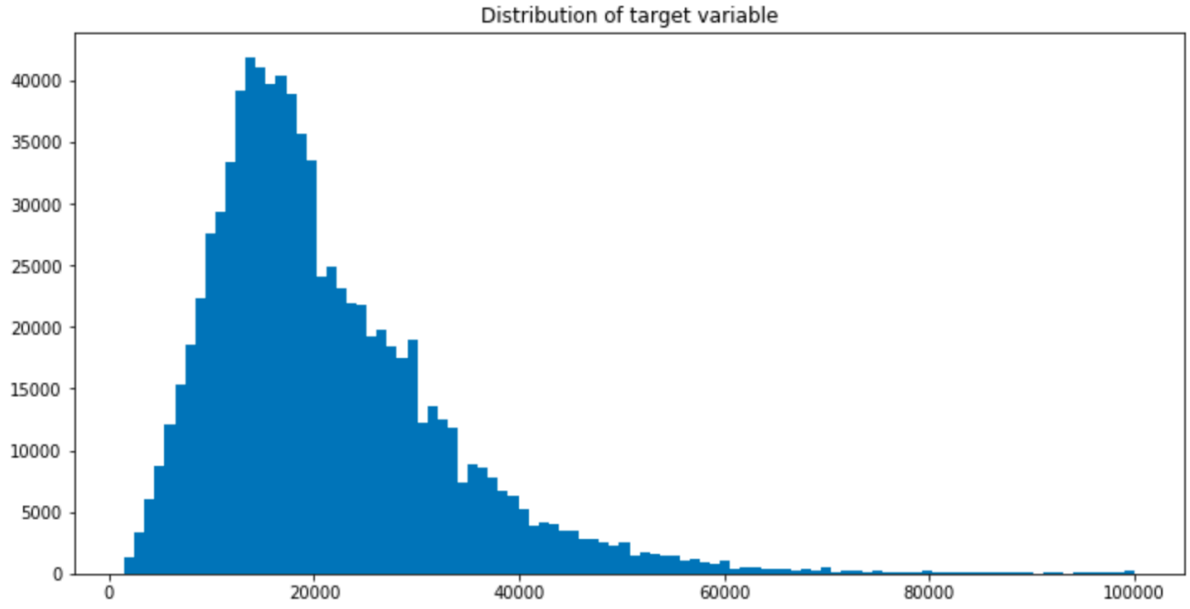
Project design:

1) EDA of the relationship between VIN features and price
    a. Split VIN into its sub-features and see distribution of them (how many different manufacturers, fuel_types, etc. captured by the VINs in our sample)
    b. See which VIN features have most explanatory power with respect to listing price (e.g., grouping, target encoding, visualizations)
2) Train Neural network (optimize hyperparmeters with cross-validation)
    a. Plan to use 3 layers:
        i. Embedding layer
        ii. Hidden layer
        iii. Final layer (output variable might be categorical (price) / or categorical (grouping prices)
3) Analyses of test set results

2.)Analysis

Data exploration / Exploration Visualization:

As this is a very data-driven project and focuses on only one variable, there is not so much opportunity for data exploration.



The distribution of the target variable is somewhat close to a normal distribution but has a strong right skew, given that prices have a lower bound (zero), but no upper bound. As a benchmark for future predictions, I wanted to see how large our prediction error would be, if we randomly draw the target price, from a normal distribution (with parameters as observed by the real target variable). In we naively predict (by drawing randomly from the distribution) our MAE would be **13,933** €. So if the VIN has informational value, we should substantially reduce the MAE with our predictions.

After exploring the individual components, that are part of the VIN, I came to the conclusion that there are 4 components that could be useful for our prediction task.

- The built year of the car:
    - This part of the VIN (digit-8) can be directly mapped to built year and be used as a numerical variable.
- The built_location of the car:
    - 23 unique manufacturers
- The manufacturer of the car:
    - 229 unique manufacturers
- The model_engine_fueltype combination of the car:
    - 12671 unique categories

Algorithms and Techniques:

After the data exploration part, it becomes clear that the largest problem is the vast amount unique of categories of our explanatory variables. We could not one-hot encode them, as this would lead to ~ 13,000 features.

Instead, I think that it makes sense to train a neural network with an embedding layer. The embedding layer (or feature layer) will map the categorical variables to dense vectors consisting of lower dimensions (compared to one hot encoding them). Thus, we will use a neural network for this prediction task.

Benchmarks:

- MAE of naïve prediction: 13,933€
- R-squared of comparable project (https://www.kaggle.com/vbmokin/used-cars-price-prediction-by-15-models): approximately 80%

3. Modelling

Data preprocessing:

- Mapping of the VIN to 4 different variables (as described above)
- Exclusion of rows where:
    - Feature variables are missing
    - Feature variables cannot be mapped (e.g., for some cars the built year could not be derived from the VIN)
    - Car price above 100,000 € (extreme outliers as these prices are unusual for used cars)
- Create input pipeline with Tensorflow:
    - Translate dataset to feature columns

Model training / Implementation and refinement:

- Main parameters and their best setting:
    - Number of epochs: 10
    - Batch size: 200

- Dimension of embedding features: Power(original_categories, 0.6)
- Layers:
  - Feature layer size:
  - Dense layer 1 size: 128
  - Dense layer 2 size: 64
  - Output layer size: 1
- Optimizer: Adam

→ The most sensitive parameters were the optimizer and the dimension of the embedding features. After playing around with them, I got the best MAE result, with the parameters as set above.

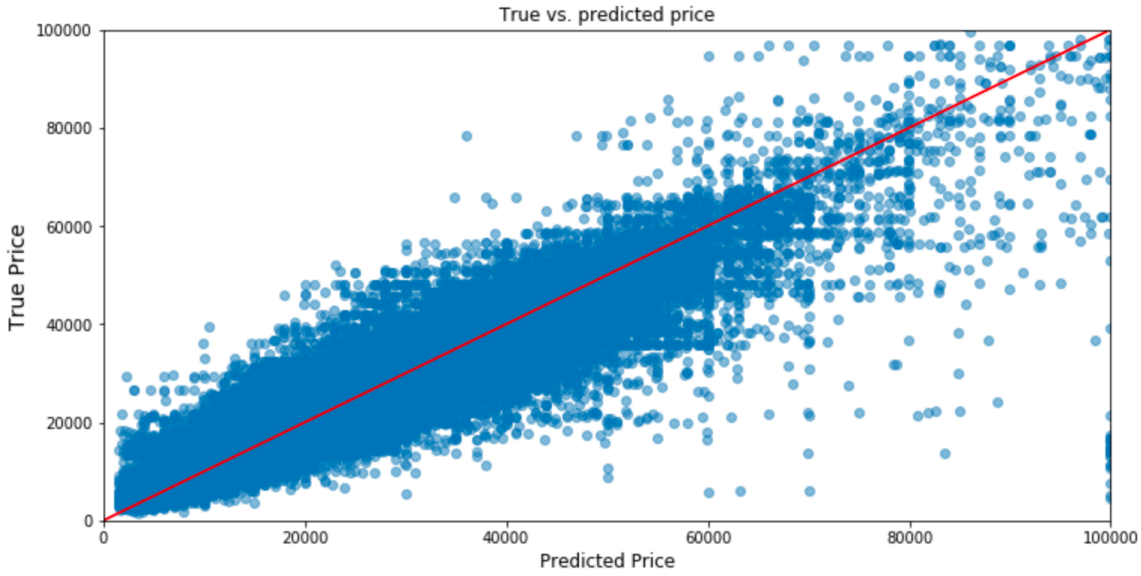
→ Please note again that the aim of this project is not to maximize the metrics, but do get a feeling for how much information value there is in the VIN

As you see below, the MAE converges at roughly 2,700 €

```
2727/2727 [==============================] - 91s 33ms/step - loss: 5041.0041 - mae: 5041.1655 - val_loss: 0.0000e+00
- val_mae: 0.0000e+00
click to scroll output; double click to hide
2727/2727 [==============================] - 83s 31ms/step - loss: 3165.8171 - mae: 3166.0005 - val_loss: 3078.7759 -
val_mae: 3079.0676
Epoch 3/10
2727/2727 [==============================] - 88s 32ms/step - loss: 2999.5091 - mae: 2999.4944 - val_loss: 2987.6181 -
val_mae: 2987.8203
Epoch 4/10
2727/2727 [==============================] - 89s 33ms/step - loss: 2920.8757 - mae: 2921.0645 - val_loss: 2930.0140 -
val_mae: 2930.2329
Epoch 5/10
2727/2727 [==============================] - 90s 33ms/step - loss: 2872.1121 - mae: 2872.0107 - val_loss: 2900.0714 -
val_mae: 2900.2166
Epoch 6/10
2727/2727 [==============================] - 87s 32ms/step - loss: 2840.2453 - mae: 2840.2717 - val_loss: 2883.6121 -
val_mae: 2883.7815
Epoch 7/10
2727/2727 [==============================] - 89s 33ms/step - loss: 2816.8771 - mae: 2816.9155 - val_loss: 2845.7364 -
val_mae: 2845.8892
Epoch 8/10
2727/2727 [==============================] - 88s 32ms/step - loss: 2796.7402 - mae: 2796.7168 - val_loss: 2835.7607 -
val_mae: 2835.9258
Epoch 9/10
2727/2727 [==============================] - 90s 33ms/step - loss: 2783.5055 - mae: 2783.3408 - val_loss: 2833.1728 -
val_mae: 2833.3601
Epoch 10/10
2727/2727 [==============================] - 89s 33ms/step - loss: 2774.2281 - mae: 2774.2620 - val_loss: 2812.9997 -
val_mae: 2813.1584
```

Results:

- R-squared of 85% (and thus better than 80% of comparable Kaggle projects that use way more variables)
- MAE of 2750 € and thus much better than naïve prediction (which had close to 14,000 € MAE)



Conclusion / Reflection / Improvement:

- Taking only the information from one variable (VIN), we are able to predict 85% of the variance of used car prices (R-squared on test set)
- Compared to a naive (random) prediction, which yields an MAE of 14,000 USD, we decrease the MAE to below 2,800 USD
- Thus, it looks like there is important informational value in the VIN that could be used to improve existing used car models
- The model could certainly be further optimized, but this is not the aim of this capstone. Instead, we wanted to get a general feeling for how much informational value the VIN contains