Car price prediction: Summary

May 24, 2021

1 Motivation and problem definition

The clients (salesmen) are in need of reasonable suggestions on car prices based on their specifications. For this purpose, the automotive group of the company are handed over the data of used car sales to us, which can be utilized to make suggestions.

The goal is to train a model to suggest the price of the car. This can be achieved by building and training a machine learning model. The data to be used for this has following specifications/features:

Feature name	Description	Datatype
`make`:	maker of the car (BMW, Toyota etc.)	category (`str`)
`model`:	model of the car	category (`str`)
`vehicle_year`:	year when car was manufactured	continuous (`int`)
`mileage`:	the total distance travelled by the car since its manufacture	continuous (`float`)
`engine_capacity`:	the cylinder volume swept by all of the pistons of a piston engine, excluding the combustion chambers in cubic centimeters (cc)	continuous (`float`)
`engine_power`:	units of horsepower	continuous (`float`)
`gearbox`:	type of transmission	category (`str`)
`fuel_type`:	type of fuel (e.g. diesel or petrol)	category (`str`)
`damaged`:	is vehicle damaged (1) or not (0)	category (`Bool`)
`is_business`:	the seller is a business (1) or an individual (0)	category (`Bool`)
`target_price`:	price of the car in Złotys	continuous (`float`)

We wish to predict the dependent variable `target_price` based on other features (independent variables).

2 Model deployment

We opt for a *Random Forest Regressor* model to be trained. The data exploration gives us an idea that the functional dependency of `target_price` on the independent variables is fairly non-linear, which makes it a feasible candidate model. `RandomForestRegressor` method from Scikit-learn module is used. Random Forest is a universal machine learning technique which can model data of mixed category features. It has fewer statistical assumptions compared to other models, needs less feature engineering, and can also work well on large datasets. Along with the machine learning model, it is important to determine the metric to be used for assessing the model performance. Here we use *RMSLE (Root-Mean-Squared-Log-Error)* between the actual and predicted car prices.
For evaluation purposes, we split the data into separate training and validation data sets. The validation dataset size is 50000 entries and rest is the training dataset, i.e., 156207 entries. After the model learns using the training data, its predictions are expressed in terms of validations score (R^2) and RMSLE.

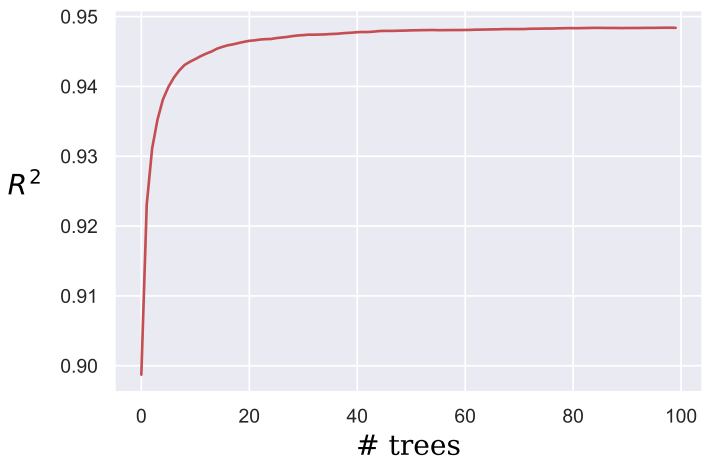

Figure 1: Validation score (R^2) of random forest regressor as a function of the number of estimators (trees)

3 Model evaluation

On evaluation, for a default set of hyperparameters (100 fully grown trees till the last leaf), the model gives a $R^2 \approx 0.95$ and RMSLE of 0.23. The effect of number of trees (estimators) in determining R^2 of the model was calculated, which is shown in Fig. 1. It can be seen that R^2 reaches a plateau at high number of trees. In other words, the number of trees set to 40 will result in approximately same R^2 as that for 100 trees. For more details on the model evaluation, please refer to the Jupyter Notebook attached herewith.

4 Challenges and further actions

- The validation score of the current random forest model can be improved by finding the optimum set of hyperparameters using GridSearch or RandomSearch methods and considering the Out-Of-Bag (OOB) score as the metric.

- Random forest regression model learns and predicts the current data very well. However, the model poses a major challenge that it cannot extrapolate outside unseen data. As an example, if the new testing data contains the feature values which are outside the range of the current data, random forest will have trouble giving accurate predictions. To circumvent this issue, it might be beneficial to employ other regression algorithms (e.g. linear regression) which can develop a generalizing relationship function between features and target value. An important aspect to look at, while implementing these models is how to encode the categorical features. For example, the features `model` and `make` are nominal variables with cardinalities of 1165 and 108, respectively, which is very large. In such cases, encoding methods such as one-hot encoding will blow up the dataset due to creation of new sparse columns and lead to serious memory issues. In this respect, random forest model has very good advantage, since it can function well even if one simply turns the categories into numbers in the same column. However, to use other regression models, we can perhaps look at other ways of numerical encoding such as feature hashing or binary encoding, in order to minimize the creation of new columns, along with efficient regularization techniques to curb overfitting. One other way to reduce overfitting could be to group all the categorical feature (high cardinality ones such as `model`, `make`) elements with say less than 10 instances into a separate group `other`.

- The prediction accuracy can also be improved with the help of gradient boosting algorithm by using the models such as XGBoost or LightGBM and careful parameter tuning.