A Work Project presented as part of the requirements for the Award of a Master's Degree in Finance from the NOVA School of Business and Economics.

Reducing Information Asymmetry in used-car markets by using Machine Learning Models

Leonard Kocks - 33845

A Project carried out on the Master's in Finance Program, under the supervision of:

Qiwei Han (NOVA School of Business and Economics)

Enrico Di Lello (OLX Group)

3rd of January 2020

Abstract

Information asymmetry in used-car markets results from knowledge differences between buyers and sellers about used cars. Naturally, someone who owns a used car for a certain period, develops a deeper understanding of the real value opposed to someone who did not. The goal of this work is to attempt to reduce information asymmetry in used-car markets by using state-of-the-art machine learning models. With data provided by a Polish used-car online marketplace, a price range estimation as well as a point estimation will be made for every car. A Median Absolute Percentage Error of 7.86% and Target Zone of 58.38% are achieved.[1]

Keywords: Information Asymmetry, Machine Learning, Price Range Estimation, Used-car markets

[1] Link to the corresponding GitLab code repository: https://gitlab.com/leo9226/thesis-repository

1. Introduction

Information asymmetry is a phenomenon that exists in markets with an imbalance of knowledge between two parties (Akerlof, 1970). It is especially relevant for used-car markets which are described as markets for "Lemons" by Akerlof. After owning a car for a certain period, sellers have a very good idea of the quality of their cars while a buyer is unable to form such a comprehensive opinion of the car's condition. The motivation behind this work is twofold. First, the aforementioned information asymmetry and resulting adverse selection lead to market inefficiencies and in the worst case market failure (Akerlof, 1970). Second, Otomoto, a used-car online marketplace in Poland and subsidiary of OLX Group that provides the data for this work, wants to create the best possible buying experience for the buyers. By attempting to reduce the information asymmetry by mediating the user a sense of the true value of the car, the user is provided with valuable information one often would not have been able to acquire. Furthermore, transparency is increased and the buyer's probability to overpay decreases. An ideal outcome of this project would be that, in the long term, more buyers are attracted by the platform which consequently will attract more sellers.

This shall be achieved by providing the buyer with a point estimation of the car's value together with a surrounding price range estimation. These values will be computed based on accrued car listing data over the last year. The machine learning model used to compute these values is called LightGBM (Ke et al, 2017). The point estimate, i.e. predicted price, is the median prediction of a quantile regression while the upper and lower boundaries of the price range are the 75^{th} and 25^{th} percentiles, respectively.

The rest of this work is built up as follows. First, literature underlining the relevance of the problem and the importance of a price range is discussed. In Section 3, the methodology, including all

methods used in order to predict the price ranges, is explained in detail. Section 4 deals with the results of the machine learning models, recommendations for Otomoto, a critical discussion of the results, limitations of the work and possible subsequent studies. Finally, the work is concluded.

2. Literature Review

Information Asymmetry and the Market for "Lemons"

In his paper from 1970, Akerlof introduced the concept of the Market for "Lemons". In a used-car market, sellers have more information about the real condition of their car, leading to information asymmetry. Due to the fact that the buyer is unaware of which car is of high and which car is of low quality, he is only willing to pay the average price of all cars. Sellers with high quality cars ("peaches") consequently leave the market because the price offered by the buyers is too low. This phenomenon is commonly referred to as adverse selection. The buyer then adjusts his willingness to pay, causing more "peaches" to leave the market. This cycle carries on until there are only bad cars ("lemons") left in the market. To counteract these effects, Akerlof mentions interventions by the government or warranties and brand names to guarantee the buyer a certain quality standard of the product in perspective (1970).

Price Ranges

To reduce information asymmetry and the resulting adverse selection in used-car markets, an exact price estimate as well as price range estimations will be constructed for each car. At first it may sound counterintuitive to introduce uncertainty to a certain prediction to increase certainty, but in the following this decision will be explained. The reasons to use a price range instead of just a point estimate are of both intrinsic and extrinsic nature. A major extrinsically motivated reason is that the values of car characteristics are subjective and differ from individual to individual. To be able to reflect these perceptions, a price range seems to be an effective tool. Another argument for a

range is price dispersion of identical goods (Kaplan and Menzio, 2015). In their paper, Kaplan and Menzio analyzed 300 million transactions by 50,000 households and showed that the price distribution for the same bundle of goods is approximately normal with a standard deviation between 9% and 14%. Although not in the context of used cars, it is shown that similar goods can have different prices. To verify if this phenomenon also occurs in used-car markets, the distributions of the true prices (*local_gross_price)* for undamaged (i.e. *damaged = 0)* Volkswagen Golf, Opel Astra and Skoda Octavia model cars with similar *vehicle_year, engine_power* and *mileage* are visualized. A similar coefficient of variation and standard deviation can be observed for all three cars, supporting the hypothesis that a price dispersion for similar cars exists in used-car markets. Further explanations can be found in Appendix B.

When applying the Range Theory from Volkmann (1951) to behavioral pricing, it indicates that humans construct a price expectation consisting of a lower and upper boundary based on memorized prices (Janiszewski, 1999). It can be derived that humans intuitively build price ranges, hence supporting the construction of a price range around the median car price prediction. Within research on acceptable price ranges, it was found that consumers tend to perceive relatively low prices as less acceptable, as they create the impression of low quality (Coulter, 2013). Moreover, high prices also lead to lower acceptance, because they are perceived as too expensive. Hence, Monroe suggested that consumers have a lower and upper price threshold (1973). This is similar to the extremeness aversion, a psychological concept that states that an option with more extreme values is perceived as less attractive opposed to an option with rather moderate values (Simonson and Tversky, 1992). A deduction that can be made from this concept is that consumers are more likely to be addressed by narrower price ranges instead of wider, more uncertain ranges. The intrinsic motivation to use price ranges arises from the fact that the model is not perfect and rarely predicts the exact, correct price. In most cases, there is a deviation to the ground truth.

Big Data

The prediction of the point estimate and construction of the price range shall be achieved by leveraging the use of big data and sophisticated machine learning models. Machine learning algorithms fed with big data have shown very promising results in various fields. Amongst them are healthcare (Chen et al, 2017), online car-hailing prediction (Huang et al, 2019) and molecular and materials science (Butler et al, 2018). Just by looking at the large amount of applications, it seems that this problem is ideally suited for machine learning. Otomoto has the necessary high-quality data readily available and a strong IT infrastructure to deploy large-scale machine learning models. Although there is not much literature about big data and the possible implications on information asymmetry available, one paper, however, finds in the context of financial markets that artificial intelligence reduces the degree of information asymmetry and increases market efficiency (Marwala and Hurwitz, 2017). Another paper shows that big data may reduce information asymmetry in the Peer-to-Peer lending industry through the reduction of signaling and search costs (Yan et al, 2015).

Best Practices

A prominent example of such a price range estimation is Zillow (Zillow, 2019). Based on state-of-the-art machine and statistical learning, Zillow forecasts a point estimate together with an estimated sales range for real estate objects. The larger the estimated sales range around the point estimate, the higher the degree of uncertainty of the models regarding the accuracy of the point estimate. One example in the context of used markets for motorized vehicles is Tradus (Tradus, 2019). They predict a point estimate as well as a price range estimation for a certain percentage of the listed vehicles. The prediction is calculated based on the vehicle's features and current and past listings of similar vehicles. Finally, a recommendation for the buyer is made, where Tradus rates the vehicles from *Very low price* to *Very high price.* Tradus' implementation of such a price range

estimation shows that big data can be used to attempt to reduce information asymmetry in used-vehicle markets.

3. Methodology

Data Origin and Curation

The data used throughout this work was provided by Otomoto. It includes cars that were listed from 10/12/2018 (dd/mm/yyyy) to 09/12/2019, allowing the author to use data of in total 365 days. It includes all the features of an advertisement that can be specified during the posting flow including advertisement id and advertisement specific features. Only advertisements with a status other than *moderated, draft, removed_by_moderator, unpaid* and *new* are considered. After removing irrelevant features (i.e. ad-specific, repetitive and invariant), the dataset contains 116 features and approximately 2 million rows. The goal of this work is to predict the variable *local_gross_price* (in Złoty), the price for which a car is listed on the marketplace. The listing price will function as the label, because Otomoto is not acting as a payment provider and thus does not have knowledge of the transaction amounts. The car features given by the user when posted on the website can be bundled into three groups: basic car parameters, extended set of car parameters and car equipment. A detailed explanation of the features can be found in Appendix A, also showing a range of visualizations regarding their frequencies in the dataset. Because the data was heavily curated by Otomoto before being provided, the curation applied throughout this work is very little. A detailed description of the *preparation* function responsible for the data curation can be found in the GitLab repository.

Important Metrics

The two key metrics that will be used to assess the quality of the machine learning models are the **M**e**d**ian **A**bsolute **P**ercentage **E**rror (MdAPE) and **Ta**rget **Z**one (TaZ) (2). The MdAPE is defined

as the median of the **A**bsolute **P**ercentage **E**rrors (APE) of all predictions. The APE itself is defined as follows:

$$APE = \frac{y - \hat{y}}{y} * 100 \quad (1)$$

$$TaZ = \frac{c}{n}, \quad (2)$$

where y is the label, $\hat{y}$ is the prediction, c is the number of all predictions with an APE smaller than or equal to 10% and n is the number of all predictions.

The Model

The model chosen for this work is LightGBM (Ke et al, 2017). LightGBM originates from the family of **G**radient **B**oosting **M**achines (GBM) or rather **G**radient **B**oosting **D**ecision **T**rees (GBDT). Decision trees are tree-like structures that consist of root, internal and terminal nodes, also known as leaves (Rokach and Maimon, 2005). The tree makes the predictions by splitting the dataset into subsets on each level of the tree based on certain criteria. The beginning is marked by the root node, which is split on the best predictor. The end of a tree is marked by its leaves, which, in this case, contain the car price predictions. For visual support, a decision tree can be found in Appendix D. GBDT construct an additive model, where in each iteration a decision tree is trained on the residuals of the previous iteration (Friedman, 1999). These residuals are the gradient of the loss function that is being minimized, hence the name Gradient Boosting. The final prediction for a single car is then calculated by the sum of all the leaves the car ended up in. Because the trees are trained with the residuals, negative values in the leaves are possible. GBDT models have proven to deliver cutting edge performance in a range of tasks (e.g. Li, 2012; Richardson et al, 2007). Some of the reasons are high accuracy, short training and prediction times and low memory consumption (Si et al, 2017). Though, with the ever-growing amounts of data, GBDT is facing

computational complexities. The main reason is that the most time-consuming part of learning a decision tree is finding the optimal split points (Ke et al, 2017). LightGBM introduces two novel techniques, namely **G**radient-based **O**ne-**S**ide **S**ampling (GOSS) and **E**xclusive **F**eature **B**undling (EFB) to target this issue. Ke et al claim that a speed up of the training time compared to established GBDT algorithms by up to over 20 times can be achieved while maintaining practically the same accuracy. GOSS focuses on data instances with a large gradient due to their higher contribution to the information gain in a split compared to data instances with a small gradient. Out of the instances with small gradients, a subsample is dropped, therefore the reduction in training speed. EFB on the other hand bundles mutually exclusive features together to reduce the total number of features. This is especially effective for sparse datasets, where features rarely take non-zero values simultaneously. Another advantage of LightGBM over GBDT algorithms like XGBoost (Chen and Guestrin, 2016) is the possibility of using unencoded categorical features, i.e. strings, as input features.

LightGBM will be used for quantile regressions throughout this work. There are two main reasons for choosing quantile over classical regressions. First, due to their robustness against outliers (John, 2015). Second, quantile regressions allow the prediction of pre-specified percentiles, thus making them an ideal candidate to construct the boundaries of the price range. The point estimation is the median, while the lower and upper bound are the 25^{th} and 75^{th} percentiles, respectively.

EDA

Due to the limited space of this work, the insights gained, and visualizations created throughout the **E**xploratory **D**ata **A**nalysis (EDA) will be presented in Appendix C.

Boruta Feature Selection

In the next step, the feature selection algorithm Boruta is applied on the dataset. In the era of big data, datasets have become increasingly large, often equipped with more features than actually needed to build accurate machine learning models (Kursa, Rudnicki, 2010). The two major problems arising from this issue are of computational nature and decrease in accuracy. While the former slows down algorithms, the latter can have a much worse impact. When the number of features is significantly higher than most favorable, the accuracy of the model's predictions can suffer (Kohavi and John, 1997). Therefore, the goal of feature selection is to find features of high relevance for model construction. Here, BorutaPy (Homola, 2018), a Python package implementing the Boruta feature selection algorithm (Kursa and Rudnicki, 2010), comes into play. Boruta is a wrapper built around a random forest, which is an ensemble of various decision trees. Opposed to GBDT, a random forest grows the trees simultaneously, where the final prediction is made through a vote by all decision trees (Breiman, 2001). Boruta selects the most important features by carrying out a range of steps. First, one shadow feature for each original feature in the dataset is created and their values randomly shuffled. It then runs a random forest on the extended dataset and computes Z scores for each feature. The benchmark the original features need to attain is the **M**aximum **Z** score among **S**hadow **A**ttributes (MZSA). Features that score lower than the MZSA undergo a two-sided test of equality with the MZSA. In case the importance is significantly higher, they are deemed as important and kept in the dataset, otherwise they will be permanently removed. Then, all shadow attributes are removed, and the procedure is repeated until all features are labelled as either important or unimportant. One restriction from using Boruta is that random forests require all features to be numerical and not containing any **N**ot **a** **N**umber (NaN). NaNs in the numerical features are imputed with the median of the specific feature and NaNs in categorical features are replaced with *'Other'*. The string categorical features are encoded using target

encoding (McGinnis, 2016). Even though necessary, this procedure leads to a different dataset. To account for this, the parameter *perc* of BorutaPy, which determines the threshold for comparison between shadow and real features, is set to 80 (Homola, 2018).

Splitting the Dataset

Having selected the most meaningful features, the next step is to split the dataset into training (80%), validation (10%) and test (10%) set. During the hyperparameter optimization, the model will be trained with the training set, while the validation set will be used to calculate the prediction errors. The test set acts as a holdout set and will be used to assess the generalization abilities of the trained model on previously unseen data (Hastie et al, 2008).

Goodness-of-fit tests

Because the dataset is divided into three subsets, it is important to validate that the distributions of the validation and test sets are similar to the training set. If not given, a model will not be able to generalize well and produce biased predictions (Chung et al, 2019). Because different data types require different goodness-of-fit tests, three different statistical tests will be applied. The **K**olmogorov-**S**mirnov test (Massey, 1951) will be used for continuous features, the chi-square contingency test (Pearson, 1900; Pearson, 1904; Crack, 2018) for binary categorical features and the chi-square test for string categorical features (Pearson, 1900; Baird, 1983; Crack, 2018). For all of the above-mentioned goodness-of-fit tests, the null hypothesis is that the two distributions are similar. The hypothesis will be evaluated with the p-values computed by the tests assuming a significance level of 0.05.

Hyperopt Hyperparameter Optimization

Hyperparameters are high-level parameters that control the way a machine learning model works and must be defined prior to the training of a model (Zheng, 2015). In achieving an accurate model, tuning these parameters is paramount, because in some cases a poorly trained model might perform

worse than chance while the same model with tuned hyperparameters produces highly accurate predictions (Hutter et al, 2014). In the case of LightGBM, the pool of tunable hyperparameters is big (Microsoft, 2019). To account for this, the Python library Hyperopt will be utilized (Bergstra et al, 2019). Hyperopt optimizes the hyperparameters of machine learning algorithms and is built for cases where the search space is too big to perform classical optimization techniques such as grid search (Bergstra et al, 2013). Based on **S**equential **M**odel-**B**ased **O**ptimization (SMBO), Hyperopt explores the pre-defined search space and picks the next set of hyperparameters based on the performance of the previous trial (Hutter et al, 2011). Furthermore, SMBO is capable of quantifying parameter importance and parameter interactions. To utilize the Hyperopt library, one must define an objective function and configure a search space (Bergstra et al, 2011; Bergstra et al, 2013). The search space includes all hyperparameters and the respective value ranges. The objective function, in this case, runs a LightGBM model with a set of hyperparameters chosen by the SMBO from the configured search space and returns the MdAPE on the validation set as its loss. This process is repeated for a set number of trials given by the user. The overall goal is to minimize the MdAPE on the validation set. After each Hyperopt run, a visualization of the performance with one plot for each hyperparameter is created and analyzed. Ideally, clusters of points can be observed, meaning that this value area promises good results for this specific hyperparameter. In the succeeding Hyperopt run, the focus will lie on the clustered areas, gradually decreasing the value ranges in the search space.

Cross-validation

Followed by the hyperparameter optimization process, cross-validation will be applied on a LightGBM model trained with the best set of hyperparameters. Cross-validation is a statistical method that, among others, helps to assess the generalization abilities of a machine learning model

by testing it on previously unseen data (Stone, 1974). A common form of cross-validation, also used in this work, is called k-fold cross-validation (Refaeilzadeh et al, 2008). First, the data is split into k folds of equal size. Thereafter, k different models are trained, each on k-1 folds of the data and the performance of this model will be assessed on the held-out fold. The performance of each of the k models will be tracked with the metrics MdAPE and TaZ, both for the test (held-out fold) and the training (k-1 folds) instances. After completion, the metrics of the k models, divided by train and test performance, will be averaged and the standard deviation calculated. Ideally, two things can be observed. The standard deviation is only marginal, meaning that the performance over all the folds is somewhat stable and the difference between the training and test performance is as small as possible, indicating that the model is not overfitting. The number of folds chosen is $k = 5$ so that for each iteration 80% of the data is used to train the algorithm.

After the assessment of the quality of the model, another LightGBM model will be trained on the training and validation sets (90%) and tested on the test set (10%). The result will be visualized using the custom function *plot_performance.*

SHAP Model Explainability

This part of the methodology focuses on model interpretability, something that becomes increasingly important in the age of big data and complex machine learning models whose behavior at times is not apparent (Lakkaraju et al, 2017). As Lundberg et al stated, model interpretability "is important for trust, actionability, accountability, debugging and many other tasks" (2019: 1). To interpret the results of the LightGBM model, the concept of **SH**apley **A**dditive ex**P**lanation (SHAP) values will be applied (Lundberg and Lee, 2017). SHAP values are based on game theory applications and are unique, consistent and locally accurate attribution values to explain model predictions. The authors claim that other feature attribution methods are inconsistent in a way that

they can decrease feature importance although the true importance increases. Lundberg et al furthermore developed the concept of SHAP interaction values (2019). An extension of SHAP values that captures pairwise interaction effects of features and enables consistency even for single predictions. To analyze the feature attributions for the best LightGBM model, one plot will be presented in Section 4, which will show the individualized feature attributions on a global level. In Appendix F more SHAP plots are presented.

Quantile Regressions

To construct the price ranges, two quantile regressions will be built. The lower boundary of the price range will be the 25th percentile prediction while the upper boundary will be the 75th percentile prediction. Hence, 50% of the median predictions will fall into this interval. A prediction interval covering 50% of the predictions is chosen in accordance with the literature presented in section 2. It includes a reasonable amount of predictions while at the same time not allowing the price range to become too large. The metrics **P**rediction **I**nterval **C**overage **P**robability (PICP) (3), **M**ean **I**nterval **P**rediction **W**idth (MPIW) (4), MPIW captured (5) and **R**elative **M**ean **P**rediction **I**nterval **W**idth (6) will measure the quality of the intervals (Pearce et al, 2018). They are defined as follows.

$$PICP = \frac{c}{n} \tag{3}$$

$$MPIW = \frac{1}{n}\sum_{i=1}^{n} \hat{y}_{Ui} - \hat{y}_{Li} \tag{4}$$

$$MPIW\ captured = \frac{1}{c}\sum_{i=1}^{c} \hat{y}_{Ui} - \hat{y}_{Li} \tag{5}$$

$$RMPIW = \frac{1}{n}\sum_{i=1}^{n} \frac{\hat{y}_{Ui} - \hat{y}_{Li}}{\hat{y}_i} \tag{6}$$

where n is the total number of predictions, c is the number of median predictions that fall within the prediction interval [25; 75] (i.e. captured), $\hat{y}_{Ui}$ ($\hat{y}_{Li}$) is the upper (lower) boundary of the interval for car i and $\hat{y}_i$ is the median prediction for car i.

Comparison to Baseline

To measure the relative accuracy of the LightGBM model trained with the boruta features and optimized hyperparameters, a comparison to four different models will be conducted. First, two LightGBM models will be trained, one with all features (1) and one with basic features (2). Subsequently, an XGBoost regression (Chen and Guestrin, 2016) as well as a scikit-learn random forest (Pedregosa et al, 2011) will be fit on the boruta features. The comparison with XGBoost and the random forest is not fully representative, because the hyperparameters are not tuned (i.e. default set of hyperparameters) and they require the inputs to be numerical. Furthermore, random forests do not accept NaNs. To replace these values, the same heuristics already used within the feature selection with the Boruta algorithm are applied.

4. Results, Recommendations, Discussion, Limitations & Future Research

Results

After applying the Boruta feature selection algorithm on the dataset, 51 features out of 115 are selected and thus, deemed as important to predict the price of a car. This reduction in size of about 55% is rather drastic and indicates that most of the features in the original dataset are of poor quality for this specific problem.

The result of the goodness-of-fit tests is very satisfying. In total, 12 tests for continuous, 70 tests for binary categorical and 20 tests for string categorical features are conducted. Except for four binary tests, for every single test the conclusion is that the respective features from the training and validation/test set come from a similar distribution.

In order to tune the hyperparameters, three Hyperopt runs were conducted. The result of the first run is illustrated by Figure 1 (only 3 subplots are shown).

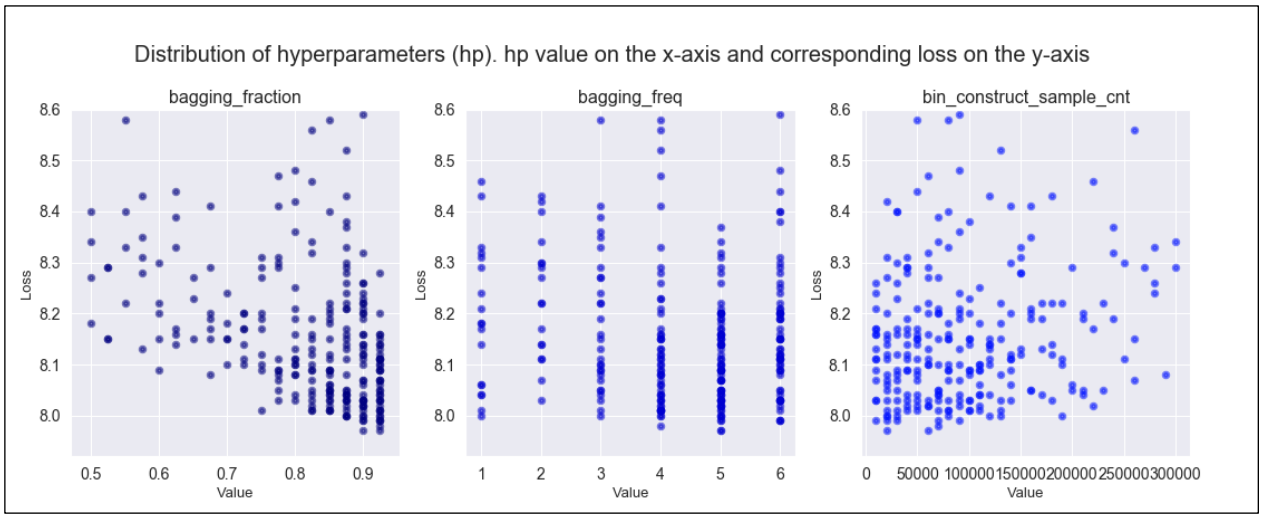

Figure 1 Scatterplot for the distribution of the hyperparameters during a Hyperopt run

Each of the subplots is structured in the same way. The title refers to the hyperparameter, the x-axis shows the value range and the y-axis shows the loss (MdAPE) of the iteration where the value of this hyperparameter is the corresponding point on the x-axis. For instance, *bagging_fraction* shows a cluster of points in the range from 0.8 to 0.925, indicating that this area promises good results. All subplots will be analyzed and the value range, if possible, reduced. Appendix E contains all used search spaces, one plot like Figure 1 for each Hyperopt run and the final set of hyperparameters yielding the lowest MdAPE. Moreover, an explanation of the hyperparameters is included.

The cross-validation performed with the set of hyperparameters chosen by Hyperopt shows a mean MdAPE of 7.92% over the 5 folds on the test set with a mean TaZ of 58.2%. The standard deviation of the MdAPE and TaZ on the test set are 0.03% and 0.13%, respectively. The mean MdAPE on the train set over the 5 folds lies at 5.36% with a standard deviation of 0.01%. Both the standard deviation on the test set and the difference between test and training set performances are very satisfactory. Hence, an adequate quality of the hyperparameters is confirmed.

Next, the performance of a LightGBM model with the best hyperparameters will be evaluated by plotting the results (Figure 2).

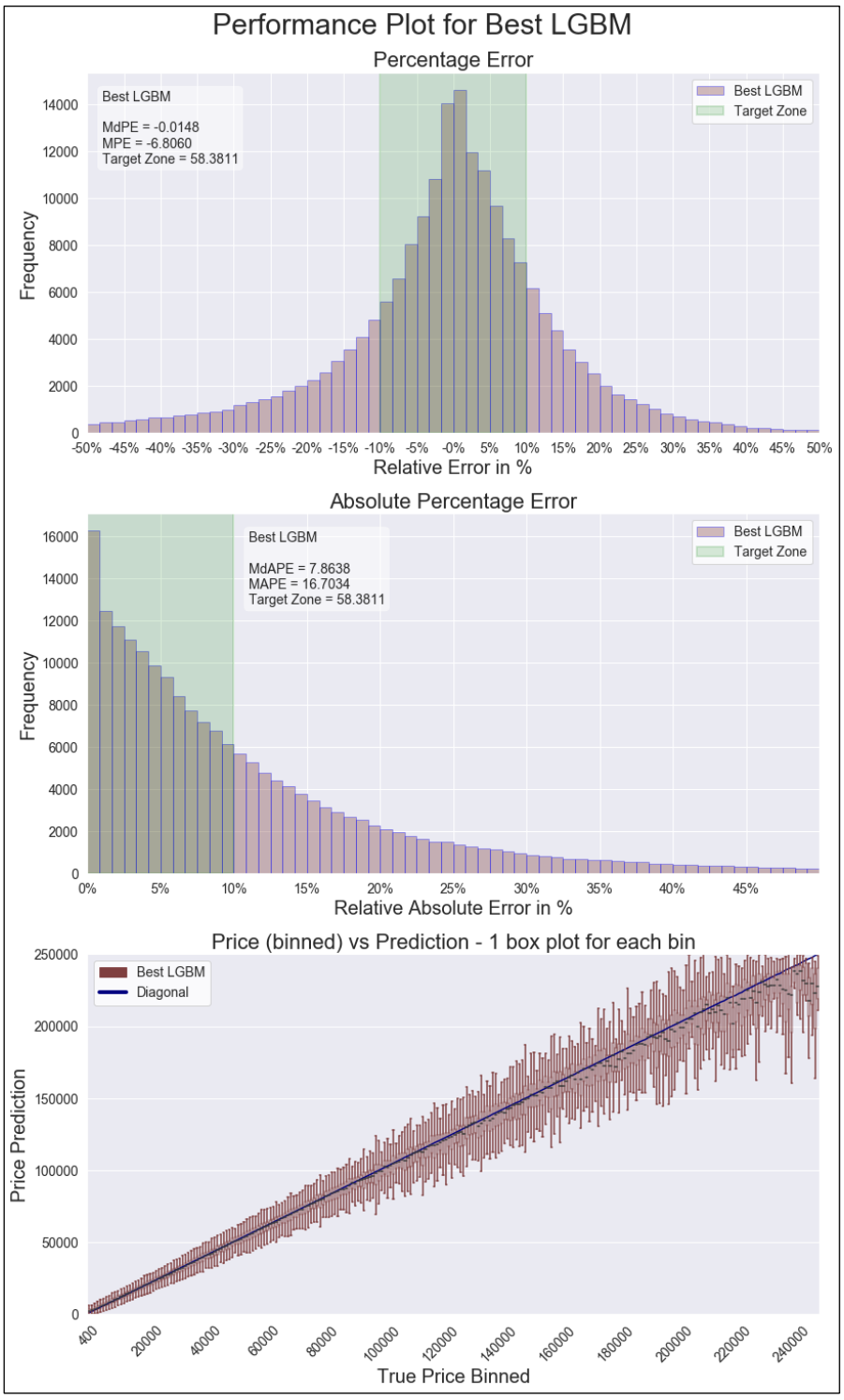

Figure 2: Performance Plot for the best LightGBM Model

Subplot 1 shows a **P**ercentage **E**rror (PE) histogram with PE on the x- and frequency on the y-axis. It can be noted that the distribution looks similar to a bell curve and has a median PE of around 0%, meaning that the model is evaluating the prices fairly. Subplot 2 shows an APE histogram with the APE on the x- and frequency on the y-axis. While the MdAPE and TaZ are 7.86% and 58.38% respectively, a large proportion of the predictions has an APE of close to 0%. An MdAPE of around 0% may be a signal for an overfitting model. The purpose of subplot 3 is to analyze the variance of the predictions for different price levels. To do so, the true price is binned into 250 bins and each prediction is assigned into one bin. For each bin, one box plot is plotted, showing the variance of the predictions. The diagonal shows the ideal prediction value as it is composed by the median values of each of the 250 bins. The main takeaway from this plot is that the variance increases at a steady, but slow pace until around 160,000 Złoty (Zł). From there, the length of the box plots starts to fluctuate more heavily.

Figure 3 below is plotted with the SHAP library and summarizes the feature importance for the 7 most important predictors. On the y-axis is the feature, on the secondary y-axis a color scale ranging from red (corresponds to high value) to blue (corresponds to low value) and the x-axis represents the SHAP value, which is a measurement for the impact on the prediction of the model. I.e., a SHAP value for mileage of 12,250 Zł increases the prediction by exactly 12,250 Zł. For each feature, the SHAP value of every predicted instance is represented as a dot. It immediately becomes apparent that the most influential predictors of the car price are *vehicle_year*, *engine_power*, *mileage*, *make* and *model*. In the case of *vehicle_year*, it can be stated that a newer car has positive SHAP values and thus, increases the prediction, whereas older cars are more onto the blueish side of the color scale and have a negative influence on the car price prediction. String categorical features are in grey, because the strings are nominal. Due to the limited scope of this work, three

SHAP plots, an analysis of a single prediction, a feature importance bar plot, and the full summary plot (Figure 3) with 20 features will be presented in Appendix F.

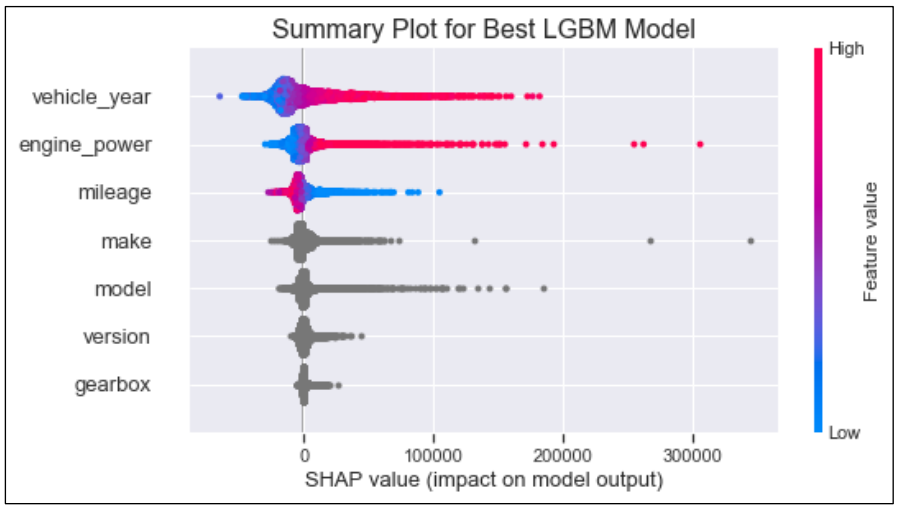

Figure 3: Summary plot for the 7 most important features

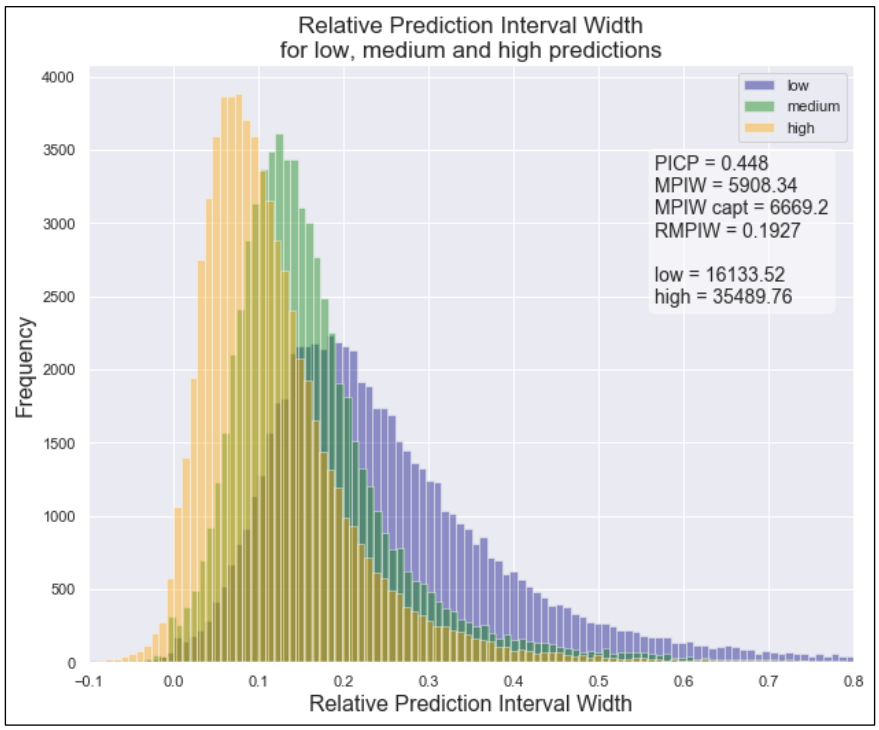

Figure 4: Relative Prediction Interval Width for cars with low, medium and high predictions

The results of the two quantile regressions, 25th and 75th percentile, are presented above in Figure 4, which shows three histograms illustrating the distributions of the relative prediction interval

width for cars with low, medium and high predictions. To get equal subsets, the thresholds to divide the instances are the 33rd and 66th percentiles. The density is on the y-axis. Theoretically, the PICP should be 0.5, because 50% of the points fall within the interval [25; 75]. Practically, the PICP is 0.448, meaning that 44.80% of the true prices fall within their individual [25; 75] prediction interval. The MPIW for all predictions is 5908.34 Zł, while the MPIW captured is 6669.2 Zł. The overall RMPIW is 0.1927, showing that on average the length of the prediction interval is 1/5 of the size of the median prediction. The distributions of the three subsets furthermore show that with an increasing prediction, the relative prediction interval width is decreasing.

Finally, the results from the comparison of the best LightGBM model to the baseline models are presented in Table 1 below. The results show that a LightGBM model with all features shows a slightly better MdAPE. Therefore, the feature selection process needs to be reconsidered as accuracy is lost. Possible explanations can be the loss of important interactions between features and the altered dataset used throughout the feature selection. Appendix G shows two plots: one comparing the random forest and XGBoost; and one comparing the three LightGBM models.

Algorithm	**Features**	**Preprocessing**	**Tuned**	**MdAPE test**	**TaZ test (%)**	**MdAPE train**	**TaZ train (%)**
LightGBM	Boruta	None	Yes	7.89	58.23	5.52	67.56
LightGBM	All	None	Yes	7.73	59.09	5.14	69.02
LightGBM	Basic	None	Yes	9.66	51.21	7.88	57.73
XGBoost	Boruta	Target Encoding	No	15.58	34.22	15.54	34.29

Algorithm	Features	Preprocessing	Tuned	MdAPE test	TaZ test (%)	MdAPE train	TaZ train (%)
Random Forest	Boruta	Target encoding & NaNs	No	7.58	59.03	3.0	86.64

Table 1: Comparison of 5 different learning algorithms

Recommendations

An important question is how to use the predictions or more specifically, how to present the point estimate and price range estimations to the buyer on the marketplace. Given an example where the median prediction, 25th percentile and 75th percentile are 6,500, 6,250 and 7,000 Zł, respectively, a recommendation for the buyer could involve a color bar and the following information:

"The suggested price for this car is 6,500 Zł.

50% of the sellers sell a similar car for 6,250 – 7,000 Zł."

For visual support, a color bar shows the upper and lower boundaries and median prediction. Moreover, it is crucial to not leave the buyer in the dark about the computation of the suggested prices. Explanations regarding the technology, methodology and used car characteristics need to be included. By clarifying these things, the buyer can retrace the whole process and the probability of developing trust towards the suggested prices and a deeper understanding increases.

Discussion

The fact that the target group of introducing price range estimations is the buyers leads to a problem that could cause severe consequences for Otomoto. As Otomoto does not sell cars, they are dependent on sellers that use their platform. If a seller is unsatisfied with the predictions, e.g. because he believes the true value of the car is higher, the probability of delisting the car and moving to a competitor increases. In case the car is indeed overpriced, it is both beneficial for the

buyer, as he avoids purchasing an overpriced car, and for the algorithm, as the quality of the data improves. On the other hand, if the car is fairly priced, Otomoto loses a valuable seller and the number of good cars listed on their website decreases. Even though the predictions might not satisfy the expectations of the seller, at the same time they are a reason to stay on the platform. Because they are so informative and beneficial for the buyers, the number of users on the platform and hence, possible buyers for a car increases. Thus, the increasing buyer potential is a strong motivator to stay on the platform. Another problem is created by the price range itself. Wide range indicate uncertainty and instead of helping the buyers to form an opinion about the true value of a car, it can create confusion.

Limitations

Throughout this work, a local machine is used. This comes with computational restrictions, especially concerning the hyperparameter optimization and feature selection. Another limitation arises from the feature selection with Boruta, which utilizes random forests and therefore requires all features to be numerically encoded and without NaNs. A different approach for feature selection could be chosen to mitigate this limitation. A limitation concerning the price of the cars is that Otomoto does not record transaction data of actual purchases. Therefore, only the listing price on the platform is available to build a learning algorithm. Nevertheless, the feature *is_business* indicates if a seller is private or business/commercial. In general, business retailers have a better understanding of the real value of the cars they are selling. The model should be able to learn this pattern and the MdAPE on cars sold by business retailers smaller. This can be observed with 10.26% MdAPE for private and 6.07% for business sellers. A possible heuristic is to learn the model just on cars of business sellers. Yielding in a more accurate model, this leaves out 49.0% of the data and thus a lot of possibly important variance. More importantly, business sellers focus on a smaller range of cars than private sellers, because they promise higher profits. The model would

have problems generalizing well on non-business seller cars once in production. Lastly, this model has not been deployed. Consequently, no observations have been made and the effect of a possibly reduced information asymmetry cannot be measured.

Future Research

Future research should focus on two main topics. First, the model offers space for improvements. Approaches could involve feature engineering, new approaches for feature selection, further tuning of the hyperparameters or trying different learning algorithms including neural networks. Second, the implications on the real world of the introduction of such a model need to be researched. It is interesting to know how sellers, buyers and Otomoto are affected and if the overall goal, a reduction of information asymmetry in used-car markets, is reached.

5. Conclusion

The purpose of this work is to attempt to reduce information asymmetry in used-car markets by using machine learning models, namely LightGBM. To achieve this, a point as well as a price range estimation can be presented to the buyer. Feature selection, goodness-of-fit tests and hyperparameter optimization are applied on a dataset with one year of car listing data provided by a Polish used-car online marketplace. Lastly, two quantile regressions are built for the upper and lower boundaries of the price range. The model shows a good performance with a Median Absolute Percentage Error of 7.86% and a Target Zone of 58.38%. The Relative Mean Prediction Interval Width of the price ranges is 0.1927. Based on the model's performance and accuracy, a deployment on Otomoto for a trial period should be considered. To make adequate use of the results, Otomoto should present the point estimation together with the price ranges and a supportive, colored graphic. Since the model has not been deployed, the effects on all affected parties cannot be observed.

References

Akerlof, George A. 1970. “The Market for "Lemons": Quality Uncertainty and the Market Mechanism.” *The Quarterly Journal of Economics*, 84(3): 488-500.

Baird, Davis. 1983. “The Fisher/Pearson Chi-Squared Controversy: A Turning Point for Inductive Inference.” *The British Journal for the Philosophy of Science*, 34(2): 105-118.

Bergstra, James, Dan Yamins and David D. Cox. 2013. “Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms.” Proceedings of the 12th Python in Science Conference. (SCIPY 2013).

Bergstra, James, Dan Yamins and David D. Cox. 2019. Hyperopt. https://github.com/hyperopt/hyperopt (accessed December 23, 2019).

Bergstra, James, Rémi Bardenet, Yoshua Bengio, Balázs Kégl. 2011. “Algorithms for Hyper-Parameter Optimization.” https://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf

Breiman, Leo. (2001). “Random Forests.” *Machine Learning*, 45: 5–32.

Butler, Keith T., Daniel W. Davies, Hugh Cartwright, Olexandr Isayev and Aron Walsh. 2018. “Machine learning for molecular and materials science”. *Nature*, 559: 547–555.

Chen, Min, Yixue Hao, Kai Hwang, Lu Wang and Lin Wang. 2017. "Disease Prediction by Machine Learning Over Big Data From Healthcare Communities," *IEEE Access*, 5: 8869-8879.

Chen, Tianqi, and Carlos Guestrin. 2016. “XGBoost.” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16.

Chung, Yeounoh, Peter J. Haas, Tim Kraska, Eli Upfal. 2019. „Learning Unknown Examples For ML Model Generalization“. https://arxiv.org/abs/1808.08294

Coulter, Keith S. 2013. “Commentary on: “an appraisal of behavioral price research (Part I)””. *AMS Review*, 3(3): 135-140.

Crack, Timothy Falcon. 2018. “A Note on Karl Pearson's 1900 Chi-Squared Test: Two Derivations of the Asymptotic Distribution, and Uses in Goodness of Fit and Contingency Tests of Independence, and a Comparison with the Exact Sample Variance Chi-Square Result.” https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3284255

Friedman, Jerome H. 2001. “Greedy Function Approximation: A Gradient Boosting Machine.”, IMS 1999 Reitz Lecture.

Hastie, Trevor, Robert Tibshirani, Jerome Friedman. 2008. “The Elements of Statistical Learning. Data Mining, Inference, Prediction.” Springer.

Homola, Daniel. 2018. https://github.com/scikit-learn-contrib/boruta_py (accessed December 23, 2019).

Huang, Zihao, Gang Huang, Zhijun Chen, Chaozhong Wu, Xiaofeng Ma, Haobo Wang. 2019. „Multi-Regional Online Car-Hailing Order Quantity Forecasting Based on the Convolutional Neural Network.” *Information*, 10, 193.

Hutter, Frank, Holger Hoos and Kevin Leyton-Brown. 2011. „Sequential Model-Based Optimization for General Algorithm Configuration (extended version).” https://www.cs.ubc.ca/~hutter/papers/10-TR-SMAC.pdf.

Hutter, Frank, Holger Hoos and Kevin Leyton-Brown. 2014. „An Efficient Approach for Assessing Hyperparameter Importance.“ Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP Volume 32.

Janiszewski, Chris and Donald R. Lichtenstein. 1999. “A Range Theory Account of Price Perception” *Journal of Consumer Research*, 25(4): 353-368.

John, Onyedikachi O. 2015. “Robustness of Quantile Regression to Outliers.” *American Journal of Applied Mathematics and Statistics*, 3(2): 86-88.

Kaplan, Greg, and Guido Menzio. 2015. “The Morphology of Price Dispersion” *International Economic Review*, 56(4): 1165-1205.

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. 2017. „LightGBM: A Highly Efficient Gradient Boosting Decision Tree”. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Kohavi, Ron and George H. John. 1997. “Wrappers for feature subset selection.”, *Artificial Intelligence*, 97: 273-324.

Kursa, Miron B. and Witold R. Rudnicki. 2010. “Feature Selection with the Boruta Package”. *Journal of Statistical Software*, 36(11).

Lakkaraju, Himabindu, Ece Kamar, Rich Caruana and Jure Leskovec. 2017. "Interpretable & Explorable Approximations of Black Box Models." KDD'17, Workshop on Fairness, Accountability, and Transparency in Machine Learning.

Li, Ping. 2012. "Robust LogitBoost and Adaptive Base Class (ABC) LogitBoost" https://arxiv.org/ftp/arxiv/papers/1203/1203.3491.pdf.

Lundberg, Scott M. and Su-In Lee. 2017. "A Unified Approach to Interpreting Model Predictions." 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Lundberg, Scott M., Gabriel G. Erion and Su-In Lee. 2019. "Consistent Individualized Feature Attribution for Tree Ensembles" https://arxiv.org/abs/1802.03888.

Marwala, Tshilidzi and Evan Hurwitz. 2017. "Artificial Intelligence and Economic Theories", https://arxiv.org/ftp/arxiv/papers/1703/1703.06597.pdf

Massey, Frank T. 1951." The Kolmogorov-Smirnov Test for Goodness of Fit." *Journal of the American Statistical Association*, 46(253): 68-78.

McGinnis, Will. 2016. https://contrib.scikit-learn.org/categorical-encoding/targetencoder.html (accessed December 23, 2019).

Microsoft. 2019. Microsoft Corporation Revision. https://lightgbm.readthedocs.io/en/latest/Parameters.html (accessed December 23, 2019).

Monroe, Kent B. 1973. "Buyers' Subjective Perceptions of Price" *Journal of Marketing Research*, 10(1): 70-80.

Pearce, Tim, Mohamed Zaki, Alexandra Brintrup and Andy Neely. 2018. "High-Quality Prediction Intervals for Deep Learning: A Distribution-Free, Ensembled Approach." Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

Pearson, Karl. 1900. "On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *Philosophical Magazine Series* 5, 50(302): 157–175.

Pearson, Karl. 1904. "On the Theory Of Contingency and Its Relation To Association and Normal Correlation" appearing in Drapers' Company Research Memoirs, Biometric Series I, Mathematical Contributions to the Theory of Evolution, part XIII, Dulau and Co., London.

Pedregosa et al. 2011. "Scikit-learn: Machine Learning in Python" *Journal of Machine Learning Research*, 12: 2825-2830

Refaeilzadeh, Payam, Lei Tang and Huan Liu. 2008. „Cross-Validation". http://leitang.net/papers/ency-cross-validation.pdf

Richardson, Matthew, Ewa Dominowska and Robert Ragno. 2007. "Predicting clicks: estimating the click-through rate for new ads." In Proceedings of the 16th international conference on World Wide Web, pages 521–530. ACM.

Rokach, Lior, and Oded Maimon, 2005. "Decision Trees." In *Data Mining and Knowledge Discovery Handbook*, 165-192, Boston, MA: Springer.

Si, Si, Huan Zhang, S. Sathiya Keerthi, Dhruv Mahajan, Inderjit S. Dhillon and Cho-Jui Hsieh .2017. "Gradient Boosted Decision Trees for High Dimensional Sparse Output". Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

Simonson, I., and Tversky, A. 1992. "Choice in context: Tradeoff contrast and extremeness aversion" *Journal of Marketing Research*, 29(3): 281–295.

Stone, M. 1974. "Cross-Validatory Choice and Assessment of Statistical Predictions." *Journal of the Royal Statistical Society*. Series B (Methodological), 36(2): 111-147.

Tradus. 2019. Tradus B.V. https://www.tradus.com/en/transport/vans/closed-vans/renault/renault-trafic-kasten-l1h1-2-7t-komfort-2017-4699529 (accessed December 23, 2019).

Volkmann, John. 1951. "Scales of Judgment and Their Implications for Social Psychology," in John H. Rohrer and Muzafer Sherif (Eds.), Social Psychology at the Crossroads, New York: Harper.

Yan, Jiaqi, Wayne Yu and J. Leon Zhao. 2015. "How signaling and search costs affect information asymmetry in P2P lending: the economics of big data". *Financial Innovation*, 1(19).

Zheng, Alice. 2015. "Evaluating Machine Learning Models". O'Reilly Media, Inc.

Zillow. 2019. Zillow Group. https://www.zillow.com/zestimate/ (accessed December 23, 2019).

Appendices

A – Description of different groups of features

The following explanations of the three groups of features were adapted from an internal document.

1. Basic car characteristics:

- make
- model
- vehicle_year
- mileage
- fuel_type
- engine_capacity
- engine_power
- gearbox

This subset of car parameters is relevant because the concept of "similar cars" used in Sourcing Insights. engine_capacity, gearbox and engine_power are non-mandatory, but except gearbox (96.41%), appear in 100% of the ads. Figure 5 represents the percentage of ads for which a value is specified for the corresponding feature.

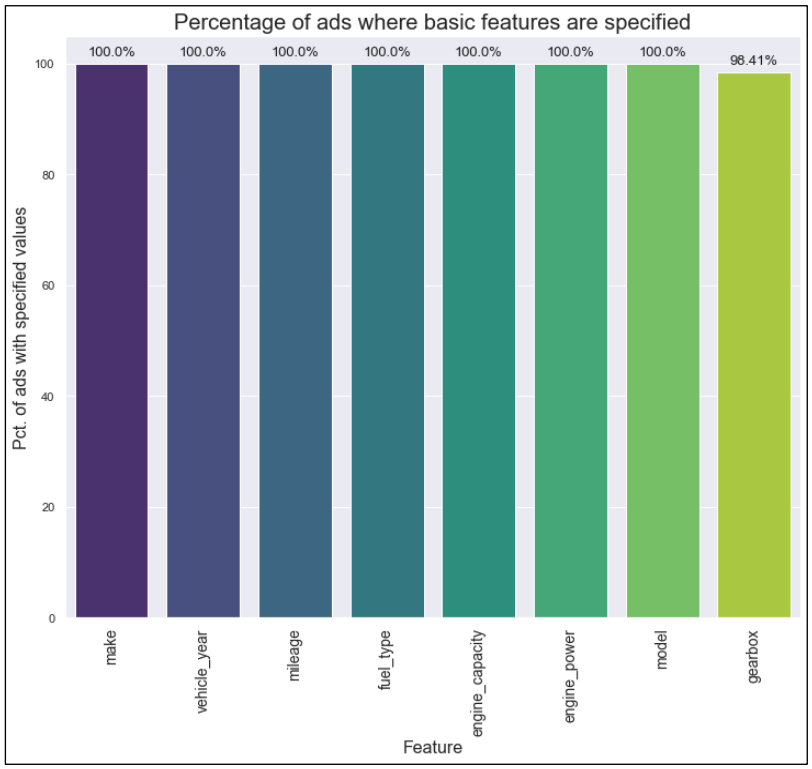

Figure 5: Occurrence of basic car features

2. Extended car characteristics:

The second set is composed by all the fields contained in the <params> field, except for the 'features' field, which contains the car equipment features and is analyzed later. The features included in this subset are:

- price_gross_net, price_currency, make, vehicle_year, fuel_type, color, body_type, model, price, mileage, has_vin, has_registration, engine_capacity, metallic, gearbox, no_accident, registered, service_record, engine_power, original_owner, door_count, vat_discount, nr_seats, financial_option, vat, rhd, transmission, damaged, leasing_concession, version, peal, particle_filter, tuning, approval_for_goods, historical_vehicle, matt, country_origin, date_registration, vin, registration, engine_code, monthly_payment, co2_emissions, video, remaining_payments, residual_value, down_payment, authorized_dealer, acrylic

Similar graphs for the extended car characteristics are represented by Figures 6 and 7 below.

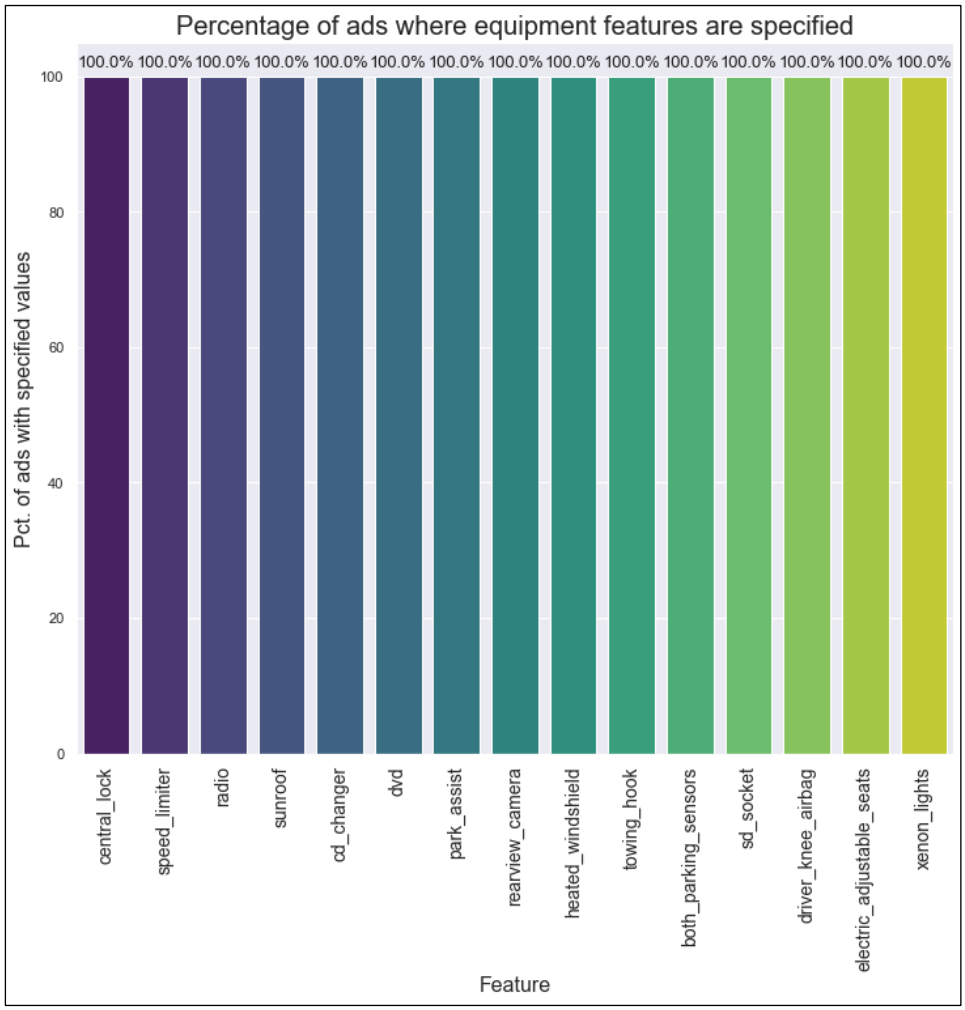

Figure 6: Occurrence of extended car features no. 1

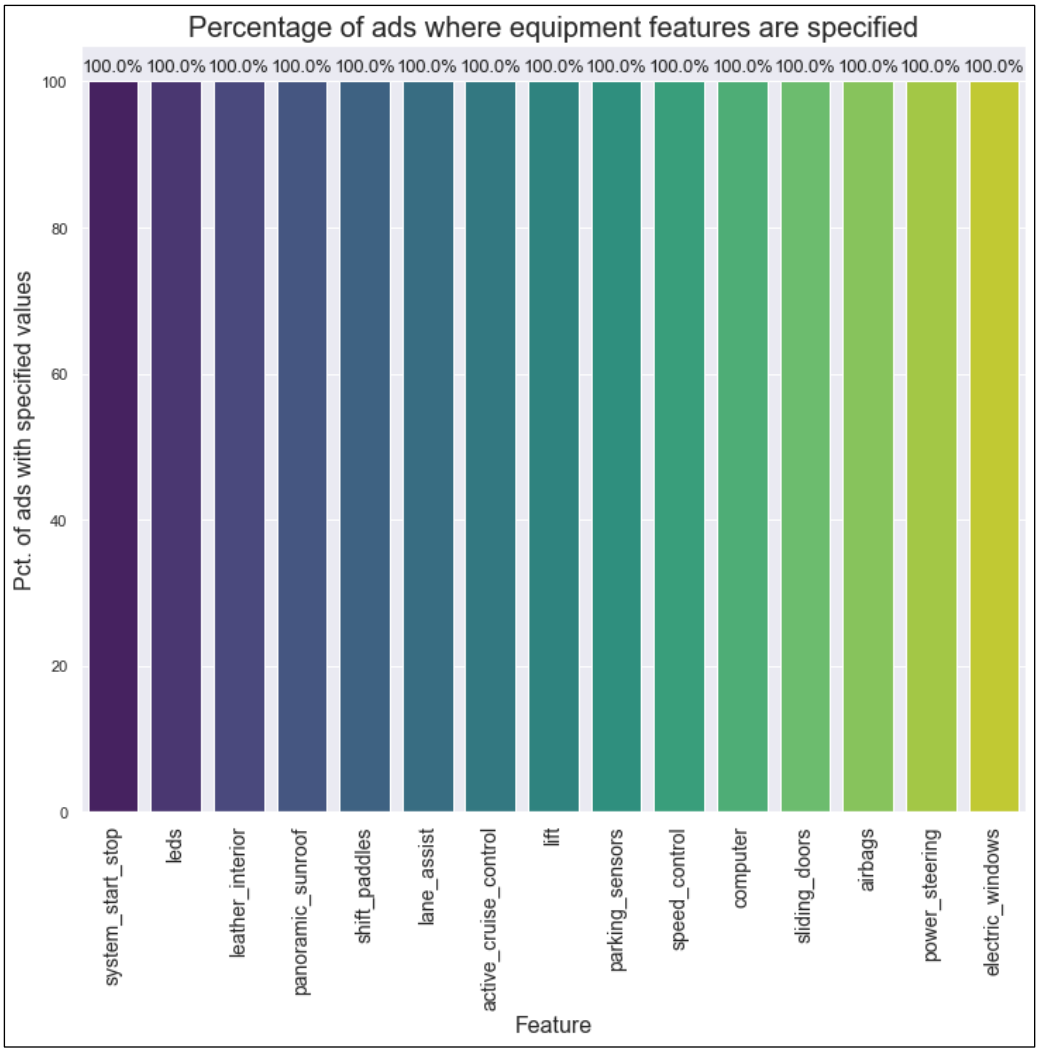

Figure 7: Occurrence of extended car features no. 2

3. Car equipment characteristics:

The car equipment features are encoded in a different way. There are 85 possible equipment features, and during the data extraction process, every feature is counted if present or set to zero if absent. The features included in this subset are:

- central_lock, abs, front_airbags, assisted_steering, front_electric_windows, front_passenger_airbags, electronic_rearview_mirrors, electronic_immobiliser, cd, original_audio, onboard_computer, alloy_wheels, esp, fog_lights, asr, front_side_airbags, steering_wheel_commands, automatic_air_conditioning, rear_electric_windows, heated_rearview_mirrors, cruise_control, isofix, side_window_airbags, alarm, rear_parking_sensors, automatic_wipers, mp3, aux_in, bluetooth, automatic_lights, dual_air_conditioning, electric_interior_mirror, daytime_lights, front_heated_seats, usb_socket, tinted_windows, air_conditioning, rear_passenger_airbags, velour_interior, gps, roof_bars, both_parking_sensors, electric_exterior_mirror, leather_interior, leds, system_start_stop, xenon_lights, electric_adjustable_seats, driver_knee_airbag, sd_socket,

towing_hook, speed_limiter, heated_windshield, rearview_camera, park_assist, dvd, cd_changer, sunroof, radio, panoramic_sunroof, shift_paddles, lane_assist, active_cruise_control, blind_spot_sensor, adjustable_suspension, quad_air_conditioning, auxiliary_heating, rear_heated_seats, head_display, tv, electric_windows, power_steering, airbags, sliding_doors, computer, speed_control, parking_sensors, lift, passenger_airbag

Because of the encoding applied, there are no NaNs in the equipment features. The bar plots can be found in the folder *Graphs* in the GitLab repository.

B – Deviation of prices for similar cars

All cars were grouped by *make* and *model* and the following cars were picked for a more detailed analysis:

- Volkswagen Golf (3rd most common car)
- Opel Astra (most common car)
- Skoda Octavia (7th most common car)

In order to only compare similar cars, the cars listed above were filtered for the following criteria:

- *130 <= engine_power <= 150*
- *2018 <= vehicle_year <= 2019*
- *mileage <= 10000*
- *damaged = 0*

The following histograms (Figures 8, 9 and 10) are all structured the same. On the y-axis is the true price (*local_gross_price)* and on the x-axis is the frequency. Under the legend is a text box with important information regarding the number of cars plotted (n), the standard deviation (std), mean and coefficient of variation. For all cars, the coefficient of variation lies between 12.3 and 15.6%, supporting the hypothesis that similar cars have a varying price.

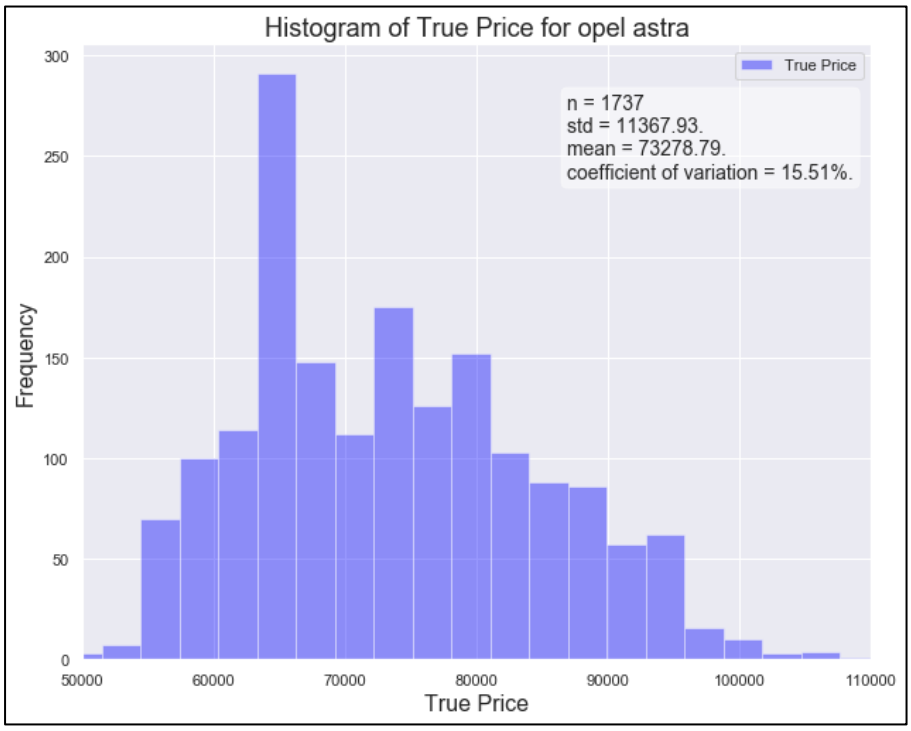

Figure 8: Histogram for *local_gross_price* of similar Opel Astra cars

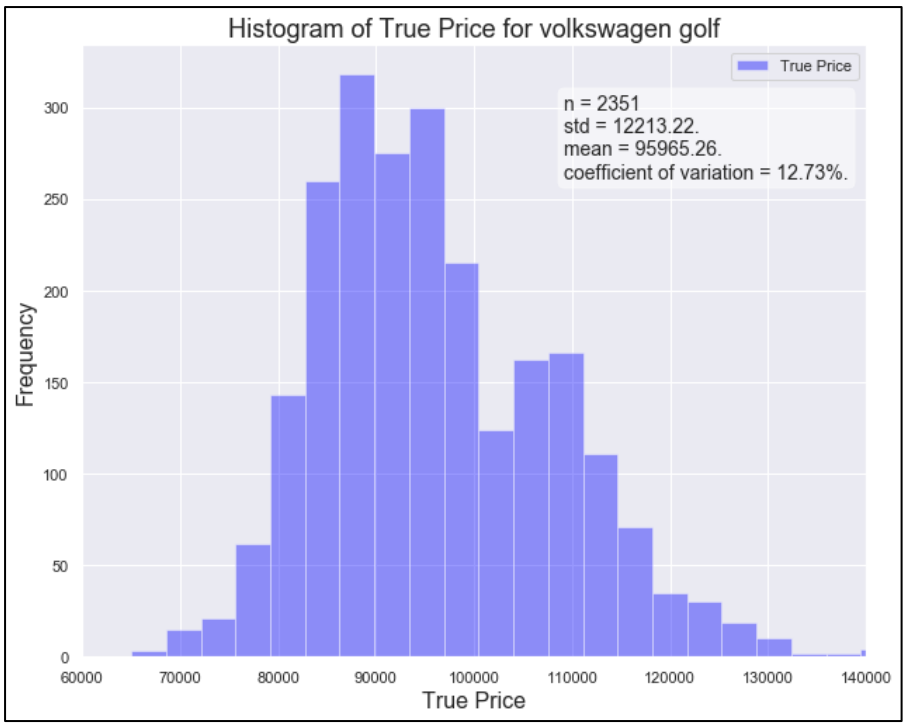

Figure 9: Histogram for *local_gross_price* of similar Volkswagen Golf cars

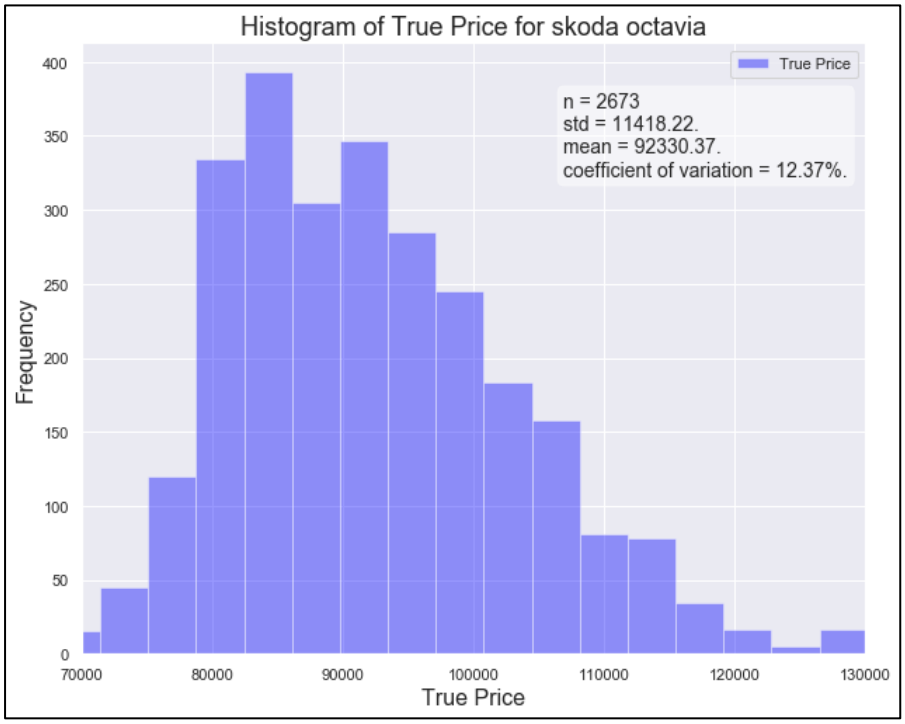

Figure 10: Histogram for *local_gross_price* of similar Skoda Octavia cars

C – Exploratory Data Analysis

Hereafter, the visualizations created, and insights gained throughout the EDA will be presented and discussed.

Figure 11 represents two waffle charts, where each square stands for a certain number of cars. The waffle charts show the distribution of *body_type* and *transmission*, respectively. It becomes apparent that the top 6 body types, *combi, compact, suv, sedan, city-car and minivan* make up the lion's share of all body types. Looking at the distribution of *transmission*, one can see that *front-wheel* makes up for around 75% of all transmissions while *rear-wheel, all-wheel-auto* and *all-wheel-permanent* are rather equally distributed.

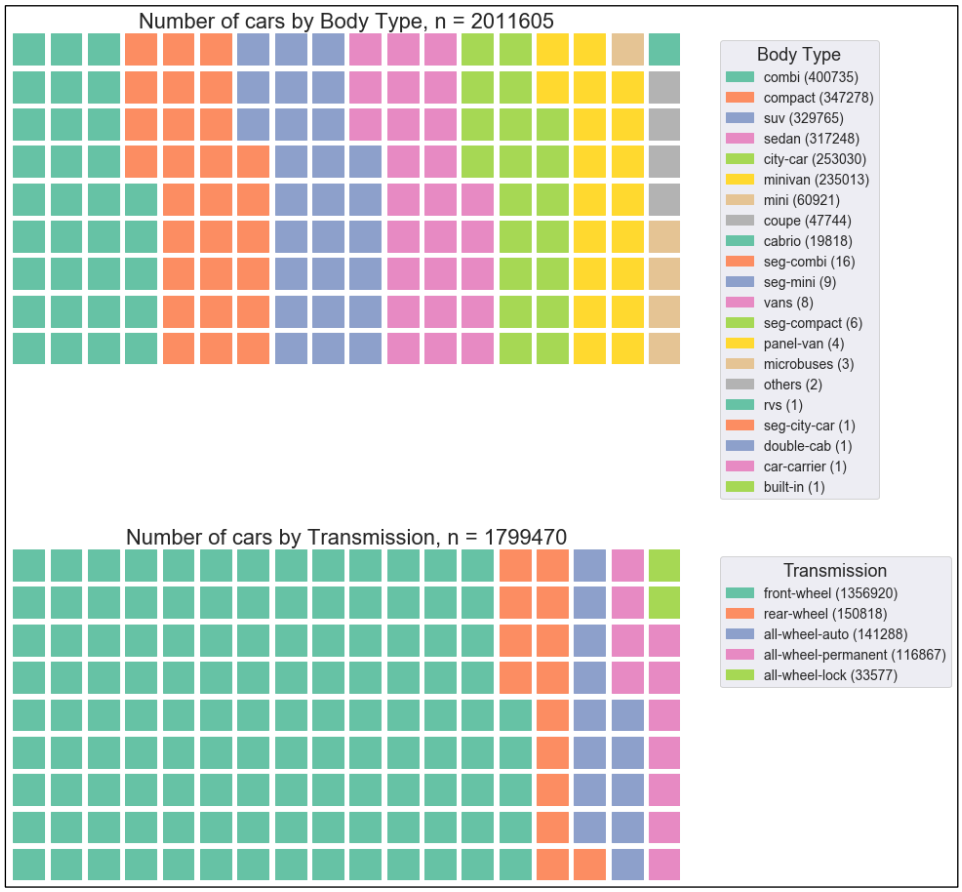

Figure 11: Waffle Chart for number of cars by *body_type* and *transmission*

Figure 12 below shows the distribution of the features *color* and *fuel_type* in the form of a bar chart. For both plots, the colors/fuel types are on the x- and the corresponding counts on the y-axis. One can observe that the colors of the cars are rather traditional with *black, silver, grey, white* and blue making up the majority. Though, the number of colorful cars, e.g. *red, green or dark-red* is not insignificant.

The distribution of *fuel_type* shows a similar picture than *transmission*. *diesel* and *petrol* occur roughly equally alike and together make up around 94% of all fuel types. Interestingly, among the 2 million cars there are only 57 electric ones. This can be explained by the fact that otomoto is a

used-car platform and that electric vehicles are rather new. Hence, it is unlikely that they are already sold on used-car platforms.

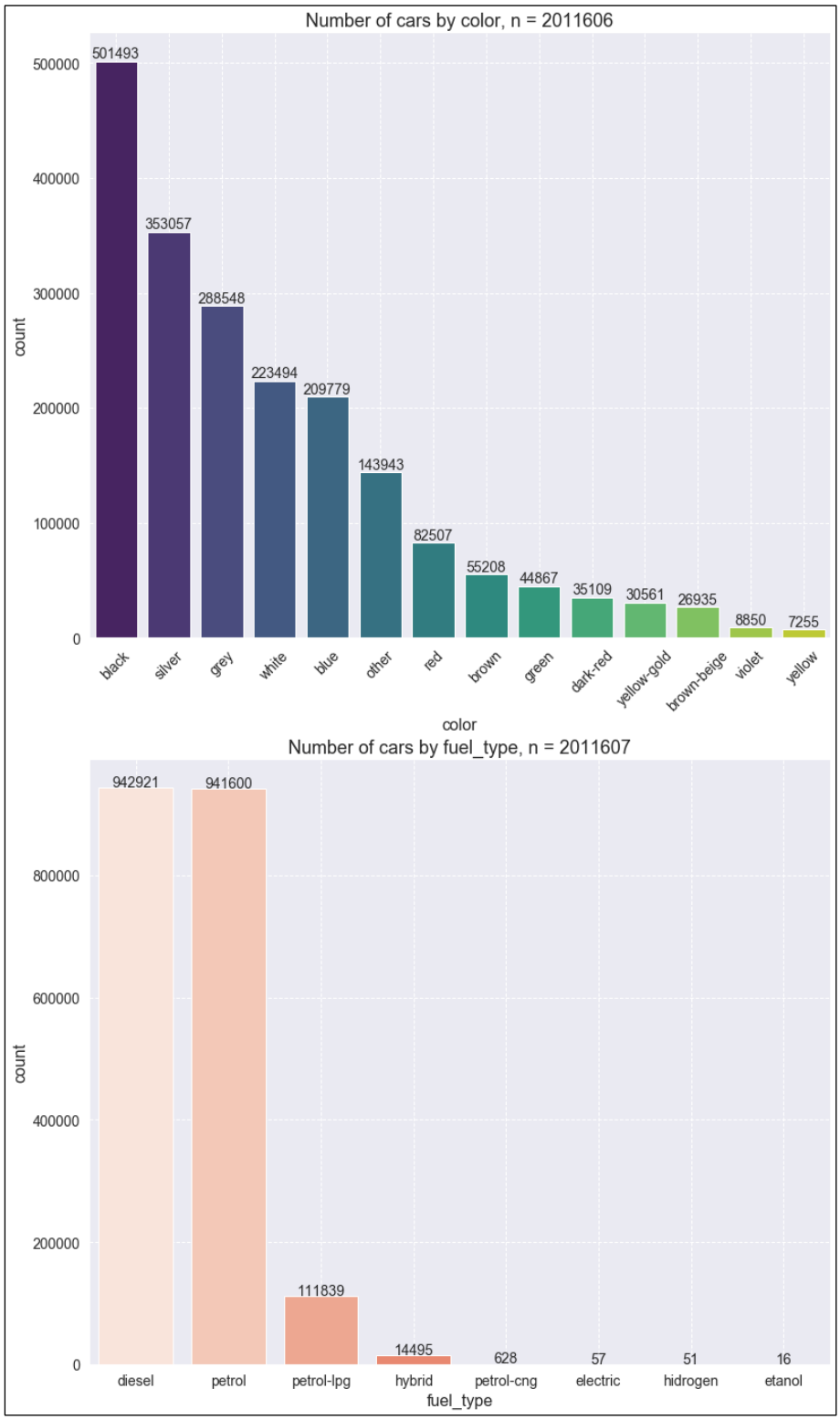

Figure 12: Bar plots for number of cars by *color* and *fuel_type*

Figure 13 is a regression plot, where *vehicle_year* is binned into 60 bins and each price is allocated into one bin. Then, the median of all prices within a bin were plotted as a scatter point. The points are widely scattered until around 1935, mainly due to a low number of cars from these years and because cars of this age are considered as old timers and thus, have a higher price. The median price then steadily decreases until a tipping point around 1998. From thereon, the median prices steadily increase, with big jumps from 2016 to 2019.

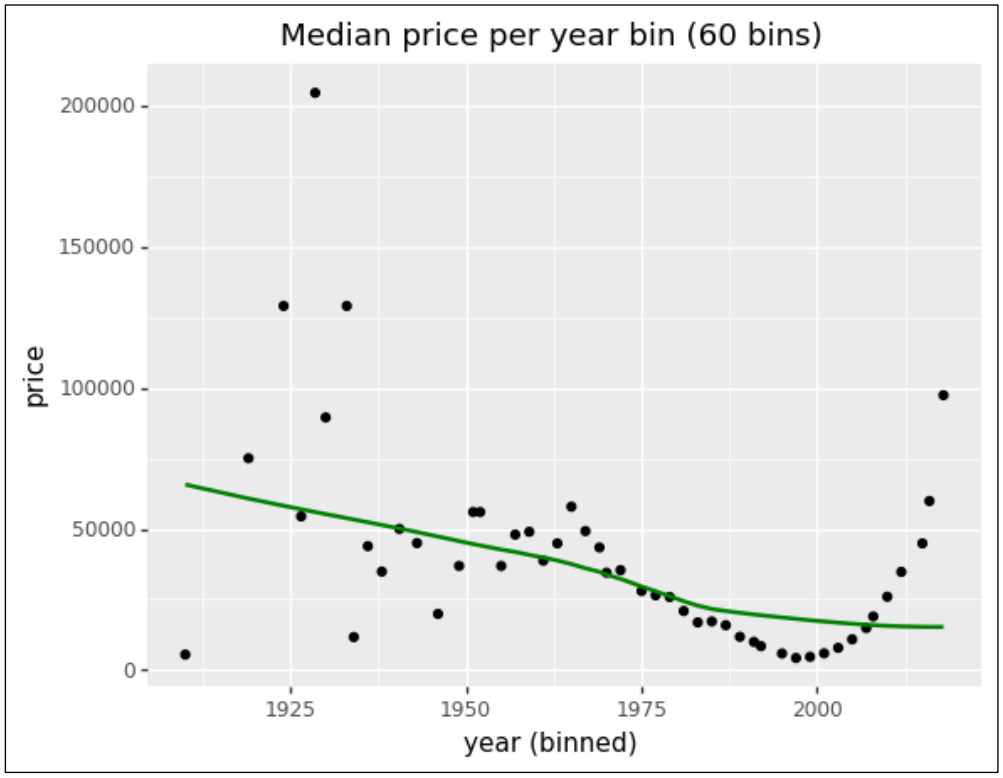

Figure 13: Regression-Scatter plot for binned years (x-axis) and *local_gross_price* (y-axis)

Figure 14 below illustrates the price difference for the top 20 makes (by number of cars). For each car of a top 20 make, the z-score is computed. Then, the z-scores of all cars from a top 20 make were averaged. The three German brands mercedes-benz, bmw and audi are the most expensive cars with volvo as number four. Because these four brands are much more expensive than an average top 20 make car, most brands have a red bar, indicating that their average price is below the average of all top 20 make cars.

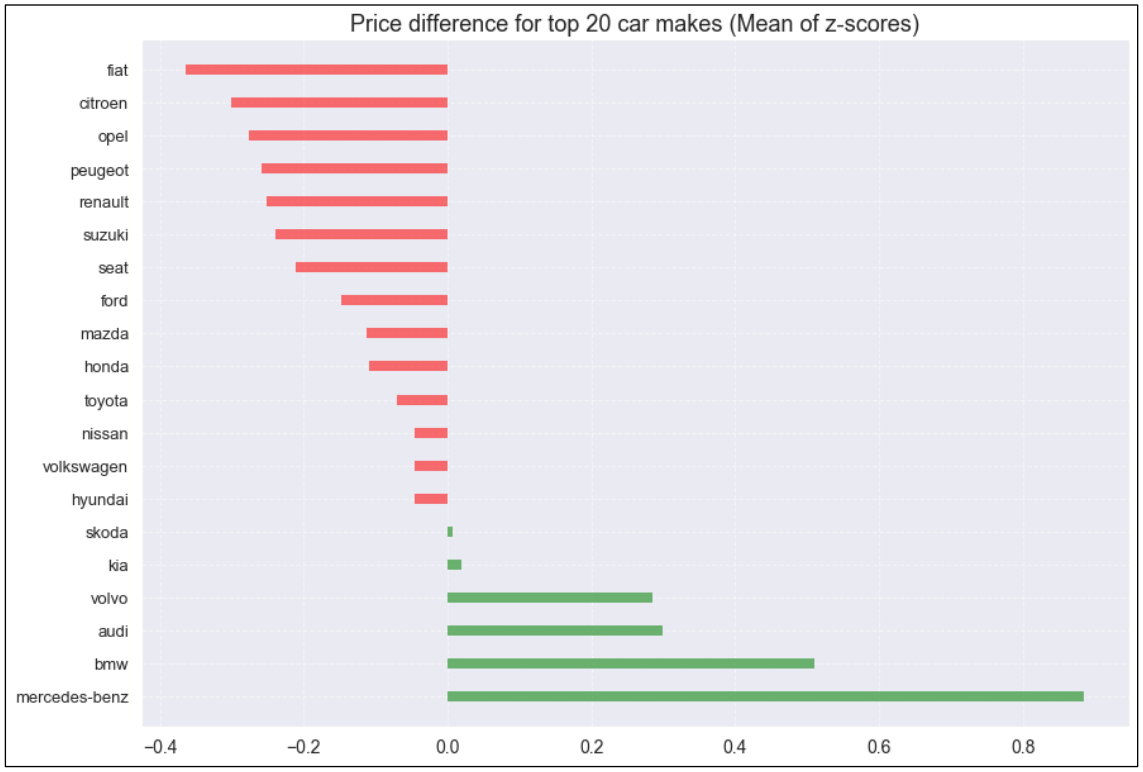

Figure 14: Diverging bar chart showing *local_gross_price* differences for top 20 makes with most cars

Figure 15 below is similar to figure 14 above, but instead of price, mileage is compared amongst the top 20 makes. The methodology behind calculating the z-scores is identical. German cars again dominate with 5 out of the top 6 brands. Asian cars, on average, have a lower mileage with the brands kia, hyundai, suzuki and Nissan ranking at the bottom.

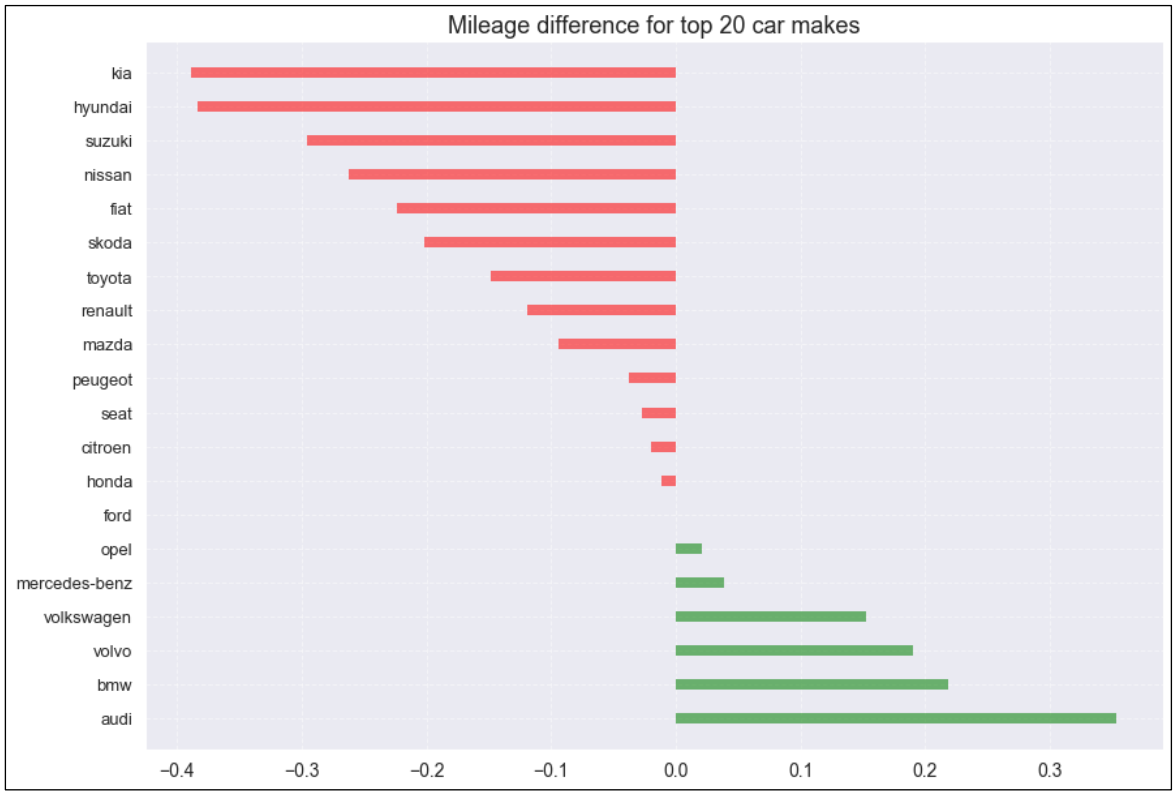

Figure 15: Diverging bar chart showing *mileage* differences for top 20 makes with most cars

The graph below (Figure 16) is a heatmap representing the Pearson correlations between all numerical features in the dataset. The most important correlations are in the bottom row, because a high correlation with the label *local_gross_price* indicates that the feature is a good predictor to explain the variance of *local_gross_price*. Indeed, the correlations with *engine_power*, *mileage* and *vehicle_year* are 0.59, -0.42 and 0.51, respectively. This intuition is also confirmed by the SHAP summary plot presented in the main part of this work, where these three features are deemed as very important for explainability of *local_gross_price.*

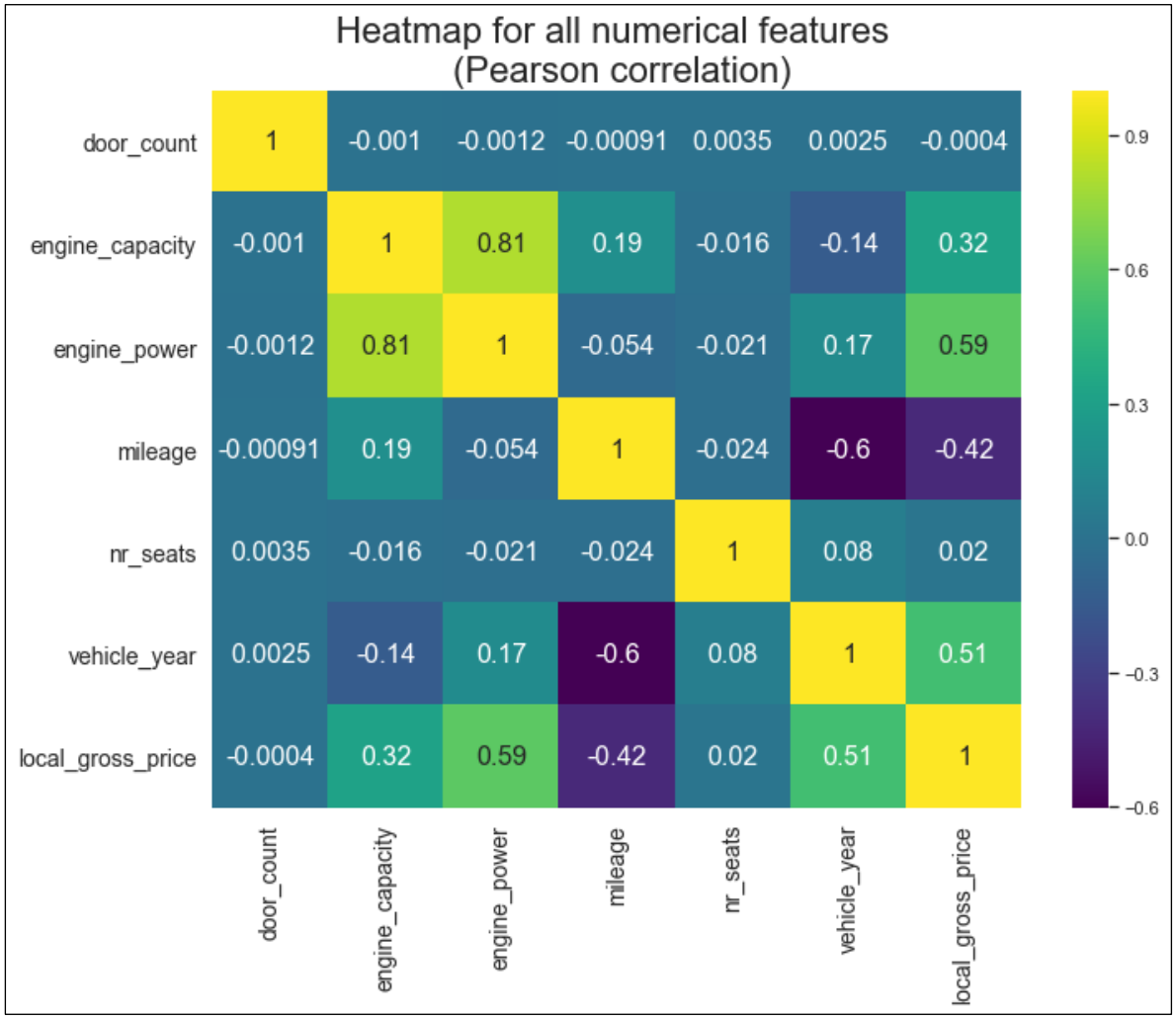

Figure 16: Heatmap (Pearson correlation) for all numerical (excluding binary categorical) features

The last figure of the EDA is presented below. Figure 17 has two subplots, with *vehicle_year* on the x- and *engine_power* on the y-axis. In the first subplot, the median engine power for each of the top 10 makes is plotted as a scatter point. Up until around 2000, the scatter points are distributed without any clear structure. From 2000 on, this changes as the median engine power for all makes slowly increases and converges. Car producers are hence producing, regarding engine power, more and more similar cars.

With respect to the second subplot, illustrating the mean engine power per make, the same pattern can be observed. In addition, two different groups are emerging, colored in orange and green. Here, the rate of engine power increase is slightly higher, meaning that there is a significant number of cars with very high engine power. It seems that producers focus on either the lower or higher engine power group and streamline their production.

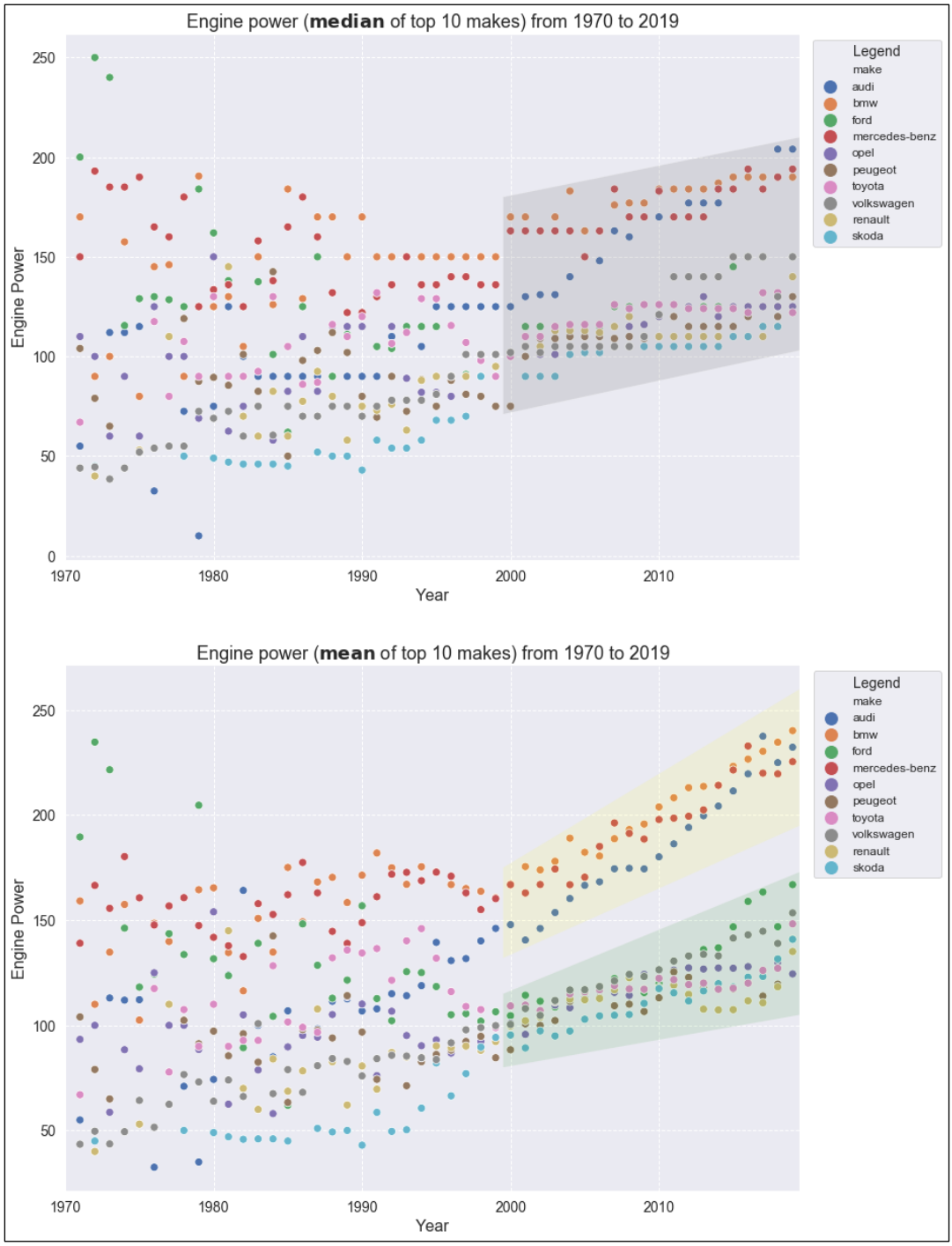

Figure 17: Scatterplots with *vehicle_year* (x-axis) and *engine_power* (y-axis). Colors represent top 10 makes with most cars. First subplot represents median *engine_power,* second subplot the mean

D – Decision Tree

Figure 18 represents a decision tree of a dataset only containing the features *vehicle_year*, *make* and *engine_power*. The root node is split on *vehicle_year* with the criteria *<= 2009.50*. Depending on *vehicle_year*, the next split is then conducted on either *engine_power <= 125.50* or *vehicle_year <= 2016.50.* Although *make* is a string categorical feature, it appears in numerical values, because LightGBM performs transformations under the hood. The leaves contain the final prediction of the tree. E.g. the value of leaf 0 is -898.35. In case a car ends up in this leaf (if vehicle_year <= 2009.50 and engine_power <=125.50), 898.35 is deducted from the overall prediction of all trees.

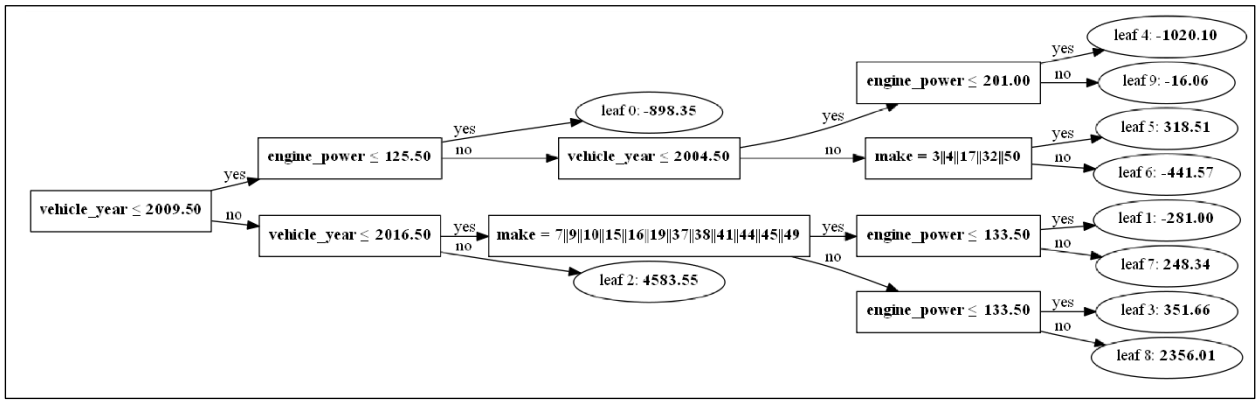

Figure 18: Decision Tree of a LightGBM model for a reduced dataset with only three features.

E – Search space and hyperparameters

Table 2 below shows the search space for the first Hyperopt run.

Name	Data type	Minimum	Maximum	Step size	Distribution
bagging_fraction	float	0.5	0.95	0.025	uniform
bagging_freq	Int	1	6	1	uniform
bin_construct_sample_cnt	int	10,000	300,000	10,000	uniform
cat_l2	float	6	14	0.25	uniform
cat_smooth	float	6	14	0.25	uniform
cegb_tradeoff	float	0.93	1.0	0.01	uniform
feature_fraction	float	0.5	0.925	0.025	uniform
lambda_l1	float	0.1	1.0	0.05	uniform
lambda_l2	float	0.1	1.0	0.05	uniform
learning_rate	float	0.05	0.15	0.005	uniform
min_sum_hessian_in_leaf	float	0.06	0.18	0.01	uniform
min_data_in_leaf	int	6	40	2	uniform
n_estimators	int	50	100	10	uniform
num_leaves	int	3400	6400	200	uniform

Table 2: Search space for first Hyperopt run

Below, the full result of the first Hyperopt run with the search space from Table 2 is shown (Figure 19). The best loss achieved is an MdAPE of 7.97%.

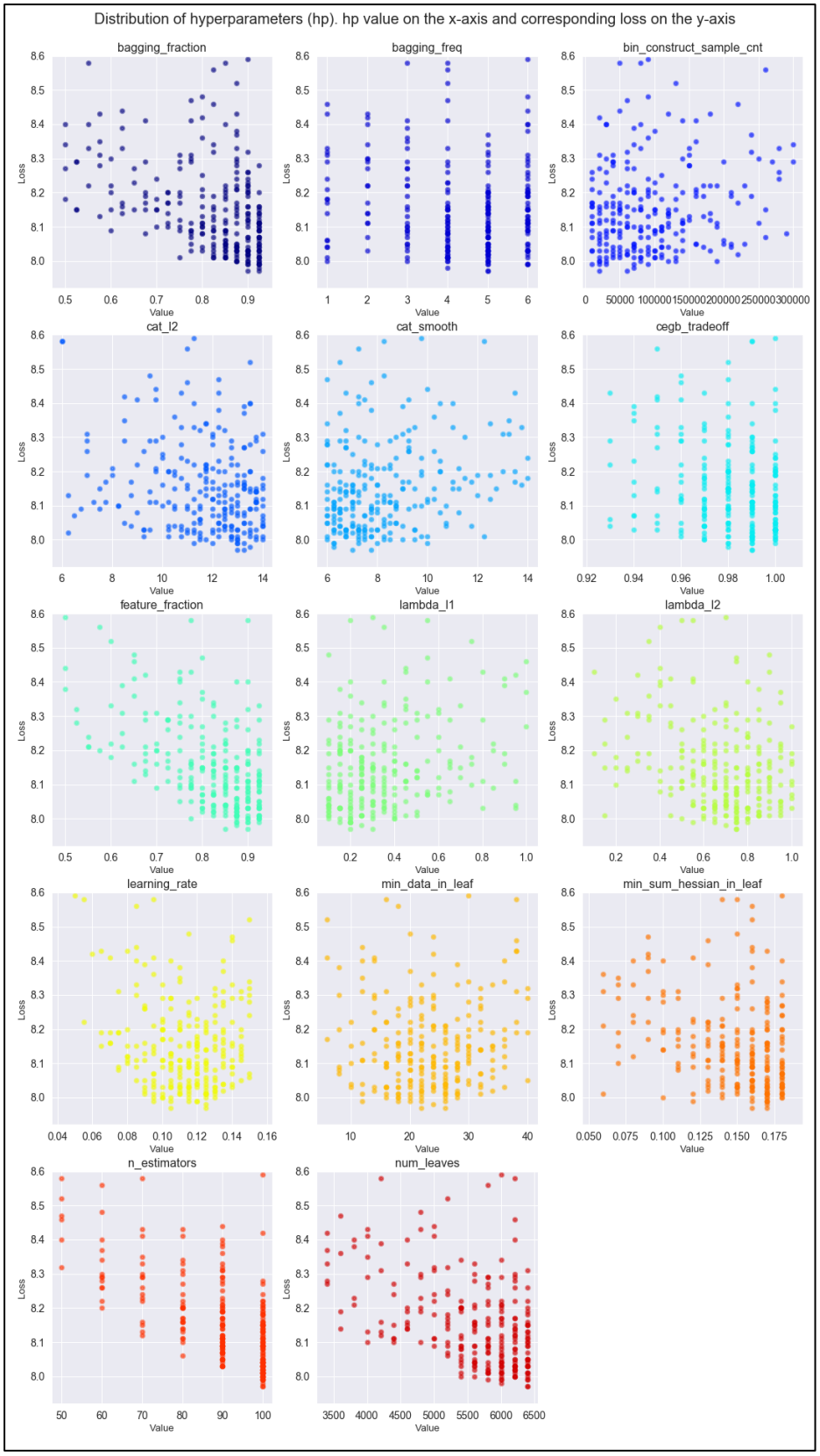

Figure 19: Scatterplot for the distribution of the hyperparameters during a Hyperopt run no. 1

Table 3 below shows the search space for the second Hyperopt run:

Name	Data type	Minimum	Maximum	Step size	Distribution
bagging_fraction	float	0.75	0.925	0.025	uniform
bagging_freq	Int	4	6	1	uniform
bin_construct_sample_cnt	int	10,000	150,000	10,000	uniform
cat_l2	float	11	16	0.25	uniform
cat_smooth	float	5	10	0.2	uniform
cegb_tradeoff	float	0.97	1.0	0.01	uniform
feature_fraction	float	0.75	0.925	0.025	uniform
lambda_l1	float	0.1	0.5	0.01	uniform
lambda_l2	float	0.5	0.85	0.01	uniform
learning_rate	float	0.085	0.13	0.005	uniform
min_sum_hessian_in_leaf	float	0.12	0.20	0.01	uniform
min_data_in_leaf	int	15	30	1	uniform
n_estimators	int	80	120	10	uniform
num_leaves	int	5000	6400	100	uniform

Table 3: Search space for second Hyperopt run

Below, the full result of the second Hyperopt run with the search space from Table 3 is shown (Figure 20). The best loss achieved is an MdAPE of 7.92.

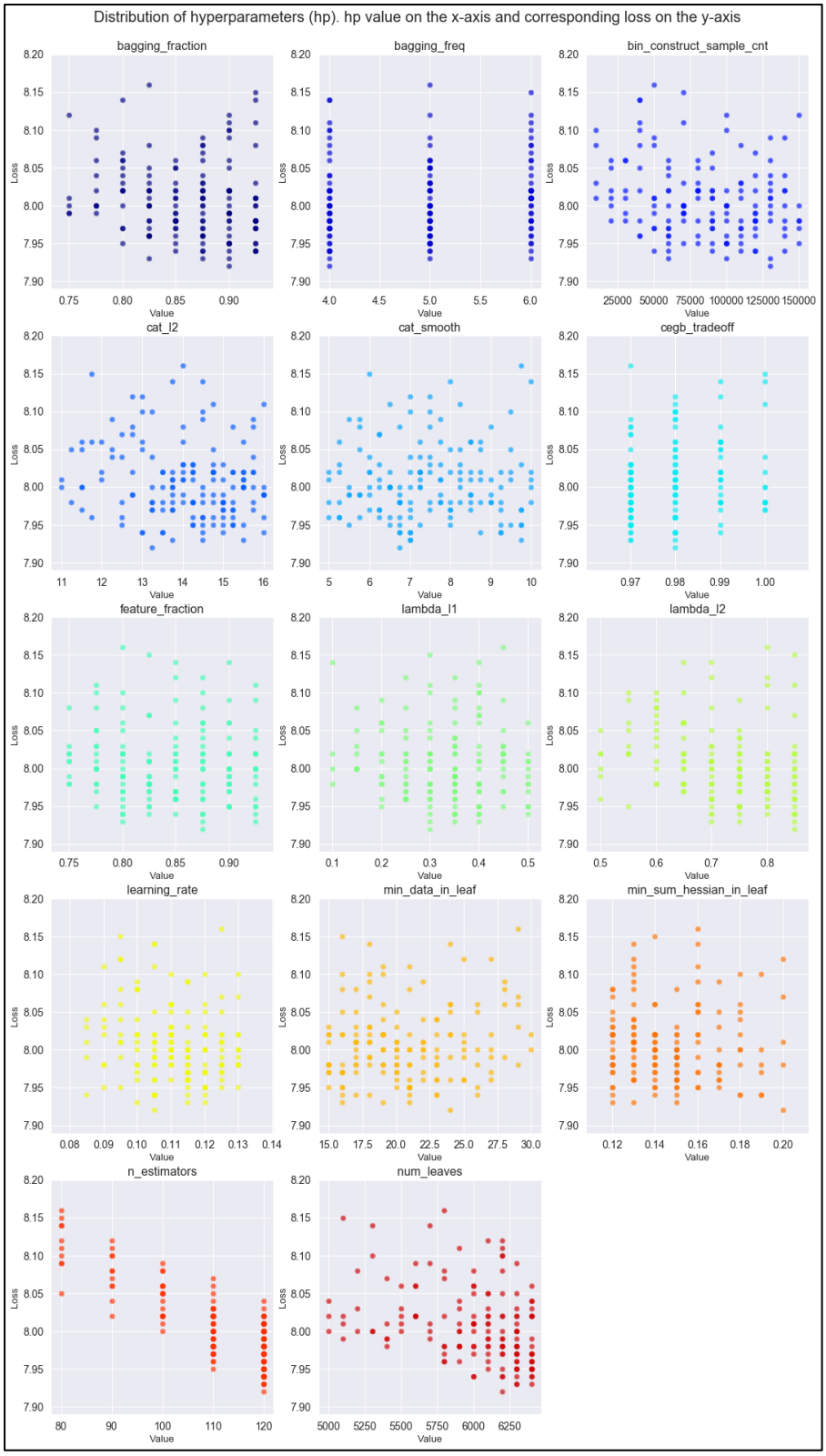

Figure 20: Scatterplot for the distribution of the hyperparameters during a Hyperopt run no. 2

Table 4 below shows the search space for the last Hyperopt run:

Name	Data type	Minimum	Maximum	Step size	Distribution
bagging_fraction	float	0.80	0.90	0.025	uniform
bagging_freq	Int	4	4	-	-
bin_construct_sample_cnt	int	50,000	130,000	10,000	uniform
cat_l2	float	13	15.5	0.1	uniform
cat_smooth	float	6.5	10	0.1	uniform
cegb_tradeoff	float	0.97	0.98	0.005	uniform
feature_fraction	float	0.80	0.925	0.025	uniform
lambda_l1	float	0.3	0.5	0.025	uniform
lambda_l2	float	0.7	0.95	0.025	uniform
learning_rate	float	0.1	0.125	0.005	uniform
min_sum_hessian_in_leaf	float	0.11	0.16	0.01	uniform
min_data_in_leaf	int	15	25	1	uniform
n_estimators	int	80	120	10	uniform
num_leaves	int	5500	6400	50	uniform

Table 4: Search space for the third Hyperopt run

Below, the full result of the third Hyperopt run with the search space from Table 4 is shown (Figure 21). The best loss achieved is an MdAPE of 7.9%.

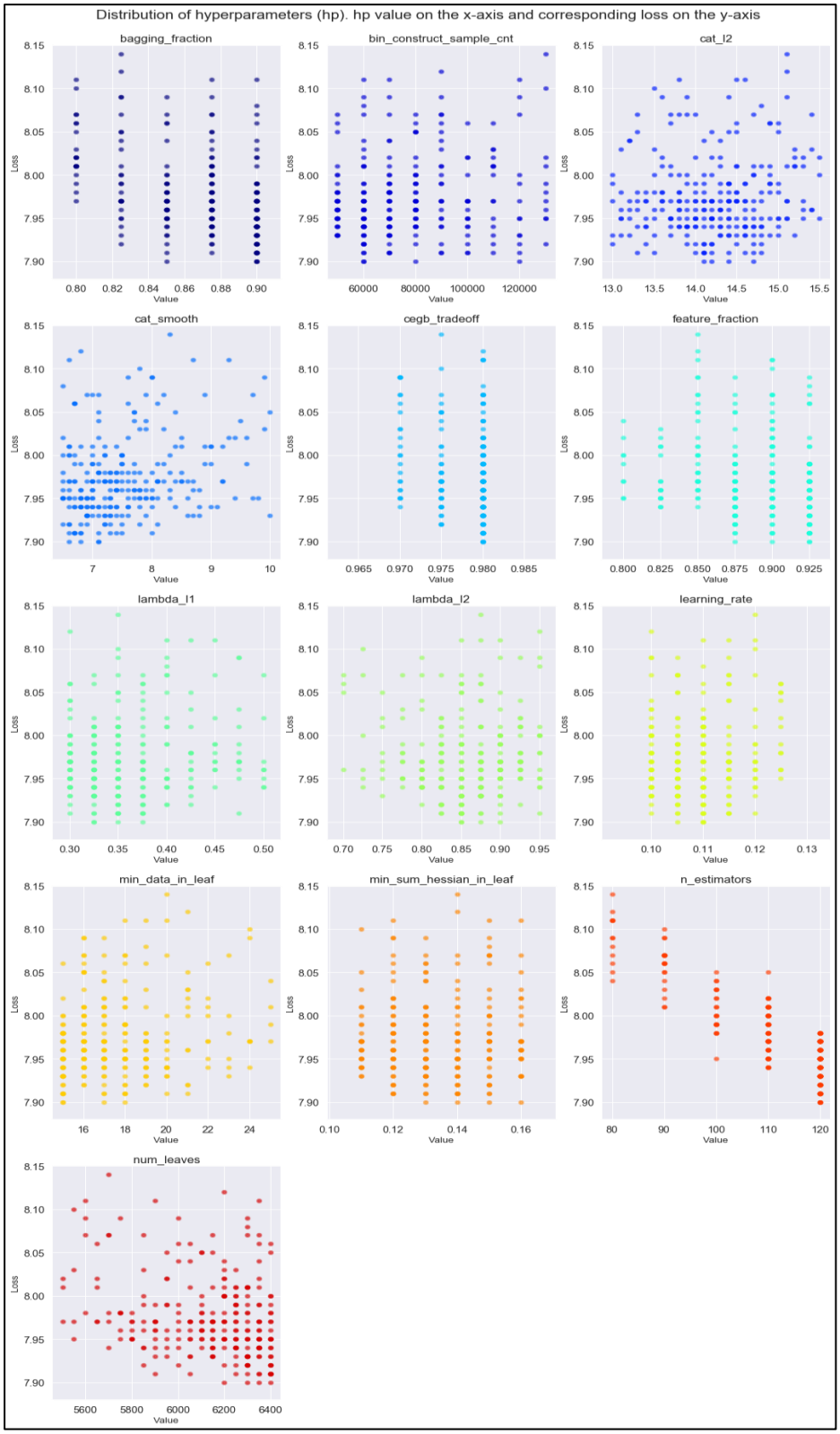

Figure 21: Scatterplot for the distribution of the hyperparameters during a Hyperopt run no. 3

The final values and an explanation (LightGBM documentation, see *References*) for the hyperparameters are in Table 5 below. The table also includes the parameters *alpha*, *early_stopping_round*, *max_depth*, *metric*, *objective* and *seed*, which are all fixed to a certain value.

Name	**Final value**	**Explanation**
bagging_fraction	0.9	will randomly select part of data without resampling
bagging_freq	4	frequency for bagging, performs bagging at every kth iteration
bin_construct_sample_cnt	60,000	number of data that sampled to construct histogram bins
cat_l2	14.1	L2 regularization in categorical split
cat_smooth	7.1	can reduce the effect of noises in categorical features, especially for categories with few data
cegb_tradeoff	0.98	cost-effective gradient boosting multiplier for all penalties
feature_fraction	0.925	will randomly select part of features on each iteration
lambda_l1	0.325	L1 regularization
lambda_l2	0.85	L2 regularization
learning_rate	0.105	shrinkage/learning rate, eta
min_sum_hessian_in_leaf	0.13	minimal sum hessian in one leaf
min_data_in_leaf	17	minimal number of data in one leaf
n_estimators	120	number of boosting iterations/decision trees
num_leaves	6400	maximum number of leaves in one tree
alpha	0.5	predicted quantile
early_stopping_round	10	will stop training if metric doesn't improve in last 10 rounds
max_depth	-1	limit the max depth for trees, -1 means no limit
metric	'quantile'	metric to be evaluated on the evaluation set
objective	'quantile'	objective of the LightGBM model
seed	42	ensure reproducibility

Table 5: Hyperparameters with corresponding value (chosen by Hyperopt) and explanation (extracted from Documentation)

F – SHAP plots

In the following, three SHAP plots will be presented. Figure 22 shows the explanation for single prediction. The *base value* is the average model output over the training dataset that is passed to the SHAP explainer (equal to *explainer.expected_value*). Features that push the prediction higher are colored in red, and those pushing the prediction lower are colored in blue. The *output value* is the prediction for this specific car. *vehicle_year* with a value of 2013.0 is rather new compared to other cars and thus pushes the prediction up by around 6,000 Zł. Mileage on the other hand is relatively high (301,000) and causes a decrease in value of around 10,000. It is interesting to note that feature interaction really matters. A *vehicle_year* value of 2013.0 not always by circa 6,000. The effect on the prediction depends on the values of the other features.

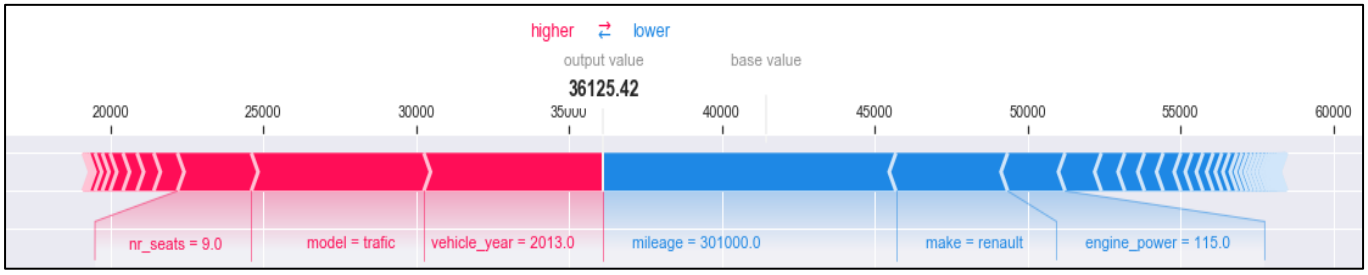

Figure 22: SHAP Force Plot explaining a single prediction

Figure 23 below is a feature importance bar chart for the 20 most influential features. On the y-axis are the features and on the x-axis are the mean absolute SHAP values. They stand for the average impact on model output, i.e. if the mean absolute SHAP value for *vehicle_year* is 17,500, then, on average, *vehicle_year* increases or decreases the prediction by 17,500. One can observe that *vehicle_year* by far is the most important predictor with a mean absolute SHAP value of around 19,000. *engine_power* and *mileage*, also numerical features, lie in between 5,000 and 7,500. *Make* and *model* are the first string categorical features and have mean absolute SHAP values of around 3,000. The overall conclusions are that only a handful of features determine the price of a car and that the heatmap (Figure 16) correctly predicted the most important (numerical) predictors.

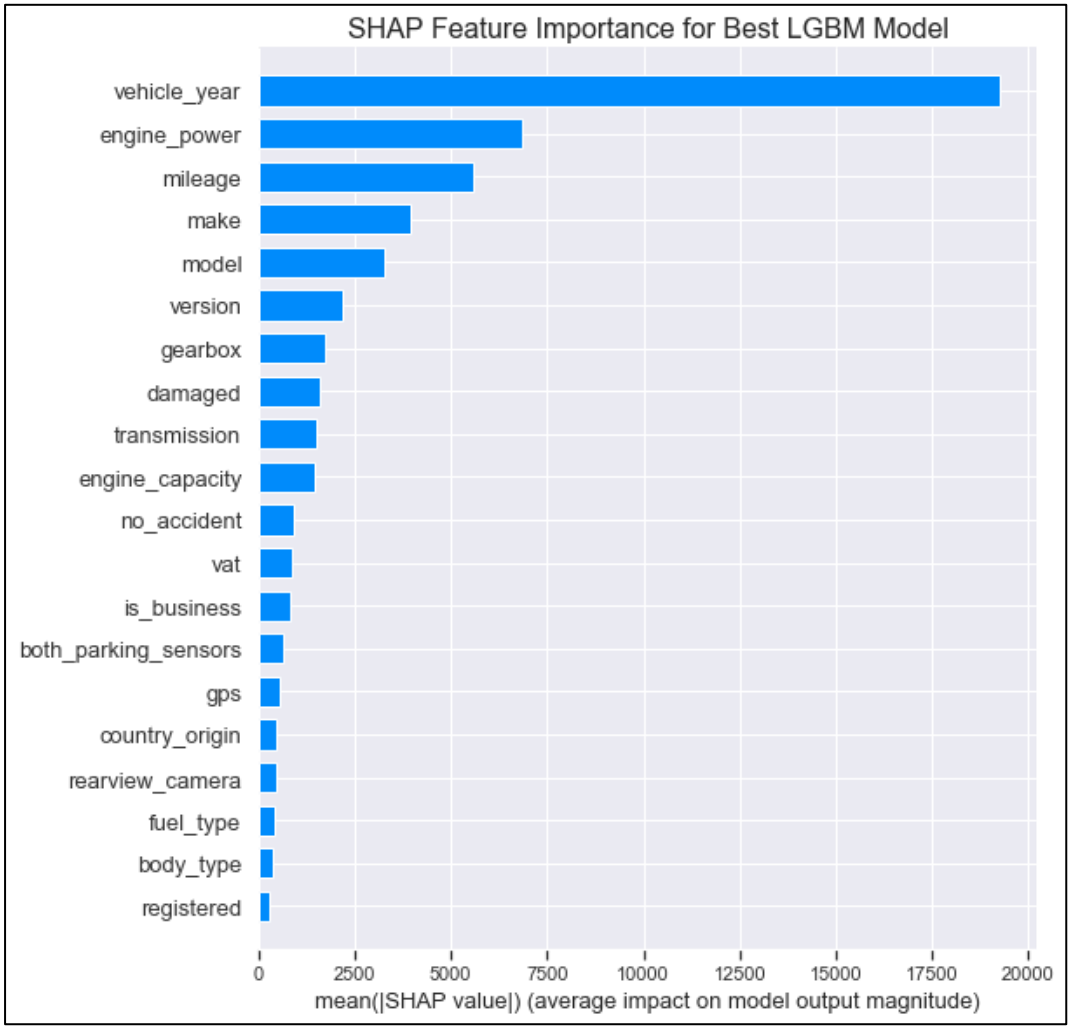

Figure 23: Bar Plot illustrating the feature importance, features on the x-axis and mean absolute SHAP values on the y-axis

Figure 24 below shows the full summary plot of Section 4 of the main part of this work.

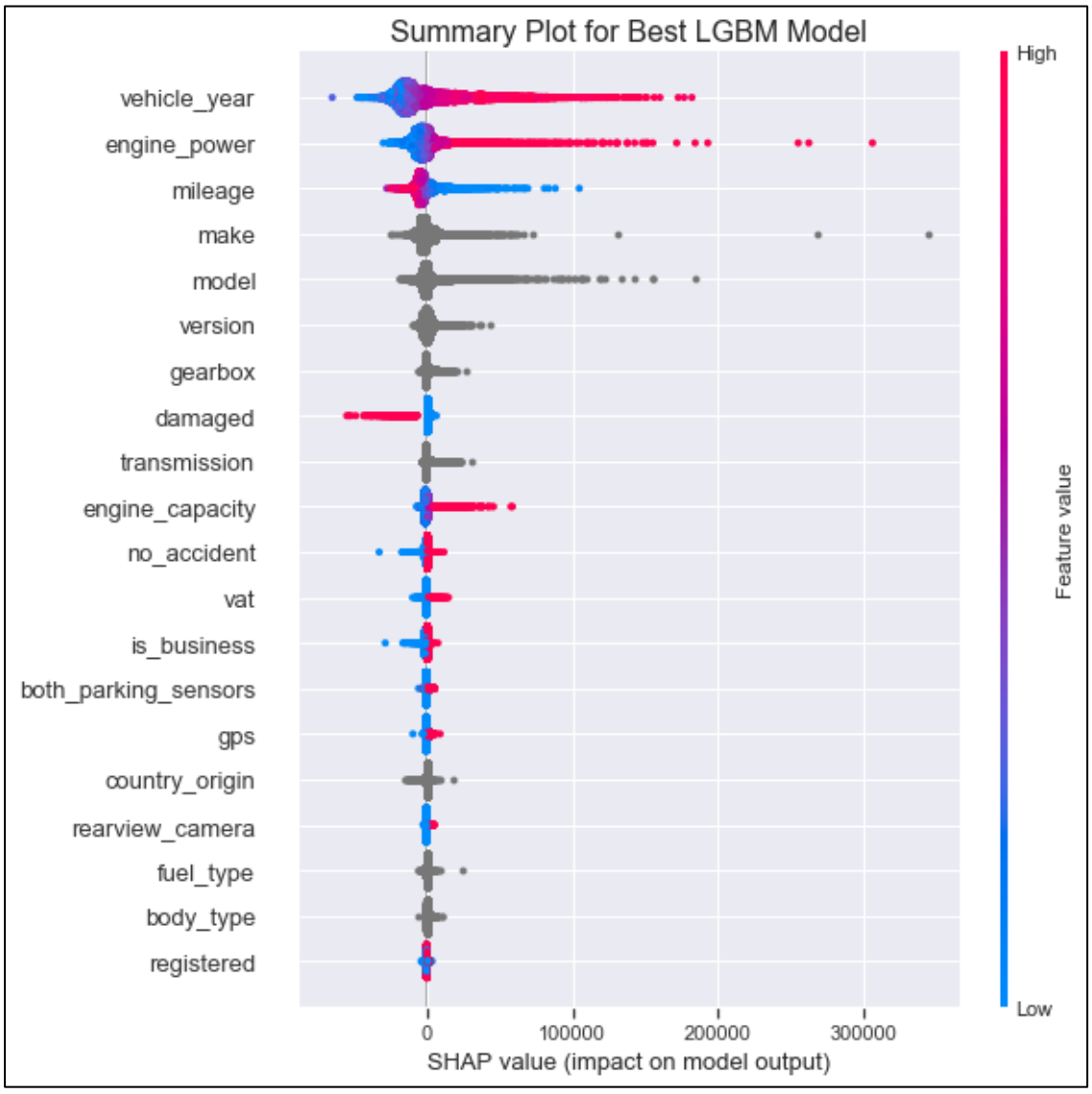

Figure 24: Summary plot for the20 most important features

G – Comparison to Baseline Visualizations

The first plot (Figure 25) compares the performance of the three LightGBM models with optimized hyperparameters but trained with different features. One model is trained with the features selected by the Boruta algorithm, one is trained with all features and one only with the basic features.

The second plot (Figure 26) on the other hand illustrates the performance of the XGBoost and random forest models.

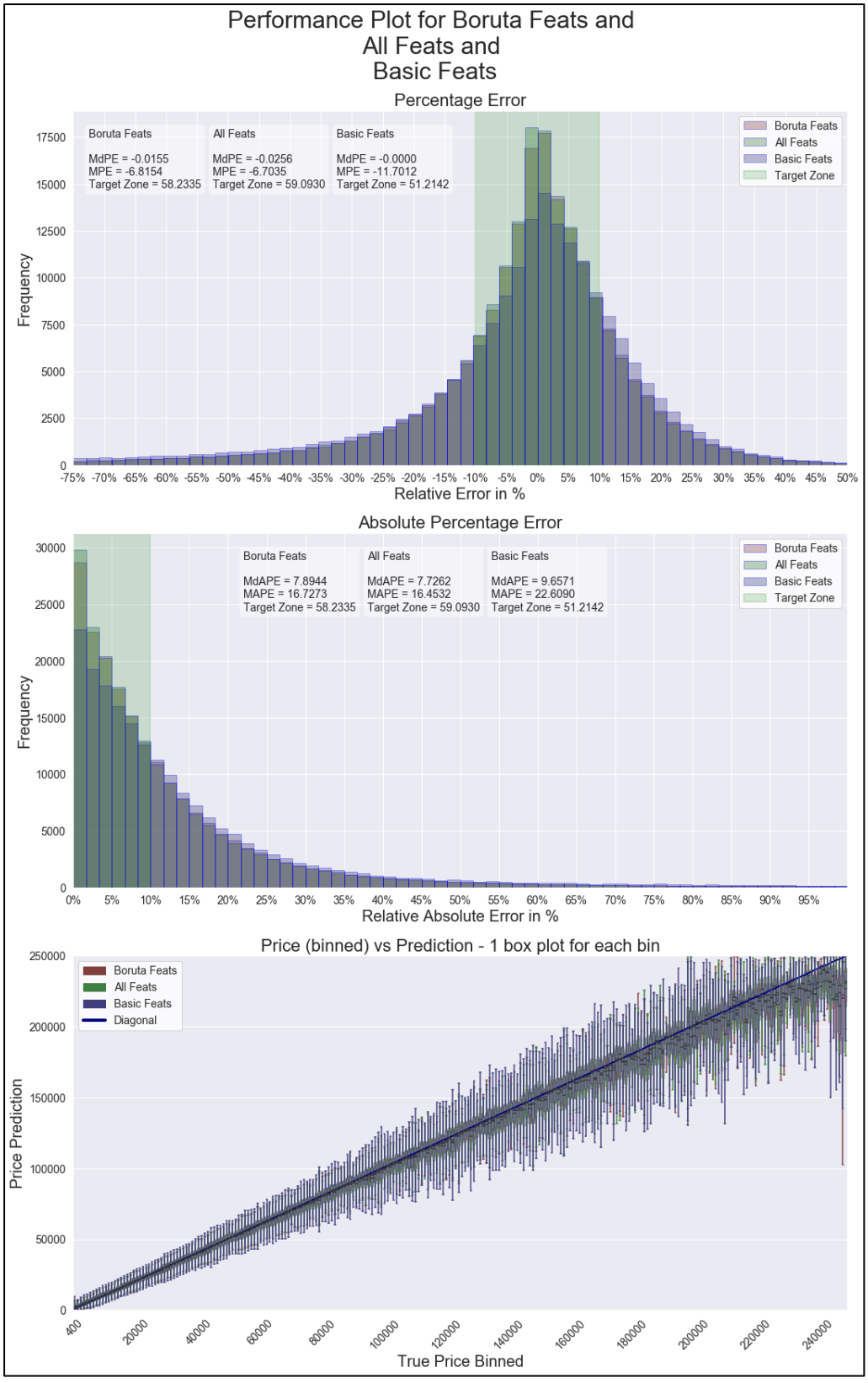

Figure 25: Comparison of performance of three LightGBM models with different features (Boruta vs All vs Basic)

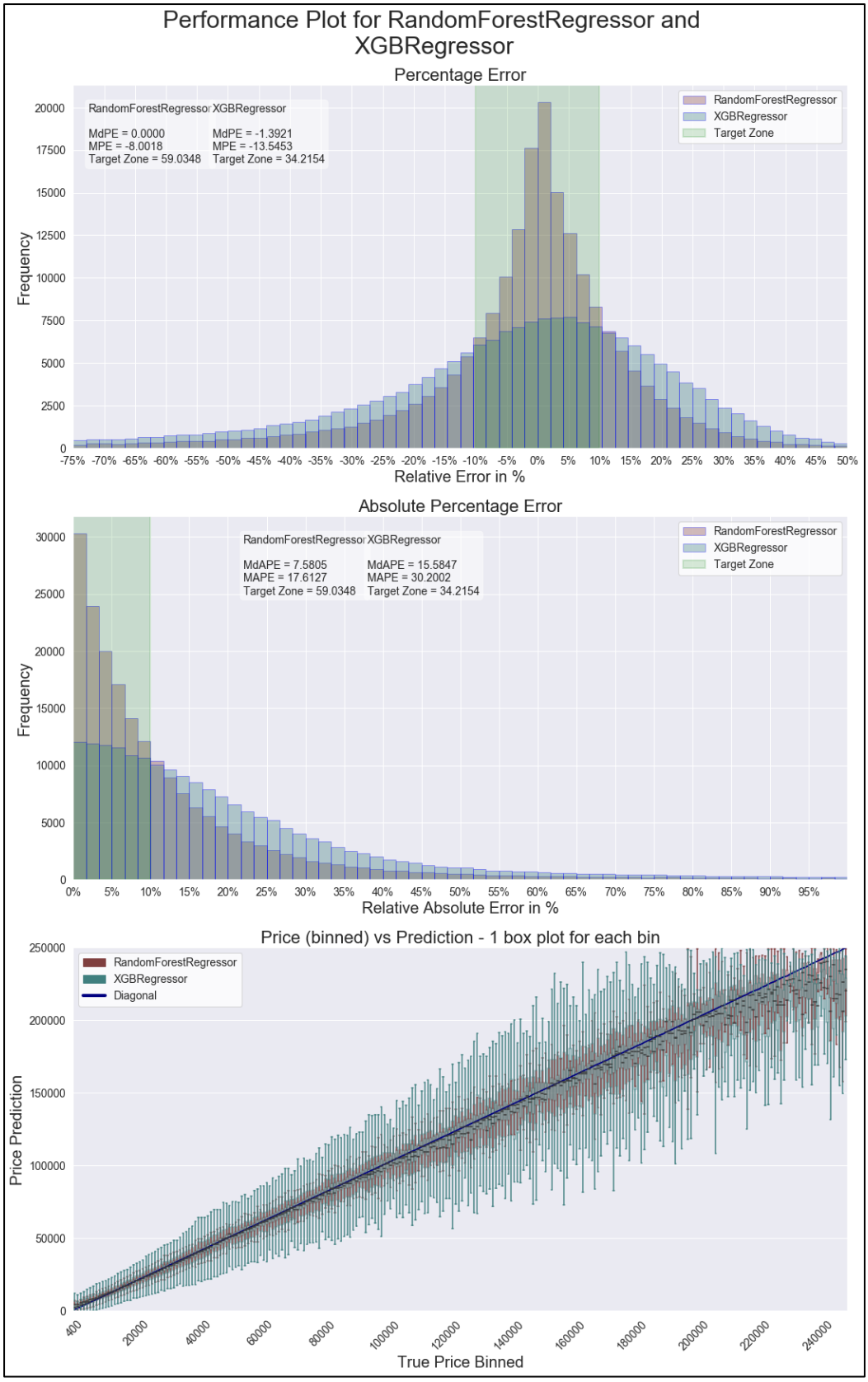

Figure 26: Comparison of performance of XGBoost and random forest models