WB

A PREPRINT

Kacper Grzymkowski
Faculty of Mathematics and Information Science
Warsaw University of Technology
Poland
kacper.grzymkowski.stud@pw.edu.pl

Dominik Kędzierski
Faculty of Mathematics and Information Science
Warsaw University of Technology
Poland
dominik.kedzierski.stud@pw.edu.pl

Jakub Piwko
Faculty of Mathematics and Information Science
Warsaw University of Technology
Poland
jakub.piwko.stud@pw.edu.pl

May 26, 2022

ABSTRACT

Machine Learning has become powerful tool for data scientists in wide variety of applications. Particularly interesting problem, which shows efficiency of artificial intelligence algorithms, is real estate appraisal. This paper examines and compares popular or recently developed tree-based models including Random Forest, XGBoost, LightGBM and Catboost for this regression task in R. Research is conducted on data describing numerous aspects of residential homes in Ames, Iowa, United States. We perform exploratory data analysis to investigate set characteristics. Preprocessing operations were selected in order to achieve more intuitive way of variables coding. Next step is building models and tuning hyperparameters. Different methods for finding the most optimal parameters set were used for each model. We concentrate on diagnosis designed to determine the best model regarding minimizing of RMSE scores but also their sustainability. Finally, explainable artificial intelligence methods were used to unveil decision-making process of best model. We were able to obtain knowledge about most influential variables and how they contribute to final price prediction.

Keywords Machine Learning · Explainable Artificial Intelligence · Real estate · Property appraisal

1 Introduction

Machine Learning had been developed remarkably recently to become powerful tool for data scientists. The idea of giving a computer huge amount of data and receiving straightforward results has become solution to many modern world problems. Help of artificial intelligence seems to be convenient and effective way of reducing human effort, especially in cases when plenty of information is to be inspected. Machine Learning algorithms are used extensively around us, whether we are aware of it or not. Self driving cars, speech recognition, natural language processing, product recommendations are only a few applications from wide range of areas where people rely on artificial intelligence.

In this paper we will focus on the problem of real estate appraisal. Machine learning algorithms can be successfully used to predict price having knowledge about property conditions. Inspired by published works Masías et al. [2016], Wei et al. [2022], Turnbull and van der Vlist [2022], Tchuente and Nyawa [2021], van de Minne et al. [2021] addressing this issue, we want to study how efficient are artificial intelligence algorithms in matter of house price prediction. For this particular research, only a few of algorithmic techniques available in R language are taken into account. We put emphasis on tree based models. Well-known Random Forest, novel XGBoost, LightGBM and Catboost were subjected to the test. While Random Forest is a basic ensemble learning method, which is thoroughly explained in Breiman [2001], the latter are examples of algorithms using gradient boosting technique that combines weak decision tree

learners into a single strong learner by minimizing errors. Each of tree models have different methods of manipulating data as outlined in Chen and Guestrin [2016] for XGBoost, Ke et al. [2017] for LightGBM and Prokhorenkova et al. [2017] for Catboost. Thanks to such a choice, election of best model regarding real estate appraisal will be easier.

Our research is conducted on data describing 2919 properties located in Ames, Ihoa in United States. Set consists of 80 columns describing house features or their conditions and target variable `SalePrice`. Performance of Exploratory Data Analysis shows that there is relatively large number of missing values and many variables express quality scale of chosen characteristics. In consequence, preprocessing lead mainly to encoding change of NA values and quality related columns. After getting acquainted with algorithmic techniques of our models, the next step is training models. The most important part of this stage is hyperparameters tuning. This operation is performed with use of Caret package in R Kuhn [2008]. Furthermore, we are creating our own simple methods to frisk best hyperparameters values. Corresponding parameters grids of different values are being fitted to models. As a result, a large collection of models is received. Subsequently, models are evaluated with emphasis on Root Mean Squared Error (RMSE). Analysis determines 5 best models of each type. Final phase is comparison of these models. Examination of residuals values and stability leads to selection of one best model, which is Catboost with specific parameters. The final stage of our research is running Responsible Machine Learning methods using DALEX package Biecek [2018]. Results help explain black box model and understand which variables are most influential to final prediction and how each feature contribute to the price.

Diagnosis of the final Catboost model is showing that it minimizes RMSE score on both train and test set. Although we trained several models that appeared to have better stability, none of them reached preferable metric values. Examination of residuals leads to suppose that model have tendency to overestimate house prices. Explainable Artificial Intelligence analysis reveals that the most powerful variables are those expressing overall quality and connected to living area of properties. Moreover, our model is likely to value property base on number of rooms and presence of garage and basement.

2 Data

2.1 Data Description

Data analysed in our article was originally collected by Dean De Cock for use in data science education. It is publicly available through an advanced regression techniques machine learning competition on kaggle.com. The data describes 2919 dwellings located in Ames, Ihoa, United States. It is split between a training set of 1460 observations and a test set of 1459 observations. There are 79 feature columns describing various aspects of the dwelling such as its lot area, quality of amenities or the distance to the nearest railroad. The label is the sale price of the real estate.

2.2 Conclusions from Exploratory Data Analysis

Early analysis found that there are significant counts of missing values within the dataset. However, upon further inspection, those missing values were tied closely to the structure of the data. Certain features, such as *Basement Quality* or *Swimming Pool Quality* do not make sense when those features are missing. As such, we decided to impute the missing data by creating new categories representing a missing feature.

What is interesting in our data set is presence of many attributes that hold some kind of rating. For example, *Overall Quality* provides information about the overall quality of dwelling. We could not find a concrete methodology with which this data was collected. It is possible for these ratings to be subjective. Additionally some of these variables were in a categorical format, and some already encoded into an integer format. We categorized all feature columns into 47 discrete, 16 ordered discrete, and 16 continuous features.

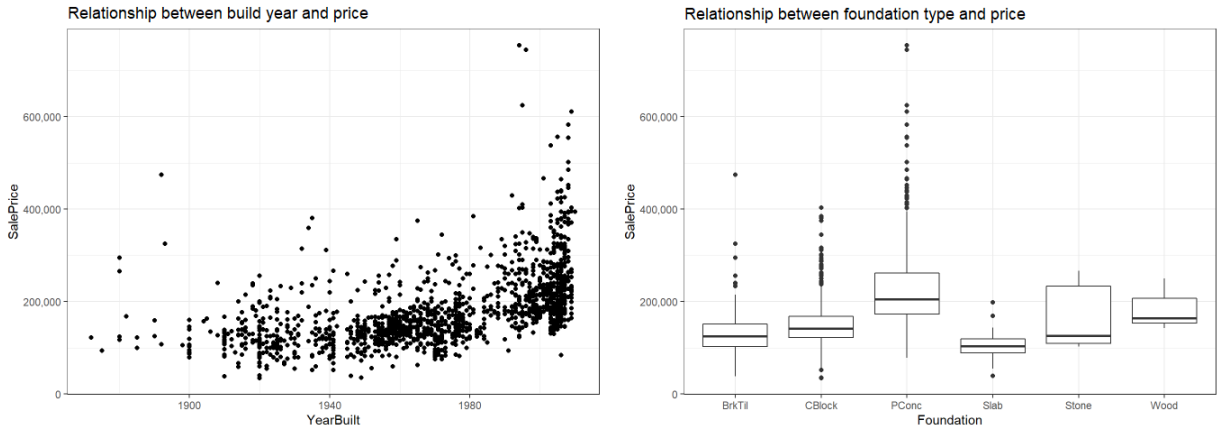

Figure 1: Relationship between dependant and target variable

Next step was analysing some one-dimensional and multi-dimensional dependencies by visualising them in histograms, box plots and scatter plots. As a result, we found some variables that seem to impact appraisal of real estate as shown on plots 2.2. For example, built year, type of foundation and area showed some relation with price of dwellings, so we hope that these columns will be significant in predicting prices.

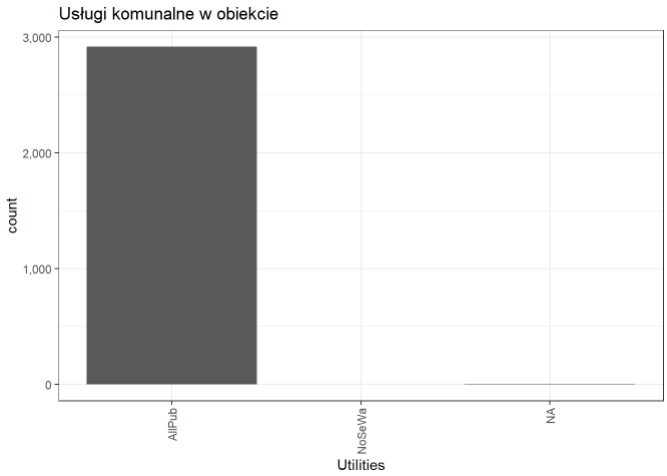

Figure 2: Type of utilities count

Exploratory Analysis outcome has also implied that in some columns there is a huge dominance of one value, while other types of observations are negligible as presented on plot 2.2. There are also variables related to extra features like pool or tennis courts and only a small part of dwellings presented in frame own them. This is a hint to encode this kind of columns in a different way or just not pay that much attention to them in a process.

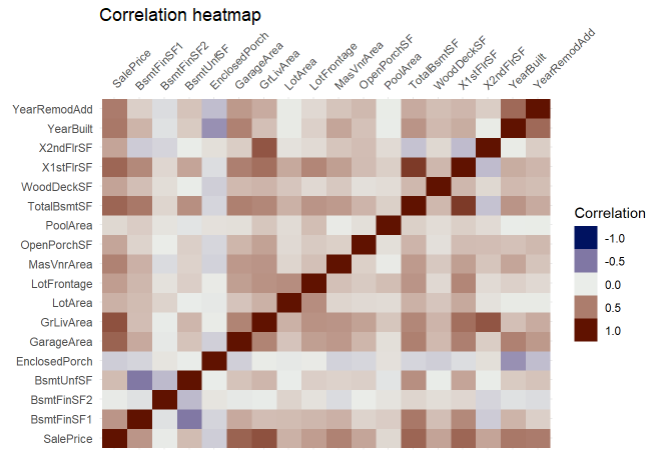

Figure 3: Correlation heatmap

Our final stage was to look more broadly at continuous variables, so we created correlation heatmap 2.2. We haven't found any particularly high correlations between variables. But we observed that negative correlations are being outnumbered, which can be related to specification of our data.

3 Model selection

Data are explored and properly processed. The next crucial step of our research is building models. Studying this particular problem of property appraisal is a great opportunity to test some of lately popular tree-based models. In our research we are focusing on following algorithms:

- Random Forest
- XGBoost
- LightGBM
- Catboost

All of them rely on ensemble of many decision trees to improve forecasting, but use different types of variables coding, individual methods for finding split points and gradient boosting in case of 3 last types. All of these factors raise expectations about their performance and begs the question, which one is going to be the best in case of real estate price prediction.

Models training was preceded by process of writing code that were to wrap building procedures for each model type into single function. As said before, our models differs if it comes to preparation of data or model construction, so our research group divided and worked on automation of training models for each algorithm separately.

After reaching the goal, parameters tuning is to be conducted. Every model type has its own and in some cases exceptionally long list of parameters to choose from. Decision was made to perform and adjust tuning for every model kind and then diagnose them using proven metrics and visualisations, with emphasis of root mean square error.

3.1 CatBoost

CatBoost hyperparameters were selected by a combined grid and random search approach. The search was performed over parameters exported from the `catboost` Prokhorenkova et al. [2017] package into the `caret` Kuhn [2008] API, namely: `iterations`, `depth`, `l2_leaf_reg`, `learning_rate`, `rsm`. Initial grid search found `learning_rate` and `iterations` to be the most important hyperparameters. These findings were used to refine the search space and a random search was performed afterwards. Results of tuning are presented in table 1.

3.2 LightGBM

Attempts of LightGBM training started with the moment of preprocessing. Examination of different operations preceding learning has shown that the best scores are reached by selecting only automatic handling data with pa-

RMSE Train	RMSE Test	depth	iterations	learning_rate	l2_leaf_reg	rsm
3312.41	24471.06	5	800	0.1	0.05	0.75
3906.97	24483.43	5	700	0.1	0.05	0.75
5604.79	24563.16	5	500	0.1	0.05	0.75
3280.82	25975.99	4	1400	0.1	0.01	0.90
397.50	24430.67	6	1200	0.1	0.01	0.90

Table 1: Catboost hyperparameter tuning results.

rameters `autofactor` and `forceconvert` set to TRUE. These variables are available in function implemented beforehand. Decision was made that the best way to find the most sustainable set of training parameters is to create our own grid of hyperparameters and pass them randomly to models. Tuning principles for lightGBM were taken into consideration during selection of values. As a result, optimal random grid of parameters was was constructed, including: `max_depth, num_leaves, bagging_fraction, feature_fraction, learning_rate, num_iterations, min_data_in_leaf, lambd_l1`. Then again, after fitting 200 models, we examined the RMSE and MAE scores on train and test sets and chose best 5 regarding these metrics. Table 2 shows final values for each parameter of our top 5 models.

	max_depth	num_leaves	num_ iterations	learning_rate	bagging_ fraction	feature_ fraction	lamba_l1	min_data_ in_leaf
1	3	2	242	0.06619538	0.572464	0.4015881	1.37272030	31
2	4	2	182	0.18479657	0.5591560	0.7143656	0.02852181	39
3	4	2	54	0.14623834	0.8445833	0.6481415	1.24955739	27
4	3	2	347	0.17079440	0.5978079	0.4352913	0.72251183	33
5	3	2	88	0.14964913	0.9925450	0.6002572	0.40267346	21

Table 2: Parameters of best LightGBM models

In table 3 RMSE of models for training and test set in corresponding order was collected. Surprisingly, models generated with lightGBM are very stable. The difference between scores on train and test set is diminutive. It turned out that this kind of tree model is the best if to comes to sustainability in our research. Unfortunately, metrics alone are not so impressive. The threshold of 40000 on test data was barely crossed.

	RMSE_train	RMSE_test
1	39869.42	39968.16
2	39101.91	40065.58
3	40663.25	41350.35
4	37729.43	41060.23
5	39295.34	41700.98

Table 3: RMSE for best LightGBM models

3.3 XGBoost

For training XGBoost models, firstly we performed preprocessing on our data – the same we did while exploring data. It included imputation of missing values and coding of variables concerning quality or conditions rate.

For parameters tuning caret package mentioned before was used. By default, list of hyperpyrameters to perform matching included: `eta, max_depth, gamma, colsample_bytree, min_child_weight, subsample, nrounds`.

Over 200 models were trained using cross-validation and 'gbtree' option. In next step 40 best were picked. Evaluation was carried out with RMSE and MAE metrics taken into account. We were trying to choose models that not only performed well on test set, but also had low difference between metrics on test and train sets to cope with overfitting. Then chosen parameters were fitted to functions implemented beforehand and again models performance was diagnosed by measuring rmse and mae. Finally, selection of 5 best in our opinion was completed. Table 4 shows corresponding parameters values.

	eta	max_depth	gamma	min_child_weight	subsample	colsample_bytree	nrounds
1	0.04462844	3	3.580246	1	0.5339151	0.6035825	305
2	0.05525418	3	1.062662	10	0.9131714	0.6054747	305
3	0.19592307	2	8.526370	16	0.5217781	0.6626087	81
4	0.20411763	2	3.240579	1	0.6497563	0.6905933	405
5	0.08603091	2	3.549911	0	0.6369859	0.4167207	163

Table 4: Parameters of best XGBoost models

Error scores for best XGBoost models are contained in table 5. They reached pretty low values on a test set and significantly lower when predicted train values, which implies poor stability. Sadly, none of other models have reduced the gap between scores more satisfactorily. Although, XGBoost performance is acceptable and these top 5 models can compete with others.

	RMSE_train	RMSE_test
1	12005.92	28219.85
2	13583.38	30127.77
3	22388.00	34033.15
4	10474.84	29589.40
5	24614.30	36718.85

Table 5: RMSE for best XGBoost models

3.4 Random Forest

To build the random forest model as in previous models we started with data preprocessing. We removed columns containing missing data. We binarized data frame columns that were characters and had only two unique values inside each other. In the last step we performed one hot encoding on given dataframe. OHE was performed on character columns with more than two unique categories only.

To have some reference in further tuning the hyperparameters we started by building our model on the default parameters. We obtained RMSE values on it of 14025.22 for the training set and 64501.26 for the test set.

Hyperparameter tuning was performed using the random search method. For this purpose, we prepared a grid of the following hyperparameters:

- `num.trees` - number of trees in model, 400 to 1000 increasing in 20 steps
- `max.depth` - Maximal tree depth, 5 to 100 increasing in 5 steps
- `min.node.size` - Minimal node size, 1 to 10 increasing in 1 steps
- `splitrule` - Splitting rule, extratrees or vaiance

We trained 100 models from which we selected the top 5 models based on the average RMSE values on the training and test sets. They obtained the following RMSE values:

	RMSE_train	RMSE_test
1	49542.65	38349.32
2	43751.19	43696.35
3	43422.84	44024.12
4	30349.42	55754.17
5	30553.19	55836.22

3.5 Comparing models

Once we had selected the top 5 models for random forest, xgboost, lightgbm and catboost we decided to compare them to each other. To do this we created a point chart of the RMSE values for each of the 20 selected models.

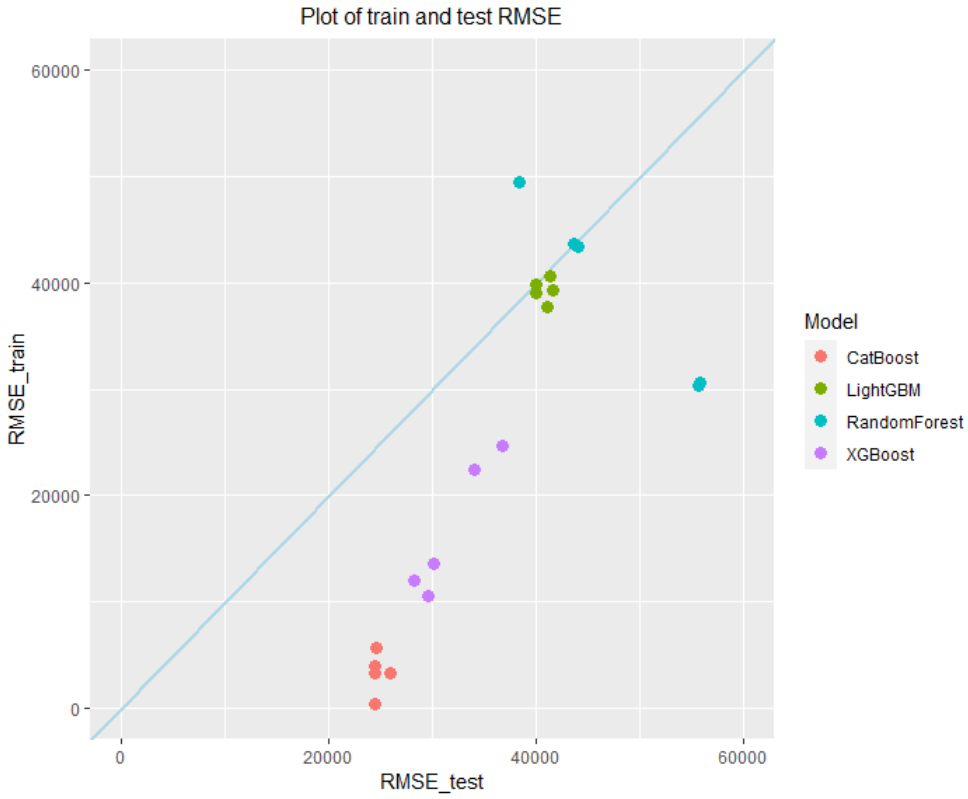

Figure 4: Comparing RMSE

The analysis of the models turned out to be fairly easy. It is straightforward to see that random forest turned out to be the worst model. For this model both the training and test RMSE values turned out to be very large compared to the other models. The second worst model is LightGBM, the only advantage of this model is that the training and test RMSE values are very close to each other, but they are still very large. Third turned out to be XGBoost. The best model on the other hand is CatBoost. Models built with CatBoost have a very small training RMSE, test RMSE is much larger, but it is still much smaller than for random forest, xgboost and lightgbm.

3.6 Selection and diagnosis of the best model

Of all the models, we chose the best one with a training RMSE of 397.50 and a test RMSE of 24430.67. This is the CatBoost model with the following parameters:

- depth - 6
- n_estimators - 1200
- learning_rate - 0,1
- l2_leaf_reg - 0,01
- colsample_bylevel - 0,9

We started the diagnostics of the selected model by calculating the MAE (Mean absolute error) value. It is 254.4542 for the training set and 14987.81 for the test set.

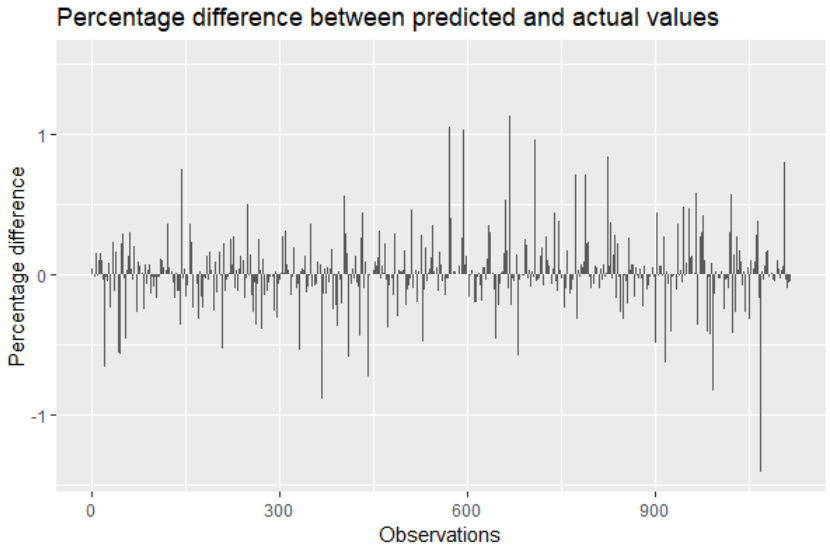

Figure 5: Percentage difference - train

As you can see, the difference between the true and predicted values for the training set is almost always less than 1

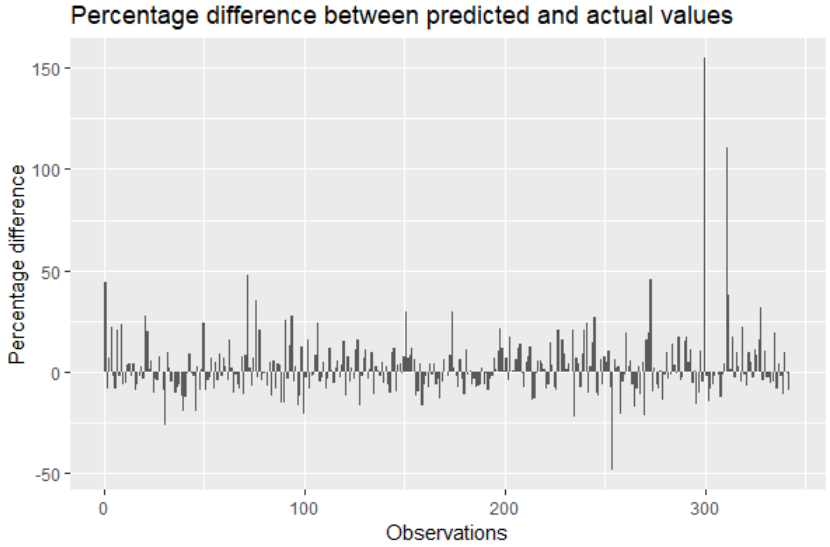

Figure 6: Percentage difference - test

For the test set, the difference is significantly larger, but except for a few outlier observations, for the vast majority the difference is only a few percent. We can also see from the graph that our model more often tends to overestimate rather than underestimate property values.

4 Explainable Artificial Intelligence

Catboost models are examples of black boxes in machine learning. In most cases it is crucial to understand what affects decisions that model makes. Comprehension of algorithm, fairness of process and further major choices depends on what is inside so called black box. To tackle problem of poor interpretability of our best catboost model, Explainable Artificial Intelligence procedures will be applied. Both local and global methods are equally important, so the analysis will concern very broad perspective on this issue with usage of `DALEX` package.

4.1 Permutational importance of variables

Firstly, permutational importance of variables was analysed. This method explains how important a variable is to the model, that is, how much a change in the value of a variable affects the final outcome of the model.

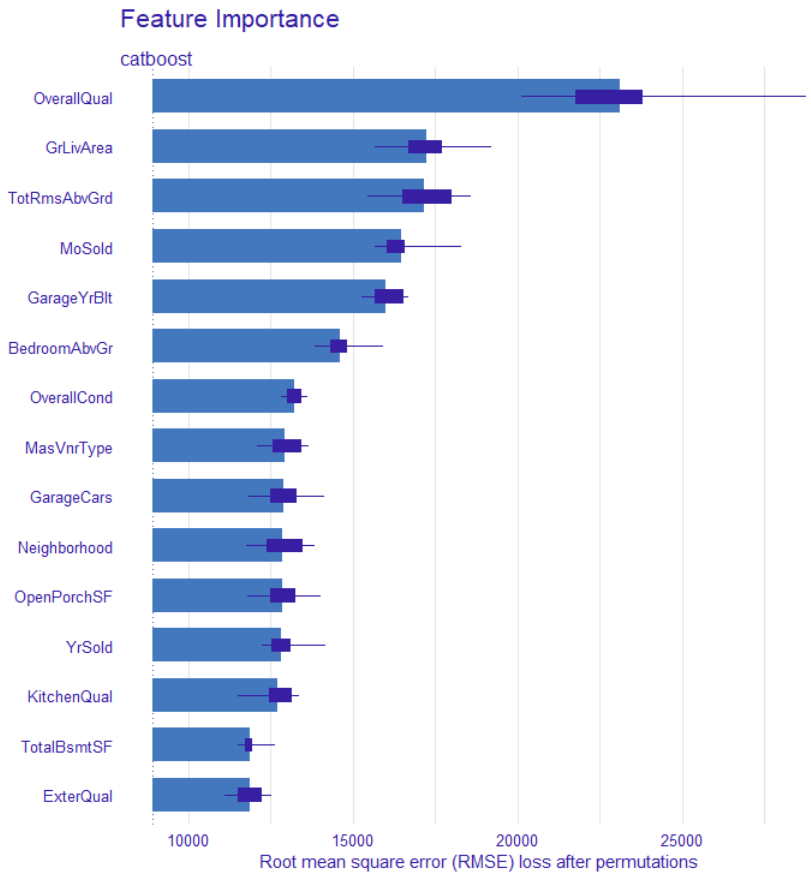

Figure 7: Permutational importance of variables

For the model, the column with biggest influence on the price of a property is its quality. It is very understandable that the better quality of the property, the price will be much higher than the value of the property with the same other parameters but of lower quality. The next most important variables are the size of the property and the number of rooms contained there. The larger the size of the property, the more rooms located in it usually. It seems logical that the larger the property, the higher its value. The constructed model confirms this observation.

4.2 Partial Dependence Profiles and Accumulated Local Dependence

Partial dependence profiles and accumulated local dependence show how property values change when the value of a particular variable changes. This also shows whether a particular variable has a linear or other effect on the model's predictions.

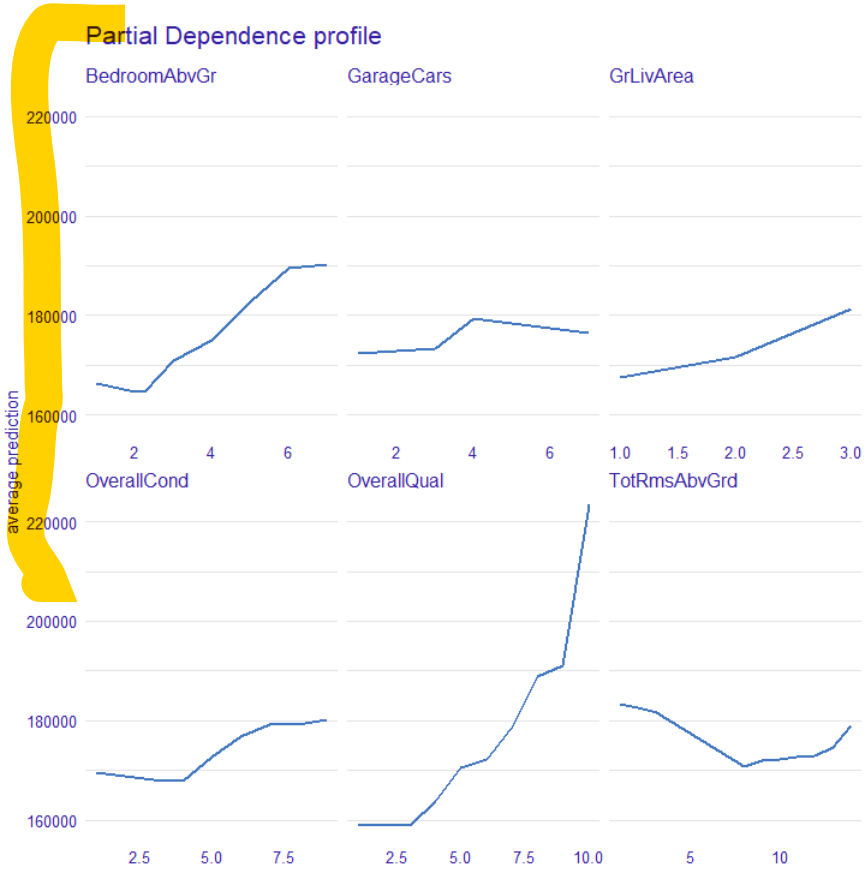

Figure 8: Partial Dependence Profiles

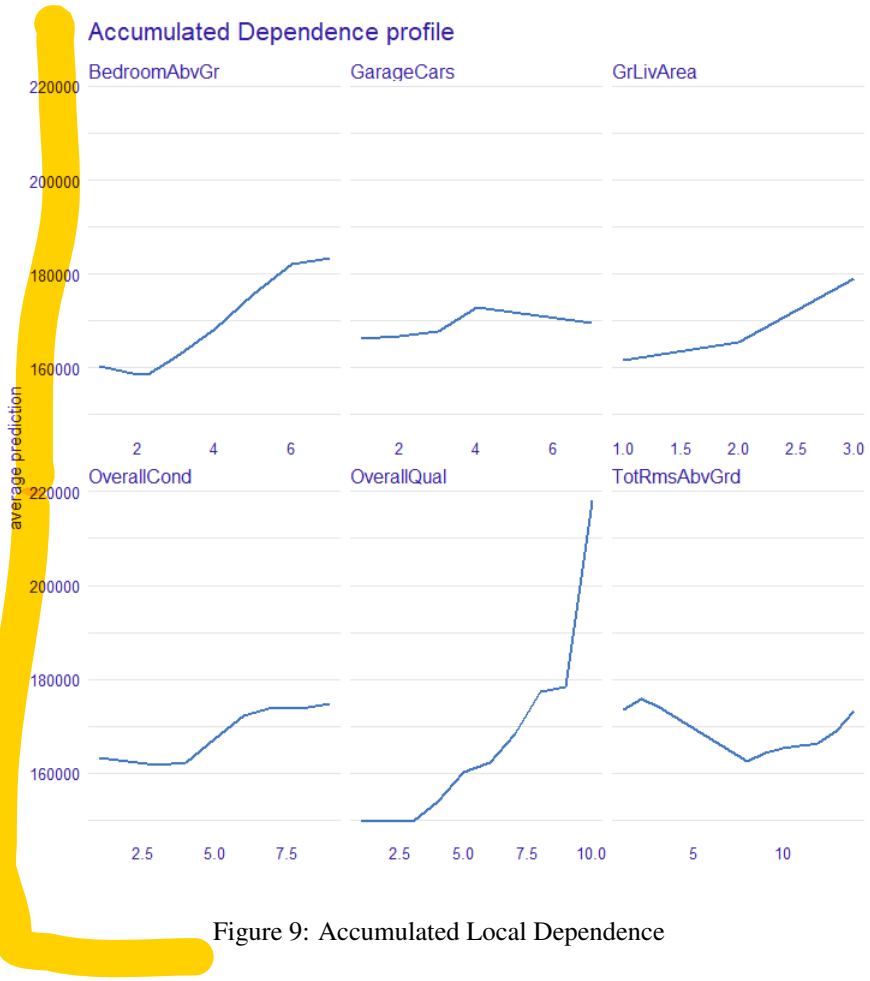

Figure 9: Accumulated Local Dependence

For the variables that had some of the highest feature importance values, the graphs and of Partial dependence profiles 4.2 and accumulated local dependence 4.2 look almost identical, so that clear conclusions can be drawn from them. Property values increase with increasing number of bedrooms. However, one-bedroom apartments are valued higher than two-bedroom apartments. Also, an interesting dependence is that increase in the price of an apartment slows down very significantly when the number of bedrooms exceeds 6. As for the number of garage spaces, it can be seen that the valuation increases until there are 4 such spaces and then it decreases. This seems logical as hardly anyone needs more than 4 garage spaces. The area of the apartment has an almost linear influence, the bigger it is, the higher the value of the property. With the overall condition of the apartment we see the biggest difference when going from level 4 to level 6. This is an interesting observation from a sales point of view. If we want to renovate an apartment in order to sell it, it is most profitable to buy an apartment with quality level 4 and then renovate it to level 6. With the overall quality of an apartment we can see a huge increase in valuation at the highest level. What is interesting, when it comes to the number of rooms we can see that with the increase in the number of rooms up to 8 the price decreases and next it increases. It is difficult to explain this behavior of the model and find a logical reason for it.

4.3 Break Down Profiles

Break Down Values of chosen records will help understand which variables have had the biggest influence on the final prediction for specific observations and to analyze how different values for one feature can lead to changing contribution from negative to positive and other way around.

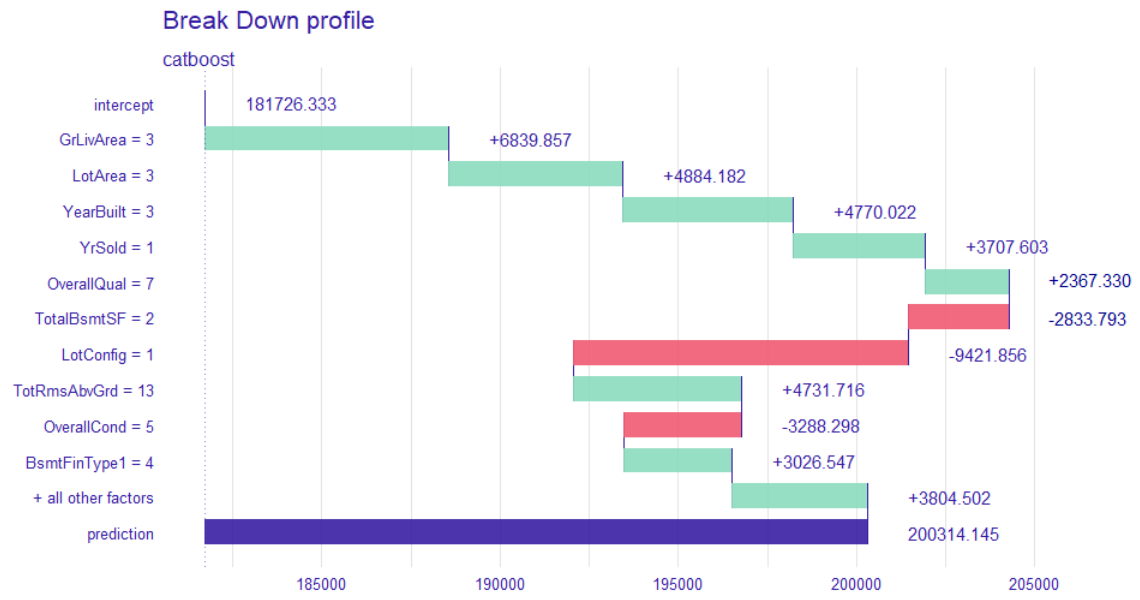

Figure 10: Break down values for observation with the lowest difference between actual and predicted price

Firstly, break down values profile for observation on plot 4.3 for which difference between actual and predicted price value was prepared. Prediction error is about 300$ in this case. The most positively influential variables are `GrLivArea` and `LotArea`. Both of them corresponds to living area above the ground. Value 3 on a scale of 4 indicates that this property is rather spacious, so the positive contribution is clear. Other significant columns increasing the price were related to build year - build not a long time ago and overall quality of the house with value 7 meaning very good standard. On the other hand, `LotConfig` remarkably decreased predicted price. This variable describes position of lot against other lots. Value 1 indicates that this property is surrounded by other properties.

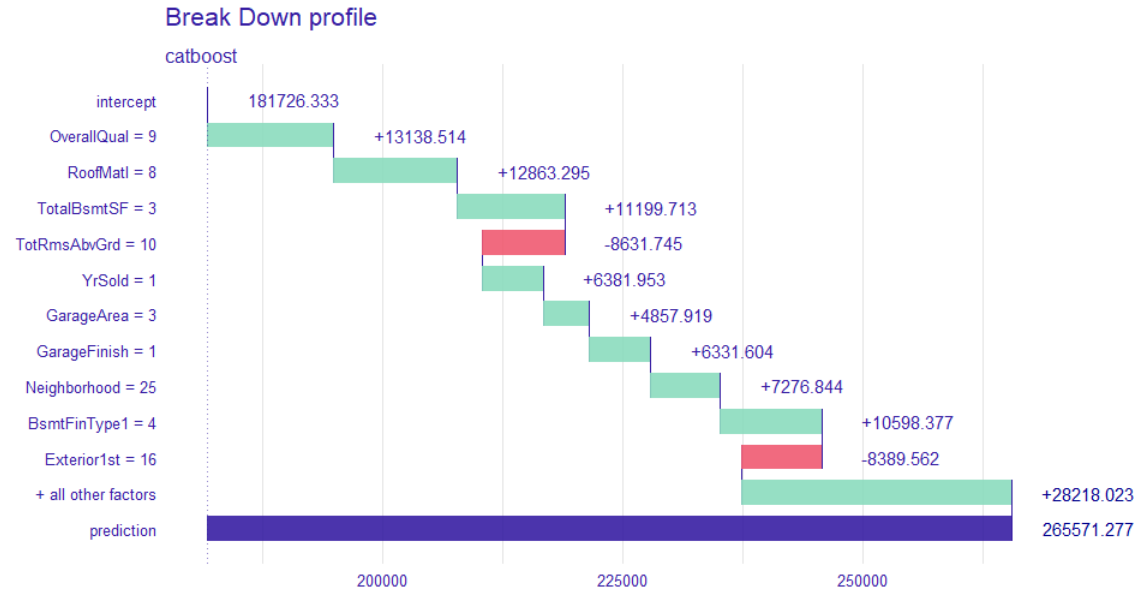

Figure 11: Break down values for observation with the biggest difference between actual and predicted price

Break down values for property with the biggest error between actual and predicted price is presented on plot 4.3. Although majority of features were adding to final prediction, this property is still underestimated and error amounts to 110,000$. Closer look at data showed that this property is the most expensive one in our set. Variables connected to basement and garage are powerful positive contributors. It is very common for houses in this set that presence of this two places in a property is a huge price booster. Property has an excellent quality of finish on level 9. Also, district of city Ames represented by variable `Neighborhood` is a positive factor in this case. Surprisingly, number of rooms in house equals 11, which is a huge amount in comparison. Yet, this value decreases final price prediction. This behaviour confirms our observations from partial dependence profiles. Features associated with property shows why this house real price is on such a high level. Our model predicted price incorrectly, because this observation can be treated as an outlier, thus algorithm have not learn to cope with such outstanding occurrence.

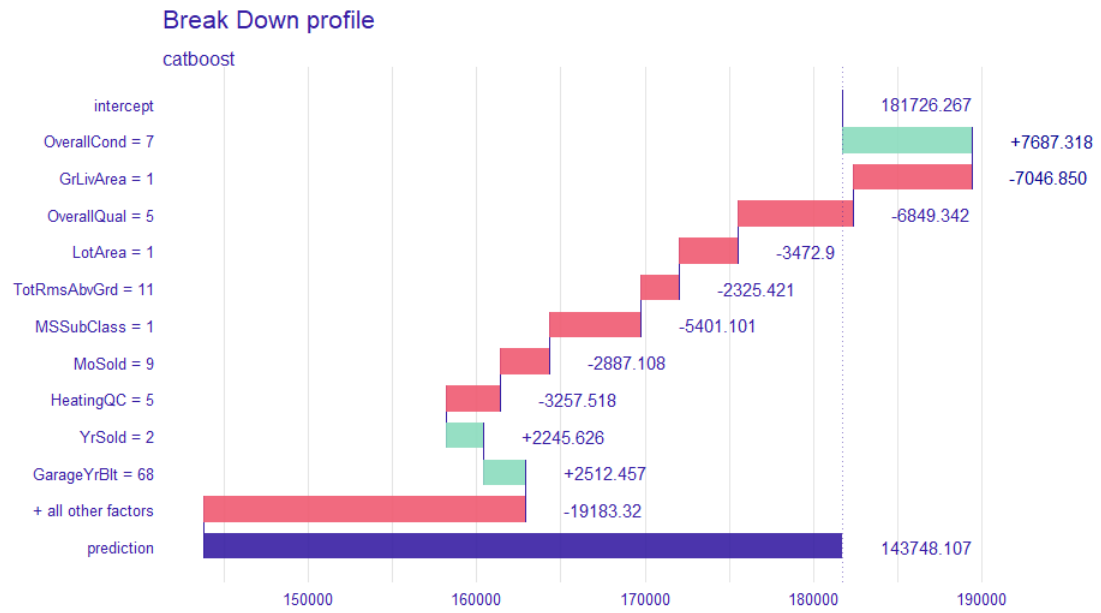

Figure 12: Break down values for observation showing contrasting contributions

Analysing break down value algorithm for many properties shows that the most influential variables are similar for different observation. They coincide with the most powerful variables regarding feature importance, although it can be observed that changing values cause significant prediction changes. On the plot 4.3 one can see variables contribution to final prediction in observation, that shows some contrasting variables values. In this case, previously most price increasing features - above grade living area in square foot, overall quality of finish and lot size, have lower values. Value 1 for area relate columns indicates little room and quality on level 5 means that house have average standard of finish. As a result, this features impact negatively on final price. On the contrary, overall condition level is higher and equals to 7 meaning good conditions. Thereby contribution is clearly positive. In conclusion, our model values positively mainly properties with lot of space and of very good quality. Otherwise, decision to decrease prediction will be made. In addition, algorithm highlights appearance of garage and basement.

5 Discussion

References

Víctor Hugo Masías, Mauricio Valle, Fernando Crespo, Ricardo Crespo, Augusto Vargas Schüler, and Sigifredo Laengle. Property valuation using machine learning algorithms: A study in a metropolitan-area of chile. 01 2016.

Cankun Wei, Meichen Fu, Li Wang, Hanbing Yang, Feng Tang, and Yuqing Xiong. The research development of hedonic price model-based real estate appraisal in the era of big data. *Land*, 11(3):334, February 2022. doi:10.3390/land11030334. URL https://doi.org/10.3390/land11030334.

Geoffrey K. Turnbull and Arno J. van der Vlist. After the boom: Transitory and legacy effects of foreclosures. *The Journal of Real Estate Finance and Economics*, February 2022. doi:10.1007/s11146-021-09882-w. URL https://doi.org/10.1007/s11146-021-09882-w.

Dieudonné Tchuente and Serge Nyawa. Real estate price estimation in french cities using geocoding and machine learning. *Annals of Operations Research*, February 2021. doi:10.1007/s10479-021-03932-5. URL https://doi.org/10.1007/s10479-021-03932-5.

Alex van de Minne, Marc Francke, and David Geltner. Forecasting US commercial property price indexes using dynamic factor models. *Journal of Real Estate Research*, 44(1):29–55, December 2021. doi:10.1080/08965803.2020.1840802. URL https://doi.org/10.1080/08965803.2020.1840802.

L. Breiman. Random forests. 45, pages 5–32. Springer Nature, 2001. doi:10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

Tianqi Chen and Carlos Guestrin. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, aug 2016. doi:10.1145/2939672.2939785. URL https://doi.org/10.1145%2F2939672.2939785.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features, 2017. URL https://arxiv.org/abs/1706.09516.

Max Kuhn. Building predictive models in r using the caret package. *Journal of Statistical Software, Articles*, 28(5): 1–26, 2008. ISSN 1548-7660. doi:10.18637/jss.v028.i05. URL https://www.jstatsoft.org/v028/i05.

Przemyslaw Biecek. Dalex: Explainers for complex predictive models in r. *Journal of Machine Learning Research*, 19 (84):1–5, 2018. URL https://jmlr.org/papers/v19/18-416.html.