Price Prediction of Motorcycles using an ensembled Machine Learning Model

Tabea Hacheney

tabea.hacheney@tu-dortmund.de

Due Date: 31st of July 2024

Contents

1. Motivation

To make reasonable buying decision of objects, let those be clothing, a new CPU or even a motorcycle, one has to have an approriate price in mind. This might be more or less easy for common items like food and clothing, but a quiet challenging task for more complex and expensive things, like houses, rare **PoKéMoN** cards or motorcycles. A potential buyer is almost obligated to invest multiple hours to understand how the price of an item comes to be and to check if it is approriate in comparison to the market.
This is a very tidious task. By using Machine Learning algorithms, using up to date market prices and appropriate attributes, which influence the price, one can get an appropriate price with minimal time and effort (*at least for the end user*). When the author of this report was trying to sell her last motorcycle on the internet, she was met with the problem: "*What is the highest possible profit I could get out of this while not seeming absurd?*". Which brings us to the content of the following report, in which an ensembled model of multiple regressor model will be used to estimate an appropriate market price according to the age, mileage, brand, model and the with that connected bike specifications.

2. Dataset

To train a machine learning model and achieve reasonable good results, a good data base is needed. For getting accurate market price predictions for motorcycles, one needs up to date market data of motorcycles. This is given by the dataset *"USA Comprehensive Motorcycles Dataset 9k+"*[1] and *"Motorcycle Specifications Dataset"*[2], which both can be found on **kaggle.com** and underly a *CC0:Public Domain* License.
The first dataset provides market prices, model, mileage and year of manifacture for the brands *BMW*, *KTM*, *Royal Enfield*, *Suzuki*, *Yamaha* and *Ducati* up to 2023. Although only six different brands are provided, these already cover most of the current motorcycle market. The second dataset provides additional information for the single motorcycle models like displacement, power or the number of cylinders.

2.1. Preprocessing

A common first step for machine learning tasks is the preprocessing of the dataset. According to the quality of the dataset, this can be more or less tidious. As for the final dataset used in this report, this step took longer than hoped for. As the data for all brands were provided seperately, as well as the dataset for the model specifications one of the major task was to merge the bike specification columns (displacement, ...) to the datasets containing the prices. An excerpt of the used datesets is shown in Figure 1. An enlarged view on the datasets is shown in the Appendix A.

mileage	price	Bike	Types and Used Time	Year
500 miles	$19,994	r18	2022 BMW Cruiser	2022
16,479 miles	$20,995	k1600b	2019 BMW Touring	2019
123,456 miles	$21,000	r602	1966 BMW Classic / Vintage	1966
7,709 miles	$20,000	k1600b	2019 BMW Cruiser	2019
20,311 miles	$19,595	k1600b	2018 BMW Touring	2018
...	...	...	...	...
2 miles	$14,770	ce04	New 2023 BMW Scooter	2023
2 miles	$26,005	r1250gs	New 2023 BMW Dual Sport	2023
5 miles	$15,365	rninetscrambler	New 2023 BMW Standard	2023
2 miles	$28,820	k1600b	New 2023 BMW Touring	2023
1 miles	$25,580	r1250gs	New 2023 BMW Dual Sport	2023

Brand	Bike	Year	Category	Rating	Displacement (ccm)	Power (hp)	Torque (Nm)	Engine cylinder	Engine stroke
bmw	450 sports enduro	2008	Enduro / offroad	3.5	449.0	49.6	48.0	Single cylinder	four-stroke
bmw	blechmann r18	2020	Prototype / concept model	NaN	1800.0	90.0	158.0	Two cylinder boxer	four-stroke
bmw	c 400 gt	2019	Scooter	3.5	350.0	34.0	35.0	Single cylinder	four-stroke
bmw	c 400 gt	2020	Scooter	NaN	350.0	34.0	35.0	Single cylinder	four-stroke
bmw	c 400 gt	2022	Scooter	NaN	350.0	34.0	35.0	Single cylinder	four-stroke

Figure 1: Excerpt of one of the datasets containing the price and selling specifications (left) and the dataset containing the motorcycle specifications (right).

Before merging the datasets, the columns have to be properly cleaned. For this purpose, a descriptional column of the datasets containing the prices was discarded, as it does not give additional information in a uniform way. Some entries had information about the bike condition or the limitation of the model in that column. Secondly, the year of manifacture was extracted from the *Types and Used Time* column and all entries with no information on the price and mileage were dropped. The *Bike* column of the price datasets and the *Bike* column of the specifications dataset were brought to the same formatting and style, such that a merge between those datasets is possible.
Finally, the datasets are merged, based on the bike model and the year of manifacture. For those cases, where there is no matching bike model for the exact year of manifacture in the specifications dataset, the entries are merged with another year of manifacture of that exact model. Some bike models of the specifications dataset were missing some column entries like *Category* for some years of manifacture. These entries were filled with

matching bike models, but different years of manifacture. It does happen, that some specifications like power or displacement changes over the different years of manifacture slightly, but the used method should yield a very good approximation of the real column entries. At this point, all entries for which the column *Category* is empty, are dropped. Next, NaN entries are filled with the mean value of the corresponding bike category, the *mileage* and *price* column data types are fixed to be integers instead of strings and calculated into kilometres and euros instead of miles and dollars.

All brand subsets are concatenated to form one big dataset and the final columns used for further analysis are chosen as *Mileage [km], Price [€], Bike, Brand, Category, Displacement [ccm], Power [hp], Torque [Nm], Condition and Age [a].* The condition column contains boolean values for Used (`True`) and New (`False`) based on the mileage. The age is the relative used age of the motorcycle (2023 − Year).

2.2. Exploratory Data Analysis

Before throwing the dataset into some machine learning algorithms and hoping for good results, a logical approach is to look at the data distributions, correlations of the attributes with each other and the price and look for useful data scaling methods. As some of the columns contain categorical data (like *Bike*), they need to be transformed into numerical data, to make use of them in a plot. For the *Bike* and *Category* column, a frequency mapping was chosen, meaning that the string values were matched with the frequency of that relative string. This is a common approach for categorical columns, with a broad varity of entries. For the *Brand* column, a dummy column is created. This means, that for each unique value of the respective column, a new column with boolean values is created.

To get a general understanding of the data distributions, a scatter matrix containing all columns (except the dummy columns) is presented in Figure 2. It shows, that there are in general many newer motorcycles with a low age amount and little amount of mileage. The displacement, power and torque are more or less normal distributed, with significant peaks in the mean value, which is due to the filling of `NaN` values with the mean. Many

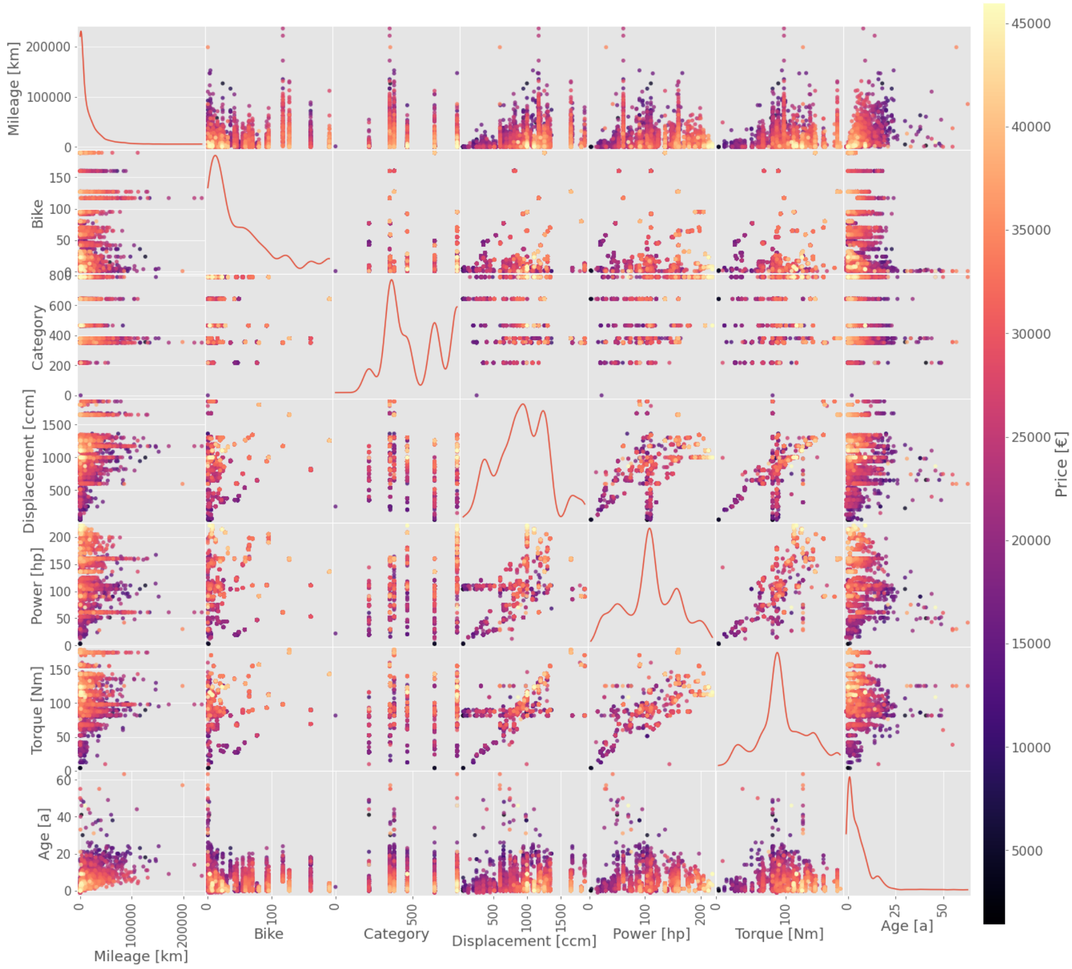


Figure 2: Scatter matrix of all dataset columns (except dummy columns).

scatter plox show already significant correlations, like the *Power [hp]* - *Torque [Nm]* (which is very much expected as these two metrics are causally linked). The Figure 3 also shows clear dependencies of the price on the used attributes, making them useful for regression problems. Motorcycles with a low *age* and *mileage* and a high amount of *power* should yield the highest prices.

When handling big datasets, a high amount of attributes while also having a very high amount of data entries can lead to long training times, overtraining and long waiting times for hyperparameter optimisation. Hence, one wants to look at the correlation of different used attributes. The correlation matrix for the used numerical variables is shown in Figure 4.

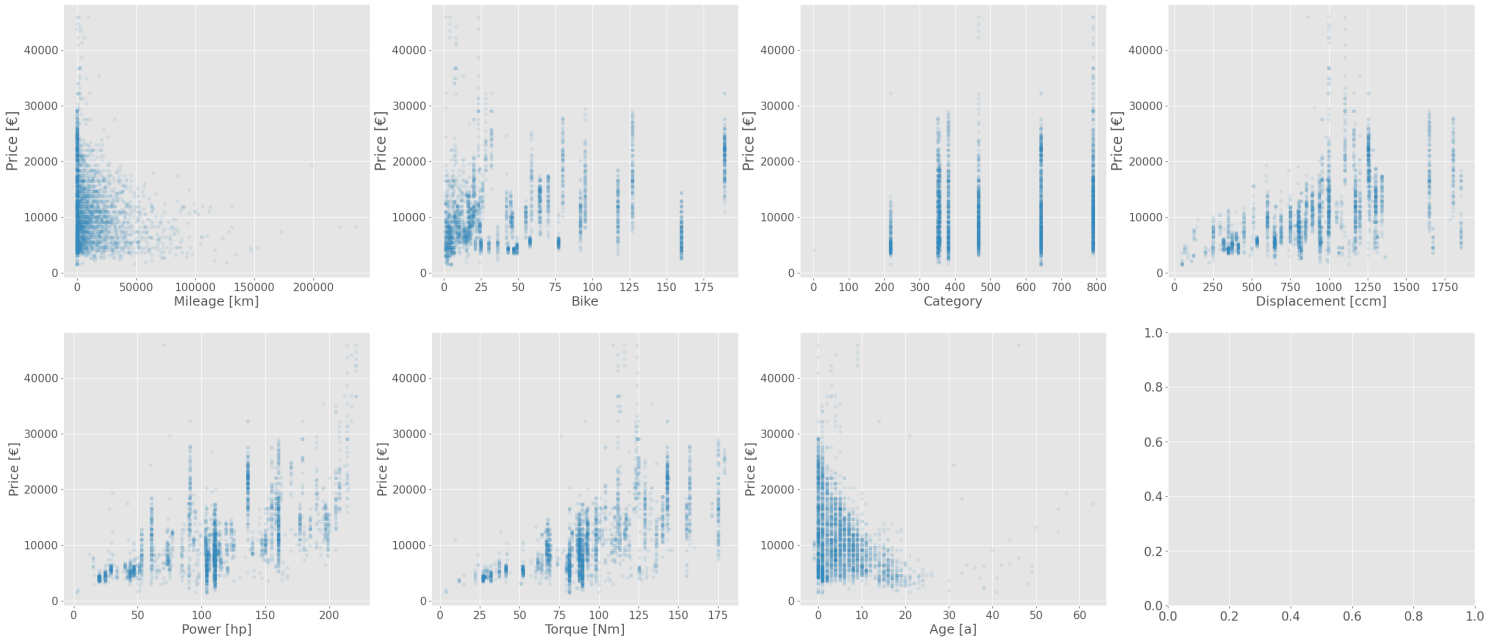


Figure 3: Scatter plots of the training attributes with the target value (*Price [€]*).

Some attributes like the torque, power and displacement show high correlation, which is expected to to their mechanical origin. Also, the mileage and age, as well as the brand *Royal Enfield* and the engine specifics are highly correlated. However, no variables will be discarded at this step, due to the already low number of attributes and the later performed feature selection.

Another useful method is to look at the behaviour of the data after different scaling are applied. As for the provided attributes, the different scaling methods are observed for the *Mileage* and *Age* attribute. As they have the most fluent distributions and nice correlations with the price. The scatter plots of these two attributes for *Standard*, *Robust*, *Gaussian*, *Min-Max*, *Max-Abs* and *Uniform* scaling are shown in Figure 5. The scalers have been used from the **sklearn.preprocessing**[3] library. The online documentations offers more insides on the type of scaling. In the training of the single models, the standard, robust and normalised scaling are compared with no scaling in more detail.

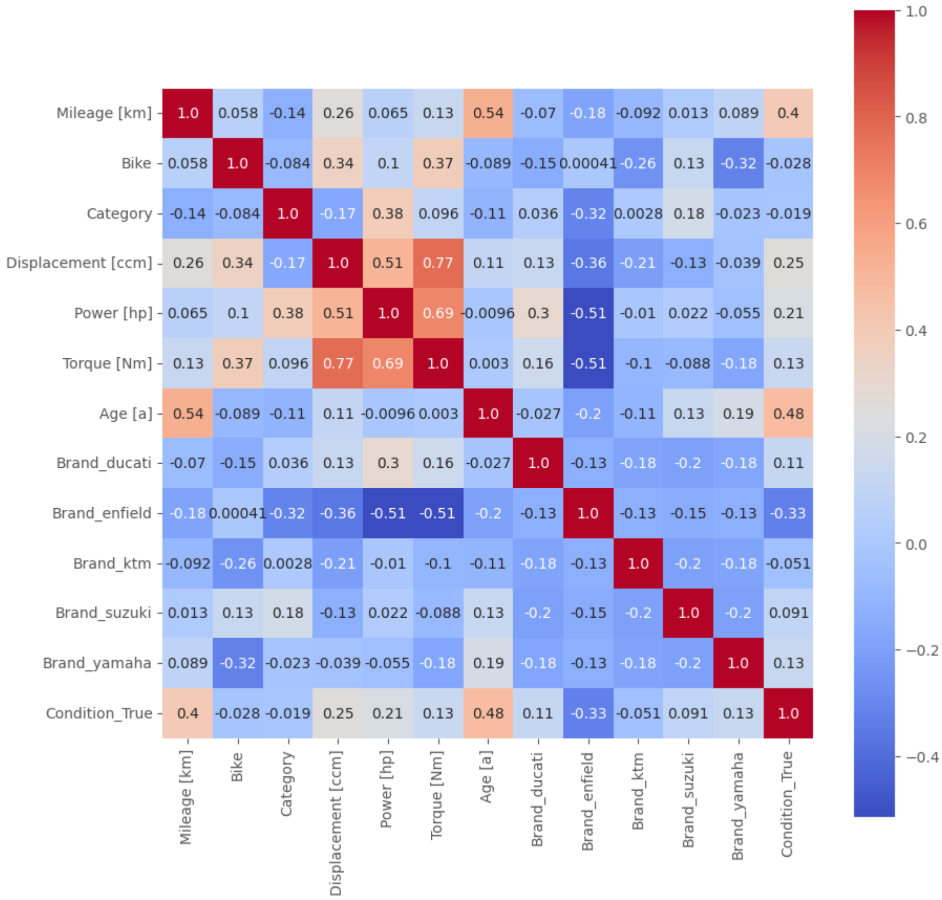
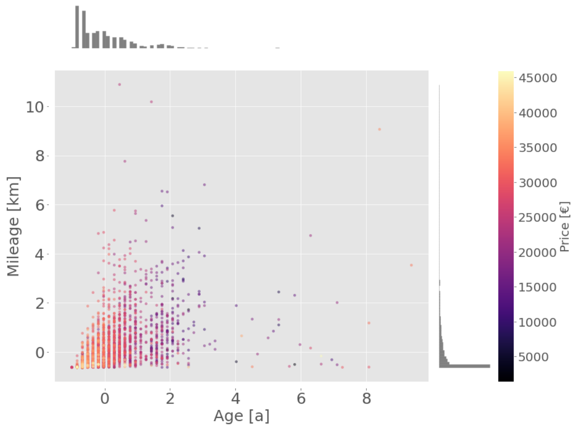
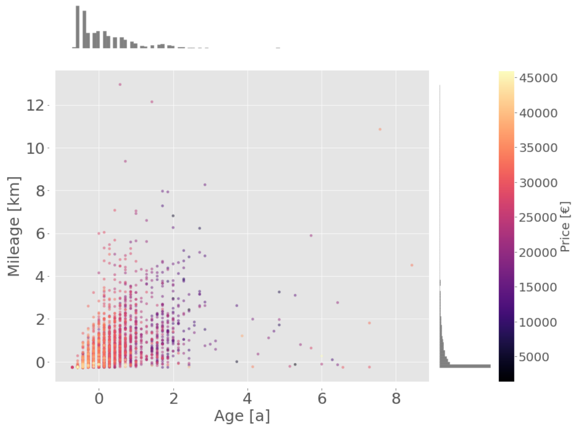


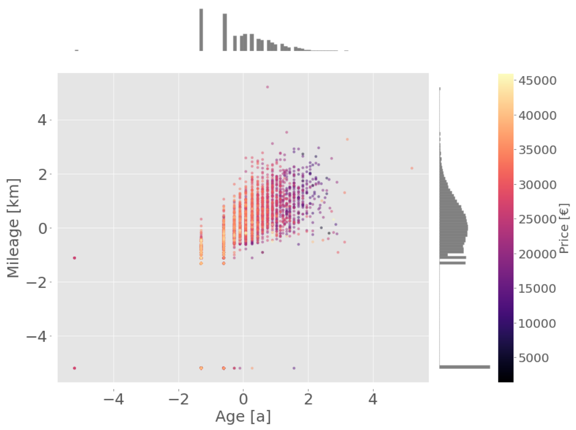
Figure 4: Correlation matrix of the provided dataset attributes.



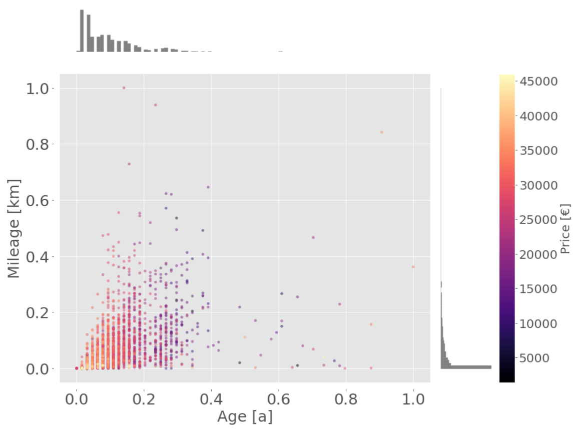
(a) Standard Scaling.



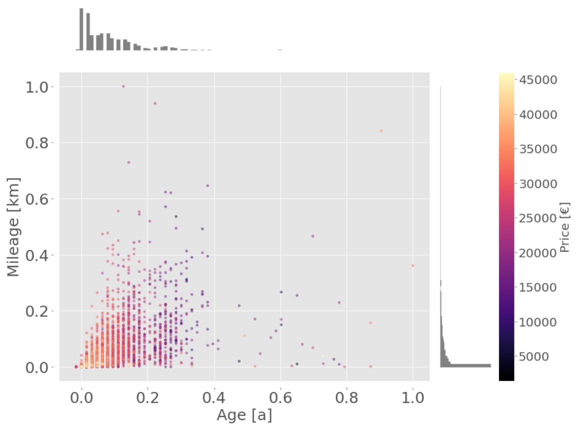
(b) Robust Scaling.



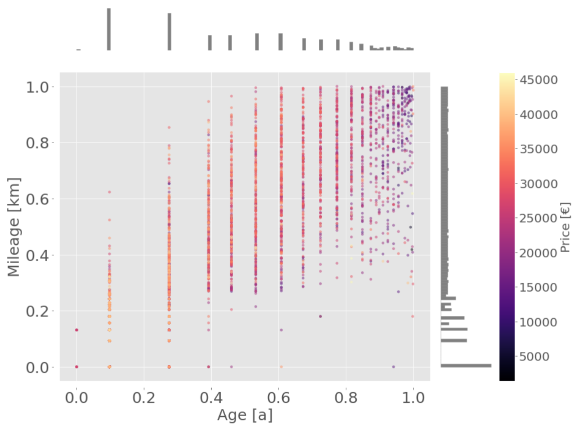
(c) Gaussian Scaling.



(d) Min-Max Scaling.



(e) Max-Abs Scaling.



(f) Uniform Scaling.

Figure 5: Scatter plot of *Mileage* and *Age* with different scaling methods.

3. Machine Learning Models

As price prediction is a common regression task, multiple regressors are trained seperately and the best performing models are ensembled using a VotingRegressor.

3.1. Regression Models

For the single regressor models, the models *XGBoost* (one tree based, one based on linear models), *CatBoost*, *AdaBoost* and *ExtraTrees* were trained and compared using the three scaling methods discussed in subsection 2.2 and with no scaling.
The target variable *Price* is scaled logarithmic. Training based on a Min-Max scaling of the price yielded worse results on average. The training and testing split was done by using the train test split function of the sklearn.modelselection library and set as $30\,\% - 70\,\%$.
For the training of the individual models, a Pipeline, involving a *preprocessor*, a *feature selection* and the respective *regressor*. The preprocessor consists of a numeric and categorical transformer. The numeric transformer uses a SimpleImputer to fill missing entries with the mean and also scales entries according to the chosen scaling method. The categorical transformer suses a SimpleImputer to fill `NaNs` with the most frequent entry and handles categorical values with a OneHotEncoder which creates dummy columns.
The feature selection is done with an Extra Trees Regressor, which evaluates the most important features, which are then used for the training. The hyperparameters are optimised by using a Randomised Search with 10-fold Cross Validation. The best parameters for each scaling for each model can be found in the Appendix A.
As evaluation metrics, a 10-fold Cross Validation of the Root Mean Squared Error (RMSE) is performed and the Mean Squared Error (MSE), Mean Absolute Error (MAE), R^2 are evaluated for the test sub set and the Training Time (TT) is extracted. The results are shown in **??**. The most important parameters chosen by the ExtraTrees Regressor are visually represented in **??**.

3.2. Ensembled Model

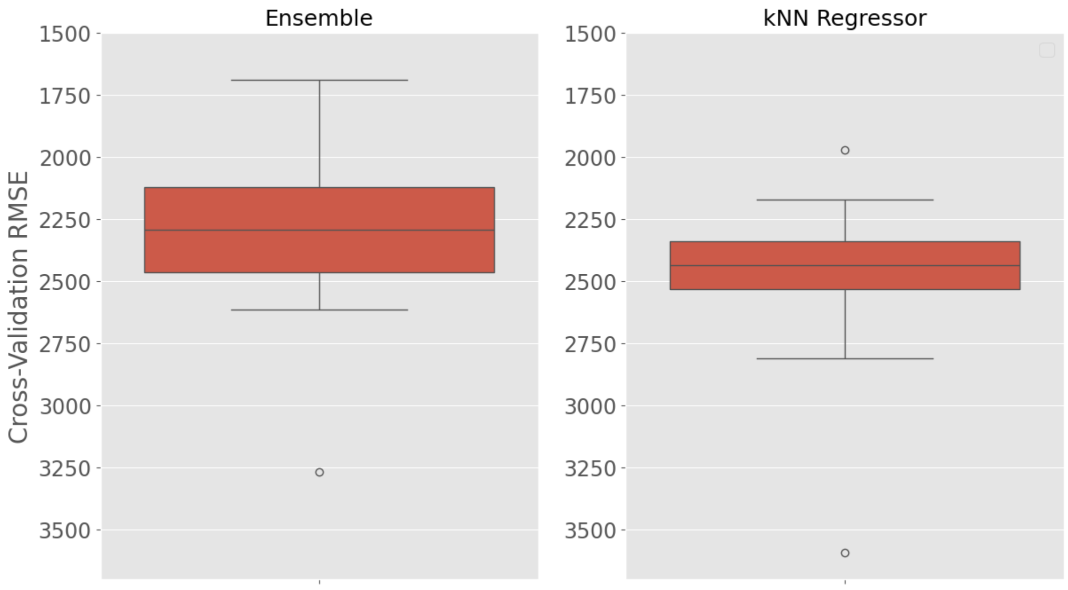


Figure 6: .

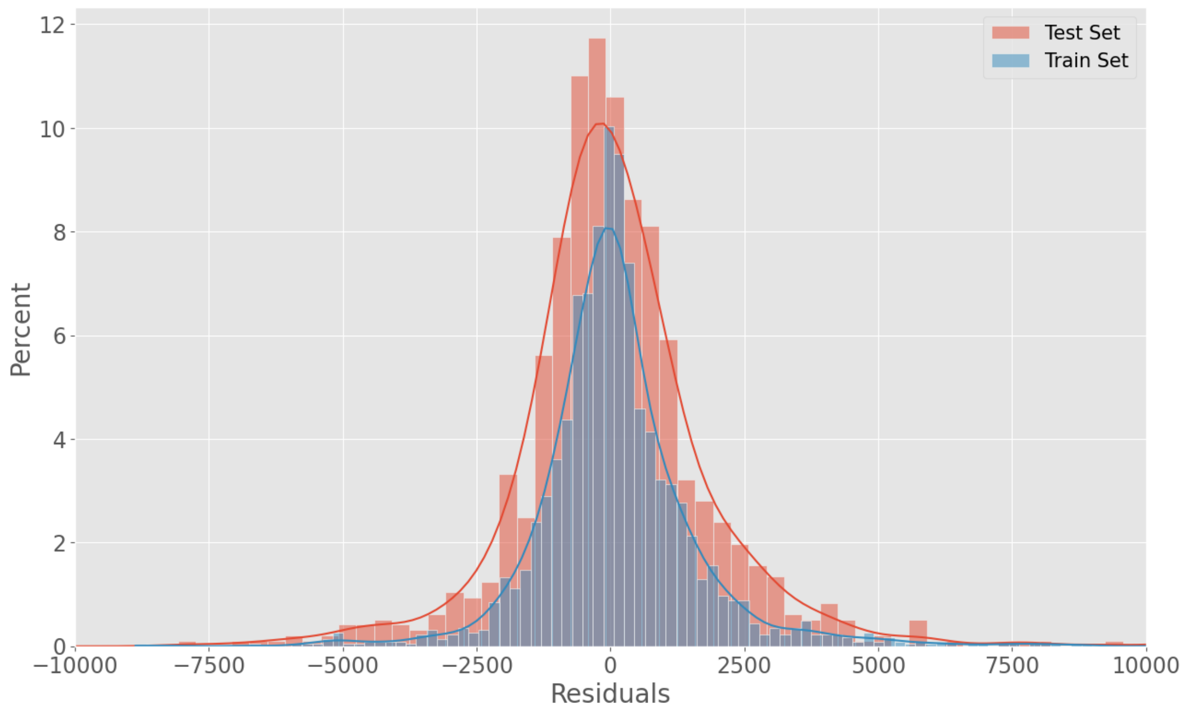


Figure 7: .

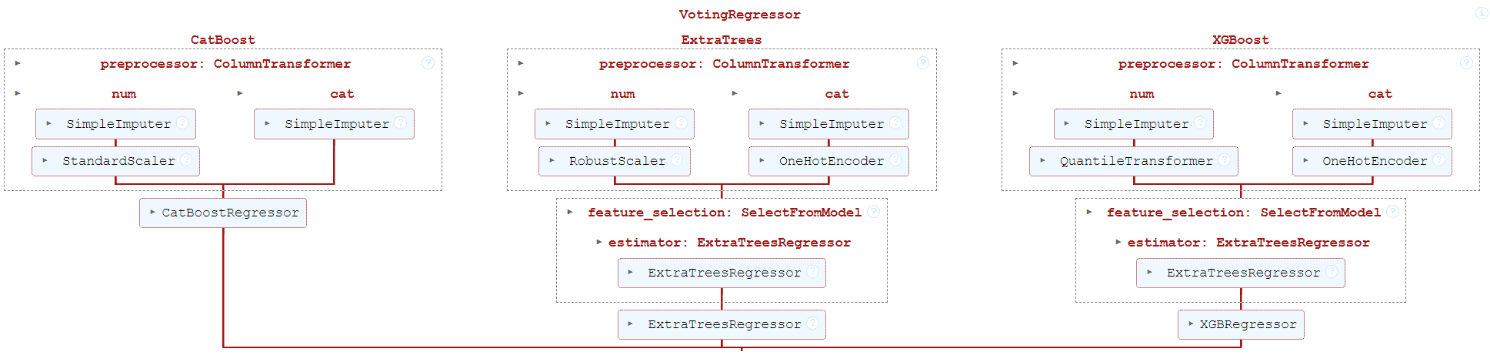


Figure 8: .

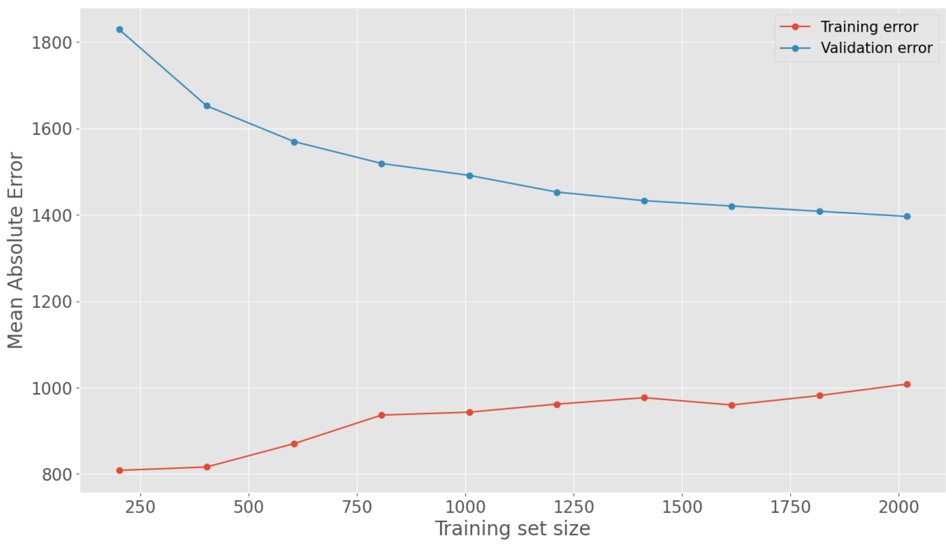


Figure 9: .

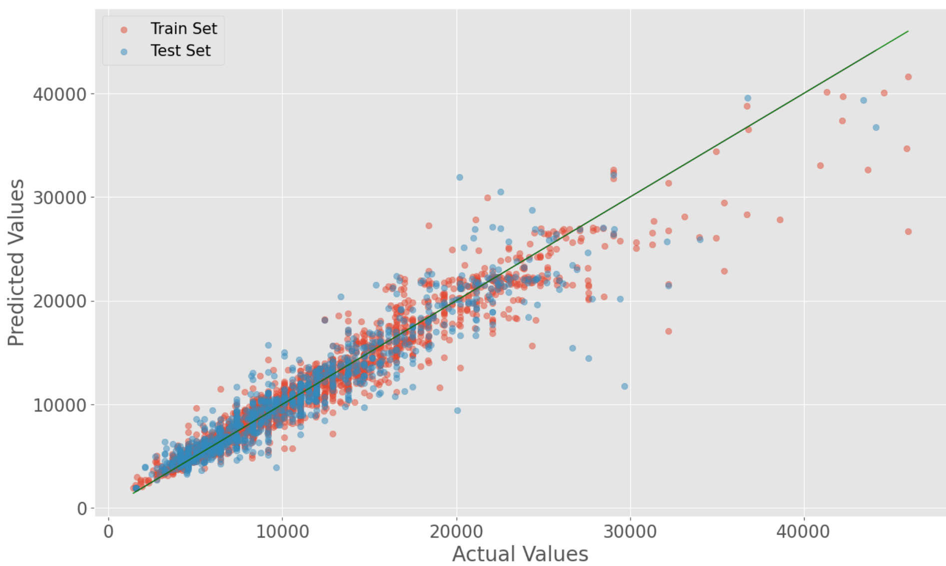


Figure 10: .

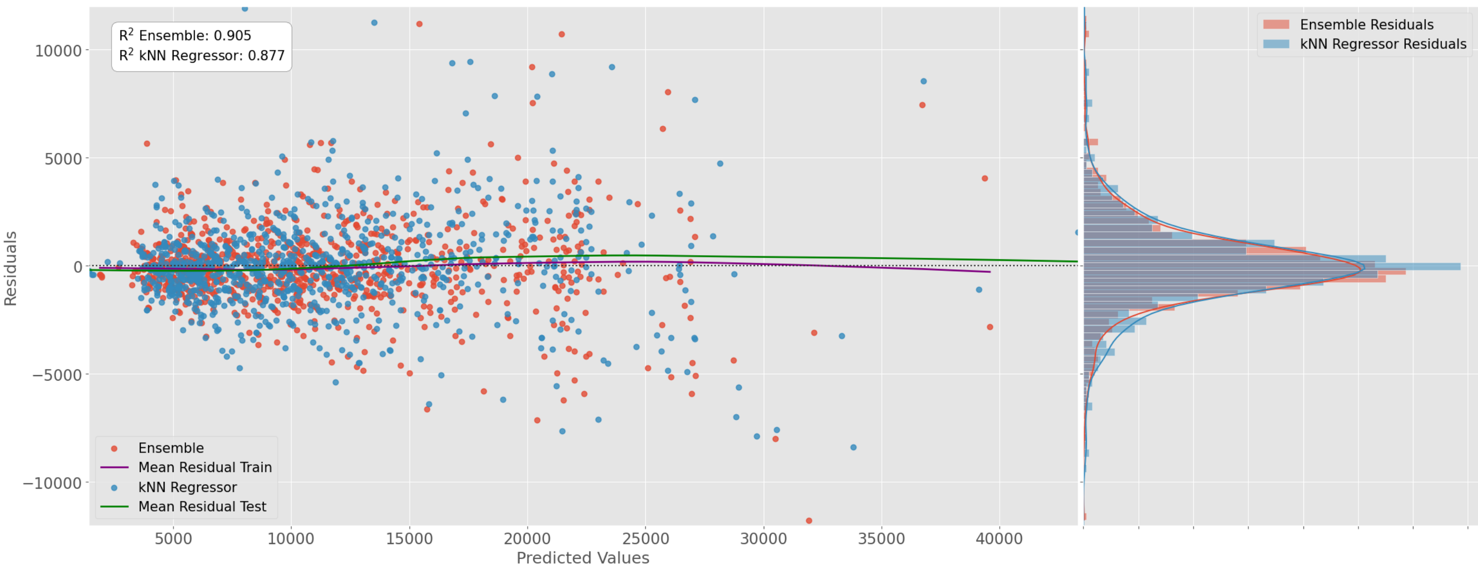


Figure 11: .

Brand	Bike	Category	Power [hp]	Displacement [ccm]	Torque [Nm]	Mileage [km]	Age [a]	True Prices [€]	Predicted Prices [€]	Difference [€]
suzuki	hayabusa	Sport	197	1,340	155	9,442	8	11,499.08 €	11,500.48 €	1.40 €
ktm	rc390	Sport	44	373	37	27	1	5,335.08 €	5,338.08 €	3.00 €
suzuki	boulevardc50t	Touring	111	819	90	51,512	16	5,510.80 €	5,507.67 €	3.13 €
nfield	continentalgt	Allround	29	535	41	8	1	5,518.16 €	5,514.88 €	3.28 €
ktm	390adventure	Enduro / offroad	44	373	37	3	1	6,255.08 €	6,264.98 €	9.90 €
suzuki	gsx-r750	Sport	106	749	90	2	0	11,024.36 €	11,047.52 €	23.16 €
maha	yz85	Enduro / offroad	104	85	82	48	3	3,675.40 €	3,698.80 €	23.40 €
ducati	monsterplus	Naked bike	111	937	93	9,012	17	6,900.00 €	6,924.77 €	24.77 €
ducati	848evo	Sport	138	849	98	11,507	12	9,384.00 €	9,358.37 €	25.63 €
suzuki	gsx250r	Sport	24	248	22	33,226	5	4,508.00 €	4,482.18 €	25.82 €

Figure 12: .

Brand	Bike	Category	Power [hp]	Displacement [ccm]	Torque [Nm]	Mileage [km]	Age [a]	True Prices [€]	Predicted Prices [€]	Difference [€]
ducati	mh900e	Sport	75	904	76	251	21	29,670.00 €	11,618.95 €	18,051.05 €
bmw	r18	Custom / cruiser	91	1,802	157	31,382	2	27,595.40 €	14,430.90 €	13,164.50 €
ducati	panigalev4	Sport	215	1,103	123	483	1	20,171.00 €	31,884.39 €	11,713.39 €
bmw	rninet	Naked bike	108	1,170	119	8	0	26,680.00 €	15,362.04 €	11,317.96 €
bmw	r1250gs	Enduro / offroad	136	1,254	143	1,287	1	32,200.00 €	21,486.68 €	10,713.32 €
ducati	monsterplus	Naked bike	111	937	93	349	4	20,056.00 €	9,414.69 €	10,641.31 €
bmw	s1000rr	Sport	190	999	112	161	0	29,440.00 €	20,163.23 €	9,276.77 €
bmw	m1000rr	Sport	205	999	113	575	1	34,040.00 €	26,005.01 €	8,034.99 €
ducati	panigalev4	Sport	214	1,103	124	8	0	22,535.40 €	30,563.01 €	8,027.61 €
bmw	r18	Custom / cruiser	91	1,802	157	8	1	27,820.80 €	20,233.53 €	7,587.27 €

Figure 13: .

4. Alternative Method : kNN Regressor

4.1. Comparison to Ensembled Regression Model

References

[1] *USA Comprehensive Motorcycles Dataset 9k+*. kaggle. URL: `https://www.kaggle.com/datasets/joyshil0599/comprehensive-motorcycles-dataset` (visited on 30/07/2024).

[2] *Motorcycle Specifications Dataset*. kaggle. URL: `https://www.kaggle.com/datasets/emmanuelfwerr/motorcycle-technical-specifications-19702022` (visited on 30/07/2024).

[3] 'scikit learn : 6.3 Preprocessing'. Version 1.5.1. In: (2024). URL: `https://scikit-learn.org/stable/modules/preprocessing.html`.

A. Appendix