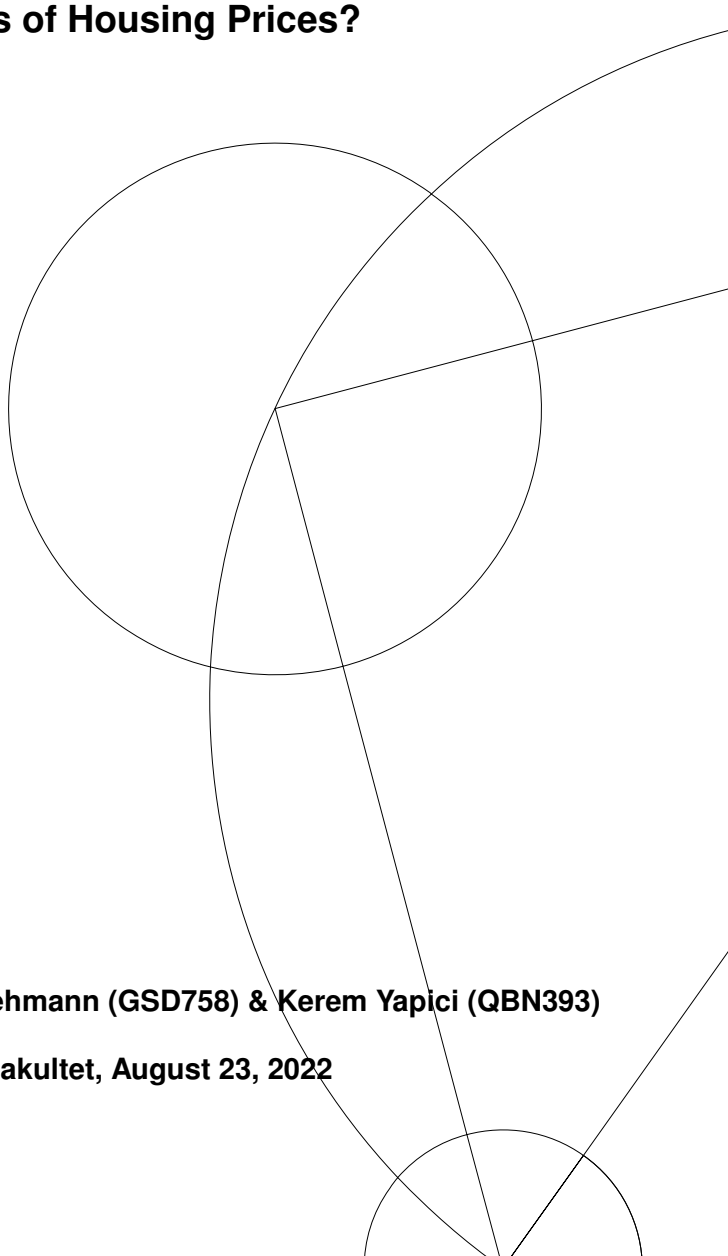
KØBENHAVNS UNIVERSITET

Predicting Housing Prices in Copenhagen using Machine Learning

Can Geospatial Data Improve Predictions of Housing Prices?



Julius Løve Fischer(NBM205), Hans Christian Jul Lehmann (GSD758) & Kerem Yapici (QBN393)

Institut for økonomi - det samfundsvidenskabelige fakultet, August 23, 2022

Abstract

In our desire to uncover whether geospatial data can help predict housing prices in Copenhagen, we determine which of our considered machine learning algorithms are most fit for predicting house prices in Copenhagen. The linear regression algorithms, OLS and LASSO, and the non-linear Random Forest Regressor are applied to examine about ∼33.000 housing transactions over the period 2018-2022 in Copenhagen. In the appraisal of the best prediction performance, we find that the Random Forest Regressor outperforms the linear OLS and LASSO algorithms significantly based on the performance metric root mean squared error (RMSE). We conclude that including geospatial attributes improves the prediction performance of all the algorithms employed in alignment with the literature reviewed.

Contributions:

Joint:
Section: 5.1

NBM205:
Sections: 2.4.2, 2.6, 3, 3.3, 4, 4.1 & 6

GSD758:
Sections: 1, 2, 2.1, 2.2, 2.2.1, 2.2.2, 3.1 & 5.2

QBN393:
Sections: 2.3, 2.3.1, 2.4, 2.4.1, 2.5 & 3.2

Contents

1 Introduction

Historically, housing has served as a base providing people with necessary shelter and security. Over time, housing has become not just a base but an integral part of the economy. Whether we look at the household life cycle, in which a home purchase is the most significant investment, or at the macro level, where housing market dynamics are used as a vital indicator of how the general economy is performing. In recent years, however, house prices seem to have drifted away from the general economic trend, especially in Copenhagen. With sharp fluctuations in the upward direction, this creates uncertainty for first-time buyers, housing developers, and policymakers relying on economic attributes for predicting house prices. This uncertainty raises the question, what characteristics can then be used to predict house prices? Moreover, can geospatial data help improve the predictions of housing prices?

Traditionally, social scientists have used a hedonic approach, where characteristics associated with the good such as the number of bathrooms, lot size, etc., help to explain and predict house prices. Following the Alonso-Muth-Mill urban economic model of the late 1960s, where agents live around a central business district, Kain and Quigley (1970) [2] include proximity to the central business district in their prediction of house prices.
Since then, the field has developed rapidly, making the hedonic model one of several popular methods for predicting house prices. Zulkifley et al. (2020) [11] shows that several recent studies have opted for other techniques such as Random Forest Regression, XGBoost, and Artificial Neural Networks. Today, social scientists are equipped with advanced statistical tools and open source data on a large number of data. They are no longer limited by assumptions such as the homocentric assumption of a central business district. As Salganik states, "big data systems are always-on" Salganik (2019) [7]. Zulkifley et al. (2020) [11] summarises that the most commonly used geospatial measures in the literature are distances to shopping centers, schools, and public transportation. Park B and Bae JK (2015) [6] find that machine learning algorithms improve the predictability of house prices. Koktashev et al. (2019) [3] finds using Random Forest Regression that the location of residential properties significantly affects house value. Although machine learning algorithms exempt us from certain constraints on assumptions and conventional estimation methods, the study of research is still relatively new, as noted in [3].

In this article, we aim to investigate whether geospatial data can help improve the prediction of house prices. We use data on housing transactions in Copenhagen from Boliga and geospatial data primarily from OpenStreetMaps. Based on a filtered version of the data, we first employ a baseline model that predicts our target variable house prices solely based on structural characteristics as features. We will compare the prediction performance of the baseline model with an extended model incorporating geospatial data using nearest distances. We apply the algorithms OLS, LASSO, and Random Forest Regression. Based on the root mean squared error (RMSE), we find that the Random Forest Regressor vastly outperforms the linear regression algorithms. The same conclusion can be drawn when accounting for polynomial features. Furthermore, geospatial attributes help improve the performance prediction of all the algorithms mentioned above.

The remainder of the article is organized as follows: section 2 describes the structural and geospatial data used in the analysis, how we obtained the data, cleaned the data, and conducted feature selection. Section 3 briefly introduces the machine learning algorithms employed. Section 4 presents the implementation of the algorithms and the main results. Section 5 discusses the results and the limitations of the algorithms and data used, and in section 6 we conclude on our findings.

2 Data

The notebook "isds2022_group9.ipynb" provides all the code and is publicly available on GitHub[1].

2.1 Fetching Data

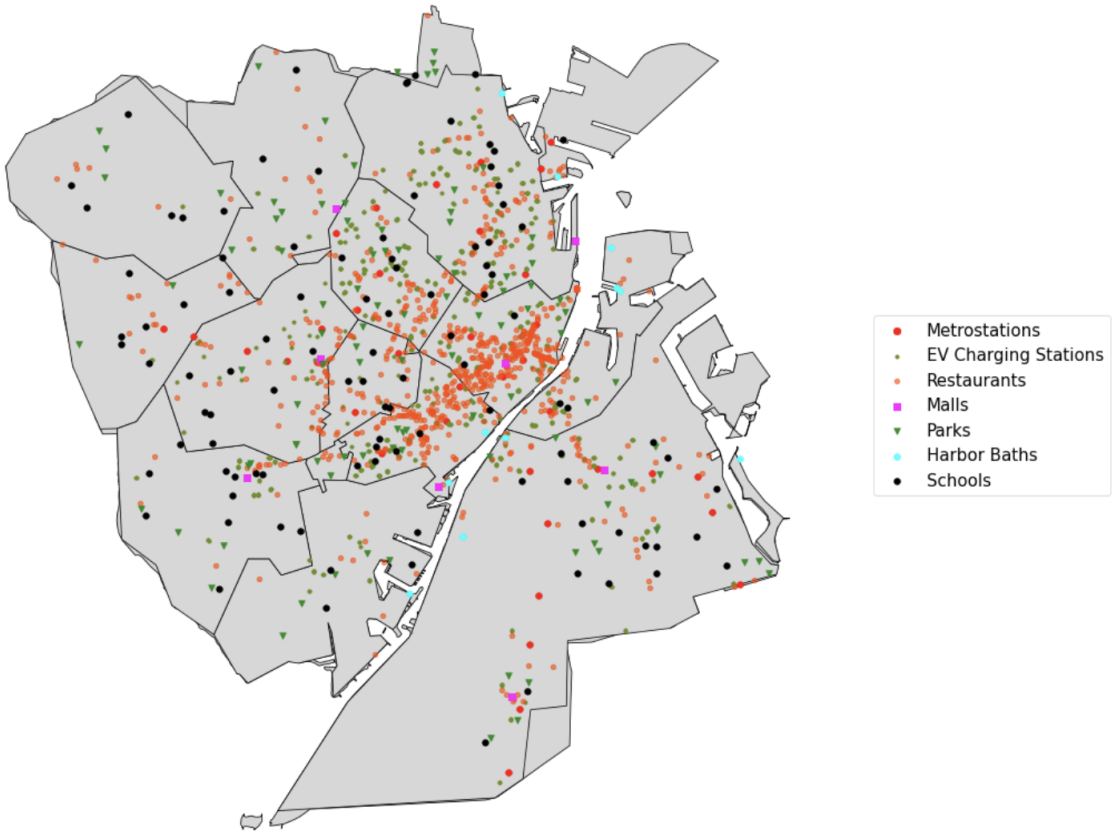
The data used in this article can be split into two main categories: The structural attributes and geospatial attributes. Considering the purpose of this paper, i.e. analyzing whether geospatial data can help improve the prediction of housing prices, this distinction allows for a convenient analysis of the effects of geospatial attributes.

2.2 Geospatial Attributes

Based on the literature presented in the introduction section, a property's proximity to schools and malls is among standard amenities when predicting house prices. In addition to these amenities, we opt to explore geospatial features such as distance to parks and harbor baths under the assumption that recreational areas should increase house prices. In addition, we include distance to restaurants as this may be a measure of urban vibrancy.
A great source for retrieving geospatial data is OpenStreetMaps from here on OSM. To query data from OSM we use the Overpass API, which we access through the wrapper overpy[2]. Although OSM is an excellent source of information about geospatial data, the server load is often too high when sending queries via the Overpass API. This can be circumvented by sending a new query, but it may reflect the trade-off of using an open source data provider. Moreover, the contributions to the OSM are based on volunteers. The data are not always curated correctly, which places a greater demand on the data cleaning process. Due to this, the label of harbor baths in OSM is inconsistent, and extracting the coordinates of harbor baths proves to be cumbersome. Instead, we have used the automation tool Selenium to extract coordinates from Visit Copenhagen. We have also extracted data from Open Data DK on charging stations for electric cars and metro stations. Using Open Data DK ensures the data is consistently curated and continuously updated by the authorities. The geospatial attributes used in the article have been plotted in figure 1 using GeoPandas.

[1]Link to GitHub repository: https://github.com/yapicikerem/ISDS_G9
[2]Link to overpy: https://pypi.org/project/overpy/

Figure 1: Locational Attributes in Copenhagen



Note: border lines indicate postal codes.

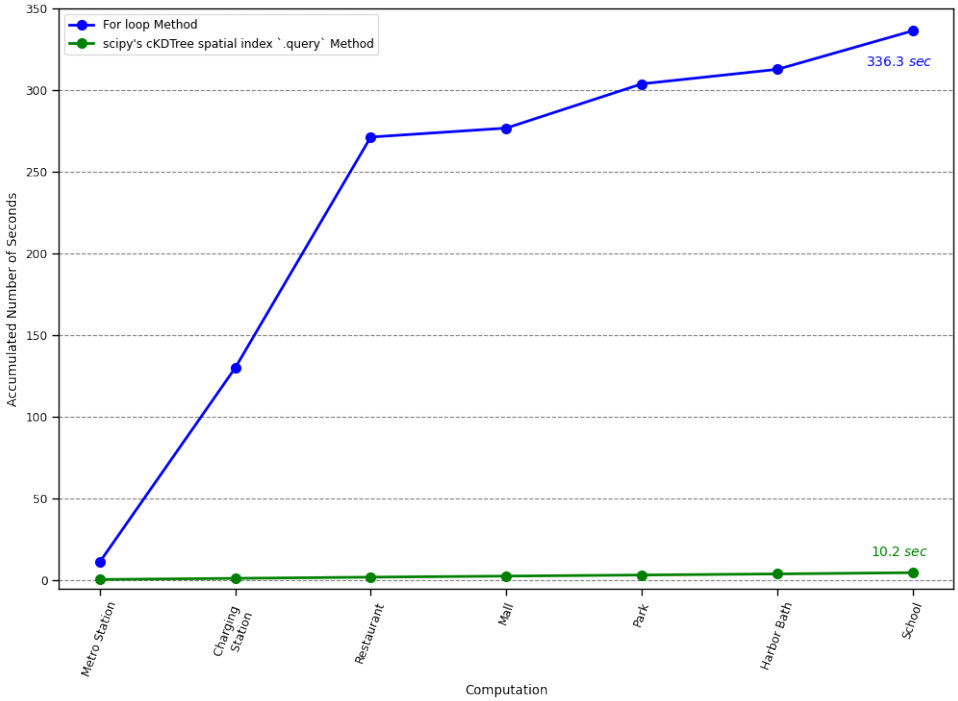
2.2.1 Calculating Nearest Distance

Since the housing market reflects spatial fixity, we calculate each property's proximity to the various geospatial attributes. Working with spatial data from both the Overpass API and the GeoJson files from Open Data DK, the coordinates have to be extracted by converting the geometry column to a string and then using regex to extract longitudes and latitudes. Furthermore, we convert the projection of the GeoDataFrame to easting/northing coordinates to measure distances in meters in Copenhagen correctly.

2.2.2 The Need for Speed

GeoPandas allows us to calculate the distance between geometric objects using a simple for-loop. However, this turns out to be computationally cumbersome because it requires that for each property in the GeoDataFrame, one computes the distance to all geospatial attributes and then chooses the shortest distance. As an alternative, we found that scipy's spatial cKDTree query method significantly improves computation time as the package utilizes vectorization. From Figure 2, it is evident that a for-loop is quite expensive, with a computation time of about 5 minutes and 30 seconds compared to less than 11 seconds for the cKDTree package.

Figure 2: Computing Nearest Distance



2.3 Structural Attributes

Having focused on geospatial attributes, we proceed to structural attributes. Our analysis of the historical property prices in Copenhagen is based on data from Boliga.dk's API, api.boliga.dk. Using this API, we have gained access to approximately ∼118,000 properties sold between the year 2000-2022. The data we obtain here is relatively sparse, but each of these data records has a unique code for its corresponding BBR (Bygnings- & Boligregistreret) information. We thus obtain the BBR information allocated in the API and merge this with the sold properties to obtain detailed attributes for each property unit. The BBR data has approximately 33 extra variables, including basement size, toilet quantity, bathroom quantity, heating code etc., for each property. Structural attributes are likely to determine the price of the housing transaction.
Our method for retrieving data first involved understanding Boliga.dk's html and XHR/API structure. We created a base-url definition in Python and combined it with a query-url with specifications. This includes, e.g. a time limitation period with the start year 2000 to the 27th of July 2022, area limitation (Copenhagen: 101, Frederiksberg: 147), and property types (Villa: 1, Terraced house: 2, Apartment: 3). Subsequently, we have scraped our list of query-url's and saved it as a .json file. This has been done both for housing data and in addition BBR data.

2.3.1 Data Ethics

Data on the Danish housing market are publicly available and can be used freely. The largest housing portals in Denmark are Boliga.dk and Boligsiden.dk. Boliga.dk, in contrast to Boligsiden.dk, has made it possible to obtain their data through an API (api.boliga.dk). Although it is possible to gain access to information about each property on the respective pages freely, data scraping can still be a gray zone on such

portals since it can be classified as misuse (taking data and processing it out of their portal/website). Scraping can further result in server overload.[3] In this project, we did not want to make use of Boliga.dk's data for resale purposes but only used it in connection with an investigation of the danish housing market in an educational context. To avoid the aforementioned server overload, we have set a sleep timer that, after each iteration of the queries, waited up to 0.2 seconds for the following query. As we need to make a query for each single BBR request (unlike sold houses with 50 properties per request), this has also meant that we have waited up to 15 hours to get the entire data set for our project. The presentation of data will further be in aggregated form to avoid presenting specific data for a house that may be in breach of GDPR legislation.

2.4 Data Preprocessing

Throughout the data preprocessing steps, we ensure that no transformations are made that can cause data leakage before splitting our data into train-test splits.

2.4.1 Duplicates and Missing Values

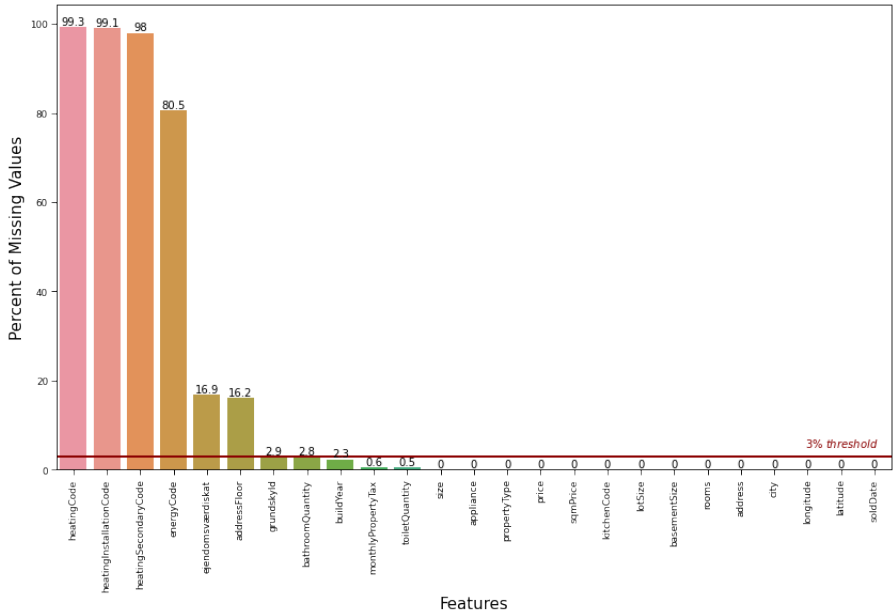
To begin with, relevant data transformations for each variable have been carried out. Duplicates and other non-relevant variables were identified and removed - leaving us with 25 relevant ones for each property. We have removed duplicates based on the address attribute, such that duplicates with the oldest sale dates are removed. This is done under the assumption that the BBR information is correct for the newest sold date[4].
For the remaining variables, we have screened for missing values (given as '-' and '0' in the data set[5]) and removed those with a high proportion of missing values (3% or more). As seen in figure 3 variables such as 'addressFloor', 'ejendomsværdiskat', 'energyCode', 'heatingSecondaryCode', 'heatingInstallationCode', and 'heatingCode' were removed.

[3]See this old case between Boligsiden.dk and Boliga.dk (https://ing.dk/blog/boligsidendk-spaerrer-af-boliga-75381)

[4]This is important if, e.g., housing extensions have been made and recorded in BBR, but these new changes are then reflected in old sale dates records in which the extension was not conducted.

[5]While some properties could have, e.g., zero bathrooms, there are several large villas in Frederiksberg and Valby that are recorded as having zero bathrooms, which is not very likely. We thus regard these as missing values.

Figure 3: Percent of Missing Values



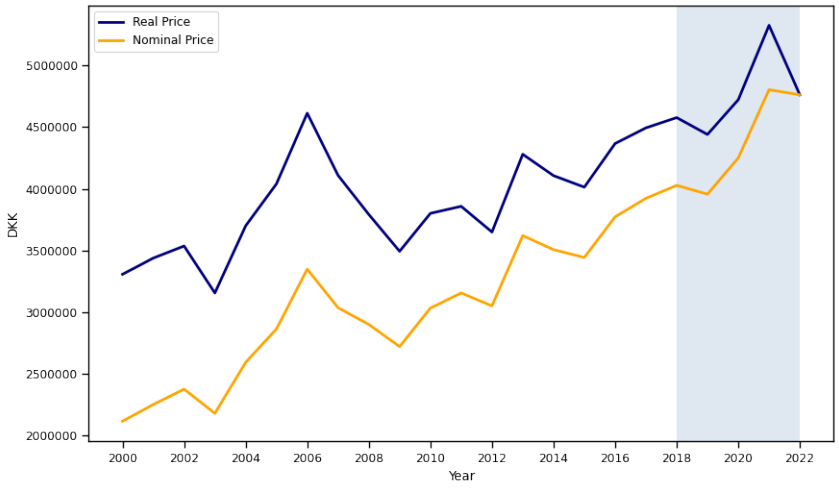
2.4.2 Merging Net Consumer Price Index from Statistics Denmark

Next, we have merged our Boliga data with the Net Consumer Price Index (Net CPI) from Statistics Denmark to adjust property prices for inflation. We opt for the net price index instead of the consumer price index because it adjusts for taxes and is commonly used in the indexation of housing contracts. The base index has been changed from 2015 to 1^{st} of July 2022. By adjusting for net inflation, we uncover the real growth of the property prices and potentially mitigate bias in our data set. Furthermore, we expect this adjustment to yield a more stationary distribution of prices, particularly with regard to the distributions being identical across time. This latter point is important within generalization theory for supervised machine learning where e.g., VC (Vapnik–Chervonenkis) theory[6] builds upon the i.i.d. assumption. The requirement in regards to generalization theory is quite intuitive since ML algorithms trained on housing prices drawn from a distribution under *one* regime is not expected to generalize well if observations in the test set follow a distribution from a different regime. To get a more accurate picture of the development in housing prices, the variable real_sqmPrice has been screened for extreme values, and upper- and lower limits have been set. The upper limit will be DKK 160.000 since this is the highest square meter price for a sold property in Copenhagen historically[7], however lower upper caps could have been considered. The lower limit will be set to DKK 10.000. This is justified on the basis that houses with a real_sqmPrice under DKK 10.000 are subject to other dependencies. Similarly, higher lower caps could have been considered.

[6]Details about VC theory can be found in Abu-Mostafa et al. (2012) [1]

[7]https://www.boligsiden.dk/nyheder/udbud/danmarks-dyreste-kvadratmeterpriser-der-spraenger-budgettet/

Figure 4: Nominal- and Real House Prices Adjusted for Net Inflation*



*Period: 2000-2022

Figure 4 shows the price development from 2000 to 2022. The figure shows that net inflation can account for some but not all the growth in house prices. One immediate explanation could be that an increase in Copenhagen's population since the early 2000s has driven up housing demand and hence prices. We thus include time dummies in our regression in hopes that it can account for time-varying effects, herein the distributional non-identically across time. We will focus on housing transactions in the period spanning from 2018 to 2022 for two main reasons: First, a shorter time span makes concerns about time-dependent effects far less severe. However, this comes at the price of less data. This trade-off is generally favored in terms of having quality data rather than a lot of data, as discussed in[8]. Second, we wish to use distances to the metro lines M3 and M4 and to charging stations for electric vehicles. These are relatively new phenomena in Copenhagen, and we thus risk introducing spurious correlations by including old housing properties in which sales prices did not reflect this information.

2.4.3 Descriptive Statistics

For the analysis of housing prices, we will consider the logarithm of real housing prices. The reason for this is twofold: First, it is more natural to think of the relationship between housing prices and features in terms of a multiplicative structure (relative changes) rather than an additive structure (nominal changes). Second, linear regression models that minimize squared residuals are sensitive to outliers (due to the squaring), which the log transformation circumvents to some degree.
Before proceeding with log prices, we briefly showcase selected properties in table 1 with focus on real prices.

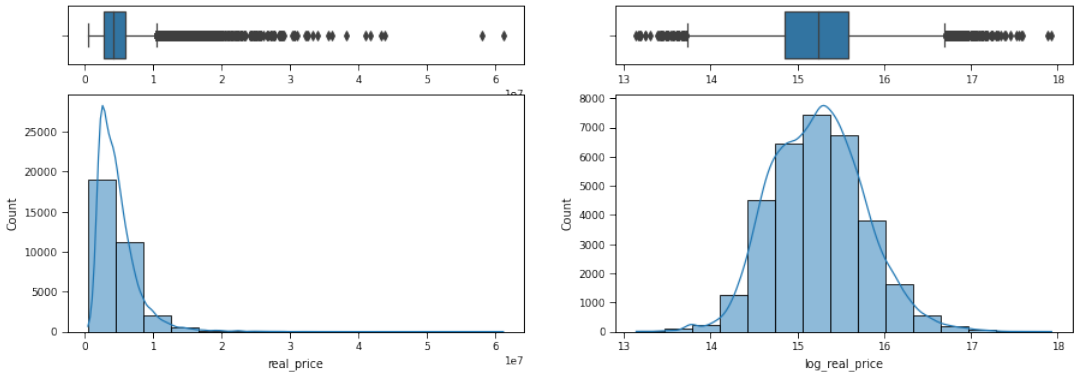
[8]See Meng (2018) [4] for a discussion.

Table 1: Descriptive Statistics on Selected Property Variables

	real_price	real_sqmPrice	size	rooms	buildYear
count	32,929.00	32,929.00	32,929.00	32,929.00	32,929.00
mean	4,811,305.00	51,225.82	91.78	3.15	1,943.07
std	3,005,766.00	15,255.33	40.17	1.34	44.58
min	511,363.60	10,013.33	24.00	1.00	1,623.00
25%	2,805,556.00	41,471.05	61.00	2.00	1,908.00
50%	4,133,333.00	49,454.55	84.00	3.00	1,935.00
75%	5,892,256.00	58,891.84	112.00	4.00	1,974.00
max	61,150,000.00	159,813.08	616.00	14.00	2,022.00

Table 1 shows us that the sample size after the initial preprocessing and cleaning of the raw dataset leads to 32,929 observations. The average home in Copenhagen appears to have a size of ∼92 square meters and with ∼3 rooms with a square meter price of DKK 51,225.82.
Next, we demonstrate some of the implications from using log prices. In figure 2 below, we see that the distribution for real house prices is right-skewed, whereas the log transformation makes the distribution more centered, although still fat-tailed.

Figure 5: Housing Prices: Normal and Log-Transformed



Lastly, through spatial join in GeoPandas, we have merged the filtered data on houses with the different postcodes in Copenhagen and Frederiksberg municipalities in order to create a heat map. The postcodes are colored according to their average transaction price in the period 2018-2022.

Figure 6: Average Real House Prices by Zip Code

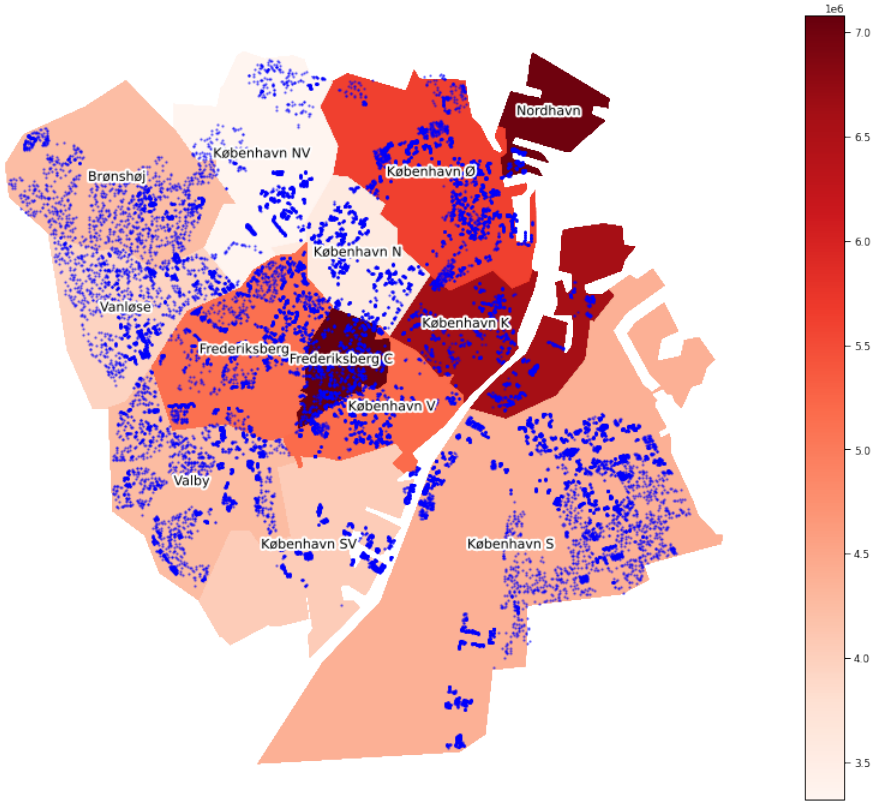


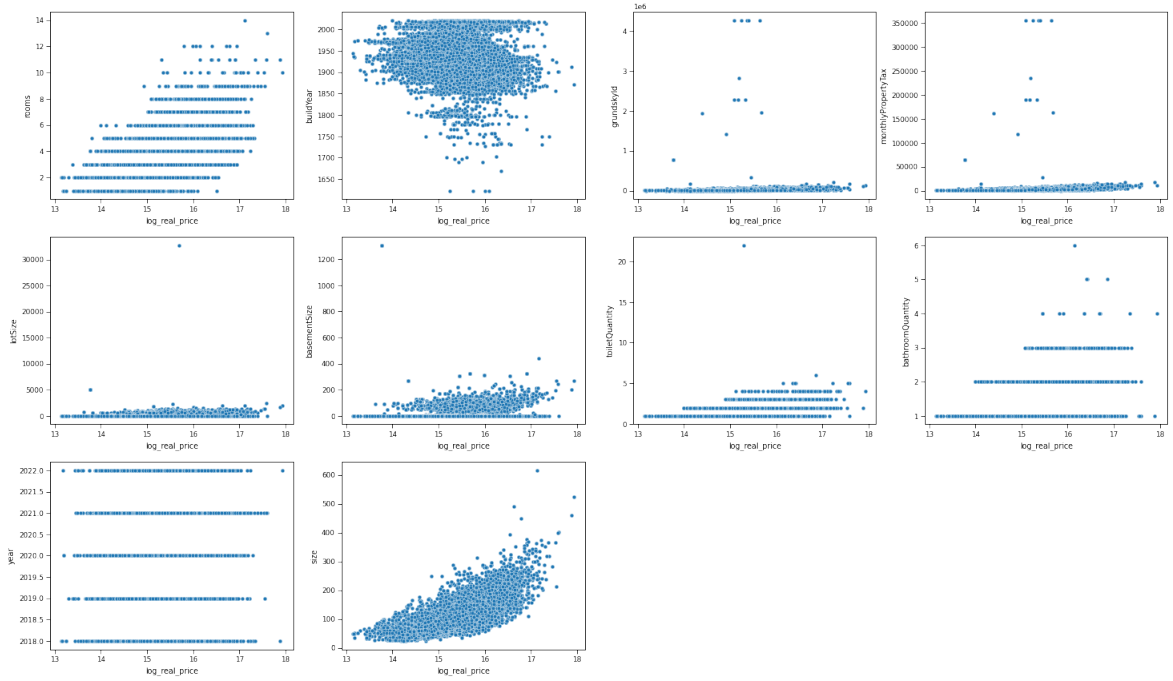
Figure 6 shows the average real house prices by postcode where areas of the highest average house prices include 2150 (Nordhavn), 1800-2000 (Frederiksberg), 1000-1499 (Kbh K), 2100 (Kbh Ø) and 1500-1499 (Kbh V). Prices in these areas are approx. DKK 5.5m and up. The house price level outside the inner city is significantly lower than in the inner city.

2.5 Feature Selection

In this section, we base feature selection for our baseline linear regression model on scatter plots and correlation plots in figure 7 and 8, respectively. Due to readability, the pairwise relationships between the numeric variables and the target are visualized in 7. We refer to appendix[9] for the complete scatter plot.
As several of the scatter plots that all include log_real_price seemingly contain non-linear relationships, we investigate the importance of including polynomial features in our linear regression analysis. In particular, we settle for polynomial features of degree 2. The scatter plots also enable us to detect outliers that should be accounted for[10].

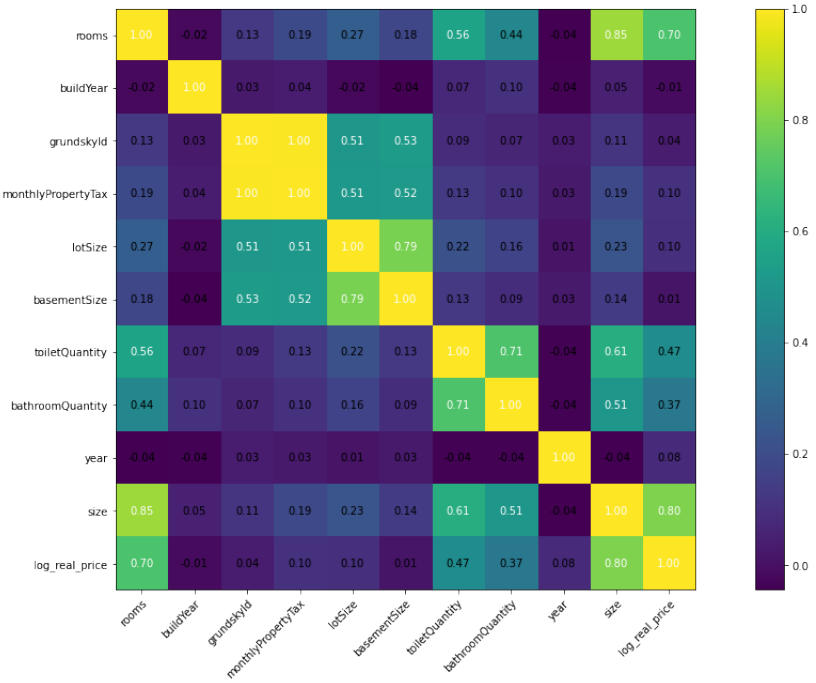
[9] See figure 11
[10] As an example, we discard the observation with 20+ toilets.

Figure 7: Scatter Plot for Selected Variables Against Target



Based on the correlation plot in 8 using Pearson's correlation coefficient, we argue that buildYear, grundskyld, and basementSize are likely unimportant for explaining housing prices due to their low correlation patterns, although there could be non-linear dependencies. In particular, basement size is likely to introduce unwanted correlation, as most properties in Copenhagen (specifically expensive properties) have zero basements.

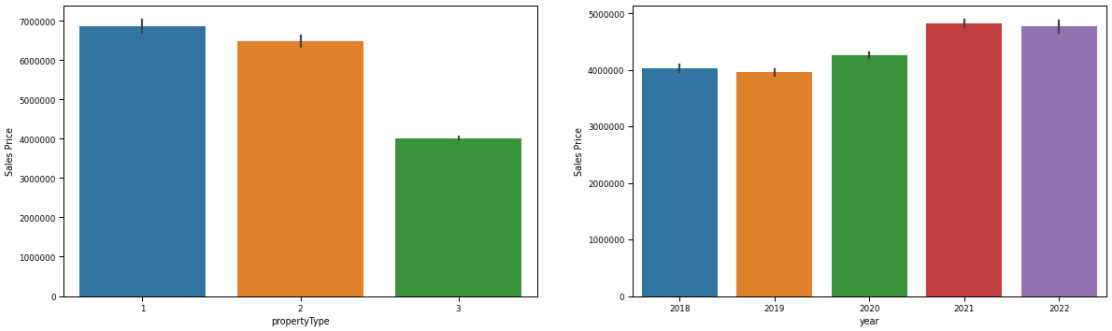
Figure 8: Log Real Price Correlation Matrix



For categorical variables, we include the variable propertyType and the year variable

using dummy encoding. An overview of housing prices across these categorical variables is shown in figure 9 below. The bar plot shows the mean value with an associated error band. The small error bands may indicate a concentration of the data implying that the plotted mean is very likely.

Figure 9: Bar Plot of House Prices by Categorical Variable



Note: propertyType: Villa: 1, Terraced house: 2, Apartment: 3

3 Machine Learning Algorithms

In this section we briefly introduce the machine learning algorithms used for predicting house prices. For the linear models we wish to estimate the following model:

$$Y = X\beta + \varepsilon \tag{1}$$

Where Y is a $n \times 1$ vector, X is a $n \times p$ matrix, β is a $1 \times p$ vector, and ε is a $n \times 1$ vector, where n denotes the number of observations and p denotes the number of features including the intercept.

3.1 Ordinary Least Squares (OLS)

The objective in OLS is to estimate (1) by choosing the argument, β, that minimizes the sum of squared errors:

$$\hat{\beta} = \arg\min_{\beta} = \varepsilon'\varepsilon = (Y - X\beta)'(Y - X\beta) = ||Y - X\beta||_2^2 \tag{2}$$

Where $\hat{\beta}$ is the well-known OLS estimator. We use sklearn's linear regression package which is built on the scipy's least-squares solution involving matrix algebra to minimize (2) [11] Due to the inclusion of polynomial (cross)-terms, OLS is likely to overfit as we have a lot of features. The next algorithm tries to avoid this by regularization.

3.2 LASSO

LASSO regression[8] is very similar to OLS except for the fact that it includes a penalization term in the objective function in (2):

$$\hat{\beta}_\lambda = \arg\min_{\beta} = ||Y - X\beta||_2^2 - \lambda||\beta||_1 \tag{3}$$

[11] Documentation for scipy's least-squares solution method:
https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.lstsq.html

The LASSO regression thus uses the $L-1$ norm to penalize large weight coefficients associated with our features, where λ denotes the strength of regularization. The choice of λ thus has to be chosen such that it delicately balances the bias-variance tradeoff. Later, we will use a variant of k-fold cross validation designed to deal with the time series aspect of our data to fine-tune the hyperparameter. In conventionel k-fold cross validation, randomly partitioned subsets will break the time dependent structure in our data. This is circumvented simply by doing k-fold cross validation on a rolling basis instead.

3.3 Random Forest Regression

Next, we consider Random Forest regression which constitutes of an ensemble of decision trees[12]. These decision trees are binary in the sklearn package. As decision trees are mappings that are non-linear in nature, Random Forest regression is typically good at recognizing non-linear relationships. Decision trees work by splitting the nodes in the tree based on an information gain measure, which is to be maximized. This measure is defined in terms of an impurity measure, in which case we will consider the mean squared error[13]. As the structure of individual decision trees is highly sensitive to changes in the training data, they tend to have high variance (overfitting). The Random Forest regressor solves this by bagging (bootstrap aggregating)[14], where it creates bootstrap subsets of the data for randomly selecting features. The Random Forest regressor then outputs the average prediction across the trees.
For cross-validation we consider an exhaustive grid search over the number of trees ranging from 64-128 and the max depth of each tree ranging from 2-6. Once again, this is done on the development data with k-folds on a rolling basis.

4 Performance Evaluation

In this section, we consider predicting housing prices based on two models: A baseline model and an extended model. The baseline model includes features solely based on structural attributes, whereas the extended model extends the baseline model with our geospatial attributes. An overview of the included variables in the baseline model and extended model is given in the appendix[15]. In what follows, we will consider an 80/20 train-test split following the Pareto principle.

To evaluate the performance of our models, we choose the root mean squared error (RMSE) as our performance measure. When computed on the test data, this can be thought of as a proxy for the true underlying out-of-sample RMSE (the more test data, the better the proxy). For the linear models, we evaluate performance both with the inclusion and exclusion of polynomial features. For fine-tuning the hyperparameter in LASSO, we construct a fine grid from -4 to 4 on log scale. We use sklearn's

[12]See Mirjalili et al. (2019) [5] chapter 3 for a thorough explanation of decision trees in the classification scenario.

[13]See chapter 10 in in Mirjalili et al. (2019) [5] for a detailed discussion.

[14]See chapter 7 in Mirjalili et al. (2019) [5] for a detailed discussion.

[15]See table 4

TimeSeriesSplit() function with 5 folds for cross validation[16]. The results for OLS and LASSO are shown in table 2. For OLS, the results clearly indicate the importance of finding a fine balance between a overfitting and underfitting: With too little information, i.e., a baseline model with no polynomials, OLS is on average 11 million DKK away from the mean. In contrast, when we extend the model and include polynomial features, the amount of features increases significantly (as cross-terms are also included), so the out-of-sample error explodes (notice the low in-sample error). On the contrary, LASSO's performance across models is a lot more stable due to the inherent regularization. Further, the extended LASSO model with polynomial features appears to be the best performing linear model.

Table 2: Linear Regression Results

	Baseline model				Extended model			
	Polynomials		No Polynomials		Polynomials		No Polynomials	
	Train RMSE	Test RMSE	Train RMSE	Test RMSE	Train RMSE	Test RMSE	Train RMSE	Test RMSE
OLS	1,641,873.94	4,149,513.50	2,689,223.56	11,413,071.54	1,452,110.28	INF	2,484,321.06	8,524,496.93
LASSO	2,026,218.76	4,279,993.58	2,026,218.76	4,279,993.58	2,002,229.23	4,065,299.39	2,024,042.83	4,270,531.99

Note: All values in DKK.

Table 3: Random Forest Regression Results

	Baseline Model				Extended model			
	Train MSE		Test MSE		Train MSE		Test MSE	
	Avg.	Std. dev	Avg.	Std. dev	Avg.	Std. dev	Avg.	Std. dev
Random Forest Regression	1,535,525.25	869.30	1,905,546.53	3,702.49	1,480,653.47	2,283.00	1,843,235.76	3,614.66

Note: All values in DKK. Results are based on 5 repetitions.

In table 3, we show the results for the Random Forest Regression. For cross-validation we have again used the TimeSeriesSplit() function with 5 folds. Because of the inherent randomness in the algorithm, we assess the robustness of the algorithm by training it repeatedly 5 times with different seeds. We then average over the outcomes and provide standard deviations. Evidently, Random Forest Regression outperforms the linear models significantly[17].

To interpret the implication of our best performing model, we are on average about 1.8 million DKK away from the mean. At first, this may sound absurdly high and indicate that our model is performing very poorly. However, it turns out that our model performs rather well on "normal" housing labels. By this, we mean that whenever we observe typical housing labels in the range of 1-15 million DKK, it manages to predict a price very close to the label. On the other hand, it does a very poor job at predicting very expensive housing prices. The maximum prediction Random Forest Regression provides is around 27.5 million DKK, whereas the highest label in the test set is around 61.5 million DKK. Further, as was evident from the descriptive analysis, there are many points in our data set where the sales price is very expensive. These two factors explain why the RMSE intuitively seems bad, and one may therefore want to consider a different performance measure to more rightfully assess the performance of the model. In hindsight, it would probably be better to measure performance by how far off prices are relatively: A prediction that is 2 million DKK off for a house sold at 1 million DKK is far worse than being 2 million DKK off for a house sold at 61

[16]Details for the function can be found here:
https://scikit-learn.org/stable/modules/generated/sklearn.model$_s election.TimeSeriesSplit.html$

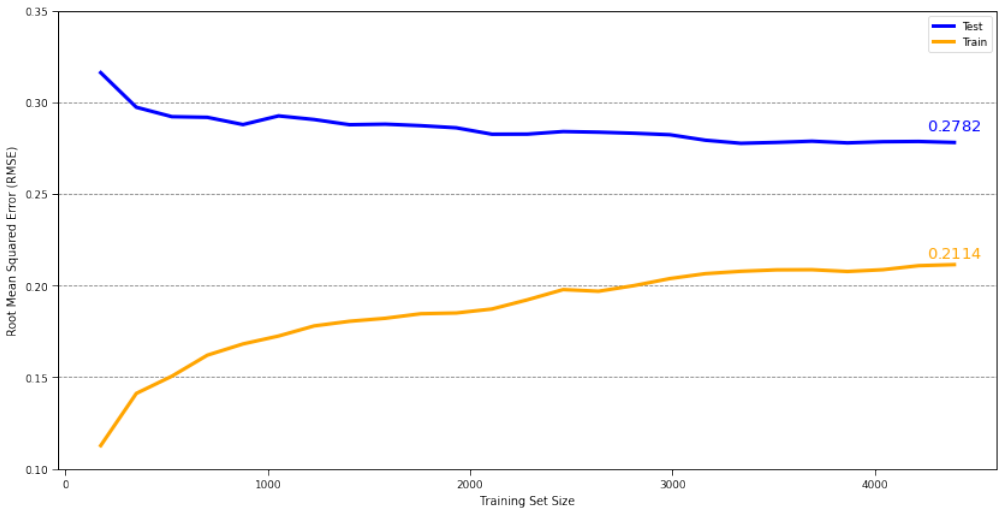
[17]Due to the far better performance, we chose to disregard checking the robustness of the extended model with LASSO regression with regards to the choice of hyperparameter.

million DKK (most people would probably even consider this great!). At the end of the day, we wish to assess whether spatial attributes can help improve the prediction of housing prices. Based on the results for LASSO and Random Forest Regression (and even in OLS disregarding the overfitting scenario), we find evidence that spatial attributes can in fact improve housing prices.

4.1 Learning Curve for Extended Model with Random Forest Regression

In figure 10 below we report the learning curve for our preferred model with Random Forest Regression for *log(real_prices)*[18]. It has the classical appearance with indications of overfitting for sufficiently small samples, but this issue is remedied as the sample size increases. As we have limited our data set to observations from 2018-present for reasons provided earlier, it is of interest to assess whether we may have been better off by increasing the sample size. While the learning curve indicates that some improvements are feasible, it seems to flatten fairly quickly. It is therefore not likely that a larger sample size can improve the results significantly.

Figure 10: Learning Curve for the Extended Model with Random Forest Regression



Note: The y-axis denotes the RMSE in terms of log(real_prices).

5 Discussion

5.1 Data

In addition to structural and geospatial attributes, we could have explored the realm of neighborhood qualities such as the efficiency of public education, crime rates, and pollution, among others. Alternatively, economic attributes such as interest rates would have been of interest, as conventional macro theory suggests that user costs rise with interest rates. However, as noted in Zulkifley et al. (2020) [11] economic and

[18] We have used sklearns integrated learning_curve() function, however this function does not allow for transformation to real prices before measuring the RMSE. The qualitative conclusions in terms of log(real_prices) is the same.

neighborhood attributes prove more challenging to define and measure.
Although we briefly describe the housing market by postal code in section 2.4.3, we do not use postal code dummies as features in the machine learning models as it increased the polynomial feature complexity significantly. However, we find it plausible that some structural differences within each postal code might have been relevant to investigate further.
Moreover, the data used depends on the prediction purpose of the model. If the purpose is to get an algorithm to value more expensive houses, expensive houses should be included. Alternatively, we could redesign the project to exclude expensive houses in the data cleaning process and presumably improve our model's performance prediction.

5.2 Methods

We had initially planned to evaluate the performance using the XGBoost algorithm, but due to time constraints, we have not done so. XGBoost is a natural extension to our analysis and is known to perform well on housing prices. For example, Zhou (2020) [10] found that XGBoost outperforms the models employed in this article.

Generally, the purpose of machine learning models is purely performance prediction. This article does not seek to quantify a causal relationship between house prices and any specific geospatial data, as this is generally not the purpose of machine learning. Although, as pointed out in Varian (2014) [9] machine learning may help in estimating the causal impact of, say an intervention of the housing market, however, this is beyond the scope of the article.

6 Conclusion

In this paper we have examined how well different machine learning algorithms performs on recent housing data in Copenhagen retrieved from Boliga. In order to enhance model performance, we have investigated the data and used descriptive statistics to arrive at a representative data set with relevant features. Overall, we find that all of our considered machine learning algorithms perform better when we include geospatial attributes. Further, the Random Forest Regressor gives the best performance in terms of having the lowest root mean squared error.

7 Appendix

Figure 11: Scatter Plot Matrix for Numerical Variables

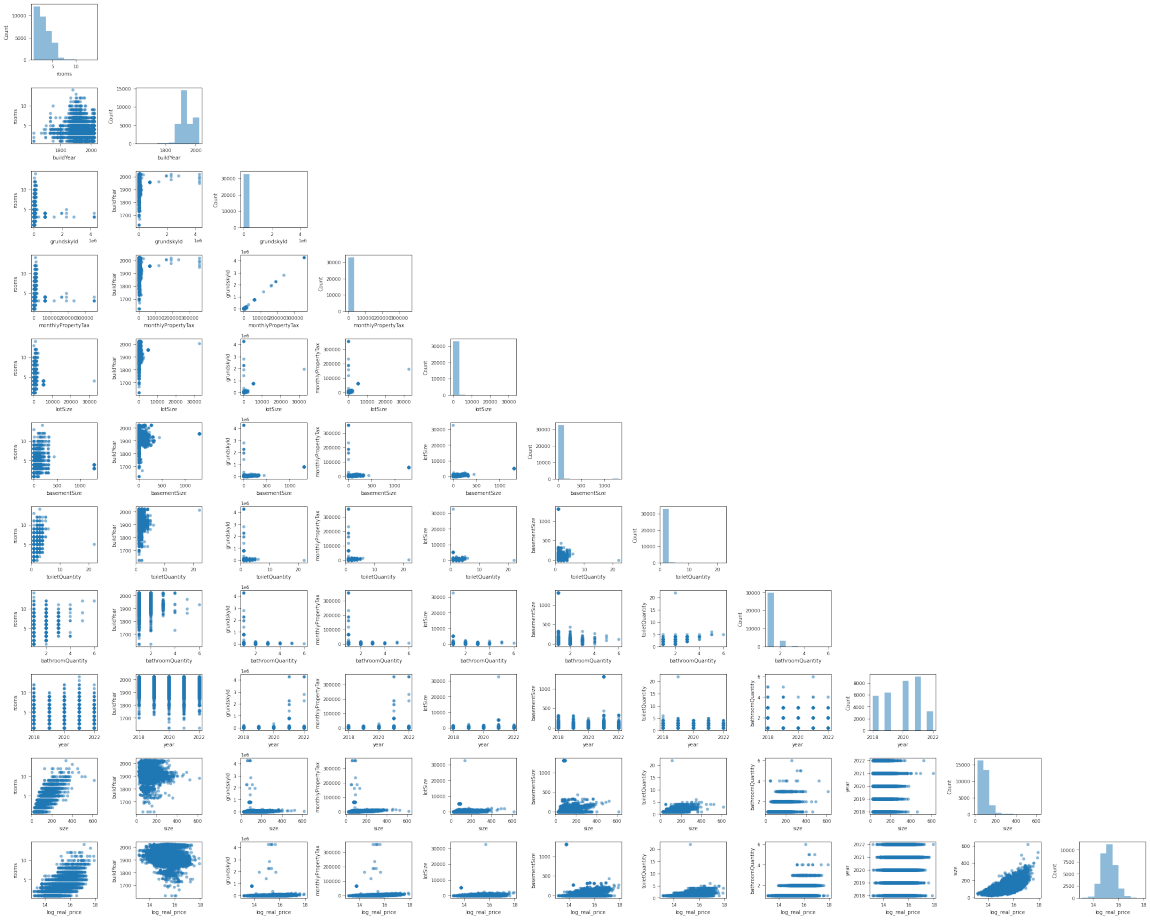


Table 4: Variable Selection in Baseline and Extended Model

Variable	Baseline model	Extended model	N/C
log_real_price	x	x	N
size	x	x	N
rooms	x	x	N
monthlyPropertyTax	x	x	N
lotSize	x	x	N
toiletQuantity	x	x	N
bathroomQuantity	x	x	N
propertyType	x	x	C
year	x	x	C
nearest_dist_charging_station		x	N
nearest_dist_metrostation		x	N
nearest_dist_restaurant		x	N
nearest_dist_mall		x	N
nearest_dist_park		x	N
nearest_dist_school		x	N
nearest_dist_bath		x	N

Notes: N/C: Numerical/Categorical variable

References

[1] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning From Data*. 2012.

[2] J. F. Kain and J. M. Quigley. Measuring the value of housing quality. *Journal of the American Statistical Association*, 65(330):532–548, 1970.

[3] V. Koktashev, V. Makeev, E. Shchepin, P. Peresunko, and V. V. Tynchenko. Pricing modeling in the housing market with urban infrastructure effect. *Journal of Physics: Conference Series*, 1353(1):012139, nov 2019.

[4] X.-L. Meng. Statistical paradises and paradoxes in big data (i): Law of large populations, big data paradox, and the 2016 us pre. *The Annals of Applied Statistics*, 2018.

[5] V. Mirjalili and S. Raschka. *Python Machine Learning - Third Edition*. 2019.

[6] B. Park and J. K. Bae. Using machine learning algorithms for housing price prediction: The case of fairfax county, virginia housing data. *Expert Systems with Applications*, 42(6):2928–2934, 2015.

[7] M. Salganik. *Bit by Bit: Social Research in the Digital Age*. Princeton University Press, 2019.

[8] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 1996.

[9] H. R. Varian. Big data: New tricks for econometrics. *Journal of Economic Perspectives*, 28(2):3–28, 2014.

[10] Y. Zhou. Housing sale price prediction using machine learning algorithms. 2020.

[11] N. Zulkifley, S. Rahman, U. Nor Hasbiah, and I. Ibrahim. House price prediction using a machine learning model: A survey of literature. *International Journal of Modern Education and Computer Science*, 12:46–54, 12 2020.