FACULTY OF SOCIAL SCIENCES
Department of Economics
University of Copenhagen

Exam in Social Data Science

51, 53, 65 & 95

Finding the best predictor for housing prices in the greater Copenhagen area

ECTS points: 7.5

Date of submission: 01/09/2018

Keystrokes: 17 standard pages

Contributions

This project is written jointly and we all support every part of the project. But as it is required to show who contributed with writing which part of the report, we have divided the sections between us in the following manner:

Co-writtten Section 1 and 8.

51 has written Section 2 par. intro-2, Section 3.3, Section 4 par. 3, Section 5, Section 5.2.2, Section 5.3.2, Section 6.2.1 and Section 7.3.

53 has written Section 2 par. 3-5, Section 3.4, Section 4 par. 4, Section 5.1, Section 5.2.3, Section 6 and Section 6.3.

65 has written Section 3.1, Section 4 par. intro-1, Section 4 par. 5, Section 5.2, Section 5.3, Section 6.1 and Section 7-7.1.

95 has written Section 3.2, Section 4 par. 2+6-7, Section 5.2.1, Section 5.3.1, Section 6.2 and Section 7.2.

1 Introduction

In this paper we aim to investigate which machine learning technique gives the best prediction of sales prices in the greater Copenhagen area.

An accurate prediction of property prices is of utmost interest, as it has multiple applications.
First of all an accurate prediction of property prices at a given point in time is important, since 80 percent of the danish people view their housing as part of their pension scheme.[1] So whether the price is high or low, in terms of debt in the house, has a big impact on the living standards for these people.
Secondly both investors, the house sellers and house buyers are interested in knowing whether the property is over or under priced.
Lastly public institutions such as SKAT might be interested in attaining the value of a given property to make precise valuations of property taxes.

The focus of this project is the housing market of the greater Copenhagen area, since the demographic development predicts that more and more people are going to live there in the coming years.[2]

In our effort to do this we scrape data from *Boliga.dk* and employ various machine learning techniques. We have chosen to compare Ordinary Least Squares (OLS), Lasso and Ridge. Lasso and Ridge are employed, as we feel that these are a natural improvement of OLS in terms of prediction. To evaluate our model we calculate the Mean Squared Error (MSE).

Given our prior knowledge we would expect that the Lasso and Ridge outperform the OLS, as fitting an OLS with many features risks producing a model with overfitting and thus causing bad out-of-sample predictions. If one is interested in getting a more simple model the Lasso model has the advantage of making a more parsimonious model, whereas Ridge handles exploding coefficients well. However if one values prediction accuracy higher, the features of the Ridge model is more compelling. This is due to the penalty used in the Ridge model, which allows for more features and a better prediction, but not a very simple model.

[1] Article from Politiken:
https://politiken.dk/oekonomi/bolig/art5492292/Pas-p%C3%A5-de-3-f%C3%A6lder-Er-din-bolig-ogs%C3%A5-din-pensionsopsparing

[2] Article from DR:
https://www.dr.dk/nyheder/regionale/hovedstadsomraadet/hovedstaden-har-voksevaerk-100000-flere-koebenhavnere-om-11-aar

The remainder of the project is ordered as follows, Section 2 gives a brief review of existing literature. Section 3 presents our scraping method and data cleaning, as well as a discussion of any ethical and legal concerns. Section 4 is our descriptive analysis. Section 5 gives an overview of the machine learning methods we apply. Section 6 presents our prediction method and our results. Section 7 includes a discussion of our results, how the model could be improved and which other machine learning methods could be applied to our problem and finally we conclude in Section 8.

2 Literature review

This section reviews some of the literature both on using machine learning as a social scientist and on using machine learning to make predictions on the housing market.

Hindman (2015) talks about how machine learning can be used in the social sciences to make more accurate predictions, even with small data sets. He explains *"building simplified models, social scientists should avoid standard OLS and logit models, and instead use penalized regression techniques such as the Lasso"*. He elaborates saying that supervised learning models such as classification and regression trees, dimension reduction/singular, value decomposition, nearest neighbor methods, support vector machines should be of great interest to social scientists. Further he especially highlights the penalized regression techniques saying that they are *"an important subplot of the big data revolution"*. Giving praise to the Lasso model for producing parsimonious models, which he says is one of the building blocks of a good model along with good prediction. Hindman (2015) concludes saying *"In a few years—if we are lucky—fitting statistical models without a holdout sample will seem similarly absurd"*.

In the paper by Borde et al. (2017) they use different machine learning algorithms to best predict the real estate price trends in Mumbai. They employ Linear Regression both with Least Squares and using gradient descent, K Nearest Neighbor regression and Random Forest regression. Their features include e.g. distance to airport, hospital count and a variable containing year/quarter. Using the different algorithms they obtain the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and the Mean Absolute. Comparing the different errors for each algorithm, the algorithm with the smallest error is chosen. They conclude that the Random Forest algorithm does the best job when it comes to predicting the real estate price trends. However they also conclude that it is only able to predict near future trends with good accuracy, as the time/quarter feature was not incorporated accurately.

Similarly the paper by Masías et al. (2016) compares the predictive performance of Ordinary Least Squares Regression with that of the Random Forest regression, the Neural Network and Support Vector Machine approaches using real data on the housing market of Santiago in Chile. Their features include among others number of bedrooms, usable floor area in square meters, latitude and longitude of property locations and property type. They find that the Random Forest algorithm gave the best prediction by comparing RMSE, MAE and the R squared. Furthermore they conclude *"that machine learning techniques can provide a useful set of tools for acquiring information on housing markets"*.

Based on the demographic development in the greater Copenhagen area, predicting the housing market is therefor of special interest. And based on the reviewed literature we feel confident that using machine learning algorithms, and more specifically the Lasso and Ridge methods, is a solid method to predict property purchasing prices.

3 Getting and cleaning the data

In this section we first discuss our ethical considerations with regards to scraping data from *Boliga.dk*, then our scraping procedure is described, followed by our data cleaning and lastly there is a brief description of how one makes a choropleth map.

3.1 Legal and ethical concerns

With regards to our method of scraping the purchase prices of properties in the greater Copenhagen area from *Boliga.dk* some legal and ethical concerns were taken into consideration. One such concern was a privacy concern. Scraping down to the level of address would enable further analysis (distance to different points of interest), but it could also be misused to find specific people and their economic situation. Therefore, to respect privacy, we have chosen to use data only at zip code level.

Another concern we have had is that automatized scraping is in violation of *Boliga.dk*'s terms of service agreement,[3] as it bans all scrapers. This may be due to not wanting undue stress on their servers, as their terms of service agreement also very explicitly states that undue stress to their infrastructure is forbidden. We follow the thoughts of Soeller et al. (2016) and therefore we try to alleviate this problem by having a 5-6 second delay between each request. This reduces our request-frequency to (or slower

[3] *https://www.boliga.dk/vilkaar-og-betingelser*

than) a person using *Boliga.dk* for regular use. It should be noted that we only scrape their archives, and not any data regarding current properties for sale. Furthermore we only scrape part of their archive, from the period 2010 to 2018 and for certain zip codes.

It is also noted that *Boliga.dk* sells this data,[4] and therefore it could be considered stealing. If we were to publish and/or create a commercial product, we would feel obliged to buy the data, but as this is merely a test-run, we do not believe this to be neccesary.
However, we think that the decision to scrape from *Boliga.dk* can be justified. This is due to the fact that we are not using the scraped data to create a competing service or trying to undermine their business. Furthermore the data is scraped only for research and educational purposes with no intent of publishing. By reducing the information we scrape about each property we also feel that no persons' privacy is violated. On these grounds we see it as justifiable to scrape part of *Boliga.dk*'s archive.

3.2 Web-scraping procedure

For our analysis we scraped data from *Boliga.dk*. We wanted to look at purchase prices for the Copenhagen and Frederiksberg area and went with *Boliga.dk*'s own classification of which zip codes this covers. This means that the data covers most of the greater Copenhagen area. For a list of the included zip codes see Table A.1 in the Appendix.

The data needed could be obtained from their archive and information on sales from January 1st 2010 to January 1st 2018 was scraped. *Boliga.dk* has data on sales all the way back to 1992, but due to new rules from SKAT about how the purchase price is reported onwards from August 2009 [5], we chose to scrape data only back to 2010.

In order to scrape the data three loops were employed. The first loop iterates over zip codes. The second loop iterates over the page number. This means that we visit every page for each zip code area. The third and final loop iterates over each cell i.e. the information for every sale and the variables *zip_code, purchase_price, sales_date, sales_type, kr_m2, rooms, property_type, m2* and *built* are generated. Finally for each page the list with variables are combined to a data frame, that are then saved as a csv-file. The frequent saving is done to minimize the loss of data in case of a malfunction with *Boliga.dk*'s server.
The scraping leaves us with a data set with 9 variables and 84,573 observations ready for cleaning.

[4] *https://www.boliga.dk/boligdata*
[5] *https://www.boliga.dk/salg, "Notat om salgspriserne"*

The scraping process can be examined in Section 1 of our code file.

3.3 Data cleaning

The data we scraped does however need some cleaning, as we have some missing values and outliers. Examining our data shows the variable *kr_m2* has 423 missing values and as this variable is a combination of the variables *purchase_price* and *m2* we drop *kr_m2*. However the variables *m2* and *rooms* also have missing values. 133 rows are dropped to remove missing values from *m2* and 2 rows are dropped from *rooms*. Further we remove 7 rows from January 1st 2018 to ensure we only have whole years.
We chose to divide the purchase price with 100,000 to make data easier to read.

As mentioned we want to predict purchase prices of properties in the greater Copenhagen area, and therefore we only want to look at properties that have been bought and sold under regular market conditions. To achieve this, only sales that have been made as 'normal sales' (category 'Alm. Salg' on *Boliga.dk*) are included. Thus rows with the sales type 'family transfers' (6,727 observations), 'auctions' (940 observations) and the category 'other' (1,316 observations) are removed. This leaves us with a total of 75,448 observations that we feel represent the whole free market of property sales in the greater Copenhagen area. Outliers within the free market will be investigated after our descriptive analysis, which gives us some insights into which outliers there are.
Consult Section 3 in our code file for the data cleaning code.

3.4 How to map the greater Copenhagen area

To get a quick and visualized overview of how the housing prices have evolved from 2010 to 2017 and how it differs between the different zip code areas, we made a map of the greater Copenhagen area. This is included in our Descriptive analysis as Figure 1. This was done by downloading a GeoJSON file covering Denmark according to zip code from the GitHub user Neogeografen.[6] In the GeoJSON file we located our 17 zip code areas and hereafter converted the file into a Shapefile, which we used in our Jupyter Notebook with GeoPandas.
See Section 4, Figure 1 in our code file.

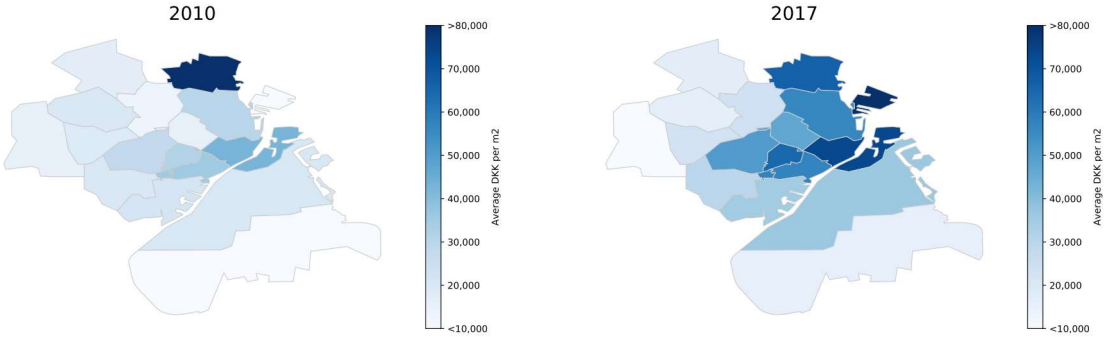
[6] *https://github.com/Neogeografen/dagi*

4 Descriptive analysis

In this section the cleaned data will be presented both visually and with summary statistics.

Figure 1 shows the average price in DKK per square meter for each zip code in the greater Copenhagen area for 2010 and 2017 respectively. The darker the color is, the more expensive the area is. For a map over the areas and which zip codes it covers see the figure in appendix A.2.

Figure 1: Average DKK per m^2 in the greater Copenhagen area



Source: Boliga.dk

From Figure 1 it can be seen that the overall price level in the greater Copenhagen area has gone up over the 7 years, as the map for 2017 is darker. However, some of the outer zip code areas seem to have gone down in price per square meters. The change is most significant in Hellerup, where the price per square meter on average has dropped with around 10,000 DKK.

Figure 1 also shows that Copenhagen and Frederiksberg have seen an increase in prices. København K, Frederiksberg C and the surrounding areas appears to have gone up with around 20,000 DKK per square meter from 2010 to 2017, while København S, København SV and Valby have gone up with around 10,000 DKK per square meter.

It is worth noting that Nordhavn is a relatively new neighborhood, as it got its own zip code in 2013. It does therefore not make sense to see the change in price per square meters as a real change, as there was none to very few housing opportunities in 2010. It is however worth noting that as of 2017 Nordhavn is one of the most expensive areas.

To look a bit further into the evolution of the housing prices Table 1 and Table 2 show the summary statistics across all zip codes for year 2010 and 2017 respectively.

Table 1: Summary statistics across zip codes and for year 2010

	Count	Mean	Std.	Min	Max
Purchase price	7,306	25.26	21.88	0.28	391.69
Rooms	7,306	3.44	1.49	1.0	12.0
m^2	7,306	100.75	44.58	12.0	568.0
DKK/m^2	7,306	24,799.24	22,259.17	436.0	470,566.0

Note: Purchase price is in 100,000 DKK
Source: Boliga.dk

Table 2: Summary statistics across zip codes and for year 2017

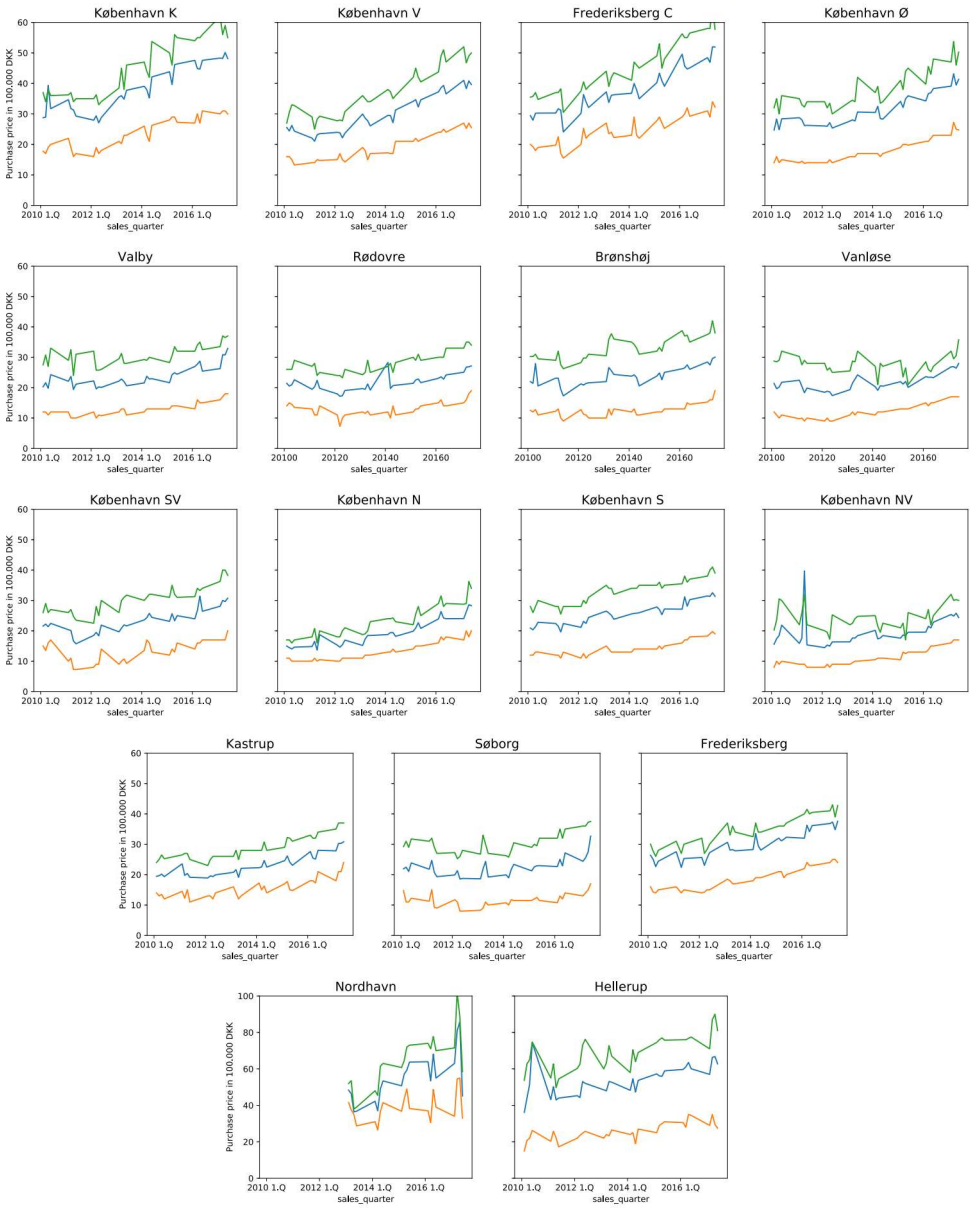
	Count	Mean	Std.	Min	Max
Purchase price	11,240	35.61	24.13	0.50	460.00
Rooms	11,240	3.20	1.44	1.0	20.0
m^2	11,240	92.72	41.92	12.0	563.0
DKK/m^2	11,240	37,779.06	14,804.15	253.0	625,862.0

Note: Purchase price is in 100,000 DKK
Source: Boliga.dk

Table 1 and Table 2 shows that the number of sales in 2017 was 11,240 compared to 7,306 in 2010, an increase of almost 4,000. This could be due to a greater demand for housing in the greater Copenhagen area, as mentioned in the introduction. Another explanation could be the fact that more housing is being constructed, such as the area of Nordhavn, making the supply greater. The average purchase price across all zip codes has gone up with around 1 million DKK from 2010 to 2017. The minimum and maximum purchase prices have gone up; the minimum has increased from 28,000 DKK to 50,000 DKK, while the maximum has increased from around 39 million DKK to 46 million DKK over the examined 7 years. This overall increase in price also shows in the price per square meter, which has gone up with around 13,000 DKK over the 7 years. This is due to the number of rooms and the size of the housing in square meters being roughly the same. The average housing size has fallen with around 7 square meters.

Figure 2 shows the development of purchase prices for each of the 17 zip codes included in our project.

Figure 2: Purchase price for the 17 zip code areas from 2010 to 2017



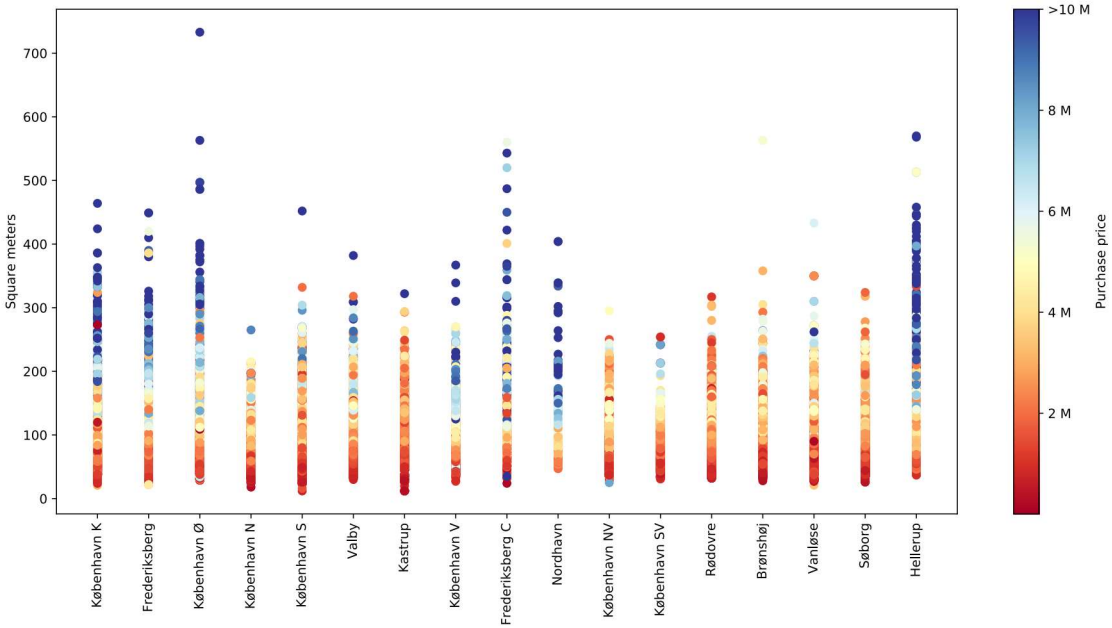
Note: The blue line shows the mean, the green line the 75th percentile and the orange line is the 25th percentile. The plots of Nordhavn and Hellerup have a different y-axis than the other plots.
Source: Boliga.dk

The figure above supports the increasing tendency we saw in Figure 1 and also shows that the purchase

prices display volatility. This volatility might be partially explained by seasonal effects and abnormally big sales in the form of one very expensive property. The development is shown for quarters instead of months to try and correct for some of these effects, but some of them still carry through in the plots.
In Figure 2 it can be seen that most of the areas in Copenhagen, Frederiksberg as well as Hellerup and Kastrup have seen an increase in purchase prices from 2010 to 2017. Especially København K and Frederiksberg C, the central areas of Copenhagen and Frederiksberg, have a clear increasing trend. Nordhavn seems to have an increasing trend from 2013 to 2017 as well. Here it is important to keep in mind that Nordhavn and Hellerup is plotted with a higher maximum value on the y-axis, as they have a higher overall price level compared to the other areas. Nordhavn has a mean of about 9 million DKK around mid 2017 before taking a dive towards 5 million DKK at the end of 2017. To compare both København K and Frederiksberg C has the highest mean of around 4,5 million DKK in the end of 2017. For some of the outer areas the trend is less clear. Valby, Rødovre, Brønshøj, Vanløse and Søborg seem to fluctuate a bit over the period but end with only a small or no increase compared to the mean in the first quarter of 2010.

Figure 3 shows the distribution of the property size in the different areas across the 7 years. The color bar to the right shows the purchase price of a given apartment.

Figure 3: Property size and purchase price in the greater Copenhagen area



Source: Boliga.dk

In Figure 3 it can be seen that some areas have a higher purchase price for a given number of square meters compared to other areas in the greater Copenhagen area. Comparing Hellerup to Søborg, two things are worth noticing. First Hellerup has a greater amount of large properties compared to Søborg. Secondly the overall price level is higher in Hellerup for a property with the same number of square meters. This can be seen as Hellerup has multiple properties with a high purchase price (marked with blue) starting from just under 200 square meters, whereas Søborg has none despite having properties with over 200 square meters. However the figure also shows that some areas have roughly the same price level for the same square meters e.g. Frederiksberg, København K, København Ø and Hellerup. When looking at this graph it is worth keeping in mind that multiple factors play a part in the price setting of a property and a direct comparison of price only based on number of square meters might not be representative. Thus the graph should be used only to obtain an overview over the distribution of prices and property sizes across the different neighborhoods.

As seen in Table 1 and 2 some of the properties sold in 2010 and 2017 were very cheap, with 28,000 and 50,000 DKK for the cheapest in 2010 and 2017 respectively. Similarly some were very expensive at 39.2 million and 46 million DKK in 2010 and 2017 respectively. In Figure 2 we see that the mean purchase price in some quarters jumps to a higher level e.g. Rødovre in 2013, Nordvest in 2011 and Nordhavn in 2017. This is indicative of some very large outliers that are able to sway the mean. To combat this we decide to remove some outliers. The upper and lower boundary for our inliers purchase price have been chosen by examining our data. This leads us to set a minimum purchase price of 100,000 DKK and a maximum purchase price of 25 million DKK. The minimum purchase price results in the removal of 32 observations and the maximum purchase price results in the removal of 71 observations. This brings our data set down to 75,345 observations.

From our descriptive analysis it is clear that both the zip code and sales year have an effect on the property prices. To account for this we include both dummies for the different zip codes and for the different sales years. We will dive into this in Section 6.

5 Machine Learning methods

In this section the different machine learning methods used to make our predictions will be introduced.

As we are interested in which machine learning method works best in relation to predicting property prices we look at different approaches. No algorithm is superior across all possible scenarios and

the performance is also highly dependent on the available data used. It is therefore interesting to see how the different algorithms perform on the same data, to see which predicts the most accurately.

The purpose of prediction models is to increase the precision of out-of-sample prediction. Our machine learning models can be seen as optimization problems, where we want to minimize the out-of-sample errors, which depends on the algorithm we want to use. It is by definition hard to measure how the out-of-sample errors perform. A way to overcome this is by splitting our observed sample into a training set and a test set (the holdout method, a simple form of cross validation). We use the training set to estimate the model and the test set to evaluate the models performance.

In the following subsections we will go through some linear regression models; first the classic OLS and then the L1 (Lasso) and L2 (Ridge) regularization methods. Then we go through LOOCV and K-fold cross validation which allows us to enhance our out-of-sample prediction accuracy.

5.1 OLS

OLS is constructed to minimize the squared errors in the sample and therefore performs well for in-sample predictions. This is done as follows:

$$\arg\min_{w} E[(y_0 - \hat{f}(x_0))^2].$$

We use w notation instead of β notation as this is a project about machine learning mostly. As most economists know, OLS is the best linear unbiased estimator given the Gauss-Markov assumptions, which among others include that we know the true form of the model. These unbiased estimates of the coefficients also allow us to analyze causation, which in many situations is useful. When the prediction object does not change, seldom does one receive an extra square meter, the need for unbiased estimates disappear.
However, OLS performs poorly when out-of-sample, as it is prone to overfitting, due to the fact that an unbiased estimator does not allow for bias. This then again leads to variance, due to the bias-variance tradeoff. As we are interested in out-of-sample properties, this is not optimal.

5.2 Regularization

A way to overcome the problem with overfitting, is to use regularization and introduce a penalty term to the cost function. This allows us to do polynomial expansion without overfitting. In our project we are looking at the following two methods using different penalty terms, Lasso and Ridge. When

regularizing, the penalty term is scaled with a hyperparameter, and how to pick this hyperparameter is covered in Section 5.2.3. It is also important to standardize the features such that they have variance 1 and mean 0. If not, then how our features are measured will influence which size the weights have, which again will have an influence on the penalty incurred from the penalty term.

5.2.1 Lasso

The penalty or regularization term when using the Lasso method is the absolute value of the weights and thus the minimization problem is as follows:

$$\arg\min_{w} E[(y_0 - \hat{f}(x_0))^2] + \lambda \sum_{j=1}^{p} |w_j|.$$

The Lasso method overcomes the problem with overfitting by punishing irrelevant weights w by setting them to zero, as taking the absolute value results in corner solutions. This means that in most cases you end up with a more simple model including fewer features, which is an attractive feature in model building.

5.2.2 Ridge

The penalty or regularization term when using the Ridge method is the weights squared and thus the minimization problem is as follows:

$$\arg\min_{w} E[(y_0 - \hat{f}(x_0))^2] + \lambda \sum_{j=1}^{p} w_j^2.$$

The Ridge regression makes the weights smaller, but does not reduce them to zero, as a quadratic penalty results in interior solutions. This means that the method does not get rid of irrelevant features but minimizes their impact on the trained model. The Ridge method is also faster to run, as it uses an analytic solution.

5.2.3 Hyperparameters

As mentioned earlier, the penalty term is scaled with a hyperparameter. Hyperparameters are not learned by training, but are values that are set before the learning process begins. In other words the model designers have to choose it. To specify the correct hyperparameter, one has to iterate through several and then choose the one that results in the most accurate model, without using the test data! To do this, the training data, hence known as development data, can be split into a new training and validation set, more about this in Section 5.3. This allows us to train and test the model with different hyperparameters

without using the test data. Thus the model designers can iterate through a list of hyperparameters and evaluate each hyperparameter's performance and select the best hyperparameter.

5.3 Cross Validation

As mentioned in the beginning of Section 5, we split the data into training and test data, which is a simple form of cross validation. In Section 5.2.3 the idea of splitting up the development data into a training and validation set is introduced. In this section we look at more advanced cross validation methods.

The problem with only one split of the development set is that the hyperparameter is sensitive to which sample is chosen when splitting the development data into training and validation data. One way of getting more information from the development set, and thereby reduce sensitivity to our chosen sample, is to do more advanced forms of cross validation. This allows us to reduce the hyperparameter's sensitivity to the chosen sample by rotating which parts of the development set are used for training and validation. The model is then trained and evaluated several times, and the average of the models accuracy is the chosen hyperparameter's accuracy.
For this project, we are evaluating our models using the MSE as the measure of accuracy. The MSE is calculated as follows:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{n}.$$

It is very important to note that we only use our test set when we are testing the model at the end - this is to ensure that the final test resembles an out-of-sample prediction problem.
The following two sections go into different cross validation methods.

5.3.1 Leave-one-out cross validation

We would like to use Leave-one-out cross validation (LOOCV) seeing as it is the most robust method, in regards to the hyperparameter's sensitivity to the chosen sample. This method uses each observation in the development set as a test set and then uses the rest of the development data to train. This process is repeated for every observation in the development set, and thus makes as many models as you have observations in the development set. The average of the MSE's from every step is the accuracy of the model, which reduces the sensitivity to the chosen sample. This is a very computing intensive option, and therefore we did not choose this option.

5.3.2 K-fold cross validation

A less computing intense option is K-fold cross validation. This cross validation method splits the development data into K even bins. Each bin is used as test data and the remaining bins are used for training, and this process is done K times, once for each bin. This allows all of the data to be used for testing and the remaining $(100 - \frac{100}{K})\%$ to be used for training. Again the average of the K MSE's is the accuracy of the model, and thus the sensitivity to the chosen sample has been reduced. It is easy to see that if the number of bins is equal to the number of observations, this is in fact equal to LOOCV. As this is less computing intensive, this is the method we chose.

6 Predicting

In this section we use the above mentioned methods to find the most accurate predictor for property purchase prices in the greater Copenhagen area. We calculate MSE and MAE and use them to evaluate our algorithms.

We chose to use the following features, which were available through *Boliga.dk*, for our prediction: *rooms*, m^2 and a total of 54 dummies. This included 6 dummies for sales, leaving 2010 as the reference year, 9 dummies for property type, leaving the category *other* as the reference and 16 dummies for the different zip code areas, leaving København K as the reference zip code and lastly dummies for the year the building were built. To avoid having dummies for every year possible, we grouped the building years. From 1900 and onwards the dummies covered a 10 year period. The 1800s were divided into two and everything built before 1800 were left out as the reference.
By having the sales year included as a dummy we try to capture the development in housing prices over the years and control for this in our model.
Having more features which would be relevant for housing prices could have been included. This is touched more upon in Section 7.2.

6.1 OLS

To do linear regression using OLS we have to assume a functional form. To do this, we draw on prior experience. It seems reasonable to assume that there is a non-linear relationship between purchase prices and square meters, such that the difference in purchase price between having 25 square meters and 30 square meters is bigger than the difference between having 100 square meters and 105 square meters. To accommodate this a quadratic term for square meters is included, as it is commonly done in similar

econometric models. Of course one could argue some interaction terms should be included, but as the model already contains many features adding more features to the model could cause or increase a problem with overfitting. Therefore interaction terms is not included.

6.2 Lasso & Ridge

One of the benefits of using regularized linear regression is that we do not need to know, or assume to know, the true functional form. Instead we can do polynomial expansion. We do polynomial expansion using Scikit Learn's PolynomialFeatures to the second degree, as third degree is too computing intensive. The polynomial expansion squares everything and interacts all features. We include a bias so that our models' biases is the base purchase price for our reference categories, that is other property type sold in 2010 in København K and built before 1800, given 0 rooms and 0 square meters. Polynomial expansion, with bias, transforms our 56 features to 1,653. This procedure is done both for Lasso and Ridge.
When developing our models we use 10-fold cross validation. For the Lasso this results in convergence warnings for the first 14 hyperparameters, which are the lowest. To ensure convergence we increase max iterations to 100,000 when fitting the whole development set when preparing to test.

6.2.1 Code specifics

This section quickly goes through the machine learning code, and the decisions we have made.

We have decided to do 10-fold cross validation, as we think this is a fair trade-off between robustness and computing intensiveness.

The first development-test split is 80%-20%, and then the development set is split into training and validation sets of 75%-25%. This results in a 60%-20%-20% split (train, validation, test).
For both Lasso and Ridge we iterate through a list of hyperparameters that consists of 50 samples in the range 10^{-4} to 10^{4}. For each hyperparameter we create a pipeline, and then we do 10-fold cross validation. The MSE for each fold is calculated and appended to a list, which after all 10 folds is then appended to a list, which creates a list of lists. We take the mean of each list, and the hyperparameter that results in the lowest average MSE is chosen as the optimal hyperparameter. The optimal hyperparameter for Lasso is $2.947 * 10^{-3}$ and for Ridge is 35.564.
Then the model is fitted using the **whole** development set, and the model predicts the test-set, which, again, is only used this once. It is noted that both our models converge.

For OLS we include a bias using Scikit Learn's PolynomialExpansion with degree equal to 1. We fit to the development set and then predict the test set. It is noted that we use a different data set that includes $(m^2)^2$, and therefore the test and training set for the OLS is not the same as the test and development set for the regularized linear regression.

6.3 Results

As mentioned, we are using MSE as a measure of accuracy. As the MSE has no obvious interpretation in a biased estimator, we also include the MAE. The smaller the number, the smaller the error, which can be interpreted as being a more precise prediction model. In Table 3 the MSE and MAE for our three models are presented.

Table 3: Results

	OLS	Lasso	Ridge
MSE	$1.603 \cdot 10^{21}$	100.862	100.254
MAE	$326 \cdot 10^{6}$	5.125	5.144

We see that our Ridge model gives the most precise prediction, as it has the lowest MSE with 100.254. Although it is only slightly better than the Lasso model, with a MSE of 100.862. It is noted that even though the Lasso is worse at predicting outliers than the Ridge, it has a slightly lower MAE. The OLS model performs significantly worse than the other two, which is probably due to very significant overfitting.

7 Discussion

In this section our results are discussed, further we include a discussion of our model and which other machine learning methods might be applied to this problem.

7.1 Our results

As mentioned in Section 1 we expected our Ridge model to be most precise, which we saw was the case when looking at the MSE. Trying to interpret the results, we saw that both Lasso and Ridge got a MAE of about 510,000 DKK. With a mean purchase price in 2017 of 3,561,000 the MAE's is quite substantial, arguably making our prediction models unfit for use.

This could be an indicator that predicting property purchase prices is a very complex task, which

we also see in the real world, an example being SKAT as briefly mentioned earlier. The difficulty of predicting can be attributed to the housing market being very heterogeneous, which also is the case with our data sample even with the elimination of significant outliers mentioned in both Section 3 and Section 4.

7.2 Improving the model

To improve the accuracy of the model, one possibility is to take more features into account, due to the many factors that we can imagine having an impact on the purchase price of a property, when using a hedonistic pricing model, every characteristic can be included. Some factors that might be included could be dummies that denote whether or not the property has a balcony, floor level (if an apartment), how many floors, how big a garden, how high the crime rate in the area is, how good the nearby schools are, how close the supermarket is, ocean-view and many more possible features to include in the model. The zip code could be used as a proxy indicator for some of the location specific features. It should be noted that many of these distance-features, and more specific location-features than zip codes, would require us to scrape the address, and turn it into GPS coordinates, which would raise privacy concerns. This may have resulted in a better prediction model, but also put the data-scraping in a more grey area ethically. This is because we would have to get street address information thus making it easier to identify people and their properties.

Our regression models only predicts purchase prices of properties that have already been sold, which means we have a whole year of data about the general price level that year. To be able to predict what a property would be sold for in the next quarter, we need to be able to predict the trend of the prices in the future, a problem that Borde et al. (2017) analyses with only near-future success. To really improve our models predicting powers, this would need to be improved upon.
Had we had a more powerful computer, we would also have liked to increase our search for the optimal hyperparameters; both with bigger initial searching spaces, perhaps 500 samples between 10^{-4} and 10^{4}, and iterate over a smaller range around the found optimal hyperparameters.

7.3 Other machine learning algortihms

Instead of using more features, maybe additional machine learning methods could have been applied. These could have been either Random Forest, which is an ensemble machine learning method that combines a set amount of decision trees, K-nearest neighbors, which predicts the target as the average target of the K-nearest neighbors, or another regularization penalty term that combines L1 and L2

regularization, the elastic net, which both penalizes exploding coefficients and selects features. Random Forest is of particular interest as both Borde et al. (2017) and Masías et al. (2016) finds that it gives the most precise predictions.

Ultimately we chose to use the Ridge and Lasso regularization as it is very similar to OLS and thus very intuitive for social scientists. Our choice is further supported by Hindman (2015), who argue that penalized regression techniques have many positive attributes and favors Lasso.

8 Conclusion

The aim of our project was to find the most precise prediction model for purchase prices in the greater Copenhagen area. In our descriptive analysis we saw that the housing market in the greater Copenhagen area has seen some relatively large changes over the last 7 years, with an increase in the mean purchase price of almost 1 million DKK. Further we found significant local differences between zip codes. Having an accurate prediction of the purchase, or sales, price of a property is of great interest to both house owners, investors and public institutions, making this project relevant.

When doing analysis in the social sciences, the tools from social data science are very useful. Web scraping and machine learning are both powerful tools, as they automate processes and therefore saves time. Furthermore machine learning does not require the right functional form, which can be a problem when using OLS as seen in this project.
It is however essential to keep ethical and legal concerns in mind when doing so.

Our analysis found that the Ridge model gave the most accurate predictions, but with a relatively high error. We thus concluded that our model is unfit for real world predictions in its current state and needs to be improved upon. This could be done by including additional features in the model or applying other machine learning methods. The reviewed literature finds that the Random Forest model gives the most precise prediction and an obvious extension to our project would be to include this model.

In conclusion we find great potential within the project, but there are several areas of the model which could be improved upon.

9 References

Raschka, S., Mirjalili, V. (2017). *Python machine learning. 2nd ed.* Packt Publishing Ltd.

Salganik, M. J. (2017). *Bit by Bit: Social Research in the Digital Age*. Princeton, NJ: Princeton University Press. Open review edition. Chapter 6.4.4

Borde, S., Rane, A., Shende, G., Shetty, S. (2017). *Real Estate Investment Advising Using Machine Learning*.

Masías, V.H., Valle, M.A., Crespo, F., Crespo, R., Vargas, A. and Laengle, S., *Property Valuation using Machine Learning Algorithms: A Study in a Metropolitan-Area of Chile*. In Selection at the AMSE Conferences-2016 (p. 97).

Practical machine learning: Ridge regression vs. Lasso. (2017). https://codingstartups.com/practical-machine-learning-ridge-regression-vs-lasso Visited August 29th 2018.

Hindman, M. (2015). *Building Better Models: Prediction, Replication, and Machine Learning in the Social Sciences*. Annals of the American Academy of Political and Social Science, 659, 48.

Soeller, G et al.(2016) *MapWatch: Detecting and Monitoring International Border Personalization on Online Maps*. WWW '16 Proceedings of the 25th International Conference on World Wide Web Pages 867-878, Chapter 3.2

Appendix

A.1 - Table of zip codes and corresponding area/city

Zip code :	Area / City
$1050 - 1549$:	København K
$1550 - 1799$:	København V
$1800 - 1999$:	Frederiksberg C
2000 :	Frederiksberg
2100 :	København Ø
2150 :	Nordhavn
2200 :	København N
2300 :	København S
2400 :	København NV
2450 :	København SV
2500 :	Valby
2610 :	Rødovre
2700 :	Brønshøj
2720 :	Vanløse
2770 :	Kastrup
2860 :	Søborg
2900 :	Hellerup

A.2 - Map of where the different areas are located