Introduction to Social Data Science

The Danish Housing market

Determining the fair price relative to other available houses

∼ **Summer School 2020** ∼

Contents

1 List of contributions

No. 148 : 2, 3.1, 6.1, 6.3, 6.6, 7, 8

No. 149: 3.3, 5.2, 4, 6.4, 6.7, 7, 8

No. 155: 3.2, 5.1, 6.2, 6.5, 7, 8

2 Introduction

Housing makes up an essential part of the aggregated danish economy as well as the individual households. For individuals housing is important because, for obvious reasons, a living place is necessary, but also because a lot of Danes own their home. As an asset housing differs radically from other assets as equities, cash etc. because it is illiqiud. In 2019 the total housing supply, this is only counting livable and holiday houses, in Denmark was 615,715 and of these were only 12.8 pct. traded (Statistics Denmark, EJEN88 & BOL101). Illiqiud assets, such as housing, are often difficult to price correctly, due to the fact not enough trades are completed, and are therefore priced inefficiently. In 2018 housing made up 44 percent of danish individuals assets and equity from housing was around one third of the total equity (Statistics Denmark, FORMUE1), this posses a problem because one would wish to determine the right price to ask before putting such an important asset on the market.

This paper will examine the possibility of creating an efficient model, based on normal characteristics of a listing and if it improves the predictions of model to interpret text-as-data. The relevance of this topic from a Social Data Science perspectives derives from the fact, that housing in general is normal and a part of every ones live, yet it varies in many different underlying parameters that are not always tangible like size and distance to the nearest grocery shop - for instance how does a description of a nice view or a lovely location affect the price.

We will through machine learning develop a model which will be able to predict the asking price for a house about to enter the market, in such a way that the price will be equal to the market price given the features of the particular house. This involves both quantitative data in terms of specific features on a given listing, but furthermore qualitative data derived from real estate agents description of the listing.

2.1 Litterature review

The application of ML in predicting house prices is commonly used. Whether its to predict general price trends, incorporating alternative data (Wu, L. & Brynjolfsson, 2009) or cross-sectional (Geron,A. 2017). But the vast majority of machine learning models, fitted to house data, uses house core data as size, number of bathrooms etc. We try to go beyond the scope of this basic approach by implementing text data in our model.

3 Data collection process

3.1 Collecing data at Boliga.dk

"Boliga.dk" contains valid information about all houses that are currently for sale. This includes price, price change, address, size and a link to the real estate company selling the house. Because of the static environment in the housing market, within a given period, we are able to consider the data as cross-sectional with respect to the price changes. In order to maximize the precision of the model and create room for dropping unspecified rows, there has been gathered data from every house on Boliga, which corresponds to approximately 52,000 houses. To scrape Boliga we tap into their own API, which makes it possible for us to make a JSON call for each house and transform this to a structured dataframe. But in order to utilize their

API, we needed to have the ID for every active house. We gathered the IDs by creating a function that ran through all search pages on Boliga and attained the specific IDs.

3.2 Collecting data from danskejernbaner.dk

To collect information about the distance between houses and the nearest public transportation railroad station we have added additional information from ”danskejernbaner.dk”. This website contains information about all currently and formerly active railroad stations in Denmark. With scraping we identified all currently operating station on the website and retrieved their respective location coordinates. With this set of coordiantes we cross checked the location in our house dataframe, using Geopy we are able to calculate the distance between every station for every house. We then found the smallest distance. Afterwards we separated the distances into four bins split by the distance and created a dummy for each house.

3.3 Collecting data from real estate brokers

In an attempt to further improve our model, we added text-as-data. We did this by collecting body text from the houses real estate page, by doing so we hope to find certain keywords that works as a proxy for key attributes for example sea view, close to nature etc. Key attributes we think can have an impact on the house price. To do so, we drew advantage from having the link to every house realtors page. An initial investigating of our dataset, showed that a total of 295 companies represents all house sales in Denmark. Creating a scraping function for the web page of every company, to get every house description, would be both time consuming and inefficient. But further analysis shows that out of the 295 companies, 36 realtors represents approximately 47,000 houses, 90 pct of the data set. These 36 real estate agents are primarily national franchises such as Home, Danbolig, Nybolig etc. For every web page, out of the 36 real estate agents, we analysed the body text. After gathering the text, we had to preprocess the text data in order to determine the key attributes. Because of the lack of well functioning POS-taggers as well as lemmatizers for the danish language, in the native NLTK python package, we need to utilize other open source packages packages (Nielsen, 2020). A lot of considerations went into identifying the correct packages, facing the trade off between finding a reasonable sized package while ensuring a high accuracy. For lemmatization we use the package called ”Lemmy”, a package trained on words authorized by ”Dansk Sprognævn”. We use this package in cooperation with ”spaCy”s POS-tagger package which provides a 99 pct accuracy for lemmatization (Lind). The POS-tagger from ”spaCy” is a pre-trained model with 94.13 pct accuracy on POS tagging(spaCy). The POS-tagger creates the ability to separate the nouns from the rest of the words. By separating we could analyze the specific attributes mentioned by the real estate agent. By running all our text data through the prepossessing we were are able to create a sorted list with the top 300 nouns. We then assigned each keyword, noun, to one of five different groups representing our understating of key attributes, the groups are nature, view, location, other and lastly a list of non-relevant words. This enables us to classify a total of 1,737,594 occurrences of nouns. By using these classifications, we were are able to find matches between keywords classifications and nouns in each house description and thereby assign grades to the houses within the 5 groups and add this to the model. The function created for scraping each realtors homepage were very extensive, and created a lot of issues. In order to reach the assignment deadline, we restricted the number of house descriptions collected to 35.000 houses, but because of our restriction to 36 realtors we only gained data from 27,000.

4 Data ethics

When scraping Boliga.dk we have to mention the concerns regarding data ethics. In the Terms and Conditions it is clearly stated that as a third party user, you are not allowed to:

> "*Benytte nogen former for robot, søgerobotter, scrapers eller andre automatiske midler til at få adgang til websites hostet af Boliga ApS og tjenester udbudt af Boliga ApS og indsamle indhold til kommerciel brug.*"
> - (Boliga Terms and Conditions)

Which translates to the following: "Use any robots, search robots, scrapers or other automatic tools to access websites hosted by Boliga Aps and services offered by Boliga Aps and collect data for commercial use." We are aware, that we have broken the Terms and Conditions by scraping the website. This scraping could therefore be considered stealing since Boliga.dk offers any third party users to buy the data. The data collected for this paper is of investigative matter and will not be distributed beyond the scope of this course and will therefore not be used for any commercial purposes. In addition to this we are not allowed to cause any extensive pressure on the infrastructure of the website - we have dealed with this by implementing the time.sleep() function.

Real estate brokers has house descriptions for the purpose of promoting a given estate and later sell it. Therefore we do not believe, that we face the same ethical issue as in the instance of Boliga. Since the text is publicly available, with the purpose of reaching as many potential buyers as possible, we don't believe this could be considered stealing and we have also implemented a time.sleep() function to avoid overloading the real estate brokers individual websites. The same goes for data collected through "danskejernbaner.dk" they don't have any Terms and Conditions available on their website and combined with not overloading their website we don't believe to disrupted or disturbed the website or its operators in any way.

5 Descriptive statistics

5.1 Quantitative data

The descriptive statistics has been produced on the data we have achieved by scraping Boliga.dk. We have removed listings only containing land and forced sales. This serves the purpose of an overview of the relevant listings. The full process of data can be seen in section 6.1:

As stated within table 1 the total count of houses is 47,456. Within the data we are dealing with multiple outliers. This is surely affected by the data collecting. However, they *will* be included, because they give are a picture of the big differences within the housing market across the nation.

The price of the houses vary a lot. The most expensive is set to a price of 85,000,000 DKK. The average listing is set to 2,403,305 DKK. In addition to this it must be emphasised that our data also contains values for summerhouses, which are typically cheaper compared to regular real estate.
When inspecting the size, there are once again a lot of variation - this is explained by the fact, that our data both contains sizes of great villas and small apartments this surely correlates with the variation in rooms.

	price	size	squaremeterPrice	rooms	daysForSale
count	47,456	47,456	47,456	47,456	47,456
mean	2,403,305	140.29	17,820.59	4.52	277.74
std	2,475,032	99.06	27,167.79	1.97	428.08
min	0.00	0.00	0.0	0.00	0.00
25%	1,045,000	95.00	8,200.75	3.00	50.00
50%	1,795,000	131.00	13,552.00	4.00	133.00
75%	2,995,000	172.00	22,616	5.00	335.00
max	85,000,000	9073.00	3,750,000	56.00	4981.00

Table 1: Housing data, source: Boliga.dk and own calculations

Looking at the square meter price we find a standard deviation of 27,167.79 and the highest price 3,750,000 DKK per square meter (estate registrered as 1 sqm).
The average house is for sale for a total number of 277 days. what can be seen from the table is that the time varies a lot. Looking at the lower quartile we see that 25 pct. of the listed houses are for sale for less than 50 days. Also it must be said, that the average value of this particular variable is quite far from the median, which means that it is influenced by outliers.

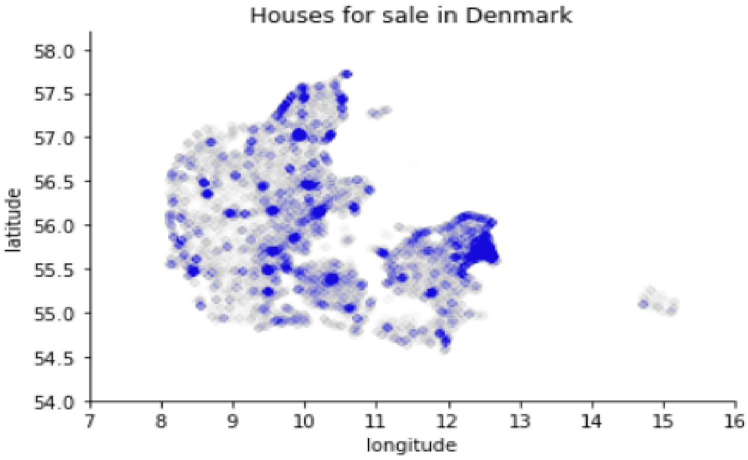


Figure 1: Geographical map of houses for sale in Denmark

In the listings some tendencies are obvious. Most of the listings are placed in the big cities like Copenhagen, Odense, Aarhus and Aalborg, cf. figure 1. This is naturally affected by the fact that there are many apartments in these areas and the population density is higher compared to more remote areas. Another plausible, but less probable, explanation could be that people are moving away from cities to more rural areas. When examining the geographical price patterns closer, cf. figure 2, it becomes clear that North Zealand and especially the area immediately north of Copenhagen has a brighter coloring which indicates the prices are higher. If we zoom further in around Copenhagen we see the higher priced houses even clearer.

Besides the fact, that the concentration of high price properties, it is also worth mentioning, that there is a clear pattern regarding the placement, since the most of the properties are placed close to the sea. When

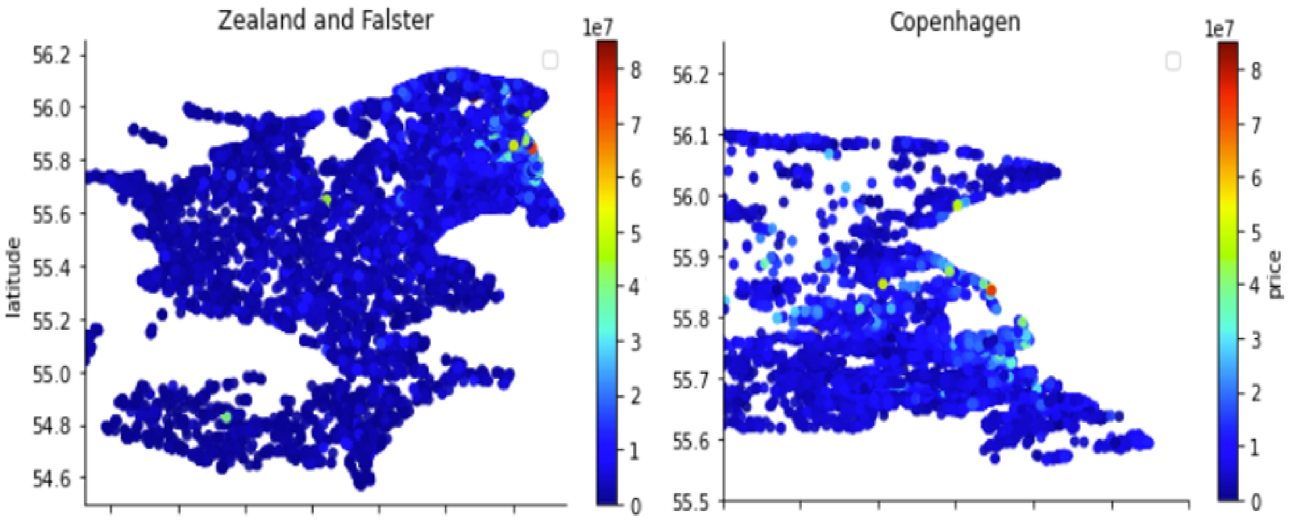


Figure 2: Mapping of houses for sale, with prices, by latitude and longtitude

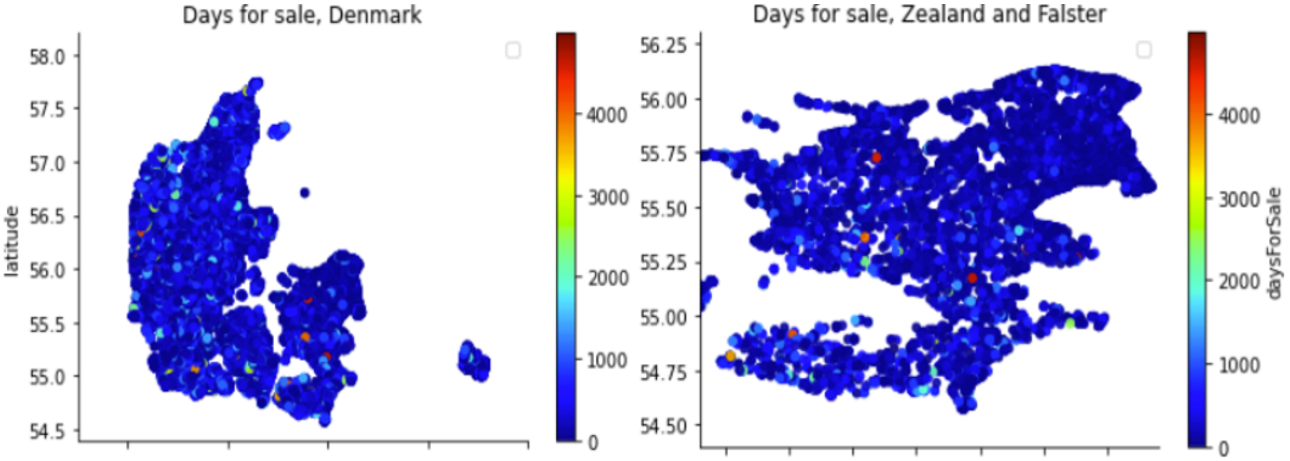


Figure 3: Mapping of houses for sale, with days for sale, by latitude and longtitude

looking at figure 4, we see that views ("udsigt") are one of the most used words.

Another takeaway from the geographical overview is the variance in the number of days a unit has been on the market. This is represented in figure 3, here one can see that especially Copenhagen and North Zealand is very dark blue, which is equivalent to the number of days a house has been for sale is low, while areas in Jutland and South Zealand is brighter or even red, which indicates a house has been available on the market for an extended period. For Zealand the picture is even clearer on the picture, where there has been zoomed in. The further away from Copenhagen you come the longer time a house has been on the market.

5.2 Text data

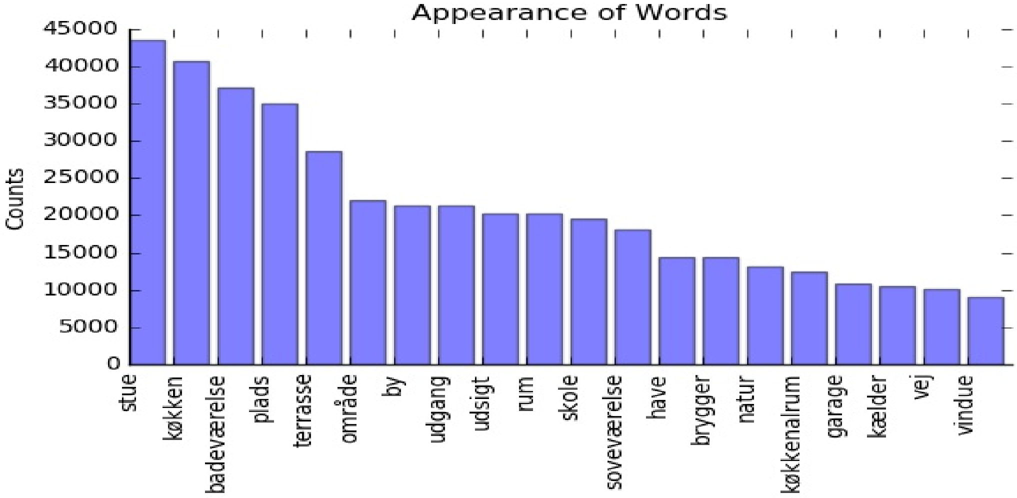


Figure 4: The 20 most common Nouns in the text

Figure 4 shows the most frequent used words in house descriptions except most common words like house, flat etc. The graph clearly shows a tendency for the nouns to highlight some main attributions like the area, the view or close to a school even though the content of the sentence, where the nouns is used, is not clear. It is also clear from the graph, that within the house, the real estate brokers tend to highlight the same rooms or areas. The concept of using nouns as a proxy for qualitative features is partly confirmed by the figure. Another interesting feature to look at, is the most commonly used words for different price ranges. This is shown in figure 5.

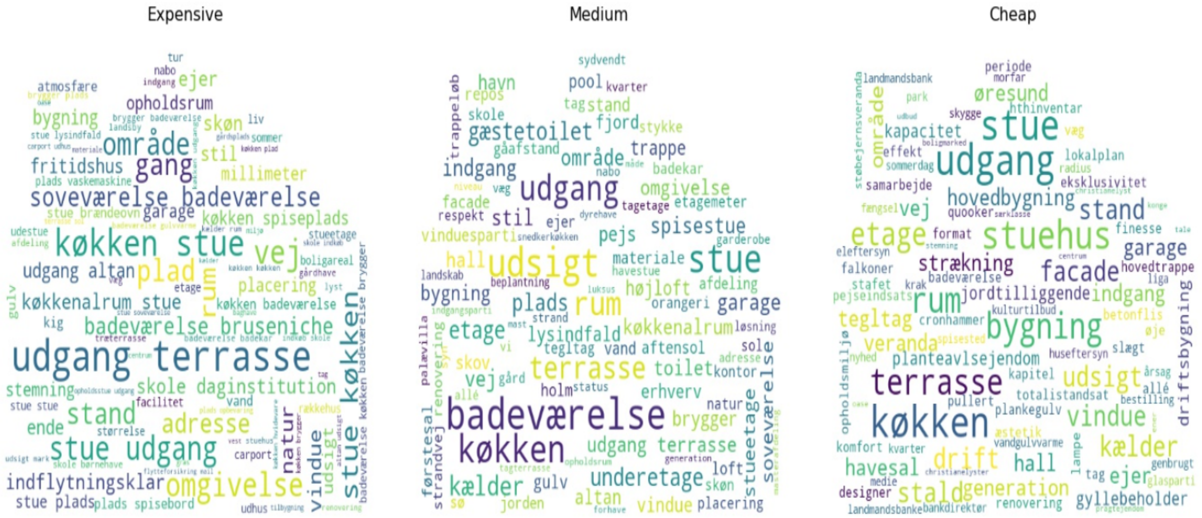


Figure 5: Word clouds by price category

Figure 5 shows the 100 most represented words for 3 price ranges; Expensive, Medium and Cheap. The wordcloud shows how the nouns differentiate between the different price ranges. In the expensive price range, nouns like beach, exit, flat, yard etc. is represented. Words typically associated with more high class houses or flats. Where as more basic nouns like living room, kitchen is more represented in the cheaper houses. The size of the words represents the frequency of the word. This indicates that the word usage could be possible to use within our model. An interesting point is the representation of rural nouns in the cheap houses, words like stable, slurry tank and farmer bank indicating that the geographic price difference is represented in the words. This is also shown in figure 2, where the most expensive houses are placed at the water or nearby.

6 Theory and Methodology

In the following section we will describe our usage of Machine Learning and elaborate on the theoretical foundation laying behind the processes of certain elements of Machine Learning.

We will be using supervised Machine Learning with the purpose of creating a regression model, because the target values takes continuous values. To answer our research questions we are developing the model with the purpose of predicting the housing price, given the current market situation and features of the house.

The process behind creating a Machine Learning model consists of four different steps, 1) Preproccesing, 2) Learning, 3) Evaluation and 4) Prediction. Preprocessing, the first step, has already partly been described in the section regarding data collection, the information in section 6.1 is therefore more about cleaning the data. Section 6.2 through 6.4 is concerning the regressions, which is part of the learning step. While section 6.5 describes how to evaluate and validate the different models. The last section will describe the models predictions.

The Root Mean Squared Error (RMSE) is the most common used tool the evaluate how a model is performing. RMSE shows the standard deviation of the errors. The mathematical definition of RMSE is the following:

$$RMSE(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h(\mathbf{x}^{(i)}) - y^{(i)}\right)^2} = \sqrt{MSE}$$

RMSE is a useful quantitative measure for evaluating models, because it is easy to compare between different regressions and it is easy to use in regards to tuning via grid search or cross-validation because it normalizes.

6.1 Preprocessing

Regarding the raw data the process is mainly described within the section "Data collection". In addition, we added a municipality dummy to each house, floor dummy and house type dummy adding a total of 117 dummies. To further preprocess the data, we are using sklearn.preprocessing train-test-split This function splits the data randomly into a training part consisting 2/3 of the entire data and a test part consisting of the remaining 1/3, we are able to use the uneven distribution due to our large dataset. This enables a out-of-sample validation of the final model. Our model will then use the training part to find patterns within the data - afterwards this will be tested on the remaining part to see how well it actually fits the data. After creating several dummies we also needed to transform our non-dummy data. We did so by creating

PolynomialFeatures with 3 degrees and StandardScaling. The data processing are shown below in figure 6.

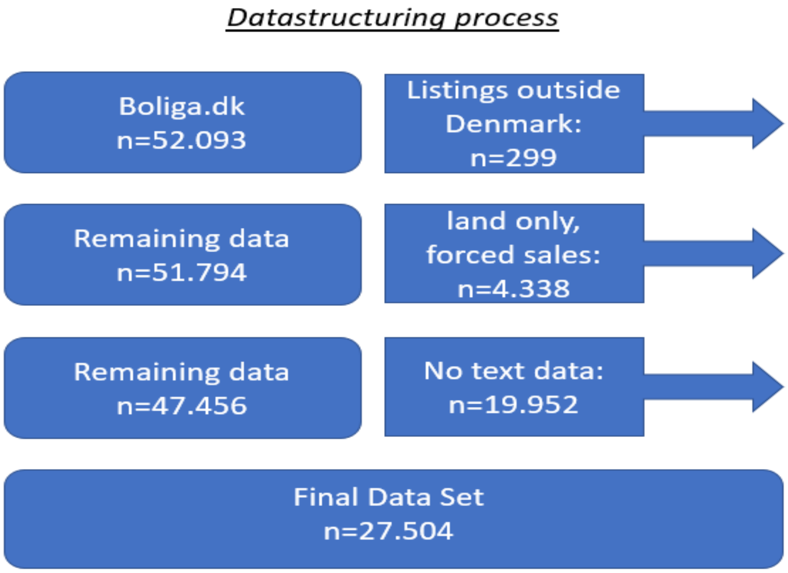


Figure 6: Datastructirong process decomposed

The removal of data points was caused by several reasons. First and foremost we removed all houses not in Denmark, this removed 300 houses. Secondly we removed all listing with no house just soil and properties with ”Propertytype” 9 and 10. These properties are forced sales, and are not included in our model since they are listed on other terms than the regular listing. This removed around 4.338 listings. Thirdly we removed all NA values. Lastly our added body text of each house demanded a removal of 19,952 listings. Leaving us with a final data set of 27,504 listings.

6.2 Linear regression

We will be using multiple linear regression since we are trying to model multiple features. All linear models takes the following form

$$\hat{y} = \theta^T \cdot \mathbf{x}$$

Where θ is a matrix of the parameters containing a bias term θ_0 and the feature weights θ_1 through θ_n and x is a matrix of the features.

The optimal solution is choosen so the parameters will minimize the cost function. Which is the average of the squared errors. An error is measured by the difference between the actual and predicted output.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(\underbrace{y^{(i)} - \hat{y}^{(i)}}_{\text{Error term}} \right)^2$$

6.3 Lasso regression

A Lasso regression is a linear regression but it differs from a standard linear regression due to its L1 regularization. L1 regularization is characterized by its ability to penalize the sum of absolute weight coefficient. Because it uses the absolute weights it is more common for the optimal solution to have some weights being 0. In other words, Lasso regressions favours sparsity and is well equipped to be used in regressions with many features, such as our dataset. It is also favourable in Machine Learning because it minimizes number of features and removes those irrelevant. The following cost function consists of the linear regression cost function and the penalty term from L1 regularization, the optimal solution will minimize this cost function (Raschka & Mirjalili, p. 482).

$$J(\theta) = \text{MSE}(\theta) + \alpha \sum_{i=1}^{n} |\theta_i|$$

From the definition it becomes clear that in the case of $\alpha = 0$ the cost function for the Lasso regression will be equivalent to that of an ordinary linear regression and the results will not differ between the two estimations.

6.4 Random Forest Regression

A Random Forest model is a non-linear regression model. The model can be understood as multiple decision trees put together in one model. An advantage regarding Random Forest is the need to choose good hyperparameters isn't as big compared to other models due to its robustness. The only important choice is the number of decision trees. The idea behind the model is that, when one decision tree turns out to have a high variance, it is possible to generate a more general model by scaling the number of trees, and thereby creating a model which is less exposed to overfitting. (Raschka & Mirjalli, p. 170-171). When using Sklearn this is done by averaging the outcomes of the individual decision trees (Random Forest documentation, sklearn). The model aims towards minimizing mean squared error as stated above.

The model works by letting the algorithm run from the root through the terminal notes. With our data set the root might for instance be apartment type. This generates an output to which patterns within the data will be recognisable. A great feature within both the decision tree model and Random Forest is that we do not need to worry about scaling the parameters because the models are scale invariant. However, because the model consists of multiple decision trees running the model can be quite demanding in terms of computational power. Furthermore there is no need to reprocess the data by making interaction with polynomials because Random Forest is able to derive non-linear relations on its own.

6.5 Hyperparameters

In order to tune our models to the specific data set we tune our hyperparameters. Hyperparameters are not learned from the data set but represents the knobs of a model that we can turn to maximise performance(Raschka & Mirjalili, p. 56). There are multiple build-in functions that can help determine the optimal hyperparameter, this includes GridSearchCV and RandomSearchCV. In our assignment we use GridSearchCV which uses brute force to validate the optimal parameter. GridSearch uses cross-validation, explained in later sections, to ensure that the hyperparamter is not overfit to the training data (Geron, p. 30). In our models the GridsearchCV shows the following results regarding optimizing. Notice, that the optimal number of estimators decline from 58 to 51 when including text into the model, also the paramter of alpha is quite small minimizing the penalty as mentioned in section 6.3.

Random Forest		
	Model without text-as-data	Model with text-as-data
Number of estimatores	58	51
Max features	6	6
Lasso Regression		
	Model without text-as-data	Model with text-as-data
Alpha	0.00053	0.0001

Table 2: Optimizing hyperparameters and features in models

6.6 Cross validation

To validate our models we use the cross-val-score function. This function makes use of the K-fold method. The K-fold method has its mathematical definition as below, where K-fold is set to 10.

$$Train = 100 - \frac{100}{k}, \text{ where } k = 10$$

It should be noted, that since our main data set are approximately 47.000 observations, we have chosen to make the K-fold equal to 10, which means that 4.700 observations will be used for testing and the rest for training each iteration. The K-fold works as follows: In the first iteration it splits the traning data and test data into 90 pct. training data and 10 pct. test data. Next iteration it will still split up the data by 90 pct. training data and 10 pct. test data, however this iteration will use another 10 pct. as test data.

The advantage of using this method is that we are testing and training the data 10 times, which gives us more precise results regarding RMSE and does therefore advise in greater detail on which model performs the best measured by RMSE. We have used the K-fold method on all of the regressors. We could in theory have chosen to validate our model by using LOOCV, however a unwanted feature with the use of this model is that it is extremely computing intensive. Furthermore, we use GridSearchCV to find the optimal number of estimators in our RandomForest model.

6.7 Predictions

The models yielded the following results.

	Models without text-as-data			Models with text-as-data		
	Linear	Lasso	Random Forrest	Linear	Lasso	Random Forrest
RMSE	2.71	2.18	1.52	3.18	3.12	1.71
Std. dev.	1.80	1.13	0.77	3.49	3.48	0.76

Table 3: Cross validation of different models - scaled y

The results from our models are as presented in table 2. Within the model without text-as-data we see that Random Forest consisting of 58 estimators has the lowest RMSE of 1.52 and in addition to this the lowest standard deviation of 0.77. This means that this is the preferred model. The result is not unexpected, since our data set consist of many parameters. When applying the optimal hyperparameters for the Lasso regressor, we see that it is preferred to the linear regression model. But as described in the theory section regarding Lasso regression a low hyperparameter is indicative of the Lasso regression being close to the linear regression, which is the case. This means L1 regularization may not be necessary, and could be interpreted as a sign that overfitting is not a particular issue in the linear model.

When looking at the models including text-as-data we see that the Random Forest model yield a RMSE of 1.71 and a std. dev. of 0.76. Another important take-away is that none of the models performs better with text-as-data compared to the models excluding text-as-data. One possible explanation for this will be discussed further in section 7.

7 Discussion and Analysis

When using the housing prices from Boliga we get the price set by the real estate agent and not the selling prices. The housing market is not efficient and therefore our predictions will give no indication of a fair or fundamental price. It could be a reasonable presumption that houses within the medium range price range is sold at approximately same price. But the outliers is expected to have a sizeable difference between the offered price and sales price.

We choose to maximize our number of observations, thereby incorporating houses from the whole country. This has inevitably created unnecessary noise in our model. If we restricted the model to certain parts of the country, that being Copenhagen, Bornholm, Sourthen of Jutland the predictions could have yielded a more robust model. We have reason to believe so, because a geographically smaller model will focus on specific areas, where we assume that the parameters in general are much more alike compared to our model. If the estate is located within a given municipality, for example Copenhagen, our model does not differ between Østerbro or Amager, which probably has a significant impact upon the price. This can clearly be seen in our descriptive statistics section, where price and days for sale differ on a local plan. However, the purpose of our assignment were to create a nationwide usable model, that could determine the fair price for any given house, and we chose therefore to incorporate all houses.

Performing analysis within a non-native language such as danish creates a few obstacles and demands specific restrictions to the use of text processing packages. There does exist a lot of open source well developed packages for POS tagging and lemmatization(Nielsen, 2020), but the two leading are nlpstanford and spaCy (in cooperation with lemmy). To determine which model to choose we looked at a random sample of ten houses and compared the POS-tag and lemmatization result. Here the spaCy/lemmy combination proved to be more accurate. When gathering text data for the house we enter a trade off between fewer houses and more information about each house. We chose to prioritize more information about each house, because we believe that the key attributes has a significant effect on the houses. Furthermore, by doing a text analysis of the real estate agents description of the houses, using this as a proxy for the house key attributes we assume that all estate agents are equally good at highlighting the pros. This is a rather harsh assumption, in a competitive market like real estate there will most likely be agents better at mentioning the primary pros of houses. Our key focus on the nouns and not the semantic construction of the house description, is because of a presumption that all real estate agents talk positive about their houses. If this is the case, a sentiment analysis would prove no use. But if this was not the case, a further analysis of the sentiment construction could have added further functionality to the model.

Further it might be meaningful to discuss our choice of models reflected in the prediction section. The fact, that we are applying this much data to our models comes the downside of smaller ability to try out different regressors. The linear regression was chosen, with the purpose of showing, that it would not be suitable as a valid model, with this many parameters. When testing Lasso, we expected it to perform better than the linear regression, when applied with optimal value of hyperparameter. As mentioned above Random Forest outperformed both models with and without text-as-data.

An interesting result appeared when we introduced text data in the models. First and foremost all models performed worse. One possible reason for this is a methodical we made. We only collected text data regarding around 27 thousand houses, cf. figure 6, due to increasing complexity because of more real estate brokers. We saw this as a trade-off between more observations with fewer datapoints, the models without text data, and fewer observations with more datapoints, the models with text data. Because of this choice the models without text-as-data were tested upon 47 thousand observations and the models with text-as-data were tested upon 27 thousand observations. This significantly decreases the number of observations available for training the models which can explain the decrease in performance. Another approach could have been to decrease the number of observations for the models without text-as-data but then these models might have performed worse.

If we had more time we would have tried to model the text data further and collect text data for even more houses to compare the models. We could also have build another grading system for the nouns, this could have been done by either expanding the number of different categories or attributing more points to specific groups of nouns. Another field which we could work further on is the choice of machine learning models. In the section below some of these models will be discussed.

7.1 Other potential models

A Ridge regression is also a linear model and defined mathematically as below:

$$J(\theta) = \text{MSE}(\theta) + \alpha \sum_{i=1}^{n} \theta_i^2$$

The difference between Lasso and Ridge is found within the regularization. Lasso uses L1 regularization, where Ridge uses L2 regularization. The model therefore introduces a greater and squared penalty. This might affect the model by setting even further parameters equal to zero. This model is therefore considered relevant and might as well has been used (Bhattacharyya, 2018). L2 regularization and Ridge regressions are great at dealing with exploding coefficients, while L1 regularization and Lasso regressions are great at dealing with too many irrelevant features. Since our dataset consists of more than 100 dummies we were more concerned about irrelevant features than exploding coefficients. To investigate this matter further we only employed Lasso, instead of a ElasticNet, which could be interesting to train in further investigation.

Another possible model is a Kernel regression. This regression is build upon the assumption that the data is distributed by Gaussian as shown below:

$$K(x) = \frac{1}{h\sqrt{2\pi}} \exp\left(\frac{-1}{2}\left(\frac{x - x_i}{h}\right)\right)^2$$

Where h is the bandwidth of the parameters. For using the regressor the data has to fulfill the criteria of being normally distributed. The regressor uses the distribution for the purpose of giving each parameter a weight. This might be usefull because the model therefore focus on the most observed values. When looking at our descriptive statistics i section 5, we get the impression, that there might be a high degree of curtosis. This indicates our data is not symmetrical. Because of this we believe, that this estimator would not perform well despite it's great features.

8 Conclusion

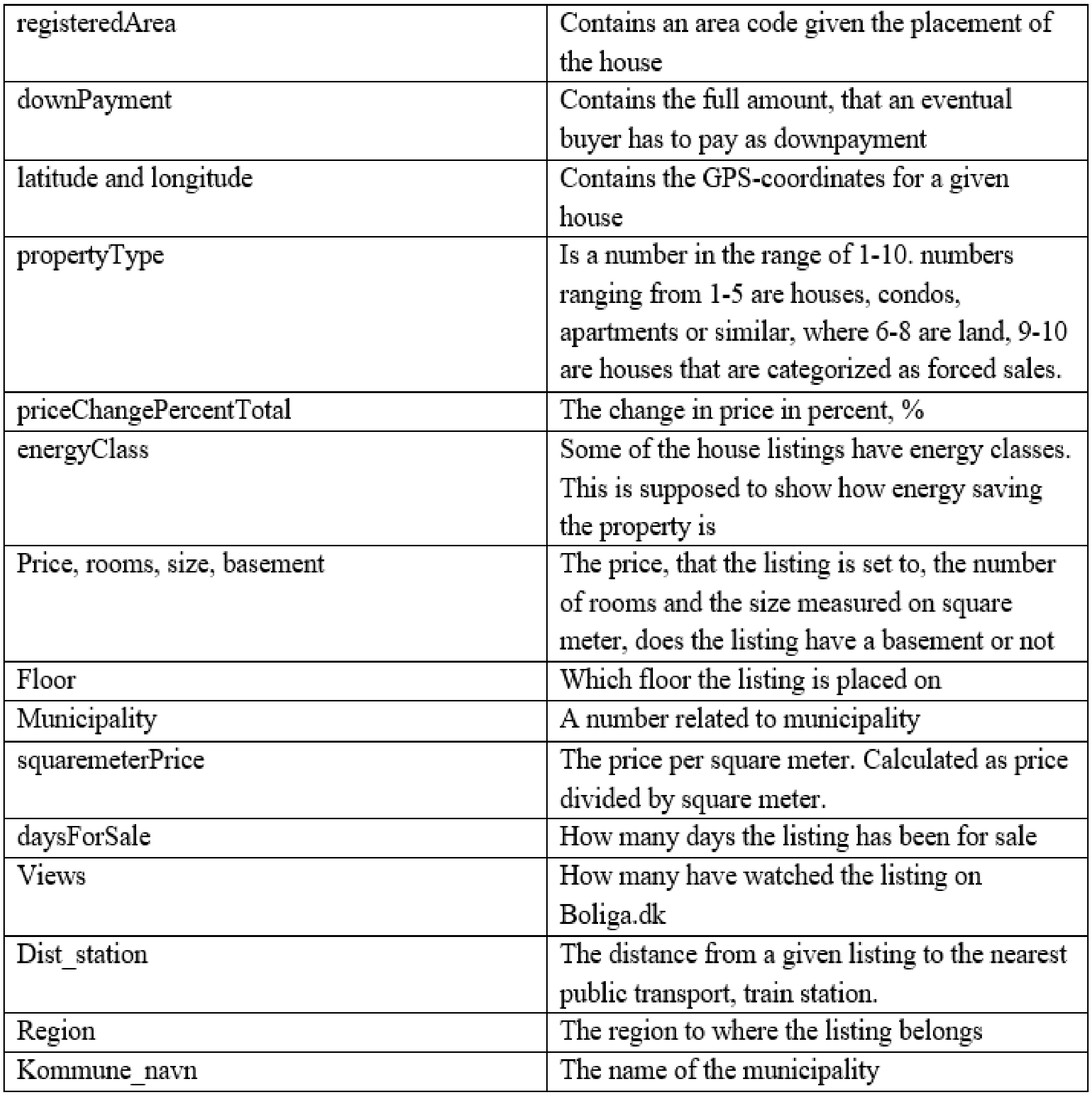
We can conclude, that we have managed to retrieve, process and handled huge amounts on data. From our descriptive analysis we see that there tend to be a clear pattern between words used in sales texts and the price of the given listing (figure 5). The machine learning model, that we prefer overall are the Random Forest estimator, which has the lowest standard deviation and RMSE compared to Linear regression and Lasso when looking at the model excluding text-as-data. When incorporating text data into the machine learning models we find that they in general perform worse, however Random Forest performs much better compared to Linear Regression and Lasso.

As mentioned in the discussion this might be caused by the fact that we use less data, but it might as well be caused by the fact that the text is hard to process and furthermore, that it is difficult to derive the true meaning of a text used for sales purposes especially in danish. The conclusion might therefore be, that by using our methods, including text as data does not improve the precision and should therefore not be used. This subject also calls for further investigation with possibilities with more detailed text data or more complex supervised/unsupervised Machine Learning models. We believe that we have created a Random Forest model, without/with text-as-data, which is capable of predicting a asking price for a house in line with others available houses on the basis of characteristics such as rooms, size, location, distance to public transportation etc.

9 Literature and References

- Raschka, S. & Mirjalili, V. (2017): "*Python Machine Learning*", 2nd edition. Birmingham: Packt Publishing.
- Geron, A. (2017): "*Hands on Machine Learning with SciKit-Learn & TensorFlow*". CA: O'Reilly Media
- McKinney, W. (2012): "*Python for Data Analysis*". CA: O'Reilly Media.
- Boliga.dk (2020). URL: Housing data, Terms and Conditions
- Statistics Denmark data. URLs: FORMUE 1, EJEN88 & BOL101
- Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60
- Nielsen, F. (2020): "*Danish resources*", 2020. URL: Package for POS and lemmatization
- Documentation regarding sklearn. URL: Documentation for Random Forest
- Danish lemmatizer, Lemmy, developed by Søren Lind. URL: Github repository
- Danish multi-task CNN trained model. URL: Spacy.io
- Nagpal, Anuja. 2017. L1 and L2 Regularization Methods. URL: Theory Lasso, Ridge
- SODAS2020 Group 2 Github. URL: Github Link
- Wu, L. and Brynjolfsson, E. (2009). The future of prediction: How google searches foreshadow housing prices and sales.
- Bhattacharyya,Saptashwa. 2018. Ridge and Lasso Regression: L1 and L2 Regularization. URL: Ridge and Lasso regressions in scikit learn

10 Appendix



registeredArea	Contains an area code given the placement of the house
downPayment	Contains the full amount, that an eventual buyer has to pay as downpayment
latitude and longitude	Contains the GPS-coordinates for a given house
propertyType	Is a number in the range of 1-10. numbers ranging from 1-5 are houses, condos, apartments or similar, where 6-8 are land, 9-10 are houses that are categorized as forced sales.
priceChangePercentTotal	The change in price in percent, %
energyClass	Some of the house listings have energy classes. This is supposed to show how energy saving the property is
Price, rooms, size, basement	The price, that the listing is set to, the number of rooms and the size measured on square meter, does the listing have a basement or not
Floor	Which floor the listing is placed on
Municipality	A number related to municipality
squaremeterPrice	The price per square meter. Calculated as price divided by square meter.
daysForSale	How many days the listing has been for sale
Views	How many have watched the listing on Boliga.dk
Dist_station	The distance from a given listing to the nearest public transport, train station.
Region	The region to where the listing belongs
Kommune_navn	The name of the municipality

Figure 7: Variables regarding the data