NTNU
Norwegian University of Science and Technology

Developing and Evaluating an Automated Valuation Model for Residential Real Estate in Oslo

Kristoffer Birkeland
Allan Daniel D'Silva

Industrial Economics and Technology Management
Submission date: May 2018
Supervisor: Are Oust, IØT

Norwegian University of Science and Technology
Department of Industrial Economics and Technology Management

Problem Description

The residential real estate market is an integral part of any nation's economy. In Norway, a strong tradition of home-ownership exists, with as many as 84% of the population living in a self-owned home. Furthermore, during the past few decades, the price of real estate in Norway's capital Oslo has experienced strong growth. These factors have established a keen interest for the understanding of real estate prices among private individuals, commercial actors and policymakers. Moreover, the housing market is characterised by substantial complexity, illiquidity and transaction costs, making it a far from frictionless market. In total, this motivates the research of residential real estate prices.

In this study, we aim to develop an automated valuation model (AVM) to estimate the selling price of individual dwellings. Our approach is largely novel in the field of real estate finance and combines concepts from data science communities with techniques from traditional real estate research. Specifically, we leverage the concepts of *stacked generalisation* and *comparable market analysis*. Stacked generalisation, or stacking, is a machine learning technique where a meta-estimator is trained to combine the predictions of underlying submodels. We implement five different submodels; four of these are ensemble learning methods, while the fifth model is based on a repeat sales index. We apply the concept of comparable market analysis by selecting a set of comparable previous transactions tailored for each value estimation.

The AVM produces a value estimate for a dwelling by analysing data describing comparable dwellings' characteristics and previous transactions. To assess the model, we implement it for the residential real estate market in Oslo and evaluate its performance out-of-sample on all transactions in Oslo in the first quarter of 2018.

Preface

This thesis concludes our Master of Science degree in Industrial Economics and Technology Management at the Norwegian University of Science and Technology (NTNU). It is original and independent work performed by Kristoffer B. Birkeland and Allan D. D'Silva, written during the spring of 2018.

We would like to thank our supervisor, Are Oust, Associate Professor at NTNU Business School, for his valuable advice and guidance. His interest in our work has been beneficial and motivating during the process of writing this thesis. Moreover, this study is developed in close collaboration with industry experts. We thank Henrik Langeland, CEO of Alva Technologies, for his invaluable input and his unrestricted availability for questions during this process.

ABSTRACT

In this thesis, we develop an automated valuation model (AVM) for the residential real estate market, and evaluate it by analysing its accuracy on all dwellings sold in the first quarter of 2018 in Oslo, Norway. We design the AVM to utilise data on dwellings' transactions and characteristics by leveraging the concepts of stacked generalisation and comparable market analysis. In particular, we combine four novel ensemble learning methods with a repeat sales method and tailor the data selection for each value estimate. By developing our AVM, we aim to aid in creating a more efficient real estate market and to further the research of applying machine learning to real estate finance.

We calibrate and evaluate the model for the residential real estate market in Oslo, by producing out-of-sample value estimates for the 1 979 dwellings sold in Q1 2018. This yields highly promising results; the AVM achieves a median absolute percentage error (MdAPE) of 5.4%, and more than 96% of the dwellings are estimated within 20% of their actual selling price. This is comparable to the accuracy of local estate agents in Oslo, and we observe similar performances by the industry leader for AVMs in the U.S.

These results demonstrate that our AVM is a viable tool for both industrial and private applications. Furthermore, we find our novel approach of applying stacked generalisation to residential real estate valuation to be successful and encourages the research into the application of machine learning to the field of real estate finance.

SAMMENDRAG

I denne oppgaven utvikler vi en automatisert verdsettelsesmodell (AVM) for boligmarkedet, og evaluerer den ved å analysere presisjonen på alle boliger solgt i første kvartal 2018 i Oslo. Vi utformer AVMen slik at den utnytter data om boligers transaksjoner og egenskaper ved å benytte konseptene "stacked generalisation" og sammenlignbar markedsanalyse. Modellen kombinerer fire "ensemble"-læringsmetoder med en "repeat sales"-metode og skreddersyr dataseleksjonen for hvert estimat. Ved å utvikle vår AVM tar vi sikte på å bidra til å skape et mer effektivt eiendomsmarked, og å fremme forskningen innen anvendelse av maskinlæring i boligøkonometri.

Vi kalibrerer og tester modellen for boligmarkedet i Oslo ved å produsere estimater for de 1 979 boligene som ble solgt i første kvartal 2018. Dette gir svært lovende resultater. AVMen oppnår en median absolutt prosentfeil på 5,4%, og mer enn 96% av boligene er estimert innenfor 20% av den faktiske salgsprisen. Dette er sammenlignbart med presisjonen til eiendomsmeglere i Oslo, og vi observerer lignende resultater hos markedslederen for automatiserte verdsettelsesmodeller i USA, Zillow.

Disse resultatene viser at vår automatiserte verdsettelsesmodell er et konkurransedyktig verktøy for både industrielle og private anvendelser. Videre finner vi vår tilnærming ved å anvende "stacked generalisation" til verdsettelse av boliger å være lovende og anbefaler videre forskning innenfor anvendelsen av maskinlæring til boligøkonometri.

Contents

List of Figures

List of Tables

Acronyms

ANN artificial neural network

AVM automated valuation model

BP bagging predictor

ET extra trees

HPM hedonic pricing method

MAE mean absolute error

MAPE mean absolute percentage error

MdAPE median absolute percentage error

NOK Norwegian krone

OLS ordinary least squares

PPSM price per square meter

RF random forest

RSM repeat sales method

USM usable square meters

WLS weighted least squares

XGB XGBoost

1 Introduction

The decision of purchasing or selling a dwelling is typically the most significant financial decision in a person's life (Smith & Tan, 2015). It is both a financial investment, as it yields potential capital gains or losses, and a consumption decision, as it influences life quality. Furthermore, the residential real estate market is a vital part of a nation's economy, affecting banks, policymakers and investors, in addition to homeowners. It is a market characterised by significant transaction costs, low liquidity and high information asymmetry between its professional actors and private individuals (Ibbotson & Siegel, 1984; Levin, 2001). Also, the residential real estate market has hardly been impacted by digitalisation and automation, compared to other markets (Sittler, 2017).

A neutral automated value estimator can benefit several actors in the residential real estate market. First, a publicly available service that provides such estimates can benefit private individuals, both home buyers and sellers. When applying for a loan, searching for a new place to live or when considering to sell a residence, these estimates can simplify the process. Second, we believe that automated value estimates also can be of great importance to professional actors like banks, asset managers or policymakers. Situations where automated value estimates could be beneficial are when diversifying loans, considering new asset placements or calculating discount rates.[1]

This paper aims to develop an automated valuation model (AVM), which is an automated tool that produces real estate property valuations based on statistical modelling. By doing so, we seek to create a more efficient real estate market. The AVM should be capable of leveraging both historical transaction data and attribute-specific data. We strive to develop a flexible AVM, that can be applied to any real estate market given sufficient source data, and implement and evaluate it in Oslo.

This study may be placed in the intersection of i) the research surrounding AVMs, ii) the application of *ensemble learning* in the field of econometrics and iii) the research in the construction of real estate indices. In the two latter research areas, we find extensive literature from the past five decades. Ensemble learning methods by Breiman (1996a) and Schapire (1989) have been applied with success in relatively few, albeit increasing number of, econometric works (Graczyk, Lasota, Trawiński, & Trawiński, 2010; Inoue & Kilian, 2008).[2] The research into the construction of real estate indices has had few improvements since the work by Case and Shiller (1987), building on Bailey, Muth, and Nourse (1963). These indices are acclaimed by academia (Balk, De Haan, & Diewert, 2011) and implemented by industrial actors (S&P Dow Jones Indices, n.d.).

We find the publicly available research regarding AVMs to be fragmented, and that a methodological consensus has yet to emerge. Traditionally, methods based on ordinary least squares (OLS) have been the prominent approaches (Pattichis, 1999). More recently, geographically optimised regressions have been explored to better fit the dynamics of real estate (Moore & Myers, 2010). However, both these methods suffer from multicollinearity issues related to their parametric form (Wheeler, Tiefelsdorf, Wheeler, & Tiefelsdorf, 2005). Today, much of the publicly available development of AVMs flourishes amongst machine learning communities like Kaggle[3], where contemporary techniques such as artificial neural networks and ensemble learning are the most popular (Adam-Bourdarios, Cowan, Germain, & Guyon, 2015; D. Nielsen, 2016).

[1] Downie and Robson (2007) provides a global perspective on the use of AVMs across commercial and government applications.

[2] We particularly recommend Mullainathan and Spiess (2017) as an excellent overview of machine learning applications in econometrics.

[3] https://www.kaggle.com/zillow. Retrieved May 27th 2018

Our approach to developing an AVM is fairly novel in academic finance, and is largely inspired by innovations of data science communities (Kaggle, 2018) and leading international industrial actors (Zillow, 2017). First, it combines the well-established repeat sales method (RSM) by Case and Shiller (1987) with four machine learning methods by using a concept known as *stacked generalisation* (D. H. Wolpert, 1992). Second, our model includes a form of valuation known as *comparable market analysis* (Rattermann, 2007), which seeks to value dwellings based on transactions with a close spatial and temporal proximity.

The four machine learning methods, which we combine with the RSM, are known as *ensemble learning methods*. Ensemble learning is a class of modern machine learning methods which combines multiple models into one model in order to increase its *out-of-sample* predictive force (Opitz & Maclin, 1999). The four submodels are from two classes of ensemble learning techniques; *bagging* (Breiman, 1996a) and *boosting* (Freund & Schapire, 1996; Schapire, 2013). The use of these ensemble learning methods allows our model to learn patterns in the underlying data, without making any assumptions regarding the data. We hypothesise that these ensemble learning methods are suitable for use in the residential real estate market, due to the amount of available source data and the complexity of the modelling task.

The AVM developed in this study shows several encouraging results. We evaluate our model by producing out-of-sample estimates for the 1 979 dwellings sold in Oslo in Q1 2018. Our AVM values these dwellings with a median absolute percentage error (MdAPE) of 5.4%, with more than 96% of the dwellings being estimated within 20% of their actual selling price. We compare the model to traditional AVMs, estate agents and the industry leader for commercial AVMs, Zillow. The performance of our AVM is comparable to the accuracy of estate agents, which has been at 5.3% in Oslo over the past two years, and superior to the precision of Zillow in several of the cities in which they provide official performance statistics.

This thesis has two main implications. First, by making our AVM publicly available through a web service, it can directly aid actors in the real estate market in Oslo. We aim to do this by continuing to work with our industrial partner, Alva Technologies. Second, the combination of modern ensemble learning techniques and traditional real estate indices has yielded promising results. We believe that further research within the combination of these fields will be beneficial for future AVMs.

The remainder of this study is structured as follows: In Chapter 2, the residential real estate market in Oslo is presented, with an emphasis on concepts critical to our model. Next, in Chapter 3 we describe the datasets and data pre-processing steps used by our AVM. In Chapter 4, we develop and justify the technical aspects of our AVM. Chapter 5 presents the results of our model, and evaluates its performance. Finally, in Chapter 6, we summarise the performance of our AVM and suggest future extensions and areas of research within this field.

2 THE RESIDENTIAL REAL ESTATE MARKET IN OSLO

Real estate valuation models are known to be highly dependent on local conditions (Tsatsaronis & Zhu, 2004). Therefore, we use this chapter to introduce critical factors specific for real estate in Oslo. In particular, we seek to understand the factors affecting the selling prices in the residential real estate market in Oslo, with the aim of adapting our AVM to the city's conditions. To do this we i) present fundamental background information on the residential real estate market in Oslo, ii) provide a brief description of the ownership types of residential real estate in Norway and iii) discuss some underlying market drivers that affect residential real estate prices.

2.1 BACKGROUND

The residential real estate market in Norway is characterised by a strong tradition of home-ownership, with 84 percent of Norwegians live in a self-owned home, compared to 69 and 65 percent for Sweden and Denmark, respectively (Eurostat, 2015). By January 2018, the population of Oslo was 673,468 (Statistics Norway, 2018a). The city has experienced large growth over the past few decades, and metropolitan Oslo has contributed to roughly 50% of the population growth of Norway (Statistics Norway, 2010). The dwellings located in central parts of Oslo are typically characterised by four- and five-storey brick apartment buildings. Historically, western parts of Oslo have generally had larger, more expensive houses, while eastern parts have had smaller, less expensive apartments. Oslo is divided into 15 districts, in addition to the city centre, as illustrated in Figure 1.

Figure 1: Map of Oslo, including district borders.

Two official registers, *Matrikkelen* and *Grunnboken*, define real estate property and ownership relations for the Norwegian real estate market (The Norwegian Mapping Authority, 2017). Every dwelling is described in *Matrikkelen*. It contains information about the location and boundaries of the dwelling, its size, attached property and, in the case of apartments, the building in which it is contained. *Grunnboken* describes ownership relationships for both private property and cooperatives.

In the Norwegian real estate market it is a common practice to define the selling price as the sum of the transaction amount the buyer pays and the common debt which the buyer incurs (Ministry of Local Government and Modernisation, 2009; Statistics Norway, 2017). We use this definition in our model.

2.2 Ownership Types and Common Debt

There are two main ownership types in Norway; condominiums and cooperatives. While condominiums are owned directly by individuals, cooperatives are owned by a legal entity, known as a *cooperative association*. Individuals can buy and sell shares of this association to acquire and relinquish the right to use a dwelling. There are important tax-related distinctions between the two categories; purchasing a condominium results in a document tax of 2.5% of the selling price, while no such tax is levied when acquiring a dwelling in a co-operative association. Furthermore, the cooperative association is a separate legal entity, responsible for the debt incurred from construction or renovation, while every cooperative member is responsible for the common costs, which essentially maintains the common debt. Besides, there may be further regulations enforced by the cooperative, e.g. regulating subletting. The combined effect of all these restrictions on the sold price is thought to be minimal (Boligmani, 2015; E24, 2014). We analyse the differences between ownership types in our dataset in Chapter 3.

Common debt is the debt that is owned by a cooperative or a group of condominiums and is incurred when purchasing a dwelling. The debt originates from the construction of dwellings in a cooperative, or renovation projects for both cooperatives or groups of condominiums. Apart from potential differences in interest rates and the handling of default scenarios, this debt is equivalent to any other debt obtained when purchasing a dwelling.

2.3 Market Dynamics

Developments in observed explanatory variables cannot fully explain the observed variation in house prices. There are underlying macroeconomic drivers, as well as market dynamics, that have significant short-term effects on residential real estate values (Adams & Füss, 2010). The level and expectations of interest rates, policy decisions and the GDP of Norway are important macroeconomic drivers, while the price of substitute goods (rent prices), population growth, amount of debt in the housing market and rate of construction are important market dynamics (Harris, 1989; Sufi & Mian, 2014). We do not analyse any of these variables, as we consider this to be outside of the scope of this paper. As explained in Chapter 4, we aim to use the repeat sales method (RSM) to capture the appreciation or depreciation in the market.

3

Data Analysis

3.1 The Datasets

Our primary datasets are consolidated from *Grunnboken* and *Matrikkelen*, introduced in Chapter 2.1, by Alva Technologies. They have conducted a thorough data cleaning process and provided us with three datasets. These can be summarised as follows:

i) **Address dataset:** Consists of all dwellings in Norway, where 276 780 of them are located in Oslo. The dataset contains most of the descriptive variables displayed in Table 1.

ii) **Enhanced transaction dataset:** Consists of 18 401 transactions from Oslo between August 2016 and April 2018. Proprietary dataset with improved quality and additional variables.

iii) **Historical transaction dataset:** Consists of all registered residential real estate transactions in Oslo between January 1993 and May 2018, illustrated in Figure 2. In total 220 898 separate transactions are mapped to Oslo-addresses. Includes *unique address identifier*, *sold price* and *sold date*, but is missing data on *common debt* and *usable square meters (USM)*.

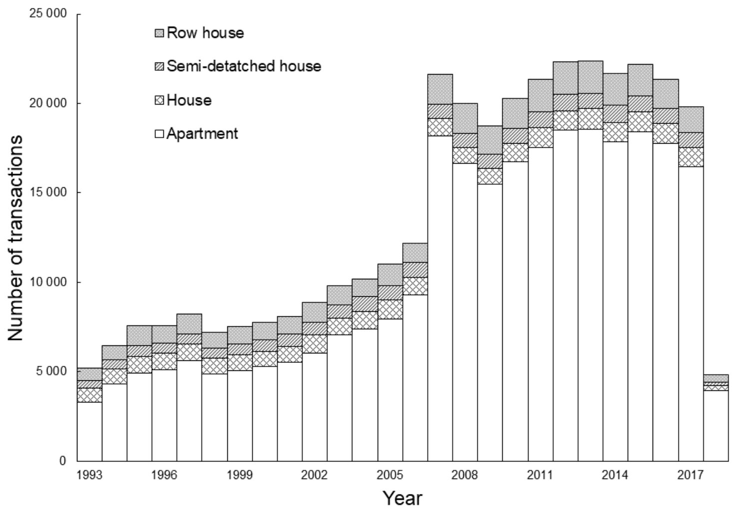


Figure 2: All recorded transactions in the dataset (1993-2018) by year of transaction. Note that sales from cooperatives are added around 2007 causing a sharp increase in the number of transactions.

Table 1 illustrates the attributes that are gathered from these datasets and used by the ensemble learning methods in Chapter 4.2. The dwellings' locations are described by both geographical coordinates and districts. The area within the walls of an individual dwelling is defined as its usable square meters (USM). Dwellings in Norway are typically divided into four main unit types; *apartments*, *row houses*, *semi-detached houses* and *houses*. Apartments are defined as dwellings which occupy only part of a larger building, while the three other unit types denote variations of dwellings that are built separately. When modelling, we explicitly distinguish between apartments and non-apartments (i.e. row house, semi-detached house or house), due to differences discussed in Chapter 3.3. We do, however, use the unit type as a categorical variable in the ensemble learning methods.

Table 1: Overview of the exogenous variables used by the attribute-based pricing methods.

Variable	Type	Description
USM	Numeric	The dwelling's usable square meterss
Coordinates	Numeric	The geographical coordinates of a dwelling
Storey	Numeric	The floor on which the dwelling is located
# of rooms	Numeric	The number of rooms in the dwelling
# of days since sale	Numeric	Number of days since the occurrence of the transaction
Rank	Numeric	Measure of proximity to target dwelling, increasing with distance
Build year	Numeric	Construction year of the dwelling
Common debt	Numeric	The dwelling's part of the debt held by the group of properties
Sold month	Categorical	The year and month of the occurrence of the transaction
District	Categorical	The district in which the dwelling is located
Unit type	Categorical	The unit type of the dwelling
Build type	Categorical	The type of the dwelling, as by NS3457 (2013)
Ownership type	Categorical	The dwelling's ownership type, as by Chapter 2.2
Elevator	Categorical	Whether or not the building of the dwelling has an elevator

In addition to the datasets provided by Alva Technologies, we use a separate dataset to analyse estate agents' aggregate precision and compare it with that of our model. This dataset, gathered from Eiendomsverdi, contains 15 786 transactions in Oslo from 2016 and 2017 and includes both the *asking price*, provided by the estate agent, and the final *sold price*.

3.2 Data Pre-Processing

As shown in Figure 3, there is a significant degree of missing data in our datasets, with certain attributes such as *number of rooms* and *storey* missing values in roughly 30 % of its records. Also, we have been advised by industry experts that the *historical transaction dataset* is prone to contain erroneous data. Specifically, this seems to be due to the occurrence of sales under special circumstances, poor data management, and poor data integration of data from multiple sources.

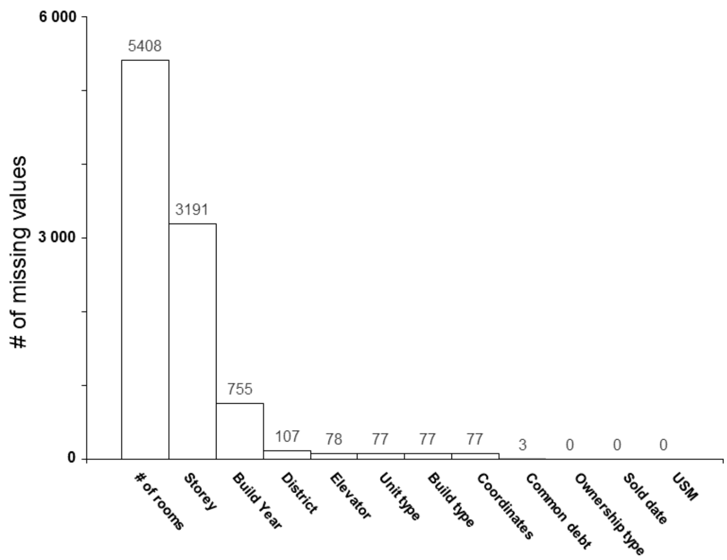


Figure 3: The number of missing values for each explanatory variable. The total number of transactions is 18 073.

The issues above motivate the development of a pre-processing procedure prior to developing our AVM. To handle missing data we consider a selection of methods presented in Allison (2001), including *imputation* and *listwise deletion*, and build on these concepts. To develop a cleaning procedure for the transaction data, we rely on advice from industry experts and practices prevalent in academia. We perform a pre-processing procedure consisting of two steps; i) handling missing values and ii) additional cleaning for the historical transaction dataset.

First, we handle dwellings with missing data, either by removing these dwellings (and associated transactions) from our dataset or by imputing substituted values. First, we remove all dwellings where we do not have data regarding the *district*, which in total affects less than 1 % of the dwellings. By doing so, we simultaneously remove all data points with missing values for *elevator*, *unit type*, *build type* and *coordinates*. Next, we impute values for data points with missing values for *storey*, *build year* and *number of rooms*, with the mean value of all remaining dwellings. This technique is known as *mean substitution* and is a standard method for imputation (Hariharakrishnan, Mohanavalli, Srividya, & Sundhara Kumar, 2017). In total the imputation affects around 30 % of the transactions.

Second, we further clean the **historical transaction dataset**. The outline for this data cleaning step is given below in Procedure 1.

PROCEDURE 1: Cleaning of historical transaction dataset

1) Remove transactions with a selling price below 100 kNOK or above 70 MNOK. This accounts for less than 1 % of the transactions.
2) Remove transactions where the same property is sold twice within three months. This might be due to distressed sales or speculative transactions, and are therefore not likely to explain the real appreciation or depreciation in that given period (Jansen, De Vries, Coolen, Lamain, & Boelhouwer, 2008).
3) Remove transactions where the ratio of the sold prices between two transactions is larger than five. This is an unlikely observation, and we expect it to be due to an error in the logging phase or large renovation projects.
4) Remove dwellings that have more than ten previous transactions within the recorded time period. This high frequency of re-selling is unlikely, and the dwellings will typically not be representative, as argued by Case and Shiller (1987).

We note that our AVM is evaluated on a test set with transactions solely from the enhanced transaction dataset. Therefore removing transactions from the historic transaction dataset will not bias the selection of our test set and overstate our results.

3.3 Exploratory Data Analysis

In this section, we analyse the datasets to summarise and discuss their main characteristics, especially those concerning location, sold price, common debt and usable square meters, for the different unit types and ownership types. With this, we aim to be able to adapt our AVM to the datasets. We start by analysing the interdependencies between the different explanatory attributes by discussing the correlations of the numerical attributes. Next, a detailed illustration of the most recent sales in Oslo is used to highlight the local differences in sold price per square meter. Finally, we provide statistics on the dwellings with multiple previous sales and discuss changes over time for the historical transaction dataset. This lays a foundation for the development of our AVM, and multiple choices made in Chapter 4 are due to findings in this section.

On Multicollinearity and Omitted Variable Bias

When analysing the impact of a dwelling's attributes on its selling price, one needs to be cautious of potentially skewed estimates due to interdependencies of the explanatory variables (Walpole, Myers, Myers, & Ye, 2016). As shown in Figure 4, there are clear linear relationships between the attributes in our dataset. For instance, we observe a positive correlation between *build year* and *longitude*, and also a negative correlation between *USM* and *storey*.[4] These observations indicate that the dwellings located in western parts of Oslo are, on average, older and that dwellings on higher floors tend to be smaller in size. This inherent multicollinearity of explanatory variables in the residential real estate market has to be accounted for when developing an AVM, since many traditional real estate pricing methods build on an assumption of independent attributes which can be priced individually (Wheeler et al., 2005).

Another potential issue with the dataset is the sufficiency of the observed and recorded set of explanatory variables. When price-affecting characteristics of a dwelling are excluded, then the model may suffer from what is known as *omitted variables bias*. This bias may carry over to the predicted prices and affect their accuracy. In practice, the bias is impossible to avoid, as detailed information capturing structural characteristics, neighbourhood quality, among others, can be hard to obtain. Therefore, the effect of the *omitted variable bias* depends on the model specification.

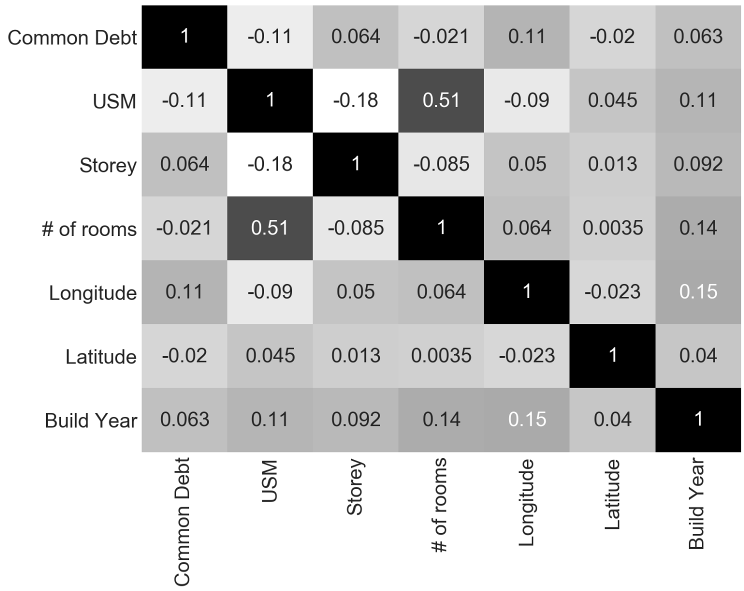


Figure 4: A plot of Pearson correlation coefficients between the numerical variables in our datasets, illustrating linear dependencies between several of the variables. These dependencies present challenges when pricing real estate using linear regression models.

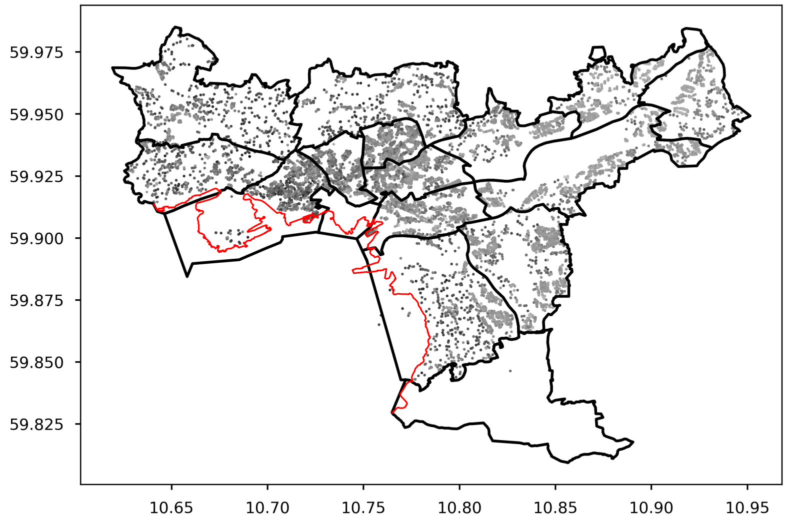
Location

As discussed in Chapter 2 the location of a dwelling is known to be a major driver of its selling price. Table 2 illustrates the variations in price per square meter in some of the districts in our dataset, and Figure 5 shows prices per square meter for all transactions in the enhanced transaction dataset. Both Table 2 and Figure 5a highlight the clear variations in prices between districts, which motivates the use of districts as an explanatory variable. However, we also observe large local differences within districts. Figure 5b shows this in Nordstrand, which is a district with both proximity to the coastal line and areas with large high-rise buildings. This motivates the inclusion of the dwelling's geographical coordinates as a finer grained location measure.

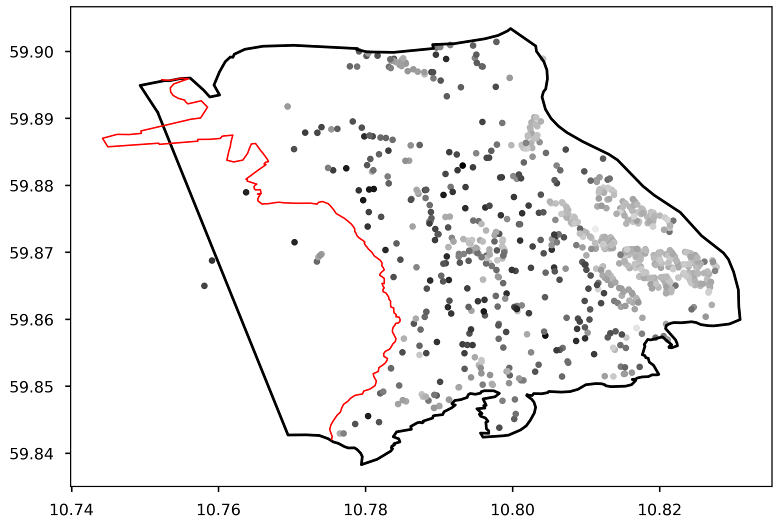
[4]Both of which are significant with a p-value of < 0.001.

Table 2: The 25th, 50th (median) and 75th percentile of the price per square meter (PPSM), for a selection of the districts in Oslo, based on the enhanced transaction dataset.

	Alna	Østensjø	Nordstrand	Grünerløkka	Frogner
25 %	41 kNOK	49 kNOK	52 kNOK	65 kNOK	76 kNOK
Median	47 kNOK	54 kNOK	57 kNOK	72 kNOK	85 kNOK
75 %.	55 kNOK	60 kNOK	64 kNOK	81 kNOK	96 kNOK



(a) All recorded transactions in Oslo (2016-2018), in total 18 073 transactions.



(b) All recorded transactions in the district Nordstrand (2016-2018), in total 1 167 transactions.

Figure 5: All recorded transactions in Oslo (2016-2018). A darker colour represents a higher price per square meter (PPSM). The bolded black lines represent the district borders, while the red line is the coastline. We observe large local differences, highlighted by the the plot of Nordstrand.

Common debt

In Chapter 2 we introduced some of the main practical and jurisdictional differences between the two ownership types. In Table 3 we present basic descriptive statistics for the ownership types, based on the transactions in the enhanced transaction dataset. It indicates major differences in the amount of common debt for the two ownership types. There are also noticeable differences in the *sold price*, but these variations might also be due to factors other than the ownership types; there are observed differences in location, build year etc. While the median-sized dwellings in the two ownership types are almost identical in size, the median amount of *common debt* for condominiums is less than 3 % of that of cooperatives. The low level of common debt for condominiums implies that the historical transaction dataset can be used, even though it lacks data regarding common debt.

Table 3: Descriptive statistics on the *usable square meters (USM)*, *sold price* and *common debt* for cooperatives and condominiums, based on the enhanced transaction dataset, illustrating differences in common debt.

	Cooperative			**Condominum**		
	USM	Sold Price	Common Debt	USM	Sold Price	Common Debt
Mean	58	3.7 MNOK	230 kNOK	67	5.1 MNOK	41 kNOK
St.Dev.	18	1.0 MNOK	318 kNOK	27	2.0 MNOK	72 kNOK
25%	46	3.0 MNOK	76 kNOK	49	3.6 MNOK	0 NOK
50%	59	3.4 MNOK	141 kNOK	60	4.5 MNOK	4 kNOK
75%	69	4.0 MNOK	228 kNOK	82	6.1 MNOK	58 kNOK

Repeat sales

We have registered transactions on 185 961 of the dwellings in our dataset, and 100 268 of these are sold at least twice. We define two consecutive transactions of a dwelling as a *repeat sale*. Figure 2 illustrates the number of dwellings sold each year, and Appendix A.5 displays the number of recorded transactions per month for each district. We note that we do not have data on the sales of *cooperatives* from 1993 to 2004, and in the period 2005-2006, only a minority of cooperative sales are registered. Hence, we observe a spike in Figure 2 in 2007, when sales from cooperatives are added. Also, we observe an increasing number of transactions each year. During the last ten years, we observe a doubling in the number of transactions in Oslo. This sparsity of transactions before 2007 should be considered when modelling the development in historical prices. At the same time, the large total number of repeat sales motivates the use of previous sales when valuing dwellings.

Unit types

The dwellings from the four unit types (apartment, house, semi-detached house and row house) differ in both size and sold price. Table 4 illustrates this for all recorded transactions from 2016 until 2018. An important aspect here is that roughly 90 % of the transactions are from apartments, while only 10 % of the transactions are from non-apartments.

Table 4: Descriptive statistics on *usable square meters (USM)* and *sold price*, based the enhanced transaction dataset, illustrating differences and similarities between unit types

	Apartment		**Row house**		**Semi-detached house**		**House**	
	USM	Sold Price	USM	Sold Price	USM	Sold Price	USM	Sold Price
Mean	63	4.2 MNOK	137	6.3 MNOK	137	8.5 MNOK	192	11.6 MNOK
St.Dev.	24	1.9 MNOK	45	2.9 MNOK	45	2.9 MNOK	58	4.0 MNOK
25%	48	3.0 MNOK	105	4.3 MNOK	105	6.2 MNOK	156	9.0 MNOK
50%	62	3.6 MNOK	135	5.7 MNOK	135	8.5 MNOK	187	11.2 MNOK
75%	73	4.8 MNOK	128	7.9 MNOK	165	10.4 MNOK	222	13.6 MNOK

Table 4 demonstrates that apartments are much smaller than the other three categories in size. The 75th percentile usable square meters for apartments is smaller than the 25th percentile for row houses. Semi-detached houses are more expensive than row houses, and dwellings tagged as houses are by far the largest and most expensive. The median-sized house is more than three times that of the median-sized apartment. We observe smaller differences between the categories in *sold price* than in *USM*, indicating that there might be a decreasing marginal sold price for increasing dwelling size. The large share of transactions involving apartments, as well as the apparent similarity of the non-apartment unit types, suggest a different modelling approach for apartments and non-apartments.

Sold date

The dynamics of the housing market discussed in Chapter 2.3 often causes large short-term fluctuations in real estate prices. The precise date of a transaction is therefore critical when modelling these prices. From here on we denote *sold date* as the date when the buyer and seller agree upon a selling price, and *official date* as the date when the dwelling is handed over to the buyer. The official date is registered in the official registers ("Grunnboken"), and can be several months after the sold date. In our data, we find the average difference between official date and sold date to be 46 days. We argue that the sold date is the most interesting when modelling house prices, as it indicates the time at which the sold price was determined. All the transactions in the enhanced transaction dataset have a recorded sold date. However, for the historical transaction dataset we only have sold date on some transactions. Therefore, we use the official date as a proxy for the sold date for these transactions.

4 METHODOLOGY

4.1 METHODOLOGY OVERVIEW

In this chapter, we develop our automated valuation model (AVM). First, we introduce two key concepts of our AVM, namely *stacked generalisation* and *comparable market analysis*. Next, we describe the most prominent value estimation technique for AVMs, namely attribute-based pricing methods. In particular, we present four recently developed machine learning methods that aim to circumvent some of the shortcomings of traditional parametric attribute-based pricing methods. Further, we describe the repeat sales method, a widely used method for creating *real-estate indices*, and how we apply it to real estate valuation. Finally, we present our AVM, which combines the aforementioned concepts.

Although we aim to predict the selling price, we model the price per square meter (PPSM), as this is a common measure and recommended in academia (Bonnet, 2012; Kettunen, 2012) and industry (Statistics Norway, 2018b).

Stacked Generalisation

When producing a value estimate for a dwelling, our AVM uses multiple underlying methods, known as *submodels*, to create individual value estimates for every dwelling in the training data. A separate model, known as a *model-stacker*, then analyses the individual value estimates *in-sample*, to determine the *out-of-sample* prediction for the dwelling. This technique is referred to as stacked generalisation[5], and was pioneered by D. H. Wolpert (1992) and refined by Breiman (1996b). Their idea was that the model-stacker should use the training data and the individual predictions to identify the biases made by the underlying algorithms, thus reducing the bias when predicting out-of-sample. The stacking procedure is outlined in Figure 6, and will also be detailed in Chapter 4.4.

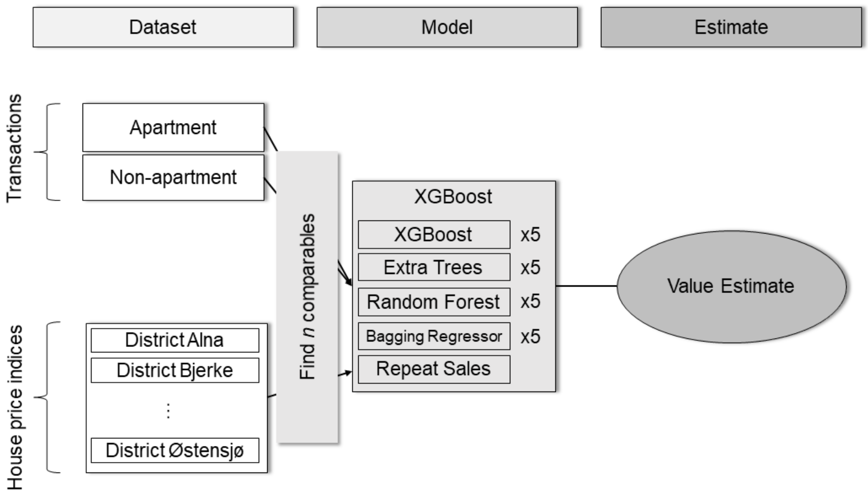


Figure 6: Summary of the automated valuation model (AVM). The model is trained for each individual value estimate it produces. The model extracts the n[6]comparable transactions. The repeat sales method (RSM) requires pre-processed indices, based on the historical transaction dataset, while the four other ensemble learning algorithms are trained on five *folds* of the training data. The XGBoost-algorithm combines the underlying models to produce one value estimate.

[5]We refer to Chapter 2 of D. H. Wolpert (1992) for a rigorous definition of stacking. However, we do quote said paper and note that "*it is in the nature of stacked generalization that presenting it in full generality and full rigor makes it appear more complicated than it really is*".

[6]$n = 10000$ when the dwelling is an apartment, and $n = 2000$ otherwise.

Comparable Market Analysis

Our AVM tailors the *training data* for each value estimate based on the dwelling it aims to value, thus fitting a unique model to each particular dwelling. The training data is selected to mimic the dwelling as closely as possible. This concept, known as comparable market analysis, is a prevalent valuation principle often applied to real estate valuation (Rattermann, 2007). In particular, estate agents use nearby, recent, sales as a starting point when valuing a dwelling. Automated valuation models can mimic this behaviour by tailoring its source data to include transactions of dwellings in close geographical proximity to the dwelling in question. This is a key concept of our model, and is described in Chapter 4.4.

4.2 Attribute-Based Pricing Methods

Traditionally, hedonic pricing methods (HPMs) have been prevalent in academic literature for residential real estate valuations (Balk et al., 2011). HPMs build on the assumption that goods are typically sold as a package of inherent attributes, and implicit prices of these attributes can be estimated from observed prices of differentiated products and the specific amounts of characteristics associated with them (Rosen, 1974). Using these implicit attribute prices, one can predict the selling price of a dwelling from the value of its underlying attributes. However, due to the high degrees of multicollinearity between key variables and potential *omitted variable bias*, as discussed in Chapter 3.3, traditional HPMs may suffer from model misspecification (Balk et al., 2011; Wheeler et al., 2005).

We propose an alternative approach, where four *ensemble learning methods* are used to generate four independent value estimates. Each method creates multiple submodels, fitted to independently sampled input data. When predicting the value of an unseen dwelling, the methods combine each submodel's prediction to determine the new selling price. The four methods we apply are bagging predictor (BP), random forest (RF), extra trees (ET)[7] and XGBoost (XGB).

Before we describe the four ensemble methods, we briefly introduce the underlying machine learning concepts they rely upon, namely *decision trees* and *bootstrapping*, in addition to how these methods are adjusted to the datasets through *hyperparameter tuning*. In the following subsections we present i) our selected ensemble learning methods and ii) the rationale and challenges around their use.

Ensemble Learning - Key Concepts

Each of our four ensemble methods builds multiple submodels known as *decision trees* and combines every tree's prediction to produce one value estimate. The number of trees in each ensemble model, as well as the rules for building each tree, are critical choices for the success of the ensemble method. We will here introduce central concepts related to ensemble learning.

A *decision tree* is a simple, but powerful tool for predictive modelling (Lior & Others, 2014). Informally, a decision tree is a tree-structure alike with a flowchart, as illustrated in Figure 7, where each internal node denotes a test of an attribute, the subsequent branching represents the outcome of the test, and each leaf node holds a prediction. Specifically, each internal node in the decision tree splits the dataset into two disjoint sets, on a particular binary test related to a *cut-point* value of a given attribute. The attribute and its cut-point are chosen to minimise an objective function, typically the mean squared error, of each branch. The predictions in the leaf nodes of our decision trees are determined by the average PPSM[8] of the dwellings in the training data directed into that branch.

There are several *hyperparameters*, regarding the construction of decision trees in each

[7] Extra Trees is actually an acronym derived from **Ext**remely **ra**ndomised trees.

[8] As mentioned in Chapter 4.1 we model the PPSM of the dwellings, rather than the selling price.

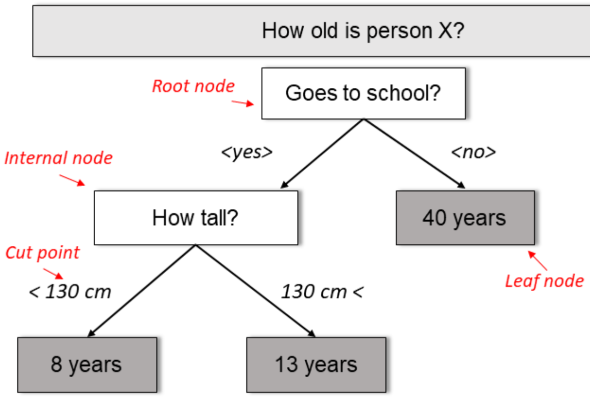


Figure 7: A descriptive example of a decision tree. Illustrating how binary choices divide the dataset into leaf nodes, where predictions are given on basis of the training set.

ensemble method, which determine the strength of the model. Two essential hyperparameters are the *number* of decision trees to build, and the *depth* of each decision tree. Another, more subtle, hyperparameter is the option to use *bootstrapping* or not. By using bootstrapping the methods pick random transactions from the training data with replacement. This sampling procedure produces separate datasets for each decision tree.

There is a trade-off to consider, when determining the hyperparameters above, between the explanatory power of the model and its computational complexity. In general, increasing the number of trees will yield a higher explanatory power, but is more computationally burdensome. A model with high explanatory power captures the prevalent relationships in the data (i.e. avoiding *underfitting*) while at the same time avoiding identifying non-existing relationships between the attributes (i.e. avoiding *overfitting*).

Optimising the Individual Methods - Hyperparameter Tuning

Both the bagging and boosting algorithms require tuning of hyperparameters to be fit to any particular dataset. This is due to differences in the amount of available data, the number of attributes in the dataset and the structure of the dataset. Some hyperparameters are common for all our algorithms, like the number of decision trees to create and the maximum depth of each tree, while others are specific for each model. Table 5 and Table 6 provide a description of the different hyperparameters, and their tuned values. We refer to the documentation from SKLearn and XGBoost for further descriptions of the parameters and their default values (Chen & Guestrin, 2018; Pedregosa et al., 2011).

In optimising the parameters for the bagging predictor, random forest and extra trees methods we follow Hauck (2014), which provides an in-depth analysis of parameter tuning. When tuning the parameters for XGBoost we follow DMLC (2016), a guide provided by the machine learning community which developed XGBoost.

The parameter tuning was implemented by using the *CrossValidation* package in *SKlearn*, to search over a set of possible parameters. The search was done iteratively, decreasing the range for each parameter successively to find the optimal value. The cross-validation procedure divides the training data into five separate training and validation sets and runs each model with a given set of hyperparameters on each dataset. It then averages the prediction errors on the validation sets and chooses the optimal combination of hyperparameters based on a scoring function. We use mean absolute error as the scoring function when selecting the optimal hyperparameters. Note that the hyperparameter tuning is done on the training data, *not* the testing data (i.e. the data from the first quarter of 2018). Hyperparameter tuning on the testing data would lead to overfitting and thus overstated performance estimates.

Bagging - Bagging predictor, Random forest and Extra trees

Bagging is an ensemble technique in which the underlying decision trees are trained in parallel and independently. We utilise three bagging methods, specifically *bagging predictor (BP)*, *random forest (RF)* and *extra trees (ET)*. Random forest is an extension of bagging predictor, while extra trees extends random forest again. The methods have subtle, but distinct, variations in the procedure of building decision trees. We will first introduce their common concepts by describing the most general bagging method, namely the bagging predictor, as by Breiman (1996a). Subsequently, we present the extensions made in random forest, as by Breiman (2001), and extra trees, as by Geurts, Ernst, and Wehenkel (2006). We note that we present these algorithms with their tuned hyperparameters.

The **bagging predictor** method builds a fixed number of independent decision trees, by sampling the training data with bootstrapping. When constructing each decision tree, the method searches over each attribute and each cut-point to find the attribute that best splits the data at a given node. When calculating the sold price of an unseen dwelling, the bagging predictor averages the estimates from all the decision trees.

Random forest differs from bagging predictor in the method used for "growing" the underlying decision trees. Random forest builds the trees by sampling from only a randomly selected subsample of the attributes at each node split. This is known as *feature bagging*. As noted by Breiman (2001), the prediction error of ensembles of tree predictors depends on the strength of the individual trees, as well as the correlation between them. By using feature bagging at each node split, the random forest will tend to reduce the correlation between trees, thus yielding a more robust model for out-of-sample predictions. Feature bagging also has the added benefit of being less computational burdensome.

Instead of using feature bagging, as in the random forest method, **extra trees** randomises the choice of cut-point of each attribute to learn decorrelated trees. That is, it arbitrarily chooses a value (cut-point) for each attribute when splitting the trees, instead of trying all possible cut-points. Doing so increases the randomness, in addition to reducing the computational burden of the algorithm.

Table 5: The hyperparameters of bagging predictor (BP), random forest (RF) and extra trees (ET) - descriptions and tuned values.

Variable	Applicable for	Description	Tuned Value
Number of trees	BP, RF, ET	The total number of decision trees created	250, 150, 100
Bootstrapping	BP, RF, ET	Whether or not to pick subsamples with replacement	True
Maximum depth	RF	The (maximum) depth of each tree	50
Share of attributes	RF	The share of attributes to use when creating a split	0.33

Boosting - XGBoost

Boosting is an ensemble technique introduced by Schapire (1989) which trains the underlying decision trees *sequentially*, with each tree fitted to improve the errors made by preceding trees. As with bagging methods, bootstrapped training data is used to train the underlying trees. In contrast to bagging methods, each new tree improves on the predictions of the previous tree by attempting to improve its "shortcomings". The *AdaBoost* method, proposed by Freund and Schapire (1995), was the first proposal for a boosting algorithm, and was generalised by J. H. . Friedman (2001) into the *Gradient Boosting Machine*. During the past five years, there have been several contributions to the development of boosting methods as a result of the increased interest in data science (Chen & Guestrin, 2016; Schapire, 2013).

XGBoost is a recently developed boosting method, that has proved successful in a variety of machine learning competitions[9]. We will describe the fundamental concepts of the method, and how we have used and tuned the method to fit our purpose, but refer to Chen and Guestrin (2016) for a thorough explanation of the implementation details of the method.

XGBoost is implemented to minimise an objective function consisting of a loss function plus a regularisation term at each iteration. Regularisation is a term added to constrain the model from overfitting. We write the objective function as

$$Obj = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{1}$$

where $l(\cdot,\cdot)$ is the loss function, $\Omega(\cdot)$ is the regularisation term, f_k is the k^{th} decision tree, and $\hat{y}_i$ and y_i are the predicted and actual selling price, respectively, of the i^{th} dwelling.

The trees are built by splitting leaf nodes on the attribute value which minimises the prespecified objective function. This is done recursively until the trees reach a prespecified maximum depth. In our implementation, the loss term of the objective function is chosen to be the *mean absolute error*.

The trees are built by adding two new leaf nodes to an existing leaf node, split on the value (cut-point) of the attribute which minimises the objective function.

Each leaf contains a weight, determined by the first and second order differential of the loss function and the regularisation term[10]. Each decision tree is trained on the residuals from the previous iteration, continually improving the estimates. When creating a value estimate for a dwelling x_i, XGBoost sums the selected weight for each tree, as shown in (2).

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{2}$$

As with the bagging methods, XGBoost has several important hyperparameters that one needs to set when developing the method. Their tuned values and practical implications are summarised by Table 6[11].

Table 6: The tuned hyperparameters of XGBoost, both as a model-stacker and as an underlying method - descriptions and tuned values.

Variable	Description	Tuned Value
Learning rate	The step-size shrinkage used in each update	0.005
Number of iterations	The number of trees to build	1 000
Gamma	The minimum loss reduction required to make a split	0
Maximum depth	The (maximum) dept of each tree	5
Subsample	The share of data points to used when building a tree	0.8
Colsample by tree	The share of attributes to used when building a tree	0.8
Evaluation Metric	The loss function the algorithm aims to minimise	MAE

Model Rationale and Challenges

We argue that there are several advantages with ensemble learning methods compared to other alternatives. The primary rationale of these methods is that they are non-parametric, i.e. they do not require any rich *a priori* knowledge regarding the underlying data generating process. This allows the method to adapt to the underlying data and model poten-

[9]See D. Nielsen (2016) for a comprehensive analysis of the method's success.

[10]See equation 5, and related descriptions, of Chen and Guestrin (2016) for the technical rationale of this.

[11]We use the default values for the remaining hyperparameters in XGBoost.

tial non-linear relationships. In contrast, the traditional HPMs do not model non-linear relationships and tend to make strict assumptions about the data, such as linearity and homoscedasticity.

Further, we find the ensemble learning methods to be ideal for the amount of relevant and available training data. More complex non-linear models, such as artificial neural networks (ANNs), often require many more degrees of freedom to yield high predictive power (Bishop, 2006, Chapter 5), which quickly may result in *overfitting* in the case of limited data supply. Similarly, simpler models such as HPMs based on ordinary least squares (OLS) may not be able to fully utilise the dataset, due to a constrained functional form. We discuss the use of OLS and ANNs in Chapter 5.3 and Appendix A.1 with references to achieved empirical results with such models.

We also note that the selected ensemble learning methods require minimal *feature engineering*[12], especially concerning the grouping of attributes into categorical variables, but also transformations of continuous variables. The underlying decision trees are constructed to learn such patterns without requiring significant domain knowledge. This makes our model simpler to apply and more robust in dynamic real-estate markets.

On the other hand, we acknowledge that the ensemble methods do not provide the same degree of transparency as the simpler OLS-based HPMs. The HPMs yield price estimates of individual attributes, while the ensemble learning methods, which are more elaborate, can be harder to interpret. However, there are certain methods for producing *attribute importances* for the underlying attributes. In Chapter 5.3 we give a demonstration of one such attribute importance calculation. We believe that these techniques provide a sufficient degree of transparency for most applications of AVMs.

Another potential challenge with ensemble learning methods is that, as they do not make any underlying assumptions of the distribution of the data, they may struggle to generalise beyond the observed training data. They are therefore heavily reliant on an extensive dataset to yield robust results. We do not expect this to be an issue for our model as we use a comparable pricing approach to select and engineer the training data, as we will elaborate in Chapter 4.4. That is, we select training data which is tailored to suit the dwelling in question, thus making the model highly likely to generalise to the given dwelling.

4.3 Repeat Sales Method

We use the repeat sales method (RSM) to create a separate index for each district in Oslo, with the aim of capturing the appreciation or depreciation in the market over time, as discussed in Chapter 2.3. The indices are used to produce price estimates for all previously sold dwellings in our dataset, by adjusting each dwelling's previous selling price with the appropriate index.

The RSM was introduced in the seminal paper by Bailey et al. (1963) and builds an index model by considering dwellings which have been sold more than once. The model is exceedingly simple and provides a basis to predict the price of any previously sold dwelling. The method's core idea is that the ratio of selling prices, for the same dwelling, at two distinct times can be thought of as a ratio of an (unobservable) index for the local area at the two selling times multiplied by an error term. This idea is justified under a constant-quality assumption, i.e. an assumption that the quality of a dwelling does not deteriorate or improve drastically over time.

We opt for using a repeat sales method based on the specifications described in Case

[12]This is the process of transforming the attributes to create new attributes which capture domain-knowledge of the prediction problem. An example could be creating an attribute for the existence of an elevator in the dwelling's building *and* the dwelling being in the third storey or above.

and Shiller (1987). Although there are other implementations of the RSM[13], we note that the RSM by Case and Shiller (1987) is deemed the favourable implementation both in academia and for industrial applications (Balk et al., 2011; S&P Dow Jones Indices, n.d.). The model is stated as a weighted least squares (WLS), with the logarithm of the ratios of the selling prices as the endogenous variables and the selling times as exogenous variables. The full implementation details are left for Appendix A.3.

We note that we only consider condominiums when constructing the indices. This is due to our *historical transactions dataset* missing data regarding common debt and the significant share of common debt in cooperatives, as described in Chapter 3.1 and Chapter 3.3, respectively. Although this may lead to partially biased indices, we find that this alternative provides the better utilisation of our dataset.

Model Rationale and Challenges

The repeat sales method (RSM) has the major advantage of isolating actual increases in the price of a dwelling without requiring detailed information about the characteristics of individual properties. As discussed in Chapter 2, there are several explanatory variables not included in our dataset. In theory, the RSM provides unbiased results of the price increases, by controlling for location at the finest level, that is to say, the specific address.

A considerable weakness of the RSM is that, as it does not require the measurement of the dwelling quality or characteristics, it places the assumption of constant quality for individual dwellings across time. In the case of our dataset, spanning more than a quarter-century, we are prone to encounter dwellings that have been upgraded or altered in some significant form. This will possibly bias our results. However, by using the RSM as by Case and Shiller (1987), the repeat sales with longer time spans between them are potentially attributed less weight in the estimation of the model.

4.4 Comparable Pricing and Stacking of Models

In this section, we combine the aforementioned concepts and methods to complete the description of our AVM. In particular, we describe the procedure for selecting the comparable transactions and the stacking of individual methods.

Comparable Pricing

The ensemble learning submodels of the AVM are trained separately for each value estimate. That is, the ensemble learning methods are trained on *comparable recent sales*, which are selected based on the location and type of the dwelling. Specifically, we select the n geographically nearest transactions[14][15] of dwellings that are apartments, or non-apartments, depending on the unit type we aim to value. Further, we add a *ranking* variable to these transactions, i.e. rank $\in [1, 2, \dots, n]$, which is a proximity measure, increasing with distance from the target dwelling.

The comparable pricing approach enables the ensemble methods to explicitly recognise geographically close, recent, transactions when valuing a dwelling. We note that, as the comparable transactions are selected from the enhanced transactions dataset, they are all from August 2016 or later. Therefore, we do not eliminate any transactions based on the sold date when selecting the comparables.

[13] Two prominent alternatives for creating real estate indices are by Bailey et al. (1963) and Calhoun (1996), which only vary in their assumptions of the heteroscedasticity of the underlying regression problem.

[14] $n = 10000$, if the dwelling is an apartment, and $n = 2000$ if not. The distinction is made due to data availability of transactions for the two types of dwellings.

[15] Note that distance between two points in spherical geometry is not given by the Pythagorean formula, but by the Haversine formula. See p. 159 of Sinnott (1984) for a description of this formula and the rationale.

Stacking of Models

In addition to using XGBoost as an underlying method, we leverage it as a stacking method, or *model-stacker*. As described in the introduction to Chapter 4, XGBoost is used as a meta-estimator with both the exogenous variables and the value estimates from the individual models as input. This stacking procedure is described in detail in Procedure 2 below.[16]

PROCEDURE 2: The automated valuation model

1) Let the dwelling, whose current selling price we wish to predict, be denoted by u.
2) Select n comparable transactions, as described above, and denote the set of these transactions as *training data* or X
3) For each model $m \in \{$XGBOOST, RANDOM FOREST, EXTRA TREES, BAGGING PREDICTOR$\}$:
    a) Divide the training data into $k = 5$ random subsets of equal size, denoted by X_i, $i = 1 \ldots k$. For each of the subsets X_i:
        i) Fit m to training data *not* in X_i to get a fitted model m_i
        ii) Use m_i to get estimates of sold price of data in X_i. Denote these estimates by $\hat{P}_i$, and extend the training data X_i with $\hat{P}_i$.
        iii) Use m_i to get an estimate of the sold price of u. Denote this estimate by $\hat{u}_i$
    b) Average the k predictions of the price of dwelling u to get $\hat{u} = \frac{1}{k}\sum_{i=1}^{k} u_i$, and extend the dwelling data u with $\hat{u}$.
4) For each data point in the training data X and for the dwelling u
    a) Find all previous sales for the dwelling.
    b) Use the Repeat Sales price indices to generate price estimates based on each of the previous sales.
    c) Extend the data for the relevant dwelling with an average of the Repeat Sales price estimates, as well as the number of predictions. If there are no previous sales we extend the data with 0 and 0.
5) Finally, fit another XGBOOST model to the, now extended, training data X and use the model to predict a selling price for u.

Model Rationale and Challenges

Using estimates from a diverse set of estimators enables our AVM to deliver robust results with high predictive power. Although it is a contemporary approach in the field of econometrics, the idea of stacking different ensemble learning methods is gaining traction both in academia (Campos, Canuto, Salles, de Sá, & Gonçalves, 2017; D. Wolpert & Macready, 1996) and in international data science competitions (Adam-Bourdarios et al., 2015; Alves, 2017; Kaggle, 2018). In essence, we choose a powerful stacking method, namely XGBoost, due to its ability to detect when each base model performs well or poorly, and combine the underlying predictions correspondingly. Stacking is highly effective when the underlying models are diverse, and we do this by selecting both an array of ensemble learning methods and a repeat sales method.

Nevertheless, there are some drawbacks of combining stacked generalisation with comparable market analysis. By applying comparable market analysis we require one instance

[16]We refer the interested reader to our Python implementation of steps 3-5 of Procedure 2, which can be found in Appendix A.7.

of the model to be trained for each value estimation, and by stacking multiple individual models, each such estimation becomes increasingly complex. Specifically, the model encompasses five ensemble learning method, of which four are trained five times. The combined effect of these choices is that of increased model training time. We analyse and discuss the practical implications of this in Chapter 5.5.

4.5 Out-of-Sample Prediction

As our motivation for the development of the algorithm is the real-world application, we seek to train and evaluate our model using realistic data. Therefore we opt for an evaluation of the model's *out-of-sample* predictions, that is the model's predictions on unseen data. To achieve this, we partition our datasets as described below.

When evaluating our model we divide our data into two disjoint sets, one which is used as a *training set* and the other being *test set*. We define the training set and the test set as the transactions before and after a given point in time, denoted as the *split*. By training our model on data from the training set and evaluating it on the test set, we simulate a real-life scenario where the model is trained on data recorded up to a given day and produces value estimates on possible transactions the next day. We perform this split on a monthly basis, while in practice one would update the data supply every day. Hence, our results should be interpreted as a conservative performance estimate of the out-of-sample predictive power of the model. When evaluating our model in Chapter 5, we make three such partitions using the following *splits*:

i) January 1st 2018, 00:00

ii) February 1st 2018, 00:00

iii) March 1st 2018, 00:00

For each dwelling in the test set, we choose the comparable transactions from the corresponding training set, as discussed in Chapter 4.4. Similarly, when applying the RSM to predict previously sold dwellings in the test set we use indices built solely on the training set. As we have the attribute *sold month* in our dataset, we set this to be equal to the previous month for all dwellings in the test set. Thus, when making predictions with both the ensemble learning methods and the RSM, we are predicting the selling price as if the sold date was the first day of the month.

5

Results and Discussion

In this chapter we analyse and discuss the performance of our automated valuation model and its submodels, in addition to discussing important model choices and potential challenges. We begin, in Chapter 5.1, by evaluating the performance of our AVM by comparing its precision to estate agents, as well as the American industry leader for real estate AVMs, namely Zillow[17]. Then, in Chapter 5.2 we justify the use of *stacked generalisation* in our model by comparing the performance of the AVM to the underlying submodels. Next, in Chapter 5.3 and Chapter 5.4 we make empirical justifications for the choice of submodels. Finally, in Chapter 5.5 we discuss the strengths and weaknesses of our AVM in the light of the presented results.

5.1 Evaluation of the Performance of our AVM

To analyse the performance of our AVM, and discuss its potential commercial applications, we here examine the distribution of the value predictions of our AVM and compare it to estate agents and industry leaders.

Our AVM achieves an overall median absolute percentage error (MdAPE) of 5.4 % in January, February and March 2018, as illustrated in Table 7. When comparing the performances of our AVM with that of the estate agents, by analysing Table 7 and Table 8, we find very similar performances. Both the quantiles and the MdAPEs are close to identical, while the mean absolute percentage error (MAPE) of our AVM is slightly better than the MAPE of the estate agents. We note that the tables are not directly comparable, as the transactions are from different time periods. The difference in time periods is due to a lack of data from *Eiendomsverdi* for transactions in 2018, and insufficient training data in our enhanced transactions dataset to produce value estimates in 2016 and 2017.

Table 7: The share of the predictions of the **automated valuation model (AVM)** that are within 5 %, 10 % and 20 % of the correct value and the median absolute percentage error (MdAPE) and mean absolute percentage error (MAPE) - out-of-sample performance for Q1 2018

	Within 5 %	Within 10 %	Within 20 %	MdAPE	MAPE
Q1 2018	46.9 %	76.4 %	96.3 %	5.4 %	7.2 %

Table 8: The share of the predictions of **estate agents** that are within 5 %, 10 % and 20 % of the actual selling price and the overall median absolute percentage error (MdAPE) and mean absolute percentage error (MAPE) - Data from *Eiendomsverdi*, including 15 786 transactions in Oslo in 2016 and 2017

	Within 5 %	Within 10 %	Within 20 %	MdAPE	MAPE
2016/2017	47.8 %	74.0 %	96.5 %	5.3 %	7.6 %

Further, we compare our AVMs performance with the performance of Zillow, the American industry leader for real estate AVMs. Zillow creates similar value estimates for more than 100 millon dwellings in the U.S., and publish their aggregated performances for a handful selected cities. Table 9 illustrates some of these performances.

[17] `https://www.zillow.com/corp/About.htm`. Retrieved May 27th 2018

Table 9: The share of Zillow's Zestimates [1] which are within 5 %, 10 % and 20 % of the actual selling price and the overall median absolute percentage error (MdAPE). Provided for a selection of U.S. cities.

	Within 5 %	Within 10 %	Within 20 %	MdAPE
Baltimore, MD	54.6 %	73.6 %	85.1 %	4.3 %
Boston, MA	53.9 %	78.1 %	89.9 %	4.5 %
Charlotte, NC	52.3 %	72.3 %	84.1 %	4.7 %
Chicago, IL	56.7 %	76.8 %	88.5 %	4.1 %
Cincinnati, OH	46.4 %	68.4 %	84.0 %	5.5 %
Cleveland, OH	44.8 %	65.4 %	80.2 %	6.0 %
Dallas-Fort Worth, TX	33.1 %	57.2 %	79.6 %	8.2 %
Denver, CO	65.5 %	86.1 %	94.5 %	3.3 %
Detroit, MI	50.9 %	71.9 %	85.6 %	4.8 %

1) The data is gathered from www.zillow.com/zestimate/ on May 27th 2018.

As we see in Table 9, the MdAPEs of Zillow vary between 3.3% and 8.2%, which is both considerably better and worse than our model's performance. In addition, we observe that none of the cities have a higher amount of estimations within 20% of the selling price than Denver, whose value is 94.5%. Here our model clearly outperforms Zillow, with above 96% of the estimations being within 20% of the selling price. In addition, Zillow has data describing roughly 1-2 millions dwellings in each of the presented cities, which is a considerably larger amount of training data than we have had access to. It is clear that one needs to interpret the comparison of the performances with caution, due to the obvious differences between markets and data availability. However, the comparison does illustrate some of the potential commercial value of our AVM.

On a final note, we remark a behavioural finance aspect of the comparisons made in this section. The predictions of both the estate agents and Zillow are made (and published) prior to the selling price being established, and thus, are likely to influence the buyer and seller. We believe that this can have two effects: Estate agents might aim to price a dwelling lower than the expected selling price to attract many potential buyers, and hence start a bidding war. We observe that roughly 61 % of value estimates (asking prices) by estate agents are lower than the final selling price. In addition, by the *anchoring-and-adjustment heuristic*, as presented in psychological literature[18], such reference points are prone to bias the transaction participants, and the final selling price is likely to be insufficiently adjusted away from the *anchor*.

5.2 Evaluation of the Stacked Model

To justify the application of stacked generalisation in our AVM, we perform a thorough evaluation of the effect of stacking. First, we compare the accuracy, measured in MdAPE, of the stacked model with that of the selected ensemble learning methods, the repeat sales method (RSM) and a simple OLS-based HPM. Then, we analyse the performance against training time of our models, to find the optimal number of comparables to use at each valuation. Finally we reason for the choice of stacking the four ensemble learning methods, by analysing and discussing the correlations of the out-of-sample residuals of the underlying ensemble learning methods and the model-stacker.

Comparison of the Stacked Model to the Individual Models

To justify the use of stacked generalisation in our AVM, we compare the accuracy of the stacked model to that of the underlying models. The comparison is done out-of-sample for January, February and March 2018, as well as in aggregate for the three months. The

[18]Seminal works by Slovic and Lichtenstein (1971) and Kahneman and Tversky (1972) introduce and discuss this heuristic, while Northcraft and Neale (1987) considers it empirically in the setting of the residential real estate market in Tuscon, Arizona, finding that the *"subject populations were significantly biased by listing prices"*.

results are presented in Table 10. We observe that the automated valuation model (AVM) performs significantly better than the individual models on average. We also note that the RSM performs considerably poorer than all the other underlying methods. The inclusion of this method has been justified theoretically in Chapter 2.3 and Chapter 4.3, and will be justified empirically below.

Table 10: The median absolute percentage error (MdAPE) of the automated valuation model (AVM) compared to the ensemble learning methods and the repeat sales method (RSM), as well as an OLS-based hedonic pricing method - out-of-sample performance Q1 2018.

	Ensemble Learning				Repeat Sales[1]	Traditional	Stacked Model
	BP	RF	ET	XGB		OLS	XGB-S
2018 - January	6.23 %	6.31 %	6.21 %	6.13 %	8.93 %	8.95 %	**5.49** %
2018 - February	5.60 %	5.70 %	5.71 %	5.56 %	8.86 %	8.85 %	**4.94** %
2018 - March	5.81 %	5.47 %	5.64 %	5.90 %	9.42 %	8.46 %	**5.50** %
Total	5.95 %	5.90 %	5.92 %	5.99 %	9.05 %	8.77 %	**5.36** %

1) The MdAPE for the RSM is only calculated for the dwellings which have previous sales, which constitutes 77% of the dwellings in the test set.

We further examine the correlations between the individual models residuals, to compare their value estimates and discuss the potential gain of stacking. Figure 8 illustrates the correlations between the ensemble learning methods residuals out-of-sample.[19] An apparent observation is the high positive correlation between the individual methods. We choose to include all the four methods since no single method yields strictly better results, and believe that the stacking algorithm should be able to choose the optimal combination of them. It is clear from Figure 8 that the model-stacker (XGB-S) is able to detect relationships in the data not captured by the individual methods, hence yielding better out-of-sample results.

Figure 8: The Pearson correlation coefficients of the residuals of the automated valuation model (denoted by XGB-S) and the submodels (XGBoost (XGB), bagging predictor (BP), random forest (RF) and extra trees (ET)). A lighter colour, and higher positive number, indicates a higher positive correlation. We observe high correlations between the residuals.

Selecting the Number of Comparable Transactions

Figure 9 displays how training time and accuracy, represented by the MdAPE, increases with the number of comparable transactions for each valuation, for February 2018. We observe that the training time is more or less linear in the number of comparable transactions. With only 50 comparable transactions, the model is able to score an out-of-sample MdAPE

[19] We do not include the RSM here, due to it only covering 77% of the transactions.

of about 6%. The further gain towards 5% MdAPE, however, is computationally burdensome. We observe no further improvements after including 10 000 comparable transactions, and choose this as the number of comparables in our final model, due to training time constraints.

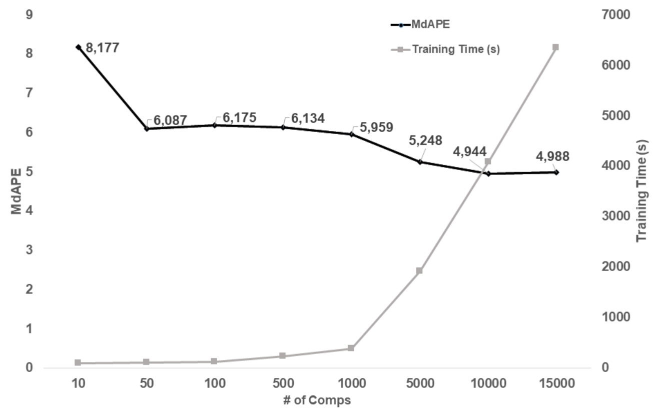


Figure 9: Accuracy and training time[20]of the automated valuation model (AVM) for varying number comparable sales. Results from February 2018 (470 transactions). *Note that the x-axis is not linear in number of comparable sales*

5.3 Evaluation of the Attribute-Based Pricing Methods

Here we assess the choices made during the selection and tuning of the attribute-based pricing methods. We provide empirical reasoning for the choice of decision tree-based methods, by comparing the out-of-sample performance of our selected methods with a HPM based on ordinary least squares (OLS). Subsequently, we analyse the importance of the individual attributes in one of our ensemble learning methods and discuss the absence of substantial feature engineering of our data.

Boosting and Bagging vs. OLS - Out-of-Sample Accuracy

Table 10 illustrates the out-of-sample performance of the selected ensemble learning methods and our AVM, compared to a regular OLS-based HPM[21]. We note that the OLS-based HPM uses the same set of attributes as the ensemble learning methods. The performance of the ensemble learning methods is superior to OLS in each of the test months, as they outperform the HPM by more than 30 % on average. We acknowledge that the HPM might suffer from the lack of feature engineering. That is, it might obtain better results with transformation and grouping attributes in a pre-processing step. However, we find that performing a feature engineering process of a HPM to be outside of the scope of this paper, and rather note that the superiority of the ensemble learning methods justifies their inclusion in our model.

Attribute Importance

As introduced in Chapter 4.2, AVMs using ensemble learning are harder to interpret than OLS-based HPMs. However, this can be addressed by analysing the models' underlying *decision trees*, either by simply displaying the individual trees, as illustrated in Appendix A.2, or by analysing aggregated statistics from each model, such as *attribute importance*. This is a score given to each attribute based on its importance in increasing the model's performance. There are several methods for calculating this based on the aggregate frequency of

[20]The training time is achieved with our implementation as it is described in Appendix A.6.
[21]See p. 50 of Balk et al. (2011) for a presentation of this method.

occurrence and placement of the attributes in the decision trees[22]. Here we give an example of analysing attribute importance for the XGBoost-algorithm using the definition given in Chen and Guestrin (2018).

For the XGBoost-algorithm, attribute importance is given as the share of predictive power brought by including a particular attribute in the decision trees (Chen & Guestrin, 2018). These scores are denoted as $\alpha_i \in [0,1]$, where $\sum_i \alpha_i = 1$.

Since we build a separate model for each dwelling, we analyse the model built for one particular apartment, sold in Oslo in March 2018. The apartment is a condominium situated in the district Gamle Oslo and has 82 USM. It was constructed in 2013, has four rooms and is located on the fifth floor. Figure 10 illustrates the attribute importance of the most important attributes when building the decision trees for the XGBoost-algorithm for the discussed apartment.

We observe that the numerical attributes are considered to be far more important than the categorical attributes when building the decision trees. Specifically, the location, represented by longitude and latitude, size of dwelling represented by USM and date of the transaction, represented by the *Days since sale*-attribute, are the most important attributes. We observe some variables with 0.0 attribute importance score, such as districts far from the selected dwelling.

We find these observations to be reasonable. The numerical variables have a larger number of possible *cut-points* than the binary categorical variables and therefore can be used more often to split the dataset into good partitions. The variables that have received the highest attribute scores are the ones most often associated with selling prices of dwellings. The fact that some attributes receive a score of 0.0 for one particular model does not hinder the attribute from being important in another model. This illustrates the model's ability to adapt to the underlying data.

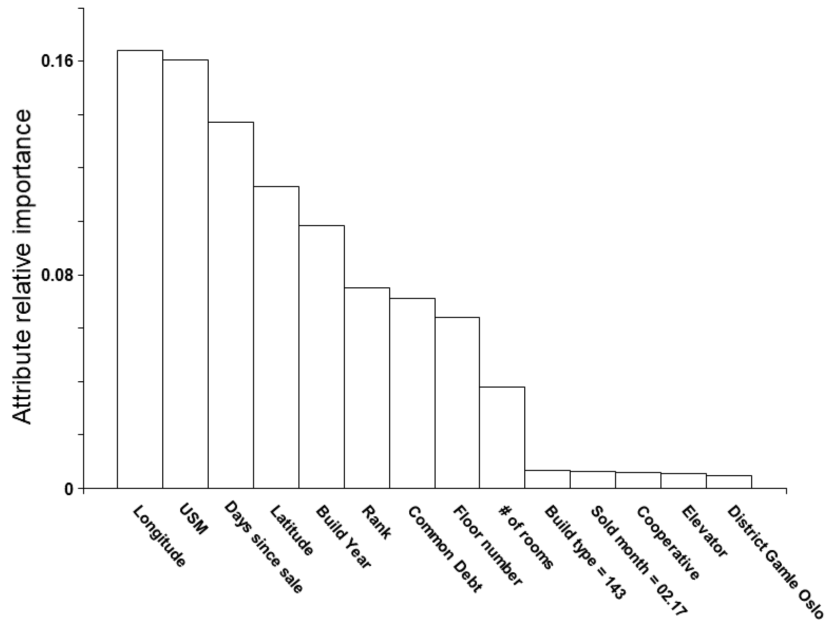


Figure 10: The relative importance of the most important attributes when valuing the dwelling specified above, calculated as a ranking score of the underlying XGBoost-model.

[22]We refer to Chapters *10.13* and *15.3* of J. Friedman, Hastie, and Tibshirani (2001) for an overview of variable importance estimations.

A Summary of the Explored Attribute-Based Pricing Methods

The choice of using ensemble learning methods as our attribute-based pricing methods was made on an extended exploratory analysis of the state-of-the-art machine learning techniques. Here we briefly summarise the choices explored, and refer to Appendix A.1 and Table 10 for comparable results.

First, we acknowledge that traditional hedonic methods, such as those based on ordinary least squares (OLS), might yield favourable results in some situations. Given a smaller dataset, less training time or a desire to study more familiar statistics such as condition numbers, coefficients of determination or other statistical measures, we believe that traditional hedonic methods could be applied with decent precision. We refer to Table 10 for comparable results.

Second, we believe that there are potential gains from a further investigation of ensemble learning techniques. In particular there are promising techniques like *CatBoost* by Yandex (2018) and *LightGBM* by Microsoft's Ke et al. (2017) that have gained attraction in the Kaggle-community recently. The drawback of these techniques is usually their computational burden, which is also the reason why we have opted not to include them in our AVM.

Third, we considered, and explored, the use of artificial neural network (ANN) as a submodel in our AVM. ANNs have recently gained much attention in both commercial and machine learning communities. The findings from our exploration, however, is that these techniques are considerably hard to apply for an econometrician, due to their large number of hyperparameters and low interpretability. We provide a brief summary of our experiences developing an ANN as a submodel for our AVM, and a discussion of why the model was not included, in Appendix A.1. In short, we found the process of designing the network to be a highly specialised engineering process, with extremely many design choices and only a handful of established guidelines. Whether or not these techniques can yield superior results is therefore left as an open question.

5.4 Evaluation of the Repeat Sales Method

As illustrated in Table 10 the RSM has a considerably poorer performance than all the other individual models, and similar performance to the OLS-based HPM in Table 10. One might therefore question the inclusion of the model in the AVM. However, as argued in Chapter 2.3 and Chapter 4.3, the RSM is trained on a separate dataset and aims to capture different market movements than the ensemble learning methods. We therefore believe that the inclusion of the RSM in the AVM has a benefit. To empirically justify this decision, we run the AVM as described in Procedure 2, but without Step 4), and compare the performance to that of the AVM's. We present this comparison in Table 11, and note that the MdAPE increases by 8% overall when excluding the RSM from the model. This indicates a considerable benefit of including the RSM. We note that the number of previous sales for the dwelling is included in the training data for the model-stacker, in addition to RSM's prediction. This attribute is also likely to yield significant explanatory power, as dwellings sold often might have special characteristics.

Table 11: The share of the predictions that are within 5 %, 10 % and 20 % of the selling price and the median absolute percentage error (MdAPE) and mean absolute percentage error (MAPE) using the **automated valuation model (AVM) with and without repeat sales method (RSM)** - out-of-sample performance for Q1 2018.

	Within 5 %	Within 10 %	Within 20 %	MdAPE	MAPE
AVM incl. RSM	46.89 %	76.36 %	96.31 %	5.36 %	7.17 %
AVM excl. RSM	45.52 %	73.23 %	95.08 %	5.81 %	7.43 %

5.5 Model Discussion

In this section we discuss and critique our model, both with regards to the hypothesised model rationales in Chapter 4 and in the context of the empirical results provided above.

We begin by reiterating three of the main conjectured strengths of our model and elaborate on these in turn. First, we argued that applying stacked generalisation would allow our model to combine the predictions of several submodels with improved results. In the results Table 10 and Table 11, we find this to be the case, as the stacked model's performance clearly improves upon the individual methods.

Second, while developing the model, we hypothesised that the use of comparable market analysis would be beneficial as the model would benefit the most from nearby transactions. However, as Figure 9 illustrates, we observed that including up to 10 000 transactions, with the additional ranking variables, yielded superior results.

Third, we emphasise the benefits of ensemble learning techniques within the field of econometrics. The non-parametric nature of the methods makes them applicable to a wide range of tasks. Even though ensemble learning methods are novel tools in econometrics, they often provide superior results to traditional methods and therefore becoming increasingly popular.

The major drawback of applying stacked generalisation in a model is the training time it demands, due to the required predictions made on the training data by the individual folds. Depending on the quality of the implementation[23] the model runs in 60-400 seconds, for a single prediction. For practical applications, this imposes certain constraints on the design of the service. However, we argue that the value of a strong model exceeds the drawback of time complexity. Furthermore, we note that the model can easily be adapted to the demands of training time. In practice, several instances of the model could be run in parallel to give the user improved results as the different instances are completed.

A challenge for the non-parametric ensemble learning methods is that they require sufficiently diverse training data to achieve strong out-of-sample predictive force. Specifically, they do not generalise well outside the observed range of attribute values, as they do not make any prior assumptions about the underlying data. However, as we discovered in the results above, the model is not only able to predict with high accuracy, but also with a strong precision. That is, that in addition to a low MdAPE, it has a high share of predictions within a 20% deviation. This is compelling evidence of the model's ability to generalise well to the given dataset, and consequently, its suitability for use in AVMs.

Another drawback is the model's lacks of transparency and, to a certain extent, its lack of underlying model assumptions. Although we have explained how the model may be visualised using attribute importance and by displaying the underlying decision trees, we do concede that this may be insufficient for use in public policy. A major transition from traditional HPMs to ensemble learning AVMs, is the shift from a *theory-driven* to a *data-driven* approach, where the econometrician defines fewer of the model's assumptions.

[23] Quality of the implementation depends largely on choices regarding programming language, parallelisation and pre-processing. See Appendix A.6 for an overview and discussion of our implementation.

6 Conclusion

6.1 Motivation, Findings and Remarks

The aim of this study was to develop a commercially viable automated valuation model (AVM) and to aid in progressing the field of real estate finance. To assess the model, we sought to implement and evaluate this model on the residential real estate market in Oslo. By *stacking* four different ensemble learning methods and the repeat sales method (RSM) into an AVM, and evaluating it on transaction data for real estate in Oslo, we achieve these goals. Specifically, we train the model on data comprising 25 years of transactions and two years of enhanced data including key dwelling attributes and test the model out-of-sample on all transactions in Oslo in the first quarter of 2018. The model estimates the value of the 1 979 dwellings sold in Q1 of 2018, with a median absolute percentage error (MdAPE) of 5.4%. Comparing this to the precision of estate agents in Oslo and Zillow, the industry leader in the U.S., we find our AVM to have produced highly promising results.

This study has two main implications, regarding the performance our AVM and the methodological techniques applied to construct it. The comparison of our AVM with the accuracy of Zillow indicates that we achieve a sufficient precision to be commercially viable. We seek to continue to work with Alva Technologies to improve upon their models, and create a service that provides access to a publicly available AVM for Oslo. We also find the novel combination of ensemble learning and real estate indices to be a substantial addition to the current field of research, which is both fragmented and to some degree hidden by private industrial actors.

Two aspects of the model which were the subject of concern during its development were its computational complexity and degree of interpretability. To achieve our goal of making the AVM valuable in real-life applications, we opted to design a model with substantial computational complexity. At the same time, we have shown that one could significantly decrease this complexity, at the price of a lower model precision. We have also addressed how available tools have been designed to increase the degree of interpretability of the ensemble learning methods, and believe that they are sufficient for the main applications of an AVM.

6.2 Further Research

To conclude, we wish to focus on possible improvements that can be made to the AVM. We begin by briefly discussing alternative applications of the AVM, as well as the corresponding adaptions which would be needed to be made. Further, we discuss enrichment of the source data, as well as enhancements of the individual submodels and the overall model design.

Applications

As introduced in Chapter 1, there are several direct applications of AVMs, including the needs of homeowners, policymakers and loan providers. But with minor adjustments, we believe that the model could have applications within other realms. We briefly exemplify two such possibilities: general asset pricing and real estate development. Traditional asset pricing methods are built on the two polar approaches of *absolute* and *relative* pricing (p. 8 Cochrane, 2009), which price assets by measuring exposure to fundamental sources of risk and by comparing similar asset prices, respectively. We view the use of AVMs for both the methods as a feasible approach, as the transactional data which exists for most assets is often of higher quality than that of the residential real estate markets. We also believe that leveraging the AVM for use in real estate development is highly feasible. By calculating *marginal attribute prices*[24], one could estimate the marginal value of individual attributes given a specific dwelling.

Enrichment of source data

One aspect to which we have paid little attention is that of extending the primary datasets which we have received, as we have deemed this to be outside of the scope of our work. We do, however, believe that there are several industrial and government actors who have potentially beneficial data sources, such as *Finn.no*[25] and *The Norwegian Mapping Authority*. These data could be *structured data*, regarding dwellings' characteristics, but also complex *unstructured data*, such as images and natural language descriptions of dwellings.

Structured data, if acquired for the dwellings in the enhanced dataset, would likely be easy to incorporate into the model, and would only require retuning the hyperparameters of the ensemble learning techniques. If data regarding the dwellings in the historical transactions were acquired, then more complex models for the RSM, such as by Abraham and Schauman (1991), may be applied with potential success.

In the case of acquiring unstructured data, one would need to pre-process it before potentially including it in the model. The development and implementation of such processing steps are out of the scope of most econometricians' work, but we consider a brief example here; The application of keyword extraction from classified advertisements[26] could yield additional attributes for our model, further increasing the model's performance[27].

In addition to enriching and extending our data, we note that improving the quality and quantity of the existing datasets is likely to be fruitful to the model performance.

Alternatives to the individual and stacking methods

The use of a repeat sales method in combination with the ensemble learning methods in an AVM is a novel approach and has yielded impressive results. We find the study of combining methods based on transactional data with methods based on attribute-specific data to be an interesting and little-explored field of research. Therefore, we suggest further exploration of the combinations of such methods.

It is clear that there are an array of available regression methods, particularly from the realm of machine learning, that can be applied to AVMs. The amount and variety of different regression techniques are increasing, and we have implemented and tested some of the most common, as discussed in Chapter 5.3. We believe that there are exciting developments, particularly within the field of *boosting* and *artificial neural networks*, that can be beneficial for future AVMs.

[24]The marginal price of an attribute for a given dwelling could be estimated simply by adjusting the attribute's value by an appropriately small value (in the case of numerical attributes) or by changing the attribute's label (in the case of categorical attributes). The change in the model's prediction would yield an estimate of the *marginal attribute price* for the given attribute.

[25]Finn.no is a Norwegian classified advertisements website with advertisements for nearly all dwellings sold on the free market.

[26]Examples of relevant keywords may be "kitchen renovated in 2016", "balcony", "needs renovation", etc.

[27]The interested reader may view Bharti and Babu (2017) for an overview over relevant approaches for keyword extraction approaches, as a processing procedure for unstructured data in various domains.

Appendix

A

A.1 An Evaluation of Artificial Neural Networks as AVMs

Background

ANNs are a class of machine learning techniques which models learning tasks by combining a collection of units, known as *artificial neurons*, each of which applies a simple threshold function, known as *activation functions*, to its input. These neurons are typically organised in layers, with connections between neurons in adjacent layers which can transmit the outputs of a layer as inputs to the following layer, adjusted by a certain *weight*. The activation functions, the learning method and the structure of neurons and layers are determined by the user, while the model learns the weights of the network from the relevant training data.

Although ANNs are known to be able to approximate any finite mathematical function[28] and have been in existence for several decades, they can be hard to adapt and apply to many learning problems. This is due to their complex training procedure and non-parametric structure.

Implementation

To design our implementation of an ANN we relied heavily on Bengio (2012) and Goodfellow, Bengio, and Courville (2016) as conceptual guides, as well as a plethora of forums in the data science community, including Kaggle and TowardsDataScience[29]. To implement the design, we used the **MLPRegressor**-package by *scikit-learn* for Python. This package provides a sufficiently broad toolkit for our exploration.

There are a few generally accepted practices for applying ANNs, such as normalisation of input data and the use of mini-batches[30], which we apply. For the model's hyperparameters we use a selection of default and recommended parameters[31], as well as an educated guess combined with a grid-search and cross-validation. The most challenging task we faced here was the structure of the neurons, that is, the number of layers, the number of neurons per layer and the connections between each layer. The choices here are numerous, and the use of a grid-search alone to determine the appropriate choice is infeasible due to the exponential increase in computational requirements. Our final model is the most stable result, with fairly consistent results. We provide the selected hyperparameters in Table 13.

Results and Discussion

The results of our work are presented in Table 12, which also includes the stacked AVM for comparison. We see that our implementation of an ANN performs far poorer than the AVM. Although we believe there to be superior implementations of ANNs for this problem, we can not determine it by any structured approach and have to rely largely on guesswork and grid-searches. When considering the solutions used in several Kaggle-competitions, we find that ANNs are prevalent as submodels, but rarely used without a model-stacker. Furthermore, researching recent literature[32] reveals that the inherent struggle of training ANNs is an established issue.

[28]This is known as the universal approximation theorem (Cybenko, 1989).

[29]`www.towardsdatascience.com`. Retrieved May 27th 2018

[30]Mini-batches are randomised subsamples of the training data, which lets the network train faster and with less memory.

[31]Neurons' activation functions are set to *ReLU* (Glorot, Bordes, & Bengio, 2011) and the weight optimiser is set to *Adam* (Kingma & Ba, 2014). Both these choices are well-established for regression problems, although even these have several prominent alternatives.

[32]See Chapter 5 of M. A. Nielsen (2015) for a conceptual understanding of some of the challenges in designing ANNs, and Glorot and Bengio (2010) for a more technical treatment of the reasons.

Our conclusion from this exploration is that, although ANNs are universal approximators, they require a great deal of experience to apply with success, and are currently closer to an art-form than an engineering process. We believe ensemble learning methods to be more consistent and more straightforward to apply, and prefer these for regression problems in the field of econometrics. Although we do not disregard the capabilities of ANNs, we do not view them as ripe for use by econometricians in an as intuitive and statistically-grounded manner as many other machine learning techniques.

Table 12: The share of the predictions of an **artificial neural network (ANN)** that are within 5 %, 10 % and 20 % of the correct value and the median absolute percentage error (MdAPE) and mean absolute percentage error (MAPE). Also, our AVM (for comparison) - out-of-sample performance for Q1 2018.

	Within 5 %	Within 10 %	Within 20 %	MdAPE	MAPE
AVM	46.89 %	76.36 %	96.31 %	5.36 %	7.17 %
ANN	24.20 %	46.18 %	77.30 %	10.96 %	13.99 %

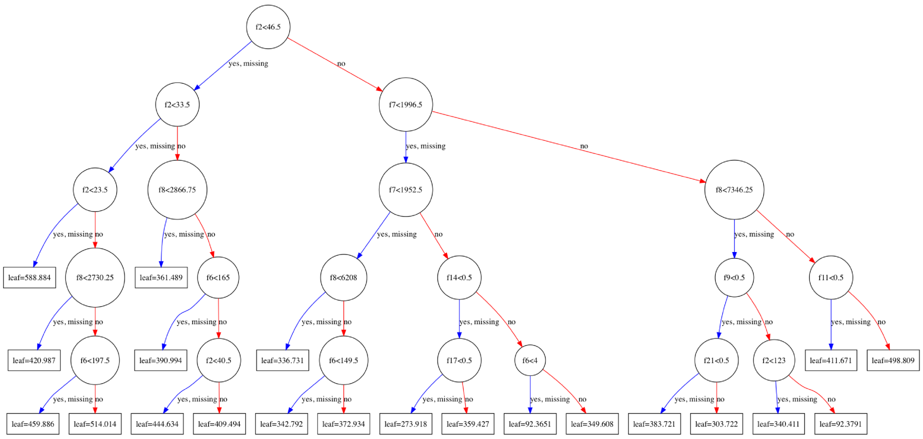
Table 13: The hyperparameters of our artificial neural network (ANN) - descriptions and selected values.

Variable	**Description**	**Selected Value**
Optimiser	The solver used for weight-optimisation	Adam[1]
Activation function	The activation function used in the neurons	ReLU[2]
# of hidden layers	The number of layers	4
Hidden layers	The number of nodes per layer	[64,64,32,32]
Maximum number of iterations	The maximum number of iterations of the training data	1 000
Learning rate	The step-size used in updating the weights	0.01
Alpha	A regularisation parameter, to prevent overfitting the data	0.0001

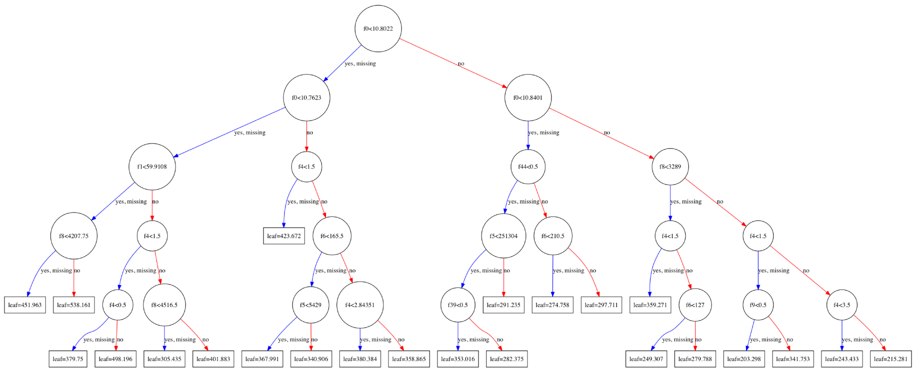
1) See Kingma and Ba (2014).
2) See Glorot et al. (2011).

A.2 Decision Trees for the XGBoost-Method

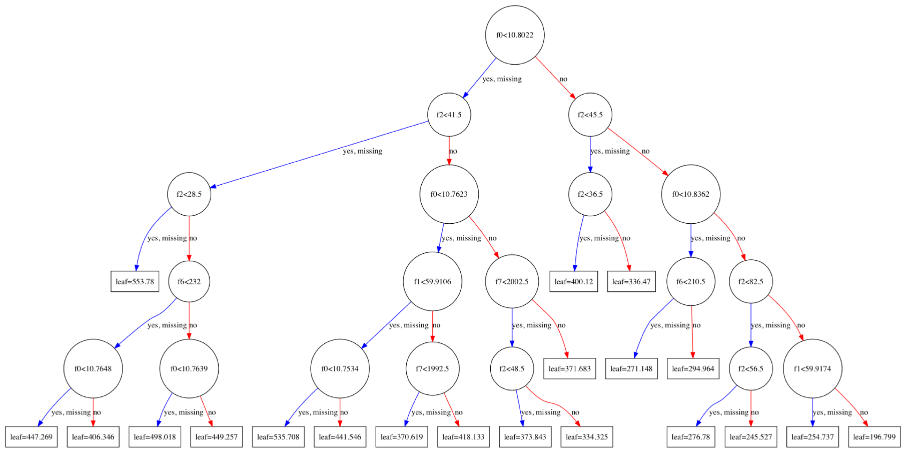
The XGBoost-method produces several binary decision trees, which are grown sequentially to improve on the previous tree's residual, as described in Chapter 4.2. Below we present an example of the three first trees grown for a randomly selected dwelling[33]. The attribute names are given as **f0**, **f1**, . . . due to the nature of our selected graphing tool. We note a few of the important attributes; **f0** is the longitude, **f1** is the latitude, **f2** is the usable square meters, **f4** is the number of rooms, **f6** is the days since sale, **f7** is the build year, **f8** is the common debt.



(a) Tree #1 of 1000



(b) Tree #2 of 1000



(c) Tree #3 of 1000

Figure 11: The first three decision trees grown for an instance of the XGBoost-method.

[33] A three-room, cooperative, apartment in the district of Østensjø.

A.3 Methodological and Implementation Details of the Repeat Sales Method

In this appendix we give a succinct presentation of the repeat sales method (RSM) which we introduced in Chapter 4. The RSM is implemented as developed by Case and Shiller (1987). We also present the rationale and procedure for applying *index smoothing* to our indices.

The RSM is implemented using a three-step regression. We first run a model-specific pre-processing, as noted and justified in Chapter 3. We treat multiple resales as independent observations, a practice recommended in the literature (See Shiller, 1991). Finally, dwellings which have been sold more than once are grouped in their corresponding district, as the indices are calculated for each unique district [34]. We then run the procedure given below

Procedure A.1: The Case Shiller repeat sales method method

1) In the first stage, the natural logarithm of the price difference of the sales in a given district is regressed on a set of indicator variables, one for each time period in the sample except the first. For each observation, the indicator variables are zero in every quarter except the quarters in which the two sales occurred. For the quarter of the first sale, the dummy variable is -1, and for the quarter of the next sale, the dummy variable is 1.
2) The residuals from the regression in the first stage are then squared and regressed on a constant term and the time between sales[35]
3) In the third stage, a weighted least squares (WLS) regression is run similarly to that of the stage one regression, weighted with the reciprocal of the square root of the fitted values in the second stage.

After running the above three-step regression on the transaction data, we obtain a WLS estimation. The exponential function is applied to the coefficients of the final estimation to produce an index. This index can then be used to adjust a previously sold dwelling's previous sold price to the last observed index observation, to predict today's selling price. Note that the RSM can produce multiple estimates for dwellings with numerous repeat sales. In that case, we average the estimates. After the application of the RSM as described above, we use an index smoothing step, which we describe next.

Index Smoothing

When developing indices one needs to determine the geographical and temporal granularity of each point in the index[36]. The econometrician is tasked with determining the optimal grouping of observations, such as to allot sufficient data in each group and to let each group contain adequately similar data. On the one hand, one seeks to allot sufficient data for each time period in each index. On the other hand, the indices should have sufficient geographical homogeneity and adequate temporal granularity. In our case, we considered the choice between creating a single monthly index for Oslo and creating quarterly indices for each district. Based on industry expertise we have opted to create *smoothed quarterly indices* for

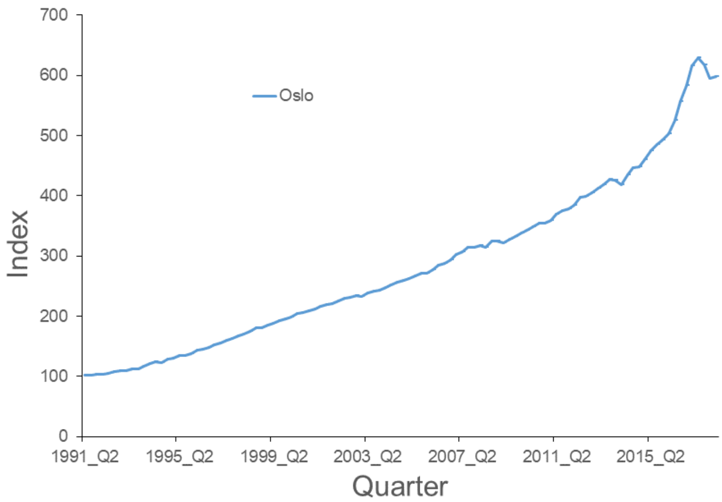
[34] We also produce an index for all of Oslo, for use in the index smoothing process described in below.

[35] This stage is used to create weights for the WLS. Two prominent alternatives for creating real estate indices by Bailey et al. (1963) and Calhoun (1996) vary merely in their assumptions of the heteroscedasticity of this regression stage. The former assumes no heteroscedasticity, while the latter assumes non-linear heteroscedasticity, and thus includes a term with time squared.

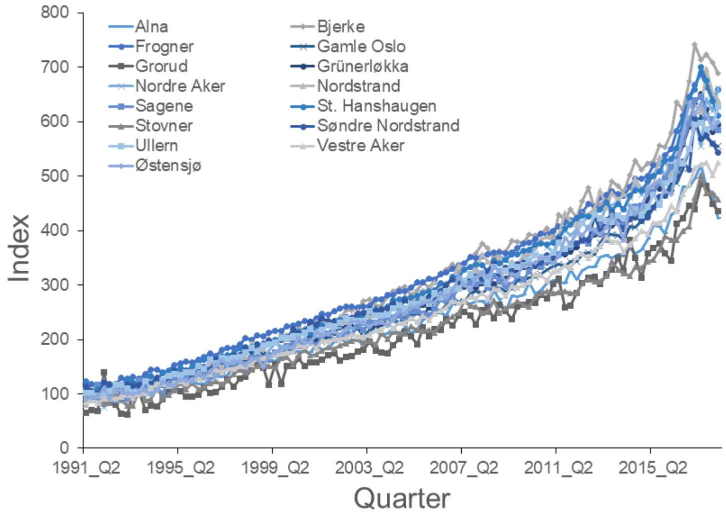
[36] For example, one may consider one index based on transactions grouped per week, created for each street in Norway, and another index grouped per year, created for all of Norway. It should be clear to the reader that the former would not allocate sufficient data for each point in the index, while the latter would group too much data in each index. Both would be unwise choices.

each district. That is, we smooth the indices with an aggregated Oslo-index by averaging the quarterly observations. This is done due to the sparsity of our dataset, particularly prevalent for the older data, as shown in Appendix A.5. The individual districts' indices, the index for Oslo and the smoothed indices are found in Appendix A.4.

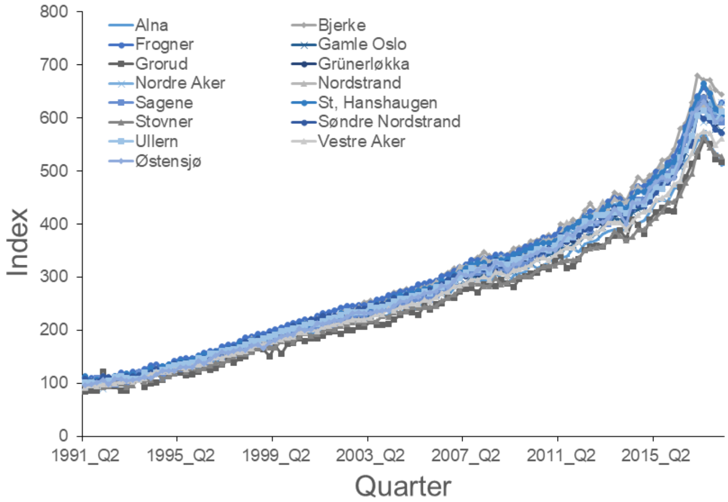
A.4 Plots of the Real Estate Indices by the Repeat Sales Method



(a) Aggregated index for Oslo



(b) Indices for each district



(c) Smoothed indices for each district

Figure 12: Real Estate Price Indices for Oslo from Q2 1991 to Q1 2018, we apply version (c) in our AVM.

A.5 Number of Transactions Per District and Quarter in Oslo

Table 14: Number of transactions recorded per district and per quarter, between 2nd quarter 1993 and 1st quarter 2018.

	Alna	Bjerke	Frogner	Gamle Oslo	Grorud	Grünerløkka	Nordre Aker	Nordstrand	Sagene	St. Hanshaugen	Stovner	Søndre Nordstrand	Ullern	Vestre Aker	Østensjø	Oslo
1991Q2	22	27	230	135	6	55	32	33	35	79	31	20	69	69	47	890
1991Q3	21	29	186	86	7	67	65	49	33	125	30	32	57	99	33	919
1991Q4	26	39	258	142	17	76	44	58	44	110	41	58	92	121	52	1178
1992Q1	17	33	188	73	7	51	26	41	22	64	22	70	60	114	25	813
1992Q2	22	17	158	129	6	54	35	35	30	51	22	57	72	89	18	795
1992Q3	25	22	170	99	9	48	26	40	20	90	32	23	78	82	41	805
1992Q4	17	39	186	140	12	95	54	50	18	97	38	42	76	80	36	980
1993Q1	19	19	175	150	4	48	25	39	28	73	15	23	39	76	30	763
1993Q2	23	26	217	83	8	40	37	37	23	81	25	17	78	72	30	797
1993Q3	36	37	248	80	11	80	53	65	49	115	42	48	72	114	35	1085
1993Q4	72	35	360	90	18	106	70	78	60	158	50	50	103	135	37	1422
1994Q1	25	26	205	108	6	75	49	42	39	87	39	27	64	88	38	918
1994Q2	48	49	257	163	18	86	46	66	60	116	47	76	97	103	46	1278
1994Q3	36	57	369	116	17	92	92	131	48	191	51	67	121	122	51	1561
1994Q4	44	51	359	113	19	152	74	119	59	188	56	68	107	143	77	1629
1995Q1	30	44	243	75	12	93	57	67	61	120	92	68	146	104	45	1257
1995Q2	63	53	299	85	16	90	48	88	44	159	73	90	126	114	51	1399
1995Q3	60	57	369	96	27	131	91	92	91	189	72	138	190	140	90	1833
1995Q4	79	80	397	148	15	148	101	108	70	197	86	134	191	199	78	2031
1996Q1	50	39	338	126	9	95	63	68	71	126	34	70	91	130	46	1356
1996Q2	56	68	387	126	39	94	75	80	94	204	73	101	135	131	65	1728
1996Q3	70	104	440	150	24	136	79	99	89	225	69	105	127	163	85	1965
1996Q4	45	90	486	194	16	166	76	137	106	233	92	81	143	197	92	2154
1997Q1	34	72	378	129	24	146	74	66	68	157	44	83	103	166	78	1622
1997Q2	52	85	419	194	19	156	64	101	86	245	81	84	132	145	70	1933
1997Q3	74	100	486	207	70	250	119	142	100	263	98	147	155	168	118	2497
1997Q4	73	117	561	219	53	215	101	111	87	224	77	169	178	222	92	2499
1998Q1	55	62	350	172	25	154	61	82	55	127	56	112	104	150	76	1641
1998Q2	53	92	421	169	28	176	67	101	88	223	63	132	131	181	52	1977
1998Q3	79	74	533	178	30	213	118	136	128	249	96	162	185	206	96	2483
1998Q4	62	89	446	162	26	161	136	135	112	194	90	99	105	169	71	2057
1999Q1	47	71	319	112	20	142	91	116	60	156	64	112	96	139	76	1621
1999Q2	41	73	445	132	29	141	89	124	87	208	67	150	145	131	83	1945
1999Q3	89	110	521	188	22	210	126	162	105	263	103	145	157	170	125	2496
1999Q4	76	103	611	195	39	241	111	136	128	235	109	153	199	181	126	2643
2000Q1	47	62	420	128	25	186	77	135	94	162	64	87	110	123	45	1765
2000Q2	66	83	444	168	17	212	93	113	98	210	95	102	141	162	73	2077
2000Q3	77	106	494	189	46	217	109	197	146	236	101	144	139	165	94	2460
2000Q4	79	119	565	169	48	214	188	160	131	288	152	170	134	222	138	2777
2001Q1	41	72	381	115	23	154	91	85	109	162	79	137	131	121	71	1772
2001Q2	44	113	485	173	26	158	147	93	120	252	86	149	147	143	85	2221
2001Q3	91	106	625	220	42	256	136	170	94	338	118	180	176	205	132	2889
2001Q4	85	152	515	237	41	178	142	140	149	296	110	137	182	185	110	2659
2002Q1	57	102	469	198	25	192	75	120	120	203	78	94	149	141	70	2093
2002Q2	78	114	606	163	34	202	136	132	126	307	108	181	156	197	111	2651
2002Q3	81	97	537	237	43	213	136	147	173	279	84	125	160	180	111	2603
2002Q4	95	109	617	203	44	232	171	181	126	291	112	135	196	194	161	2867
2003Q1	49	59	414	177	16	154	84	117	151	345	74	130	116	146	56	2088
2003Q2	57	144	529	152	30	285	112	181	150	342	100	97	133	175	90	2577
2003Q3	67	147	560	211	38	319	128	160	129	343	110	135	155	185	121	2808
2003Q4	108	147	782	254	55	457	149	202	142	374	151	174	212	230	128	3565
2004Q1	67	81	619	235	33	269	101	129	146	298	88	111	158	170	74	2579
2004Q2	75	98	635	203	42	348	120	141	198	352	93	133	175	193	122	2928
2004Q3	78	103	641	226	32	333	167	164	193	358	110	137	185	179	84	2990
2004Q4	86	86	687	253	42	282	156	197	217	331	94	122	186	238	99	3076
2005Q1	88	143	516	153	34	253	116	124	146	249	71	97	129	145	62	2326
2005Q2	84	110	584	231	38	313	143	156	178	318	98	103	171	180	94	2801
2005Q3	98	119	714	317	47	344	155	160	243	357	117	149	189	213	136	3358
2005Q4	68	173	777	245	31	441	162	173	262	340	107	142	270	309	132	3632
2006Q1	56	99	563	182	28	369	144	134	188	288	85	88	145	157	101	2627
2006Q2	90	117	696	299	56	421	118	147	182	405	93	113	345	177	98	3357
2006Q3	68	128	670	283	34	442	238	149	240	463	141	114	230	223	108	3531
2006Q4	63	141	755	293	39	361	349	178	222	396	95	120	171	248	118	3549
2007Q1	77	156	549	184	24	232	186	126	152	343	92	103	138	136	78	2576
2007Q2	76	120	553	350	36	318	212	119	216	327	107	119	214	156	108	3031
2007Q3	97	131	634	321	45	371	241	159	249	430	124	167	192	224	108	3493
2007Q4	74	145	727	338	34	366	201	172	216	405	98	132	198	249	95	3450
2008Q1	74	83	477	243	28	321	213	121	153	283	73	150	159	183	92	2653
2008Q2	70	95	539	247	31	393	189	163	275	409	102	103	275	239	76	3206
2008Q3	60	155	540	264	38	382	189	176	207	292	86	136	190	183	96	2994
2008Q4	47	97	461	232	24	301	157	125	121	296	64	83	175	128	58	2369
2009Q1	29	60	389	162	25	187	125	79	116	174	42	57	115	134	50	1744
2009Q2	40	86	479	235	25	328	185	153	277	257	50	86	166	172	81	2620
2009Q3	41	164	606	268	34	372	248	141	268	339	61	98	200	215	89	3144
2009Q4	65	119	630	236	31	374	203	165	262	313	67	92	185	222	91	3055
2010Q1	40	87	493	215	22	248	131	143	200	244	58	79	141	165	80	2346
2010Q2	64	100	563	216	25	378	143	135	257	282	68	82	163	150	75	2701
2010Q3	64	129	686	309	40	383	230	121	302	417	81	93	245	246	102	3448
2010Q4	71	113	714	244	27	440	174	147	253	336	97	90	230	238	92	3266
2011Q1	46	102	508	184	17	310	102	152	199	255	39	71	126	154	71	2336
2011Q2	79	110	615	250	27	324	149	162	216	332	58	104	165	176	101	2868
2011Q3	68	145	685	289	35	461	205	162	304	398	115	114	205	175	98	3459
2011Q4	54	123	740	295	47	413	236	192	212	329	78	113	234	229	79	3374
2012Q1	53	100	536	224	24	278	145	116	164	246	58	89	131	178	57	2399
2012Q2	63	90	533	296	42	330	174	144	215	251	60	111	173	211	82	2775
2012Q3	66	107	585	338	47	433	192	145	328	358	73	117	202	221	103	3315
2012Q4	71	126	569	264	25	427	182	166	232	286	78	73	156	209	95	2959
2013Q1	41	108	409	265	43	314	136	111	180	213	95	79	137	150	80	2361
2013Q2	50	76	465	193	39	324	130	127	341	246	55	97	174	182	65	2564
2013Q3	64	102	567	268	23	338	196	147	361	287	68	138	196	210	137	3102
2013Q4	42	109	473	253	33	401	172	156	214	272	65	71	143	148	90	2642
2014Q1	40	62	372	179	19	250	94	108	134	163	45	85	106	134	56	1847
2014Q2	54	82	478	215	20	367	137	151	250	222	62	142	157	145	75	2557
2014Q3	58	93	565	268	30	393	141	128	272	297	69	96	180	190	94	2874
2014Q4	66	96	509	259	30	372	123	156	250	292	46	86	176	193	73	2727
2015Q1	41	71	435	162	20	280	77	121	167	202	49	81	134	137	62	2039
2015Q2	56	73	519	240	18	314	119	106	237	245	49	77	150	177	75	2455
2015Q3	59	81	555	252	26	319	156	120	302	277	74	94	135	206	67	2723
2015Q4	43	77	484	254	21	302	130	150	212	228	66	90	142	186	74	2459
2016Q1	40	58	333	137	16	204	103	88	127	152	39	72	115	121	41	1646
2016Q2	56	85	403	185	18	286	109	94	166	205	51	75	125	126	70	2054
2016Q3	65	119	540	258	42	408	169	156	330	301	84	70	181	219	92	3034
2016Q4	64	119	644	226	42	380	216	195	240	272	108	89	213	259	80	3147
2017Q1	78	121	572	266	49	395	186	205	266	289	98	65	234	251	120	3195
2017Q2	72	146	522	286	39	427	204	215	308	282	78	80	200	286	139	3284
2017Q3	76	115	436	242	21	339	185	170	264	237	78	73	174	181	88	2679
2017Q4	53	114	404	242	35	319	150	156	232	248	73	52	151	178	92	2499
2018Q1	19	43	133	90	13	116	54	52	76	74	28	31	54	51	42	876

A.6 Implementation Overview

Here we provide a brief overview of the implementation of the AVM, with an emphasis on the choices related to the programming details, external dependencies and hardware.

Our stacked AVM is implemented in full in **Python**. The implementation is carried out with the aim of producing many estimates in parallel, to be able to quickly create value estimates for all dwellings in a given month, and therefore not optimised to create single estimates quickly. We make use of *mpi4py*[37], a standardised API for parallel computing, to divide the estimates into a given number of parallel cores, all running one instance of the AVM. To fully exploit the capabilities this provides, we run the implementation on a HP bl685c G7 server computer, employing four 2.2GHz AMD Opteron 6274 CPUs, each with 16 logical cores.

We rely on two different libraries for the individual methods; *SKlearn* and *XGBoost*. SKlearn is a large library, consisting of packages for many commonplace statistical methods. Amongst these we use:

i) **BaggingRegressor** - Contains all required methods for the bagging predictor algorithm

ii) **RandomForestRegressor** - Contains all required methods for the random forest algorithm

iii) **ExtraTreeRegressor** - Contains all required methods for the extra trees algorithm

iv) **GridSearchCV** - Contains the cross validation algorithm to search for hyperparameters

The XGBoost library is provided by an open source community, and it contains all the necessary methods to both tune and run the XGBoost-algorithm.

Training Time for the Individual Algorithms

The combined training time for one value estimate is, as mentioned in Chapter 5.5, about 400 seconds, given one core and the hardware-specifications above. Table 15 illustrate the training times for the individual algorithms.

Table 15: The average training times for the individual algorithms, given 10 000 comparable transactions and no parallelising of the submodels.

	Ensemble Learning			
	BP	RF	ET	XGB
Average Training Time	177.1 s	42.4 s	31.5 s	131.3 s

We note that this can be significantly optimised by parallelising the individual methods, if the goal is to create *one* quick value estimate.

[37]See http://mpi4py.readthedocs.io/en/stable/ for an overview of this package.

A.7 Stacked Generalisation - Python Code for Procedure 2

Here we present the Python code used to implement steps 3-5 in Procedure 2[38].

```python
class Ensemble(object):
   def __init__(self, n_splits, stacker, base_models):
      self.n_splits = n_splits
      self.stacker = stacker
      self.base_models = base_models

   def fit_predict(self, X, y, T, comp_transer, test_unit):
      X = np.array(X)
      y = np.array(y)
      T = np.array(T)

      folds = list(KFold(n_splits=self.n_splits,
          shuffle=True).split(X, y))
      S_train = np.zeros((X.shape[0], len(self.base_models)))
      S_test = np.zeros((T.shape[0], len(self.base_models)))

      for i, clf in enumerate(self.base_models):
         S_test_i = np.zeros((T.shape[0], self.n_splits))

         for j, (train_idx, test_idx) in enumerate(folds):
            X_train = X[train_idx]
            y_train = y[train_idx]
            X_holdout = X[test_idx]
            y_holdout = y[test_idx]

            clf.fit(X_train, y_train)
            y_pred = clf.predict(X_holdout)[:]

            S_train[test_idx, i] = y_pred

            predictions = clf.predict(T)[:]
            S_test_i[:, j] = predictions
         S_test[:, i] = S_test_i.mean(axis=1)

      if run_RS:
      rs_train_pred = []
      for id, unit in comp_transer.iterrows():
         rs_train_pred.append(list(prepred_rs_dict[id].values()))

      rs_test_pred =
          [list(prepred_rs_dict[test_unit.name].values())]
      S_train = np.concatenate((S_train, rs_train_pred), axis=1)
      S_test = np.concatenate((S_test, rs_test_pred), axis=1)
      x_and_s_train = np.concatenate((S_train, X), axis=1)
      x_and_s_test = np.concatenate((S_test, T), axis=1)

      self.stacker.fit(x_and_s_train, y)
      stack_ppsm = self.stacker.predict(x_and_s_test)[:]

      return stack_ppsm, S_test
```

[38] This code is inspired by the following Kernel: https://www.kaggle.com/serigne/stacked-regressions-top-4-on-leaderboard/notebook. Retrieved February 27th 2018

REFERENCES

Abraham, J. M., & Schauman, W. S. (1991). New Evidence on Home Prices from Freddie Mac Repeat Sales. *Real Estate Economics*, *3*, 333–352. Retrieved from http://dx.doi.org/10.1111/1540-6229.00556

Adam-Bourdarios, C., Cowan, G., Germain, C., & Guyon, I. (2015). The Higgs boson machine learning challenge. *NIPS Workshop on High-energy Physics and Machine Learning*, 19–55. Retrieved from http://www.jmlr.org/proceedings/papers/v42/cowa14.pdf doi: 10.1088/1742-6596/664/7/072015

Adams, Z., & Füss, R. (2010). Macroeconomic determinants of international housing markets. *Journal of Housing Economics*, *19*(1), 38–50. Retrieved from http://dx.doi.org/10.1016/j.jhe.2009.10.005 doi: 10.1016/j.jhe.2009.10.005

Allison, P. D. (2001). *Missing Data*. SAGE Publications. Retrieved from https://books.google.no/books?id=LJB2AwAAQBAJ

Alves, A. (2017). Stacking machine learning classifiers to identify Higgs bosons at the LHC. *Journal of Instrumentation*, *12*(05), T05005.

Bailey, M. J., Muth, R. F., & Nourse, H. O. (1963). A Regression Method for Real Estate Price Index Construction. , *58*(304), 933–942.

Balk, B., De Haan, J., & Diewert, E. (2011). *Handbook on Residential Property Prices Indices (RPPIs)* (No. November 2009). Retrieved from http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Handbook+on+Residential+Property+Prices+Indices+(+RPPIs+){#}1{%}5Cnhttp://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Handbook+on+Residential+Property+Prices+(RPPIs){%}230 doi: 10.2785/34007

Bengio, Y. (2012). Practical Recommendations for Gradient-Based Training of Deep Architectures. In G. Montavon, G. B. Orr, & K.-R. Müller (Eds.), *Neural networks: Tricks of the trade: Second edition* (pp. 437–478). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-35289-8{_}26 doi: 10.1007/978-3-642-35289-8_26

Bharti, S. K., & Babu, K. S. (2017). Automatic Keyword Extraction for Text Summarization: A Survey. *arXiv preprint arXiv:1704.03242*(April). doi: 10.1145/2980258.2980442

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Vol. 4) (No. 356). Secaucus, NJ, USA: Springer-Verlag New York, Inc. Retrieved from http://www.library.wisc.edu/selectedtocs/bg0137.pdf doi: 10.1641/B580519

Boligmani. (2015). *Selveier og andelsleilighet.* Retrieved 2018-04-01, from http://www.boligmani.no/aktuelt/boligmarkedet/selveier-og-andelsleilighet-fordelene-og-ulempene-du-bor-kjenne-til/6779

Bonnet, O. (2012). Hedonic prices in the Paris housing market and an analysis of the willingness to pay for a high-income neighborhood. *Institut detudes politiques de Paris*. Retrieved from http://econ.sciences-po.fr/sites/default/files/file/Odran{_}Bonnet{_}Master{_}dissertation.pdf

Breiman, L. (1996a). Bagging predictors. *Machine Learning*, *24*(2), 123–140. Retrieved from http://link.springer.com/10.1007/BF00058655 doi: 10.1007/BF00058655

Breiman, L. (1996b). Stacked regressions. *Machine Learning*, *24*(1), 49–64. Retrieved from http://link.springer.com/10.1007/BF00117832 doi: 10.1007/BF00117832

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. doi: 10.1023/A:

1010933404324
Calhoun, C. A. (1996). OFHEO house price indexes: HPI technical description. *Office of Federal Housing Enterprise Oversight*, *20552*(March), 1–15.
Campos, R., Canuto, S., Salles, T., de Sá, C. C., & Gonçalves, M. A. (2017). Stacking Bagged and Boosted Forests for Effective Automated Classification. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '17*, 105–114. Retrieved from http://dl.acm.org/citation.cfm?doid=3077136.3080815 doi: 10.1145/3077136.3080815
Case, K., & Shiller, R. (1987). Prices of Single Family Homes Since 1970: New Indexes for Four Cities. (September). Retrieved from http://www.nber.org/papers/w2393.pdf doi: 10.3386/w2393
Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Retrieved from http://arxiv.org/abs/1603.02754{%}0Ahttp://dx.doi.org/10.1145/2939672.2939785 doi: 10.1145/2939672.2939785
Chen, T., & Guestrin, C. (2018). *Understand your dataset with XGBoost — xgboost 0.71 documentation.* Retrieved 2018-05-22, from http://xgboost.readthedocs.io/en/latest/R-package/discoverYourData.html{#}creation-of-new-features-based-on-old-ones
Cochrane, J. H. (2009). *Asset Pricing:(Revised Edition).* Princeton university press.
Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, *2*(4), 303–314. Retrieved from https://doi.org/10.1007/BF02551274 doi: 10.1007/BF02551274
DMLC. (2016). *Notes on Parameter Tuning - xgboost 0.71 documentation.* Retrieved 2018-04-24, from http://xgboost.readthedocs.io/en/latest/how{_}to/param{_}tuning.html
Downie, M. L., & Robson, G. (2007). Automated Valuation Models : an international perspective. *RICS Automated Valuation Models Conference:AVMs Today and Tomorrow*(November), 1–78.
E24. (2014). *E24: Boligduellen.* Retrieved 2017-12-10, from https://e24.no/privat/eiendom/boligduellen-selveier-eller-andelsbolig/23281342
Eiendomsverdi. (n.d.). *Eiendomsverdi AS.* Retrieved 2018-01-30, from https://eiendomsverdi.no/
Eurostat. (2015). *Distribution of population by tenure status, type of household and income group.* Retrieved 2017-12-17, from https://tinyurl.com/y8yodafw
Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. , *139*, 23–37. Retrieved from http://link.springer.com/10.1007/3-540-59119-2{_}166 doi: 10.1007/3-540-59119-2_166
Freund, Y., & Schapire, R. E. (1996). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*. *journal of computer and system sciences*, *55*. Retrieved from https://ac.els-cdn.com/S002200009791504X/1-s2.0-S002200009791504X-main.pdf?{_}tid=a3d0cfd7-6278-40ef-98e2-9649c97e1c64{&}acdnat=1526896094{_}abe3abc903521693dbdcc430990cf3ca
Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer series in statistics New York.
Friedman, J. H. . (2001). Greedy Function Approximation : A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1189–1232. doi: 10.1214/009053606000001389
Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, *63*(1), 3–42. doi: 10.1007/s10994-006-6226-1
Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward

neural networks. *Pmlr*, *9*, 249–256. Retrieved from http://machinelearning.wustl.edu/mlpapers/paper{_}files/AISTATS2010{_}GlorotB10.pdf doi: 10.1.1.207.2059

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *AISTATS '11: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, *15*, 315–323. doi: 10.1.1.208.6449

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. *Nature*, *521*(7553), 800. Retrieved from http://goodfeli.github.io/dlbook/{%}0Ahttp://dx.doi.org/10.1038/nature14539 doi: 10.1038/nmeth.3707

Graczyk, M., Lasota, T., Trawiński, B., & Trawiński, K. (2010). Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. In N. T. Nguyen, M. T. Le, & J. Świ\katek (Eds.), *Intelligent information and database systems* (pp. 340–350). Berlin, Heidelberg: Springer Berlin Heidelberg.

Hariharakrishnan, J., Mohanavalli, S., Srividya, & Sundhara Kumar, K. B. (2017). Survey of pre-processing techniques for mining big data. *International Conference on Computer, Communication, and Signal Processing: Special Focus on IoT, ICCCSP 2017*. doi: 10.1109/ICCCSP.2017.7944072

Harris, J. C. (1989). The Effect of Real Rates of Interest on Housing Prices. *Journal of Real Estate Finance and Economics*, *2*, 47–60. Retrieved from https://link.springer.com/content/pdf/10.1007{%}2FBF00161716.pdf

Hauck, T. (2014). *scikit-learn cookbook* (2nd ed.). Packt Publ. Retrieved from http://cds.cern.ch/record/1975064

Ibbotson, R. G., & Siegel, L. B. (1984). Real estate returns: a comparison with other investments. *Real Estate Economics*, *12*(3), 219–242.

Inoue, A., & Kilian, L. (2008). How Useful Is Bagging in Forecasting Economic Time Series? A Case Study of U.S. Consumer Price Inflation. *Journal of the American Statistical Association*, *103*, 482–511. Retrieved from http://www.tandfonline.com/action/journalInformation?journalCode=uasa20 doi: 10.1198/016214507000000473doi.org/10.1198/016214507000000473

Jansen, S. J., De Vries, P., Coolen, H. C., Lamain, C. J., & Boelhouwer, P. J. (2008). Developing a house price index for the Netherlands: A practical application of weighted repeat sales. *Journal of Real Estate Finance and Economics*, *37*(2), 163–186. doi: 10.1007/s11146-007-9068-0

Kaggle. (2018). *Kaggle | Zillow Prize: Zillow's Home Value Prediction (Zestimate) - Kernels.* Retrieved 2018-05-01, from https://www.kaggle.com/c/zillow-prize-1/kernels

Kahneman, D., & Tversky, A. (1972). Subjective probability: A judgment of representativeness. *Cognitive Psychology*, *3*(3), 430–454. Retrieved from http://www.sciencedirect.com/science/article/pii/0010028572900163 doi: https://doi.org/10.1016/0010-0285(72)90016-3

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Retrieved from https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

Kettunen, P. (2012). *FI – Finland* (Tech. Rep.). Statistics Finland. Retrieved 2018-05-02, from https://www.dallasfed.org/{~}/media/documents/institute/houseprice/finland.pdf

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Levin, J. (2001). Information and the Market for Lemons. *The RAND Journal of Economics*, *32*(4), 657. Retrieved from http://doi.wiley.com/10.2307/2696386 doi: 10

.2307/2696386

Lior, R., & Others. (2014). *Data mining with decision trees: theory and applications* (Vol. 81). World scientific.

Ministry of Local Government and Modernisation. (2009). *Til deg som vurderer å kjøpe en andel i borettslag med høy fellesgjeld* (Tech. Rep.). Kommunaldepartementet. Retrieved 2018-05-25, from https://www.husbanken.no/-/media/Boligjuss/Kjoper{_}web.ashx?la=no

Moore, J. W., & Myers, J. (2010). Using geographic-attribute weighted regression for CAMA modelling. *Journal of Property Tax Assessment and . . .*, *7*(3), 5–28. Retrieved from http://library.iaao.org/fulltext/pa1003005.pdf

Mullainathan, S., & Spiess, J. (2017). Machine Learning: An Applied Econometric Approach. *Journal of Economic Perspectives*, *31*(2), 87–106. Retrieved from http://pubs.aeaweb.org/doi/10.1257/jep.31.2.87 doi: 10.1257/jep.31.2.87

Nielsen, D. (2016). Tree Boosting With XGBoost: Why Does XGBoost Win "Every" Machine Learning Competition? *NTNU Tech Report*(December), 2016. Retrieved 2018-05-27, from https://brage.bibsys.no/xmlui/bitstream/handle/11250/2433761/16128{_}FULLTEXT.pdf?sequence=1{&}isAllowed=y

Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press. Retrieved from http://neuralnetworksanddeeplearning.com/chap5.html

Northcraft, G. B., & Neale, M. A. (1987). Experts, amateurs, and real estate: An anchoring-and-adjustment perspective on property pricing decisions. *Organizational Behavior and Human Decision Processes*, *39*(1), 84–97. Retrieved from http://www.sciencedirect.com/science/article/pii/074959788790046X doi: https://doi.org/10.1016/0749-5978(87)90046-X

NS3457. (2013). *NS3457 Standard.* Retrieved 2018-05-28, from https://www.standard.no/nyheter/nyhetsarkiv/bygg-anlegg-og-eiendom/2014/ny-tabell-for-bygningstyper---ns-3457-3/

Opitz, D., & Maclin, R. (1999). Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligent Research*, *11*, 169–198. Retrieved from http://www.jair.org/papers/paper614.html doi: 10.1613/jair.614

Pattichis, C. S. (1999). Computer Assisted Valuation : Multiple Regression Analysis in Cyprus. (January 1999).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rattermann, M. (2007). *Valuation by Comparison: Residential Analysis & Logic*. Appraisal Institute. Retrieved from https://books.google.no/books?id=6n{_}pQN2noAQC

Rosen, S. (1974). Hedonic Prices and Implicit Markets: Product Differentiation in Pure Competition. *Journal of Political Economy*, *82*(1), 34–55. Retrieved from http://www.journals.uchicago.edu/doi/10.1086/260169 doi: 10.1086/260169

Schapire, R. E. (1989). The Strength of Weak Learnability (Extended Abstract). *Machine learning*, *227*(October), 28–33. Retrieved from http://link.springer.com/article/10.1007/BF00116037

Schapire, R. E. (2013). Explaining AdaBoost. Retrieved from http://rob.schapire.net/papers/explaining-adaboost.pdf

Shiller, R. (1991). Arithmetic Repeat Sales Price Estimators. *Journal of Housing Economics*, *1*(1), 110–126.

Sinnott, R. W. (1984). Virtues of the Haversine. *Sky Telesc.*, *68*, 159.

Sittler, P. (2017). Digitalization in Real Estate. In *24th annual european real estate society conference.* Delft, Netherlands. Retrieved from https://eres.architexturez

.net/doc/oai-eres-id-eres2017-128
Slovic, P., & Lichtenstein, S. (1971). Comparison of Bayesian and regression approaches to the study of information processing in judgment. *Organizational Behavior and Human Performance*, *6*(6), 649–744. Retrieved from http://www.sciencedirect.com/science/article/pii/003050737190033X doi: https://doi.org/10.1016/0030-5073(71)90033-X
Smith, P., & Tan, N. (2015). Bulletin December Quarter 2010. , 59–66.
S&P Dow Jones Indices. (n.d.). *S&P CORELOGIC CASE-SHILLER HOME PRICE INDICES.* Retrieved 2017-12-14, from http://us.spindices.com/index-family/real-estate/sp-corelogic-case-shiller
Statistics Norway. (2010). *Befolkningsvekst rundt Oslo.* Retrieved 2018-05-23, from https://www.ssb.no/befolkning/artikler-og-publikasjoner/befolkningsvekst-rundt-oslo
Statistics Norway. (2017). Price index for existing dwellings. , 7–11. Retrieved 2018-05-23, from https://www.ssb.no/en/bpi/
Statistics Norway. (2018a). *Population and population changes - quarterly.* Retrieved 2018-05-13, from https://www.ssb.no/en/befolkning/statistikker/folkemengde/kvartal
Statistics Norway. (2018b). *Prices per square metre of detached houses - SSB.* Retrieved 2018-05-23, from https://www.ssb.no/en/priser-og-prisindekser/statistikker/kvadenebol
Sufi, A., & Mian, A. (2014). *House of Debt*.
The Norwegian Mapping Authority. (2017). *Eiendomsinformasjon.* Retrieved 2017-12-17, from https://www.kartverket.no/eiendom/eiendomsinformasjon/
Tsatsaronis, K., & Zhu, H. (2004). What drives housing price dynamics : cross-country evidence. *BIS Quarterly Review*(March), 65–78.
Walpole, R. E., Myers, R. H., Myers, S. L., & Ye, K. E. (2016). *Probability and Statistics for Engineers and Scientists*. Pearson Education Limited. Retrieved from https://books.google.no/books?id=pP6SjwEACAAJ
Wheeler, D., Tiefelsdorf, M., Wheeler, D., & Tiefelsdorf, M. (2005). Multicollinearity and correlation among local regression coefficients in geographically weighted regression. *J Geograph Syst*, *7*, 161–187. Retrieved from https://link.springer.com/content/pdf/10.1007{%}2Fs10109-005-0155-6.pdf doi: 10.1007/s10109-005-0155-6
Wolpert, D., & Macready, W. (1996). Combining stacking with bagging to improve a learning algorithm. *Santa Fe Institute Technical Report*, 1–28. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.53.9933{&}rep=rep1{&}type=pdf
Wolpert, D. H. (1992). Stacked Generalization (Stacking). , *5*, 241–259.
Yandex. (2018). *CatBoost - Parameter tuning - Yandex Technologies.* Retrieved 2018-04-24, from https://tech.yandex.com/catboost/doc/dg/concepts/parameter-tuning-docpage/
Zillow. (2017). *How Accurate is Zillow's Zestimate?* Retrieved 2017-12-13, from https://www.zillow.com/zestimate