The Predictability of House Prices: "Man Against the Machine"*

Kristoffer B. Birkeland† Allan D. D'Silva§ Roland Füss‡ Are Oust¶

This version: November 10, 2019

* **Acknowledgement:** For helpful comments we are grateful to Jon Olaf Olaussen and Alois Weigand. We are also indebted to *Alva Technologies* for providing us the data.

† NTNU Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology (NTNU), Gløshaugen, NO-7491 Trondheim, Norway, Phone: +47 7359-3511, Email:kristoffer.birkeland@ntnu.no.

§ NTNU Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology (NTNU), Gløshaugen, NO-7491 Trondheim, Norway, Phone: +47 7359-3511, Email: allan.dsilva@ntnu.no.

‡ Swiss Institute of Banking and Finance (s/bf), University of St.Gallen, Unterer Graben 21, CH-9000 St.Gallen, Switzerland, Research Fellow at Center for Real Estate and Environmental Economics, NTNU Business School, Trondheim, Norway, and Research Associate at the Centre for European Economic Research (ZEW), Mannheim, Germany; Phone: +41 (0)71 224-7055, Fax: +41 (0)71 224-7088; Email: roland.fuess@unisg.ch.

¶ NTNU Business School, Norwegian University of Science and Technology (NTNU), Gløshaugen, NO-7491 Trondheim, Norway, Phone: +47 7355-9963, Email: are.oust@ntnu.no.

The Predictability of House Prices: "Man Against the Machine"*

Abstract

We develop an automated valuation model (AVM) for the residential real estate market by leveraging the concepts of stacked generalization and comparable market analysis. Specifically, we combine four novel ensemble learning methods with a repeat sales method and tailor the data selection for each value estimate. We calibrate and evaluate the model for the residential real estate market in Oslo by producing out-of-sample estimates for the value of 1,979 dwellings sold in the first quarter of 2018. Our novel approach of stacked generalization achieves a median absolute percentage error of 5.4%, and more than 96% of the dwellings are estimated within 20% of their actual sales price. When we compare the valuation accuracy of our AVM to that of the local estate agents in Oslo, it generally demonstrates its viability as a valuation tool. However, in stable market phases, the machine cannot beat the man.

Keywords: Automated valuation models; housing market; machine learning; repeat sales approach; XGBoost.

JEL Classification: *C*55, *R*31

1 Introduction

This paper develops an automated valuation model (AVM) that will leverage both historical transaction data and attribute-specific data for real estate property valuations. First, our novel automated valuation model (AVM) approach combines the well-established repeat sales method (RSM) of Case and Shiller (1987) with four machine learning methods based on the concept of stacked generalization (Wolpert, 1992). Second, it takes advantage of comparable market analysis (Rattermann, 2007), which seeks to value dwellings based on transactions with a close spatial and temporal proximity.

Hence, our empirical study may be placed in the intersection of research on *AVMs* in real estate valuation, ensemble learning from the field of econometrics, and index construction methodology in real estate markets. In the two latter research areas, we find extensive literature over the past five decades. Ensemble learning methods by Breiman (1996a) and Schapire (1989) have been successfully applied in a relatively few but growing number of econometric works (Graczyk et al. 2010; Inoue and Kilian, 2008).[1] The research into the construction of real estate indices has had few improvements since the work by Bailey, Muth, and Nourse (1963) and Case and Shiller (1987). Construction methodologies include median house price indices (e.g., Crone and Voith, 1992; Gatzlaff and Ling, 1994), hedonic pricing model (HPM) (e.g., Balk, de Haan, and Diewert, 2011; Geltner, 2015; Fisher, Geltner, and Webb, 1994) Value), repeat sales model (RSM) (e.g., Calhoun, 1996), and hybrid models (e.g., Quigley, 1995; Meese and Wallace, 1997).

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 that combines multiple models into one model to increase its out-of-sample predictive power (Opitz and Maclin, 1999). The four sub-models are from two classes of

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

ensemble learning techniques: bagging (Breiman, 1996a) and boosting (Schapire, 2013; Freund and Shapire, 1997). The use of these ensemble learning methods allows our model to learn patterns in the underlying data without making any assumptions about the data generating process. We hypothesize that, due to the amount of available data and the model complexity, these ensemble learning methods are suitable for use in the residential real estate market.

Over the last two decades, machine learning has found its way into the real estate sector. It was proved as a tool for mass appraisal and has been tested for different residential and commercial real estate markets. For instance, Antipov and Pokryshevskaya (2002) show that for price predictions in the residential apartment market of Saint-Petersburg, random forest is superior to techniques such as Chi-square Automatic Interaction Detectors (CHAID), Classification and Regression Trees (CART), multiple regression analysis, artificial neural networks, and Boosted Trees. Peterson and Flanagan (2009) and Ma, Chen, and Zhang (2015) demonstrate that artificial neural networks outperform linear hedonic regression in terms of pricing errors, out-of-sample pricing precision, and extrapolation of volatile pricing environments. Chiarazzo et al. (2014) use artificial neural networks to illustrate how accessibility, land-use, and environmental quality variables improve the appraisal of home sales prices in the city Taranto in Italy. Barr et al. (2017) construct home price indices based on a gradient boosted model. The authors show that their approach is superior to traditional median sale or repeat sales indices. Manganelli, De Mare, and Nesticò (2015) and Park and Bae (2015) demonstrate the effectiveness in modeling the relationship between the price and location of homes as well as between closing and listing price based on genetic and classification algorithms, respectively. Furthermore, Kok, Koponen, and Martinez-Barbosa (2017) support the superiority of AVM over traditional appraisal approaches for the

commercial real estate sector. Thereby, the authors emphasize the importance of location information.

We contribute to the recent literature on automated valuation models by mimicking the behavior of estate agents who often use recent sales of comparable dwellings as well as earlier transaction prices of the same property as a starting point for the valuation. More precisely, we propose a novel approach by combining ensemble learning methods with the RSM in an AVM. In addition to the RSM's prediction, the number of previous dwelling sales is included in the training data for the model-stacker. As frequently sold dwellings might have distinctive characteristics, this attribute is also likely to yield significant explanatory power. The RSM is trained on a separate dataset and aims to capture different market movements than the ensemble learning methods.

The four ensemble learning methods—bagging predictor (BP), random forest (RF), extra trees (ET), and XGBoost (XGB)—are used to generate four independent value estimates. In addition, we leverage XGBoost as a stacking method. We utilize the model-stacker XGBoost as a meta-estimator with both exogenous indicators and value estimates from the individual models as input variables. As a powerful stacking method, XBoost can detect each base model's performance and combine the underlying predictions accordingly. In our case, when both an array of ensemble learning methods and a repeat sales method is selected, i.e., the underlying models are diverse, stacking is highly effective. We employ a Norwegian dataset with 18,401 transactions from Oslo between August 2016 and April 2018. In Norwegian real estate markets, existing residential dwellings are sold by English auctions. This market setting provides an exclusive opportunity to test AVM models and compare the price predictions with those of real estate agents. Norwegian real estate agents are not allowed to advertise dwellings with a teaser price, i.e., an asking price lower than the seller's reservation price. Hence, real estate agents are incentivized to set their asking price close to the expected

market price. Our data also reflects this length arm's valuation behavior and thus, challenges our machine learning approach more than in other markets such as OTC markets or organized exchange markets.

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 the first quarter of 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 sales price. We compare the model to traditional AVMs, estate agents, and the U.S. industry leader Zillow for commercial AVMs. The performance of our AVM is comparable to the accuracy of Norwegian estate agents, which, in Oslo, has been at a MdAPE of 5.3% over the past two years. Our AVM is also superior to the precision of Zillow for a selection of cities, for which official performance statistics are available. Specifically, when we compare the valuation accuracy of our AVM to local estate agents, the out-of-sample results are of the same accuracy as the estate agents demonstrate in the training period of our model. This period includes a very dynamic development of prices in the Oslo housing market. However, when we directly compare the results for the first quarter of 2018, which consist of mostly stable house prices, the machine cannot beat the man.

The remainder of the paper is structured as follows: Section 2 describes our data from the residential real estate market in Oslo and introduces the data pre-processing steps used by our AVM. In Section 3, we develop the technical framework of our AVM. Section 4 presents the results of our model and evaluates its performance. Section 5 summarizes the performance of our AVM and concludes.

2 Data

Real estate valuation models are known to be highly dependent on local conditions (Tsatsaronis and Zhou, 2004). Therefore, we seek to identify the factors affecting sales prices in the residential real estate market of Oslo. The identification of these factors allows us to adapt our AVM to the city's market conditions.

2.1 The Residential Market of Oslo

The residential real estate market in Norway is characterized by a strong tradition of home-ownership, with 84% of Norwegians living in a self-owned home (Eurostat, 2015). By January 2018, the population of Oslo amounts to 673,468 (Statistics Norway, 2018). The city has experienced significant 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 characterized by four- and five-story brick apartment buildings. Historically, western parts of Oslo have generally had larger, more expensive houses, while eastern parts have had smaller, less expensive apartments. As illustrated in Figure 1, Oslo is divided into 15 districts and the city center.

[INSERT Figure 1 here]

2.2 Merged Datasets

Our primary datasets are consolidated from two official registers, *Grunnboken* and *Matrikkelen*, and extended by proprietory data.[2] *Alva Technologies*, a Norwegian real estate IT and services company, has provided us with three datasets: (i) The *address dataset* consists of all dwellings in Norway, where 276,780 of them are located in Oslo. The data contains the variables displayed in Table 1; (ii) The *enhanced transaction dataset* covers 18,401

[2] *Matrikkelen* and *Grunnboken* define real estate property and ownership relations for the Norwegian real estate market (The Norwegian Mapping Authority, 2017). For every dwelling in *Matrikkelen* information is provided on its location and boundaries, size, property type and, in the case of apartments, the building in which it is located. *Grunnboken* describes ownership relationships for both private properties and cooperatives.

transactions from Oslo between August 2016 and April 2018. This proprietary data takes into account improved quality and additional variables collected from the dwellings sale advertisements; and (iii) The *historical transaction dataset* includes all registered residential real estate transactions in Oslo between January 1993 and May 2018 as illustrated in Figure 2. In total, 220,898 separate transactions are mapped to Oslo addresses. The data includes a unique address identifier, the sales price, and sale date, but misses data on common debt and usable square meters (USM).

[INSERT Figure 2 here]

Table 1 lists the attributes that are gathered from these datasets and used by the ensemble learning methods in Sub-section 3.2. Both geographical coordinates and districts describe the dwellings' locations. The size of an individual dwelling is defined by its usable square meters (USM) and the number of rooms. Dwellings in Norway are typically divided into four main unit types: *apartments*, *rowhouses*, *semi-detached houses*, and detached *houses*. While apartments are part of a larger building complex, the three other unit types denote variations of dwellings that are single-family homes. In the ensemble learning methods, we include unit type as a categorical variable by differentiating between apartments and non-apartments (i.e., rowhouse, semi-detached house or detached house) due to their specific characteristics discussed in Sub-section 2.4.[3]

[INSERT Table 1 here]

In addition to the datasets provided by *Alva Technologies*, we use a separate dataset to analyze estate agents' aggregate valuation precision and utilize it as a benchmark for our model. This dataset is gathered from *Finn.no* and contains 15,786 transactions in Oslo from

[3] The correlation structure among the different housing attributes ranges between -0.11 and +0.15, and thus, shows no clear linear relationships between the attributes in our dataset. The only exception is the positive correlation of 0.51 between *USM* and number of rooms, which is significant at a 1% level.

2016 and 2017 and 3,009 transactions in Oslo from the first quarter of 2018. The data includes both *asking prices* provided by estate agents and final *sales prices*.

2.3 Data Pre-Processing

Our sample shows missing data for specific characteristics. For instance, values are missing in roughly 30% of its records, such as the *number of rooms*. The explanatory variables *story* and *build year* have 3,191 and 755 missing values, respectively, out of a total number of 18,073 observations.[4] In addition, the *historical transaction dataset* is prone to contain erroneous data due to the occurrence of sales under particular circumstances, inadequate data management, and imprecise matching of data from multiple sources.

To handle missing data, we remove all dwellings for which we do not have data regarding the *district*. By doing so, we simultaneously remove all data points with missing values for the *elevator*, *unit type*, *build type*, and *coordinates*. In total, this affects less than 1% of the dwellings. Next, we impute values for missing data points with the mean value of all remaining dwellings for the variables *story*, *build year*, and the *number of rooms*. We further clean the *historical transaction dataset*. We first remove transactions with a sales price below 100,000 NOK or above 70 million NOK, which accounts for less than 1% of the transactions. We then eliminate transactions where the same property is sold twice within three months. These are most likely distressed sales or speculative transactions, and therefore, do not reflect the real appreciation or depreciation in that given period (Jansen et al., 2008). We also remove transactions where the ratio of the sales prices between two transactions is larger than five. Finally, we exclude dwellings that have more than ten previous transactions within the recorded time period. Such a 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 taken from the enhanced

[4] For the following variables missing values (count) are: *District* (107), *Elevator* (78), *Unit Type* (77), *Build Type* (77), *Coordinates* (77), and *Common Debt* (3).

transaction dataset. Therefore, removing transactions from the historic transaction dataset will not bias the selection of our test set.

2.4 Exploratory Data Analysis

This subsection provides summary statistics on our primary variables *location*, *sales price*, *common debt*, and *usable square meters (USM)* for the different unit and ownership types. We provide a detailed report on the historical transactions as well as the spatial and temporal distribution of the most recent sales in Oslo.

Figure 3 shows prices per square meter for all transactions in the enhanced transaction dataset. Panel A of Figure 3 highlights the spatial variation in prices, which motivates the use of districts as an explanatory variable. There are also sizeable local price variations within districts. Panel B of Figure 3 shows a high variation in *price per square meter (PPSM)* for the district Nordstrand, which has proximity to the coastal line and areas with large high-rise buildings. Therefore, we include the dwelling's geographical coordinates as a finer-grained location measure.

[INSERT Figure 3 here]

Table 2 presents descriptive statistics for the ownership types introduced in Subsection 2.1 derived from the enhanced transaction dataset. The parameters indicate significant differences in the amount of common debt for the two ownership types. 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 without concerns about the lack of common debt data. Noticeable differences also exist in the *sales price*. These variations might be due to factors other than the ownership types (e.g., differences in location, build year, etc.).

[INSERT Table 2 here]

We have registered transactions on 185,961 of the dwellings in our dataset, of which 100,268 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. We note that we do not have data on the sales of *cooperatives* for the period 1993 to 2004. Also, 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, and at that point, an increasing number of transactions each year. During the last ten years, the number of transactions in Oslo doubled. Similarly, the large total number of repeat sales indicates that previous sales contain useful information for the valuation of dwellings.

The dwellings from the four-unit types (apartment, detached house, semi-detached house, and rowhouse) differ in both size and sales price. Table 3 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 the remaining dwelling types.

[INSERT Table 3 here]

Table 3 demonstrates that apartments are much smaller in size compared to the other three categories. For instance, the 75^{th} percentile usable square meters for apartments is smaller compared to the 25^{th} percentile for rowhouses. Semi-detached houses are more expensive than rowhouses, and detached houses are by far the largest and most expensive. The median-sized house is more than three times the size of the median-sized apartment. We observe smaller differences between the categories in *sales price* than in *USM*, indicating that there might be a decreasing marginal sales price for increasing dwelling size.

The dynamics of the housing market often cause large short-term fluctuations in real estate prices. Therefore, the precise date of a transaction is critical when modeling these prices. The *sold date* is the date when the buyer and seller agree upon a sales price, whereas

the *official date* is the date when the dwelling is handed over to the buyer. In our data, we find the average difference between sold and official date to be 46 days. We argue that the sold date is the most interesting when modeling house prices as it indicates the time at which the sales price was determined. All the transactions in the enhanced transaction dataset have a recorded sold date. For some historical transactions, only the official registration date is available. Therefore, for these observations, we use the official date as a proxy for the sold date.

3 Methodology

In this section, we develop our AVM. First, we introduce two key concepts of our AVM – *stacked generalization* and *comparable market analysis*. We then describe the attribute-based pricing methods as the most prominent value estimation technique for AVMs. The four recently developed and widely-used machine learning methods aim to circumvent some of the shortcomings of traditional parametric attribute-based pricing methods. Furthermore, we describe the RSM and its application to real estate valuation. Finally, we present our AVM as a combination of the concepts mentioned above.

3.1 Stacked Generalization and Comparable Market Analysis

When producing a value estimate for a dwelling, our AVM uses multiple underlying *sub-models* to create individual value estimates for every dwelling in the training data. Subsequently, a *model-stacker* as a separate model analyzes the individual value estimates *in-sample* to determine the *out-of-sample* prediction for the dwelling. This technique is referred to as stacked generalization, which was pioneered by Wolpert (1992) and refined by Breiman (1996b). The model-stacker uses the training data and individual predictions to identify and reduce the biases made by the underlying algorithms when predicting out-of-sample. The stacking procedure is outlined in Figure 4 and will be detailed in Subsection 3.4.

[INSERT Figure 4 here]

Our AVM tailors the *training data* for each value estimate based on the dwelling it aims to value and fits a unique model to each particular dwelling. The training data is selected to mimic the dwelling as closely as possible. This concept, known as comparative market analysis, is a prevalent valuation principle often applied to real estate valuation (Rattermann, 2007). Specifically, estate agents use nearby, recent sales as a starting point when valuing a dwelling. Automated valuation models can mimic this behavior by tailoring its source data to include transactions of dwellings in close geographical proximity to the dwelling in question. This fundamental concept of our model is described in Subsection 3.4.

3.2 Attribute-Based Pricing Methods

Traditionally, hedonic pricing models (HPMs) have been prevalent in academic literature for residential real estate valuations (Balk, De Haan, and Diewert, 2011). HPMs build on the assumption that goods are typically sold as a bundle of inherent attributes and their implicit prices can be estimated from observed prices of characteristics associated with them (Rosen, 1974). Using these implicit attribute prices, one can predict the sales price of a dwelling from the value of its underlying attributes. However, due to the potential existence of multicollinearity, non-linearity, and omitted variable bias, traditional HPMs may suffer from model misspecification (Wheeler and Tiefelsdorf, 2005; Balk, De Haan, and Diewert, 2011).

We propose an alternative approach, where four ensemble learning methods are used to generate four independent value estimates. Each method creates multiple sub-models, fitted to independently sampled input data. When predicting the value of an unseen dwelling, the methods combine each sub-model's prediction to determine the new sales price. We apply the four widely spread methods bagging predictor (BP), random forest (RF), extra trees (ET)[5],

[5] Extra Trees is actually an acronym derived from **Ext**remely **Ra**ndomized Trees.

and XGBoost (XGB). Each of the four ensemble methods builds multiple sub-models known as *decision trees*, which combine every tree's prediction to produce one value estimate. The number of trees in each ensemble model and the rules for building each tree are critical for the success of the ensemble method.

A *decision tree* is a simple but powerful tool for prediction modeling (Lior and Maimon, 2015). Informally, a decision tree is a tree-structure alike with a flowchart as illustrated in Figure 5, 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 based on a particular binary test related to a *cut-point* value of a given attribute. The attribute and its cut-point are chosen to minimize 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 dwellings' average PPSM in the training data that directs into that branch.[6]

Figure 5 exemplifies one binary decision tree. The XGBoost method produces several of such binary decision trees, which are grown sequentially to improve on the previous tree's residual as described in Subsection 3.2. Below, we present an example of the first tree for a randomly selected dwelling. The dwelling is a three-room, cooperative apartment in the district of Østensjø. 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: longitude ($f0$), latitude ($f1$), usable square meters ($f2$), number of rooms ($f4$), days since sale ($f6$), build year ($f7$), and common debt ($f8$).

[INSERT Figure 5 here]

The ensemble technique, *bagging*, trains the underlying decision trees in parallel and independently. We utilize three bagging methods, specifically bagging *predictor (BP)*,

[6] As mentioned in Subsection 3.2, we model the PPSM of the dwellings rather than the sales price.

random forest (RF), and *extra trees (ET)*. Random forest is an extension of the bagging predictor, while extra trees extend random forest further. The methods have subtle but distinct variations in the process of building decision trees. The most general bagging method is the bagging predictor of Breiman (1996a), which 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 sales price of an unseen dwelling, the bagging predictor averages the estimates from all the decision trees.

Random forest differs from the bagging predictor in terms of the method used for "growing" the underlying decision trees. Random forest builds the trees by sampling from only a randomly selected sub-sample of the attributes at each node split. This approach 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 among 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. In addition, feature bagging has the added benefit of being less computationally burdensome.

Instead of using feature bagging, as in the random forest method, *extra trees* of Geurts, Ernst, and Wehenkel (2006) randomizes the choice of cut-points for each attribute to learn about decorrelated trees. In so doing, it arbitrarily chooses a value (cut-point) for each attribute when splitting the trees in lieu of trying all possible cut-points. As a result, it increases the randomness by simultaneously reducing the computational burden of the algorithm.

The ensemble technique, *boosting*, introduced by Schapire (1989) trains the underlying decision trees *sequentially* with each tree being 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 boosting algorithm, *AdaBoost*, was first proposed by Freund and Schapire (1997) and then generalized into the *Gradient Boosting Machine* by Friedman (2001).[7] *XGBoost* is a recently developed boosting method that has proved successful in a variety of machine learning competitions.[8] The algorithm minimizes an objective function consisting of a loss function and a regularization term. At each iteration, the regularization term is added to constrain the model from overfitting. The objective function is defined 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 regularization term, f_k is the kth decision tree, and $\hat{y}_i$ and y_i are the predicted and actual sales prices of the ith dwelling, respectively.

The trees are built by splitting leaf nodes on the attribute value (cut point), which minimizes the pre-specified objective function. This is done recursively until the trees reach a pre-specified maximum depth. In our implementation, the loss term of the objective function is chosen to be the *mean absolute error*. Each leaf contains a weight, determined by the first- and second-order differential of the loss function and the regularization term.[9] Estimates continually improve by training each decision tree on the residuals from the previous iteration. When creating a value estimate for a dwelling x_i, XGBoost sums up the selected weight for each tree:

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

[7] See Schapire (2013) and Chen and Guestrin (2016) for a detailed overview on the development of boosting methods.

[8] Nielsen (2016) provides a comprehensive analysis of the methodology. For implementation details, we refer to Chen and Guestrin (2016).

[9] For the technical rationale see Equation (5) and the related descriptions in Chen and Guestrin (2016).

For the construction of decision trees in each ensemble method, several *hyperparameters* are available to determine the strength of the model. Two essential hyperparameters are the *number* of decision trees to build and the *depth* of each tree. Another, more subtle, hyperparameter is achieved by using bootstrapping. The bootstrapping methods pick random transactions from the training data with replacement. This sampling procedure produces separate datasets for each decision tree.

We note a trade-off between the explanatory power of the model and its computational complexity when determining hyperparameters. 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 (avoidance of *underfitting*) while simultaneously avoids identifying non-existing relationships between the attributes (avoidance of *overfitting*).

Both the bagging and boosting algorithms (XGBoost) require tuning of important hyperparameters to fit any particular dataset. This step is necessary due to the differences in the amount of available data, the number of attributes, and the structure of the data. Some hyperparameters are common for all our algorithms, such as the number of decision trees to create and the maximum depth of each tree, while others are specific for each model. Panels A and B of Table 4 describe the different hyperparameters and their tuned values.[10]

[INSERT Table 4 here]

We follow Hauck (2014) in optimizing the parameters for the bagging predictor, random forest, and extra trees methods. The tuning of the parameters for XGBoost is guided by

[10] We refer to the documentation from SKLearn and XGBoost for further descriptions of the parameters and their default values (Pedregosa, et al., 2011; Chen and Guestrin, 2018). We use the default values for the remaining hyperparameters in XGBoost.

DMLC (2016).[11] 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).[12] Hyperparameter tuning on the testing data would lead to overfitting, and thus, would overstate our performance estimates.

The ensemble methods as non-parametric models do not require any rich *a priori* knowledge regarding the underlying data generating process. This relaxation allows the method to adapt to the underlying data and the potential non-linear relationships among the variables. In contrast, traditional HPMs do not model non-linear relationships and make strict assumptions about the data such as linearity and homoscedasticity. Furthermore, ensemble learning methods are 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), which quickly may result in *overfitting* in the case of limited data availability. Similarly, simpler models, such as HPMs based on ordinary least square (OLS), may not be able to fully utilize the dataset, due to their constrained functional form. We discuss the use and demonstrate the performance of OLS and ANNs in Subsections 4.3 and the Appendix, respectively.

[11] 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. Subsequently, the prediction errors on the validation sets are averaged and the optimal combination of hyperparameters based on a scoring function is chosen.

[12] We refrain from using a validation set approach due to the limited time dimension in our sample. The division between training and validation is more common for techniques like ANN which are more likely to be affected by the problem of overfitting.

We also note that the selected ensemble learning methods require minimal feature engineering[13], 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. Thus, our model is simpler to implement and more robust to applications in dynamic real estate markets. On the other hand, the ensemble methods do not provide the same degree of transparency as the simpler OLS-based HPMs, which yield price estimates of individual attributes. In Subsection 4.3, we calculate *attribute importance* for the underlying characteristics to provide a sufficient degree of transparency for the application of our AVM.

As ensemble learning methods do not make any underlying assumptions of the distribution of the data, they may struggle to generalize beyond the observed training data. Therefore, ensemble learning methods are 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 (see Subsection 3.4). More precisely, we select training data, which is tailored to suit the dwelling in question; thus, making the model highly likely to generalize to the given dwelling.

3.3 Repeat Sales Method

We use the repeat sales method (RSM) introduced in the seminal paper by Bailey, Muth, and Nourse (1963) to create a separate index for each district in Oslo to capture the appreciation or depreciation in the market over time. The indices are used to produce price estimates for all previously sold dwellings in our dataset by adjusting each dwelling's previous sales price with the appropriate index. The RSM has the advantage of isolating actual increases in the price of a dwelling without requiring detailed information about the

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

characteristics of individual properties. As discussed in Section 2, there are several explanatory variables not included in our dataset.

The method's core idea is that the ratio of sales 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 dates. This idea is justified under a constant-quality assumption, i.e., the quality of a dwelling does not deteriorate or improve substantially over time. In theory, the RSM provides unbiased results of the price appreciation by controlling for all attributes that do not change over time, and in particular, for the micro-location. In our dataset, spanning more than a quarter-century, we are prone to encounter dwellings that have been upgraded or altered in some significant form. Hence, assuming constant quality will possibly bias our results. However, by using the RSM as proposed by Case and Shiller (1987), the repeat sales with extended longer periods between them are potentially attributed less weight in the estimation of the model. Although there are other implementations of the RSM, we note that the RSM by Case and Shiller (1987) is deemed the favorable implementation both in academia and for industrial applications (Balk, De Haan, Diewert (2011), S&P Dow Jones Indices, 2017).[14] The model is stated as a weighted least squares, with the logarithm of the ratios of the sales prices as the endogenous variables and the selling times as exogenous variables. We only consider condominiums for index construction, because the *historical transactions dataset* has missing data regarding common debt, which has a significant share in cooperatives as described in Subsections 2.2 and 2.4. Although this may lead to partially biased indices, we find that this alternative provides a better fit for our data. Figure 6 shows, in Panels A and B, the repeat sales indices for Oslo (from 1991Q1 to 2018Q1) and its districts (from 1993Q2 to 2018Q1), respectively.

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

[INSERT Figure 6 here]

3.4 Comparable Pricing and Stacking of Models

The ensemble learning sub-models 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[15] of dwellings for each of the unit types (apartments and non-apartments) separately. Further, we add a *ranking* variable to these transactions, i.e., $rank \in [1, 2, \ldots, n]$ based on the proximity, i.e., it increases with distance from the target dwelling.[16]

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

In addition to using XGBoost as an underlying method, we leverage it as a stacking method or *model-stacker*. As described, XGBoost is used as a meta estimator with both the exogenous variables and the value estimates from the individual models as an input. This stacking procedure can be described in detail as follows:

[15] With n = 10,000, if the dwelling is an apartment, and n = 2,000 if not. The distinction is made due to data availability of transactions for the two types of dwellings.

[16] The distance between two points in spherical geometry is given by the Haversine formula that adjusts for the earth's curvature, particularly becoming relevant for distances larger than 50km (Sinnott, 1984).

Procedure of Automated Valuation Model

Step 1: Denote the current sales price of the dwelling to be predicted as *u*.

Step 2: Select *n* comparable transactions, as described above, and denote the set of these transactions as *training data* or *X*.

Step 3: For each model $m \in [XBoost, Random\ Forest, Extra\ Trees, Bagging\ Predictor]$:

a) Divide the training data into $k = 5$ random subsets of equal size X_i with $i = 1 \ldots k$. For each of the subsets X_i:

i) Fit *m* observations to the training data *not* included in X_i to get a fitted model m_i.

ii) Use m_i to get estimates of the sales price, $\hat{P}_i$, for the data in X_i and extend the training data X_i with $\hat{P}_i$.

iii) Use m_i to get an estimate of the sales price of *u* denoted as $\hat{u}_i$.

b) Average the *k* predictions of the price of dwelling *u* to get and extend the dwelling data *u* with $\hat{u}$.

Step 4: For each data point in the training data *X* and 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, extend the data with a 0 for sale and resale.

Step 5: Fit another XGBoost model to the, now extended, training data *X*, and use the model to predict a sales price for *u*.

Using estimates from a diverse set of estimators enables our AVM to deliver robust results with high predictive power. We choose the powerful stacking method XBoost due to its ability to detect each base model's performance to combine the underlying predictions accordingly. Stacking is highly effective when the underlying models are diverse as in our case, i.e., when both an array of ensemble learning methods and a repeat sales method is selected. The drawback of combining stacked generalization with comparable market analysis is that the latter requires one instance of the model to be trained for each value estimate. By

stacking multiple individual models, the estimation becomes increasingly complex. Specifically, the model encompasses five ensemble learning methods, of which four are trained five times. The combined effect of these choices leads to increased model training time. We analyze and discuss the practical implications of our approach in Subsection 4.6.

3.5 Out-of-Sample Prediction

When evaluating our model, we divide our data into two disjoint sets, one which is used as a *training set* and the other being the *test set*[17]. By training our model on data from the training set and evaluating it on the test set, we simulate a real-life scenario. Thereby, the model is trained on data recorded up to a given day and produces value estimates on possible transactions for the next day. We perform this split monthly, while in practice, one would update the data every day. Hence, the results of our model can be interpreted as conservative in terms of both estimates and out-of-sample predictive power. When evaluating our model in Section 4, we make three such partitions using the following *splits*: January 1st, 2018, 00:00, February 1st, 2018, 00:00, and 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 Subsection 3.4. 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 *sales 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 sales price as though the sales date was the first day of the month.

[17] This gives an approximate 90-10 split between the training set and the test set.

4 Empirical Results

In this section, we evaluate the prediction accuracy of our AVM and its sub-models. Subsequently, we discuss essential model choices and potential challenges.

4.1 Performance Evaluation of the AVM

To analyze the performance of our AVM, we examine the distribution of its value predictions and compare it to the valuation provided by estate agents and industry leaders. Our AVM achieves an overall median absolute percentage error (MdAPE) of 5.4% in the first quarter of 2018 (see Panel A of Table 5). When comparing the performances of our AVM with the precision of the estate agents in Panels A and B of Table 5, we find that the performance of our AVM was very similar to the performances of the estate agents in 2016/17 (the training dataset period). 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. Looking at the more directly comparable first quarter of 2018, real estate agents perform better than the AVM along the forecast performance ratios. The main reason for the outperformance of the estate agents in the first quarter of 2018 compared to 2016/17 is probably due to the dynamics in the market during the later period. While the house price development in the first quarter of 2018 was quite stable, house prices in Oslo increased in 2016 by 23%, before they decreased by 6% in 2017.

[INSERT Table 5 here]

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 million dwellings in the U.S. and publishes their aggregated performances for a handful of selected cities. Panel C of Table 5 illustrates some of these performances. 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 sales price than Denver, whose value is 94.5%. Here, our model outperforms Zillow, with above 96% of the estimates being within 20% of the sales price. In addition, Zillow has data describing roughly 1-2 million dwellings in each of the presented cities, which is a considerably more significant amount of training data than we have had access to. While the comparative analysis illustrates the value-added of our AVM, we must point out that such a comparison of performances is limited due to the apparent differences between markets and data availability.

On a final note, we remark a behavioral finance aspect of the comparisons made in this section. The predictions of both the estate agents and Zillow are made (and published) prior to establishing the sales price; thus, they are likely to influence the buyer and seller. We believe this can have two effects: Estate agents might aim to price a dwelling lower than the expected sales 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 sales price in the 2016/2017 data, and roughly 43% lower in the Q1 2018 data. In addition, by the anchoring-and-adjustment heuristic, as presented in psychological literature[18], such reference points are prone to bias market participants' valuation. Thus, the final sales price is likely to be insufficiently adjusted away from the anchor.

4.2 Performance Evaluation of the Stacked Model

To justify the application of stacked generalization in our AVM, we perform a thorough evaluation of the effect of stacking. First, we compare the accuracy, measured by 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 analyze the performance against the

[18] Seminal works by Slovic and Lichtenstein (1971) and Kahnemann and Tversky (1972) introduce and discuss this heuristic, while Northcraft and Neale (1987) consider it empirically in the setting of the residential real estate market in Tuscon, Arizona. They find that the "*subject populations were significantly biased by listing prices."* (Northcraft and Neale (1987), p. 95)

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.

To justify the use of stacked generalization in our AVM, we compare the accuracy of the stacked model to that of the underlying models. The comparison is made out-of-sample for January, February, and March 2018, as well as in aggregate for the three months. The results are presented in Table 6. We observe that, on average, the AVM performs significantly better than the individual models. We also note that the RSM performs considerably weaker than all the other underlying methods.

[INSERT Table 6 here]

We further examine the correlations between the individual model residuals to compare their value estimates and discuss the potential gain of stacking. Figure 8 illustrates the correlations between out-of-sample residuals of the ensemble learning methods.[19] An apparent observation is the highly positive correlation between the methods. We choose to include all 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. Figure 8 shows that the model-stacker (XGB-S) can detect relationships in the data not captured by the individual methods; hence, yielding better out-of-sample results.

[INSERT Figure 7 here]

Figure 8 displays how training time and accuracy, represented by the MdAPE, increases with the number of comparable transactions for each valuation for the period of 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 can score an out-of-sample MdAPE of about 6%. However, the further gain towards 5% MdAPE is computationally burdensome. Because we do not observe further improvements after including 10,000 comparable

[19] We do not include the RSM here, because it only covers 77% of the transactions.

transactions and due to training time constraints, we choose this number of comparables in our final model.

[INSERT Figure 8 here]

4.3 Performance Evaluation of the Attribute-Based Pricing Methods

Table 6 illustrates the out-of-sample performance of the selected ensemble learning methods and our AVM compared to a regular OLS-based HPM. 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 and might obtain better results by including transformation and grouping attributes in a pre-processing step.

As introduced in Subsection 3.2, AVMs using ensemble learning are harder to interpret than OLS-based HPMs. However, this weakness can be addressed by analyzing the models' underlying decision trees, either by displaying the individual trees or by analyzing aggregated statistics from each model, such as *attribute importance*. Thereby a score is given to each attribute based on its importance in increasing the model's performance. To calculate this ratio, several methods exist based on the aggregate frequency of occurrence and the placement of attributes in the decision trees.[20] We give an example of analyzing 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 and Guestrin, 2018). These scores are denoted as $\alpha_i \in [0,1]$, where $\sum_i \alpha_i = 1$.

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

Since we build a separate model for each dwelling, we analyze the model built for one particular apartment sold in Oslo in March 2018. The apartment is a condominium situated in the district Gamle Oslo with 82 USM. The dwelling was constructed in 2013, has four rooms, and is located on the fifth floor. Figure 9 illustrates the importance of the most relevant 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 critical than the categorical attributes when building the decision trees. Specifically, the location, represented by longitude and latitude, the size of dwelling, represented by USM, and the date of the transaction, represented by the *days since sale*-attribute, are the most significant attributes. We observe some variables with a *zero* attribute importance score, such as districts far from the selected dwelling.

[INSERT Figure 9 here]

These observations are not only reasonable but are also confirmed by real estate theory. The numerical variables have a larger number of possible *cut-points* than the binary categorical variables; therefore, they can be used more often to adequately partitioning the dataset. The variables that have received the highest attribute scores are the ones most often associated with sales prices of dwellings. The fact that some attributes receive a score of *zero* for one particular model does not hinder the attribute from being necessaryin another model. This variation illustrates the model's ability to adapt to the underlying data.

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. We acknowledge that traditional hedonic methods, such as those based on ordinary least squares (OLS), might yield favorable results in some situations. Given a smaller dataset, less training time, or more detailed inference statistics, we believe that

traditional hedonic methods could be applied with decent precision (see Table 6 for comparable results).

Moreover, we consider the use of an artificial neural network (ANN) as a sub-model in our AVM in the Appendix. Due to a large number of hyperparameters and low interpretability, our findings demonstrate their high-level complexity for real-world applications. We provide a summary of our results when including ANN as a sub-model for our AVM in the Appendix. In short, we found the process of designing the network to be a highly specialized engineering process, with extremely many design choices and only a handful of established guidelines.

4.4 Performance Evaluation of the Repeat Sales Method

As illustrated in Table 6, the RSM shows a considerably weaker performance compared to other individual models and similar performance to the OLS-based HPM. Therefore, one might question the inclusion of the model in the AVM. However, as argued in Subsections 2.1 and 3.3, the RSM is trained on a separate dataset and aims to capture different market movements than the ensemble learning methods. Hence, we 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 the procedure of Subsection 3.4—without step 4—and compare the performance to that of the AVM. We present this comparison in Table 7 and note that the MdAPE increases by 8% overall when excluding the RSM from the model. This decline in predictability 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 frequently sold dwellings often might have distinctive characteristics.

[INSERT Table 7 here]

5 Conclusion

This study develops an automated valuation model (AVM) by stacking four different ensemble learning methods and the repeat sales method (RSM). We evaluate the predictive power of our model on transaction data of the residential real estate market in Oslo. The novel approach of combining ensemble learning and real estate indices in the form of stacked generalization leads to substantial improvements compared to individual methods. It reveals that the use of comparable market analysis is valuable information as the model benefits the most from nearby transactions. Thus, our findings support the understanding of how the quality of AVM can be improved to become a valuable instrument for commercial use.

Our AVM estimates the value of the 1,979 dwellings sold in Q1 of 2018 with a median absolute percentage error (MdAPE) of 5.4%. This performance is comparable to the accuracy of Norwegian estate agents and is superior to the precision of Zillow, the industry leader in the U.S., for a selection of cities, for which official performance statistics are available. In summary, we conclude that in very dynamic markets, the valuation accuracy of our AVM is similar to the one of estate agents' appraisals. However, in more stable market phases, the machine cannot beat the man.

A drawback of applying stacked generalization in a model is the training time it demands due to the required predictions made on the training data by individual folds. Another challenge for the non-parametric ensemble learning methods is that they require sufficiently diverse training data to achieve strong out-of-sample predictive power. Specifically, they do not generalize well outside the observed range of attribute values, as they do not make any prior assumptions about the underlying data. However, our model does not only predict with high accuracy but also with strong precision. In addition to a low MdAPE, it has a high share of predictions within 20% of the actual sales price, which indicates the model's ability to generalize well to the given dataset.

A significant transition from traditional HPMs to ensemble learning AVMs is the shift from a theory-driven to a data-driven approach. Hence, another well-known drawback is the model's lack of transparency and, to a certain extent, its lack of underlying model assumptions. We have addressed how available tools, such as the measure of attribute importance and the display of decision trees, have been designed to increase the degree of interpretability of the ensemble learning methods. 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.

Appendix: An Evaluation of Artificial Neural Networks as AVMs

Artificial Neural Networks (ANNs) are a class of machine learning techniques which model 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 organized 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[21], they can be hard to apply to many learning problems. This is due to their complex training procedure and non-parametric structure.

Implementation: We conceptually follow Bengio (2012) and Goodfellow, Bengio, and Courville (2016) to design our ANN and use the *MLPRegressor* package by *scikit-learn* for Python for the implementation. There are a few generally accepted practices for applying ANNs, such as normalization of input data and the use of mini-batches.[22] For the model's hyperparameters, we use a selection of default and recommended parameters[23], as well as experience combined with a grid-search and cross-validation. The most challenging task is the choice of the structure of neurons; i.e., the number of layers, the number of neurons per layer, and the connections between each layer. The choices are numerous, and the use of a grid-search alone to determine the appropriate choice is infeasible due to the exponential increase

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

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

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

in computational requirements. Our final model is the most stable with reasonably consistent results. We provide the selected hyperparameters in Table A.1.

Table A.1: ANN Hyperparameters

Variable	Description	Selected Value
Optimizer	The solver used for weight-optimization.	Adam (Kingma and Ba, 2014)
Activation function	The activation function used in the neurons.	ReLU (Glorot, Bordes, and Bengio, 2011)
# of hidden layers	The number of layers.	4
Hidden layers	The number of nodes per layer.	[64, 64, 32, 32]
Maximum number of iteration	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 regularization parameter to prevent an overfitting of the data	0.0001

Results and Discussion: The results of our work are presented in Table A.2, which also includes the stacked AVM for comparison. We see that our implementation of an ANN performs far weaker than the AVM. Although there might be superior implementations of ANNs for this problem, we cannot determine it by any structured approach. Instead, we rely primarily on experience and grid-searches. When considering the solutions used in several Kaggle-competitions (Kaggle, 2018), we find that ANNs are prevalent as submodels, but rarely used without a model-stacker. Furthermore, recent literature reveals that the inherent struggle of training ANNs is an established issue.[24]

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

Table A.2: Prediction Accuracy of Artificial Neural Network (ANN)

	Within 5%	**Within 10%**	**Within 20%**	**MdAPE**	**MAPE**
AVM	46.89%	76.36%	96.31%	5.35%	7.17%
ANN	24.20%	46.18%	77.30%	10.96%	13.99%

We conclude from this exploration that, although ANNs are universal approximators, they require a great deal of experience to apply with success and miss a structured engineering process. Therefore, we believe ensemble learning methods to be more consistent and more straightforward to apply in practice.

References

Antipov, E.A., and E.B. Pokryshevskaya (2002): Mass appraisal of residential apartments: An application of random forest for valuation and a cart-based approach for model diagnostics, *Expert Systems With Applications* 39, 1772-1778.

Bailey, M.J., R.F. Muth, and H.O. Nourse (1963): A Regression Method for Real Estate Price Index Construction, *Journal of the American Statistical Association* 58(304), 933-942.

Balk, B., J. De Haan, and E. Diewert (2011): Handbook on Residential Property Prices Indices (RPPIs), no. November 2009.

Barr, J., E. Ellis, A. Kassab, and C. Redfearn (2017): Home price index: a machine learning methodology, *International Journal of Semantic Computation* 11, 111-133.

Bengio, Y. (2012): Practical Recommendations for Gradient-Based Training of Deep Architectures, In Tricks of the Trade, volume 7700 of Theoretical Computer Science and General Issues, Springer-Verlag, pp. 437-478.

Bishop, C.M. (2006): Pattern Recognition and Machine Learning (Information Science and Statistics), vol. 4. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Breiman, L. (1996a): Bagging predictors, *Machine Learning* 24(2), 123-140.

Breiman, L. (1996b): Stacked regressions, *Machine Learning* 24(1), 49-64.

Breiman, L. (2001): Random forests, *Machine Learning* 45(1), 5-32.

Calhoun, C.A. (1996): OFHEO house price indexes: HPI technical description, Office of Federal Housing Enterprise Oversight, 20552(March), 1-15.

Case, K., and R. Shiller (1987): Prices of Single Family Homes Since 1970: New Indexes for Four Cities, *New England Economic Review*, (September/October), 45-56.

Chen, T., and C. Guestrin (2016): XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 785-794, San Francisco, CA.

Chen, T., and C. Guestrin (2018): Understand your dataset with XGBoost – xgboost 0.71 documentation, from: https://xgboost.readthedocs.io/en/latest/R-package/discoverYourData.html#creation-of-new-features-based-on-old-ones accessed: July 15. 2019.

Chiarazzo, V., L. Caggiani., M. Marinelli, and M. Ottomanelli (2014): A neural network based model for real estate price estimation considering environmental quality of property location, *Transportation Research Procedia* 3, 810-817.

Cybenko, G. (1989): Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems* 2(4), 303-314.

Crone, T. M., and R. Voith (1992): Estimating house price appreciation: A comparison of methods. *Journal of Housing Economics* 2(4), 324-338.

DMLC (2016): Notes on Parameter Tuning – xgboost 0.71 documentation, from: https://xgboost.readthedocs.io/en/latest/tutorials/param_tuning.html accessed: July 15. 2019.

Eurostat (2015): Distribution of population by tenure status, type of household and income group, EU – SILC survey.

Fisher, J. D., D. Geltner, and R.B. Webb (1994): Value Indices of Commercial Real Estate: A Comparison of Index Construction Methods. *Journal of Real Estate Finance and Economics* 9(2), 137-164.

Freund, Y., and R.E. Schapire (1997): A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *Journal of Computer and System Sciences* 55, 119-139.

Friedman, J., T. Hastie, and R. Tibshirani (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.

Gatzlaff, D.H., and D.C. Ling (1994): Measuring Changes in Local House Prices: An Empirical Investigation of Alternative Methodologies. *Journal of Urban Economics* 35(2), 221-224.

Geltner, D. (2015): Real Estate Price Indices and Price Dynamics: An Overview from an Investments Perspective. *Annual Review of Financial Economics* 7(1), 615-633.

Geurts, P., D. Ernst, and L. Wehenkel (2006): Extremely randomized trees, *Machine Learning* 63(1), 3-42.

Glorot, X., and Y. Bengio (2010): Understanding the difficulty of training deep feedforward neural networks, *Proceedings of Machine Learning Research (PMLR)* 9, 249-256.

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

Goodfellow, I., Y. Bengio, and A. Courville (2016): Deep Learning, *Nature* 521(7553), 800.

Graczyk, M., T. Lasota, B. Trawiński, and K. Trawiński (2010): Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal, In: N.T. Nguyen, M.T. Le, and J. Świątek, et al. (eds.), Intelligent Information and Database Systems, LNCS (LNAI), Vol. 5991, pp. 340-350, Springer Heidelberg.

Hauck, T. (2014): scikit-learn cookbook. Packt Publ., 2nd ed..

Inoue, A., and L. Kilian (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.

Jansen, S.J., P. De Vries, H.C. Coolen, C.J. Lamain, and P.J. Boelhouwer (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.

Kaggle (2018): Kaggle|Zillow Prize: Zillow's Home Value Prediction (Zestimate) - Kernels.

Kahneman, D., and A. Tversky (1972): Subjective probability: A judgment of representativeness, *Cognitive Psychology* 3(3), 430-454.

Kingma, D.P., and J. Ba (2014): Adam: A method for stochastic optimization, *ICLR conference paper*.

Kok, N., E.-L. Koponen, and C.A. Martinez-Barbosa (2017): Big data in real estate? From manual appraisal to automated valuation, *The Journal of Portfolio Management* 43(6), 202-211.

Lior, R., and O. Maimon (2015): Data mining with decision trees: theory and applications, vol. 81. World scientific.

Ma, H., M. Chen, and J. Zhang (2015): The Prediction of Real Estate Price Index Based on Improved Neural Network Algorithm, *Advanced Science and Technology Letters* 81, 10-15.

Manganelli, B., G. De Mare, and A. Nesticò (2015): Using Genetic Algorithms in the Housing Market Analysis, Computational Science and Its Applications – ICCSA 2015, 36-45.

Meese, R.A., and N.E. Wallace (1997): The Construction of Residential Housing Price Indices: A Comparison of Repeat-Sales, Hedonic-Regression, and Hybrid Approaches. *Journal of Property* 14(1/2), 51-73.

Mullainathan, S., and J. Spiess (2017): Machine Learning: An Applied Econometric Approach, *Journal of Economic Perspectives* 31(2), 87-106.

Nielsen, D. (2016): Tree Boosting With XGBoost: Why Does XGBoost Win "Every" Machine Learning Competition?, *NTNU Tech Report*, (December), 2016.

Nielsen, M.A. (2015): Neural Networks and Deep Learning. Determination Press.

Northcraft, G.B., and M. A. Neale (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.

NS3457 (2013): NS3457 Standard, https://www.standard.no/nettbutikk/produktkatalogen/produktpresentasjon/?ProductID=665100.

Opitz, D., and R. Maclin (1999): Popular Ensemble Methods: An Empirical Study, *Journal of Artificial Intelligent Research* 11, 169-198.

Oust, A., S.N. Hansen, and T.R. Pettrem (2019): Combining Property Price Predictions from Repeat Sales and Spatially Enhanced Hedonic Regressions. *Journal of Real Estate Finance and Economics*, forthcoming.

Park, B., and J.K. Bae (2015): Using Machine Learning Algorithms for Hosuing Price Prediction: The Case of Faifax County, Virginia Housing Data, *Expert Systems with Applications* 42(6), 2918-2934.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Van-derplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011): Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research* 12, 2825-2830.

Peterson, S., and A.B. Flanagan (2009): Neural network hedonic pricing models in mass real estate appraisals, *Journal of Real Estate Research* 31(2), 147-164.

Quigley, J.M. (1995): A Simple Hybrid Model for Estimating Real Estate Price Indexes. *Journal of Housing Economics* 4, 1-12.

Rattermann, M. (2007): Valuation by Comparison: Residential Analysis & Logic. Appraisal Institute.

Rosen, S. (1974): Hedonic Prices and Implicit Markets: Product Differentiation in Pure Competition, *Journal of Political Economy* 82(1), 34-55.

Schapire, R.E. (1989): The Strength of Weak Learnability (Extended Abstract), *Machine Learning* 227(October), 28-33.

Schapire,R.E.(2013):ExplainingAdaBoost, https://www.cs.princeton.edu/~schapire/papers/explaining-adaboost.pdf.

Slovic, P., and S. Lichtenstein (1971): Comparison of Bayesian and regression approaches to the study of information processing in judgment, *Organizational Behavior and Human Performance* 6(6), 649-744.

S&P Dow Jones Indices (2017): S&P Corelogic Case-Shiller Home Price Indices, https://us.spindices.com/index-family/real-estate/sp-corelogic-case-shiller.

Statistics Norway (2010): Befolkningsvekst rundt Oslo, https://www.ssb.no/befolkning/artikler-og-publikasjoner/befolkningsvekst-rundt-oslo.

Statistics Norway (2018): Population and population changes – quarterly, https://www.ssb.no/en/folkemengde/.

Tsatsaronis, K., and H. Zhu (2004): What drives housing price dynamics: cross-country evidence, *BIS Quarterly Review*, (March), 65-78.

Wheeler, D., and M. Tiefelsdorf (2005): Multicollinearity and correlation among local regression coefficients in geographically weighted regression, *Journal of Geographical System* 7(2), 161-187.

Wolpert, D. H. (1992): Stacked Generalization, *Neural Networks* 5(2), 241-259.

Figures:

Figure 1: Districts of Oslo

The figure shows the administrative districts of Oslo with the price/m^2 ranking for 2017 given by Oust, Hansen, and Pettrem (2019). Data for district CBD (denoted by grey color) were not available, while district Søndre Nordstrand is not represented in our dataset and is thus excluded from the map.

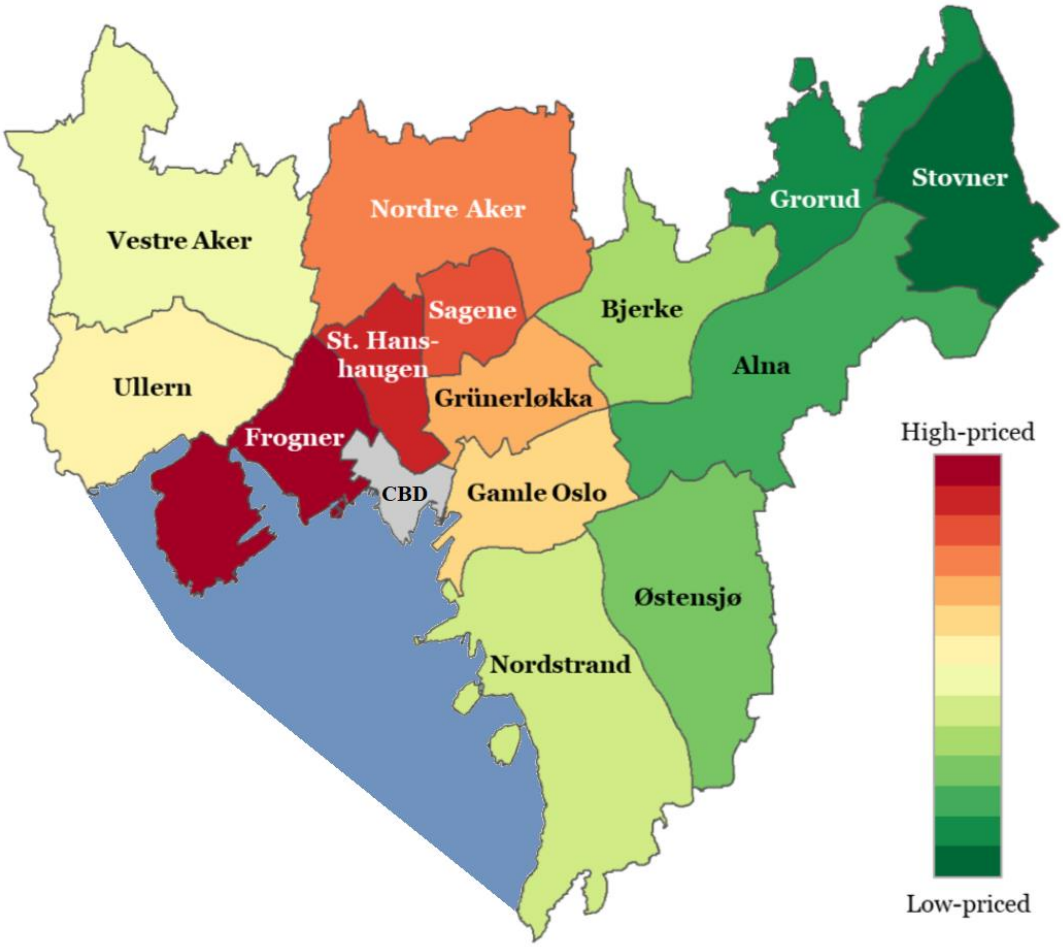


Figure 2: Number of Transactions

The figure shows all recorded transactions by year of transaction in the dataset for the period from 1993 to 2018. Sales from cooperatives are added around 2007, which causes a sharp increase in the number of transactions. Note that the bar for the year 2018 is small as the data covers only the period until May.

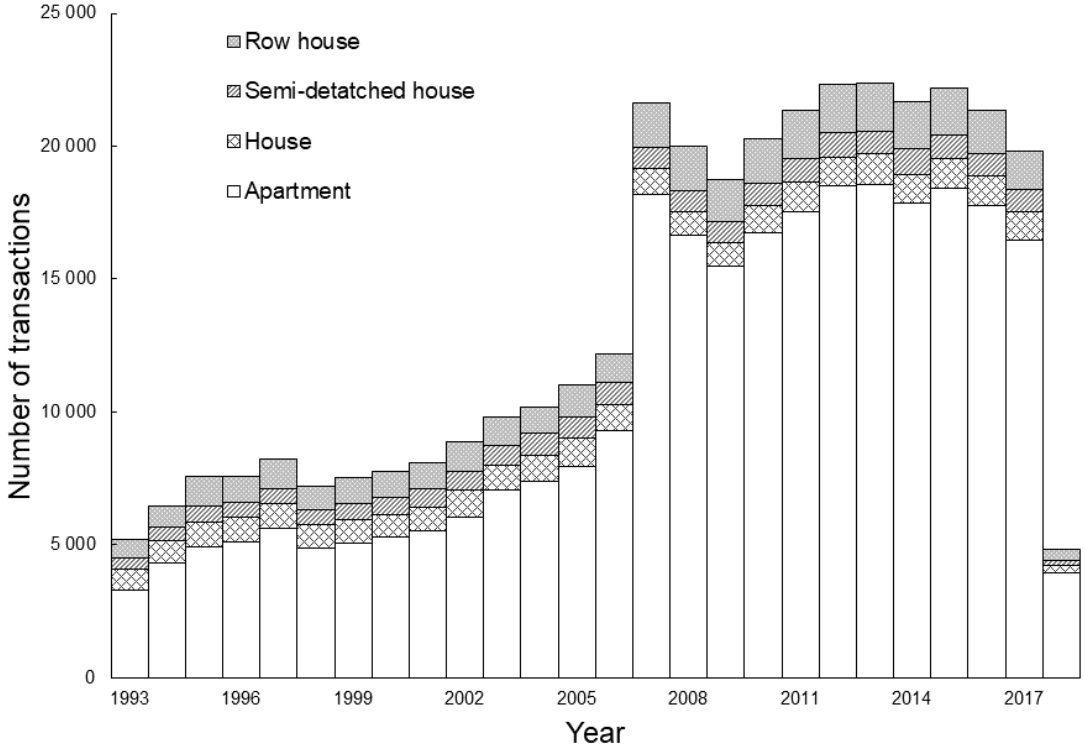
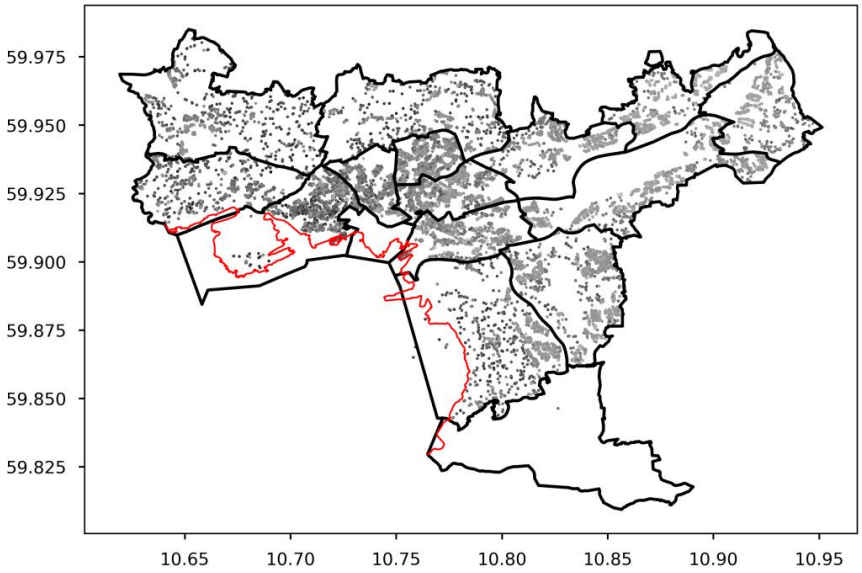


Figure 3: Spatial Distribution of Total Transactions

Panel A of the figure illustrates all recorded transactions in Oslo from 2016-2018, which covers 18,073 transactions in total. Panel B shows all recorded transactions in the district Nordstrand from 2016-2018, which results in a total of 1,167 transactions. A darker color represents a *higher price per square meter (PPSM)*. The bold black lines represent the district borders, while the red line is the coastline. Values for longitude and latitude are reported on the *x*- and *y*-axis.

Panel A: All recorded transactions in Oslo



Panel B: All recorded transactions in the district Nordstrand

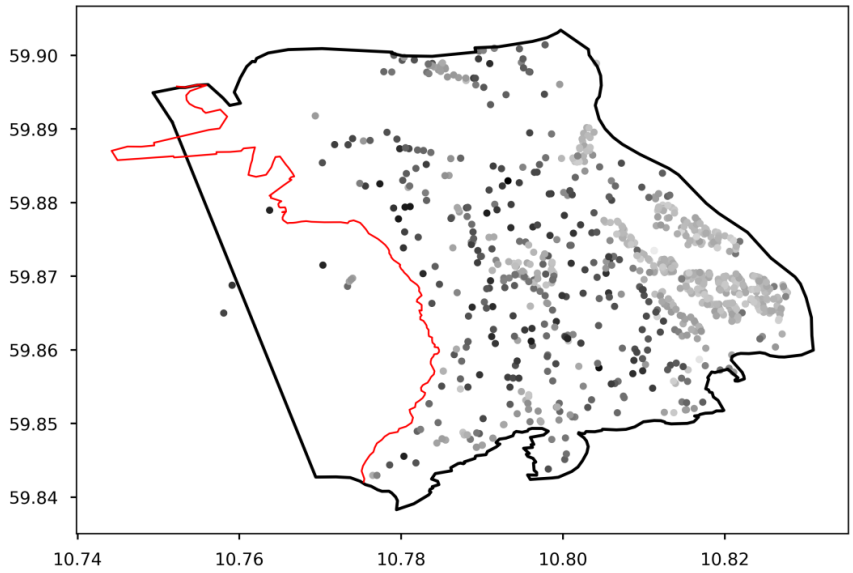


Figure 4: Automated Valuation Model

The figure describes the automated valuation model (AVM). The model is trained for each individual value estimate it produces. The model extracts the *n* comparable transactions, where *n* = 10,000 if the dwelling is an apartment, and *n* = 2,000 otherwise. 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.

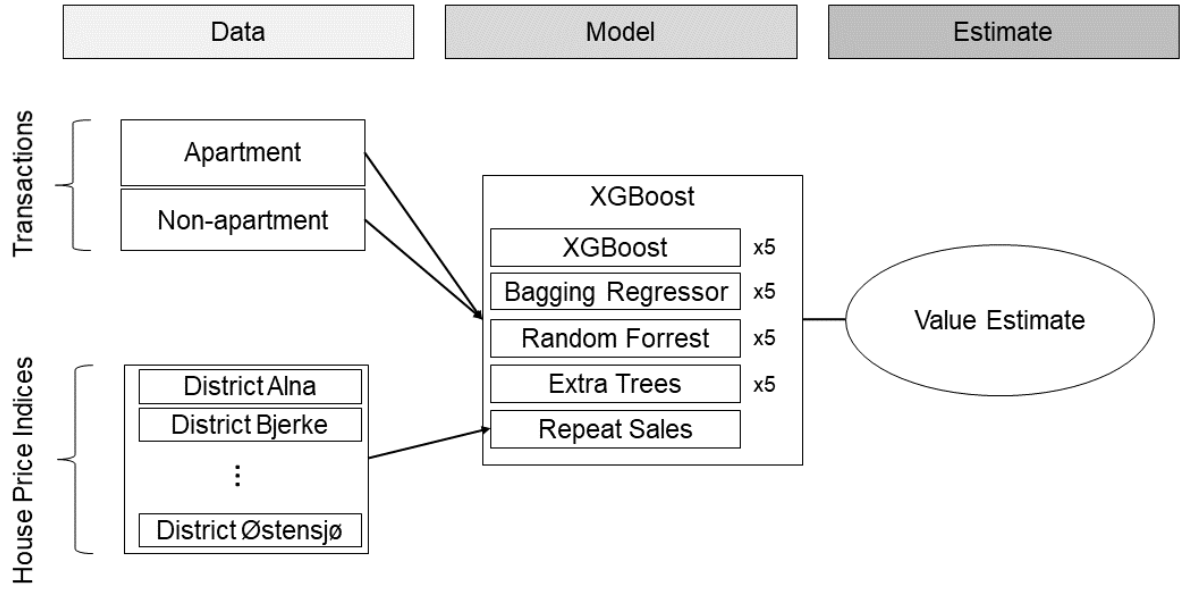


Figure 5: Example of a Decision Tree for the XGBoost-Method

The figure shows a binary decision tree produced by the XGBoost-method for a randomly selected dwelling. The dwelling has three-rooms and is a cooperative apartment in the district of Østensjø. The attribute names are given as: longitude (*f*0), latitude (*f*1), USM (*f*2), number of rooms (*f*4), days since sale (*f*6), build year (*f*7), and common debt (*f*8). The vertical distance between the nodes indicate the attribute importance.

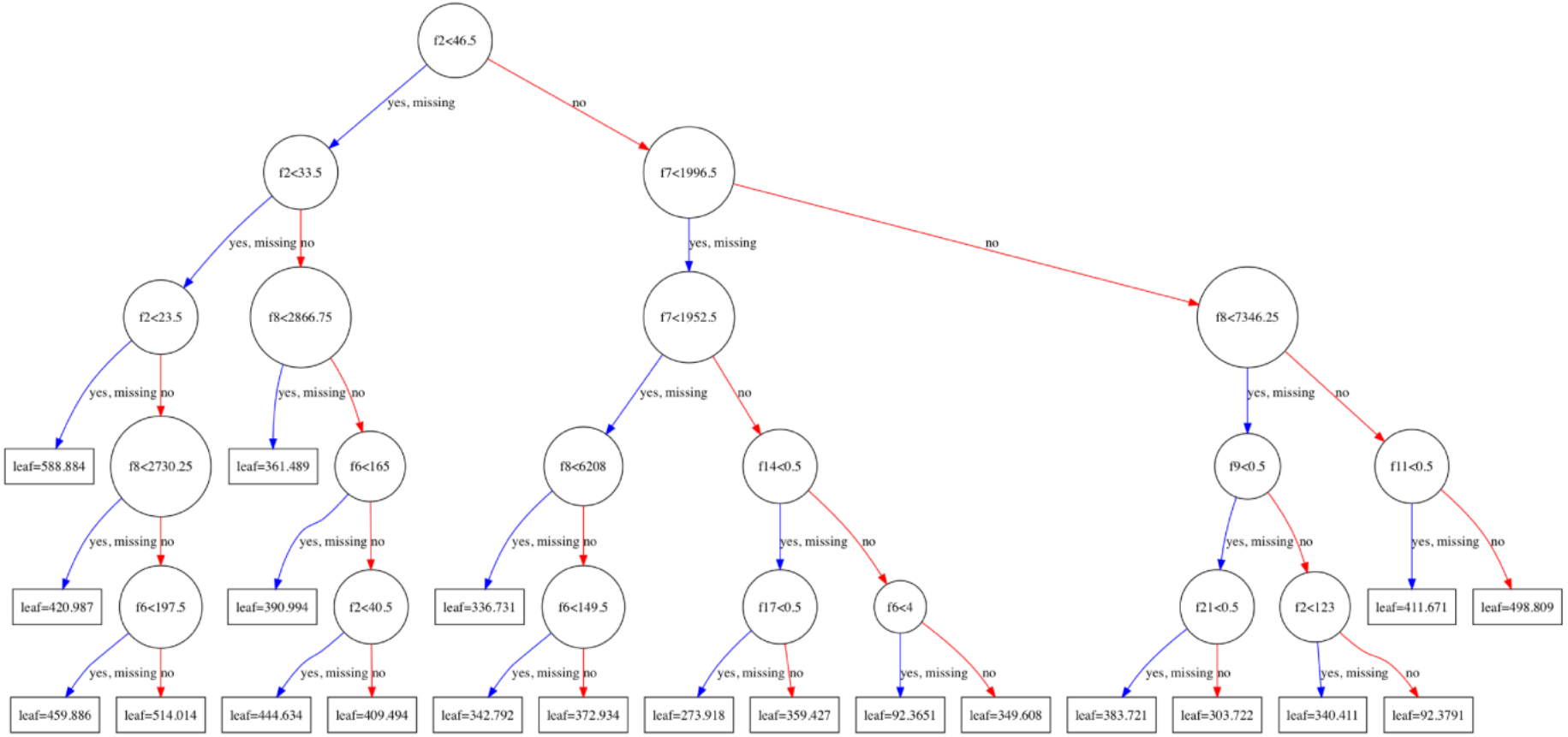
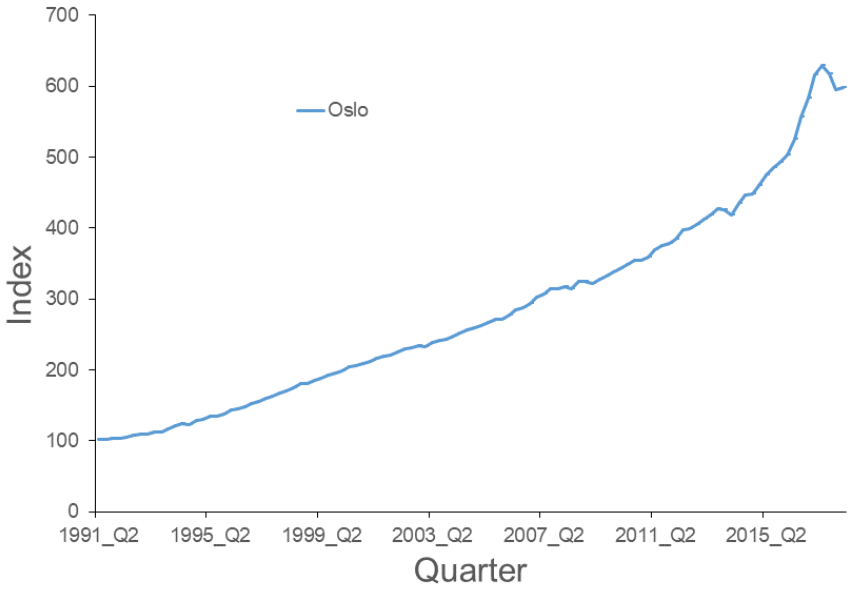


Figure 6: Aggregated Index for Oslo

The figure shows the index for the residential market of Oslo (Panel A) and its administrative districts (Panel B) based on RSM by Case and Shiller (1987).

Panel A: Repeat Sales Index for the Residential Market of Oslo



Panel B: Repeat Sales Index for the Administrative Districts of Oslo

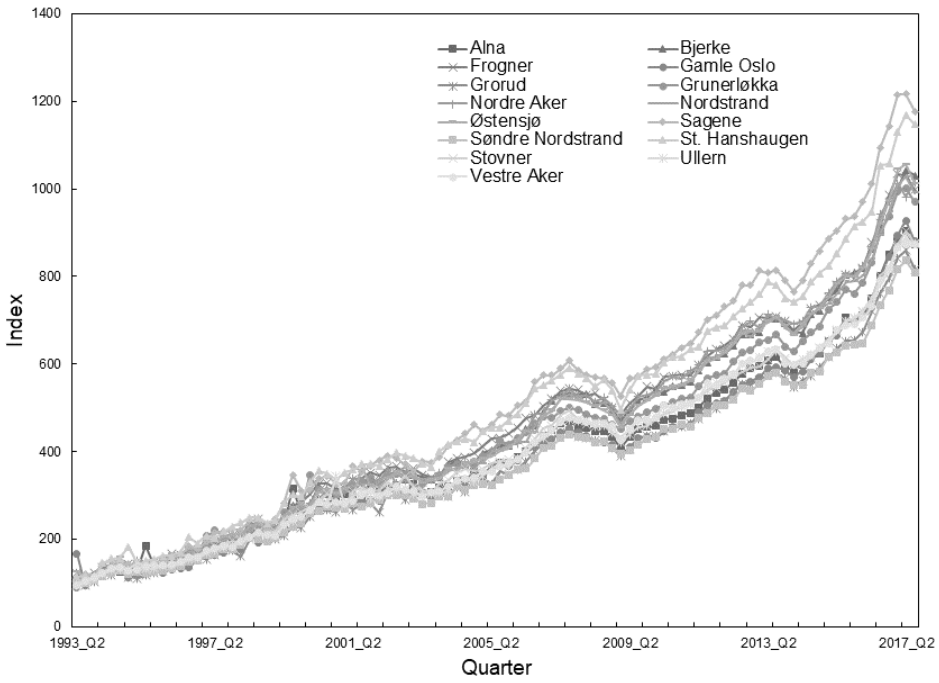


Figure 7: Correlations of Sub-models

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

Figure 8: Model Performance of the AVM for Various Number of Comparables

The figure presents the accuracy and training time of the automated valuation model (AVM) for varying number of comparable sales. The results are for the out-of-sample period February 2018 (470 transactions). Note that the *x*-axis is not linear in the number of comparable sales. Median absolute percentage error (MdAPE) is shown on the left-hand *y*-axis, whereas training time is plotted on the right-hand.

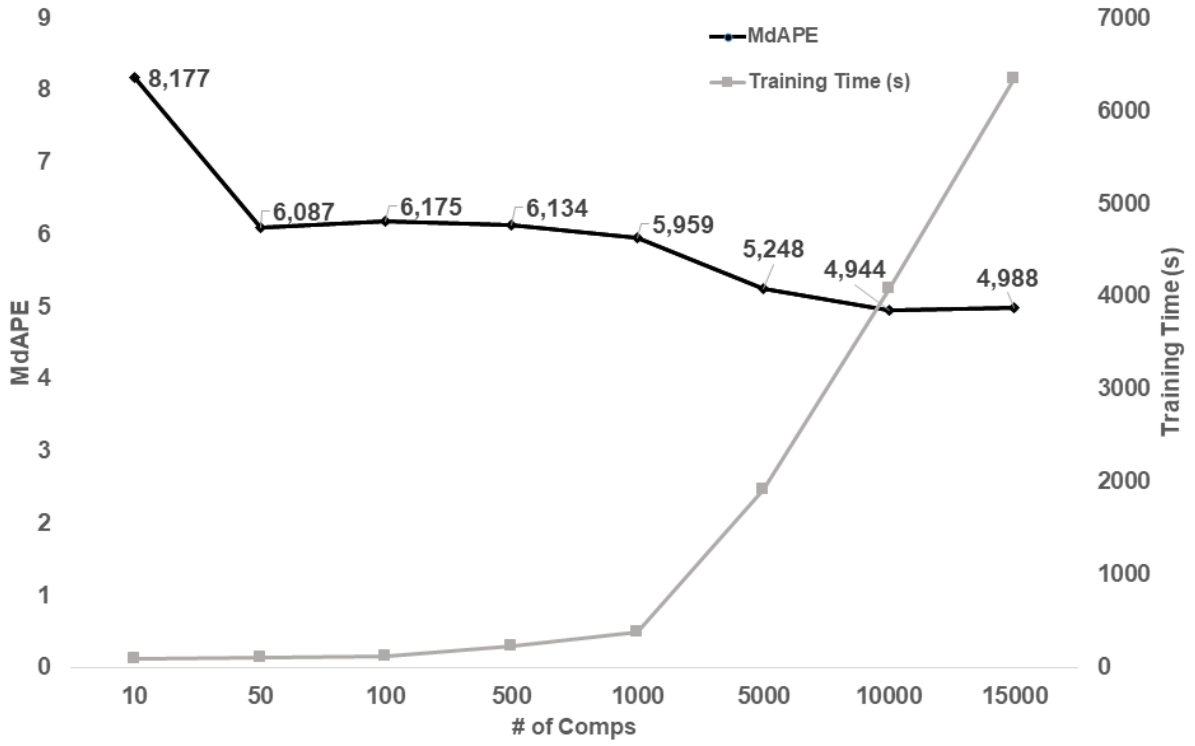
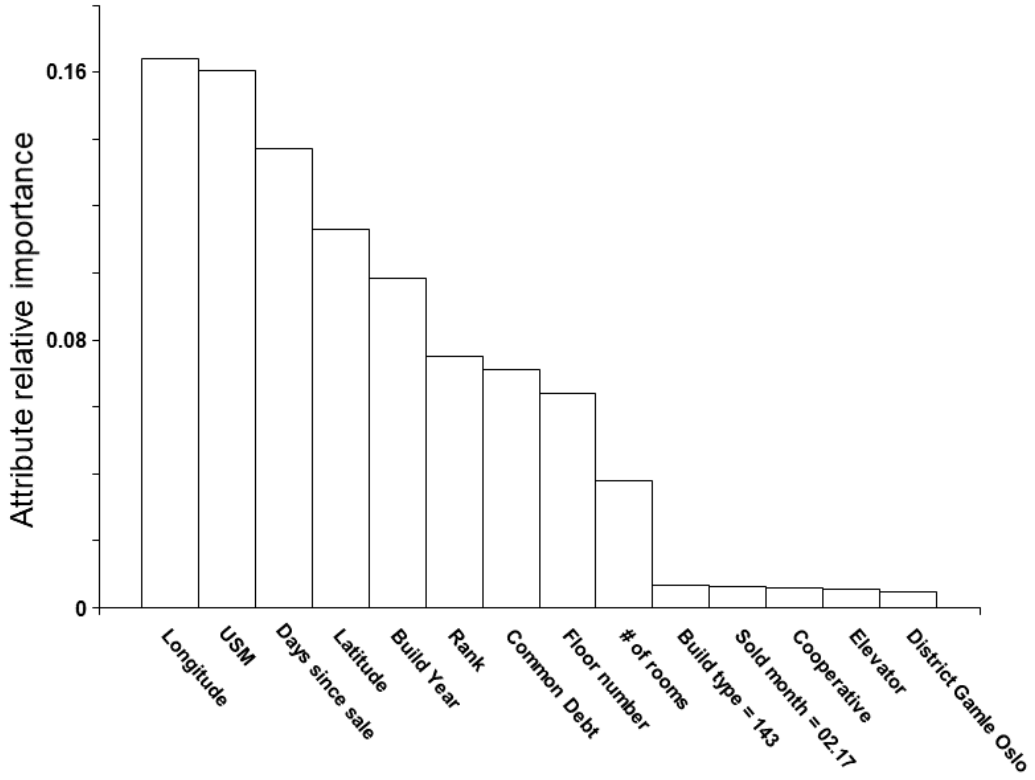


Figure 9: Attribute Importance

The figure shows the relative importance of the most important attributes when valuing a dwelling 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. The relative importance is calculated as a ranking score of the underlying XGBoost-model.



Tables:

Table 1: Overview of Exogenous Variables

The table gives an overview of the exogenous variables used by the attribute-based pricing methods. The attributes are classified as numeric and categorical variables.

Variable	Type	Description
USM	Numeric	The dwelling's usable square meters.
Log(USM)	Numeric	The natural logarithm of the USM.
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	The number of days since the occurrence of the transaction.
Rank	Numeric	A measure (increasing with distance) of proximity to target dwelling.
Build year	Numeric	The 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 from Subsection 2.1.
Elevator	Categorical	Whether or not the building of the dwelling has an elevator.

Table 2: Descriptive Statistics for Cooperatives and Condominiums

The table shows descriptive statistics on the *usable square meters (USM)*, *sales price* in millions of NOK (MNOK), and *common debt* in thousands of NOK (kNOK) for 4,089 cooperatives and 3,076 condominiums based on the enhanced transaction dataset.

	Cooperative			**Condominum**		
	USM	**Sales Price (MNOK)**	**Common Debt (kNOK)**	**USM**	**Sales Price (MNOK)**	**Common Debt (kNOK)**
Mean	58	3.7	230	67	5.1	41
St.Dev.	18	1.0	318	27	2.0	72
Min	14	1.3	0	16	1.1	0
25%	46	3.0	76	49	3.6	0
50%	59	3.4	141	60	4.5	4
75%	69	4.0	228	82	6.1	58
Max	183	12.9	2,900	339	18.0	1,200

Table 3: Descriptive Statistics on Unit Type

The table shows descriptive statistics on the size, measured by *usable square meters (USM)*, and the *sales price* of different dwelling types based the enhanced transaction dataset. Both measures are provided for apartments and non-apartments (row house, semi-detached house, and detached house) denoted in million NOK (MNOK). We show the parameters for mean, standard deviation (St.Dev.), median, as well as 1st and 2nd quartile.

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

Table 4: Hyperparameters

The table presents the descriptions and tuned values of the bagging hyperparameters bagging predictor (bp), random forest (RF), and extra trees (ET) in Panel A. Panel B shows the descriptions and tuned values of the hyperparameters of XGBoost, both as a model-stacker and as an underlying method. By running the algorithm repeated rimes it turned out that the tuned values provide the highest parameter stability.

Variable	**Applicable for**	**Description**	**Tuned Value**
Panel A: Bagging hyperparameters			
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 used when creating a split.	0.33
Panel B: XGBoost hyperparameters			
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 used when building a tree.	0.8
Colsample by tree		The share of attributes used when building a tree.	0.8
Evaluation Metric		The loss function the algorithm aims to minimize.	MAE

Table 5: Prediction Accuracy: Machine versus Man

The table shows the prediction accuracy for the automated valuation model (AVM), estate agents, and Zillow. The evaluations for the automated valuation model (AVM) in Panel A and *estate agents* in Panel B are based on the share of the predictions within the range of 5%, 10%, and 20% of the correct value or actual sales price, respectively, as well as the median absolute percentage error (MdAPE) and mean absolute percentage error (MAPE). For the AVM the out-of-sample performance refers to the period Q1 2018. For the estate agents, the data is from *Finn.no*, including 15,786 transactions in Oslo in 2016 and 2017 as well as 3,009 transactions for Q1 2018. Panel C shows the share of Zillow's *Zestimates* within 5%, 10%, and 20% of the actual sales price and the overall MdAPE for a selection of U.S. cities as reported by Zillow. The data is gathered from `www.zillow.com/zestimate/` on May 27, 2018.

	Within 5%	**Within 10%**	**Within 20%**	**MdAPE**	**MAPE**
Panel A: Comparable prediction accuracy of the automated valuation model (AVM)					
Q1 2018	46.9%	76.4%	96.3%	5.4%	7.2%
Panel B: Comparable prediction accuracy of estate agents					
2016/2017	47.8%	74.0%	96.5%	5.3%	7.6%
Q1 2018	70.6%	93.8%	99.5%	3.0%	5.1%
Panel C: Comparable prediction accuracy of Zillow					
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%	-

Table 6: Sub-model Accuracy Compared to the AVM

The table shows 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 for the out-of-sample performance period Q1 2018. The MdAPE for the RSM is only calculated for dwellings with previous sales, which constitutes 77% of the dwellings in the test set.

	Ensemble Learning				**Repeat Sales (RSM)**	**Traditional OLS**	**Stacked Model XGB-S**
	BP	**RF**	**ET**	**XGB**			
January 2018	6.23%	6.31%	6.21%	6.13%	8.93%	8.95%	**5.49%**
February 2018	5.60%	5.70%	5.71%	5.56%	8.86%	8.85%	**4.94%**
March 2018	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%**

Table 7: Prediction Accuracy of AVM with and without RSM

The table shows the share of the predictions within the range of 5%, 10%, and 20% of the sales price, the median absolute percentage error (MdAPE), and the mean absolute percentage error (MAPE) derived from the automated valuation model (AVM) with and without the repeat sales method (RSM) for the out-of-sample performance period 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%