Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Economics and Management
NTNU Business School

Elias Bjørgve
Cato Sandnes

Improving AVM Prediction Performance in the Housing Market: A Stacked Generalization Approach with Diverse Model Selection

Master's thesis in Economics and Business Administration
Supervisor: Are Oust
Co-supervisor: Ole Jakob Sønstebø
May 2023

NTNU
Norwegian University of
Science and Technology

Elias Bjørgve
Cato Sandnes

Improving AVM Prediction Performance in the Housing Market: A Stacked Generalization Approach with Diverse Model Selection

Master's thesis in Economics and Business Administration
Supervisor: Are Oust
Co-supervisor: Ole Jakob Sønstebø
May 2023

Norwegian University of Science and Technology
Faculty of Economics and Management
NTNU Business School

NTNU
Norwegian University of
Science and Technology

Preface

This thesis concludes the authors' Master of Science degree in Economics and Business Administration at the Norwegian University of Science and Technology (NTNU). The thesis has a scope of 30 ECTS, and is written by Elias Bjørgve and Cato Sandnes during the spring semester of 2023, both with a major in Business Analytics. The content presented in this thesis is the authors' sole responsibility.

First and foremost, we want to thank our supervisors Are Oust and Ole Jakob Sønstebø for excellent guidance and constructive feedback throughout the whole process. Their expertise in the field has been invaluable. We would also like to thank Ulf Jakob Flø Aarsnes and the rest of the team at Solgt.no for providing us with data. Lastly, we want to thank our friends and families for their help and support along the way.

Abstract

This thesis investigates the potential enhancement of predictive accuracy in Automated Valuation Models (AVMs) by employing diverse model selections in stacked generalization. The aim is to reduce inaccuracies in AVMs through the use of this advanced technique, which prior research has shown to be beneficial. Building on these findings, the study integrates multiple valuation techniques, including the Comparable Sales Method (CSM), Least Absolute Deviation (LAD), and XGBoost (XGB), to capture diverse patterns and insights. These techniques were combined in various ways to form three stacked models, which were then evaluated for their predictive performance through a comparative analysis between the three individual models and the three stacked models.

Our models were tested on a dataset consisting of 164,652 apartment transactions from Oslo, sold between 2007 and 2022, generating out-of-sample predictions using 25% of the data. The findings suggest that the stacked model involving XGB and CSM emerged as the most accurate of all models examined in the study, achieving a median absolute percentage error (MdAPE) of 5.35%. However, the study also reveals that most of the accuracy of the stacked models is derived from the XGB model. This best-performing individual model closely follows the stacked models, achieving an MdAPE of 5.47%. Despite these results, our stacked models face significant challenges related to computational complexity and interpretability, suggesting that further XGB model development might be more efficient.

Further tests were conducted to understand how different valuation techniques and stacked models perform with varying data sizes. The findings highlight the limitations faced by individual models when dealing with smaller datasets and reveal a decrease in the effectiveness of the stacked models as the size of the training data is reduced, suggesting stacked generalization might not improve prediction performance for smaller datasets. In a separate evaluation, our models were tested using an out-of-time test set on the full data, demonstrating that our initial methodology can be generalized.

The study concludes that using stacked generalization to combine distinct modeling techniques can improve the predictive performance of our models, but these improvements are relatively small and require a certain amount of data. Overall, this thesis offers valuable insights into the application of stacked generalization in real estate valuation and provides interesting findings in a relatively unexplored field.

Sammendrag

Denne oppgaven undersøker den potensielle forbedringen av prediktiv nøyaktighet til automatiserte verdsettelsesmodeller (AVM-er) ved å bruke metodisk forskjellige modeller i "stacked generalization". Målet er å redusere unøyaktigheter i AVM-er ved å bruke denne avanserte teknikken, som tidligere forskning har vist å være fordelaktig. Basert på disse funnene, anvender denne studien flere verdsettelsesteknikker, inkludert salgssammenligningsmetoden, "Least Absolute Deviation" (LAD) og "XGBoost" (XGB), for å fange forskjellige mønstre og innsikt. Disse teknikkene ble kombinert på ulike måter for å danne tre stablede modeller, som deretter ble evaluert for deres prediktive ytelse gjennom en komparativ analyse mellom de tre individuelle modellene og de tre stablede modellene.

Modellene våre ble testet på et datasett bestående av 164 652 leiligheter solgt i Oslo mellom 2007 og 2022, hvor det ble generert prediksjoner "out-of-sample" ved å bruke 25% av datasettet. Funnene tyder på at den stablede modellen som involverer XGB og salgssammenligningsmetoden gir den mest nøyaktige prediksjonen av alle modellene som ble undersøkt i studien, og oppnådde en median absolutt prosentfeil (MdAPE) på 5,35%. Studien viser også at mesteparten av nøyaktigheten til de stablede modellene er hentet fra XGB-modellen. Denne individuelle modellen med best ytelse følger tett de stablede modellene, og oppnår en MdAPE på 5,47%. Til tross for disse resultatene, står våre stablede modeller overfor betydelige utfordringer knyttet til beregningsmessig kompleksitet og tolkbarhet, noe som tyder på at videreutvikling av XGB-modellen kan være mer effektivt.

Ytterligere tester ble utført for å forstå hvordan ulike verdsettelsesteknikker og stablede modeller presterer med varierende datastørrelser. Funnene belyser begrensningene som individuelle modeller møter ved arbeid med mindre datasett og avslører en svekkelse i effektiviteten til de stablede modellene når størrelsen på treningsdataene reduseres. Dette antyder at "stacked generalization" kanskje ikke fører til forbedret prediksjonsevne for mindre datasett. I en egen undersøkelse ble modellene våre testet ved hjelp av et "out-of-time" testsett på de fullstendige dataene, som viste at vår første metodikk kan generaliseres.

Studien konkluderer med at bruk av "stacked generalization" for å kombinere metodisk forskjellige modeller kan forbedre den prediktive ytelsen til våre modeller, men disse forbedringene er relativt små og krever en viss mengde data. Samlet sett gir denne oppgaven verdifull innsikt i anvendelsen av "stacked generalization" i eiendomsvurdering og gir interessante funn i et relativt lite utforsket felt.

Table of Contents

List of Figures

List of Tables

Acronyms

AVM	Automated Valuation Model
BP	Bagging Predictor
CSM	Comparable Sales Method
ET	Extra Trees
EXF	Expert Function
GAM	Generalized Additive Model
GBM	Gradient Boosting Machine
HPM	Hedonic Pricing Method
KNN	K-Nearest Neighbors
LAD	Least Absolute Deviation
LASSO	Least Absolute Shrinkage and Selection Operator
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MdAPE	Median Absolute Percentage Error
MRA	Multiple Regression Analysis
MSE	Mean Squared Error
NOK	Norwegian kroner
OLS	Ordinary Least Squares
PPE(10)	Percentage Prediction Error at 10%
PPE(20)	Percentage Prediction Error at 20%
PPE(50)	Percentage Prediction Error at 50%
R^2	R-squared
RF	Random Forest
RMSE	Root Mean Squared Error
SHAP	SHapley Additive exPlanations
sqmP	Price per square meter
XGB	eXtreme Gradient Boosting *or* XGBoost

1 Introduction

In the continuously evolving landscape of the real estate industry, technological advancements have set in motion significant shifts in valuation techniques. Traditional methodologies, which heavily relied on human appraisers, have increasingly given way to more modern, automated methods. This change can largely be attributed to the rise of Automated Valuation Models (AVMs), which offer a more cost-effective, time-saving, and notably accurate means of valuing residential properties in comparison to traditional methods. However, despite the clear advantages they present, concerns have been raised as they may not provide an accurate representation of the local market conditions or consider the unique features of each property (Mooya, 2017; Tretton, 2007). This article examines these concerns and explores how stacked generalization, a technique used to combine multiple models, can improve the prediction performance of AVMs in the housing market.

The main idea underlying stacked generalization is to dynamically weight the input from base models based on their respective performances on a validation set (Smyth & Wolpert, 1997). Each base model captures different patterns or insights from the data, and the meta-model learns the optimal combination of these base model predictions to generate the final prediction output. By leveraging the diversity among the base models, the meta-model effectively adjusts the ensemble of predictions, potentially resulting in improved performance compared to using any single base model alone (Breiman, 1996). This approach can yield a more robust and reliable prediction model, benefiting stakeholders like real estate professionals, investors, and homebuyers. It can also help refine valuation techniques in the real estate industry and promote informed decision-making in the housing market. Therefore, our primary research question is whether the use of stacked generalization with diverse model selection leads to better prediction performance.

Previous research has investigated the use of stacked generalization to improve prediction performance in real estate valuation (Birkeland et al., 2021; Kansara et al., 2018; Nnadozie et al., 2022; Truong et al., 2020). These studies have demonstrated that models using stacked generalization not only enhance the stability of predictions, but also outperform other models in terms of predictive accuracy. As the literature up to this point has mainly focused on integrating various machine learning methods, our study aims to extend the literature by investigating the implementation of stacked generalization with methodically diverse modeling techniques. Specifically, we integrate a traditional appraisal method, the Comparable Sales Method (CSM), which, to our knowledge, has not been used in this context before. In addition, we include a hedonic regression in Least Absolute Deviation (LAD), and a machine learning technique in XGBoost (XGB). The selection of diverse models aims to leverage the diversity among the base models and ensures that a wide range of approaches is considered, which may lead to further improvement in performance. Furthermore, by incorporating the CSM into a stacked generalization framework, we are exploring a new direction in real estate valuation. This approach could provide valuable insights into the potential benefits and challenges present in this procedure.

The dataset used in this study consists of 164,652 apartment transactions in Oslo,

spanning from May 2007 to December 2022. Our models were trained on 75% of the data, and tested out-of-sample on the remaining 25%. To answer the research question, we developed AVMs for the three individual models. We then fed their predictions into an XGB meta-model as part of the stacked generalization process. This enabled the development of three additional stacked models: (XGB + CSM), (XGB + LAD), and (XGB + CSM + LAD). All six models were then compared against each other using a range of different evaluation metrics. Additional insights were gained by examining both the feature importance of each model, and the discrepancy between the predicted and the actual values. Lastly, we conducted a series of tests designed to simulate the performance of our models out-of-sample across a variety of smaller training sets, each representing a wide range of potential scenarios.

Our findings revealed that the application of stacked generalization with methodically diverse models led to a marginal improvement in prediction performance. Notably, the XGB, when used as a standalone model, already produced robust results when tested on an extensive dataset from Oslo's housing market. Our analysis also indicated that the stacked models derive most of their accuracy from the XGB base model, suggesting a strong reliance on this particular algorithm. However, the practicality of this remains a concern due to the computational expense of stacked generalization. Despite this, integrating XGB with the CSM through stacked generalization resulted in a minor but significant improvement across all performance metrics, with median absolute percentage error (MdAPE) reducing from 5.47% to 5.35%. When experimenting with smaller datasets, we found that stacked generalization might be less effective as the dataset decreases in size. The stacked model, composed of XGB and CSM, retained its position as the best-performing model until the training data was reduced to approximately 12,000 observations. Beyond this threshold, the individual XGB emerged as the best-performing model and showed its dominance.

The limited enhancements observed with stacked models in this study, along with their time-consuming implementation, raise questions about the practicality of using stacked generalization in real-world applications. Nonetheless, as this field is relatively unexplored, further research is required to substantiate these findings.

To provide a comprehensive understanding of our work, we will explore the following key sections in our study: In Section 2, we offer a comprehensive review of related literature, establishing the theoretical foundation for our research. Section 3 explores the dataset and details the cleaning process and rationale for variable selection. Section 4 illustrates the development of our AVMs, including the application of the stacked generalization approach and details about the individual models, along with the SHAP framework used to get feature importances, and an overview of the evaluation metrics used in the study. In Section 5, we then evaluate and discuss the models' performance under various conditions, while Section 6 concludes the study with final remarks and suggestions for future research.

2 Literature Review

The following section introduces the relevant literature for the study. This includes the use of stacked generalization in previous work, followed by a deep dive on each individual model used in this study. Alternative models are also discussed, justifying the model choices.

2.1 Stacked Generalization

Stacked generalization is an ensemble learning technique that combines the predictions of multiple base models to make a final predictive model with improved accuracy and robustness. It was pioneered by Wolpert (1992) as a way to address the limitations of single-model approaches for classification tasks, and later refined by Breiman (1996) who used it for regressions. He also found that stacking models that are not overly similar yielded better results. The seminal work of Smyth and Wolpert (1997) shed light on the applicability of stacked generalization to unsupervised learning tasks. Their findings provided evidence that stacked generalization could also be effectively employed in unsupervised learning settings, and expanded the horizons of stacked generalization beyond supervised learning. This unlocked new possibilities for its utilization in various unsupervised learning tasks, thereby contributing to the advancement of the field of machine learning.

In the context of real estate valuation, there have been several studies that have implemented stacking. Graczyk et al. (2010) conducted a research study that investigated the effectiveness of six different algorithms, including two neural network algorithms, two decision trees, linear regression, and support vector machine, when used as base models in ensemble learning applied to real estate appraisal. Three meta-learners, namely Additive Regression, Bagging, and Stacking, were employed to create ensembles, and their performance was compared in terms of prediction accuracy. The findings revealed that bagging was the most stable technique, while stacking yielded the lowest prediction errors in most cases. In a study conducted by Truong et al. (2020), the prediction of house prices in Beijing was investigated using three distinct machine learning models, specifically Random Forest (RF), XGB, and LightGBM. Subsequently, stacked generalization was employed to compare the predictive performance of the individual models against the stacked model. The findings revealed that RF outperformed the stacked model on the training set and was the best-performing model among the three individual models. However, it was noted that RF exhibited a tendency towards overfitting, whereas the stacked model demonstrated superior performance on out-of-sample data, indicating its potential for enhanced generalization performance. Birkeland et al. (2021) proposed an AVM for the residential real estate market in Oslo, utilizing a stacked ensemble approach that combined multiple base models including Bagging Predictor (BP), RF, Extra Trees (ET), and XGB, with another XGB model serving as the meta-model. The stacked ensemble model demonstrated superior performance compared to the individual models when evaluated on out-of-sample data, as evidenced by low MdAPE and a high percentage of predictions within 20% of the true values. However, the

authors acknowledged that the stacked generalization approach is computationally expensive and may pose challenges in interpreting the output due to potential hidden factors that could influence the model's predictions. Further insights can be gained by incorporating Linear Regression (Kansara et al., 2018) and ElasticNet Regression (Nnadozie et al., 2022), in addition to a subset of the previously mentioned base models. In both studies, XGB emerged as the most effective individual model, with the stacked model demonstrating superior performance.

2.2 Comparable Sales Method and Its Variations

CSM is a widely used valuation approach by real estate professionals to determine the value of a property. This method involves analyzing recent sales of comparable properties located in the same area as the subject property, and adjusting the sales prices based on differences in various relevant factors such as size, location, age, and other relevant features. This approach leads to an estimation of the value of the subject property. In its simplest form, commonly referred to as an expert algorithm, it imitates the appraisal process undertaken by professional appraisers. The algorithm finds comparables that match the target property in its characteristics, and computes the average value of their transaction price to estimate a prediction for the target property. Examples of using such approach in recent literature include Larraz et al. (2021); Stang et al. (2022); Trawiński et al. (2017), albeit with results that were surpassed by more complex models. However, Stang et al. (2022) showed that their Expert Function (EXF) was better suited to estimate the price in some of the districts in Germany, especially those with fewer observations.

An alternative approach is to use a K-Nearest Neighbors (KNN) algorithm to calculate distances between the sample and target property to find a number of similar properties (K). Antipov and Pokryshevskaya (2012) and Valier (2020) got more promising results using this approach rather than the simpler expert algorithm, although the tree-based machine learning algorithms still yielded better results. Building upon this approach, Oust et al. (2020) proposed a variation by incorporating a K-means algorithm as a function of longitude, latitude and price per square meter on the training set to generate more homogeneous artificial districts. Subsequently, K-Nearest Neighbors was applied to the test set to classify dwellings based on the newly constructed districts, employing the Haversine formula to measure distance. The utilization of this innovative approach resulted in hedonic regressions that were enhanced spatially, effectively serving as a CSM. Birkeland et al. (2021) utilized a similar method with the Haversine formula, known as the comparable pricing method, to train ensemble learning sub-models of an AVM for estimating the value of different types of dwellings. The method involves selecting geographically nearest transactions of the same type of dwelling as the target property, with a ranking variable based on proximity, enabling the ensemble methods to recognize recent sales of similar properties located in close proximity.

2.3 Hedonic Regression Models

One of the earliest and most widely used AVMs is the Hedonic Pricing Model (HPM) based on Multiple Regression Analysis (MRA) proposed by Rosen (1974). This model is often used as a benchmark for testing other models. Recent studies, such as the one conducted by Doumpos et al. (2021), have shown that even simple linear models can be effective tools for building accurate AVM systems, as long as they account for spatial effects. The study found that their locally weighted linear regression models, including Ordinary Least Squares (OLS), Least Absolute Shrinkage and Selection Operator (LASSO), and LAD, performed well in capturing spatial price variation compared to global and unweighted local approaches. These models were also found to be easy to implement in a large-scale context and required only a small sample of data for model fitting.

However, Kilpatrick (2011) highlighted the limitations of hedonic models, noting that these models rely on certain assumptions, such as independent and identically distributed data, strict exogeneity, no multicollinearity, spherical errors, and normality, which, if violated, can lead to biased or unreliable results. He suggested it was imperative to explore alternative methodologies in real estate that possess robust statistical characterization, while avoiding the strict assumptions of hedonic models.

LAD was first introduced by Koenker and Basset (1978) as an alternative to the more commonly used linear regression models. The study conducted by Yoo (2001) found that the LAD model generates more robust and accurate estimates of hedonic regression models compared to the conventional least squares method. This was demonstrated by applying the LAD method to a dataset of residential property sales in Seoul, South Korea. These findings demonstrate the advantages of using the LAD method for HPM analysis and its potential to provide more reasonable results and robustness of such models.

2.4 Boosting Techniques

Boosting is an ensemble technique first introduced by Schapire (1990). It involves training decision trees sequentially, where each subsequent tree is fitted to improve the errors made by the preceding trees. Later, this technique was developed further and the AdaBoost method was introduced by Freund and Schapire (1995), which was widely adopted in various machine learning applications. Gradient Boosting Machine (GBM) was developed by Friedman (2001) as a generalization of AdaBoost to achieve higher accuracy, handle complex interactions between variables, and optimize any differentiable loss function. However, this algorithm can be computationally expensive and can also lead to overfitting. To address these issues, Chen and Guestrin (2016) implemented the eXtreme Gradient Boosting (XGB) algorithm. The model is a powerful and computationally efficient implementation of the gradient boosting model. It is designed to address overfitting issues by imposing regularization on the model structure, which limits its complexity and generalizes better for out-of-sample data.

The XGB model has since been popular in AVMs and is well documented in previous literature. Kumkar et al. (2018) conducted a comparative analysis of four tree-based ensemble methods, including XGB, to evaluate their efficiency in the appraisal of property values in Mumbai, India. The results of the study demonstrate that the XGB model outperforms the other three models in terms of accuracy and predictive power. Stang et al. (2022) found that due to its flexibility the XGB function achieved the highest overall accuracy compared to the OLS regression, Generalized Additive Model (GAM), and their EXF in their AVM for real estate properties in Germany. Hjort et al. (2022) used a large dataset containing 126,719 observations where their variant of the XGB model achieved the best results on 4 out of 5 of the performance metrics compared to other models, where only their RF model performed better on median error percentage. Although XGB has been shown to be a superior model in some contexts, it has been reported to be outperformed by at least one other model in various comparisons (Law et al., 2019; Sangani et al., 2017; Wang et al., 2020).

3 Data

This study utilized data provided by the Norwegian proptech company Solgt.no. The dataset includes detailed information of housing transactions listed on Finn.no, the largest online marketplace for private properties in Norway. The transactions are from the Norwegian capital, Oslo, and span from May 2007 to December 2022. This section provides an overview of the key variables present in the dataset, as well as the techniques used for filtering and cleaning the data to ensure accurate analysis and interpretation.

3.1 Data Pre-Processing

The dataset was first examined to assess the distribution of housing types before delving into specific variables. Out of the total 199,370 transactions, a significant majority of 87.7% were identified as apartments, as shown in Table 1. This finding aligns with data from Bydelsfakta (2023b), which indicates that apartments make up 76% of all housing types in Oslo. It is worth noting that the higher proportion of apartments in our dataset may be attributed to the fact that Solgt.no exclusively deals with homes valued at less than NOK 13 million (Solgt.no, 2023).

Considering the known price disparities between housing types, focusing on one specific type of housing is considered advantageous when developing an AVM. Apartments generally have a higher price per square meter compared to other types of housing, even though standalone houses often command a higher total price (Oslo kommune, 2023a). These disparities in price are known to vary across districts, leading to an uneven distribution of housing types. By focusing exclusively on apartments in our analysis, we aim to create a more homogeneous market and reduce the number of unsatisfactory predictions. This deliberate decision allows us to maintain a consistent and focused study.

In preparation for analysis, data cleaning was conducted as a necessary step. The dataset's transactions revealed missing values in certain variables and the possibility of typographical errors in the source ads, requiring thorough attention during the cleaning process. Consequently, the focus of the cleaning process was on addressing these issues, with careful consideration given to minimizing data loss when removing missing values, as such a loss could have significant implications for further analysis.

The approach taken in this study was partially influenced by the methodology of Helgaker et al. (2022), who employed a strict cleaning process to eliminate unrealistic observations from an earlier version of the same dataset. However, some modifications were made to the process in order to retain a greater quantity of data. A summary of the full cleaning process is provided in Table 1.

Initially, Helgaker et al. (2022) removed observations with sales prices exceeding twice the list price or falling below half the list price. This step did not affect our dataset, as it was either corrected before we received the data or handled by keeping only apartments. Instead, we removed observations with sales prices below 500,000 to exclude potential housing types, such as garages, wrongly labeled as

apartments. Subsequently, we retained observations with living areas between 9 and 300 square meters to further remove wrongly labeled apartments. For instance, garages are often registered with a living area of 0 or 1 square meter in the data. Apartments with living areas exceeding 300 square meters are rare and typically due to typing errors. Moreover, we kept observations with build year newer than 1600 and observations with less than 10 bedrooms, despite not using any of these variables in the final dataset. Doing so enhances the dataset's quality, as inaccuracies in one variable may indicate issues in other variables.

Next, we removed observations with faulty coordinates. This included instances where latitude and longitude values were incorrectly interchanged, causing a misplacement of data points, as well as observations that were geographically distant from our area of focus, Oslo. For example, due to these errors, some coordinates corresponded to locations in Bergen and Lillestrøm. Subsequently, we retained observations with floor number less than 21 and observations with fewer than four bathrooms. Instead of the latter step, Helgaker et al. (2022) removed all observations with zero bathrooms, which led to a considerable loss of data. Since "bathrooms" is not listed as a key attribute on Finn.no, many realtors do not include the number of bathrooms in their housing advertisements. This can lead to observations in the dataset appearing with "0" as the number of bathrooms. Moreover, the number of bathrooms is rarely used when valuing a property. Removing such observations would risk a loss of valuable information. Hence, we kept those observations with the intention of not using the bathrooms variable in the final dataset. Lastly, we removed the two observations within the Marka district.

Table 1: Data pre-processing steps of the original dataset

Data pre-processing	Observations
Apartment	174,815
Standalone house	10,331
Row house	7,898
Semi-detached house	6,066
Other	260
Original data	199,370
Keep only apartments	174,815
Keep observations with sales price more than 500,000	174,598
Keep observations with living area larger than 9 m^2 and less than 300 m^2	174,484
Keep observations with build year newer than 1600	174,198
Keep observations with less than 10 bedrooms	165,077
Remove observations with faulty coordinates	164,674
Keep observations with floor less than 21	164,671
Keep observations with less than 4 bathrooms	164,654
Remove observations within the district "Marka"	164,652
Final data	164,652

3.2 Data Overview and Variable Selection

The following subsection includes a detailed explanation of the variables chosen from the original dataset, along with the rationale for their selection. Additionally, it describes how some of the existing variables were transformed or calculated into new variables to suit the specific needs of the analysis. Table 2 shows the descriptive statistics for all the variables in the final dataset.

Table 2: Summary statistics for the final dataset

Variable	Unit	Mean	St. Dev.	Min	Max	Type
Price per square meter	NOK (1000)	65.17	23.29	12.50	278.33	Numeric
Days since	days (1000)	3.49	1.38	0.00	5.68	Numeric
Living area	m^2	66.31	25.04	10.00	299.00	Numeric
Latitude	degrees	59.92	0.03	59.82	59.98	Numeric
Longitude	degrees	10.78	0.06	10.63	10.95	Numeric
Latitude in radians	radians	1.05	0.00	1.04	1.05	Numeric
Longitude in radians	radians	0.19	0.00	0.19	0.19	Numeric
	%	N				
Size categories						
10-49m^2	23.31	38,380				Binary
50-69m^2	40.40	66,511				Binary
70-99m^2	27.39	45,101				Binary
100-139m^2	7.29	12,003				Binary
140-179m^2	1.28	2,113				Binary
Above 180m^2	0.33	544				Binary
District						
Alna	5.88	9,678				Binary
Bjerke	4.37	7,188				Binary
Frogner	12.55	20,672				Binary
Gamle Oslo	11.96	19,691				Binary
Grorud	2.94	4,836				Binary
Grünerløkka	13.36	21,995				Binary
Nordre Aker	4.86	8,004				Binary
Nordstrand	4.78	7,868				Binary
Sagene	11.54	19,005				Binary
St. Hanshaugen	8.77	14,444				Binary
Stovner	2.58	4,257				Binary
Søndre Nordstrand	2.47	4,066				Binary
Ullern	3.83	6,305				Binary
Vestre Aker	3.68	6,063				Binary
Østensjø	6.43	10,580				Binary
Sales year						
2007	0.01	16				Binary
2008	1.56	2,564				Binary
2009	2.30	3,795				Binary
2010	2.89	4,756				Binary
2011	4.81	7,921				Binary
2012	6.72	11,067				Binary
2013	7.17	11,812				Binary
2014	7.02	11,561				Binary
2015	7.84	12,907				Binary
2016	7.19	11,838				Binary
2017	7.49	12,338				Binary
2018	8.36	13,763				Binary
2019	9.27	15,257				Binary
2020	9.53	15,689				Binary
2021	10.16	16,731				Binary
2022	7.67	12,637				Binary
Sales month						
January	6.11	10,066				Binary
February	7.28	11,983				Binary
March	9.54	15,704				Binary
April	8.44	13,891				Binary
May	10.80	17,784				Binary
June	11.20	18,446				Binary
July	4.35	7,158				Binary
August	10.04	16,537				Binary
September	10.75	17,700				Binary
October	9.48	15,612				Binary
November	8.08	13,300				Binary
December	3.99	6,571				Binary
Balcony	67.99	111,954				Binary
Garage	36.34	59,828				Binary
Quiet	59.78	98,432				Binary
View	34.99	57,618				Binary
Renovation object	6.64	10,938				Binary
Two bathrooms or more	4.47	7,365				Binary
New apartment	6.93	11,417				Binary
High floor and no elevator	5.05	8,321				Binary
First floor and inner city	8.18	13,465				Binary

Note: May 24th, 2007 serves as the starting point of the dataset. The variable "days since" counts the days that have elapsed since this date, and was normalized to a range of 0 to 1 thereafter. Price per square meter was log transformed, and the radians for latitude and longitude were scaled to unit variance.

To optimize the accuracy of the value estimate for housing, the use of price per square meter as the dependent variable was determined to be more appropriate than the regular sales price variable. This decision was based on the understanding that the size of a property significantly impacts its sales price. When developing an AVM for Norwegian data, it is imperative to incorporate both the sales price and the common debt into the calculation of the price per square meter. This approach, initially proposed by Oust et al. (2020), is essential in accurately capturing the total cost associated with acquiring a property, and thereby providing a more comprehensive representation of the property's market value. Furthermore, using the natural log as presented in Equation (1) has several advantages over using the actual price per square meter as the dependent variable. Firstly, it helps to normalize the distribution of the dependent variable, which can improve the accuracy of the model's predictions. Additionally, the natural log transformation can help reduce the influence of extreme values, which may distort the analysis.

$$ln(price\ per\ square\ meter) = ln(\frac{sales\ price_i + common\ debt_i}{living\ area_i}) \tag{1}$$

The regular variable exhibited a wide range of values, with the minimum and maximum prices recorded as NOK 12,500 and NOK 278,331, respectively. This substantial variability within the dataset underscores the presence of disparities and justifies the need for a transformation.

To represent size, the dataset included both gross area and living area. Although both variables were available, we made the decision to use living area as the primary measure of size for property valuation purposes. This was based on the belief that living area provides a better representation of the usable space for potential occupants, and thus, is more relevant to property value. As the study involves the development of various models, it was necessary to make certain adjustments. While the XGB model demonstrates strong performance in handling continuous variables, both CSM and LAD have some limitations in this regard, as discussed in greater detail in Section 4.2 and 4.3, respectively. In the context of hedonic regression models, categorical variables are often preferred due to their linearity. Thus, for the LAD and CSM, we adopt a categorical approach. Despite our CSM utilizing a KNN algorithm and not being linear in form, our experimentation revealed that the categorical approach yielded superior results.

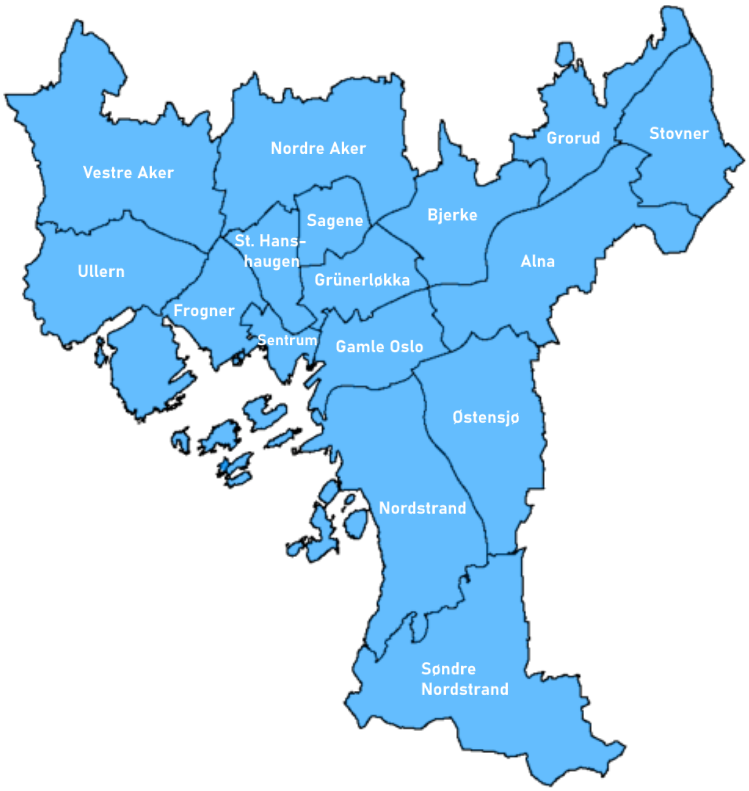


Figure 1: Districts of Oslo

When appraising a property, the location plays a crucial role in determining its value. The original dataset encompasses several means of expressing location, such as latitude and longitude coordinates, postcode, address, and district information. Extracting information from the full address is a complex task, and since postcodes have already been utilized as a baseline to derive the district variable, which is more interpretable, we opted to exclude both address and postcode from our analysis. Although the district variable may not be as precise as the coordinates, it was necessary to employ it as an alternative in the LAD model to ensure linearity. To enhance the accuracy of the district variable, another approach could involve generating new districts based on the coordinates using K-Means (Oust et al., 2020). However, as this would only be helpful for the LAD model, which is not the main focus of our study, we proceeded with using the district variable for this model.

It is worth noting that there are 15 administrative districts in Oslo (Oslo kommune, 2023b), however, the original dataset also comprises information on "Sentrum" and "Marka". As mentioned in Section 3.1, we excluded observations from *Marka* as they fall outside the outer line in Figure 1. *Sentrum* refers to the city center and lacks its own administration. Due to the small number of observations and this characteristic, we reallocated them to *St. Hanshaugen*.

For the XGB and the CSM, we deemed it more appropriate to utilize the latitude and longitude coordinates to generate more accurate predictions. In order to calculate the distances in the CSM, we generated two new variables by utilizing Equation (2) and (3) to transform the coordinates from degrees to radians. To avoid potential numerical instabilities during the computation, we utilized StandardScaler from the

machine learning library scikit-learn. This library provides a wide range of functions for data pre-processing, model selection, evaluation, and other essential machine learning tasks. The StandardScaler function standardizes the data by subtracting the mean and scaling it to unit variance (Pedregosa et al., 2011), ensuring accurate distance calculations and precise analysis results.

$$rad_{lat} = \frac{latitude * \pi}{180} \tag{2}$$

$$rad_{lng} = \frac{longitude * \pi}{180} \tag{3}$$

Despite being geographically constrained to Oslo, the dataset reveals a considerable variation in prices across various districts. Figure 2 depicts the distribution of the price per square meter based on latitude and longitude coordinates, indicating a discernible trend that prices increase as the proximity to the city center intensifies. The five districts *Gamle Oslo*, *Grünerløkka*, *Sagene*, *St. Hanshaugen* and *Frogner* represent the "inner city" (Oslo kommune, 2023b), where apartments are generally smaller due to their central location (Bydelsfakta, 2023c). This leads to a relatively higher price per square meter compared to that in other areas in Oslo. Additionally, the graph demonstrates a modest increase in prices for the western regions of Oslo, which can be attributed to the relatively greater affluence of the local population in contrast to their counterparts who live in the eastern regions (Bydelsfakta, 2023d). Specifically, Frogner and St. Hanshaugen have the highest average price per square meter, while the lowest are observed in Stovner and Søndre Nordstrand, which align with the statistics provided by Bydelsfakta (2023a).

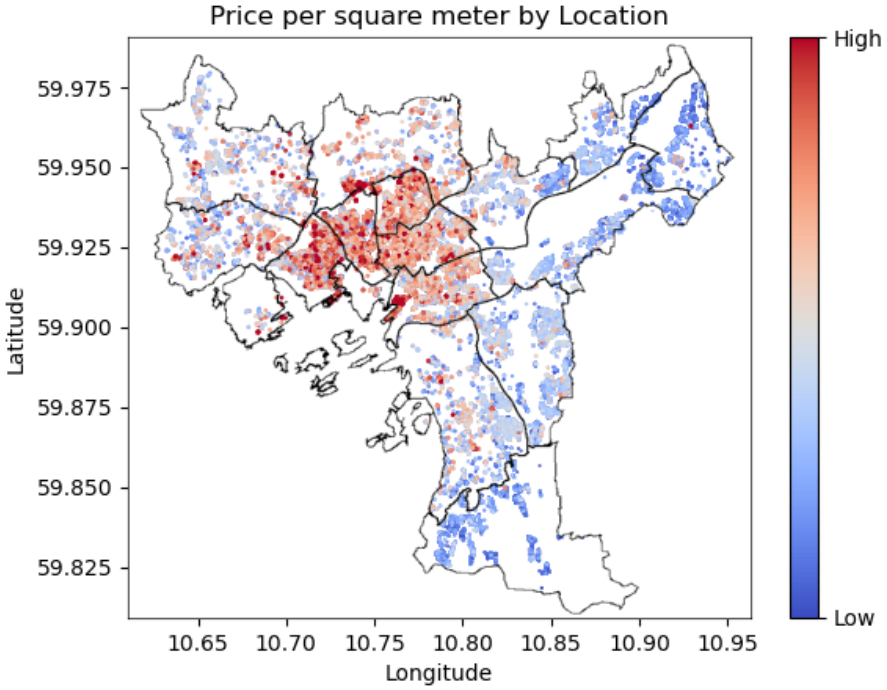


Figure 2: Price per square meter based on latitude and longitude coordinates

The original dataset contained a variable denoting sales date. To facilitate the implementation of the hedonic model, we partition this variable into its year and month components and apply one-hot encoding on the resulting data. For the other models, we construct a new variable, referred to as "days since", which records the number of days that have elapsed since the first transaction in the dataset. To enhance computational efficiency and minimize the influence of the CSM, we employ the MinMaxScaler function from the scikit-learn library, which rescales the data to fall within the range of 0 to 1 (Pedregosa et al., 2011). This rescaling process does not adversely affect the performance of the XGB model.

The following set of variables pertains to the facilities of dwellings. To provide insights into the features of apartments, a collection of binary variables is incorporated. These variables are sourced from the Finn.no advertising interface and can be selected to refine search results while browsing the platform. It is important to note that their inclusion in an advertisement is optional, and different real estate agents may have varying perspectives on which facilities to highlight as relevant. It is plausible that some advertisements may lack certain facilities despite the underlying apartment possessing the corresponding attribute. Therefore, a degree of uncertainty is associated with the facility variables, and it is crucial to carefully consider which facilities are relevant when explaining house prices. In total, there are 22 facilities available for consideration. After assessing the quality and relevance of each variable, we selected only the most significant facilities, resulting in a set of four facility variables, including "balcony", "garage", "view" and "quiet". Each of these variables exhibits a distribution that appears reasonable and intriguing from a predictive standpoint, as displayed in Table 2.

The dataset also included a binary variable indicating whether the examined apartment is a renovation project or not. This variable was assigned a value of 1 if certain keywords[1] in the ad title suggested the apartment required renovation, and 0 otherwise. Typically, renovation projects require additional investments after the initial purchase, which results in their value being lower than that of an equivalent apartment that does not require these extra costs. Although there may be some uncertainty associated with this variable, it is common to include this type of information in the ad title to attract certain types of homebuyers. Out of the 164,652 transactions remaining after the cleaning process in Table 1, 10,938 were for renovation projects, which likely captures most apartments in need of renovation.

Ultimately, we proceeded to generate new variables for inclusion in our dataset. As previously discussed in Section 3.1, we opted not to incorporate the "bedrooms", "bathrooms" and "build year" variables in our final dataset. This decision was informed by several considerations. Specifically, we noted that the "bedrooms" variable exhibited a high degree of correlation with the "living area" variable, as shown in Figure 8. Consequently, we concluded that including this variable would likely introduce redundant information into our analysis. Additionally, we observed that a substantial proportion of observations in the "bathrooms" variable exhibited unrealistic values, with a large number of apartments reportedly having no bathrooms. However, despite retaining these observations, we concluded that incor-

[1]The variable included word-recognizing for the keywords "oppussing" and "renovering" (which translates to "remodeling" and "renovation") in the ad title.

porating this variable into our analysis would introduce more confusion than clarity. As an alternative, we created a binary variable to derive meaningful insights from the "bathrooms" variable. The binary variable takes a value of 1 if an apartment has two or more bathrooms, and 0 if it has one or zero bathrooms. This allows us to eliminate the unreliable values from the original variable without removing the observations from the dataset.

Regarding the "build year" variable, we chose to differentiate between new and old dwellings by establishing a binary variable. We assigned a value of 1 to apartments built in 2010 or later, and 0 otherwise. This approach is motivated by the fact that younger dwellings depreciates more rapidly than older dwellings (Grether & Mieszkowski, 1974), acknowledging that beyond a certain age, the influence of age on the property value becomes less evident.

Upon examination of the remaining unused variables in the dataset, we identified two variables, namely the number of floors and the presence of an elevator. While not of individual interest, they could be informative in combination with other variables. To extract valuable insights from these variables, we generated two new binary variables. The first binary variable was constructed such that it took a value of 1 if the apartment was situated on the fifth floor or higher and lacked an elevator, and 0 otherwise. The second binary variable was assigned a value of 1 if the apartment was located in any of the five districts in the "inner city", and was situated on the first floor, otherwise 0. The rationale for the latter is that we hypothesized that apartments located on the first floor in these districts would likely experience a negative impact on their value due to increased foot traffic and limited privacy.

4 Methodology

In the following section, a detailed explanation of the development process for our AVMs is presented, encompassing both the stacked generalization approach and the individual models used in the ensemble. Furthermore, this section describes the evaluation metrics and techniques used to assess model performance. Our primary goals were to optimize the performance of each individual model and to determine whether incorporating stacked generalization could improve the predictive ability of the selected models. In addition, gaining a deeper understanding of the underlying mechanisms of each model was considered crucial to our investigation.

4.1 Stacked Generalization

The idea of stacked generalization is to exploit the strengths of a series of base models by combining them with a meta-model. For the meta-model, we opted for another XGB model, which has proven to be well suited for this purpose in previous literature (Birkeland et al., 2021). This decision was based on the XGB model's capability to handle a large number of features and capture non-linear relationships between input features and the target variable. The meta-model is trained on a new dataset consisting of predictions from the individual base models that have undergone k=5 folds of cross-validation to prevent overfitting, as visualized in Figure 3. Further in this subsection, we will go through the stacking process step-by-step.

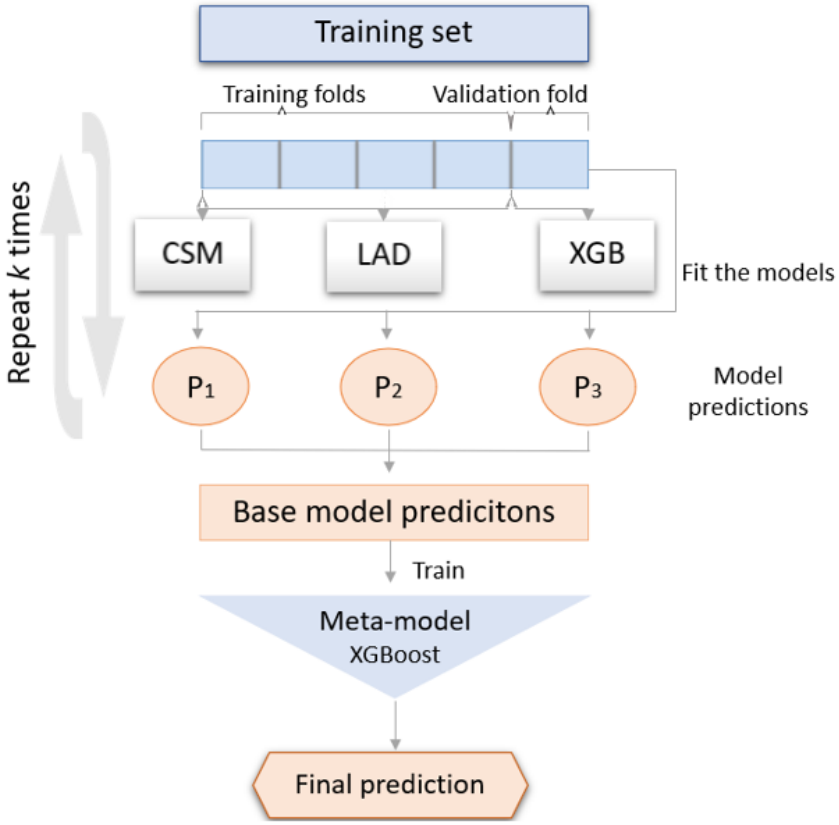


Figure 3: The process of stacked generalization

In order to prepare for the construction of predictive models, the dataset was initially divided into a training set and a test set using random allocation, with a 75-25 split[2]. This allocation ensured that the models had access to a substantial amount of data for training, while also providing a sufficiently large sample size for evaluating their performance. The training set comprised 123,489 observations, while the test set contained 41,163 observations.

The second step is to train multiple prediction models individually. CSM and LAD are models that have been widely used in the real estate industry, while XGB is a more complex machine learning model that has shown promising results in predicting property values in previous studies, outperforming other machine learning methods (Hjort et al., 2022; Kumkar et al., 2018; Stang et al., 2022). By utilizing these different models, we aim to capture a wide range of features and relationships within the data, and to assess the relative strengths and weaknesses in each of the models. To prevent overfitting in the stacked ensemble, we performed a 5-fold cross-validation for each base model by dividing the original training set into training folds and a validation fold. This allows us to train and evaluate the ensemble multiple times on different subsets of the data, ensuring that the models are not overfitting to a particular subset. With the use of the trained base models, we can generate predictions for the properties in the dataset. These predictions are then combined to create a new dataset, which is used as input features for training the meta-model. Once the meta-model is trained, it is used to make predictions on a holdout dataset (our test dataset).

In order to optimize the performance of our stacked generalization model, we employed a common approach of using grid search with cross-validation to identify the optimal hyperparameter values, which will be described further in Section 4.2 and 4.4. Through careful selection of the hyperparameters and their corresponding grids, we can achieve better model performance. Note that while we used XGBoost as our meta-model, the hyperparameters used in the stacked generalization model differ from those used in the individual XGB model to increase model diversity. The hyperparameters used for the stacked generalization model can be found in Appendix A.2.

Lastly, the stacked model, along with the individual models, is evaluated by using different metrics such as R^2, RMSE, MAPE, MdAPE, PPE(10), PPE(20), and PPE(50), which will be described further in Section 4.6. This allows for assessment of the effectiveness of the stacked model in comparison to the individual models.

In our study, all individual and stacked models were implemented using the Python programming language. We utilized the StackingRegressor function provided by scikit-learn for the implementation of the stacked model. This function facilitated the inclusion of individual models in the stacking ensemble and the selection of the meta-estimator for generating the final prediction.

[2]To verify the generalizability of our 75-25 split, we later conducted out-of-time predictions, as detailed in Appendix A.7.

4.2 Comparable Sales Method

The KNN algorithm serves as the fundamental methodology in our CSM. This approach involves identifying a set of K neighboring apartments that are most similar to the data point in question and estimating the target value by averaging the target values of these K nearest neighbors. To determine the optimal value for K, we employed grid search, a hyperparameter tuning technique, testing values ranging from 1 to 20 and utilizing mean absolute error (MAE) as the performance evaluation metric. Our experiments revealed K=10 as the optimal value, as it yielded the lowest MAE.

We selected KNN as the foundational methodology for its scalability, a marked contrast to other CSM approaches that typically involve filtering and constrain the study to a smaller dataset. By choosing KNN, we are able to utilize the entirety of the data, potentially extracting valuable information from a larger sample.

Our KNN algorithm calculates the distance between each property using the Euclidean distance formula, as defined by Equation (4), where x and y represent the data points being compared, x_i and y_i denote the feature values of the data points along each dimension, and n represents the number of dimensions. The Euclidean distance metric allows us to measure the similarity between data points in a multi-dimensional space, which is particularly useful in situations where the calculation must take into account multiple features beyond just the spatial location.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4}$$

After calculating the distances, the KNN algorithm identifies the K nearest neighbors. These nearest neighbors are then used to estimate the target value of the data point in question by averaging their target values. This approach assumes that properties with similar feature values are likely to have similar target values, making it a suitable technique for predicting house prices. A practical example of how our CSM works is demonstrated in Appendix A.3.

As mentioned in Section 3.2, a variable named "days since" was constructed to represent the time of the sale date for the properties. However, during the testing phase, it was observed that leaving this variable unscaled resulted in biased predictions, as all the K comparables happened to be sold on the same day as the target property. To address this issue, the variable was scaled within the range of 0 to 1, reducing its weighting and expanding the range of days. This led to improved predictions, though there is a possibility they were overly accurate because they were based on similar dates, which were both before and after the target property's sale date. To mitigate this concern, a custom metric incorporating both the Euclidean distance and consideration of only using prior dates was developed, as shown in Appendix A.4. However, it is worth noting that this modification significantly increased the computation time from 14 seconds to over 6 hours. As a result, implementing stacked generalization, which requires multiple iterations, became impractical due

to the prolonged computation time. Hence, the custom metric was not implemented, and we acknowledge it as a weakness of our CSM.

4.3 Least Absolute Deviation

The aim of our hedonic regression model is to estimate the relationship between the square meter price of an apartment ($sqmP_i$) and a set of independent variables (X_{ki}), such as the size, location, and other facilities of the apartment. As discussed in Section 3.2, we use the natural logarithm of the price per square meter as the dependent variable. Thus, in our model, the natural logarithm of the square meter price ($\ln(P_i)$) is assumed to be a function of the independent variables. This transformation allows for a more flexible model that can better capture the potentially non-linear relationship between the square meter price and the independent variables, improving the accuracy of the model and the estimated coefficients. The regression equation in our study can be represented by:

$$\ln(sqmP_i) = \beta_0 + \sum_{k=1}^{K} \beta_k X_{ki} + \epsilon_i, \qquad (\epsilon \sim \text{i.i.d.}) \tag{5}$$

By estimating the coefficients β_k of the independent variables, the model can identify the marginal impact of each variable on the square meter price, holding all other variables constant. The intercept term β_0 shows the value of the square meter price when all independent variables are equal to zero. The error term ϵ_i captures the random variation in the square meter price that is not explained by the independent variables, and is assumed to be independently and identically distributed (i.i.d) across observations. The aim is to estimate the coefficients β_k that both provide the best fit to the data and minimize the loss function.

In this study we use LAD instead of the more traditionally used OLS method, due to the robustness of LAD in the face of outliers within the dataset, as noted by Yoo (2001). The difference between OLS and LAD lies in the loss function where OLS minimizes the squared prediction errors to find the feature coefficients β_k, while LAD seeks to minimize the absolute value of these errors. The LAD model in our study is implemented using optimization algorithms that minimize the sum of absolute differences between the predicted and actual values between all apartments in the dataset (i=1,2,3...), and can be represented by the following equation:

$$\arg\min_{\beta_0,\beta} \sum_{i=1}^{n} |\ln(sqmP_i) - (\beta_0 + \beta X_i)| \tag{6}$$

In this equation, the objective is to minimize the sum of absolute differences between the natural logarithm of the observed values $\ln(sqmP_i)$ and the combined effect of the intercept term β_0, as well as the product of the slope parameter β with the independent variable X_i. The prediction of the dependent variable is computed by summing the intercept term and the product of the vector of coefficients, β, with

the vector of predictor values, X_i. The arg min notation indicates the search for values of β_0 and β that yield the minimum sum of absolute differences.

One inherent advantage of using hedonic regression models like LAD is that they do not require any specific hyperparameter tuning. The optimization objective of the LAD model is solely based on absolute differences, making it a simpler and less time-consuming alternative compared to more complex machine learning models such as XGB.

4.4 XGboost

XGB is an advanced machine learning algorithm known for its remarkable performance in numerous competitions (Nielsen, 2016). It operates by repeatedly incorporating decision trees into a model, where each subsequent tree aims to rectify errors made by the previous trees. During the training process, the XGB model optimizes a loss function that quantifies the disparity between predicted and actual values (Chen & Guestrin, 2016). It then uses gradient descent to update the weights of the decision trees and minimize the loss function by successively adjusting the parameter values. To address the issue of overfitting, XGB employs regularization techniques, such as L1 and L2 regularization. These techniques involve the addition of penalty terms to the objective function during training, discouraging the model from fitting the noise in the dataset and penalizing the complexity of the model. Notably, the optimized implementation of XGB allows for efficient handling of large datasets, making it a compelling option for machine learning tasks that entail substantial volumes of data. The objective function utilized in our study to train the XGB model is defined as:

$$Obj^{(t)} = \sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)) + \sum_{i=1}^{t} \Omega(f_i) \tag{7}$$

In this equation, the loss function denoted by $\mathcal{L}$ measures the difference between the predicted prices and the actual prices, with the goal of minimizing the difference. y_i is the true price, while $\hat{y_i}^{(t-1)}$ is the predicted price at the previous iteration. The term $f_t(x_i)$ represents the contribution of the new decision tree added to the ensemble at iteration t, which aims to rectify errors made by the previous trees and enhance the overall model performance. $\Omega(f_i)$ represents the regularization techniques applied to the tree to prevent overfitting by penalizing complex trees. The MAE was chosen as the loss term for the objective function in our case. For further technical details, we refer to Chen and Guestrin (2016).

Optimal tuning of hyperparameters is essential in complementing the objective function to enhance the performance of the XGB models. However, finding the right balance between model complexity and accuracy is crucial, as overly complex models can be computationally expensive. To identify the optimal hyperparameter values, we employed grid search, the same brute-force approach as we used to find the optimal number of K in Section 4.2. However, this time there were a larger number of parameters to search over. The algorithm trains and evaluates the model using

each combination of hyperparameter values within the grid, ultimately returning the best values for each parameter. In our study, we performed grid search in combination with 5-fold cross-validation, aiming to minimize the MAE as our performance metric. Table 3 summarizes the tuned values and the predefined grids, with a brief explanation of each parameter.

Table 3: Hyperparameter tuning for the XGBoost model

Hyperparameter	Hyperparameter grid	Tuned value	Description
max_depth	[3, 5, 7, 9]	7	Maximum depth of each tree
min_child_weight	[1, 3, 5, 7]	1	Minimum sum of instance weight in a child node.
gamma	[0.01, 0.05, 0.1, 0.2, 0.3]	0.01	Minimum loss reduction for splitting a node.
subsample	[0.6, 0.7, 0.8, 0.9, 1]	0.8	Fraction of samples used for training each tree.
colsample_bytree	[0.6, 0.7, 0.8, 0.9, 1]	0.6	Fraction of features used for training each tree.
learning_rate	[0.005, 0.01, 0.05, 0.1]	0.1	Step size for model weight updates.
n_estimators	[100, 200, 300, 400, 500]	500	Number of trees.
reg_alpha	[0.1, 0.5, 1, 2.5]	1	L1 regularization.
reg_lambda	[0.1, 0.5, 1, 2.5]	0.5	L2 regularization.

Note: The rest of the available parameters remained at their default values. For more information about these parameters and their default values, we recommend referring to DMLC (2022), a thorough guide provided by the developers of the xgboost package.

Despite the remarkable performance of the XGB model in previous studies, a critical consideration when optimizing XGB models is the trade-off between accuracy and explainability. Comprehending how the model arrives at its predictions can be challenging due to its complex decision-making process. Model interpretation techniques, such as feature importance analysis (to be discussed in Section 4.5), can provide insights into the complex interactions among the ensemble of decision trees in the model, thereby enhancing its overall interpretability.

4.5 SHAP

In our pursuit of understanding the behavior of each individual model within a stacked ensemble, a closer examination of the various features, both as standalone models and in the ensemble, was necessary. SHapley Additive exPlanations (SHAP) was first introduced by Shapley (1953), as an application to coalitional game theory, and serves our purpose. More recently, Štrumbelj and Kononenko (2010) explained how this concept could be implemented on machine learning algorithms. This has led to SHAP becoming a useful framework to interpret the feature importances of more complex models.

Mathematically, the general formula for SHAP values is given by Equation (8). In this equation, $\phi_i(x)$ represents the SHAP value for the i-th feature, which provides an attribution of the prediction for an instance to that particular feature. S is the subset of all features N excluding the i-th feature, while $|S|$ denotes the number of elements (cardinality) of subset S. The weight assigned to each subset S is given by the term $|S|!(|N| - |S| - 1)!$ divided by $|N|!$, and the term $f(x_S \cup \{x_i\}) - f(x_S)$ represents the difference in the model's output when the i-th feature is included or excluded.

$$\phi_i(x) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f(x_S \cup \{x_i\}) - f(x_S)] \tag{8}$$

The advantage of using SHAP as an explanation of features is that it can be used on all types of models (Lundberg & Lee, 2017), which makes it a practical choice for scenarios involving multiple models. However, it also has limitations, particularly in terms of computation time for non-tree-based models. For instance, tree-based models such as XGB offer more efficient ways of computing feature importance in Python, such as Tree SHAP, which provides faster SHAP values compared to the regular Kernel SHAP (Lundberg et al., 2019). Similarly, for linear models like our LAD model, computations are generally faster due to the availability of Linear SHAP, which handles linear models effectively. However, for models like our CSM that employ a KNN algorithm, neither linear nor tree-based methods are applicable, creating computational challenges. As a result, Kernel SHAP, which is computationally intensive, must be used, leading to prolonged computation times for exact SHAP values. This issue also impacts the stacked models where the CSM is included. To address these computational challenges, a common approach is to utilize sampling techniques to reduce the size of the dataset. For instance, Yuan et al. (2023) conducted multiple simulations using random extractions from a training set and investigated the effect of different background sample sizes (ranging from 100 to 1,000 observations) on SHAP computations. They found that SHAP explanations tend to fluctuate when smaller sample sizes are used, suggesting that larger background datasets should be preferred. However, considering the tradeoff between computation time and accuracy, we decided to use a randomly drawn background sample size of 300 observations and a randomly drawn test sample of 100 observations to get the feature importances in this study. We conducted simulations with the entire dataset for both LAD and XGB models and found no differences in the rankings of variables. Hence, we assumed that our reduced random sample was a good representation of the entire dataset, and this reduced version was used for comparison across all models.

4.6 Evaluation Metrics

To measure the accuracy of models and compare their performance, it is important to select appropriate evaluation metrics to gain valuable insights from the different model outputs. In this subsection, we will discuss the metrics we used in detail and their relevance in evaluating our models.

R-squared (R^2) is a statistical measure used to evaluate the performance of regression models. It represents the proportion of the variance in the dependent variable that is explained by the independent variables in the model. R-squared values range from 0 to 1, with 0 indicating no relationship between the variables and 1 indicating a perfect fit. A higher R-squared value means that the model more accurately captures the variability of the dependent variable. The advantage of including R-squared is that it is well-known and easy to interpret. However, it may not accurately capture the relationship between the dependent and independent

variables in nonlinear models (Kvalseth, 1985). Therefore, it is important to use R-squared in conjunction with other metrics to gain a more complete understanding of the model's performance.

Root Mean Squared Error (RMSE) is a metric used to measure the difference between the actual and predicted values in a regression model. To calculate RMSE, the differences between each predicted and actual value are squared, and then the mean is calculated to get the Mean Squared Error (MSE). Then, the square root of the MSE is taken to obtain the RMSE value. In our case, the dependent variable is log-transformed, meaning that the RMSE values will be in the log scale, implying that our RMSE calculations are based on relative errors. While this can make the results more challenging to interpret, it is usually an acceptable trade-off for better inference (Hodson, 2022). Lower RMSE values indicate better model performance in terms of predictive accuracy.

Mean Absolute Percentage Error (MAPE) and Median Absolute Percentage Error (MdAPE) are metrics used to assess the accuracy of predictions in a regression model by quantifying the percentage difference between actual and predicted values. While MAPE is more commonly employed, MdAPE also holds value as it relies on median values, which are less sensitive to outliers than mean values (Levenbach, 2015). This means that MdAPE provides more robust estimates. In both cases, a lower percentage value indicates increased prediction accuracy.

Percentage Prediction Error (PPE) evaluates the accuracy of predictions in a regression model based on specific percentage thresholds. In our study, we included PPE(10), PPE(20), and PPE(50), which measure the proportion of predictions that fall within 10%, 20%, and 50% of the actual values, respectively. For each threshold, the metric checks whether the prediction error for each data point is within the specified percentage range. Then, it calculates the proportion of data points that meet this criterion. Since our actual and predicted values were log-transformed, we reverted them to their original scale before performing the calculations to obtain a more accurate representation of PPE values. A higher PPE value indicates better model performance, as a higher percentage of predicted values fall within a certain range of the actual values.

5 Results and Discussion

In this section, we aim to provide a comprehensive analysis and discussion on the performance of our AVMs. Starting in Section 5.1, we evaluate the models' effectiveness by comparing the individual and stacked models. Next, we delve into the feature importance of each model in Section 5.2, identifying the key variables utilized in their predictions. In Section 5.3, we then assess model performance at a district level, investigating for any observable patterns. Finally in Section 5.4, we conduct a series of tests designed to evaluate how our models adapt and perform under different data sizes.

5.1 Model Performance

Table 4 shows the performance of the individual and stacked models. The XGB model performs best on all performance metrics compared to the individual models. The CSM ranks second, while the LAD model expectedly falls short in comparison. When introducing stacked generalization, we can see that the models gets marginally better than the individual XGB model. Stacking only XGB and LAD does not seem to improve much, with nearly identical results on most of the metrics, and it actually performs worse on PPE(50). The combination with XGB and CSM seems to be the best model overall, outperforming all other models on almost all metrics. The inclusion of LAD in addition to XGB and CSM performs slightly better on PPE(20), but slightly worse elsewhere. This indicates that the LAD model fails to bring any new valuable information that is not already covered by the CSM.

Table 4: Comparison of model performance

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8576	0.1372	10.35%	7.86%	60.18%	87.35%	99.29%
LAD	0.8377	0.1465	11.16%	9.01%	54.68%	84.79%	99.69%
XGB	0.9342	0.0933	7.01%	5.47%	76.30%	96.13%	**99.91%**
(XGB + CSM)	**0.9367**	**0.0915**	**6.87%**	**5.35%**	**77.23%**	96.30%	**99.91%**
(XGB + LAD)	0.9346	0.0930	6.99%	5.47%	76.61%	96.14%	99.90%
(XGB + CSM + LAD)	0.9361	0.0919	6.90%	5.38%	77.03%	**96.33%**	**99.91%**

To better represent the prediction errors, histograms were generated for each model. Figure 4 shows the relative error rate in percentage of the square meter price on the x-axis and the probability density on the y-axis. The dashed lines mark the +- 20% range. The results indicate that XGB outperforms CSM and LAD, exhibiting a more narrow error distribution and a higher proportion of errors within the dashed lines, implying a superior model. The three stacked models appear identical, exhibiting a distribution pattern similar to that of XGB alone, suggesting that the stacked models are predominantly reliant on the XGB algorithm.

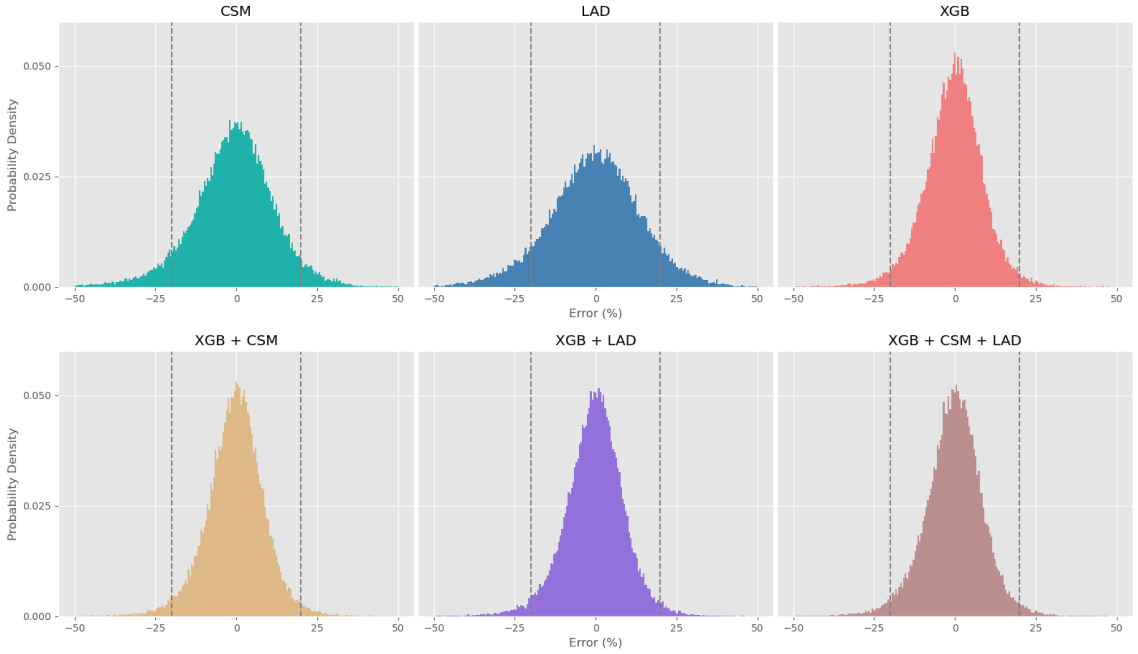


Figure 4: Histograms of relative errors in percent from each model

In summary, we can observe that the XGB model performs exceptionally well on its own, raising the question of whether stacked generalization is worth the added complexity and resources required for training and prediction. Although the number of predictions falling within 50% of actual price does not improve with stacking, we do see improvements within 20% and especially within 10%. Stacking XGB and CSM improves by nearly 1%-point compared to the individual XGB model within the 10% range. This improvement translates to correctly predicting 383 more apartments within 10%, which may justify using stacked generalization.

Furthermore, while these findings shed light on the performance of the models, it is essential to acknowledge potential limitations. One such limitation is the absence of out-of-time predictions in our models, which hinders their evaluation in real-world scenarios. We acknowledged this concern for the CSM in Section 4.2, offering a possible solution for that specific model in Appendix A.4. However, to address this limitation comprehensively across all our models, a simpler and more effective approach involves modifying the division of training and test sets. Thus, we conducted a simulation to account for the temporal perspective and assess its impact on our conclusions.

The results of this simulation demonstrated that our initial methodology could be generalized, as the division of the dataset did not result in substantial differences in the relative performance of the models. For a more comprehensive analysis of this simulation, please refer to Appendix A.7.

5.2 Feature Importance

In order to enhance our comprehension of each model, we have examined the outputs produced by the SHAP method, as described in Section 4.5. Figure 5 portrays the SHAP values derived from all models, offering a comprehensive overview of each variable across the models. Beeswarm plots from each model can be found in Appendix A.8. As the models utilize distinct features to represent time, location, and size, we have aggregated the SHAP values to facilitate comparison. Specifically, the *time* feature incorporates all categorical variables pertaining to year and month for the LAD model, whereas CSM and XGB employ the "days since" variable. The *location* feature encompasses all categorical district variables in the LAD model, while XGB employs latitude and longitude, and CSM employs latitude and longitude represented in radians. *Size* is represented by the "living area" variable for XGB, whereas LAD and CSM include categorical size variables, as detailed in Section 3.2. All other variables are common to all models.

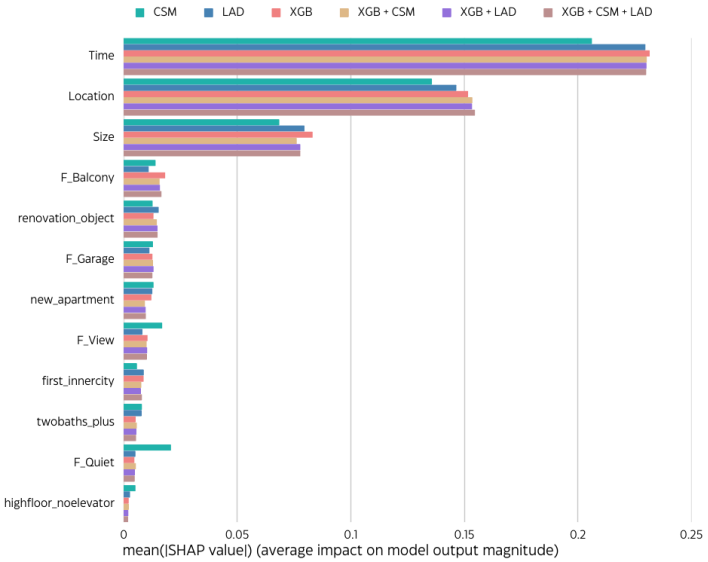


Figure 5: Mean absolute SHAP values from all models

The analysis reveals that the *time* feature is the most influential variable across all models, followed by *location* and *size*. While the XGB model exhibits greater sensitivity to time and size, the stacked models are more responsive to location, with size playing a less significant role in comparison to the individual XGB. Of note, the CSM deviates substantially from the others in terms of variable weighting. While it ranks the top three variables equally with the other models, it assigns lower importance to these variables and instead prioritizes variables that other models do not consider as important, particularly the facility variables, with "quiet" ranking fourth in CSM, but eleventh in other models. LAD, on the other hand, assigns higher importance to "renovation object" and rank it as fourth, while XGB and the stacked models rank "balcony" fourth. The variable "high floor and no elevator" is ranked the least important across all models, even though CSM allocates a little more weight to this variable than the rest. Despite this, the variable is still significant for the LAD model, as evidenced in Table 6, and is also included in the final decision tree for the XGB model (Figure 10b).

5.3 Model Performance at District Level

Our investigation also involved the identification of any potential patterns in the districts. Figure 6 shows the average absolute error rate in each district, with a darker shade indicating a smaller prediction error. Our findings suggest that the predictive models perform better in the eastern region of Oslo compared to the western region. One possible explanation for this result is that apartments in the eastern region are more homogeneous, mostly consisting of apartment complexes with similar sizes and prices, while the western region possesses more variation due to the generally higher income levels of the local population as discussed in Section 3.2. Additionally, our results reveal that micro-location is a critical factor when predicting prices, which may explain why the LAD model performed poorly. Notably, the prediction errors observed in the XGB model and the three stacked models are very similar, further substantiating the effectiveness of the XGB model.

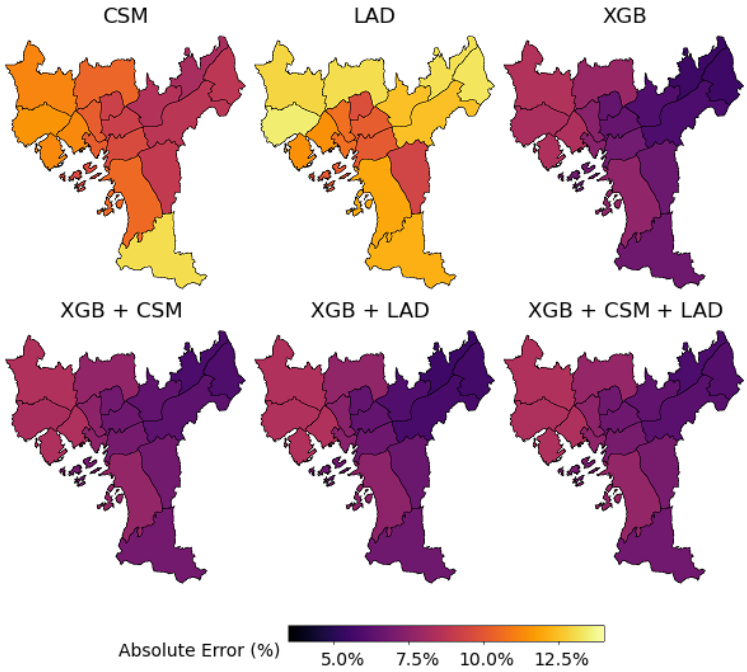


Figure 6: Absolute error per district from each model

5.4 Model Performance under Different Data Sizes

To gain a deeper understanding of how the different valuation techniques as well as the stacking perform under different data sizes, we conducted additional tests to simulate such scenarios. Our objective was to evaluate the models' ability to generalize well to new data with fewer observations and determine their adaptability to limited or scarce data. A model's adaptability to smaller datasets is a reflection of its robustness and flexibility, which are essential qualities given that real-world scenarios often require dealing with data constraints. The nature of the test is essentially a stress test for the models, evaluating their capabilities to adapt to less than ideal circumstances.

Previous research, such as the study conducted by Stang et al. (2022), has shown that traditional methods may outperform the XGB model in areas with limited observations, suggesting that the XGB model may be more sensitive to having less information compared to the CSM and the LAD model. Thus, we aimed to investigate how our models would perform out-of-sample on a variety of smaller training sets, which represent a wide range of potential scenarios. To achieve this, we systematically reduced the original training set of 123,489 observations to varying sizes, all the way down to 123 (0.1%) observations. To minimize the influence of randomness, we maintained a constant test set size of 41,163 observations throughout the experiment. Figure 7 illustrates how the MdAPE differentiates across the various sizes of training data.

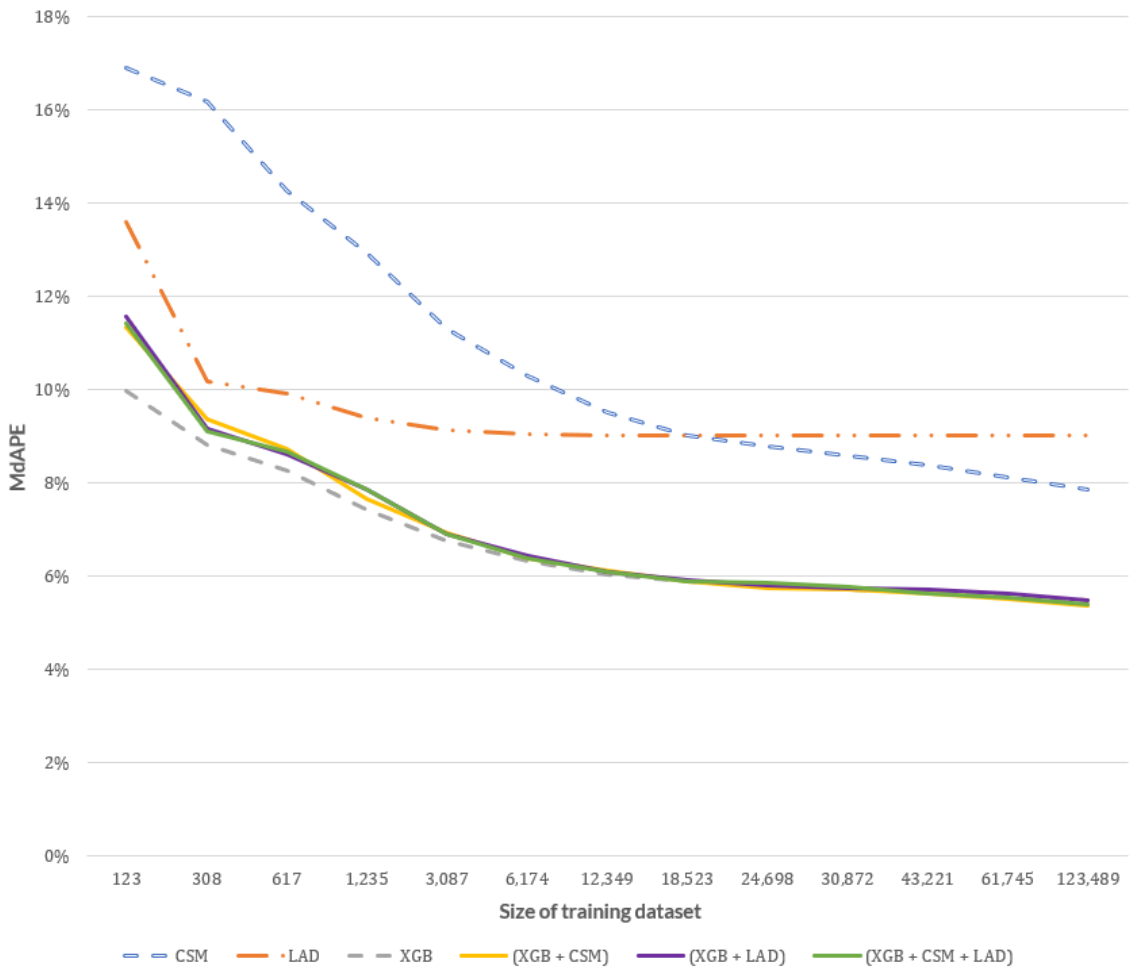


Figure 7: MdAPE for training data of varying sizes, ranging from 123 to 123,489 observations

The graph illustrates that even when the training set is halved (61,745 observations), the stacked model combining XGB and CSM remains the best-performing model in terms of MdAPE. However, it is evident at this point that the stacked model with XGB and LAD now perform worse than the individual XGB. This further supports the notion that LAD struggles to provide any significant valuable information in stacked generalization.

Moving forward, it can be observed that all three stacked models closely mimic the performance of the individual XGB, with the combination of XGB and CSM consistently outperforming the others by a slight margin until the training set is reduced to 10% (12,349 observations). At this point, a shift occurs, and XGB emerges as the best-performing model, maintaining its superiority as the data is further reduced. These results indicate that stacked generalization fails to improve prediction performance for datasets of this size and below.

LAD demonstrates strong stability compared to the other models and outperforms CSM in terms of MdAPE when the training data is reduced to 10% and beyond. However, there is no clear evidence that stacking with this model is beneficial for very small datasets. It can be observed that while the other models gradually exhibit higher MdAPE as the dataset size decreases, LAD manages to maintain a consistent performance around 9% for an extended period, indicating a potential to outperform XGB in the long run. However, when the training data is reduced to a mere 0.1% (123 observations), LAD experiences a sharp increase in MdAPE, whereas XGB continues to perform well with less impact. This indicates that XGB remains the best individual model regardless of the data size.

A possible reason for XGB's sustained precision could be its ability to leverage a greater amount of information about each individual property compared to LAD, specifically in terms of location and time of sale. On the other hand, CSM, while also able to incorporate more information about location and time of sale than LAD, becomes vulnerable when the dataset size is too small. This may be due to its reliance on having a sufficient number of similar properties to the one being evaluated.

We observed trends in the other evaluation metrics similar to those observed with MdAPE. The stacked model that combines XGB and CSM demonstrated superior performance across all metrics until the training data was reduced to 10%, emphasizing the strength of this model. However, there were a few exceptions worth noting. At the 25% data size mark (30,872 observations), XGB slightly outperformed the others on PPE(10), while the combination of all three models showed better performance on PPE(20). When the training data was reduced to 10%, the point where XGB first began outperforming the stacked models in terms of MdAPE, XGB also demonstrated superior performance on MAPE and PPE(10). Despite this, the other metrics continued to favor stacked generalization. When the training data was further reduced to 5% (6,174 observations), and below, XGB dominated the results. For comprehensive details on these results, please refer to Appendix A.9.

These findings confirm the results presented in Table 4, indicating that they hold true until the training data is reduced to approximately 12,000 observations. Beyond this point, the performance of the CSM is significantly penalized, resulting in stacked generalization with XGB and CSM being ineffective for improving model performance. The effectiveness of stacked generalization with XGB and LAD was limited even with large datasets, and although the LAD model demonstrated stability across different dataset sizes, it failed to provide any meaningful insights in smaller datasets.

6 Conclusion and Further Work

This study develops and compares six distinct Automated Valuation Models (AVMs), of which three are individual models and the remaining three are different combinations of the initial models using stacked generalization. The objective of the study was to determine whether employing stacked generalization could enhance prediction accuracy when the ensemble incorporated methodologically diverse models. Furthermore, we analyzed how each model within the stacked ensemble functioned to identify the most suitable conditions for their use.

Consistent with previous literature, it has been confirmed that XGBoost (XGB) stands out as the most effective individual model when tested on an extensive dataset from the housing market in Oslo. This is evidenced by metrics such as the median absolute percentage error (MdAPE), as low as 5.47%, and 96.13% of predictions falling within 20% of the actual price. Remarkably, when XGB is combined with the Comparable Sales Method (CSM) using stacked generalization, prediction performance across all evaluation metrics improves, including MdAPE decreasing to 5.35% and the proportion of predictions within 20% of the actual price rising to 96.30%. The inclusion of Least Absolute Deviation (LAD) in addition to the others in the stack further increases this proportion to 96.33%. However, this addition somewhat negatively impacts performance on other metrics.

Additional tests, designed to simulate model performance under varying data sizes, revealed that the stacked models' effectiveness declined as the dataset size decreased. The stacked model, composed of XGB and CSM, outperformed all other models until the training data was reduced to approximately 12,000 observations. This turning point marked the individual XGB model's rise to the top, suggesting that stacked generalization might not improve prediction performance for smaller datasets. Despite the LAD model demonstrating robustness with decreasing dataset size, its incorporation within a stacked model did not yield any substantial improvement in prediction performance.

During the development of our models, two aspects that arose as significant concerns were the computational complexity of the stacked models and their level of interpretability. Our findings indicate that the stacked models derive most of their accuracy from the XGB base model, suggesting a strong reliance on this particular algorithm. Nonetheless, we were unable to precisely determine the individual contributions of each model within the stacked ensemble, which would have provided valuable insights. In addition, the inclusion of CSM in the stacked models made it challenging to obtain reliable results regarding the feature importance. Due to computational complexity, we had to analyze a smaller version of the dataset regarding this context. Consequently, our analysis of feature importance might not fully capture the true underlying patterns and relationships within the data.

In conclusion, our study has shown that using stacked generalization to combine distinct modeling techniques can improve predictive performance, but these improvements are relatively small and require a certain amount of data. The limited enhancements observed with stacked models, along with their time-consuming implementation, raise questions about the practicality of using stacked generalization

in real-world applications. Given the superior performance of the XGB as a standalone model, it may be more efficient and advantageous to focus on incorporating relevant data and refining the features used within the XGB model. This approach could yield more substantial gains in predictive accuracy while reducing the time and resources needed for execution. Nonetheless, further investigation is required to substantiate these results and delve into alternative strategies for optimizing predictive models, such as the inclusion of demographic and economic factors.

In light of the unsatisfactory results with stacked generalization on smaller datasets present in this study, it might be beneficial to consider using a different meta-model instead of XGB. During our research, we employed a grid search to optimize the hyperparameters for both the individual XGB and the XGB meta-model. However, the output for the meta-model revealed a less complex model, as shown in Table 3 and Table 5. This finding suggests that a simpler model, such as linear regression, might be a more appropriate choice for this purpose. Consequently, the use of a simpler meta-model could potentially make stacked generalization more effective, especially when dealing with smaller datasets.

It may also be worth investigating whether the creation of district-specific AVMs can lead to improved prediction performance. To use the dataset in our study as an example, the dataset could be divided into 15 separate parts, each representing a different district, and a unique AVM could be trained for each district. This would allow for the exploration of whether micro-locality plays a significant role in predicting housing prices, and whether certain districts benefit more from their own individual models due to greater homogeneity in the housing or other unique characteristics. Additionally, this approach would allow for an examination of whether different models perform better in certain districts compared to others, and whether incorporating these district-specific models into AVMs can lead to improved overall performance and predictive power. Overall, the exploration of district-specific AVMs is an exciting opportunity to deepen the understanding of factors that influence housing prices and to improve the accuracy of AVM predictions.

Another promising area for future research is to delve deeper into the CSM or other largely unexplored models within a stacked generalization framework. Our implementation of the CSM in this context showed promising results when tested out-of-sample on datasets above a certain size, highlighting the potential of using this model as one of the base models. However, current technological limitations pose challenges to the interpretability of this model within a stacked generalization framework due to long computation times. As technology evolves, the CSM could become an increasingly suitable candidate for inclusion. Similarly, while Support Vector Machines (SVM) have seen some use in the context of stacked generalization, there is still a potential for broader exploration and application.

Addressing computational and interpretability challenges within this framework will open the door to a wider range of models that can be incorporated into stacked generalization, including deep learning models that could potentially lead to more powerful and robust AVMs. These future developments could provide valuable insights and contribute to our collective understanding of the framework, bringing advancements in predictive performance in real estate valuation and other fields.

Bibliography

Antipov, E. A., & Pokryshevskaya, E. B. (2012). 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*(2), 1772–1778. https://doi.org/10.1016/j.eswa.2011.08.077

Birkeland, K., D'silva, A., Füss, R., & Oust, A. (2021). The Predictability of House Prices: "Human Against Machine". *24*, 139–183. https://www.researchgate.net/publication/353878679_The_Predictability_of_House_Prices_Human_Against_Machine

Breiman, L. (1996). Stacked Regressions. *Machine Learning*, *24*(1), 49–64. https://doi.org/10.1007/BF00117832

Bydelsfakta. (2023a). *Boligpriser: Boligpris for blokkleiligheter*. Retrieved 24th March 2023, from https://bydelsfakta.oslo.kommune.no/bydel/alle/boligpriser

Bydelsfakta. (2023b). *Bygningstyper: Boliger etter bygningstype*. Retrieved 21st March 2023, from https://bydelsfakta.oslo.kommune.no/bydel/alle/bygningstyper

Bydelsfakta. (2023c). *Husholdninger: Husholdningstyper*. Retrieved 24th March 2023, from https://bydelsfakta.oslo.kommune.no/bydel/alle/husholdninger

Bydelsfakta. (2023d). *Levekårsindikatorer: Levekår*. Retrieved 24th March 2023, from https://bydelsfakta.oslo.kommune.no/bydel/alle/levekaar

Chen, T., & Guestrin, C. (2016). Xgboost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

DMLC. (2022). *XGBoost Parameters*. Retrieved 16th April 2023, from https://xgboost.readthedocs.io/en/stable/parameter.html

Doumpos, M., Papastamos, D., Andritsos, D., & Zopounidis, C. (2021). Developing automated valuation models for estimating property values: A comparison of global and locally weighted approaches. *Annals of Operations Research*, *306*. https://doi.org/10.1007/s10479-020-03556-1

Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, 23–37. https://doi.org/10.1007/3-540-59119-2_166

Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1189–1232. http://www.jstor.org/stable/2699986

Graczyk, M., Lasota, T., Trawiński, B., & Trawiński, K. (2010). Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. *Intelligent Information and Database Systems.*, *5991*, 340–350. https://doi.org/10.1007/978-3-642-12101-2_35

Grether, D., & Mieszkowski, P. (1974). Determinants of real estate values. *Journal of Urban Economics*, *1*(2), 127–145. https://doi.org/https://doi.org/10.1016/0094-1190(74)90013-8

Helgaker, E., Oust, A., & Pollestad, A. J. (2022). Adverse selection in iBuyer business models—don't buy lemons! *Zeitschrift für Immobilienökonomie*. https://doi.org/10.1365/s41056-022-00065-z

Hjort, A., Pensar, J., Scheel, I., & Sommervoll, D. E. (2022). House price prediction with gradient boosted trees under different loss functions. *Journal of Property Research*, *39*(4), 338–364. https://doi.org/10.1080/09599916.2022.2070525

Hodson, T. O. (2022). Root-mean-square error (RMSE) or mean absolute error (MAE): When to use them or not. *Geoscientific Model Development*, *15*(14), 5481–5487. https://doi.org/10.5194/gmd-15-5481-2022

Kansara, D., Singh, R., Sanghvi, D., & Kanani, P. (2018). Improving Accuracy of Real Estate Valuation Using Stacked Regression, 2321–9939. https://www.researchgate.net/publication/341271380

Kilpatrick, J. (2011). Expert systems and mass appraisal, 529–550. https://doi.org/10.1108/14635781111150385

Koenker, R., & Basset, G. (1978). Regression Quantiles. *Econometrica*, *46*(1), 33–50. https://doi.org/10.2307/1913643

Kumkar, P., Madan, I., Kale, A., Khanvilkar, O., & Khan, A. (2018). Comparison of Ensemble Methods for Real Estate Appraisal. *2018 3rd International Conference on Inventive Computation Technologies (ICICT)*, 297–300. https://doi.org/10.1109/ICICT43934.2018.9034449

Kvalseth, T. O. (1985). Cautionary Note about R^2. *The American Statistician*, *39*(4), 279–285. https://doi.org/10.2307/2683704

Larraz, B., Alfaro-Navarro, J.-L., Cano, E. L., Alfaro-Cortes, E., Garcia, N., & Gámez, M. (2021). A computer-assisted expert algorithm for real estate valuation in Spanish cities. *Environment and Planning B: Urban Analytics and City Science*, *48*(6), 1712–1727. https://doi.org/10.1177/2399808320947729

Law, S., Paige, B., & Russell, C. (2019). Take a Look Around: Using Street View and Satellite Images to Estimate House Prices. *ACM Trans. Intell. Syst. Technol.*, *10*(5). https://doi.org/10.1145/3342240

Levenbach, H. (2015). The Myth of the MAPE... and How to Avoid It. *CPDF: Cerfified Professional Demand Forecaster*.

Lundberg, S. M., Erion, G. G., & Lee, S.-I. (2019). Consistent Individualized Feature Attribution for Tree Ensembles. https://doi.org/10.48550/arXiv.1802.03888

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, *30*.

Mooya, M. M. (2017). Automated Valuation Models and Economic Theory. In M. d'Amato & T. Kauko (Eds.), *Advances in Automated Valuation Modeling: AVM After the Non-Agency Mortgage Crisis* (pp. 33–57). Springer International Publishing. https://doi.org/10.1007/978-3-319-49746-4_3

Nielsen, D. (2016). Tree boosting with XGBoost. *Norwegian University of Science and Technology*.

Nnadozie, L., Matthias, D., & Bennett, E. (2022). A model for Real Estate Price Prediction using MultiLevel Stacking Ensemble Technique. *European Journal of Computer Science and Information Technology*, *10*(3), 33–45.

Oslo kommune. (2023a). *Boligpriser*. Retrieved 31st March 2023, from https://www.oslo.kommune.no/statistikk/boligpriser-boforhold-og-byggevirksomhet/boligpriser/

Oslo kommune. (2023b). *Geografiske inndelinger*. Retrieved 24th March 2023, from https://www.oslo.kommune.no/statistikk/geografiske-inndelinger/

Oust, A., Hansen, S. N., & Pettrem, T. R. (2020). Combining Property Price Predictions from Repeat Sales and Spatially Enhanced Hedonic Regressions.

The Journal of Real Estate Finance and Economics, *61*(2), 183–207. https://doi.org/10.1007/s11146-019-09723-x

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

Rosen, S. (1974). Hedonic Prices and Implicit Markets: Product Differentiation in Pure Competition. *Journal of Political Economy*, *82*(1), 34–55. http://www.jstor.org/stable/1830899

Sangani, D., Erickson, K., & Hasan, M. A. (2017). Predicting Zillow Estimation Error Using Linear Regression and Gradient Boosting. *2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 530–534. https://doi.org/10.1109/MASS.2017.88

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227. https://doi.org/10.1007/BF00116037

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games*, 307–317.

Smyth, P., & Wolpert, D. (1997). Stacked Density Estimation. https://www.researchgate.net/publication/2718463_Stacked_Density_Estimation

Solgt.no. (2023). *FAQ*. Retrieved 21st March 2023, from https://solgt.no/faq

Stang, M., Krämer, B., Nagl, C., & Schäfers, W. (2022). From human business to machine learning—methods for automating real estate appraisals and their practical implications. *Zeitschrift für Immobilienökonomie*. https://doi.org/10.1365/s41056-022-00063-1

Štrumbelj, E., & Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, *11*, 1–18.

Trawiński, B., Telec, Z., Krasnoborski, J., Piwowarczyk, M., Talaga, M., Lasota, T., & Sawiłow, E. (2017). Comparison of expert algorithms with machine learning models for real estate appraisal. *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, 51–54. https://doi.org/10.1109/INISTA.2017.8001131

Tretton, D. (2007). Where is the world of property valuation for taxation purposes going? *Journal of Property Investment & Finance*, *25*(5), 482–514. https://doi.org/10.1108/14635780710776684

Truong, Q., Nguyen, M., Dang, H., & Mei, B. (2020). Housing Price Prediction via Improved Machine Learning Techniques. *Procedia Computer Science*, *174*, 433–442. https://doi.org/10.1016/j.procs.2020.06.111

Valier, A. (2020). The Cross Validation in Automated Valuation Models: A Proposal for Use. *Computational Science and Its Applications – ICCSA 2020*, 585–596. https://doi.org/10.1007/978-3-030-58814-4_45

Wang, J., Sui, J., Zhang, Z., Qi, J., Liu, N., & Lv, J. (2020). Empirical analysis of I-GBDT to improve the accuracy of mass appraisal method. *Journal of Physics: Conference Series*, *1550*(3), 032074. https://doi.org/10.1088/1742-6596/1550/3/032074

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*(2), 241–259. https://doi.org/10.1016/S0893-6080(05)80023-1

Yoo, S.-H. (2001). A robust estimation of hedonic price models: Least absolute deviations estimation. *Applied Economics Letters*, *8*(1), 55–58. https://doi.org/10.1080/135048501750041303

Yuan, H., Liu, M., Kang, L., Miao, C., & Wu, Y. (2023). An empirical study of the effect of background data size on the stability of SHapley Additive exPlanations (SHAP) for deep learning models. https://doi.org/10.48550/arXiv.2204.11351

Appendix

A.1 Additional Data Overview

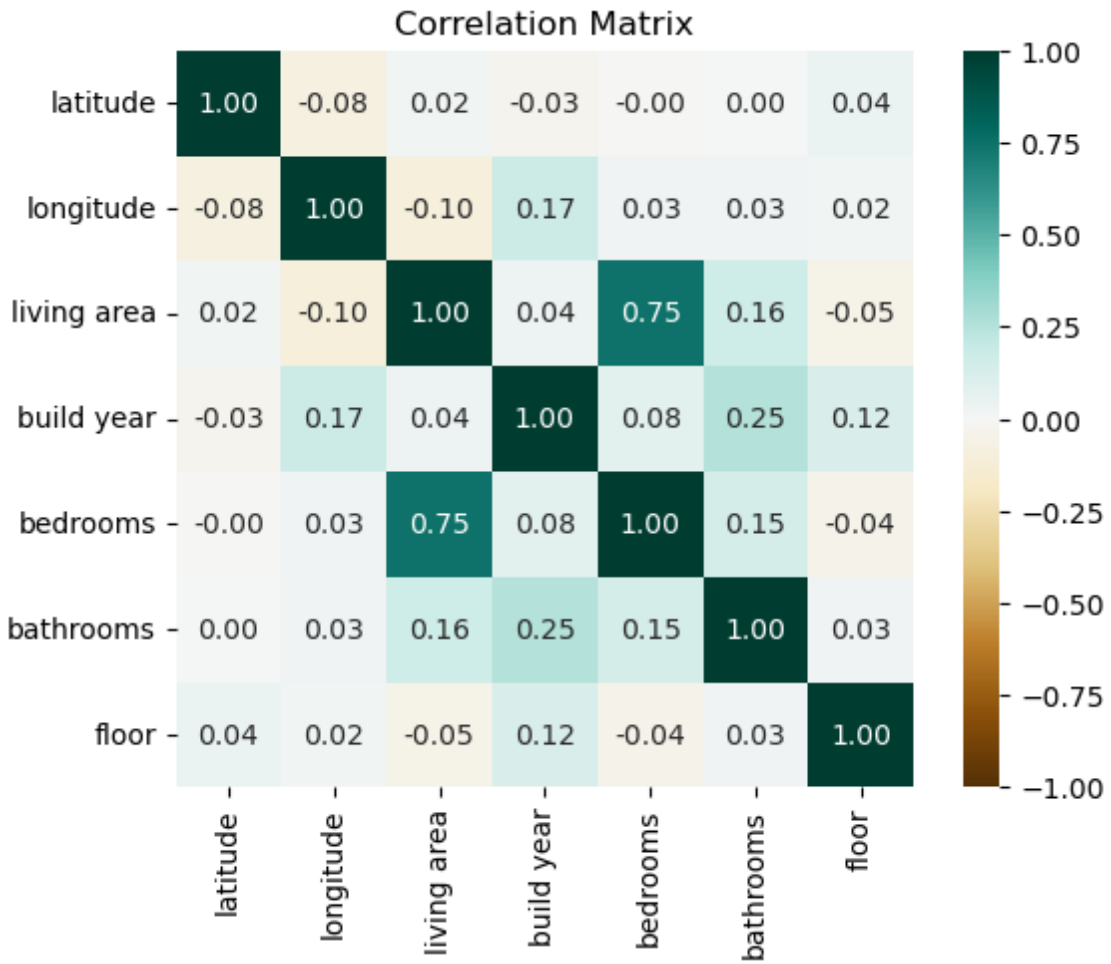


Figure 8: Correlation matrix for the continuous variables in the original dataset. The variable "bedrooms" was not included in the final dataset due to its correlation with "living area". While the continuous variables "build year", "bathrooms", and "floor" were also excluded from the final dataset, they were utilized to generate binary variables.

A.2 Hyperparameters for the Meta-Model

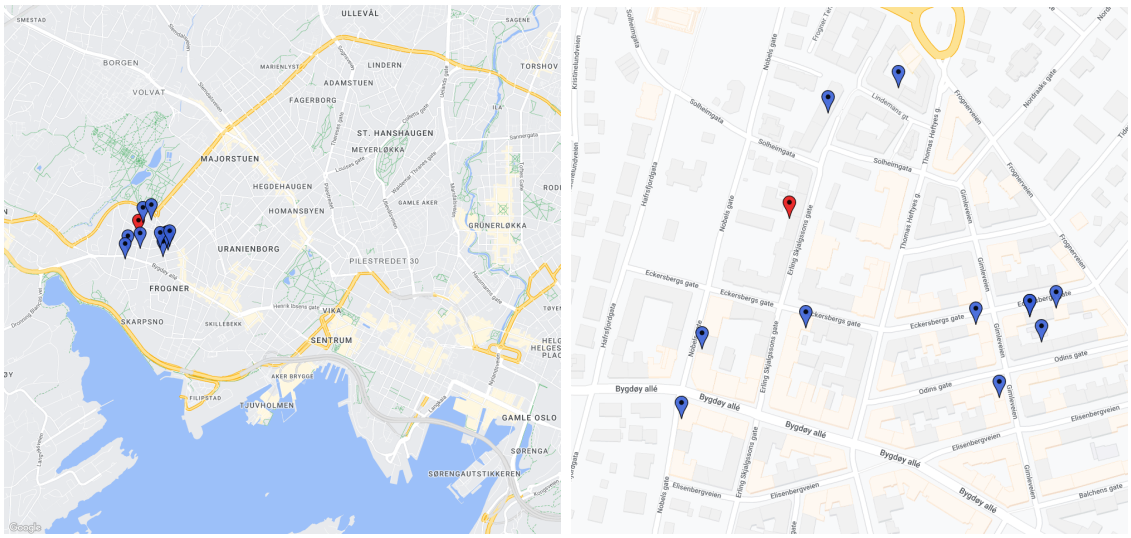
The same set of values was used in the grid search for the meta-model as for the individual XGB model, as described in Section 4.4. However, the resulting tuned values were different, leading to a less complex meta-model compared to the individual XGB model. This is likely because a stacked model tends to perform better when there is more diversity (Breiman, 1996).

Table 5: Hyperparameter tuning for the meta-model

Hyperparameter	Hyperparameter grid	Tuned value
max_depth	[3, 5, 7, 9]	5
min_child_weight	[1, 3, 5, 7]	3
gamma	[0.01, 0.05, 0.1, 0.2, 0.3]	0.1
subsample	[0.6, 0.7, 0.8, 0.9, 1]	0.7
colsample_bytree	[0.6, 0.7, 0.8, 0.9, 1]	1
learning_rate	[0.005, 0.01, 0.05, 0.1]	0.1
n_estimators	[100, 200, 300, 400, 500]	100
reg_alpha	[0.1, 0.5, 1, 2.5]	1
reg_lambda	[0.1, 0.5, 1, 2.5]	1

A.3 Example of Comparables for a Random Property - CSM

This example illustrates the practical application of our model. The target property, sold in June 2021 and located in Frogner, is an apartment featuring a living area of 62 m^2, one bathroom, and a balcony. In Figure 9, the property under consideration is represented by a red marker, while the blue markers indicate the ten properties most similar to it. In addition to geographical location, other variables taken into account in the calculation include the presence of a bathroom, a balcony, a construction date before 2010, living area sizes ranging between 50 and 69 m^2, and similar sale dates.



(a) Zoomed out **(b)** Zoomed in

Figure 9: Comparables for a random property in Frogner

The actual value of the price per square meter for this specific property is NOK 119,355. Our model predicts a value of NOK 103,065, indicating a PPE of 13,6%. Thus, this prediction qualifies for inclusion in the presentations of PPE(20) and PPE(50), but not PPE(10).

A.4 Python Code for Custom Metric - CSM

We present below the code we developed to consider only the K nearest neighbors that were sold prior to the target property for prediction. However, given its high computational demand, especially when used in stacked generalization, we chose not to include this code in our final analysis.

```
def custom_distance(x1, x2):
    days_since_diff = x1[-1] - x2[-1]
    if days_since_diff < 0:
        return float('inf')
    else:
        return np.linalg.norm(x1[:-1] - x2[:-1])
```

The custom distance function in the code snippet factors in the "days since" variable, representing the time elapsed from the transaction date. Specifically, the "days since" variable is normalized such that the oldest transaction date has a value of 0, the newest has a value of 1, and all other transaction dates have values in between this range. The function compares the "days since" values of the potential comparables and the subject property to determine if they were sold before or after the subject property. If the difference between the values in the "days since" variable is negative, it means that the property was sold after the subject property, and the function returns a distance of infinity, effectively excluding that comparable property from consideration. Conversely, if the "days since" value for a comparable property is positive, implying it was sold before the target property, the function computes the Euclidean distance using "np.linalg.norm(x1[:-1] - x2[:-1])".

Please note that in our analysis, we used the regular Euclidean distance calculation for the KNN algorithm, as described in Section 4.2. We only included the code snippet to show the possibility of doing it a more proper way when splitting the training and test sets randomly. However, we do not recommend this approach for large datasets.

A.5 Regression Summary - LAD

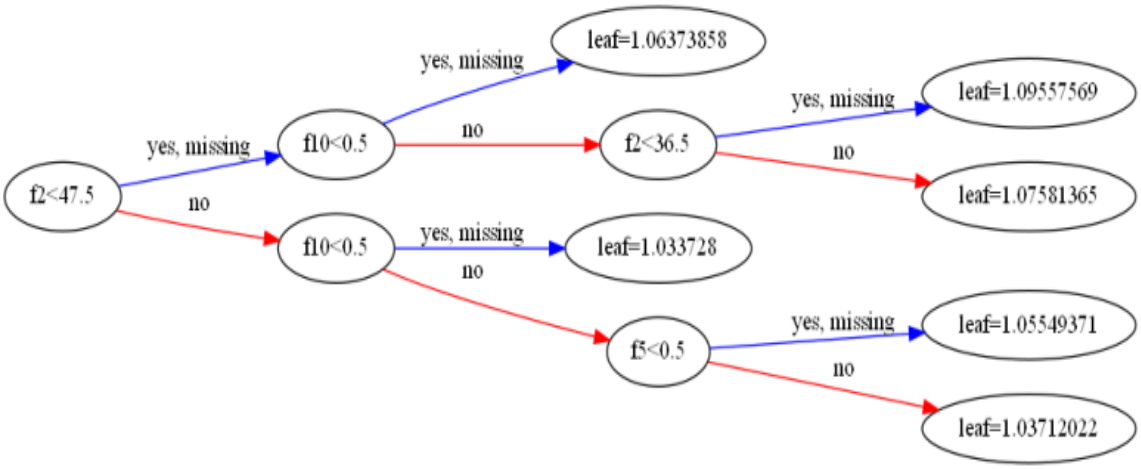
Table 6: Regression summary for the LAD model

ln(Price per square meter)	Coefficients	Std Err
Size		
10-49m^2	0.172***	(0.001)
70-99m^2	-0.056***	(0.001)
100-139m^2	-0.090***	(0.002)
140-179m^2	-0.092***	(0.004)
Above 180m^2	-0.099***	(0.007)
District		
Alna	-0.534***	(0.002)
Bjerke	-0.383***	(0.002)
Gamle Oslo	-0.229***	(0.002)
Grorud	-0.584***	(0.003)
Grünerløkka	-0.180***	(0.002)
Nordre Aker	-0.154***	(0.002)
Nordstrand	-0.347***	(0.002)
Sagene	-0.132***	(0.002)
St. Hanshaugen	-0.065***	(0.002)
Stovner	-0.701***	(0.003)
Søndre Nordstrand	-0.672***	(0.003)
Ullern	-0.126***	(0.002)
Vestre Aker	-0.229***	(0.003)
Østensjø	-0.422***	(0.002)
Year		
2007	-0.021	(0.042)
2008	-0.826***	(0.004)
2009	-0.815***	(0.003)
2010	-0.738***	(0.003)
2011	-0.649***	(0.002)
2012	-0.569***	(0.002)
2013	-0.534***	(0.002)
2014	-0.514***	(0.002)
2015	-0.402***	(0.002)
2016	-0.247***	(0.002)
2017	-0.187***	(0.002)
2018	-0.188***	(0.002)
2019	-0.154***	(0.002)
2020	-0.096***	(0.002)
2022	0.060***	(0.002)
Month		
January	-0.028***	(0.002)
February	-0.017***	(0.002)
March	-0.012***	(0.002)
April	-0.007***	(0.002)
May	-0.000	(0.002)
July	0.004*	(0.002)
August	0.017***	(0.002)
September	0.007***	(0.002)
October	0.006***	(0.002)
November	0.004**	(0.002)
December	0.010***	(0.002)
Balcony	0.023***	(0.001)
Garage	0.025***	(0.001)
Quiet	0.011***	(0.001)
View	0.018***	(0.001)
First floor and inner city	-0.046***	(0.002)
High floor and no elevator	-0.021***	(0.002)
New apartment	0.102***	(0.002)
Renovation object	-0.117***	(0.002)
Two bathrooms or more	0.074***	(0.002)
const	11.493***	(0.002)
R-squared: 0.8367		
Number of observations: 123,489		

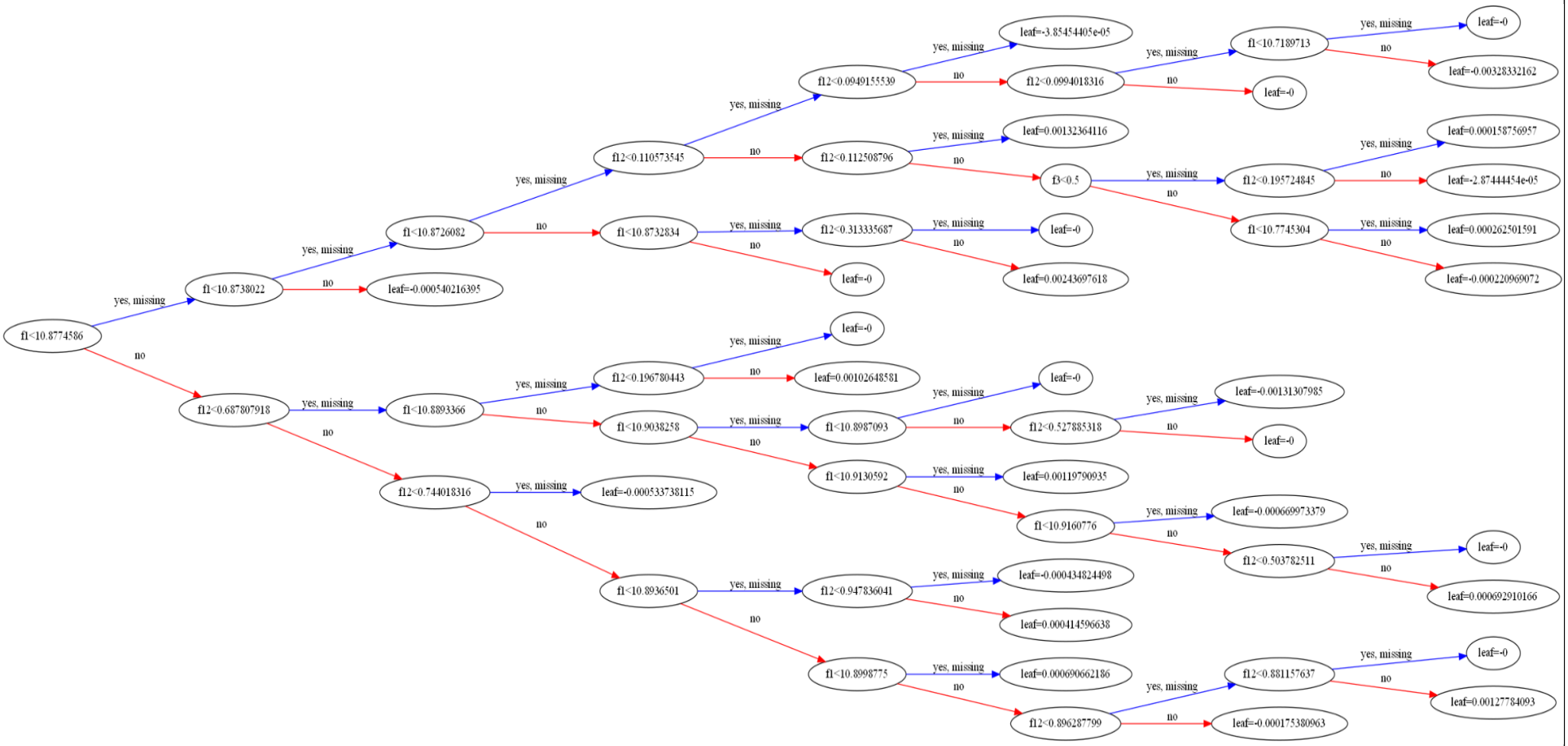
***Significant at the 1% level.
**Significant at the 5% level.
*Significant at the 10% level.

A.6 First and Last Decision Trees - XGB

We developed an XGBoost model with 500 available trees. The trees are grown sequentially, and is improving for each tree. The first tree is relatively small with few branches, making it easy to interpret. However, the last tree becomes significantly more complex with many branches. The variable names are set as **f0**, **f1**, ..., **f12** for visualization purposes, and represents the variables (in this order): "latitude", "longitude", "living area", "high floor and no elevator", "two bathrooms or more", "renovation object", "first floor and inner city", "new apartment", "balcony", "garage", "quiet", "view" and "days since".



(a) First tree



(b) Last tree

Figure 10: XGBoost decision trees

A.7 Simulation with Out-of-Time Predictions

A potential drawback of our models is that they do not incorporate out-of-time predictions, which is necessary to evaluate the performance of the models in real-world situations. We provided a possible solution for the CSM in Appendix A.4. However, to address this limitation across all our models, a simpler and more effective approach involves modifying the division of the training and test sets. Therefore, we conducted a simulation dealing with the time perspective to examine its potential impact on our conclusions. In this simulation, we sorted the dataset by sales date and selected the 2,000 most recent observations as the test set, while the training set consisted of the remaining data (162,652 observations). The test data, spanning from mid-September to the end of December 2022, provided an opportunity to assess our models' performance with out-of-time predictions. The results of this simulation are presented in Table 7.

Table 7: Comparison of model performance with out-of-time predictions

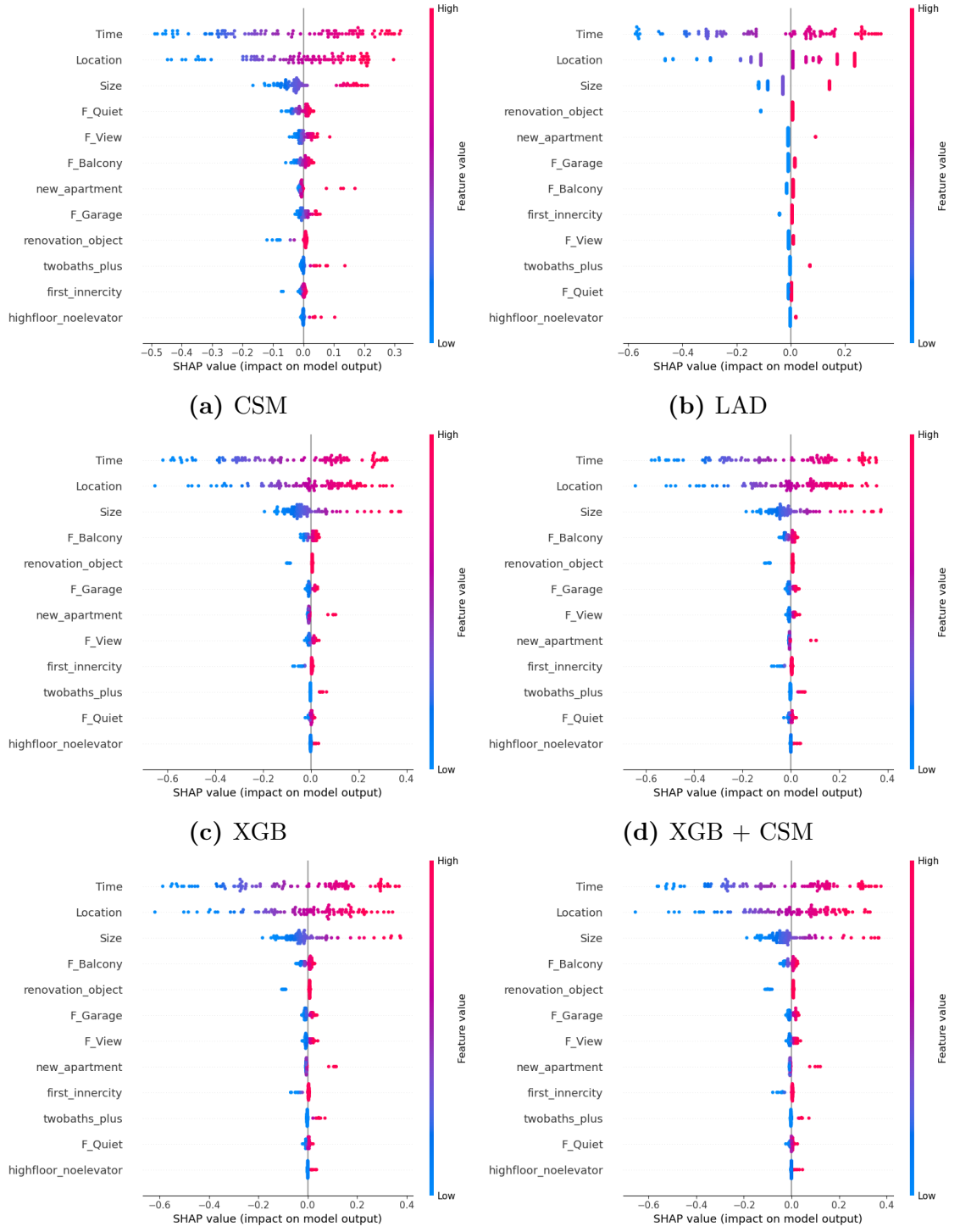
Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.7777	0.1330	10.02%	7.85%	59.95%	88.40%	99.80%
LAD	0.7112	0.1516	12.03%	9.67%	51.15%	81.15%	99.35%
XGB	0.8881	0.0944	7.48%	5.82%	74.50%	**95.20%**	99.85%
(XGB + CSM)	0.8918	0.0928	7.39%	5.81%	74.15%	**95.20%**	99.85%
(XGB + LAD)	0.8887	0.0941	7.49%	5.92%	73.35%	**95.20%**	**99.90%**
(XGB + CSM + LAD)	**0.8920**	**0.0927**	**7.33%**	**5.78%**	**75.00%**	95.05%	99.85%

As expected, the overall performance drops significantly when dealing with the time perspective. However, we can see that the models' relative performance remains consistent to the models trained with the original 75/25 train-test-split approach, as shown in Table 4. XGB remains the best-performing individual model, while LAD continues to fall short. Furthermore, all stacked models still show improvements compared to the individual XGB on a large portion of the evaluation metrics.

While the results are somewhat less distinct than before, our main conclusion remains consistent. CSM continues to contribute more to improved prediction accuracy than LAD when applied to a large dataset using stacked generalization. However, one noticeable difference is that stacking all three models now yields slightly better results compared to only stacking the XGB and CSM models, which contradicts our previous findings. One advantage of XGB and CSM over LAD is their ability to fine-tune sales timing more accurately. However, when this advantage is diminished, the comparative advantage of these models over LAD is reduced, and the inclusion of LAD in a stacking approach becomes more beneficial. We should note that minor differences in outcomes might be due to using the same hyperparameters from our previous analysis, without performing a grid search on the new training set. Adjustments to these hyperparameters could potentially yield different results.

However, these findings demonstrate that our initial methodology can be generalized, and the division of the dataset is not a critical factor in determining model performance.

A.8 Beeswarm Plots - SHAP



(a) CSM **(b)** LAD

(c) XGB **(d)** XGB + CSM

(e) XGB + LAD **(f)** XGB + CSM + LAD

Figure 11: SHAP beeswarm plots for all models

A.9 Simulation with Different Data Sizes - All Tables

Table 8: Comparison of model performance with different sizes of training data (% of original data)

(a) 61,745 observations (50%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8545	0.1387	10.52%	8.11%	59.05%	87.00%	99.36%
LAD	0.8377	0.1465	11.17%	9.01%	54.56%	84.79%	99.69%
XGB	0.9314	0.0953	7.15%	5.56%	75.55%	95.76%	**99.91%**
(XGB + CSM)	**0.9331**	**0.0941**	**7.07%**	**5.50%**	**76.08%**	**95.94%**	**99.91%**
(XGB + LAD)	0.9310	0.0955	7.18%	5.63%	75.50%	95.72%	**99.91%**
(XGB + CSM + LAD)	0.9323	0.0946	7.11%	5.55%	75.75%	95.87%	99.90%

(b) 43,221 observations (35%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8435	0.1438	10.93%	8.38%	57.55%	85.86%	99.19%
LAD	0.8377	0.1465	11.18%	9.02%	54.49%	84.77%	99.68%
XGB	0.9288	0.0970	7.28%	5.64%	74.97%	95.34%	99.90%
(XGB + CSM)	**0.9302**	**0.0961**	**7.21%**	**5.62%**	**75.36%**	**95.63%**	99.90%
(XGB + LAD)	0.9290	0.0969	7.29%	5.70%	74.83%	95.40%	**99.91%**
(XGB + CSM + LAD)	0.9300	0.0962	7.24%	5.63%	75.06%	95.52%	99.89%

(c) 30,872 observations (25%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8331	0.1485	11.30%	8.57%	56.35%	84.78%	99.03%
LAD	0.8376	0.1466	11.17%	9.03%	54.55%	84.77%	99.68%
XGB	0.9268	0.0984	7.37%	5.72%	**74.53%**	95.16%	99.87%
(XGB + CSM)	**0.9274**	**0.0980**	**7.34%**	**5.70%**	74.52%	95.16%	**99.90%**
(XGB + LAD)	0.9255	0.0992	7.44%	5.75%	73.76%	95.11%	99.89%
(XGB + CSM + LAD)	0.9269	0.0983	7.38%	5.76%	74.06%	**95.23%**	99.89%

(d) 24,698 observations (20%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8252	0.1520	11.57%	8.78%	55.35%	83.91%	98.93%
LAD	0.8373	0.1467	11.17%	9.01%	54.48%	84.78%	99.69%
XGB	0.9243	0.1000	7.47%	5.79%	73.76%	94.92%	99.88%
(XGB + CSM)	**0.9254**	**0.0993**	**7.43%**	**5.74%**	**74.03%**	**95.12%**	**99.89%**
(XGB + LAD)	0.9241	0.1002	7.51%	5.81%	73.58%	95.03%	**99.89%**
(XGB + CSM + LAD)	0.9251	0.0995	7.46%	5.85%	**74.03%**	94.98%	99.88%

(e) 18,523 observations (15%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.8153	0.1563	11.90%	9.03%	54.25%	83.16%	98.72%
LAD	0.8370	0.1468	11.18%	9.01%	54.49%	84.74%	99.69%
XGB	0.9216	0.1018	7.60%	**5.87%**	73.08%	94.52%	99.87%
(XGB + CSM)	**0.9223**	**0.1013**	**7.58%**	**5.87%**	**73.12%**	**94.76%**	99.87%
(XGB + LAD)	0.9215	0.1019	7.64%	5.91%	72.77%	94.59%	**99.89%**
(XGB + CSM + LAD)	0.9218	0.1017	7.62%	5.89%	73.05%	94.62%	99.88%

(f) 12,349 observations (10%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.7992	0.1630	12.50%	9.52%	52.03%	81.30%	98.57%
LAD	0.8366	0.1470	11.20%	9.02%	54.50%	84.76%	99.69%
XGB	0.9173	0.1046	**7.83%**	**6.02%**	**71.95%**	94.09%	99.85%
(XGB + CSM)	0.9172	0.1047	7.85%	6.11%	71.70%	**94.22%**	99.83%
(XGB + LAD)	0.9172	0.1046	7.86%	6.10%	71.37%	94.13%	**99.86%**
(XGB + CSM + LAD)	**0.9176**	**0.1044**	7.85%	6.08%	71.84%	94.13%	**99.86%**

(g) 6,174 observations (5%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.7655	0.1761	13.63%	10.30%	48.78%	78.03%	98.00%
LAD	0.8357	0.1474	11.24%	9.06%	54.09%	84.59%	99.67%
XGB	**0.9102**	**0.1089**	**8.15%**	**6.32%**	**70.14%**	**93.38%**	99.83%
(XGB + CSM)	0.9082	0.1101	8.24%	6.37%	69.63%	93.12%	99.83%
(XGB + LAD)	0.9085	0.1100	8.26%	6.44%	69.38%	93.16%	**99.86%**
(XGB + CSM + LAD)	0.9087	0.1099	8.23%	6.37%	69.88%	93.08%	99.84%

(h) 3,087 observations (2.5%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.7201	0.1924	15.11%	11.32%	44.98%	74.06%	97.13%
LAD	0.8337	0.1483	11.32%	9.12%	54.04%	84.33%	99.63%
XGB	**0.8982**	**0.1160**	**8.74%**	**6.74%**	**66.99%**	**91.91%**	99.78%
(XGB + CSM)	0.8952	0.1177	8.88%	6.92%	66.11%	91.57%	99.77%
(XGB + LAD)	0.8946	0.1180	8.88%	6.91%	66.12%	91.50%	**99.81%**
(XGB + CSM + LAD)	0.8952	0.1177	8.84%	6.89%	66.52%	91.67%	99.79%

(i) 1,235 observations (1%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.6540	0.2139	17.10%	12.92%	40.17%	68.68%	95.70%
LAD	0.8225	0.1532	11.71%	9.40%	52.63%	83.23%	99.50%
XGB	**0.8815**	**0.1252**	**9.50%**	**7.41%**	**62.91%**	**89.81%**	**99.74%**
(XGB + CSM)	0.8730	0.1296	9.82%	7.66%	61.47%	88.96%	99.66%
(XGB + LAD)	0.8717	0.1302	9.92%	7.85%	60.67%	88.62%	99.71%
(XGB + CSM + LAD)	0.8710	0.1306	9.95%	7.87%	60.41%	88.78%	99.68%

(j) 617 observations (0.5%)

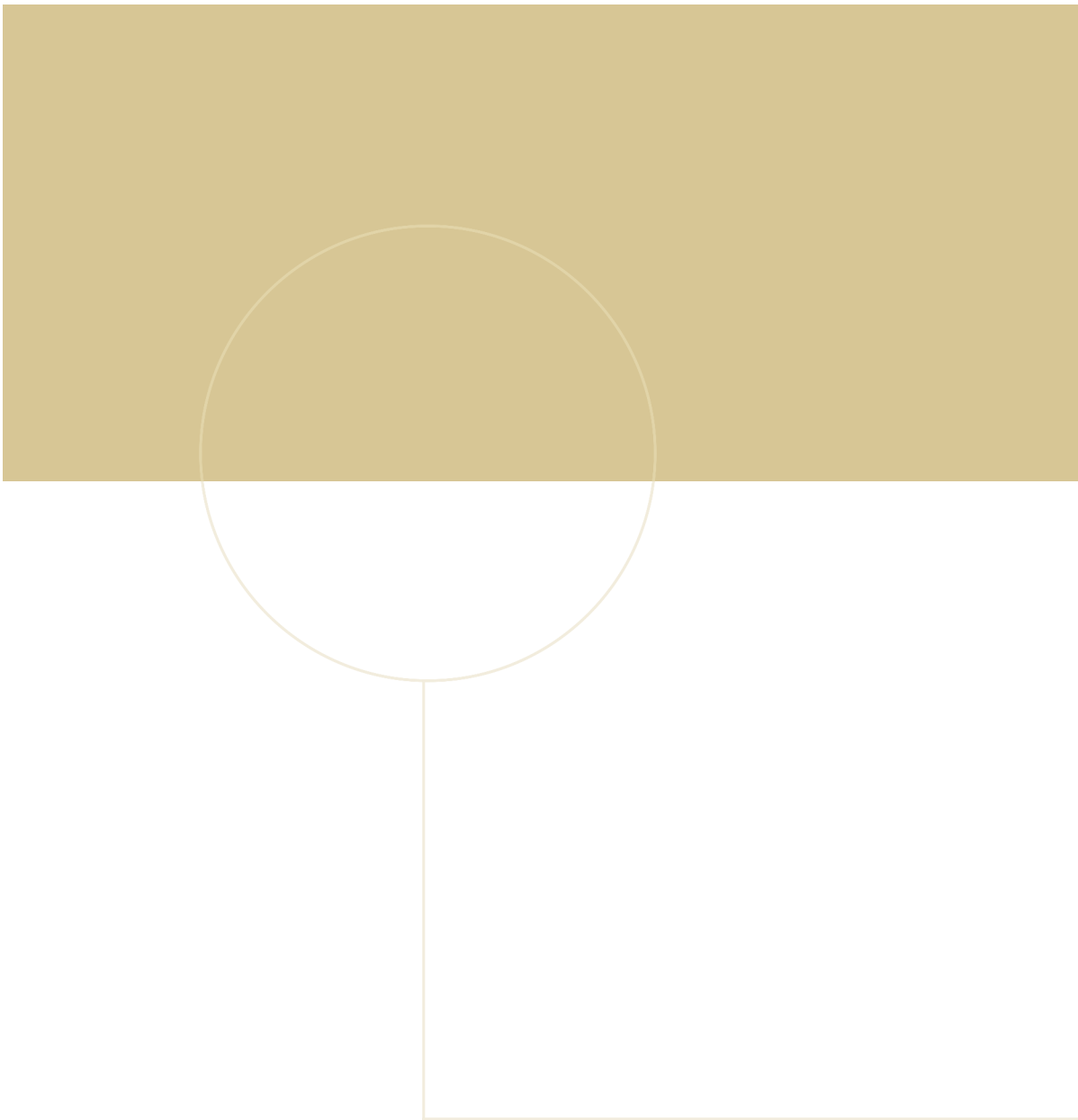
Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.5858	0.2340	18.97%	14.27%	36.63%	64.55%	94.04%
LAD	0.8009	0.1623	12.53%	9.92%	50.34%	80.80%	99.11%
XGB	**0.8607**	**0.1357**	**10.47%**	**8.25%**	**58.43%**	**87.18%**	**99.61%**
(XGB + CSM)	0.8470	0.1422	11.00%	8.73%	55.78%	85.54%	99.51%
(XGB + LAD)	0.8475	0.1420	10.97%	8.62%	56.47%	85.43%	99.53%
(XGB + CSM + LAD)	0.8457	0.1428	11.03%	8.68%	55.99%	85.16%	99.49%

(k) 308 observations (0.25%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.4983	0.2576	21.41%	16.17%	32.87%	59.05%	91.98%
LAD	0.7954	0.1645	12.69%	10.17%	49.35%	80.28%	99.11%
XGB	**0.8403**	**0.1453**	**11.29%**	**8.80%**	**55.29%**	**84.38%**	99.40%
(XGB + CSM)	0.8162	0.1559	12.05%	9.37%	52.66%	82.26%	99.10%
(XGB + LAD)	0.8311	0.1494	11.52%	9.16%	53.61%	83.72%	**99.41%**
(XGB + CSM + LAD)	0.8295	0.1502	11.51%	9.09%	53.94%	83.90%	99.31%

(l) 123 observations (0.1%)

Models	R^2	RMSE	MAPE	MdAPE	PPE(10)	PPE(20)	PPE(50)
CSM	0.4685	0.2651	21.86%	16.92%	31.13%	57.28%	92.12%
LAD	0.6413	0.2178	17.42%	13.60%	37.98%	67.29%	96.19%
XGB	**0.8081**	**0.1593**	**12.26%**	**9.97%**	**50.14%**	**81.32%**	**99.34%**
(XGB + CSM)	0.7546	0.1801	13.99%	11.33%	44.90%	75.70%	98.70%
(XGB + LAD)	0.7474	0.1827	14.13%	11.56%	44.02%	75.27%	98.88%
(XGB + CSM + LAD)	0.7475	0.1827	14.06%	11.42%	44.52%	75.30%	98.91%



NTNU

Norwegian University of
Science and Technology