Bachelor Thesis:
Econometrics & Operational Research

Big data mining:

Investigating potential improvement of parsimonious factor analysis with machine learning, principal component variations and regularization.

Author:
Richie Lee *(505917KL)*
Date:
July 2, 2021

Supervisor:
Philip Hans B.F. Franses
Second assessor:
Carlo Cavicchia

Abstract

In recent years, "Big data" has grown in recognition and cultivated in what we can now consider an indispensable scientific field in the world of data mining. Though commonly associated with extensive datasets, it has also found benefits in parsimonious modelling. These models, characterized by their great explanatory power from minimal resources, are especially valuable in cases of limited or low-frequency data.

This thesis will investigate the potential of parsimonious factor analysis extension through machine learning and shrinkage methods (Boosting, Elastic Net), as well as principal component alternatives (Independent components, Sparse principal components).

While forecasting national house prices, empirical analyses show promising results in favor of the model extensions. However, because traditional models remained significantly relevant, it is concluded that model variations are better suited to be used in conjunction with traditional factor models, rather than replacing them entirely. The resulting explanatory power improvements are briefly touched upon using mean model combinations, which displayed the greatest performance overall by considerable margins.

Keywords: Prediction - Boosting - Elastic Net - Independent component analysis - Sparse component analysis - Mean model combinations

Contents

1 Introduction

In recent years, "Big data" has grown in recognition and cultivated in what we can now consider an indispensable scientific field in the world of data analysis. Though commonly associated with extensive datasets, it has also found benefits in parsimonious modelling. These models, characterized by their great explanatory power from minimal resources, are especially valuable in cases of limited or low-frequency data. A study conducted by Kim and Swanson (2018) explores this topic with parsimonious models by investigating a wide variety of topics such as machine learning, data mining, variable selection, dimension reduction, and shrinkage. Additionally, there is attention for Independent component analysis (ICA) and Sparse principal component analysis (SPCA) as alternatives to principal component analysis (PCA). Overall, this large repertoire of modelling features was shown to have promising results while forecasting macroeconomic variables. This thesis seeks to extend this literature by analyzing whether these methods maintain their functionality when forecasting Dutch house prices.

Wei and Cao (2017) call attention to house price forecasts as important indicators of the real estate market's health and stability, providing value for house investors, real estate developers and government regulators. Moreover, as emphasized by Leung (2004), there has been a growing recognition of the importance of the interactive connection between the real estate market and the macroeconomy. For this reason, macroeconomic variables are used to forecast house prices.

This paper will mainly focus on factor analysis techniques, which are not only useful in "higher" dimension analyses, but also provide robustness to multicollinearity amongst the macroeconomic covariates. To refine the predictions, multiple hybrid models will be examined. PCA, ICA, SPCA, Boosting (an ensemble machine learning method), Elastic nets (a penalty-based shrinkage method), and Mean model combinations will be applied to improve aspects such as bias, variance, variable selection and overfitting.

However, rather than outperforming current state-of-art prediction models, the chief goal of this thesis will be to provide the reader a representative illustration of the potential of parsimonious, (hybrid) factor models. This research is intended to lay out some groundwork for future research, where optimizing particular features could improve factor modelling for low-frequency data cases in general. This leads to the following research question:

RQ: *Under which conditions can parsimonious factor models be improved through the usage of machine learning, shrinkage methods and principal component variations?*

The results in Section 5 show promising potential for both machine learning and principal component alternatives. However, traditional methods still remain relevant, which is why (hybrid) model selection is concluded to be case-specific. For this reason, hybrid models are probably better suited to accompany traditional factor models, rather than replacing them.

A simple implementation of this idea was explored through mean model combinations. These models displayed promising results, by achieving the overall best performance. Therefore it is hypothesized that (more advanced) model combination variations are potentially the most appealing options in parsimonious modelling settings.

2 Literature

The foundation of this thesis, *Mining big data using parsimonious factor, machine learning, variable selection and shrinkage methods* by Kim and Swanson (2018) focuses on the usefulness of factor models in the context of prediction using big data. In particular, this research examines performance of models using low-frequency macroeconomic variables. Furthermore, they also consider a wide array of model variations, extensions, and principal component alternatives. Overall, the findings suggest dimension reduction associated with the specification and estimation of factors, as well as machine learning and shrinkage methods, to be very useful for forecasting macroeconomic variables when analyzing big data. Additionally, substantial empirical evidence was found suggesting Independent component analysis (ICA) and Sparse principal component analysis (SPCA) to be interesting alternatives to principal component analysis (PCA) when estimating factors.

Secondly, I consider whether the assumption of substantial causation between macroeconomic variables and house prices is reasonable. According to existing literature, this decision could be justified by considering findings of Leung (2004); Wei and Cao (2017) and Galati et al. (2011), of which the latter analyzed the Dutch market as well. Statistical tests are applied as well with the Granger-causality test that evaluates the H_0 of sufficient explanatory power of variables. The testing results are included in Appendix B.

3 Data

The data that is used in this paper originates from CBS (Centraal Bureau Statistiek), the national statistics institute of the Netherlands. This organisation is backed by the government and provides open-source information through their databank known as "StatLine".

This thesis uses 2 datasets[1] with basic information provided in Table 1. Firstly, the input data: "Key figures by sector". This dataset consists of a number of Dutch key figures of the sector accounts. These indicators provide the most important information on the total economy and on the main institutional sectors of the economy: non-financial corporations, financial corporations, general government, households including non-profit institutions serving households (NPISHs), and the rest of the world. After removing unusable variables (due to missing values), 57 variables (N) remain, which are reported quarterly, spanning from the first quarter of 1999 to the last of 2020 ($T = 88$ quarters). The raw input variables descriptions alongside variable selection and missing value details are presented in Appendix G.

Secondly, the target data: "Existing own-homes, price indices", which are based on registered transactions from *Kadaster* and WOZ-values of all homes in the Netherlands. This dataset reports 9 variables that describe price indices, average selling prices of new homes, and amount of transactions. This paper will focus on 2 of those 9 variables, *price indices* and *average selling prices* (Figure 1). These figures are available per quarter as well and include observations from 1995 up to and including 2021's first quarter ($T = 105$ quarters). One key distinction between *price indices* and *average selling price* is the fact that the latter is unable to capture price changes of existing homes. For this reason, both price indicators will be analyzed separately.

[1] Both datasets are both accessible online, through following links for the key figures and house price data respectively: *https://opendata.cbs.nl/statline/#/CBS/nl/dataset/84097NED/table*, *https://opendata.cbs.nl/#/CBS/nl/dataset/83906NED/table*

Table 1: Data summary

Table 1 reports the period with corresponding quarters, amount of quarters (T), distinct variables (N), and missing observations. These values were measured after removing unused variables (7 for both datasets) with no additional data preprocessing.

Data set	Period	T	N	Missing values
Key figures	1999:1 - 2020:4	88	57	55 (0.01 %)
House price data	1995:1 - 2021:1	105	2	0 (0.00 %)

3.1 Data preprocessing

The data preprocessing procedure is quite minimal due to the relatively high raw data quality. No modifications to index-based variables were required as both the input and target datasets were computed with 2015 as baseline.

For the *key figures* data set (input), I first omit a single variable "*net external assets, market value*", due to the absence of data prior to 2015. Additionally, there are 6 variables that are only recorded a on year basis in December. For these figures, I will use lagged yearly observations at quarter Q_{4t} to forecast quarters Q_{4t+1} up to and including Q_{4t+4} to avoid data leakage (using future data to forecast the past), as information recorded in December should not be available prior in that year. The key figures dataset also includes 55 missing values which are all located at the start of their respective series. This is an issue because they cannot be filled with values based on past data (since they are the first observations). Future data usage through e.g. arithmetic averages or random sampling from variable distributions does not work either, as it would cause data leakage. However, due to the scale of missing data being approximately 0.01%, I decided to leave these values empty as these are expected to not significantly hurt the analyses. Besides, this missing data will be present across all the models, yielding a comparison that remains fair (details on missing values are provided in Appendix G).

The *house price* dataset (target) requires attention for both non-stationarity and seasonality. These undesired properties are managed by converting the observed values to yearly growth rates using the following data-transformation:

$$y_{\text{growth rate, } t} = \log(\frac{y_t}{y_{t-4}}) \times 100\%. \tag{1}$$

By using the changes in the target variables, stationary time series are obtained with not only better out-of-sample forecasts but also a new interpretation. Instead of the price (index) itself, the models will now forecast (1-quarter ahead) target yearly growth or decline[2]. In addition, seasonality is solved as well by using ratios of observations at t and $t-4$, in which the same quarters are used for every computation.

Afterwards, 7 variables are omitted, as these provide no additional relevant information when compared to the selected 2. This dataset contains no empty values, however. the first 4 years of data (1995-1998) are lost when merging to match the input data dimensions.

[2] In other words, the target variable is now interpreted as forecasting how much growth the next quarter will have seen, when compared to the same quarter a year prior. Hence a yearly growth rate, as the growth spans over a period of a year.

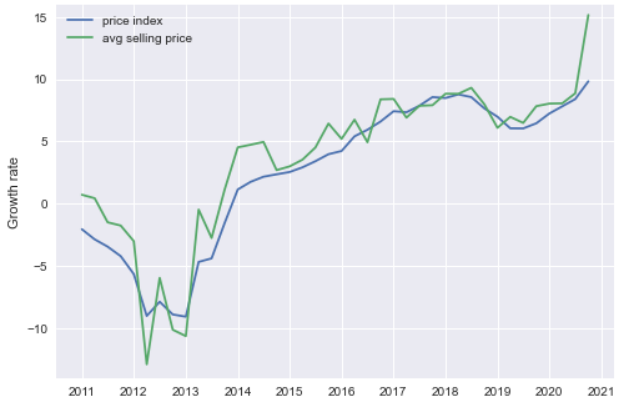


Figure 1: Target variables

Figure 1 displays the target variables after the growth rate transformation (only the test period is shown). For example, a growth rate of 101 indicates that the current quarter has grown 1% when compared to the same quarter of the year prior. Furthermore, the original realized values are included in Appendix A.

3.2 Train-test split

With the exception of the autoregressive benchmark model, all models require tuning. For this purpose, a train/validation/test split is implemented alongside an expanding window (also known as recursive estimation). This method uses a consistent validation size (5 quarters), a consistent testing size (1 quarter) and increases the training size by one observation with every iteration. For instance, let τ denote the current time, then this would yield the following split:

Table 2: Train-test split at $t = \tau$

Training	Validation	Test
$[0, \tau - 5)$	$[\tau - 5, \tau]$	$\tau + 1$

4 Methodology

This thesis is intended to extend the research of Kim and Swanson (2018) by investigating whether their promising findings for (machine learning hybrid) factor models still uphold when forecasting alternative datasets (Dutch house prices instead of macroeconomic variables). The methodology will also be very similar to the original research, in which multiple factor models, extended by machine learning and shrinkage, are introduced. These methods will then be compared to more traditional time series methods to benchmark performance.

In my experiments, I decided to construct all the hybrid models as follows: factors are first constructed using the full dataset, with each of PCA, ICA and SPCA. These factors will then be the input data for 1-step ahead forecasts (1 quarter). With the goal of covering a wide variety of machine learning and shrinkage functionalities, I included Boosting, an ensemble machine learning algorithm, and Elastic Net, a combination of penalty-based shrinkage methods.

The performance of these hybrid models is also dependent on the hyperparameter selection (as shown in Appendix D). The hyperparameters, as well as the number of components, are tuned for every iteration as follows: First, the hybrid model hyperparameters are tuned through a *grid search*. In other words, a predetermined set of hyperparameter combinations are evaluated with the validation dataset after which the best performing combination is used for testing (the predetermined combinations are presented in each hybrid models' respective subsections). After obtaining the set of machine learning hyperparameters, I will then tune the number of components (k) through trial and error as well (on the validation set). Due to computational limitations, I decided to limit the component tuning by restricting the options to multiples of 2, less or equal to 20 ($k = 2, 4, 6, \cdots, 20$). Additionally, the machine learning hyperparameters are held constant during component tuning. The hyperparameter selections are all included in Appendix H.

4.1 Model variations

4.1.1 First order autoregression with drift [AR(1)]

An AR(1) autoregressive process is one in which the predicted current value ($\hat{y}_t$) is based on the immediately preceding value (y_{t-1}). In addition, slow steady changes are accounted for as well, through the introduction of drift. This yields the following specification:

$$\hat{y}_t = y_{t-1} + \eta. \tag{2}$$

In this equation, drift is denoted by η and can be interpreted as a constant.

4.1.2 Ordinary least squares [OLS]

Ordinary least squares (OLS) refers to a linear regression model that assumes a linear relationship between input variables and the target variable. With a single input variable, this relationship is a line, and with higher dimensions, this relationship can be thought of as a hyper-plane that connects input variables to the target variable. The coefficients are estimated through optimization of a loss objective function ($\mathcal{L}$) that seeks to minimize the sum of squared errors between the predictions ($\hat{y}$) and true target values (y):

$$\mathcal{L}_{OLS} = \sum_{t=1}^{N} (\hat{y}_t - y_t)^2 \tag{3}$$

A common problem with linear regression is that coefficients can become large, resulting in a model which is sensitive to its input variables and potentially unstable. This is especially true for models with few observations/samples (n) and models with fewer observations than input predictors or variables (p).

A variation on linear regression is component regression, in which the input variables are substituted with latent variables which are obtained through component analyses. The OLS extensions included in this thesis are Principal component regression (PCR), Independent component regression (ICR), and Sparse principal component regression (SPCR). These variations are more elaborately discussed in Section 4.2.

4.1.3 Elastic net

Elastic net (EN) is a machine learning method, that extends linear regression by implementing regularization penalties to the loss function that encourage simpler models with smaller coefficient values. More specifically, it combines both L2 (Ridge regression) and L1 (LASSO) penalty functions.

Ridge regression uses L2 penalties which extend OLS with shrinkage estimators (ridge estimators) which theoretically produce new estimators that are shrunk closer to the "true" population parameters. This is especially useful when multicollinearity (highly-correlated explanatory variables) is present. In addition, it does not require the number of observations (n) to exceed the number of parameters (p). It is important to note that ridge regression does not omit variables. Instead, variables coefficients converge to 0 as their utility declines.

Additionally, Elastic nets also include L1 penalties, which are obtained using LASSO (Least Absolute Shrinkage and Selection Operator). This machine learning method operates like ridge regression but distinguishes itself by completely omitting variables, contrary to ridge regression, which only converges coefficients to 0. In general, LASSO tends to omit all the variables but one, when handling correlated parameter sets. This property makes it especially useful in estimations with lots of redundant (highly-correlated) variables.

Elastic net seeks to capture the best of both worlds, by implementing both penalties alongside tunable hyperparameters. This paper considers the following Elastic net specification:

$$P_{EN} = (\alpha * P_{L1}) + ((1\text{–}\alpha) * P_{L2}) \tag{4}$$

$$\mathcal{L}_{EN} = \mathcal{L}_{OLS} + \lambda * P_{EN}, \tag{5}$$

in which $P_{\{L1,L2,EN\}}$ denote L1, L2 and Elastic net penalty functions respectively. $\mathcal{L}_{\{EN,OLS\}}$ refer to the loss functions for Elastic net and OLS, of which the latter is shown in equation 3. The first hyperparameter, *alpha* (α), is a value between 0 and 1 that is used to weigh the relative contribution of L1 and L2 regularization (i.e. $\alpha = 1$, uses LASSO only). The second parameter is *lambda* (λ), which controls the weight of the penalty functions combined. Tuning of these hyperparameters is done through grid search, by searching the best performing parameter pair (during validation) from all combinations of $\alpha \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$ and $\lambda \in \{0.1, 0.25, 0.5, 1, 1.5\}$.

4.1.4 Boosting

Gradient boosting refers to a set of ensemble machine learning algorithms, suitable for both classification and regression predictive modelling. Ensembles are constructed from decision tree models (sets of sequential, hierarchical decisions that ultimately lead to some final result). Trees are added one at a time to the ensemble and fit to correct the prediction errors of previous models.

Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm, hence the name, "gradient boosting", as the loss gradient is minimized as the model is fit (in a similar way as neural networks). This implementation should not only improve variance and stability but can also contribute by reducing forecast bias. However, in return, there is an increased risk of overfitting.

More specifically, this thesis implements Extreme Gradient Boosting (XGBoost), an algorithm initially developed by Chen and Guestrin (2016) to achieve computational efficiency while maintaining strong performance.

XGBoost uses 5 hyperparameters, of which two (*eta* & *max depth*) were tuned in this research. Firstly, *n_estimators*, which refers to the number of trees in the ensemble. Secondly, *max_depth* which controls the complexity of trees. Thirdly, *eta*, the learning rate that weighs every model. Fourthly, *subsample*, the number of samples (rows) used in each tree. And lastly, *n_features*, the number of features (columns) used in each tree. Randomness is used in the construction of the model. This means that each time the algorithm is run on the same data, it may produce a slightly different model.

Tuning of these hyperparameters is done through grid search, by searching the best performing parameter pair (during validation) from all combinations of *eta* $\in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 1, 1.25, 1.5, 2\}$ and *max depth* $\in \{1, 2, 3, 5, 6, 10\}$.

4.1.5 Mean model combination

Mean model combinations form predictions by taking the arithmetic average of other model predictions. This thesis aims to investigate the potential of improving individual models through combinations, as a means to capitalize on the variety of the model advantages. For this purpose, all model combinations are computed, evaluated and sorted by performance. This method is intended to illustrate which models excel in combination environments and how much forecast accuracy can be enhanced.

This paper only considers time-invariant model selection. In other words, the same combination is used throughout the entire testing period. Alternatively, it could be interesting to investigate stochastic combination selection, in which the best combination (evaluated on the validation set) would be selected for every iteration. However, exploration of this idea is left for future research.

4.2 Principal component variations

4.2.1 Principal component analysis [PCA]

Firstly, Principal Component Analysis (PCA), a well-established method that seeks to reduce dimensions while preserving as much information as possible. After taking the original variables as inputs, PCA derives a set of independent latent variables which are ordered by captured variance. These latent variables are also known as *principal components* (denote k = number of components used). The underlying idea is that the first k principal components would capture the majority of the information, under the assumption that the components that capture the most variance would be the most informative (generally, but not necessarily true). It is then a matter of optimizing the trade-off between model simplicity (by reducing k) and total information (by increasing k)

Reducing dimensionality improves both computational efficiency and overfitting, as these aspects suffer from large sets of variables. This is especially true for machine learning algorithms. By eliminating correlations amongst the components, PCA also improves performance and visualization options. However, in return for these benefits, PCA sacrifices information contained within the variables as well as their interpretability.

4.2.2 Independent component analysis [ICA]

Furthermore, this paper also includes Independent Component Analysis (ICA). When originally introduced by Comon (1994), the concept of ICA was presented as an extension of PCA, capable of extracting hidden factors (often referred to as *sources*) within data by transforming a set of variables to a new set of maximally independent components.

The procedure (Oja and Hyvarinen (2000)) in which independent components ($\mathbf{Y}$) are obtained starts by representing the components as linear combinations of original explanatory variables ($\mathbf{X}$), with weight matrix $\mathbf{\Omega}$:

$$\mathbf{Y} = \mathbf{X\Omega} = \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_n x_n. \tag{6}$$

The ICA procedure can then be further understood by observing that all input variables ($\mathbf{X}$) can be interpreted as linear combinations of hidden sources ($\mathbf{S}$), weighted by unobserved matrix $\mathbf{\Psi}$:

$$\mathbf{X} = \mathbf{S\Psi}, \tag{7}$$

which can be rewritten as:

$$\begin{aligned} x_1 &= \psi_{11} s_1 + \psi_{12} s_2 + \cdots + \psi_{1k} s_k, \\ x_2 &= \psi_{21} s_1 + \psi_{22} s_2 + \cdots + \psi_{2k} s_k, \\ &\vdots \\ x_n &= \psi_{n1} s_1 + \psi_{n2} s_2 + \cdots + \psi_{nk} s_k. \end{aligned}$$

Now, the target variables can be explained through the sources as well, by plugging equation 7 in equation 6 and collecting coefficients $\mathbf{\Psi}$ for each variable ($\theta_i = \sum_{j=1}^{n} \psi_{ij}$). The target variable can then expressed as follows:

$$\mathbf{Y} = (\mathbf{S\Psi})\mathbf{\Omega} = \mathbf{S\Theta} = \theta_1 s_1 + \theta_2 s_2 + \cdots + \theta_k s_k. \tag{8}$$

With the equality of equations 6 and 8 in mind, ICA relies on two key assumptions which require independent components and sources to be one, statistically independent, and two, non-Gaussian.

Assumption 2 of non-Gaussianity is used to quantify independence. More specifically, it quantifies how far the distribution of a random variable is from being Gaussian, through measures such as negentropy or kurtosis. The usefulness of these measures follows from the Central Limit Theorem stating that the sum of multiple independent random variables will have a distribution that is closer to Gaussian than either of the original variables. Hence, given the independence of the sources (by assumption 1), a sum of multiple s_i's, or equivalently multiple non-zero θ_i's, will be closer to Gaussian and thus reduce independence.

In other words, given the assumptions, the non-Gaussianity of component y_i is maximized when it is directly proportional to one of the sources. And as a result, ICA can be framed as an optimization problem:

$$\max_{\mathbf{\Omega}} \quad kurtosis(\mathbf{X\Omega}), \tag{9}$$

in which non-Gaussianity is maximized by solving for weight matrix $\mathbf{\Omega}$. If the assumptions are (sufficiently) correct and if optimization is successful, ICA then obtains independent components (y) each corresponding to a source (s).

The algorithm that this thesis applies is "FastICA", an efficient and popular ICA algorithm invented by Hyvarinen (1999). Input variables were preprocessed through PCA, which besides covering necessary *prewhitening* (centering and whitening) has been shown to be helpful in general for ICA (Draper et al. (2003)). Additionally, to help independence among sources, I performed variable selection in which 26 (out of the 57) "potentially Gaussian" original variables are omitted through a Jarque-Bera normality test [3]

Lastly, notice that the number of inputs and outputs are the same, and since the outputs are mutually independent there is no obvious way to drop components like in Principal Component Analysis (PCA). For this reason, I set the number of selected components (k) to 26, for every iteration using ICA.

4.2.3 Sparse principal component analysis [SPCA]

Lastly, Sparse Principal Component Analysis (SPCA). Compared to PCA, Sparse principal components yield a more parsimonious and interpretable representation, clearly emphasizing which of the original features contribute to the differences between samples. This addresses one of the drawbacks of PCA i.e. the non-zero loadings which are hard to interpret.

SPCA reduces dimensionality of data by introducing sparsity structures to the input variables. Because sparse principal components are not restricted to orthogonal basis vectors, they allow more adaptability in representing the data. As a result, it finds linear combinations with minimal input variables, by setting redundant variables to 0.

There are many different formulations that achieve this result. The one implemented here is based on Mairal et al. (2009) (and similar to Zou and Hastie (2005)) and can broadly be explained through a unsupervised learning method known as *Sparse coding*. This method aims to find a latent representation ($\mathbf{h}^{(t)}$) that reconstructs the original input ($\mathbf{x}^{(t)}$) as well as possible, while maximizing sparsity (number of zeros in vector $\mathbf{h}^{(t)}$). This yields the following objective function:

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} \|\mathbf{x}^{(t)} - \mathbf{D}\mathbf{h}^{(t)}\|_2^2 + \lambda \|\mathbf{h}^{(\mathbf{t})}\|_1, \tag{10}$$

in which $\mathbf{D}$ denotes a so-called "dictionary", constrained to having columns of norm 1[4]; $\mathbf{D}\mathbf{h}^{(t)}$ the reconstruction; $\|\mathbf{x}^{(t)} - \mathbf{D}\mathbf{h}^{(t)}\|_2^2$ the *reconstruction error*; $\|\mathbf{h}^{(\mathbf{t})}\|_1$ the *sparsity L1 penalty* (similar to LASSO); and λ the hyperparameter which controls the balance between reconstruction and sparsity control (default λ is used).

Equation 10 can be interpreted as a trade-off, in which I minimize the sum of the negative correlated terms: reconstruction error (quality of data representation) and L1 penalty (degree of sparsity). The inner loop seeks to optimize for all training examples ($\forall t \in \{1, \cdots, T\}$) by finding the best (penalized) latent representation ($\mathbf{h}^{(t)}$):

[3] The Jarque-Bera test uses a more severe confidence level of 0.01% due to the small number of observations (T = 81), with which asymptotic approximations do not hold.

[4] If this constraint on $\mathbf{D}$ is not satisfied, $\mathbf{D}$ could grow too big, resulting in a $\mathbf{h}^{(t)}$ that is unable to satisfy the prior.

$$\mathbf{h}(\mathbf{x}^{(t)}) = \arg\min_{\mathbf{h}^{(t)}} \frac{1}{2}\|\mathbf{x}^{(t)} - \mathbf{D}\mathbf{h}^{(t)}\|_2^2 + \lambda\|\mathbf{h}^{(\mathbf{t})}\|_1. \tag{11}$$

Then, in the outer loop of equation 10, the entire training set is used to find the best dictionary matrix ($\mathbf{D}$) that optimizes reconstruction and sparsity (while mindful of the constraints on $\mathbf{D}$). This application of $\mathbf{D}$ is comparable to *dictionary learning* (Mairal et al. (2009)), although detailed derivations of this topic are beyond the scope of this research.

In short, SPCA can in broad terms be considered a PCA optimization problem (reconstruction error minimization) that cleverly uses L1 penalties on the components to introduced sparsity, which contribute to reducing noise when handling few training samples. A more elaborate discussion of the exact programming algorithm applied is presented in the Scikit-Learn documentation[5].

4.3 Performance evaluation

The main analysis methods in this paper will include various performance evaluations and comparisons, while mindful of the following aspects:

1. Are machine learning and shrinkage methods capable of contributing to principal component analysis through hybrid models?
2. Are independent/sparse component analyses capable of improving principal component analysis?

In order to illustrate the contributions of machine learning to traditional factor modelling, principal component regression (OLS - PCA) will be used as a benchmark in comparison to hybrid model performance (Boosting - PCA and Elastic Net - PCA). Afterwards, I will expand the analyses to various component types, in which model variations are held consistent for a fair comparison. The interpretation of these empirical results will then be based on their statistical performance metrics and forecast behaviour overall.

The first performance metric is unmodified errors:

$$e_t = \hat{y}_t - y_t. \tag{12}$$

The main motivation for this straightforward measure lies in its convenient interpretability in terms of error magnitude, forecast bias, and variance. Other popular alternatives such as squared errors or absolute errors are not visualized directly, because errors magnitudes are visibly displayed through error distributions already (Figure 10). However, squared errors are used during computations of the R^2 and Diebold-Mariano statistics, to avoid errors "cancelling each other out" while taking sums. Moreover, summary statistics of the errors are included as well (Table 5), which will be help investigate potential occurrences of bias-variance trade-offs.

[5] Scikit Learn SPCA user guide: *https://scikit-learn.org/stable/modules/decomposition.html#sparsepca*

The second performance measure is the R^2. This statistical measure shows the percentage of the dependent variable variation that the linear model explains, and is computed as follows:

$$R^2 = 1 - \frac{SSE}{SSR} = 1 - \frac{\sum_{t=1}^{N}(\hat{y}_t - y_t)^2}{\sum_{t=1}^{N}(y_t - \bar{y})^2}. \tag{13}$$

The SSE and SSR are abbreviations for sum squared errors/residuals and represent the variation in the forecast errors and dependent variable respectively.

The final relevant performance metric is the Diebold-Mariano [DM] test. This statistical test compares the performance (forecast error magnitudes) of 2 forecast models with a H_0 of equal prediction accuracy. More specifically, this paper applies a modified Diebold-Mariano implementation, in line with Gu et al. (2020) in which comparison of the cross-sectional average of prediction errors from models A and B yields the following expression:

$$DM_{AB} = \frac{\bar{d}_{AB}}{\widehat{\sigma}_{AB}}, \tag{14}$$

where

$$\bar{d}_{AB,t} = \frac{1}{T}\sum_{t=1}^{T}\left((\widehat{e}_t^{\,\mathrm{A}})^2 - (\widehat{e}_t^{\,\mathrm{B}})^2\right). \tag{15}$$

In short, this method computes loss differentials for every t and regresses the resulting series on a constant. The DM-statistic is then obtained through the t-statistic of the regression.

Given the loss differential in equation 15, the DM-statistic is interpreted as favorable for model A when its coefficient is (significantly) negative and vice versa. However, it is important to note that DM-statistic magnitudes are not designed to be used as measures of performance difference.

5 Results

The section will present the results which were obtained through the evaluation methods described in subsection 4.3. The following paragraphs will first compare traditional factor modelling with machine learning and shrinkage variations. Subsequently, these model comparisons will be extended to component alternatives. These performance evaluations will be performed for both target variables (growth rate: price index / average selling price). It is important to note that these findings are empirical conclusions based on minimal testing ($N_{test} = 40$), and the statistical significance of the findings should be considered accordingly.

5.1 Improving factor models with machine Learning and shrinkage

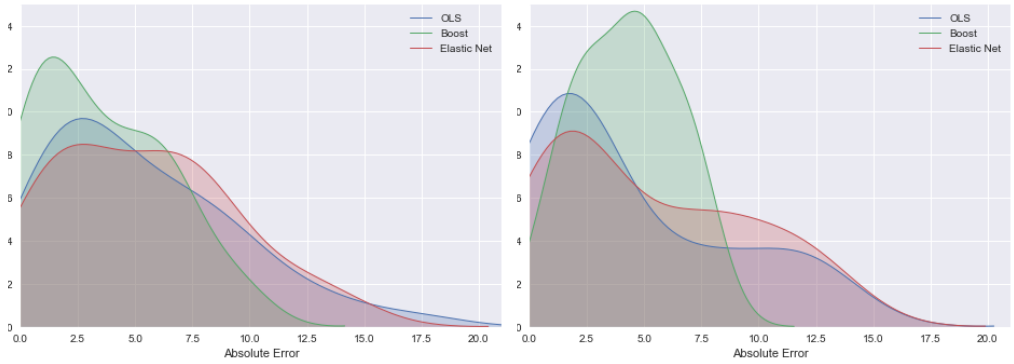
Table 3 displays Diebold-Mariano test statistics which are used to compare predictive abilities of models (more elaborately discussed in section 4.3). The following table contains traditional and hybrid factor models which use principal components, to evaluate potential benefits of machine learning and shrinkage extensions.

Table 3: Diebold-Mariano statistics (PCA)

Table 3 displays the Diebold-Mariano test statistics. Forecast performance for the target variables (with (Δ_y) denoting yearly growth rates) is compared in which negative statistics (**bold**) indicate preference towards the row model (e.g. the value **-1.19** is interpreted as Boosting (row) outperforming OLS/PCR (column) for *Price index* forecasting). The magnitude of the coefficients contain no valuable interpretation.

	Price index (Δ_y)			Average Selling Price (Δ_y)		
	OLS	Boosting	Elastic net	OLS	Boosting	Elastic net
OLS	-	1.19	**-2.59**	-	6.43	4.19
Boosting	**-1.19**	-	**-1.62**	**-6.43**	-	**-4.50**
Elastic net	2.59	1.62	-	**-4.19**	4.50	-

Table 3 indicates promising results for hybrid models. Both target variables experienced better results for machine learning, in which both *Price index* and *Average selling price* showed preference towards Boosting through coefficients **-1.19, -1.62** and **-6.43, -4.50** respectively. Furthermore, Elastic net outperformed OLS for *Average selling price* (**-4.19**), but not for *Price index* (2.59). The Diebold-Mariano is not geared towards measuring the magnitude of performance difference amongst the models. For this reason, Figure 2 is introduced to visualize a error magnitude comparison for PCA variations through usage of absolute error distributions.



(a) Price index **(b)** Average selling price

Figure 2: Absolute error distribution (PCA)

This figure contains absolute forecast error distributions for PCR, alongside the hybrid models. The x and y-axis ranges have been made consistent across the subfigures for comparison convenience.

Figure 2 shows the difference amongst the models in terms of absolute error distributions. This visualization of error magnitudes for PCA variations displays the overall difference in performance amongst the model variations, in which densities near absolute errors of zero are preferred [6].

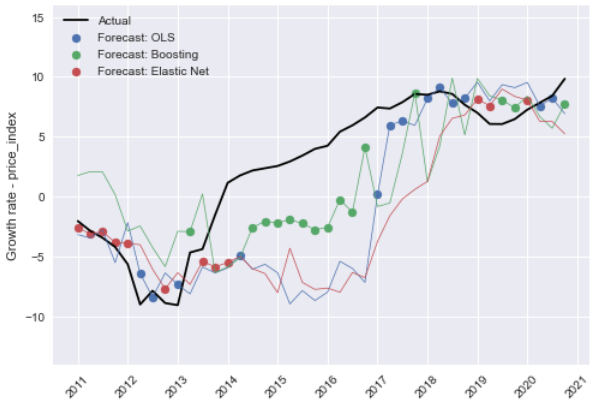
The two target variables show little difference in performance overall. One could consider this to be intuitive as both variables are highly correlated while sharing a largely similar economic interpretation. A comparison of both target variables is provided in Figure 1 (section 3.1). I hypothesize the difference in overall volatility to have been the cause of the slight performance

[6] An alternative visualization of the difference in error distributions is presented in Appendix C. These figures use regular errors, which have the added benefit of distinguishing between positive and negative biases at the expense of error magnitude visibility.

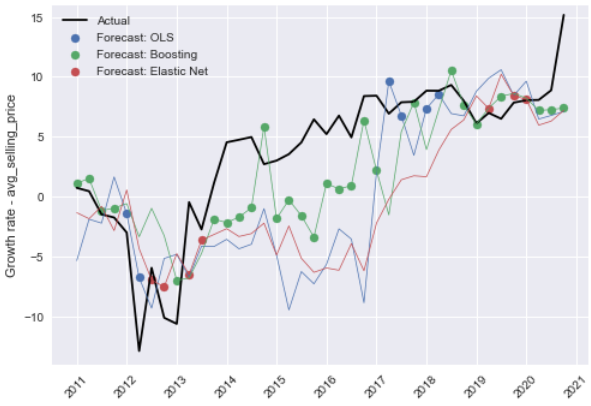
difference, because volatility is often believed to harm performance through increase of overfitting risk. Under this presumption, a favorable robustness for hybrid models is concluded, because of the little performance difference between *price index* and the more volatile target variable *Average selling price*.

Furthermore, a preference towards Boosting is observed as well, as its distributions display significantly less fat tails, alongside larger densities at the lower absolute error values.

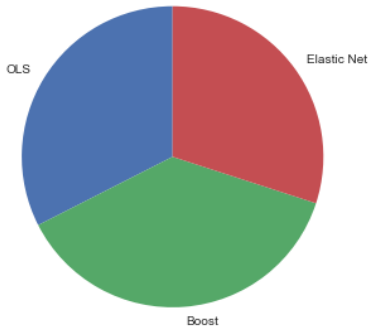
Thirdly, I explored the model performance changes over time. Figure 3 presents the forecast development over time, while also highlighting the best performing model for every T.



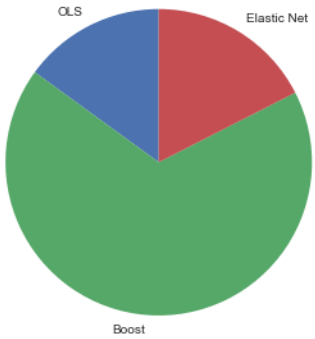
(a) Forecast development - Price index



(b) Forecast development - Average selling price



(c) Model selection - Price index



(d) Model selection - Average selling price

Figure 3: Model selection over time (PCA)

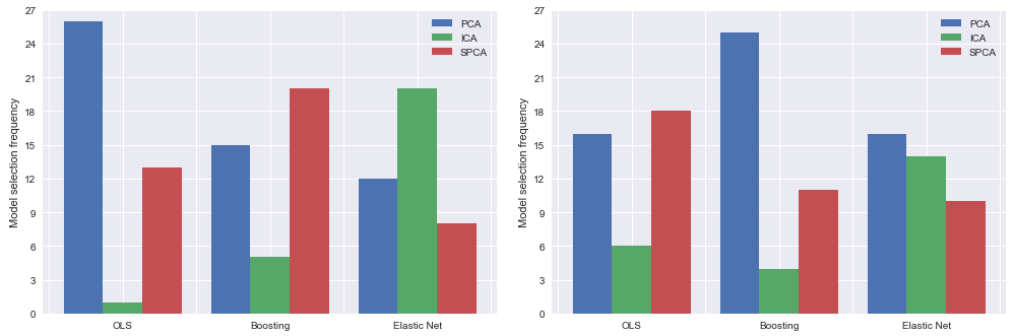
Subfigures 3a & 3b display how the predictions developed over time, with dots representing the best performing models for that particular quarter (the y-axis' growth rates are yearly growth rates. For instance, a value of 5 is interpreted as 5% growth when compared to the preceding year). Afterwards, the amount of selections are summed and represented in the following Pie charts.

Figure 3 displays model selection over time, in which particular models appear to excel for particular extended periods. This could be due to exogenous developments in the target/input data, resulting in compatibility of specific models. Moreover, the significant presence of OLS in the model selection proportions in figure 3c and 3d appear to contradict findings in table 3 that indicated overall preference towards machine learning implementations.

I hypothesize this difference to originate from the difference in evaluation methods. Though both the Diebold-Mariano and model selection frequency can be considered indicators of performance, they differ significantly in their approach. In particular, model selection considers how often the model performed well while disregarding its poor performance (other than it resulting in the model not being selected). The Diebold-Mariano on the other hand uses poor performance significantly more in its evaluation. When considering the different conclusions in table 3 and Figure 3, I suspect OLS/PCR to suffer from "bad periods" (such as 2015-2017, figure 3b) more than the hybrid counterparts. This could be due to overall forecast volatility or due to the frequency of negative outliers in the predictions.

5.2 Improving factor models with principal component variations

This subsection compares principal components (PCA), independent components (ICA) and sparse principal components (SPCA), which are introduced in section 4.2. Figure 4 contains model selection frequencies for the different component variations, which are intended to reflect potential benefits of PCA alternatives.



(a) Price index **(b)** Average selling price

Figure 4: Model selection - Component Type Comparison

Figure 4 displays model selection frequencies for different component variations. These frequencies are obtained by evaluating forecast errors for every t and while tracking how often models performed the best. Every model type (OLS, Boosting, Elastic net) has been considered individually for this analysis. The vertical axis' range has been made consistent across the subfigures for comparison convenience.

Figure 4 shows improvement potential through both Independent and Sparse component analysis. However, PCA still remains preferred frequently and for this reason, I would not consider ICA and SPCA to be replacements for PCA in this instance. Instead, these extensions could be more valuable in complement with PCA.

Contrary to model-type selection (Figure 3), component variation forecast development did not show clear patterns or clustering of model selection over time (Appendix E). However, it is important to note that ICA often failed to sufficiently optimize its objective function during computation (Section 4.2.2), resulting in unsatisfactory independent components for OLS and Boosting (due to computational limitations and data properties). Therefore, its potential could perhaps be undervalued in these results as this method was optimized the least.

During the comparison of component types, a significant difference in training time across component types became apparent. While tuning every iteration, PCA and ICA required several minutes to complete training and testing, whereas SPCA required 1, 4 and 9 hours for OLS, Elastic net and Boosting respectively. In a parsimonious analysis environment such as this one (81 observations total) these computational restrictions are manageable, however as data sets increase in size, this restriction could become more relevant.

In return for demanding training times, significant performance benefits become available. Table 4 shows overall performance difference when applying ICA and SPCA, through Diebold-Mariano test results.

Table 4: Diebold Mariano (Component Variations)

Table 3 displays the Diebold-Mariano test statistics. Forecast performance for the target variables (with (Δ_y) denoting yearly growth rates) is compared in which negative statistics (**bold**) indicate preference towards the row component types (e.g. the value **-14.79** in the OLS category is interpreted as PCA (row) outperforming ICA (column) for *Price index* forecasting). The magnitude of the coefficients contain no valuable interpretation.

		Price Index (Δ_y)			Average Selling Price (Δ_y)		
		PCA	ICA	SPCA	PCA	ICA	SPCA
OLS	PCA	-	**-14.79**	**-4.77**	-	**-12.40**	**-2.71**
	ICA	14.79	-	14.72	12.40	-	12.38
	SPCA	4.77	**-14.72**	-	2.71	**-12.38**	-
Boosting	PCA	-	**-13.34**	10.29	-	**-12.82**	**-5.55**
	ICA	13.34	-	15.84	12.82	-	11.82
	SPCA	**-10.29**	**-15.84**	-	5.55	**-11.82**	-
Elastic Net	PCA	-	**-5.83**	**-9.02**	-	**-6.53**	**-6.10**
	ICA	5.83	-	2.86	6.53	-	5.39
	SPCA	9.02	**-2.86**	-	6.10	**-5.39**	-

Table 4 shows overall preference towards PCA, with the exception of Boosting (*Price index*) with the coefficients **-10.29** and **-15.84** indicating favor towards SPCA. In conclusion, I conclude PCA to generally be preferred, however as seen in figure 4, the alternatives ICA and SPCA still hold valuable information as improvement when compared to PCA.

5.3 Comparison of all models

Table 5 evaluates the performance of all the models through the R^2 (section 4.3), forecast bias, and error variance.

Table 5: Performance Metrics (All Models)

Table 5 contains performance metrics for all the models for both target variables (with Δ_y denoting yearly growth rates). For every model type (OLS, Boosting, Elastic Net) the best performing component variation is highlighted in **bold** for every metric. The best model overall is highlighted as well in **blue** (per metric).
For instance, the highlighted value **-0.15** shows that for category OLS, PCA performs the best in terms of R^2 when forecasting *Price index*. Additionally, **0.49** shows that AR(1) performs the best across all models and component variations in terms of R^2 when forecasting *Price index*.

		Price index (Δ_y)			Average Selling price (Δ_y)		
		R^2	Bias	Variance	R^2	Bias	Variance
OLS	PCA	**-0.15**	-3.24	**27.95**	**-0.26**	-3.49	33.05
	ICA	-10.20	-6.74	330.68	-18.15	**-0.68**	690.50
	SPCA	-0.27	**-3.13**	32.86	-0.34	-4.22	**29.96**
Boosting	PCA	0.31	**-1.38**	21.40	**0.41**	-1.70	**18.26**
	ICA	-0.82	-1.82	57.80	-1.71	-3.98	81.64
	SPCA	**0.45**	-1.46	**16.25**	0.34	**-1.66**	20.89
Elastic Net	PCA	**-0.38**	-4.18	**28.68**	**-0.26**	-4.05	**28.53**
	ICA	-0.59	-3.84	38.31	-0.67	**-3.81**	45.46
	SPCA	-0.47	**-4.15**	31.77	-0.44	-4.30	32.96
AR(1)	Regular	**0.49**	**0.54**	**12.32**	0.22	**0.64**	23.76

Table 5 highlights the benefits of machine learning and shrinkage, as well as component variations. The improvements by hybrid models can be observed by comparing the R^2 of OLS models with the hybrid models. Furthermore, a negative bias for all factor models can be observed. The machine learning hybrid models perform well in terms of variance as well, by significantly reducing variance when compared to OLS (Boosting in particular). The variance of the ICA models is especially high for OLS and Boosting models, because these model categories failed to construct adequate independent components (by failing to sufficiently optimize equation 9, section 4.2.2).

For *price index*, no factor model outperformed the AR(1) benchmark ($R^2_{pi} = \mathbf{0.49}$). I hypothesize this to be the result of the autoregressive model's ability to better capture positive trends (Figure 1). On the other hand, multiple Boosting models did outperform the benchmark when forecasting *average selling price* ($R^2_{asp} = \mathbf{0.41}$, 0.34).

The differences in best-performing models across the target variables suggest that both model and component variation selection are situation-dependent. Hence, it can be concluded that based on these results, no absolute superior model can confidently be established. Instead, it could be worthwhile to explore a wider array of variations, such as in table 5. Exploring multiple models could be especially useful as this thesis has found minimal differences in target data to considerably affect model selection.

5.4 Mean model combinations

This subsection explores mean combination models in which forecasts are obtained through arithmetic averages of sets of models. This method seeks to capitalize on the case-dependency of model selection by taking advantage of multiple models and their perks. All combinations of the 10 models (1023 combinations) have been computed and evaluated using the R^2. Figure 5 highlights the 100 best performing combinations to illustrate the potential of successful combinations.

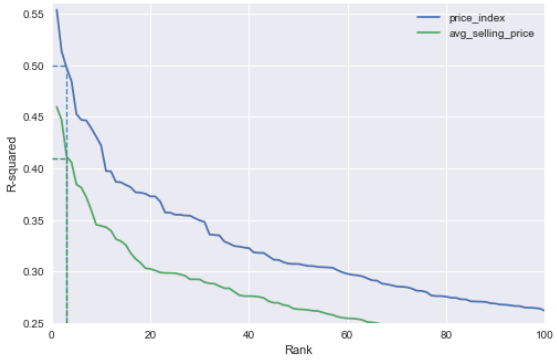
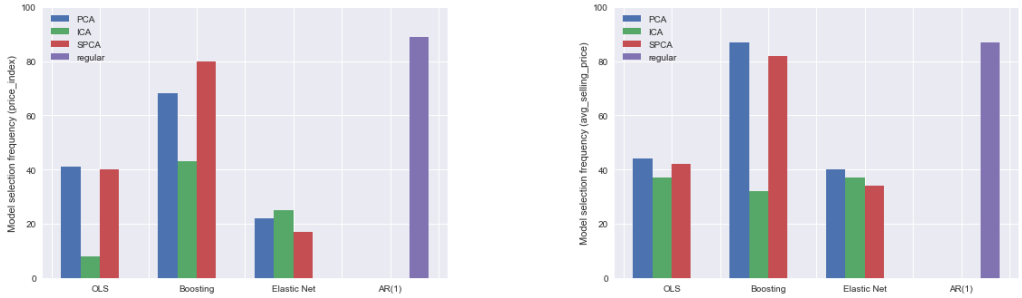


Figure 5: R^2 of top 100 best performing mean combinations

Figure 5 shows the R^2 of the 100 best performing mean combinations. Furthermore, dashed lines are included which highlight the R^2 and Rank of the best performing individual model (mean combinations containing 1 model).

Figure 5 shows significant improvement potential when implementing mean model combinations. Both target variables improved the best performing individual models (indicated with dashed lines) with roughly 5% in terms of R^2. For *price index*, the best individual model performance originated from AR(1), which 2 mean combinations managed to surpass. *Average selling price* was best predicted by BOOST-PCA, and surpassed by 2 combinations as well[7]. Overall these improvements in explanatory power are promising results for mean combination models when paired with autoregressive and (hybrid) factor modelling.

Model selection is key when applying mean model combinations. For this reason, I included Figure 6, which shows how often models were included in the top 100 combinations.



(a) Price index **(b)** Average selling price

Figure 6: Model selection - Top 100 best mean combinations

Figure 6 displays how often models were included in the top 100 best performing mean combinations (based on R^2).

Figure 6 shows an overall favor towards including autoregressive models in the combinations. When considering the forecast developments in Figure 3, it could be the AR(1) model's ability to capture changing trends that adds value, as factor models struggled with this more (especially in 2013-2016, Figure 3).

[7] Extra information that focuses on the set of models outperforming single models are included in Appendix F, as well as the best performing mean combinations.

In short, it shows that a wide variety of model and component variations contribute well to overall performance[8]. As shown in prior sections, there is promising potential in expanding factor models with machine learning, shrinkage, and/or component variation. However, perhaps the greatest success can be achieved through combinations of the models as shown in this section.

6 Conclusion

This section will briefly summarize the main findings of this research in which a wide variety of factor model extensions and alternatives were examined (when forecasting Dutch house prices). The thesis' objective is summarized in the following research question:

RQ: *Under which conditions can parsimonious factor models be improved through the usage of machine learning, shrinkage methods and principal component variations?*

in which machine learning methods: Boosting and Elastic Net, as well as principal component alternatives: Independent component analysis (ICA) and Sparse principal component analysis (SPCA) were considered. The following findings were obtained through empirical analysis of 2 target variables, in which 40 quarters (2011-2020) were used for testing.

Overall, there have been promising results favoring both machine learning extensions and principal component alternatives. This finding of improvement potential through traditional model variations is also in line with Kim and Swanson (2018).

Both Boosting and Elastic Net showed significant performance improvements when compared to traditional principal component regression (PCR), with a preference for Boosting. However, despite its seemingly overall inferior explanatory power, PCR (OLS) still frequently produces the most-accurate predictions. For this reason, machine learning methods are suggested to not entirely replace traditional factor models, but to be regarded as promising case-dependent alternatives. A related finding supporting this "case-dependency", is the fact that models displayed "clustered" periods in which they outperformed the alternatives, hinting at periods in which certain conditions favored certain models (Figure 3). In general, machine learning models were found to be more useful for data sets with weaker trends and volatility.

Furthermore, ICA and SPCA have been shown to be interesting alternatives to principal component analysis (PCA). Overall, a clear compatibility relation between model selection (PCR, Boosting, Elastic Net) and component variations is observed. This effect is especially apparent for ICA, which can be seen in Figure 4 in which its performance improved immensely when paired with Elastic Net[9]. For PCA and SPCA, the performance remains comparable, with relative performance differing across the models as well. Moreover, both ICA and SPCA suffer from computational limitations, resulting in a risk of failing to construct adequate components for ICA and large computation time requirements for SPCA. All things considered, I conclude both ICA and SPCA to be valuable alternatives to PCA, when mindful of model compatibility and computational restrictions.

[8] In figure 6, Independent components analysis (ICA) is likely underrepresented for Boosting and OLS, because the ICA implementations are considerably less optimal (due to computational difficulty, Section 4.2.2). [9] For OLS and Boosting, ICA suffered considerably from computational limitations. For this reason, its potential for these models could be undervalued.

Lastly, mean combination models are briefly discussed as a means to potentially capitalize on the general finding, which suggests that a wide variety of models appears to be capable of excelling, given the right conditions. Every mean combination series is obtained by evaluating the arithmetic average of a set of models forecasts. After selecting the best forecasts from all combinations[10] significant improvement potential (approximately 5% in R^2) was found (Figure 5).

In conclusion, empirical results generally showed preference towards hybrid models which implement Boosting. However, instead of regarding Boosting as superior, it seems more appropriate to consider the findings in this thesis to be indications of model selection case-dependency, highlighting each model's ability to excel under specific (currently largely unknown) conditions. A simple way of capitalizing on this finding was successfully explored, with the overall best performance of mean combination models.

7 Discussion

This section briefly discusses future research suggestions from which I would expect potentially relevant results. These suggestions are based on the main findings and limitations of this Thesis.

7.1 Limitations

Small testing size: The full dataset includes 81 quarterly observations of which 40 were used for testing purposes. The relatively low amounts of data is a "drawback" that could not be avoided, due to this Thesis' focus on parsimonious models, characterised by their small data sets. However, I would like to highlight that the findings/conclusions of this thesis are thus better interpreted as indications for patterns and behaviour that are likely to be present. It is also not guaranteed that the results of this paper represent larger datasets as well.

Tuning of amount of components (k): In order to save computation time, the feasible amount of components for principal component analysis (PCA) and Sparse principal component analysis (SPCA) have been restricted. Instead of evaluating all feasible values ($k \in [1, 52]$), I only allowed multiples of 2 less or equal than 20 to be selected ($k \in \{2, 4, \cdots, 20\}$). While considerably reducing computational power requirements, I expected this approach to adequately represent the models' abilities. As a result, the overall findings can be considered a "minimum" of achievable performance through this methodology.

Furthermore, for the hybrid models, tuning of k had been executed with a predetermined set of tuned hyperparameters. Perhaps, including k in the grid search process of machine learning hyperparameters could significantly improve hybrid model performance, though it would require substantially longer computing times.

[10] All combinations include all possible sets of models (arbitrary length). Initial individual models are also included.

7.2 Future research potential

Model selection clustering: Figure 3 displays model selection over time, in which it appears that model selections often remain the same for extended periods of time (hence the "clustering"). This behaviour is hypothesized to be related to underlying conditions such as particular data properties such as trends or exogenous developments (relevant explanatory variables which are not included in the model). In the best-case scenario, particular general data characteristics are identified, capable of contributing to model selection.

Model combinations: Section 5.4 showed promising results for mean model combinations. The application of arithmetic averages can however be considered relatively basic. Perhaps more advanced variations of (weighted) model combinations can improve forecasts even more. Furthermore, only consistent combinations throughout the entire testing period[11] were considered, though introducing stochastic model selection (e.g. reselecting a new combination for every iteration) could enhance performance as well. Especially because this method could potentially exploit the finding of model selection clustering , under the presumption that model selection clustering increases the likelihood of validation sets sharing the same properties with test sets (given that testing sets immediately follow the validation set in time).

[11] "Consistent combinations" imply that the same model combination is used to predict all values.

References

T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

P. Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

B. A. Draper, K. Baek, M. S. Bartlett, and J. R. Beveridge. Recognizing faces with pca and ica. *Computer vision and image understanding*, 91(1-2):115–137, 2003.

G. Galati, F. Teppa, and R. J. Alessie. Macro and micro drivers of house price dynamics: An application to dutch data. 2011.

S. Gu, B. Kelly, and D. Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273, 2020.

A. Hyvarinen. Fast ica for noisy data using gaussian moments. In *1999 IEEE international symposium on circuits and systems (ISCAS)*, volume 5, pages 57–61. IEEE, 1999.

H. H. Kim and N. R. Swanson. Mining big data using parsimonious factor, machine learning, variable selection and shrinkage methods. *International Journal of Forecasting*, 34(2):339–354, 2018.

C. Leung. Macroeconomics and housing: a review of the literature. *Journal of Housing Economics*, 13(4): 249–267, 2004.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696, 2009.

E. Oja and A. Hyvarinen. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

Y. Wei and Y. Cao. Forecasting house prices using dynamic model averaging approach: Evidence from china. *Economic Modelling*, 61:147–155, 2017.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

Appendix A

Target variables

(a) Price indices **(b)** Average selling price

Figure 7: Target variables (raw)

Figure 7 presents the development of the realized values of the 2 target variables: house price indices and average selling price.

(a) Price indices **(b)** Average selling price

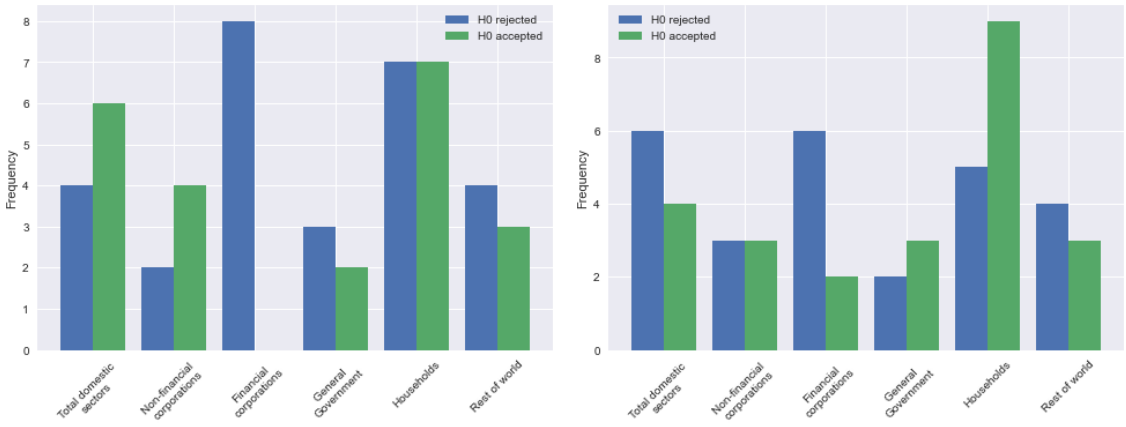
Figure 8: Target variables (growth rates)

Figure 8 presents the development of the growth rate values of the 2 target variables: house price indices and average selling price. (computation is elaborated on in section 3.1).

Appendix B

Data selection justification: Granger causality test

In order to draw meaningful conclusions, it is important that forecasts achieve adequate performance. However, it is not trivial that macro-economic variables are capable of producing reasonable house-price predictions. To justify expectations for input data compatibility, I consider existing literature (section 2) alongside a Granger causality test. This statistical hypothesis test is used for determining whether one time series is useful in forecasting another. Based on the amount of variables expected to have significant predictive power (measured by amount of H_0 rejections (blue) in Figure 9), I hypothesize the input data selection to be sensible given the target data.



(a) Price indices **(b)** Average selling price

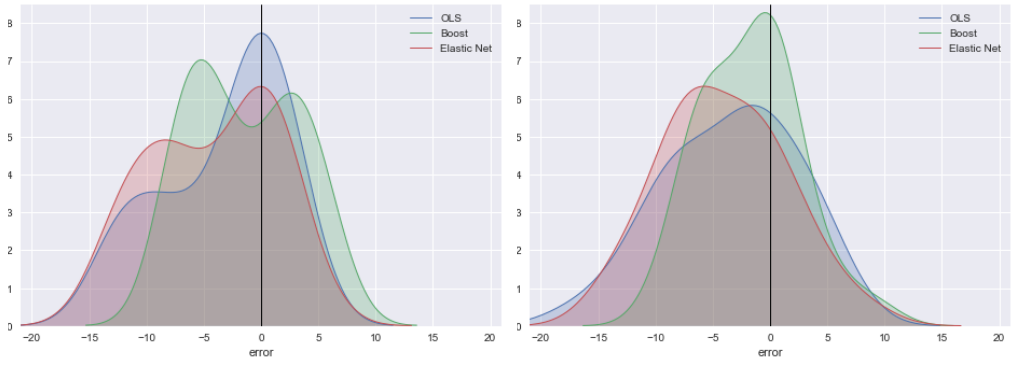
Figure 9: Granger causality for all input variables

Figure 9 reports Granger causality test results. The null hypothesis is that x (input variable) does not Granger-cause y (price index, average selling price). In other words, it is accepted (green) for insufficient predictive causality and rejected (blue) for adequate levels. All tests use a lag length of 2, and significance level of 5%. The analyses were executed using "EViews" (econometric software), before data cleaning.

Appendix C

Regular error distributions

This appendix includes distributions of regular errors. The main benefit of these models when compared to Figure 2 is the added benefit of distinguishing between positive and negative forecast biases. In general a negative bias was observed for all PCA models which is displayed in Figure 10 and Table 5.



(a) Price index **(b)** Average selling price

Figure 10: Error distribution (PCA)

Figure 10 contains forecast error distributions for PCR, alongside the hybrid models. The x and y-axis ranges have been made consistent across the subfigures for comparison convenience.

Appendix D

Hybrid model tuning comparison

(a) Price Index - Boosting

(b) Average selling price - Boosting

(c) Price Index - Elastic Net

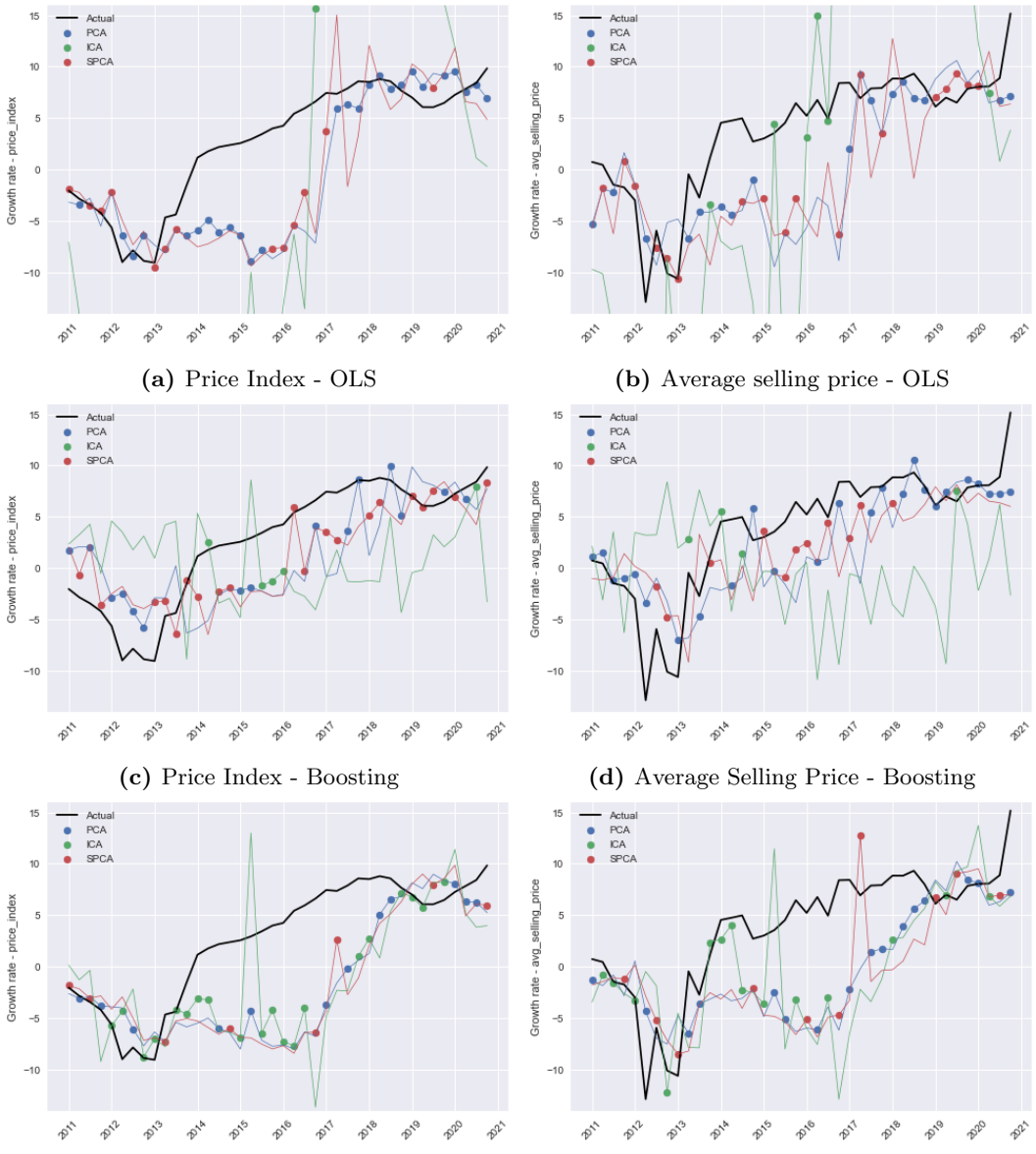
(d) Average Selling Price - Elastic Net

Figure 11: Comparison: Tuned and Not Tuned Forecasts

Figure 11 compares hybrid models (PCA) which were tuned for every iteration ("**Tuned**") with models which were only tuned for the first iteration ("**Not Tuned**").
The y-axis' growth rates are yearly growth rates. For instance, a value of 101 is interpreted as 1% growth when compared to the previous year (same quarter).

Appendix E

Component variation comparison



(a) Price Index - OLS

(b) Average selling price - OLS

(c) Price Index - Boosting

(d) Average Selling Price - Boosting

(e) Price Index - Elastic Net

(f) Average Selling Price - Elastic Net

Figure 12: Forecast development and model selection

Figure 12 displays how the predictions developed over time, with dots representing the best performing models for that particular T.
The y-axis' growth rates are yearly growth rates. For instance, a value of 101 is interpreted as 1% growth when compared to the previous year (same quarter).

Appendix F

Mean model combinations, additional information.

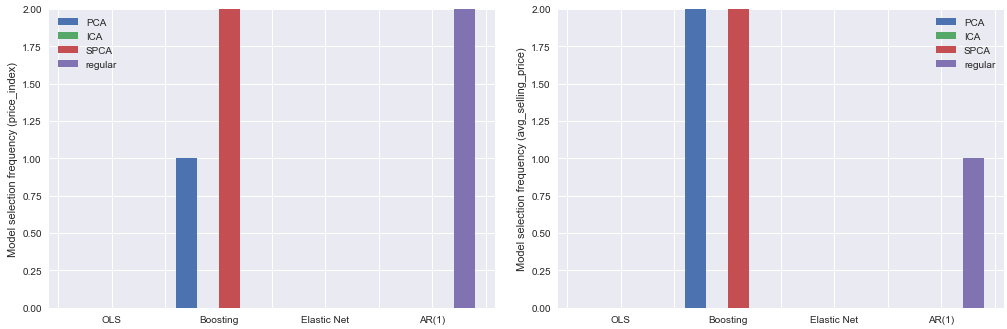
This appendix contains additional information on mean model combinations. In particular, it focuses on the best performing model and the set of models that outperformed the best performing individual models (based on R^2):

- *Price index*: AR(1)
- *Average selling price*: BOOST-PCA

Table 6: Best model combination

Table 6 shows the content of the best model combination, alongside a R^2 comparison of the best combination with the best individual models. Δ_y is used to indicate yearly growth rates of the target variables.

		Price Index (Δ_y)	Average Selling Price (Δ_y)
OLS	PCA		
	ICA		
	SPCA		
Boosting	PCA		×
	ICA		
	SPCA	×	×
Elastic Net	PCA		
	ICA		
	SPCA		
AR(1)	Regular	×	
R-squared	(Best combination)	0.55	0.46
R-squared	(Best individual)	0.49	0.41



(a) Price index **(b)** Average selling price

Figure 13: Model selection - Combinations outperforming best single model

Figure 13 displays how often models were included in the set of combinations which outperformed the best single models.

Appendix G

Input variable (selection) details

Table 7: Input variable selection and descriptions (7 variables omitted)

Table 7 presents all the "Key sectors" raw variables (source: CBS). Variables without available quarterly data are highlighted **red**. These variables can also be classified in either the *omitted* variables column or the *yearly* column in which missing quarterly data are replaced with yearly observations. Lastly, the number of *missing* quarters (after substituting yearly data) are shown in the last column. If no data is missing, a variable contains 88 quarterly observations.

	Description	Omitted	Yearly	Missing
1	Total domestic sectors/Gross domestic product (million euros)			0
2	Total domestic sectors/Consumption of fixed capital (million euros)			0
3	Total domestic sectors/Gross operating surplus and mixed income (million euros)			0
4	Total domestic sectors/Gross national income (million euros)			0
5	Total domestic sectors/Gross disposable national income (million euros)			0
6	Total domestic sectors/Gross national saving (million euros)			0
7	Total domestic sectors/Gross fixed capital formation (million euros)			0
8	Total domestic sectors/Net lending (+) or net borrowing (-) (million euros)			0
9	Total domestic sectors/Net lending to private sector (% GDP)			**3**
10	Total domestic sectors/Lending to private sector, end of period (% GDP)			**3**
11	**Total domestic sectors/Labour input of employed persons (1,000 full-time equivalent jobs)**		×	**4**
12	Non-financial corporations/Gross value added (million euros)			0
13	Non-financial corporations/Gross operating surplus (million euros)			0
14	Non-financial corporations/Gross profits before taxes (million euros)			0
15	Non-financial corporations/Profits from foreign subsidiaries (million euros)			0
16	Non-financial corporations/Profit ratio (% value added)			0
17	Non-financial corporations/Capital formation ratio (% value added)			0
18	**Non-financial corporations/Labour input of employees (1,000 full-time equivalent jobs)**		×	**4**
19	Financial corporations/Gross value added (million euros)			0
20	Financial corporations/Gross profits before taxes (million euros)			0
21	Financial corporations/Profits from foreign subsidiaries (million euros)			0
22	Financial corporations/Financial net worth (million euros)			0
23	Financial corporations/Property income received (% total assets)			**3**
24	Financial corporations/Property income paid (% total liabilities)			**3**
25	Financial corporations/Liquidity ratio mon. fin. institutions (% total assets)			0
26	Financial corporations/Financial assets of pension funds (million euros)			0
27	**Financial corporations/Labour input of employees (1,000 full-time equivalent jobs)**		×	**4**
28	General government (consolidated)/Total revenue (% GDP)			**3**
29	General government (consolidated)/Taxes and social security contributions (% GDP)			**3**
30	General government (consolidated)/Total expenditure (% GDP)			**3**
31	General government (consolidated)/Government debt (EMU) (% GDP)			**3**
32	General government (consolidated)/Balance general government sector (EMU) (% GDP)			**3**
33	**General government (consolidated)/Labour input of employees (1,000 full-time equivalent jobs)**		×	**4**
34	Households including NPISHs/Gross operating surplus and mixed income (million euros)			0
35	Households including NPISHs/Mixed income (million euros)			0
36	Households including NPISHs/Gross disposable income (million euros)			0
37	Households including NPISHs/Real disposable income (% volume changes)			**7**
38	Households including NPISHs/Adjusted disposable income (million euros)			0
39	Households including NPISHs/Final consumption expenditure (million euros)			0
40	Households including NPISHs/Free / individual savings (million euros)			0
41	Households including NPISHs/Savings ratio (% disposable income)			0
42	Households including NPISHs/Households' capital formation ratio (% disposable income)			0
43	Households including NPISHs/Savings deposits and other deposits (million euros)			0
44	Households including NPISHs/Insurance, pension and guarantee schemes (million euros)			0
45	Households including NPISHs/Pension entitlements and claims (million euros)			0
46	Households including NPISHs/Home mortgages; closing balance (million euros)			0
47	Households including NPISHs/Home mortgages; net lending (million euros)			0
48	**Households including NPISHs/Labour input of employees (1,000 full-time equivalent jobs)**		×	**4**
49	**Households including NPISHs/Labour input of self-employed persons (1,000 full-time equivalent jobs)**		×	**4**
50	Rest of the world/Net exports share (% GDP)			0
51	Rest of the world/Net exports (million euros)			0
52	Rest of the world/Net primary income abroad (million euros)			0
53	Rest of the world/Net current transfers abroad (million euros)			0
54	Rest of the world/Net capital transfers abroad (million euros)			0
55	Rest of the world/Surplus on current transactions (million euros)			0
56	Rest of the world/Net external assets (million euros)			0
57	**Rest of the world/Net external assets; market value (million euros)**	×		**64**

Appendix H

Hyperparameters (OLS, Boosting, Elastic Net)

- PI = price index (growth rate)
- ASP = Average selling price (growth rate)

Table 8: Hyperparameters / Amount of components: OLS

	PCA		ICA		SPCA	
Date	$\mathbf{k}_{PI}$	$\mathbf{K}_{ASP}$	$\mathbf{k}_{PI}$	$\mathbf{K}_{ASP}$	$\mathbf{K}_{PI}$	$\mathbf{K}_{ASP}$
2011-01-01	6	20	26	26	10	10
2011-04-01	6	10	26	26	8	12
2011-07-01	4	4	26	26	14	14
2011-10-01	4	2	26	26	4	2
2012-01-01	2	2	26	26	2	2
2012-04-01	2	2	26	26	2	2
2012-07-01	2	2	26	26	2	2
2012-10-01	20	16	26	26	20	2
2013-01-01	10	10	26	26	10	10
2013-04-01	2	2	26	26	10	10
2013-07-01	2	2	26	26	14	16
2013-10-01	2	2	26	26	20	14
2014-01-01	2	2	26	26	20	20
2014-04-01	10	2	26	26	14	14
2014-07-01	10	2	26	26	14	14
2014-10-01	8	8	26	26	10	8
2015-01-01	16	8	26	26	8	8
2015-04-01	16	16	26	26	10	10
2015-07-01	8	2	26	26	8	8
2015-10-01	8	2	26	26	8	16
2016-01-01	8	8	26	26	8	8
2016-04-01	16	16	26	26	18	8
2016-07-01	8	16	26	26	18	18
2016-10-01	20	20	26	26	8	20
2017-01-01	20	20	26	26	18	16
2017-04-01	18	18	26	26	20	18
2017-07-01	18	18	26	26	18	18
2017-10-01	16	16	26	26	18	18
2018-01-01	18	20	26	26	20	20
2018-04-01	20	20	26	26	18	18
2018-07-01	18	20	26	26	18	20
2018-10-01	18	6	26	26	16	16
2019-01-01	16	6	26	26	20	6
2019-04-01	6	8	26	26	8	8
2019-07-01	6	6	26	26	8	6
2019-10-01	8	8	26	26	8	20
2020-01-01	8	8	26	26	12	18
2020-04-01	4	8	26	26	10	20
2020-07-01	8	8	26	26	8	8
2020-10-01	8	8	26	26	8	8

Table 9: Hyperparameters: Boosting - PCA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ETA**	**MAX_DEPTH**	**k**	**ETA**	**MAX_DEPTH**	**k**
2011-01-01	1.5	1	12	2	1	16
2011-04-01	2	2	12	2	5	10
2011-07-01	0.01	3	12	1.5	1	8
2011-10-01	1	3	10	0.01	5	4
2012-01-01	1	1	12	0.01	3	4
2012-04-01	0.3	6	16	2	5	10
2012-07-01	1.5	2	18	0.01	1	10
2012-10-01	2	3	12	2	2	10
2013-01-01	0.05	1	2	0.05	5	14
2013-04-01	0.2	2	10	1	2	10
2013-07-01	1.5	1	18	1	1	14
2013-10-01	2	5	16	1.25	1	8
2014-01-01	1.5	5	20	0.01	3	18
2014-04-01	1	1	10	1.25	5	10
2014-07-01	2	5	12	2	3	10
2014-10-01	2	3	8	2	2	6
2015-01-01	0.01	3	4	2	3	10
2015-04-01	0.01	1	2	0.1	5	10
2015-07-01	0.01	1	2	0.01	10	20
2015-10-01	0.01	1	2	0.01	1	2
2016-01-01	0.01	1	2	2	5	10
2016-04-01	0.01	6	4	0.01	6	18
2016-07-01	0.01	3	4	0.01	10	4
2016-10-01	2	6	10	1.25	2	20
2017-01-01	0.01	3	4	1.5	5	8
2017-04-01	0.01	2	20	1.25	5	10
2017-07-01	0.5	1	18	1.25	1	10
2017-10-01	1.5	3	12	2	1	10
2018-01-01	2	1	16	1.5	1	6
2018-04-01	2	6	8	1.25	5	6
2018-07-01	1.5	2	6	2	1	8
2018-10-01	1	2	10	2	5	18
2019-01-01	2	2	10	0.5	2	10
2019-04-01	1.25	2	20	1.25	10	8
2019-07-01	1.5	1	8	1.25	10	2
2019-10-01	1	5	10	0.2	3	10
2020-01-01	1	5	18	0.1	3	10
2020-04-01	0.3	3	10	1.5	6	10
2020-07-01	1	6	10	1	3	10
2020-10-01	0.1	2	10	2	5	6

Table 10: Hyperparameters: Boosting - ICA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ETA**	**MAX_DEPTH**	**k**	**ETA**	**MAX_DEPTH**	**k**
2011-01-01	0.01	10	26	0.01	6	26
2011-04-01	0.01	3	26	2	5	26
2011-07-01	0.1	5	26	0.01	5	26
2011-10-01	0.01	2	26	1.25	1	26
2012-01-01	1.25	6	26	0.01	6	26
2012-04-01	0.3	1	26	0.01	1	26
2012-07-01	0.01	6	26	0.01	2	26
2012-10-01	0.01	5	26	0.05	3	26
2013-01-01	0.01	6	26	1.25	10	26
2013-04-01	0.1	3	26	1.25	3	26
2013-07-01	0.5	1	26	1.5	3	26
2013-10-01	1.5	5	26	1.5	6	26
2014-01-01	1.5	3	26	0.2	5	26
2014-04-01	1	10	26	2	2	26
2014-07-01	1.5	10	26	0.01	10	26
2014-10-01	0.3	6	26	1.5	10	26
2015-01-01	0.05	6	26	0.01	6	26
2015-04-01	0.05	10	26	0.05	5	26
2015-07-01	0.01	1	26	0.01	6	26
2015-10-01	0.01	1	26	0.01	1	26
2016-01-01	0.01	1	26	0.01	1	26
2016-04-01	0.01	1	26	0.05	3	26
2016-07-01	0.01	1	26	0.01	1	26
2016-10-01	1.25	5	26	0.1	3	26
2017-01-01	0.01	1	26	1.5	6	26
2017-04-01	0.01	1	26	1	2	26
2017-07-01	0.01	1	26	0.01	5	26
2017-10-01	0.01	1	26	0.01	1	26
2018-01-01	0.01	1	26	0.01	10	26
2018-04-01	0.01	1	26	0.1	6	26
2018-07-01	0.3	10	26	0.01	1	26
2018-10-01	1.5	2	26	1.25	10	26
2019-01-01	0.01	1	26	1.5	6	26
2019-04-01	1	2	26	1.5	10	26
2019-07-01	1.5	6	26	0.2	5	26
2019-10-01	1	10	26	1	6	26
2020-01-01	0.3	1	26	1.5	5	26
2020-04-01	1.5	5	26	1	10	26
2020-07-01	0.5	2	26	0.05	10	26
2020-10-01	0.2	6	26	0.1	2	26

Table 11: Hyperparameters: Boosting - SPCA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ETA**	**MAX_DEPTH**	**k**	**ETA**	**MAX_DEPTH**	**k**
2011-01-01	0.01	3	20	1.25	1	10
2011-04-01	1.5	2	20	2	1	8
2011-07-01	1.5	1	14	0.3	1	20
2011-10-01	2	5	16	2	1	10
2012-01-01	0.5	1	18	0.01	1	12
2012-04-01	1	6	10	0.01	2	8
2012-07-01	0.05	1	10	0.01	1	10
2012-10-01	1	5	16	0.5	6	10
2013-01-01	0.05	1	10	0.05	1	10
2013-04-01	0.05	1	10	1.25	10	10
2013-07-01	1.5	2	10	1	5	20
2013-10-01	2	2	16	1.25	3	10
2014-01-01	1.25	1	14	1.5	3	10
2014-04-01	2	5	14	1.5	6	18
2014-07-01	1.5	6	18	1.25	1	20
2014-10-01	0.3	1	14	1.25	1	10
2015-01-01	2	3	10	2	3	8
2015-04-01	0.01	2	10	1.25	2	10
2015-07-01	0.01	1	10	0.01	1	10
2015-10-01	0.01	1	10	1.25	6	10
2016-01-01	0.01	1	12	1.5	5	6
2016-04-01	2	1	10	1.5	3	6
2016-07-01	0.01	6	6	1	2	10
2016-10-01	1.25	5	10	0.5	2	10
2017-01-01	1.25	6	10	0.5	3	14
2017-04-01	1.25	5	10	1.25	3	14
2017-07-01	0.2	2	14	1.25	3	14
2017-10-01	1	3	10	2	10	10
2018-01-01	0.1	2	10	1.5	5	10
2018-04-01	1	5	10	1.25	10	10
2018-07-01	2	3	14	1.25	6	10
2018-10-01	1.5	1	10	0.2	10	10
2019-01-01	1.5	5	10	1.25	5	20
2019-04-01	1.25	6	10	1.25	10	6
2019-07-01	1.5	3	10	1.25	6	10
2019-10-01	0.5	1	6	0.5	1	10
2020-01-01	0.5	1	6	0.5	10	6
2020-04-01	0.1	5	10	0.1	5	6
2020-07-01	0.05	6	14	0.1	1	18
2020-10-01	1.5	2	10	0.05	1	4

Table 12: Hyperparameters: Elastic Net - PCA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ALPHA**	**L1_RATIO**	**k**	**ALPHA**	**L1_RATIO**	**k**
2011-01-01	0.1	0.9	6	0.1	0.1	8
2011-04-01	0.1	0.5	6	0.1	0.1	10
2011-07-01	0.25	0.5	6	0.5	0.9	4
2011-10-01	1	0.25	4	1.5	0.9	4
2012-01-01	1.5	0.75	12	1.5	0.9	2
2012-04-01	1.5	0.9	2	1.5	0.9	2
2012-07-01	1.5	0.9	2	1.5	0.9	2
2012-10-01	1.5	0.9	2	1.5	0.9	2
2013-01-01	1.5	0.9	2	0.1	0.9	10
2013-04-01	0.5	0.9	6	0.25	0.1	2
2013-07-01	0.5	0.75	2	0.5	0.9	2
2013-10-01	0.5	0.9	2	1	0.9	2
2014-01-01	0.5	0.9	2	1	0.9	2
2014-04-01	0.1	0.1	10	1	0.9	2
2014-07-01	0.1	0.1	10	1	0.9	2
2014-10-01	1.5	0.9	2	0.25	0.9	8
2015-01-01	1.5	0.9	6	1.5	0.9	8
2015-04-01	1.5	0.9	6	1.5	0.9	8
2015-07-01	1.5	0.5	6	1.5	0.9	8
2015-10-01	1.5	0.5	6	1.5	0.9	12
2016-01-01	1.5	0.5	6	1.5	0.9	12
2016-04-01	1.5	0.5	6	1.5	0.75	12
2016-07-01	1.5	0.25	8	1	0.9	12
2016-10-01	0.5	0.9	8	0.5	0.9	8
2017-01-01	0.5	0.9	12	1	0.9	12
2017-04-01	0.25	0.9	18	0.5	0.9	8
2017-07-01	0.25	0.9	16	0.5	0.9	6
2017-10-01	0.25	0.9	16	0.5	0.9	6
2018-01-01	0.25	0.9	16	1.5	0.9	6
2018-04-01	0.25	0.9	12	0.25	0.9	12
2018-07-01	0.1	0.1	18	0.1	0.1	20
2018-10-01	0.1	0.1	18	0.1	0.1	6
2019-01-01	0.1	0.9	16	0.1	0.1	6
2019-04-01	0.1	0.9	6	0.25	0.1	6
2019-07-01	0.1	0.9	6	0.1	0.1	6
2019-10-01	0.5	0.9	6	0.5	0.9	8
2020-01-01	0.5	0.9	18	1	0.9	6
2020-04-01	1	0.5	6	1	0.75	8
2020-07-01	1.5	0.25	8	1	0.5	8
2020-10-01	1.5	0.25	8	1.5	0.1	10

Table 13: Hyperparameters: Elastic Net - ICA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ALPHA**	**L1_RATIO**	**k**	**ALPHA**	**L1_RATIO**	**k**
2011-01-01	0.1	0.75	26	0.1	0.75	26
2011-04-01	0.1	0.9	26	0.25	0.25	26
2011-07-01	0.5	0.5	26	0.25	0.9	26
2011-10-01	0.25	0.1	26	1.5	0.9	26
2012-01-01	1	0.25	26	1.5	0.9	26
2012-04-01	1.5	0.1	26	1.5	0.75	26
2012-07-01	1	0.75	26	1.5	0.9	26
2012-10-01	0.1	0.75	26	0.1	0.25	26
2013-01-01	0.1	0.9	26	0.1	0.75	26
2013-04-01	0.1	0.1	26	0.1	0.5	26
2013-07-01	0.25	0.1	26	0.1	0.9	26
2013-10-01	0.1	0.1	26	0.1	0.9	26
2014-01-01	0.5	0.1	26	1.5	0.25	26
2014-04-01	0.25	0.25	26	0.1	0.25	26
2014-07-01	0.1	0.1	26	0.25	0.9	26
2014-10-01	1	0.1	26	0.25	0.75	26
2015-01-01	1.5	0.9	26	1.5	0.9	26
2015-04-01	1.5	0.9	26	1.5	0.9	26
2015-07-01	1.5	0.9	26	1	0.9	26
2015-10-01	1.5	0.9	26	1.5	0.9	26
2016-01-01	1.5	0.9	26	1.5	0.75	26
2016-04-01	1.5	0.75	26	1.5	0.9	26
2016-07-01	1.5	0.9	26	1.5	0.9	26
2016-10-01	0.25	0.1	26	1.5	0.9	26
2017-01-01	0.1	0.5	26	1.5	0.75	26
2017-04-01	0.1	0.1	26	0.1	0.1	26
2017-07-01	0.1	0.1	26	0.1	0.9	26
2017-10-01	0.1	0.75	26	0.1	0.5	26
2018-01-01	0.1	0.1	26	0.1	0.9	26
2018-04-01	0.25	0.25	26	0.1	0.75	26
2018-07-01	0.1	0.9	26	0.1	0.5	26
2018-10-01	0.1	0.75	26	0.1	0.25	26
2019-01-01	0.1	0.75	26	0.1	0.9	26
2019-04-01	0.1	0.75	26	0.1	0.75	26
2019-07-01	0.1	0.9	26	0.1	0.75	26
2019-10-01	0.1	0.25	26	0.1	0.1	26
2020-01-01	0.1	0.1	26	0.1	0.75	26
2020-04-01	0.25	0.75	26	0.25	0.25	26
2020-07-01	0.25	0.75	26	0.25	0.1	26
2020-10-01	0.5	0.5	26	0.25	0.5	26

Table 14: Hyperparameters: Elastic Net - SPCA

	Price Index (Δ_y)			Average Selling Price (Δ_y)		
Date	**ALPHA**	**L1_RATIO**	**k**	**ALPHA**	**L1_RATIO**	**k**
2011-01-01	0.1	0.25	12	0.1	0.1	12
2011-04-01	0.1	0.1	8	0.1	0.1	12
2011-07-01	0.1	0.1	14	1	0.1	6
2011-10-01	1	0.5	12	1.5	0.9	12
2012-01-01	1.5	0.9	6	1.5	0.75	2
2012-04-01	1.5	0.75	2	1.5	0.75	2
2012-07-01	1.5	0.9	2	1.5	0.9	2
2012-10-01	1.5	0.9	20	1.5	0.9	16
2013-01-01	0.5	0.5	20	0.1	0.1	10
2013-04-01	0.1	0.1	10	0.1	0.1	16
2013-07-01	0.5	0.25	18	1	0.1	18
2013-10-01	0.5	0.9	20	1	0.5	16
2014-01-01	0.25	0.75	20	0.5	0.5	2
2014-04-01	0.1	0.75	16	0.25	0.75	10
2014-07-01	0.1	0.1	14	0.1	0.9	14
2014-10-01	1.5	0.1	10	0.1	0.9	10
2015-01-01	1.5	0.1	8	1.5	0.9	10
2015-04-01	1.5	0.1	10	0.5	0.9	10
2015-07-01	1.5	0.1	8	1.5	0.9	8
2015-10-01	1.5	0.9	10	1.5	0.9	8
2016-01-01	0.1	0.1	8	0.1	0.9	8
2016-04-01	1.5	0.1	8	0.25	0.9	8
2016-07-01	0.1	0.75	8	0.25	0.9	10
2016-10-01	0.1	0.5	8	0.1	0.9	8
2017-01-01	0.1	0.9	10	0.1	0.9	18
2017-04-01	0.1	0.1	20	0.1	0.9	20
2017-07-01	0.1	0.1	10	0.1	0.9	10
2017-10-01	0.5	0.1	8	0.25	0.1	8
2018-01-01	0.25	0.1	20	1	0.1	8
2018-04-01	0.1	0.1	18	0.5	0.1	18
2018-07-01	0.1	0.9	20	0.25	0.5	20
2018-10-01	0.1	0.9	20	0.5	0.25	8
2019-01-01	0.1	0.9	10	0.1	0.9	6
2019-04-01	0.1	0.9	8	1	0.1	8
2019-07-01	0.1	0.9	18	0.1	0.9	6
2019-10-01	0.25	0.1	8	0.5	0.1	8
2020-01-01	0.25	0.1	8	0.5	0.1	20
2020-04-01	0.5	0.9	18	0.5	0.1	20
2020-07-01	1.5	0.25	12	1	0.5	12
2020-10-01	1.5	0.1	14	1	0.25	10