Universiteit Leiden
The Netherlands

Opleiding Informatica

Optimally Weighted Machine Learning Model Ensembles using Quadratic Optimization in Python

Maxim Janssen

Supervisors:
Dr. M.T.M. Emmerich, Dr. H. Wang & P. Echtenbruck (external)

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

14/11/2023

Abstract

For both novice and experienced users alike, choosing the right model for a regression task can be a tedious task that requires a lot of knowledge of the inner workings of these models. While combining these models in ensembles can yield better results, most ensemble methods can be either too simplistic or complicated and opaque, complicating optimization attempts. This paper evaluates a recently proposed method of using quadratic programming for the creation of an optimally weighted ensemble through multiple easily accessible datasets. Additionally, this study rewrites the approach in Python to increase its accessibility while expanding upon the method by expanding the algorithmic notation and re-evaluating the approach on fitting the models to datasets. The results indicate that this ensemble method outperforms model selection on multiple different datasets, but dataset size is an important prerequisite for the effectiveness of the ensemble. The code used in this research was made available through github: https://github.com/maximjanssen/optimally-weighted-ensembles.

Contents

1 Introduction

When working with complex data, choosing the right model for machine learning can be a challenge. Choosing the right model requires deep knowledge of its inner workings because the difference between the performance of two models can be large. To simplify this process a data scientist has two different options: to automate model selection (through tools such as AutoML) or to combine multiple models in a 'smart' ensemble.

In 1979 already it was proposed to combine multiple different models to create a model combination or ensemble [5]. Nowadays a wide range of ensembles exists. Within these ensembles two different categories can be distinguished: simple methods (such as Voting and Averaging) and more complicated methods (such as Boosting). The problem with the first method could be that complicated problems should not be simplified because such simplified ensembles will not yield the best result. The latter method could be opaque, meaning that for the user it becomes difficult to know which model in the ensemble has the most impact or how the ensemble could be optimized. When the user doesn't have sufficient knowledge this can lead to a 'trial and error' approach, increasing the risk of overfitting. Creating a weighted average by assigning different weights to different models (with the weights totalling 1) can be a good middle ground between the two methods: assigning weights to 'better' models can yield better results, while the weights clearly convey a level of importance to different models. Additionally, the weights can be easily tweaked without having to refit the models in the ensemble or without even letting the models create new predictions. The main challenge with this approach is that finding the optimal combination of weights can be a challenge, especially for regression tasks as the possibilities are technically endless. Friese et al [7] first proposed this approach as an alternative to model selection and searched for the best weights using box-constrained optimization. Benítez-Peña et al. [1] proposed an interesting approach by presenting the challenge to find the optimal combination of weights as a convex equation using Quadratic Programming and making this challenge more doable. In 2022 Echtenbruck et al. [6] expanded upon this method by replacing it with an exact optimization algorithm.

These studies have opened up a lot of possibilities for a flexible, transparent ensemble algorithm that can serve as an alternative to model selection. They do, however, both focus on a relatively small dataset while there is a large potential for their methods on a wide variety of data. This research will try to expand upon their work by testing the method on multiple different datasets and trying to draw conclusions from it. Additionally, we will rewrite their algorithm in Python using Sci-Kit Learn to make this method more widely available and easy to copy for novice users. By these comparisons we will try to answer the following research question: What is the performance of the Optimally Weighted Ensemble compared to selection of the optimal single model? For this comparison we will compare the ensemble to the best possible method in Sci-Kit Learn, thus comparing this ensemble method to a hypothetically optimal model selection process. As a secondary research question we will try to answer the question: Is a large assigned weight determined by individual model performance?

In the second section of this paper we will define the tools used for this research and describe all the five datasets that we will evaluate the Optimally Weighted Ensemble on. In Related Work we will give an introduction to ensemble machine learning and quadratic programming while also introducing the ensemble method that we will use in more detail. In the Approach section we will expand the definitions given in the provided papers, discuss how this was rewritten in Python and give a description of the experiments that we will run. In the Results section we will discuss

the results of the experiments, while in the Conclusions section we will try to answer the research question and give recommendations for future research.
In this paper vectors will be denoted with bold text. For example, $\boldsymbol{a}_n$ is an n-dimensional vector with a_m the m-th element of this vector. Matrices are noted with capital letters, unless stated otherwise.

2 Definitions

2.1 Scikit-learn

Scikit-learn is a Python module that integrates many machine learning algorithms for medium-sized problems, both supervised and unsupervised. The module is designed with a focus on ease of use for non-specialists [14]. Among others, this ease of use is achieved by imposing a unified input/output data convention and a fixed procedure for model fitting and testing [9].
Ease of use is the main reason Scikit-learn was used for this research: as the main goal of this research is to increase the accessibility of an existing ensemble method, it makes sense to use this popular module for implementation. Additionally, Scikit-learn is fully open source and the goal of this research is to standardize the method in such a way that it can be uploaded to the project's github. Because both the fitting and testing procedure of each Scikit-learn module is unified, this research was able to look at a large number of different models with relative ease.

2.2 Datasets

For this paper we will analyse the method defined in section 3 by testing it on 5 datasets from the UCI Machine Learning Repository. The UCI Machine Learning Repository is a widely used repository that includes databases usable for analysis of machine learning algorithms. The archive was created in 1987 and has been cited over a 1000 times, meaning it is one of the 100 most cited 'papers' in computer science [10]. At the time of writing the UCI repository contains 622 datasets that can be used for testing methodologies for classification, regression, clustering or a mixture of the three [16]. As the UCI Machine Learning Repository is popular and accessible it ensures a great level of reproducability for all analyses performed in this research.

2.2.1 Abalone

In 1994 Nash et al. [12] cut open 4177 abalones (a type of marine snail) to determine their age by counting the rings on their shells (the number of rings +1.5 gives their age). This dataset includes 7 input variables (1 nominal value, sex, and 6 continuous variables) while the output variable, the number of rings on a shell, is an integer.
The Abalone dataset is the second most popular regression dataset on the UCI website [16]. It's relatively easy to implement because all its input variables except for one are numerical, while the nominal value (sex) can simply be mapped to the numbers 0, 0.5 and 1.

2.2.2 Wine Quality

The Wine Quality dataset consists of two datasets, as red wine and white wine are separate datasets. Both will be used for this research. Cortez et al. [4], the creators of the dataset, note that both datasets combine objective datapoints as input and have subjective values as output; a score between 1 and 10 given by wine experts to the different types of wine. The red wine dataset consists of 1599 instances while the white wine dataset is significantly larger with 4898 instances. All 11 input attributes are continuous numeric values.
The Wine Quality dataset is the most popular regression dataset on UCI at the time of writing [16]. Because for input it only has continuous numeric variables, and as output it has an integer, creating regression tasks is relatively low effort and this increases the reproducability of this studies. Additionally, because the only significant difference between the two datasets is the number of instances, these datasets can give us a good indication of the impact of dataset size on the performance of the Optimally Weighted Ensemble. For these reasons both datasets have been included in this research.

2.2.3 Forestfires

In the Forestfires dataset Cortez et al. [3] created a dataset that is disliked by regression models and students alike. In the description of the dataset's original paper, there is a warning that the Forestfires dataset is a very difficult regression task. The regression task associated with this dataset serves to predict the number of hectares burned in forest fires based on attributes like coordinates, date and weather data. The dataset includes 517 instances and it uses 12 input attributes, 10 of which are continuous and 2 of which are nominal.
The Forestfires dataset was included in this research because it is a famously difficult regression task: the dataset heavily skews to zero while there is an unknown number of outliers. The dataset was included in this paper for its different character: the errors of regressors tend to be very high so we are expecting a big impact when comparing model selection with model combination through ensembles. Additionally this dataset is the only 'pure' regression task in this paper: no type of classification would be possible for the requested output value as it is a continuous numeric attribute.

2.2.4 Obesity

Research by Mendoza and De la Hoz looks at the obesity levels of people from Mexico, Peru and Colombia. The dataset assigns a nominal 'Obesity Level' value to multiple people while also registering their eating habits, physical condition and general information (gender, height, weight and age). For eating habits the dataset includes data such as number of main meals and frequency of consumption of high caloric food, while for physical condition attributes such as main method of transportation and frequency of physical activity are used. The dataset includes 2111 instances, 77% of the data was synthetically generated while 23% was directly collected through a survey [13]. The dataset includes 17 attributes, 8 numeric and 9 nominal ones, of which 5 are binary nominal attributes (yes/no and Male/Female). This research will leave out one numeric attribute (weight) to increase the difficulty of the regression task while using a non-binary nominal attribute (NObeyesdad, the level of obesity) as a target attribute. All binary nominal attributes have been converted to 0's and 1's while all non-binary attributes have been converted to ranges from 0 to (the number

of options -1). While this approach makes sense for most attributes, one could critique this approach because of the attribute MTRANS (mode of transportation). These modes range from 'Automobile' to 'Walking'. While we believe this approach is sufficient for the purpose of this research, as converting these values to numerical values makes it possible to compare a wider range of regressors, this conversion of classification to numerical values is not the optimal approach for when the performance of all models needs to be maximized, as statements such as 'walking is twice as effective in preventing overweight as using a bike' cannot be made without proper research.

3 Related Work

3.1 Quadratic Programming

Quadratic Programming is a methodology to solve a quadratic cost function $f : \mathbb{R}^n -> \mathbb{R}$ defined on a subset $D \subseteq \mathbb{R}^n$ and by a set of constraints $\Omega \subseteq \mathbb{R}^n$ described by linear equalities $h_i(x) = 0$ and inequalities $g_j(x) \leq 0$. The goal of Quadratic Programming is to find a set of variables to minimize or maximize the result of the quadratic cost function f. To achieve this goal, it is necessary to express the quadratic cost function in the following form:

$$\frac{1}{2}\boldsymbol{x}^T Q\boldsymbol{x} + c^T\boldsymbol{x} \tag{1}$$

with $\boldsymbol{x}$ an n-dimensional vector that needs to be calculated to find the lowest total result possible. Solving a Quadratic Program is an intensive task for which multiple different solvers exist. A possible risk with Quadratic Programming solvers is that a minimum that can be found by a solver can be a local minimum that is not the same as a global minimum. In case a cost function is convex, any local minimum that is found is also the global minimum. Proving convexity is thus a vital part of Quadratic Programming.

3.2 Ensemble Machine Learning

Ensemble machine learning is a technique which combines multiple machine learning models in order to obtain a better performance [11]. While each individual model has its strengths and weaknesses, ensemble machine learning aims to combine multiple models in an optimal way to obtain a single prediction. In 1979 Dasarathy and Sheela [5] proposed to combine multiple classifiers to gain better results, creating the first ensemble. In 1990 Hansen and Salamon [8] proved that creating an ensemble of multiple neural network classifiers superiorly outperformed the individual classifiers.
Most methods of Ensemble Machine Learning tend to be based on relatively simple methods of ensembles (such as majority voting or simple averaging) or can be highly complex and hard to understand of the inexperienced user [6].

3.3 Weighted Averaging

Weighted Averaging is a method that has been used as a method to create ensembles for a while, but a recently proposed method expands upon it and has given new life to the research around it. Given a set of n base regressors $\mathcal{F}$ with $f_1, \ldots, f_n$ the regressors in the set, we can define a set of weights

$\Omega = \left\{\boldsymbol{\alpha} \in \mathbb{R}^n : \sum_{f\in\mathcal{F}} \alpha_f = 1, \alpha_f \geq 0,\ \forall f \in \mathcal{F}\right\}$ so that we create an ensemble $F = \sum_{f\in\mathcal{F}} \alpha_f f$. By assigning these weights to the different regressors in an optimal way, we can thus create an Optimally Weighted Ensemble that is flexible and easy to understand: the user can easily check assigned weights and conclude a sense of 'importance' based on the assigned weight. Additionally, we can look for a method that can exclude any regressors with bad performance by assigning them the weight 0.
To calculate the efficiency of an ensemble or regressor we define the loss function $\mathcal{L}$ with $\mathcal{L}_f$ the individual loss of a regressor f. To create an optimal ensemble we need to find an optimal combination of $\boldsymbol{\alpha} \in \Omega$ such that $\mathcal{L}(F)$ is as small as possible. Benítez-Peña et al. [1] define $\mathcal{L}(F)$ as

$$\mathcal{L}\left(\sum_{f\in\mathcal{F}} \alpha_f f\right) + \lambda \sum_{f\in\mathcal{F}} \alpha_f \mathcal{L}_f \tag{2}$$

where $\lambda \geq 0$ is a regularization parameter that trades off the importance given to the loss of the ensemble regressor and to the selective sparsity of the regressors used. With this we can define the main optimization problem as

$$\min_{\boldsymbol{\alpha}\in\Omega} \left\{\mathcal{L}\left(\sum_{f\in\mathcal{F}} \alpha_f f\right) + \lambda \sum_{f\in\mathcal{F}} \alpha_f \mathcal{L}_f\right\} \tag{3}$$

Given a training sample $\mathcal{I}$ with s responses where each individual $i \in \mathcal{I}$ is characterized by its feature vector $\boldsymbol{x}_i \in \mathbb{R}^p$ and its response y_i. Then, if we let $\mathcal{L}$ be the empirical loss of Ordinary Least Squares regression for $\mathcal{I}$,

$$\mathcal{L}\left(\sum_{f\in\mathcal{F}} \alpha_f\right) = \sum_{i\in\mathcal{I}} \left(y_i - \sum_{f\in\mathcal{F}} \alpha_f f(\boldsymbol{x}_i)\right)^2 \tag{4}$$

and

$$\mathcal{L}(F) = \sum_{i\in\mathcal{I}} \left(y_i - \sum_{f\in\mathcal{F}} \alpha_f f(\boldsymbol{x}_i)\right)^2 + \lambda \sum_{f\in\mathcal{F}} \left(\alpha_f \sum_{i\in\mathcal{I}} \left(y_i - f\left(\boldsymbol{x_i}\right)\right)^2\right) \tag{5}$$

3.4 Matrix Notation

To rewrite equation 5 to the shape of 1, Echtenbruck et al. [6] define the matrix $A = (a_{ij}) \in \mathbb{R}^{s\times n}$ with $a_{ij} = f_j(\boldsymbol{x}_i)$ for $j = 1, \ldots, n$ and $i = 1, \ldots, s$, a matrix where each column is an s-dimensional vector containing the predictions made by a different base regressor of $\mathcal{F}$.
We can then take $Q = A^T A$ with $c = A^T \boldsymbol{y}$ to create the matrix notation of equation 4. For this notation Echtenbruck et al. proved convexity [6]. We can thus find the global minimum (the lowest possible error of a combination of weights) through Quadratic Programming using this definition.

4 Approach

4.1 Extensions

To achieve the matrix notation of equation 5 and combine the definitons by Benítez-Peña et al. and Echtenbruck et al. we take the vector $\boldsymbol{m} \in \mathbb{R}^n$ with $m_k = \sum_{i=1}^{s} \left(y_i - f_k\left(\boldsymbol{x}_i\right)\right)^2$ and $k = 1, \ldots, n$.

We can then take $c = A^T\boldsymbol{y} + \lambda\boldsymbol{m}$ and keep $Q = A^T A$ to formulate equation 5 as

$$L(F) = \frac{1}{2}\boldsymbol{\alpha}^T \left(A^T A\right)\boldsymbol{\alpha} + \left(A^T\boldsymbol{y} + \lambda\boldsymbol{m}\right)^T\boldsymbol{\alpha} \tag{6}$$

with the following limitations:

$$h_i(\boldsymbol{\alpha}) = \mathbf{1}_n\boldsymbol{\alpha} - 1 = 0 \tag{7}$$

$$g_i(\boldsymbol{\alpha}) = I_n\boldsymbol{\alpha} \geq \mathbf{0} \tag{8}$$

with $\mathbf{1}_n$ the 1-vector of size n and I_n the identity matrix of size n. We can now use quadratic programming solvers to find the minimum set of weights $\boldsymbol{\alpha}$.

4.2 Experiment Setup

For this research we will rewrite the approach defined in 4.1 into Python using Sci-Kit Learn. As Sci-Kit Learn is one of the most popular machine learning repositories, the aim of this research is to increase the accessibility of the Convex Combinations algorithm.
For Quadratic Programming we will use the qpsolvers package [2]. This package is a flexible package that works with 16 different quadratic solvers, thus allowing for more flexibility for future use and research. The package is defined in such a way that no matter the solver you choose, you only need to enter the variables from equation 1 along with its equalities and inequalities in matrix form. For this research we have chosen to use the OSQP solver [15]. This package is library-free, free to use and the only requirement to use this package is for the cost function to be convex.
For both model selection and ensemble creation we will select a set of regressors from the Sci-Kit Learn package. To broaden the scope of the research we have chosen to not limit ourselves by selecting a limited number of regressors, but instead chose to include almost all regressors included in the Sci-Kit Learn package. Only the following regressors are excluded from this research:

- All ensembles included in Sci-Kit Learn. Since we are comparing our ensemble to model selection, the performance of these ensembles is not relevant and we do not want to include ensembles in our ensemble as that would complicate the research.
- All Multi Output Regressors are excluded because all dataset challenges are single output.
- All Multi Task Regressors are excluded since all datasets include only one task to be performed.
- QuantileRegressor is excluded, as this regressor takes significantly longer to fit than other regressors while not having a significantly lower error.
- RadiusNeighborsRegressor and SGDRegressor are both excluded because both have an extremely large error for all datasets, skewing all ensembles and making the comparisons useless.
- All regressors not converging or throwing any error when the regressors are fitted to the dataset are excluded. Because of this reason there is a small variance in the number of regressors used to build the ensemble and compare the performance with.

One strength of this study is that it expands upon previous approaches by preventing overfitting by separating the datasets into four different subsets. The training set is used to fit all individual models, while the test set is used to test the performance of both the individual models and the performance of the created ensembles. For this research an 80/20 split was made between these two datasets.
Previous experiments concluded that fitting the Optimally Weighted Ensemble on the same training set as the regressors were trained on yields worse results due to overfitting. For this reason another 80/20 split is made in the training set. The smaller training set (80%) is used to fit the individual models while the second set is used to create the ensembles and is inserted in equation 6. The individual models are trained on the bigger training set to achieve a 'fair' comparison between model selection and ensemble creation. This means that the individual models have a larger training set than when the same models are included in an ensemble.
Previous experiments came to the conclusion that a λ value of 200 in equation 5 yields the best result for the Optimally Weighted Ensemble, so this value will be used for all datasets. To evaluate the performance of the Weighted Average created through the Optimally Weighted Ensemble method, a Simple Average ensemble was created for comparison. The Simple Average ensemble takes the average of the predictions by all individual models trained on the larger training set without assigning any weights to the different predictions.

5 Results

In the Results section we will look at the average error of the Weighted Average created through the Optimally Weighted Ensemble method and Simple Average Ensemble and compare their errors to the average errors of the individual regressors included in the ensembles. For each dataset we will look at the average error on the test set. In the cases in which it is interesting to look at, we will also show the error on the training set and see which model got assigned which weight in the Weighted Average.

5.1 Abalone

For the Abalone dataset 35 different models were combined in the ensembles and compared to. This is the largest number of all datasets studied in this paper. Notably, LinearSVR and PoissonRegressor were only used in the Abalone dataset, while Abalone was only one of two dataset to exclude the MLPRegressor.
In figure 1 we can see the average error of the Weighted Average, Simple Average and the individual models on the test set of the Abalone dataset. We can see that the Weighted Average has a lower average error than any individual model on the test set. The Simple Average Ensemble has a higher average error than roughly half of the regressors.
When we look at the performance of the ensembles on the training set (figure 2) we see that for both ensembles the average error is one of the lowest, with only four regressors having a lower error than the Simple Average.
When we look at the weights assigned to the models in the Weighted Average ensembles (figure 3), we see that the model that has the highest error for both the test set and the training set (PLSCanonical) is only excluded once out of ten runs and that it is in the top half of assigned

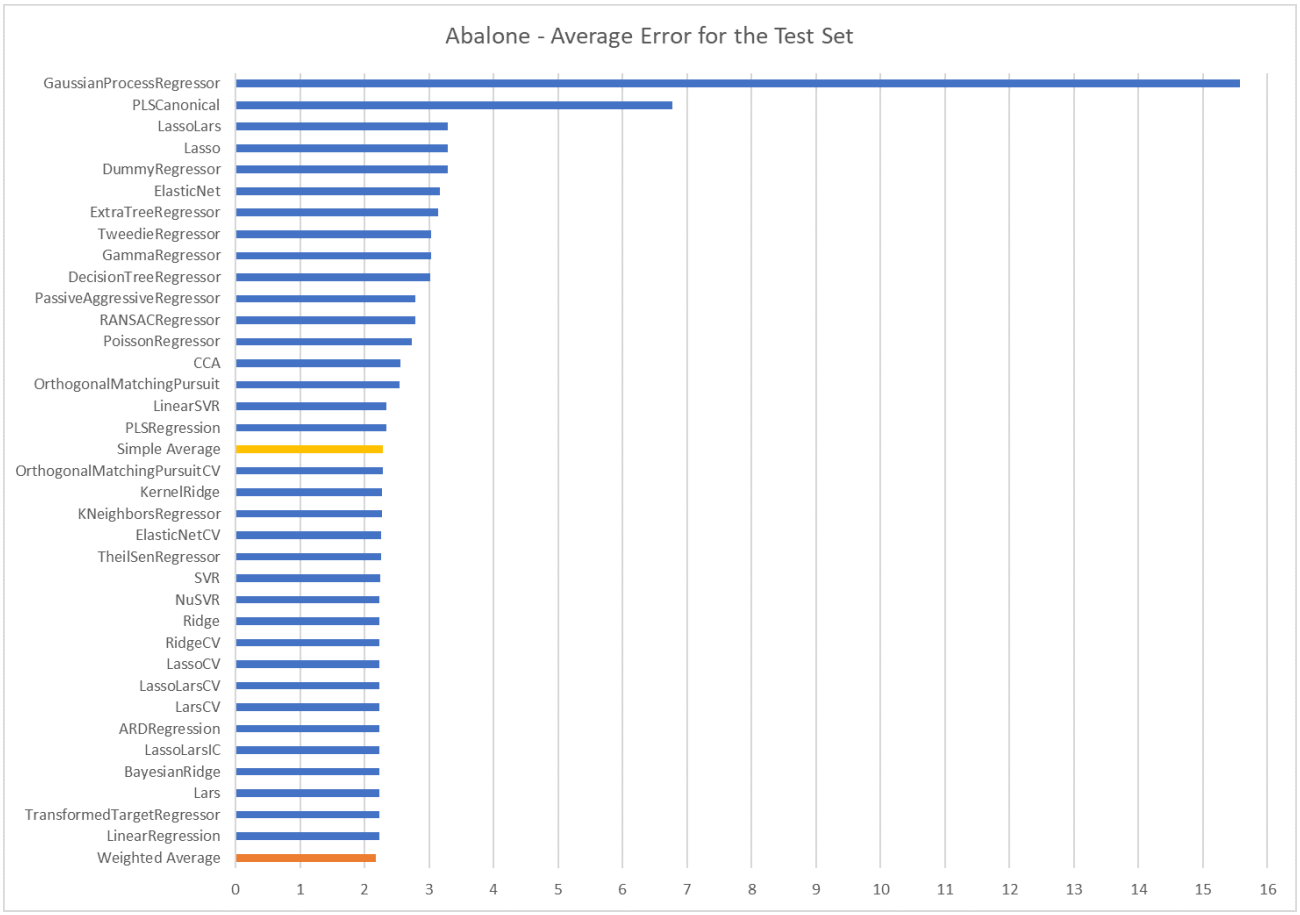


Figure 1: Average errors of different models and ensembles for the test set of the Abalone dataset.

average weights. Additionally, we can see that there is no linear relationship between the average weight assigned to a model and how often it is excluded from the ensemble: some models (such as Ridge and SVR) never get excluded, but have a very low assigned average weight, while, for example, PassiveAgressiveRegressor is excluded the third most out of all models but still has an assigned weight that is the ninth highest. Additionally, we can see that a lower individual error does not necessarily mean that the average assigned weight is higher. For example, the TheilSenRegressor has the (significantly) highest average assigned weight but falls roughly in the middle in terms of the average error on both the test set and the training set. Conversely, PassiveAgressiveRegressor has a relatively high error on the test set, but also has a relatively high average assigned weight.
To get an idea of the risk of overfitting for the Weighted Average Ensemble we can also look at the difference between predicted and actual value for both the test set and the training set. When we look at figure 4 we can see that there is no significant difference between the differences in values for the test set and the training set. This aligns with the results shown in figures 1 and 2: the average error for the training set and the test set are relatively similar.

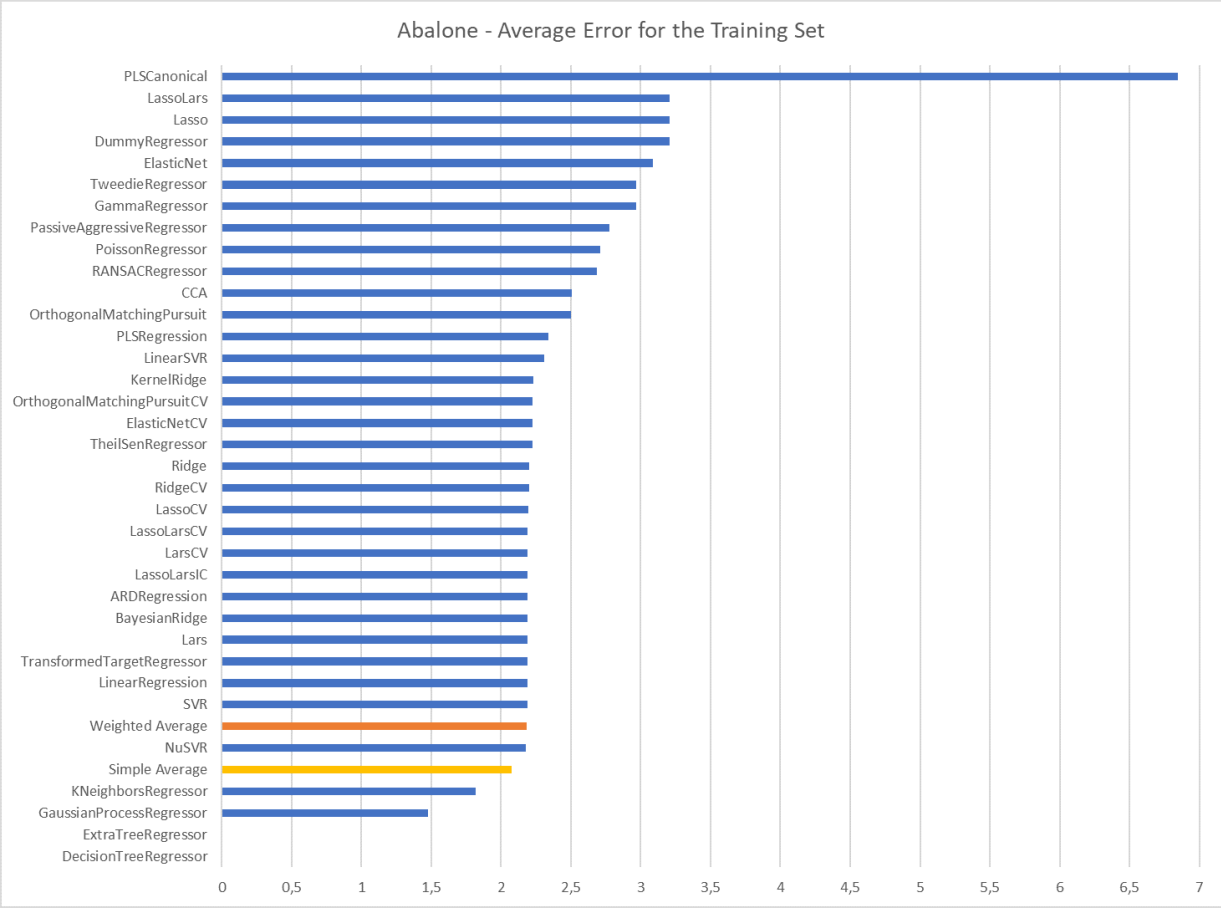


Figure 2: Average errors of different models and ensembles for the training set of the Abalone dataset.

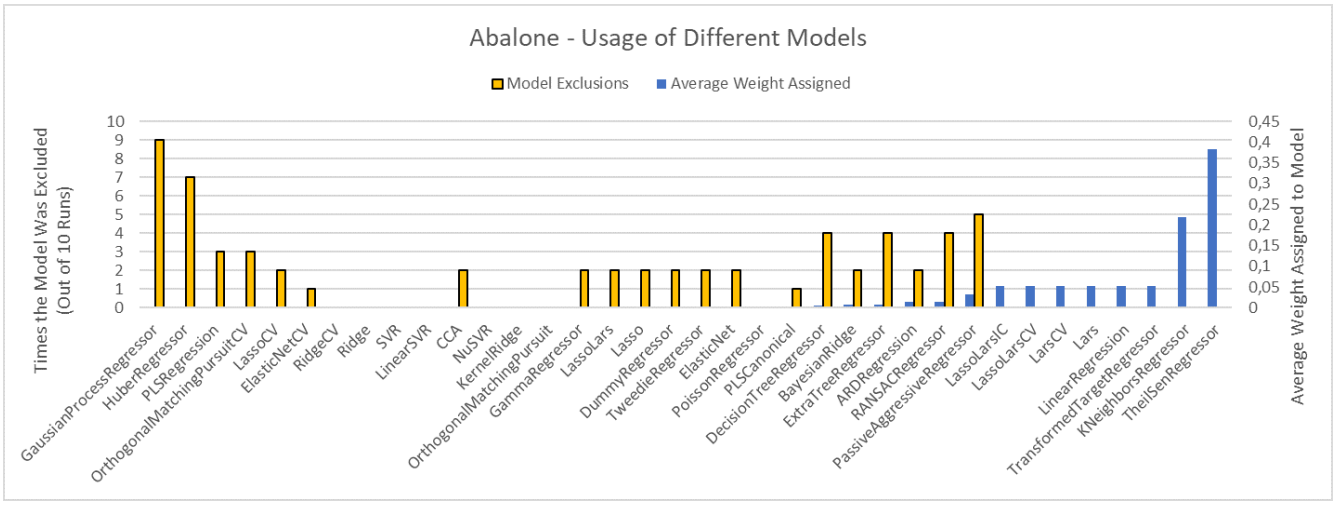
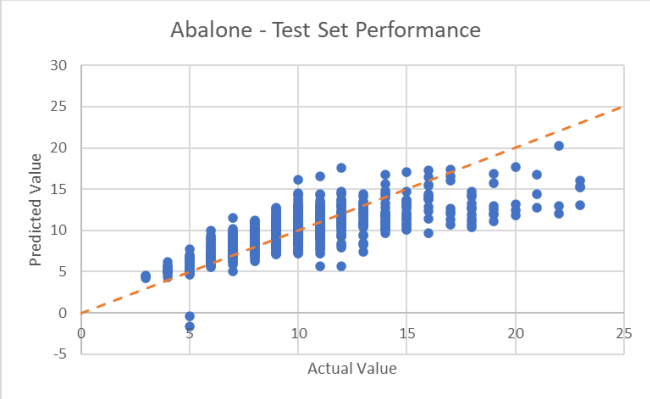


Figure 3: Comparison of the number of times a model was excluded from the ensemble with the average weight assigned to the model for the Weighted Average Ensemble on the Abalone dataset.



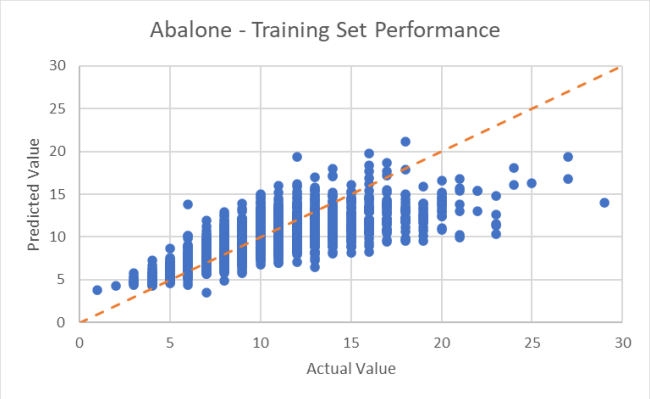


Figure 4: Comparison between the actual values and the predicted values of the Abalone dataset for the test set and the training set

5.2 Wine Quality White

For the Wine Quality White dataset 34 regressors were used to build the ensemble. The MLPRegressor is included in the ensemble for this dataset. As we can see in figure 5, for the test set the Weighted Average has the lowest average error, followed by the Simple Average ensemble. The range of different average errors between the different regressors is relatively small, with the GaussianProcessRegressor being a notable exception as it has a significantly higher average error compared to the other models and ensembles compared for this study.
When we look at the average error on the training set (see figure 6), we see that the average training errors for both ensemble methods are some of the lowest, but they are not significantly lower than in most other models. It is notable that the GaussianProcessRegressor (the model with the highest average error on the test set) has an average error of zero, along with two other regressors (ExtraTreeRegressor and DecisionTreeRegressor).
In figure 7 we can see how the ensemble has used each regressor that was included in this studies. When a higher weight is assigned to a model it can be included less often, but it is clear that there is no direct linear relationship between the two. It is interesting to note that the model with the worst performance on the test set (the GaussianProcessRegressor) was excluded only three times. It does have a near-zero assigned weight, but there are eleven models with a lower average assigned weight. Additionally, it is also interesting to note that getting a high assigned weight does not necessarily mean that the performance of the individual model is high. For example, TheilSenRegressor has the third highest average weight but has the ninth lowest error on the test set and ranks halfway in performance on the training set.

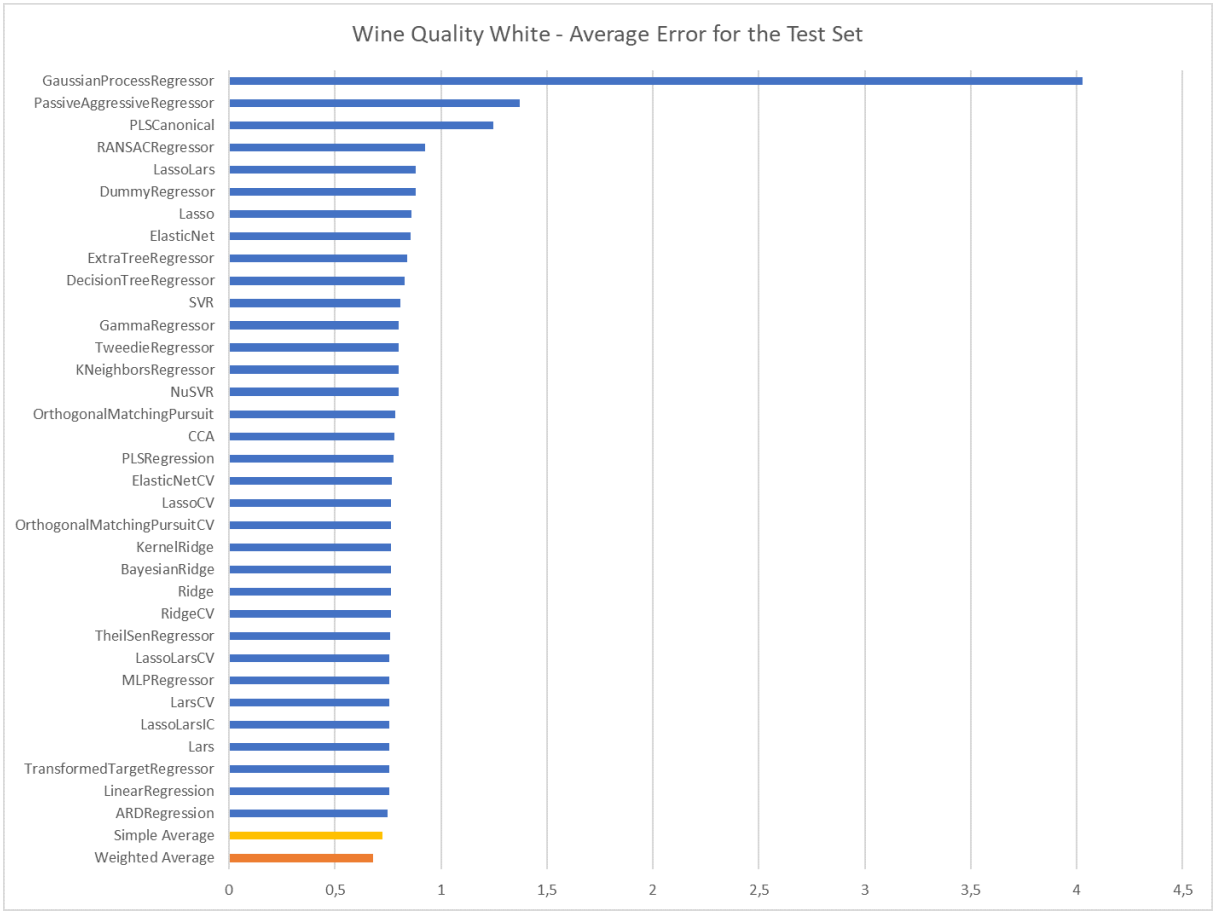


Figure 5: Average errors of different models and ensembles for the test set of the Wine Quality White dataset.

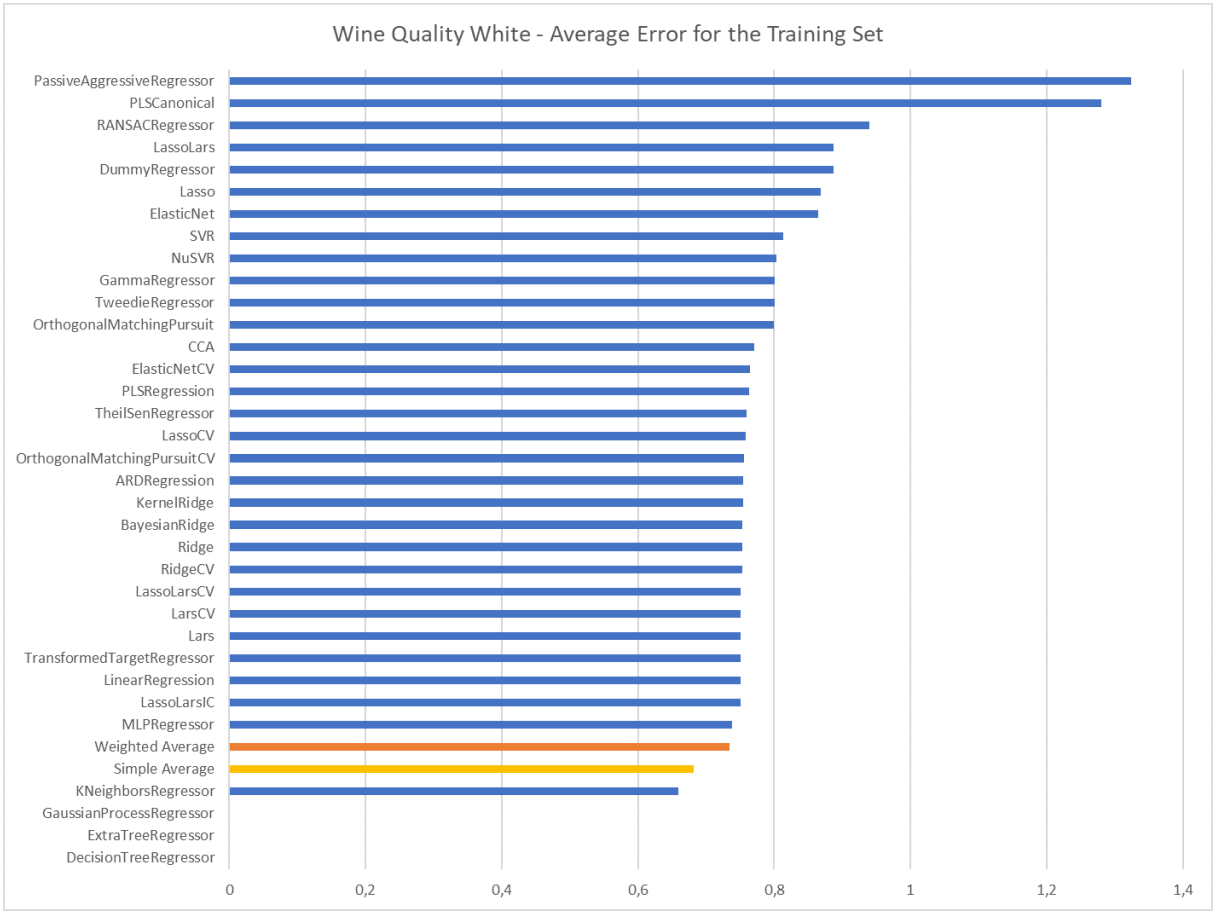


Figure 6: Average errors of different models and ensembles for the training set of the Wine Quality White dataset.

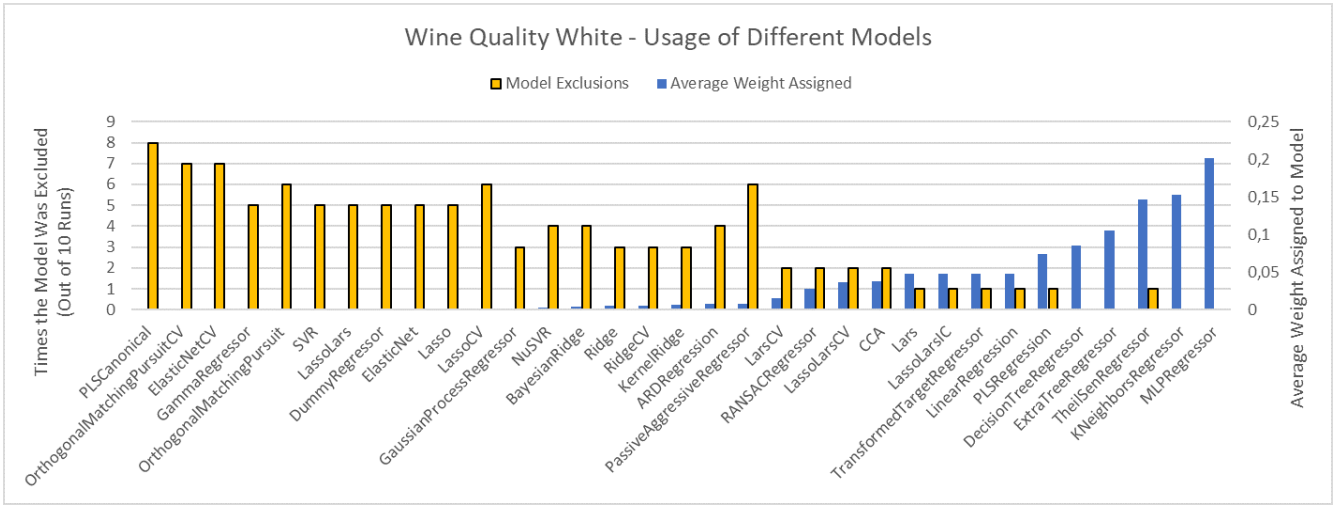


Figure 7: Average errors of different models and ensembles for the training set of the Wine Quality White dataset.

5.3 Wine Quality Red

34 regressors were included in the ensemble for the Wine Quality Red dataset, the same regressors as used in the Wine Quality White dataset. In figure 8 we can see the average error of the Weighted Average and the Simple Average on the test set of the Wine Quality Red dataset compared to the average error of the individual models. We can see that the Weighted Average does not have the lowest average error, but is one of the five models with the lowest average error. It also has a lower average error than the Simple Average method, but the differences between the 19 best-performing models and ensembles are relatively small.

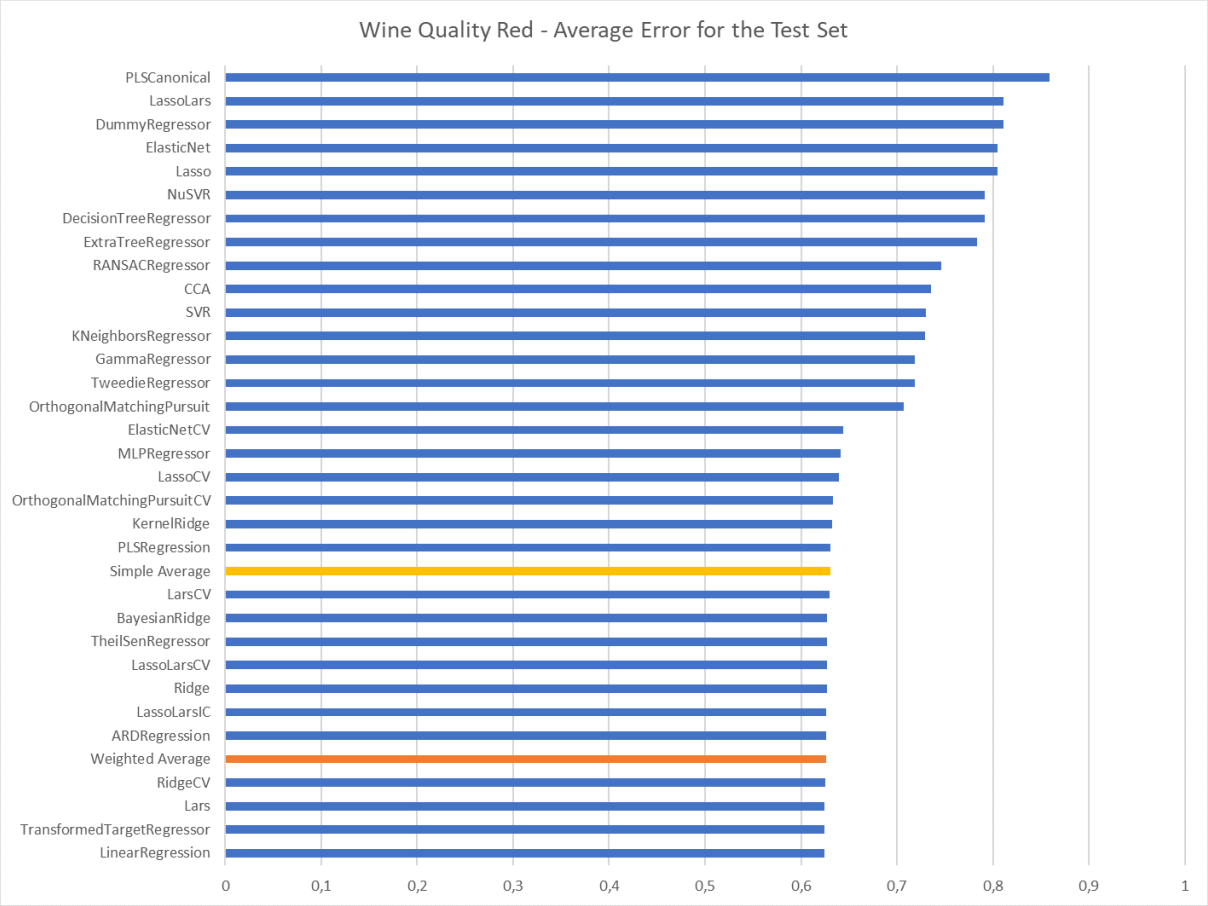


Figure 8: Average errors of different models and ensembles for the test set of the Wine Quality Red dataset.

5.4 Obesity

For the Obesity dataset 33 regressors were combined in the ensemble. For this dataset the MLPRegressor was also excluded. In figure 9 we can see that the Weighted Average has the lowest average error for the test set of the Obesity dataset, while the Simple Average ensemble has a higher error that three models. At first glance, the Simple Average seems to be significantly held back by the GaussianProcessRegressor as it has a significantly higher error than any other model. It is notable that the relative difference between the best individual model and the Weighted Average is highest of all datasets used in this research.

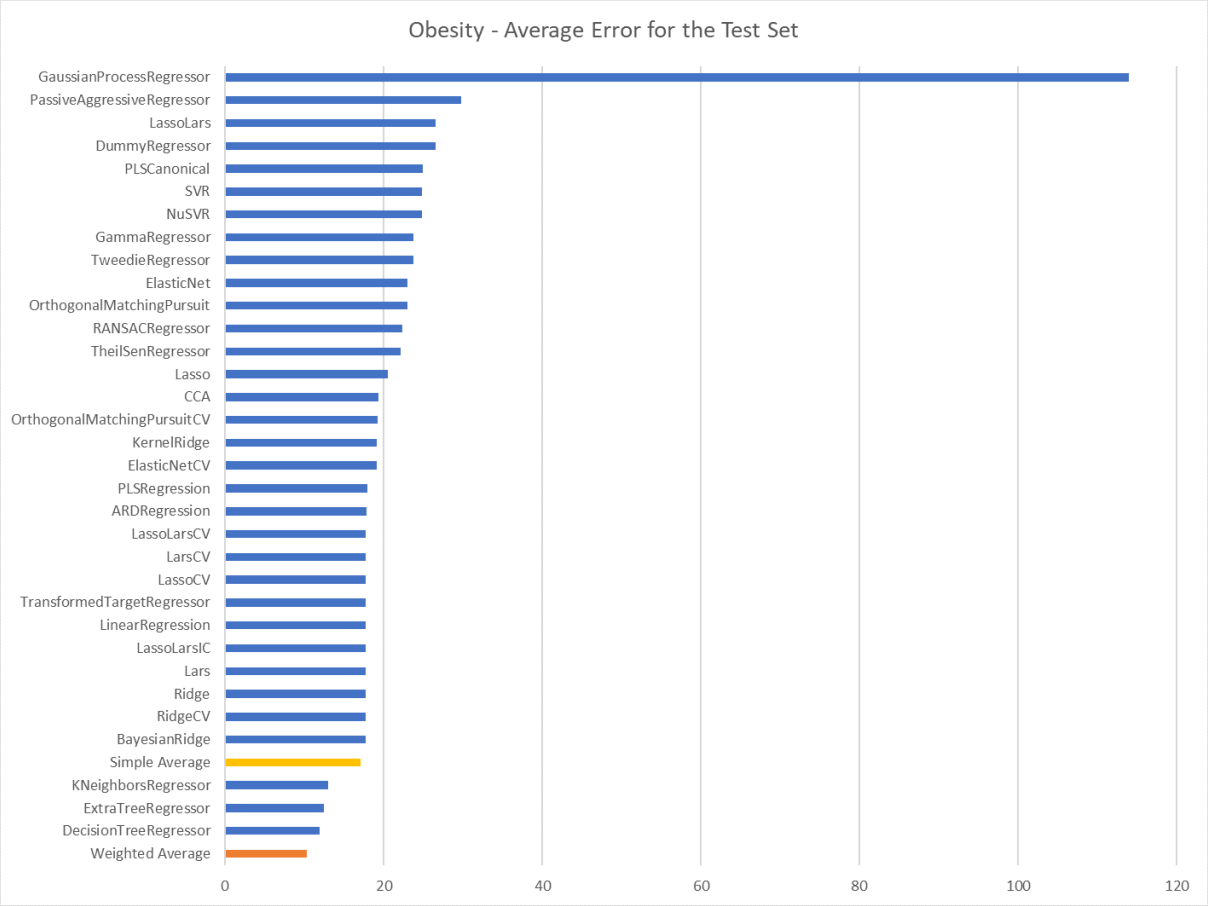


Figure 9: Average errors of different models and ensembles for the test set of the Obesity dataset.

5.5 Forestfires

For the Forestfires dataset 32 models were combined into the ensemble, the lowest number of all datasets. The regressors are the same as in the Wine Quality datasets, except for the exclusion of the GammaRegressor and the TweedieRegressor. In the Forestfires dataset (figure 10 we can see that the Weighted Average has one of the highest errors for the test set. The Simple Average has the lowest average error apart from one model. It is important to note, however, that the differences are relatively small. The graph in figure 10 does not start at zero for readability purposes.
When we look at the performance of the ensembles for the training set of the Forestfires dataset, we can see a similar story in figure 11. The Weighted Average ensemble has the second highest error, while the Simple Average ensemble has the fifth lowest average error. The relative difference in errors between the training set and test set is the highest of this study: the errors in the training set are around 45 while the errors in the test set are around 110.

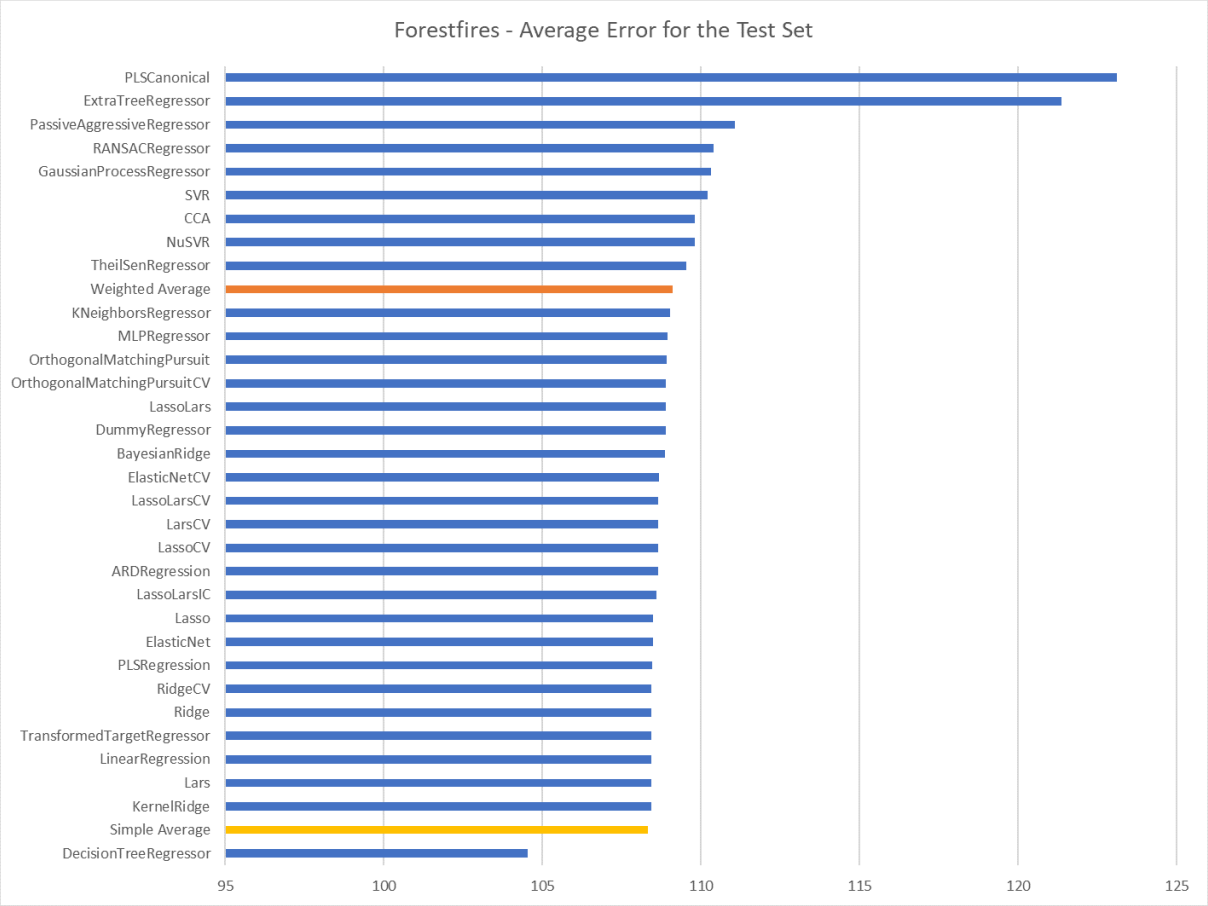


Figure 10: Average errors of different models and ensembles for the test set of the Forestfires dataset. Important note: the x-axis starts at 95.

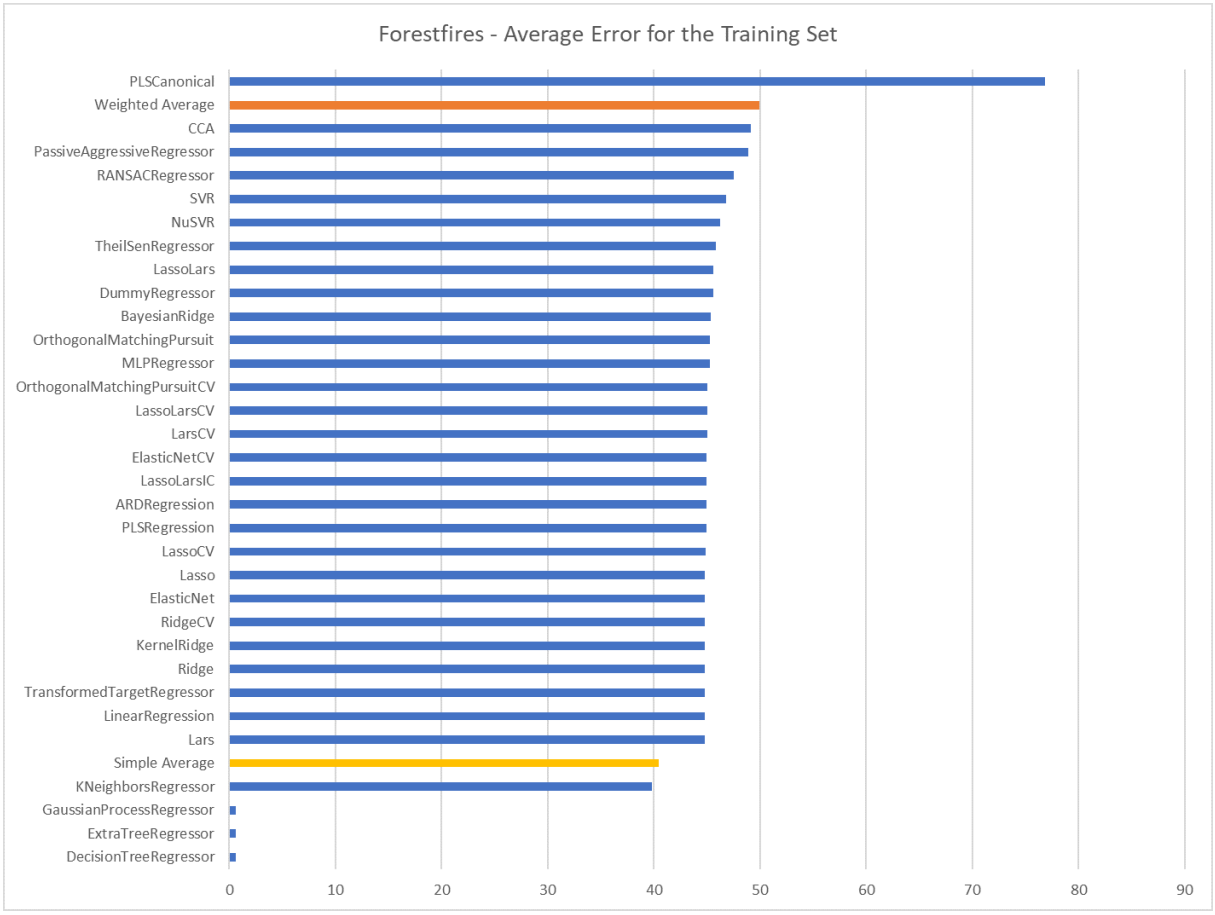
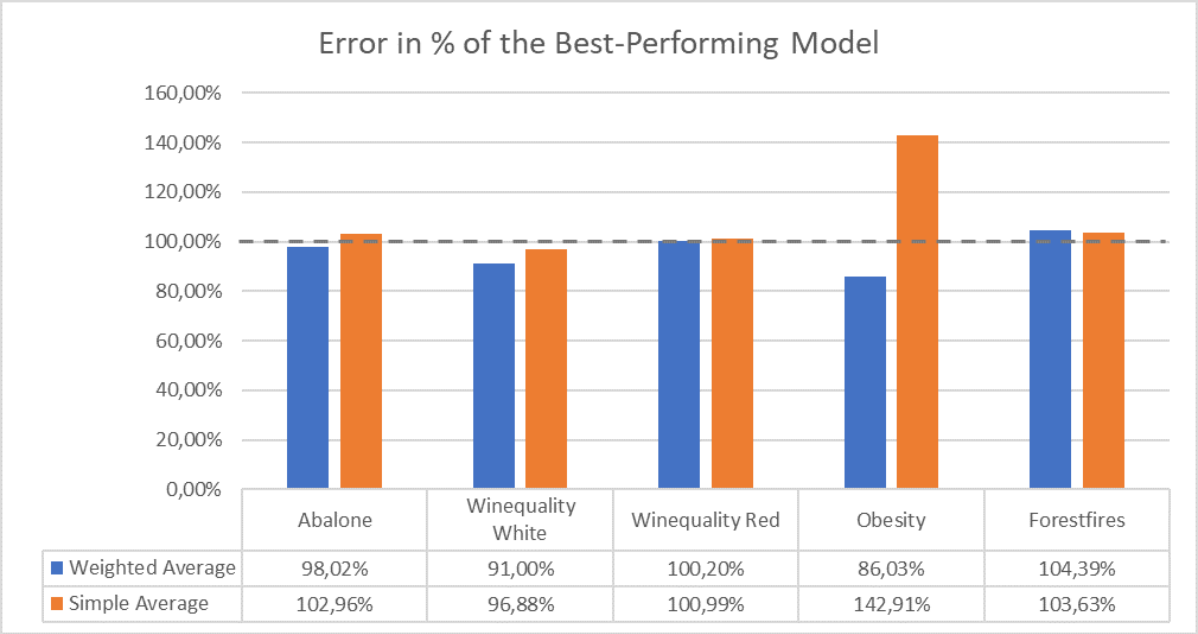


Figure 11: Average errors of different models and ensembles for the training set of the Forestfires dataset.

5.6 Overview

In figure 12 we can see a complete overview of the relative performance of the different models compared to the best possible model. We can see that for three of the datasets (Abalone, Wine Quality White and Obesity) the Weighted Average has a lower error than the ensemble. For the other two datasets (Wine Quality Red and Forestfires) the error is higher, but the relative difference is very small. It is important to note that for this graph the best possible model is selected directly by taking the model with the lowest error on the test set, a model selection method cannot be expected to select the best possible model 100% of the time.
For the Simple Average ensemble it is almost always outperformed by the Weighted Average. In the two cases that it has a (slightly) lower error than the Weighted Average, it still has a higher error than the best-performing model.



	Abalone	Winequality White	Winequality Red	Obesity	Forestfires
Weighted Average	98,02%	91,00%	100,20%	86,03%	104,39%
Simple Average	102,96%	96,88%	100,99%	142,91%	103,63%

Figure 12: Overview of average error on the test set of the different models as a percentage of the error of the best-performing model.

6 Conclusions and Future Research

A big risk for machine learning is the risk of overfitting. Before we evaluate the performance of the ensembles we should ensure that this is not the case. When we look at the scatterplots in figure 4, we can see that there is no significant difference between the performance on the test set and training set, so we can conclude that the risk of overfitting for the Optimally Weighted Ensembles is relatively low. This conclusion is strengthened when we note that overall performance on the training sets is comparable to that of other individual models (see figures 2, 6 and 11).
When we look at the complete overview (figure 12), we can see that the Optimally Weighted Ensembles outperform the best model selection for all but two datasets, with the difference between

the hypothetically best model and the best Weighted Average being negligible in the cases where the ensemble is outperformed. We can thus conclude that model combination yields better results than or results similar to model selection, even when selecting the best-performing individual model through a hypothetically optimal model selection process.
For the Forestfires database the Optimally Weighted Ensemble was outperformed by most models and even by the Simple Average Ensemble. When compared to the other datasets used in this study, the main difference between them and the Forestfires dataset is that the latter is heavily skewed with an unknown number of outliers. We can thus conclude that the Optimally Weighted Ensemble is outperformed by model selection when the dataset is heavily skewed.
On the Wine Quality Red dataset the Optimally Weighted Ensemble is also (slightly) outperformed by the best-performing model. The only difference between Wine Quality White and Red is the number of instances. For this reason, it is clear that dataset size has a large impact on the performance of the Optimally Weighted Ensemble. Additionally, the Forestfires dataset has an even lower number of instances than any other dataset. In the case of the Forestfires dataset this is not surprising, as the train/test split used in this research means that the weights were calculated using only 82 data points. We can thus conclude that the Optimally Weighted Ensemble benefits from a larger dataset and for small datasets model selection might be a more optimal solution than model combination.
When we look at the Simple Average ensemble we can see that it is almost always outperformed by the Optimally Weighted Ensemble (except for the Forestfires dataset, see above). At the same time the Simple Average only once outperforms the best-performing model. We can thus conclude that simple averaging does not yield the same performance as the Optimally Weighted Ensembles and, even though ensembles outperform model selection, we can also conclude that a simple method like averaging results usually does not yield better results than model selection.
When we look at the distribution of weights in the usage graphs we can conclude that there is no direct linear relationship between the performance of an individual model and the weight the ensemble assigns to it. While 'better' models are included more often and get a higher weight, we can conclude that including models with a lower individual performance can yield better results. It is thus an interesting challenge to find out in advance which models to exclude from the ensemble.
When we look back at the previously posed research questions we can conclude that in general the Optimally Weighted Ensemble outperforms model selection, but factors like dataset size and skew can impact results. Additionally, we can answer that a large assigned weight is not determined by individual performance, but is clearly influenced by it.

6.1 Future Research

For future research this study can be extended to compare the performance of the Optimally Weighted Ensemble to other ensembles, both the ones included in SciKit Learn and ones outside of its scope. Additionally, incorporating parameter optimization of the individual models could yield interesting results: does a better performance of the individual models lead to better performance of model combinations? While we would generally assume that this is the case, the question of model selection vs model combinations might be answered differently.
A big challenge for the Optimally Weighted Ensembles method is the time it takes to fit the ensembles. Reducing the number of models used in the ensemble would reduce calculation time but could cause a drop in performance. We propose a follow-up study to look for a way to exclude

models based on their individual performance before including them in an ensemble. This might lead to interesting results as this research has shown that this is not directly linked to the performance of the individual model.

References

[1] Sandra Benítez-Peña, Emilio Carrizosa, Vanesa Guerrero, M. Dolores Jiménez-Gamero, Belén Martín-Barragán, Cristina Molero-Río, Pepa Ramírez-Cobo, Dolores Romero Morales, and M. Remedios Sillero-Denamiel. On sparse ensemble methods: An application to short-term predictions of the evolution of covid-19. *European Journal of Operational Research*, 295(2): 648–663, 2021. ISSN 0377-2217. doi:10.1016/j.ejor.2021.04.016.

[2] Stéphane Caron, Daniel Arnström, Suraj Bonagiri, Antoine Dechaume, Nikolai Flowers, Adam Heins, Takuma Ishikawa, Dustin Kenefake, Giacomo Mazzamuto, Donato Meoli, Brendan O'Donoghue, Adam A. Oppenheimer, Abhishek Pandala, Juan José Quiroz Omaña, Nikitas Rontsis, Paarth Shah, Samuel St-Jean, Nicola Vitucci, Soeren Wolfers, @bdelhaisse, @MeindertHH, @rimaddo, @urob, and @shaoanlu. qpsolvers: Quadratic Programming Solvers in Python, April 2023. URL https://github.com/qpsolvers/qpsolvers.

[3] Paulo Cortez, Aníbal Morais, José Neves, and Manuel Filipe Santos. *A Data Mining Approach to Predict Forest Fires using Meteorological Data*, page 512–523. APPIA, 2007. ISBN 13 978-989-95618-0-9. URL http://www3.dsi.uminho.pt/pcortez/fires.pdf.

[4] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009. ISSN 0167-9236. doi:10.1016/j.dss.2009.05.016. Smart Business Networks: Concepts and Empirical Evidence.

[5] B.V. Dasarathy and B.V. Sheela. A composite classifier system design: Concepts and methodology. *Proceedings of the IEEE*, 67(5):708–713, 1979. doi:10.1109/PROC.1979.11321.

[6] Patrick Echtenbruck, Martina Echtenbruck, Joost Batenburg, Thomas Bäck, Boris Naujoks, and Michael Emmerich. Optimally weighted ensembles of regression models: Exact weight optimization and applications. 2022. doi:10.48550/arXiv.2206.11263.

[7] Martina Friese, Thomas Bartz-Beielstein, Thomas Bäck, Boris Naujoks, and Michael Emmerich. Weighted ensembles in model-based global optimization. *AIP Conference Proceedings*, 2070(1): 020003, 02 2019. ISSN 0094-243X. doi:10.1063/1.5089970.

[8] L.K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990. doi:10.1109/34.58871.

[9] Jiangang Hao and Tin Kam Ho. Machine learning made easy: A review of scikit-learn package in python programming language. *Journal of Educational and Behavioral Statistics*, 44(3): 348–361, 2019. doi:10.3102/1076998619832248.

[10] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The UCI machine learning repository. URL https://archive.ics.uci.edu. Accessed: 2023-06-11.

[11] Ibomoiye Domor Mienye and Yanxia Sun. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access*, 10:99129–99149, 2022. doi:10.1109/ACCESS.2022.3207287.

[12] Warwick Nash, Tracy Sellers, Simon Talbot, Andrew Cawthorn, and Ford Wes. Abalone. UCI Machine Learning Repository, 1995. doi:10.24432/C55C7W.

[13] Fabio Mendoza Palechor and Alexis de la Hoz Manotas. Dataset for estimation of obesity levels based on eating habits and physical condition in individuals from Colombia, Peru and Mexico. *Data in Brief*, 25:104344, 2019. ISSN 2352-3409. doi:10.1016/j.dib.2019.104344.

[14] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bartrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. URL https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf.

[15] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, feb 2020. doi:10.1007/s12532-020-00179-2.

[16] UCI Machine Learning Repository: Data Sets. UCI machine learning repository: Data sets. URL https://archive.ics.uci.edu/datasets. Accessed: 2023-06-11.