FEATURE SELECTION METHODS WITH APPLICATIONS IN ELECTRICAL LOAD FORECASTING

OSCAR UTTERBÄCK

LUND
UNIVERSITY

Master Thesis

Centre for Mathematical Sciences
Faculty of Engineering
Lund University

Project performed under the supervision of
Anders Heyden, LTH
Pawel Herman, KTH
Mattias Jonsson, Expektra

ABSTRACT

The purpose of this thesis is two-fold: implement and evaluate a method, the Fast Correlation-Based Filter (FCBF) by Yu et al. [33], for feature selection applied on a meteorological data set consisting of 19 weather variables from 606 locations in Scandinavia, and investigate whether geography can be exploited in the search for relevant features. Four areas are chosen as target areas where load prediction error is evaluated as a measure of goodness. A subset of the total data set is used to lower the computation time; only Swedish locations were used, and only data from SMHI was used.

The impact of using different subsets of weather features as well as selecting features from several locations is investigated using FCBF and ϵ-Support Vector Regression. A modification to the FCBF algorithm is tested in one of the experiments, using Pearson correlation in place of symmetrical uncertainty. An investigation of how the relationships between features change with distance is performed and the results are then used to motivate a greedy feature selection method.

FCBF, even when implemented with the naive approximation of marginal and conditional entropy, filtered the total data set from 3180 to approximately 20 features with a prediction error of less than 1% for three of the target areas and 1.71% for the fourth. Further tests lowered the numbered of features even further without significantly affecting the prediction error. Using FCBF to rank the weather variables for a single area proved less than optimal which may be attributed to many of the extremely small intra-feature SU values. Selecting locations based on distance from target area resulted in prediction errors better than random sampling and comparable to the filter while still keeping the number of features low.

The very best feature selection results were only slightly lower than a base case, suggesting that the present experimental setting may not be enough to draw definitive conclusions regarding the efficacy of the selection methods. Two possible contributing factors are the unoptimized model used, and the choice to investigate the impact on average load over a 24 hour window. Future studies may also wish to extend the geographical investigation to use coordinates or direction in conjunction with distance from the target area, as some indication of latitude dependent behavior was found, most likely contributed by the elongated shape of Sweden.

ACKNOWLEDGMENTS

I want to thank everyone who has had a hand, knowing or unknowing, in me reaching this far. To my excellent supervisors, Pawel and Mattias, for all their insight, pedagogy, patience and support. It has been an absolute pleasure working with you. Adam, for being an awesome roommate during my time in Stockholm. Jakob and Hilda for their immense moral support. To my family, Per, Ingela, Sophia and Niklas, for cheering me on all the way. To Claus, without whom I wouldn't have done it.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 BACKGROUND AND PROBLEM FORMULATION

Forecasting near-future electrical load and production is an important part of the energy industry's daily processes. The nature of electricity and its extremely wide range of uses in modern society makes it a product that is difficult to store and difficult to predict the demand of. In the early days of electricity the grids had relatively few usage areas, few providers as well as little geographical reach. As innovation flourished throughout the 20th century the number of applications grew; factories exchanging steam-run engines with electrical ones, infrastructure making use of electrical access, and consumer appliances becoming more common, all contribute to the growing complexity of energy demand, and subsequently load prediction. Other factors accounting for this is the deregulation of energy markets leading to new vendors in the market, as well as globalization which enabled international trade of energy [12].

There are a number of reasons behind the desire to achieve accurate load forecasts. Most prominent is the need to ensure that the equipment in use is able to handle the load on the electrical grid in order to maintain function within given specification as well as to avoid outages [12]. Of importance is also the economical factor, where for example brokers on the energy market frequently need to determine the spot price of electricity as energy in various forms are traded between producers, providers and utility companies [6]. As such, operators in the energy industry rely on forecasting methods, the efficiency and accuracy of which has a direct effect on operating expenses [12].

Load forecasting is divided into a number of categories depending on the lead time of the forecast. Although there is no formal definition, nor are the delimitations completely agreed upon, there are four divisions commonly found in literature: very short term load forecasting (VSTLF), typically entailing forecasting less than 24 hours ahead, short term load forecasting (STLF), applicable for day-ahead forecasting up to about a week, medium term load forecasting (MTLF) denoting forecasts of up to a year into the future, and long term load forecasting (LTLF), covering time spans longer than a year [10]. The lead time is chosen based on the usage area; VSTLF and STLF carries importance when planning day-to-day operations as well

as in spot price determination, whereas M/LTLF may be involved in long time planning of contracts, customer capacity determination, grid expansion and machinery maintenance[10].

Electric load has long been known to correlate with factors such as weather, social events and season on a daily, weekly and yearly basis [10][30]. The information used when performing load prediction depends strongly on the lead time of interest. In (V)STLF it is argued that social and economic variations may be of little importance[10], while the daily air temperature has a very high correlation with load even at an hourly resolution. The state of the weather is generally considered to be the most important factor, but both climatic and regional influences as well as consumer behavior play an important role [8].

Connecting these variables to load forecasting is done using a number of different approaches. Forecast models based on regression analysis as well as statistical time series analysis are typical approaches with historical and present use in the industry[10]. Another approach uses numerical simulation methods to predict the load via the modelling of a dynamical system. Vitec's system Aiolos is one example of a software package that provides load forecasting via a combination of statistical analysis and dynamical modelling using weather as the primary component [30].

In the past two decades computational methods based on machine learning and artificial intelligence have begun increasing in popularity[31] and subsequently made their way into the forecasting market. There are numerous reports detailing success in applying different machine learning methods to problems in time series forecasting, both in energy load prediction and in other areas [2]. In [5], [8], [23], and [24] support vector machines and artificial neural networks are used as models for load prediction and compared with traditional regression methods. The possibility for these methods to model complex non-linear interactions between points in the data set is consistently highlighted as a strength. There has been interest in applying different pre-processing methods, different feature selection methods and different optimization routines, and research is still on-going.

Expektra AB, with whom this thesis is done in collaboration, is one company that offers data analysis with applications within the energy market. One of their services, Expektra Predict[7], aims to analyze large amounts of data to perform short term load prediction for areas around Scandinavia using sophisticated machine learning methods based on artificial neural networks. The data set used includes weather predictions for variables such as temperature, wind speed and humidity from the meteorological institutes SMHI[28] and Yr[32], measured observations of the weather, as well as historical load data.

The data set consists of a number of locations, each with their own data series containing the weather predictions, observations and load. The locations are identified

by a set of coordinates related to areas for which the load information is collected and each location is specified by corresponding utility companies responsible for the electrical grids. Since weather is a localized phenomenon and the density of grid areas within different regions is irregular, it is hypothesized that there will be various grades of redundancy in the weather data set as some of these grid areas are closer together, resulting in less spatial variation for the weather to differ.

Expektra has previously determined[14] that using multiple data series describing a single variable, e.g. temperature, from different locations may improve load prediction accuracy. However, the large amount of locations makes it unfeasible to use every single one as the computational complexity grows as more data is added. Additionally, redundancy may render the increase in accuracy negligible or statistically insignificant and irrelevancy may have a negative impact on the prediction results.

Performing feature selection, i.e. determining a combination of input variables that optimally increase the accuracy while keeping the number of inputs small, is an ongoing research area within machine learning and data analysis. While such methods are applied within Expektra Predict, it is of interest to perform a study on the weather data to investigate how redundancy can be identified in the set, how it varies geographically and whether geographical properties can be used as a way of identifying valuable sources of information.

Within this thesis project, a number of delimitations will be set within which the data set will be analyzed with a few goals in mind. The analysis will consist in investigating whether there is some level of saturation with regards to information describing a single variable, identifying redundancy between data series as well as identifying relevancy in terms of increasing prediction accuracy. Central will be the geographical aspect, i.e. the relationship between the locations of the data series and whether or not it can be exploited in determining how to optimally identify useful information for predicting short term electrical load.

The main goal this thesis will work towards is determining the impact that the different weather variables have on predicting the load, both from the geographical aspect and the different weather descriptors. The work will center around two research questions:

- How can the data set be used to efficiently minimize prediction error?
- Can geography be exploited to find good feature subsets?

1.2 OUTLINE

Chapter 2 begins by introducing the reader to literature describing the state of the art of methods used within machine learning and feature selection for electrical load forecasting. Following is a section on data mining and preprocessing, discussing different methods for extracting information and enhancing the connection between weather and load prediction. An explanation of the different methods and measures used to quantify redundancy and relevancy is given, and how they relate to prediction accuracy in the context of machine learning. Finally, support vector machines are presented together with a description of model evaluation.

Chapter 3 begins by discussing the data set and the delimitations to the project are introduced. The experiments performed and the motivation behind them will be laid out along with a description of the machine learning model used for evaluation.

In chapter 4 the experiments are described in detail along with their results. The chapter starts with a number of baseline experiments used for comparison together with a statistical analysis of the data set. This is followed by experiments making use of geography and FCBF with the intent of maximizing prediction accuracy.

Finally, chapter 5 will present some conclusive thoughts together with suggestions for future work.

2

THEORETICAL BACKGROUND AND RELATED WORK

2.1 SHORT TERM ELECTRIC LOAD FORECASTING

> Perhaps the most challenging aspect of the operation of power systems is their ability to meet load requirements instantaneously and at all times. This is not the result of a magic trick. And it is certainly not true that energy travels from the generating station to the load that is situated hundreds of kilometers away at the blink of an eye. What happens is rather a transaction between the generator and the load. The load requests electric power which is duly supplied to it. However, this extra power requirement cause the generator to slow down, thus reducing the system frequency. Automatic generation control (AGC), a feedback control mechanism, senses the frequency drop and opens the valve to allow more steam to flow through the turbine and thus increase the speed to its nominal value. This procedure completes the cycle and more electric power is supplied at the same frequency.
>
> Kyriakides & Polycarpou: Short Term Electric Load Forecasting: A Tutorial

Machine learning methods for short term load forecasting has been gaining in popularity since the beginning of the 90's. The desire for more accurate forecasting techniques has for the last twenty or so years been driven in large by the expansion of the electric markets with the addition of new vendors, deregulations as well as globalization. Energy is traded not only between different suppliers, utilities and providers, but also between nations and it has moved from being a local commodity to a global phenomenon that is traded with in the same manner as physical goods or stocks[17].

STLF falls under the category of supervised learning and involves the prediction of load demand within the span of 24 hours and up to a week into the future with both financial as well as technological applications. Supervised learning is the problem of predicting new values from unseen data points given a set of input and target data. In load forecasting the known target data is historically recorded load.

2.2 MACHINE LEARNING IN LOAD FORECASTING

Machine learning in the context of load forecasting has been an active research area since the early 90's. Machine learning in itself is a booming research field that continues to grow as computational power becomes more prevalent in society with load forecasting being but one of the many areas where new modeling approaches based on computational intelligence methods shows promise. There is an extensive library of methods available for performing prediction in regression problems and research in load forecasting applications grows with it.

A large number of studies show promising results in applying machine learning methods for load forecasting compared with traditional time series and regression methods. While it is nowhere near conclusive the best choice of method or model, actors in the industry have begun applying machine learning to load forecasting with successful results.

In 1991 Park et al.[24] suggested using an artificial neural network for performing one-hour and 24-hour ahead forecasts of energy load over the period of 1/11/1988 to 30/1/1989 using temperature data. They employed a network with one hidden layer consisting of five nodes and attempted to forecast the peak load and total load, and a ten-node network to forecast the hourly load. In the prior two cases the minimum, average and maximum temperature were used as input data, whereas the hourly forecast used the hourly temperature as well as the load data for the past two hours. Noteworthy is that the authors divide the data set into five sets of one week each where one day is left out for testing, and subsequently six days are used for training, leaving the data set rather small. The resulting predictions show an increase in performance, from 4% error to approximately 2%, when compared to the predictions of Puget Sound Power and Light Co., who supplied the data.

McMenamin and Monforte[23] published a study on short term forecasting with neural networks in 1998. Their data set includes weather and load information over all of 1993 for an electric utility in the southwest U.S. This study uses not only weather information and historical load data but also a number of variables describing the calendar. Weekday, season, holidays as well as sunrise and sunset were all identified as variables affecting the load demand of the following day. Inspection of the data leads them to conclude that temperature differences affect the load demand in different ways depending on weekday and season. Interactions between the weather variables may also change as a result of the season; a higher wind speed implies a chilling effect which may lead to reduced usage of cooling equipment in summertime whereas heating equipment is turned up in the wintertime. Neural networks were argued to be a suitable model for forecasting that is able to capture complex non-linear dependencies between the included variables and the load data.

In 2001 Chen et al. applied support vector machines to forecast mid-term load, up to 31 days into the future, in a competition which yielded them the winning position [5]. Their competitors used a variety of models, including not only classical time-series approaches but also neural networks, self-organizing maps and fuzzy expert systems.

One study on performing hourly short term load forecasts in Portugal using a number of regression techniques was published in 2007 [8]. K-Nearest Neighbors, Multiple Linear Regression, Regression Trees, Neural Networks and a number of others were in a preliminary step ranked against each other where artificial neural networks stood out as the most efficient. Forecasting is then performed using a combination of load from the past two weeks and a week-day variable, resulting in a mean absolute percentage error of 1.5% to approximately 2%. Subsequent analysis investigates the effect of temperature as an input variable with the conclusion that, despite its prevalence as one of the most important variables in load forecasting, no advantage is provided from it in the present case. Possible explanations include Portugal's temperate climate, the inclusion of temperature information in the load series as well as the quality of the temperature data itself.

A comparative study of machine learning models for time series forecasting was done in 2010 by Ahmed et al.[2] in which eight popular machine learning methods for preprocessing and forecasting were compared: multilayer perceptrons, bayesian neural networks, radial basis function neural networks, generalized regression neural networks, K nearest neighbor regression, classification and regression trees, support vector machines, and gaussian processes. The study was not done with the explicit goal of performing STLF but for time series prediction on business data. The data set used, part of the monthly M3 time series competition data[20], consists of 1045 time series with 81-126 data points each from different econometric areas. The last 18 points of each series was left out of model training, and for each of the models one-step-ahead forecasting was performed on these 18 points. The models were ranked in three different categories based on the preprocessing methods used: No preprocessing, one-step backward differencing and moving averages. It is concluded that there is an unambiguous ranking that suggests that certain models are more suited than others for time series forecasting with the caveat that this applies to business-type data.

2.3 MACHINE LEARNING METHODOLOGY

2.3.1 *Terminology*

Terminology in machine learning literature can sometimes be used haphazardly and may lend itself to some confusion. Several definitions are available depending on

author and context but the following definitions will be used in this report.

Definition 1. *A **variable** is a descriptor from the original data set, e.g. temperature, humidity, load.*

Definition 2. *A **model** is a class of learning algorithms. In the present case of regression it is a function $f(\boldsymbol{x};\boldsymbol{\theta}) : \mathbb{R}^m \to \mathbb{R}$ that has undergone learning from the data set by optimization of the parameters θ via a cost function.*

Examples of models are artificial neural networks (ANN's), support vector machines (SVM's), or linear regression.

Definition 3. *A **feature** is an input variable in the context of machine learning. It is used as an input to the model and may have undergone preprocessing.*

Definition 4. *The **target** is the value sought to be predicted by the model.*

Features can be combinations of variables, a function of one or more variables, or encoded meta-data such as time of day or day of the week. The original variables may of course be used as features.

The data used in machine learning can be structured or unstructured. In the prior case it is commonly represented in matrix form. Denoting the input data as $X \in \mathbb{R}^{n\times m}$, the dimension n refers to the number of data points and m refers to the number of variables. Each row i of X is a vector $\boldsymbol{x}_i \in \mathbb{R}^m$ which thus constitutes part of a data point of m different variables. The target corresponding to $\boldsymbol{x}_i$ is typically denoted $y_i \in \mathbb{R}$.

Definition 5. *A **data point** $\{\boldsymbol{x}_i, y_i\} \in \mathbb{R}^{m+1}$ is a sample from the data set where each component in $\boldsymbol{x}_i$ describes a different feature.*

2.3.2 *Model evaluation*

In order to evaluate the performance of a machine learning model it is imperative to be able to quantify how the model generalizes from the learned data set to unseen data. Working with a finite data set it is customary to split the data set into two disjoint sets: a training set $X^{train}, \boldsymbol{y}^{train}$ and a test set $X^{test}, \boldsymbol{y}^{test}$. The training set is used to optimize the cost function by learning the optimal parameters $\boldsymbol{\theta}$ of the model such that $f(\boldsymbol{x};\boldsymbol{\theta})$ represents a useful relationship between the input and the target data. The test set is then used after training to evaluate an error function $e(f(X^{test};\boldsymbol{\theta}), \boldsymbol{y}^{test})$. Since this set was not observed during the training phase it can be used to estimate how the model generalizes.

One issue that arises is the choice of the training and test sets. Optimally, training would make use of all data available but doing so would leave the researcher in the dark as to whether it generalizes well or just approximates the data set. Further, the choice of training and testing set will further influence the performance of the model since different divisions of the set may result in different relationships between input and output appearing during training which will impose a bias on the model.

One way of overcoming this issue is by performing k-fold cross-validation. The data set is randomly split into k equally sized and disjoint subsets. One subset is set aside for testing while the union of the other $k-1$ subsets is used for training. Once training is done, the test set is used to evaluate the performance. This is then repeated k times such that each subset is used as a test set and the final evaluation is taken to be the average of the k prediction errors.

2.3.3 *Data preprocessing*

Data preprocessing may be an important step when performing machine learning predictions and which methods one should apply is highly dependent on the data and model used. When investigating large data sets consisting of a multitude of different variables, spanning a large number of dimensions, the quality of the data may vary. This may be due to any number of reasons such as the way in which the data has been collected, variability in the data the reason of which may not have been recorded or changes in the data structure occuring after the data collection has begun.

If a data point is missing information in one or more variables, the point is said to be incomplete[11]. If this is the case, one has the choice of either omitting the point from the sample set or inserting a value in place of the missing attribute. There are a number of choices available to insert a missing value; e.g. a predetermined global constant, mean or median of all samples of the variable or a computed value based on regression or inference methods. Although filling in a missing value may be an attractive choice in order to keep the data set large it imposes a bias dependent on the method chosen[11]. Any value entered will have a chance of being incorrect and as such may alter any following computations including these values.

Subsequently, when working with data describing different variables it is often the case that they take on wildly differing ranges. In the present case of load forecasting the data set may for example contain just temperature, pressure and electric load. While temperature may range from something like -40 to 40 $^{\circ}C$, the electric load accumulated for a small city is in the range of hundreds of thousands of kilowatt hours, and air pressure takes on a rather small range of values centered around 101 kPa. A vector representation of the temperature, pressure, and load from some

previous point in time will be highly skewed in certain directions and the data set will most likely be offset far from the origin. Rescaling and translating the variables as a preprocessing step is recommended in most machine learning applications for a wide variety of reasons:

- The initialization of the weights in a neural network are dependent on the scale of the inputs[26].
- Models making use of distance functions will interpret variables with larger numerical values as more significant[13][26].
- Large variations in numerical values between the different variables may render the training and prediction processes ill-conditioned[13][26].
- Regularization such as weight decay benefits from standardized inputs[26].
- Many optimization routines are in some sense sensitive to scaling[26].

There are a number of ways to rescale the data but the most common in practice is to modify each variable so that it has mean 0 and standard deviation 1[26]. This entails computing the sample mean and standard deviation for each column of X, subtracting the mean and dividing by the standard deviation. With subscript i denoting row i, subscript j denoting column j, corresponding to sample i and feature j respectively, the sample mean is given by

$$\bar{X}_j = \frac{\sum_i X_{ij}}{n} \tag{1}$$

whereas the sample standard deviation is given by

$$s_j = \sqrt{\frac{\sum_i (X_{ij} - \bar{X}_j)^2}{n-1}}. \tag{2}$$

Each column is then rescaled as

$$\tilde{X}_j = \frac{X_j - \bar{X}_j}{s_j}. \tag{3}$$

These statistics are computed on the training set. The motivation behind this is that the training set is the only observed data available, and only the observed data can contribute to knowledge of the mean and standard deviation of the underlying distribution. Thus $\bar{X}_j^{train}$, s_j^{train} are computed and used to rescale both the training and test set via $\tilde{X}_j{}^{train} = \frac{X_j^{train} - \bar{X}_j^{train}}{s_j^{train}}$ and $\tilde{X}_j{}^{test} = \frac{X_j^{test} - \bar{X}_j^{train}}{s_j^{train}}$.

Rescaling the target data is "typically more a convenience ... than a necessity"[26] when working with a single target variable, and rescaling is done using the same methods as for the input data. It is however important to apply the inverse transformation on the prediction results since the model will make predictions within the same range as the target training data.

2.3.4 *Feature selection*

One major subproblem in machine learning is that of feature selection, i.e. the process of finding relevant variables to be used with the predictive model, and identifying irrelevant or redundant variables. As machine learning is used to address problems with larger feature spaces and as more data becomes available in all areas of research, research in feature selection methods progresses in parallel.

The definition of an optimal subset of features is both dependent on the problem and the needs of the analyst, and prediction accuracy may need to be weighed against computational complexity, memory availability, bandwidth and data storage. While certain applications may be given plenty of CPU time to run, some areas require the model to feasibly be run in real-time on devices with less than high-end hardware. The number of input parameters as well as the complexity of the model will have a direct effect on both runtime and prediction accuracy.

Features are often characterized as relevant, irrelevant or redundant, but also in this case the definitions vary depending on author, research area and utility of the concept. In [15] the following, slightly vague definitions are given:

Definition 6. *A feature is **relevant** if it has an influence on the output that cannot be assumed by other features.*

Definition 7. *A feature is **irrelevant** if it has no influence on the output and its values are generated at random for each example.*

Definition 8. *A feature is **redundant** if it can be replaced by another feature.*

This definition of relevant puts some weight behind a variable being a unique descriptor of the target but quantifying relevance is done heuristically in the context of machine learning experiments. Quantifying relevance and subsequently selecting the 'best' set of features is a non-trivial problem. Within a finite data set random correlations will appear between irrelevant features and the target, and using these features in training may worsen the performance of the predictor[21].

Furthermore, Guyon makes a strong case that feature relevancy, irrelevancy and redundancy are notions that change when investigating combinations of variables[9]. In a handful of illustrative examples, it is shown that apparently redundant variables can be used in combination to improve prediction performance, that highly correlated variables in combination may still provide complementary information about the target, and that two by themselves irrelevant variables can become relevant when used together.

There will always be the need for trade-offs, or feature selection would not be an actual problem. The benefits of feature selection include enabling visualization of the data, reducing training and prediction times, improving prediction accuracy[9], and reducing the complexity of the model[16], and each method delivers their own benefits and weaknesses.

Many feature selection methods can be described by just two components: a search strategy and an evaluation measure. If one wishes to avoid performing an exhaustive search of all the variables, a search strategy will devise how the method investigates different combinations of variables. Exhaustive searches can be incredibly expensive, depending on the size of the data set and the evaluation measure used and so there is extensive literature investigating different strategies for combining subsets of features in search of an optimum. Typically they are divided into *forward selection* and *backward elimination* algorithms, where the former starts with an empty set and successively adds more features whereas the latter begins with all features and successively eliminates them, but alternative approaches are also available.

Feature selection methods can roughly be divided in three categories: filter methods, wrapper methods and embedded methods, while methods combining the properties of these categories are called hybrid methods. Filter methods perform feature selection before involving a machine learning model and as such only works with the data set itself. Wrapper methods use machine learning models as score functions to rank combinations of variables based on their predictive performance for the particular model. Embedded methods perform variable selection during the training of the machine learning model[9]. Only wrappers and filters will be addressed in this thesis.

Wrapper methods make effective use of machine learning models as scoring functions and as such they have an inherent flexibility to them. A wrapper method couples a search strategy with an evaluation metric and a model, and searches for optimal subsets of features by training the model on different subsets of features and evaluating the results. Since the model is independent from the wrapper it is easily configured with any model of the users choice.

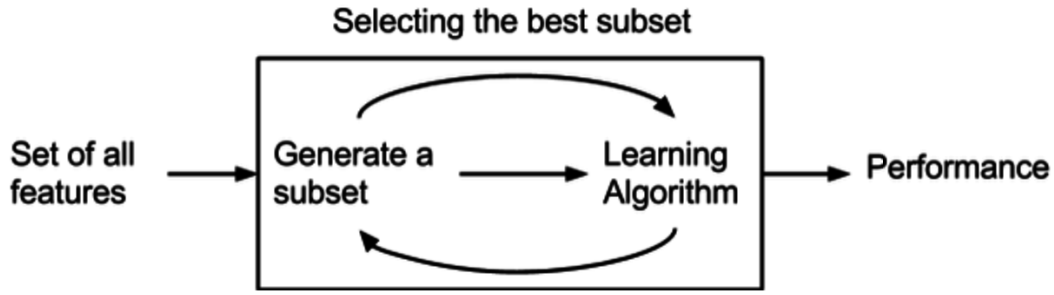


Figure 1: Basic outline of wrapper methods for feature selection. The learning algorithm is used to iteratively evaluate the goodness of the generated subset until some predetermined stop criterion is met. Image from Wikipedia[18].

A lot of criticism towards wrapper methods is based on the computational complexity of training a model each time a new subset of features is chosen as well as the risk of biasing the feature set to the selected model. The cost of performing training has however led researchers to investigate new methods for accelerating the evaluation[4] and greedy search strategies have been applied to largely reduce the number of combinations of features needed to reach an acceptable optimum subset of features[9]. Efficient choice of search strategy has also been shown to reduce the induced model bias[9].

Filter methods on the other hand are entirely independent from the inductive model. A filter works by evaluating some attribute relating the features to each other and to the target and evaluating their goodness without performing predictions. Common attributes used are the Pearson correlation, symmetrical uncertainty or Spearman correlation. Their name is derived from being used as a preprocessing step in machine learning, *filtering* out features.

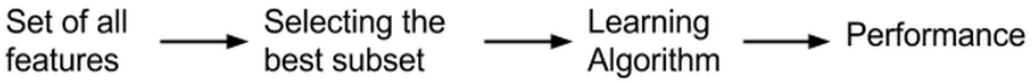


Figure 2: Basic outline a filter algorithm. The optimal subset is selected based on intrinsic attributes of the data set, without involving a learning algorithm. Image from Wikipedia[19].

The biggest benefit of filters is argued to be their computational efficiency; compared with wrappers and embedded methods the amount of computation needed is much less. Filter methods result in a generic choice of features since they are independent of the inductive model chosen[9], and as such they only make use of the inherent characteristics of the data set investigated. However, this adds the possibility of missing interdependent features that a wrapper method of sufficient complexity may find[15]. There are approaches to mitigate this, such as constructing sums or products of features, exponentiating features or logarithmizing features.

2.3.5 *Pearson correlation*

The Pearson product-moment correlation coefficient is a statistical tool for measuring linear dependence between two continuous stochastic variables. It is defined as

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}. \tag{4}$$

For a sample population, the covariance and the standard deviations are replaced by the sample estimates of the respective measures, yielding

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{5}$$

where $\bar{x}$ and $\bar{y}$ represent the sample mean of the two distributions. The measure takes values between -1 and 1, where the extreme values occur when there is perfect negative or positive correlation between the two sample distributions.

2.3.6 *Fast Correlation-Based Filter*

One article by Yu et al[33] proposes an algorithm for feature selection that uses a variant of information entropy as the evaluation measure. Given two random variables X and Y, let $p(x)$, $p(y)$ be the marginal probability for $x \in X$ and $y \in Y$ respectively, and let $p(x|y)$ represent the conditional probability of x occurring given y. The marginal entropy for a random variable is then defined as

$$H(X) = -\sum_i p(x_i) \log p(x_i) \tag{6}$$

and the conditional entropy between two variables X and Y is defined as

$$H(X|Y) = -\sum_j p(y_j) \sum_i p(x_i|y_j) \log p(x_i|y_j). \tag{7}$$

The mutual information is then given by $I(X;Y) = H(X) - H(X|Y)$. By normalizing the mutual information, the authors then define the symmetrical uncertainty as

$$SU(X,Y) = 2\frac{I(X;Y)}{H(X) + H(Y)}. \tag{8}$$

The measure is symmetrical, as $SU(X,Y) = SU(Y,X)$, and the normalization puts it in the range $[0,1]$. An SU of 0 indicates that there is no information to be gained

about one variable via knowledge of the other whereas 1 indicates that knowledge of one variable completely predicts the value of the other.

The probabilities occurring in the formulae for the entropies pose some issue that needs addressing. Since all features in the data set are sampled values of continuous variables and the original probability density function is unknown, the marginal and conditional probabilities need to be approximated. In the implementation of the algorithm for this thesis the approximation is done via uniform discretization of the values with a fixed and equal number of bins for both the X and Y variables which may impose an error source.

The authors argue that the chosen search strategy enables the algorithm to find features that are both relevant to the prediction target as well as identify redundancy between the features themselves.

Initially all feature-target correlations $SU(F_i, L)$, where L denotes load, are measured and sorted to evaluate the relevance to the target. A threshold $\delta \geq 0$ is used to immediately sort out those features that the user deems irrelevant by removing any features whose feature-class correlation is lower than δ.

In order to evaluate redundancy feature-feature correlations need to be measured. For a data set of N features this would entail the computation of N^2 correlation values and a greedy search strategy is heuristically designed to lower this number drastically. Denoting the feature F_i with the highest feature-class correlation the predominant feature, all correlations $SU(F_i, F_j)$ are computed and those that have a higher correlation with the predominant feature than with the target, i.e. $SU(F_i, F_j) > SU(F_j, L)$, are deemed redundant and removed. This is repeated for each feature, in descending order of $SU(F_i, L)$. If on average half of the remaining features are removed each iteration, the time complexity is now $O(N\log N)$.The algorithm is given below:

```
Let Load = L, Feature i = F_i
input: S(F_1, F_2, ..., F_n, L)
Set delta >= 0
Compute SU(L, F_i) = SU_Li for all i
Let S' = List of features sorted by SU(L, F_i)
    -values in descending order
Remove all features with SU(F_i, L) < delta
For each F_p in S':
  For each F_q in S' after F_p:
    If SU(F_p, F_q) >= SU(L, F_q) remove F_q from S'
Return S'
```

Since the list is ordered by descending correlation with the target, the resulting set of features will have a low correlation with each other but a high correlation to the target.

2.3.7 *Support Vector Machines*

Support Vector Machines originated in the 1960's but its current form, encompassing non-linear learning via kernels and soft-margins, has its roots in the 90's, developed by Vladimir Vapnik and his colleagues at the AT&T Bell Laboratories[29]. Support Vector Machines are a versatile class of algorithms that include methods for both classification and regression, aptly named Support Vector Classification (SVC) and Support Vector Regression (SVR). Historically they have been used for problems in optical character recognition, object recognition, natural language processing, and as previously stated load prediction.

The objective function of SVR is motivated by the objective function of SVC, where the goal is to find the maximum-margin hyperplane that separates two linearly separable classes. Finding the hyperplane entails finding a linear function $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ such that, for the given training set X^{tr} with corresponding outputs $\boldsymbol{y}^{tr}$, $f(\boldsymbol{x})$ does not deviate by more than a given margin ϵ for each data point. The distance between the two margins is $\frac{2\epsilon}{||\boldsymbol{w}||}$, and maximizing this distance is equivalent with minimizing $||\boldsymbol{w}||$. The objective function is taken as the square of the norm, $||\boldsymbol{w}||^2$, both due to its regularization properties but also to be able to make use of quadratic programming algorithms. Maximum-margin is a desired property in both SVR and SVC as a larger margin means a greater possibility to generalize. This is a convex optimization problem of the form:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}||\boldsymbol{w}||^2 \\ \text{s.t.} \quad & y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b \leq \epsilon \\ & \boldsymbol{w}^T\boldsymbol{x}_i + b - y_i \leq \epsilon \end{aligned} \tag{9}$$

Since this problem will not generally have a feasible solution with values of ϵ that are of interest to the application, the margin is softened by introducing slack variables ξ, $\xi*$ which allow points to deviate by more than ϵ from f by virtue of increasing

the cost function linearly with the magnitude of the slack variables. This leads to the form:

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} \xi_i + \xi_i^* \\ \text{s.t.} \quad & y_i - \boldsymbol{w}^T\boldsymbol{x}_i - b \leq \epsilon + \xi_i \\ & \boldsymbol{w}^T\boldsymbol{x}_i + b - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i* \geq 0 \end{aligned} \tag{10}$$

The constant parameter C determines how much the softening of the margin will penalize the cost function compared with the magnitude of the weights, and plays a part in determining the complexity of the model.

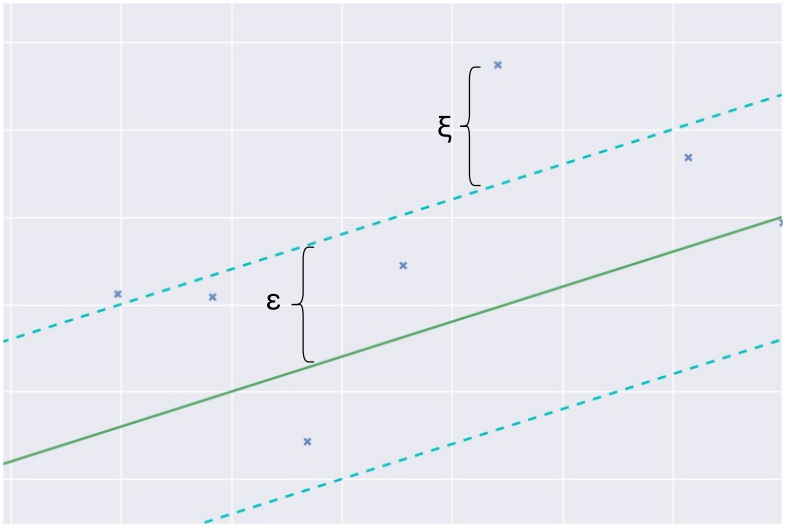


Figure 3: The margin parameter ϵ and the soft margin variable ξ. Only data points further than ϵ from the function $f(x)$ at the green line will contribute to the cost function.

The optimization problem as stated in (10) has a dual form which can be solved by using the Karush-Kuhn-Tucker criteria. This is computationally cheaper than solving the primal problem when the number of variables is greater than the number of samples in the data set, but the dual form is also a key step in allowing the SVR to be extended with non-linear transformations via kernel methods.

From (10) the Lagrangian can be constructed as:

$$\begin{aligned} L =& \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} \xi_i + \xi_i^* - \sum_{i=1}^{n} \eta_i \xi_i + \eta_i^* \xi_i^* \\ & - \sum_{i=1}^{n} \alpha_i (\boldsymbol{w}^T \boldsymbol{x}_i + b - y_i + \epsilon + \xi_i) \\ & - \sum_{i=1}^{n} \alpha_i^* (y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b + \epsilon + \xi_i^*) \end{aligned} \tag{11}$$

where the Lagrange multipliers $\alpha_i, \alpha_i^*, \eta_i, \eta_i^*$ are constrained to be positive.

Theorem 1, chapter 8 of [3] states that if $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*; \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\eta}, \boldsymbol{\eta}^*)$ is a saddle point of L above, then

1. $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ solves the primal problem

2. $\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\eta}, \boldsymbol{\eta}^*$ satisfy the KKT conditions.

Thus the primal variables must satisfy

$$\partial L_{\boldsymbol{w}} = 0, \quad \partial L_b = 0, \quad \partial L_{\boldsymbol{\xi}} = 0, \quad \partial L_{\boldsymbol{\xi}^*} = 0 \tag{12}$$

and taking the partial derivatives of L with respect to the given variables yield the constraints

$$\partial L_{\boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \boldsymbol{x}_i = 0, \tag{13}$$

$$\partial L_b = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0, \tag{14}$$

$$\partial L_{\boldsymbol{\xi}_i} = C - \alpha_i - \eta_i = 0, \tag{15}$$

$$\partial L_{\boldsymbol{\xi}_i^*} = C - \alpha_i^* - \eta_i^* = 0. \tag{16}$$

The variables $\boldsymbol{\eta}$ and $\boldsymbol{\eta}^*$ can immediately be eliminated via reshaping (15) and (16). Additionally, since η_i and $\eta_i^* \geq 0$ for all i, α_i and $\alpha_i^* \leq C$ for all i. (14) puts a constraint on the remaining Lagrange multipliers, and (13) provides an explicit expression for the weights $\boldsymbol{w}$ as a function of the data and the Lagrange multipliers.

Inserting (13), (15) and (16) in (11) yields, together with the constraints on $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$, the dual problem

$$\begin{aligned} \max_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} \quad & -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\boldsymbol{x}_i^T\boldsymbol{x}_j \\ & -\epsilon\sum_{i=1}^{n}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{n}y_i(\alpha_i-\alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)=0 \\ & \alpha_i,\alpha_i^*\in[0,C] \end{aligned} \tag{17}$$

Information about b can be had from the complementary slackness principle ([3] chapter 8, eq. 18) that states

$$\begin{aligned} \alpha_i(\epsilon+\xi_i-y_i+\boldsymbol{w}^T\boldsymbol{x}_i+b)&=0 \\ \alpha_i^*(\epsilon+\xi_i^*+y_i-\boldsymbol{w}^T\boldsymbol{x}_i-b)&=0 \\ (C-\alpha_i)\xi_i&=0 \\ (C-\alpha_i^*)\xi_i^*&=0 \end{aligned} \tag{18}$$

for all i at the solution. If any of ξ_i or ξ_i^* is non-zero, corresponding to $|\boldsymbol{w}^T\boldsymbol{x}_i+b-y_i|\geq\epsilon$, then by necessity the corresponding α_i or $\alpha_i^*=C$. It is also immediately apparent that only one of the two constraints $y_i-\boldsymbol{w}^T\boldsymbol{x}_i-b\leq\epsilon+\xi_i$ and $\boldsymbol{w}^T\boldsymbol{x}_i+b-y_i\leq\epsilon+\xi_i^*$ will be active, i.e. become an equality, which means that at most one of α_i and α_i^* will be non-zero.

Now, if $\alpha_i<C$, then the constraint $y_i-\boldsymbol{w}^T\boldsymbol{x}-b\leq\epsilon$ is satisfied which means $y_i-\boldsymbol{w}^T\boldsymbol{x}-\epsilon\leq b$, and if $\alpha_i>0$ then either $\epsilon-y_i+\boldsymbol{w}^T\boldsymbol{x}_i+b=-\xi_i$ or $\epsilon-y_i+\boldsymbol{w}^T\boldsymbol{x}_i+b=0$, in the case of $\alpha_i<C$ and $\alpha_i=C$ respectively. This can be formulated as $\epsilon-y_i+\boldsymbol{w}^T\boldsymbol{x}_i+b\leq 0\Rightarrow b\leq-\epsilon+y_i+\boldsymbol{w}^T\boldsymbol{x}_i$.

Similarly, if $\alpha_i^*<C$, then $\xi_i^*=0$ and $\boldsymbol{w}^T\boldsymbol{x}_i+b-y_i\leq\epsilon$ is satisfied, which leads to $b\leq\epsilon+y_i-\boldsymbol{w}^T\boldsymbol{x}_i$. For $\alpha_i^*>0$ it instead holds that $\epsilon+y_i-\boldsymbol{w}^T\boldsymbol{x}_i$. Through all these inequalities b now has a bounding on both sides that depends on the sample index i. Furthermore, if any α_i or α_i^* lies strictly within $(0,C)$ then both inequalities for that given Lagrange multiplier will hold and b will have a fixed value.

Linear models are however not very representative of many real world data sets and thus desirable to allow for the model to represent non-linear relationships. This is done via the kernel trick which builds on the fact that support vector machines only explicitly make use of the dot product between data points. A non-linear mapping of the data can be achieved implicitly by replacing the dot products with kernel functions that take as input two data points. There are a number of conditions that

need to be fulfilled for a function to be usable as kernels[29] but these are omitted.

Definition 9. *A kernel is a function* $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ *that can be represented as a dot product on an auxiliary feature space* $\mathbb{F} \subset \mathbb{R}^m$*. Given a mapping* $\phi(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{F}$ *k can be defined as* $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j))$

The strength of kernels lies in being able to implicitly map the data to a feature space where the number of dimensions is customizable by the researcher without actually having to compute the representation in this feature space. Two typically used kernels are the polynomial kernel of degree p: $\text{ﻉ}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^T \boldsymbol{x}_j + c)^p$, where c may be taken to be 0 for a homogeneous polynomial, and the radial basis function (RBF) kernel: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma^2})$.

For a polynomial kernel of degree 2, $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^T \boldsymbol{x}_j)^2$, used on a data set with two variables, the auxiliary mapping $\phi(\boldsymbol{x}_i) = (x_{i,1}^2, \sqrt{2} x_{i,1} x_{i,2}, x_{i,2}^2)$ maps the data set from $\mathbb{R}^2$ to $\mathbb{R}^3$. The mapping $\phi(\boldsymbol{x}_i)$ is never actually computed.

The RBF kernel maps the data to $\mathbb{R}^\infty$[1]. Without loss of generality, let $n = 2$ and $\sigma^2 = 1/2$. Then

$$\begin{aligned} k(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \exp\left(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2\right) \\ &= \exp\left(-(x_{i,1} - x_{j,1})^2 - (x_{i,2} - x_{j,2})^2\right) \\ &= \exp\left(-x_{i,1}^2 + 2x_{i,1}x_{j,1} - x_{j,1}^2 - x_{i,2}^2 + 2x_{i,2}x_{j,2} - x_{j,2}^2\right) \\ &= \exp\left(-||\boldsymbol{x}_i||^2\right) \exp\left(-||\boldsymbol{x}_j||^2\right) \exp\left(2\boldsymbol{x}_i^T \boldsymbol{x}_j\right) \end{aligned}$$

and finally a Taylor expansion of the last factor yields $\exp\left(2\boldsymbol{x}_i^T \boldsymbol{x}_j\right) = \sum_{n_0}^{\infty} \frac{(2\boldsymbol{x}_i^T \boldsymbol{x}_j)^n}{n!}$. Since this is an infinite sum of polynomial kernels, the RBF will implicitly map the data set into infinite-dimensional space.

The kernel function then takes the place of the dot products $\boldsymbol{x}_i^T \boldsymbol{x}_j$ in 17 which yields

$$\begin{aligned} \max_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} \quad & -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(\boldsymbol{x_i}, \boldsymbol{x}_j) \\ & - \epsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \\ & \alpha_i, \alpha_i^* \in [0, C]. \end{aligned} \tag{19}$$

The weight vector $\boldsymbol{w}$ now takes the form $\boldsymbol{w} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi(\boldsymbol{x}_i)$ and the resulting function is thus

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)k(\boldsymbol{x}_i, \boldsymbol{x}) + b.$$

3

METHOD

3.1 THE DATA SET

The meteorological institutes SMHI and Yr provide open access to large amounts of weather data. Among other things, one can find predictions for different weather variables over Sweden, Norway, Finland and Denmark. Predictions are made by collecting present data from as many sources as possible to form an initial state from which to base predictions on. Data processing and meteorological methods are then applied to extrapolate the data into the future at different temporal and spatial resolutions. Expektra subscribes to this data in a form that has been interpolated to match the coordinates of the grid areas for which the load is predicted.

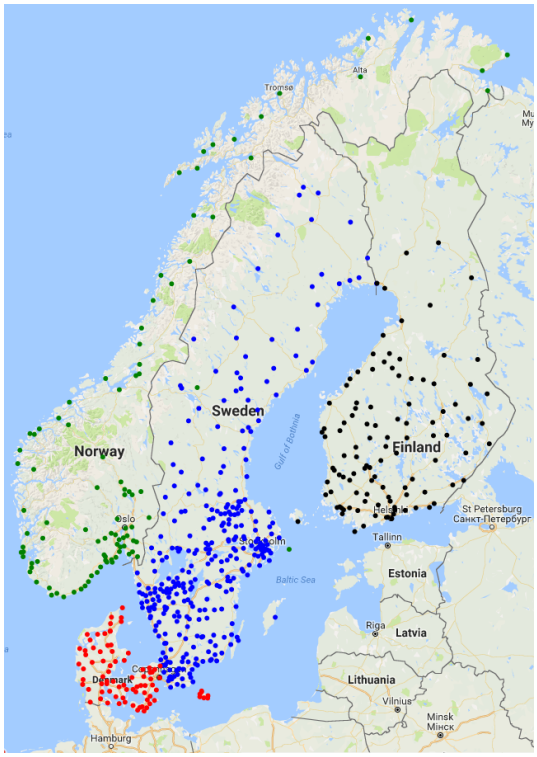


Figure 4: Each point represents a coordinate matching a grid area with the color coding representing the country. The weather data has been interpolated to correspond to each grid area.

In total there are 606 locations, each with 9 and 10 different variables for Yr and SMHI respectively, described at an hourly resolution. The locations are specified by an area within which the load is computed and summed into a single scalar, also presented at an hourly resolution, called grid areas. These grid areas are determined by energy suppliers and brokers, and they vary in size and density depending on the population density and municipality borders. Each grid area is represented by a single coordinate, depicted in Figure 4.

The variables describing the weather are tabulated in Table 1.

Variable	Unit	Data source
Temperature	$^{\circ}C$	SMHI, Yr
Pressure	hPa	SMHI, Yr
Humidity	Percent	SMHI, Yr
Probability of thunderstorm	Percent	SMHI
Precipitation	mm	SMHI, Yr
Wind direction	Degrees	SMHI, Yr
Wind speed	m/s	SMHI, Yr
Wind gust	m/s	SMHI
Total cloud cover	Percent	SMHI
Cloudiness	Percent	Yr
Visibility	km	SMHI
Fog	Percent	Yr
Dewpoint Temperature	$^{\circ}C$	Yr

Table 1: Weather variables provided by the meteorological institutes.

With 10 weather variables from SMHI and 9 variables from Yr, each with 606 locations, the total dimensionality of the data set is $606 \cdot (10 + 9) = 11{,}514$. The data used in this thesis runs from 2014/04/01 to 2016/05/31, clocking approximately 20,000 hours and subsequently around 20,000 sampled values per variable. As such it easy to see the need for investigating what information carries any value, and what may be discarded while still retaining accurate prediction results.

Along with the weather data, Expektra has access to continuously updated information about the load for the different grid areas. Due to the process by which the load is measured and the manner in which the information is collected, the data comes with a quality marking. In essence, the load information is approximated during the time it is being collected, and as such, there is a time frame of approximately 7 days back in time during which the load information is less accurate.

3.2 EXPERIMENTAL SETUP

3.2.1 *Approach*

The experiments in this project wishes to address a number of hypotheses.

- There is a limit to accuracy gained by adding more data of the same variable
- The different weather variables can be ordered by gain in predictive power of the load
- Geographical location plays a role in the predictive power of the weather variables

The experiments revolve around using the correlation measures and the filter algorithm described in Chapter 2 to perform feature selection on different subsets of the data set, and evaluating the resulting feature sets by performing load prediction on the four target areas. The experiments are varied by changing the number of locations to investigate, and changing which weather variables to include. Experiments begin by using temperature as the single variable with which to investigate the different hypotheses, and then expands into further analysis using combinations of variables.

All experiments were run on a personal computer equipped with an Intel Core i7-950 CPU running at 3.07 GHz, 6 GB of RAM, running Windows 10 with 64-bit architecture and Python 3.5 64 bit. Data processing has been performed with Pandas 0.18.1[22]. The Python package scikit-learn 0.18-dev[25] was used for the machine learning algorithms.

3.2.2 *Delimitations*

A number of delimitations have been imposed on the experiments in order to maintain feasible computational complexity.

The data set was downloaded from a remote SQL database and locally stored during the project. This procedure was time-consuming due to the loading times of SQL queries of the magnitude involved, and had the effect that the data set was not continuously updated during the project.

The data set was first pruned to only include Swedish grid areas. Partly because this halved the number of locations from 606 to 318 but also because the locally stored data set showed a large discrepancy in the number of samples between the Swedish locations (approximately 17-20k samples per variable and location) and the rest of

Scandinavia (approximately 11,000 samples per variable and location). Furthermore only data from SMHI was used. This choice was arbitrary.

Four target areas were chosen for which to predict the load. These will throughout the report be denoted with A, B, C and D[1]. The areas were chosen by their geographical locations as well as the density of other areas surrounding them.

3.2.3 *Preprocessing*

All data analysis and prediction has been performed with the same preprocessing steps. A moving average is first applied with a 24 hour window to each feature as well as the output. The intention here is to make the problem a little less complex; prediction of load at an hourly resolution is a difficult problem that spans outside the scope of this thesis. A snapshot of how load in a certain area behaves over a time period of one week in October 2015 is presented in Figure 5. The unprocessed load contains several distinct extreme values in a single day while the processed load is much smoother.

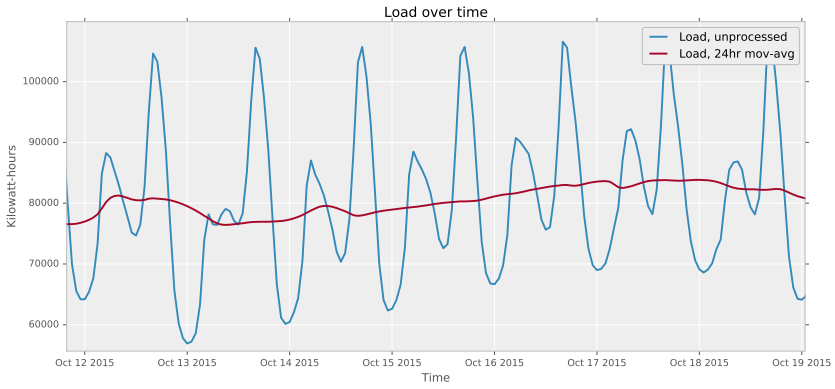


Figure 5: Load of area A as a function of time over approximately one week. The difference in complexity between the processed and unprocessed data series is very distinct.

The input variables are preprocessed in the same way. Unprocessed and processed temperature predictions from SMHI is shown in Figure 6.

1 Area names are obfuscated.

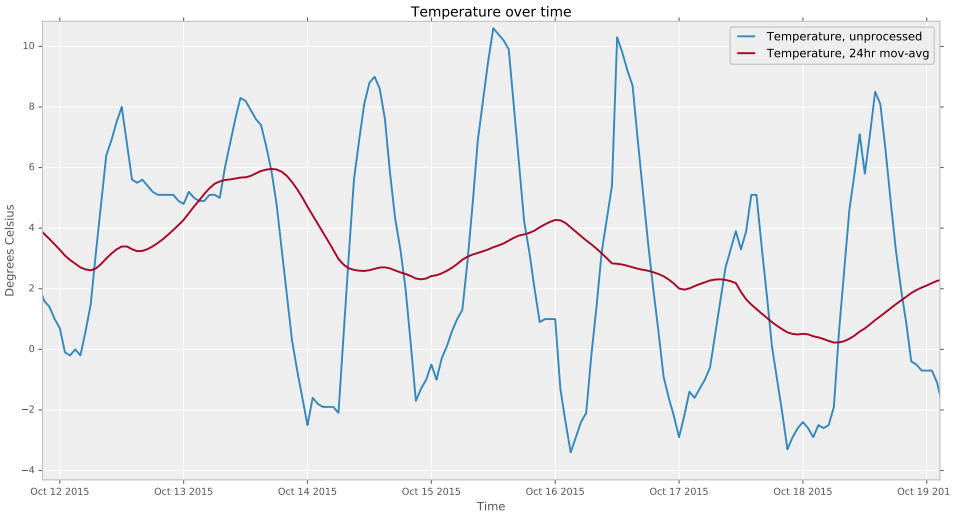


Figure 6: Temperature of area A as a function of time over the same week as the load.

Samples with missing values in any feature are dropped from the data set so as not to impose any bias into the data. In all cases, no points were found to be missing on the interior of the data set. Removing missing values merely had the effect of cutting off either the tail or the head of the time series, ensuring that each series had the same beginning and end time.

All features are normalized so that they have mean 0 and variance 1.

3.2.4 *Discretization of variables for computation of symmetrical uncertainty*

Prior to being able to use the fast correlation-based filter, a discretization of the variables need to be performed. As previously stated this was done uniformly and equally in both variables, and subsequently the number of bins need to be determined.

Iterating over bin numbers ranging from 10 up to 170, the FCBF algorithm was run on two different feature sets for each of the four target areas and then ranked by performing a prediction of the load for each area using the resulting feature subsets as input variables. Feature set 1 was chosen to contain all weather variables from the target area resulting in 10 features. Feature set 2 used temperatures from all locations, resulting in 318 features.

For feature set 1, setting the number of bins between 30 and 70 resulted in the best prediction scores. For feature set 2, changing the number of bins made little difference. The filter returned different locations with different bin settings, but the difference in prediction accuracy was negligible. Averaging the number of bins

yielding the best results from feature set 1 gives 51.5 which was subsequently rounded down to 50.

3.2.5 *Learning model*

For all experiments, support vector machines were used with the radial basis function kernel and default settings as described in [27], $\epsilon = 0.1$ and $\gamma = 1/n$ with n being the number of features. The parameter σ of the RBF kernel is in scikit-learn defined as $\gamma = 1/(2\sigma^2)$

3.2.6 *Evaluation*

For evaluating the prediction errors, symmetric mean absolute percentage error (SMAPE) is used. With $\hat{y}_t$ denoting the predicted load and y_t denoting the actual load at time t, SMAPE is defined as:

$$\text{SMAPE} = \frac{1}{n}\sum_{t=1}^{n}\frac{|\hat{y}_t - y_t|}{|\hat{y}_t| + |y_t|}. \tag{20}$$

Throughout this report SMAPE has been presented as percentages by replacing the above definition with $100\cdot$ SMAPE.

The number of folds during k-fold cross-validation is set to 8.

4

RESULTS AND DISCUSSION

4.1 BASELINE RESULTS

As a starting point, a set of results are formed against which the subsequent predictions can be compared against. For this purpose two experiments are performed in which the features used for prediction are chosen heuristically. First, persistence is used, meaning that the energy load from a previous time is used as a predictor for future load. To do this, the data set will take the shape $x_t = L_{t-k}, y_t = L_t$ and the goal for the learner will be to approximate a function

$$f(L_{t-k}) = L_t.$$

This corresponds to the simplest case where no other information is available than the variable that is sought to be predicted. As stated in Section 3.1 the quality of the load information is subject to change up to approximately 7 days into the future. This is used as a guideline for the parameter k which is divided in steps of 12, starting from 120 hours and going up to 196 hours. 168 hours is used for comparison and other values are given for reference.

Hours of lag	Area A	Area B	Area C	Area D
120	3.52	2.84	3.06	5.94
132	3.56	2.84	3.03	6.04
144	3.59	2.83	2.99	6.12
156	3.64	2.83	2.96	6.20
168	**3.70**	**2.86**	**2.96**	**6.30**
180	3.76	2.95	3.03	6.45
196	3.81	3.06	3.16	6.61

Table 2: SMAPE from predicting load using the lagged load for training.

The second feature set to be used as a baseline result is the set of temperatures from all locations in the data set. As has been established in Chapter 2 temperature is one of the strongest predictors of energy consumption, and this feature will be used in various experiments pertaining to establishing the impact of geographical

location on prediction accuracy. Selecting n_a areas to use the temperature from, a data point at time t may be expressed as

$$\mathbf{x}_t = \mathbf{T}_t = (T_t^1, T_t^2, ..., T_t^{n_a})$$

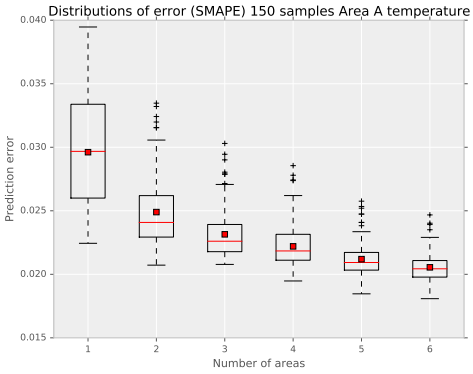
and the model is now trained to learn a relationship expressed by

$$f(\mathbf{T}_t) = L_t.$$

The feature set is built via uniformly and randomly sampling n_a unique areas and then training and evaluating a learner using temperature from these locations as input. n_a samples are drawn 150 times from the pool of 318 areas, for each n_a ranging from 1 to 6. The SMAPE for each prediction is tabulated and mean, median, standard deviation, minimum and maximum are computed for each n_a. Statistics as well as box plots and describing the distributions of errors are given for area A in Figure 7.

A number of things can be read from these statistics. First and foremost, it is easy to achieve better results by picking one or few areas anywhere in Sweden and use the temperature as a predictor of load, than to use persistence. The median SMAPE is lower than persistence for each of the four target areas whereas the maximum SMAPE becomes lower than persistence with $n_a = 2, 5, 3$ and 1 for areas A, B, C and D respectively.

Second, the range of the SMAPE shrinks with each new area added, suggesting that there is saturation to the information that can be gained by adding more temperature series. The difference in mean error between $n_a = 5$ and $n_a = 6$ is $0.07\%, 0.06\%, 0.05\%$ and 0.04% for areas A, B, C and D respectively, indicating that the gain in accuracy will be small if random sampling is used to select these areas.



	SMAPE				
n_a	Mean	Std	Min.	Mdn.	Max.
1	2.96	0.42	2.24	2.97	3.95
2	2.49	0.26	2.07	2.41	3.35
3	2.31	0.19	2.08	2.26	3.03
4	2.22	0.17	1.95	2.18	2.86
5	2.12	0.13	1.85	2.09	2.58
6	2.05	0.11	1.81	2.04	2.47

Figure 7: Statistics of prediction results from random sampling of n_a areas from which to use the temperature to predict the load, with area A as target area. 150 different combinations were sampled for each value of n_a. The boxes of the box plots range from the lower to the upper quartile of the data and the whiskers extend the box by $1.5 \cdot IQR$, where IQR is the difference between the upper and lower quartile. Anything else is considered an outlier and marked with a $+$. The line marks the median and the square marks the mean.

4.2 CORRELATIONS AND PREDICTION ACCURACY

To enable an understanding of what makes a variable relevant as a predictor of the load, it is of interest to quantify the relationship between load and the different variables. Three choices of measurements are chosen: Pearson correlation, symmetrical uncertainty, and prediction error measured with SMAPE. For each of the weather variables the correlation and symmetrical uncertainty are computed with the weather variables taken from the same area as the load.

Evaluating the error of each feature is done by taking the feature as sole input x and training the model to approximate the function

$$f(x_t) = L_t.$$

The quantities are listed in table 3 for areas A and B, and table 4 for areas C and D.

One thing that immediately stands out is the complete dominance of temperature's correlation with load, and subsequently how well it performs as a predictor. However, this is just confirmation of earlier research since, as stated in chapter 2, temperature has successfully been used as a predictor of load for several decades.

For each of the four areas, the symmetrical uncertainty between feature and load takes on very small values, where only temperature and probability of thunderstorm

	Area A			Area B		
Feature	Corr.	SU	SMAPE	Corr.	SU	SMAPE
Temperature	-0.952	0.45	2.40	-0.946	0.38	2.69
Prob. thunderstorm	-0.540	0.24	6.20	-0.540	0.22	6.19
Humidity	0.452	0.11	9.40	0.452	0.10	8.40
Visibility	-0.273	0.12	11.01	-0.273	0.12	9.69
Cloud cover	0.193	0.08	10.73	0.193	0.07	9.64
Pressure	-0.129	0.11	10.49	-0.129	0.11	9.36
Wind speed	0.111	0.07	11.52	0.111	0.07	9.92
Wind direction	0.102	0.13	12.08	0.102	0.14	10.38
Precipitation	-0.073	0.04	11.67	-0.073	0.05	10.46
Wind gust	-0.066	0.12	11.97	-0.066	0.12	10.07

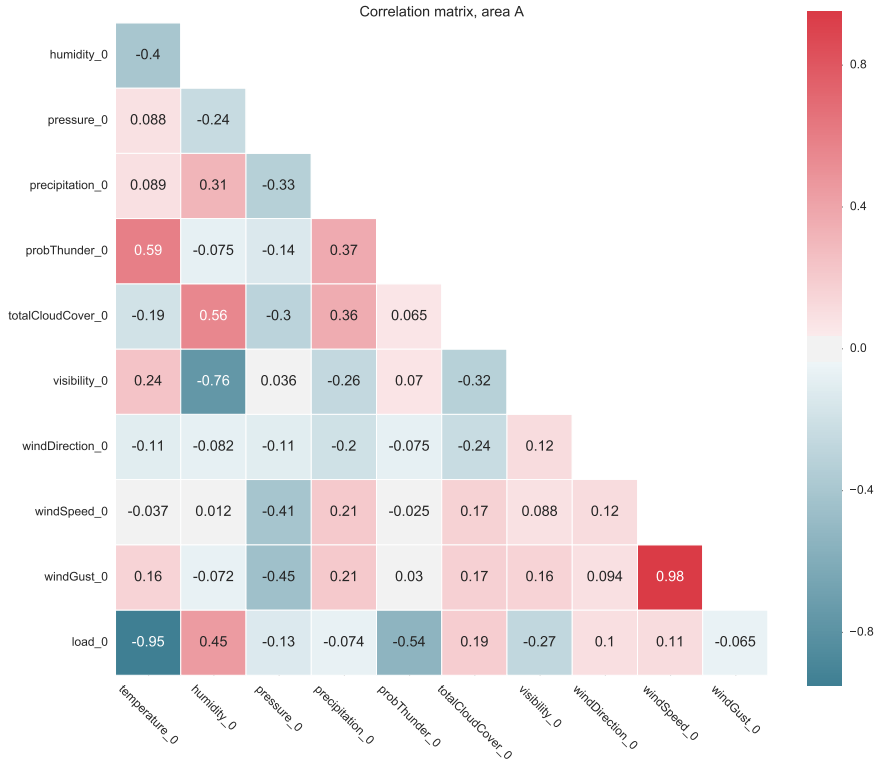
Table 3: Correlation between features and load, where feature and load originate from the same location. A 24hr moving average has been applied to both features and load.

stand out somewhat. By the definition of symmetrical uncertainty, this indicates that little information is gained about the load by investigating these features. Lacking is however a description of little, and further investigations will have to decide the actual impact of these features on load prediction.

It is also apparent that for the present set of variables taken from the target area, performing a ranking based on the variables relationship to the load would yield different rankings if Pearson correlation or symmetrical uncertainty is used.

Further, it is of interest to investigate how the features relate to each other. For this purpose, the Pearson correlation and symmetrical uncertainty are computed for each feature pair and visualized for area A via the heatmaps in Figure 8.

Not surprisingly the heatmap for correlation displays a wide range of values. This is exaggerated by the fact that the variables have been preprocessed by a 24 hour moving average. The heatmaps for symmetrical uncertainty shows that most intra-feature relationships are extremely small, of a magnitude around 0.1 or less, barring a handful of the variables.



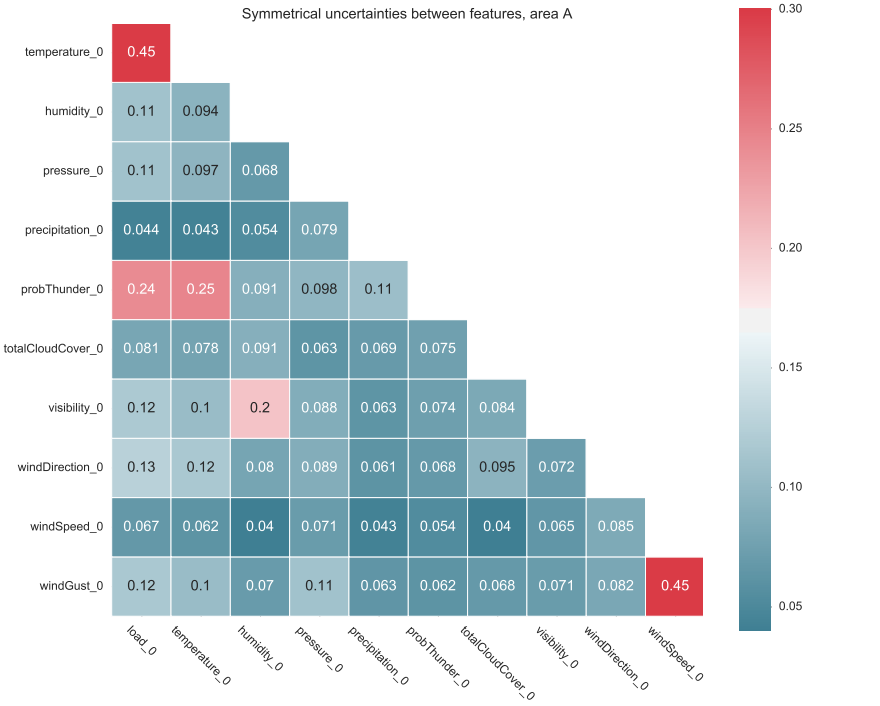


Figure 8: Heatmaps of the correlations and symmetrical uncertainties between the features and load, located at area A.

	Area C			Area D		
Feature	Corr.	SU	SMAPE	Corr.	SU	SMAPE
Temperature	-0.952	0.38	2.62	-0.945	0.42	4.08
Prob. thunderstorm	-0.575	0.25	5.28	-0.509	0.26	9.10
Humidity	0.463	0.10	8.15	0.490	0.13	13.64
Visibility	-0.242	0.12	9.39	-0.415	0.13	15.65
Cloud cover	0.272	0.07	9.02	0.219	0.07	18.18
Pressure	-0.139	0.11	8.91	-0.183	0.12	17.12
Wind speed	0.182	0.07	9.67	0.032	0.06	19.25
Wind direction	0.094	0.12	10.24	-0.060	0.11	18.52
Precipitation	-0.049	0.05	9.93	-0.112	0.06	19.20
Wind gust	0.054	0.11	10.19	-0.156	0.11	18.74

Table 4: Correlation between features and load, where feature and load originate from the same location. A 24hr moving average has been applied to both features and load.

In order to investigate how redundancy manifests itself in the data set, two learning experiments are performed and compared. First, the temperature is taken from each area to use as a single feature to approximate the function

$$f(T_t^{other}) = L_t^{target}, \tag{21}$$

and then the temperature from each area is paired with the temperature from the target area to approximate

$$f(T_t^{other}, T_t^{target}) = L_t^{target}. \tag{22}$$

Histograms of the resulting errors are plotted for area A in Figure 9. For each of the four areas, the difference between the highest and lowest error are 2.21%, 1.57%, 1.70% and 3.35% for areas A, B, C and D respectively. It is apparent that there are certain series much more suitable for predicting the load at the four targets. The resulting error using the temperature from the target areas are 2.40%, 2.69%, 2.62% and 4.08%.

When adding the temperature from the target area the range shrinks by approximately one order of magnitude in all four cases. The difference between maximum and minimum is now 0.32%, 0.59%, 0.46% and 0.64%. The maximum of each case corresponds to the single input prediction from the target area, and no result from this experiment resulted in a worse score. Visible in the histograms for areas A and C are now sharp peaks close to the highest error illustrating that a lot of the temperature series provide very little value to load prediction. For these areas, the 30th

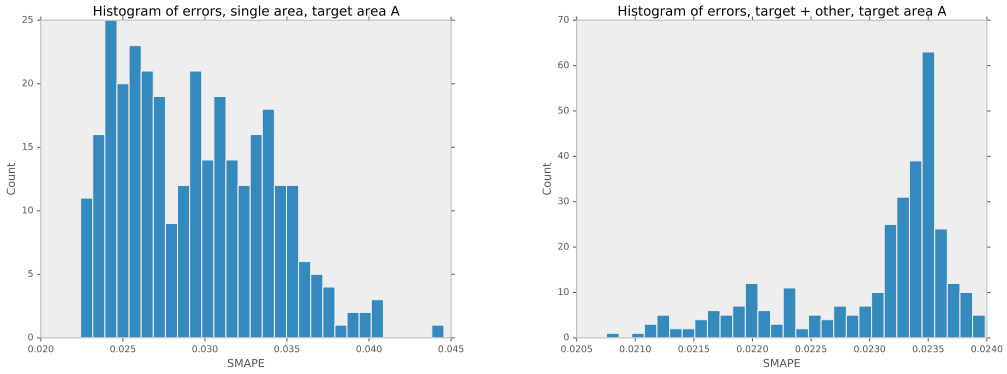


Figure 9: Histogram of prediction errors from using temperature as input. On the left is the single-variable case, on the right is the two-input case.

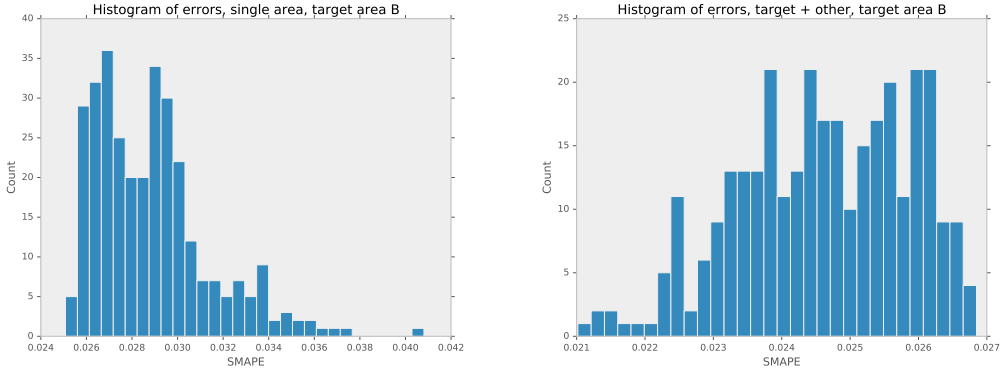


Figure 10: Histogram of prediction errors from using temperature as input. On the left is the single-variable case, on the right is the two-input case.

percentile is within 0.1% of the maximum. While the range of errors for area B is small the distribution here is much flatter, with the 30th percentile being slightly below the mean of the scores, or approximately 0.3% below the maximum. Area D displays the complete opposite behavior with a wide peak further away from the max indicating that many of the other areas may provide complementary information that can be used to improve the prediction score.

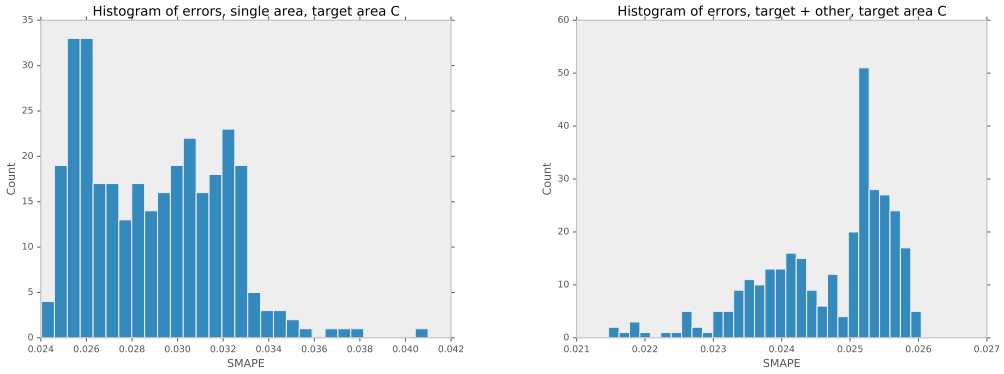


Figure 11: Histogram of prediction errors from using temperature as input. On the left is the single-variable case, on the right is the two-input case.

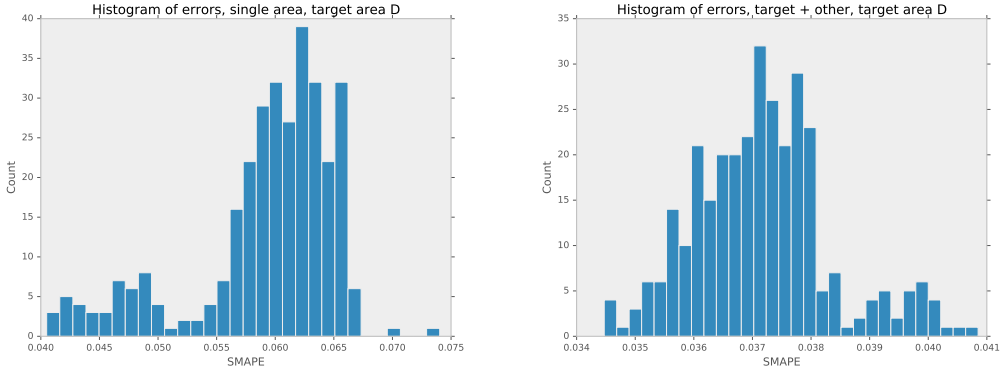


Figure 12: Histogram of prediction errors from using temperature as input. On the left is the single-variable case, on the right is the two-input case.

4.3 CORRELATIONS AND DISTANCE

For each area in the data set, the geographical latitude and longitude were provided. For each of the four target areas, the distance from the target area to all other areas is computed. The distance was subsequently used to investigate the geographical properties of the data set. Figure 13 displays for each area how many other areas are within a certain distance. Due to area A, B and C being in relatively close proximity to each other their curves are naturally very similar.

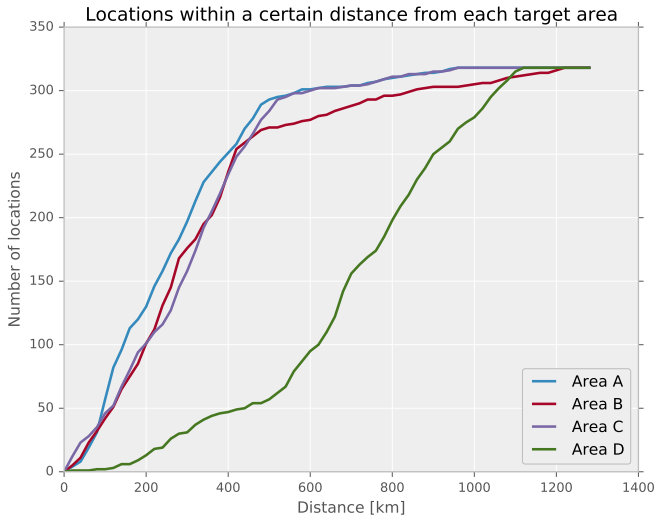


Figure 13: A visualization of the number of locations within a certain distance from the four chosen target areas.

To visualize how the relationships between the variables change over distance, the correlations between the load and temperature of the target area are computed against all other temperature series. The results are plotted against distance in Figure 14. In both sets of plots, a strong linear relationship reveals itself. The results of computing the correlation between distance and correlation are shown in Table 5. While the coefficients are significantly smaller for the temperature-load correlations, it is clear that distance from the target area lends a lot of information about the relationship between local and distant temperature.

Area	Corr(T, T)	Corr(T, L)
Area A	-0.96	0.85
Area B	-0.98	0.87
Area C	-0.94	0.81
Area D	-0.97	0.92

Table 5: The correlation coefficients for the distance-correlation data.

Following the same idea, plotting the results from the previous section against distance shows two things. First, in the single area case, errors diminish as areas closer to the targets are chosen. This is intuitive since the correlation drops off with distance, and when using a single variable as a predictor, the difference in just linear correlation may affect the performance significantly. In the two area case there is some tendency to achieving better results by pairing the temperature from the target area with a temperature series from an area further away, in all four cases. This

Figure 14: The left figure describes the correlations between the temperature from the target area and all other areas, whereas the right describes the correlations between target load and all other temperatures. These are plotted as a functions of distance. A strong linear relationship can be seen in both plots for all areas.

follows the idea from [33]; a feature is a good predictor if it correlates highly with the target but not with the other features. As can be seen from figure 14 the magnitude of the correlation with both load and temperature from the target area declines with distance but is still in the vicinity of 0.9 for the areas furthest away.

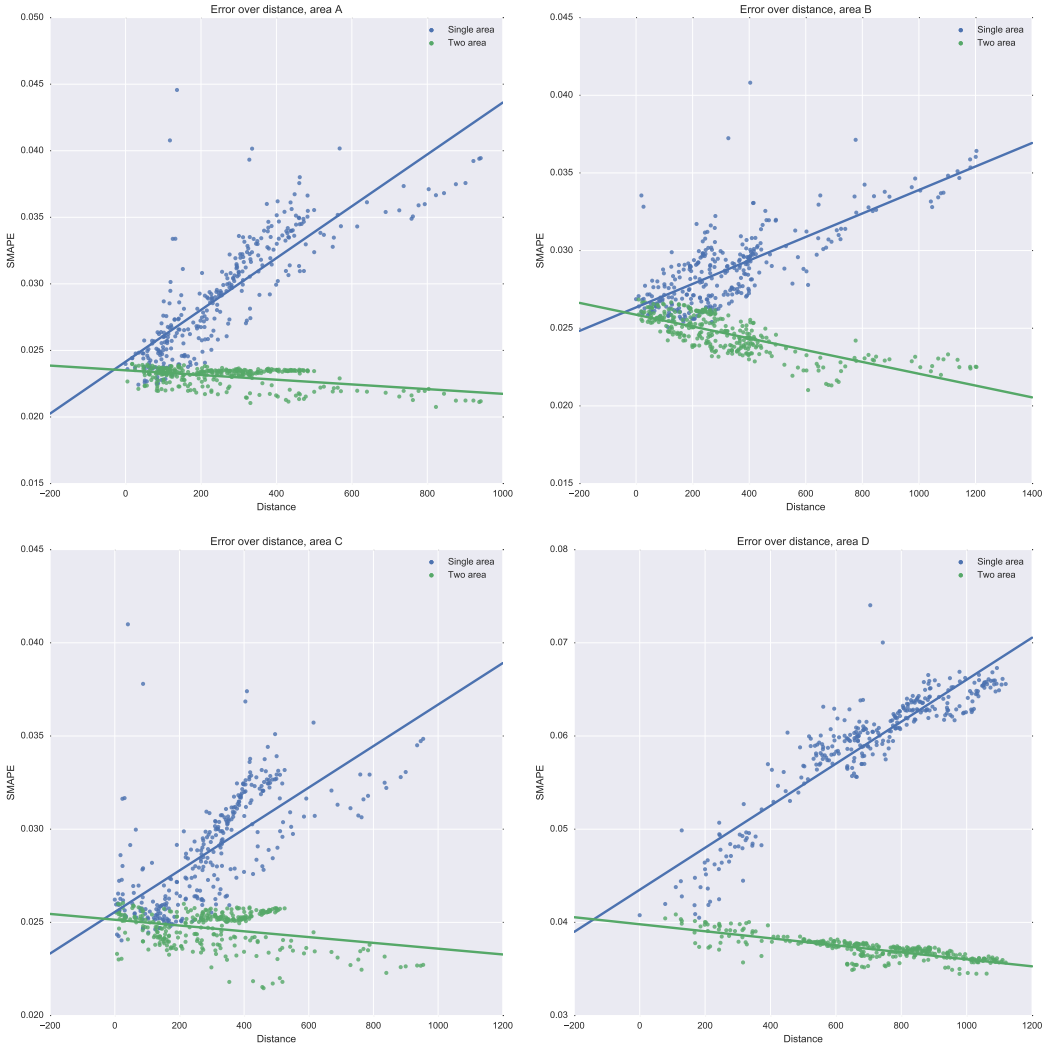


Figure 15: Regression analysis of the sets of prediction errors from the single and two area case. For a single area, prediction accuracy shows a trend to increase further away from the target area. Pairing the temperature from the target area with another source shows that the two areas should be far apart in order to decrease the error.

4.4 FAST CORRELATION-BASED FILTER

The Fast Correlation-Based Filter is employed to perform two things. First it is used to rank the different weather variables against each other and hopefully remove those that provide very little detail in light of the others. This is done by applying it to the set of variables from the target area only and as such it begins with 10 variables. Load is naturally used as the class C in the description of the algorithm as detailed in Section 2.3.6, and $n_{bins} = 50$ as explained in Section 3.2.4. Denoting the set of features returned from the filter as $\boldsymbol{x}^{FCBF}$ training is performed to approximate the function

$$f(\boldsymbol{x}_t^{FCBF}) = L_t \tag{23}$$

for all four areas. The resulting features along with test errors are listed in Table 6.

Area	$\boldsymbol{x}^{FCBF}$	SMAPE
A	Temperature, humidity, wind gust, wind direction	1.52
B	Temperature, prob. thunderstorm, visibility, wind gust	1.85
C	Temperature, visibility, pressure, prob. thunderstorm, wind direction	1.63
D	Temperature, visibility, pressure, prob. thunderstorm	2.84

Table 6: FCBF was applied to the set of weather variables from the target areas. From the 10 features, 4 remain for areas A, B and D whereas 5 remain for area C.

Next, the filter is used to search the set of areas for useful features. This can be done in a number of ways: for each area included in the set there is a choice to make of which weather variables to include. Including all variables means the filter will compare all weather variables from all areas. Another option is to run the filter on carefully chosen subsets of variables in the hope that a priori knowledge may help the filter keep variables of interest that otherwise may not be considered relevant. It may also exclude computations that are known to be of little interest. The very least this will accomplish is to enable certain features to become predominant and ensure their selection, and shorten the runtime of the algorithm.

First, let $\boldsymbol{x}^{all}$ denote the set of weather variables regardless of area. With the help of the filter, 9 feature sets are formed and used for training and testing.

FCBF, ALL AREAS, $\boldsymbol{x}^{all}$ All weather features from all areas are pooled together and entered into FCBF.

FCBF, ALL AREAS, $\boldsymbol{x}^{FCBF}$ The features returned form FCBF for the single area case are for all areas entered into FCBF.

FCBF, ALL AREAS, $\boldsymbol{x}^{all}$, PER FEAT For each weather feature, FCBF is run on all areas and thus only sorts through the geographical locations without regard for intra-feature correlations.

FCBF, ALL AREAS, $\boldsymbol{x}^{FCBF}$, PER FEAT As above, but ignores the features that were filtered out by FCBF in the previous experiment.

The resulting features for each experiment[1] and these sets are subsequently used for training and testing. The results are displayed in Table 7 along with the result from using three other feature sets; the complement of $\boldsymbol{x}^{FCBF}$, the complement of $\boldsymbol{x}^{FCBF}$ together with temperature, and $\boldsymbol{x}^{all}$.

	SMAPE			
Feature set	Area A	Area B	Area C	Area D
Target, $\boldsymbol{x}^{all}$	1.13	1.13	1.00	2.10
Target, $\boldsymbol{x}^{FCBF}$	1.52	1.85	1.63	2.84
Target, $\boldsymbol{x}^{all} \setminus \boldsymbol{x}^{FCBF}$	4.05	4.59	6.38	8.99
Target, temperature $\cup$ $(\boldsymbol{x}^{all} \setminus \boldsymbol{x}^{FCBF})$	1.23	1.21	1.50	2.48
FCBF all areas, $\boldsymbol{x}^{all}$	0.96	0.83	0.77	1.71
FCBF all areas, $\boldsymbol{x}^{FCBF}$	0.99	0.86	0.83	1.71
FCBF all areas, $\boldsymbol{x}^{all}$, per feat	0.98	0.83	0.77	1.72
FCBF all areas, $\boldsymbol{x}^{FCBF}$, per feat	0.99	0.85	0.82	1.73
FCBF all areas, temperature $\cup$ $(\boldsymbol{x}^{all} \setminus \boldsymbol{x}^{FCBF})$, per feat	0.97	0.86	0.82	1.85

Table 7: Comparison of prediction results from a number of feature sets. In the present case it appears that the fast correlation-based filter succeeded in providing a good subset of relevant features. The accuracy is 0.2 to 0.4% higher than simply taking all weather variables from the target area. However, using the fact that temperature is a relevant feature and combining it with the complement of the resulting features from running FCBF on the variables at just the target area actually provided a better result than using the features suggested by FCBF.

1 Listed in a separate document.

	No. features			
Feature set	Area A	Area B	Area C	Area D
FCBF all areas, $\boldsymbol{x}^{all}$	19	21	21	20
FCBF all areas, $\boldsymbol{x}^{FCBF}$	11	16	17	17
FCBF all areas, $\boldsymbol{x}^{all}$, per feat	30	31	31	29
FCBF all areas, $\boldsymbol{x}^{FCBF}$, per feat	11	17	17	18
FCBF all areas, temperature $\cup$ ($\boldsymbol{x}^{all} \setminus \boldsymbol{x}^{FCBF}$), per feat	21	16	16	13

Table 8: Resulting number of features.

There is one result that stands out here: training and testing on temperature together with the complement of $\boldsymbol{x}^{FCBF}$ yields a slightly better result than that of $\boldsymbol{x}^{FCBF}$, most notably for area B with a 0.6% improvement and the score is comparable to using all features, except in the case of area C. Excluding temperature from this feature set show remarkably poor results compared to all other cases so far.

All four tests using the full pool of areas reveal very little difference in performance. The largest difference of 0.05% is smaller than the variation between the cross-validation sets which is of the order of $\pm 0.1\%$ around the reported error. It is here seen that using the complementary features with temperature yield comparative performance to that of the filtered feature set.

It is also worth noting that $\boldsymbol{x}^{all}$ from the target area and the filtered results from the full data set perform similarly with a difference in less than 0.2% for areas A, B and C.

4.5 HOW TO SEARCH FOR BETTER RESULTS

With the relationship between distance and correlations in mind, it seems natural to suggest that distance itself can be used as a very cheap way of performing feature selection. In the following, n_a areas are chosen such that they are equidistant, or as close as possible to it. This is ensured by selecting areas via the following scheme:

```
Let A be the set of all areas
Order A by distance from target area
Let n_a be the number of areas to select
Let d_max be the maximal distance
Let q = d_max/n_a-1
chosen_areas = empty list
for i = 1 to n_a:
    x_chosen = None
    min_distance = None
    for all x in A:
        distance_from_split = abs(x.distance - (i-1)*q)
        if x_chosen == None or distance_from_split < min_distance:
            x_chosen = x
            min_distance = abs(x_chosen.distance - (i-1)*q)
    chosen_areas.append(x_chosen)
return chosen_areas
```

For each subset of areas, testing and training is then performed using three sets of features from each area: temperature, $\boldsymbol{x}^{FCBF}$ and $\boldsymbol{x}^{all}$. The results are displayed in Table 9, 10 and 11 respectively. The first row in each set is a repetition of old results as the chosen area is the target area.

In the case of using n_a temperature variables as input, it is apparent that for $n_a = 2$ through 6, the resulting prediction error is comparable to the minimum of the error achieved by random sampling in section 4.1 being within 0.1%. The difference in error between $n_a = 2$ and $n_a = 9$ is $0.34\%, 0.39\%, 0.35\%$ and 0.59% for areas A, B, C and D respectively. When adding additional weather variables the difference in error between $n_a = 2$ and $n_a = 9$ shrinks; Using $\boldsymbol{x}^{FCBF}$ areas B, C and D still achieve some 0.3%, but with $\boldsymbol{x}^{all}$ there is essentially no difference at all.

	SMAPE			
n_a	Area A	Area B	Area C	Area D
1	2.40	2.69	2.62	4.08
2	2.12	2.25	2.27	3.57
3	2.02	2.11	2.21	3.52
4	2.00	2.11	2.15	3.48
5	1.95	2.00	2.07	3.32
6	1.93	1.97	2.03	3.24
7	1.86	1.94	1.98	3.16
8	1.84	1.80	1.97	3.22
9	1.78	1.86	1.92	2.98

Table 9: Prediction results from selecting areas based on distance. Temperature is used as the single weather variable so the size of the feature set is equal to n_a.

	SMAPE			
n_a	Area A	Area B	Area C	Area D
1	1.52	1.85	1.63	2.84
2	1.06	1.25	1.01	2.24
3	0.96	1.01	0.89	1.99
4	0.98	0.91	0.87	1.92
5	0.93	0.88	0.81	1.87
6	0.97	0.86	0.82	1.81
7	0.94	0.84	0.79	1.84
8	0.97	0.82	0.82	1.78
9	0.95	0.84	0.79	1.79

Table 10: Prediction results from selecting areas based on distance, using the features suggested by the fast correlation-based filter. The size of the feature set is $4n_a$, $4n_a$, $5n_a$ and $4n_a$ for A, B, C and D respectively.

	SMAPE			
n_a	Area A	Area B	Area C	Area D
1	1.13	1.13	1.00	2.10
2	1.00	0.82	0.81	1.74
3	0.97	0.81	0.79	1.72
4	0.98	0.80	0.80	1.74
5	0.96	0.82	0.78	1.73
6	0.98	0.79	0.78	1.71
7	0.96	0.80	0.76	1.70
8	0.99	0.93	0.79	1.77
9	0.97	0.79	0.77	1.78

Table 11: Prediction results from selecting areas based on distance, using all weather variables. The size of the feature set is $10n_a$ for all areas.

In the last experiment, the pool of areas is divided into n_s equidistant segments and FCBF sorts through each segment. In each segment, all weather features are included. Here, a variation of FCBF is also used whereby the symmetrical uncertainty measure is replaced with the Pearson correlation. This new variant is also evaluated on the set of all areas and all weather features. For each of the four areas, FCBF in its original form provides absolutely nothing interesting when dividing the set of areas into multiple segments. The difference from $n = 1$ to $n = 2$ is of the order of 0.05% which in no way justifies almost doubling the amount of features. The Pearson variant sees a small improvement for areas B, C and D[2] of 0.2% to 0.5%.

Area A				
	FCBF(SU)		FCBF(Pearson)	
n_s	No. feats.	SMAPE	No. feats.	SMAPE
1	11	0.99	7	1.06
2	25	0.94	11	1.00
3	31	0.93	13	0.96
4	37	0.94	18	0.93
5	38	0.95	20	0.95
6	47	0.93	25	0.92
7	50	0.95	28	0.94
8	58	0.96	31	0.92

2 Listed in a separate document.

Table 12: After dividing the pool of areas equally by distance from the target area the fast correlation-based filter is applied to the set of temperatures from the resulting subsets. These are subsequently used for training and testing of load prediction for each target area.

4.6 AUXILIARY FINDINGS

During these experiments it was also found that for temperature, the linear Pearson correlation had close to a monotone relationship with symmetrical uncertainty, as can be seen in the plots below. Wikipedia states that if two random variables X and Y have normal marginal distributions, and their joint distribution is a bivariate normal distribution, then mutual information, i.e. the enumerator in symmetrical uncertainty, can be expressed as a function of correlation. Mutual information is the difference between marginal and conditional entropy $I = H(X) - H(X|Y)$, and the relationship is given by $I = -\frac{1}{2}\log(1-\rho^2)$. Computing this function and plotting against the mutual information confirms this, as does a calculation of the correlation of the functions. The results are plotted and tabulated below.

Area	Corr.	p-value
A	0.993	3.00e-299
B	0.994	7.81e-307
C	0.992	2.45e-289
D	0.995	1.17e-321

Table 13: Correlation coefficient and p-value of the mutual information from the data set with the computed mutual information $I = -\frac{1}{2}\log(1-\rho^2)$.

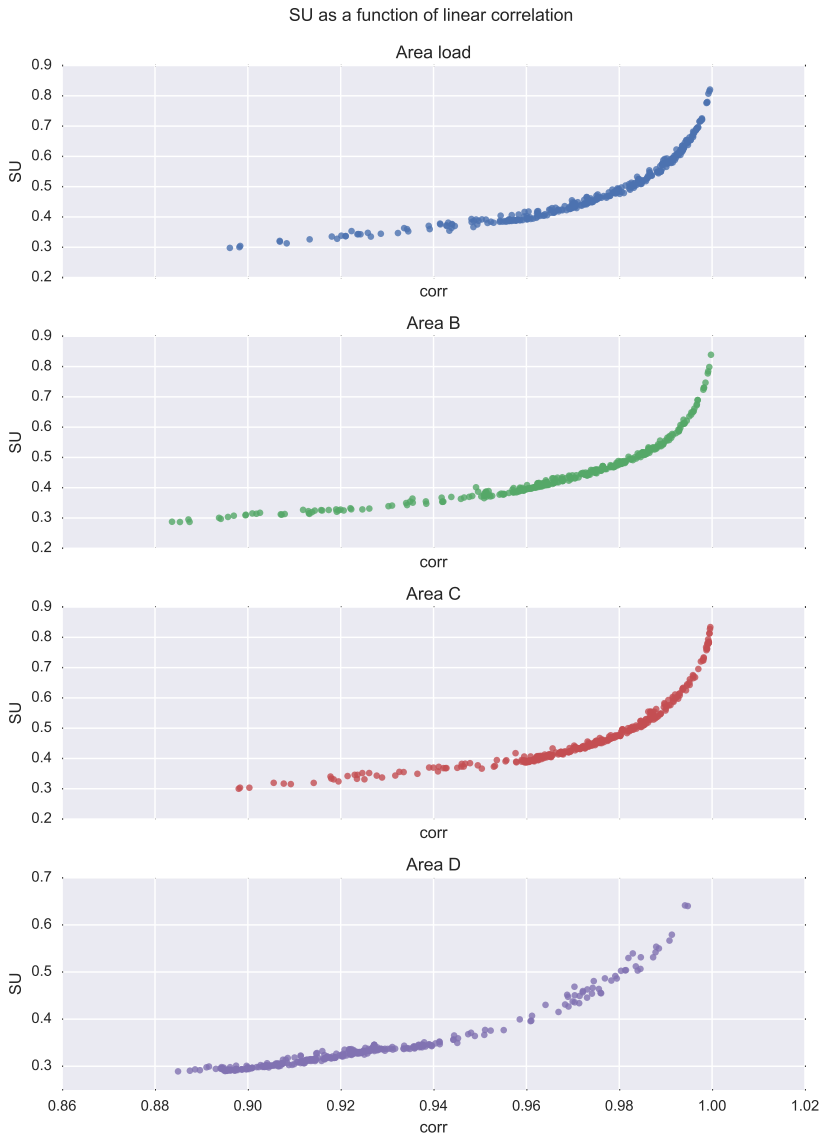


Figure 16: Scatter plots of correlations against symmetrical uncertainty for all four target areas.

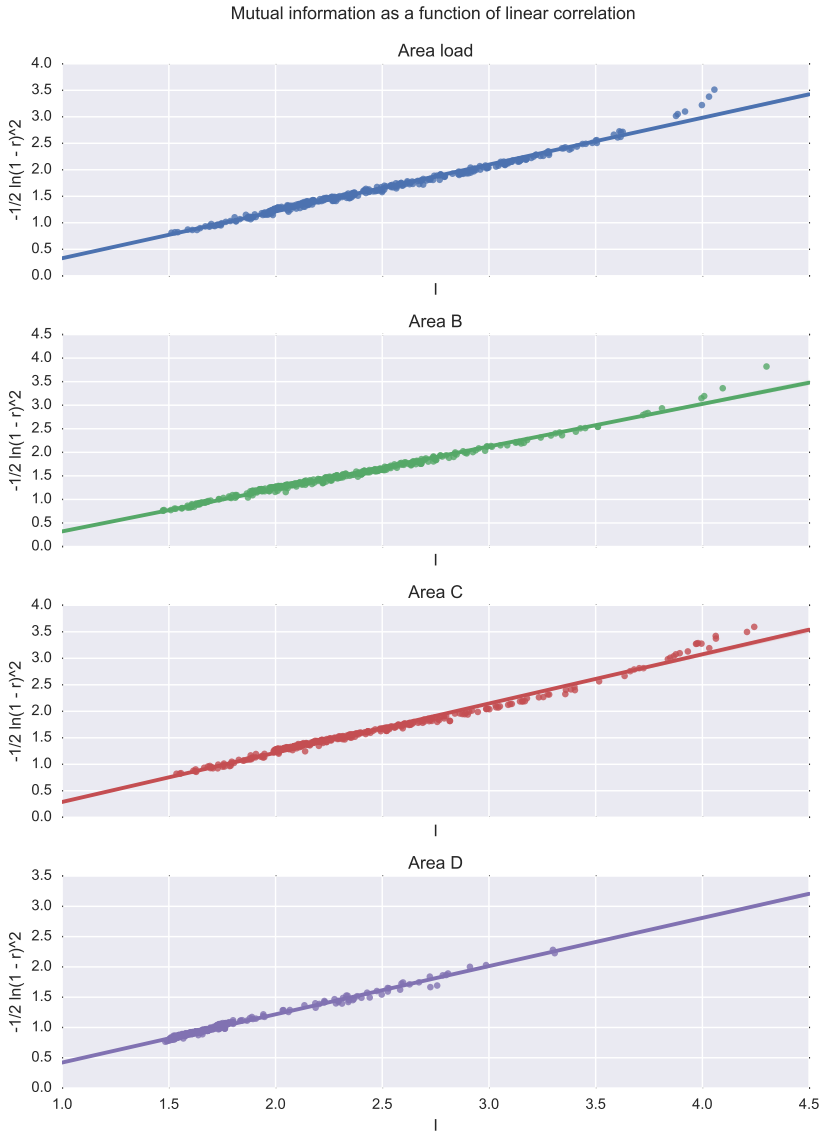


Figure 17: Scatter plots of the mutual information computed from correlations against the mutual information computed from the data. The solid line displays the result of performing linear regression on the data

5

DISCUSSION AND CONCLUSION

This thesis sets out with two goals in mind: investigating how to efficiently use the data set to minimize prediction accuracy, and attempting to exploit geography in order to select features.

The approach taken here has been to first see how each weather variable from the target area in and of itself perform as a predictor of load to form a baseline of results. Then, from these baseline results achieve an understanding of how much there is to be gained from combining different weather features as well data from different geographical locations. In the single feature case where temperature was used, random sampling of areas indicated two things: features seem to become redundant with just a handful of areas, and how the areas are chosen matter when optimizing for prediction accuracy. It would have been interesting to include random sampling experiments using more features than just temperature but these experiments took several days to run, and due to quantum fluctuations, human error, or both, they had to be redone several times. The importance of the choice of area was subsequently enhanced by visualization of the error distributions in the one and two area cases.

In the following experiments distance as a single variable was used to investigate how geography affects the statistics and the relationship between the features. One thing that is immediately clear is that for almost all of the different weather features there is a strict decline in correlation to the weather at the target as a function of distance, as can be seen in the plots in appendix **??**. This is not a huge surprise however, the distances involved are only of the order of 100 - 1000 km and the moving-average applied to the features will most certainly lessen any high-frequency behavior in the weather that could have attributed to increasing the spatial difference.

The fast correlation-based filter proved useful in the following experiments but the single area case revealed that it is by no means flawless. Adding temperature as an a priori known good predictor of load to the features that FCBF filtered out yielded slightly better results. Including additional areas did however remove this discrepancy and there was very little difference in prediction error. In any case, the filter performed well in all regards, reducing the number of areas from 3180 to below 20 when taking the entire data set into account, yielding a prediction accuracy of

0.99%, 0.86%, 0.83% and 1.71% for areas A, B, C and D respectively and this is about as good as any of the following feature sets performed.

The experiments involving distance as a feature selection tool also proved comparative to the best results. In the case of using temperature as the single weather variable, selecting equidistant areas compared to the lowest errors of random sampling in all cases. In the case of $\boldsymbol{x}^{FCBF}$ and $\boldsymbol{x}^{all}$, the scores approached their limit with just 2 or 3 areas, providing no real further gain with each new addition. Since intra-feature correlation decays with distance, not just for temperature but for all features, selecting areas based on distance proved to be a cheap variant of a correlation-based filter. However, since correlation is not strictly a monotone function of distance, this selection method is cheap but not fool-proof.

In the last experiment, the pool of areas was divided into equally sized segments based on distance, and FCBF was used to select the best features in each segment, now using both symmetrical uncertainty and Pearson correlation as the correlation measure. The intuition behind this was to investigate whether the very similar scores between most of the previous experiments could be improved in any way by pure feature selection. In all cases using symmetrical uncertainty, the number of features grew but the gain in accuracy is completely negligible. The experiment did however indicate that Pearson correlation may be successfully employed with the search strategy of FCBF and that performing the geographical segmentation did amend any loss in accuracy when looking at the entire pool of areas.

This thesis project made a few assumptions in the beginning that can be seen as having shaped these results. First and foremost, using four target areas proved both incredibly computationally taxing and made some tasks prone to error. Certain experiments took an extremely long time to run on a home desktop computer using no parallelization techniques. While it has been reassuring to have several cases to confirm experiments against, this demanded a lot of time that could have been used to advance the experiments further.

Secondly, investigating the data set preprocessed by a 24-hour average most likely imposed some restrictions on what could actually be achieved. The very best results achieved were only slightly better than a base case, and several of the experiments approached these lowest results. While the problem of performing load forecasting with an hourly resolution is far too complex for the present setup it would be interesting to see how these results vary with smaller averaging windows. Doing so could reinforce the efficiency of the methods presented and determine for what resolutions they are useful.

BIBLIOGRAPHY

[1] Online. URL: http://stats.stackexchange.com/questions/58585/how-to-understand-effect-of-rbf-svm.

[2] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. "An Empirical Comparison of Machine Learning Models for Time Series Forecasting." In: *Econometric Reviews, 29: 5, 594 — 621* (2010).

[3] Lars-Christer Böiers. *Mathematical Methods of Optimization*. Ed. by Studentlitteratur. Studentlitteratur, 2010.

[4] Avrim Blum and Pat Langley. "Selection of Relevant ffeature and Examples in Machine Learning." In: *Artificial Intelligence* 97.1-2 (1997), pp. 245–271.

[5] Bo-Juen Chen, Ming-Wei Chang, and Chih-Jen Lin. "Load Forecasting Using Support Vector Machines: A Study on EUNITE Competition 2001." In: *IEEE Transactions on Power Systems* 19 (2004), pp. 1821–1830.

[6] *Energimarknader*. Swedish. Vattenfall AB. URL: https://corporate.vattenfall.se/om-energi/energimarknader/.

[7] *Expektra AB: Expektra Predict*. English. URL: http://expektra.se/?page_id=444.

[8] J. Nuno Fidalgo and Manuel A. Matos. "Forecasting Portugal Global Load with Artificial Neural Networks." In: *Lecture Notes in Computer Science* 4669 (2007), pp. 728–737.

[9] Isabelle Guyon and Andre Elisseeff. "An introduction to Variable and Feature Selection." In: *Journal of Machine Learning Research 3* (2003).

[10] Heiko Hahn, Silja Meyer-Nieberg, and Stefan Pickl. "Electric Load Forecasting Methods: Tools for Decision Making." In: *European Journal of Operational Research* (2009).

[11] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Ed. by Diane Cerra. Morgan Kauffmann Publishers, 2012.

[12] Tao Hong and David A. Dickey. *Electric Load Forecasting: Fundamentals and Best Practices*. Visited: September 2016. 2013. URL: https://www.otexts.org/elf.

[13] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. *A Practical Guide to Support Vector Classification*. 2016. URL: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

[14] Mattias Jonsson. Personal Communication. 2016.

[15] M. Karagiannopoulos, D. Anyfatis, S.B. Kotsiantis, and P.E. Pintelas. "Feature Selection for Regression Problems." University of Patras, Greece.

[16] Vipin Kumar and Sonajharia Minz. "Feature Selection: A literature Review." In: *Smart Computing Review* 4.3 (2014), pp. 211–229.

[17] Elias Kyriakides and Marios Polycarpou. "Short Term Electric Load Forecasting: A Tutorial." In: *Studies in Computational Intelligence* 35 (2007), pp. 391–418.

[18] Lastdreamer7591. 2014. URL: https://en.wikipedia.org/wiki/File:Feature_selection_Wrapper_Method.png.

[19] Lastdreamer7591. 2015. URL: https://commons.wikimedia.org/wiki/File:Filter_Methode.png.

[20] *M3-Competition*. URL: https://forecasters.org/resources/time-series-data/m3-competition/.

[21] David J C MacKay. "Probable Networks and Plausible PrPredicti - A Review of Practical Bayesian Methods for Supervised Neural Networks." Cavendish Laboratory, University of Cambridge.

[22] Wes McKinney. "Data Structures for Statistical Computing in Python." In: *Proceedings of the 9th Python in Science Conference* (2010), pp. 51–56. URL: http://conference.scipy.org/proceedings/scipy2010/mckinney.html.

[23] Stuart McMenamin and Frank Monforte. "Short Term Energy Forecasting with Neural Networks." In: *The Energy Journal* 19.4 (1998).

[24] D.C. Park, M.A. El-Sharkawi, R.J. Marks II, L.E. Atlas, and M.J. Damborg. "Electric Load Forecasting Using An Artificial Neural Network." In: *IEEE Transactions on Power Systems* 6.2 (1991).

[25] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[26] Warren S. Sarle. *Neural Network FAQ, part x of 7: Subtitle*. Online. 2002. URL: ftp://ftp.sas.com/pub/neural/FAQ.html.

[27] *sklearn.svm.SVR - Scikit-learn 0.18.1 documentation*. URL: http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR.

[28] *SMHI Open Data API Docs - Meteorological Forecasts*. URL: http://opendata.smhi.se/apidocs/metfcst/index.html.

[29] Alex J. Smola and Bernhard Schölkopf. "A Tutorial on Support Vector Regression." In: *Statistics and Computing* 14 (2004), pp. 199–222.

[30] Håkan Törenvik. “Theories and Experience with Aiolos - An Algorithm for the Predicition of Heat and Electricity Load.” In: *None* (Unknown).

[31] *What Is Machine Learning?* URL: http://www.mlplatform.nl/what-is-machine-learning/.

[32] Yr.no. URL: http://api.met.no/weatherapi/documentation.

[33] Lei Yu and Huan Liu. “Efficiently Handling Feature Redundancy in High-Dimensional Data.” In: *SIGKDD* (2003).