DEGREE PROJECT IN MATHEMATICS,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2018

Machine Learning Algorithms for Regression Modeling in Private Insurance

MAI NGUYEN

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

Machine Learning Algorithms for Regression Modeling in Private Insurance

MAI NGUYEN

Degree Projects in Financial Mathematics (30 ECTS credits)
Degree Programme in Applied and Computational Mathematics
KTH Royal Institute of Technology year 2018
Supervisor at Pensionsmyndigheten: Erik Ferm
Supervisor at KTH: Tatjana Pavlenko
Examiner at KTH: Tatjana Pavlenko

TRITA-SCI-GRU 2018:362
MAT-E 2018:77

Royal Institute of Technology
School of Engineering Sciences
KTH SCI
SE-100 44 Stockholm, Sweden
URL: www.kth.se/sci

Abstract

This thesis is focused on the Occupational Pension, an important part of the retiree's total pension. It is paid by private insurance companies and determined by an annuity divisor. Regression modeling of the annuity divisor is done by using the monthly paid pension as a response and a set of 24 explanatory variables e.g. the expected remaining lifetime and advance interest rate. Two machine learning algorithms, artificial neural networks (ANN) and support vector machines for regression (SVR) are considered in detail. Specifically, different transfer functions for ANN are studied as well as the possibility to improve the SVR model by incorporating a non-linear Gaussian kernel. To compare our result with prior experience of the Swedish Pensions Agency in modeling and predicting the annuity divisor, we also consider the ordinary multiple linear regression (MLR) model. Although ANN, SVR and MLR are of different nature, they demonstrate similar performance accuracy. It turns out that for our data that MLR and SVR with a linear kernel achieve the highest prediction accuracy. When performing feature selection, all methods except SVR with a Gaussian kernel encompass the features corresponding to advance interest rate and expected remaining lifetime, which according to Swedish law[1] are main factors that determine the annuity divisor. The results of this study confirm the importance of the two main factors for accurate modeling of the annuity divisor in private insurance. We also conclude that, in addition to the methods used in previous research, methods such as MLR, ANN and SVR may be used to accurately model the annuity divisor.

[1] Swedish law: 5 kap. 12 § lagen (1998:674) om inkomstgrundad ålderspension

Sammanfattning

Denna uppsats fokuserar på tjänstepensionen, en viktig del av en pensionärs totala pension. Den utbetalas av privata försäkringsbolag och beräknas med hjälp av ett så kallat delningstal. Regressionsmodellering av delningstalet görs genom att använda den månatliga utbetalda pensionen som svar och en uppsättning av 24 förklarande variabler såsom förväntad återstående livslängd och förskottsränta. Två maskininlärningsalgoritmer, artificiella neuronnät (ANN) och stödvektormaskiner för regression (SVR) betraktas i detalj. Specifikt så studeras olika överföringsfunktioner för ANN och möjligheten att förbättra SVR modellen genom att införa en ickelinjär Gaussisk kärna. För att jämföra våra resultat med tidigare erfarenhet från Pensionsmyndigheten vid modellering och förutsägande av delningstalet studerar vi även ordinär multipel linjär regression (MLR). Även om ANN, SVR och MLR är av olika natur påvisar dem liknande noggrannhet. Det visar sig för vår data att MLR och SVR med en linjär kärna uppnår den högsta noggrannheten på okänd data. Vid variabel urvalet omfattar samtliga metoder förutom SVR med en Gaussisk kärna variablerna motsvarande förväntad återstående livslängd och förskottsränta som enligt svensk lag är huvudfaktorer vid bestämning av delningstalet. Resultatet av denna studie bekräftar betydelsen av huvudfaktorerna för noggrann modellering av delningstalet inom privat försäkring. Vi drar även slutsatsen att utöver metoderna som använts i tidigare studier kan metoder såsom ANN, SVR och MLR användas med framgång för att noggrant modellera delningstalet .

Acknowledgments

First of all, I would like to thank Erik Ferm, Tommy Lowén and Ole Settergren at Pensionsmyndigheten for giving me the opportunity to do this thesis, helping me in the progress and to find useful data. I would also like to give my thanks to my supervisor Associate Professor Tatjana Pavlenko at the Department of Mathematics at KTH for her valuable feedback, ideas for my thesis and Mikael Andblom who has given me support and spent his time for useful mathematical discussions during the time I did the thesis.

Contents

1 Introduction

1.1 Background

In Sweden *the Swedish Pensions Agency*[2] handles the pension for retirees. In 1913 the Swedish parliament decided to introduce the national pension system in Sweden for the first time. It was also the first pension system in the world entitled to all residents in the country. There has been many changes and in 1990 we got a complete pension system that has been used till today. In Sweden the retirement age is 65 but residents can retire once they are 61 years old. The older they are the higher pension they will receive at retirement.

The pension consists of three main parts, *Public Pension*[3], *Occupational Pension*[4] and *Private Pension*[5]. Public Pension consists of two parts, *Notional Defined Contribution NDC*[6] and *Financial Defined Contribution FDC*[7] are calculated by the Swedish Pensions Agency and Occupational Pension by private insurance companies. An illustration of this can be seen in Figure 1.

In 1994 the Swedish parliament introduced the annuity divisor to determine the yearly pension for retirees. The yearly pension is calculated by dividing the total pension by the annuity divisor. The Swedish Pensions Agency clearly states the use of the annuity divisor to determine the Public Pension. However, within the Occupational Pension, every private insurance company is free to define their own annuity divisor which is optional to publish. Although a lot of information about private insurance companies is available through the website of *KFB*[8]. It is therefore possible to approximate how the annuity divisor is formulated using said data.

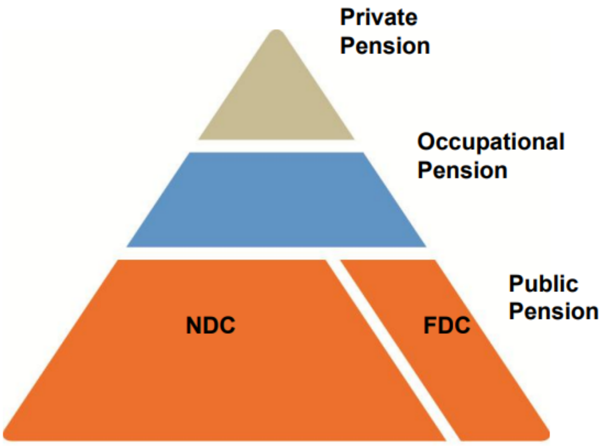


Figure 1: Pension schemes in Sweden [1]

[2] Sw. Pensionsmyndigheten
[3] Sw. allmän pension
[4] Sw. tjänsterpension/avtalspension
[5] Sw. eget sparande till pension
[6] Sw. inkomst pension
[7] Sw. primär pension or Premium Pension
[8] Konsumenternas Försäkringsbyrå

1.2 Problem Formulation

By Swedish law the annuity divisor should be based on two main factors, life expectancy at the time of retirement and an advance interest rate. Furthermore, there exist many other variables that might influence the annuity divisor such as return rates and administrations fees. On the website of KFB many different tables of various features that private insurance companies may use to formulate their annuity divisor can be accessed.

There are three main tasks to consider. The first one is to study which variables have the most impact on the annuity divisor. The second one is to obtain the most parsimonious model for the annuity divisor, i.e. a simple model with high explanatory and predictive power. The last question to be answered is whether the features life expectancy and advance interest rate have a major impact on the annuity divisor.

2 Available Data

In Sweden 9 out of 10 employees have right to Occupational Pension from their *Collective agreement*[9]. One who does not have an Occupational Pension will get lower pension when retired. The yearly pension for a retiree depends on which employment contract he has. There exist 7 different types of employment contracts:

- ITP1: employment contract for civil servants who are born after 1979.
- ITP2: employment contract for civil servants who are born before 1979.
- SAF-LO: employment contract for private workers.
- AKAP-KL: employment contract for municipality and county employees who are born after 1986.
- KAP-KL: employment contract for municipality and county employees.
- PA-KFS-09: employment contract for municipality employees who are born after 1954.
- PA-03: employment contract for State employees.

2.1 Selected data

On the website of KFB the information about these 7 types of employment contracts is available for everyone. For every type of contract there are about 10-20 private insurance companies and for every company there are about 65 observed values that can potentially affect their annuity divisor. Every type of value is studied in order to decide which ones are most likely to be included in the calculation of the annuity divisor in private insurance companies.

After reading carefully about all 65 various types of values, what they mean and how they are used in the pension system, 17 of them are chosen to constitute the database for the investigation. Below are some of our most important rules to select features:

[9] Sw: kollektivavtal

- Many features are constant over all observations which makes it reasonable to assume that they do not have any impact on the annuity divisor. Such features are therefore excluded from the database.
- Only features with no missing values are considered.
- Features that cannot be properly represented numerically such as names are ignored.

The selected 17 features are denoted and explained in Table 1. Later when we model employment contracts we will add 7 additional features, hence the enumeration of the variables starts at 8.

There are three different responses for every observation: monthly pension at 65 years, 75 years and 85 years i.e. how much pension a retiree will get every month if he decides to retire at 65 years, 75 years or 85 years. Every month the employer pays 4-5%[10] of the salary to the total Occupational Pension amount. This means that the monthly pension amount is dependent on the previous salaries of the retiree. Let c denote the total Occupational Pension amount in hundreds of thousands Swedish crowns (sek). To account for this dependency KFB uses $c = 500$ when calculating the monthly pension at age of retirement 65 years, 75 years and 85 years. To get more comprehensible numbers and to create responses that are approximately independent of the assumption $c = 500$ we divide the observed monthly pension by c to define the standardized responses $y^{(65)}$, $y^{(75)}$ and $y^{(85)}$ as shown in Table 2. This means that e.g. $y^{(65)}$ is defined as the monthly pension at retirement age 65 divided by c. When we want to keep the age of retirement unspecified the response is simply denoted by y.

The yearly pension is determined by dividing the total pension by the annuity divisor. Thus the annuity divisor can be calculated by dividing the total pension by the yearly pension. The Occupational Pension is paid equally every month thus the yearly pension is determined by multiplying the monthly pension with 12 months. Therefore, the annuity divisor D for every retirement age used in private insurance companies can be estimated as

$$D^{(65)} = \frac{1000c}{12cy^{(65)}} = \frac{1000}{12y^{(65)}}, \quad D^{(75)} = \frac{1000}{12y^{(75)}}, \quad D^{(85)} = \frac{1000}{12y^{(85)}}. \tag{1}$$

This means that your monthly pension is calculated as $\frac{250c}{3D}$.

[10] Information taken from https://www.pensionsmyndigheten.se/forsta-din-pension/tjanstepension/ta-reda-pa-om-du-har-tjanstepension

Table 1: The selected features

Feature	Name	Unit	Comment
x_8	Expected Remaining Lifetime at Year 65	years	
x_9	Advance Interest Rate	%	expected future interest rate
x_{10}	Fixed Annual Fee	‰	
x_{11}	Return in 2017	%	return from investments of pension savings
x_{12}	Return in 2016	%	
x_{13}	Return in 2015	%	
x_{14}	Return in 2014	%	
x_{15}	Return in 2013	%	
x_{16}	Move Fee	‰	fee to move pension savings to another insurance company
x_{17}	Insurance Fee	%	
x_{18}	Insurance	yes/no	some part of pension savings is insured
x_{19}	Old Savings Covered	yes/no	new fees are applied to old pension savings
x_{20}	Free Consulting	yes/no	
x_{21}	Fund Management Fee	%	insurance company's fee for managing pension funds
x_{22}	Asset Management Fee	%	insurance company's fee for managing pension assets
x_{23}	Expected Remaining Lifetime at Year 75	years	see section 2.2
x_{24}	Expected Remaining Lifetime at Year 85	years	see section 2.2

Table 2: The responses using $c = 500$

Feature	Name	Unit
$500y^{(65)}$	Monthly pension at retirement age 65	sek/month
$500y^{(75)}$	Monthly pension at retirement age 75	sek/month
$500y^{(85)}$	Monthly pension at retirement age 85	sek/month

2.2 Life Expectancy Modeling

We would like to know what happens to $y^{(65)}$, $y^{(75)}$ and $y^{(85)}$ if a change is made in x_8. Let the expected remaining lifetime at age x be denoted by $l(x)$ and the total amount paid to a retiree that retires at age x by $c(x)$. Note that the total amount paid is proportional to the remaining lifetime multiplied by the monthly pension e.g. $c(75) \propto l(75)y^{(75)}$. Note that $l(65) = x_8$ and that every private insurance company chooses their own value for x_8. We assume that the choice of x_8 has an impact on $l(75)$ and $l(85)$ but not on $c(x)$ meaning the total amount paid remains. If we then adjust $l(65) = x_8$ to $l^*(65)$ we will get new values for $l(75)$ and $l(85)$ denoted by $l^*(75)$ and $l^*(85)$ and analogously for $y^{(65)*}$, $y^{(75)*}$ and $y^{(85)*}$. Since we want $c(x)$ to be independent of x_8 we can obtain for instance $y^{(75)*} = \frac{l(75)}{l^*(75)}y^{(75)}$.

The mortality rate at age x denoted by $u(x)$ is sometimes assumed to be exponential at the middle age and above[11]. Data for $u(x)$ in the US has been collected in [2] and is displayed in Figure 2. We will now investigate if we can model $u(x)$ as exponential for $x \geq 40$. To do this we will fit the approximative model $\hat{u}(x) = \hat{\beta}_0 e^{\hat{\beta}_1 x}$ to the data. By taking the logarithm we get $\log u(x) = \log \hat{\beta}_0 + \hat{\beta}_1 x$. An ordinary least squares fit of this can be seen in Figure 3 where $\hat{\beta}_1 = 0.084912$ displaying an almost perfect fit. The exponential assumption is therefore reasonable. We will from now on use $u(x) = \alpha e^{0.085x}$. Additionally we assume this formula holds in Sweden as well. We also assume that the change in expected remaining lifetime $l(x)$ is proportional to the change in mortality rate $u(x)$ i.e. $l'(x) \propto u'(x)$. This means we can model $l(x) = ae^{0.085x} + b$. We use $l(40) = 43.4$, $l(50) = 33.83$ and $l(90) = 4.155$ as fixed points. Each $l(\cdot)$ is taken as the average of the expected remaining lifetime between males and females in Sweden using data from SCB[12] 2017.

Given a value for $l^*(65)$ we can then with linear regression approximate numerical values for a and b to obtain $l^*(x)$. Then it is possible to evaluate $l(75)$, $l(85)$, $l^*(75)$ and $l^*(85)$ (which correspond to the regressors x_{23} and x_{24}) to obtain $y^{(65)*}$, $y^{(75)*}$ and $y^{(85)*}$.

All contract types are combined in the tables shown in Appendix A.1. While in Appendix A.2 new data points for ITP2 have been generated by adjusting x_8 which with the methodology above implies new values for x_{23}, x_{24}, $y^{(65)}$, $y^{(75)}$ and $y^{(85)}$.

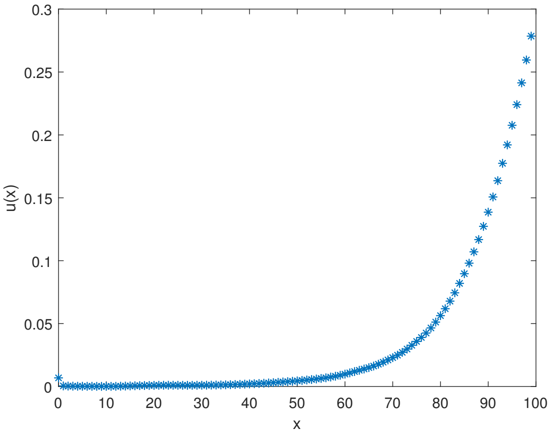


Figure 2: Data from [2] displaying the mortality rate $u(x)$ as a function of the age x

[11] This is known as the Makeham formula

[12] Statistiska Centralbyrån

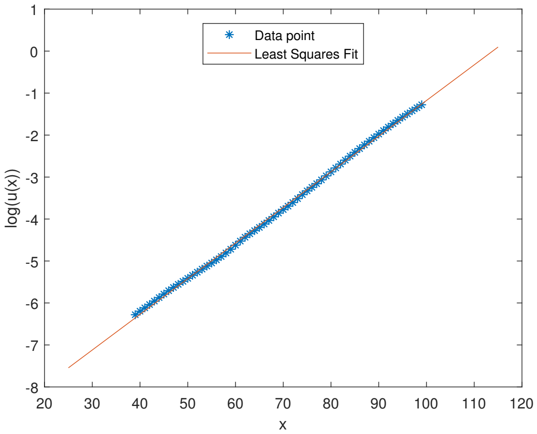


Figure 3: Data from [2] displaying the log mortality rate $\log(u(x))$ as a function of the age x together with a linear least squares fit that illustrates the high accuracy of the model

3 Mathematical Modeling

3.1 Regression Analysis

To limit the scope of methods we consider only methods that yield parametric models. A parametric model contains a finite number of parameters or weights that are determined by minimizing some error on the data set. The final model contains the model assumption and numerical parameter values. An advantage of such a model is that it is better suited for quantitative analysis compared to a nonparametric model. Unlike nonparametric models, parametric models can be explicitly defined as mathematical functions to enable e.g. differentiation and integration.

We will not model the annuity divisor D directly but instead the scaled monthly pension y. Suppose the true relationship is

$$y = f(\boldsymbol{x}, \boldsymbol{\beta}) + \epsilon, \tag{2}$$

where $\boldsymbol{x}$ is the p-dimensional feature vector, $\boldsymbol{\beta}$ is a vector of the unknown model parameters and ϵ is an error term which in some cases are assumed to be normally distributed. The idea is then to find an approximate model $\hat{f}(\boldsymbol{x}, \hat{\boldsymbol{\beta}})$ that as closely as possible follows the true model $f(\boldsymbol{x}, \boldsymbol{\beta})$, i.e.

$$\hat{f}(\boldsymbol{x}, \hat{\boldsymbol{\beta}}) \approx f(\boldsymbol{x}, \boldsymbol{\beta}), \tag{3}$$

by using the data set of n observations $(\boldsymbol{x}_i, y_i)$, $i = 1, \ldots, n$. We have a choice to model all employment contracts separately or simultaneously. If we model them separately we will have to fit $\hat{f}(\cdot)$ to 7 different data sets each corresponding to a contract of type C_i. In that case we will have in its general form

$$y = f(x_8, \ldots, x_{24}, \boldsymbol{\beta}) + \epsilon. \tag{4}$$

To model all contract types simultaneously we will introduce 7 additional regressors as seen in Table 3 and the general form is

$$y = f(x_1, \ldots, x_{24}, \boldsymbol{\beta}) + \epsilon. \tag{5}$$

Keep in mind that we also have three different ages of retirement and each age will require their own model. We will only fit Equation (4) to ITP2 but for all three retirement ages while Equation (5) will be fitted to all contract types simultaneously.

Three different approaches of estimating the model function $f(\cdot)$ are considered, artificial neural networks (ANN), support vector machines for regression (SVR) and multiple linear regression (MLR). Each method will be covered in separate sections but first we will look at an overview of their key differences.

Table 3: Employment's contract binary features

Feature	Name	Unit
x_1	ITP1 - C_1	yes/no
x_2	ITP2 - C_2	yes/no
x_3	SAF-LO - C_3	yes/no
x_4	AKAP-KL - C_4	yes/no
x_5	KAP-KL - C_5	yes/no
x_6	PA-KFS-09 - C_6	yes/no
x_7	PA-03 - C_7	yes/no

3.2 Machine Learning Algorithms vs MLR

ANN, SVR and MLR are all parametric models where the parameters are optimized over the data set. In MLR the model $\hat{f}(\boldsymbol{x}, \hat{\boldsymbol{\beta}})$ is defined such that the regression coefficients appear linearly. For all these approaches an error is defined on the data set and the coefficients are chosen such that the error is minimized. Different error measures are considered in Section 3.3.

The simplicity of the MLR assumption makes the optimization of the regression parameters straightforward and is done in one step. Additionally, under normality assumption of the error term, statistical inferential procedures are available. On the other hand machine learning algorithms such as ANN and SVR are rather complex, especially ANN which is nonlinear in the model parameters.

When designing the models the risk of overfitting must be kept in mind. Overfitting is when the model function follows the data set too closely and not the general pattern meaning that it will perform badly when new data is introduced. How this is countered in each of the methods will be discussed later in their corresponding section. We will specify some pros and cons of the methods.

Ordinary MLR:

+ Full control over model assumptions.

+ Can create a relatively simple model that is easy to understand and work with e.g. the derivative may be easy to compute.

+ Due to the normality assumption of the random error standard statistical inferences can be performed.

- Hard to find a high accuracy model when the underlying function $f(\boldsymbol{x})$ is non-linear. One may have to transform both the response and the features. Some experience may be required.

- To avoid overfitting, one has to be careful not to introduce too many variables. There is not a clear predetermined limit of what is meant by "too many" variables but if $p > n$ the inversion of the design matrix, i.e. $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$, does not exist and therefore the regression coefficients can not be determined.

- If one wants to make statistical inferences validation of the normality assumption has to be made.

ANN:

+ High accuracy.

+ Can essentially follow an arbitrary function f without too much design effort.

+ Easily extended to vector valued output. This is useful if we want to model $y^{(65)}$, $y^{(75)}$ and $y^{(85)}$ simultaneously.

+ Not as sensitive to the number of features due to the large amount of model parameters also known as the "large p small n setting".

- Less control over model assumptions.

- Nontrivial model assumption that may be difficult to work with.

- Unpredictable results due to weight initialization and data partitioning.

- Highly nonlinear optimization which cause high computational complexity.

- Not suited for statistical inference to conclude something about the estimation uncertainty.

SVR:

+ High accuracy.

+ Can essentially follow an arbitrary function without too much design effort.

+ The parameter optimization can be formulated as a quadratic program.

- A suitable kernel function must be found. Sometimes a simple linear kernel will suffice.

- Additional observations may introduce additional support vectors meaning the model may grow with the amount of observations.

- Not suited for statistical inference to conclude something about the estimation uncertainty.

We consider training ANN on two data sets using the three responses $y^{(65)}$, $y^{(75)}$ and $y^{(85)}$. One data set for ITP2 found in Appendix A.2. The second data set for all types of contracts found in Appendix A.1. MLR and SVR will be trained on the data set for all employment contracts using only the response $y^{(65)}$. The reason why we study these two data sets is to see if we can explain all contracts simultaneously or if there should be different models for each contract type.

3.3 Evaluation of the Model Performance Accuracy

When numerical values for the parameters of a regression model $\hat{\boldsymbol{\beta}}$ has been determined the performance of the model $\hat{f}(\boldsymbol{x}, \hat{\boldsymbol{\beta}})$ can be evaluated in many different ways. Imagine we have n samples in the data set and let the ith sample have feature values $\boldsymbol{x}_i$. Furthermore, let $\hat{f}(\boldsymbol{x}_i, \hat{\boldsymbol{\beta}}) = \hat{y}_i$ and the corresponding target be denoted by y_i. Some common error measures for function approximation are

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \quad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \quad \text{RMSE} = \sqrt{\text{MSE}}. \tag{6}$$

If the response is a vector, the error is then calculated elementwise and averaged. MSE is shorthand notation for mean square error while MAE stands for mean absolute error. The MSE is better suited for differentiation than MAE. It may be good to keep in mind that MSE is more sensitive to outliers than MAE. While MSE may be easier to work with MAE is often easier to interpret. RMSE stands for root mean square error and serves the same purpose as MSE except that it has the same unit as the output unlike MSE which has the unit in square. A common measure that is used in the MLR setting is the coefficient of determination R^2 defined as

$$R^2 = 1 - \frac{\text{SS}_{\text{Res}}}{\text{SS}_{\text{T}}}, \quad \text{SS}_{\text{Res}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \quad \text{SS}_{\text{T}} = \sum_{i=1}^{n}(y_i - \bar{y})^2, \tag{7}$$

where $\bar{y}$ is the mean of the response. Observe that $R^2 \in [0, 1]$ and should be maximized.

3.4 Cross Validation

For a lot of regression and machine learning techniques the error on the data set used for training, i.e. training error, can in practice be reduced to zero. For instance in MLR you can introduce as many regression coefficients as observations to obtain a perfect fit. In such a case overfitting has most likely occurred. Therefore, only looking at the training error can be misleading. What one really wants to know is how well the model performs in the future on new data, i.e. how well the method generalizes. The error on new unseen data is referred to as the test error. A method with a smaller test error is generally preferred. There are two main issues connected to determining the test error. Firstly obtaining new data may be difficult or sometimes even impossible therefore the data set is usually split into a training set $\mathcal{S}_{\text{train}}$ and a test set $\mathcal{S}_{\text{test}}$. Where $\mathcal{S}_{\text{train}}$ is used to determine the values for the model parameters and $\mathcal{S}_{\text{test}}$ is used to estimate the test error. The second issue is that the data used to estimate the test error could have been used in the training set to improve the fit. Hence the test error may be overestimated. Regardless of this critique the test error can still be a valuable measure and we will consider three ways of estimating it.

3.4.1 Leave One Out Cross Validation

The idea behind leave one out cross validation (abbreviated LOOCV) is as follows. Assume that we have n observations of the form $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)$. Let the training set include all except the ith observation i.e. $\mathcal{S}_{\text{test}} = \{(\boldsymbol{x}_i, y_i)\}$. The model is then trained on $\mathcal{S}_{\text{train}}$ and the test error is calculated on $\mathcal{S}_{\text{test}}$. The test error (MSE) is calculated by leaving out the ith observation and defined as

$$\mathrm{MSE}_i = (y_i - \hat{y}_i)^2, \tag{8}$$

where the hat indicates it is the fitted value obtained from the trained model on the training set. The LOOCV is calculated as the average of all MSE_i where $i = 1, \ldots, n$. Formally we define LOOCV as

$$\mathrm{CV}_{(n)} = \mathrm{LOOCV} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{MSE}_i. \tag{9}$$

In the case of MLR, LOOCV may be calculated as

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2, \tag{10}$$

where h_i is the leverage defined as the ith diagonal element of the projection matrix $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ and $\boldsymbol{X}$ is the design matrix defined as

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & \ldots & x_{p1} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \ldots & x_{pn} \end{bmatrix}, \tag{11}$$

where x_{ij} is the jth observation of the ith feature.

3.4.2 k-fold Cross Validation

This approach splits the data randomly into k approximately equally sized groups $\mathcal{S}_1$, $\mathcal{S}_2$, $\ldots$, $\mathcal{S}_k$ also known as folds. Define the integer division between n and k as $n_f = n \backslash k$ and the remainder as r such that $n = kn_f + r$. Let the number of elements in $\mathcal{S}_i$ be n_f if $i \leq k - r$ and $n_f + 1$ if $i > k - r$. The model is trained on $k - 1$ of the folds and the test set will consist of the remaining fold. Similarly to LOOCV MSE_i is calculated on the test set but instead of the training set excluding observation i it now excludes fold i. The k-fold cross validation is defined as

$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i. \tag{12}$$

Notice that k-fold cross validation is identical to LOOCV if $k = n$.

3.4.3 TVT Partitioning

When considering methods that iteratively adjust parameters to improve the training performance it is desired to stop adjusting the parameters at some point before overfitting occurs. The idea is then to create an additional data set called the validation set $\mathcal{S}_{\mathrm{val}}$. Each time the parameters are adjusted to reduce the train error the validation error is calculated. If the validation error does not decrease over multiple iterations the optimization procedure stops. We now have three sets abbreviated as TVT: training, validation and test. Finally the test error is calculated on the test set, sometimes referred to as the TVT error. The partitioning is random and typically training consists of 70% of the data, validation 15% and test 15%. To counter the randomness of this approach we will partition the data set and train the method multiple times and calculate the average test error.

3.4.4 Comparison Between the Test Error Estimates

Some pros and cons of the previously mentioned test error estimates are
LOOCV:

+ Predictable result. You will get the same test error every time i.e. non random result.

+ Low risk of overestimating the test error due to only using a single observation in each calculation.

- Highly computationally requiring except for the MLR case where one can apply Equation (10).

k-fold CV:

+ Less computationally requiring than LOOCV.

- Result is random due to random data splitting.

TVT partitioning:

+ Good counter measure against overfitting when considering methods that adjust model parameters iteratively.

+ Less computationally requiring than LOOCV.

- Result is random due to random data splitting.

- Even less points used in training due to introducing an additional data set $\mathcal{S}_{\text{val}}$ which increases the risk of overestimating the test error.

3.5 Feature Scaling

In practice it is common that features have different units and scales. For instance one feature may be the age of a person in years which may be in the range $[20, 60]$ while another feature may be their yearly income in SEK which is usually of order hundreds of thousands. This may cause different types of problems depending on the choice of method and as a result it may be a good idea to consider feature scaling. We will later discuss what the specific effects are for our methods but for future reference we formulate two common ways of scaling. Imagine we have a feature x and that we have observed this feature multiple times. Let x_{min} and x_{max} be the smallest and largest observed values respectively. Let μ_x and σ_x be the mean and standard deviation of the observed values. Finally, let $\hat{x}$ be the scaled feature. Then we define

$$\hat{x}_{\text{norm}} = \frac{x - \mu_x}{\sigma_x} = \text{mapstd}(x), \tag{13}$$

$$\hat{x}_{\text{minmax}} = \frac{2x - x_{\text{max}} - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} = \text{mapminmax}(x, -1, 1). \tag{14}$$

Equation (13) is referred to as normalization or standardization and is centered around the mean. Equation (14) is a linear transformation such that the largest and smallest values correspond to $\hat{x} = 1$ and $\hat{x} = -1$ respectively. If we have multiple features the minmax transformation will ensure that all features lie within the same interval $[-1, 1]$.

3.6 Dimensionality Reduction

Suppose we wish to visualize n observations of p features. One way of doing this is to create multiple 2D scatter plots. In order to view all combinations of features one would have to create $\binom{p}{2} = p(p-1)/2$ scatter plots which for $p = 24$ equals 276 plots! Furthermore, if we wish to visualize a regression surface in detail we are only allowed to have $p = 2$ features. The need to in some cases perform dimensionality reduction is therefore motivated. The method we will consider is called principal component analysis or PCA for short.

3.6.1 Principal Component Analysis

PCA is an orthogonal transformation of the feature space. PCA creates as many new variables, called principal components, as the initial dimension size. In essence, the components are ordered in such a way that the first principal components explains most of the variance in the data, the second component explains second most etc. This means that if we want to reduce the feature dimension down to 2 we should keep the first two principal components. Formally it works as follows.

Let $\hat{\boldsymbol{X}}$ be the design matrix defined in Equation (11) without the first column of ones and centered such that each column has zero mean, in other words

$$\hat{\boldsymbol{X}} = \begin{bmatrix} x_{11} - \bar{x}_1 & \dots & x_{p1} - \bar{x}_p \\ \vdots & & \vdots \\ x_{1n} - \bar{x}_1 & \dots & x_{pn} - \bar{x}_p \end{bmatrix}, \tag{15}$$

where $\bar{x}_i$ is the mean of the ith feature. We are now going to consider the first principal component. Define the vector $\boldsymbol{t}_1 = \begin{bmatrix} t_{11} & \dots & t_{1n} \end{bmatrix}^T$ as

$$\boldsymbol{t}_1 = \hat{\boldsymbol{X}}\boldsymbol{w}_1, \tag{16}$$

where $\boldsymbol{w}_1 = \begin{bmatrix} w_{11} & \dots & w_{1p} \end{bmatrix}^T$ is referred to as the first principal component and is chosen such that the variance of $\boldsymbol{t}_1$ denoted by σ_1^2 is maximized. Formally we have $\sigma_1^2 = \frac{1}{n-1}\sum_{i=1}^n (t_{1i} - \mu_1)^2$ where $\mu_1 = \frac{1}{n}\sum_{i=1}^n t_{1i}$. Remember that the columns of $\hat{\boldsymbol{X}}$ has mean zero which gives us $\mu_1 = 0$. We have to restrict the length of $\boldsymbol{w}_1$ otherwise the variance can be made arbitrarily large. Specifically we will set it to unit length i.e. $||\boldsymbol{w}|| = 1$. To find $\boldsymbol{w}_1$ we want to solve

$$\boldsymbol{w}_1 = \underset{||\boldsymbol{w}||=1}{\arg\max}\ \sigma_1^2(\boldsymbol{w}) = \underset{||\boldsymbol{w}||=1}{\arg\max}\ \{\sum_{i=1}^n t_{1i}^2\}, \tag{17}$$

or in matrix form

$$\boldsymbol{w}_1 = \underset{\boldsymbol{w}}{\arg\max}\ \frac{||\hat{\boldsymbol{X}}\boldsymbol{w}||^2}{||\boldsymbol{w}||} = \underset{\boldsymbol{w}}{\arg\max}\ \frac{\boldsymbol{w}^T\hat{\boldsymbol{X}}^T\hat{\boldsymbol{X}}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{w}} := \underset{\boldsymbol{w}}{\arg\max}\ R(\boldsymbol{w}). \tag{18}$$

Define the symmetric matrix $\boldsymbol{M} = \hat{\boldsymbol{X}}^T\hat{\boldsymbol{X}}$. Note that $\hat{\boldsymbol{X}}$ is of size $n \times p$ which implies that $\boldsymbol{M}$ is of size $p \times p$ and observe that

$$R(\boldsymbol{w}) = \frac{\boldsymbol{w}^T\boldsymbol{M}\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{w}}, \tag{19}$$

is a Rayleigh quotient [3] which is maximized when $\boldsymbol{w} = \boldsymbol{v}_{\max}$ where $\boldsymbol{v}_{\max}$ is the eigenvector corresponding to the largest eigenvalue $\lambda_{\max}$ of $\boldsymbol{M}$ which is also the attained maximum value. Therefore we have $\boldsymbol{w}_1 = \boldsymbol{v}_1$ and $\sigma_1^2 = \lambda_1$ where λ_1 is the largest eigenvalue of $\hat{\boldsymbol{X}}^T\hat{\boldsymbol{X}}$ and $\boldsymbol{v}_1$ is the corresponding eigenvector. For component $1 < k \leq p$ we define

$$\hat{\boldsymbol{X}}_k = \hat{\boldsymbol{X}} - \sum_{i=1}^{k-1} \hat{\boldsymbol{X}}\boldsymbol{w}_i\boldsymbol{w}_i^T, \tag{20}$$

then the kth principal component is given by $\boldsymbol{w}_k = \boldsymbol{v}_k$ where $\boldsymbol{v}_k$ is the eigenvector corresponding to the largest eigenvalue λ_k of $\hat{\boldsymbol{X}}_k^T\hat{\boldsymbol{X}}_k$.

Suppose you are interested in reducing the feature dimension from p down to k. The data set represented by $\hat{\boldsymbol{X}}$ is then transformed into a data set represented by $\tilde{\boldsymbol{X}}$ calculated as

$$\tilde{\boldsymbol{X}} = \hat{\boldsymbol{X}} \begin{bmatrix} \boldsymbol{w}_1 & \dots & \boldsymbol{w}_k \end{bmatrix}, \tag{21}$$

and the total explained variance in percent σ_{exp}^2 by this transformation is defined as

$$\sigma_{\text{exp}}^2 = 100\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^p \lambda_i}. \tag{22}$$

Similarly a new observation $\boldsymbol{x}$ of all the p features is transformed into $\tilde{\boldsymbol{x}}$ of length k by

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{w}_1^T \\ \vdots \\ \boldsymbol{w}_k^T \end{bmatrix} \boldsymbol{x}. \tag{23}$$

3.7 The Annuity Divisor for the Premium Pension

In this section, the annuity divisor model from the Swedish Pensions Agency will be studied in order to do a comparison between our models and theirs. The model is described in the Orange Rapport [4] and used to calculate the annuity divisor for the Premium Pension in Sweden. The annuity divisor D is defined as

$$D(x) = \int_0^\infty e^{-\delta t}\frac{l(x+t)}{l(x)}dt, \tag{24}$$

$$\delta = ln(1+r) - \epsilon, \tag{25}$$

$$l(x) = e^{-\int_0^x (1-s)\mu(t)dt}, \tag{26}$$

$$\mu(x) = \begin{cases} a + be^{cx} & \text{for } x \leq 100, \\ \mu(100) + (x-100)\cdot 0.01 & \text{for } x > 100, \end{cases} \tag{27}$$

with variables defined as in Table 4. This annuity divisor formula will be used to do a comparison in Section 7.5 with other models. Observe that the notation follows the one in the Orange Rapport and is only used in this section.

Table 4: List of variables in the Premium Pension formula from Swedish Pensions Agency

Variable	Comment
D	Annuity divisor
x	Exact age at time of calculation
r	Interest rate
ϵ	Interest intensity of operating costs
s	Mortality charge
$l(x)$	Survival function used in the calculation of inkomst pension NDC
$\mu(x)$	Mortality function (or Makeham's formula) used for calculating the risk of death within one year
a, b, c	Statistics Sweden forecast of remaining life expectancy in the year 2015-2110 for individuals born in 1938, 1945 and 1955.

3.8 Previous Research

A similar study has been done by Tommy Lowén at the Swedish Pensions Agency in 2016. The annuity divisor is estimated by using data from the website KFB and regression analysis. Following our notations the annuity divisor D is modeled using a log linear relationship

$$\ln(D) = \beta_0 + \beta_1 \ln(x_9) + \beta_2 \ln(x_8) + \epsilon, \tag{28}$$

where ϵ is an error term. Note that only the expected remaining lifetime at 65 x_8 and advance interest rate x_9 are used. Here the interpretations of β_1 and β_2 are how much the advance interest rate and expected remaining lifetime, given that the other variable in the model is fixed affect the annuity divisor respectively. More about the results of this annuity divisor model will be shown in Section 7.5.

4 Artificial Neural Networks

4.1 Overview

The first algorithm we focus on to estimate $f(\cdot)$ is called Artificial Neural Networks, ANN for short. This method is to some degree based on the biological neural networks in the brain. Mathematically speaking ANN is a parametric nonlinear regression model.

An ANN can be divided into four parts: inputs, hidden layers, output layers and outputs. These parts can be combined and used in an enormous amount of ways, therefore to be able to say something more specific we will look at a special kind of ANN. In this thesis the ANN that is used will consist of one input, one hidden layer, one output layer and one output. We will also limit the flow of information to only go forward i.e. the feed forward network will be considered. One may think these restrictions might greatly reduce the usefulness of the network but in practice such a network is still very powerful and also commonly used for fitting regression models. One should however keep in mind that due to the nonlinear optimization problem large data sets are computationally intensive.

4.2 Universal Approximation Theorem

The Russian mathematician Kolmogorov showed in 1957 that continuous functions of multiple arguments can always be expressed as a finite sum of single variable functions.

The theorem reads

Theorem 1 *For any continuous function $f(\boldsymbol{x})$ defined on a compact subset $\Omega \subset \mathbb{R}^p$ and any $\epsilon > 0$ there exists an integer n_h and constants $b_0 \ldots, b_{n_h}$ and $w_{01}, \ldots, w_{n_h p}$ such that we may define*

$$F(\boldsymbol{x}) = b_0 + \sum_{i=1}^{n_h} b_i \sigma \left(w_{i0} + \sum_{j=1}^{p} w_{ij} x_j \right), \tag{29}$$

as an approximation of $f(\boldsymbol{x})$ where $\sigma(\cdot)$ is a sigmoidal activation function. The approximation satisfies

$$|F(\boldsymbol{x}) - f(\boldsymbol{x})| < \epsilon, \tag{30}$$

for all $\boldsymbol{x} \in \Omega$.

A sigmoidal function is bounded, differentiable and satisfies $\frac{d}{dx}\sigma(x) \geq 0 \, \forall x \in \mathbb{R}$. We say that $\sigma(\cdot)$ is symmetric if $\sigma(x) + \sigma(-x) = 1$ and asymmetric if $\sigma(x) + \sigma(-x) = 0$. The variable n_h is referred to as the number of hidden layer nodes.

In words, a feedforward network with a single layer is sufficient to represent any function, but the layer may be infeasibly large and may fail to learn and generalize correctly [5].

4.3 Preprocessing

The first step in a neural network is data preprocessing. This includes removing constant features and scaling the remaining features. The reason why constant features are removed is because they provide no additional information to the problem and would have had the same effect as bias which is already included in the model. We consider scaling methods such as standardization and mapping the maximum and minimum to +1 and -1 respectively as described in Section 3.5. The reason why feature scaling is important in neural networks is because the adjustment of the weight elements in each training epoch is proportional to the gradient of the error. If the features are unscaled the gradient will have different lengths in different directions and this may impact the convergence. It is said that gradient descent methods converges much faster with feature scaling compared to without [6].

4.4 Input to Output

The input is a vector $\boldsymbol{x}$ consisting of p elements corresponding to, in our case, the different features about the pension. The output is a vector $\hat{\boldsymbol{y}}$ consisting of m elements. If we model the three ages of retirement simultaneously we have $m = 3$. The elements in the network are iteratively optimized such that the observed features will yield outputs as close as possible to the observed responses $\boldsymbol{y}$. Each input x_i is first sent through a preprocessing function $g(x_i)$ which outputs $\hat{x}_i$. Then $\hat{x}_i$ is sent to the hidden layer where a linear combination of the elements is to be created by matrix multiplication with a weight matrix $\boldsymbol{W}_h$ of the size $n_h \times p$ with elements denoted by w_{ij}^h. This results in a new vector of length n_h. A bias vector $\boldsymbol{b}_h$ of length n_h with elements denoted by b_i^h is then added, the result is denoted by $\boldsymbol{x}_h$. Formally we have

$$\hat{\boldsymbol{x}} = g(\boldsymbol{x}), \tag{31}$$

$$\boldsymbol{x}_h = \boldsymbol{W}_h \hat{\boldsymbol{x}} + \boldsymbol{b}_h. \tag{32}$$

The index h here indicates that it is a part of the hidden layer. With this notation we say that the hidden layer has n_h nodes. The vector $\boldsymbol{x}_h$ is sent through an activation function (or transfer function) $\sigma(\cdot)$ which is typically sigmoid. Some examples of transfer functions are

$$\begin{aligned} \tanh(n) &= \frac{2}{1+e^{-2n}} - 1, \\ \text{logsig}(n) &= \frac{1}{1+e^{-n}}, \\ \text{purelin}(n) &= n. \end{aligned} \tag{33}$$

The transfer function will be applied elementwise and we denote the output of the hidden layer by $\sigma_h(\boldsymbol{x}_h)$. This vector is then passed through the output layer in a similar fashion as in the hidden layer except for potentially a different transfer function $\sigma_o(\cdot)$ and with m layer nodes. The output can then be explicitly defined as

$$\boldsymbol{y} = \sigma_o\left(\boldsymbol{W}_o \sigma_h(\boldsymbol{W}_h g(\boldsymbol{x}) + \boldsymbol{b}_h) + \boldsymbol{b}_o\right), \tag{34}$$

where $\boldsymbol{W}_o$ is a weight matrix of size $m \times n_h$ and $\boldsymbol{b}_o$ is a bias vector of length m. Observe that if $\sigma_o(n) = \text{purelin}(n)$ we get the same model as in Equation (29).

The model now includes multiple unknown parameters. The unknown parameters are the elements in the weight matrices and biases, which means we have $m+mn_h+n_h+n_hp = n_h(1+p+m)+m$ parameters to determine. They will be "learned" by training with examples. See Figure 4 and Figure 5 for a simplified and detailed overview respectively of a single layer feed forward neural network using our notation.

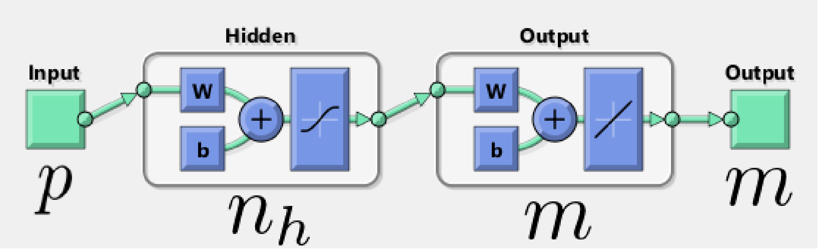


Figure 4: Overview of a feed forward artificial neural network

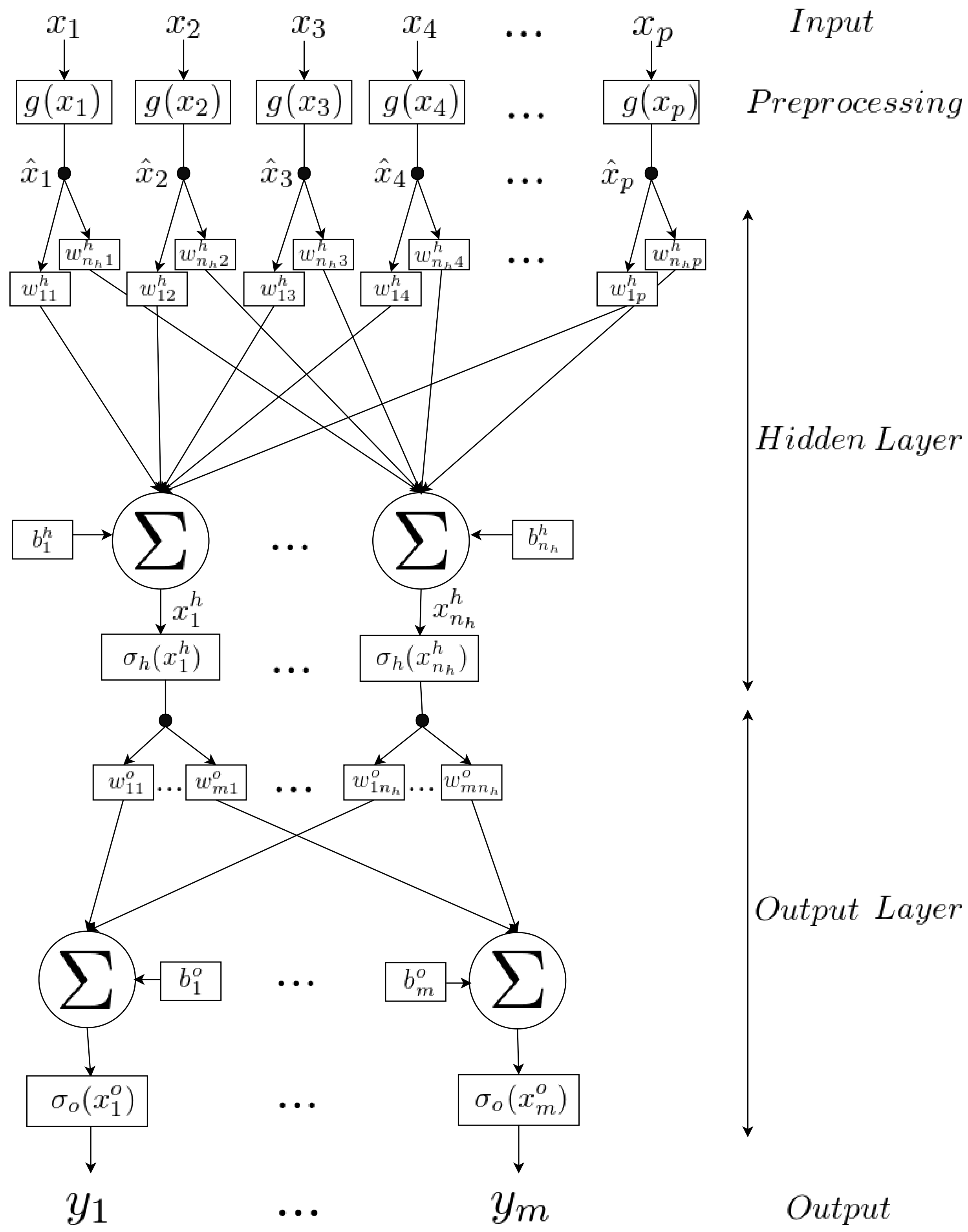


Figure 5: Representation of a single hidden layer, feed-forward neural network where Σ indicates a summation node. The output of a summation node is the sum of the inputs

4.5 Training the Network

The relationship between the error as defined in Section 3.3 and the weights of the network is nontrivial and non convex. This makes the optimization problem tricky to solve and approximate local optimality will have to suffice meaning an explicit solution will not be determined. Instead the error will be reduced iteratively until convergence and the final error will then unfortunately depend on the initial values of the weights and also on the partitioning of the data as will be seen later.

In principle the error is calculated using the current values for the weights. The gradient of the error is calculated using an algorithm called backpropagation. The gradient is then used to determine the optimal weight adjustment $\boldsymbol{\delta}$ and the process restarts and continues until convergence. Specifically the Levenberg-Marquardt algorithm [7] is considered to find the weight adjustment $\boldsymbol{\delta}$. Assume we want to minimize the sum of square deviations

$$S(\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{n} \left(y_i - \hat{f}(\boldsymbol{x}_i, \hat{\boldsymbol{\beta}}) \right)^2 . \tag{35}$$

If the weights $\hat{\boldsymbol{\beta}}$ are adjusted to $\hat{\boldsymbol{\beta}} + \boldsymbol{\delta}$ and $\hat{f}(\cdot)$ is approximated by its linearizion we obtain

$$S(\hat{\boldsymbol{\beta}} + \boldsymbol{\delta}) \approx \sum_{i=1}^{n} \left(y_i - \hat{f}(\boldsymbol{x}_i, \hat{\boldsymbol{\beta}}) - (\nabla_{\hat{\boldsymbol{\beta}}} \hat{f}(\boldsymbol{x}_i, \hat{\boldsymbol{\beta}}))^T \boldsymbol{\delta} \right)^2 , \tag{36}$$

where $\nabla_{\hat{\boldsymbol{\beta}}}$ denotes the gradient with respect to $\hat{\boldsymbol{\beta}}$. Denote the Jacobian matrix of $\hat{f}(\boldsymbol{x}_i, \hat{\boldsymbol{\beta}})$ with respect to $\hat{\boldsymbol{\beta}}$ by $\boldsymbol{J}_{\hat{\boldsymbol{\beta}}}$ and note that

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{f}(\boldsymbol{x}_1, \hat{\boldsymbol{\beta}}) \\ \vdots \\ \hat{f}(\boldsymbol{x}_n, \hat{\boldsymbol{\beta}}) \end{bmatrix} , \tag{37}$$

we can then formulate Equation (36) in matrix form as

$$S(\hat{\boldsymbol{\beta}} + \boldsymbol{\delta}) \approx (\boldsymbol{y} - \hat{\boldsymbol{y}})^T (\boldsymbol{y} - \hat{\boldsymbol{y}}) - 2(\boldsymbol{y} - \hat{\boldsymbol{y}})^T \boldsymbol{J}_{\hat{\boldsymbol{\beta}}} \boldsymbol{\delta} + \boldsymbol{\delta}^T \boldsymbol{J}_{\hat{\boldsymbol{\beta}}}^T \boldsymbol{J}_{\hat{\boldsymbol{\beta}}} \boldsymbol{\delta}. \tag{38}$$

Differentiating this expression w.r.t. $\boldsymbol{\delta}$ and setting the result to zero yields

$$\boldsymbol{J}_{\hat{\boldsymbol{\beta}}}^T \boldsymbol{J}_{\hat{\boldsymbol{\beta}}} \boldsymbol{\delta} = \boldsymbol{J}_{\hat{\boldsymbol{\beta}}}^T (\boldsymbol{y} - \hat{\boldsymbol{y}}), \tag{39}$$

in the Levenberg-Marquardt algorithm this expression is replaced with the "damped version"

$$(\boldsymbol{J}_{\boldsymbol{\beta}}^T \boldsymbol{J}_{\boldsymbol{\beta}} + \mu \boldsymbol{I}) \boldsymbol{\delta} = \boldsymbol{J}_{\boldsymbol{\beta}}^T (\boldsymbol{y} - \hat{\boldsymbol{y}}), \tag{40}$$

where μ in this section refers to a non negative damping factor and $\boldsymbol{I}$ is the identity matrix. When μ is small the method is similar to Newton's method while μ is large it is similar to gradient descent. Newton's method is faster and more accurate near the error minimum and the idea is therefore to try and reduce μ after each step if it would result in a decrease of $S(\hat{\boldsymbol{\beta}})$.

TVT partitioning as explained in Section 3.4.3 will be used meaning the Levenberg-Marquardt algorithm continues until convergence or until the error on the validation set has not improved over 5 iterations. This means that any reported error/accuracy using ANN in this thesis is actually an estimate of the test error/accuracy.

4.6 Weights Initialization

The weights and biases has to be initialized in some way in order to start the learning process. A neural network transfer function usually yields values approximately in $[-1, 1]$ and the same goes for the preprocessing scaling functions. It is therefore reasonable for the weights to be in a similar range. We consider two initialization methods. The first one is called Nguyen-Widrow initialization (NW) and the second is called random symmetric initialization (RS). NW is a more sophisticated method and is supposed to be overall better than the purely random RS. Consider a layer with $s > 1$ nodes and a transfer function with an active region $[x_1, x_2]$. For instance the active region for $\tanh(x)$ is $x \in [-2, 2]$ and for satlins(x) is $x \in [-1, 1]$ as defined in Table 7. NW requires a finite active region, e.g. purelin(x) is not allowed, and works in the following steps:

1. Calculate $\alpha = 0.7s^{1/n}$, $n_1 = \frac{x_2 - x_1}{2}$ and $n_2 = \frac{x_2 + x_1}{2}$.
2. Generate a matrix of size $s \times n$ with randomized elements from the uniform distribution $U(-1, 1)$. Rescale the elements such that the row vectors of the matrix have unit length. Denote the resulting matrix by $\hat{\boldsymbol{W}}$.
3. Let $\boldsymbol{W}_s = \alpha \hat{\boldsymbol{W}}$.
4. Define a vector $\hat{\boldsymbol{b}}$ elementwise as $\hat{b}_i = t_i \alpha \operatorname{sign}(w_i)$ for $i = 1, \ldots, s$. Where w_i is the ith element of the first column in $\boldsymbol{W}_s$ and $t_i = \frac{1+s-2i}{1-s}$.
5. The weight matrix is then defined as $\boldsymbol{W} = n_1 \boldsymbol{W}_s$.
6. The bias vector is defined as $\boldsymbol{b} = n_1 \hat{\boldsymbol{b}} + n_2$.

RS on the other hand generates its elements directly from $U(-1, 1)$.

4.7 Tunable Network Properties

A lot of decisions has to be made when designing a neural network. Some of the most important choices are the number of nodes and transfer function for the hidden layer. Additionally the initial randomization of the weights can be done in different ways, the summation nodes can be replaced with multiplication nodes and there are multiple preprocessing functions to consider. The optimal set of design choices depend on the data set and will be determined by comparing different resulting performances. The design choices considered in this report are the number of nodes and transfer function of the hidden layer, initialization method and preprocessing functions. The full list of transfer functions can be seen in Table 5 to Table 7, the initialization methods are defined in Section 4.6 and the preprocessing functions are defined in Section 3.5.

We will not consider scaling of the target when working with ANN and therefore we need an output transfer function that can yield any real number. This means we have to use purelin or netinv. For simplicity and to obtain a network model as in Equation (29) we use purelin. For the hidden layer any of the transfer functions can be used. The question is which transfer function will give us the highest accuracy. To answer this each transfer function will be tested in the hidden layer and a performance average is calculated. The optimal number of nodes in the hidden layer depend on the transfer function which means we have to change the nodes when testing each function.

When the best transfer function and number of nodes for the data has been determined the preprocessing function and weight initialization method is tested in a similar fashion.

Table 5: Neural network layer transfer functions. The input is denoted by $\boldsymbol{n}$ and the output by $\boldsymbol{a}$. The ith element is denoted by n_i and a_i respectively for $i = 1, \ldots, k$. Part I/III

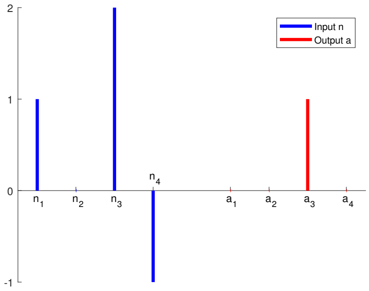
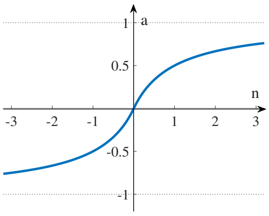
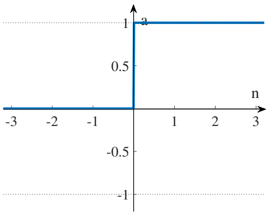
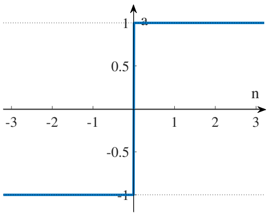
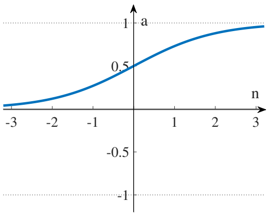
Name	Full Name	Mathematical Formula	Graph of Transfer Function
compet	Competitive	$\boldsymbol{a} \cdot \boldsymbol{n} = \max\limits_{i} n_i$	
elliotsig	Elliot sigmoid	$a_i = \dfrac{n_i}{1 + \lvert n_i \rvert}$	
hardlim	Positive hard limit	$a_i = \begin{cases} 1 & \text{if } n_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$	
hardlims	Symmetric hard limit	$a_i = \begin{cases} 1 & \text{if } n_i \geq 0 \\ -1 & \text{otherwise} \end{cases}$	
logsig	Logarithmic sigmoid	$a_i = \dfrac{1}{1 + e^{-n_i}}$	

Table 6: Neural network layer transfer functions. The input is denoted by $\boldsymbol{n}$ and the output by $\boldsymbol{a}$. The ith element is denoted by n_i and a_i respectively for $i = 1, \ldots, k$. Part II/III

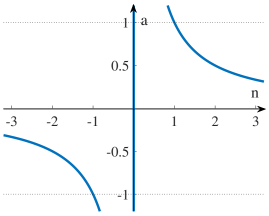
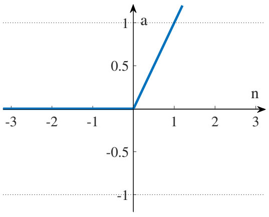
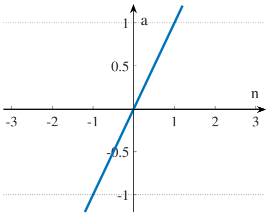
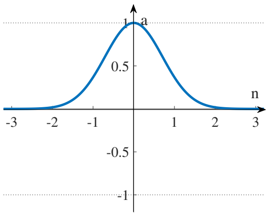
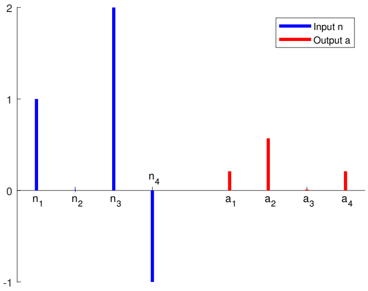
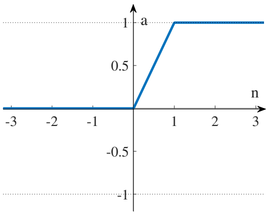
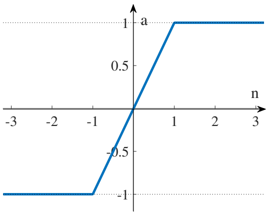
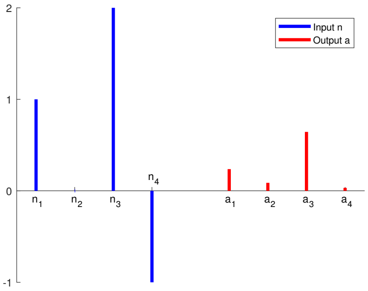
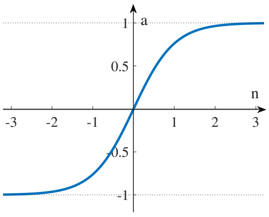
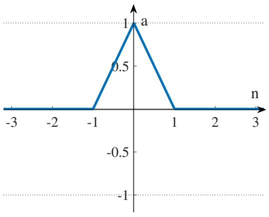
Name	Full Name	Mathematical Formula	Graph of Transfer Function
netinv	Inverse	$a_i = \dfrac{1}{n_i}$	
poslin	Positive linear	$a_i = \begin{cases} n_i & \text{if } n_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$	
purelin	Linear	$a_i = n_i$	
radbas	Radial basis	$a_i = e^{-n_i^2}$	
radbasn	Radial basis normalized	$a_i = \dfrac{e^{-n_i^2}}{e^{-n_1^2} + \ldots + e^{-n_k^2}}$	

Table 7: Neural network layer transfer functions. The input is denoted by $\boldsymbol{n}$ and the output by $\boldsymbol{a}$. The ith element is denoted by n_i and a_i respectively for $i = 1, \ldots, k$. Part III/III

Name	Full Name	Mathematical Formula	Graph of Transfer Function
satlin	Positive saturating linear	$a_i = \begin{cases} 1 & \text{if } n_i \geq 1 \\ n_i & \text{if } 0 \leq n_i < 1 \\ 0 & \text{otherwise} \end{cases}$	
satlins	Sym. saturating linear	$a_i = \begin{cases} 1 & \text{if } n_i \geq 1 \\ n_i & \text{if } 0 \leq n_i < 1 \\ -1 & \text{otherwise} \end{cases}$	
softmax	Soft max	$a_i = \dfrac{e^{n_i}}{e^{n_1} + \ldots + e^{n_k}}$	
tansig	Hyperbolic tangent	$a_i = \dfrac{2}{1 + e^{-2n_i}} - 1$	
tribas	Triangular basis	$a_i = \begin{cases} 1 - \lvert n_i \rvert & \text{if } -1 \leq n_i \leq 1 \\ 0 & \text{otherwise} \end{cases}$	

4.8 Feature Selection

In order to determine which features that can explain the response most efficiently the network will first be trained with all features. The network is created and trained multiple times to obtain an averaged MSE. Then one feature at the time is set constant such that it is removed in the preprocessing step. The network is then created and trained multiple times again to calculate the new averaged MSE. The removed feature that resulted in the lowest increase or highest decrease of the average MSE is removed from the model. The process then restarts and stops before the average MSE has grown significantly or when the number of features is at a desired level. The standard deviation of the MSE is calculated to measure the reliability of the model. This means that the average MSE is used to evaluate the feature importance unless the standard deviation is exceptionally large.

5 Support Vector Machines

The second algorithm we consider in this study is support vector machines (SVM) which was initially developed for classification problems. The basic idea is to transform the data to higher dimension where it can be linearly separated by a hyperplane. The hyperplane is found by maximizing its margin i.e. its distance to the closest data points. The result can be expressed as a set of vectors called support vectors.

5.1 Duality Principle

The duality principle can be used to transform a nonlinear inequality constrained optimization problem into a quadratic inequality constrained optimization problem using Lagrange multipliers. Suppose we want to minimize a nonlinear convex objective function $f(\boldsymbol{x})$ with k nonlinear inequality constraints formally stated as

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, k. \end{aligned} \tag{41}$$

Introduce the Lagrangian

$$\mathcal{L}_p(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{k} \lambda_i g_i(\boldsymbol{x}), \tag{42}$$

where the subscript p indicates that it is the *primal* Lagrangian and λ_i in this section are the nonnegative Lagrange multipliers. Define locally the vector variable $\boldsymbol{h}$ as

$$\boldsymbol{h}(\boldsymbol{\lambda}) = \arg \min_{\boldsymbol{x}} \mathcal{L}_p(\boldsymbol{x}, \boldsymbol{\lambda}), \tag{43}$$

in other words, $\boldsymbol{x} = \boldsymbol{h}$ minimizes the primal Lagrangian. An additional Lagrangian is introduced as

$$\mathcal{L}_d(\boldsymbol{\lambda}) = \mathcal{L}_p(\boldsymbol{h}(\boldsymbol{\lambda}), \boldsymbol{\lambda}), \tag{44}$$

where the subscript d indicates that it is the *dual* Lagrangian. Due to convexity $\boldsymbol{h}$ satisfies $\nabla_{\boldsymbol{x}} \mathcal{L}_p(\boldsymbol{x}, \boldsymbol{\lambda})|_{\boldsymbol{x}=\boldsymbol{h}} = 0$. Let $\boldsymbol{x}^*$ denote the optimal solution to Equation (41). The duality principle then states that $\boldsymbol{x}^* = \boldsymbol{h}(\boldsymbol{\lambda}^*)$ where $\boldsymbol{\lambda}^*$ is the solution to the quadratic dual problem

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{maximize}} \quad & \mathcal{L}_d(\boldsymbol{\lambda}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0. \end{aligned} \tag{45}$$

This problem can in some cases be easier than solving Equation (41).

5.2 Support Vector Regression

Support vector machines can be extended to the regression setup. In linear regression and in support vector regression (SVR) we want to find optimal model parameters to the approximative model

$$\hat{f}(\boldsymbol{x}) = \beta_0 + \boldsymbol{x}^T\hat{\boldsymbol{\beta}}, \tag{46}$$

to a data set of points $(\boldsymbol{x}_i, y_i)$ for $i = 1, \ldots, n$ where $\boldsymbol{x}, \boldsymbol{x}_i \in \mathbb{R}^p$ are column vectors. An error or loss function $L(y, \hat{f}(\boldsymbol{x}))$ can be defined in multiple ways. In least squares regression the following loss function is used

$$L_0(y, \hat{f}(\boldsymbol{x})) = \left(y - \hat{f}(\boldsymbol{x})\right)^2, \tag{47}$$

while in SVR we often use

$$L_1(y, \hat{f}(\boldsymbol{x})) = \max\{0, |y - \hat{f}(\boldsymbol{x})| - \epsilon\}, \tag{48}$$

$$L_2(y, \hat{f}(\boldsymbol{x})) = \max\{0, (y - \hat{f}(\boldsymbol{x}))^2 - \epsilon\}, \tag{49}$$

where ϵ in SVR is a parameter controlling the width of the margin where no penalty is given to errors. Sometimes $\epsilon = 0.1$ is used. It can be shown that only data points outside the margin produce support vectors meaning a smaller value of ϵ will increase the number of support vectors used in the model and thus increase the flexibility of the fit. The max function is defined as

$$\max\{a, b\} = \begin{cases} a & \text{if } a \geq b, \\ b & \text{if } b > a. \end{cases} \tag{50}$$

The first loss function L_1 is usually referred to as the linear ϵ-insensitive loss function while L_2 is referred to as the quadratic ϵ-insensitive loss function. We say that L_1 and L_2 are ϵ-insensitive because they produce zero error for fitted values within the ϵ-tube which for L_1 is the region $\{\forall \boldsymbol{x} \in \mathbb{R}^p : |y - \hat{f}(\boldsymbol{x})| \leq \epsilon\} = \Omega$. From now on let us focus on the loss function L_1. Consider points above the ϵ-tube. These points have the distance $\xi_i = y_i - \hat{f}(\boldsymbol{x}_i) - \epsilon$ to Ω. Points below the ϵ-tube have a distance $\xi_i' = \hat{f}(\boldsymbol{x}_i) - y_i - \epsilon$ to Ω. We refer to ξ_i and ξ_i' as the set of slack variables. The slack variables are set to zero within Ω. The total error of the fit is then the sum of slack variables. Note that $\xi_i \geq y_i - \hat{f}(\boldsymbol{x}_i) - \epsilon$, $\xi_i' \geq \hat{f}(\boldsymbol{x}_i) - y_i - \epsilon$ and ξ_i, $\xi_i' \geq 0$. To avoid overfitting we introduce a norm penalty of the regression coefficients also known as regularization. Consider therefore the primal optimization problem

$$\begin{aligned} \underset{\hat{\beta}_0, \hat{\boldsymbol{\beta}}, \boldsymbol{\xi}, \boldsymbol{\xi}'}{\text{minimize}} \quad & \frac{1}{2}||\hat{\boldsymbol{\beta}}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \\ \text{subject to} \quad & y_i - \hat{f}(\boldsymbol{x}_i) - \epsilon - \xi_i \leq 0, \\ & \hat{f}(\boldsymbol{x}_i) - y_i - \epsilon - \xi_i' \leq 0, \\ & -\xi_i \leq 0, \\ & -\xi_i' \leq 0, \quad i = 1, \ldots, n. \end{aligned} \tag{51}$$

where $\boldsymbol{\xi} = \left[\xi_1, \dots, \xi_n\right]^T$, $\boldsymbol{\xi}' = \left[\xi_1', \dots, \xi_n'\right]^T$ and C is a parameter controlling the flatness of the function $\hat{f}(\boldsymbol{x})$. The primal Lagrangian is then calculated as

$$\mathcal{L}_p = \frac{1}{2}||\hat{\boldsymbol{\beta}}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') + \sum_{i=1}^{n} a_i\{y_i - \hat{f}(\boldsymbol{x}_i) - \epsilon - \xi_i\}$$
$$+ \sum_{i=1}^{n} b_i\{\hat{f}(\boldsymbol{x}_i) - y_i - \epsilon - \xi_i'\} - \sum_{i=1}^{n} c_i\xi_i - \sum_{i=1}^{n} d_i\xi_i', \quad (52)$$

where a_i, b_i, c_i and d_i are the nonnegative Lagrange multipliers. Taking the partial derivatives and setting them equal to zero yields

$$\frac{\partial \mathcal{L}_p}{\partial \hat{\beta}_0} = \sum_{i=1}^{n}(b_i - a_i) = 0, \quad (53)$$

$$\frac{\partial \mathcal{L}_p}{\partial \hat{\boldsymbol{\beta}}} = \hat{\boldsymbol{\beta}} + \sum_{i=1}^{n}(b_i - a_i)\,\boldsymbol{x}_i = 0, \quad (54)$$

$$\frac{\partial \mathcal{L}_p}{\partial \xi_i} = C - a_i - c_i = 0, \quad (55)$$

$$\frac{\partial \mathcal{L}_p}{\partial \xi_i'} = C - b_i - d_i = 0. \quad (56)$$

Define $\alpha_i = a_i - b_i$. By combining Equation (55) and (56) we also have $\alpha_i = d_i - c_i$. This means that

$$\hat{\boldsymbol{\beta}} = \sum_{i=1}^{n}\alpha_i\boldsymbol{x}_i, \quad (57)$$

$$0 = \sum_{i=1}^{n}\alpha_i. \quad (58)$$

This means that the optimal regression coefficients can be expressed as a linear combination of the observation vectors $\boldsymbol{x}_i$. By the structure of the problem only some α_i will be nonzero meaning only a set of observations $\boldsymbol{x}_i$ (the so called support vectors) contribute to the value of the optimal regression coefficients. Inserting Equation (57) into Equation (46) yields

$$\hat{f}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{n}\alpha_i\boldsymbol{x}^T\boldsymbol{x}_i. \quad (59)$$

The dual Lagrangian can now be written as

$$\begin{aligned}
\mathcal{L}_d &= \frac{1}{2}||\hat{\boldsymbol{\beta}}||^2 + \sum_{i=1}^{n} a_i\{y_i - \hat{\beta}_0 - \boldsymbol{x}^T\hat{\boldsymbol{\beta}} - \epsilon\} + \sum_{i=1} b_i\{\beta_0 + \boldsymbol{x}^T\boldsymbol{\beta} - y_i - \epsilon\}, \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{x}_i^T\boldsymbol{x}_j + \sum_{i=1}^{n}\left(\alpha_i\{y_i - \sum_{j=1}^{n}\alpha_j\boldsymbol{x}_i^T\boldsymbol{x}_j\} + (a_i + b_i)\epsilon\right), \\
&= \sum_{i=1}^{n}\alpha_i y_i + \epsilon\sum_{i=1}^{n}(a_i + b_i) - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\boldsymbol{x}_i^T\boldsymbol{x}_j, \\
&= \boldsymbol{\alpha}^T\boldsymbol{y} + \epsilon\sum_{i=1}^{n}(a_i + b_i) - \frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{G}\boldsymbol{\alpha},
\end{aligned} \quad (60)$$

where $\boldsymbol{G}$ is the Gramian matrix with elements $G_{ij} = \boldsymbol{x}_i^T \boldsymbol{x}_j$. The dual problem can thus be stated as the following quadratic program

$$\begin{aligned} \underset{\boldsymbol{a},\boldsymbol{b}}{\text{maximize}} \quad & \boldsymbol{\alpha}^T \boldsymbol{y} + \epsilon \sum_{i=1}^{n} (a_i + b_i) - \frac{1}{2} \boldsymbol{\alpha}^T \boldsymbol{G} \boldsymbol{\alpha} \\ \text{subject to} \quad & \boldsymbol{\alpha} = \boldsymbol{a} - \boldsymbol{b}, \\ & \sum_{i=1}^{n} \alpha_i = 0, \\ & 0 \leq a_i \leq C, \quad i = 1, \dots, n \\ & 0 \leq b_i \leq C, \quad i = 1, \dots, n \end{aligned} \tag{61}$$

where $\boldsymbol{a} = \begin{bmatrix} a_1 & \dots & a_n \end{bmatrix}^T$ and $\boldsymbol{b} = \begin{bmatrix} b_1 & \dots & b_n \end{bmatrix}^T$. The condition $a_i \leq C$ comes from Equation (55) that gives $a_i = C - c_i \leq C$ since $c_i \geq 0$. Analogously for $\boldsymbol{b}$. The advantage of Problem (61) compared to Problem (51) is that it is quadratic which has well optimized numerical solvers.

The Euclidean inner product between two vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in $\mathbb{R}^p$ is referred to as the dot product or scalar product calculated as $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^T \boldsymbol{x}_j$ and is used in Equation (59) and in the Gramian matrix. In SVM $K(\cdot)$ is known as the kernel function. In some cases, a considerably higher accuracy can be obtained if the kernel function or inner product takes other functional form. Another common kernel function is the Gaussian kernel defined as

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2). \tag{62}$$

Fortunately SVR is easily extended to using a general kernel $K(\cdot)$. The approximate model is changed to

$$\hat{f}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}), \tag{63}$$

and the Gramian is calculated as $G_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x})$. All else remain. Intuitively speaking applying the kernel function is a way of transforming the data into higher dimension where the function pattern is linear without actually doing the computations in the high dimensional space. This is known as the "kernel trick".

6 Multiple Linear Regression

For the purpose of comparison of machine learning approaches to the conventional linear regression modeling we consider multiple linear regression (MLR).

6.1 Overview

Consider a finite set of regressors x_1, x_2, $\dots$, x_p and a single response y. Let the true relationship between the regressors and response be defined as

$$y = f(x_1, x_2, \dots, x_p) + \epsilon, \tag{64}$$

where ϵ is a normally distributed random error. In practice $f(\cdot)$ is unknown and the task is to obtain an approximation $\hat{f}(\cdot)$ of this function. In multiple linear regression we let

$$\hat{f}(x_1, x_2, \dots, x_p) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \dots + \hat{\beta}_p x_p, \tag{65}$$

where $\hat{\boldsymbol{\beta}} = \begin{bmatrix}\hat{\beta}_0 & \hat{\beta}_1 & \dots & \hat{\beta}_p\end{bmatrix}$ is referred to as the regression coefficients. To determine the coefficients we first obtain n observations. Every observation consists of a measured response y_i and regressors $x_{i1}, x_{i2}, \dots, x_{ip}$ for $0 \leq i \leq n$. The corresponding approximated response can be calculated as $\hat{y}_i = \hat{f}(x_{i1}, x_{i2}, \dots, x_{ip})$. In vector notation we want to choose the regression coefficients $\hat{\boldsymbol{\beta}}$ such that the approximated responses $\hat{\boldsymbol{y}}$ are as close as possible to the observed responses $\boldsymbol{y}$. This means that we have n equations and $p+1$ unknowns. In practice $n > p+1$ meaning that we have an overdetermined system and an exact solution is not obtainable. An unique solution can however be obtained if we reformulate the problem as a certain minimization problem instead. If we use the mean square error defined in Equation (6) and minimize this error, the solution is unique and given by

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}, \tag{66}$$

where $\boldsymbol{X}$ is the design matrix defined in Equation (11). This is known as the ordinary least squares (OLS) parameter estimate.

6.2 Transformation of the Regressors

The assumption that the response depends linearly on the regressors as stated in Equation (65) may not always be suitable. It is often good practice to plot the regressors versus the response to see if linearity is a reasonable assumption. If one obtain a plot similar to the one in Figure 6 it may be relevant to introduce a transformed regressor of the form $x_{p+1} = x_1^2$. Keep in mind though that every newly introduced regressor introduces an additional regression coefficient which will increase R^2 but also the risk of overfitting. A common strategy is to introduce multiple transformed regressors and do feature selection upon this set to obtain a smaller set of features to increase performance and to avoid overfitting. Usual transformations of the regressors are logarithmic, square root and power.

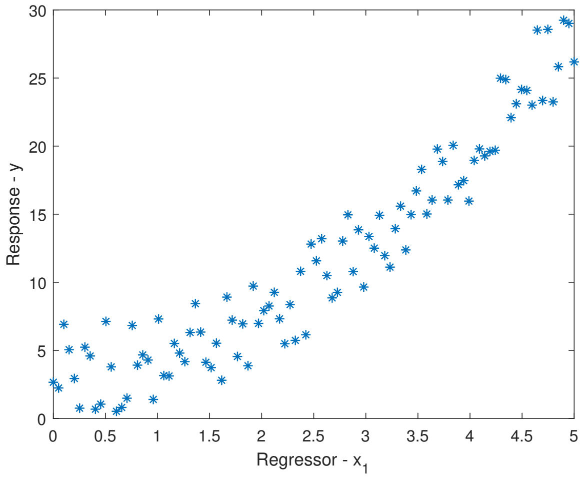


Figure 6: A seemingly quadratic relationship between a regressor and the response

6.3 Feature Scaling

In MLR, feature scaling has no statistical impact and may be disregarded without any accuracy loss. It may however be useful to do feature scaling for readability reasons. Without scaling one regression coefficient may be of magnitude 10^{-6} while another is of magnitude 10^3 and the expression may become hard to read. If we scale as in Equation (13) the intercept β_0 is the value of the response if all features are set to their mean. If we do not scale, the intercept is the value of the response if all features are set to 0 which may not be a realistic scenario. If we perform feature scaling the size of the coefficients tell us how large impact their corresponding feature has on the response.

6.4 Transforming the Response

Imagine we want to model

$$\hat{f}(x_1) = \hat{\beta}_1 e^{x_1}. \tag{67}$$

This equation is linear in $\hat{\boldsymbol{\beta}}$ but not in x_1. Linearity in the regressor can be achieved by letting $x_2 = e^{x_1}$ and obtain a form as in Equation (65). If we however would like to model

$$\hat{f}(x_1) = \hat{\beta}_1 e^{\hat{\beta}_2 x_1}, \tag{68}$$

which is nonlinear in $\hat{\boldsymbol{\beta}}$ we cannot work around that by transforming the regressor. If we instead transform the response into $\hat{y}_t = \ln \hat{y}$ we get

$$\begin{aligned} \hat{y}_t &= \ln\left(\hat{\beta}_1 e^{\hat{\beta}_2 x_1}\right), \\ &= \ln \hat{\beta}_1 + \hat{\beta}_2 x_1, \\ &= \hat{\beta}_0 + \hat{\beta}_2 x_1, \end{aligned} \tag{69}$$

which is linear.

Observe that a linear model where we log transform both the input and the output such as

$$\ln \hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \ln x_1 \tag{70}$$

is equivalent to the following model

$$\hat{y} = \hat{\beta}_0 x_1^{\hat{\beta}_1}. \tag{71}$$

This means that response transformation enables more complex models which can better fit nonlinear relationships.

6.5 Stepwise Regression

A systematic way of performing feature selection is to use a method called stepwise regression. In stepwise regression you first start with a model with only the intercept. The following is then done for all of the regressors. Add only the current regressor to the model and calculate its coefficient. A null hypothesis is then stated. The null hypothesis states that the true value of the coefficient is zero i.e. the regressor has no effect on the response. Given that the null hypothesis is true, the estimated probability that we would get a coefficient at least as extreme as the one obtained is referred to as the p-value. The p-value is calculated for all the coefficients. If any p-value is below the threshold (usually

0.05) the regressor with the smallest p-value is added to the model. The p-values for the regressors not in the model is recalculated to see if any additional regressor can be added to the model. This step is repeated until no p-value is below the threshold. Then the p-value for the regressors in the model is calculated. If any p-value is above the threshold the regressor with the largest p-value is removed from the model. The previous step starts anew. In this way regressors will be added and removed in steps until the method has converged. This results in a model where the included regressors have p-values below the threshold and the p-values for the excluded regressors are above the threshold. Note that this method cannot guarantee global optimality and if the initial model is something else than just the intercept the final model may be different.

6.6 Multicollinearity

Multicollinearity is a problem that arises when there are near-linear dependencies among the regressors. In the case of perfect multicollinearity the design matrix $\boldsymbol{X}$ is singular and thus the moment matrix $\boldsymbol{X}^T\boldsymbol{X}$ used in Equation (66) cannot be inverted and the OLS estimator $\hat{\boldsymbol{\beta}}$ does not exist. If the multicollinearity is not perfect but still present the variance of $\hat{\boldsymbol{\beta}}$ is high meaning that another sample set may produce highly different values of $\hat{\boldsymbol{\beta}}$ and the result is therefore unreliable. To avoid this problem the regressors that depend linearly on other regressors should be removed. A measure of the degree of this linear dependency is variance inflation factor (VIF).

6.6.1 Variance Inflation Factor

Let $\boldsymbol{x}_i$ be the $(i+1)$th column of the design matrix $\boldsymbol{X}$, i.e. all observations of feature i. The ith variance inflation factor is then defined as

$$\mathrm{VIF}_i = \frac{1}{1 - R_i^2}, \tag{72}$$

where R_i^2 is the coefficient of determination as defined in Equation (7) of the following fit

$$\boldsymbol{x}_i = \alpha_0 + \alpha_1\boldsymbol{x}_1 + \cdots + \alpha_{i-1}\boldsymbol{x}_{i-1} + \alpha_{i+1}\boldsymbol{x}_{i+1} + \cdots + \alpha_p\boldsymbol{x}_p. \tag{73}$$

This means that R_i^2 measures how well the ith regressor can be expressed as a linear combination of the other regressors. As R_i^2 approaches 1, VIF_i approaches $+\inf$. A commonly used rule of thumb is to say that multicollinearity is present if any $\mathrm{VIF}_i > 5$ which is equivalent to $R_i^2 > 0.8$. Generally the coefficient of determination increases as additional regressors are introduced which means that the risk of having multicollinearity increases with the number of features in the model.

6.7 Model Selection

As mentioned earlier plotting the regressors and the response can give some ideas on how to formulate a model. Parts of the model can sometimes be determined by knowing something regarding the real world problem. For instance we know the underlying relationship that extended the tables in Appendix A.1 to the tables in Appendix A.2 and if we wanted we could have this relationship explicitly in our model. However since we already know this relationship we will focus on the data set corresponding to all contract types when doing MLR. The feature selection will be done by using the method stepwise regression as described in Section 6.5.

7 Results

7.1 Artificial Neural Networks

In this section ANN is applied to the data sets corresponding to ITP2 and all contract types. Unless otherwise specified, the following design choices hold. Hidden layer transfer function is tansig and output layer transfer function is purelin. Preprocessing is removal of constant features and mapminmax$(x, -1, 1)$. Optimization function is Levenberg-Marquardt backpropagation (see Section 4.5) which is a usual procedure when performing nonlinear least squares curve fitting. The weight initialization method is Nguyen-Widrow. All errors are calculated using TVT-partitioning as defined in Section 3.4.3.

Firstly we perform feature selection by iteratively removing features such that the error is minimized in each step. The stepwise removal of variables can be seen in Table 8 and in Table 9.

Then we consider the all contracts data set in more detail. We are interested in what hidden layer transfer function is most suitable for the learning problem. The result of that study can be found in Table 10. Since Nguyen-Widrow initialization is to be studied we cannot use purelin and we can also see that purelin requires considerably more nodes than the others. Therefore tribas is used as transfer function in the hidden layer from now on.

To decide if the simple weight initialization method RS is more powerful than NW a comparison has been made in Table 11. There we have also listed the average required number of epochs before convergence during training. If the weights are initialized poorly the number of epochs until convergence may be large. From this table we conclude that using NW in both hidden and output layer is advised.

Finally we study if the preprocessing method impact on the performance. A study for this can be seen in Table 12. From this it seems like mapstd coupled with PCA is desired.

The final model will then use tribas as hidden layer transfer function and purelin as output layer transfer function. The initialization will be done using Nguyen-Widrow. The preprocessing will be removal of constant features, mapstd and PCA with a lower limit of 2% variance explanation. A visualization of this can be seen in Figure 7 where the number of hidden layer nodes is varied. The PCA is applied to reduce the feature dimension to two. In the figure it is clear that increasing the number of nodes increases the curvature. The reduced error may seem tempting but the risk of overfitting should not be neglected.

Table 8: Stepwise exclusion of variables using ANN for employment's contract type ITP2 using $n_h = 6$ hidden layer nodes. The network is created and trained 20 times with displayed average test error MSE and error standard deviation denoted by STD. The first row indicated by a "None" represents the full model with all features included

Excluded Variable	MSE	STD
None	0.00212973	0.00683795
x_{16}	0.00027524	0.00033226
x_{18}	0.00021227	0.00023113
x_{10}	0.00024906	0.00040883
x_{12}	0.00031536	0.00038009
x_{20}	0.00110724	0.00308297
x_{14}	0.00018007	0.00033603
x_{19}	0.00017751	0.00026194
x_{21}	0.00015600	0.00021418
x_{17}	0.00015304	0.00013784
x_{13}	0.00028566	0.00040830
x_{24}	0.00047484	0.00041215
x_{23}	0.00053146	0.00061147
x_{22}	0.00111509	0.00118574
x_9	0.00649019	0.00757104
x_{15}	0.08196508	0.07217501
x_{11}	0.53880618	0.28099432
x_8	0.96684102	0.19209143

Table 9: Stepwise exclusion of variables using ANN using all contract types with $n_h = 5$ hidden layer nodes. The network is created and trained 20 times with displayed average test error MSE and error standard deviation denoted by STD. The first row indicated by a "None" represents the full model with all features included

Excluded Variable	MSE	STD
None	0.2391	0.0835
x_{21}	0.2263	0.1157
x_1	0.2189	0.1075
x_{17}	0.1920	0.0821
x_{16}	0.2155	0.1104
x_5	0.1898	0.0653
x_{23}	0.1993	0.0896
x_3	0.1719	0.0710
x_{10}	0.2201	0.0947
x_7	0.1519	0.0604
x_4	0.1601	0.0557
x_2	0.1764	0.1247
x_6	0.1647	0.0951
x_{13}	0.1513	0.0864
x_{14}	0.1263	0.0726
x_{12}	0.1479	0.0600
x_{15}	0.1427	0.0752
x_{11}	0.1470	0.0995
x_8	0.1049	0.0587
x_{18}	0.1287	0.0671
x_{19}	0.1262	0.0719
x_{20}	0.1436	0.0828
x_{22}	0.2246	0.1286
x_{24}	0.2503	0.1060
x_9	0.4004	0.1196

Table 10: Network performance averaged over 20 trails for all of the transfer functions defined in Table 5 to Table 7 used in the hidden layer

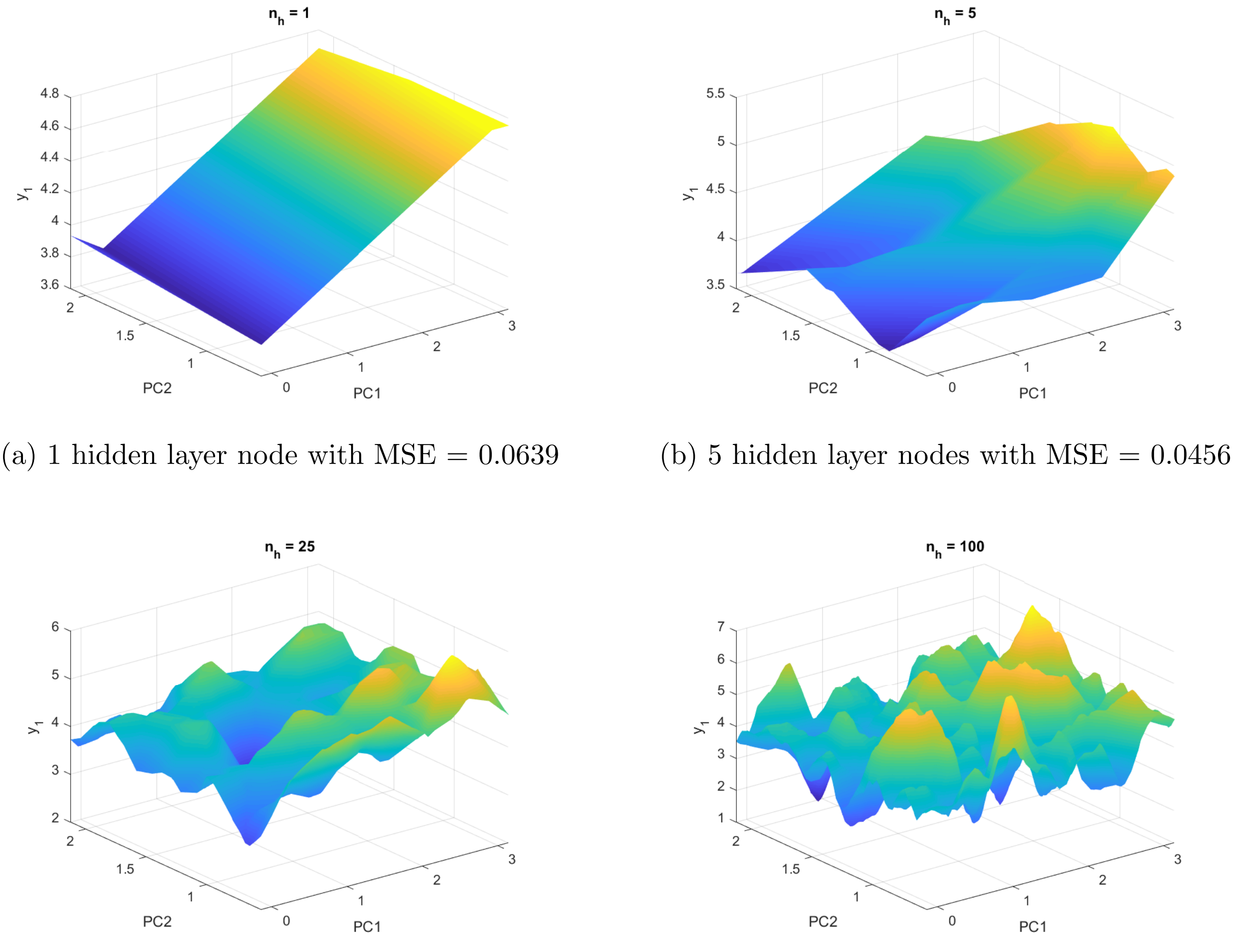
Transfer Function	Optimal Number of Nodes	MSE	STD
compet	1	0.3530	0.1139
elliotsig	4	0.3719	0.1589
hardlim	7	0.3530	0.1093
harmlims	4	0.3350	0.0820
logsig	2	0.3717	0.1805
netinv	1	0.4275	0.2000
poslin	1	0.4101	0.1466
purelin	8	0.3150	0.1269
radbas	1	0.3821	0.1113
radbasn	2	0.3467	0.1137
satlin	3	0.3625	0.1104
satlins	1	0.3686	0.1237
softmax	2	0.3506	0.1264
tribas	1	0.3217	0.0707
tansig	2	0.4103	0.1087

Table 11: Network performance for the different weight initialization methods as defined in Section 4.6 on the hidden and output layer. Each row corresponds to the network being created and trained 20 times on the all contracts data set. The performance average is denoted by MSE and the performance standard deviation by STD. The average number of training iterations until convergence also known as epochs is also displayed. An asterisk denotes best performance for a given choice of initialization methods

Hidden/Output initialization	n_h	MSE	STD	epochs
NW/NW	1	0.3113	0.1154	14.85
	2	0.2027	0.0775	21.95
	3	0.2668	0.1506	15.25
	4	0.2224	0.0985	14.90
	5	0.2411	0.1588	17.50
	6	0.2849	0.2626	22.40
*	7	0.1567	0.0778	15.05
	8	0.2490	0.1674	14.85
	9	0.1763	0.0750	16.05
RS/NW	1	0.3447	0.1331	12.85
	2	0.3013	0.1499	21.15
	3	0.2399	0.0787	17.90
	4	0.2714	0.1643	73.80
	5	0.2670	0.1943	15.40
	6	0.2612	0.1619	19.45
*	7	0.2069	0.1087	21.85
	8	0.2100	0.0888	14.25
	9	0.2117	0.0979	16.95
RS/RS	1	0.3440	0.1428	21.15
	2	0.3059	0.1538	14.95
	3	0.3075	0.1678	19.75
	4	0.3045	0.1801	16.40
*	5	0.2304	0.1090	16.75
	6	0.3732	0.3951	18.55
	7	0.2527	0.1480	16.90
	8	0.2524	0.1335	15.60
	9	0.2702	0.1247	14.70
NW/RS	1	0.2880	0.1322	16.70
	2	0.2059	0.1307	18.75
	3	0.2561	0.1905	17.80
	4	0.2089	0.0939	15.95
	5	0.2116	0.0839	17.10
	6	0.2786	0.2888	14.50
*	7	0.1989	0.1060	18.80
	8	0.1996	0.0882	17.95
	9	0.2577	0.2341	15.85

Table 12: Network performance for the different preprocessing methods. Each row corresponds to the network being created and trained 20 times on the all contracts data set using tribas as hidden layer transfer function. The performance average is denoted by MSE and the performance standard deviation by STD. PCA is applied such that only principal components with an explained variance larger than 2% are included. An asterisk denotes best performance for a given choice of preprocessing method

Preprocessing	n_h	MSE	STD
mapminmax	1	0.2628	0.1456
	2	0.2632	0.1290
	3	0.2356	0.1090
	4	0.2366	0.1230
	5	0.2353	0.1423
	6	0.4263	0.5679
	7	0.2963	0.2157
	8	0.2274	0.1942
*	9	0.2217	0.1640
mapstd	1	0.2926	0.1208
*	2	0.2006	0.1110
	3	0.2546	0.1407
	4	0.2159	0.1069
	5	0.2542	0.1012
	6	0.2110	0.1265
	7	0.2281	0.0926
	8	0.2788	0.1851
	9	0.2648	0.1717
mapstd + pca	1	0.2964	0.0965
	2	0.2081	0.1220
	3	0.1768	0.0938
	4	0.2435	0.1173
	5	0.2219	0.1716
*	6	0.1645	0.0841
	7	0.2101	0.1319
	8	0.2265	0.1287
	9	0.2084	0.2168



(a) 1 hidden layer node with MSE = 0.0639

(b) 5 hidden layer nodes with MSE = 0.0456

(c) 25 hidden layer nodes with MSE = 0.0622

(d) 100 hidden layer nodes with MSE = 0.0191

Figure 7: ANN trained on all contracts data set after feature selection using the first two principal components explaining 81% of the variance in the data set. The number of nodes in hidden layer is varied and the resulting increase in curvature is apparent. The z-axis is the scaled monthly pension at 65 years denoted by y_1

7.2 Multiple Linear Regression

In this section we try to find a mathematical function $\hat{f}(\cdot)$ to best fit the data to $y^{(65)}$ using techniques from MLR. We start by considering the following model

$$\hat{f}(\boldsymbol{x}) = \hat{\beta}_0 + \boldsymbol{x}^T\hat{\boldsymbol{\beta}}, \tag{74}$$

where $\boldsymbol{x}^T = \begin{bmatrix} x_1 & x_2 & \dots & x_{24} \end{bmatrix}$. Notice that if we use this model we have fallen into the so called dummy variable trap. This is because we have categorical variables x_1 up to x_7 that together describe all possible contract types for the data set. This means that the design matrix

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & \dots & x_{24,1} \\ 1 & x_{1,2} & x_{2,2} & \dots & x_{24,2} \\ & & & \vdots & \\ 1 & x_{1,79} & x_{2,79} & \dots & x_{24,79} \end{bmatrix} \tag{75}$$

where $x_{i,j}$ is the jth observation of the ith feature, is singular because the sum of column 2 up to column 8 equals column 1. In other words we will have multicollinearity. To avoid this we can either remove the intercept β_0 (the first column of the design matrix) or one of the dummy variables from the model. Some tools in MLR require the intercept to be present such as the coefficient of determination R^2 and some formulas for statistical inference. We chose to remove the feature x_1 meaning that if all the features x_2 up to x_7 equals 0 we know that the contract type is equal to the one corresponding to x_1.

Now that the dummy variable trap is avoided we will investigate if multicollinearity is present between the remaining features. To do this we study variance inflating factors (VIFs) as shown in Table 13. We choose to remove features with VIF > 5 meaning that x_8, x_{23}, x_3, x_{13} and x_{18} are removed to avoid multicollinearity.

Since multicollinearity is no longer present we can study the following model

$$\hat{f}(\boldsymbol{x}) = \hat{\beta}_0 + \sum_{i \in \mathcal{I}} \hat{\beta}_i x_i, \quad \mathcal{I} = \{1, 2, \dots, 24\} \backslash \{1, 3, 8, 13, 18, 23\}, \tag{76}$$

where $\mathcal{I}$ is a set of indices corresponding to the features that are present in the model. Two residual plots are shown in Figure 8 and in Figure 9 for the corresponding fit. The normal probability plot indicates that the residuals are non normal with heavier tails than the normal distribution. The residual versus fitted values plot displays a double bow pattern indicating non constant variance of the error. Therefore we consider a variance stabilizing transformation of the response. Let y_t be the transformed value of the observed response y defined as

$$y_t = \sin^{-1}\left(\sqrt{\text{mapminmax}(y, 0, 1)}\right), \tag{77}$$

as inspired by [8] to counter the double bow variance pattern.

We will try to find an approximative function to describe the transformed response of the form

$$y_t \approx \hat{f}_t(\boldsymbol{x}) = \hat{\beta}_{t0} + \sum_{i \in \mathcal{I}} \hat{\beta}_{ti} x_i, \quad \mathcal{I} = \{1, 2, \dots, 24\} \backslash \{1, 3, 8, 13, 18, 23\}, \tag{78}$$

where the subscript t indicates that it is a part of the response transformed model. Again two residual plots are shown in Figure 10 and in Figure 11 for the corresponding fit. The errors now look more normal with less heavy tails and the variance is stable.

Now that we have no multicollinearity and approximately normally distributed error with constant variance we will perform stepwise regression as explained in Section 6.5 to obtain a set of significant regressors. The result from the stepwise regression can be seen in Table 14. We have

$$\hat{f}_t(\boldsymbol{x}) = \hat{\beta}_{t0} + \sum_{i \in \mathcal{I}} \hat{\beta}_{ti} x_i, \quad \mathcal{I} = \{9, 10, 11, 12, 20, 24\}. \tag{79}$$

This model yields $R^2 = 0.9323$ and LOOCV $= 0.0127$. The fit diagnostics can be seen in Figure 12 displaying overall healthy behavior.

Table 13: Features with largest variance inflation factor (VIF). Each row displays the feature with the highest VIF in the data set which afterwards is deleted and the new maximum VIF feature is calculated on the following row. Initially the features x_2 up to x_{24} are included

Feature	VIF
x_8	8.1289e+08
x_{23}	7.6868e+07
x_3	13.1869
x_{13}	9.4876
x_{18}	6.6513
x_4	3.5612

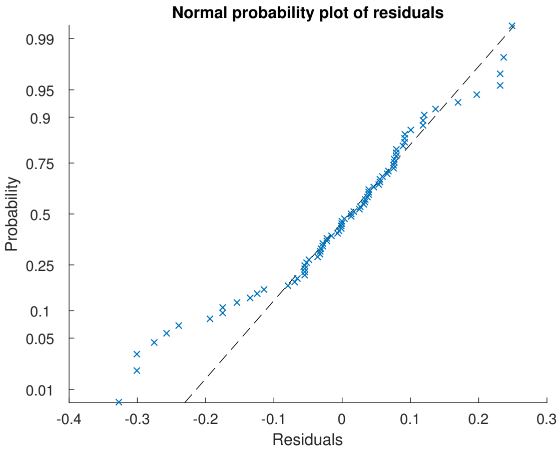


Figure 8: Normal probability plot of Equation (76)

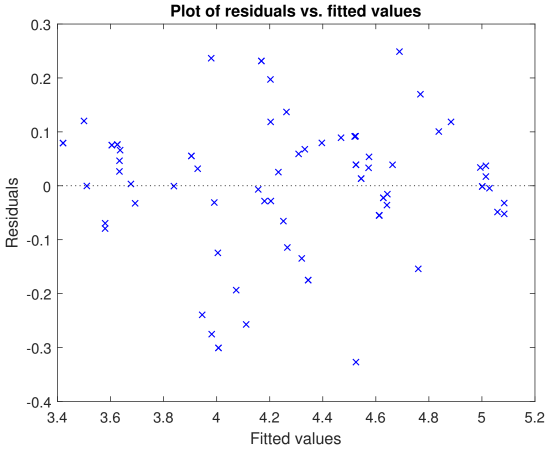


Figure 9: Residuals vs fitted values plot of Equation (76)

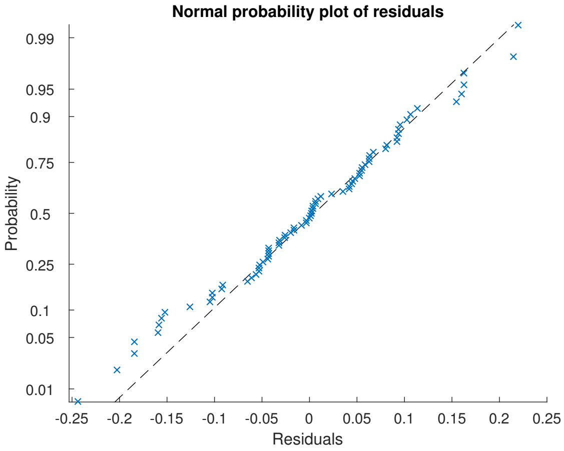


Figure 10: Normal probability plot of Equation (78)

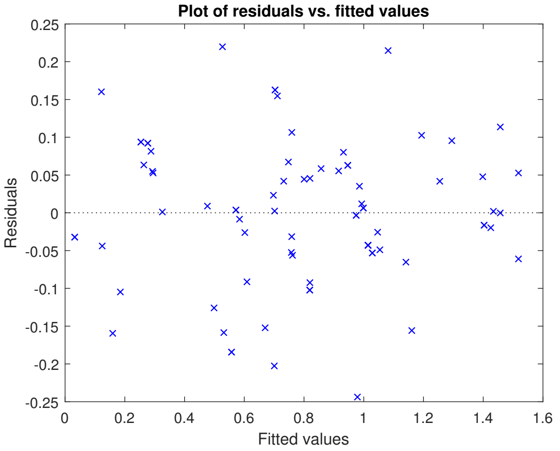
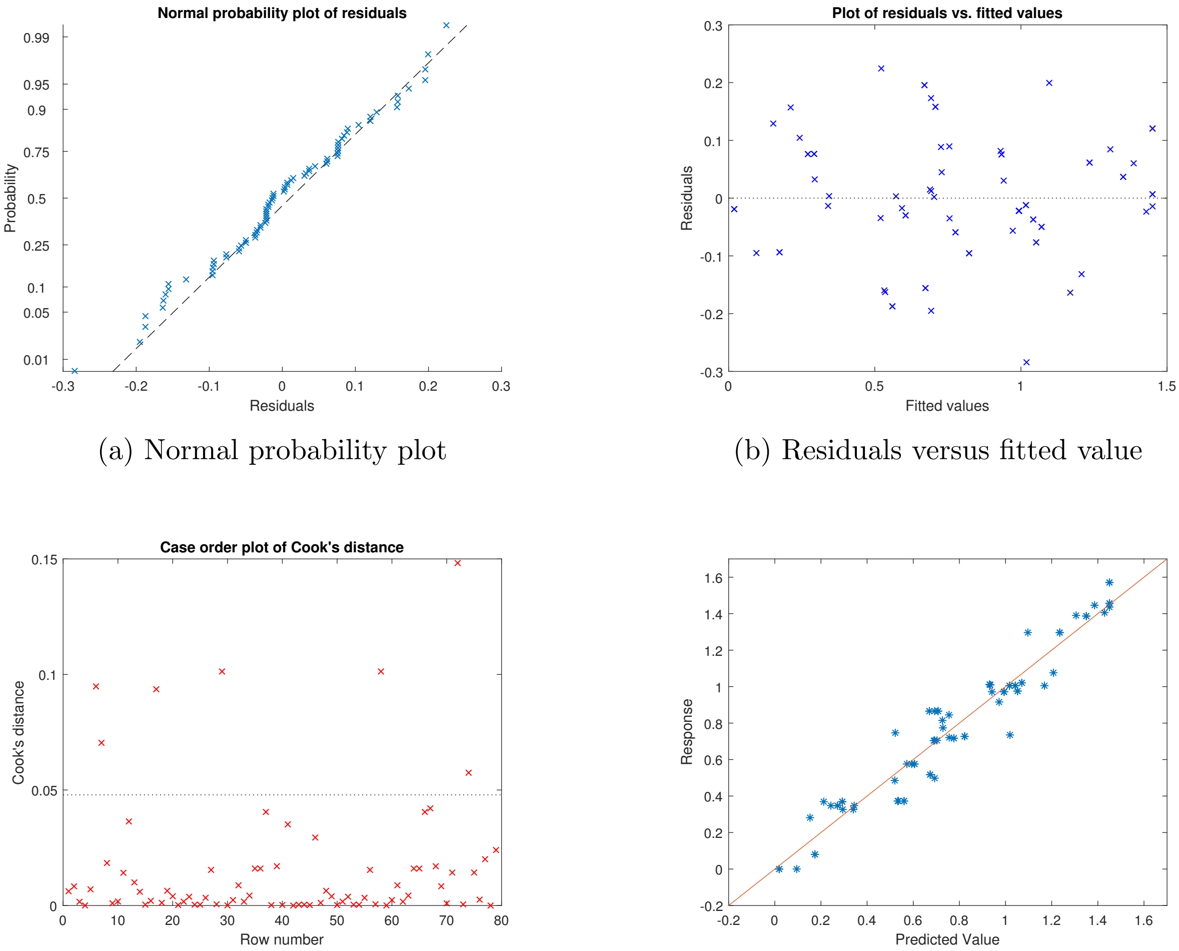


Figure 11: Residuals vs fitted values plot of Equation (78)

Table 14: Stepwise regression performed on the model in Equation (78) starting with only the intercept. The threshold p-value to enter the model is set to 0.05 and to be removed is 0.1

Step	Action	p value
1	Add x_9	1.86029e-25
2	Add x_{24}	3.19164e-13
3	Add x_{20}	3.55275e-06
4	Add x_{10}	0.000455106
5	Add x_{12}	0.0438494
6	Add x_{11}	0.0425742



(a) Normal probability plot

(b) Residuals versus fitted value

(c) Cook's distance versus row number

(d) Response versus fitted value

Figure 12: Fit diagnostics for the regression model in Equation (79)

7.3 Support Vector Machines for Regression

For the feature selection we will use the same approach as in ANN. This means that we will remove features iteratively and stop before the MSE starts increasing considerably.

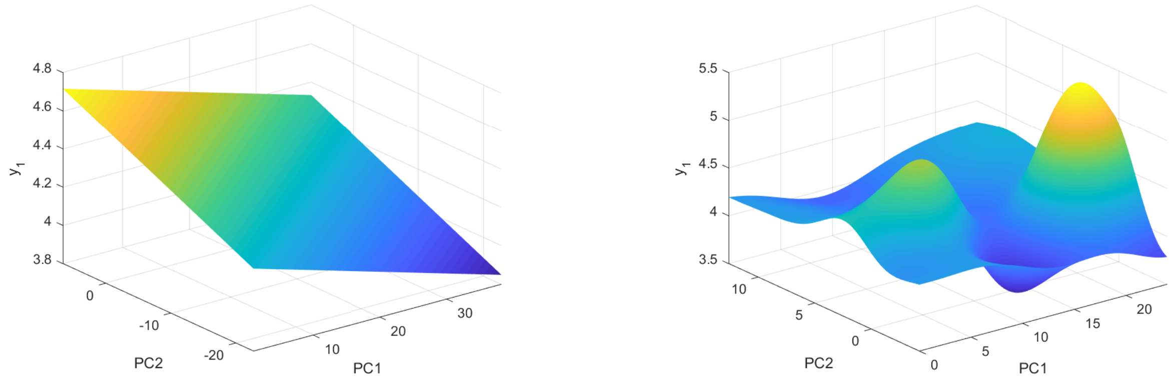
First we want to explore how well a linear kernel can be used to fit our data and what features are important for the MSE. The result can be seen in Table 15. The last excluded variable is x_2 with a resulting MSE $= 0.0147$

We are now interested in seeing the improvement in applying a nonlinear kernel such as the commonly used Gaussian kernel. The result of the stepwise removal of variables can be seen in Table 16. The last excluded variable is x_7 with a resulting MSE $= 0.0043$. This is promising result but does the model overfit the data? To examine this we calculate the leave one out cross validation as $\text{CV}_{(79)} = 0.0485$ indicating some degree of overfitting.

A visualization of the fits using linear and Guassian kernel can be seen in Figure 13. The increased complexity from using a Gaussian kernel is apparent.

Table 15: Stepwise exclusion of features using SVR with a linear kernel and standardization

Excluded Variable	MSE
none	0.0132
x_{14}	0.0131
x_{19}	0.0130
x_{16}	0.0129
x_6	0.0129
x_7	0.0129
x_{23}	0.0129
x_{24}	0.0129
x_{21}	0.0130
x_4	0.0134
x_5	0.0134
x_{22}	0.0136
x_{17}	0.0138
x_3	0.0139
x_1	0.0139
x_{12}	0.0144
x_2	0.0147
x_{11}	0.0159
x_{15}	0.0175
x_{13}	0.0180
x_{18}	0.0188
x_{10}	0.0218
x_{20}	0.0273
x_8	0.0459



(a) Linear kernel MSE = 0.2223 and $\sigma^2_{\text{exp}} = 91\%$ (b) Gaussian kernel MSE = 0.1179 and $\sigma^2_{\text{exp}} = 97\%$

Figure 13: SVR with two different kernels trained on all contracts data set after feature selection using the first two principal components. The z-axis is the monthly pension at 65 years denoted by y_1

Table 16: Stepwise exclusion of features using SVR with a Gaussian kernel and standardization

Excluded Variable	MSE
none	0.0081
x_{17}	0.0080
x_{14}	0.0080
x_{11}	0.0079
x_{24}	0.0079
x_{23}	0.0077
x_8	0.0067
x_{18}	0.0064
x_{13}	0.0062
x_{19}	0.0061
x_{20}	0.0059
x_2	0.0059
x_1	0.0055
x_6	0.0054
x_{16}	0.0053
x_5	0.0053
x_4	0.0045
x_3	0.0044
x_7	0.0043
x_{22}	0.0044
x_{15}	0.0062
x_{10}	0.0133
x_{21}	0.0301
x_{12}	0.0439

7.4 Feature Selection

For ANN and SVR the feature selection is done by removing features in steps until a satisfactory model is found. Meanwhile for linear regression the feature selection is done by stepwise regression. For ANN and ITP2 the stepwise removal of features can be seen in Table 8 and the last variable to be removed is decided to be x_{22}. For ANN on all contract types the stepwise removal of features can be seen in Table 9 and the last variable to be removed is decided to be x_8. For the linear regression one should look at the final model in Equation (79). In summary the final set of features for the various methods can be found in Table 17.

Table 17: Results of feature selection for the various methods

Method	Number of Features	List of Features
ANN for contract type ITP2	5	$x_8, x_9, x_{11}, x_{15}, x_{22}$
ANN for all contract types	6	$x_9, x_{18}, x_{19}, x_{20}, x_{22}, x_{24}$
MLR for all contract types	6	$x_9, x_{10}, x_{11}, x_{12}, x_{20}, x_{24}$
SVR - Linear for all contract types	8	$x_8, x_9, x_{10}, x_{11}, x_{13}, x_{15}, x_{18}, x_{20}$
SVR - Gaussian for all contract types	6	$x_9, x_{10}, x_{12}, x_{15}, x_{21}, x_{22}$

7.5 Comparison

In this section we will compare the following different ways of calculating the monthly pension amount

1. ANN for contract type ITP2 using the data in Appendix A.2.
2. ANN for all contract types using the data in Appendix A.1.
3. MLR using the data in Appendix A.1. The explicit formula is stated in Equation (79).
4. SVR using a linear kernel and the data in Appendix A.1.
5. SVR using a Gaussian kernel and the data in Appendix A.1.
6. The log linear relationship developed by Tommy Lowén displayed in Equation (28).
7. The premium pension formula from the Swedish Pensions Agency as defined in Section 3.7.

The performance of methods 1-5 are displayed in Table 18. All methods (1-7) will be compared using a real life example. We want to calculate what annuity divisor and how much in Occupational Pension a person will get every month if he decides to retire at 65 years. Assume that he is born in 1970 and works as a bank official, i.e. his employment contract belongs to the type ITP2. Furthermore, the pension savings are assumed to belong to any of three Swedish private insurance companies Skandia, Swedbank and Folksam in order to compare how much the Occupational Pension may vary between companies. Therefore relevant feature values from these companies are collected. Finally, let the total Occupational Pension amount to 500 000 sek. The 7 methods are displayed in Table 24 using feature values from the three selected private insurance companies with the corresponding annuity divisors.

Table 18: Final performance for our methods. The ANN models now only show the error for the first response $y^{(65)}$. For ANN MSE_{pred} is equal to the TVT error while the other methods use LOOCV

Method	MSE	MSE_{pred}
ANN on ITP2		$6.714 \cdot 10^{-5}$
ANN on all contract types		0.0497
MLR	0.0157	0.0191
SVR - linear	0.0132	0.0171
SVR - Gaussian	0.0020	0.0491

Table 19: Retiree's pension amount per month for three different Swedish insurance companies using the formula obtained from ANN for contract type ITP2

Feature	Name	Skandia	Swedbank	Folksam
x_8	Expected Remaining Lifetime at Year 65(years)	22.18	22.18	22.18
x_9	Advance Interest Rate(%)	1.25	1.5	3.0
x_{11}	Return in 2017(%)	4.2	8.58	8.12
x_{15}	Return in 2013(%)	3.5	21.95	9.67
x_{22}	Asset Management Fee(%)	0.42	0.3	0.8
	Response $y^{(65)}$	3.97	4.49	4.99
	Retiree's Pension Amount per Month(sek)	1985.00	2245.00	2495.00

Table 20: Retiree's pension amount per month for three different Swedish insurance companies using the formula obtained from ANN for all types of employment contracts

Feature	Name	Skandia	Swedbank	Folksam
x_9	Advance Interest Rate(%)	1.25	1.5	3.0
x_{18}	Insurance(yes/no)	yes	no	yes
x_{19}	Old savings Covered(yes/no)	yes	yes	yes
x_{20}	Free Consulting(yes/no)	yes	yes	yes
x_{22}	Asset Management Fee(%)	0.42	0.3	0.8
x_{24}	Expected Remaining Lifetime at Year 85(years)	7.47	7.47	7.47
	Response $y^{(65)}$	4.27	4.82	4.74
	Retiree's Pension Amount per Month(sek)	2135.00	2412.25	2372.40

Table 21: Retiree's pension amount per month for three different Swedish insurance companies using the formula obtained from MLR displayed in Equation (79)

Feature	Name	Skandia	Swedbank	Folksam
x_9	Advance Interest Rate (%)	1.25	1.5	3.0
x_{10}	Fixed Annual Fee (%)	0.17	0	0
x_{11}	Return in 2017(%)	4.2	8.58	8.12
x_{12}	Return in 2016(%)	15	7.23	7.39
x_{20}	Free Consulting(yes/no)	yes	yes	yes
x_{24}	Expected Remaining Lifetime at Year 85(years)	7.47	7.47	7.47
	Response $y^{(65)}$	4.25	4.42	4.92
	Retiree's Pension Amount per Month(sek)	2126.70	2209.75	2461.05

Table 22: Retiree's pension amount per month for three different Swedish insurance companies using the formula obtained using SVR with a linear kernel

Feature	Name	Skandia	Swedbank	Folksam
x_8	Expected Remaining Lifetime at Year 65(years)	22.18	22.18	22.18
x_9	Advance Interest Rate (%)	1.25	1.5	3.0
x_{10}	Fixed Annual Fee(‰)	0.17	0	0
x_{11}	Rerurn in 2017(%)	4.2	8.58	8.12
x_{13}	Return in 2015(%)	8.3	22.34	17.16
x_{15}	Return in 2013(%)	3.5	21.95	9.67
x_{18}	Insurance(yes/no)	yes	no	yes
x_{20}	Free Consulting(yes/no)	yes	yes	yes
	Response $y^{(65)}$	4.12	4.49	4.82
	Retiree's Pension Amount per Month(sek)	2060.00	2245.00	2410.00

Table 23: Retiree's pension amount per month for three different Swedish insurance companies using the formula obtained using SVR with a Gaussian kernel

Feature	Name	Skandia	Swedbank	Folksam
x_9	Advance Interest Rate (%)	1.25	1.5	3.0
x_{10}	Fixed Annual Fee(‰)	0.17	0	0
x_{12}	Rerurn in 2016(%)	15	7.23	7.39
x_{15}	Return in 2013(%)	3.5	21.95	9.67
x_{21}	Fund Management Fee(%)	0	0.545	0
x_{22}	Asset Management Fee(%)	0.42	0.3	0.8
	Response $y^{(65)}$	3.94	4.27	4.88
	Retiree's Pension Amount per Month(sek)	1970.00	2135.00	2440.00

Table 24: Comparison of different annuity divisor formulas

Formula	Annuity Divisor	Retiree's Pension Amount per Month(sek)
ANN for employment contract type ITP2		
Skandia	20.99	1985.00
Swedbank	18.56	2245.00
Folksam	16.70	2495.00
ANN for all types of employment contracts		
Skandia	19.52	2135.00
Swedbank	17.27	2412.25
Folksam	17.56	2372.40
MLR for all types of contracts		
Skandia	19.61	2126.70
Swedbank	18.85	2209.75
Folksam	16.93	2461.05
Linear SVR for all types of contracts		
Skandia	20.23	2060.00
Swedbank	18.56	2245.00
Folksam	17.29	2410.00
Gaussian SVR for all types of contracts		
Skandia	21.15	1970.00
Swedbank	19.52	2135.00
Folksam	17.08	2440.00
Tommy Lowén's formula	16.68	2498.00
Premium Pension formula-Swedish Pension Agency	16.53	2520.00

8 Discussion

To answer the question which variables have the most impact on the annuity divisor we considered two methods for feature selection. For ANN and SVR we removed features iteratively to obtain a better model with respect to MSE and the number of features. In the MLR setting we used the systematic method stepwise regression. The result from the feature selection can be found in Table 17. The key factors are expected remaining lifetime at 65 x_8 and advance interest rate x_9. Note that variables x_{23} and x_{24} are calculated using the value of x_8 as explained in Section 2.2. In the table we see that the key factors survived the feature selection for all methods except for Gaussian SVR. This confirms the importance of the previously stated key factors for accurate modeling of the annuity divisor in private insurance. However the contract type variables x_1 up to x_7 do not seem to play an important role in determining the annuity divisor. Note that our models use features corresponding to information regarding the private insurance companies such as return of investments from pension savings x_{11} up to x_{15} and asset management fee x_{22} unlike the formulas from Tommy Lowén and the Swedish Pensions Agency. We feel like using these features should be more realistic but it also increases the complexity of the model which sometimes may be undesired.

ANN with stepwise removal of features required more work and time than the systematic stepwise regression but was still doable. SVR stepwise removal of features was faster because of a simpler optimization problem and because the performance is deterministic meaning we did not need to retrain multiple times to obtain a performance average. For large scale problems stepwise regression may be preferred because of its ease of use.

When MLR was considered we had to introduce a variance stabilizing transformation of the response to not violate underlying assumptions. Initially multicollinearity was present which was solved by removing features with high VIF and by removing x_1 to avoid the dummy variable trap.

The results from ANN have relatively high standard deviation even with 20 trials. This has the effect that the performance could vary a lot between different trials and the result shown in Table 24 would change noticeably if we recreated and retrained the network. Therefore unlike MLR and SVR modeling, the result is quite unpredictable. The randomness in the results comes from the random weight initialization, random data partitioning and that the optimization problem is non convex with potentially multiple local minima which means that the starting point is crucial. We estimate that increasing the number of observations would increase the predictability because that would likely reduce the variation in the random data partitioning. However it is practically unavoidable to have some variance in the result without sacrificing performance for ANN. The high standard deviation also makes the comparison between design choices not as clear as we would like it to be. We have however disregarded this and made design choices based on average MSE meaning that our final result is not guaranteed to be optimal. Overall the methods in Table 24 yield similar results.

When comparing ANN, MLR and SVR we can look at Table 18. Here we see that ANN is well suited for the annuity divisor modeling when only considering a single employment contract at the time. ANN seems to perform worse than MLR when studying all employment contracts simultaneously while SVR has the best accuracy. When looking at the prediction accuracy estimated by LOOCV or TVT partitioning, SVR with a linear kernel yields the best result. The high accuracy from the Gaussian SVR seems to be from overfitting indicating that the control parameters C and ϵ in Equation (51) may have to be further tuned.

One also has to keep in mind that the private insurance companies most likely do not publish all information that goes into the calculations of the annuity divisor meaning the optimal performance is therefore limited.

All in all ANN, MLR and SVR seem like suitable methods for this type of problem. However we are concerned about the high standard deviation in the results from ANN and the overfitting of Gaussian SVR and would therefore suggest the usage of MLR or SVR with a linear kernel.

A Appendix

A.1 Data for All Employment Contracts

Table 25: All data gathered from *Konsumenternas försäkringsbyrå* for all employment's contracts differentiated by color. Color codes: PA03 AKAP-KL2014 ITP1 PA-KFS09 KAP-KL2011 ITP2 SAT-LO - Part I/III

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
22	2.9	0.15	-6.8	7.9	14.9	17.3	16.8	0	0.12	1	0	0	0	0.03	15.1769	7.4174	5.028	6.308	8.006
23.4	2.5	0.17	8.7	6.3	18	24.5	8.7	0	0	0	0	0	0.55	0	15.5778	7.8553	4.616	6.122	8.154
23.4	1.75	0.17	10.5	9.6	16	14.3	10.5	0	0.15	1	0	0	0	0.03	15.5778	7.8553	4.176	5.934	8.498
21	0	0.17	13.2	10	22.2	22.1	13.2	0	0	0	0	1	0.725	0	14.8906	7.1046	3.96	6.28	10
22.7	0	0.17	9.2	6.3	20.3	25.4	12.9	0	0	0	0	1	0.875	0	15.3774	7.6363	3.68	5.76	9.02
22.4	2.5	0.17	-3.42	7.44	17.35	17.07	9.63	0	0.1	1	0	1	0	0.08	15.2915	7.5425	4.198	5.686	7.716
22.5	0	0.17	23.72	5.38	19.83	23.72	23.72	0.1	0	0	0	1	0.55	0	15.3201	7.5738	3.702	5.824	9.156
21.75	2	0.012	5	3	12.7	12	8	0	0.07	1	1	0	0	0.05	15.1053	7.3392	4.702	6.472	8.964
22.8	2.75	0.17	5.5	3.7	13.4	8	7.2	0	0.13	1	1	1	0	0.05	15.406	7.6676	4.558	5.902	7.642
22.8	0	0.16	-8	5.43	19.36	17.74	9.68	0.8	0	0	1	1	0.55	0	15.406	7.6676	3.66	5.768	9.026
22.5	1	0.17	11.19	6.01	21.97	19.4	11.19	1	0	0	0	1	0.6	0	15.3201	7.5738	3.706	5.818	9.098
23.9	0	0.16	-10.59	4.52	24.33	19.16	11.98	0	0	0	0	1	0.65	0	15.721	8.0117	3.5	5.5	8.766
21.3	0.75	0.17	-5.37	3.03	11.35	9.91	6.66	1	0	0	0	1	0.75	0	14.9765	7.1985	4.322	6.178	8.192
22.7	2.4	0.17	5.3	15	8.3	4.3	3.5	1	0.12	1	0	1	0	0.08	15.3774	7.6363	4.476	5.908	8.068
21.9	0.75	0.12	-10.12	4.59	22.25	21.43	10.34	1	0	0	0	1	0.65	0	15.1483	7.3861	4.15	6.134	9.058
21.9	0.25	0.168	-10.12	4.59	22.25	21.43	10.34	1	0	1	0	1	0	0.2	15.1483	7.3861	3.838	5.726	8.428
23.1	1.5	0.14	-12.58	7.97	21.63	22.53	7.75	1	0	0	1	1	0.6	0	15.4919	7.7615	4.4	5.9	7.96
22	2.9	0.13	5.8	7.9	14.9	17.3	9	0.6	0	1	1	0	0.17	0.03	15.1769	7.4174	5	6.276	7.962
23.5	2.5	0.13	11	6.3	18	24.5	11	0.8	0.33	0	1	0	0	0	15.6064	7.8866	4.612	6.118	8.15
23.5	1.75	0.13	8.6	9.6	16	14.3	8.6	0.8	0	1	1	0	0.15	0.03	15.6064	7.8866	4.17	5.928	8.492
23.3	3	0	15.91	5.11	26.3	21.22	12.24	0	0.765	0	1	1	0	0	15.5492	7.824	4.606	5.95	7.69
21	0	0.13	9.44	8.58	21.14	19.64	11.7	1	0.62	0	1	1	0	0	14.8906	7.1046	3.96	6.2	9.74
22.5	0	0.13	11.54	5.38	19.83	23.72	11.68	1	0.445	0	1	1	0	0	15.3201	7.5738	3.702	5.824	9.16
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0	1	1	1	0.06	0.05	15.406	7.6676	4.558	5.936	7.73
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0	1	1	1	0.06	0.05	15.406	7.6676	4.558	5.936	7.73
22.7	0	0.13	11	6.8	17.3	25	12.9	1	0.58	0	1	1	0	0	15.3774	7.6363	3.68	5.76	9.02
22.5	1	0.13	12.31	6.15	22.49	12.4	13.46	1	0.895	0	1	1	0	0	15.3201	7.5738	3.706	5.818	9.098
23.9	0	0.13	13.7	4.5	23.1	11.79	12.6	0.6	0.71	0	1	1	0	0	15.721	8.0117	3.5	3.496	8.656
23.1	1.5	0.13	18.33	18.33	18.33	9.3	18.33	1	0.475	0	1	1	0	0	15.4919	7.7615	4.4	3.9	7.96
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.032	6.336	8.036
23.3	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	15.5492	7.824	4.628	6.15	8.21
23.3	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	15.5492	7.824	4.186	5.928	8.452

Table 26: All data gathered from *Konsumenternas försäkringsbyrå* for all employment's contracts differentiated by color. Color codes: PA03 AKAP-KL2014 ITP1 PA-KFS09 KAP-KL2011 ITP2 SAT-LO - Part II/III

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
23.3	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	15.5492	7.824	4.606	6.026	7.886
22.4	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	15.2915	7.5425	4.938	8.306	8.038
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.052	6.336	8.036
23.9	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	15.721	8.0117	3.51	5.616	9.008
22.7	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	15.3774	7.6363	3.88	5.72	8.82
21.9	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.1483	7.3861	4.152	6.304	9.57
23.1	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	15.4919	7.7615	4.4	5.98	8.14
22	2.9	0	5.8	7.9	14.9	17.3	9	0	0.15	1	1	0	0.15	0.03	15.1769	7.4174	5.024	6.3	7.992
23.3	3	0	15.91	4.8	25.8	20.9	12.22	0	0	0	1	0	0	0	15.5492	7.824	4.606	5.95	7.69
22.7	0	0	12.8	4.6	22.2	21.4	10	0	0	0	1	1	0	0	15.3774	7.6363	3.68	5.58	8.44
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0.08	1	1	1	0.08	0.05	15.406	7.6676	4.558	5.936	7.73
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0.08	1	1	1	0.08	0.05	15.406	7.6676	4.558	5.936	7.73
22.8	0	0	11.47	5.61	22.32	20.82	10.38	0.6	0	0	0	1	0	0	15.406	7.6676	3.66	5.782	9.074
22.7	2.3	0.34	8.9	10.15	15	11.55	11	1	0.23	1	0	1	0.23	0.042	15.3774	7.6363	4.368	5.9	8.268
22	2.9	0.13	5.8	7.9	14.9	17.3	9	0.6	0	1	1	0	0.17	0.03	15.1769	7.4174	5	6.276	7.962
23.5	2.5	0.13	11	6.3	18	24.5	11	0.8	0.33	0	1	0	0	0	15.6064	7.8866	4.612	6.118	8.15
23.5	1.75	0.13	8.6	9.6	16	14.3	8.6	0.8	0	1	1	0	0.15	0.03	15.6064	7.8866	4.17	5.928	8.492
23.3	3	0	15.91	5.11	26.3	21.22	12.24	0	0.765	0	1	1	0	0	15.5492	7.824	4.606	5.95	7.69
21	0	0.13	9.44	8.58	21.14	19.64	11.7	1	0.62	0	1	1	0	0	14.8906	7.1046	3.96	6.2	9.74
22.5	0	0.13	11.54	5.38	19.83	23.72	11.68	1	0.445	0	1	1	0	0	15.3201	7.5738	3.702	5.824	9.16
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0	1	1	1	0.06	0.05	15.406	7.6676	4.558	5.936	7.73
22.8	2.75	0.096	6.7	3.7	13.4	8	4.8	1	0	1	1	1	0.06	0.05	15.406	7.6676	4.558	5.936	7.73
22.7	0	0.13	11	6.8	17.3	25	12.9	1	0.58	0	1	1	0	0	15.3774	7.6363	3.68	5.76	9.02
22.5	1	0.13	12.31	6.15	22.49	12.4	13.46	1	0.895	0	1	1	0	0	15.3201	7.5738	3.706	5.818	9.098
23.9	0	0.13	13.7	4.5	23.1	11.79	12.6	0.6	0.71	0	1	1	0	0	15.721	8.0117	3.5	3.496	8.656
23.1	1.5	0.13	18.33	18.33	18.33	9.3	18.33	1	0.475	0	1	1	0	0	15.4919	7.7615	4.4	3.9	7.96
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.032	6.336	8.036
23.3	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	15.5492	7.824	4.628	6.15	8.21
23.3	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	15.5492	7.824	4.186	5.928	8.452
23.3	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	15.5492	7.824	4.606	6.026	7.886
22.4	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	15.2915	7.5425	4.938	8.306	8.038
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.052	6.336	8.036

Table 27: All data gathered from *Konsumenternas försäkringsbyrå* for all employment's contracts differentiated by color. Color codes: PA03 AKAP-KL2014 ITP1 PA-KFS09 KAP-KL2011 ITP2 SAT-LO - Part III/III

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
23.9	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	15.721	8.0117	3.51	5.616	9.008
22.7	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	15.3774	7.6363	3.88	5.72	8.82
21.9	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.1483	7.3861	3.854	5.702	8.428
23.1	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	15.4919	7.7615	4.4	5.98	8.14
22	2.9	0.13	1	1	0	0	0	0	0.17	1	1	0	0	0.03	15.1769	7.4174	5.002	6.276	7.962
21.3	2.5	0	1	1	0	0.29	0	0.8	0	0	0	0	0.575	0	14.9765	7.1985	5.01	6.66	8.896
21.3	1.75	0.22	1	0	0	0	0	0.8	0	1	0	0	0	0.05	14.9765	7.1985	4.564	6.59	9.572
22.6	0	0	1	0	0	0.29	1	0	0	0	0	0	0.525	0	15.3487	7.6051	4.216	6.832	9.088
21	0	0.1	1	1	0	0.27	1	0	0	0	0	1	0.5	0	14.8906	7.1046	3.96	6.28	10
22.4	3	0.14	1	1	0	0	0	0	0	1	0	1	0	0.08	15.2915	7.5425	4.938	6.392	8.278
21.3	1.75	0.22	1	0	0	0	0	0.8	0	1	0	0	0	0.05	14.9765	7.1985	4.564	6.59	9.572
22.8	2	0.168	1	0	0	0	0	0.6	0	1	1	1	0	0.09	15.406	7.6676	4.258	6.13	8.744
22.5	1	0.16	1	1	0	0	0	0.8	0	1	0	1	0	0.08	15.3201	7.5738	3.706	5.96	9.544
21.9	0.75	0	1	1	0	0.3	1	0	0	0	0	1	0.95	0	15.1483	7.3861	4.152	6.266	9.454
23.1	0	0.16	1	0	0	0	0	0.3	0	1	0	1	0	0.06	15.4919	7.7615	3.62	5.88	9.64

A.2 Data for Contract Type ITP2

Table 28: Data for employment contract ITP2 with added observations calculated using the life expectancy assumptions as described in section 2.2 - Part I/IV

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.032	6.336	8.036
21	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.8906	7.1046	5.2716	6.4578	8.3898
20	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.6042	6.7919	5.5352	6.5844	8.7760
19	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.3179	6.4791	5.8265	6.7161	9.1997
18	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.0315	6.1663	6.1502	6.8532	9.6664
17	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	13.7452	5.8535	6.512	6.9959	10.1830
23	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.4633	7.7302	4.8132	6.2186	7.7108
24	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.7496	8.043	4.6126	6.1056	7.4109
25	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.0359	8.3557	4.4281	5.9965	7.1336
26	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.3223	8.6685	4.2578	5.8913	6.8761
27	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.6086	8.9813	4.1001	5.7898	6.6367
23.3	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	15.5492	7.824	4.628	6.15	8.21
22	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	15.1769	7.4174	4.9014	6.3008	8.6600
21	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	14.8906	7.1046	5.1348	6.4220	9.0413
20	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	14.6042	6.7919	5.3916	6.5479	9.4575
19	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	14.3179	6.4791	5.6753	6.6788	9.9141
18	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	14.0315	6.1663	5.9906	6.8152	10.4171
24	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	15.7496	8.043	4.4930	6.0717	7.9864
25	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	16.0359	8.3557	4.3132	5.9633	7.6875
26	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	16.3223	8.6685	4.1474	5.8587	7.4101
27	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	16.6086	8.9813	3.9937	5.7577	7.1520
28	2.5	0	11	6.3	18	24.5	8.7	0.8	0	0	0	0	0.33	0	16.895	9.2941	3.8511	5.6601	6.9113
23.3	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	15.5492	7.824	4.186	5.928	8.452
22	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	15.1769	7.4174	4.4333	6.0529	8.9153
21	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	14.8906	7.1046	4.6444	6.1693	9.3078
20	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	14.6042	6.7919	4.8766	6.2902	9.7363
19	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	14.3179	6.4791	5.1333	6.4160	10.2064
18	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	14.0315	6.1663	5.4185	6.5470	10.7241
24	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	15.7496	8.043	4.0639	5.8328	8.2218
25	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	16.0359	8.3557	3.9013	5.7286	7.9141
26	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	16.3223	8.6685	3.7513	5.6281	7.6285
27	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	16.6086	8.9813	3.6123	5.5311	7.3629

Table 29: Data for employment contract ITP2 with added observations calculated using the life expectancy assumptions as described in section 2.2 - Part II/IV

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
28	1.7	0.1	8.6	9.6	16	14.3	10.5	0.8	0.2	1	0	0	0	0.3	16.895	9.2941	3.4833	5.4373	7.1150
23.3	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	15.5492	7.824	4.606	6.026	7.886
22	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	15.1769	7.4174	4.8781	6.1738	8.3182
21	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	14.8906	7.1046	5.1104	6.2925	8.6845
20	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	14.6042	6.7919	5.3659	6.4159	9.0843
19	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	14.3179	6.4791	5.6484	6.5442	9.5229
18	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	14.0315	6.1663	5.9622	6.6777	10.0060
24	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	15.7496	8.043	4.4716	5.9493	7.6712
25	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	16.0359	8.3557	4.2927	5.8431	7.3841
26	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	16.3223	8.6685	4.1276	5.7405	7.1177
27	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	16.6086	8.9813	3.9748	5.6416	6.8698
28	3	0	16	7.85	26.27	25.25	8.73	0	0	0	1	1	0.43	0	16.895	9.2941	3.83285	5.5459	6.6386
22.4	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	15.2915	7.5425	4.938	8.306	8.038
23	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	15.4633	7.7302	4.8091	8.1781	7.8428
24	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	15.7496	8.043	4.6088	8.0294	7.5378
25	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	16.0359	8.3557	4.4244	7.8860	7.2557
26	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	16.3223	8.6685	4.2542	7.7477	6.9938
27	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	16.6086	8.9813	4.0967	7.6141	6.7503
21	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	14.8906	7.1046	5.2672	8.4926	8.5334
20	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	14.6042	6.7919	5.5305	8.6592	8.9263
19	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	14.3179	6.4791	5.8216	8.8323	9.3572
18	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	14.0315	6.1663	6.1450	9.0126	9.8319
17	3	0	8.12	7.39	17.16	17.08	9.67	0.8	0.17	1	0	1	0	0.8	13.7452	5.8535	6.5065	9.2003	10.3573
22	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.1769	7.4174	5.052	6.336	8.036
23	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.4633	7.7302	4.8323	6.2186	7.7108
24	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	15.7496	8.043	4.631	6.1056	7.4109
25	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.0359	8.3557	4.4457	5.9965	7.1336
26	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.3223	8.6685	4.2747	5.8913	6.8761
27	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	16.6086	8.9813	4.1164	5.7898	6.6367
21	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.8906	7.1046	5.2925	6.4578	8.3898
20	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.6042	6.7919	5.5572	6.5844	8.7760
19	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.3179	6.4791	5.8496	6.7161	9.1997

Table 30: Data for employment contract ITP2 with added observations calculated using the life expectancy assumptions as described in section 2.2 - Part III/IV

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
18	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	14.0315	6.1663	6.1746	6.8532	9.6664
17	2.9	0	5.8	7.9	14.9	17.3	16.8	0	0.1	1	1	0	0	0.3	13.7452	5.8535	6.5378	6.9959	10.1830
23.9	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	15.721	8.0117	3.51	5.616	9.008
24	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	15.7496	8.043	3.4953	5.6058	8.9729
25	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	16.0359	8.3557	3.3555	5.5057	8.6371
26	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	16.3223	8.6685	3.2265	5.4091	8.3254
27	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	16.6086	8.9813	3.107	5.3158	8.0355
28	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	16.895	9.2941	2.9960	5.2257	7.7650
22	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	15.1769	7.4174	3.8131	5.8173	9.7297
21	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	14.8906	7.1046	3.9947	5.9291	10.1581
20	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	14.6042	6.7919	4.1944	6.0454	10.6258
19	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	14.3179	6.4791	4.4152	6.1663	11.1387
18	0	0	9.74	8.68	29.41	24.77	12.09	0	0	0	0	1	0.455	0	14.0315	6.1663	4.6605	6.2922	11.7038
22.7	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	15.3774	7.6363	3.88	5.72	8.82
23	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	15.4633	7.7302	3.8293	5.6882	8.7128
24	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	15.7496	8.043	3.6698	5.5848	8.3740
25	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	16.0359	8.3557	3.5230	5.4851	8.0606
26	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	16.3223	8.6685	3.3875	5.3888	7.7697
27	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	16.6086	8.9813	3.2620	5.2959	7.4991
21	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	14.8906	7.1046	4.1940	5.9069	9.4800
20	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	14.6042	6.7919	4.4038	6.0228	9.9165
19	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	14.3179	6.4791	4.6355	6.1432	10.3952
18	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	14.0315	6.1663	4.8931	6.2686	10.9226
17	1.25	0.17	4.2	15	8.3	4.3	3.5	0.4	0.23	1	1	1	0	0.42	13.7452	5.8535	5.1809	6.3992	11.5063
21.9	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.1483	7.3861	4.152	6.304	9.57
22	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.1769	7.4174	4.1331	6.2921	9.5296
23	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.4633	7.7302	3.9534	6.1755	9.1440
24	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	15.7496	8.043	3.7887	6.0633	8.7883
25	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	16.0359	8.3557	3.6371	5.9550	8.4594
26	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	16.3223	8.6685	3.4972	5.8505	8.1542
20	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	14.6042	6.7919	4.5464	6.5388	10.4072
19	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	14.3179	6.4791	4.7857	6.6696	10.9096

Table 31: Data for employment contract ITP2 with added observations calculated using the life expectancy assumptions as described in section 2.2 - Part IV/IV

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$y^{(65)}$	$y^{(75)}$	$y^{(85)}$
18	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	14.0315	6.1663	5.0516	6.8057	11.4631
17	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	13.7452	5.8535	5.3487	6.9475	12.0756
16	0.75	0	13.8	5.54	23.31	22.37	22.37	0.4	0	0	0	1	0.325	0	13.4589	5.5407	5.6830	7.0952	12.7574
23.1	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	15.4919	7.7615	4.4	5.98	8.14
24	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	15.7496	8.043	4.235	5.8821	7.8551
25	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	16.0359	8.3557	4.0656	5.7771	7.5611
26	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	16.3223	8.6685	3.9092	5.6757	7.2882
27	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	16.6086	8.9813	3.7644	5.5779	7.0344
28	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	16.895	9.2941	3.63	5.4833	6.7977
22	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	15.1769	7.4174	4.62	6.1041	8.5176
21	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	14.8906	7.1046	4.84	6.2214	8.8926
20	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	14.6042	6.7919	5.082	6.3434	9.3020
19	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	14.3179	6.4791	5.3494	6.4703	9.7511
18	1.5	0	8.58	7.23	22.34	21.95	21.95	0.3	0	0	0	1	0.545	0	14.0315	6.1663	5.6466	6.6023	10.2457

References

[1] A. Paulsson, “Pension reform in sweden,” 2012.

[2] E. Arias, “United states life tables, 2003,” vol. 54, no. 14, pp. 8–9, 2007.

[3] R. A. Horn, R. A. Horn, and C. R. Johnson, *Matrix analysis.* Cambridge university press, 2 ed., 1985.

[4] O. Settergren and K. Birkholz, *Orange Rapport 2016 - Annual Report of the Swedish Pension System.* Pensionsmyndigheten, 2017.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[6] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015.

[7] D. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[8] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to linear regression analysis*, vol. 821. John Wiley & Sons, 2012.

TRITA -SCI-GRU 2018:362