Norwegian University
of Life Sciences

Master's Thesis 2020 30 ECTS
Faculty of Science and Technology

Analysis of Proteins From Cerebrospinal Fluid Tests in Search of Biomarkers Characterizing Multiple Sclerosis

Karim El-Hajj Eid
Master of Science in Data Science

Preface

This thesis is written at the Faculty of Science and Technology at the Norwegian University of Life Sciences (NMBU) in 2019. The thesis consists of 30 ECTS credits and marks the conclusion of a two-year masters degree in Data Science. This thesis has been carried out in collaboration with matforskningsinstituttet-Nofima.

I would like to thank my supervisors at NMBU, Professors Kristian Hovde Liland and Oliver Tomic, Faculty of Science and Technology (REALTEK), for their excellent feedback.

I would also like to thank my supervisor at Nofima Professor Ellen F. Mosleth for her excellent guidance, support and feedback in addition to her incomparable passion for research which i found very engaging.

And to my family and friends, very far and very wide, thank you for your feedback, your encouragement, and your support during my Master's thesis.

Ås, 26th May, 2020

Karim El-Hajj Eid

Abstract

There are contradicting theories describing Multiple sclerosis (MS). This study attempts to understand MS through interpreting the bio-markers of MS. Recursive feature elimination with cross validation (RFECV) was used to select bio-markers (proteins) for MS and to detect inflammation in patients. Principal components plots of proteins selected to distinguish between patients with MS and patients without MS were not only successful in separating patients with MS and patients without MS, but also showed two types of MS patients. The first type was MS patients with inflammation and the second type was MS patients without inflammation. This finding proposes inflammation to be a secondary effect of MS instead of a primary effect. The proteins in the principal component plots were related to inflammation and neuron development/regeneration. The types of proteins found together with the separate groupings of MS patients strengthen the hypothesis describing MS as a consequence of a defect in neuron regeneration where inflammation can sometimes but not always occur. This differs from the 'outside in model' that is often referred to when explaining MS which depicts the autoimmune response as the primary cause of neuro-degeneration. Finally, Logistic regression and support vector classifiers were trained to classify patients with MS and patients with inflammation. Models distinguishing MS had a score of 90 percent on the test data, while models classifying inflammation had a score of 98 percent on the test data.

Contents

List of Figures

Chapter 1

Introduction

1.1 Multiple sclerosis

Approximately two and a half million people are afflicted with a disease called Multiple sclerosis (MS) worldwide [1]. During the earliest descriptions of MS, it was observed that MS occurs more in women than in men, where women outnumber men with a ratio of two to one [2][3]. The average age of patients with MS is 34, but children and teens are also susceptible to the disease [4]. The risk of developing MS increases If the persons parents or siblings has had MS [5]. A widely accepted assertion is white people, particularly those of Northern European descent, are at a higher risk of developing MS. People of Asian, African or Native American descent have the lowest risk [6]. However, recent research suggests no correlation between ethnicity and MS [7]. Multiple sclerosis is not a completely understood disease [8]. There are conflicting theories which explain the origin behind MS [9].

1.1.1 Types

People affected by MS can be divided into 4 categories. Approximately 85% of patients belong to a category called relapsing-remitting MS (RR-MS). Patients belonging to this category experience a sudden start of worsening symptoms followed by complete or partial recovery. These episodes occur over a period. Patients may experience a gradual progression of symptoms between these episodes or experience full/partial recovery [10].

Approximately 50% of patients in RR-MS category may convert to secondary progressive MS (SP-MS), within 10 years from when the disease began. Patients in this category (SP-MS) experience gradual worsening of symptoms. Later, relapsing episodes may occur.

Around 10% of patients suffering from MS are categorized as primary progressive MS (PP-MS). These patients experience gradual worsening of symptoms from the start and without relapses [10].

5% of patients belong to the progressive relapsing category (PR-MS). They experience gradual worsening of symptoms from disease onset accompanied by one or more relapses. Much like secondary progressive MS, but the difference is patients suffering from MS in progressive relapsing category experience worsening of symptoms from the start [10].

1.1.2 Signs and symptoms

The signs and symptoms of MS depend on how much the nerves have been damaged and the location of the damaged nerves. Mobility, hand function, vision, fatigue, cognition, bowel and bladder function, sensory, spasticity, pain, depression, and coordination are all domains affected by MS [11]. People affected by MS experience some degree of impairment in most of these domains as early as the first year of disease. The degree of damage to these domains increases with disease duration, but the patterns of disability accumulation differ from person to person. There is no cure for MS, but treatments can help in managing the symptoms.

The following is a list of possible symptoms for people with MS [5]

- Electrical shocks associated with neck movements
- Problems with coordination due to tremors
- Loss of vison, often one eye at a time accompanied by pain during eye movement
- Double and blurry vision
- Problems with speech
- Fatigue and dizziness
- Problems with sexual, bowel and bladder function
- Low levels of vitamin D is associated with MS

1.1.3 The Outside in Model

Multiple sclerosis (MS) is a disease that affects the central nervous system. It is often characterized by Lesions appearing in the brain and the spinal cord [12]. MS is often described as an autoimmune, inflammatory disease that affects the central nervous system [5].

The 'outside in model' is the model that is often referred to when explaining the origin behind MS. The outside in model describes the autoimmune response as the

primary cause of neuro-degeneration. Autoimmune diseases are conditions where a person's immune system attacks its own organs. Usually the immune system sends out white blood cells when it detects foreign invaders. The immune system is able to distinguish between foreign cells and a person's own cells. Autoimmune diseases confuse organs in a person's body as foreign. As a result, the immune system releases proteins called autoantibodies that attack these mistaken cells [13]. The destruction or disruption of the body's own tissues by the immune system results from a complex interaction of genetic and environmental factors [14]. Autoimmune diseases can affect nearly every organ in the body. Three to five percent of the general population experience autoimmune diseases [15].

In the case of MS, the immune system attacks the protective layer called myelin sheath that covers nerve fibers [2]. The damage to this insulating layer slows down the speed of signals travelling through the nervous system causing communication problems between the brain and the rest of the body. The driving factors of MS have been separated into two mechanisms, inflammation and neurodegeneracy. These two mechanisms overlap each other and are intertwined, [8].

1.1.4 The Inside Out Model and Alternatives

There is overwhelming evidence of which the conclusion always attributes inflammation as a symptom of MS. This is due to assuming MS as an autoimmune disease. As mentioned previously, the Immune system is assumed as the primary driving force of neurodegenetation (outside in model). However, clinical experience has raised some troubling inconsistencies that cast doubt on this assumption, and it has been proposed that MS might instead be driven by a primary neurodegenerative disorder which triggers the immune system [8][10]. This model is known as the inside out model [16]. There are different factors that can cause neruodegeneration such as dysfunction in the lipid metabolism [17], vitamin B deficiency [18], and virus infections [19].

An altenative theory suggests that a lack of neuron building power disables the neuron to regenerate properly. A lack of neuron regeneration causes a poor myelin layer build. The Disturbed neuron development is found to be a characteristic of MS from early symptoms of the disease [20].

1.2 Problem statement

No single clinical feature or diagnostic test is enough for the diagnosis of MS [21]. There is currently a need for MS biomarkers that can help with earlier diagnosis, prediction and evaluation of disease progression, and for monitoring treatment response [22] [23]. The aim is to find biomarkers that can be measured to classify which group a patient belongs to. Different medication may then be prescribed to

patients with MS accordingly. Furthermore, the biomarkers that help in the diagnosis of MS may also shed some light on the primary cause of MS.

Chapter 2

Theory

2.1 Algorithms

A machine learning model is able to make predictions based on a set of observations. If the machine learning model is good, it is able to make accurate predictions. Accurate predictions are achieved by training the machine learning model. A machine learning model has the ability of learning from its mistakes. Training a model is done on a set of data that is usually called the training data. There are different types of machine learning models. These models distinguish themselves from each other in the way they make predictions. Logistic regression and support vector are two different types of machine learning algorithms.

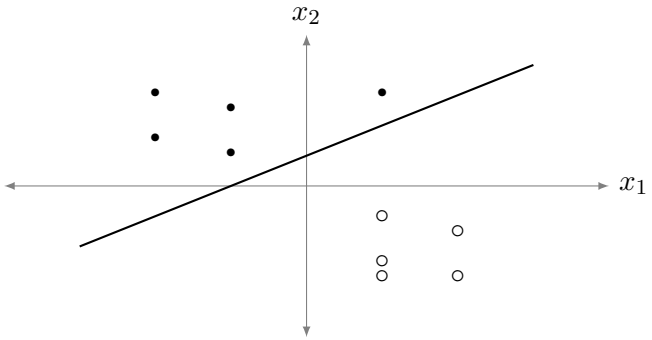

Figure 2.0: *Two linearly separable classes represented by black dots and circles*

Linearly separable classes are classes that can be separated by a hyperplane. Two linearly separable classes are shown in figure 2.0. In two dimensions, the hyperplane becomes a straight line.

x_1 and x_2 in figure 2.0 represent features. Features are input data to the machine

learning model. It is based on the features that the model is able to make predictions. The more relevant the input features are to the predictions the better the model is able to perform. The features that are relevant to predicting the output value must first be chosen when building a machine learning model. Each feature will have a weight associated with it.The features are often denoted as x_n and the weights as w_n.

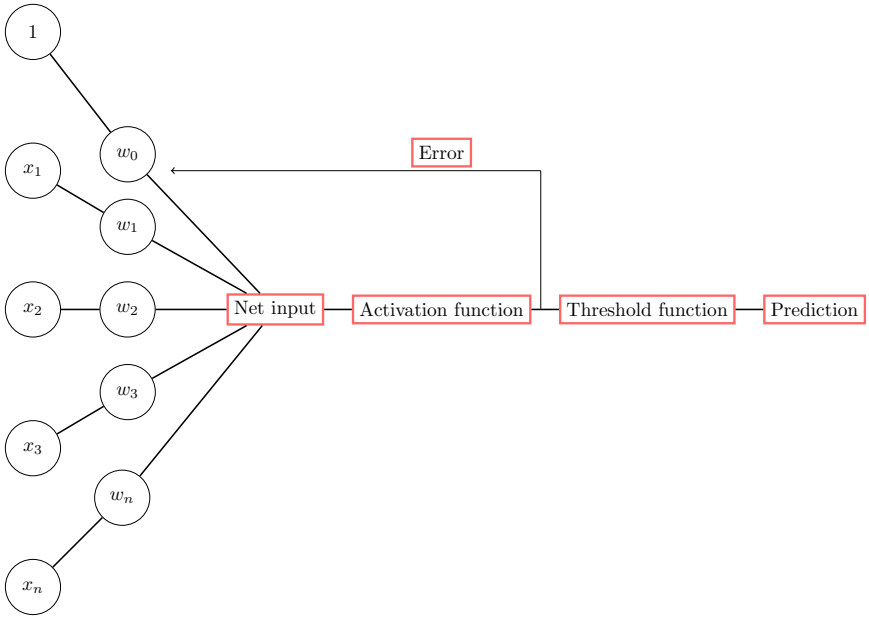

Figure 2.1: *Building blocks of a machine learning model*

Generally, a machine learning model consists of weights, an activation function, a cost function, an optimization algorithm, and a threshold function. The net input (z) in figure 2.1 is calculated by summing over the products of the features and the associated weights of the features.

$$z = \sum_{n=1}^{j} x_n * w_n, \tag{2.1}$$

Where *j* represents the number of input features.

2.1.1 Logistic Regression

Logistic regression is one of the most widely used algorithms for classification in industry[24]. It is a binary classification model for linearly separable classes. Binary classification is the task of classifying data which belongs to two classes.

2.1.2 Activation function

The net input (z) is passed through an activation function. The activation function is typically a fingerprint of the classifier. Changing the activation function can lead to a different model. To explain the activation function of the logistic regression model, the odds function is first introduced in equation 2.2.

$$O(p) = \frac{p}{1-p} \tag{2.2}$$

where p denotes the probability of an event happening. The odds function $O(p)$ describes the odds of an event occurring. The higher the odds function value the more likely it is for the event to occur. p describes the probability of the event occurring and it ranges from 0 to 1. A plot of the odds values as a function of the probability p is shown in figure 2.2.

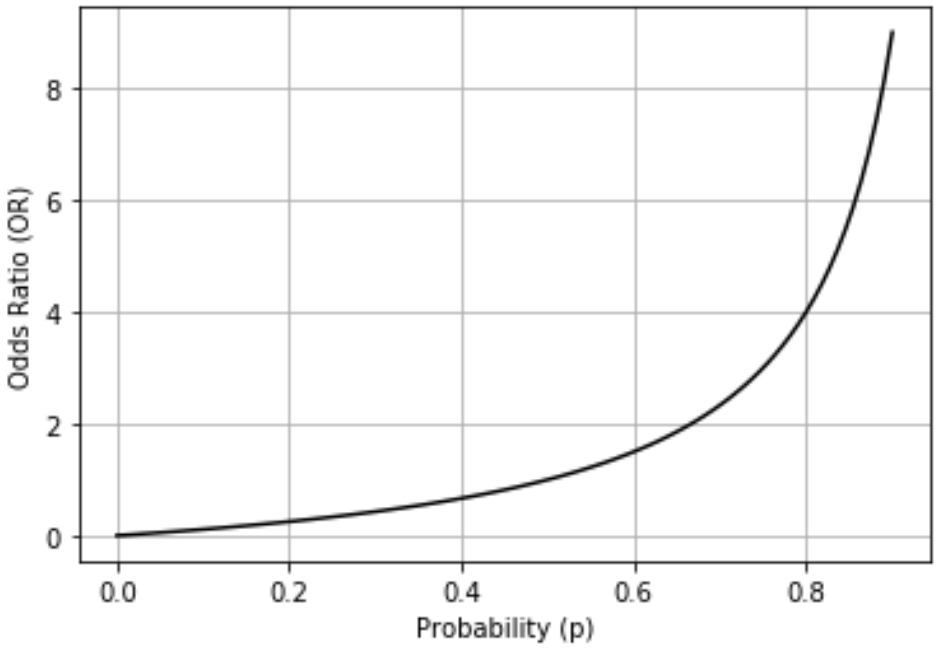

Figure 2.2: *Plot of the Odds Ratio (OR) as a function of the probability (p)*

The odds ratio can have values between 0 and up to positive infinity.

The logarithm of the odds function is called the logit function and its shown in equation 2.3.

$$logit(p) = log\frac{p}{1-p} \tag{2.3}$$

The purpose of the logarithm function is to take in probability values (in the range of 0 to 1) and map the values to the entire real number range. Figure 2.3 displays how the probability is mapped to the real values.

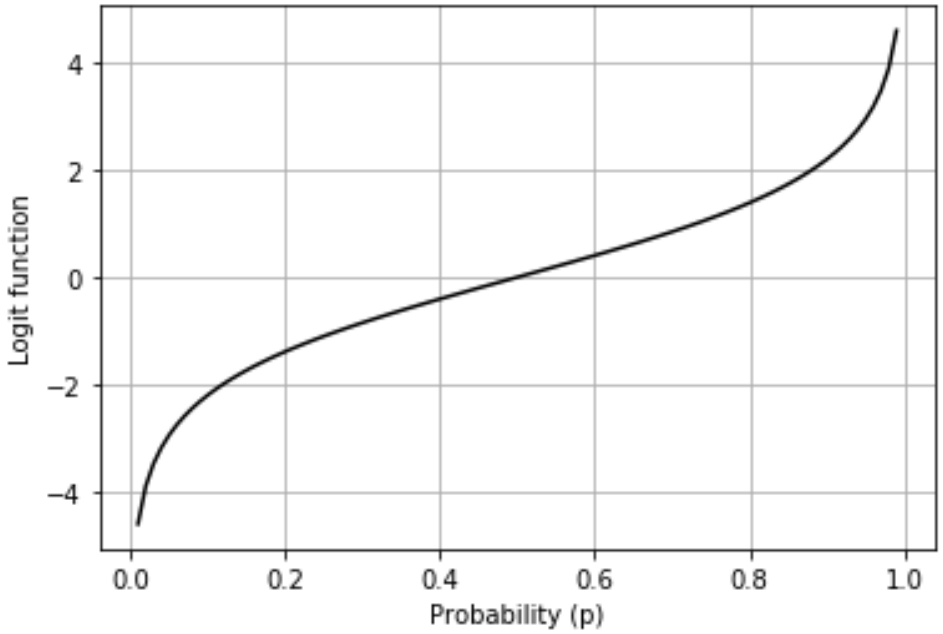

Figure 2.3: *Plot of the Logit function as of function of the probability (p)*

The logistic regression model predicts the probability that a certain sample belongs to a certain class. This is the inverse of the logit function. The logit function takes in a probability value and maps it to a value in the real number line. Therefore, the inverse of the logit function accepts a real value and transforms it to a probabilistic value. The inverse of the logit function is shown in equation 2.4.

$$\phi(z) = \frac{1}{1+e^{-z}} \tag{2.4}$$

The inverse of the logit function is called the logistic function and is the activation function used in logistic regression models. It is also called the sigmoid function due to its 's' shape. The sigmoid function is shown in figure 2.4.

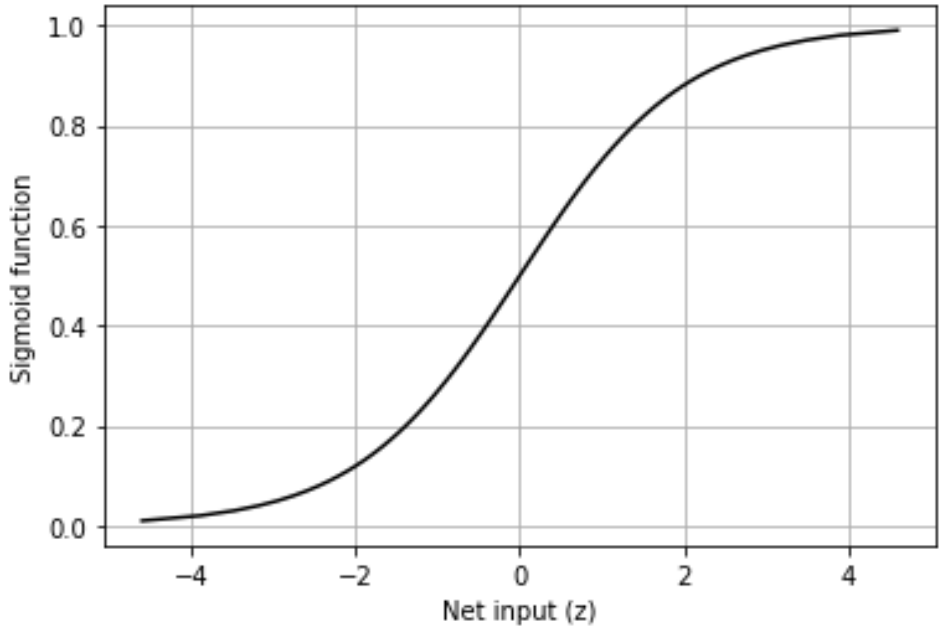

Figure 2.4: *Plot of the sigmoid function as of the net input (z)*

A sample is classified based on the value of the sigmoid function. The output of the sigmoid function is interpreted as the probability of a certain sample belonging to class 1. If the output of the sigmoid function for a particular sample is 0.1, this means that the probability of the sample belonging to class 1 is 0.1. The probability that the sample belongs to class 2 is 1-0.1 = 0.9, and hence the sample is classified as class 2.

The output of the sigmoid function can be converted to a binary outcome by the following threshold function

$$\hat{y} = \begin{cases} 1, & \phi(z) \geq 0.5 \\ 0, & \text{otherwise.} \end{cases} \tag{2.5}$$

The threshold function can also be expressed as a function of the net input (z).

$$\hat{y} = \begin{cases} 1, & z \geq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{2.6}$$

2.1.3 Cost Function

The weights of the logistic regression model are chosen randomly at first. The randomly chosen weights in combination with the sample values will not be able to

correctly classify the samples into their respective classes. To improve the model, the weights must be optimized in a way that minimizes the cost function.

The aim of a cost function is to compute the error of a wrongly classified sample. The purpose of minimizing the cost function is to update the weights in such a way that gives the model a higher predictive accuracy. To understand the cost function of the logistic regression classifier lets first introduce the likelihood L that is maximized when building a logistic regression model as shown in equation 2.7.

$$L(w) = \Pi_{i=1}^{n} P(y|x;w) = \Pi_{i=1}^{n} \phi(z^i)^{y^i}(1-\phi(z^i)^{1-y^i}) \tag{2.7}$$

Where $\phi(z^i)$ is the previously defined sigmoid function and *i* is used to denote the ith sample data. The Likelihood function L is the product of all the probabilities. Where a probability is that of a sample belonging to a class y given the weights of the model. The higher the likelihood L, the more certain the logistic regression model is of its own predictions. It is more practical to maximize the natural log of this equation. The natural log of the likelihood function is called the log-likelihood function:

$$l(w) = logL(w) = \sum_{i=1}^{n}[y^i log\phi(z^i) + (1-y^i)log(1-\phi(z^i))] \tag{2.8}$$

The products of the probabilities in the likelihood function conveniently becomes a summation task in the log-likelihood function minimizing the potential of numerical overflows and making it easier to calculate derivatives. $l(w)$ becomes the cost function of the logistic regression model. The cost function in this case is maximized instead of minimized. The cost function can be manipulated so that it gets minimized instead. The cost function is minimized or an alternative is to maximize the log likelihood function depending on the optimization algorithm that is chosen. In this case, an optimization algorithm is used to maximize the log likelihood function.

$$J(w) = \sum_{i=1}^{n}[-y^i log\phi(z^i) - (1-y^i)log(1-\phi(z^i))] \tag{2.9}$$

To better understand the log-likelihood function lets consider a single sample.

$$J(\phi(z), y; w) = [-ylog\phi(z) - (1-y)log(1-\phi(z))] \tag{2.10}$$

The value y has two possible values since the model is a binary classification model. The possible y values are 0 and 1. Depending on the y-value either the

first term or the second term becomes zero in equation 2.10. This is further emphasized in equation 2.11.

$$J(\phi(z), y; w) = \begin{cases} -log(\phi(z)), & y = 1 \\ -log(1 - \phi(z)), & y = 0 \end{cases} \tag{2.11}$$

A plot of the cost function with two different y values paints a clearer picture in the underlying structure of the cost function.

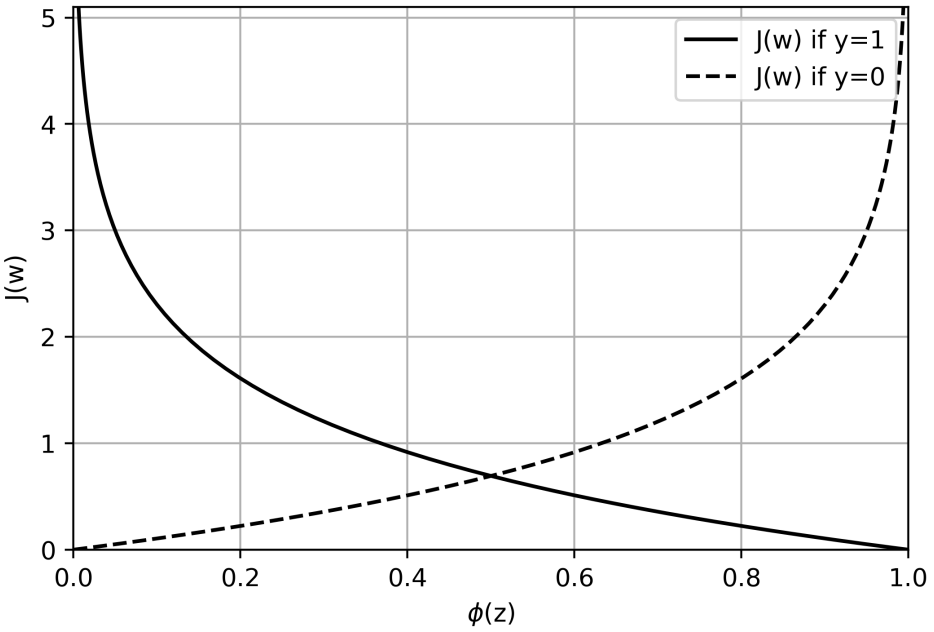

Figure 2.5: *Plot of the cost function for the two possible y-values given the activation* $\phi(z)$

The plot of figure 2.5 shows how the cost function changes depending on the y-value. If y = 1, the cost function goes towards 0 as $\phi(z)$ outputs values closer to 1. The output $\phi(z)$ denotes the probability that a sample belongs to class 1. Therefore it makes sense that the cost function decreases as the model correctly classifies the sample as belonging to class 1. For cases where the target value is 0, the cost function is represented by the dashed line. An increase in the model's output in this case increases the cost function as the model is classifying the sample as belonging to class 1 whereas in reality it belongs to class 0. This construction of the cost function allows for penalizing wrong predictions with a high value of the cost function. Gradient descent which is introduced later in this chapter, gives a direction for the weights to update in order to minimize the cost function and make correct predictions.

2.1.4 Support Vector Classifier

The goal of a support vector classifier is to find a M dimensional hyperplane that separates the data points correctly while maximizing the margin. The margin is defined as the distance between the hyperplane and the closest samples to it. The samples closest to the hyperplane are called the support vectors.

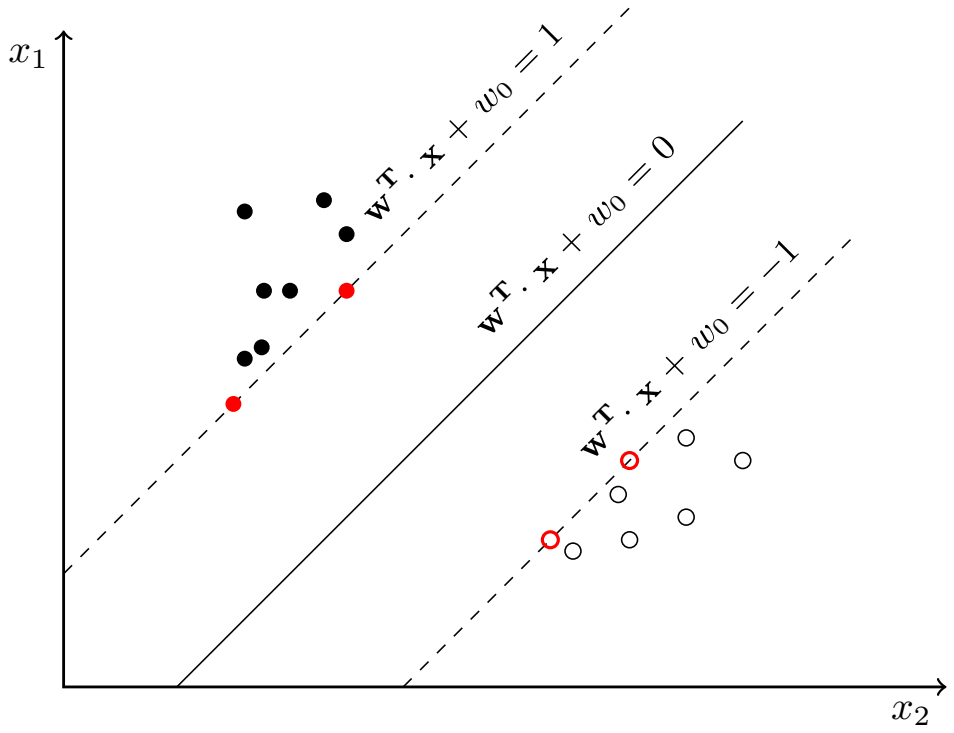

Figure 2.6: *A support vector classifier with an optimized margin showing the support vectors*

In figure 2.6, the dots represent samples that belong to one class and the circles represent samples that belong to another class. The support vectors are represented in the figure by the red dots and the red circles. They are the closest samples to the hyperplane boundary. The margin is the distance between the support vector machines. The decision boundary is shown in the figure as a straight line between the two hyperplanes.

The idea behind maximizing the margin is that models with high margins will be less prone to overfitting. For many datasets, multiple margins can be drawn which correctly classify the samples. The decision boundary that maximises the margin however, is the margin with the biggest separation between the boundary cases. Therefore, small changes in the sample values will be less likely to affect the classification. In the case of binary classification, the hyperplane divides the

space into two classes. This is done the following way:

$$\mathbf{w^T x_{pos}} + w_0 = 1 \tag{2.12}$$

$$\mathbf{w^T x_{neg}} + w_0 = -1 \tag{2.13}$$

x_{pos} represents samples that are located on the positive side hyperplane and x_{neg} represents samples that are located on the negative hyperplane.

Substracting equation 2.13 from equation 2.12 yields equation 2.14.

$$\mathbf{w^T}(\mathbf{x_{pos}} - \mathbf{x_{neg}}) = 2 \tag{2.14}$$

To normalize the result, the length of the vector w is calculated the following way:

$$||w|| = \sqrt{\sum_{j=1}^{m} w_j^2}. \tag{2.15}$$

Dividing equation 2.14 by the norm of w gives equation 2.16.

$$\frac{\mathbf{w^T}(\mathbf{x_{pos}} - \mathbf{x_{neg}})}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||} \tag{2.16}$$

Due to the way equation 2.16 is formalized, the left hand side of the equation can be interpreted as the gap between the negative hyperplane and the postive hyperplane. This gap is referred to as the margin, where the objective is to maximize the margin by maximizing $\frac{2}{||w||}$ while correctly classifying the samples.

$$y^{(i)} = \begin{cases} 1, & \mathbf{w^T x^{(i)}} + w_0 \geq 1 \\ -1, & \mathbf{w^T x^{(i)}} + w_0 \leq -1 \end{cases} \tag{2.17}$$

The constraints of correctly classifying the samples is expressed mathematically in equation 2.17. for $i = 1...N$, where N is the total number of samples. All samples with a negative score should be placed in the negative hyperplane, and all the positive samples should be placed in the positive hyperplane.

It is not always the case that the data are perfectly linearly separable. Even in the case of linearly separable data, there might be outliers that cross a linear boundary between the two hyperplanes. These outliers can be a result of a systematic error due to the instruments used or it can be due to random noise error that has found its

way in to the data. In or to adjust for non-linearly separable data, slack variables are introduced. Slack variables soften the constraints of linearly separable data, allowing convergence despite the presence of misclassifications.

$$y^{(i)} = \begin{cases} 1, & \mathbf{w^T x^{(i)}} + w_0 \geq 1 + \xi^{(i)} \\ -1, & \mathbf{w^T x^{(i)}} + w_0 \leq -1 + \xi^{(i)} \end{cases} \tag{2.18}$$

The slack ξ is added to the linear constraints as shown in equation 2.18. As in the previous equation $i = 1...N$, where N is the total number of samples. In this way, the objective that is maximized becomes:

$$\frac{||\mathbf{w}||^2}{2} + C(\sum_i \xi^{(i)}). \tag{2.19}$$

The variable C in equation 2.19 is a regularization term. It is used to control the cost of a misclassification. The bigger the value of C, the larger the penalty of misclassification. If a bigger bound of error is tolerated, a smaller value of C is used. This way, the value of C controls the width of the margin.

2.2 Optimiser

The models built in a machine learning context have to have an objective that ought to be optimized for the model to perform at its best. In logistic regression models, it is either the Maximum Likelihood function that is maximized or the cost function which is minimized. In support vector classifiers, it is the margin that is optimized. An optimizing algorithm fulfills this role. There are different types of optimizing algorithms. Gradient descent, coordinate descent, and L-BFGS are examples of optimizing algorithms used in machine learning models. These three optimization algorithms are explained below.

2.2.1 Gradient descent

Gradient descent is one of the most popular optimizing algorithms used to optimize the objective function during the training phase. The objective function to be optimized is the cost function, which depending on the machine learning algorithm is defined accordingly. The goal of gradient descent is to find the weights that minimize the cost function.

Gradient descent can be visualized as going down a hill until a local or global minimum is reached. The hill represents the value of the cost function. Traversing down the hill is equivalent to finding weights which reduce the value of the cost function. Mathematically this is equivalent to taking a step in the opposite direction

of the gradient. The gradient of a function at a given point describes the steepness of the graph at that point. Taking a step in the opposite direction at that point, becomes equivalent to travelling down the hill. The size of the step taken down hill is controlled by the learning rate. The higher the learning rate the bigger the step taken down hill.

The learning rate determines the size of the step taken down hill. It is important to choose a suitable learning rate for the problem at hand. If the learning rate chosen is too small gradient descent takes a long time to finish running. This is shown in the figure below.

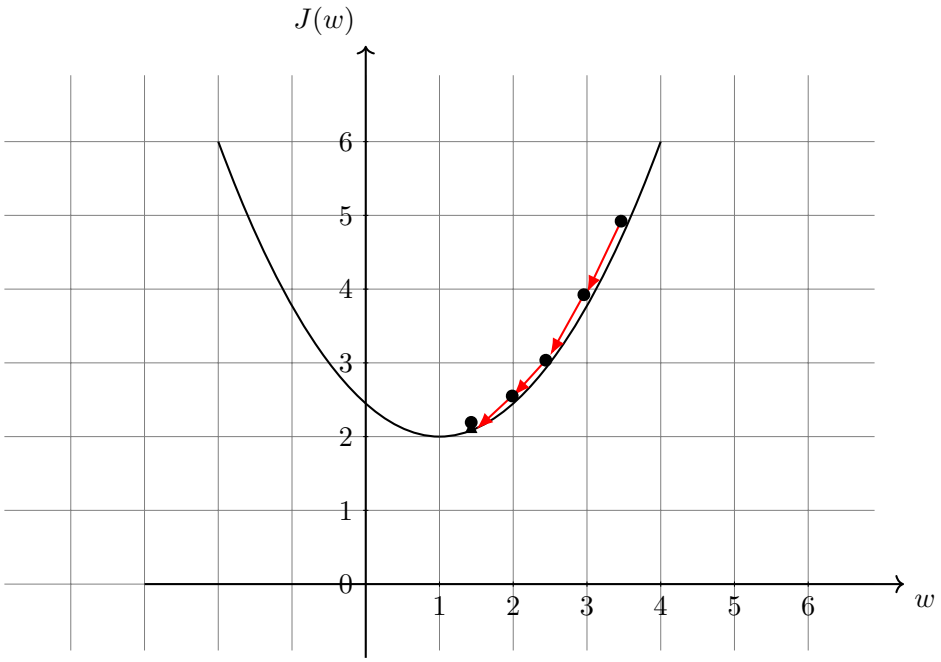

Figure 2.7: *Plot showing gradient descent with slow learning rate*

However if the learning rate is too big, gradient descent might not converge. The step taken downhill overshoots the minimum spiraling outwards as shown in the the figure below.

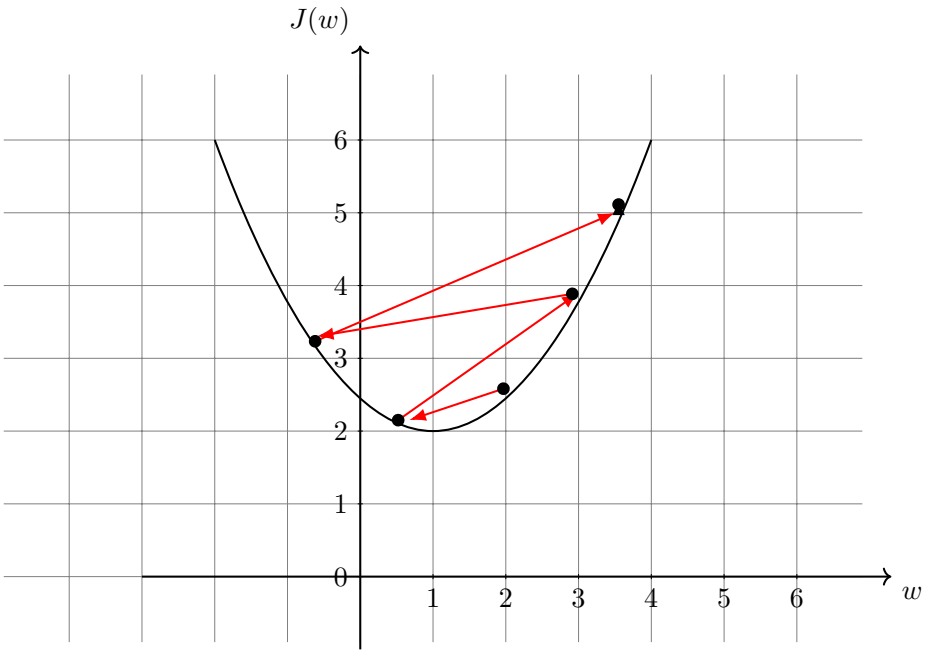

Figure 2.8: *Plot showing gradient descent with fast learning rate*

Each iteration of gradient descent updates the weights by traversing a step opposite the direction of the gradient of the cost function. The weights are updated the following way:

$$w := w + \Delta w \tag{2.20}$$

where tΔw is calculated by multiplying the learning rate by the negative gradient of the cost function.

$$\Delta w = -\eta \nabla J(w) \tag{2.21}$$

If the cost function contains two weights gradient descent can be visualized in a contour graph. Instead of traversing down the cost function in one direction, a second direction is taken into account when finding the direction of steepest descent. In practice however, it is common to have many more weights making the visualization of the actual decent impossible. Figure 2.9 shows the path taken of the weight vector guided by gradient descent for a cost function containing two weights.

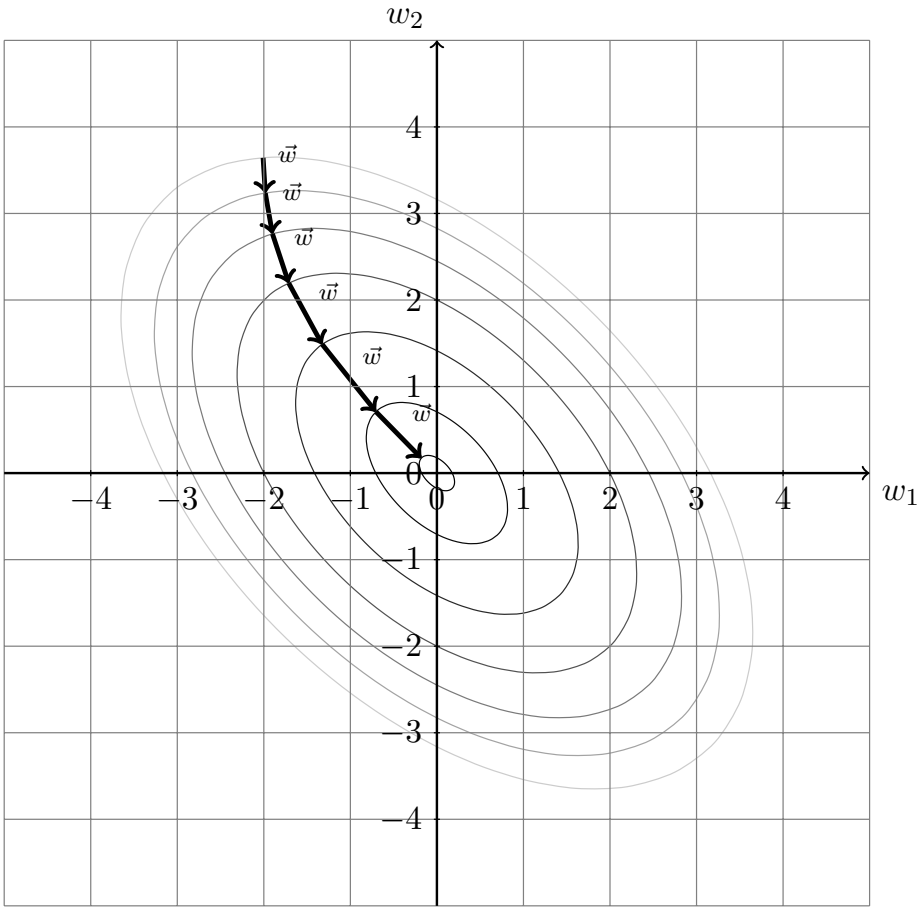

Figure 2.9: *Plot showing gradient descent of a cost function having two weights*

2.2.2 Coordinate descent

Coordinate descent is an optimization algorithm used in the Library for Large Linear Classification (Liblinear) [25]. It has the same objective as Gradient descent, to find the weights which minimize the cost function. However, it does so slightly differently. Coordinate descent computes the gradient of the cost function for one weight at a time keeping the rest of the weights constants.

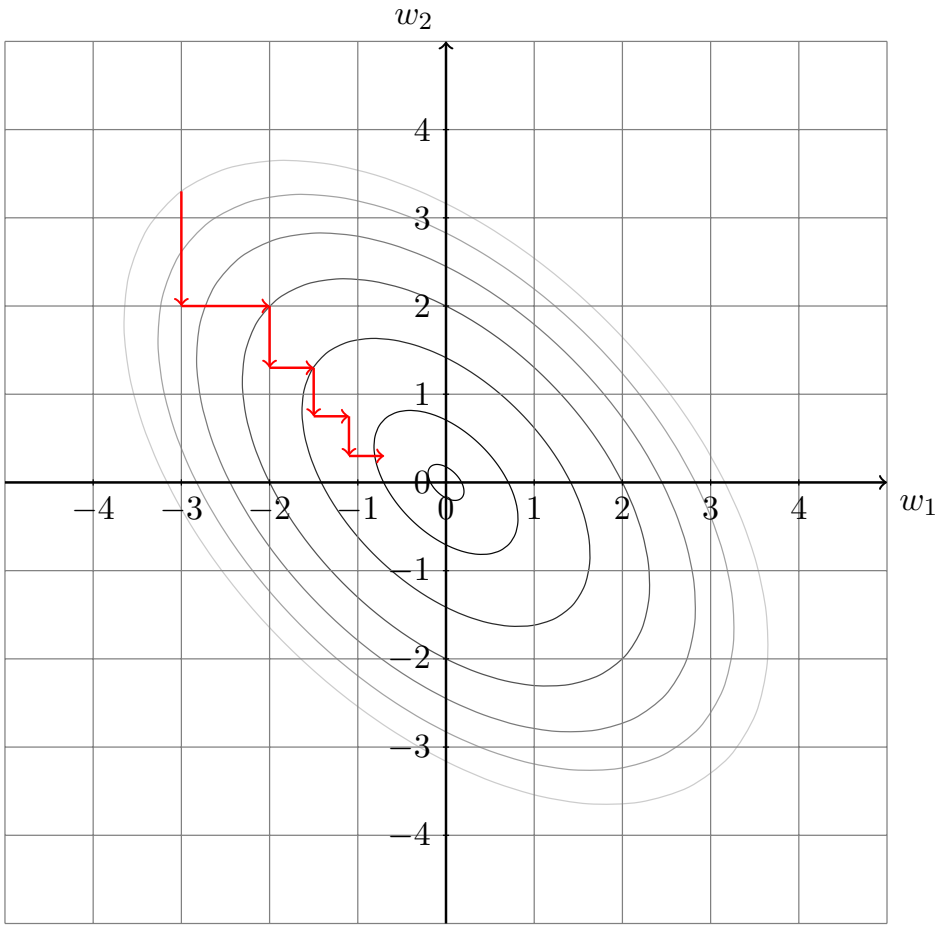

Figure 2.10: *Plot showing the path traversed by the weights due to coordinate descent*

The coordinate descent algorithm for a cost function with two weights produces a stair like path as illustrated in figure 2.9. One weight is optimized at a time, moving closer to the local or global minimum

Solving the optimization problem using coordinate descent is less complex than using gradient descent [26]. Coordinate descent can be used most of the time even when there is no closed form solution for the cost function. However, coordinate descent,can get stuck in certain functions, due to it only being able to move along one direction at a time.

2.2.3 Limited-Memory Broyden-Fletcher-Goldfarb-Shanno Algorithm

Limited-Memory Broyden-Fletcher-Goldfarb-Shanno Algorithm (L-BFGS) is an optimization algorithm used in machine learning models. The main difference between this approach and the previous optimization algorithms is that L-BFGS in addition to using information on the first derivatives of the cost function, second derivatives are also used to find the direction towards the minimum.

The optimization algorithm was created by Jorge Nocedal at the Argonne Antional Laboratory and Northwestern University (reference). This optimization algorithm is derived from BFGS with the objective of using less memory. Hence, the name limited memory BFGS (L-BFGS). This makes L-BFGS suited for large datasets.

The L-BFGS algorithm belongs to the Quasi-Newton family. The Quasi-Newton algorithms locate maximums/minimums of functions utilizing Newton's method.

A brief introduction to L-BFGS is presented since a more encompassing explanation will require heavy mathematical manipulation.

Assume a start value w_n, the goal is to estimate w_{n+1} such that $f(w_{n+1}) < f(w_n)$.

Points close to $f(w)$ can be approximated using a quadratic approximation of $f(w)$. This is done using a Taylor expansion:

$$f(w + \Delta w) \approx f(w) + \Delta w^T \nabla f(w) + \frac{1}{2}\Delta w^T (\nabla^2 f(w))\Delta w, \tag{2.22}$$

where $\Delta w = w_{n+1} - w_n$, $\nabla f(w)$ is the gradient of f, and $\nabla^2 f(w)$ is the second derivative of f.

For simplicity's sake, equation 2.22 can be rewritten in terms of quadratic approximation h_n. Equation 2.22 becomes:

$$h_n(\Delta w) = f(w_n) + \Delta w^T g_n + \frac{1}{2}\Delta w^T H_n \Delta w \tag{2.23}$$

g_n represents the gradient of f at w_n, H_n is the Hessian of f at w_n. The Hessian is defined as a square matrix containing the partial second derivatives of a function.

The objective is to find the direction which minimizes the quadratic approximation of f (which is h_n) at the point w_n. A partial derivative with respect to Δw follows:

$$\frac{\partial h_n(\Delta w)}{\partial \Delta w} = g_n + H_n \Delta w \tag{2.24}$$

Setting equation 2.23 to zero and assuming H_n to be positive definite, gives a way of finding the minimum of h_n. The result is shown in equation 2.25.

$$w_{n+1} = w_n - H_n^{-1} g_n \tag{2.25}$$

The inverse Hessian and the gradient of a function f at a point w_n, can be used to find a new w-value which is closer to the minimum of f. It can be proven given a convex function, that the above procedure converges to a global minimum independent of the start value w. However for functions that are non-convex, convergence is only guaranteed to local minimums. This is still Newton's method.

The inverse Hessian matrix that is required for Newton's method to work is often difficult to calculate. Machine learning models are often built with many input features. The inverse of the Hessian matrix becomes computationally expensive or even impossible to calculate as the total number of dimensions increase.

As a result, Newton's method is not used for large problems. To combat this obstacle, the Hessian of f is approximated using previously computed w-values, gradients of f, and hessian matrices. This method of approximating the Hessian is named the BFGS update. The formula for BFGS update is not presented as it will not provide further understanding.

The BFGS update requires storage of all the previously computed gradients of f, and their coordinates. The limited BFGS (L-BFGS) uses only the last m number of gradients and their coordinates instead.

2.3 Feature Extraction and Feature Selection

Features are key ingredients to a machine learning model. The model can only make predictions based on the information in the input features. Therefore it is of utmost importance that all features contain important information about the predicted value. Feature selection and feature extraction becomes very important once the number of features increases. Having too many features compared to the number of samples often results in what is called the curse of dimentionality.

The curse of dimentionality is a phenomena where sparsity of data occurs due to a high number of dimensions. The n-dimensional space quickly becomes sparsely populated when the number of dimensions n grow. Sparsity of data can be thought of as 'closeness of data'. The volume of represented space grows very fast as the number of dimensions increase. As a result, the data becomes sparse. In addition, the higher the number of features, the more samples are required to successfully train a model.

Often in real world problems, a dataset is presented containing many features. The first challenge when building a model is to find the relevant features to the model.

Having irrelevant features as inputs to a model produces good results on the training set but the model performs poorly on the test set. The model uses the irrelevant features to overfit the data during the training phase.

Feature extraction and feature selection are two different dimensionality reduction techniques that are used when dealing with high dimensional data set.

2.3.1 Feature Selection

The purpose of feature selection, like feature extraction, is to reduce the dimensionality of the data by selecting only the relevant features to the model.

After feature selection is done, the features that are selected can be directly found in the original dataset. This is great for directly interpreting which feature is relevant to the model's predictions. Sometimes it can be important to know what features the model needs to predict the output. It can help to understand the underlying phenomena that is present. During feature extraction however, feature interprebility might not be as explicit.

There are many techniques used in feature selection. L1 regularization and recursive feature elimination can be used a feature selection technique.

Recursive Feature Elimination

Recursive feature elimination is a type of backward selection technique. A model is built using all of the available features. The chosen model has to have an importance score such as coefficients or feature importance. This is because after the model has been trained on the whole set of features, coefficients or feature importances are computed for each feature. The features with the lowest importance scores are then dropped. This process is continued recursively until the desired number of features is reached or all the features are eliminated.

Recursive feature elimination with cross validation utilizes recursive feature elimination in addition to finding the cross validation score for every step. Where a step is when a feature is eliminated. The score is calculated on the validation data. The set of features that give the highest cross validation score is the final set chosen.

In essence, recursive feature elimination with cross validation performs a recursive feature elimination in a cross validation loop for outputting the best features.

To remove a feature feature importance is needed. To decide which feature set to use cross validation scores are used.

2.3.2 Feature Extraction

Feature extraction is the process of reducing the dimensionality of data through transforming or projecting the data onto a new feature subspace. The aim of feature extraction is to express the relevant information present in the original feature set but in a lower dimension of space.

Principal component analysis

Principal Component Analysis (PCA) is an unsupervised linear transformation technique. PCA is used to reduce the dimensionality of a dataset to increase the interpretability while keeping information loss to a minimum. The goal of PCA is to find directions of maximum variance in a data set and project the data onto those directions. The principal components can be interpreted as the directions of maximum variance given the constraint of having the new feature axes orthogonal to each other.

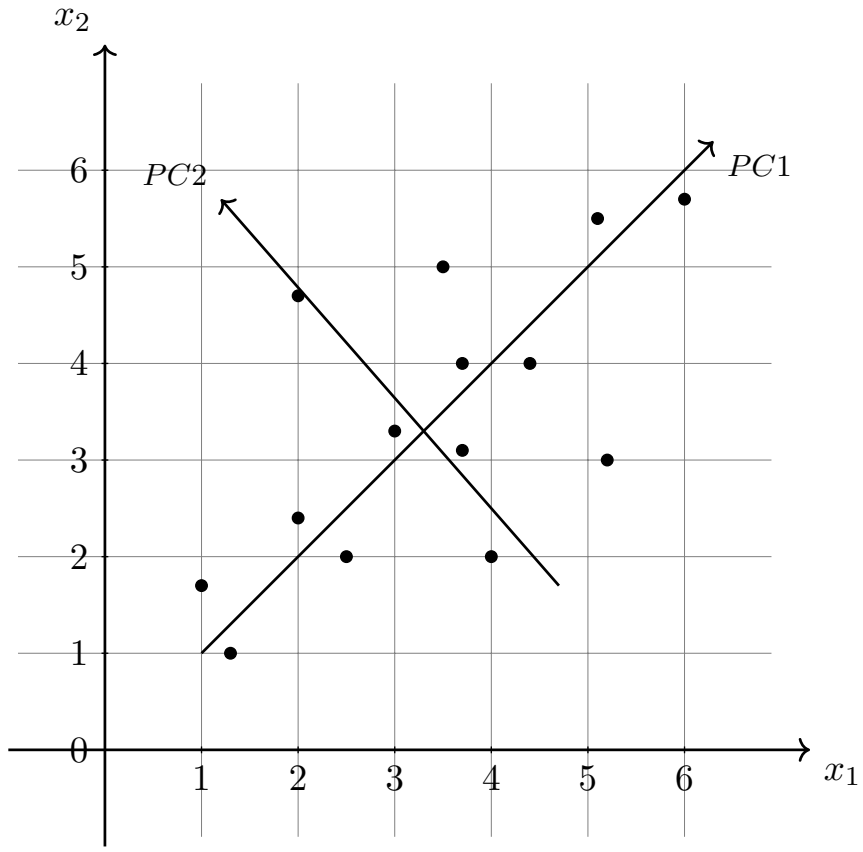

Figure 2.11: *Plot showing an illustration of a principal component decomposition into principal component (PC) 1 PC2*

In figure 2.11, x_1 and x_2 represent the original features. $PC1$ and $PC2$ represent the new feature axes.

PCA constructs a dxk matrix that transforms a vector x of d dimensions onto a new k-dimensional feature subspace. Here $d \leq k$.

$$x = [x_1, x_2, .., x_d], x \in \Re^d \tag{2.26}$$

$$xW, W \in \Re^{d \times k} \tag{2.27}$$

$$z = [z_1, z_2, .., z_k], x \in \Re^k \tag{2.28}$$

The matrix W is setup in such a way that the first component will correspond to the direction of maximum variance. The second component will have the second largest direction of variance and so forth. These components are orthogonal to each other and so they are uncorrelated. This means even if the original features are correlated, the principal components will be uncorrelated.

Since PCA searches for directions of maximum variance, standardizing the features will give different results than doing a PCA analysis on features that are not standardized. If the features are not on the same scale, the principal components will be biased towards the features with the highest variance. Standardization of features is done depending on the type of data at hand.

To find the directions of maximum variance a covariance matrix first has to be constructed. To construct the covariance matrix the features, the covariance between two features x_j and x_k must be calculated.

$$\sigma_{jk} = \frac{1}{n}\sum_{i=1}^{n}(x_j^i - \mu_j)(x_k^i - \mu_k), \tag{2.29}$$

σ_{jk} is the covariance between feature j and feature k. μ_j and μ_j are the sample means of feature j and k respectively. A positive covariance between two features indicate that the features increase or decrease together. A negative covariance means when one feature increases the other feature decreases.

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{bmatrix} \tag{2.30}$$

The eigen vectors of the covariance matrix represent the principal components and the corresponding eigen values represent their magnitude.

$$\Sigma v = \lambda v \tag{2.31}$$

Now the data needs to be projected onto the new feature space. This is done by first defining the transformation matrix W. The transformation matrix W is constructed from the top k eigenvectors.

$$X' = XW \tag{2.32}$$

Explain score and loading for the biplots.

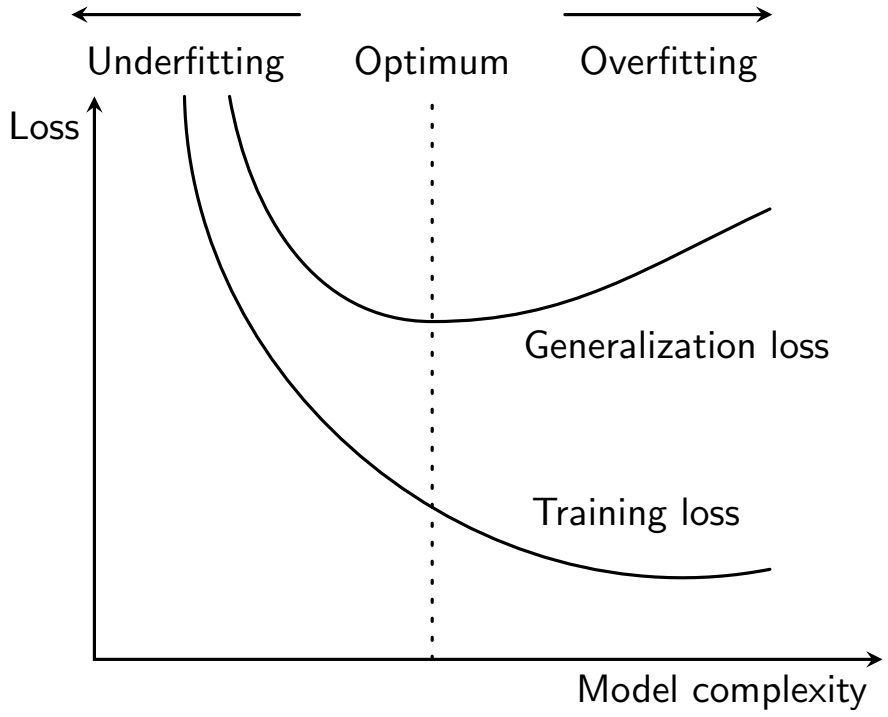

Figure 2.12: *Plot showing how model complexity can cause overfitting or underfitting*

2.3.3 Cross validation

A model without an estimate of its performance is a useless model. This is because there is no telling how good the model is. It is therefore important to measure how well a model performs. Cross validation is a method for measuring model performance on unseen data. It helps in measuring the degree of overfitting or underfitting a model might be suffering from.

Underfitting happens when the model does not learn the structure in the data. The model has low performance on both the training set and the test set. Whereas overfitting occurs when the model has high performance on the training data, but very low performance on the test data. When a model overfits, it does not generalize well to unseen data. The model is often learning noise in the training data. The optimal model has good performance on both the test and training set.

The higher the model complexity the easier it is for the model to overfit the training data. Figure 2.12 shows how the generalization loss (the ability to correctly classify unseen data) increases when the model is overfitting. This happens while decreasing the training loss. One way of increasing model complexity is by using

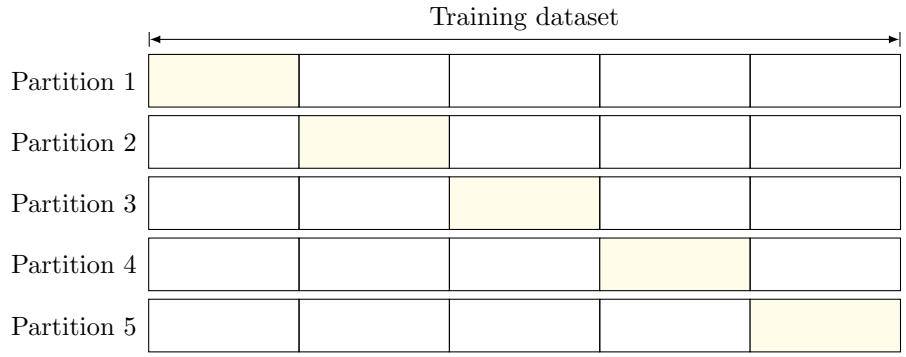

Figure 2.13: *5-fold cross validation is performed. The Yellow tainted boxes represent the held out data used to estimate model perfomance*

a high number of features. Using very few features can lead to a model that underfits. During underfitting both the generalization loss and the training loss are high. This is shown in the left side of figure 2.12. It is therefore important to build a model that is as close to the optimum point as possible. To estimate a model's perfomance the model is given unseen data and asked to make a prediction.

The dataset is split into training and test data. The ratio of the split is determined based on the size of the dataset. A typical split is a 70/30 where 70% of the data is used as training data and the remaining 30 percent is used as test data. The training data is split into K-folds. The number K is chosen based on the size of the training data. The model is trained on K-1 folds and evaluated on the fold that was held out. This is repeated for each fold. A 5-fold example is shown in figure 2.12.

If the model was to be tested on 5-fold cross validation then there would be 5 different estimates of the models performance. The average and standard deviation of the model's perfomance is calculated using the 5 measurements. This is a good estimate of the model's performance.

If the number of folds reaches the number of samples in the training set, then the technique is called leave on out. The model leaves out one sample, trains on remaining training set and evaluates the model on the left out sample. This is repeated for all samples in the training set. The negative side of leave one out technique is that it can be computationally very expensive. The positive side is that the performance variance of the model is reduced.

K-fold cross validation can be performed to find the optimal hyper parameters for a model. Hyperparameters are defined externally to a model. They cannot be directly computed from the given data set, but are approximated. The hyperparameters with the best performance using k-fold cross validation are chosen and the final estimate of the performance of the model is found on the test set after the hyperparameters

are chosen. K-fold cross validation can also be used for model selection.

K-fold cross validation gives an unbiased measure of the model performance on the training data which can be compared to the model performance on the test data.

2.3.4 L2 Regularization

L2 regularization is implemented in logistic regression models. L2 Regularization is a method of preventing overfitting. It does so by reducing the complexity of the model. The idea behind L2 regularization is to penalize large weight coefficients. To understand how L2 regularization works lets understand how it is implemented. A regularization term is added to the cost function.

$$\frac{\lambda}{2}||w||^2 = \frac{\lambda}{2}\sum_{j}^{m} w_j^2 \tag{2.33}$$

The regularization term introduces new constraints that an optimization algorithm must maintain when minimizing the loss function. λ is called the regularization parameter. The effect of adding this regularization term is to constrict the weights from becoming too big. By adjusting the value of λ, the degree of regularization in the model is tuned. An increase in the value of λ causes more regularized models.

In practice, instead of adjusting for λ values, the C-value is adjusted. The C-value is defined as the inverse of λ.

$$C = \frac{1}{\lambda} \tag{2.34}$$

Hence, a decrease in the value of C means an increase on the regularization strength of the model.

A logistic regression model with L2 regularization contains an extra term. The new regularization term is added to the cost function. The new cost function becomes:

$$J(w) = \sum_{i=1}^{n}[-y^i log\phi(z^i) - (1-y^i)log(1-\phi(z^i))] + \frac{\lambda}{2}\sum_{j}^{m} w_j^2 \tag{2.35}$$

2.4 Dataset

The dataset that is analyzed in this research paper was provided to the master student by Nofima, a food research institution. The dataset contains a total of 101

patients, of which 37 patients have MS and 64 patients do not have MS.

This research paper builds on a previous cluster analysis [27]. The previous analysis is done on the same dataset as the present research paper. The patients in the previous analysis [27] where separated into two clusters, cluster 1 and cluster 2. Both clusters contained patients with and without MS. However, further research on the cluster analysis [20] showed patients in cluster 2 having inflammation present, whereas patients in cluster 1 did not have inflammation. When cluster 1 and cluster 2 is mentioned in this paper it is referring to those clusters.

2.4.1 Data preparation

The dataset was already preprocessed when it was obtained. Among other things, a log transformation was done and a standardization of the data. Ideally the dataset should be obtained unprocessed. The test data should be standardized using the train data to simulate real unseen data. However, due to the test data and train data being processed together, information from the training data is leaked into the test data. Therefore, the test results may be skewed.

Chapter 3

Methods

3.1 Software

Python version 3.6 is used with the following libraries: scikit-learn [28], pandas [29], and numpy [30]. The analysis is performed in Jupyter notebooks [31].

3.2 Data preprocessing

The data that was obtained was already preprocessed. It was log transformed followed by a standardization of the data set. As a result the data set has a mean of 0 and a standard deviation of 1. 90 percent of the data was used as training data and the remaining 10 percent was used as test data. The 90/10 split was performed randomly. The training data was used for RFECV and model training. The test data was used to evaluate model performance. PCA was done on the whole data set using features obtained by RFECV on the training set. The dataset was not heavily class imbalanced.

3.3 Classification

The data set contains 4 classes. The first class, class 1 are non-MS patients belonging to cluster 1. The second class, class 2 are MS patients in cluster 1. The third class, class 3 are non-MS patients in cluster 2. The last class is class 4 which represents MS patients belonging to cluster 2.

Feature selection was performed using recursive feature elimination with cross validation (RFECV). The model's performance is the criterion used during RFECV with accuracy being the perfomance metric measured. RFECV was done on the

training data. RFECV was used on a Logistic Regression model with 'lbfgs' as a solver, a logistic regression model with 'liblinear' as a solver, and a linear support vector classifier.

RFECV was performed based on two objectives. The first objective was to find proteins (features) that separate the two clusters found in the cluster analysis done previously [27]. The two clusters found in that paper contain patients with MS in both clusters. The two clusters distinguished patients with inflammation and patients without inflammation as was shown by another research [20]. As previously mentioned, patients without inflammation were in cluster 1 and patients with inflammation were in cluster 2. The second objective was to find proteins that separate patients with MS and patients without MS.

RFECV was done using three algorithms for each objective. The three algorithms were logistic regression with liblinear as solver, logistic regression with lbfgs as solver, and linear support vector classifier. In total 6 models were trained for protein selection. Three models for the first objective and another three models for the second objective. ´

After the proteins were selected using RFECV, a model was trained on the training data and evaluated on the test data. The same machine learning algorithms that were used for protein selection (logistic regression with liblinear solver, logistic regression with lbfgs solver, and a linear support vector classifier), were also used to train on the dataset using the selected proteins.

After RFECV, a PCA analysis was performed using the selected features. The first and second principal components were plotted to see if the PCA plot showed distinct groups. Two biplots of the first and second principal components were done using the features that showed the best separation. The first biplot was done on the features that best separated patients into cluster 1 and cluster 2 (the first objective). The second biplot was done on the features separating patients with MS and patients without MS (the second objective).

Uniprot [32] was used to convert the protein codes used in the dataset into their original protein names.

Chapter 4

Results

The results section is divided into three sections. The first section presents the results when the objective was to find proteins that separate patients into cluster 1 and cluster 2. The second section presents the results obtained when the objective was to distinguish patients with MS and patients without MS. The third section shows the two biplots, one for each objective.

4.1 Feature Selection To Maximize Cluster separation

In this section RFECV was done to find proteins that best separate patients into cluster 1 and cluster 2. The patients' MS status was not taken into account.

4.1.1 Logistic Regression with lbfgs as solver

Recursive feature elimination using cross validation with a logistic regression model using lbfgs as solver to separate patients into cluster 1 and cluster 2 gave optimum results when 62 proteins were selected. Adding more proteins only made the model less accurate as shown in figure 4.1. This model is referred to as model 1 in this section.

The first and second component plot of the PCA performed using 62 extracted proteins is shown in figure 4.2. There is a clear separation between patients in cluster 1 and patients in cluster 2. Using the 62 extracted protiens, the PCA plot showed two distinct groups.

The training and validation accuracy of model 1 as a function of parameter C were plotted in figure 4.3. Training and validation accuracy started at around 70% when the C-value was at its lowest (0.001). The validation accuracy was highest at 90% when the the C-value was equal to 1. At this C value, the training accuracy was

100%. Further increase in the value of C gave no effect on the validation and training accuracy.

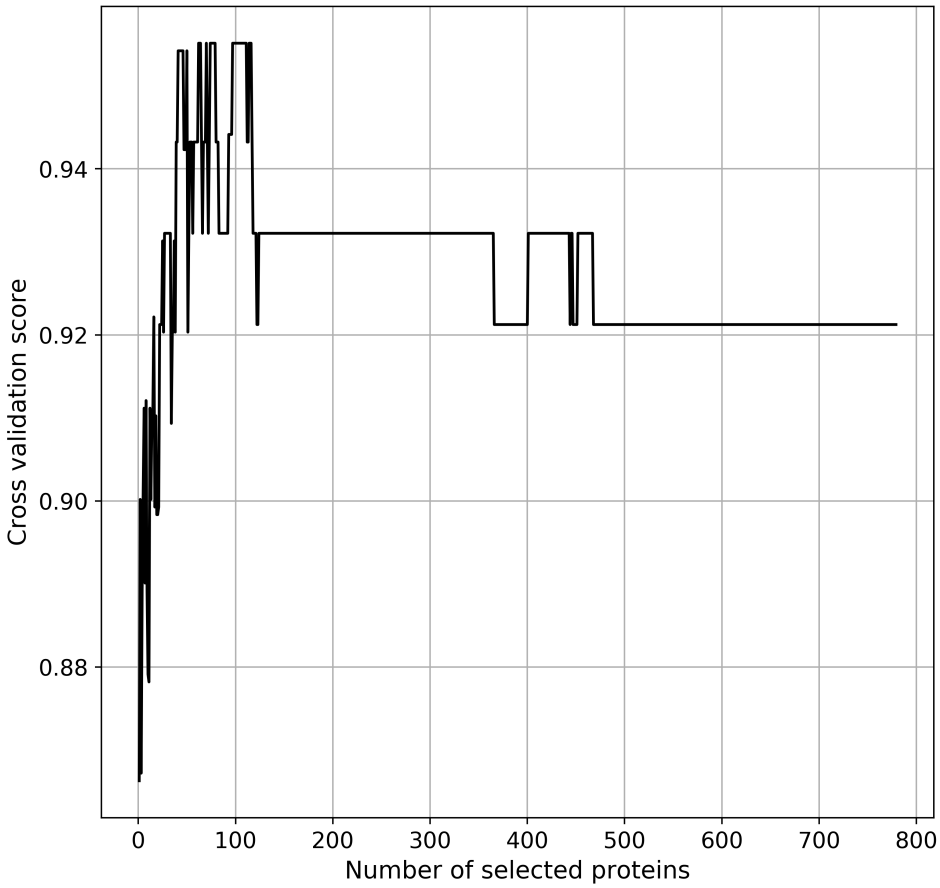

Figure 4.1: *RFECV using a Logistic Regression model with lbfgs to separate cluster 1 and cluster 2*

4.1.2 Logistic Regression with liblinear as solver

RFECV using a logistic regression model having liblinear as a solver gave completely different results compared to model 1. The optimum number of proteins to separate patients into cluster 1 and cluster 2 was 35 proteins. Adding more proteins either made the model less accurate or have the same performance. This is shown in figure 4.4. This model is referred to as model 2.

The first and second component plot of the PCA performed using 35 extracted proteins is shown in figure 4.5. Cluster separation is maintained. There is a clear separation between patients in cluster 1 and patients in cluster 2. The PCA plot clearly shows two distinct groups.

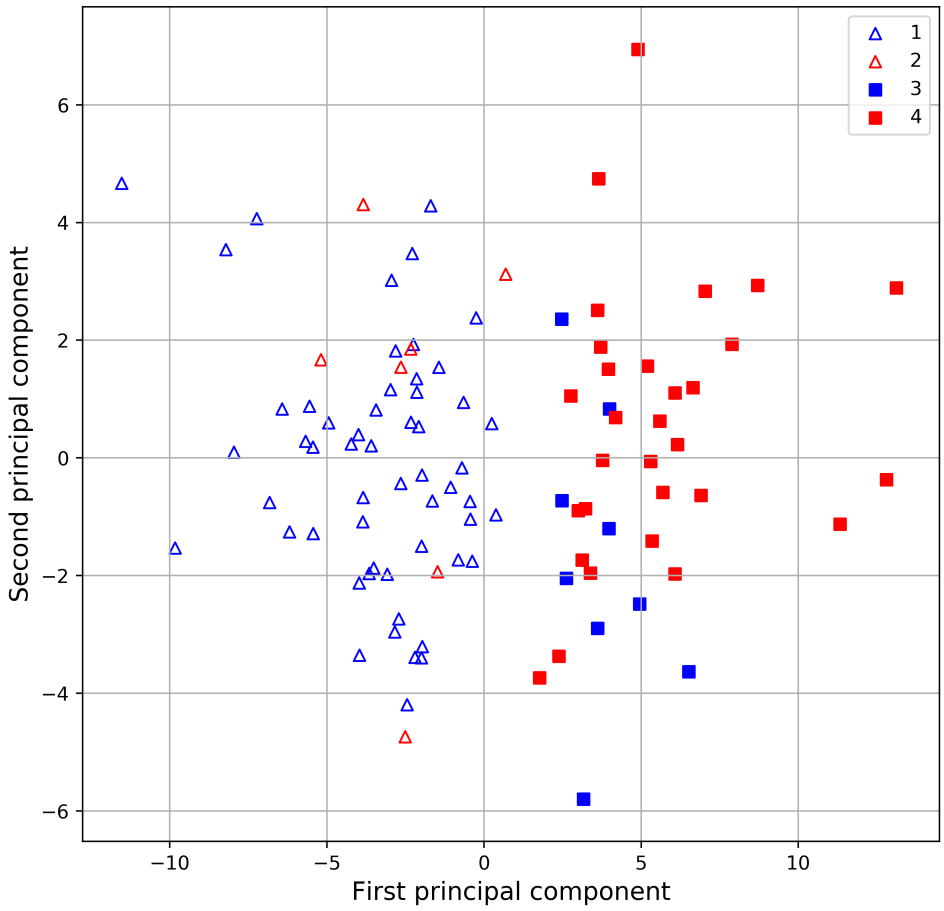

Figure 4.2: *First and second component PCA plot of the selected proteins using RFECV on logistic regression with lbfgs to separate cluster 1 and cluster 2*

The training and validation accuracy of model 2 as a function of parameter C were plotted in figure 4.6. Training and validation accuracy started at around 85% when the C-value was at its lowest (0.001). The validation accuracy was highest at 87% when the the C-value was equal to 0.1. At this C value, the training accuracy was 92%. Further increase in the value of C gave an increase in the training accuracy and a general decrease in the validation accuracy.

4.1.3 Linear Support Vector

RFECV using a linear support vector gave the fewest number of selected proteins. The optimum number of proteins to separate patients into cluster 1 and cluster 2 was 8 proteins. Adding more proteins made the model less accurate. The graph of RFECV using a linear support vector is shown in figure 4.7. This model is referred

to as model 3.

The first and second component plot of the PCA performed using the 8 extracted proteins is shown in figure 4.8. Even though there are only 8 proteins, cluster separation is still maintained. There is a clear separation between patients in cluster 1 and patients in cluster 2.

The training and validation accuracy of model 3 as a function of parameter C were plotted in figure 4.9. Training and validation accuracy was around 63% for the first three C-values. The validation accuracy was highest at 81% when the the C-value was equal to 1. At this C value, the training accuracy was 92%. Further increase in the value of C gave an increase in the training accuracy and a decrease in the validation accuracy.

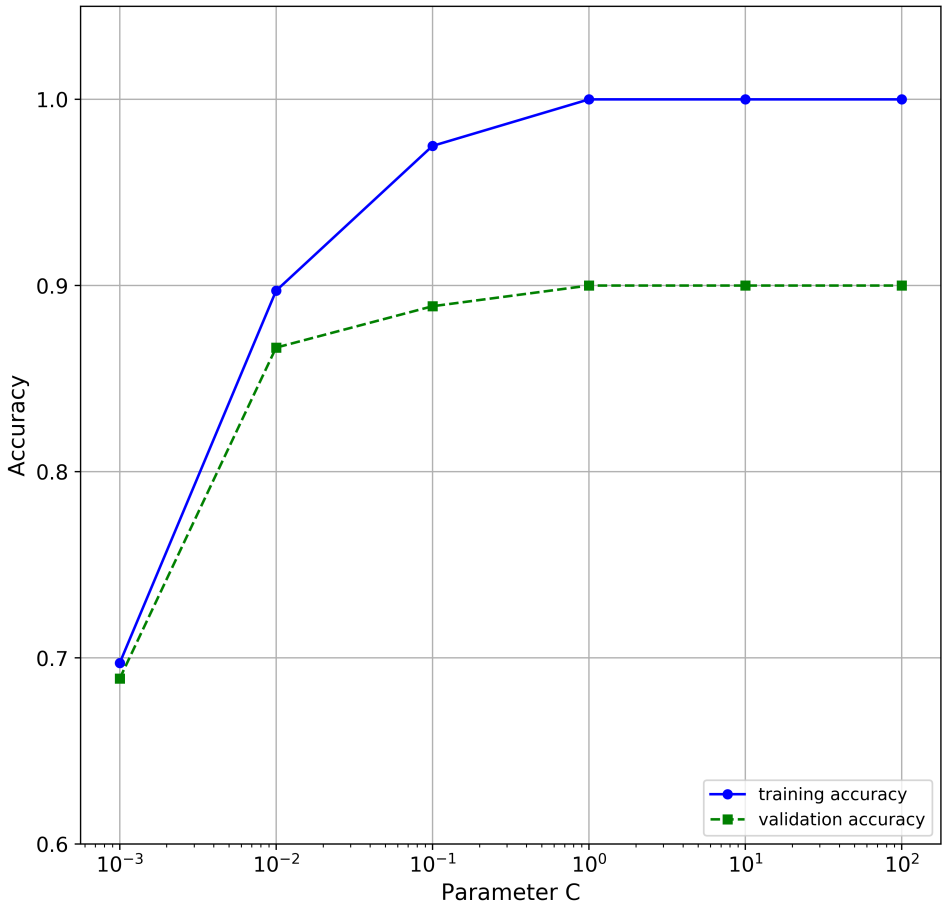

Figure 4.3: *Validation curve of a Logistic Regression model with lbfgs to separate cluster 1 and cluster 2. Class 1 (blue triangels) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

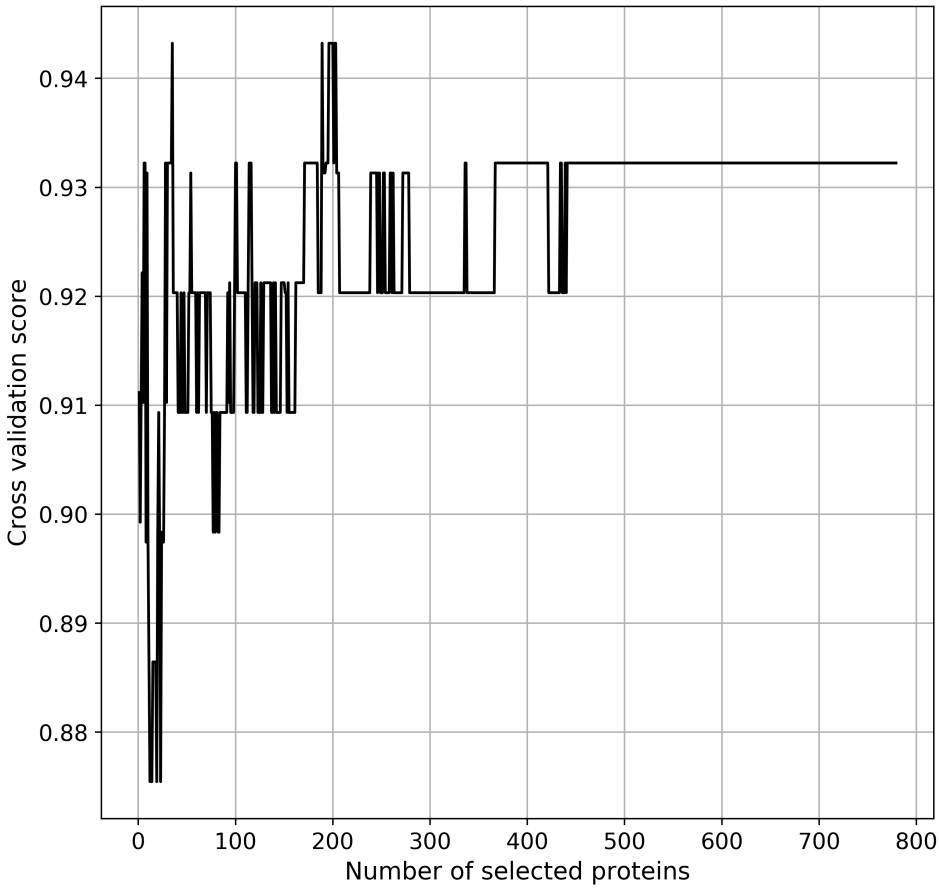

Figure 4.4: *RFECV using a Logistic Regression model with liblinear to separate cluster 1 and cluster 2*

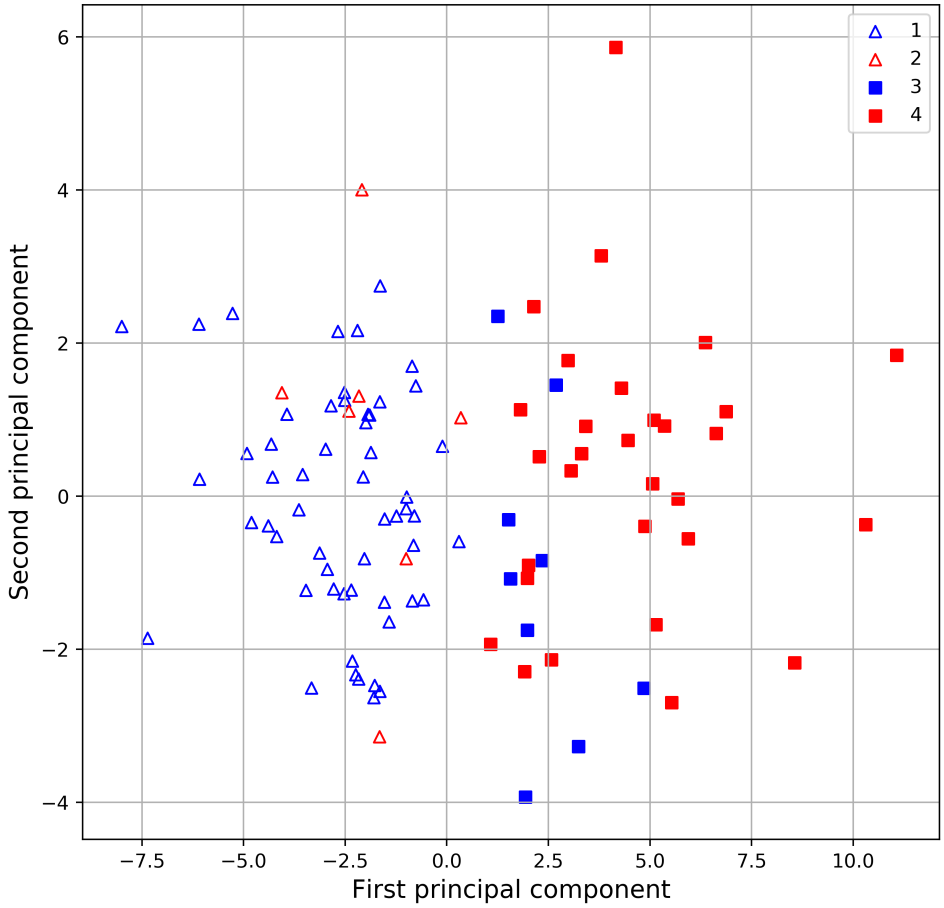

Figure 4.5: *First and second component PCA plot of the selected proteins using RFECV on logistic regression with liblinear to separate cluster 1 and cluster 2. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

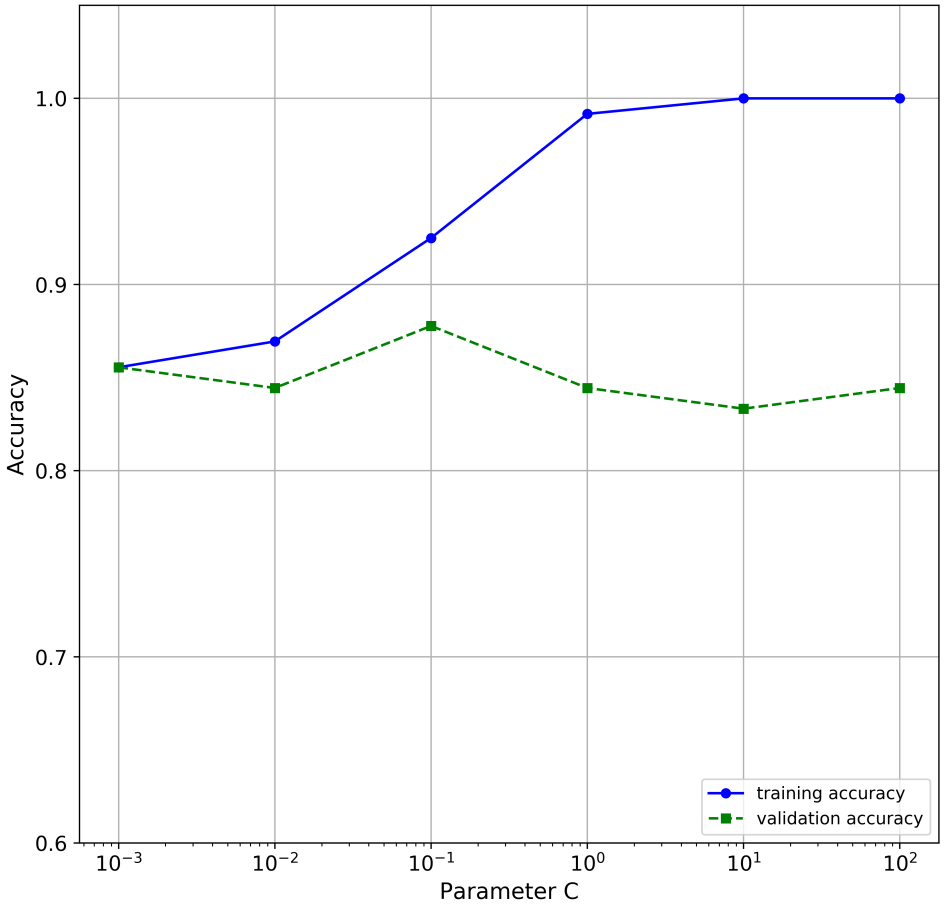

Figure 4.6: *Validation curve of a Logistic Regression model with liblinear to separate cluster 1 and cluster 2*

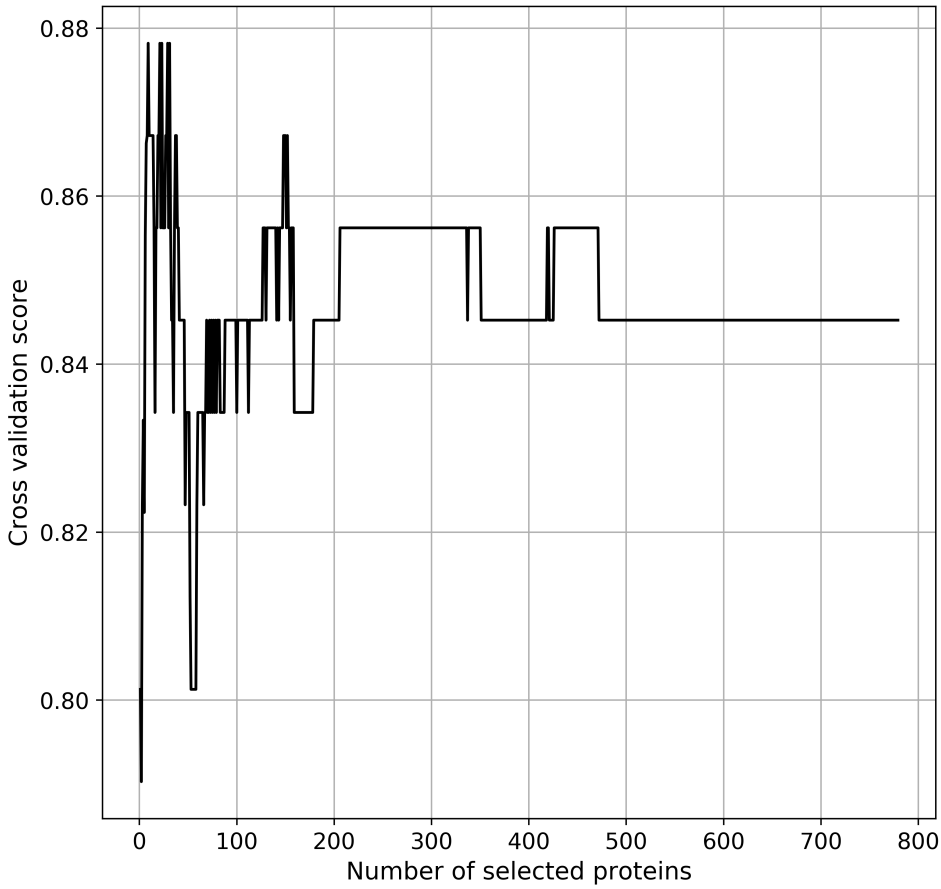

Figure 4.7: *RFECV using a Linear Support vector model to separate cluster 1 and cluster 2*

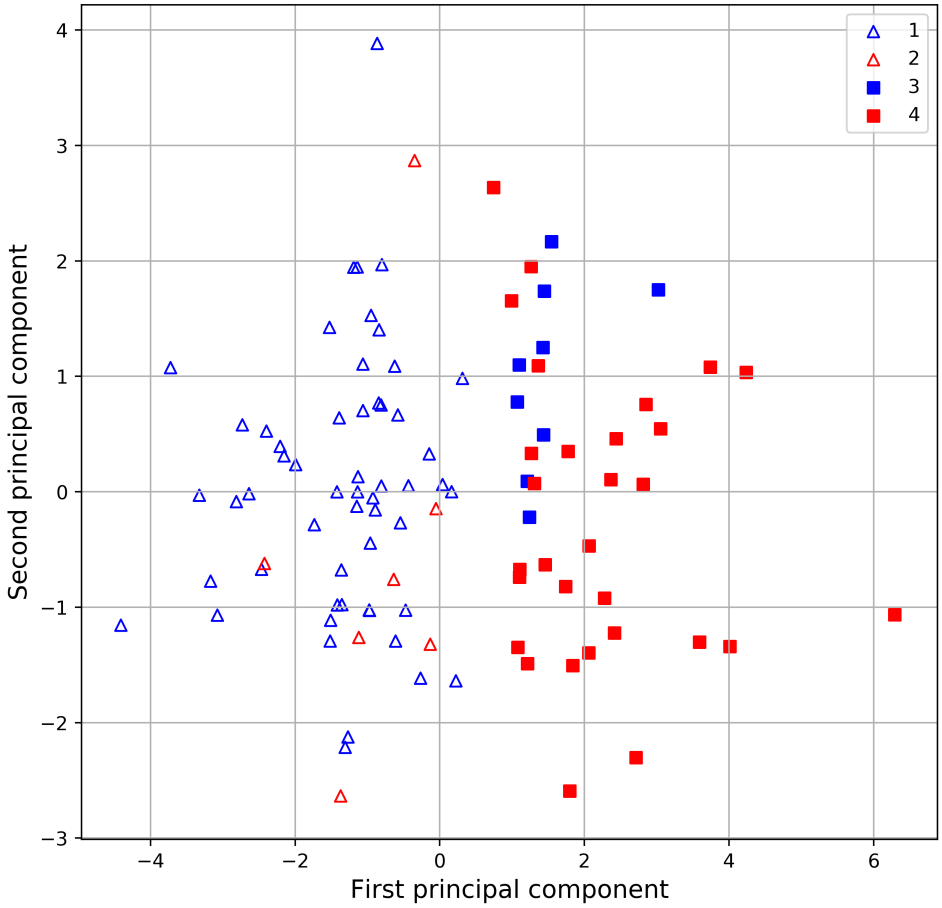

Figure 4.8: *First and second component PCA plot of the selected proteins using RFECV on a Linear Support Vector to separate cluster 1 and cluster 2. Class 1 (blue triangle) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

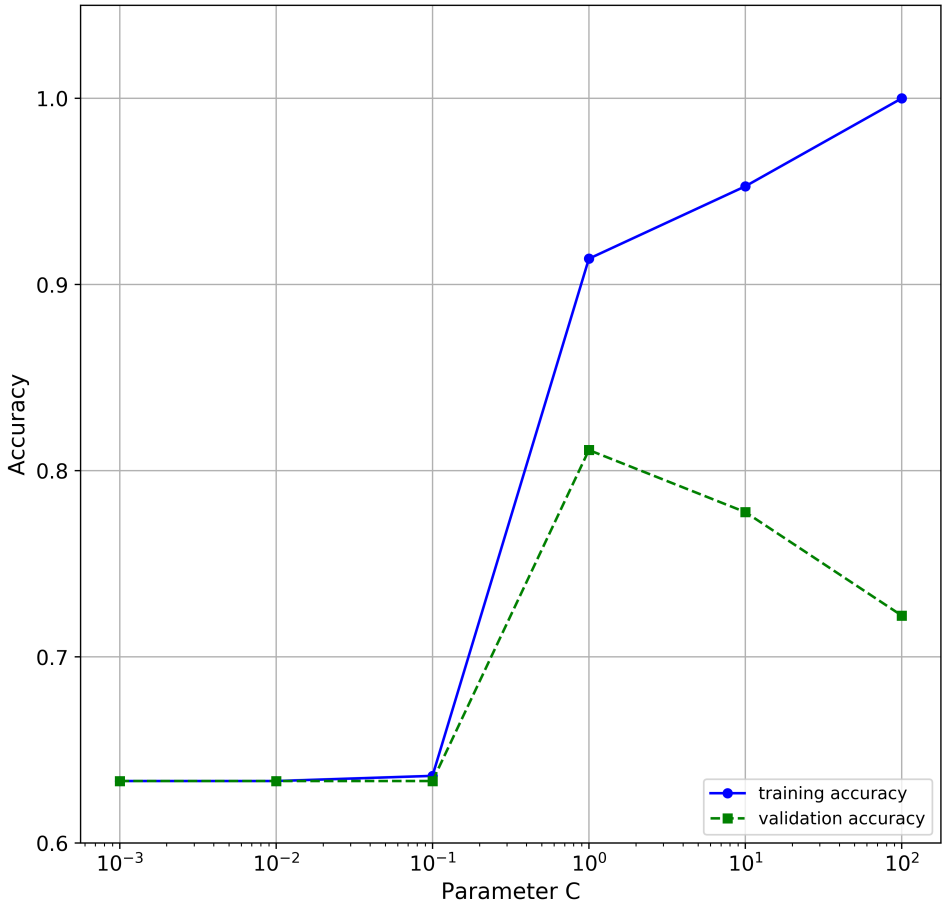

Figure 4.9: *Validation curve of a Support Vector model to separate cluster 1 and cluster 2*

4.2 Feature Selection To Separate MS and Non-MS

In this section the results of RFECV to find proteins that help in distinguishing patients with MS and patients without MS is displayed.

4.2.1 Logistic Regression with lbfgs as solver

RFECV plot using a logistic regression model having lbfgs as a solver to separate patients with MS and patients without MS is shown in figure 4.10. Based on this model, RFECV found 14 proteins that gave the highest cross validation score.

The first and second principal component plot of the PCA performed using the 14 selected proteins is shown in figure 4.11. The PCA plot shows good separation between patients with MS and patients without MS. MS separation using PCA plot is not as good as the separation seen in the PCA plot to separate cluster 1 and cluster 2. This suggests it is easier to separate cluster 1 and cluster than separating patients with MS.

The training and validation accuracy of this model as a function of the regularization parameter C were plotted in figure 4.12. Training and validation accuracy started at 63% when the C-value was at its lowest (0.001). The validation accuracy was highest at 97% when the the C-value was equal to 1. At this C value, the training accuracy was 100%. Further increase in the value of C produced no change in the training accuracy and the validation accuracy.

4.2.2 Logistic Regression with liblinear as solver

RFECV plot using a logistic regression model having liblinear as a solver to separate patients with MS and patients without MS is shown in figure 4.13. Based on this model, RFECV found 15 proteins that gave the highest cross validation score.

The first and second principal component plot of the PCA performed using the 15 extracted proteins is shown in figure 4.14. The PCA plot shows good separation between patients with MS and patients without MS. MS separation using PCA plot is not as good as the separation seen in the PCA plot to separate cluster 1 and cluster 2.

The training and validation accuracy of this model as a function of parameter C were plotted in figure 4.15. Training and validation accuracy started at 92% when the C-value was at its lowest (0.001). The validation accuracy was highest at 100% when the the C-value was equal to 0.1. At this C value, the training accuracy was 100%. Further increase in the value of C produced no change in the training accuracy and a decrease in the validation accuracy.

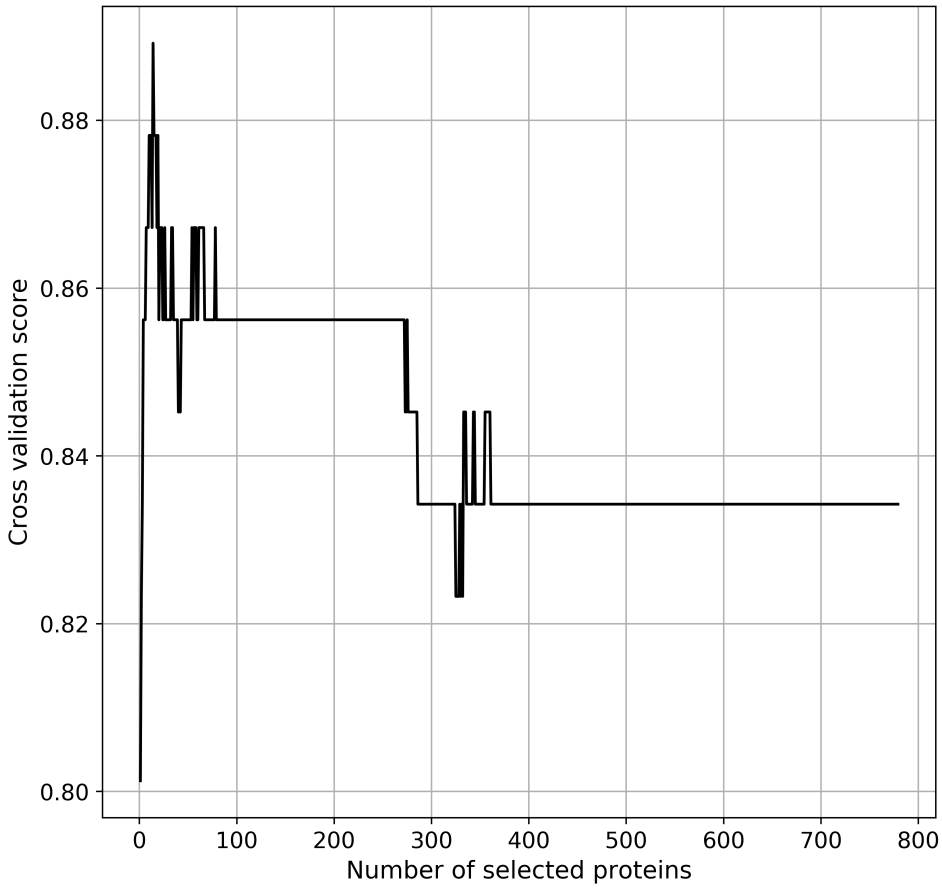

Figure 4.10: *RFECV using a Logistic Regression model with lbfgs to separate patients with MS and patients without MS*

4.2.3 Linear Support Vector Classifier

RFECV plot using a linear support vector model to separate patients with MS and patients without MS is shown in figure 4.16. Based on this model, RFECV found 9 proteins that gave the highest cross validation score.

The first and second principal component plot of the PCA performed using the 9 extracted proteins is shown in figure 4.17. The PCA plot shows good separation between patients with MS and patients without MS. One MS patient is clearly placed with Non-MS patients. The same patient is placed the same way in the PCA plot of figure 4.14 and in figure 4.11.

The training and validation accuracy of this model as a function of parameter C were plotted in figure 4.18. Training and validation accuracy started at 63% when the C-value was at its lowest (0.001). The validation accuracy was highest at 97%

when the the C-value was equal to 1. At this C value, the training accuracy was 100%. Further increase in the C-value decreased the validation accuracy whereas the training accuracy stayed at 100%.

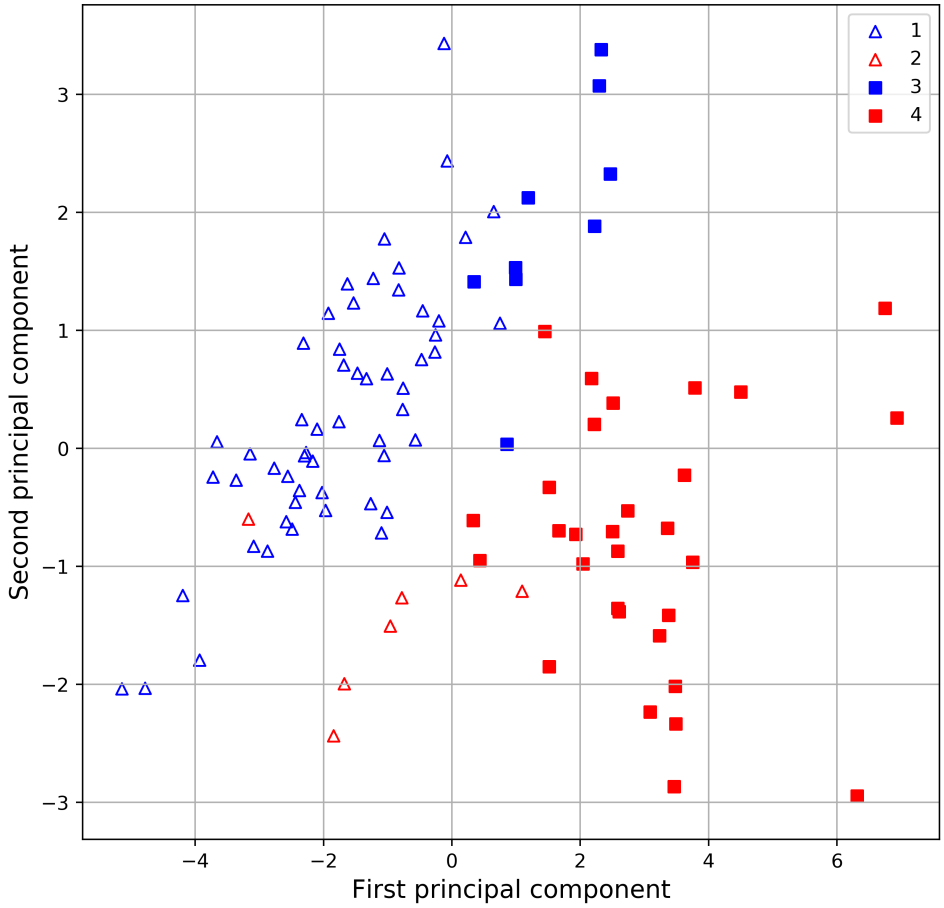

Figure 4.11: *First and second component PCA plot of selected proteins to separate patients with MS and patients without MS. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

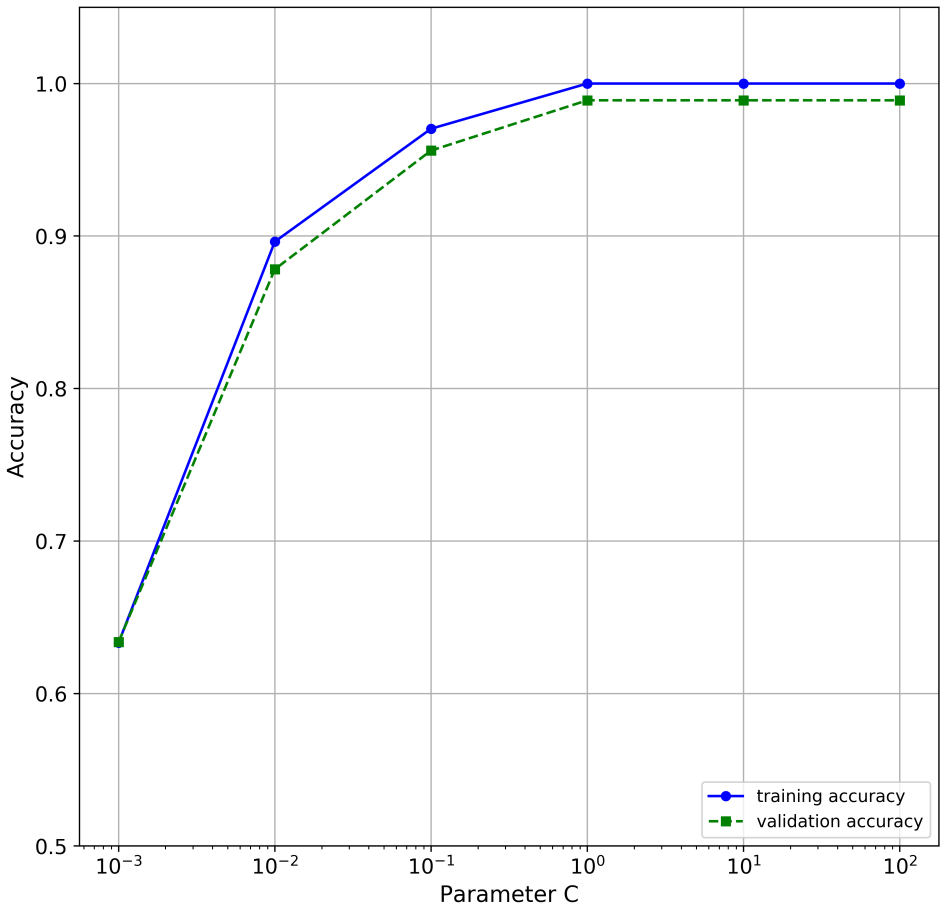

Figure 4.12: *Logistic Regression with lbfgs to separate patients with MS and patients without MS. The training accuracy and validation accuracy is shown as a function of parameter C*

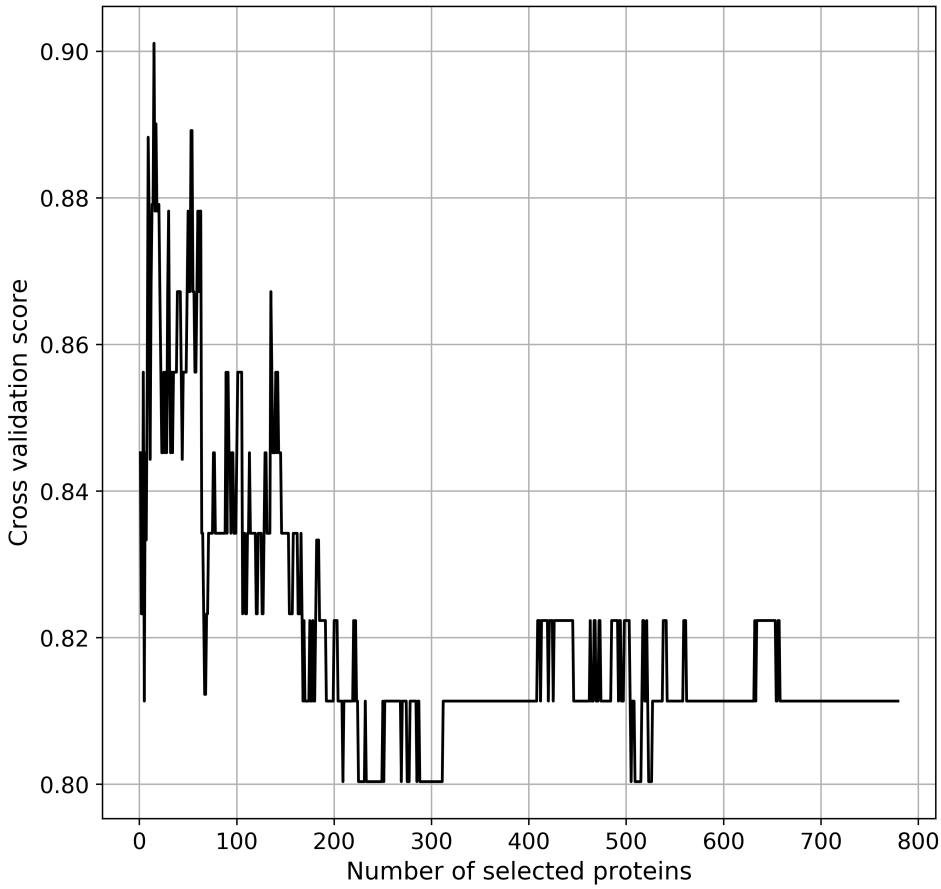

Figure 4.13: *RFECV using a Logistic Regression model with liblinear to separate patients with MS and patients without MS*

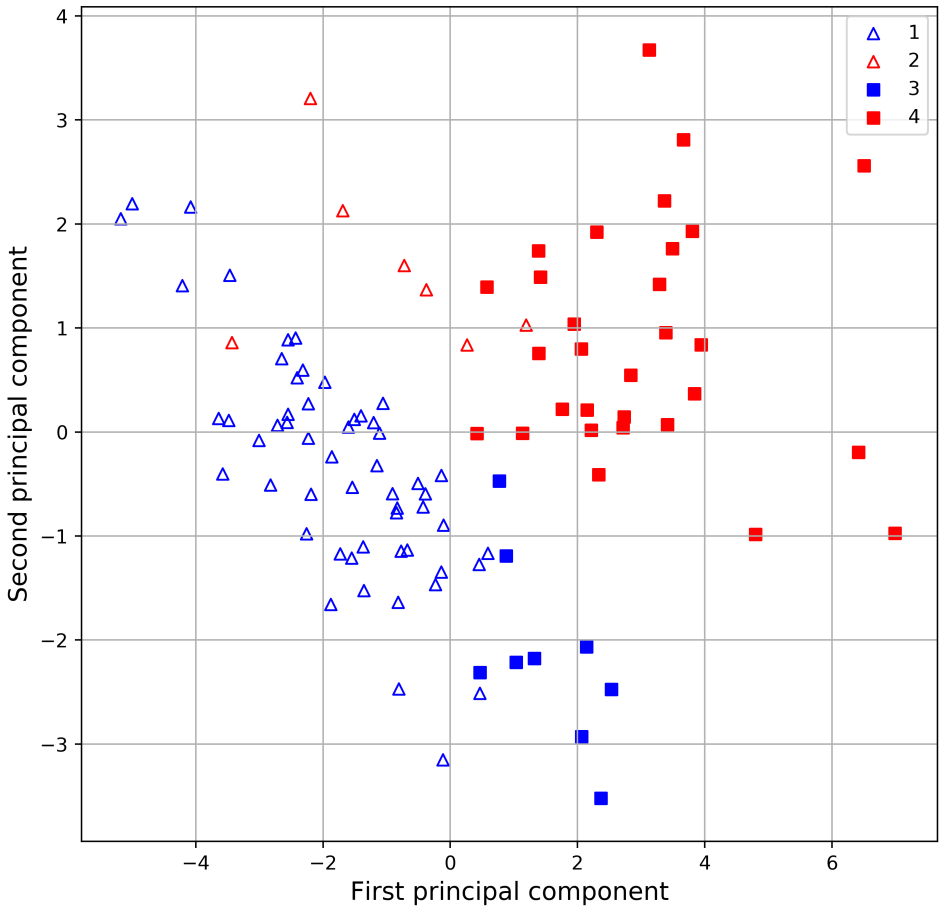

Figure 4.14: *First and second component PCA plot of proteins separating patients with MS and patients without MS. Proteins were found using RFECV on logistic regression with liblinear as solver. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

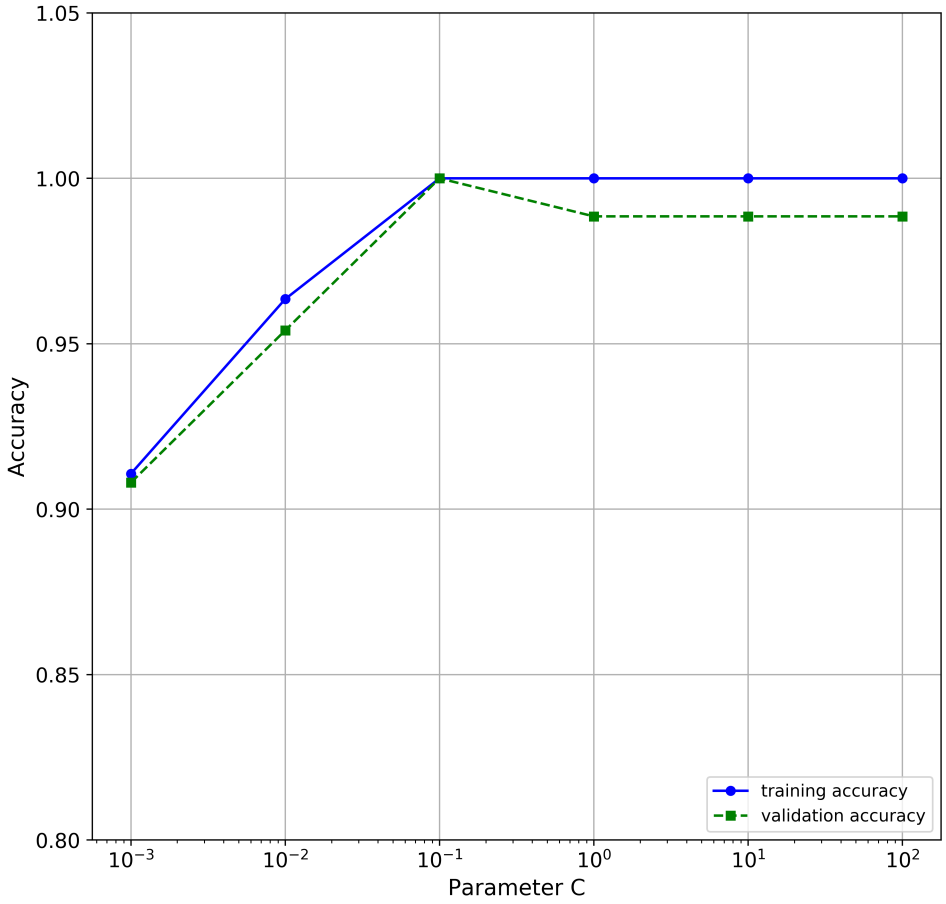

Figure 4.15: *Graph of validation accuracy and training accuracy of a Logistic Regression model with liblinear as solver as a function of parameter C*

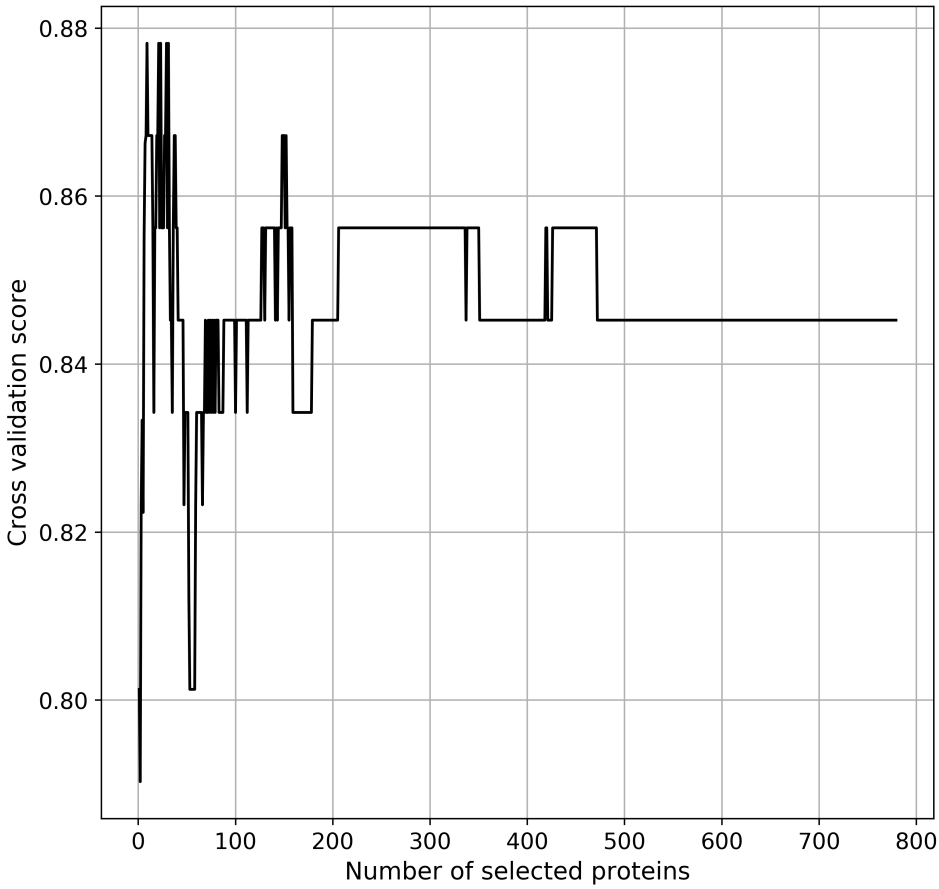

Figure 4.16: *RFECV using a Linear Support Vector model to separate patients with MS and patients without MS*

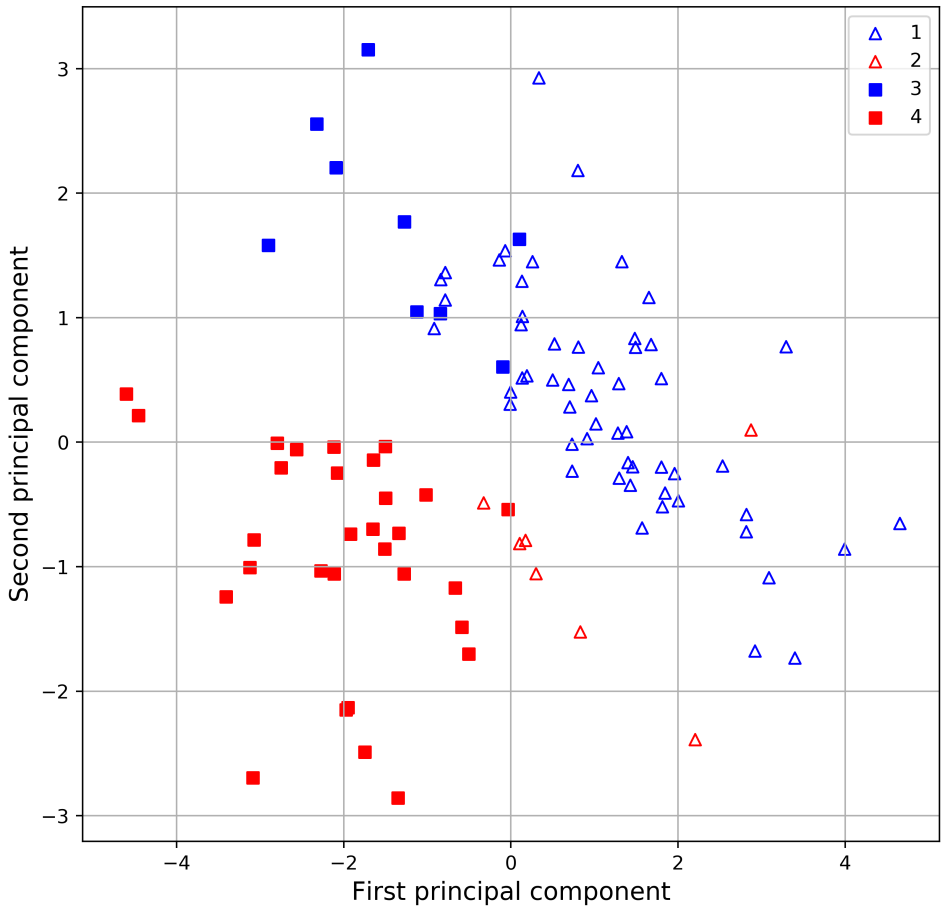

Figure 4.17: *First and second component PCA plot of the selected proteins using RFECV on a Linear Support Vector Classifier. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

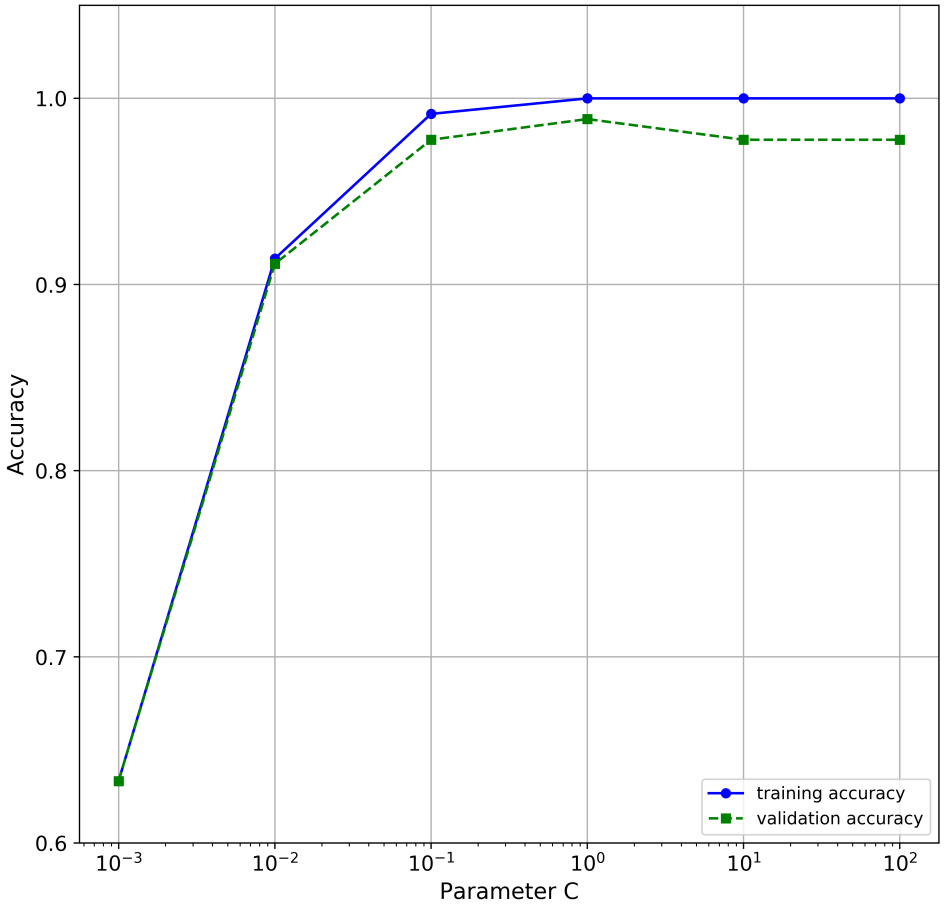

Figure 4.18: *Graph of validation accuracy and training accuracy of a linear support vector model as a function of the regularization parameter C*

4.3 Biplot using selected proteins

A biplot using the 14 proteins that were selected by a logistic regression model with liblinear as solver is shown in figure 4.19. The scores and loadings are shown.

A biplot using the 8 proteins that were selected by a linear support vector model is shown in figure 4.20. The scores and loadings are shown.

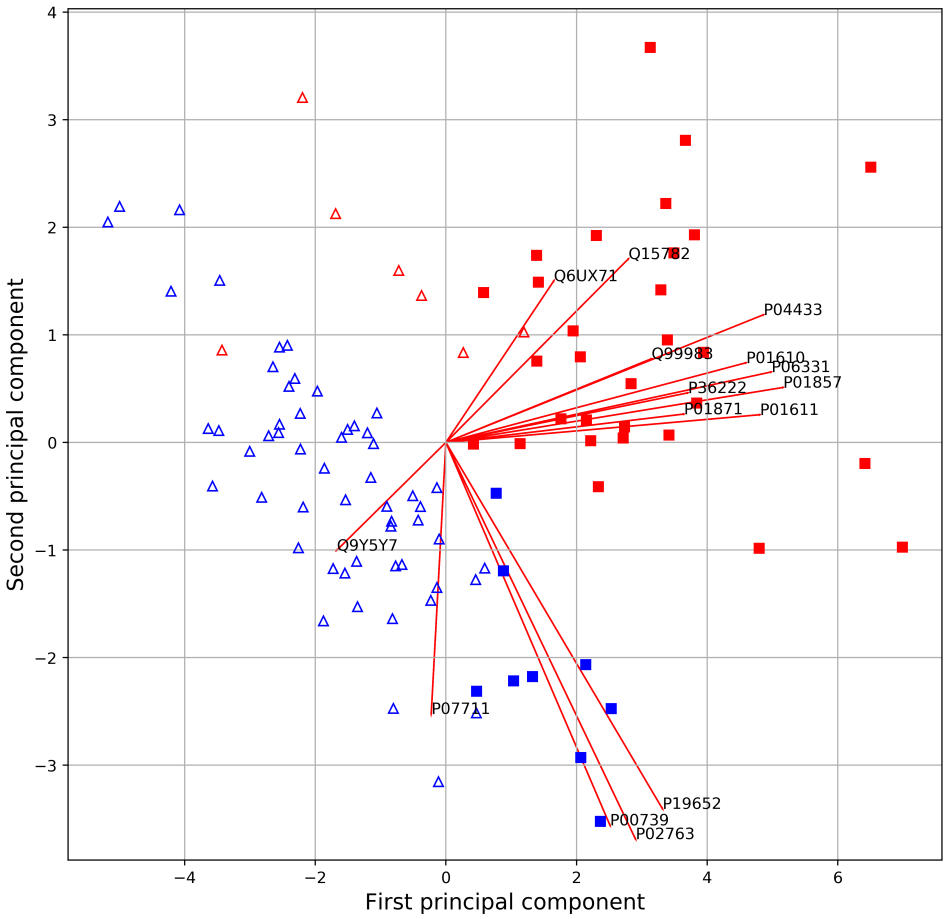

Figure 4.19: *PCA biplot using the proteins selected based on a logistic regression model with liblinear as solver to separate patients with MS and patients without MS. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

There were many immunoglobulins that were selected when training a model to distinguish between patients with MS and patients without MS. The following is

the list of immunoglobulins and their protein codes that were selected. They can be seen in the biplot of figure 4.19.

- P01611 - Immunoglobulin kappa variable 1D-12
- P01871 - Immunoglobulin heavy constant mu
- P01857 - Immunoglobulin heavy constant gamma 1
- P06331 - Immunoglobulin heavy variable 4-34
- P01610 - Immunoglobulin kappa variable 1-17
- P04433 - Immunoglobulin kappa variable 3-11

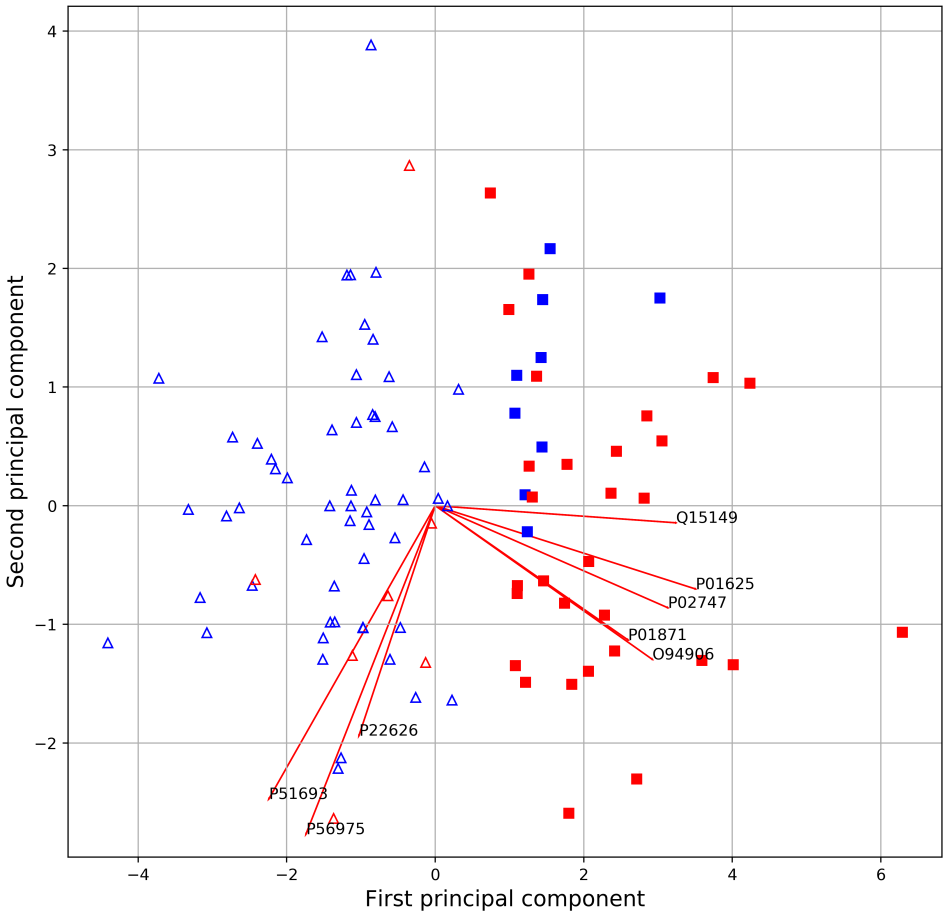

Figure 4.20: *PCA biplot using the proteins selected based on a Linear Support vector model to separate patients in cluster 1 and cluster 2. Class 1 (blue triangles) are non-MS patients belonging to cluster 1. Class 2 (red triangles) are MS patients in cluster 1. Class 3 (blue squares) are non-MS patients in cluster 2. Class 4 (red squared) are MS patients belonging to cluster 2.*

Chapter 5

Discussion

5.1 Models

The models built to classify MS and inflammation were relatively successfull. The models classifying patients with inflammation and patients without inflammation had a lower classification error than the models classifying MS. The models trained to recognize MS from protein markers seem promising. The predictive models based on protein markers will make MS diagnosis simpler and quicker than traditional methods. To deploy these methods however, further research is needed. Bigger data sets with many more patients will help asses the models to a much higher precision.

5.2 Recognizing Inflammation

In this dataset using the methods described above, it is easier to distinguish patients with inflammation than patients with MS in the two component PCA plot. This is because the patients that belong to the respective clusters were grouped after a previous unsupervised multivariate analysis was done as mentioned previously. In that analysis it was observed two clusters of which the patients received their cluster labels after seeing which cluster they were closer to. In essence, the method described in this paper classifies patients belonging to cluster 1 and cluster 2 after they were already classified by a different mechanism previously. However, the protein sets that are used to distinguish between these two clusters were mainly immunoglobulins which are proteins associated with inflammation. This reinforces the conclusions made in previous research using this dataset [20].

5.2.1 Proteins Selected for cluster separation

When separating patients into cluster 1 and cluster 2 (no inflammation and inflammation) several types of proteins were selected. Two of the proteins separating cluster 1 and cluster 2 were immunoglobulins. Their protein code and corresponding names are P01871 (Immunoglobulin heavy constant antibody Ab) and P01625 (Immunoglobulin kappa variable 4-1).

Immunoglobulins are known as antibodies. An antibody's task is to fight of pathogens such as bacteria and viruses[33] [34]. Different immunoglobulins specialize in combating different pathogens [32]. Therefore high values of immunoglobulins in patients can be interpreted as a sign of inflammation. Looking at the PCA cluster biplot in figure 4.2, it can be deduced that the first principal component is the most important component in distinguishing between cluster 1 and cluster 2. Biologically, it makes sense that immunoglobulins were selected to distinguish between patients with inflammation and patients without inflammation.

Three proteins that were picked up to distinguish between cluster 1 and cluster 2 were associated with neuron development and regulation. These proteins are P51693 (amyloid-like protein 1), P56975 (Pro-neuregulin-3), and P22626 (heterogeneous nuclear ribonucleoproteins A2/B1).

P51693 is called amyloid-like protein 1. This protein is responsible for neurite outgrowth. Neurite outgrowth happens in developing neurons where new projections (axons or dendrites) occur as a result of outside guidance [32]. Amyloid-like protein is responsible for nervous system development [35]. Although information on Amyloid precursor protiens in patients with MS is scarce, research states that it is upregulated in damaged axons, which suggests that it may constitute a reliable marker of axon demyelination [36].

P56975 (Pro-neuregulin-3), plays an important role in development, maintenance and repair of the nervous system and other organs [37]. Pro-neuregulin-3 is responsible for the negative regulation of neuron migration [32]. Neuronal migration is an essential phenomena in central nervous system development. For functioning neuronal circuits to develop, neuronal migration has to happen in the right structures. A flaw in this process may result in a neurological disorder [38].

P22626 (Heterogeneous nuclear ribonucleoproteins A2/B1 (HnRNP)) contributes to multiple essential roles in neuronal functioning and its depletion [32]. A decline in HnRNP A1 is correlated with symptoms in several neuro-degenerative diseases among which is multiple sclerosis [39].

These three proteins are grouped together in cluster 1 in the PCA biplot. The grouping can be interpreted as the proteins sharing similar characteristics to each other. This is indeed true when looking at how all three of them are associated with neuron development. In addition, these proteins are almost perpendicular to the other group of proteins containing immunoglobulins in cluster 2. This shows

that these two groupings of proteins are not correlated and do not show similar characteristics.

There are patients with MS present in both clusters, but there are only a few patients with MS in cluster 1. These are patients with MS but no signs of inflammation. A question might be raised as to why neuron related proteins are used to distinguish between cluster 1 and cluster 2 when the primary goal of cluster 1 and cluster 2 classification was to distinguish patients with inflammation. Due to the majority of patients being healthy in cluster 1, the PCA plot also uses proteins associated with developing the central nervous system to distinguish cluster 1 from cluster 2. If more patients with MS but no signs of inflammation where present in cluster 1, this may not have been the case.

5.3 Recognizing Multiple Sclerosis

When the objective was to group patients into patients with inflammation and patients without inflammation, no MS separation was observed. However, when the objective was to separate patients with MS and patients without MS, patients with inflammation were grouped separately. This suggests that MS and inflammation are two unconnected biological phenomena, and to find patients with MS, a two stage classifier is required. The first stage is separating patients with inflammation and patients without inflammation. The second stage consists of separating patients with MS and patients without MS.

5.3.1 Proteins selected

When training the model to distinguish patients with MS and patients without MS, the two component PCA plot of the selected proteins showed four different groups. Patients in the dataset had a combination of MS and inflammation conditions (MS + inflammation, MS + no inflammation, no MS + inflammation, no MS + no inflammation) and these groups were distinguished in the PCA plot.

Multiple immunoglobulins were selected in order to distinguish between patients with MS and patients without MS. Immunoglobulins are associated with the immune response, regulation of immune response, adaptive immune response, and inflammation. Proteins that are not immunoglobulins but still have something to do with the immune response and inflammation were also picked up. These include P07711 (Cathepsin L1) and P00739 (Haptoglobin-related protein).

Cathepsin L1 is responsible for the adaptive immune response [32]. When a specific antibody is made for an antigen and the body is hit a second time with the same antigen, an enhanced secondary response allows the body to fight the antigen off quicker. Also known as immunological memory [40].

Haptoglobin is a protein that is responsible for Inflammation which starts as short-

lived response to injury or an antigen. The inflammatory response begins within minutes. It can either settle within a few days or develops into a chronic inflammatory response[40]. The protein is also associated with positive regulation of cell death [40]. Cell death is regulated through activation or stoppage of vital processes within a cell causing cell death. Haptoglobin-related proteins are markers of Neuromyelitis optica (NMO), which is a neurodegenerative disease [41]. It is interesting to note that Haptoglobing was selected as a protein used to distinguish MS, while being a marker for a neurodegenerative disease.

Another interesting protein that was selected is Q9Y5Y7, Lymphatic vessel endothelial hyaluronic acid receptor 1 (LYVE-1). The functional role of LYVE-1 is not clear, several of the common neuro-lymphatic proteins are essential for brain development and neuronal function. LYVE-1 is suggested as potential physiological and pathological importance in MS [42].

Q99983 (Osteomodulin) is another protein selected to separate patients with and without MS. Osteomodulin (OMD), plays a crucial role in regulating skeletal development [43]. Increasing evidence shows MS is correlated with risks of fractures. Most MS patients are Physically inactive, a reduction on the mechanical load on the bones is likely the major contributing factor for reduced bone density in patients with MS [44]. Hence, osteomodulin becomes an indirect way of distinguishing MS.

5.4 MS hypothesis

The proteins selected to distinguish between patients with MS and patients wihout MS support the hypothesis which states MS as a defect in neuron regeneration and not an inflammatory diseases [20]. Most patients later on develop inflammation but some patients do not. The two component PCA plot showing 4 groups is interpreted in the light of this hypothesis in two categories. The first category is MS and second is inflammation. The four groups are a combination of these two categories. Furthermore, there are two types of proteins that distinguish the 4 groups. Proteins that are responsible for inflammation and protiens that are responsible for neuron developement and degeneration.

Some weak links to this hypothesis based on the findings may be that the findings are sample dependent. It should be noted that the proteins selected are dependent on the patients in the dataset. A different training set may produce different set of selected proteins. However it should also be noted that the proteins that were chosen were very relevant to MS and inflammation. They were mainly immunoglobulins and neuron related proteins. A much larger number of patients would be very helpful in verifying or refuting the findings in this paper.

Chapter 6

Conclusions

Separating patients with MS and patients without MS was successfull using RFECV. Small protein sets were found from a total of 779 protiens. The protein sets show clear separation in the two component PCA plot between patients with MS and patients without MS. In addition to separation between patients with MS and patients without MS, the two component PCA plot also distinguished between patients that have inflammation and patients that do not have inflammation even though no information was given as to which patients have inflammation. Distinguishing patients with MS resulted in also separating patients with inflammation. The findings challenge the current understanding of MS which describes MS as a combination between inflammation and neuro-degeneracy. The findings also strengthen the hypothesis which describe MS as a result of a defect in neuron regeneration, where inflammation is not a necessity in order for MS to occur [20].

Bibliography

[1] C. Lucchinetti, W. Brück, J. Parisi, B. Scheithauer, M. Rodriguez, and H. Lassmann, "Heterogeneity of multiple sclerosis lesions: Implications for the pathogenesis of demyelination," *Annals of Neurology*, vol. 47, no. 6, pp. 707–717, 2000.

[2] L. Steinman, "Multiple sclerosis: a two-stage disease," *Nature Immunology*, vol. 2, no. 9, p. 762–764, 2001.

[3] C. C. Whitacre, "Sex differences in autoimmune disease," *Nature Immunology*, vol. 2, no. 9, p. 777–780, 2001.

[4] M. C. Stöppler, "5 early multiple sclerosis symptoms: Ms treatment life expectancy," Oct 2019. [Online]. Available: https://www.emedicinehealth.com/multiple_sclerosis/article_em.htm

[5] M. F. for Medical Education and Research. (2019) Multiple sclerosis. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/multiple-sclerosis/symptoms-causes/syc-20350269

[6] G. W. Lowis, "Ethnic factors in multiple sclerosis: A review and critique of the epidemiological literature," *International Journal of Epidemiology*, vol. 17, no. 1, p. 14–20, 1988.

[7] A. Langer-Gould, S. M. Brara, B. E. Beaber, and J. L. Zhang, "Incidence of multiple sclerosis in multiple racial and ethnic groups," *Neurology*, vol. 80, no. 19, pp. 1734–1739, 2013. [Online]. Available: https://n.neurology.org/content/80/19/1734

[8] B. D. Trapp and K.-A. Nave, "Multiple sclerosis: An immune or neurodegenerative disorder?" *Annual Review of Neuroscience*, vol. 31, no. 1, pp. 247–269, 2008. [Online]. Available: https://doi.org/10.1146/annurev.neuro.30.051606.094313

[9] P. K. Stys, G. W. Zamponi, J. V. Minnen, and J. J. G. Geurts, "Will the real multiple sclerosis please stand up?" *Nature Reviews Neuroscience*, vol. 13, no. 7, p. 507–514, 2012.

[10] S. L. Hauser and J. R. Oksenberg, "The neurobiology of multiple sclerosis: Genes, inflammation, and neurodegeneration," *Neuron*, vol. 52, no. 1, p. 61–76, Oct 2006.

[11] I. Kister, T. E. Bacon, E. Chamot, A. R. Salter, G. R. Cutter, J. T. Kalina, and J. Herbert, "Natural history of multiple sclerosis symptoms," *International Journal of MS Care*, vol. 15, no. 3, p. 146–156, 2013.

[12] H. Lassman, "Multiple sclerosis pathology," *Cold Spring Harbor Perspectives in Medicine*, vol. 8, 2018.

[13] S. Watson, "Autoimmune diseases: Types, symptoms, causes, diagnosis and more," Mar 2019. [Online]. Available: https://www.healthline.com/health/autoimmune-disorders

[14] T. J. Vyse and J. A. Todd, "Genetic analysis of autoimmune disease," *Cell*, vol. 85, no. 3, p. 311–318, May 1996.

[15] P. Marrack, J. Kappler, and B. L. Kotzin, "Autoimmune disease: why and where it occurs," *Nature Medicine*, vol. 7, no. 8, p. 899–905, Aug 2001.

[16] G. Disanto, A. J. Berlanga, A. E. Handel, A. E. Para, A. M. Burrell, A. Fries, L. Handunnetthi, G. C. D. Luca, and J. M. Morahan, "Heterogeneity in multiple sclerosis: Scratching the surface of a complex disease," *Autoimmune Diseases*, vol. 2011, p. 1–12, 2011.

[17] A. P. Corthals, "Multiple sclerosis is not a disease of the immune system," *The Quarterly Review of Biology*, vol. 86, no. 4, p. 287–321, 2011.

[18] A. Miller, M. Korem, R. Almog, and Y. Galboiz, "Vitamin b12, demyelination, remyelination and repair in multiple sclerosis," *Journal of the Neurological Sciences*, vol. 233, no. 1-2, p. 93–97, 2005.

[19] I. Tsunoda and R. S. Fujinami, "Inside-out versus outside-in models for virus induced demyelination: axonal damage triggering demyelination," *Springer Seminars in Immunopathology*, vol. 24, no. 2, p. 105–125, 2002.

[20] F. M. Ellen, A. V. Christian, H. L. Kristian, M. Anette, H. B. Gerd, K. Liesbeth, B. Frode, L. Artem, J. R. Christopher, E.-H. E. Karim, A. O. Jill, T. G. Bjørn, G. Sonia, and M. Kjell-Morten, "Analysis of megavariate data in functional omics," *Chemistry, Molecular Sciences and Chemical Engineering*, In print.

[21] K. B. Digre, "Mcdonald wi, compston a, edan g, et al. recommended diagnostic criteria for multiple sclerosis: Guidelines from the international panel on the diagnosis of multiple sclerosis." *Journal of Neuro-Ophthalmology*, vol. 22, no. 2, p. 143, Jul 2002.

[22] M. Comabella and X. Montalban, "Body fluid biomarkers in multiple sclerosis," *The Lancet Neurology*, vol. 13, no. 1, p. 113–126, 2014.

[23] C. Teunissen, T. Menge, A. Altintas, J. C. Álvarez Cermeño, A. Bertolotto, F. S. Berven, L. Brundin, M. Comabella, M. Degn, F. Deisenhammer, and et al., "Consensus definitions and application guidelines for control groups in cerebrospinal fluid biomarker studies in multiple sclerosis," *Multiple Sclerosis Journal*, vol. 19, no. 13, p. 1802–1809, 2013.

[24] S. Raschka, *Python Machine Learning*. Packt Publishing, December 2019.

[25] E. F. Rong, C. Kai-Wei, J. Cho, X.-R. Hsieh, and L. Wang, Chih-Jen, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research*, pp. 1871–1874, 2008.

[26] Y. T. Lee and A. Sidford, "Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems," *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 2013.

[27] J. A. Opsahl, M. Vaudel, A. Guldbrandsen, E. Aasebø, V. V. Pesch, D. Franciotta, K.-M. Myhr, H. Barsnes, M. Berle, Torkildsen, and et al., "Label-free analysis of human cerebrospinal fluid addressing various normalization strategies and revealing protein groups affected by multiple sclerosis," *Proteomics*, vol. 16, no. 7, p. 1154–1165, 2016.

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

[29] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, S. van der Walt and J. Millman, Eds., 2010, pp. 51 – 56.

[30] T. Oliphant, "NumPy: A guide to NumPy," USA: Trelgol Publishing, 2006–, [Online; accessed ¡today¿]. [Online]. Available: http://www.numpy.org/

[31] F. Pérez and B. E. Granger, "IPython: a system for interactive scientific computing," *Computing in Science and Engineering*, vol. 9, no. 3, pp. 21–29, May 2007. [Online]. Available: https://ipython.org

[32] T. U. Consortium, "UniProt: a worldwide hub of protein knowledge," *Nucleic Acids Research*, vol. 47, no. D1, pp. D506–D515, 11 2018. [Online]. Available: https://doi.org/10.1093/nar/gky1049

[33] W. H. Fridman, "Structure and function of immunoglobulins," *Cell-Mediated Effects of Immunoglobulins*, p. 1–28, 1997.

[34] "B cells, immunoglobulin genes, and immunoglobulin structure," *Immunology Guidebook*, p. 277–309, 2004.

[35] U. Lenkkeri, M. Kestilä, J. Lamerdin, P. Mccready, A. Adamson, A. Olsen, and K. Tryggvason, "Structure of the human amyloid-precursor-like protein gene aplp1 at 19q13.1," *Human Genetics*, vol. 102, no. 2, p. 192–196, 1998.

[36] S. Moore, A. J. Khalaj, R. Patel, J. Yoon, D. Ichwan, L. Hayardeny, and S. K. Tiwari-Woodruff, "Restoration of axon conduction and motor deficits by therapeutic treatment with glatiramer acetate," *Journal of Neuroscience Research*, vol. 92, no. 12, p. 1621–1636, Mar 2014.

[37] C. J. Carlos, L. Mingli, H. Adriana, N. Annette, O. John, C. H. Watson, B. P. K, S. Wesley, A. Precious, K. Sri, and et al., "Neuregulin in health and disease," *International Journal of Brain Disorders and Treatment*, vol. 4, no. 1, 2018.

[38] M. Rahimi-Balaei, H. Bergen, J. Kong, and H. Marzban, "Neuronal migration during development of the cerebellum," *Frontiers in Cellular Neuroscience*, vol. 12, 2018.

[39] U. Bekenstein and H. Soreq, "Heterogeneous nuclear ribonucleoprotein a1 in health and neurodegenerative disease: From structural insights to post-transcriptional regulatory roles," *Molecular and Cellular Neuroscience*, vol. 56, p. 436–446, 2013.

[40] P. Gaudet, M. S. Livstone, S. E. Lewis, and P. D. Thomas, "Phylogenetic-based propagation of functional annotations within the Gene Ontology consortium," *Briefings in Bioinformatics*, vol. 12, no. 5, pp. 449–462, 08 2011. [Online]. Available: https://doi.org/10.1093/bib/bbr042

[41] A. Sempere, L. Berenguer-Ruiz, M. Lezcano-Rodas, F. Mira-Berenguer, and M. Waez, "[lumbar puncture: its indications, contraindications, complications and technique]," *Revista de neurologia*, vol. 45, no. 7, p. 433—436, 2007. [Online]. Available: http://europepmc.org/abstract/MED/17918111

[42] G. V. Chaitanya, S. Omura, F. Sato, N. E. Martinez, A. Minagar, M. Ramanathan, B. W. Guttman, R. Zivadinov, I. Tsunoda, J. S. Alexander, and et al., "Inflammation induces neuro-lymphatic protein expression in multiple sclerosis brain neurovasculature," *Journal of Neuroinflammation*, vol. 10, no. 1, 2013.

[43] Y. Sommarin, M. Wendel, Z. Shen, U. Hellman, and D. Heinegård, "Osteoadherin, a cell-binding keratan sulfate proteoglycan in bone, belongs to the family of leucine-rich repeat proteins of the extracellular matrix," *Journal of Biological Chemistry*, vol. 273, no. 27, p. 16723–16729, Mar 1998.

[44] S. Gupta, I. Ahsan, N. Mahfooz, N. Abdelhamid, M. Ramanathan, and B. Weinstock-Guttman, "Osteoporosis and multiple sclerosis: Risk factors,

pathophysiology, and therapeutic interventions," *CNS Drugs*, vol. 28, no. 8, p. 731–742, 2014.

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway