CHALMERS
UNIVERSITY OF TECHNOLOGY

Machine learning for classifying the early stage of Osteoarthritis

Based on biological data.

Master's thesis in Physics and Biomedical Engineering

LINA ÅBERG

Department of Physics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2020

Master's thesis 2020

Machine learning for classifying the early stage of Osteoarthritis

Based on biological data.

LINA ÅBERG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Chalmers University of Technology
Gothenburg, Sweden 2020

Machine learning for classifying the early stage of Osteoarthritis
Based on biological data.
LINA ÅBERG

Supervisor and examiner: Magnus Karlsteen, Department of Physics, Chalmers
Advisor: Eva Skiöldebrand, Department of Biomedical Sciences and Veterinary Public Health, SLU

Master's Thesis 2020
Department of Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2020

Machine learning for classifying the early stage of Osteoarthritis
Based on biological data.
LINA ÅBERG
Department of Physics
Chalmers University of Technology

Abstract

Osteoarthritis, or OA, is a chronic joint disease and the most common form of arthritis. It is a very common disease in human athletes, but also the most common reason for lameness and poor performance in animal athletes, such as racehorses. The traditional standard for diagnosing OA is by radiographic measurements. Unfortunately, clinically recognizable changes do not appear until the chronic destruction of the articular cartilage has progressed too far and the disease is irreversible.

In order to diagnose the disease earlier, the focus has been shifted from imaging biomarkers to biological biomarkers. Several promising biological biomarkers have been found by researchers at SLU and Sahlgrenska, each representing a different stage of the destruction process. One specific biomarker has shown to increase in both blood and synovial fluid in horses with acute lameness, corresponding to an early stage of OA. If this early OA could be identified, it would be possible to intervene in time and the chronic and painful destruction of the joint tissues could be prevented, which could greatly improve the equine welfare.

The aim of this thesis was to investigate different machine learning approaches in order to find a promising method to be used in a decision support system for practitioners. The future system should be able to help diagnose OA, and specifically identify the different progression stages of structural changes in the joint, based on biological data. A Random Forest Classifier was developed along with a Spectral Clustering Algorithm, which was trained and evaluated on datasets with samples from both synovial fluid and serum. The results indicate some promise for the future decision support system, which will have to be evaluated further once more data is collected and the biomarkers for the remaining progression stages are added in the mix.

Keywords: machine learning, engineering, biomarkers, decision support system, random forest, spectral clustering.

Acknowledgements

First, I would like to thank my supervisor and examiner Magnus Karlsteen for giving me great support and the opportunity to combine two of my large passions in life in this thesis work. Thank you for providing me with this challenging and rewarding task and welcoming me to Chalmers.

I would also like to send a special thank you to Eva Skiöldebrand and her colleagues at SLU for their encouraging words and immense knowledge about horses and Osteoarthritis. It has been a pleasure getting acquainted with your research and I look forward to see where it may lead in a not too distant future.

Last but not least I would like to thank my friends, family and fellow equestrians for showing interest in my work and giving me great feedback and cheers along the way.

Linköping, June 2020
Lina Åberg

Contents

List of Figures

List of Tables

Notation

Dictionary

Biomarker Something that can be measured in a biological system

Hyperparameter A parameter which is manually provided as an input to the machine learning algorithm

Serum Blood without red or white blood cells, platelets or clotting factors

Abbreviations

$COMP$	Cartilage Oligomeric Matrix Protein
DSS	Decision Support System
ECM	Extracellular matrix
$ELISA$	Enzyme-Linked Immunosorbent Assay
IDE	Integrated Development Environment
NGF	Nerve Growth Factor
OA	Osteoarthritis
PCA	Principal Component Analysis

1
Introduction

This master's thesis is part of a collaborative research between Chalmers University of Technology, Swedish University of Agricultural Sciences, University of Gothenburg and Sahlgrenska University Hospital. It was carried out as a first step in introducing machine learning on the biological data collected from horses in the ongoing research of the progression stages of Osteoarthritis.

1.1 Problem specification

The main goal for this thesis was to investigate different machine learning algorithms in order to find a promising method to be used in a decision support system for practitioners. This future system should be able to help diagnose Osteoarthritis, and specifically identify the stages of structural changes in the joint based on biological data. This is important in order to better understand the complex data and to be able to indicate if early cartilage degradation has started before any clinical signs has occurred and to prognoses the disease, which would greatly improve the equine welfare.

The aims are described with the following questions:

- Is it possible to diagnose Osteoarthritis by using a machine learning approach on data collected from serum and/or synovial fluid?
- How does the performance of the algorithms differ between the dataset of serum versus synovial fluid?
- Is the machine learning approach a promising method to use in a decision support system for practitioners when diagnosing Osteoarthritis?

1.2 Hypothesis

The main hypothesis of the ongoing research states that there are four main stages of Osteoarthritis and that it is possible to diagnose the disease before any clinical signs occur, thanks to one of the biomarkers. In this thesis work, the hypothesis is that the machine learning algorithms will be able to classify new data into three possible outcomes: Osteoarthritis, Septic or Healthy. Another hypothesis is that the relatively small amount of data might affect the results. This study will help in determining a possible method to be further developed when more data is available.

1.3 Delimitations

Due to the limited time frame of 20 weeks only one classification algorithm was chosen to be developed and evaluated thoroughly in Python, along with one clustering algorithm.

Two datasets were collected from different data banks: the first based on serum and the second on synovial fluid. However, the datasets became quite small after filtering out the samples with the sought-after features. A third dataset was considered for the clustering algorithm, but it was unfortunately unusable for the task.

Lastly, only the early stage of Osteoarthritis was considered in this thesis, which corresponds to the increasing amount of the molecule fragment called COMP1 in an early stage of OA. Data is yet to be collected for the middle stages as well as the late stages of progression, which will therefore be used in future work.

2
Background

Osteoarthritis (OA) is a type of joint disease and the most common form of arthritis. It is a common disease in human athletes, but also the most common reason for lameness and poor performance in animal athletes, such as racehorses [2, 3]. If the early OA can be identified it is possible to intervene in time and the chronic and painful destruction of the joint tissue can be prevented.

In this chapter a brief background of the research project will be provided in order to explain the basics of the disease as well as relevant biomarkers.

2.1 Osteoarthritis, OA

OA is a chronic disease most commonly found in synovial joints, i.e. joints where the two bones are covered by articular cartilage as shown in figure 2.1. A synovial joint has the function of a weight-bearer to transfer load and enable movement [3]. The surfaces of the bones are covered by articular cartilage which works as a shock absorber. It also distributes the load across the surface of the joint, leading to increased joint mobility. The cells of the synovial membrane, located between the cavity and the outer capsule, secretes the thick synovial fluid which is kept in place by the surrounding joint cavity. The synovial fluid along with the articular cartilage, prevents friction between the bones [4]. A detailed image of a comparison between a healthy and a diseased joint is shown in figure 2.2. Once the destruction process of the joint has reached as far as shown in the image, the disease is irreversible and the joint causes severe pain and low mobility.

The causes of OA are many, but most common is mechanical damage from abnormal joint loading, trauma, impact injuries or ageing [3]. The initial trauma can lead to inflammation of the joint which causes abnormal and degraded articular cartilage, which in turn generates a greater sensitivity to future load, stiffness and disability [5]. The inflammation also leads to fragmentation of extracellular matrix (ECM) molecules which leaks into the synovial fluid [2]. The research team has found that unique fragments of one of these molecules, the Cartilage Oligomeric Matrix Protein (COMP) shown in the upper right corner of figure 2.2, are only present in diseased cartilage. They have also shown that a specific fragment of the molecule is explicitly formed during the early destruction process of OA in humans as well as in inflamed equine cartilage, which will make it possible to diagnose the disease earlier [6].

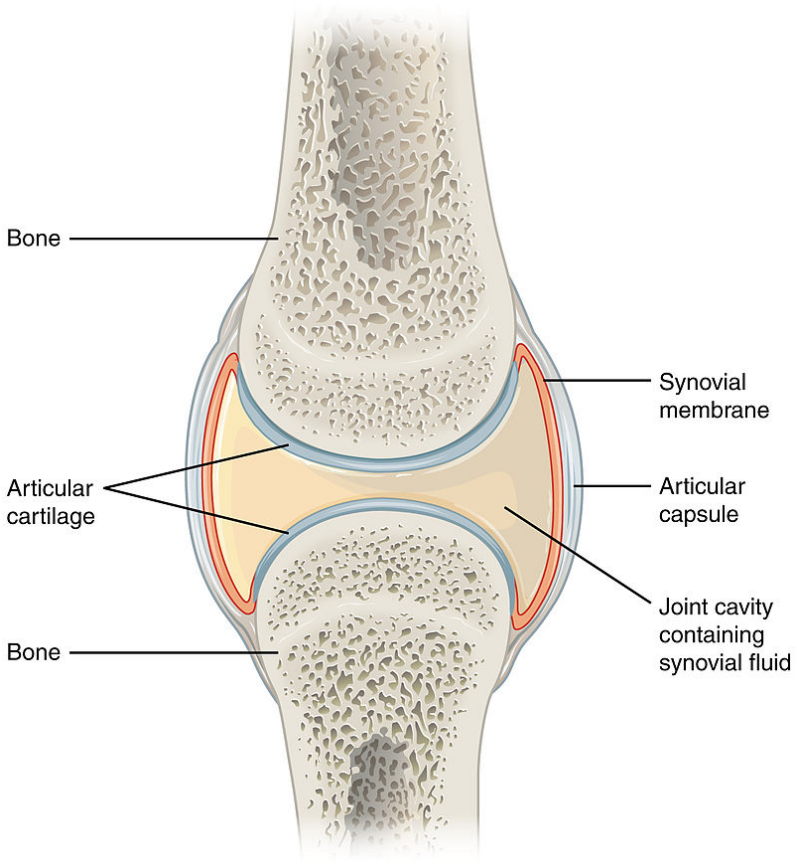


Figure 2.1: A cross section of a healthy synovial joint. (Betts G, et al. 2020)

When diagnosing OA today, different imaging techniques such as MRI or x-rays are used to examine the joint [2]. However, clinically recognizable symptoms does not appear until late in the destruction process, when the OA is already severe, irreversible and painful.

2.2 Biomarkers

In order to diagnose the disease earlier the focus has been shifted to biological markers, also known as biomarkers. A biomarker is something that can be measured in a biological system, and can be either chemical, physical or biological [7]. Biomarkers can be used as a diagnostic tool when examining a biological process, such as the destruction process of the joint tissues in OA. These diagnostic biomarkers are traceable and quantifiable, with for example an ELISA (see section 2.3.1), and indicate a biological change which happens during the progression of a disease [7].

The research team have identified four biomarkers for OA, each representing different stages of the destruction process of the articular cartilage [2]. The specific fragment from the COMP molecule, a fragment called COMP1, is one of the biomarkers. The amount of COMP1 has been found to increase in both synovial fluid [2] and serum [8] in horses with acute lameness in an early stage of OA.

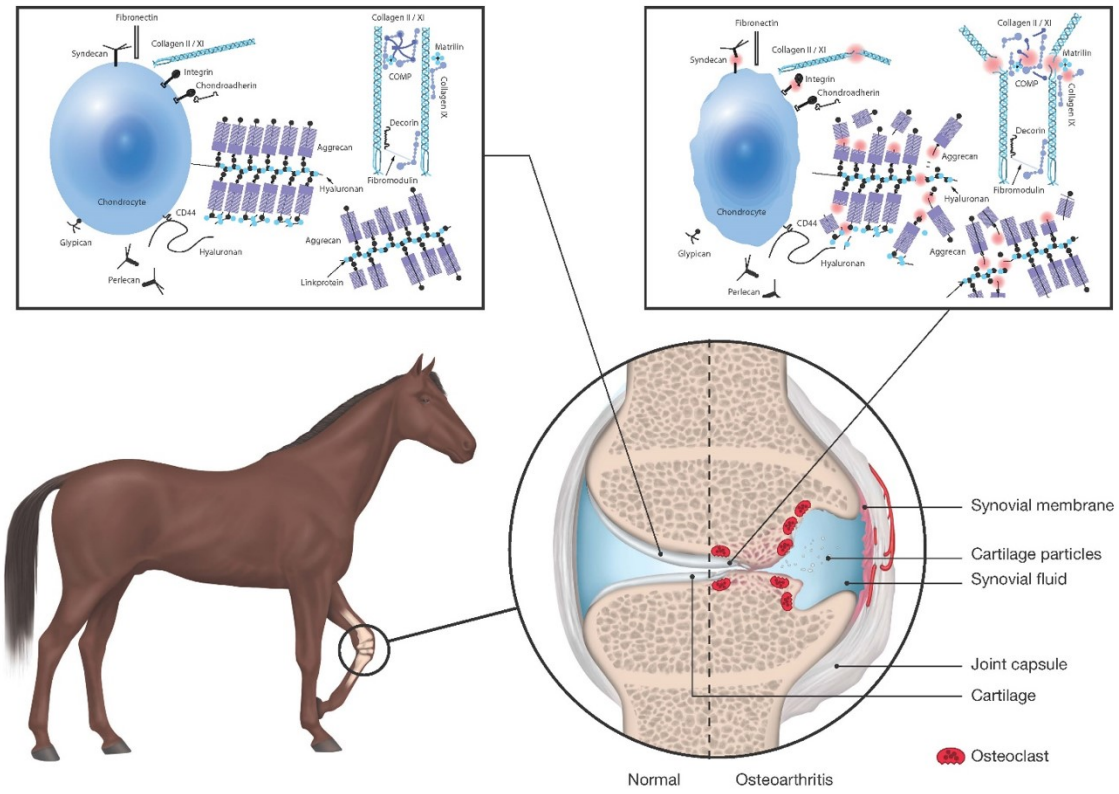


Figure 2.2: A comparison between a healthy joint and a joint with structural changes from OA. The figure shows the normal articular cartilage compared to the cartilage destruction leading to inflammation and friction between the bones. During the destructive process the ECM molecules are degraded into smaller matrix fragments, as shown in the upper right box, which leaks into the surrounding synovial fluid. (Andersson P, 2020)

Since OA leads to severe pain, another interesting biomarker is the Nerve Growth Factor (NGF) which is also included in the ongoing research. The NGF is a protein which has proven to play an important role in both pain evaluation and inflammation [9].

2.2.1 Cartilage Oligomeric Matrix Protein, COMP

The COMP protein molecule, also known as Thrombospondin 5, consists of five subunits each consisting of 755 amino acids [10]. Its appearance is shown in the detailed cartilage windows at the top of figure 2.2. It is still unknown what the exact function of the molecule is, but it has shown to interact with various cartilage ECM molecules, such as collagen, and other proteins such as the NGF [2]. The connection to collagen is of importance since once the collagen network in the joint is affected and destructed, the OA is irreversible [5, 10].

A main attribute of COMP in the OA process is that it is increased in cartilage at a very early stage [5]. The amount is then drastically decreased at the same time

as the COMP1 fragment, which is one of the subunits of the COMP molecule, is increased close to the cells. This process of molecule degradation is an indication of the early OA stage [2]. The COMP1 fragment is an antigen containing a very specific cleavage site after fragmentation, known as a neoepitope. The concentration of this specific COMP neoepitope is quantifiable in both synovial fluid and serum with an inhibition ELISA, making it possible for it to be used as a biomarker [2, 6].

2.2.2 Nerve Growth Factor, NGF

As mentioned earlier, the NGF is a protein that has been shown to increase in synovial fluid during injury, inflammation or chronic pain [11]. It therefor plays an important part in the pain and development of OA, where its existence is needed for neurons' growth and survival while it's blocking has shown promise for pain relief.

NGF was included in the research as a promising biomarker for pain evaluation, which is an essential factor in order to better understand animal behavior and progression of diseases such as OA. In human OA, the amount of NGF is increased in the synovial fluid which might indicate that it could be useful in determination of the progression stages and their pain levels [9]. However, the most recent findings indicate that they have a more complex connection to the COMP molecules than just pain management, details which are yet to be discovered. Additionally, it is not certain if the role of the NGF is the same in serum as in synovial fluid [9].

2.3 Data Collection

The biological data in form of either serum (blood without red or white blood cells, platelets or clotting factors) or synovial fluid has been collected from horses at various veterinary clinics in Sweden. In order to transform this raw data into usable data for machine learning, an inhibition ELISA was developed and performed by the research team to quantify the concentration of COMP1 in the samples.

2.3.1 ELISA

An ELISA, which stands for enzyme-linked immunosorbent assay, is a plate-based technique used for detection of target antigens, antibodies, proteins or hormones in samples [12]. In short, an antigen (or antibody) is immobilized on a plate or solid surface before being exposed to and linked with an antibody (or antigen). The connected antibody or antigen is in turn linked to an enzyme, which together with a substrate creates a final solution measurable by color. There are various forms of ELISA's depending on the antigen-antibody combination, and the one developed and used by the research team is an inhibition/competitive ELISA which will be explained briefly.

Antibodies

There are two types of antibodies that can be used in an ELISA. They are either monoclonal, polyclonal or a combination of both. For this assay polyclonal antibodies were raised against the neoepitope of the COMP fragment [2].

Polyclonal antibodies are quite complex and can vary from batch-to-batch if not tested and validated thoroughly [12]. They represent specificities to a wide representation of epitopes found in a single antigen, which can yield higher signal levels. However, they might share one or more epitope with closely related proteins which in turn might result in a higher non-specific signal. To reduce this problem, the polyclonal antisera mixture was purified by affinity chromatography to reduce the amount of unwanted substances [2].

Inhibition ELISA

An inhibition ELISA, or competitive ELISA, can be used for detection of epitopes or neoepitopes in a sample making it possible to quantify the concentration of a specific antigen. In this case, the target is a neoepitope from the antigen under investigation, which is the COMP1 fragment of the COMP molecule. Figure 2.3 shows an overview over the main steps taken during the measurement along with the incubation time in each step to get a sense of the total time spent on the assay. The following section of the inhibition ELISA process is a brief description, described more thoroughly by the research team in their published articles for synovial fluid [2] and serum [6].

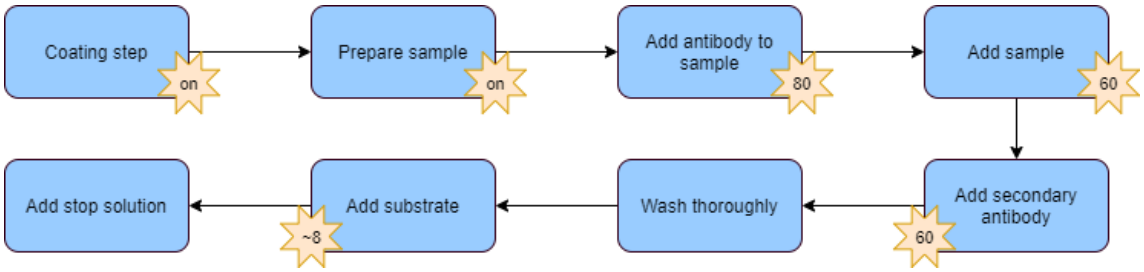


Figure 2.3: A flowchart showing the main steps of the inhibition ELISA process. The yellow stars symbolizes incubation, where *on* stands for overnight and the unit of the numbers are minutes.

Both NUNC plates and 96-well Sterilin plates were used. On the first day, the first two steps in figure 2.3 were completed. At the coating step, the NUNC plates were coated with a peptide and incubated overnight at 4°C. This peptide is considered a control antigen which will be able to connect to the same antibody. The sample preparation began with diluting the synovial fluid (or serum) sample in buffer before incubated on the Sterilin plates overnight at 25°C. These two steps allow the target molecules of the samples to immobilize on the plates.

On the second day the primary antibody was diluted in buffer before added to the Sterilin plates and incubated again for 80 min at 25°C. While waiting, the NUNC

plates with the peptide were washed and then blocked with buffer for 60 min at 25°C. The blocking step makes sure that there are no free sites in the wells left [12]. A total of $100\mu L$ of the sample from the Sterilin plates were added to the NUNC plates, which was incubated for 60 min at 25°C. The NUNC plates were then washed before the secondary antibody was added and incubated for another 60 min at 25°C. The secondary antibody is enzyme conjugated and used for the detection step.

After the final incubation, the NUNC plates were thoroughly washed six times before incubated with substrate at 25°C. The substrate is catalyzed by the enzyme to color the solution, making it readable. After about 8 minutes the stop solution of H_2SO_4 was added and the ELISA was completed. The absorbance was then read at the wavelength of 450 nm with a light microscopy.

2.3.2 Sampling

The synovial fluid and serum samples used in the datasets were originally collected for other studies. Fortunately, those studies had relevant inclusion criteria which made it possible to use the samples for this study as well. The samples come from various biobanks, consisting of samples from both live horses and horses subjected to euthanasia.

2.4 Research status

As mentioned earlier, the research team has found that the amount of the COMP1 neoepitope is significantly increased in the synovial fluid of horses with acute lameness compared to either healthy horses or horses with chronic lameness or structural OA [2]. They have also found that the neoepitope of COMP1 appears to be nonexistent in healthy articular cartilage and that it only shows in low concentrations in healthy equine joints.

They also reached the same conclusion for serum, where an increased concentration of the COMP1 neoepitope is connected to acute lameness [6]. Furthermore, the COMP1 neoepitope concentration in serum is shown not to be influenced by age, short-term exercise or time of day. Next step is to verify the three promising biomarkers connected to the middle stages as well as later stage of the degradation process.

3

Theory

In this chapter the basics of machine learning will be explained along with how the performance of the algorithms is measured in this thesis. One relevant area of use for machine learning is in different decision support systems, which will be briefly described in order to provide necessary knowledge to the reader.

3.1 Machine Learning

Machine learning is based on probability theory and automated methods for data analysis in order to detect patterns in data, predict future data or perform decision making of other kinds [13]. Two of the three main fields of machine learning will be used in this thesis, which are the Supervised and Unsupervised Machine Learning.

The supervised learning approach is predictive, meaning that the algorithm is trained in order to classify future data based on previously known data. The algorithm is given a training dataset of inputs x connected to labeled outputs y, according to the following equation [13]:

$$D = \left\{(x_i, y_i)\right\}_{i=1}^{N} \tag{3.1}$$

where D is the training set and N the number of samples. The input x_i can be multidimensional and consists of features of the data, such as age or height. Supervised machine learning can be used on either a classification problem or a regression problem. A classification problem can be described as an approximation of an unknown function f, $y = f(x)$, where the goal is to learn to generalize predictions on new inputs based on estimations of f, using $\hat{y} = \hat{f}(x)$ [13]. The output variable of a classification problem, which is the case in this thesis, is a categorical or discrete value such as a class or a diagnose. In case the output variable would be a real or continuous value it would be called a regression problem.

The unsupervised learning approach is descriptive, meaning that its purpose is to discover new knowledge in the given data. In this approach, the algorithm is only provided with inputs x_i and then searches for unknown patterns in the given data [13]. One main difference to the supervised learning, is that in unsupervised learning multivariate probability models are needed instead of just a single variable.

3.1.1 Random forest

The random forest algorithm is a supervised learning method based on a classification or regression problem. The random forest consists of several independent decision tree classifiers which together form a forest for the final classification. Each decision tree is considered a weak classifier, which is defined as a classifier which only performs slightly better than a random guess [14]. When collecting these weak classifiers into a complete forest, the result is a better classifier with a higher accuracy.

The key to the forest is to create a randomness with a lower correlation between the trees. If the randomness is low, then each decision tree in the forest will look quite similar to the other and the algorithm would be neither robust nor work well on previously unseen data [14].

Decision tree

The structure of a decision tree classifier is shown in figure 3.1 below. It consists of a series of questions asked, chosen depending on the previous answer [14]. The tree starts at the root node where a question with a binary output is asked about the input data. The answer leads to the next question from the connected child node below. If this node is not a leaf node, a new question is asked which either leads to yet another question or ends up in a leaf node. The leaf node contains the final decision, class or label which is used to classify the input data, and connects the input data with this output label.

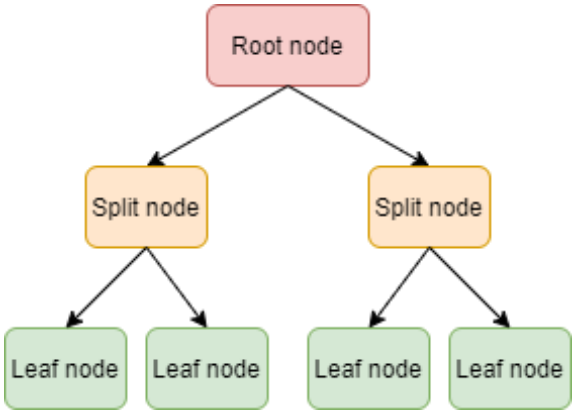


Figure 3.1: The flowchart structure of a hierarchically organized decision tree classifier.

Randomness

When the forest is created some randomness is important in order to get a robust result. The randomness provides with generalization due to that the accuracy on previously unseen data will increase if the trees used in the forest consists of different combination of nodes [14]. The result is shown in figure 3.2.

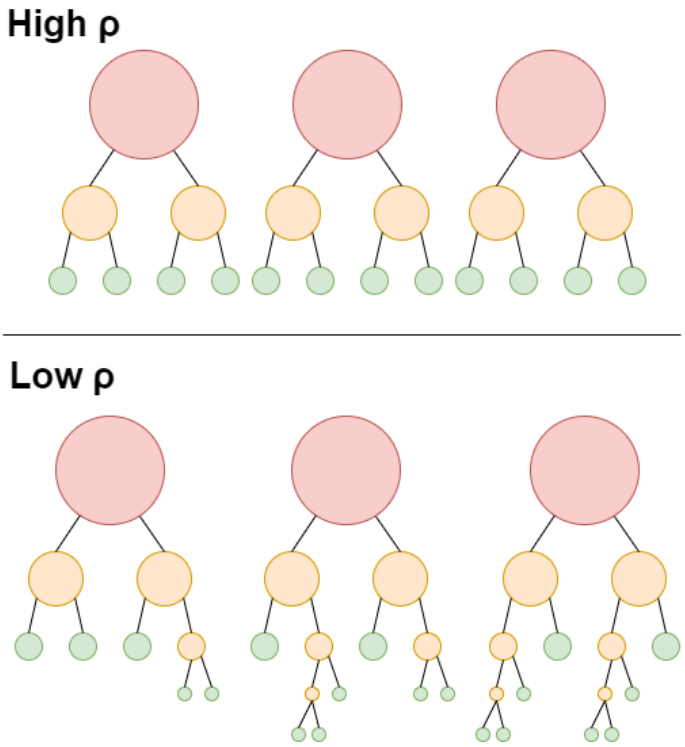


Figure 3.2: The change in tree collection depending on the value of ρ. The forest on top with a high value of ρ consists of very similar trees, meaning a low randomness and a high tree correlation. The bottom forest however receives a higher randomness with lower tree correlation when using a lower ρ.

There are two common ways in how to include this. The first technique is called bootstrap aggregation, or bagging for short. Here, new bootstrapped datasets are created by randomly sample with replacement from the training data for each decision tree. This creates slightly different datasets for each tree, leading to different decision trees. The other technique is called random subspace method, or feature bagging. It is used to reduce the correlation of the trees by using randomly sampled feature sets for each tree instead of all features.

Maximum-margin separation
The maximum-margin approach is a key property in support vector machines, but might as well be of use in random forests [14]. Given a linearly separable dataset, the goal is to find the split between the two classes which leads to a maximized margin, i.e. the most optimal separation. In the forest, it is not necessary for each tree to use maximum-margin separations, but the combination of all trees will resolve in the desired optimal separation.

Entropy
This best split can be determined by calculating the entropy of the data on both sides of the split. By looping this calculation on all potential splits the best split is found where the entropy is at its lowest. This is calculated by the following equation:

$$Entropy = -\sum_{i=1}^{c} p_i \times log_2 p_i \tag{3.2}$$

where p_i is the probability of the class i in the data [14]. This leads to that with a higher probability comes a higher certainty of the prediction. The second part, $-log_2 p_i$, is known as the uncertainty value and $\sum_{i=1}^{c} p_i$ is the weighted sum. The weighted sum increases with the number of classes, which is beneficial since a low entropy then corresponds to one or a few classes.

Important parameters
When the decision trees are collected into a complete forest, some main parameters are to be considered and optimized in order to get the best accuracy [14]. Three of the most important parameters are:

- The forest size, **T**.
  A small number of trees will result in an imperfect generalization due to a larger uncertainty.
- The tree depth, **D**.
  A maximized tree depth will result in a higher risk of overfitting, while a too small depth will lead to underfitting instead.
- The amount of randomness, ρ.
  A smaller value of p means less parameters, leading to increased randomness of each tree which reduces the correlation of the trees. A larger randomness also produces higher uncertainty between the classes, as shown in figure 3.2.

3.1.2 Spectral clustering

Clustering is a unsupervised learning method used to group together similar data points into the same group, known as a cluster. The spectral clustering technique is a relatively new method based on graph theory and linear algebra [15], where the clusters are divided based on the edge nodes in the graph. Each step included in this technique will be further explained in this section.

Principal Component Analysis, PCA Principal component analysis, or PCA, is a statistical technique used to reduce the dimensions of the data [1]. It is used in order to more easily visualize and analyze the data, which is needed in order to create a graph for spectral clustering.

PCA uses summary indices, called principal components, to express the extracted important part of the data. These components can be lines, planes or hyper-planes in a K-dimensional space, and they are defined using the least squares approximation [1]. The goal for PCA in spectral clustering is to reduce the number of dimensions to two, to simplify the visualization and transform the data onto new axes. This is shown in figure 3.3, where the red dot is the mean-centered average and the vectors PC1 and PC2 is the principal components. These two vectors will create the new transformed plane.

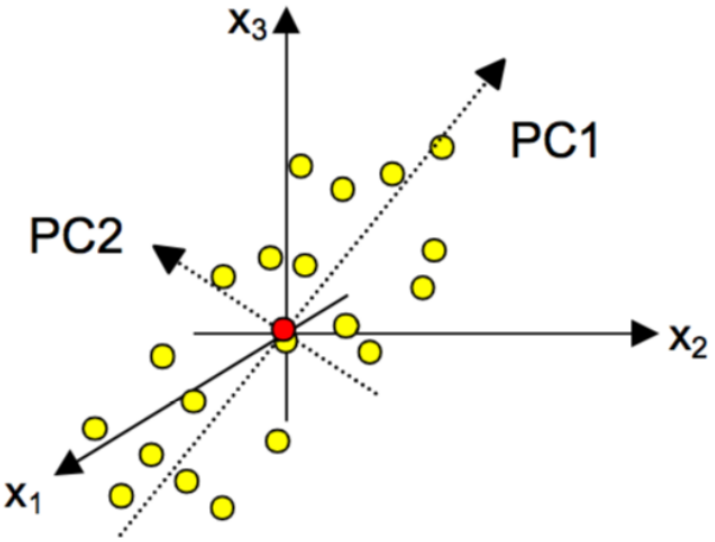


Figure 3.3: Visualization of PCA, where the two principal components are shown as PC1 and PC2 [1].

Similarity graph
A similarity graph is used to represent the data by connecting data points to each other based on a certain relationship [16]. This could be done in several different ways, but the two most common similarity graphs are based on ϵ-neighborhood or k-nearest neighbor.

The ϵ-neighborhood graph connects data points which falls inside a certain radius ϵ, where ϵ is a real value. This groups data points together based on distance [16]. The k-nearest neighbor graph however, connects data points to the k nearest neighbors regardless the distance, where k is the number of neighbors chosen. The result of the similarity graph is a $N \times N$ matrix, where N stands for the number of samples.

Graph Laplacian
The graph Laplacian, or simply the Laplacian matrix **L**, is calculated by $L = D - A$ where **A** is the adjacency matrix and **D** is the degree matrix [15].

The adjacency matrix **A** is based on the similarity matrix using distance. If the value in a cell, i.e. the distance, is smaller than the threshold, then the adjacency matrix gets the value of 1, otherwise the value is 0 [15]. The value of 1 means that there is an edge present between the nodes.

$$A_{ij} = \begin{cases} w_{ij} & \text{weight to edge } (i,j) \\ 0 & \text{if no edge between i,j} \end{cases} \tag{3.3}$$

The number of edges is then used in the degree matrix **D**, which is a diagonal

matrix [15]. The degree of a node is determined by the number of edges connected to it and is written as:

$$D_{ij} = \begin{cases} deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

where $deg(v_i)$ is the degree of the vertex i.

Eigenvalue and eigenvector
The eigenvalues and eigenvectors of the Laplacian provides useful information about the clusters. The number of eigenvalues with the value of 0 corresponds to the number of clusters in the dataset [15]. For example, if only one of the eigenvalues is 0, then that means that there is only one connected component, or cluster, in the dataset.

The first eigenvalue with a non-zero value is called the spectral gap [15]. The value of the spectral gap describes the density of the graph, for example if a graph with N=10 nodes was completely connected, then the spectral gap would get the value of 10.

The second eigenvalue describes the approximate minimum cut needed to separate the graph into two connected components [15]. This value is called the Fiedler value, which would be 0 if the graph was already divided in two connected components. The values of the Fiedler vector, which is the eigenvector of the second eigenvalue, depends on which side of the cut that each node belongs to.

If the whole graph is connected in once component, then the first eigenvalue will be 0. If the graph is already separated in to two components, then the second eigenvalue, i.e. the Fiedler value, would also be 0. The next eigenvalue will be close to zero if there are more possible cuts available [15]. The first large gap between eigenvalues gives an indication on how many clusters exist in the dataset.

For example, if there are three eigenvalues before the larger gap, then the data can be divided into three clusters. To know where to split the components the eigenvectors of the first three eigenvalues are used. These vectors can be put together and used in the next step, which is the K-means clustering.

K-means
The K-means algorithm is a centroid-based clustering algorithm which uses the number of clusters as input. It is an iterative algorithm which updates the parameters in each step [13].

With centroid-based means that the clusters are represented by centroids placed in the centers of the clusters. The initial starting points for the centroids are randomly selected, and the surrounding data points will belong the cluster which centroid is closest to them [15]. For the second iteration, the centroids are moved to be placed at the center of the cluster, and the data points will once again check to see which

centroid is closest. If none of the centroids are moved in an iteration of the algorithm, the clustering is complete. See figure 3.4 for a flowchart of the K-means algorithm.

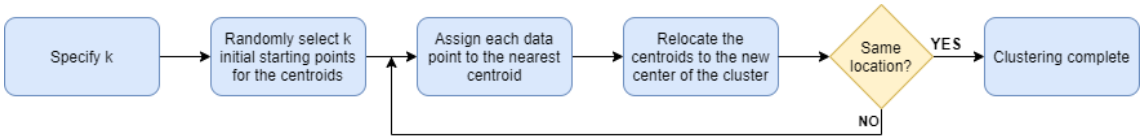


Figure 3.4: The steps of the K-means algorithm.

The K-means algorithm alone can be used as a clustering method as well, without the previous steps included in the spectral clustering method. It is a very common approach since it is easy to implement.

3.2 Performance Measurements

There are several ways in which to estimate the performance of the algorithms and measure the success accurately. Most of them are based on the confusion matrix and provide different angles to examine the result, depending on the machine learning problem at hand.

3.2.1 Confusion Matrix

A visualization of a regular confusion matrix is shown in figure 3.5 below. Each cell in this matrix provides a value to be used in further estimations. The true positive (TP) value represents the number of correctly classified positive samples, whereas the false positive (FP) value represents the number of samples which are wrongly classified as positive. The true negative (TN) value represents the correctly classified negative samples, and the false negative (FN) value represents the samples which are classified as negative but which are in fact positive.

3.2.2 Accuracy

The accuracy is the most used performance measurement since it shows how accurate the algorithm is based on all samples. In this calculation it is however important to remember to take the balance of the dataset into account to make sure that the algorithm actually does perform better than a random classifier. The accuracy is calculated with the equation below.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.5}$$

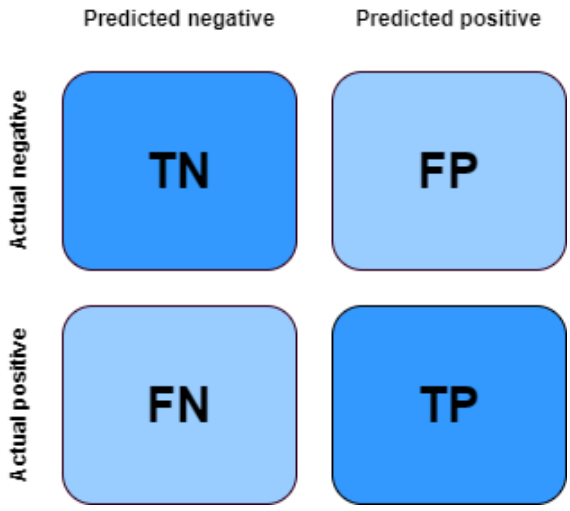


Figure 3.5: The confusion matrix table, where TN = True Negatives, TP = True Positives, FP = False Positives and FN = False Negatives.

3.2.3 Precision

Precision is a measure of quality of the algorithm, also known as the positive predicted value rate. If given the maximum value of 1, precision explains that for every sample labeled as positive by the algorithm, they indeed were positive. It is calculated by

$$Precision = \frac{TP}{TP + FP} \tag{3.6}$$

3.2.4 Recall

Recall is a measure of quantity or sensitivity, and can be seen as the true positive rate. If given the maximum value of 1, that means that out of all positive samples in the set every one of them was labeled positive by the algorithm. It is resembled by

$$Recall = \frac{TP}{TP + FN} \tag{3.7}$$

3.2.5 F_1-score

The F_β-score is used to combine the results from both precision and recall to make it easier to interpret. There are several more versions of the F_β-score depending on how to weight the parameters, but the F_1-score will be the one used in this thesis and it is calculated by the harmonic mean, namely

$$F_1 score = 2 \times \frac{precision \times recall}{precision + recall} \tag{3.8}$$

3.3 Decision Support Systems

Advance Decision support systems, DSS, are being more and more integrated in modern healthcare and are becoming of great importance due to the increasing volume of patient data [17]. The DSS are computer based systems developed to help healthcare practitioners in various decision making tasks. The first DSS were used in the 1970s with great success, leading to an improved performance of the practitioners. Since then, with each new advancement in biomedical technology comes an increased complexity and volume of the medical data, causing an even stronger need for more advanced decision support [17].

The DSS can be divided into three types, which are diagnostic support, alert- and reminder systems and patient management systems. In this thesis only diagnosis support is of relevance.

Computer-aided diagnosis support systems are available in various fields of healthcare and are most often based on either signal and image processing, journal information or biomedical informatics. A quite new field of interest is in bioinformatics, which combined with medical informatics has shown great promise for both prognosis and diagnosis [17]. The difference is that bioinformatics might need a few extra steps in order to process the raw data into usable input data.

4
Methods

Several machine learning methods were evaluated and developed in Matlab in order to find the most promising algorithm. The chosen algorithm was then developed and evaluated further in Python through the process described in this chapter.

4.1 Overview

Figure 4.1 below shows an overview of the steps taken during the implementation process. The datasets created for the classification task was prepared and scrutinized before inserted into the Matlab algorithms. The method with the highest accuracy was considered most promising, and was further developed in Python. The clustering algorithm was implemented directly in Python.

Figure 4.1: The main steps of the method implementations.

4.2 Data preparation

The two main datasets were divided into an additional two sets with fewer features. All datasets were imported as xlsx-files from Excel.

The datasets were prepared and manipulated in order to function properly with the different machine learning algorithms. The data had to be presented as either numerical or categorical data, meaning that some features had to be manually modified and coded from text to numbers, saving a key for evaluation purposes. In case of imbalanced datasets, oversampling was used in order to gain a larger amount of samples for each dataset.

4.3 Classifier implementation

There are several implementation steps involved in this method, all which will be described in this section.

4.3.1 Method selection

In order to choose the most promising machine learning method to investigate further, the first step was to evaluate six different algorithms in Matlab. The datasets were trained on each method using k-fold cross-validation, with $k = 5$, and evaluated by combining the validation results. The methods that were investigated, along with the associated Matlab-function, are shown in table 4.1.

Table 4.1: Matlab functions used for method selection

Method	**Function**
K-Nearest Neighbors, KNN	fitcknn
Support Vector Machines, SVM	fitcsvm
Decision Trees	fitctree
Random Forest	fitensamble, *bag*
AdaBoost	fitensamble, *AdaBoostM1*

The datasets were modified into healthy or non-healthy samples, since these algorithms are based on classification with only two labels. The training of each method was evaluated using Matlabs *predict*-function and visualized using *confusionmat*, before calculating the average accuracy.

4.3.2 Random forest classifier

The random forest algorithms were developed in Python (using the IDE PyCharm) in two separate parts. First, the method was implemented without any external machine learning libraries, which was then evaluated and compared to an implementation using the *Scikit-learn* machine learning library. Two additional libraries of importance that were used was *numpy* and *pandas*.

Using Scikit-learn library

The Scikit-learn library (*sklearn* in short) simplifies the implementation in many ways. Same as before, the data was divided into training data and testing data, where the training data was trained using a gaussian classifier on bootstrapped datasets. This classifier used the forest size as an input, along with number of features and maximum tree depth. It is possible to generalize this algorithm even further, but it was implemented to be similar to the other for comparison purposes. Some parameters were investigated and tuned in a later stage.

The trained forest was tested with the test set using the *predict*-function and the accuracy was calculated using *accuracy_score*. The sklearn library also provide a possibility to find out which features were of most importance in the forest.

Hyperparameter tuning
In order to find the optimal settings of the hyperparameters, meaning parameters to be manually provided to the machine learning algorithm, each algorithm was evaluated in the range described in table 4.2.

Table 4.2: Parameters to be optimized

Parameter	**Description**	**Range**
Size of training set	Percentage of data used for training	70-90 %
Size of bootstrapped set	Number of samples used for each tree	10-70
Forest size	Number of trees in the forest	10-1000
Tree depth	Longest path from root node to leaf node	2-10
Number of features	Number of features for each tree	2-6
Number of epochs	Number of training iterations	10-250

However, if this was to be evaluated for each and every setting from table 4.2, it would need at least $3 \times 7 \times 10 \times 5 \times 5 \times 6 = 31500$ iterations. Instead, a randomized combination of the settings for each parameter was evaluated using the function *RandomizedSearchCV* from sklearn. All parameters from table 4.2 were not available in this function, but those who were can be found in table 4.3 along with the possible setting for each parameter. The tuning was done using k-fold cross-validation with k=3.

Table 4.3: Settings for randomized parameter tuning

Parameter	**Setting**
Forest size	10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000
Number of features	2, 3, 4, 5, 6
Tree depth	2, 4, 6, 8, 10
Bootstrapped	True, False

The results from the randomized parameter tuning narrowed down and concentrated the range of the settings, creating a new combination to be used the sklearn-function *GridSearchCV*. This function does not randomize the sampling but evaluates all possible combinations and displays the best parameters for an optimized algorithm.

The parameters which could not be tested through the tuning functions were manually evaluated using the already optimized combination of the tuned parameters.

Without machine learning libraries
The following implementation was developed without sklearn and based on the theory in chapter 3.1.1. The data was imported and its balance investigated, before being randomly divided into training data and testing data. Bagging was then used on the training data to create bootstrapped training sets to be used for each decision tree. The random subspace method was also implemented to increase the randomness in the forest, and is used as an input parameter to the decision tree algorithm. The steps of the decision tree algorithm is shown in figure 4.2.

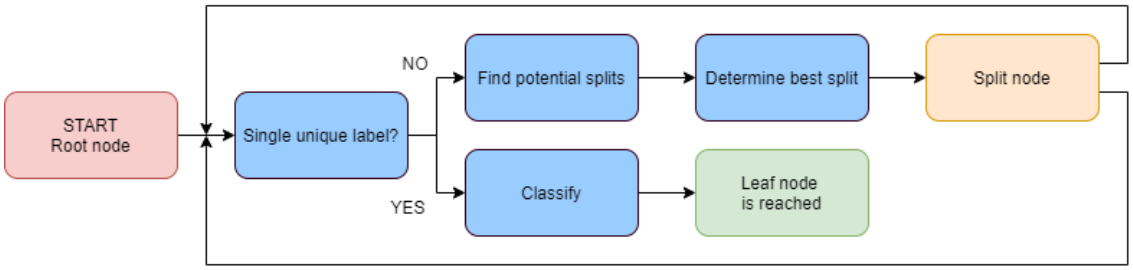


Figure 4.2: A flowchart showing the decision tree steps.

The decision tree algorithm takes in the data as a pandas dataframe and determines if the input is a leaf node. That is done by checking if the data is pure or if either the appointed maximum depth of the tree is reached or there are only a few samples left in the input data. If one of the above is true, then the data is classified to the class of majority and the loop ends. If we are not at the leaf node however, the next step is to find potential splits of the data.

All potential splits are found for a few randomly chosen features. The best split is determined by finding the split resulting in the lowest overall entropy on both sides of the split, before splitting the data. Again, both sides of the split runs through the loop which continues until all data points have been classified. The size of the complete random forest is decided with an input value for number of trees, as well as number of samples in the bootstrapped training sets, number of features to use for each tree for the random subspace method along with the maximum tree depth.

To evaluate the trained algorithm, the test set was used and the predictions was compared to the true values of the testing set. The result was visualized with a confusion matrix heatmap based on a *pandas* dataframe, as shown in figure 4.3 below. Through the confusion matrix the accuracy, precision, recall and F_1-score was calculated. The parameters used in this method was based on the knowledge from the sklearn algorithm in order to optimize the algorithm.

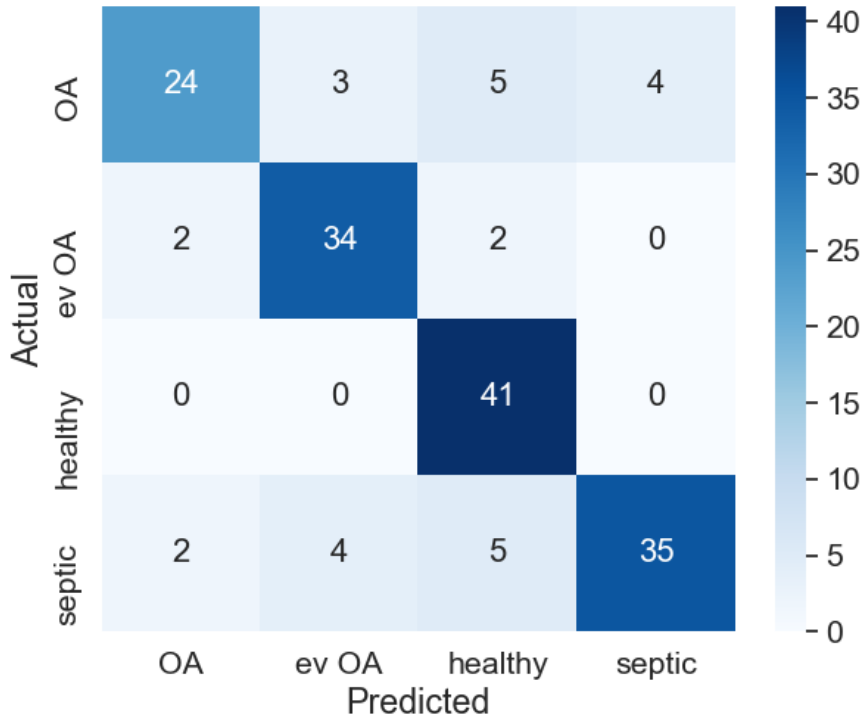


Figure 4.3: An example of the confusion matrix heatmap used for evaluation calculations. The precision and recall is based on the column and row of the OA class respectively.

4.4 Clustering implementation

The spectral clustering algorithm was implemented directly in Python with the help of several libraries, such as numpy, pandas and sklearn. The functions used from external libraries are described in table 4.4. The only variable to be manually provided to the algorithm is the number of clusters. Another variable, the distance for the adjacency matrix, was looped in order to find the most optimized setting.

The data was imported and transformed into a pandas dataframe. The features of the dataset were specified to the matrix x, while the target column containing the labels were saved to y. Before applying PCA on the data, the features were scaled with *StandardScaler*. It standardizes the features onto a unit scale to prepare the transition to the PCA.

The PCA was performed in order to make the data easier to visualize, and the dimensions were reduced to two principal components. An adjacency matrix was arranged with the *radius_neighbors_graph*-function based on distance, and using *csgraph* the graph laplacian was created. The eigenvalues and eigenvectors were then calculated with *linalg.eigh* before the final clustering was done using *KMeans*.

The algorithm was used on data with known labels, which made it possible to evaluate the resulting clusters. By using the labels saved in y, the predicted clusters were transformed into labels and the accuracy was calculated with *accuracy_score*.

Table 4.4: Functions used for the spectral clustering algorithm.

Function	**Library**
StandardScaler	sklearn
PCA	sklearn
radius_neighbors_graph	sklearn
csgraph	scipy
linalg.eigh	scipy
KMeans	sklearn
accuracy_score	sklearn

5
Results

This chapter will provide a presentation and visualization of the results gained from the implemented algorithms.

5.1 Datasets

After choosing what features to be used in the different algorithms, two datasets were created from the original datasets of serum and synovial fluid samples. In order to make larger comparisons, two additional datasets were created with less features, which resulted in datasets with more samples.

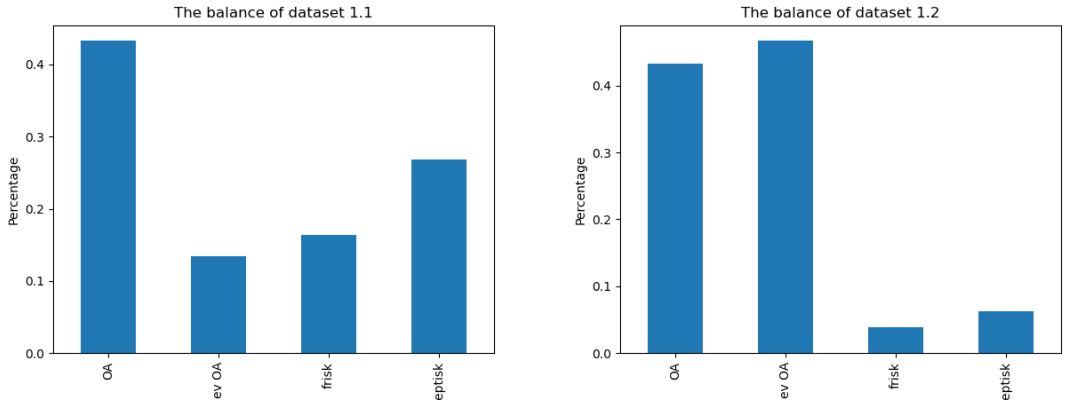
5.1.1 Serum samples

The original dataset of serum samples consisted of 335 samples collected from different biobanks, however they were not all useful for this task. After selecting the samples with the chosen features of age, gender, breed, COMP1, NGF and diagnose represented, the final dataset (dataset 1.1) consisted of 67 samples. The additional dataset (dataset 1.2), when using only the values for COMP1, NGF and diagnose, consisted of 287 samples.

The balance of these datasets are shown in figure 5.1, where it is clear that they are both highly unbalanced and over-represented of samples diagnosed with OA (or possibly OA). The results with these datasets was therefor compared to results from randomly balanced versions of the datasets. To avoid decreasing the amount of samples, the balance was corrected with oversampling which filled the other groups with duplicates. After balancing, the datasets reached a size of 116 and 536 samples respectively.

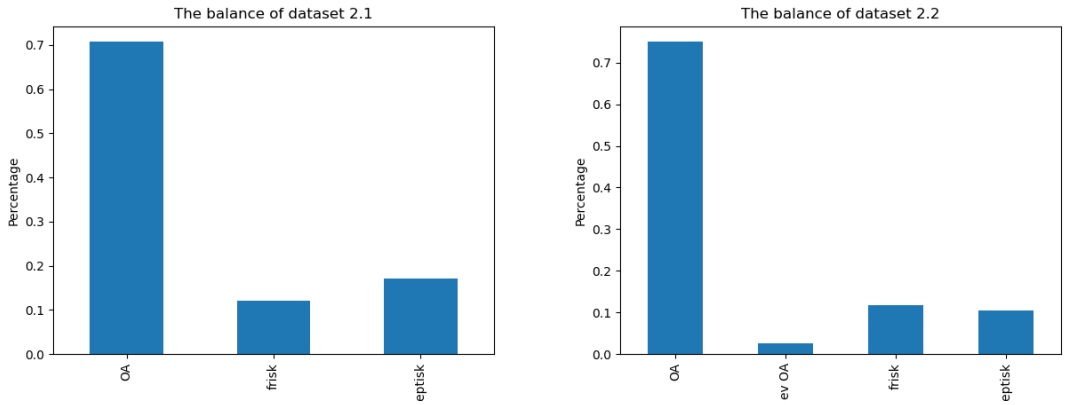
5.1.2 Synovial fluid samples

The original dataset of synovial fluid samples consisted of 178 samples collected from different data banks. The chosen features of synovial fluid are the same as for the serum, except adding the specific joint from which the sample was taken. After the selection, the final dataset (dataset 2.1) consisted of 82 samples and the additional dataset (dataset 2.2) consisted of 152 samples. Their balance is shown in figure 5.2. These datasets are even more unbalanced, leading to a randomized accuracy of around 70% by simply guessing the diagnose OA. After oversampling, the new balanced datasets consisted of 174 and 456 samples respectively.



(a) Dataset 1.1. **(b)** Dataset 1.2.

Figure 5.1: The balance of the serum datasets.



(a) Dataset 2.1. **(b)** Dataset 2.2.

Figure 5.2: The balance of the synovial fluid datasets.

5.2 Classification algorithm

This section will describe the outcome from the Matlab selection as well as the results obtained with the python algorithms for the random forests.

5.2.1 Method selection

Based on the implementation described in section 4.3.1, the method was selected in Matlab through evaluating the accuracy of the algorithms. The results are shown in figure 5.3, where the random forest algorithm got the highest accuracy on both the serum as well as the synovial fluid dataset. For this task the unbalanced datasets were used, which was considered sufficient enough since they reached an accuracy above the random limit for each dataset. The accuracy of 90.11% on the synovial fluid dataset 2.1 was considered very promising, while an accuracy of 78.89% for serum was in need of improvement in Python.

method	accuracy
'KNN'	0.69524
'SVM'	0.77302
'Decision tree'	0.69683
'Random Forest'	0.78889
'AdaBoost'	0.75714

(a) Serum, dataset 1.1.

method	accuracy
'KNN'	0.66591
'SVM'	0.76477
'Decision tree'	0.74205
'Random Forest'	0.90114
'AdaBoost'	0.86477

(b) Synovial fluid, dataset 2.1.

Figure 5.3: The results from the method selection in Matlab.

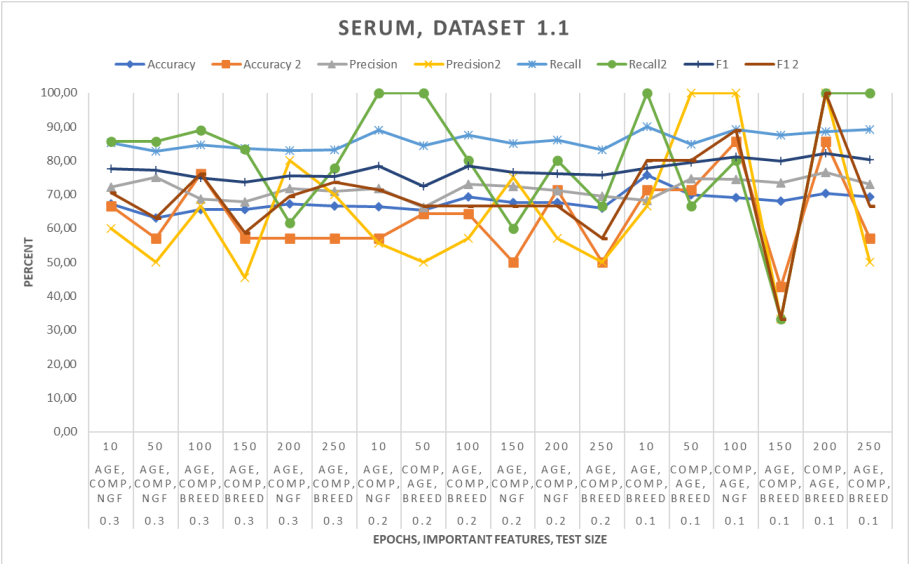
5.2.2 Random forest with Scikit-learn library

The sklearn algorithm was evaluated first in order to narrow the parameter range to be used on the original algorithm. The detailed results can be found in Appendix 1.

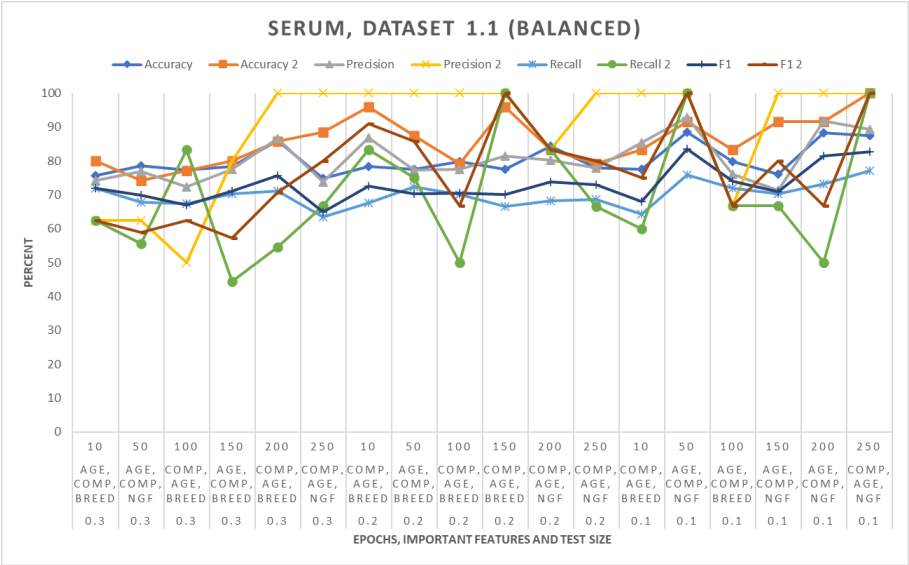
Serum dataset 1.1
The mean accuracy of the sklearn algorithm reached 63.49% after the grid search, which is a decrease with over 6% from the regular training when it got 67.81%. Unfortunately, neither of those results are considered good enough. The grid search gave the best results in a wide spread, but was improved when using the parameter setting of a maximum depth and maximum features of 4 and a forest size of 100.

The results from the balanced dataset was significantly better, obtaining the best results with the mean max depth of 6.56, max features of 4 and mean forest size of 333.33. The basic training gave a mean accuracy of 80.20%. When using the best estimators from the grid search, the accuracy improved with 7.25%, landing on 86.01%. The highest accuracy and overall performance was reached when training on 90% of the data and leaving 10% for testing. The specifics are visualized in figure 5.4.

The algorithm also calculated the most important feature during training. For the unbalanced dataset the age was considered most important, whereas the COMP1 was more important on the balanced dataset, although it shifted back and forth.



(a) Original dataset 1.1.



(b) Balanced dataset 1.1.

Figure 5.4: Results of the sklearn-algorithm on dataset 1.1, showing both the results from the original training along with the results when using the optimized parameters from the grid search, which is marked with a '2'.

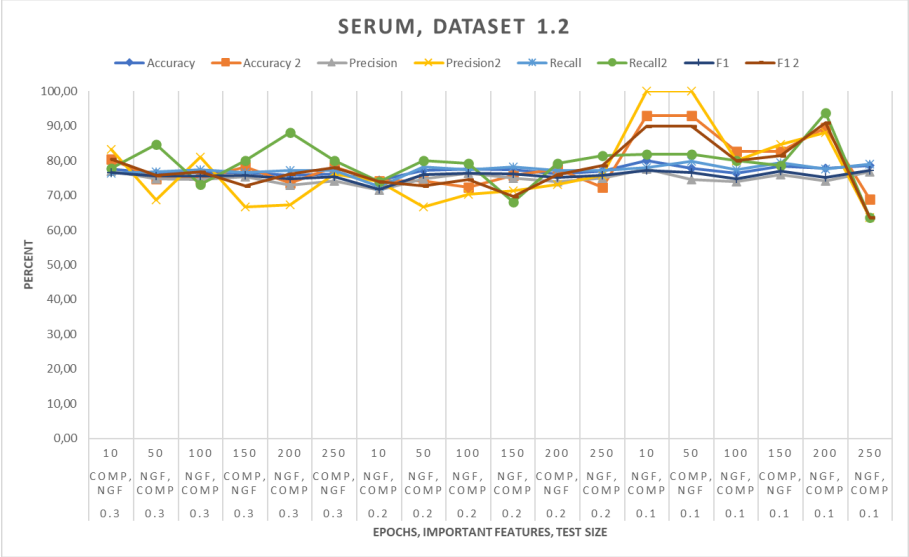
Serum dataset 1.2

The results from this dataset was overall better and more even than dataset 1.1. The mean accuracy reached 77.12% on the original dataset, which was then decreased after the grid search to a mean of 78.70%. The grid search gave the best result when using the parameter setting of a mean max depth of 5.67 and a forest size of 400. The overall highest scores were reached when using a test size of 10%.

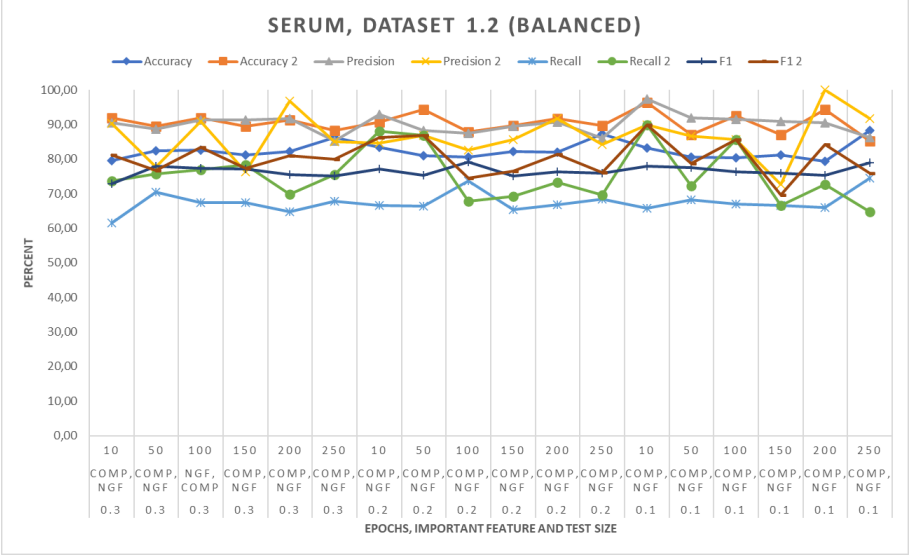
As for the balanced dataset it was again significantly better, showing best results with the mean max depth of 8, max features of 4 and mean forest size of 444.44. Its

training gave a mean accuracy of 82.42%, and when using the best estimators the accuracy improved with 9.83%, reaching a mean accuracy of 90.52%.

As seen in figure 5.5, the accuracy is improved for all but one run when using the optimized settings. The recall is similarly overall increased, at the cost of a lower precision. The algorithm also claimed the COMP-value as the most important feature in all runs but one, while the original dataset chose the NGF.



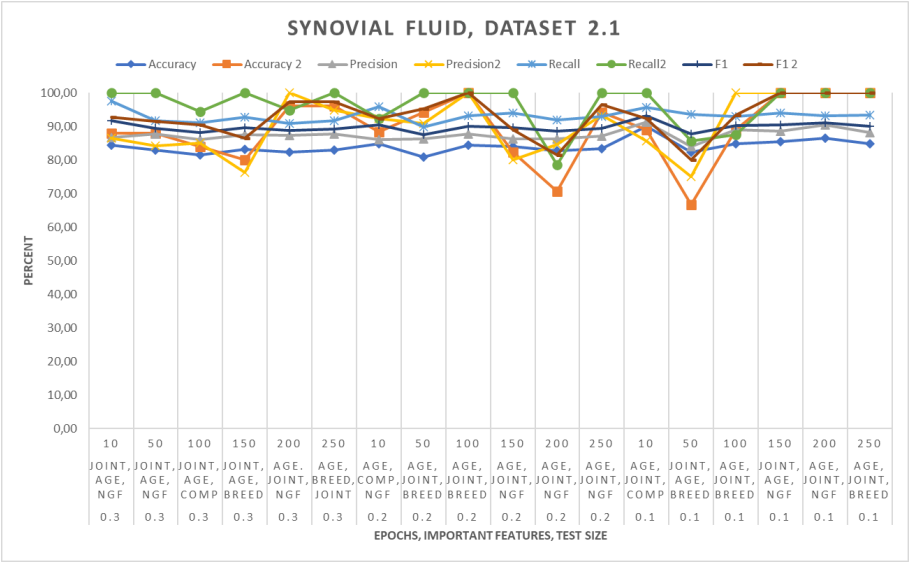
(a) Original dataset 1.2.



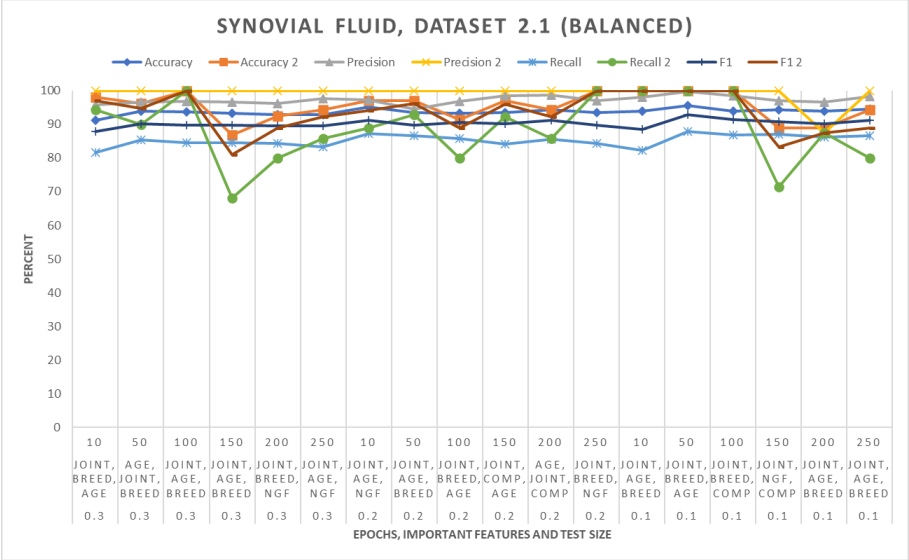
(b) Balanced dataset 1.2.

Figure 5.5: Results of the sklearn-algorithm on dataset 1.2, showing both the results from the original training along with the results when using the optimized parameters from the grid search, which is marked with a '2'.

Synovial fluid dataset 2.1
The mean accuracy for the original dataset 2.1 reached 83.94% and increased to a mean of 89.22%. The grid search gave the best result when using the parameter setting of a max depth of 5 and a mean forest size of 411.11. The results are shown in figure 5.6.



(a) Original dataset 2.1.



(b) Balanced dataset 2.1.

Figure 5.6: Results of the sklearn-algorithm on dataset 2.1, showing both the results from the original training along with the results when using the optimized parameters from the grid search, which is marked with a '2'.

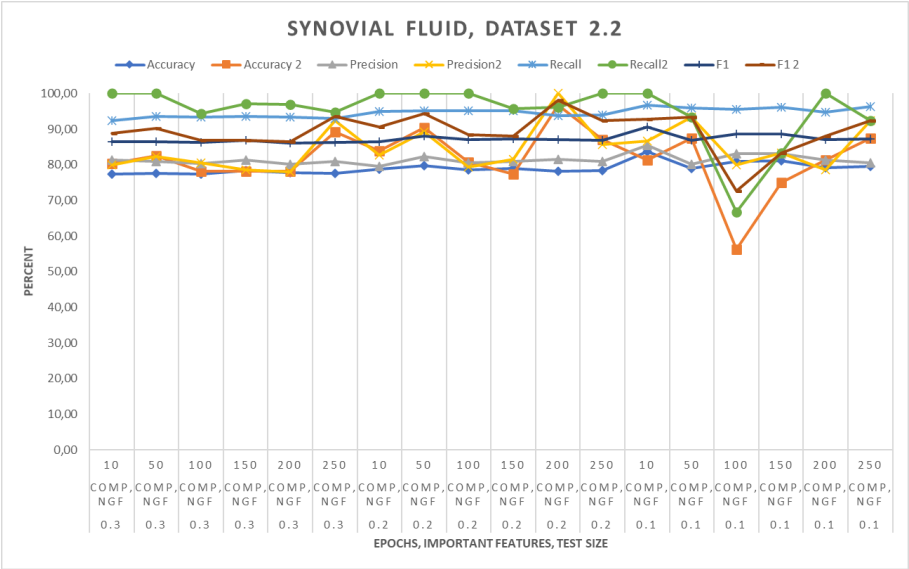
The balanced dataset provided the overall highest percentages when using a test size of 10%. Three out of six runs reached a performance of 100% on all scores after the grid search. The optimal settings were given with a mean max depth of 6, max features of 4 and mean forest size of 372.22. Its training gave a mean accuracy of 93.76%. When using the best estimators, the accuracy improved with 1.75%, reaching a mean accuracy of 95.40%.

Figure 5.6 shows an overall nice plot of high percentages. For example, the second precision is set to 100% on all but one run, followed by high recall and f_1-score. When using the test size of 10%, a total of four runs reached a perfect score of 100% on accuracy, precision, recall and f_1-score with the optimized parameters. The most important feature for the original dataset was the age, whereas for the balanced dataset it was considered the joint.

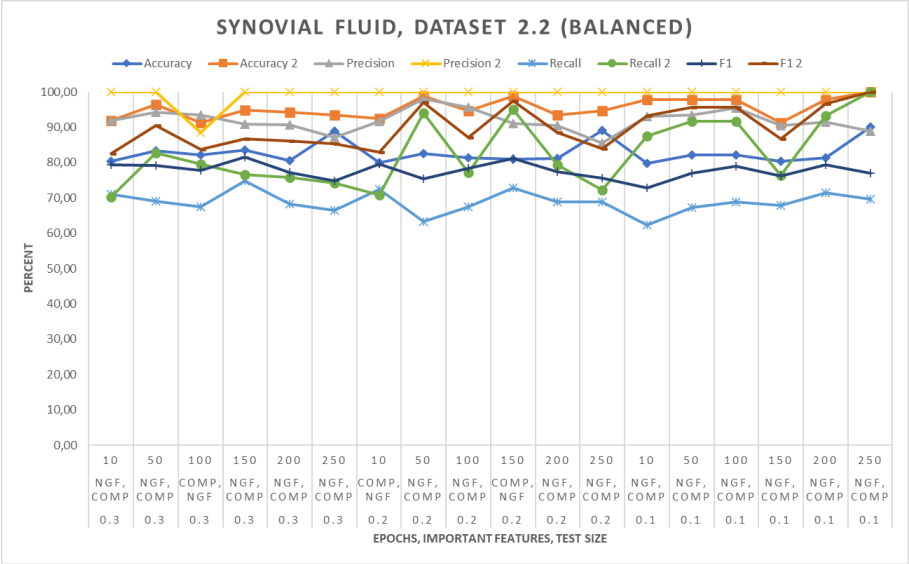
Synovial fluid dataset 2.2

The best settings for dataset 2.2 used a mean max depth of 6.33 and a forest size of 311.11. The accuracy reached 79.00% on the original dataset and increased to 81.77% after grid search.

The balanced dataset reached the best results yet when using the mean max depth of 8, max features of 4 and mean forest size of 413.89. Its training gave a mean accuracy of 82.76%, and when using the best estimators the accuracy improved greatly with 15.29% reaching a mean accuracy of 95.42%. The grid search parameters outmatched the original training in almost every score, as shown in both figure 5.7 and Appendix 1. As for the most important feature, COMP1 was selected for the unbalanced dataset and the NGF-value for the balanced dataset.



(a) Original dataset 2.2.

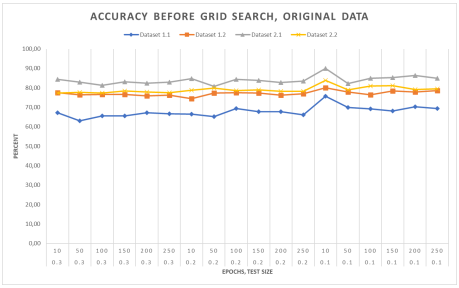


(b) Balanced dataset 2.2.

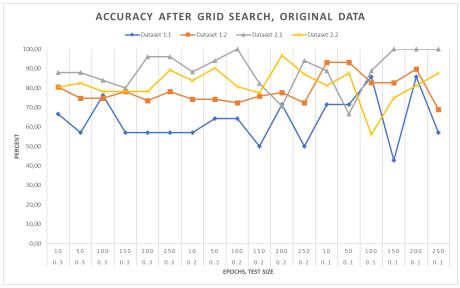
Figure 5.7: Results of the sklearn-algorithm on dataset 2.2, showing both the results from the original training along with the results when using the optimized parameters from the grid search, which is marked with a '2'.

Comparisons based on performance measurements
To make it easier to compare the performance on each dataset, they were also compared based on the results obtained from the different performance measurements.

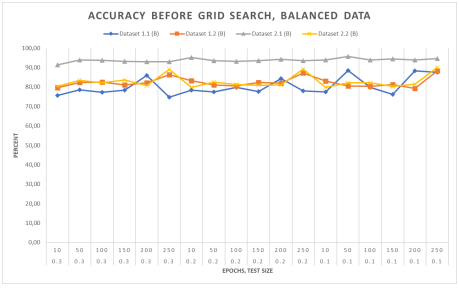
The accuracy comparison is shown in figure 5.8. Overall, dataset 2.1 reaches the highest accuracy both before and after the grid search, and dataset 1.1 yields the most poor performance. The results also gain some irregularity and fluctuations after the grid search on both the original and the balanced dataset.



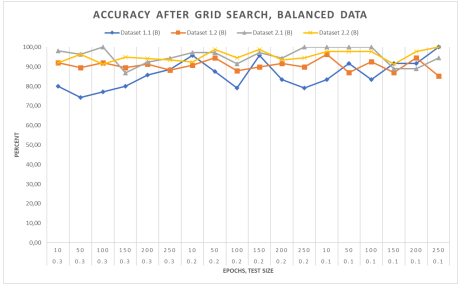
(a) Original, before.



(b) Original, after.



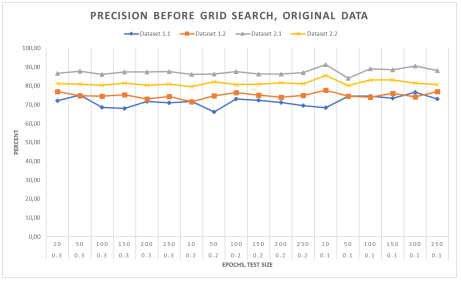
(c) Balanced, before.



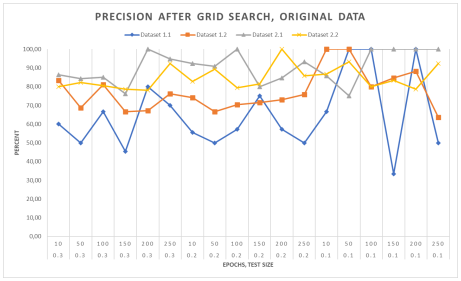
(d) Balanced, after.

Figure 5.8: A comparison of the accuracy between the original (figure a and b) and balanced (figure c and d) datasets before and after the grid search.

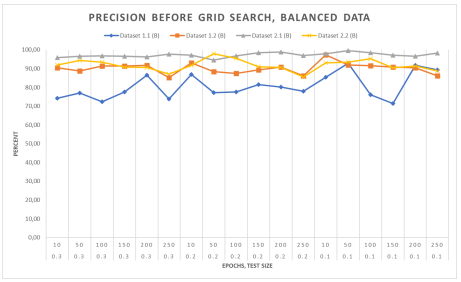
All results for the precision measurements are shown in figure 5.9. Similar to with the accuracy some fluctuations have been added after the grid search, making the graphs very irregular in figure 5.9b and 5.9d. The similarities continues with both the best and worst performance of the precision, where dataset 2.1 is continuously at the top of the graphs and dataset 1.1 at the bottom.



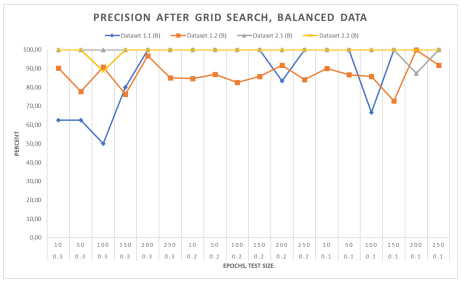
(a) Original, before.



(b) Original, after.



(c) Balanced, before.

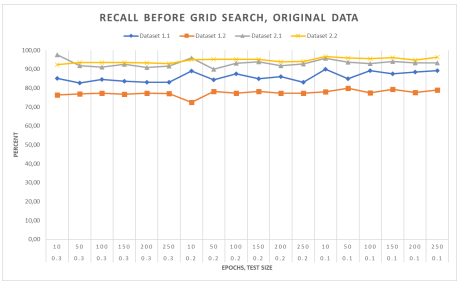


(d) Balanced, after.

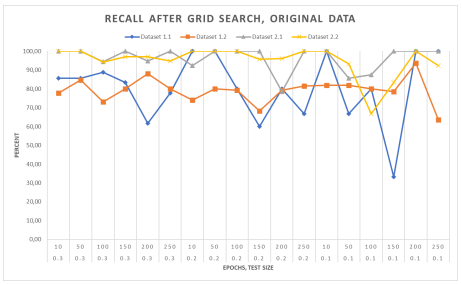
Figure 5.9: A comparison of the precision between the original (figure a and b) and balanced (figure c and d) datasets before and after the grid search.

The recall measurements in figure 5.10 resemblance the precision graphs a lot, and dataset 2.1 still outperforms the others with dataset 2.2 not far behind. Since precision and recall is somewhat dependent on the other, the fluctuations in the precision can be found in the recall graphs as well. This also corresponds to the fact that the original datasets have a overall high recall and a somewhat lower precision, and that the balanced datasets have higher precision but lower recall.

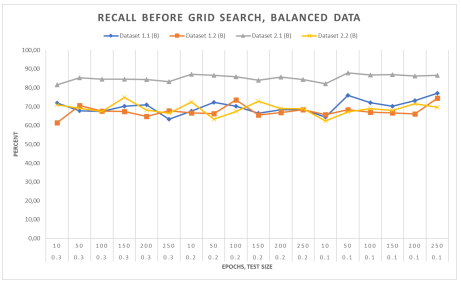
To no surprise the F_1-score follows the graphs of recall and precision and provides a kind of mean between the two, as shown in figure 5.11. Dataset 2.1 remains at the top of all the graphs and dataset 1.1 at the bottom. The fluctuations is transferred into the F_1-score as well. On the original dataset before and after grid search, the datasets of synovial fluid is clearly separated with the serum datasets, where the synovial fluid received a better performance.



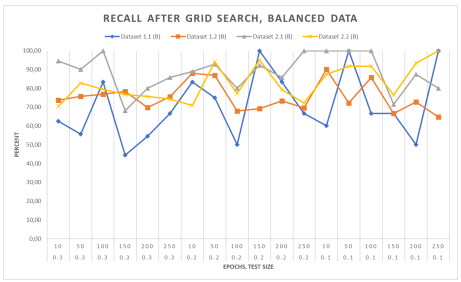
(a) Original, before.



(b) Original, after.

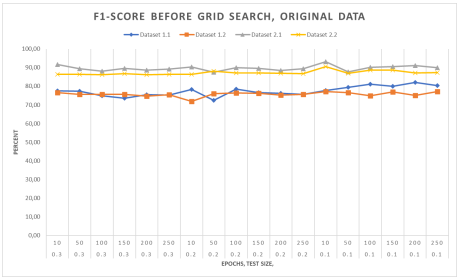


(c) Balanced, before.

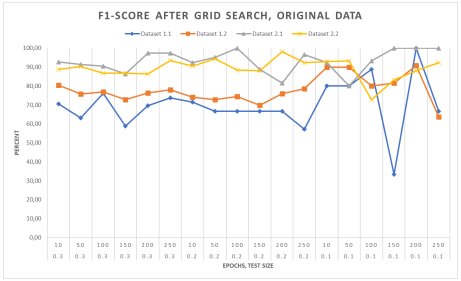


(d) Balanced, after.

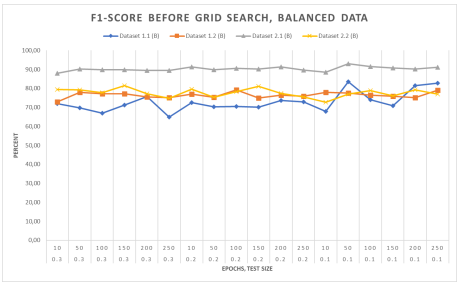
Figure 5.10: A comparison of the recall between the original (figure a and b) and balanced (figure c and d) datasets before and after the grid search.



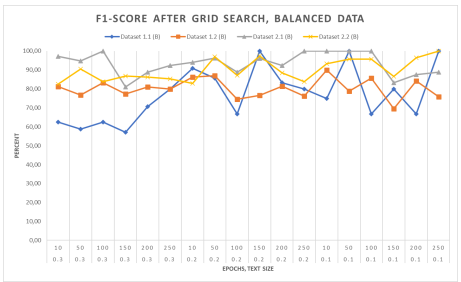
(a) Original, before.



(b) Original, after.



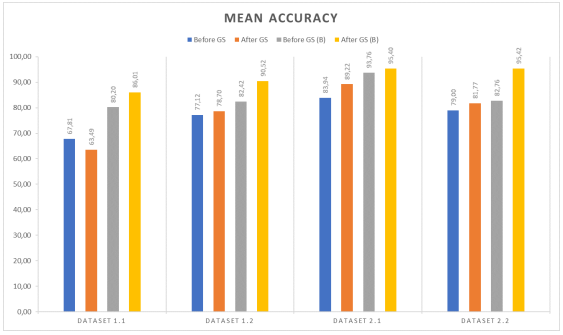
(c) Balanced, before.



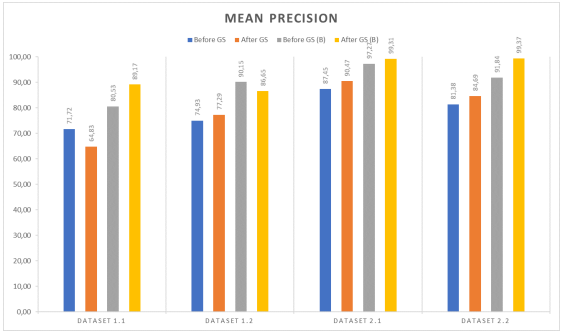
(d) Balanced, after.

Figure 5.11: A comparison of the F_1-score between the original (figure a and b) and balanced (figure c and d) datasets before and after the grid search.

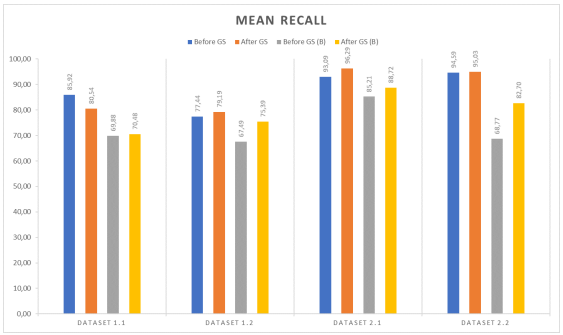
To make things even more clear, the performance measurements were also compared in a bar plot shown in figure 5.12. In this comparison the mean value of all runs are used and shown with the different datasets grouped together. This shows a pretty similar outcome for all datasets, where the synovial fluid datasets 2.1 and 2.2 performs slightly better. In these graphs it is clear that the accuracy improves with the grid search in all but one dataset.



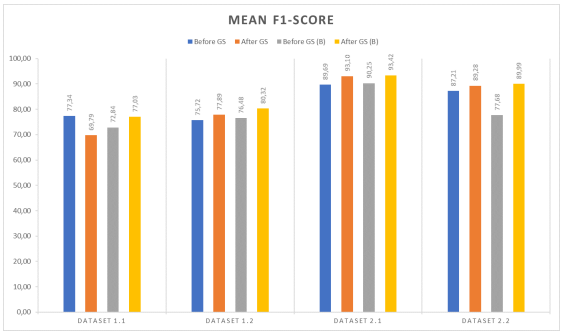
(a) Accuracy.



(b) Precision.



(c) Recall.



(d) F_1-score.

Figure 5.12: A comparison between all datasets using the mean of the measurements, where GS stands for Grid Search.

5.2.3 Random forest without machine learning libraries

The random forest algorithm was evaluated on all datasets but with fewer test runs based on the received results from the sklearn algorithm. The size of the bootstrapped training set was set to 50, and the forest size to 10. The details of these results can be found in Appendix 2.

Serum dataset 1.1
As shown in figure 5.13, the accuracy received for this dataset is below 90% for all test sizes, both for the original dataset as well as the balanced. The precision and recall is slightly higher but still not high enough, resulting in a quite low F_1-score.

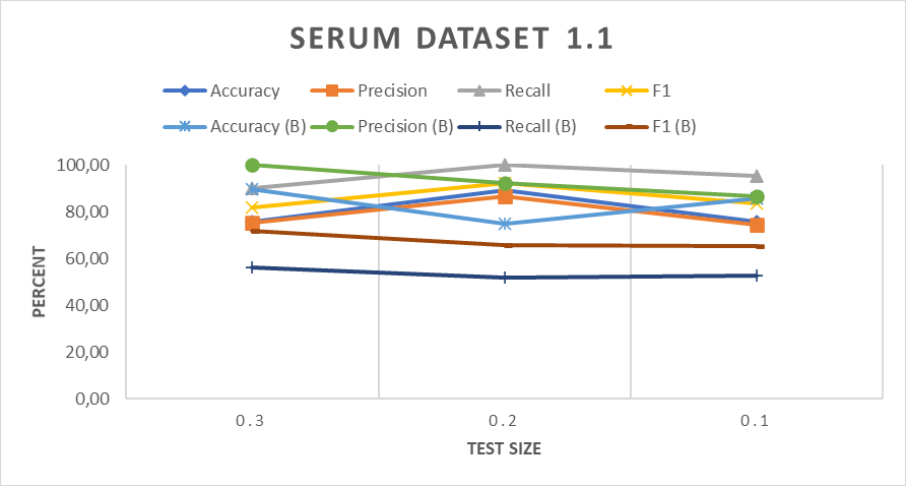


Figure 5.13: An overview of the results for dataset 1.1.

Serum dataset 1.2
The results shown in figure 5.14 are similar to dataset 1.1, but the graph is very smooth. The best results are obtained with the test size of 10, but the performance is still quite poor compared to the sklearn algorithm.

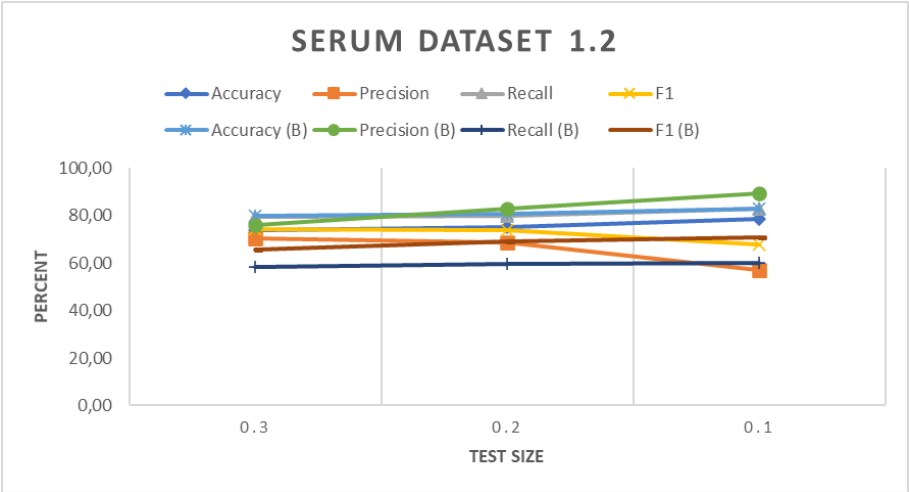


Figure 5.14: An overview of the results for dataset 1.2.

Synovial fluid dataset 2.1

As seen in figure 5.15 and similar to the results obtained from the sklearn algorithm, the performance is improved for the synovial fluid dataset 2.1, specifically when it has been balanced. The results are above or around 90% for all test sizes.

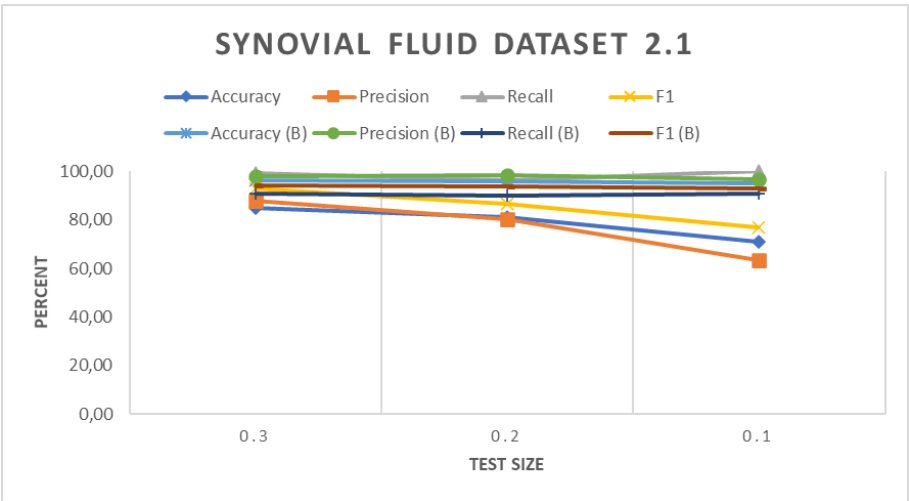


Figure 5.15: An overview of the results for dataset 2.1.

Synovial fluid dataset 2.2

The final dataset is shown in figure 5.16, which shows a wide spread of results. The performance of the algorithm on the balanced dataset has drastically decreased with a recall and F_1-score below 60%, while the accuracy is nowhere near the top scores close to 100%.

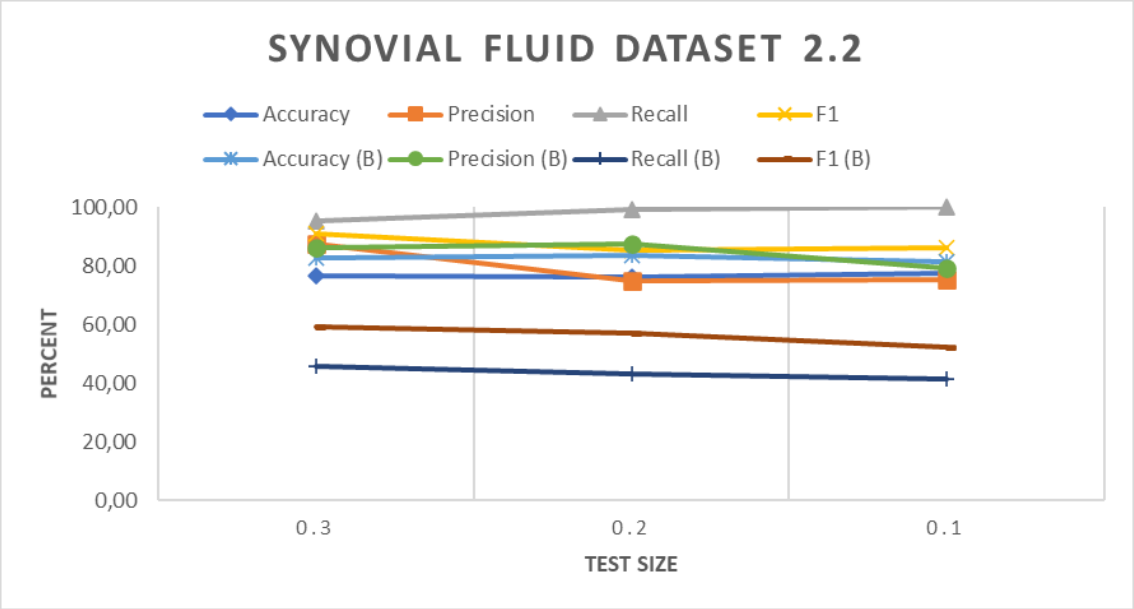
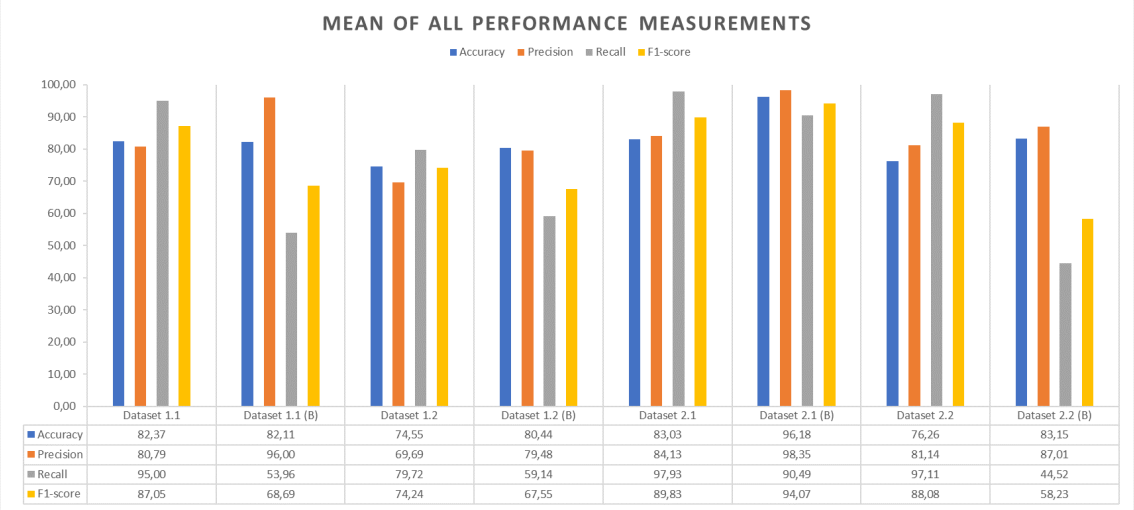


Figure 5.16: An overview of the results for dataset 2.2.

Comparisons based on performance measurements
Figure 5.17 shows the mean of the results grouped together by the different datasets, where the balanced datasets have been separated from the original. The overall best results are obtained from dataset 2.1, which was somewhat expected. However, the bar plot clearly shows the surprisingly low recall for the balanced dataset 2.2.



	Dataset 1.1	Dataset 1.1 (B)	Dataset 1.2	Dataset 1.2 (B)	Dataset 2.1	Dataset 2.1 (B)	Dataset 2.2	Dataset 2.2 (B)
Accuracy	82,37	82,11	74,55	80,44	83,03	96,18	76,26	83,15
Precision	80,79	96,00	69,69	79,48	84,13	98,35	81,14	87,01
Recall	95,00	53,96	79,72	59,14	97,93	90,49	97,11	44,52
F1-score	87,05	68,69	74,24	67,55	89,83	94,07	88,08	58,23

Figure 5.17: Mean values of all performance measurements, where (B) labels the balanced datasets.

5.2.4 Comparison between the two random forest algorithms

A comparison between the two implemented algorithms for the random forest method is shown in figure 5.18, where the mean accuracy is presented. As previous observations has shown, dataset 2.1 yields the best accuracy for all algorithms. The highest overall mean accuracy is 96.18%, which was received for the balanced dataset 2.1 using the algorithm without the sklearn library.

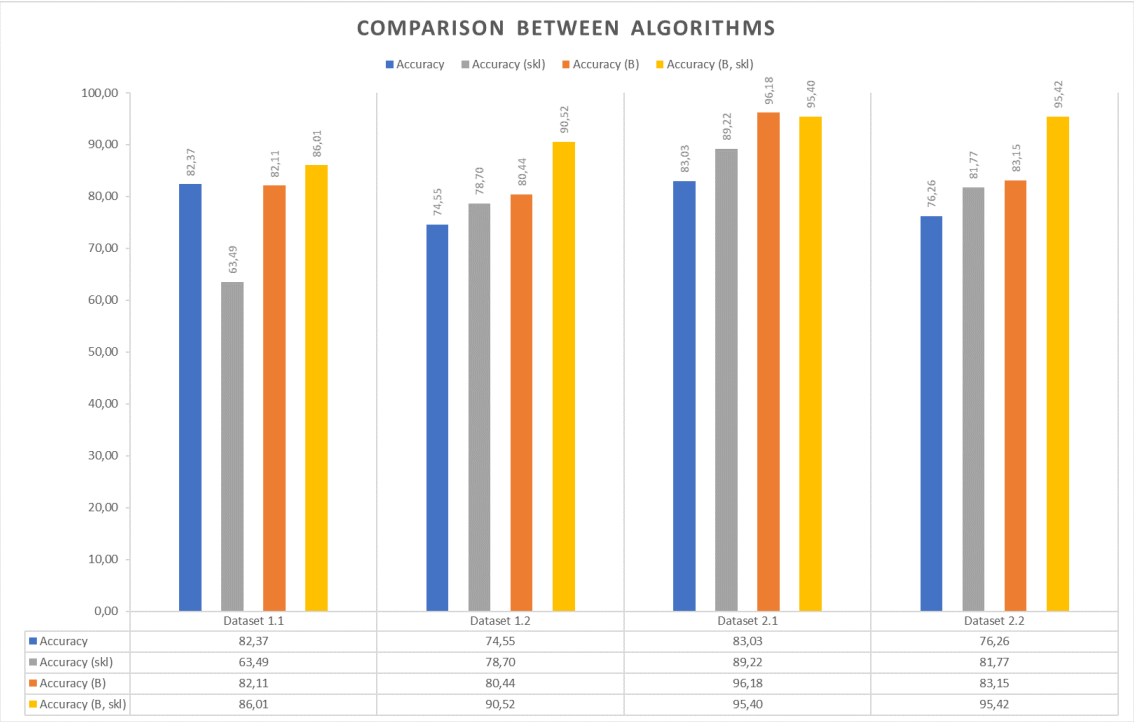


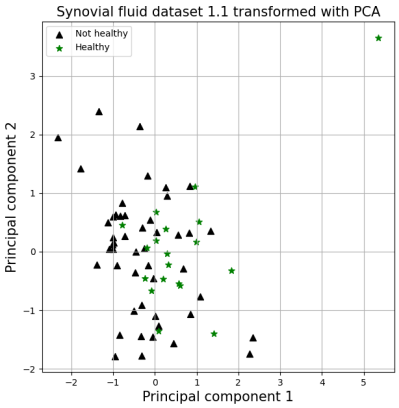
Figure 5.18: Comparison of the random forest algorithms, where (B) labels the balanced datasets and the sklearn i noted with (*skl*). The results gained after the grid search are the one used for the sklearn algorithm.

5.3 Clustering algorithm

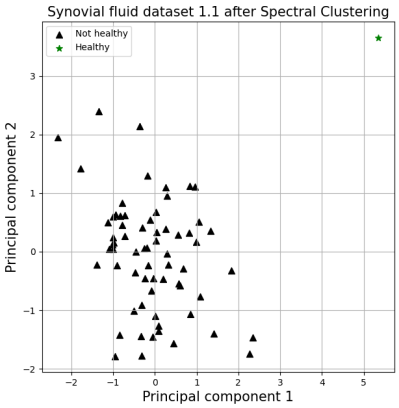
The clustering algorithm was tested on four datasets, namely dataset 1.1 and 2.1, plus the same datasets but without the NGF-value. The datasets were modified with labels of either healthy or non-healthy, where septic, OA and ev OA was considered non-healthy. Since the samples have a correct label it was possible to calculate accuracy and visualize the resulting clusters.

The results were created with a distance between 0.1 to 1.5 and the detailed result can be found in Appendix 3. In figure 5.19 the results from dataset 1.1 with and without the NGF-value are shown, and figure 5.20 shows the same except that it is based on dataset 1.2.

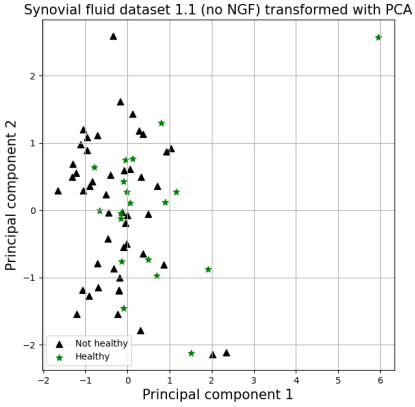
As shown in these figures, the new two dimensional data obtained with the PCA is clearly not separable in any simple way. The random samples quite far away from the other samples might also be outliers, which in that case could affect the outcome of the algorithm.



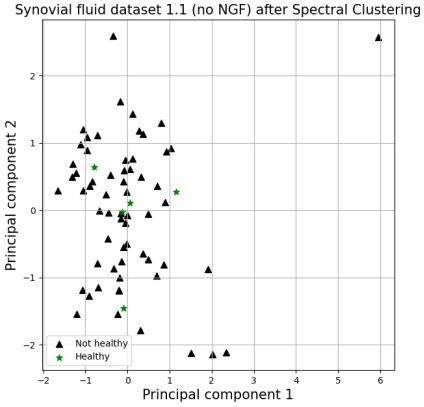
(a) Dataset 1.1 after PCA.



(b) Dataset 1.1 after spectral clustering with distance 1.1. Accuracy: 73.13%.

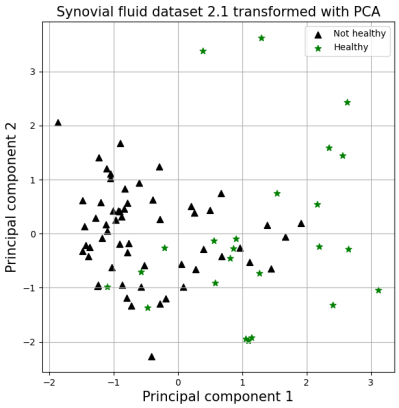


(c) Dataset 1.1 (no NGF) after PCA.

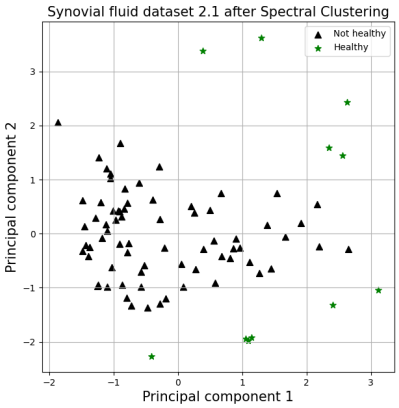


(d) Dataset 1.1 (no NGF) after spectral clustering with distance 0.1. Accuracy: 76.12%.

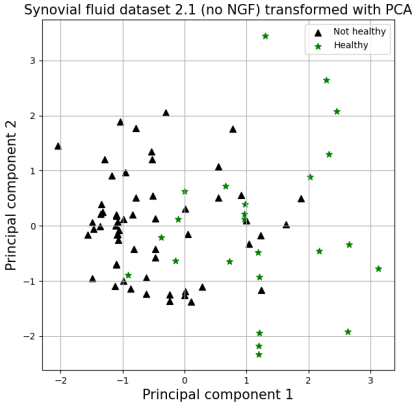
Figure 5.19: Results after PCA and Spectral clustering on dataset 1.1 with and without NGF. The distance with the highest accuracy was used.



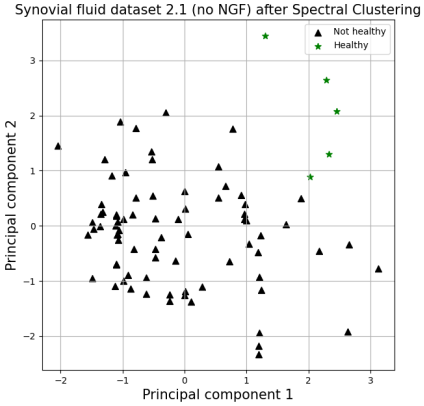
(a) Dataset 2.1 after PCA.



(b) Dataset 2.1 after spectral clustering with distance 0.8. Accuracy: 81.71%.



(c) Dataset 2.1 (no NGF) after PCA.



(d) Dataset 2.1 (no NGF) after spectral clustering with distance 1.3. Accuracy: 76.83 %.

Figure 5.20: Results after PCA and Spectral clustering on dataset 2.1 with and without NGF. The distance with the highest accuracy was used.

The datasets were not completely balanced, which means that a random guess on the majority class would give an accuracy above 50%. The value for this random guess is named the target value. The accuracy of the algorithm is presented in figure 5.21 for each distance used in the evaluation. As the figure shows, the results are almost all below 80%, which can not be considered acceptable for a decision support system.

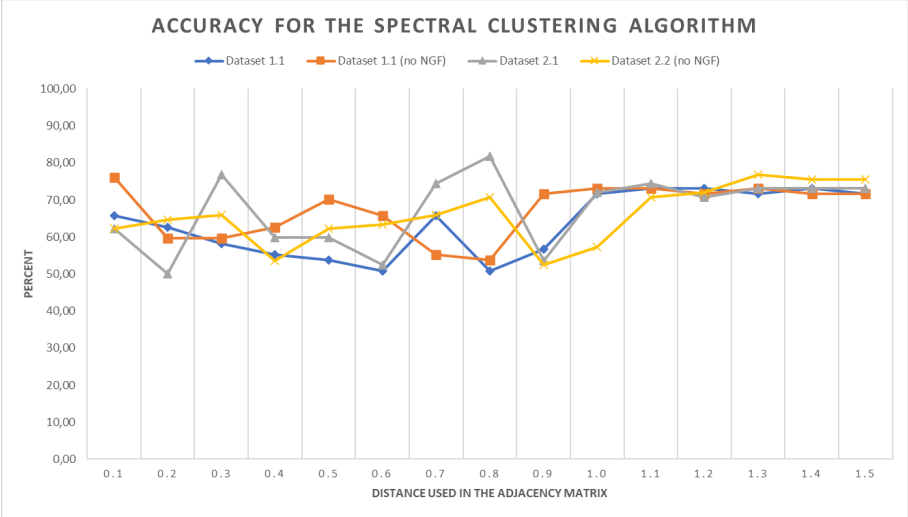


Figure 5.21: The accuracy of the spectral clustering algorithm.

The mean accuracy is shown in figure 5.22, where it is compared with the target value for each dataset along with the maximum received accuracy. Due to the somewhat irregular graph for the accuracy with a few results as low as 50%, the mean accuracy landed at between 60-70% while the maximum accuracy performed with an accuracy above the target value.

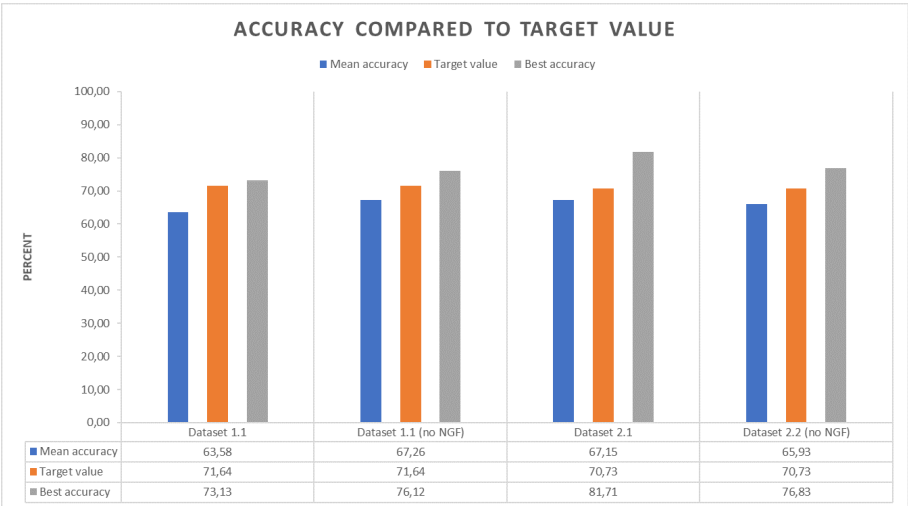


Figure 5.22: An overview of the mean accuracy compared to the target value and the maximum accuracy obtained for the most optimal distance.

6
Conclusion

This last chapter consists of a discussion part followed by the conclusion of the thesis as well as some possible future improvements.

6.1 Discussion

The discussion is divided into three parts, namely the data, the classification algorithm followed by the clustering algorithm.

6.1.1 Data

Biological data is complicated and complex in many ways. The samples of both blood and synovial fluid is collected at a specific point in time, which might affect the values for both COMP1 and NGF. Since it isn't possible to explain to the horse the importance of standing still, keeping calm or breathing slowly it is possible that the state of the horse in the moment of sampling might have an effect on the biochemical data collected. The data was also collected by several veterinarians at different clinics. Even with a protocol to follow these circumstances might lead to small differences in the samples.

Another major difficulty for the algorithm was the significant balancing issue of the final datasets. By using oversampling as described in the thesis, the algorithm risks becoming overfitted to the training data due to the amount of duplicates available. Fortunately, the random forest algorithm can handle duplicates pretty well thanks to the random subspace method, but it can nevertheless develop a large error. Say for example that there was only two breeds present among the healthy samples, and only a few healthy horses in total. This would lead to a large amount of duplicates of these horses of same breed, age and so on, which might indicate to the algorithm that the breed itself indicates a healthy sample.

The amount of samples available was also a complication to gain robust results. After reducing large original dataset of samples into datasets based on the specific features, a large amount of samples were unusable for the tasks of this thesis. The amount of data has however been growing throughout the work with the thesis thanks to researchers at SLU and Sahlgrenska, leading up to the conclusive datasets used in the final evaluation. Since the machine learning part of the research is in an early stage, this will continue to increase with time.

Out of all biomarkers found to be promising for determining the progression stage of OA in the future, only the COMP1 was available at this point. This is probably a main cause for the poor performance of the clustering algorithm, since there are parts of the disease still missing. Once they are added it might be possible to find more patterns in the data.

6.1.2 Classification algorithm

The method selection was based on the unbalanced datasets. The choice of using the random forest might have been different if the selection were to be decided on balanced datasets. However, since the accuracy reached close to or above 80%, it was considered acceptable. The reason for not investigating this further was that the final datasets were delivered quite late in the development process. The first datasets used in the selection was not complete, bur fortunately the final datasets gained similar results leading to the same conclusion.

The calculated precision was generally high for both algorithms, which is a good sign. Since the precision was calculated based on the diagnose of OA, it means that out of all samples that were labeled as OA by the algorithm, most of them had in fact OA according to the true label. This is important for a decision support system since a false diagnose of OA would put the horse out of practice. At times when the high precision was combined with a high recall the result was of course even better. The high recall implies that most of the true OA labels was actually predicted to have OA, which is a good result. At times when the high recall was combined with a low precision, this means that algorithm probably was over confident with classifying OA leading to an increase of false positives. This was most common for the unbalanced datasets.

The sklearn library was very useful and turned out be helpful even when developing and evaluating the other algorithm. The grid search function did overall improve the results, especially the accuracy. Because of this, the parameter range could be much smaller when testing the other algorithm which saved a lot of time.

Time is another aspect when developing machine learning algorithms. Thanks to the small datasets and pretty straight forward implementations, the training time was not too long. It did however take a some time to train the algorithms with the different settings, an aspect which when looking in the rear-view window should have been considered early on. When looking at the results from the sklearn algorithm, there isn't much difference in the performance. The time put on just training, where one training could be processed in around 4-8 minutes, could probably have come to better use. This time might also have been possible to decrease by using a stationary computer with a better processor, but due to COVID-19 the main parts of the thesis work had to be done on distance. Therefore, the algorithms was developed and trained on a Lenovo YOGA laptop with an Intel CORE i5 processor.

6.1.3 Clustering algorithm

The clustering algorithm was supposed to be tested on a completely different dataset from another case study, but unfortunately the data was not applicable with the task. There were too many gaps in the features to be able to provide a fair result. But since the algorithm was already developed, it was used on the already used datasets instead.

As mentioned earlier, one of the main difficulties with the clustering algorithm is the dimensions. By using PCA, the dimensions is reduced and processed into data easier to comprehend and visualize. However, in this process some value of the data might be lost. One way of adjusting this could be to introduce weights to the different features to indicate which features are more important, which could probably improve the performance.

The NGF-value was investigated more thoroughly in this task, since it is still unknown what its exact values are supposed to be in correlation to OA. This did not affect the performance of the algorithm in any significant way, which might indicate that the feature is not that important for this specific task or that its importance is equally important for both synovial fluid and serum. A large error in this assumption is that the data was modified into only healthy and non-healthy samples, when they in fact had several different diagnoses.

6.2 Conclusion

The main aim for this thesis was to investigate the possibility of diagnosing OA with a machine learning approach in order to use it in a decision support system. The results showing a mean accuracy above 90% combined with a high precision implies that it is indeed possible, and that the algorithms could be further developed to improve its performance even more once more data is available. Since the results on this first attempt of classifying OA were relatively robust, it does seem like a promising method for diagnosing the disease in the future.

The algorithms worked quite well on all datasets, but was given an overall higher accuracy when used on the synovial fluid datasets. Especially nice results were obtained from the dataset with more features than only COMP1 and NGF, and where only three diagnoses were available. This indicates a possible importance of which joint the sample is from, the breed of the horse as well as its age. However, further analysis will be needed on improved and balanced datasets in order to fully understand the complexity of the chosen features and how it affects the classification of OA.

6.3 Future Improvements

As mentioned earlier, the datasets could be improved in several ways. Most importantly, it would be beneficial to gather more data to be able to train on larger training sets. Furthermore, these new datasets would need to be balanced based on all its included features. Meaning that they should include a similar amount of samples from each breed, similar amount of samples diagnosed as healthy versus OA or septic samples, similar amount of samples from each joint, and so on.

It could also be interesting to use even more features in future algorithms, since the biological data has so many possible parameters. Maybe it would be possible to use more parameters from the blood sample to improve these results, or adding more features concerning the external parts - such as grade of lameness, the weight of the horse or some variable connected to the amount of intense training the horse has recently been exposed to.

There are of course also several more possible algorithms available for this task. Based on the method selection done in this thesis, the most interesting algorithms to implement in the future would be Support Vector Machines or AdaBoost.

Bibliography

[1] Eriksson L. (2013). *Multi- and Megavariate Data Analysis: Basic Principles and Applications*. Malmö MKS Umetrics. https://blog.umetrics.com/what-is-principal-component-analysis-pca-and-how-it-is-used, [Accessed: 2020-04-02].

[2] Skiöldebrand E et al. *Cartilage oligomeric matrix protein neoepitope in the synovial fluid of horses with acute lameness: A new biomarker for the early stages of osteoarthritis*. Equine Veterinary Journal, 2017;49(5):662-667.

[3] Schlueter A and Orth M. *Equine osteoarthritis: a brief review of the disease and its causes*. Equine and Comparative Exercise Physiology, 2003;1(4); 221 – 231.

[4] Betts G and others. (2020). *Anatomy and Physiology*. OpenStax resource, Rice University. https://openstax.org/books/anatomy-and-physiology/pages/9-4-synovial-joints, [Accessed: 2020-04-02].

[5] Heinegård D and Saxne T. *The role of the cartilage matrix in osteoarthritis.* Nat. Rev. Theumatol, 2010;7:50-56.

[6] Ekman S et al. *Effect of circadian rhythm, age, training and acute lameness on serum concetrations of cartilage oligomeric matrix protein (COMP) neoepitopes in horses.* Equine Veterinary Journal, 2019;51:674-680.

[7] Strimbu K and Tavel J.A. *What are Biomarkers?* Curr Opin HIV AIDS, 2010;5(6):463-466.

[8] Ekman A et al. *Effect of circadian rhythm, age, training and acute lameness on serum concentrations of cartilage oligomeric matrix protein (COMP) neoepitope in horses.* Equine Veterinary Journal, 2019;51(5):674-680.

[9] Kendall A and Skiöldebrand E. (2020). *Nerve Growth Factor som biomarkör för smärta och inflammation.* SLU, https://www.slu.se/fakulteter/vh/forskning/forskningsprojekt/hast/nerve-growth-factor-som-biomarkor-for-smarta-och-inflammation/., 2020. [Accessed 2020-05-03].

[10] Tseng S et al. *Cartilage Oligomeric Matrix Protein (COMP): A Biomarker of Arthritis*. Biomark Insights, 2009;4; 33 – 44.

[11] Shang X et al. *Mechanism and therapeutic effectiveness of nerve growth factor in osteoarthritis pain*. Therapeutics and Clinical Risk Management, 2017;13; 951 – 956.

[12] Life Science Group. (2017). *ELISA Basics Guide.* Bio-Rad Laboratories, Inc. https://www.bio-rad-antibodies.com/elisa-procedure.html. [Accessed: 2020-04-10].

[13] Murphy K. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press. https://doc.lagout.org/science/Artificial%20Intelligence/Machine%20learning/Machine%20Learning_%20A%20Probabilistic%20Perspective%20%5BMurphy%202012-08-24%5D.pdf. [Accessed: 2020-04-10].

[14] Criminisi A et al. *Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning.* Foundations and Trends® in Computer Graphics and Vision., 2012; 7(2–3):81–227.

[15] Fleshman W. (2019). *Spectral Clustering, Foundation and Application.* Towards Data Science. https://towardsdatascience.com/spectral-clustering-aba2640c0d5b. [Accessed: 2020-04-10].

[16] Aoullay A. (2018). *Spectral Clustering for beginners.* Towards Data Science. https://towardsdatascience.com/spectral-clustering-for-beginners-d08b7d25b4d8. [Accessed: 2020-05-02].

[17] Belle A and others. (2013). *Biomedical Informatics for Computer-Aided Decision Support Systems: A Survey.* The Scientific World Journal. https://www.hindawi.com/journals/tswj/2013/769639/, [Accessed: 2020-04-02].

A

Appendix 1: results from the algorithm using Sklearn

The following pages include the results obtained from the classification algorithm using the sklearn library. It is a PDF file extracted from the excel data file.

Green background in the second columns of the performance measurements means that the result was improved after the grid search. Red background means that the the algorithm reached a higher value without grid search.

At the bottom of each column is a calculated mean.

RESULTS, with sklearn library

Serum_data

ORIGINAL

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision2**	**Recall**	**Recall2**	**F1**	**F1 2**
0.3	age, comp, ngf	10	8	100	67,14	66,67	72,11	60,00	85,20	85,71	77,59	70,59
0.3	age, comp, ngf	50	2	900	63,05	57,14	75,14	50,00	82,65	85,71	77,25	63,16
0.3	age, comp, breed	100	4	100	65,62	76,19	68,59	66,67	84,63	88,89	74,85	76,19
0.3	age, comp, breed	150	4	100	65,65	57,14	67,93	45,45	83,61	83,33	73,61	58,82
0.3	age, comp, ngf	200	8	100	67,21	57,14	71,77	80,00	83,01	61,64	75,48	69,57
0.3	age, comp, breed	250	4	500	66,65	57,14	71,00	70,00	83,11	77,78	75,24	73,68
0.2	age, comp, ngf	10	8	900	66,43	57,14	71,73	55,56	89,05	100,00	78,35	71,43
0.2	comp, age, breed	50	6	700	65,29	64,29	66,24	50,00	84,45	100,00	72,46	66,67
0.2	age, comp, breed	100	4	300	69,36	64,29	73,06	57,14	87,46	80,00	78,39	66,67
0.2	age, comp, ngf	150	4	100	67,71	50,00	72,38	75,00	84,95	60,00	76,56	66,67
0.2	age, comp, ngf	200	4	100	67,68	71,43	71,10	57,14	86,06	80,00	76,19	66,67
0.2	age, comp, breed	250	4	100	66,06	50,00	69,43	50,00	83,12	66,67	75,66	57,14
0.1	age, comp, breed	10	8	900	75,74	71,43	68,33	66,67	90,00	100,00	77,68	80,00
0.1	comp, age, breed	50	4	300	70,00	71,43	74,60	100,00	84,83	66,67	79,39	80,00
0.1	comp, age, ngf	100	4	300	69,14	85,71	74,50	100,00	89,13	80,00	81,16	88,89
0.1	age, comp, breed	150	8	100	68,10	42,86	73,40	33,30	87,61	33,33	79,88	33,33
0.1	comp, age, breed	200	4	700	70,29	85,71	76,55	100,00	88,46	100,00	82,08	100,00
0.1	age, comp, breed	250	8	100	69,37	57,14	73,10	50,00	89,14	100,00	80,33	66,67
	age, comp, breed		**5,33**	**355,56**	**67,81**	**63,49**	**71,72**	**64,83**	**85,92**	**80,54**	**77,34**	**69,79**
					-6,36%	-4,31						

BALANCED

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision 2**	**Recall**	**Recall 2**	**F1**	**F1 2**
0.3	age, comp, breed	10	8	100	75,71	80,00	74,26	62,50	71,90	62,50	71,95	62,50
0.3	age, comp, ngf	50	4	100	78,46	74,29	76,97	62,50	67,72	55,56	69,81	58,82
0.3	comp, age, breed	100	4	100	77,34	77,14	72,44	50,00	67,47	83,33	67,00	62,50
0.3	age, comp, breed	150	6	500	78,32	80,00	77,58	80,00	70,23	44,44	71,15	57,14
0.3	comp, age, breed	200	6	700	85,89	85,71	86,51	100,00	71,02	54,55	75,68	70,59
0.3	comp, age, ngf	250	6	300	74,73	88,57	73,83	100,00	63,36	66,67	64,84	80,00
0.2	comp, age, breed	10	8	100	78,33	95,83	86,83	100,00	67,50	83,33	72,54	90,91
0.2	age, comp, breed	50	8	300	77,41	87,50	77,21	100,00	72,27	75,00	70,34	85,71
0.2	comp, age, breed	100	6	100	79,83	79,17	77,49	100,00	70,13	50,00	70,43	66,67
0.2	comp, age, breed	150	8	500	77,56	95,83	81,47	100,00	66,52	100,00	70,15	100,00
0.2	comp, age, ngf	200	6	900	84,27	83,33	80,12	83,33	68,24	83,33	73,70	83,33
0.2	comp, age, ngf	250	6	300	78,00	79,17	78,01	100,00	68,59	66,62	73,00	80,00
0.1	comp, age, breed	10	6	100	77,50	83,33	85,33	100,00	64,36	60,00	67,94	75,00
0.1	age, comp, ngf	50	8	700	88,50	91,67	92,89	100,00	75,89	100,00	83,53	100,00
0.1	age, comp, breed	100	8	500	79,75	83,33	76,15	66,67	72,00	66,67	74,02	66,67
0.1	age, comp, ngf	150	8	300	76,11	91,67	71,36	100,00	70,26	66,67	70,81	80,00
0.1	age, comp, ngf	200	6	300	88,37	91,67	91,75	100,00	73,23	50,00	81,45	66,67
0.1	comp, age, ngf	250	6	100	87,50	100,00	89,37	100,00	77,06	100,00	82,76	100,00
	comp, age,breed		**6,56**	**333,33**	**80,20**	**86,01**	**80,53**	**89,17**	**69,88**	**70,48**	**72,84**	**77,03**
					7,25%	5,81						

Led_data

ORIGINAL

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision2**	**Recall**	**Recall2**	**F1**	**F1 2**
0.3	joint, age, ngf	10	2	700	84,40	88,00	86,68	86,36	97,52	100,00	91,61	92,68
0.3	joint, age, ngf	50	6	300	82,96	88,00	87,72	84,21	91,75	100,00	89,45	91,43
0.3	joint, age, comp	100	6	900	81,36	84,00	86,12	85,00	91,05	94,44	88,10	90,47
0.3	joint, age, breed	150	4	100	83,15	80,00	87,42	76,19	92,61	100,00	89,63	86,49
0.3	age. Joint, ngf	200	4	300	82,36	96,00	87,39	100,00	90,84	94,74	88,65	97,30
0.3	age, breed, joint	250	6	100	82,99	96,00	87,65	94,74	91,60	100,00	89,22	97,30
0.2	age, comp, ngf	10	6	300	84,71	88,24	86,05	92,31	95,87	92,31	90,37	92,31
0.2	age, joint, breed	50	4	300	80,82	94,12	86,33	90,91	89,87	100,00	87,57	95,24
0.2	age, joint, breed	100	4	100	84,41	100,00	87,63	100,00	93,12	100,00	89,90	100,00
0.2	age, joint, ngf	150	6	100	83,88	82,35	86,32	80,00	93,86	100,00	89,52	88,89
0.2	age, joint, ngf	200	4	1000	82,71	70,59	86,27	84,62	91,86	78,57	88,45	81,48
0.2	age, joint, ngf	250	4	500	83,39	94,12	87,10	93,33	92,82	100,00	89,32	96,55
0.1	age, joint, comp	10	4	1000	90,00	88,89	91,30	85,71	95,64	100,00	93,23	92,31
0.1	joint, age, breed	50	4	300	82,22	66,67	84,05	75,00	93,62	85,71	87,63	80,00
0.1	age, joint, breed	100	6	700	84,89	88,89	88,99	100,00	92,99	87,50	90,11	93,33
0.1	joint, age, ngf	150	8	100	85,33	100,00	88,48	100,00	93,97	100,00	90,52	100,00
0.1	age, joint, ngf	200	6	500	86,44	100,00	90,47	100,00	93,22	100,00	91,06	100,00
0.1	age, joint, breed	250	6	100	84,84	100,00	88,11	100,00	93,33	100,00	89,99	100,00
	age, joint, ngf		**5,00**	**411,11**	**83,94**	**89,22**	**87,45**	**90,47**	**93,09**	**96,29**	**89,69**	**93,10**
					6,29%	5,28						

BALANCED

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision 2**	**Recall**	**Recall 2**	**F1**	**F1 2**
0.3	joint, breed, age	10	8	700	91,32	98,11	95,86	100,00	81,64	94,44	87,92	97,14
0.3	age, joint, breed	50	8	100	93,92	96,23	96,54	100,00	85,37	90,00	90,24	94,74
0.3	joint, age, breed	100	4	300	93,74	100,00	96,75	100,00	84,55	100,00	89,80	100,00
0.3	joint, age, breed	150	6	500	93,25	86,79	96,60	100,00	84,58	68,18	89,82	81,08
0.3	joint, breed, ngf	200	6	100	92,94	92,45	96,29	100,00	84,32	80,00	89,46	88,89
0.3	joint, age, ngf	250	6	100	92,94	94,34	97,71	100,00	83,21	85,71	89,51	92,31
0.2	joint, age, ngf	10	6	100	95,14	97,14	97,18	100,00	87,22	88,89	91,24	94,12
0.2	joint, age, breed	50	8	100	93,49	97,14	94,61	100,00	86,56	92,86	89,84	96,30
0.2	joint, breed, age	100	4	700	93,23	91,43	96,78	100,00	85,84	80,00	90,53	88,89
0.2	joint, comp, age	150	4	500	93,60	97,14	98,47	100,00	84,07	92,31	90,14	96,00
0.2	age, joint, comp	200	8	100	94,30	94,29	98,82	100,00	85,59	85,71	91,24	92,31
0.2	joint, breed, ngf	250	6	100	93,55	100,00	96,98	100,00	84,28	100,00	89,74	100,00
0.1	joint, age, breed	10	6	300	93,88	100,00	98,00	100,00	82,20	100,00	88,48	100,00
0.1	joint, breed, age	50	6	900	95,67	100,00	99,67	100,00	87,90	100,00	92,92	100,00
0.1	joint, breed, comp	100	6	700	93,89	100,00	98,47	100,00	86,75	100,00	91,49	100,00
0.1	joint, ngf, comp	150	8	100	94,37	88,89	97,10	100,00	86,98	71,43	90,78	83,33
0.1	joint, age, breed	200	4	300	93,94	88,89	96,67	87,50	86,15	87,50	90,16	87,50
0.1	joint, age, breed	250	4	1000	94,53	94,44	98,34	100,00	86,61	80,00	91,21	88,89
	joint, age, breed		**6,00**	**372,22**	**93,76**	**95,40**	**97,27**	**99,31**	**85,21**	**88,72**	**90,25**	**93,42**
					1,75%	1,64						

Serum_data2

ORIGINAL

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision2**	**Recall**	**Recall2**	**F1**	**F1 2**
0.3	comp, ngf	10	4	100	77,59	80,46	77,02	83,33	76,37	77,78	76,56	80,46
0.3	ngf, comp	50	6	100	76,39	74,71	74,73	68,75	76,86	84,62	75,63	75,86
0.3	ngf, comp	100	6	500	76,63	74,71	74,55	81,08	77,35	73,17	75,59	76,92
0.3	comp, ngf	150	8	100	76,62	78,16	75,35	66,67	76,64	80,00	75,73	72,73
0.3	ngf, comp	200	4	100	75,86	73,56	72,99	67,27	77,27	88,10	74,79	76,29
0.3	comp, ngf	250	8	100	76,23	78,16	74,26	76,19	77,19	80,00	75,37	78,05
0.2	ngf, comp	10	4	500	74,48	74,14	71,55	74,07	72,49	74,07	71,79	74,07
0.2	ngf, comp	50	6	300	77,31	74,14	74,71	66,67	78,16	80,00	76,00	72,73
0.2	ngf, comp	100	8	1000	77,55	72,41	76,40	70,37	77,33	79,17	76,45	74,51
0.2	ngf, comp	150	6	100	77,33	75,86	75,04	71,43	78,20	68,18	76,18	69,77
0.2	comp, ngf	200	4	700	76,23	77,59	73,94	73,08	77,30	79,17	75,22	76,00
0.2	ngf, comp	250	6	500	76,99	72,41	74,97	75,86	77,26	81,48	75,71	78,57
0.1	ngf, comp	10	6	300	80,00	93,10	77,65	100,00	77,99	81,82	77,23	90,00
0.1	comp, ngf	50	6	700	77,86	93,10	74,60	100,00	79,86	81,82	76,58	90,00
0.1	ngf, comp	100	6	300	76,38	82,76	73,96	80,00	77,51	80,00	74,80	80,00
0.1	comp, ngf	150	4	500	78,34	82,76	75,96	84,62	79,36	78,57	76,93	81,48
0.1	ngf, comp	200	6	900	77,79	89,66	74,17	88,24	77,69	93,75	75,13	90,91
0.1	ngf, comp	250	4	500	78,61	68,97	76,91	63,64	79,04	63,64	77,18	63,64
	ngf, comp		**5,67**	**405,56**	**77,12**	**78,70**	**74,93**	**77,29**	**77,44**	**79,19**	**75,72**	**77,89**
					2,05%	1,58						

BALANCED

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision 2**	**Recall**	**Recall 2**	**F1**	**F1 2**
0.3	comp, ngf	10	8	500	79,62	91,93	90,54	90,32	61,48	73,68	72,87	81,16
0.3	comp, ngf	50	8	300	82,32	89,44	88,70	77,78	70,53	75,68	77,89	76,71
0.3	ngf, comp	100	8	500	82,53	91,93	91,39	90,91	67,50	76,92	77,28	83,33
0.3	comp, ngf	150	8	100	81,14	89,44	91,41	76,32	67,37	78,38	77,17	77,33
0.3	comp, ngf	200	8	300	82,21	91,30	91,76	96,77	64,74	69,77	75,56	81,08
0.3	comp, ngf	250	8	900	86,35	88,20	85,24	85,00	67,75	75,56	75,14	80,00
0.2	comp, ngf	10	8	100	83,33	90,74	92,99	84,62	66,60	88,00	77,07	86,27
0.2	comp, ngf	50	8	100	81,02	94,44	88,35	86,96	66,34	86,96	75,29	86,96
0.2	comp, ngf	100	8	500	80,62	87,96	87,49	82,61	73,61	67,86	79,23	74,51
0.2	comp, ngf	150	8	1000	82,26	89,81	89,43	85,71	65,48	69,23	75,02	76,60
0.2	comp, ngf	200	8	300	82,05	91,74	90,77	91,67	66,78	73,33	76,36	81,48
0.2	comp, ngf	250	8	500	87,23	89,81	86,16	84,21	68,41	69,57	75,84	76,19
0.1	comp, ngf	10	8	300	83,15	96,30	97,33	90,00	65,79	90,00	77,96	90,00
0.1	comp, ngf	50	8	900	80,52	87,04	92,03	86,67	68,32	72,22	77,50	78,79
0.1	comp, ngf	100	8	300	80,40	92,59	91,53	85,71	67,07	85,71	76,37	85,71
0.1	comp, ngf	150	8	1000	81,22	87,04	90,83	72,73	66,61	66,67	75,90	69,57
0.1	comp, ngf	200	8	300	79,30	94,44	90,48	100,00	66,02	72,73	75,23	84,21
0.1	comp, ngf	250	8	100	88,24	85,19	86,18	91,67	74,43	64,71	79,03	75,86
	comp, ngf		**8,00**	**444,44**	**82,42**	**90,52**	**90,15**	**86,65**	**67,49**	**75,39**	**76,48**	**80,32**
					9,83%	8,10						

Led_data2

ORIGINAL

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision2**	**Recall**	**Recall2**	**F1**	**F1 2**
0.3	comp, ngf	10	4	300	77,39	80,43	81,30	80,00	92,30	100,00	86,39	88,89
0.3	comp, ngf	50	4	300	77,61	82,61	80,86	82,22	93,51	100,00	86,42	90,24
0.3	comp, ngf	100	6	100	77,37	78,26	80,36	80,49	93,41	94,29	86,19	86,84
0.3	comp, ngf	150	8	100	78,32	78,26	81,35	78,57	93,52	97,06	86,84	86,84
0.3	comp, ngf	200	8	500	77,80	78,26	80,22	78,05	93,28	96,97	86,09	86,49
0.3	comp, ngf	250	8	500	77,49	89,13	80,92	92,31	92,99	94,74	86,33	93,51
0.2	comp, ngf	10	6	100	78,71	83,87	79,63	82,76	95,00	100,00	86,45	90,57
0.2	comp, ngf	50	8	100	79,81	90,32	82,24	89,29	95,10	100,00	87,98	94,34
0.2	comp, ngf	100	4	500	78,55	80,65	80,64	79,31	95,12	100,00	87,05	88,46
0.2	comp, ngf	150	4	300	78,92	77,42	80,84	81,48	95,15	95,65	87,16	88,00
0.2	comp, ngf	200	8	300	78,26	96,77	81,54	100,00	93,81	96,15	86,96	98,04
0.2	comp, ngf	250	4	100	78,31	87,10	81,03	85,71	94,05	100,00	86,81	92,31
0.1	comp, ngf	10	8	100	83,75	81,25	85,50	86,67	96,75	100,00	90,53	92,86
0.1	comp, ngf	50	4	500	79,00	87,50	80,20	93,33	96,00	93,33	86,86	93,33
0.1	comp, ngf	100	8	700	80,88	56,25	83,15	80,00	95,47	66,67	88,55	72,73
0.1	comp, ngf	150	8	100	81,08	75,00	83,05	83,33	96,06	83,33	88,69	83,33
0.1	comp, ngf	200	8	900	79,16	81,25	81,34	78,57	94,79	100,00	87,11	88,00
0.1	comp, ngf	250	6	100	79,55	87,50	80,63	92,31	96,37	92,31	87,29	92,31
	comp, ngf		**6,33**	**311,11**	**79,00**	**81,77**	**81,38**	**84,69**	**94,59**	**95,03**	**87,21**	**89,28**
					3,51%	2,77055556						

BALANCED

			2,… , 10	*10,… , 1000*								
Test-size	**Imp feature**	**Epochs**	**max depth**	**forest size**	**Accuracy**	**Accuracy 2**	**Precision**	**Precision 2**	**Recall**	**Recall 2**	**F1**	**F1 2**
0.3	ngf, comp	10	8	300	80,29	91,97	91,86	100,00	70,98	70,27	79,35	82,54
0.3	ngf, comp	50	8	300	83,34	96,35	94,31	100,00	69,00	82,76	79,21	90,57
0.3	comp, ngf	100	8	100	82,16	91,24	93,51	88,57	67,37	79,49	77,70	83,78
0.3	ngf, comp	150	8	500	83,57	94,89	90,79	100,00	74,79	76,67	81,56	86,79
0.3	ngf, comp	200	8	500	80,65	94,16	90,67	100,00	68,18	75,76	77,28	86,21
0.3	ngf, comp	250	8	100	88,96	93,43	87,10	100,00	66,38	74,29	74,78	85,25
0.2	comp, ngf	10	8	100	79,89	92,39	91,71	100,00	72,41	70,83	79,58	82,93
0.2	ngf, comp	50	8	500	82,46	98,91	97,96	100,00	63,26	94,12	75,33	96,97
0.2	ngf, comp	100	8	700	81,27	94,57	95,57	100,00	67,48	77,27	78,30	87,18
0.2	comp, ngf	150	8	100	80,94	98,91	91,07	100,00	72,81	95,00	81,19	97,44
0.2	ngf, comp	200	8	300	81,11	93,48	90,58	100,00	68,93	79,31	77,38	88,46
0.2	ngf, comp	250	8	700	89,17	94,57	85,55	100,00	68,87	72,22	75,57	83,87
0.1	ngf, comp	10	8	700	79,78	97,83	92,98	100,00	62,34	87,50	72,77	93,33
0.1	ngf, comp	50	8	50	82,09	97,83	93,42	100,00	67,22	91,67	76,99	95,65
0.1	ngf, comp	100	8	300	82,22	97,83	95,39	100,00	68,90	91,67	78,90	95,65
0.1	ngf, comp	150	8	300	80,43	91,30	90,44	100,00	67,95	76,47	76,13	86,67
0.1	ngf, comp	200	8	1000	81,36	97,83	91,44	100,00	71,49	93,33	79,31	96,55
0.1	ngf, comp	250	8	900	90,05	100,00	88,82	100,00	69,57	100,00	76,93	100,00
	ngf, comp		**8,00**	**413,89**	**82,76**	**95,42**	**91,84**	**99,37**	**68,77**	**82,70**	**77,68**	**89,99**
					15,29%	12,65						

B

Appendix 2: results from the original algorithm

The following pages include the results obtained from the classification algorithm without the sklearn library. It is a PDF file extracted from the excel data file.

The green lines marks the best result obtained for each dataset. Calculated mean values are shown at the end of each column.

RESULTS, original algorithm without scikit-learn library

Serum_data

ORIGINAL: 67 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy	Precision	Recall	F1
0.3	600	10	6	4	50	75,50	75,14	90,00	81,83
0.2	600	10	6	4	50	89,23	86,43	100,00	92,27
0.1	600	10	6	4	50	75,71	74,17	95,00	83,30
						82,37	**80,79**	**95,00**	**87,05**

BALANCED: 116 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy (B)	Precision (B)	Recall (B)	F1 (B)
0.3	300	10	6	4	50	89,43	100,00	56,25	71,79
0.2	300	10	6	4	50	74,78	92,00	51,67	65,58
0.1	300	10	6	4	50	85,83	86,67	52,50	65,39
						82,11	**96,00**	**53,96**	**68,69**

Led_data

ORIGINAL: 82 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy	Precision	Recall	F1
0.3	500	10	4	4	50	84,80	87,87	99,41	93,07
0.2	500	10	4	4	50	81,25	80,38	96,44	86,59
0.1	500	10	4	4	50	70,83	63,33	100,00	76,85
						83,03	**84,13**	**97,93**	**89,83**

BALANCED: 174 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy (B)	Precision (B)	Recall (B)	F1 (B)
0.3	400	10	6	4	50	96,35	98,13	91,04	94,22
0.2	400	10	6	4	50	96,00	98,57	89,93	93,91
0.1	400	10	6	4	50	95,29	96,67	90,92	92,89
						96,18	**98,35**	**90,49**	**94,07**

Serum_data2

ORIGINAL: 287 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy	Precision	Recall	F1
0.3	400	10	6	4	50	73,84	70,59	79,49	74,53
0.2	400	10	6	4	50	75,26	68,78	79,95	73,95
0.1	400	10	6	4	50	78,62	57,14	82,96	67,67
						74,55	**69,69**	**79,72**	**74,24**

BALANCED: 536 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy (B)	Precision (B)	Recall (B)	F1 (B)
0.3	400	10	8	4	50	79,94	75,91	58,54	65,78
0.2	400	10	8	4	50	80,93	83,05	59,73	69,32
0.1	400	10	8	4	50	83,15	89,24	60,18	71,09
						80,44	**79,48**	**59,14**	**67,55**

Led_data2

ORIGINAL: 152 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy	Precision	Recall	F1
0.3	300	10	6	4	50	76,52	87,54	95,22	90,99
0.2	300	10	6	4	50	76,00	74,73	99,00	85,17
0.1	300	10	6	4	50	77,33	75,52	100,00	86,05
						76,26	**81,14**	**97,11**	**88,08**

BALANCED: 456 samples

Test-size	forest size	Epochs	max depth	max featu	bootstrap	Accuracy (B)	Precision (B)	Recall (B)	F1 (B)
0.3	300	10	6	4	50	82,56	86,39	45,97	59,36
0.2	300	10	6	4	50	83,74	87,63	43,07	57,10
0.1	300	10	6	4	50	81,30	79,13	41,42	52,36
						83,15	**87,01**	**44,52**	**58,23**

C

Appendix 3: results from the clustering algorithm

The following page include the results obtained from the clustering algorithm. It is a PDF file extracted from the excel data file.

The green lines marks the obtained results with a value above the specific target value for each dataset. A calculated mean is shown at the bottom of each column.

RESULTS, clustering algorithm

Dataset 1.1

Clustering_serum

ORIGINAL: 67 samples

Distance for A	Accuracy
0.1	65,67
0.2	62,69
0.3	58,21
0.4	55,22
0.5	53,73
0.6	50,75
0.7	65,67
0.8	50,75
0.9	56,72
1.0	71,64
1.1	73,13
1.2	73,13
1.3	71,64
1.4	73,13
1.5	71,64
	63,58

Target value: 71,64

Dataset 1.1 (no NGF)

Clustering_serum2

No NGF: 67 samples

Distance for A	Accuracy
0.1	76,12
0.2	59,70
0.3	59,70
0.4	62,69
0.5	70,15
0.6	65,67
0.7	55,22
0.8	53,73
0.9	71,64
1.0	73,13
1.1	73,13
1.2	71,64
1.3	73,13
1.4	71,64
1.5	71,64
	67,26

Target value: 71,64

Dataset 2.1

Clustering_led

ORIGINAL: 82 samples

Distance for A	Accuracy
0.1	62,20
0.2	50,00
0.3	76,83
0.4	59,76
0.5	59,76
0.6	52,44
0.7	74,39
0.8	81,71
0.9	53,65
1.0	71,95
1.1	74,39
1.2	70,73
1.3	73,17
1.4	73,17
1.5	73,17
	67,15

Target value: 70,73

Dataset 2.2 (no NGF)

Clustering_led2

No NGF: 82 samples

Distance for A	Accuracy
0.1	62,20
0.2	64,63
0.3	65,85
0.4	53,66
0.5	62,20
0.6	63,41
0.7	65,85
0.8	70,73
0.9	52,44
1.0	57,32
1.1	70,73
1.2	71,95
1.3	76,83
1.4	75,61
1.5	75,61
	65,93

Target value: 70,73