Universiteit Leiden
The Netherlands

Opleiding Informatica

Analyzing and classifying Borderline Personality Disorder using Datamining paradigms

Ewout Zwanenburg and Ashwini Gopal

Supervisors:
Prof. Dr. Ir. Fons J. Verbeek
Drs. C. de Vries

BACHELOR THESIS

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

13/09/2017

Abstract

Borderline Personality Disorder is a mental disorder which is mostly associated with experiencing very intense emotions. This disorder is found in 1 to 2 percent of the population. The extreme emotions the patients experience can lead to various complications like bad or low quality relationships with others but in some cases even to suicide.

During this research project we have worked on creating a model in order to correctly classify unknown subjects as either Borderline or Healthy. Before we could create such a model we had to pre-process the dataset provided using multiple fMRI manipulation tools. The pre-processing techniques used during this project are: brain extraction, slice timing correction, intensity normalization and registration to the standard template. Once the data had been pre-processed we could start the process of feature extraction, once completed we had a single *.csv* file that could be used to create the model.

Model creation was solely done in Weka, our prior experience with the software made this a easy choice. The different classifying algorithms used for model creation during this project are: J48 (or C4.5), k-nearest neighbor algorithm, naive bayes and support vector machine. In combination with these classifying algorithms we have also used the following attribute selectors: ranked gain ratio, best first and greedy step wise.

Contents

Abstract **I**

1 Introduction **1**

1.1 Work division 1

1.2 Borderline Personality Disorder 1

1.3 Research question 2

1.4 Sub-questions 2

1.5 Related work 3

1.6 Thesis overview 3

2 Data set **5**

2.1 The brain 5

2.2 fMRI 6

2.3 Voxels 6

2.4 Data collection method 7

2.5 Data explanation 8

3 Software and methods **11**

3.1 Software 11

3.1.1 MobaXterm 11

3.1.2 FSL 11

3.1.3 Python 12

3.1.4 Weka datamining tool 13

3.1.5 LLSC and the TORQUE engine 15

3.2 Methods 16

3.2.1 Manipulation using FSL tools 16

3.2.2 Voxel feature extraction 18

3.2.3 TORQUE resource manager 20

4 Experiments **23**

4.1 Data storage 23

4.2 Structure experiments . . . 25
4.3 Measurements . . . 26
4.4 Results . . . 27
4.4.1 Attribute selection . . . 27
4.4.2 Classification algorithms . . . 28
4.4.3 Onderwater and van Mil set . . . 28
4.4.4 Unsecure added to both sets . . . 29
4.5 Slice_bet_smooth . . . 30
4.6 Slice_bet_smooth_6x6x6 . . . 32
4.7 Slice_bet_smooth_regionGrowing . . . 34
4.8 Slice_bet_smooth_norm . . . 36
4.9 Slice_bet_smooth_norm_6x6x6 . . . 38
4.10 Slice_bet_smooth_norm_regionGrowing . . . 40
4.11 Onderwater and van Mil; Slice_bet_smooth_norm_6x6x6 . . . 42

5 Conclusion and discussion **46**
5.1 Research sub-questions . . . 46
5.2 Research questions . . . 47
5.3 Discussion . . . 47
5.3.1 Data pre-processing . . . 48
5.3.2 Feature extraction . . . 48
5.3.3 The best results by NaiveBayes . . . 48
5.3.4 The results by addition of the unsecure group . . . 49
5.3.5 The differences in cross validation . . . 49
5.3.6 Comparison to the Van Mil and Onderwater set . . . 49

Bibliography **52**

Chapter 1

Introduction

In this chapter we will give a short introduction to Borderline Personality Disorder, define our research question and sub-questions, discuss some relevant work, and finally give a short overview of the other chapters in this paper.

1.1 Work division

For this project we have worked in a team of two. We have done roughly the same amount of work for this project. We started off doing research into the subject together, once we had a understanding of the subject we divided the workload among us. Gopal worked on the data pre-processing and manipulation using python and FSL tool, Zwanenburg focused on datamining and model creation using Weka. During the project we assisted each other where needed, this was one of the main advantages of working in a pair. At the end of the project we evaluated the models together, and created and wrote the conclusion together.

1.2 Borderline Personality Disorder

BPD, short for Borderline Personality Disorder which is a mental disorder which is mostly associated with experiencing very intense emotions. This disorder is found at 1 to 2 percent of the population. The extreme emotions the patients experience can lead to various complications like bad or low quality relationships with others but in some cases even to suicide.

First of all as said earlier people with BPD experience intense emotions. These can hold on for some time or come in mood swings. These mood swings are common among people with BPD. Because of these intense emotions the patients can be very impulsive. This mostly leads to irrational decision making or they could even harm themselves. This self harm can vary from eating disorders to actual physical mutilation. The reason

why they can be impulsive is because they search for immediate relief from the pain they experience at that time.

Relationship are also a big problem for people with BPD. These patients are mostly scared of being abandoned by the other person. They can feel very happy when the other persons shows some sort of kindness. This can totally turn around when the other disappoints them and the emotions can go from happy to total sadness. Even though patients look for intimacy they avoid really close relationships.

Another problem with BPD patients is that they have a very bad self-image. They cant really make a realistic self image so they tend to see themselves as bad or failed which leads to a feeling of emptiness.

Causes
The cause of BPD is not really certain. There are suggestions that it leads from child abuse or other trauma but this has not been fully proven. The focus at this point is the treatment of BPD rather than the cause of it.

1.3 Research question

"How can we use Datamining paradigms to analyze and classify subjects with and without Borderline Personality Disorder into two distinct groups by using their fMRI scans." This is the research question we will try to answer at the end of this thesis.

1.4 Sub-questions

To help answer our research question as stated above, we have constructed 4 sub-questions. If we are able to answer these sub-question, we will be able to answer our research question. The following are the sub-questions we will answer in this thesis:

1. Which voxel features are relevant, and how do we process them?
   The fMRI brain scan is build out of thousands of voxels, these voxels contain information about a subjects brain activity during their fMRI scan. In order to create a model, we must extract insightful information from these voxels.

2. Which pre-processing steps should be applied on the raw data?
   Before we can extract insightful information from the data, we need to process the data into a use-able form. Many steps are required before we can obtain reliable information.

3. Which classification algorithms produce the best results?
   Different classification algorithms will result in different results. Our goal is to find a algorithm that produces the best model for our data set.

4. What are the best ways to visualize our results?
   Visualizing data is one of the simplest and most understandable ways to communicate data. Models will help understand and interpret the results.

1.5 Related work

Analyzing Borderline Personality Disorder is a fairly new subject, however, there are some papers covering the subject. One of these is the bachelor thesis from Onderwater and Van Mil, called *Finding and visualizing patterns in Borderline Personality Disorder fMRI images*. Onderwater and Van Mil have laid the groundwork for the research project we are conducting at the moment. Their project, like ours, focused on the creation of a model in order to patients as either borderline or healthy. In their thesis they discuss methods of data pre-processing, feature extraction, model creation and visualization. For our thesis we will use this as guideline, besides this we will focus more towards testing different classification algorithms, where as Onderwater and Van Mil mostly used the C4.5 decision tree algorithm.

1.6 Thesis overview

Our thesis will follow a logical order. In this chapter we have given an introduction to the subject and defined our research questions and sub-questions, and finally discussed some relevant work. In Chapter 1 we will provide an overview and explanation of our data set. Next, in Chapter 2 we will discuss the software and methods we have used during our research. After this, we will discuss the results of the experiments conducted. Finally we will give a conclusion to the experiments conducted, answer our main research question and its sub-questions and finally discuss potential future work and improvements.

Chapter 2

Data set

The brain is the the organ that controls all functions performed by the human body, together with the spinal cord it makes up the central nervous system. The brain is responsible for processing all information received by the sense organs (smell, touch, hearing, sight etc.) and making decisions based on this information, besides this the brain also controls automated functions necessary to maintain regular day-to-day life.

2.1 The brain

The brain is the the organ that controls all functions performed by the human body, together with the spinal cord it makes up the central nervous system. The brain is responsible for processing all information received by the sense organs (smell, touch, hearing, sight etc.) and making decisions based on this information, besides this the brain also controls automated functions necessary to maintain regular day-to-day life.

For a simple explanation of the functions of the brain, the brain can be divided into three different parts:

1. The cerebrum
   The cerebrum or cortex is the largest part of the brain, it is responsible for interpreting actions like speech, hearing, vision, movement and emotions. The cerebrum can be divided into four lobes:
   - Frontal lobe, responsible for tasks like: reasoning, planning, speech, movement and emotions.
   - Parietal lobe, responsible for tasks like: orientation and recognition
   - Temporal lobe, responsible for processing visual stimuli
   - Occipital lobe, responsible for tasks like: detection and recognition of sound, processing of memory and part of speech
2. The cerebellum
   The cerebellum is a smaller part of the brain when compared to the cerebrum, it is divided into three

lobes, its functions consist of maintaining a proper body posture, controlling muscle movement and keeping the body in balance. Besides this it may also be involved in functions such as attention, language and pleasure.

3. The brain stem
   The brain stem, is a small part of the three, it is located at the back of the brain. The brain stem acts as a connection point for the cerebrum and cerebellum to the spinal cord. It is responsible for automatic functions in the body like breathing, digestion, hearth rate and much more.

2.2 fMRI

fMRI, short for Functional Magnetic Resonance Imaging, is a fairly new technique that produces functional neural images by using concepts from well known MRI technology. fMRI associated blood flow in the brain with brain activity, more blood flow means higher brain activity, lower blood flow means less brain activity, this method is also called blood-oxygen-level contrast (BOLD). The result of an fMRI scan is a 4-dimensional image, the first three dimensions represent a patients brain, the fourth dimension represents the change of blood flow over time.

Whilst taking a fMRI scan the patient is asked to perform a certain predefined task. The purpose of the task is to identify different activation areas in the brain whilst a task is performed. The results produced by these tasks can be analyzed to give insight on the working of the brain. For our project we will analyze fMRI scans of borderline and non-borderline patients, we expect significant activation differences when comparing the activations produced by the respective groups, these results will be discussed in Chapter 4: Experiments.

2.3 Voxels

An fMRI scan does not provide a single 3-dimensional object, it is build up of units we call voxels. A voxel is a tiny 3D cube, fMRI scans are built of thousands of these voxels (like pictures are built of 2D pixels). A single voxel can contain information on thousands or millions of brain cells. A value is associated with each voxel, this value represents the amount of brain activity at a given time, during the period of the fMRI scan this value changes according to the actions performed by the patient.

For our research we will use these voxels as a starting point, we can analyze different voxels or groups of voxels and extract information like peaks and lows. In this thesis we will refer to this extracted information as *voxel features*.

2.4 Data collection method

The data set provided for our research consists of 103 subjects, these subject can be divided into three distinct groups, a Healthy Control group (subjects without Borderline Personality Disorder) which consists of 36 subjects. A group of subjects with low self esteem consisting of 24 subjects, and finally a group of subjects with Borderline Personality Disorder, which consisted of 42 subjects. There was also one example folder included with the data set, which did not contain any information.

Before taking the fMRI scan the subjects had the task to write down eighth memories, four neutral and four positive memories. Whilst taking the fMRI scan, the subjects had the task to read and recall these four neutral and four positive memories and experience them as vividly as possible. We can distinguish two phases in this process, the "*reading phase*" and memorize phase, in which the subject had the task of reading a memory written by them before the fMRI scan, and the "*magination phase*", in which the subject had the task to imagine the memory as vividly as possible. The task was partially self paced, the reading phase did not have a specific time limit however, the imagination phase lasted exactly 30 seconds each time. This process was repeated for each of the eighth memories. A visual representation of this process can be found in Figure 2.1.

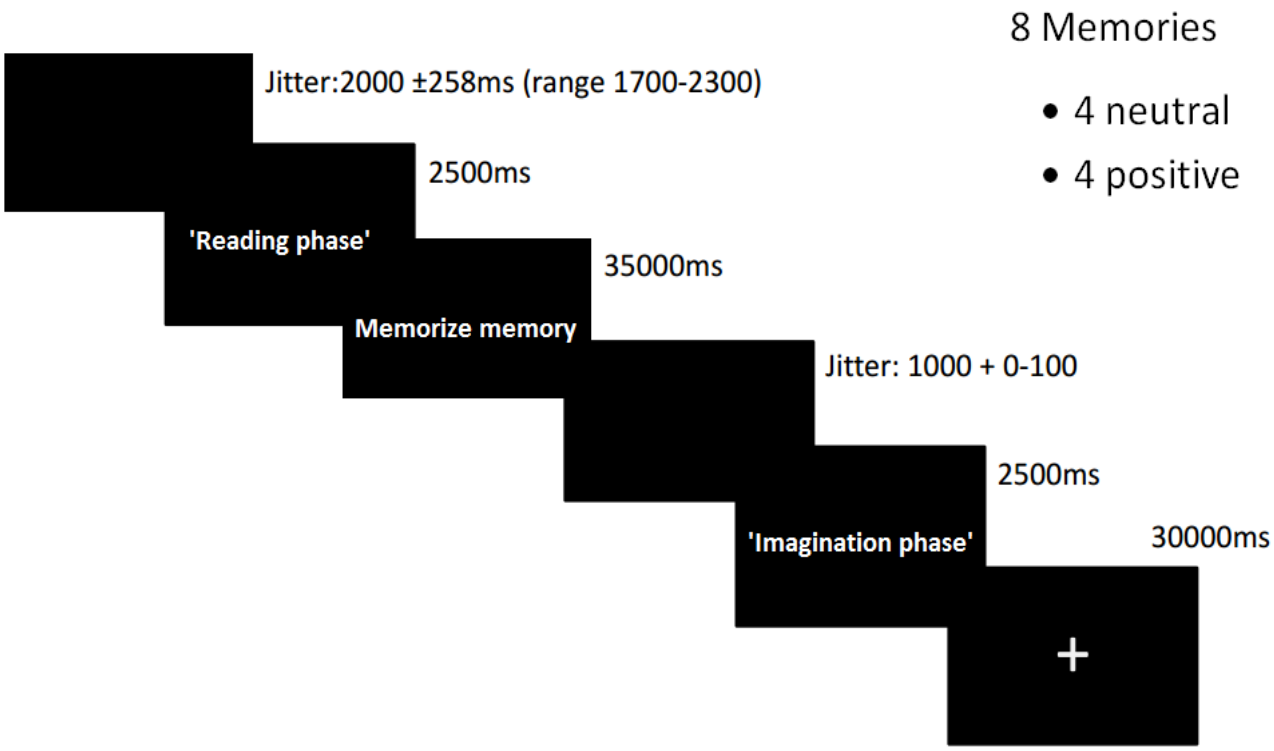

Figure 2.1: Visualization of reading phase and imagination phase

2.5 Data explanation

Our usable data set did not exist of all the 103 subject as discussed in the previous section, this is due to some scans being either not present or the scans could not be (pre-)processed. The usable data set consist of 83 subject, these are divided into the three groups as followed:

Healthy control group	32
Subjects with low self esteem	21
Subjects with Borderline Personality Disorder	30

The following subject in the data set provided were not usable:

Subject ID	*Failure reason*	*Subject ID*	*Failure reason*
B000	example file	B309	data not available
B103	processing failure	B316	data not available
B104	processing failure	B319	data not available
B105	processing failure	B320	data not available
B111	processing failure	B326	data not available
B216	data not available	B331	data not available
B222	data not available	B339	data not available
B302	data not available	B342	data not available
B305	data not available		

The data set as mentioned above was provided to us in two different ways, a set of raw fMRI scans (did not have any pre-processing steps applied), and a set of pre-processed data. The pre-processed data was provided by Charlotte van Schie and had the following pre-processing steps applied (description of these pre-processing steps can be found in Chapter 3):

- Spatial smoothing (with 5 mm kernel)
- Highpass temporal filter (with 100s)
- Motion correction
- Intensity normalization
- Registration to standard template

We decided not to use this pre-process data set provided, but instead pre-process the data ourselves. Pre-processing the data set ourselves did not only allow us to get familiar with the software and the different pre-processing techniques, but also allowed us to test and compare the difference between applying (or not applying) certain pre-processing steps on the data set.

Initially we decided not to include the subjects with a low self esteem, these subjects are neither classified as healthy or as a borderline subject. If we were to include these subject during model creation, it would

increase the amount of noise in the data and decrease the chances of finding an accurate classification model. Nonetheless, we were interested in the actual results we would get when including these subjects during feature extraction and model creation. Our main focus was model creation using only the healthy control subjects and the borderline subjects. But nearing the end, we also conducted a couple of experiments where we included the low self esteem subjects in our data set.

Chapter 3

Software and methods

In this chapter we will give an overview of the tools we have used whilst working on our research project, after that we will discuss how we have used those tools to analyze and manipulate our data set, exact results from experiments will be discussed in Chapter 4: Experiments.

3.1 Software

In this section we will cover the tools and software we have used whilst working on our research project. We will only discuss the features of each tool that were relevant for this project.

3.1.1 MobaXterm

MobaXterm is a tool that provides a linux type terminal for Windows. MobaXterm has an easy intuitive graphical user interface with tools that allowed us to easily edit and transfer file to and from the LLSC cluster. When working at from home or at the University we used MobaXterm to access and manipulate files stored on the cluster.

3.1.2 FSL

FSL, short for FMRIB Software Library, is a library of tools to analyze and process data, among that fMRI data which we used during our research project. Most of the tools provided by FSL can be run both in a graphical user interface or from the command line, we mostly used the latter one. FSL is freely available for Linux, Mac OS and Windows (through a virtual machine). The tools from the FSL library that we have used for our project can be found below.

1. FEAT Slice timing correction: slice timing correction is part of the FSL FEAT library, used to analyze and manipulate data, slice timing correction is used to correct voxel time series information.Since it takes a couple of seconds to complete each scan, each slice is taken at slightly different times, the data needs to be corrected according to the time difference between slices.

2. BET, Brain Extract Tool: deletes non-brain tissue (e.g. the skull) in the fMRI image provided.

3. MCFLIRT: a tool used to correct the data according to motion that took place whilst taking the fMRI scan. Without motion correction, the amount of noise would be much higher, thus making the subject less reliable.

4. FEAT Spatial smoothing: spatial smoothing is part of the FSL FEAT library, used to analyze and manipulate data, spatial smoothing in particular is used to reduce noise in the data without affecting the valid activation data.

5. FEAT Intensity normalization: intensity normalization is part of the FSL FEAT library, used to analyze and manipulate data, intensity normalization is used to force each volume of the fMRI scan to have the same mean intensity.

6. FLIRT, FMRIB Linear Image Registration Tool: a tool used to register a patients fMRI scan to a standard template / model.

7. FEAT Highpass temporal filtering: highpass temporal filtering is part of the FSL FEAT library, used to analyze and manipulate data, highpass temporal filtering is used to remove low frequency artifacts from the fMRI image.

8. FSLView: a tool used to view fMRI data in a 2-dimensional or 3-dimensional space.

FSL contains many more tools, we were not able to utilize all of the tools since some did not apply to our project, or the tools would increase the complexity of our project too much. We did not use the Highpass temporal filter for our own research, however the pre-processed data provided by Charlotte van Schie did use this tool.

3.1.3 Python

Python is a general purpose programming language, it is widely popular in scientific research. Pythons offers great compatibility with fMRI data, besides this van Mil and Onderwater have created many Python scripts whilst working on their research project, this was one of the factors that made use decide to use Python over other programming languages.

3.1.4 Weka datamining tool

Weka short for Waikato Environment for Knowledge Analysis, is a free datamining tool written in Java and developed at the Waikato University in New-Zealand. Weka contains tools and algorithms to pre-process, classify, cluster and visualize data among others. The large library of algorithms for classification and our prior knowledge of the software made Weka our go-to software to create our final model with.

We worked with our data as follows. After we extract our features we have a big file with all our features that we got from our data. We decided that before classifying we should try to pick the most interesting or most distinct data possible. We did this in 4 different ways:

1. Ranked InfoGain
   The first attribute selector for our data is the ranker. Weka has a special features which can give you a ranked list of attributes by how well they perform. In our case we picked InfoGain which gives the information gain for a specific feature. The ranker looks at individual features rather than how well they work together. The InfoGain is calculated by the following formula:

   $$InfoGain(Class, Attribute) = \text{H(Class)} - \text{H(Class|Attribute)}$$

   It is possible to choose how many features you want to be ranked. We (mainly) used 10 and 15 for our ranker.

2. BestFirst
   Best first is a feature attribute selector which looks at combinations of features. It start at a certain point which can vary from the beginning to the end to the middle. It will go through every feature and see if it is beneficial to add the following feature to the final set of features. It will consider all features and see if they are beneficial when they would add it in and out of the current set at a certain point.

3. GreedyStepWise
   GreedyStepWise like Bestfirst starts at a certain point in the data and looks at the remaining features to see if the others are beneficial. Unlike BestFirst, GreedyStepWise will stop immediately when none of the remaining features give additional info to the current set. This makes it a greedy way of searching for the set of features because it immediately stops instead of looking for better combinations.

4. No attribute selection
   We also decided to do a classification with all of our features to see what this does and to have a baseline (for comparison) for our results.

After the selecting the attribute of our data we start to classify our data. There are multiple ways of classifying data. We decided to go with the following 4:

1. J48

   J48 is Weka's implementation of the C4.5 decision tree algorithm. In each step C4.5 chooses an attribute to split the decision tree, an attribute is selected based on its information gain, the attribute that produces the highest information gain at a split point is chosen. This algorithm is repeated at each split point in the decision tree.

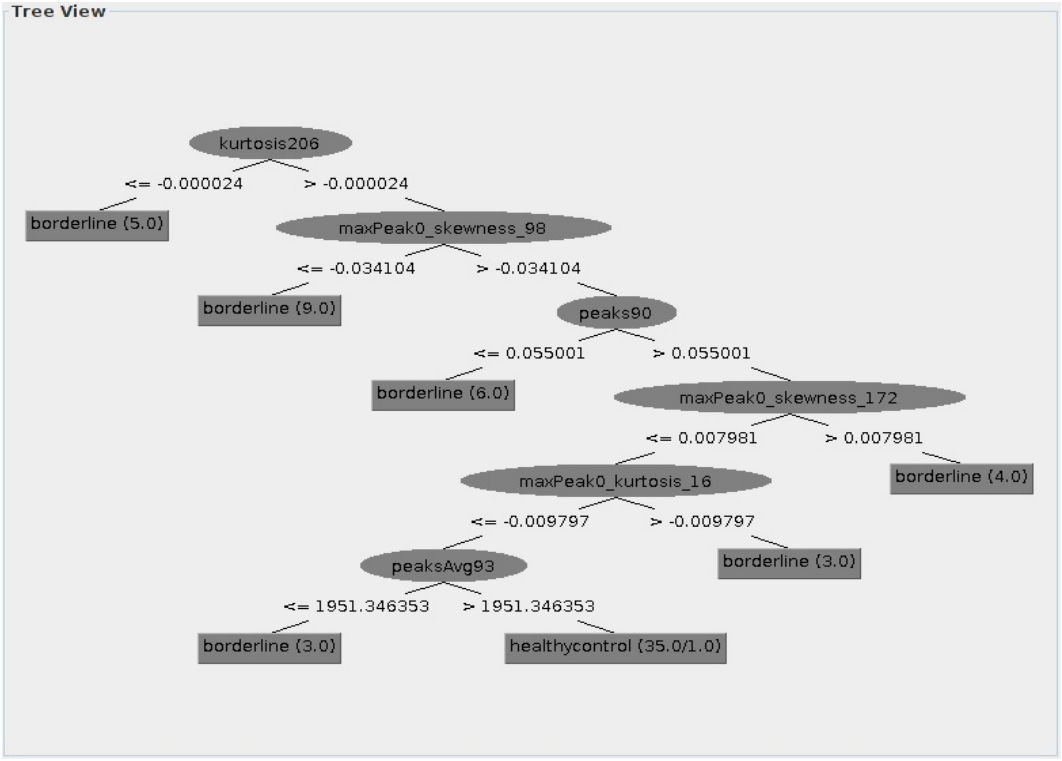

Figure 3.1: Example of a decision tree in Weka with J48

   In the example in Figure 3.1 we can see a visualization of a decision tree as build in Weka. In this example, kurtosis602 gives the highest information gain and is thus chosen as the first split point. After this, attributes with the highest information gain are recursively chosen in each split point until we get either borderline or healthy control as a leaf. Once the tree has only leafs left, the model is complete.

2. IBk

   IBk is the algorithm called k-nearest neighbor which looks at other instances in the set and looks how much they look like each other. We use 1 nearest neighbor so it only looks at the 1 nearest neighbor a specific instance has. All the instances are being evaluated and the by this results the two groups will be classified.

3. Naive Bayes

   Naive Bayes is a classification method which combines multiple different algorithms to classify the data. This way of classification looks at all the features differently instead of the correlation between these features.

4. VMO

   VMO is Weka's implementation of the support vector machine. The main principle of support vector machines is that is tries to split the data in two different areas which are split by a specific line. It tries to make 2 groups with the closest points in either group as far from the split line as possible. The different areas will now be classified as different groups.

We will also be using k-fold cross validation, the data set is divided into k parts, each of roughly equal size. Of these k parts, k-1 parts will be used to as a training set and 1 part will be used as a validation set to test the model. k = 10 is commonly used for k-fold cross validation, we will use this as a basis for testing other k values for cross validation.

3.1.5 LLSC and the TORQUE engine

LLSC, short for Liacs Life Science Cluster, is a cluster computer at the Leiden Institute of Advanced Computer Science. During our project, the data set was stored on the LLSC cluster. The LLSC cluster runs the Torque resource manager to distribute tasks to its processors. We used the Torque engine to run scripts in order to compute voxel features (see Chapter 3.2.2).

3.2 Methods

When working with the data set we had to follow a very linear working order. The tools described in the section above had to be used in a certain order to acquire usable results. Simplified, the steps can be categorized as followed:

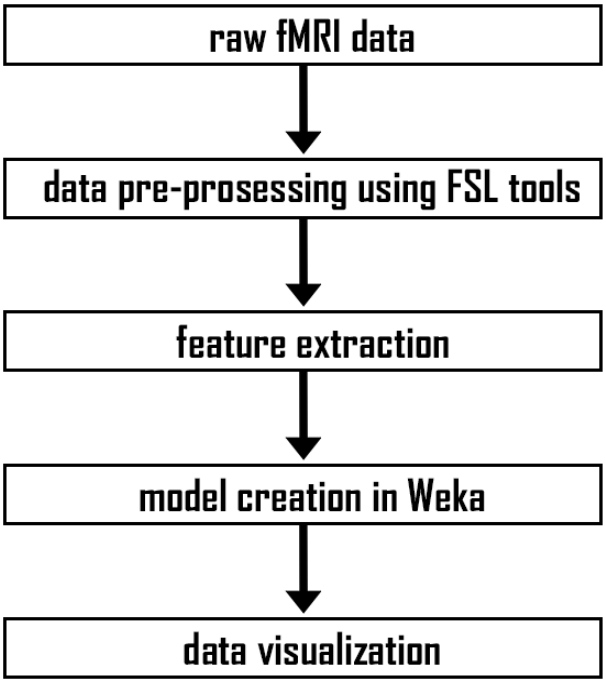

Figure 3.2: Abstract visualization of method process

These steps give an abstract description of the process we had to follow in order to create models based on our fMRI data set. Each step as shown in Figure 3.2 consists of many more steps. For each each step we will discuss what tools we used to manipulate or analyze the data, what result it had on the data and why it was necessary. The outcome of these steps will be discussed in the next chapter: Experiments.

3.2.1 Manipulation using FSL tools

Certain pre-processing steps were required to get a usable data set. The data set we received only contained raw files. We have applied the following pre-processing steps on our data set in order to get a usable data set. Not all pre-processing steps were used for each test, but when multiple pre-processing steps were applied we applied them in the following order:

Slice timing correction

Slice timing correction in used to correct time series information for each voxel. The time between the scan of the first layer and the scan of the last layer is 2.2 seconds, this means the time series information of the voxels in the first layer do not align with the information in the other layers. Slice timing correction is a method to adjust the voxel time series information between the different layers. The result is a data set in which the time series information between the layers a aligned, and thus provide more accurate information.

Brain extraction

Brain extraction is used to delete non-brain tissue from fMRI images. Unnecessary information like the skull and other noise that might be present in the scan that would result in noise in the data will be removed. The result is a fMRI scan that contains the anatomy of the brain, this scan can be easily compared and aligned with its respective MRI scan.

Motion correction

Motion correction is used to correct the scan according to the movement that took place whilst taking the fMRI scan. The slightest movement during the fMRI scan can result in artifacts in the dataset, this effect is most prominent near the edges of the scan. We use MCFLIRT to correct for this noise.

In the most simplistic way, motion correction works by by using the middle volume of the scan as a base, by comparing the base volume and its adjacent volumes it calculates a transformation value. This transformation value is then used to correct volumes beyond the first adjacent volumes.

Spacial smoothing

Spatial smoothing is a method used to to reduce noise present in the data. High voxel peak values in a certain area do not necessarily imply more brain activity. Spacial smoothing looks at a voxel's neighborhood and calculates the average activity using a Gaussian function, the result will be a group of voxels with a weighted value of the voxels in its neighborhood. There will be a high similarity between voxels of a group, and low(er) similarity between voxel groups.

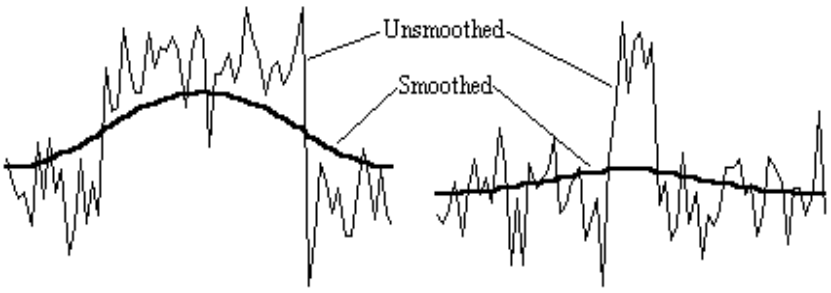

Figure 3.3: Average voxel values before and after spacial smoothing [13]

Intensity normalization

Intensity normalization is used to normalize the time series information in a fMRI scan, the result is that all the time series have roughly the same mean intensity. Groups of higher than average or lower than average values will be normalized according to the mean intensity.

Registration to standard image

Before we can extract voxel features from the fMRI images, the fMRI images have to be registered to a standard image. This is necessary so we can compare different fMRI scans with each other. Every subject has a slightly different brain size, without converting each brian to a standard template it would be impossible to accurately compare the different fMRI scans.

3.2.2 Voxel feature extraction

Once different pre-processing methods have been applied on the data set we can start extracting voxel features. In order to obtain these voxel features, we had to follow a process that consisted of multiple steps, these steps are described in Figure 3.4. The final output consisted of a single CSV file with in the top row the attributes (voxel features), and in the columns the values. The first step *create average features subject* is only necessary if we decided to use region growing. Region growing could be done in two different ways, either square regions, or regions growing with specific seed points.

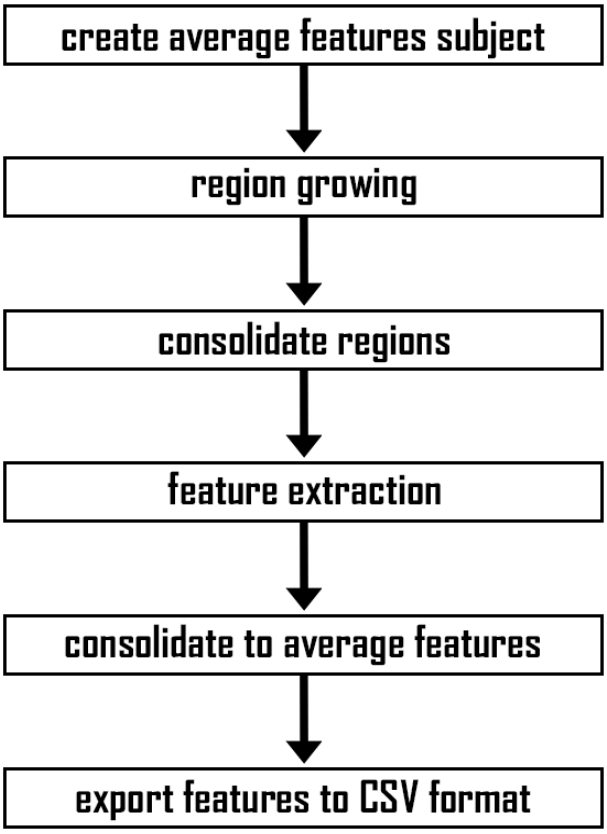

Figure 3.4: Visualization of steps in feature extraction process

Region growing

As stated above, if we choose to use region growing, it can be applied in two different ways:

1. Square regions: when using squared regions, the brain is divided in to square chunks along its axis. The number of areas used by square regions can be specified in the settings file. The result is that voxels in a square will be grouped together, this can have a negative impact on the results produced. Since the regions are decided without any knowledge of the data, the voxels are not grouped according to similarity but on which square it is in. Another negative result from using this method is that squares a the edges of the brain can contain very few voxels as compared to other brain areas. This may result in these areas resulting in a value of 0 during feature extraction. To overcome these problems, the principals of region growing are applied (see bellow). The square center points act as seed points from which the regions expand. Groups of similar voxels are clustered together.

2. Regions with seed points: when using region growing with seed points, regions are decided based on similarity between voxels. Initially, ten seed points are used, these act as a starting point for the region growing process. Neighbor voxels are compared with the seed and are added to its region if it satisfies the requirements. If a voxel does not satisfy the requirements, another voxel is chosen and this process is repeated. If no voxel can be added to a region, the process is completed for that region. The advantage of this method when compared to square regions is that voxels are grouped according to similarity instead of the square it falls into. The following seed points have been used during this project:

Brain region	*Seed point coordinates*
Caudate Nucleus	(36, 64, 47)
Cingulate gyrus	(45, 47, 47)
Frontal Pole	(48, 91, 53)
Inferior frontal gyrus	(71, 72, 42)
Insula (right)	(26, 72, 32)
Insula (left)	(64, 72, 32)
Orbitofrontal cortex	(66, 75, 29)
Precuneus	(45, 31, 55)
Superior Parietal lobe	(47, 29, 66)
Thalamus	(45, 60, 41)

In Chapter 4: Experiments, we will discuss the difference (quality of classification) between no region growing, using square regions and region growing with seed points.

Voxel features

For feature extraction we have used the scripts provided by Oderwater and Van Mil with some slight modifications. Therefor, the features extracted from our data set match the features extracted by Onderwater and Van Mil. The following features are extracted from the fMRI images:

1. Maximum value: return the maximum y-value.
2. Minimum value: return the minimum y-value.
3. Peaks: returns the amount of local maximums.
4. Average intensity: returns the average y-value.
5. Average intensity of peaks: returns the average y-values of the peaks.
6. Standard deviation of intensity values of peaks: returns the standard deviation of y-values in of the local maximums.
7. Standard deviation: returns the standard deviation of the y-values.
8. Standard deviation of distance between peaks: returns the standard deviation of the distance between peaks.
9. Skewness: returns the coefficient of skewness of the time series. (symmetry of a peak)

10. Skewness of the highest peak: returns the skewness of the highest peak.

11. Kurtosis: returns a measurement of the kurtosis of a time series. (the degree of peakedness of a peak)

12. Kurtosis of the highest peak: returns the kurtosis value of the highest peak.

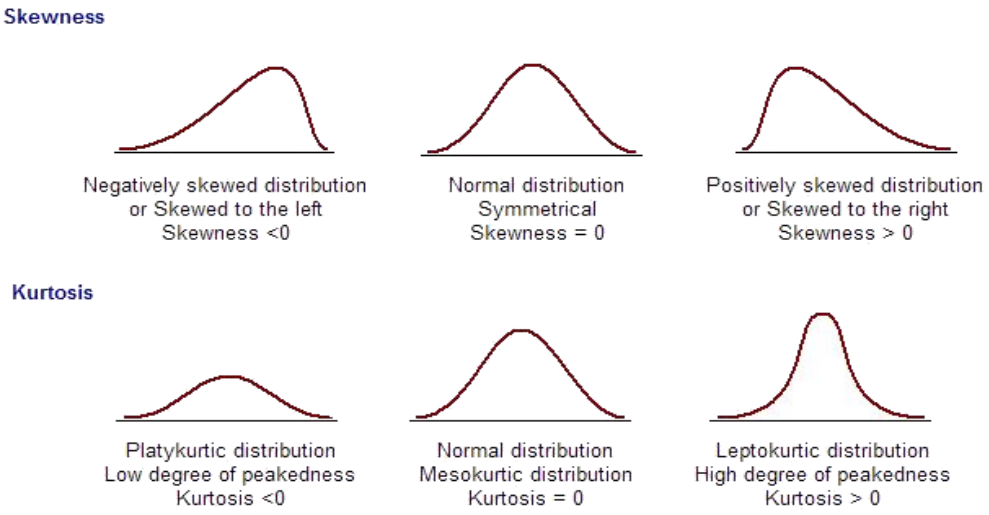

Figure 3.5: Visualization of kurtosis and skewness of a peak [24]

3.2.3 TORQUE resource manager

The process of feature extraction is time expensive, therefor most scripts we used for voxel feature extraction ran on the TORQUE engine on the LLSC. To execute code on the Torque engine, it had to be submitted in a particular format, an example with explanations can be found below.

```
1 #!/bin/bash
2 #PBS -k o
3 #PBS -l nodes=1:ppn=1,walltime=900:00
4 #PBS -l mem=4000mb
5 #PBS -m abe
6 #PBS -N create_avg_features_B102
7 #PBS -j oe
8 source /home/fswkp/pythonenv/venv/bin/activate
9 python /home/gopala/bep3/torqueJob.py create_avg_features B102
```

The most important variables can be found on lines 3 and 4. The values on line 3 are as followed: *nodes* specifies the number of nodes that must be used, *ppn* specifies the number of processors per node that can be

used, *walltime* specifies the time a process can run, $-l$ specifies that the values are a maximum limit. *mem* on line 4 specifies the maximum amount of memory a process can use. The lines after initialization contain the code that needs to be executed.

Chapter 4

Experiments

First we will give an overview of how data was stored and used for manipulation and feature extraction on the LLSC cluster. Then we will give a short description of how we keep track of what pre-processing steps and feature extraction steps are applied on the data set and the classification algorithms used. After that we will discuss and compare the actual results from the experiments conducted.

4.1 Data storage

We have used and manipulated a lot of data during this project, this data had to be stored in a clean and orderly manner.The data used during this project is stored on the LLSC cluster, the files in the output directory are stored and structured as followed:

- *slice_bet_spat*: folders titled like this contain the output after pre-processing has been applied on the raw data set. The title describes what pre-processing steps have been applied on the data set, so in this example, the following pre-processing steps have been used: slice timing correction, brain extraction and spatial smoothing. Besides these steps, all data has been registered to the standard template. Inside this main folder, there are folders for each subject, which contain the processed files. The file name describes what pre-processing techniques are applied. For feature extraction we will use the file that has the $\_flirt$ tag at the end. The pre-processing techniques are applied in the same order as specified on the file / folder name.

- ouput*_slice_bet_spat* and output_BPDaverage/*slice_bet_spat*: folders titled like this contain output after feature extraction, the text after output (in italic) describes what pre-processing steps have been applied on the data set on which feature extraction is applied. The sub-folder *output_BPDaverage* contains output files on which region manipulation is applied using borderline subject regions, the main map contains output files on which region manipulation is applied on the healthy control group. Two different kinds of region manipulations can be applied on the data set:

1. If square regions are applied on the data set (without region growing) the title will include *squareRegions_XxYxZ*. X, Y and Z specify the settings used for square regions.
2. If region growing with seed points is applied on the data set, this folder title will contain the term *regionGrowing* at the end.

If the title contains the term *unsecure*, the subjects with low self esteem (neither classified as healthy or borderline) have been included in the data set. We have used this group in three different ways: either this group has been classified on its own, so the final model has three target attributes (healthy control, borderline and unsecure), or the group has been used as part of the healthy control group or the borderline group, or finally the unsecure group has been used separately from the healhy and borderline subjects. The final csv file will only include unsecure subjects.

This output map contains the following folders and files:

- csv: contains *.csv* documents created at the end of the feature extraction process. The first row of the file contains the feature name, the columns contain the appropriate values. The values are separated by a semicolon.
- tmp_job: contains *.scripts* files for feature extraction and region growing that have been executed on the Torque engine.
- featureExtraction: contains output files (for each subject) of the feature extraction process stored as a python pickle file. The file contains a list with voxel features and its respective values.
- featuresAverage: contains files in which the average voxel feature information is stored for each subject.
- regionGrowing: if region growing is applied, this map consists of files specifying the regions found.
- settingsUsed.py: contains a copy of the settings used for the feature extraction process.
- AverageFeaturesHC*_slice_bet_spat*: is a file that contains the average features of subjects, this file is only used when region growing is applied. The part after AverageFeaturesHC (in italic) specifies what pre-processing techniques have been applied on the data set.

- CSV_output: contains all final output files of the feature extraction process, stored as a .csv file. These files are named in the same way as the folders described above. For most of our experiments we used region growing on the healthy control group, the results of these subjects can be found in the main folder. We also did a couple of experiments using region growing on borderline subjects, these results can be found in the sub-folder *BPD_Average*. These CSV files will be used to for model creation in Weka. We will discuss this in the next section.

4.2 Structure experiments

Multiple pre-processing and feature extraction steps can or cannot be applied on the data set, we have tested multiple combinations of pre-processing techniques and feature extraction methods. For each result the steps applied on the data set will be specified in a table. The following attributes can be found in these tables:

1. Title
   The title specifies what pre-processing techniques have been applied on the dataset, it can contain the following attribute:
   - Slice: slice timing correction
   - Bet: brain extraction
   - Smooth: spatial smoothing
   - Norm: intensity normalization
   - RegionGrowing: specifies that region growing has been applied with the 10 seed points
   - 6x6x6: specifies that squared regions have been used with value 6 on the x,y and z axis
2. Pre-processing
   The first value represents the pre-processing method applied for the test set, these can be either: no pre-processing, attribute selection or our features.
3. Attribute selector
   The attribute selectors are specified below the classifier, these can be either: infogain with ranker or subset evaluation. For infogain with ranker we have decided to use ranker 10 and ranker 15 (one test also includes ranker 5 for comparison), for subset evaluation we have used bestfirst and greedy step wise. In depth information on these attribute selectors can be found in Chapter 3.1.4: Weka datamining tool.
4. Classification algorithm
   The last step is selecting the classification algorithm, we have used the following four algorithms: J48, IBk, NaiveBayes and SMO. These classification algorithms have been used in combination with either 10 fold or 40 fold cross validation. The scores for correctly classified instances (CCI) and the F1 score can be found in the last two rows.

Initially we decided to focus on model creation only using the healthy control subjects and the borderline subjects. But later on we included the insecure subjects to our experiments, we have used these subjects in two different ways.

1. Unsecure part of HC: the first way was to use the unsecure subjects as part of the healthy control group. During feature extraction we classified the subjects as part of the healthy control group, we know that not every subject in this group is necessarily part of the healthy control, but we were still interested on the impact that this would have on the model.

2. Unsecure part of BPD: This is basically the same as the first option, but instead of classifying unsecure as part of the healthy control group, these subjects were classified as part of the borderline group.

Since we initially focused on model creation using only the subjects from the healthy control group and the borderline group, we have conducted more experiments on these groups and thus have more models from these groups.

4.3 Measurements

To evaluate the performance of each classifier we have decided to use the following measurements:

1. Specificity (precision)
   The specificity is a measurement that gives the proportion of positives that are correctly identified. This can also be seen as the the ability to correctly classify a subject as Healthy. The formula for the specificity is as follows:

$$Specificity = \frac{\text{TN}}{\text{TN + FP}}$$

2. Sensitivity (recall)
   The sensitivity is a measurement that gives the proportion of negatives that are correctly identified. This can also be seen as the the ability to correctly classify a subject as Borderline. The formula for the specificity is as follows:

$$Sensitivity = \frac{\text{TP}}{\text{TP + FN}}$$

3. F1 Score
   The F1 Score is a measurement of the accuracy of a test. It is calculated using the two measurements above. The following formula is used to calculate the F1 score:

$$F_1 = \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

4. Correctly classified rate (CCI)
   The correctly classified rate gives the percentage of correctly classified subjects (e.g. someone as part of the Healthy Control group being classified as Healthy Control) that the model achieved.

4.4 Results

In this sub-section we will discuss the outcome of our experiments. We will discuss and compare the performance of different classification algorithms and attribute selectors in general.

4.4.1 Attribute selection

In this section we will discuss how well the attribute selection methods performed and how well they performed in comparison to the other methods. Note that we now look at the average of the results. There are some outliers in our data but we can not give them to much value in the overall results. We will start with the results that did not use any attribute selection methods.

1. No attribute selection
   We can clearly see that this method provides the worst results in all of our data sets. As we expected this shows that selecting attributes instead of using all of the attributes that we got from our data results in better performance with our classifiers. The average results from classifying without attribute selection is mostly far below the average with a specific attribute selection.

2. Info Gain Ranker 10 and 15
   This attribute selection method which, as told before, focuses on single attributes instead of the combination of attributes clearly performs way better then using no attribute selection at all. This method gives good to even very good results in all different feature sets. When we compare the ranker that gives 10 features to the ranker that gives 15 we do not see very big differences. It differs on the feature set which amount of features gives the best results. Sometimes 10 performs slightly better and sometimes 15 does.

3. BestFirst and GreedyStepwise
   These attribute selection methods work very well on our feature sets. Most of our best results were reached with these methods. When we compare these methods with the Ranker we see that it on average gives close to the same results and in some instances they perform even better but in some other feature sets this is the other way around. BestFirst and GreedyStepwise perform overall very familiar and sometimes even exactly the same. This is because with some feature sets they chose the exact same features to classify with and this obviously gives the same results.

We wanted to use different ways of selecting our features. We both wanted to look both at the individual features as well as the combination of features. The ranking system by infoGain gave us a way of selecting a fixed number of features from our total set of features. This way we got a list of features purely based on how well these features performed on their own. Both BestFirst and GreedyStepWise gave us a way to look at the features by looking at the way the features added value in a subset. Because both BestFirst and GreedyStepwise look at how much a new feature adds to the current set of features, it looks at combinations of features rather then individual features.

4.4.2 Classification algorithms

After the selection of our attributes we had to classify them. We started with the features we were familiar with. We previously worked with both J48 and IBk. We picked these classifiers because it is easy to understand what they do exactly. We wanted these methods to be our baseline for the classifiers because even though they normally perform quite well they are still quite simple in their way of classifying data. So we searched for more sophisticated classifying algorithms. We talked with some expert and Mr. Verbeek and we choose to go with NaiveBayes and Support Vector Machines. We both do not exactly know how these algorithms work from scratch but with Weka we could easily use them without precisely knowing what the algorithm does.

Now we will discuss the results of the specific classifiers and how well the perform when we compare them to the other algorithms.

1. Base classifiers J48 and IBk
   As we expected J48 performs as a baseline for our classification. It perform slightly under average compared to the more sophisticated algorithms but overall it performs quite well. IBk is our most unreliable algorithm. Overall it performed the worst for most of our feature sets but in some instances it peaked to very high results. Still most of the times its results were close to the results from our J48 or worse.

2. Complex classifiers NaiveBayes and SMO
   Both NaiveBayes and SMO performed the best overall. With almost all feature sets SMO and NaiveBayes were the highest scoring algorithms. SMO turned out to be the most consistent. NaiveBayes sometimes dropped its results while SMO almost always ended scoring the highest or close to that. NaiveBayes on the other hand also some of the highest scores in all of the results for this project.

4.4.3 Onderwater and van Mil set

We also did some experiments with the data set used last year by Onderwater and Van Mil. We decided to look at their data in two different ways. We wanted to see what our pre-processing and classification did to their data to see if our method got good results with their data as well. We also used the features we got from our best result and picked them by hand from the features in the set Onderwater and Van Mil used. This way we wanted to see whether the features our data set delivers work as well on their data as it does on ours.

1. The same pre-processing and classifiers
   For this test we used the following pre-processing for the brain on the data Onderwater and Van Mil used. We applied slice timing correction, brain extraction, intensity normalization, squared regions with values 6x6x6 and registration to the standard template. With these features we first analyzed them as we did with any other set of features. We did attribute selection and classified after that as you can see in the results. These results were average with the exception of Naive Bayes. With all attribute selection

methods Naive Bayes performed way better then any other with in some cases even positive differences of 20 percent.

2. Own features

   We also used the features we found in our data instead of using an attribute selector. These results were not good. We used the following features and selected them by hand:

Ranker 10:	**Ranker 15:**	**BestFirst and GreedyStepwise:**
maxPeako_kurtosis_51	maxPeako_kurtosis_51	standardDeviation38
maxPeako_kurtosis_14	maxPeako_kurtosis_14	standardDeviation30
maxPeako_kurtosis_22	maxPeako_kurtosis_22	standardDeviation18
maxPeako_kurtosis_31	maxPeako_kurtosis_31	standardDeviation47
maxPeako_kurtosis_39	maxPeako_kurtosis_39	maxPeako_skewness_4
maxPeako_kurtosis_10	maxPeako_kurtosis_10	maxPeako_skewness_15
maxPeako_kurtosis_15	maxPeako_kurtosis_15	peaksIntervalStd51
maxPeako_kurtosis_52	maxPeako_kurtosis_52	peaksIntervalStd53
maxPeako_kurtosis_23	maxPeako_kurtosis_23	peaksIntervalStd18
maxPeako_kurtosis_40	maxPeako_kurtosis_40	peaksIntervalStd49
	maxPeako_kurtosis_36	peaks31
	maxPeako_kurtosis_16	maxPeako_kurtosis_39
	maxPeako_kurtosis_13	maxPeako_kurtosis_31
	maxPeako_kurtosis_25	maxPeako_kurtosis_53
	maxPeako_kurtosis_50	maxPeako_kurtosis_52
		maxPeako_kurtosis_51
		maxPeako_kurtosis_13
		maxPeako_kurtosis_10
		maxPeako_kurtosis_16
		maxPeako_kurtosis_15
		maxPeako_kurtosis_14
		maxPeako_kurtosis_22
		maxPeako_kurtosis_26
		maxPeako_kurtosis_27

Table 4.1: Own features selection set

4.4.4 Unsecure added to both sets

We also added the unsecure data to both sets to see what this did to the results. We executed the slice bet smooth norm 6x6x6 pre processing, picked the features with with the BestFirst attribute selection method and classified them using NaiveBayes. Both test ended up in worse results then we had when we didn't add the unsecure to the sets. The results are as follows:

	Borderline	**Healthy Control**
10 Fold cross validation	75.5812%	77.907%
40 Fold cross validation	73.2558%	81.3953%

Table 4.2: Results by adding the unsecure to Borderline or Healthy Control group

4.5 Slice_bet_smooth

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	68.1818%	0.681
	40 Fold:	**69.6970%**	**0.693**
IBk:			
	10 Fold:	56.0606%	0.543
	40 Fold:	53.0303%	0.522
NaiveBayes:			
	10 Fold:	65.1515%	0.620
	40 Fold:	66.6667%	0.633
SMO:			
	10 Fold:	**69.6970%**	**0.693**
	40 Fold:	68.1818%	0.678
Attribute Selection:			
InfoGain with Ranker:			
Ranked 5:			
J48:			
	10 Fold:	74.2424%	0.742
	40 Fold:	**75.7576%**	**0.758**
IBk:			
	10 Fold:	63.6364%	0.636
	40 Fold:	62.1212%	0.621
NaiveBayes:			
	10 Fold:	**75.7576%**	**0.752**
	40 Fold:	**75.7576%**	**0.752**
SMO:			
	10 Fold:	**75.7576%**	**0.754**
	40 Fold:	**75.7576%**	**0.754**
Ranked 10:			
J48:			
	10 Fold:	68.1818%	0.682
	40 Fold:	**83.3333%**	**0.833**
IBk:			
	10 Fold:	65.1515%	0.648
	40 Fold:	66.6667%	0.664
NaiveBayes:			
	10 Fold:	75.7576%	0.754
	40 Fold:	74.2424%	0.738
SMO:			
	10 Fold:	74.2424%	0.738
	40 Fold:	68.1818%	0.678

Attribute Selection:		CCI	F1 score
InfoGain with Ranker:			
Ranked 15:			
J48:			
	10 Fold:	66.6667%	0.667
	40 Fold:	**78.7879%**	**0.788**
IBk:			
	10 Fold:	68.1818%	0.681
	40 Fold:	68.1818%	0.680
NaiveBayes:			
	10 Fold:	74.2424%	0.738
	40 Fold:	74.2424%	0.735
SMO:			
	10 Fold:	77.2727%	0.770
	40 Fold:	77.2727%	0.770
Subset Evaluation:			
BestFirst:			
J48:			
	10 Fold:	65.1515%	0.652
	40 Fold:	**80.3030%**	**0.802**
IBk:			
	10 Fold:	69.6970%	0.697
	40 Fold:	71.2121%	0.712
NaiveBayes:			
	10 Fold:	69.6970%	0.687
	40 Fold:	72.7273%	0.721
SMO:			
	10 Fold:	74.7424%	0.738
	40 Fold:	75.7576%	0.752
GreedyStepWise:			
J48:			
	10 Fold:	65.1515%	0.652
	40 Fold:	**80.3030%**	**0.802**
IBk:			
	10 Fold:	69.6970%	0.697
	40 Fold:	71.2121%	0.712
NaiveBayes:			
	10 Fold:	69.6970%	0.687
	40 Fold:	72.7273%	0.721
SMO:			
	10 Fold:	74.2424%	0.738
	40 Fold:	75.7576%	0.752

4.6 Slice_bet_smooth_6x6x6

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	72.7273%	0.728
	40 Fold:	72.7273%	0.727
IBk:			
	10 Fold:	66.6667%	0.651
	40 Fold:	62.1212%	0.606
NaiveBayes:			
	10 Fold:	71.2121%	0.704
	40 Fold:	72.7273%	0.718
SMO:			
	10 Fold:	78.7879%	0.787
	40 Fold:	**81.8182%**	**0.817**
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	80.3030%	0.801
	40 Fold:	78.7879%	0.786
IBk:			
	10 Fold:	80.3030%	0.802
	40 Fold:	77.2727%	0.772
NaiveBayes:			
	10 Fold:	**84.8485%**	**0.848**
	40 Fold:	**84.8485%**	**0.848**
SMO:			
	10 Fold:	83.3333%	0.833
	40 Fold:	**84.8485%**	**0.848**
Ranked 15:			
J48:			
	10 Fold:	72.7273%	0.723
	40 Fold:	71.2121%	0.709
IBk:			
	10 Fold:	74.2424%	0.740
	40 Fold:	72.7273%	0.723
NaiveBayes:			
	10 Fold:	80.3030%	0.801
	40 Fold:	81.8182%	0.816
SMO:			
	10 Fold:	83.3333%	0.833
	40 Fold:	**86.3636%**	**0.863**

Subset Evaluation:		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	78.7879%	0.786
	40 Fold:	81.8182%	0.818
IBk:			
	10 Fold:	74.2424%	0.738
	40 Fold:	74.2424%	0.738
NaiveBayes:			
	10 Fold:	71.2121%	0.704
	40 Fold:	72.7273%	0.718
SMO:			
	10 Fold:	**87.8788%**	**0.878**
	40 Fold:	86.3636%	0.863
GreedyStepWise:			
J48:			
	10 Fold:	77.2727%	0.770
	40 Fold:	81.8182%	0.818
IBk:			
	10 Fold:	75.7576%	0.752
	40 Fold:	74.2424%	0.738
NaiveBayes:			
	10 Fold:	80.3030%	0.802
	40 Fold:	81.8182%	0.817
SMO:			
	10 Fold:	**84.8485%**	**0.848**
	40 Fold:	**84.8485%**	**0.848**

4.7 Slice_bet_smooth_regionGrowing

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	71.2121%	0.711
	40 Fold:	69.6970%	0.697
IBk:			
	10 Fold:	66.6667%	0.662
	40 Fold:	68.1818%	0.68
NaiveBayes:			
	10 Fold:	65.1515%	0.633
	40 Fold:	63.6364%	0.614
SMO:			
	10 Fold:	81.8182%	0.818
	40 Fold:	**84.8485%**	**0.848**
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	78.7879%	0.787
	40 Fold:	74.2424%	0.743
IBk:			
	10 Fold:	65.1515%	0.65
	40 Fold:	69.6970%	0.697
NaiveBayes:			
	10 Fold:	78.7879%	0.787
	40 Fold:	**81.8182%**	**0.817**
SMO:			
	10 Fold:	**81.8182%**	**0.818**
	40 Fold:	**81.8182%**	**0.818**
Ranked 15:			
J48:			
	10 Fold:	75.7576%	0.758
	40 Fold:	74.2424%	0.743
IBk:			
	10 Fold:	60.6061%	0.601
	40 Fold:	59.0909%	0.583
NaiveBayes:			
	10 Fold:	78.7879%	0.785
	40 Fold:	**81.8182%**	**0.816**
SMO:			
	10 Fold:	**81.8182%**	**0.818**
	40 Fold:	**81.8182%**	**0.818**

Subset Evaluation:		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	78.7879%	0.788
	40 Fold:	72.7273%	0.727
IBk:			
	10 Fold:	77.2727%	0.772
	40 Fold:	74.2424%	0.741
NaiveBayes:			
	10 Fold:	80.3030%	0.802
	40 Fold:	**81.8182%**	**0.817**
SMO:			
	10 Fold:	**81.8182%**	**0.818**
	40 Fold:	**81.8182%**	**0.818**
GreedyStepWise:			
J48:			
	10 Fold:	78.7879%	0.788
	40 Fold:	74.2424%	0.742
IBk:			
	10 Fold:	75.7576%	0.757
	40 Fold:	74.2424%	0.741
NaiveBayes:			
	10 Fold:	81.8182%	0.817
	40 Fold:	**84.8485%**	**0.846**
SMO:			
	10 Fold:	80.3030%	0.802
	40 Fold:	81.8182%	0.818

4.8 Slice_bet_smooth_norm

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	**71.4286%**	**0.714**
	40 Fold:	73.0159%	0.730
IBk:			
	10 Fold:	60.3175%	0.581
	40 Fold:	65.0794%	0.634
NaiveBayes:			
	10 Fold:	65.0794%	0.651
	40 Fold:	65.0794%	0.651
SMO:			
	10 Fold:	63.4921%	0.633
	40 Fold:	55.5556%	0.556
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	79.3651%	0.794
	40 Fold:	79.3651%	0.794
IBk:			
	10 Fold:	76.1905%	0.759
	40 Fold:	76.1905%	0.759
NaiveBayes:			
	10 Fold:	82.5397%	0.822
	40 Fold:	**84.1270%**	**0.839**
SMO:			
	10 Fold:	76.1905%	0.762
	40 Fold:	74.6032%	0.745
Ranked 15:			
J48:			
	10 Fold:	79.2651%	0.794
	40 Fold:	80.9524%	0.810
IBk:			
	10 Fold:	73.0159%	0.729
	40 Fold:	74.6032%	0.744
NaiveBayes:			
	10 Fold:	**84.1270%**	**0.840**
	40 Fold:	**84.1270%**	**0.840**
SMO:			
	10 Fold:	82.5397%	0.825
	40 Fold:	79.3651%	0.794

Subset Evaluation:		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	71.4286%	0.714
	40 Fold:	79.3651%	0.793
IBk:			
	10 Fold:	84.1270%	0.841
	40 Fold:	**85.7143%**	**0.857**
NaiveBayes:			
	10 Fold:	80.9524%	0.809
	40 Fold:	77.7778%	0.776
SMO:			
	10 Fold:	74.6032%	0.744
	40 Fold:	80.9524%	0.807
GreedyStepWise:			
J48:			
	10 Fold:	71.4286%	0.714
	40 Fold:	79.3651%	0.793
IBk:			
	10 Fold:	84.1270%	0.793
	40 Fold:	**85.7143%**	**0.841**
NaiveBayes:			
	10 Fold:	80.9524%	0.809
	40 Fold:	77.7778%	0.776
SMO:			
	10 Fold:	74.6032%	0.744
	40 Fold:	80.9524%	0.807

4.9 Slice_bet_smooth_norm_6x6x6

		CCI	F1 score
No Pre Processing:			
	J48:		
	10 Fold:	71.6667%	0.716
	40 Fold:	61.6667%	0.617
	IBk:		
	10 Fold:	71.6667%	0.698
	40 Fold:	76.6667%	0.757
	NaiveBayes:		
	10 Fold:	71.6667%	0.713
	40 Fold:	71.6667%	0.713
	SMO:		
	10 Fold:	80.0000%	0.799
	40 Fold:	**81.6667%**	**0.816**
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
	J48:		
	10 Fold:	80.0000%	0.800
	40 Fold:	83.3333%	0.833
	IBk:		
	10 Fold:	83.3333%	0.833
	40 Fold:	83.3333%	0.833
	NaiveBayes:		
	10 Fold:	**88.3333%**	**0.883**
	40 Fold:	86.6667%	0.867
	SMO:		
	10 Fold:	86.6667%	0.867
	40 Fold:	**88.3333%**	**0.883**
Ranked 15:			
	J48:		
	10 Fold:	80.0000%	0.800
	40 Fold:	83.3333%	0.833
	IBk:		
	10 Fold:	81.6667%	0.814
	40 Fold:	85.0000%	0.849
	NaiveBayes:		
	10 Fold:	85.0000%	0.850
	40 Fold:	85.0000%	0.850
	SMO:		
	10 Fold:	**86.6667%**	**0.867**
	40 Fold:	**86.6667%**	**0.866**

Subset Evaluation:		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	78.3333%	0.783
	40 Fold:	76.6667%	0.767
IBk:			
	10 Fold:	90.0000%	0.899
	40 Fold:	90.0000%	0.899
NaiveBayes:			
	10 Fold:	**93.3333%**	**0.933**
	40 Fold:	**93.3333%**	**0.933**
SMO:			
	10 Fold:	91.6667%	0.917
	40 Fold:	88.3333%	0.883
GreedyStepWise:			
J48:			
	10 Fold:	78.3333%	0.783
	40 Fold:	76.6667%	0.767
IBk:			
	10 Fold:	90.0000%	0.899
	40 Fold:	90.0000%	0.899
NaiveBayes:			
	10 Fold:	**93.3333%**	**0.933**
	40 Fold:	**93.3333%**	**0.933**
SMO:			
	10 Fold:	91.6667%	0.917
	40 Fold:	88.3333%	0.883

4.10 Slice_bet_smooth_norm_regionGrowing

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	81.2500%	0.813
	40 Fold:	78.1250%	0.781
IBk:			
	10 Fold:	70.3125%	0.680
	40 Fold:	70.3125%	0.674
NaiveBayes:			
	10 Fold:	70.3125%	0.700
	40 Fold:	70.3125%	0.700
SMO:			
	10 Fold:	82.8125%	0.827
	40 Fold:	**85.9375%**	**0.859**
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	84.3750%	0.844
	40 Fold:	82.8125%	0.827
IBk:			
	10 Fold:	82.8125%	0.827
	40 Fold:	82.8125%	0.827
NaiveBayes:			
	10 Fold:	**87.5000%**	**0.875**
	40 Fold:	**87.5000%**	**0.874**
SMO:			
	10 Fold:	84.3750%	0.843
	40 Fold:	84.3750%	0.843
Ranked 15:			
J48:			
	10 Fold:	84.3750%	0.844
	40 Fold:	82.8125%	0.827
IBk:			
	10 Fold:	79.6875%	0.795
	40 Fold:	79.6875%	0.795
NaiveBayes:			
	10 Fold:	85.9375%	0.859
	40 Fold:	**87.5000%**	**0.874**
SMO:			
	10 Fold:	85.9375%	0.859
	40 Fold:	84.3750%	0.843

Subset Evaluation:		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	84.3750%	0.843
	40 Fold:	**87.5000%**	**0.874**
IBk:			
	10 Fold:	84.3750%	0.844
	40 Fold:	84.3750%	0.843
NaiveBayes:			
	10 Fold:	84.3750%	0.843
	40 Fold:	**87.5000%**	**0.874**
SMO:			
	10 Fold:	84.3750%	0.843
	40 Fold:	84.3750%	0.841
GreedyStepWise:			
J48:			
	10 Fold:	84.3750%	0.843
	40 Fold:	**87.5000%**	**0.874**
IBk:			
	10 Fold:	84.3750%	0.844
	40 Fold:	84.3750%	0.843
NaiveBayes:			
	10 Fold:	84.3750%	0.843
	40 Fold:	**87.5000%**	**0.874**
SMO:			
	10 Fold:	84.3750%	0.843
	40 Fold:	84.3750%	0.841

4.11 Onderwater and van Mil; Slice_bet_smooth_norm_6x6x6

		CCI	F1 score
No Pre Processing:			
J48:			
	10 Fold:	46.1538%	0.462
	40 Fold:	50.7692%	0.504
IBk:			
	10 Fold:	53.8462%	0.492
	40 Fold:	55.3846%	0.504
NaiveBayes:			
	10 Fold:	56.9231%	0.569
	40 Fold:	**58.4615%**	**0.583**
SMO:			
	10 Fold:	**58.4615%**	**0.578**
	40 Fold:	**58.4615%**	**0.581**
Attribute Selection:			
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	53.8462%	0.539
	40 Fold:	60.0000%	0.600
IBk:			
	10 Fold:	56.9231%	0.569
	40 Fold:	56.9231%	0.569
NaiveBayes:			
	10 Fold:	**78.4615%**	**0.785**
	40 Fold:	**78.4615%**	**0.784**
SMO:			
	10 Fold:	75.3846%	0.746
	40 Fold:	69.2308%	0.686
Ranked 15:			
J48:			
	10 Fold:	58.4615%	0.585
	40 Fold:	66.1538%	0.661
IBk:			
	10 Fold:	58.4615%	0.578
	40 Fold:	60.0000%	0.578
NaiveBayes:			
	10 Fold:	78.4615%	0.784
	40 Fold:	**80.0000%**	**0.800**
SMO:			
	10 Fold:	66.1538%	0.651
	40 Fold:	63.0769%	0.623

Subset Evaluation		CCI	F1 score
BestFirst:			
J48:			
	10 Fold:	60.0000%	0.587
	40 Fold:	63.0769%	0.628
IBk:			
	10 Fold:	58.4615%	0.581
	40 Fold:	58.4615%	0.581
NaiveBayes:			
	10 Fold:	**80.0000%**	**0.798**
	40 Fold:	76.9231%	0.768
SMO:			
	10 Fold:	70.7692	0.705
	40 Fold:	69.2308%	0.688
GreedyStepWise:			
J48:			
	10 Fold:	60.0000%	0.587
	40 Fold:	63.0769%	0.628
IBk:			
	10 Fold:	58.4615%	0.581
	40 Fold:	58.4615%	0.581
NaiveBayes:			
	10 Fold:	**80.0000%**	**0.798**
	40 Fold:	76.9231%	0.768
SMO:			
	10 Fold:	70.7692%	0.705
	40 Fold:	69.2308%	0.688

Our_Features:		CCI	F1 score
InfoGain with Ranker:			
Ranked 10:			
J48:			
	10 Fold:	49.2308%	0.436
	40 Fold:	46.1538%	0.330
IBk:			
	10 Fold:	55.3846%	0.542
	40 Fold:	53.8462%	0.529
NaiveBayes:			
	10 Fold:	63.0769%	0.623
	40 Fold:	**64.6154%**	**0.640**
SMO:			
	10 Fold:	55.3846%	0.514
	40 Fold:	58.4615%	0.563
Ranked 15:			
J48:			
	10 Fold:	55.3846%	0.542
	40 Fold:	47.6923%	0.460
IBk:			
	10 Fold:	44.6154%	0.442
	40 Fold:	47.6923%	0.473
NaiveBayes:			
	10 Fold:	64.6154%	0.637
	40 Fold:	**66.1538%**	**0.651**
SMO:			
	10 Fold:	52.3077%	0.490
	40 Fold:	58.4615%	0.569
BestFirst			
J48:			
	10 Fold:	**70.7692%**	**0.708**
	40 Fold:	**70.7692%**	**0.708**
IBk:			
	10 Fold:	58.4615%	0.578
	40 Fold:	55.3846%	0.537
NaiveBayes:			
	10 Fold:	61.5385%	0.601
	40 Fold:	64.6154%	0.633
SMO:			
	10 Fold:	49.2308%	0.479
	40 Fold:	49.2308%	0.457

Chapter 5

Conclusion and discussion

In this chapter we will summarize and conclude the the research project. We will do this by answering the four sub-questions constructed at the start of this project, and afterwards answering our main research question. Afterwards we will discuss potential improvements for similar projects and discuss future work.

5.1 Research sub-questions

At the start of this research project we constructed four sub-questions that would help us answer our main research question. During the project we used these sub-questions as guidelines to help us answer our research question. Now that we have finished our project we are able to answer these sub-questions.

1. Which pre-processing steps should be applied on the raw data?
   Before we could extract useful data from the raw fMRI images, multiple pre-processing steps had to be applied on the data set. The data of unprocessed images contains a lot of noise that needed to be removed or corrected. For our research we applied the following data pre-processing techniques to our dataset: slice timing correction, brain extraction, intensity normalization and registration to the standard template. During this project we tested multiple combination of these pre-processing methods and their outcomes. Registration to the standard template was applied on all combinations, without registration comparing multiple fMRI scans would have been impractical.

2. Which steps are required before the feature extraction process?
   Once pre-processing was applied on all scans of test subjects in our dataset, we could start the process of feature extraction. As discussed in Chapter 3.2.2: Voxel feature extraction, we had to follow a very linear working order. For most experiments we used either region growing using the 10 seed points or squared regions, we expected these to give the best end results. No region manipulation always resulted in a lower scores during the model creation phase. At the end of the feature extraction process we created csv files that could be used for model creation in Weka.

3. Which classification algorithms produce the best results?
   After performing all of the pre-processing steps as discussed before we had to classify the features that were extracted. We first selected the best attributes by using specific attribute selection methods called Info Gain Rankers, BestFirst and GreedyStepwise. These attribute selectors performed quite well with BestFirst giving the overall best results. These features that were selected had to be classified afterwards. We picked 4 different algorithms called J48, IBk, NaiveBayes and Support Vector Machines. To check these classifiers we used cross validation to see how well they performed. It turned out that NaiveBayes worked out the best and gave us the best results with the support vector machines getting close to the same results. J48 and IBk overall performed as good as we expected because they work with a pretty simple algorithm compared to NaiveBayes and Support Vector Machines.

4. What are the best ways to visualize our results?
   Unfortunately we were not able to implement three dimensional visualization options during our research. We mostly used Weka visualizations like the J48 decision trees to help understand the outcome of a model.

5.2 Research questions

Now that we have discussed the outcome of the sub-question for this research paper we can answer our main research question: *"How can we use Datamining paradigms to analyze and classify subjects with and without Borderline Personality Disorder into two distinct groups by using their fMRI scans."*

Now that we have answered our research sub-questions we can finally answer our main research question. From our work it is clear that there is a significant potential to classify borderline and non-borderline subjects in distinct groups. With the use of various data pre-processing tools we can correct the raw fMRI images. This process could however be improved, intensity normalization is often referred to as a simple solution to a complicated problem. From model creation we have learned that NaiveBayes and Support Vector Machines produce better overall performances across multiple tests when compared to J48 and IBk, which are simpler algorithms. The unsecure subjects still proved to be difficult to classify. When classifying them as definite healthy control or borderline, the results worsened. Unfortunately we were not able to use the unsecure subjects as a test set during this project, however this can be used as for potential future improvement.

5.3 Discussion

In this section we will discuss the outcomes of all phases of our research. We will compare different results and methods and discuss why and how they perform differently from one another.

5.3.1 Data pre-processing

We have discussed many different data pre-processing methods and their effect on the dataset. When pre-processing the raw fMRI images, some pre-processing techniques were basically mandatory. These are brain extraction and registration to the standard template. These two techniques are not mandatory in the sense that they are not used, the result from the images will be unreliable and / or unusable. Without applying these techniques the images would contain a lot of noise and would be practically impossible to compare to each other.

Slice timing correction, spatial smoothing and intensity normalization are what we would consider optional pre-processing techniques, this in the sense that they are not mandatory as compared to brain extraction and registration to the standard template. Nevertheless, these three pre-processing techniques each improve the images compared to the raw files. The best results are produced when all three pre-processing methods are used on the dataset, this result is consistent regardless of the feature extraction process afterwards or the model creation phase in Weka. The order in which we used these is as follows: slice timing correction, brain extraction, spatial smoothing, intensity normalization and finally registration to the standard template.

5.3.2 Feature extraction

When extracting the features to be used for model creation, we have a couple of options on how to process the dataset. We can use either region growing using the 10 seed points, squared regions or neither. What we have noticed is that region growing almost always produces the best results during model creation. Squared regions are a close second, for our experiments we mostly used square regions with x,y and z values of 6. Lowering this number produces worst results, increasing would potentially produce better results. Using neither region growing nor squared regions almost always produced worse results.

Squared regions with x,y and z values of 6 performed better than region growing with the ten initial seed points. One explanation may be that since squared regions created much more seed points, the voxels grouped at each point showed higher similarity between voxels than if the voxels that would be grouped when only ten seed points are used.

These results are consistent throughout all experiments, regardless of the data pre-processing method applied or the classifiers and attribute selectors used in Weka.

5.3.3 The best results by NaiveBayes

In our results we see that the highest results were reached by the NaiveBayes algorithm. So why did this algorithm perform so well on our data? We think that this is due to the independence of our features. Our features all look at the data in different ways. From highest peak to averages. These features created don't

have a dependence on one another and that is why NaiveBayes works so well. NaiveBayes performs best when the features are independent and that is exactly the case in our data set.

5.3.4 The results by addition of the unsecure group

We can see with the results when we add the unsecure group to either the borderline or healthy control group the classification quality will go down. This is because the brains of the unsecure group differ too much from each other. The healthy control and borderline groups are more similar to each other so they can be classified better. When we add the unsecure group we add more different brains so the group as a whole won't look like each other as well as it did before. That is why our results drop when we add this group.

5.3.5 The differences in cross validation

When we tested our classifying algorithms we used cross-validation. We used both 10 Fold and 40 Fold cross-validation. We expected the 40 fold cross validation to perform better for all instances but it actually resulted in the following percentages.

	Number out of total	**Percentage out of total**
10 Fold cross validation	36	30%
40 Fold cross validation	54	45%
Same	30	25%
Total	120	100%

Table 5.1: Number of higher results between 10 and 40 fold cross validation

As we can see in the results of both of our cross validation the classification will not always be higher with more fold. Before our project we expected more fold to always perform better but this is not the case. We think that the reason for this is that with higher folds the different training sets won't differ from each other as much as the do with the 10 fold cross validation. With cross validation the set is split in k fold so when k is big one piece will be small. So there will be a lot of different training and test sets but the training sets will be quite similar if only a really small part constantly is switched out and in compared to big parts. This is why we think that even though 40 fold performs better then 10 fold sometimes 10 folds still get the better results.

5.3.6 Comparison to the Van Mil and Onderwater set

When we compare our results to the results that Van Mil and Onderwater acquired with their data set we see that we overall get a lot better results then they did. This can be explained because of the data itself. The data Van Mil and Onderwater used was as told before different from our data. Our data was a lot more in comparison to their data because the test our participants did took a bigger amount of time and different tasks were done. This way we were able to create a better classifying model. We would recommend using the test that was used on our participants rather then the one used for the set of Van Mil and Onderwater.

Even though our model worked very good on our data it worked poorly on their data. We thought that the features we extracted would work better overall because our data is bigger but we have come to the conclusion that the data sets differ too much to use the exact same features. Also looking at only their data didn't perform as well as we hoped. The results were decent and NaiveBayes actually performed quite well. This too can be explained by the independence of the features.

Bibliography

[1] J. van Mil & C. Onderwater, "Finding and visualizing patterns in Borderline Personality Disorder fMRI images," 2016.

[2] "Borderline personality disorder." https://en.wikipedia.org/wiki/Borderline_personality_disorder#Interpersonal_relationships, 31 August 2017. Accessed: 19/08/2017.

[3] "Human brain." https://en.wikipedia.org/wiki/Human_brain, 25 June 2017. Accessed: 16/06/2017.

[4] "Anatomy of the Brain." https://www.mayfieldclinic.com/PE-AnatBrain.htm, April 2016. Accessed: 16/06/2017.

[5] P. A. Kinser, "Brain Structures and their Functions." http://serendip.brynmawr.edu/bb/kinser/Structure1.html, 2000. Accessed: 18/06/2017.

[6] "Grey Matter and White Matter." http://www.indiana.edu/~p1013447/dictionary/greywhit.htm. Accessed: 19/06/2017.

[7] D. Yuhas, "What's a Voxel and What Can It Tell Us? A Primer on fMRI." https://blogs.scientificamerican.com/observations/whats-a-voxel-and-what-can-it-tell-us-a-primer-on-fmri/, 21 June 2012. Accessed: 19/06/2017.

[8] "FMRIB Software Library." https://en.wikipedia.org/wiki/FMRIB_Software_Library, 21 October 2016. Accessed: 20/06/2017.

[9] "MobaXterm." http://mobaxterm.mobatek.net/. Accessed: 18/06/2017.

[10] "NeuroWiki: Brain Extraction Tool." https://en.wikipedia.org/wiki/Cross-validation_(statistics), 30 March 2013. Accessed: 20/06/2017.

[11] "C4.5 algorithm." https://en.wikipedia.org/wiki/C4.5_algorithm, 27 June 2016. Accessed: 20/06/2017.

[12] "Cross-validation (statistics)." https://www.spinozacentre.nl/wiki/index.php/NeuroWiki:Brain_Extraction_Tool, 6 June 2017. Accessed: 20/06/2017.

[13] "Preparing MR Images for Statistical Analysis." https://users.fmrib.ox.ac.uk/~stuart/thesis/chapter_6/section6_2.html. Accessed: 22/06/2017.

[14] S. Smith, "BET." https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/BET, 05 February 2013. Accessed: 20/06/2017.

[15] "MCFLIRT & FORCE - Head Motion Correction." https://www.fmrib.ox.ac.uk/datasets/techrep/tr04ss2/tr04ss2/node13.html, 25 February 2005. Accessed: 20/06/2017.

[16] "MCFLIRT Motion Correction - User Guide." http://poc.vl-e.nl/distribution/manual/fsl-3.2/mcflirt/index.html. Accessed: 20/06/2017.

[17] "Data Preprocessing - Intensity normalization." https://sites.google.com/site/mritutorial/functional-mri-tutorials/tutorial-i-overview-of-fmri-analysis/3-data-preprocessing. Accessed: 22/06/2017.

[18] "Performance analysis of different feature selection methods in intrusion detection." https://issuu.com/ijstr.org/docs/performance-analysis-of-different-f, 10 August 2013. Accessed: 27/08/2017.

[19] "Naive Bayes classifier." https://en.wikipedia.org/wiki/Naive_Bayes_classifier, 7 August 2017. Accessed: 27/08/2017.

[20] "Support vector machine." https://en.wikipedia.org/wiki/Support_vector_machine, 9 August 2017. Accessed: 27/08/2017.

[21] "What is TORQUE, and how do I use it to submit and manage jobs on high-performance computing systems?." https://kb.iu.edu/d/avmy, 27 January 2017. Accessed: 27/06/2017.

[22] "Sensitivity and specificity." https://en.wikipedia.org/wiki/Sensitivity_and_specificity, 12 August 2017. Accessed: 24/08/2017.

[23] "Sensitivity and specificity." https://www.med.emory.edu/EMAC/curriculum/diagnosis/sensand.htm. Accessed: 24/08/2017.

[24] "Descriptive Statistics." http://www.janzengroup.net/stats/lessons/descriptive.html. Accessed: 12/09/2017.