Application of Machine Learning Algorithms for Bipolar Disorder Crisis Prediction

Axel Junestrand Leal

Bachelor's Degree Final Project in Computer Science and Engineering
Trabajo de fin de grado del Grado en Ingeniería Informática

Facultad de Informática
Universidad Complutense de Madrid

Academic year 2017/2018

Director: María Victoria López López

To my family

Index

Index of figures

Abstract

Bipolar Disorder is a complex disorder that affects millions of people in the world. It is our belief that with the use of Big Data and Machine Learning we can help both patients and doctors make a better diagnosis of this illness.

The goal of this project is to apply different Machine Learning algorithms to symptom-based patient data in order to help create a prediction model. This model would make it easier for psychiatrists to decide whether their patients might be tending towards a depression or mania episode, or staying in a euthymic state.

The first part of the project consists in the process of gathering, cleaning and visualizing data from patients with Bipolar Disorder. It includes an exhaustive analysis of the data preparation which contains code snippets and plots that help understand the data better and observe the possible relationships and dependencies between them.

The second part includes the predictive analysis of the data. In this part, different Machine Learning algorithms are analysed and applied to the patient data in order to compare prediction accuracies and select the algorithms that suit the problem the most.

The main conclusion from the project is that in order to develop a predictive model with an acceptable level of confidence, it is essential to have both an understanding of the data that is being used and the theory regarding each algorithm that is applied, as well as having enough data for the algorithms to work with.

Keywords

Machine Learning, Big Data, Python, Jupyter Notebook, Bipolar Disorder, Mood disorder prediction

Resumen

El Trastorno Bipolar es un trastorno complejo que afecta a millones de personas en todo el mundo. Creemos que con el uso del *Big Data* y el *Machine Learning* podemos ayudar tanto a pacientes como a médicos a hacer un mejor diagnóstico de esta enfermedad.

El objetivo de este proyecto es aplicar diferentes algoritmos de *Machine Learning* a datos sintomáticos de pacientes para ayudar a crear un modelo de predicción. Este modelo facilitaría a los psiquiatras la posibilidad de evaluar si sus pacientes están tendiendo hacia un episodio de depresión o manía, o si se mantendrán en un estado eutímico.

La primera parte del trabajo consiste en el proceso de recolección, limpieza y visualización de datos de pacientes con Trastorno Bipolar. Contiene un análisis exhaustivo de la preparación de los datos que incluye fragmentos de código y gráficas que permiten entender mejor los datos y observar las posibles relaciones y dependencias que puede haber entre ellos.

La segunda parte incluye el análisis predictivo de los datos. En esta parte se aplican distintos algoritmos de *Machine Learning* para comparar su precisión y de esta manera poder elegir los algoritmos que mejor se adecúan al problema.

La principal conclusión del proyecto es que para desarrollar un modelo predictivo con un nivel de confianza aceptable, es esencial entender tanto los datos que se están usando, como la teoría relacionada con cada uno de los algoritmos que se aplica, así como tener suficientes datos para que los algoritmos funcionen correctamente.

Palabras clave

Machine Learning, Big Data, Python, Jupyter Notebook, Trastorno Bipolar, Predicción en trastornos del estado de ánimo

1. Introduction

Machine Learning is becoming increasingly present in all systems that gather and process huge amounts of data, being almost an essential requirement in the development of new software applications.

One of the fields of work that could benefit from Machine Learning is, without doubt, the field of medicine. The use of Machine Learning algorithms allows the design of both classification and regression models that help with the diagnosis of different diseases, recommendation of drugs, automatic administration of drugs, etc.

On the medical branch of psychiatry, in the area of brain disorders, one of the existing disorders is Bipolar Disorder. This particular disorder is characterized by the oscillation of the patient's mood between two states, mania and depression [1], which often come accompanied by different features, both physical and psychological.

Machine Learning is the process by which certain models are created, with the help of different algorithms, that predict values based on different features and become increasingly better in making these predictions the more data they train on.

This is why Machine Learning could be a useful tool for trying to predict the episode in which a patient might be in or tend towards with the help of different Bipolar Disorder symptoms and patient data.

1.1. Research context

This final project springs from the Bip4cast project [2], which studies the appearance of crisis in patients with Bipolar Disorder in order to predict them. The main goal of the Bip4cast project is to be able to react in time and avoid the symptoms before the patients start to suffer from them.

There is a wide range of professionals working together on the Bip4cast project, including mathematicians, doctors and computer scientists, as well as patients with Bipolar Disorder. The project is developed by Clínica Nuestra Señora de la Paz, a non-profit center dedicated to Mental Health.

Various other research projects have been very relevant for this project, such as the design of a Bipolar Disorder Computer-Aided Diagnosis System [3], which combines different Machine Learning algorithms into a system in order to diagnose this disorder, or the Data Integration and Predictive Analysis for the Treatment of Drug Addiction [4], which includes a very thorough analysis on the preparation of data in a Machine Learning project. Other projects like the Analysis of a Predictive Model based on Google Cloud and Tensor Flow [5] present different Machine Learning techniques that have helped presenting other insights on the handling of huge amounts of data.

This project has been possible thanks to projects like the Design of a Big Data Architecture for the Prediction of Crisis in Bipolar Disorder [6] and the Introduction to Big Data and First Steps in a Big Data Project [7], which have helped facilitate the gathering of all the patient data.

The goal of the specific project described herein has been to try to predict the state a patient is in with the help of Machine Learning algorithms. The difficulty of this task is that not all patients behave equally when in a state or another, so if the prediction is not

completely accurate, it at least gives the psychiatrist a second opinion about the state a patient might be in or tend towards.

During the course of this project, the knowledge obtained during the Computer Science and Engineering degree has been very present, especially the one obtained with subjects like "Data Mining and the Big Data Paradigm" and "Probability and Statistics", as well as "Data Structures and Algorithms", because of the help it has brought in the understanding of algorithm structures and Machine Learning techniques.

1.2. Work plan

The procedure that has been followed in this project is: data gathering, data cleaning, Exploratory Data Analysis, algorithm comparison, algorithm selection and predictions with randomized feature values (see *Figure 1.1*).

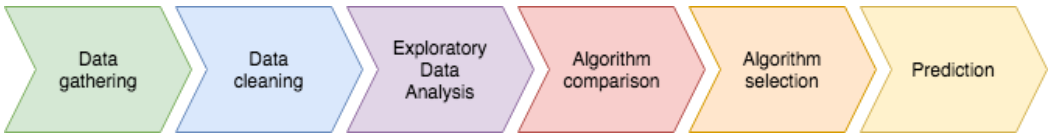

Figure 1.1. Project process

The first part of the project was to gather all the data needed for the process of Machine Learning, in order to later be able to clean and visualize it. After having gathered all the data required for the project, the next step was to clean it. This part is considered essential, as the data in this kind of projects almost always lacks entries or has empty values.

The next part of the project was the Exploratory Data Analysis (EDA), which helped us find the variables that were correlated and see the distribution of the data in the different data sets. This part also includes the identification of outliers, which are data points that are very distant from other points that can affect the prediction in a negative way.

Having a general understanding of how the data were represented in the data sets, the next part was to include different combinations of the data sets in order to find the features that could suit the problem the most.

The part were the algorithms were applied included the process of testing different baseline Machine Learning algorithms on the different data set combinations. This part was crucial in order to select the algorithms that were going to be used for the predictions. The selection was based on their accuracy, which is a good way to measure their performance.

The last part of the project consisted in the model prediction. For this, we selected the algorithms that had the best testing performance and applied them on randomized patient data in order to observe how they worked. This section also included an application to test data corresponding to different states in which the patient could be in.

The quantity of data provided for the project was not enough to produce models with a high level of confidence, but the implementation has been designed as a blueprint for future projects that include larger amounts of data.

1.3. Data used for the project

The data used for this project is anonymized patient data gathered by psychiatrists at Clínica Nuestra Señora de la Paz in Madrid. All the data was available in an Excel file with different sheets. The data has been gathered during medical appointments with four different patients that have Bipolar Disorder, but the goal for the future is that it is both recorded by the psychiatrists in appointments and with the help of mobile applications [7]. This way, the patients can actively participate in their own diagnosis.

1.4. Used technologies

The programming language used in this project is Python 2.7 [8], which has a lot of libraries that make data cleaning and visualization less complicated, as well as applying Machine Learning algorithms. The environment used is Jupyter Notebook [9].

1.4.1. Python

Python [8] is a programming language that is widely used in Machine Learning. It is especially useful for data cleaning and plotting as the language is designed for legibility and ease of use.

There are quite a few libraries designed for this task, like Pandas [10], which includes a lot of methods for data frame handling, or Seaborn and Matplotlib [11], which are data plotting libraries.

Scikit-learn [12] is the Machine Learning library of choice for this project because it includes preprocessing and cross-validation tools as well as all the known baseline Machine Learning algorithms.

1.4.2. Jupyter Notebook

Jupyter Notebook [9], the logo of which can be seen in *Figure 1.2*, is a web application consisting of "notebooks" that include live code. It is an interactive environment that contains both code and Markdown language to describe each code snippet.

Figure 1.2. Jupyter Notebook logo

The main reason this environment was chosen is the possibility it has for fast prototyping, because you can visualize the results very fast and in a cleaner way than with a terminal. The use of Markdown language is very useful too because it gives the option to explain each code snippet separately, as seen in *Figure 1.3*.

Check if there are any null values in the data set:

```
In [582]: interviews.isnull().values.any()
Out[582]: True
```

Because there are null values in the dataset, we can check which columns have null values and how many there are:

```
In [583]: null_columns=interviews.columns[interviews.isnull().any()]
          interviews[null_columns].isnull().sum()
Out[583]: irritability         1
          menstrual_cycle    647
          date                 1
          dtype: int64
```

The 'menstrual_cycle' feature values are null in all the rows, either because all the patients are men or because they were not recorded, so the smartest move is to drop the whole column, as there is no point in making up any values:

```
In [584]: interviews = interviews.drop("menstrual_cycle", 1)
```

Figure 1.3. Markdown and Python code in Jupyter Notebook

1.5. Repository

This project is shared in a public GitHub repository, which can be found at: https://github.com/AxelJunes/BDCP

This repository contains:

- A detailed description of the prerequisites and installation of all the project components.
- All the anonymized data used in the project.
- The clean data generated in the data cleaning process.
- The notebook used for the implementation of the project.
- The Random Forest classifier exported from the notebook.
- The Python script created for making predictions.

1.6. Structure of the project

This work includes a detailed description of the whole process followed in this project. It has been structured in different chapters in order to make its reading easier.

Chapters 1 and 2 include an introduction, both in English and in Spanish, in which this section is included, that contains the background of the project as well as the goals and the work plan that has been followed.

Chapter 3 is the State of the Art, which mentions the latest projects and developments currently being carried out on this field. Each project is compared with this project stating the similarities and differences with it.

Chapter 4 describes the whole data cleaning process. In the first section, which corresponds to the data gathering, each data set is described thoroughly, with an exhaustive analysis of all the features. The second section represents the data preparation, in which each data set is prepared independently. This preparation includes the removal of empty columns and the process of filling empty values in the data set.

Chapter 5 includes the Exploratory Data Analysis, by which the data is visualized and analysed to get a better understanding of it. This chapter has a section for each data set obtained in chapter 4. These sections include three types of plots: histograms, heatmaps and scatterplots. Each plot includes a thorough analysis of the most interesting relationships found in it. The correlated features are plotted with marginal and kernel density plots, with their corresponding analysis.

Chapter 6 contains the combination of the different data sets and an analysis of each of these combinations explaining why they should or shouldn't be used for the prediction. Some plots that include the distribution of the values on different patient episodes are also presented in this part, accompanied by their corresponding analysis.

Chapter 7 is the core of this project, the application of Machine Learning algorithms. This chapter contains a section for each algorithm that has been used, which includes the theory behind it as well as its application on each data set obtained in chapter 6. It also includes a section in which the algorithms are tested on randomized data. The last part of this chapter contains tests made on mocked data.

Chapter 8 presents the results obtained during this project, with a critical and reasoned discussion of each result, and their conclusions.

Chapters 9 and 10 contain a summary of the project conclusions as well as the future work that can be applied to the results obtained during this project, both in English and in Spanish.

2. Introducción

El *Machine Learning* o Aprendizaje Automático está cada vez más presente en todos los sistemas que reúnen y procesan grandes cantidades de datos, siendo prácticamente un requisito indispensable en el desarrollo de nuevas aplicaciones *software*.

Uno de los ámbitos de trabajo que podría beneficiarse del uso del *Machine Learning* es, sin duda, la medicina. El uso de algoritmos de *Machine Learning* permite el diseño de modelos de clasificación y regresión que ayudan en el diagnóstico de diferentes enfermedades, la recomendación de medicamentos, la administración automática de dosis a pacientes, etc.

En la rama médica de la psiquiatría, en el área de trastornos cerebrales, uno de los trastornos existentes es el Trastorno Bipolar. Este trastorno se caracteriza por la oscilación del estado de ánimo del paciente entre dos estados, la manía y la depresión [1], los cuales a menudo vienen acompañados de diferentes características, tanto físicas como psicológicas.

El *Machine Learning* es el proceso mediante el cual se crean modelos, con la ayuda de diferentes algoritmos, que predicen valores basados en ciertas características y cuyas predicciones mejoran a medida que reciben más información.

Esta es la razón por la que el *Machine Learning* podría ser una herramienta útil para tratar de predecir el episodio en el cual se encuentra un paciente o al cual podría estar tendiendo, con la ayuda de los diferentes síntomas del Trastorno Bipolar y datos de pacientes.

2.1. Contexto de investigación

Este trabajo surge del proyecto Bip4cast [2], el cual estudia la aparición de crisis en el Trastorno Bipolar para predecirlas. El objetivo principal del proyecto Bip4cast es poder reaccionar a tiempo y evitar los síntomas antes de que los pacientes los empiecen a sufrir.

Un amplio grupo de profesionales colaboran en el proyecto Bip4cast, incluidos matemáticos, médicos e informáticos, así como pacientes con Trastorno Bipolar. El proyecto se está llevando a cabo por la Clínica Nuestra Señora de la Paz, un centro sin ánimo de lucro dedicado a la salud mental.

Numerosos proyectos de investigación han sido muy relevantes para este trabajo, como el Diseño de un Sistema de Diagnostico Asistido por Ordenador del Trastorno Bipolar [3], que combina diferentes algoritmos de *Machine Learning* en un sistema para diagnosticar este trastorno, o la Integración de Datos y el Análisis Predictivo para el Tratamiento de la Drogadicción [4], el cual incluye un análisis exhaustivo de la preparación de los datos en un proyecto de *Machine Learning*. Otros proyectos como el Análisis de un Modelo Predictivo Basado en Google Cloud y Tensor Flow [5] presentan diferentes técnicas de *Machine Learning* que han ayudado a dar otras visiones sobre el manejo de grandes cantidades de datos.

Este trabajo ha sido posible gracias a proyectos como el Diseño de una Arquitectura Big Data para la Predicción de Crisis en el Trastorno Bipolar [6] y la Introducción al Big Data y Primeros Pasos en un Proyecto Big Data [7], los cuales han facilitado la recogida de los datos de los pacientes.

El objetivo del trabajo descrito en esta memoria ha sido tratar de predecir el estado de un paciente con la ayuda de algoritmos de *Machine Learning*. La dificultad de esta tarea es que no todos los pacientes se comportan de la misma manera cuando se encuentran en un estado u otro, por lo que si la predicción no es correcta en su totalidad, al menos otorga al psiquiatra una segunda opinión sobre el estado en el que podría encontrarse o al que podría estar tendiendo el paciente.

Durante el transcurso de este proyecto, el conocimiento obtenido durante el Grado en Ingeniería Informática ha estado muy presente, sobre todo el obtenido en las asignaturas de "Minería de Datos y el Paradigma Big Data" y "Probabilidad y Estadística", así como también "Estructuras de Datos y Algoritmos", debido a la ayuda que han aportado en la comprensión de las estructuras de los algoritmos y las técnicas de *Machine Learning*.

2.2. Plan de trabajo

El procedimiento seguido durante este proyecto ha sido: recopilación de los datos, limpieza de los datos, análisis exploratorio de los datos, comparación de los algoritmos, selección de los algoritmos y predicción con datos aleatorios (ver *Figura 1.1*).

La primera parte del proyecto consistió en recopilar todos los datos necesarios para el proceso de *Machine Learning*, para posteriormente poder limpiarlos y visualizarlos. Después de haber reunido todos los datos necesarios para el proyecto, el siguiente paso fue limpiarlos. Este proceso es esencial, ya que los datos recopilados para este tipo de proyecto casi siempre carecen de algunas entradas o contienen valores vacíos.

La siguiente etapa del proyecto fue el análisis exploratorio de datos, el cual nos ayudó a encontrar las variables correlacionadas y a observar la distribución de los datos en los diferentes conjuntos. Esta parte también incluye la identificación de valores atípicos, los cuales son puntos que se encuentran muy lejos del resto de datos y pueden afectar negativamente a la predicción.

Una vez comprendida la representación general de los datos de los diferentes conjuntos, el siguiente paso consistió en incluir distintas combinaciones de los conjuntos de datos para encontrar las características que más se ajustaran al problema planteado.

La parte en la que se aplicaron los algoritmos incluía el proceso mediante el cual se probaban diferentes algoritmos de *Machine Learning* a las combinaciones obtenidas anteriormente. Esta parte fue crucial para seleccionar los algoritmos que se iban a utilizar para las predicciones. La selección de los algoritmos se basó en la precisión de los mismos, lo cual es una buena manera de comparar su rendimiento.

La última fase del trabajo consistió en la predicción de los modelos. Para ello, seleccionamos los algoritmos que tenían mejor rendimiento con los datos de prueba y los aplicamos a datos de pacientes aleatorios para observar su funcionamiento. En esta sección también se incluyó una aplicación para probar datos correspondientes a diferentes estados reales en los que el paciente se podía encontrar.

La cantidad de datos proporcionados para el trabajo no era suficiente para producir modelos con un elevado nivel de confianza, pero la implementación se ha diseñado como una plantilla para proyectos futuros que incluyan una mayor cantidad de datos.

2.3. Datos utilizados en el proyecto

Los datos utilizados en este trabajo son datos anónimos de pacientes recopilados por psiquiatras de la Clínica Nuestra Señora de la Paz, en Madrid. Todos los datos fueron proporcionados mediante un archivo Excel con diferentes hojas de cálculo. Los datos han sido recopilados durante las citas médicas de cuatro pacientes con Trastorno Bipolar, pero el objetivo para el futuro es que estos sean recogidos tanto en citas médicas como con la ayuda de aplicaciones móviles [7]. De esta forma, los pacientes pueden participar activamente en su propio diagnóstico.

2.4. Tecnologías utilizadas

El lenguaje de programación utilizado en este proyecto es Python 2.7 [8], el cual tiene una gran cantidad de librerías que facilitan la limpieza y visualización de los datos, así como la aplicación de algoritmos de *Machine Learning*. El entorno utilizado ha sido Jupyter Notebook [9].

2.4.1. Python

Python [8] es un lenguaje de programación utilizado ampliamente para el *Machine Learning*. Es especialmente útil para la limpieza de datos y la creación de gráficas ya que está diseñado para ser legible y fácil de utilizar.

Existen bastantes librerías diseñadas para esta tarea, como Pandas [10], que incluye muchos métodos para el manejo de conjuntos de datos, o Seaborn y Matplotlib [11], que son librerías para crear gráficas.

Scikit-learn [12] es la librería de *Machine Learning* escogida para este proyecto ya que incluye herramientas de preprocesado y validación cruzada, así como todos los algoritmos básicos de *Machine Learning* conocidos.

2.4.2. Jupyter Notebook

Jupyter Notebook [9], cuyo logotipo se puede ver en la *Figura 1.2*, es una aplicación web que consiste en "cuadernos" que incluyen ejecución de código en vivo. Es un entorno interactivo que incluye tanto código como lenguaje de marcado para la descripción de los fragmentos de código.

La razón principal por la que se escogió este entorno es la posibilidad que ofrece para el prototipado, ya que se visualizan los resultados más rápido y de una forma más limpia que desde una consola de comandos. El uso del lenguaje de marcado también es muy útil ya que ofrece la opción de explicar cada fragmento de código por separado, como se puede ver en la *Figura 1.3*.

2.5. Repositorio

Este proyecto se encuentra compartido en un repositorio público de GitHub, el cual se puede encontrar en: https://github.com/AxelJunes/BDCP

Este repositorio contiene:

- Una descripción detallada de los prerrequisitos y la instalación de todos los componentes del proyecto.
- Todos los datos anonimizados que se han utilizado en el proyecto.
- Los datos limpios generados durante la fase de limpieza de los datos.
- El *notebook* utilizado para la implementación del proyecto.
- El clasificador *Random Forest* exportado desde el *notebook*.
- El *script* de Python creado para realizer las predicciones.

2.6. Estructura de la memoria

Esta memoria incluye una descripción detallada de todo el proceso seguido durante el proyecto. Ha sido estructurada en diferentes capítulos para facilitar su lectura.

Los capítulos 1 y 2 incluyen una introducción, en inglés y en castellano, que contiene tanto los antecedentes del proyecto, como los objetivos y el plan de trabajo seguido durante el mismo.

El capítulo 3 es el Estado del Arte, que menciona los últimos proyectos y desarrollos que se están llevando a cabo en este campo. Cada proyecto se compara con este proyecto indicando las similitudes y las diferencias con él.

El capítulo 4 describe todo el proceso de limpieza de los datos. En la primera sección, que corresponde a la recopilación de los datos, cada conjunto de datos se describe a fondo, con un análisis exhaustivo de todas las características. La segunda sección incluye la preparación de los datos, en la que cada conjunto de datos se prepara de manera independiente. Esta preparación incluye la eliminación de columnas vacías y el proceso de llenado de los valores vacíos de los conjuntos de datos.

El capítulo 5 incluye el análisis exploratorio de datos, mediante el cual se visualizan y analizan los datos para obtener una mejor comprensión de los mismos. Este capítulo tiene una subsección para cada conjunto de datos obtenido en el capítulo 4. Estas secciones presentan tres tipos de gráficas: histogramas, mapas de calor y gráficas de dispersión. Cada una de estas gráficas incluye un análisis en profundidad de las relaciones más importantes que se han encontrado entre los datos. Las variables correlacionadas se trazan también con gráficos marginales y de densidad de *kernel*, con su correspondiente análisis.

El capítulo 6 contiene la combinación de los diferentes conjuntos de datos y un análisis de cada combinación explicando por qué se deben, o no, usar en la predicción. También se incluyen algunas gráficas con la distribución de valores durante diferentes episodios de los pacientes, acompañadas por su correspondiente análisis.

El capítulo 7 es el núcleo del proyecto, la aplicación de los algoritmos de *Machine Learning*. Este capítulo incluye una sección para cada algoritmo utilizado, la cual contiene la teoría correspondiente a cada uno y su aplicación sobre los conjuntos de datos obtenidos en el capítulo 6. También contiene una sección en la que se prueban los algoritmos con datos aleatorios. La última parte del proyecto presenta pruebas hechas con datos simulados.

El capítulo 8 desarrolla los resultados obtenidos durante el proyecto, con una discusión crítica y razonada de cada resultado y sus conclusiones.

Los capítulos 9 y 10 consisten en un resumen de las conclusiones del trabajo así como el trabajo futuro que se puede aplicar a los resultados obtenidos, tanto en inglés como en castellano.

3. State of the art

The problem presented in this project is the prediction of a medical condition with the help of Machine Learning algorithms. This project contains the whole process of Machine Learning, ranging from data gathering to prediction, including data visualization and algorithm comparison, which can serve as a ground for other projects that are trying to predict medical conditions based on different symptoms presented by patients.

For this project to have future projection, it is important to state that the patients with Bipolar Disorder have to participate actively in gathering the data and work together with the doctors in order to get a diagnosis as accurate as possible.

3.1. Similar projects and studies

Different projects and studies are currently being carried out within the area of mood disorder crisis prediction. It is, indeed, a very new area to which apply prediction algorithms to. There are a few studies that are related to this project as in trying to predict crisis in patients with Bipolar Disorder.

3.1.1. Smartphone-based state change recognition and patient monitoring

Some studies are trying to classify the state a patient is in and detect changes in patients with Bipolar Disorder with the help of technologies such as sensors integrated in smartphones [13]. These studies are quite similar to this one in the sense that objective data is gathered from a daily survey that the patients ought to complete [14].

If we don't know the state in which the patients are when the data is gathered, it is very difficult to classify them in a state or another. This is why in this project we include the episodes of mania and depression of the patients according to the expertise of the psychiatrists.

3.1.2. Combining crisis prediction and feature importance

Knowing which variables are correlated and which features or parameters are important is essential when trying to build a model that will successfully predict the target of a study.

The goal of these studies is to investigate which features have the highest importance in health. In order to achieve this, Machine Learning algorithms and techniques are applied for feature ranking [15].

The main difference of this project with the above-mentioned is that it not only includes a study of feature importance, but it also includes the application of different algorithms to predict the state with various combinations of these features.

3.1.3. Smartphone data for symptom measuring

These studies aim to correlate Bipolar Disorder symptoms with objective data gathered from smartphones. This includes both social and physical activities that patients have to write down every evening for a certain period of time [16].

The difference with this project is that these studies only use objective data to create a model, whereas this project only uses subjective data. It is important to include subjective

data in the study because every patient might not perceive the different symptoms in the same way, and the values can be contrasted with the opinion of psychiatrists.

3.1.4. Mood charting for patient monitoring

Having in mind the involvement of Bipolar Disorder patients with their own care, the objective of these studies is to give patients a better understanding of their condition with daily charts that represent their mood [17].

These projects are quite similar to this one as they try to involve the patients with their condition and make them a part of their own treatment, but they do not include the application of algorithms to check if the different mood representations are really accurate.

4. Data cleaning

Data cleaning is an essential part of the Machine Learning process as it is the key to having concrete features and values for the model to work with. It is a difficult process and not as objective as other parts of the project might be. It consists mainly in giving quantitative values to qualitative features as these are the ones being passed to the model as parameters and they require a proper numerical scale. It also includes the identification of outliers and the handling of missing data. This process is divided in the gathering of the data, where each data set is explained thoroughly, and the cleaning of each data set.

4.1. Gathering the data

The first part of the data cleaning consisted in gathering the data that we would be working with. To do so, we exported each worksheet from the Excel file that was provided by the doctor to *csv* format.

	A	B	C	D	E	F
1	Mood	Motivation	Attention and concentration problems	Irritability	Anxiety	Sleep quality
2	2	2	3	3	3	3
3	2	2	3	3	3	3
4	2	1	3	3	3	3
5	1	2	2	2	2	3
6	1	1	3	2	2	2
7	1	2	3	2	2	3
8	1	0	2	2	2	3
9	1	1	2	2	3	2
10	1	0	3	3	3	4
11	1	0	2	2	2	1
12	1	1	2	2	1	2
13	1	0	2	2	2	1
14	1	0	2	2	2	1
15	1	0	2	2	2	2

Figure 4.1. Sample of the original Excel file with interview data (Part1)

G	H	I	J	K	L	M	N
Number of cigarettes	Caffeine	Alcohol	Other drugs	Wake up time	Going to bed time	Code	Date
34	150	No	No	06:30	23:40	D	01/06/2017
38	150	NO	No	06:45	00:15	D	02/06/2017
39	120	NO	No	07:00	00:15	D	03/06/2017
34	120	No	No	07:00	01:30	D	04/06/2017
32	150	No	No	07:00	23:45	D	05/06/2017
36	180	No	No	06:30	23:45	D	06/06/2017
32	180	No	No	06:45	23:55	D	07/06/2017
30	120	No	No	07:00	23:45	D	08/06/2017
38	180	No	No	04:50	00:30	D	09/06/2017
36	150	No	No	07:30	11:30	D	10/11/2017
32	180	No	No	07:00	23:45	D	11/06/2017
34	90	No	No	08:00	23:45	D	12/06/2017
36	60	No	No	08:00	00:10	D	13/06/2017
34	60	No	No	06:30	00:15	D	14/06/2017

Figure 4.2. Sample of the original Excel file with interview data (Part 2)

The initial Excel file was divided into five different sheets: *Episodios* (Episodes), Young, HDRS, *Diario* (Daily interviews) and *Intervenciones* (Interventions). A sample of the interview data set can be seen in *Figure 4.1* and *Figure 4.2*. To export them to a format readable by Pandas [10], we saved each sheet as *CSV UTF-8* in Microsoft Excel.

To later import the data into our notebook, we read every *csv* file into a data frame, as shown in *Figure 4.3*. The result was five different data sets.

```
episodes = pd.read_csv('./data/episodios.csv', sep=';')
young = pd.read_csv('./data/young.csv', sep=';')
hamilton = pd.read_csv('./data/hamilton.csv', sep=';')
interviews = pd.read_csv('./data/diario.csv', sep=';')
interventions = pd.read_csv('./data/intervenciones.csv', sep=';')
```

Figure 4.3. Loading the data into data frames with Pandas

4.1.1. Episode data set

The first data set, which can be seen in *Figure 4.4*, represented different episode periods of the patients (Depression/Mania). There was a total number of four patients, whose name was anonymized for privacy reasons. The patient with code "O" had constantly mixed episodes so, by recommendation of the psychiatrist, it was opted out for the study. This left us with three patients.

PATIENT	START	END	EPISODE
D	01/07/2017	24/07/2017	DEPRESSION
D	15/08/2017	11/09/2917	DEPRESSION
G	24/07/2017	07/08/2017	DEPRESSION
G	04/09/2017	01/11/2017	MANIA
O	Constantly mixed		
M	07/06/2017	01/07/2017	MANIA
M	14/07/2017	30/07/2017	DEPRESSION
M	25/09/2017	10/10/2017	DEPRESSION

Figure 4.4. Mania and depression episodes in patients

4.1.2. YMRS data set

The next file included the Young Mania Rating Scale (YMRS) [18] data from different days (see *Figure 4.5* and *Figure 4.6*), with the same patients as the ones in the Episode data set. This scale is one of the most widely used scales to rate manic symptoms. It includes eleven scoring items that are based on the patient's own report over the previous 48 hours. These scoring items and their values are:

- **Elevated Mood:**
  - 0: absent
  - 1: mildly increased on questioning
  - 2: optimistic, self-confident, cheerful
  - 3: elevated, humorous
  - 4: euphoric, singing, inappropriate laughter

- **Increased Motor Activity-Energy:**
  - 0: absent
  - 1: subjectively increased
  - 2: animated, gestures increased
  - 3: excessive energy, temporary hyperactivity
  - 4: motor excitement, continuous hyperactivity
- **Sexual Interest:**
  - 0: absent
  - 1: mildly increased
  - 2: subjective increase on questioning
  - 3: spontaneous sexual content, elaborates on sexual matters
  - 4: openly sexual towards other patients, staff or interviewer
- **Sleep:**
  - 0: no decrease in sleep
  - 1: sleeping less than normal amount by up to one hour
  - 2: sleeping less than normal amount by more than one hour
  - 3: decreased need for sleep
  - 4: denies need for sleep
- **Irritability:**
  - 0: absent
  - 2: subjectively increased
  - 4: irritable during times at interview
  - 6: frequently irritable during interview
  - 8: hostile, interview not possible
- **Speech (verbal expressions):**
  - 0: no increase
  - 2: talkative
  - 4: increased rate and amount at times
  - 6: consistently increased rate, difficult to interrupt
  - 8: uninterruptible, continuous speech
- **Language-Thought Disorder:**
  - 0: absent
  - 1: circumstantial
  - 2: distractible
  - 3: flight of ideas, difficult to follow
  - 4: incoherent, impossible to communicate with
- **Content (activities):**
  - 0: normal
  - 2: questionable plans, new interests
  - 4: special projects

- 6: grandiose or paranoid ideas
- 8: delusions, hallucinations

- **Disruptive-Aggressive Behaviour:**

  - 0: absent
  - 2: subjectively increased
  - 4: animated, gestures increased
  - 6: excessive energy, temporary hyperactivity
  - 8: motor excitement, continuous hyperactivity

- **Appearance:**

  - 0: appropriate dress and grooming
  - 1: minimally unkempt
  - 2: poorly groomed
  - 3: dishevelled, partly clothed
  - 4: completely unkempt, bizarre garb

- **Insight (about the illness):**

  - 0: present, admits illness
  - 1: possibly ill
  - 2: admits behaviour change but denies illness
  - 3: admits possible behaviour change but denies illness
  - 4: denies any behavioural change

Code	Date	Euphoria	Hyperactivity	Sexual Impulse	Sleep	Irritability
G	17/07/2017	0	0	0	0	0
G	07/08/2017	0	0	0	0	0
G	30/08/2017	0	0	0	0	2
G	04/09/2017	1	0	0	0	2
G	02/10/2017	0	0	0	0	2
G	16/10/2017	0	0	0	0	2
G	30/10/2017	0	0	0	0	2
G	08/11/2017	0	0	0	0	0
G	26/12/2017	0	0	0	0	2
M	21/06/2017	0	1	0	0	0
M	21/06/2017	0	1	0	0	0
M	03/07/2017	0	0	0	0	0

Figure 4.5. Fragment of the YMRS patient data (Part 1)

Irritability	Speech	Language-Thought Disorder	Content	Disruprive-Aggressive Behaviour	Appearance	Insight
0	0	0	0	0	0	0
0	0	0	0	0	0	0
2	0	0	0	2	0	0
2	2	0	0	0	0	0
2	0	0	0	0	0	0
2	0	0	0	0	0	0
2	0	1	0	0	0	0
0	2	0	0	2	0	0
2	0	0	0	0	0	0
0	2	0	0	2	0	0
0	2	0	0	2	0	0
0	0	0	0	0	0	0

Figure 4.6. Fragment of the YMRS patient data (Part 2)

4.1.3. HDRS data set

Another scale that was used, the Hamilton Depression Rating Scale (HDRS) [19], is the most widely used scale for depression rating. It was designed to be completed after a clinical interview. It has different versions, that include more or less rating items. For this project, we used the $HDRS_{17}$, as seen on *Figure 4.7* and *Figure 4.8*, which has 17 different items regarding depression symptoms experienced by the patients in the previous week. These 17 items and their possible values are the following:

- **Depressed Mood:**
  - 0: absent
  - 1: feeling state indicated only on questioning
  - 2: feeling state spontaneously reported verbally
  - 3: communicates feeling state non-verbally (i.e. voice)
  - 4: reports feelings only on spontaneous verbal and non-verbal behaviour
- **Feelings of guilt:**
  - 0: absent
  - 1: feels that he or she has let people down
  - 2: ideas of guilt
  - 3: delusions of guilt
  - 4: hears accusatory voices and experiences threatening hallucinations
- **Suicide:**
  - 0: absent
  - 1: feels life is not worth living
  - 2: wishes he or she was dead
  - 3: ideas of suicide
  - 4: suicide attempts

- **Insomnia (early in the night):**
    - 0: no difficulty falling asleep
    - 1: occasional difficulty falling asleep
    - 2: nightly difficulty falling asleep
- **Insomnia (middle of the night):**
    - 0: no difficulty
    - 1: restless and disturbed during night
    - 2: waking up during night
- **Insomnia (early hours of the morning):**
    - 0: no difficulty
    - 1: waking in early hours of the morning, goes back to sleep
    - 2: unable to fall asleep again if he/she wakes up
- **Work and activities:**
    - 0: no difficulty
    - 1: feelings of incapacity
    - 2: loss of interest in activity
    - 3: decrease in time spent (less than 3 hours) or productivity
    - 4: stopped working because of illness
- **Retardation:**
    - 0: normal speech and thought
    - 1: slight retardation during interview
    - 2: obvious retardation during interview
    - 3: interview difficult
    - 4: complete stupor
- **Agitation:**
    - 0: none
    - 1: fidgeting
    - 2: playing with hands, hair, etc.
    - 3: can’t sit still
    - 4: nail biting, hair-pulling, etc.
- **Psychic anxiety:**
    - 0: no difficulty
    - 1: subjective tension
    - 2: worrying about minor matters
    - 3: anxious attitude (apparent in face or speech)
    - 4: fears expressed without questioning

- **Somatic (physiological) anxiety (gastro-intestinal, cardio-vascular, respiratory, urinary frequency, sweating):**
    - 0: absent
    - 1: mild
    - 2: moderate
    - 3: severe
    - 4: incapacitating
- **Somatic symptoms (gastro-intestinal):**
    - 0: none
    - 1: loss of appetite but eating.
    - 2: difficulty eating without staff urging
- **General somatic symptoms:**
    - 0: none
    - 1: heaviness in limbs, back or head
    - 2: clear-cut symptoms
- **Genital symptoms (loss of libido, menstrual disturbances):**
    - 0: absent
    - 1: mild
    - 2: severe
- **Hypochondriasis:**
    - 0: not present
    - 1: self-absorption
    - 2: preoccupation with health
    - 3: frequent complaints
    - 4: hypochondriacal delusions
- **Loss of weight:**
    - According to patient:
        - 0: no weight loss
        - 1: probable weight loss associated with illness
        - 2: definite weight loss
        - 3: not estimated
    - According to weekly measurements:
        - 0: less than 1 *lb* (450g) loss in one week
        - 1: more than 1 *lb* (450g) loss in one week
        - 2: more than 2 *lb* (900g) loss in one week
        - 3: not estimated
- **Insight:**
    - 0: acknowledges being depressed and ill
    - 1: acknowledges illness but attributes it to other factors
    - 2: denies being ill at all

Code	Date	Depressed Mood	Feelings of guilt	Suicide	Early insomnia	Medium insomnia	Verbal Expression	Language-Thought disorders	Thougth content disorders	Late insomnia	Work and activities	Retardation
G	16/10/2017	1	0	0	0	0	0	0	0	0	1	0
G	30/10/2017	0	0	2	0	2	0	0	0	0	0	0
G	08/11/2017	0	0	1	0	0	0	0	0	0	0	0
G	26/12/2017	0	0	0	0	0	0	0	0	0	2	0
O	05/06/2017	1	0	0	1	0	0	0	0	0	3	1
O	21/06/2017	3	1	2	0	0	0	0	0	0	3	1
O	04/07/2017	0	2	2	0	0	0	0	0	0	2	1
O	24/07/2017	3	1	2	0	0	0	0	0	0	4	0
O	04/08/2017	3	0	3	0	0	0	0	0	0	3	1
O	14/08/2017	0	0	0	0	0	0	0	0	0	0	0
O	02/10/2017	3	1	2	2	0	0	0	0	0	3	2
O	23/10/2017	1	1	0	0	0	0	0	0	0	0	0
O	06/11/2017	2	0	0	0	0	0	0	0	0	4	1
O	26/11/2017	1	0	0	0	0	0	0	0	0	0	0
O	26/12/2017	0	0	0	0	0	0	0	0	0	0	0

Figure 4.7. Fragment of the HDRS patient data (Part 1)

Agitation	Psychic anxiety	Somatic anxiety	Gastro-intestinal somatic symptoms	General somatic symptoms	Genital symptoms	Hypochondriasis	Loss of weight	Insight
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	1	0	0	2	0	0	0
0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	2	1	1	1	1	0	0	0
0	3	1	2	1	3	1	0	0
0	0	0	0	0	0	1	0	0
0	1	1	0	0	3	0	0	0
0	1	2	0	0	3	1	0	1
0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Figure 4.8. Fragment of the HDRS patient data (Part 2)

4.1.4. Interview data set

The next file included data that had been gathered in interviews performed by the psychiatrist on different medical appointments with the patients, as seen in *Figure 4.9* and *Figure 4.10*.

It contained both physical and psychological items. The psychological items represent more subjective data, like anxiety, irritability or concentration problems, whereas the physical items include more objective data, like the number of cigarettes that the patient has smoked in a day or the time in which the patient woke up or went to bed. The features used in the interviews were the following:

- **Mood:** mood level of the patient, ranging from -3 to 3.
- **Motivation:** motivation level of the patient, ranging from -3 to 3.
- **Attention and concentration problems:** level of attention and concentration problems of the patient, ranging from 0 to 4.
- **Irritability:** irritability level of the patient, ranging from 0 to 4.

- **Anxiety:** anxiety level of the patient, ranging from 0 to 4.
- **Sleep quality:** quality level of the patient's sleep, ranging from 0 to 4.
- **Menstrual cycle:** whether a female patient is in her menstrual cycle.
- **Number of cigarettes:** number of cigarettes smoked by the patient since he or she woke up.
- **Caffeine:** amount of caffeine ingested by the patient since he or she woke up.
- **Alcohol:** whether the patient has consumed any alcohol.
- **Other drugs:** whether the patient has consumed any other drugs.
- **Wake up time:** when the patient woke up.
- **Going to bed time:** when the patient went to bed.
- **Code:** patient code (first letter of the patient's name).
- **Date:** when the questionnaire was completed.

Mood	Motivation	Attention and concentration problems	Irritability	Anxiety	Sleep quality	Number of cigarettes
2	2	3	3	3	3	34
2	2	3	3	3	3	38
2	1	3	3	3	3	39
1	2	2	2	2	3	34
1	1	3	2	2	2	32
1	2	3	2	2	3	36
1	0	2	2	2	3	32
1	1	2	2	3	2	30
1	0	3	3	3	4	38
1	0	2	2	2	1	36
1	1	2	2	1	2	32
1	0	2	2	2	1	34
1	0	2	2	2	1	36
1	0	2	2	2	2	34

Figure 4.9. Interview data (Part 1)

Caffeine	Alcohol	Other drugs	Wake up time	Going to bed time	Code	Date
150	No	No	06:30	23:40	D	01/06/2017
150	NO	No	06:45	00:15	D	02/06/2017
120	NO	No	07:00	00:15	D	03/06/2017
120	No	No	07:00	01:30	D	04/06/2017
150	No	No	07:00	23:45	D	05/06/2017
180	No	No	06:30	23:45	D	06/06/2017
180	No	No	06:45	23:55	D	07/06/2017
120	No	No	07:00	23:45	D	08/06/2017
180	No	No	04:50	00:30	D	09/06/2017
150	No	No	07:30	11:30	D	10/11/2017
180	No	No	07:00	23:45	D	11/06/2017
90	No	No	08:00	23:45	D	12/06/2017
60	No	No	08:00	00:10	D	13/06/2017
60	No	No	06:30	00:15	D	14/06/2017

Figure 4.10. Interview data (Part 2)

4.1.5. Intervention data set

The last file included a summary of all the medical interventions that different doctors had with the patients, as shown in *Figure 4.11*. These interventions include medical appointments, videos, phone calls and interventions at Brief Hospitalization Units (UHB in the data frame) which are hospitalization units that provide a safe environment to resolve the disorder that has caused the need for urgent care in the shortest time possible. It also contains a value for the GAF (Global Assessment of Functioning) which evaluates symptoms, normal activities and interpersonal relationships of the patient.

Code	Professional	Intervention type	Attended	Date	GAF	Relief obtained compared to previous appointment
D	DU	Appointment	yes	01/06/2017	80	5. Slightly worse
D	DU	Appointment	yes	01/06/2017	80	5. Slightly worse
D	EM	Appointment	yes	19/06/2017	80	3. Slightly better
D	DU	Video	yes	26/06/2017	85	3. Slightly better
D	DU	Appointment	yes	10/07/2017	80	6. Worse
D	DU	Phone	yes	12/07/2017	50	7. Much worse
D	DU	UHB	yes	13/07/2017	60	7. Much worse
D	EM	Appointment	yes	17/07/2017	60	5. Slightly worse
D	DU	Video	yes	24/07/2017	85	2. Much better
D	DU	Appointment	yes	21/08/2017	70	5. Slightly worse
D	EM	Appointment	yes	29/08/2017	70	5. Slightly worse
D	DU	Video	yes	11/09/2017	85	2. Much better
D	EM	Appointment	yes	19/09/2017	90	3. Slightly better

Figure 4.11. Medical interventions

4.2. Preparing the data

This part included the whole process of preparing the data to get clean data sets that could be used later for visualization and for algorithm training. Each data set was cleaned separately because they all had a different structure and size.

4.2.1. Episode data set

We started by giving the columns proper names so that all the data sets had a similar format (see *Figure 4.12*).

```
episodes.columns = ['patient', 'start', 'end', 'episode']
```

Figure 4.12. Column name translation of the Episode data set

The episodes that the patients had gone through were recorded as *DEPRESIÓN* and *MANIA*, which means depression and mania respectively. In order to make the data set cleaner, we changed the codes to D for depression and M for mania.

The dates presented in the data set were recorded in string format so, in order to have all the dates in the same format, we changed them to *datetime*, which is a Python module for manipulating dates. We iterated over every entry in the data set and changed the date, as seen in *Figure 4.13*.

```
for index, row in episodes.iterrows():
    episodes['start'][index] = datetime.strptime(row.start, '%d/%m/%Y')
    episodes['end'][index] = datetime.strptime(row.end, '%d/%m/%Y')
```

Figure 4.13. Formatting the dates in the Episode data set

4.2.2. YMRS data set

The YMRS data set had a column named *Observaciones* that was filled with NaN (Not a Number) values. It didn't add any value to the data set, so we dropped it (see *Figure 4.14*).

```
young = young.drop("Observaciones", 1)
```

Figure 4.14. Dropping Observaciones column from the YMRS data set

The next step was to rename the columns the same way as we did with the Episode data set (see *Figure 4.15*), so that we had all the names in the same format.

```
young.columns = [
        'code', 'date', 'euphoria', 'hyperactivity', 'sexual_impulse',
        'sleep', 'irritability', 'verbal_expression', 'language', 'thought',
        'aggressiveness', 'appearance', 'illness_awareness'
]
```

Figure 4.15. Column name translation of the YMRS data set

We also needed to check if there were any empty (null) values in the data set, which we did with the Pandas *isnull()* function. After confirming that there were no empty values, we converted the dates in the data set to *datetime* format, like we did with the Episode data set.

4.2.3. HDRS data set

With this data set, we started by dropping the column *Tipo de intervención* the same way as we did with the YMRS data set, because it only had NaN values. We also renamed all the columns (see *Figure 4.16*) like we did with the previous data sets.

```
hamilton.columns = [
                'code', 'date', 'depressed_mood', 'guilt', 'suicide',
                'precocious_insomnia', 'medium_insomnia',
                'verbal_expression', 'language', 'thought', 'late_insomnia',
                'work', 'retardation', 'agitation', 'psychic_anxiety',
                'somatic_anxiety',
                'somatic_gastrointestinal_symptoms','somatic_general_symptoms
                ','genital_symptoms', 'hypochondria', 'weight_loss',
                'illness_awareness'
]
```

Figure 4.16. Column name translation of the HDRS data set

With the *isnull()* function we obtained that there were empty values in the data set. We proceeded to print the values that were null. By printing the values of the rows before and after these empty values, we could get an idea of which values we could fill them with, which in this case was the same for all. So, we proceeded to fill the empty values with the ones obtained from these rows, as seen in *Figure 4.17*.

```
print(hamilton[hamilton.isnull().any(axis=1)][null_columns].head())

    retardation  agitation  illness_awareness
0           NaN        0.0                0.0
3           0.0        0.0                NaN
14          0.0        NaN                0.0

print hamilton.at[1, 'retardation']

0.0

print hamilton.at[2, 'illness_awareness']
print hamilton.at[4, 'illness_awareness']

0.0
0.0

print hamilton.at[13, 'agitation']
print hamilton.at[15, 'agitation']

0.0
0.0

hamilton = hamilton.set_value(0, 'retardation', 0.0)
hamilton = hamilton.set_value(3, 'illness_awareness', 0.0)
hamilton = hamilton.set_value(14, 'agitation', 0.0)
```

Figure 4.17. Filling empty values in the HDRS data set

The next step was to format all the values in the data set so that they had the same type, because some values had Floating Point type and others had Integer type. We changed all to Integer. After that, just as we did with the Episode and YMRS data sets, we changed the format of the dates to *datetime*.

4.2.4. Interview data set

The data set that contained the interview data from different medical appointments was the one that had the biggest amount of data and therefore presented a bigger challenge when cleaning it. The first step was to rename the columns, as we did with the other data sets (see *Figure 4.18*).

```
interviews.columns = [
                'mood', 'motivation', 'attention', 'irritability', 'anxiety',
                'sleep_quality', 'menstrual_cycle', 'nr_cigarettes',
                'caffeine', 'alcohol', 'other_drugs', 'wake up time', 'going
                to bed time','patient', 'date'
]
```

Figure 4.18. Column name translation of the Interview data set

The next step was to unify the qualitative values (*NO* and *SI*) and map them (see *Figure 4.19*) giving them a numerical value (0 and 1).

```
interviews = interviews.replace(to_replace="NO",
value="No")
interviews = interviews.replace(to_replace="SI",
value="Si")

interviews = interviews.replace(to_replace="No", value=0)
interviews = interviews.replace(to_replace="Si", value=1)
```

Figure 4.19. Categorical value unification and mapping in the Interview data set

As the sleeping time overlapped different days (for example, the patients might have slept from 10 pm one day to 9 am the day after), it was easier to know how many hours the patient had been active instead of the hours of sleep. If the patient had gone to bed after midnight, we had to add 24 hours in order to get the correct amount of time that the patient had been active.

Before we could calculate the active time, we had to format the time stamps by removing the colons, so that we could compare different hours. This whole process can be seen in *Figure 4.20*.

```
interviews = interviews.replace(to_replace=":", value="")
interviews['wake up time'] = interviews['wake up time'].str.replace(':','')
interviews['going to bed time'] =
        interviews['going to bed time'].str.replace(':','')

interviews = interviews.apply(pd.to_numeric, errors='ignore')
interviews.loc[
        interviews['going to bed time'] <
        interviews['wake up time'], 'going to bed time'
] = interviews['going to bed time'] + 2400
interviews['active_time'] =
        abs((interviews['wake up time'] -
        interviews['going to bed time']).astype(int))
```

Figure 4.20. Calculating the amount of active time in the Interview data set

After calculating the amount of active time, we could drop the *Wake up time* and *Going to bed time* columns as they were no longer relevant. The next step was to see if there were any empty values in the data set, which in this case was true. We could observe that the menstrual_cycle feature was empty in all rows, probably because all the patients in the study were males. The whole column was dropped because it didn't add any value to the data set.

We needed to print the rest of the empty values in order to see if we could fill them with other data. We compared the irritability value that was empty with the previous value and the one after it to see if they were similar. We could see that both were 1, so we filled the empty space with that value. The other value that was missing was a date, which would most surely be between two other days, so we set the value to the missing day between them. This whole process can be seen in *Figure 4.21*.

```
null_columns=interviews.columns[interviews.isnull().any()]
interviews[null_columns].isnull().sum()

irritability         1
menstrual_cycle    647
date                 1
dtype: int64

interviews = interviews.drop("menstrual_cycle", 1)

null_columns=interviews.columns[interviews.isnull().any()]
print(interviews[interviews.isnull().any(axis=1)][null_columns])

     irritability        date
306           NaN  19/08/2017
533           1.0         NaN

print interviews.at[305, 'irritability']
print interviews.at[307, 'irritability']

1.0
1.0

interviews = interviews.set_value(306, 'irritability', 1.0)

print interviews.at[532, 'date']
print interviews.at[534, 'date']

03/07/2017
05/07/2017

interviews = interviews.set_value(533, 'date',
str('04/07/2017'))
```

Figure 4.21. Completing missing values in the Interview data set

After filling the empty values, we changed all the Floating Point values to Integer and changed the dates to *datetime*, like we did with the rest of the data sets, in order to have the same format for all the numeric features and the dates.

4.2.5. Intervention data set

The first step in cleaning the intervention data set was to rename the columns, which we did with the other data sets (see *Figure 4.22*).

```
interventions.columns = [
                'code', 'doctor', 'type', 'attends', 'date', 'gaf', 'relief'
]
```

Figure 4.22. Column name translation of the Intervention data set

We could also drop the rows that contained GAF values that were NaN, because we did not want to risk making up such concrete values. The doctor and the intervention type were also dropped because they did not add any value to the data set and we wanted generic data.

The interventions that the patients did not attend to were not valuable either, so we dropped them from the data set (see *Figure 4.23*). This meant that we could get rid of the column that indicated if the patient attended or not, because at that moment we only had data from interventions that the patients had attended to.

```
interventions = interventions[interventions['attends'] != 'no']
```

Figure 4.23. Dropping entries from interventions that patients did not attend to

We only needed the numerical value of the column that indicated the relief felt by the patients on each intervention, so we split the strings and took only the number (see *Figure 4.24*), which ranged from 1 to 7 (1 being much better and 7 being much worse).

```
for index, row in interventions.iterrows():
    splits = row.relief.split('.')
    interventions['relief'][index] = splits[0]
```

Figure 4.24. Taking the numerical value from the relief column

The only thing left was to change the date to *datetime* format, the same way as we did with the other data sets, in order to obtain a clean dataset.

5. Exploratory Data Analysis

The Exploratory Data Analysis [20] is the process by which the data are visualized an analysed with the purpose of making sense of the data. This is fundamental in order to see how the data behave. With the help of different graphs and maps we could get a better understanding of all the data sets as well as the different combinations of them. The episode data set was not necessary to plot because it did not contain any numerical data, only periods of time.

In this part, we plotted each data set separately. Also, before plotting, we removed the date and code/patient columns which could not be plotted. It was also important to know which variables had more than one value in the data sets, because singular matrices cannot be plotted. For this, we made a function that showed which variables could be plotted in a data set (see *Figure 5.1*).

```
def get_plottable_columns(df):
    for column in df:
        print column, ": ",
        n_values = len(df[column].unique())
        if n_values > 1:
            print "Yes, ", n_values, " values"
        else:
            print "No, 1 value"
```

Figure 5.1. Function that returns variables in a data set that can be plotted

For each data set, we used three types of visualization techniques:

- **Histograms:** this kind of plot gives a good overview on how the data are distributed in a data set, although it depends on the bins that are chosen, which are the intervals in which the data are classified into. This might not always be the best way to represent the data, as the whole of the data set is not plotted.
- **Heatmaps:** these are graphs that show the correlation between all the features in a data set, with a colour scale that represents a higher or lower correlation between the features.
- **Scatterplots:** this technique is very useful for studying the relationship between different features in a data set, showing a distribution of all the values of each feature. In this part, we included also the kernel density plots, which are quite useful to avoid overplotting, as they show where the values are more concentrated with different colour scales, and the marginal plots, which show both the distribution and the relationship between the features.

5.1. YMRS data set

5.1.1. Histograms

The first histogram we plotted was the one where we could see all the features distributed together, as seen in *Figure 5.2*.

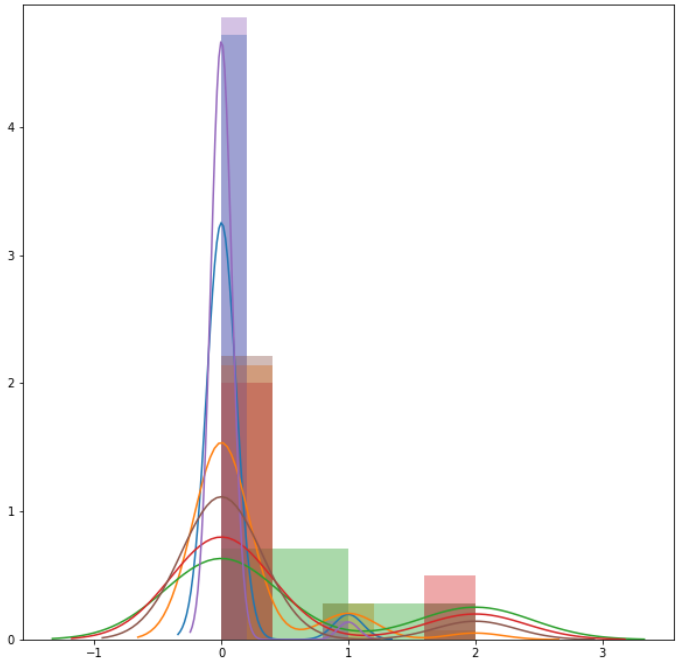

Figure 5.2. Feature distribution in the YMRS data set

This plot showed quite a similar distribution in all the features, oscillating between -1 and 3. The next step was to plot separate histograms for each feature, so we could observe how each one was distributed (see *Figure 5.3*).

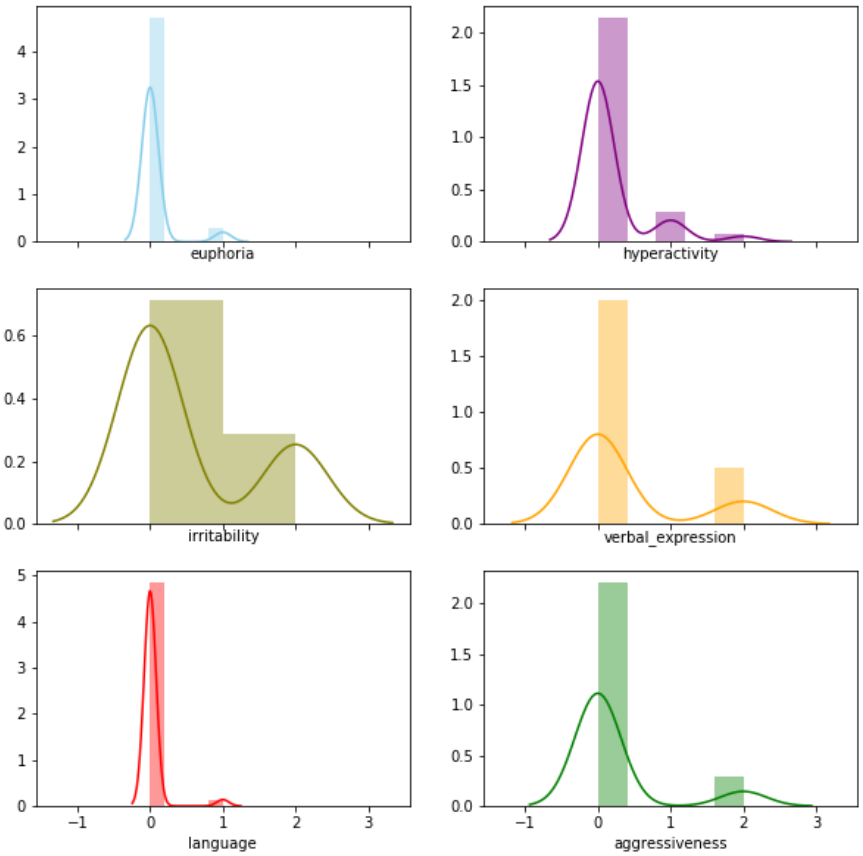

Figure 5.3. Distribution of each feature in the YMRS data set

Most of the columns had only two values in the data set, which meant that the data in the YMRS data set was very limited and would probably give quite bad results when the algorithms were tested.

5.1.2. Heatmap

We proceeded to plot a heatmap to see the correlations between the different features of the YMRS data set (see *Figure 5.4*). This helped us decide which features we could plot later with scatterplots. The heatmap shows the degree of correlation presented between two features, with black being the lowest level of correlation and white being the highest, with the associated degree of correlation, seen at the legend on the right side of the map.

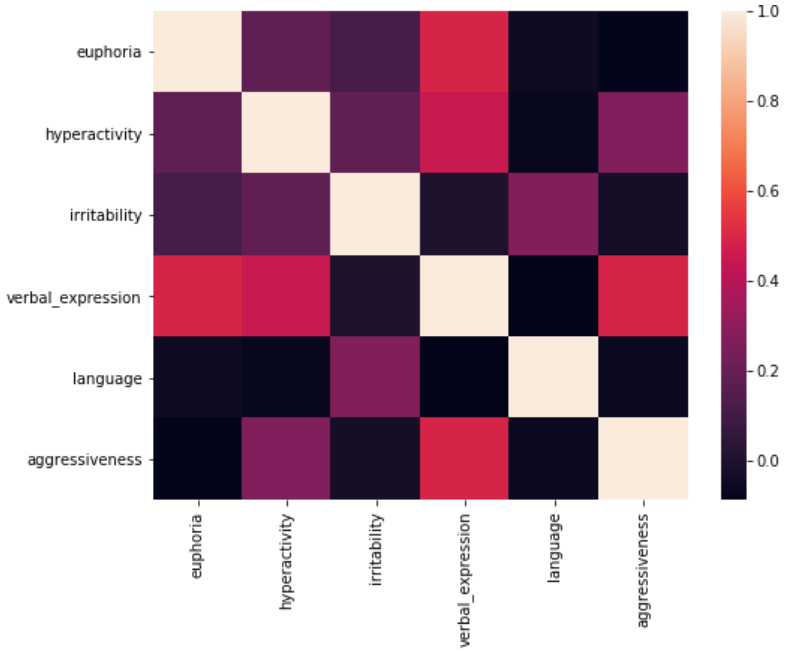

Figure 5.4. YMRS data set correlation heatmap

We saw that the features with the highest correlation were hyperactivity and verbal expression, although they were not highly correlated. It was also interesting to see that aggressiveness and verbal expression were correlated. This could mean that if the patient talks a lot (excessive speech rate), this behaviour would probably be accompanied by excessive energy or hyperactivity (Disruptive-Aggressive Behaviour).

5.1.3. Scatterplots

First, we started by plotting a matrix with all the features in the data set, as shown in *Figure 5.5*, in order to see if any features were worth plotting separately.

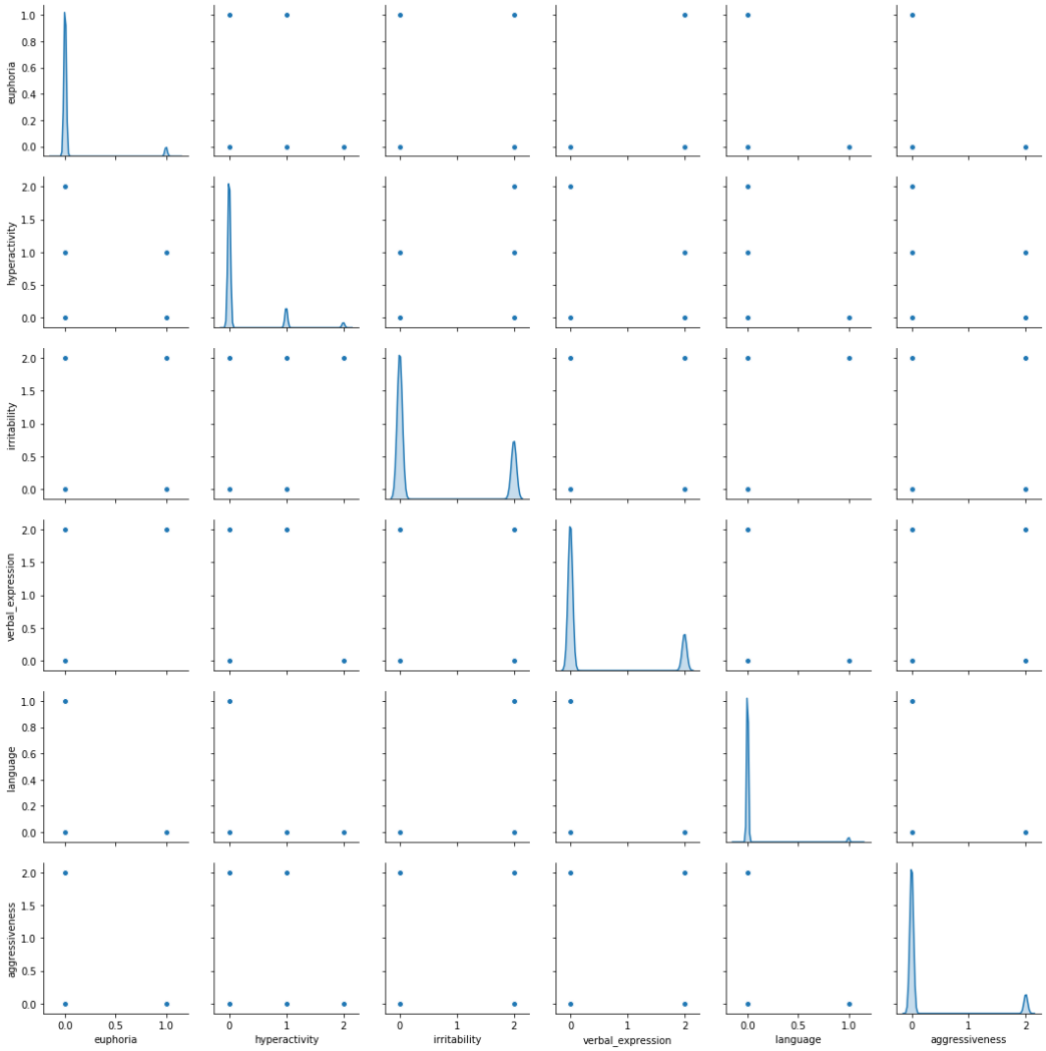

Figure 5.5. YMRS scatterplot matrix

With this scatterplot we could see that hyperactivity and irritability could have a similar distribution, as the data values had sort of an ascending form growing with the x-axis. To take a closer look at this relationship, we plotted a marginal plot (see *Figure 5.6*).

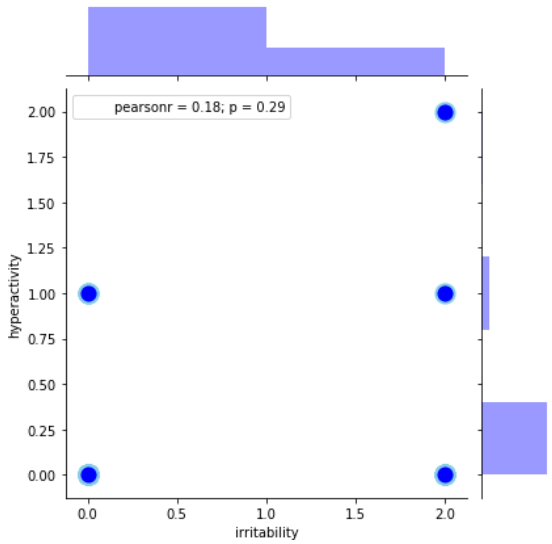

Figure 5.6. Marginal plot of hyperactivity and irritability in the YMRS data set

With such a small data set, it was very difficult to conclude that a relationship existed between these two features. In order to further visualize the relationship between them we made a 2D kernel density plot (see *Figure 5.7*).

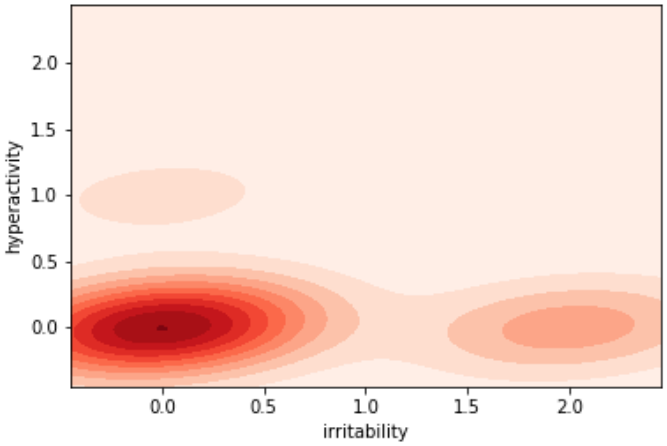

Figure 5.7. 2D kernel density plot of irritability and hyperactivity in YMRS data set

As we could observe, most of the data were concentrated around the *(0,0)* point. We couldn't clearly visualize a relationship between both features, most surely due to the lack of data.

The two other features that seemed to be correlated in the scatterplot matrix were verbal expression and euphoria, which also had an ascending distribution even though only three different values could be seen in the plot. We started by plotting a marginal plot, which showed us both the distribution and their values (see *Figure 5.8*).

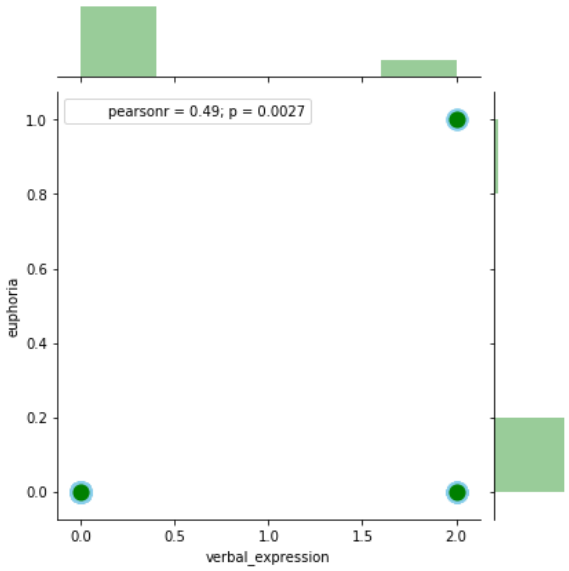

Figure 5.8. Marginal plot of verbal expression and euphoria in the YMRS data set

Both variables had a lot of values that were 0, so it was difficult to see a relationship between them in this plot. Like we did with hyperactivity and irritability, we plotted a 2D kernel density plot (see *Figure 5.9*) to see if we could conclude that a relationship existed between these two features.

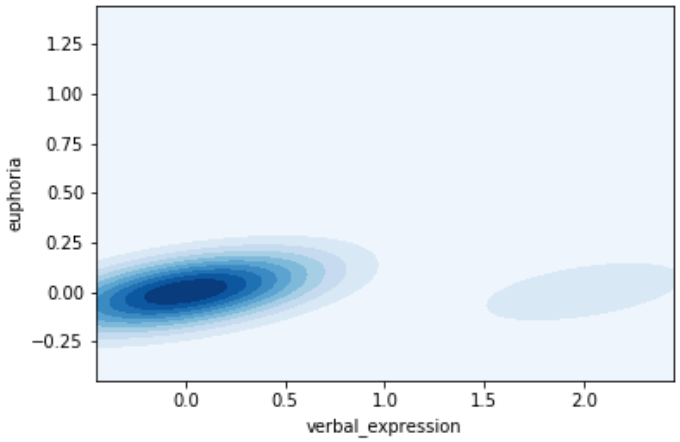

Figure 5.9. 2D kernel density plot of verbal expression and euphoria in the YMRS data set

There were some independent values of verbal expression around *(2,0)*, but it was possible to see a slight correlation between both features. This could mean that a higher rate of speech in the patient could be accompanied by a higher state of euphoria.

5.2. HDRS data set

5.2.1. Histograms

We started by plotting a histogram with all the features in it, in order to see if they were distributed equally (see *Figure 5.10*).

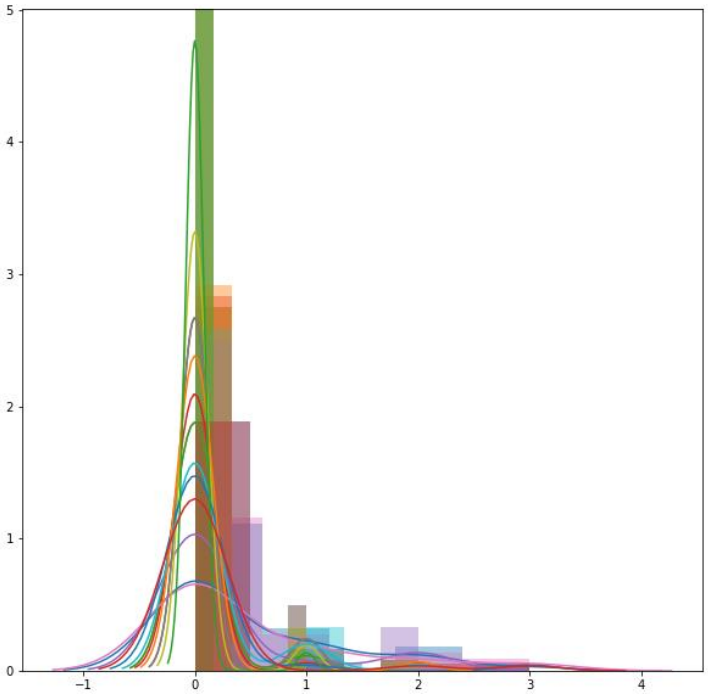

Figure 5.10. Feature distribution in the HDRS data set

The features in the HDRS data set were distributed in a similar way and all of them had a lot of values that were 0. We proceeded to plot the features in separate histograms to see how each of them was distributed (see *Figure 5.11*).

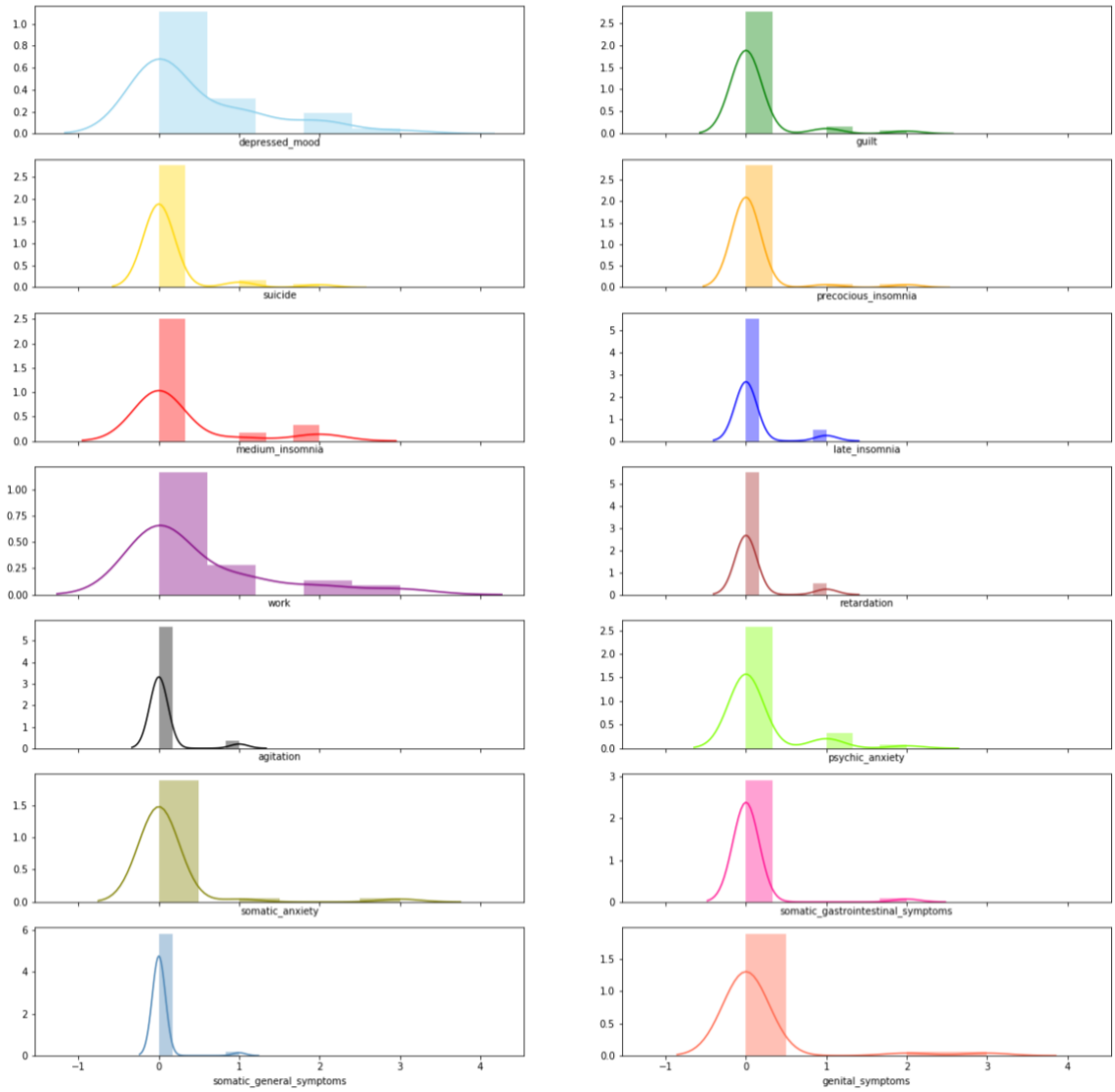

Figure 5.11. Distribution of each feature in the HDRS data set

The distribution curve was quite similar in all the features. The features that had a higher distribution were work and depressed mood, which had four different values. The rest of the features had at most three different values. Suicide and precocious insomnia (difficulty of sleep early in the night) had a very similar distribution, because the majority of the values were 0 and they had the same amount of data points concentrated where the value was 1 and 2, with no other values registered.

5.2.2. Heatmap

In order to see which features were correlated in the HDRS data set, we plotted a heatmap (see *Figure 5.12*). This is especially useful with data sets that have many features, like this one, as it gives a good overview of all the possible relationships between the features.

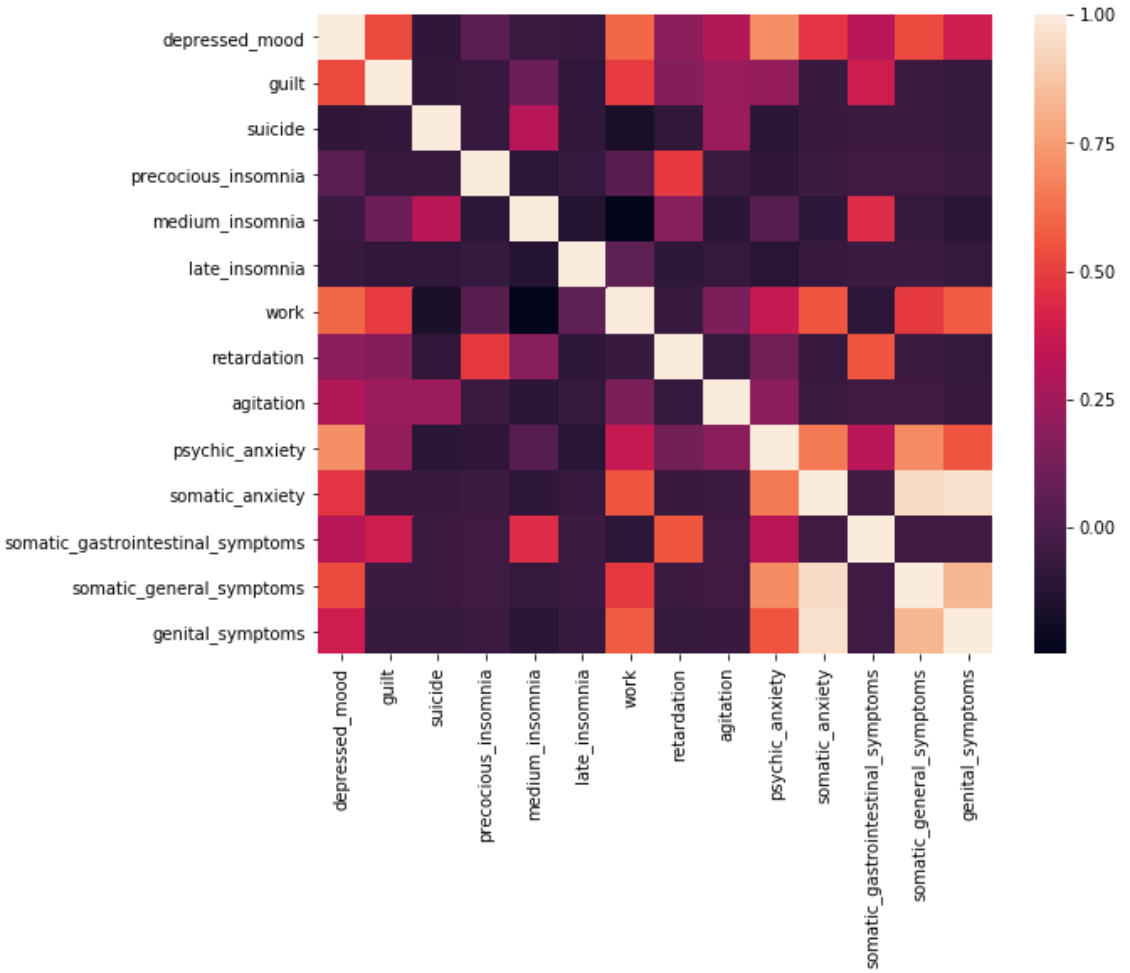

Figure 5.12. HDRS data set correlation heatmap

There were a lot of features that seemed to be correlated in this heatmap. Being black the lowest degree of correlation and white the highest, it was possible to see correlations between different features (with no particular order in the degree of correlation):

- ***Depressed mood* and *work*:** this relationship showed that the less a patient is willing to work or do other activities, the more depressed he or she will probably feel.
- ***Depressed mood* and *psychic anxiety*:** this correlation was quite interesting because it showed a relationship between the level of anxiety of a patient and a depressed mood. If a patient feels anxious, he or she will probably feel more depressed than usual.
- ***Depressed mood* and *somatic general symptoms*:** this relationship showed that if a patient is starting to feel heaviness in limbs, back or head (*General somatic symptoms=1*) or has clear-cut physical symptoms (*General somatic symptoms=2*), the probability of him or her having a more depressed mood is higher.
- ***Precocious insomnia* and *retardation*:** this meant that if the doctor starts to notice a slight or obvious retardation in the patient during an interview, the patient could probably be finding it difficult to sleep early in the night, or vice versa.
- ***Work* and *psychic anxiety*:** this relationship was also related to depressed mood, which probably meant that the three features were somehow correlated. If the

patient is less willing to go to work or do other activities, he or she will probably feel more anxious and depressed.

- ***Psychic anxiety* and *somatic anxiety*:** these two features are quite similar and showed a clear correlation. The marginal and density plots would help us confirm if this correlation existed.
- ***Psychic anxiety* and *somatic general symptoms*:** this showed us that if the patient is anxious, he or she will most surely have some kind of somatic symptom.
- ***Somatic general symptoms* and *genital symptoms*:** this relationship meant that a patient with loss of libido or menstrual disturbances will probably show other physical symptoms. We would also plot these two features with marginal and density plots in order to see if they were clearly correlated.

5.2.3. Scatterplots

As we did with the YMRS data set, we started by plotting a scatter plot matrix to see how the data points were distributed between different features. The matrix is too big to include as a figure, but it can be seen in the notebook that is associated to the project. The plot showed a quite similar distribution between depressed mood and psychic anxiety and it also showed a possible linear relationship between depressed mood and work. In order to further visualize this relationship, we made a marginal plot, as shown in *Figure 5.13*.

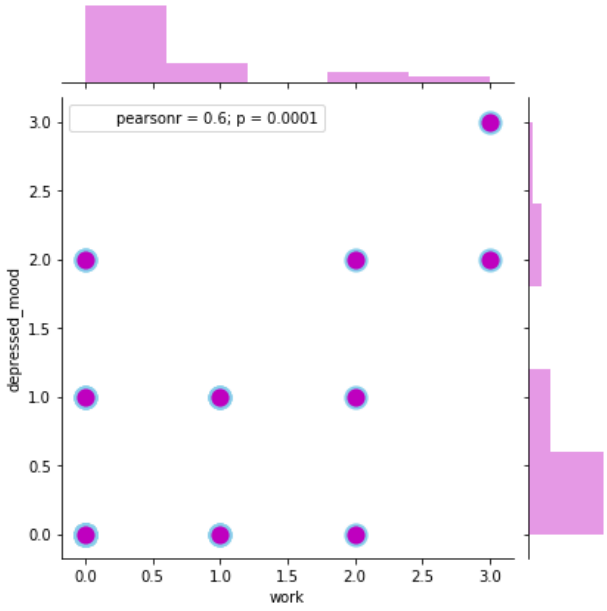

Figure 5.13. Marginal plot of depressed mood and work in the HDRS data set

As could be observed in the heatmap of *HDRS* features, depressed mood and work seemed to be correlated. We made a 2D density plot to confirm this correlation (see *Figure 5.14*).

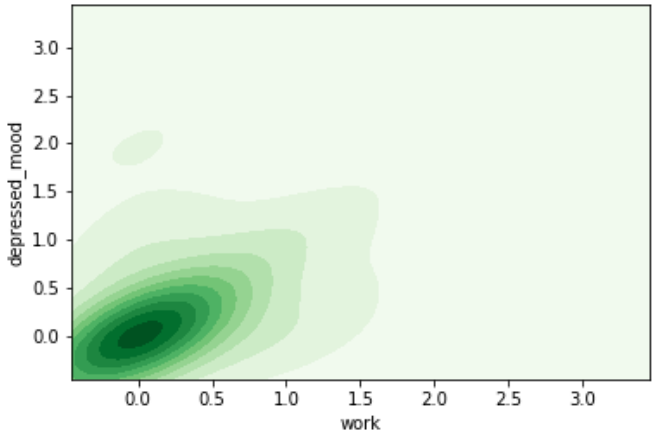

Figure 5.14. 2D kernel density plot of depressed mood and work in the HDRS data set

Although most of the values were concentrated between *(0,0)* and *(0,1)* on both features, they had a linear relationship, which showed that a decrease in a patient's activity could indicate that the patient is feeling depressed, or that the patient doesn't feel like working the more depressed he or she feels. This confirmed what we observed in the heatmap above.

We did the same thing with psychic anxiety and depressed mood, starting by plotting a marginal plot to see the distribution and the data points of both features (see *Figure 5.15*).

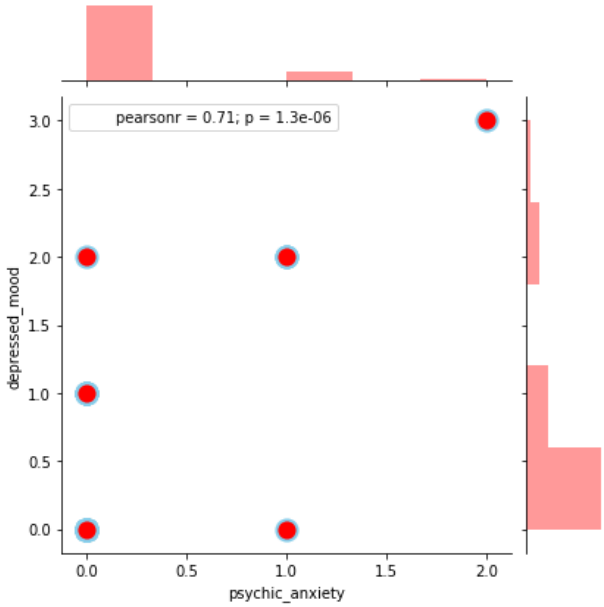

Figure 5.15. Marginal plot of depressed mood and psychic anxiety in the HDRS data set

There was a huge disparity in the distribution of the values, even if almost all the data points of depressed mood were recorded when psychic anxiety was 0. To see if there was any relationship at all between these two features we made a 2D density plot, as we did with the previous features (see *Figure 5.16*).

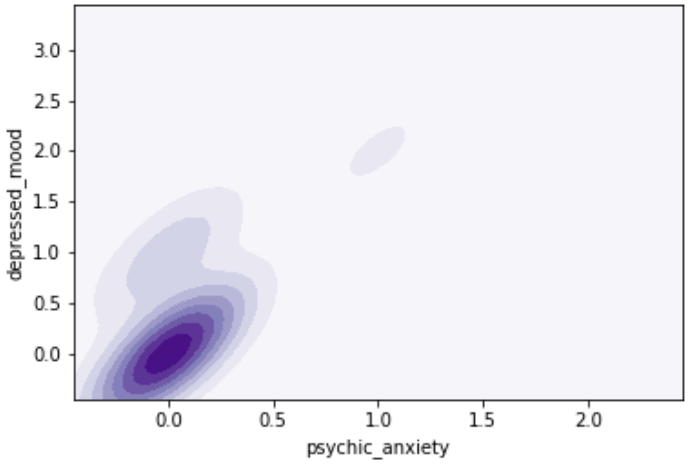

Figure 5.16. 2D kernel density plot of depressed mood and psychic anxiety in the HDRS data set

The plot didn't show a clear relationship between the features, so we decided to not make any assumptions on their possible correlation.

We plotted the same plots for somatic anxiety and psychic anxiety because we could see a certain correlation between them in the heatmap and because both features represented the general anxiety level of the patient. We started by plotting a marginal plot, which is shown in *Figure 5.17*.

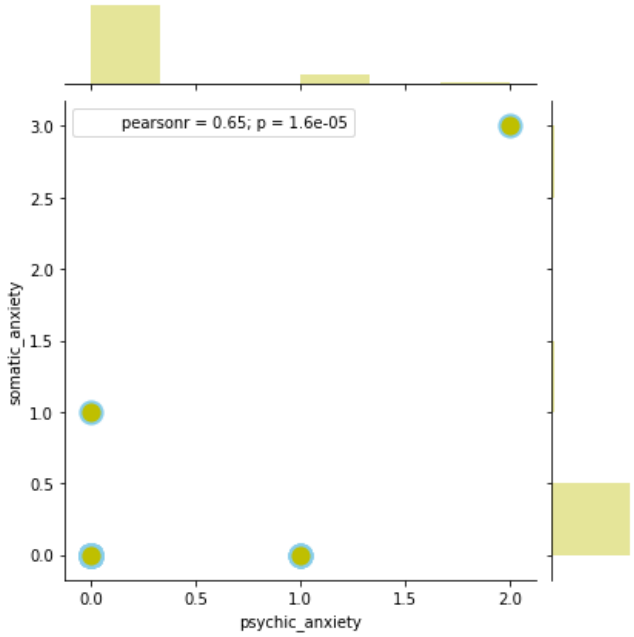

Figure 5.17. Marginal plot of somatic anxiety and psychic anxiety in the HDRS data set

Almost all the values of both features were 0, which complicated the task of trying to find a relationship between them. We needed to plot a 2D density plot (see *Figure 5.18*), which would help us deciding whether the features were correlated or not.

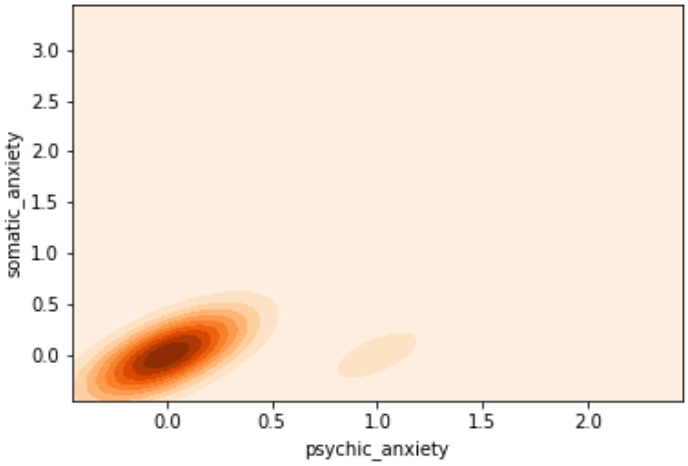

Figure 5.18. 2D kernel density plot of somatic anxiety and psychic anxiety in the HDRS data set

We observed that there was almost a linear relationship between these features which would be quite helpful when applying the different algorithms on the data because, for some algorithms, highly correlated features might lower the prediction accuracy.

5.3. Interview data set

Like we mentioned before, this data set contained a lot more data than the rest, which meant that it would be easier to see if there were any relationships between the features.

5.3.1. Histograms

We started by plotting a histogram with all the features together in order to see if the distribution of the data was similar (see *Figure 5.19*).

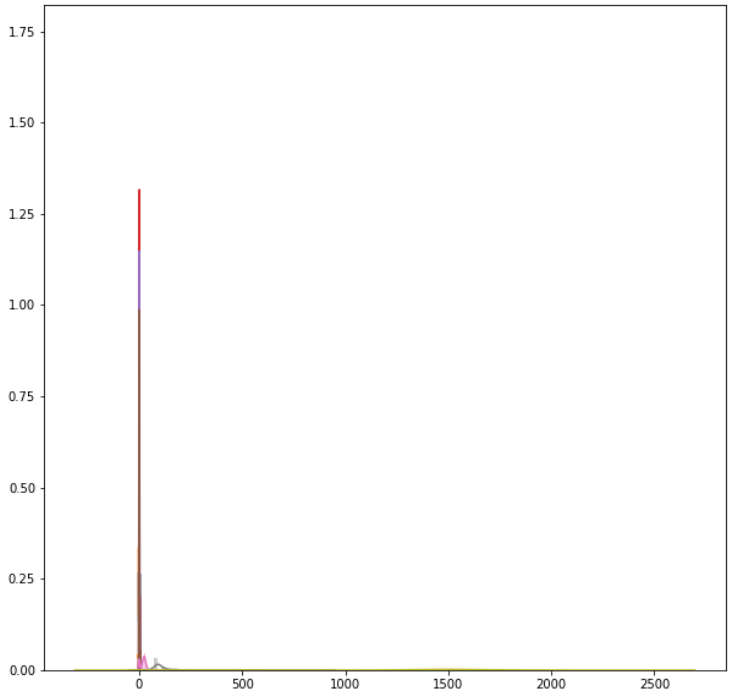

Figure 5.19. Feature distribution in the Interview data set

It was quite difficult to see how the data were distributed in the plot above. We rescaled the data so that it was easier to visualize the distribution. Rescaling means that the features are given a new value between a certain range (e.g. -1 and 1) so that they preserve their relationship but the values are much smaller, which reduces the spread of the data.

In order to scale the data, we used two scalers:

- **Standard Scaler:** this scaler assumes a normal distribution of the data. It calculates the standard deviation (SD), which is the amount of dispersion between the values, and the mean, by applying the following formula:

$$\frac{X_i - \ mean(X)}{SD(X)}$$

- **Min Max Scaler:** this scaler makes the data smaller by changing the range to *(-1, 1)*, because this data set includes both positive and negative values, and uses the following formula:

$$\frac{X_i - \ min(X)}{max(X) - min(x)}$$

The plot obtained with the Standard Scaler, shown in *Figure 5.20*, didn't really make the visualization easier because of the huge amount of data.

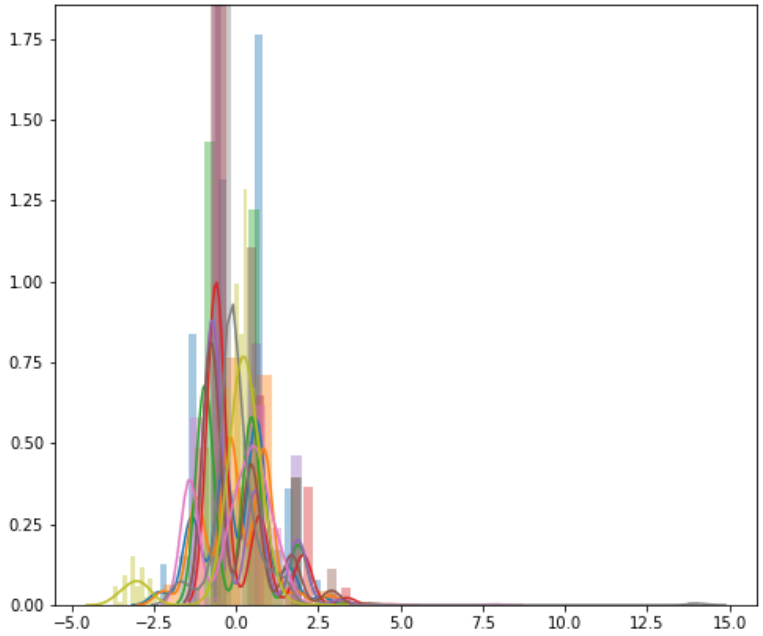

Figure 5.20. Distribution of the scaled Interview data using Standard Scaler

The Min Max Scaler generated a much cleaner plot (see *Figure 5.21*), where we could see that the data were distributed quite similarly and that it was mostly concentrated between *(0,0)* and *(0,2)*, showing a predominance of positive data.

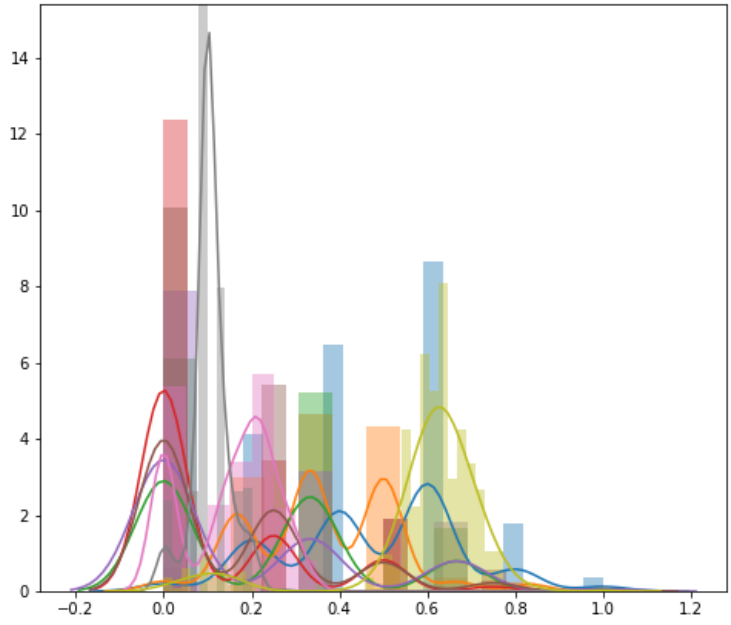

Figure 5.21. Distribution of the scaled Interview data using Min Max Scaler

In order to see how each feature was distributed, we started by plotting the features with the lowest data values, as shown in *Figure 5.22*.

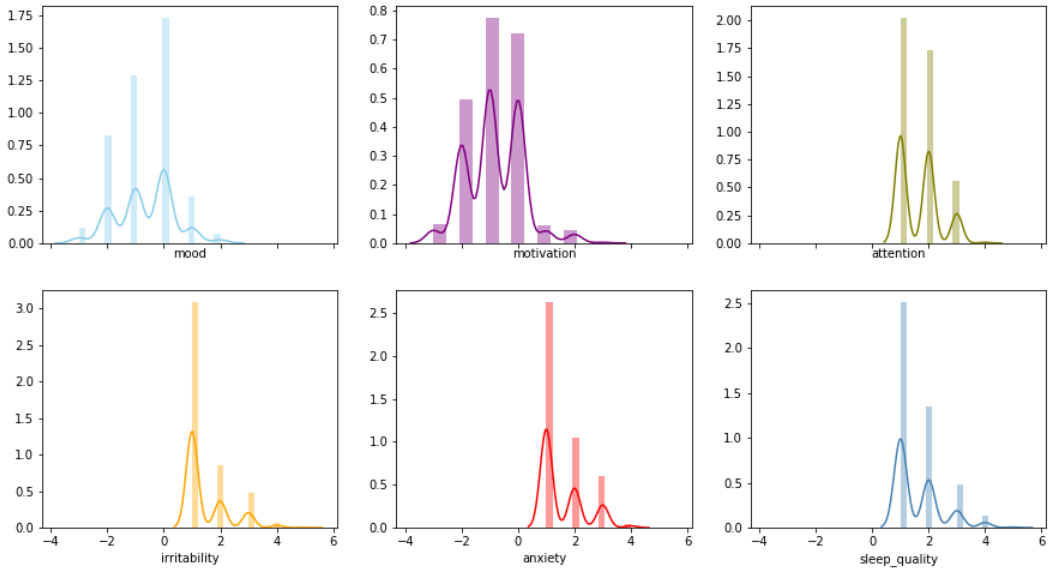

Figure 5.22. Distribution of features with lowest values in the Interview data set

In the above plot, we could see the distribution of the data properly. Mood and motivation had more negative values, ranging from -4 to 2. The rest of the features had more positive values, which ranged from 0 to 4.

After plotting the features with low data values, we proceeded to plot the active time (see *Figure 5.23*), which contained values ranging from 0 to 2400.

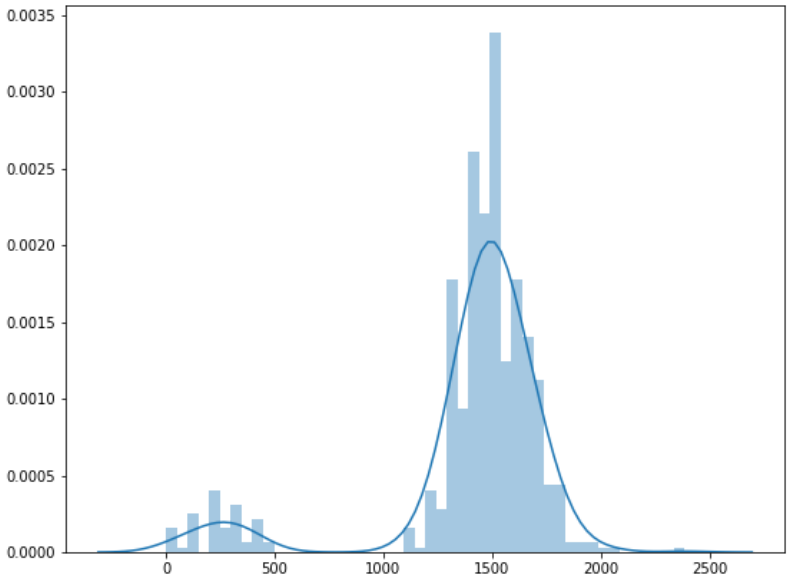

Figure 5.23. Distribution of the active time feature in the Interview data set

The above plot showed a distribution of the amount of time that the patients had been active. Two different groups could be seen, with one that had very little active time (from 0 to 5 hours) which could be data from just one patient or represent periods of time when the patients slept a lot. The next plot, shown in *Figure 5.24*, represented the amount of caffeine the patients ingested each day during the period of time when the interviews were made.

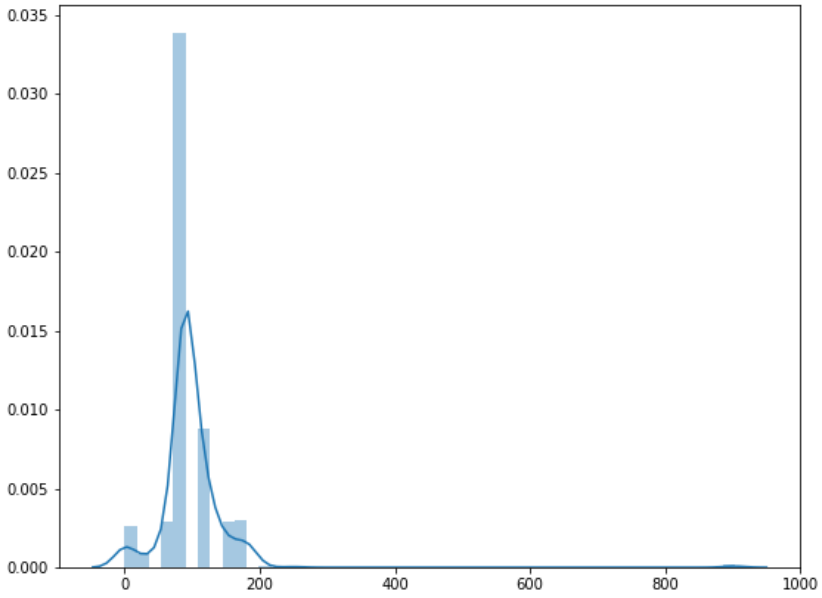

Figure 5.24. Distribution of the amount of caffeine in the Interview data set

In this plot, we could observe a slight curve between the values 800 and 1000, which meant that there could be an outlier there. A value higher than 800 had to be a writing error by the interviewer, because almost all the values of this feature ranged between 0 and 200. We checked if there were any values above 300 (see *Figure 5.25*), which was close to where the distribution curve goes down to 0.

```
print interviews_plot[interviews_plot.caffeine > 300]

     mood  motivation  attention  irritability  anxiety  sleep_quality  \
635     0           0          1             1        2              1
701    -1           0          1             1        1              3

     nr_cigarettes  caffeine  active_time
635             23       901         1455
701             19       903         1530
```

Figure 5.25. Checking if there are outliers in the Interview data set

There were in fact two values above 300 in the data set, specifically 901 and 903 which, as we mentioned above didn't make any sense if we compared them to the rest of the caffeine values in the data set. We proceeded to change the values to the same values as the rows before and after (see *Figure 5.26*), because outliers could affect the prediction with such a small amount of data as we had.

```
print "First outlier (635):"
print interviews.at[634, 'caffeine']
print interviews.at[636, 'caffeine']
print "Second outlier (701):"
print interviews.at[700, 'caffeine']
print interviews.at[702, 'caffeine']

First outlier (635):
90
90
Second outlier (701):
90
90

interviews = interviews.set_value(635, 'caffeine', 90)
interviews = interviews.set_value(701, 'caffeine', 90)
```

Figure 5.26. Changing the value of the outliers in the Interview data set

After changing the values of the outliers, we plotted the nr_cigarettes feature (see *Figure 5.27*), which represents the number of cigarettes smoked by the patients each day.

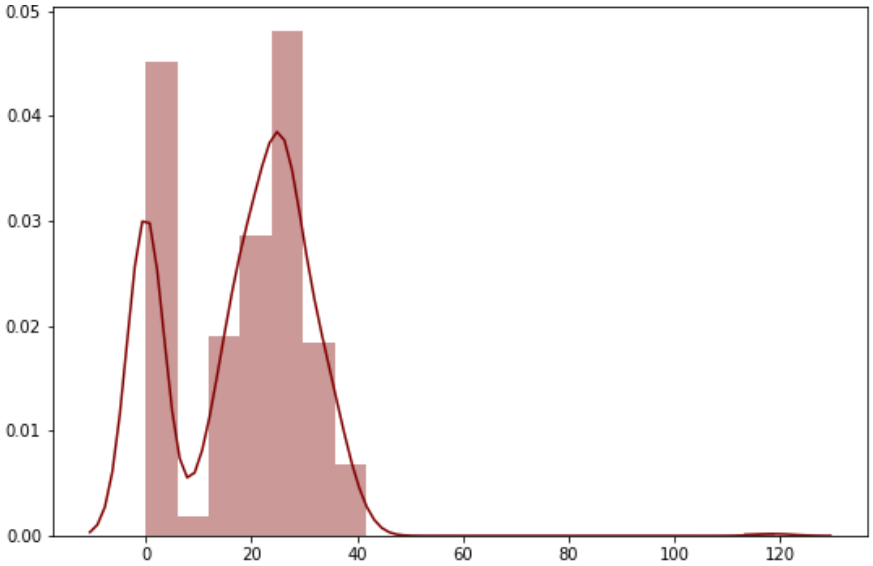

Figure 5.27. Distribution of the number of cigarettes in the Interview data set

The above plot showed that the number of cigarettes smoked by the patients were distributed between 0 and 40. We could also observe a slight curve between 100 and 120, which meant that a patient smoked between 100 and 120 cigarettes in a day, which had to be an error in this case too. We removed those values like we did with the caffeine values that were above 300, because they could affect the accuracy of some algorithms.

5.3.2. Heatmap

Plotting a heatmap of the features in the Interview data set would give us a good overview of any possible correlation between features, as seen in *Figure 5.28*.

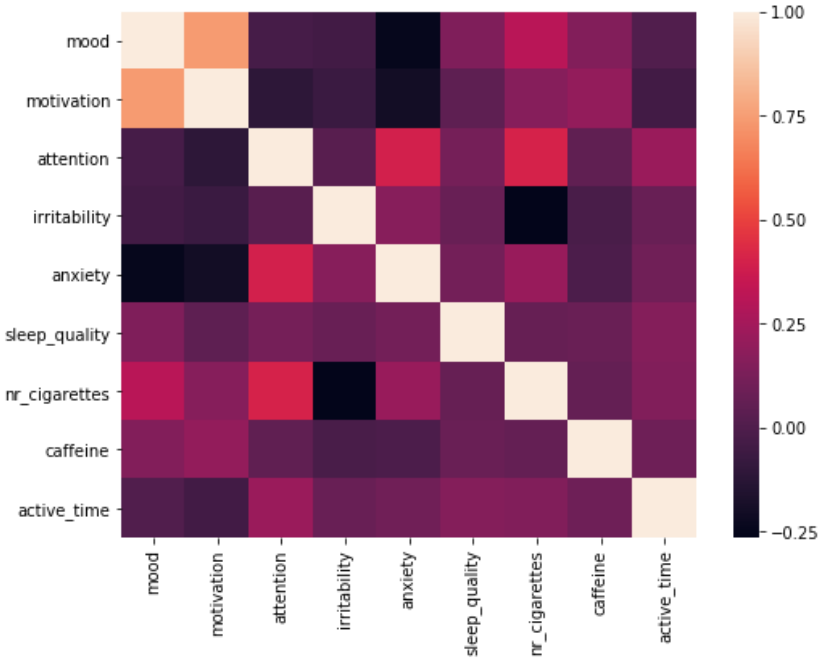

Figure 5.28. Interview data set correlation heatmap

The features that seemed highly correlated were mood and motivation. This made sense, because usually feeling unmotivated is related to a feeling of depression. The scatterplots would help confirming whether a relationship between these two features existed at all.

5.3.3. Scatterplots

As we mentioned before, the best way to see if there were any relationships between the features in the data set that were worth considering was a scatterplot matrix (see *Figure 5.29*).

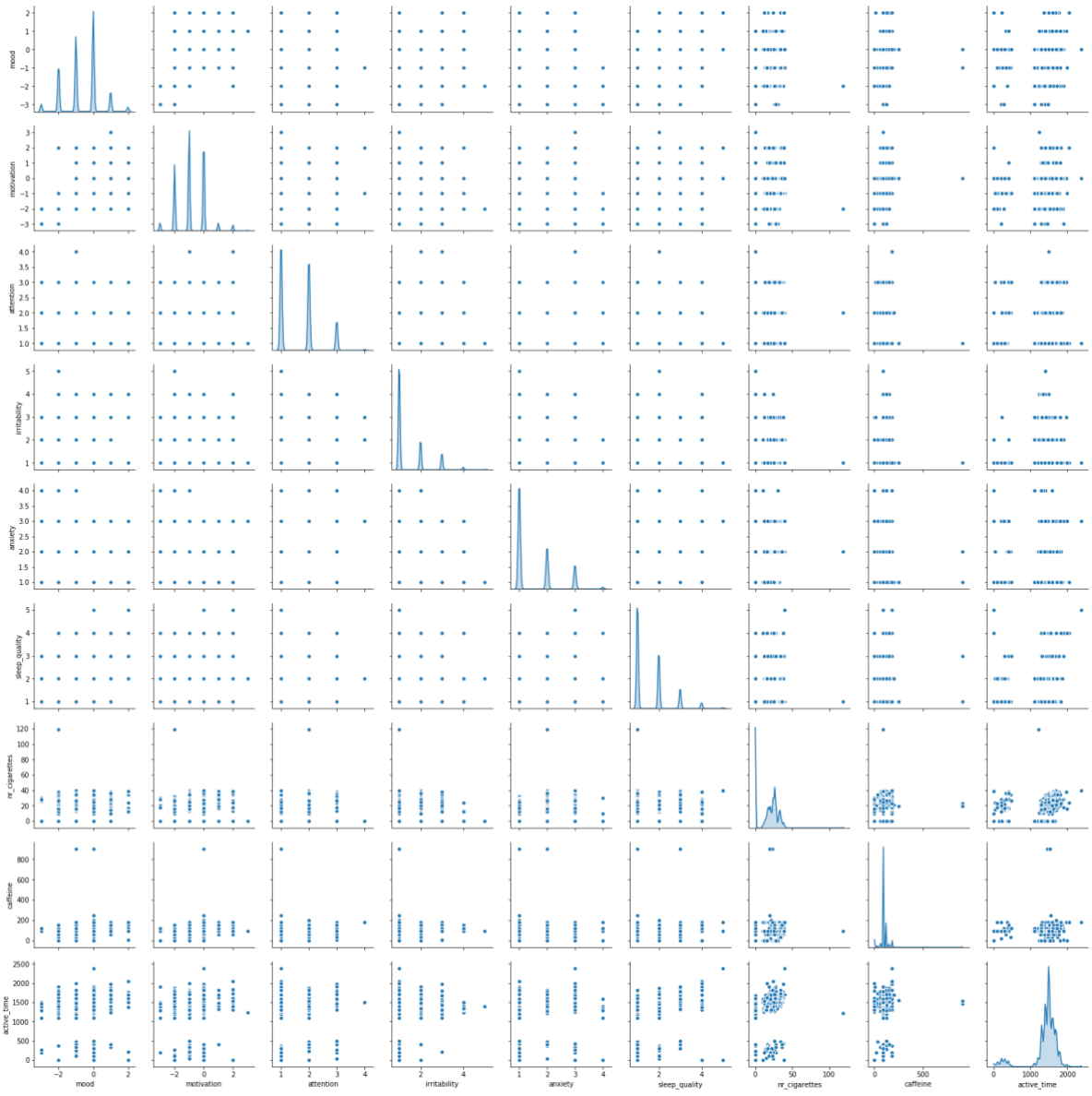

Figure 5.29. Interviews scatterplot matrix

This plot showed a possible linear relationship between mood and motivation, just as we saw in the heatmap above. We could also observe a relationship between caffeine and motivation. It was interesting to see the distribution between anxiety and sleep quality too, which showed a possible linear relationship.

There were some relationships which we didn't consider relevant like the relationship of active time with caffeine and nr_cigarettes, because it was quite obvious that the more time a patient is active, the more caffeine he or she will ingest and the more cigarettes he or she will smoke.

In order to further visualize the relationship between mood and motivation, we plotted a marginal plot which showed the data points distributed in both axes and the distribution of the values that both features presented in the data set (see *Figure 5.30*).

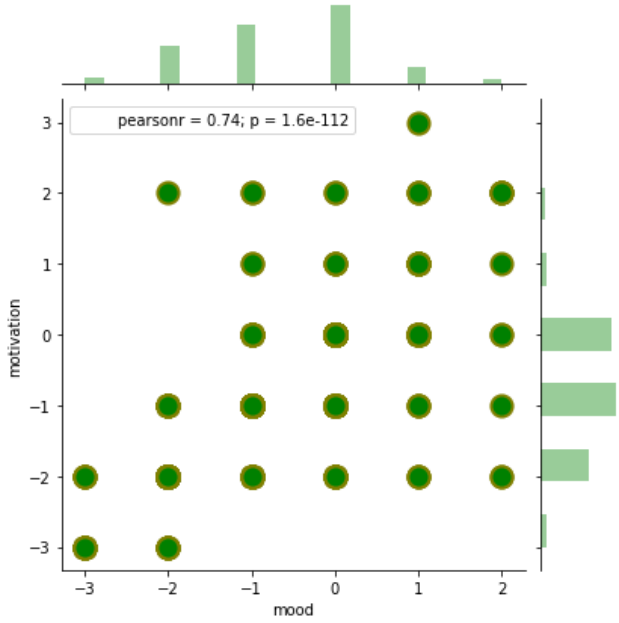

Figure 5.30. Marginal plot of motivation and mood in the Interview data set

A certain similarity could be seen between how the data was distributed and the values of the features in the axes. The data points for both features were mostly negative, as we saw in the histogram, and were concentrated between *(-3, -3)* and *(0,0)*. Plotting a 2D density plot (see *Figure 5.31*) would help us see where the data were mostly concentrated in the axes.

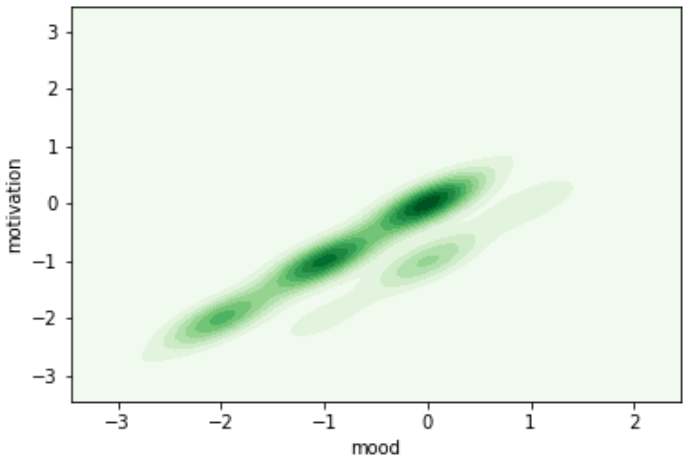

Figure 5.31. 2D kernel density plot of motivation and mood in the Interview data set

There was a clear linear relationship between mood and motivation, as seen in the density plot. The data were mostly concentrated in the points where both variables had the same value, and the data points that were mostly represented were the ones where both features were 0, where the colour was much darker.

5.4. Intervention data set

This data set contained only two features, GAF and relief, so it was not necessary to plot a heatmap or a scatterplot matrix to visualize any relationships between them.

5.4.1. Histograms

After checking that both features had different values in the data set, we could proceed to plot a histogram which showed the distribution of the data, as seen in *Figure 5.32*.

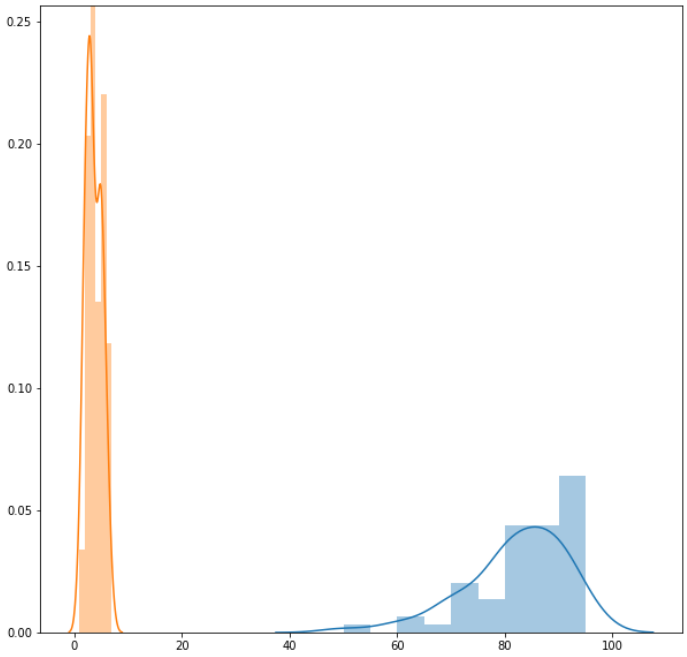

Figure 5.32. Feature distribution in the Intervention data set

As observed in the histogram, there was a huge disparity in the distribution of the data, and both features weren't in the same scale. Both features had a normal or Gaussian distribution, so we used the Standard Scaler to put them in the same scale, just as we did with the Interview data set. If we plotted them again after rescaling them, we could see if they were distributed in a similar way (see *Figure 5.33*).

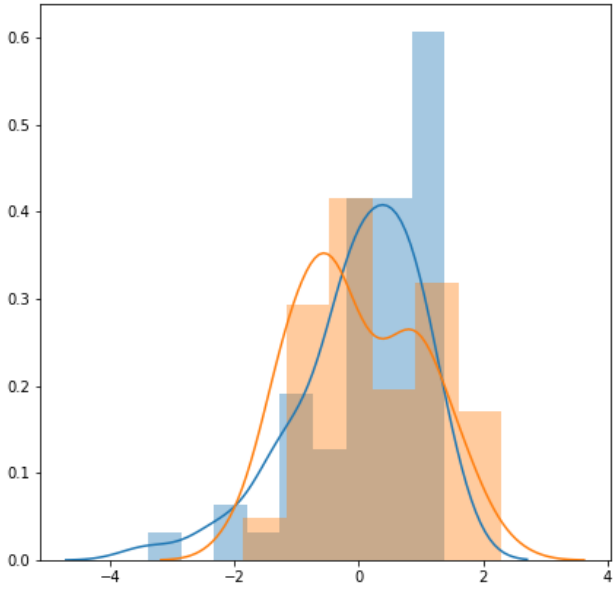

Figure 5.33. Distribution of the scaled Intervention data using Standard Scaler

This new plot showed us that the orange histogram (relief) and the blue histogram (GAF) had a similar distribution in this data set, although the distribution curve tended more towards negative values in the relief feature, which meant that there were more data points where the patients felt less relief.

5.4.2. Scatterplots

As we mentioned above, it was not necessary to plot a scatterplot matrix because the data set only contained two features. We started by plotting a marginal plot of the scaled data set with both features (see *Figure 5.34*).

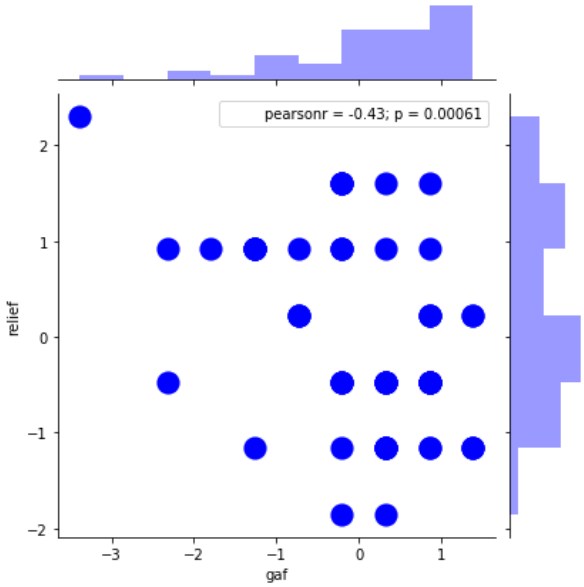

Figure 5.34. Marginal plot of relief and GAF in the Intervention data set

We couldn't find a clear relationship between the features in this plot, but it was possible to see that the relief values were more equally distributed along the axis, and the majority of the GAF values were concentrated in the highest possible value (the distribution grew in a linear way).

With a 2D density plot (see *Figure 5.35*) we would be able to see if there was any correlation between these features because it obviates values that don't have a high representation in the data set.

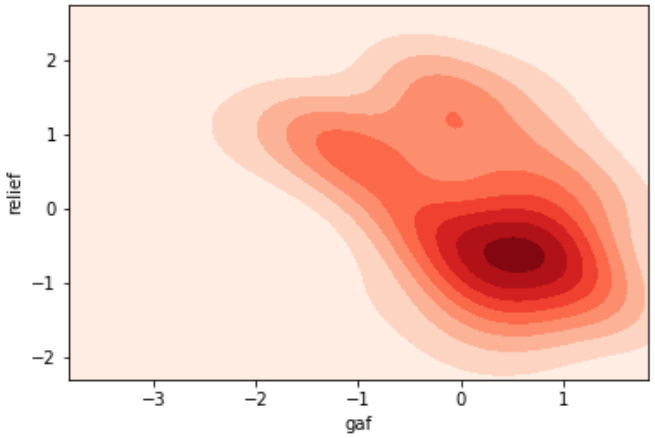

Figure 5.35. 2D kernel density plot of relief and GAF in the Intervention data set

This plot showed that relief and GAF were not correlated, and that there was a higher concentration of negative values of the relief feature, like we saw in the histogram.

6. Data combination

This part of the project represents the process of combining the data in different ways to make new data sets with different combinations of features. The goal was to find data set combinations that had enough data for the algorithms to process, so we could later see which data sets returned the highest accuracy.

In order to get the combinations right, we defined a function (see *Figure 6.1*) that obtained the date of each entry and compared it with the different episodes of depression and mania in the Episode data set, which was the target of the prediction, i.e. the value that the algorithms would try to predict.

```
def checkEpisode(date, patient):
    episode = 'N'
    ep = episodes.loc[episodes['patient'] == patient]
    for index, row in ep.iterrows():
        if date >= row.start and date < row.end:
            episode = row.episode
    return episode
```

Figure 6.1. Function that checks the episode a patient is in with a given date

For the entries or rows that were not recorded in the Episode data set, we assumed that the patient was in a euthymic state. This way, we got three possible states that a patient could be in (Depression → D, Mania → M and Euthymic/Neutral → N).

In this part, we also plotted some of the correlated features that we found in the data visualization section with scatterplots, which would help us see how the data was distributed depending on which state the patient was in. The structure of the data combination is similar to the previous parts of the project, with a section for each combination.

6.1. YMRS and Episode data sets

First, we merged the data sets with help of the function that returned the episode with the date of each row and created a new column in the YMRS data set, which we named episode. After we merged them, we dropped the code and date columns, as they were no longer needed (see *Figure 6.2*).

```
young_episodes = young.copy()
for index, row in young_episodes.iterrows():
    young_episodes.at[index, 'episode'] =
                checkEpisode(row.date, row.code)

young_episodes = young_episodes.drop('code', 1)
young_episodes = young_episodes.drop('date', 1)
```

Figure 6.2. Merging the YMRS and Episode data sets

Having merged the data sets, we could plot the features that were correlated in the YMRS data set. We found that expression and euphoria had a slight linear relationship, so we proceeded to plot them (see *Figure 6.3*) to see if there were any patterns on the distribution depending on which state the patients were in.

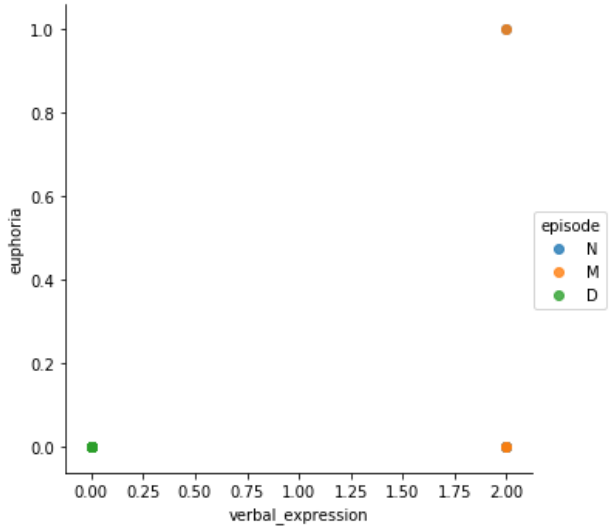

Figure 6.3. Distribution of verbal expression and euphoria on different patient states in the YMRS data set

Partly because the points can overlap each other in this kind of plot and because the data were very scarce, it was difficult to see if these features were distributed differently depending on the state of the patients.

6.2. HDRS and Episode data sets

With the HDRS data set, we did exactly the same as we did with the YMRS data set, i.e. merging it with the Episode data set. After dropping the code and date columns, we proceeded to plot the variables that we found were correlated in the Exploratory Data Analysis. We started by plotting depressed mood and work, as shown in *Figure 6.4*.

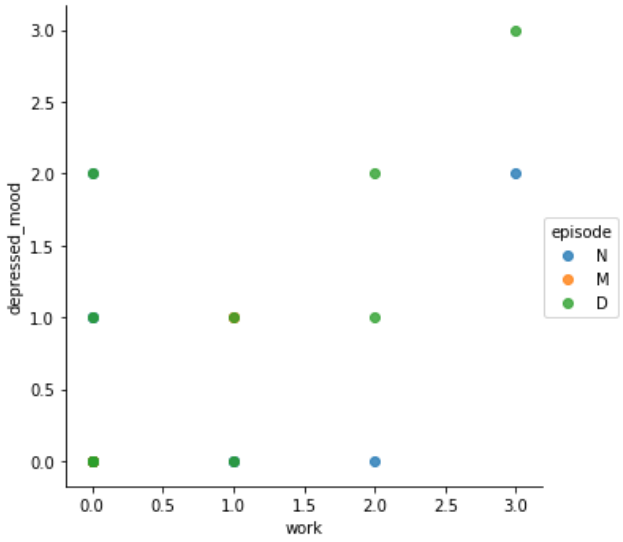

Figure 6.4. Distribution of depressed mood and work on different patient states in the HDRS data set

Even though we could only see the last painted points, taking a closer look at the plot showed that the contour of some of the green points was orange (like the *(1,1)* point), which represented mania episodes in this case. We could see that when the patients had a depression episode, the value of depressed mood was much higher, almost always between 1 and 3, which meant that they either spontaneously reported feeling depressed or they communicated feeling depressed in a non-verbal way.

We could also see that when the patients were in a depression state (because the majority of green points were recorded between 1 and 3 in the work axis), they started feeling loss of interest in activities they usually performed, or there was a decrease in the time spent in work and activities, which made perfect sense according to the HDRS rating scale.

The two other features that we found out were correlated in the Exploratory Data Analysis were somatic anxiety and psychic anxiety. We plotted them too in order to see how the different episodes were distributed (see *Figure 6.5*).

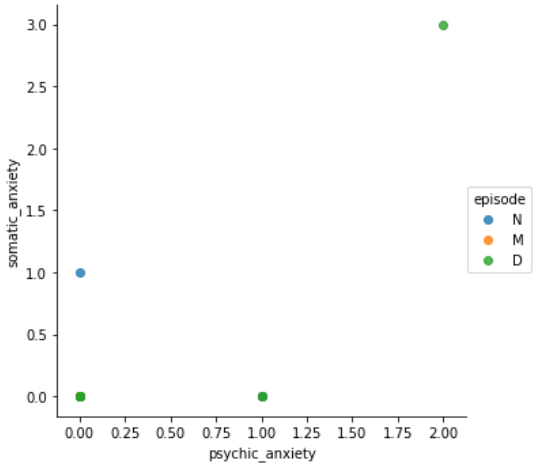

Figure 6.5. Distribution of somatic and psychic anxiety on different patient states in the HDRS data set

There was no clear pattern in the distribution of these two features depending on the states of the patient, so we couldn't make any valid conclusions with this plot.

6.3. Interview and Episode data sets

As we previously did with the YMRS and HDRS data sets, we merged the Interview data set with the Episode data set by iterating over each element and saving the episode the patient was in. We dropped the patient and date columns as we no longer needed them. The features that were correlated in the Interview data set were mood and motivation. We plotted a scatter plot like we did with the previous features (see *Figure 6.6*), in order to see the distribution on different episodes.

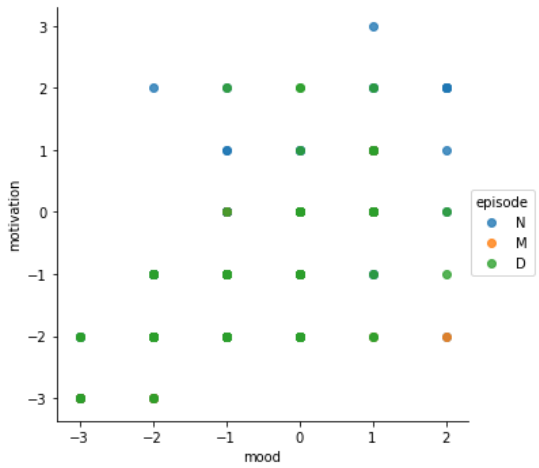

Figure 6.6. Distribution of mood and motivation on different patient states in the Interview data set

There were no clear patterns that identified the different episodes with this plot, but we made a barplot of the features in order to see the distribution based on the different episodes, as we had a lot more data in this data set (see *Figure 6.7*). The bars are a representation of the mean of the y-axis feature, in this case motivation, in each of the values of the x-axis feature (mood).

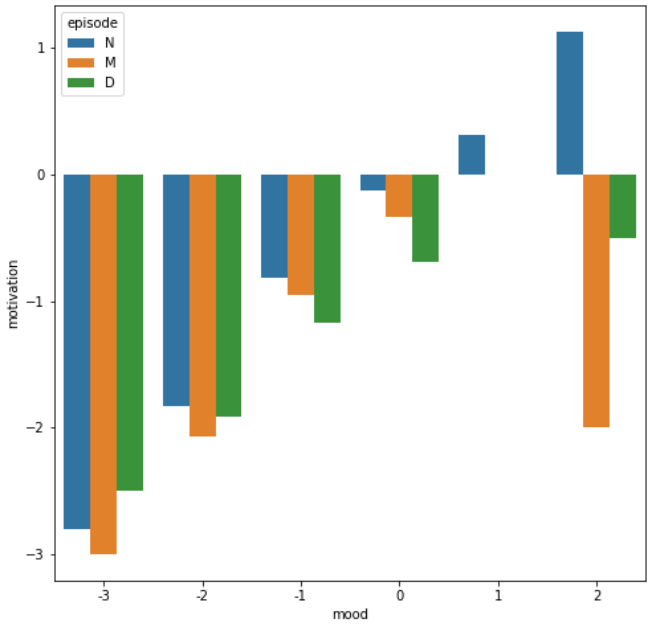

Figure 6.7. Mood and motivation barplot with patient states in the Interview data set

This plot showed a very similar distribution in all the combinations of mood and motivation in the negative values (mood level below 0 and average of motivation below 0). The only difference we could observe was in the positive values of mood, i.e. mood > 0, which showed that when the patient was in a euthymic state (blue), if he or she was in a positive mood, they would probably be motivated too (see the blue bar farthest right, which has an average of 1 in motivation while the other states had negative values in average).

By making the same plot for sleep quality and anxiety (see *Figure 6.8*), we would be able to see if we could consider that they were representative features for defining the state of a patient.

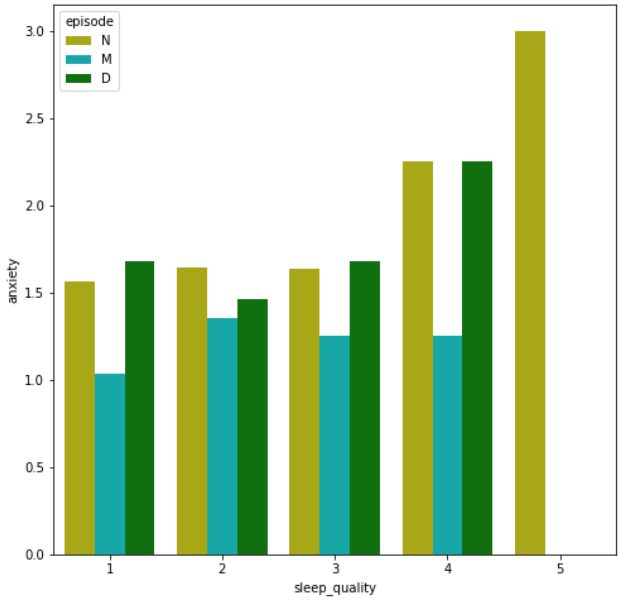

Figure 6.8. Anxiety and sleep quality barplot with patient states in the Interview data set

These two features were clearly not representative for the state of a patient, because the distribution was very similar in all the possible combinations of values. There were no values for anxiety when the sleep quality was 5 for the states of mania and depression, which could either mean that there were not enough data or that the patients going through a mania or depression episode never had the highest level of quality when they slept.

6.4. Intervention and Episode data sets

With the Intervention data set we did the exact same process of adding the episode to each entry of the Intervention data set and we dropped the code and date columns. We saw that GAF and relief did not have any relationship at all during the Exploratory Data Analysis, but we decided to make a barplot in order to check if the distribution of the data was different on each episode of the patients (see *Figure 6.9*).

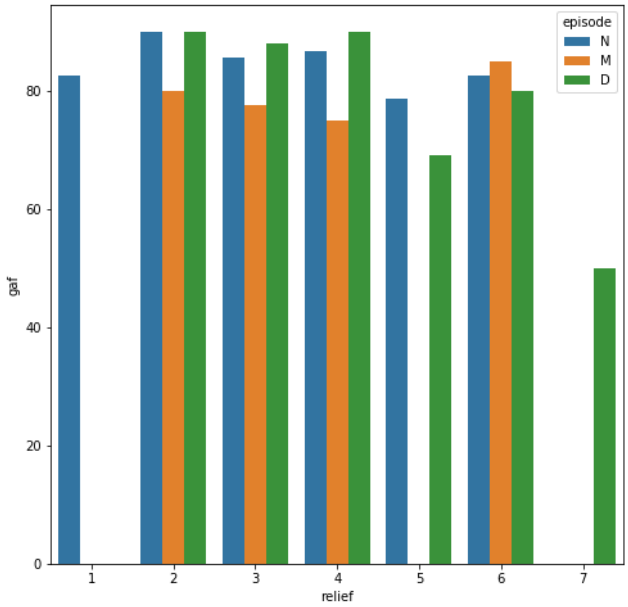

Figure 6.9. GAF and relief barplot with patient states in Intervention data set

The mean was almost the same in 4 out of 7 values of the levels of relief in the patient, which meant that the relief values between 2 and 6 were not representative of the episode a patient is in. We could see that there were no values for patients going through a mania episode in the values 1, 5 and 7 of relief, which meant that they never report extreme relief (much better or much worse).

It was useful to see that the only values recorded for level 7 of relief (feeling much worse) were from patients in a depression state, which meant that the doctor should worry when the patient reports feeling much worse, as it could indicate the start of a depression episode.

6.5. YMRS and HDRS data sets

The YMRS data set had fewer values than the HDRS data set and therefore we started by iterating over it and getting the data from the entries in the HDRS data set. In order to find the entries with the same date, we defined a function that checked if there was an entry in a data set with a given date and, if it existed, returned the index of that entry, as shown in *Figure 6.10*. This means, that for each date in the YMRS data set, we checked if it existed in the HDRS data set too by iterating over it, so that we could be able to combine the features of the rows in both data sets into a same row of the combined data set.

```
def date_in_df(date, df):
    n = None
    for index, row in df.iterrows():
        if date == row.date:
            n = index
    return n
```

Figure 6.10. Function that checks if there are any entries with the same date as the one passed as argument

After initializing the empty columns in the YMRS data set, we proceeded to combine it with the HDRS data set by iterating over every entry and checking the date in the HDRS data set, as seen in *Figure 6.11*.

```
for column in hamilton.columns:
    if(column != 'code' and column != 'date'):
        ymrs_hdrs[column] = np.nan

for index, row in ymrs_hdrs.iterrows():
    n = date_in_df(row.date, hamilton)
    if n != None and hamilton['code'][n] == row.code:
        for column in hamilton.columns:
            if(column != 'code' and column != 'date'):
                ymrs_hdrs[column][index] = hamilton[column][n]
```

Figure 6.11. Combination of YMRS and HDRS data sets

After combining the data sets, we obtained a data set that had some empty values, as not every entry on the YMRS data set also existed in the HDRS data set. The best option was to drop the rows that had NaN or empty values because it is not a good practice to completely make up new data.

After combining the data sets, we checked the episode in which each patient was in on each entry with the same function that we used with the previous data sets, and we dropped the date and code columns. This process left us with a data set of 25 entries, which was too small to get good results, but we would test anyways as it was the goal of this project to apply different algorithms on the given data.

6.6. Interviews and Interventions

We combined these two data sets in the exact same way as we did with the YMRS and HDRS data sets, by iterating over the Intervention data set which has fewer values than the Interview data set. After combining the data sets, we added an entry with the episode the patient was in, like we did with the previous data sets, and we dropped the code and date columns.

This process also generated a data set with very few entries, in this case 14, but it would be used for testing too for the same reason as the combination of the YMRS and HDRS data sets.

7. Application of the algorithms

The two most common types of Machine Learning that are used nowadays are supervised learning and unsupervised learning.

The supervised learning technique consists in creating a model, with the help of an algorithm, which will predict an output for a certain input. It often counts with a set of data for which the input and output are known. This set of data is called the training set.

Supervised learning problems can be of two types, regression and classification. Regression is a problem in which the output is quantitative, like predicting stock prizes based on previously obtained results. Classification is a problem in which the output is qualitative. The problem that is presented in this project is a classification problem, as it aims to predict whether the patient is having a mania or depression episode, or is staying in a euthymic state.

Unsupervised learning is used when the output is not known for the input values. The main type of problem in unsupervised learning is clustering, which consists in grouping the input data in clusters. Clusters are huge collections of data that have similar features [21].

In this part, we tested different classification algorithms on the data sets that we had previously obtained. The target or value that we tried to predict was the state a patient is in (Depression, Mania or Euthymic), so all the data sets had to have a column with the episode associated to the rest of the features, which we ensured in the data set combination section. The data sets on which we tested the algorithms were:

- **YMRS:** Young Mania Rating Scale data (young_episodes).
- **HDRS:** Hamilton Depression Rating Scale data (hamilton_episodes).
- **Interviews:** interview data (interviews_episodes).
- **Interventions:** intervention data (intervention_episodes).
- **YMRS-HDRS:** combination of the YMRS and HDRS data (ymrs_hdrs).
- **Interviews-Interventions:** combination of the interview and intervention data (interviews_interventions).

With such a small amount of data as was available for the project, it was almost impossible to avoid underfitting of the models. This occurs when the model has a very high training and testing error, because it fails to learn the possible relationships that can exist between the data.

Even if the risk of underfitting existed, we still tried testing which algorithms performed well, so that we could discard the ones that performed poorly for when they were to be applied to larger amounts of data.

The algorithms that we used for this part are: *Decision Tree* [22], *Random Forest* [23], *Support Vector Machines* [24] and *Logistic Regression* [25]. The reason why these algorithms have been chosen for this project is explained down below in the section that corresponds to each algorithm.

The fact that these algorithms have been applied on this project doesn't mean that they are the best option for the classification of Bipolar Disorder states, but rather that they are the most suitable ones given the amount of data and number of features used in the project. In future studies that make use of this project, if the data sets are larger, it could be interesting to apply other algorithms too, like the Naïve Bayes [26] algorithm or any kind

of Boosting algorithm [27], as to see how they perform on this particular classification problem.

Before applying each algorithm, as is necessary for every classification problem, the original data needed to be split into a training and testing set (see *Figure 7.1*). Later, the training data would be used to train the prediction models and the testing data would be used to compare the output of the model with the real targets by cross-validation, a technique presented first by M. Stone in 1974 [28] that is used widely in Machine Learning for algorithm performance comparison [29].

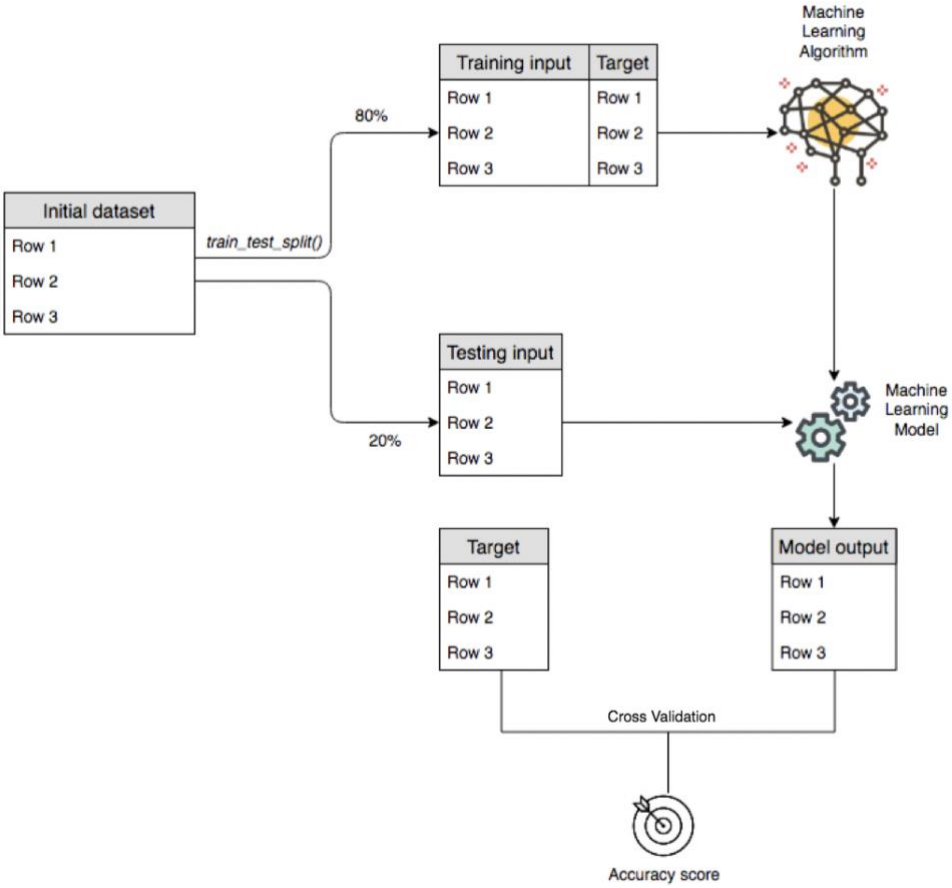

Figure 7.1. Diagram of the Machine Learning algorithm application process

The testing set is used for obtaining the accuracy of the model, as mentioned above, which is done by comparing the output obtained from the testing input and the real output of the testing set. In order to divide the original data sets into training and testing sets, we used the *train_test_split()* function from the scikit-learn library [12], an example of which can be seen in *Figure 7.2*, where the test size represents the percentage of the data that is used for the testing set (in this case 30%).

```
X_train, X_test, y_train, y_test = train_test_split(
        young_episodes.loc[:,young_episodes.columns !=
        "episode"], young_episodes["episode"], test_size=0.3
)
```

Figure 7.2. Splitting the YMRS data set into training and testing data sets

After the algorithms were applied, the *cross_val_score()* function, which is also included in the scikit-learn library, was called to evaluate the score by *k*-fold cross-validation. This process consists in dividing the test set into *k* (*cv* attribute in the function) subsets and performing a prediction with different combinations of it, as seen in the diagram shown in *Figure 7.3*.

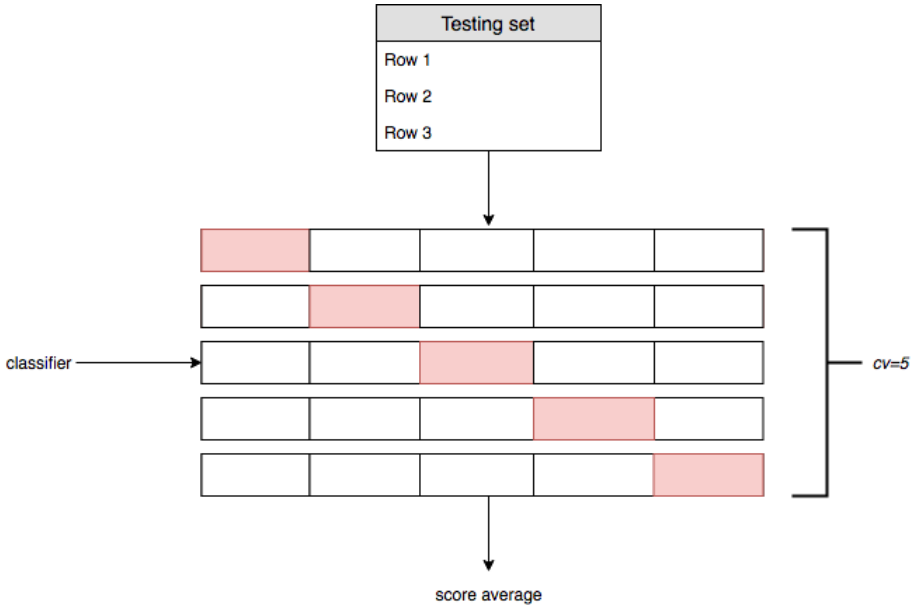

Figure 7.3. Cross validation technique with testing set

The *k*-fold cross-validation evaluates the accuracy by comparing the error of the folds, which is done by calculating the error of each fold with the result being the mean of the *k* error values, i.e. the sum of the *k* error values divided by the value of *k*:

$$E = \frac{1}{k}\sum_{i=1}^{k} E_i$$

In this part, each algorithm is presented with the mathematical theory behind it, and the process followed for its application is showed. The application of the algorithms is divided by data sets, including a comparison based on the accuracies obtained with each data set. The first section, where the Decision Tree algorithm is presented, is divided into a subsection for each data set, as a graph showing the generated tree is shown and analysed. All the algorithms are presented with a table that compares the accuracies obtained with each data set.

7.1. Decision Tree

The algorithm on which the Decision Tree algorithm is based is called *ID3*. In the original paper published by J.R. Quinlan in 1986 he mentions, as an example of classification, the diagnosis of a medical condition from different symptoms [22], which was exactly what we were trying to achieve with this project. The *ID3* algorithm uses trees that are constructed with a top-down strategy, which means that it splits each node into various nodes as it goes further down into the tree.

It was designed for training sets with many features and it uses an iterative structure, which means that the algorithm selects a subset from the initial data set and classifies the rest of the data set with it. This process continues until all elements have been classified correctly.

In this project, the Decision Tree algorithm was implemented with the *DecisionTreeClassifier* from the scikit-learn library (sklearn). This classifier uses certain hyper-parameters which can be tuned depending on the size and number of features of a data set. A hyper-parameter is a parameter that is passed to the classifier before its learning process begins.

In the case of the *DecisionTreeClassifier*, if we don't change the default hyper-parameters, the result will be a fully-grown tree with no pruning at all. This can make the tree very complex and memory consuming, so we have taken the liberty of tuning these hyper-parameters depending on the size and number of features of the data sets. The hyper-parameters that have been used for the *DecisionTreeClassifier* in this project are:

- **max_depth:** the maximum number of nodes in a branch.
- **min_samples_leaf:** the minimum number of samples required at a leaf node.

The reason why this algorithm has been used in this project is the way it discriminates different features that make wrong predictions [22] and the ease in being able to see this discrimination with some plotting libraries, such as the Graphviz [30] library for Python, which let us plot the Decision Tree classifier.

7.1.1. YMRS

The first step was to divide the data set into a training and a testing test, like we mentioned above, after which we proceeded to train the model, as seen in *Figure 7.4*.

```
clf = DecisionTreeClassifier(max_depth=5)
clf.fit(X_train, y_train)
```

Figure 7.4. Model training with Decision Tree classifier

When the model had been trained, we calculated the score with the *cross_val_score()* function and printed the mean of the scores that we obtained (see *Figure 7.5*).

```
scores = cross_val_score(clf, X_test, y_test)
print "Model accuracy: ", scores.mean()
```

Figure 7.5. Calculate model accuracy score

The max_depth in this case was set to 5 and the min_samples_leaf was set to 1 (the default value for the *DecisionTreeClassifier*), because the data were very scarce. With this model, we obtained an accuracy of 36%, which was really low and would make a very bad prediction on real data. With the help of the Graphviz [30] library, we plotted the Decision Tree classifier in order to see how each decision was made (see *Figure 7.6*).

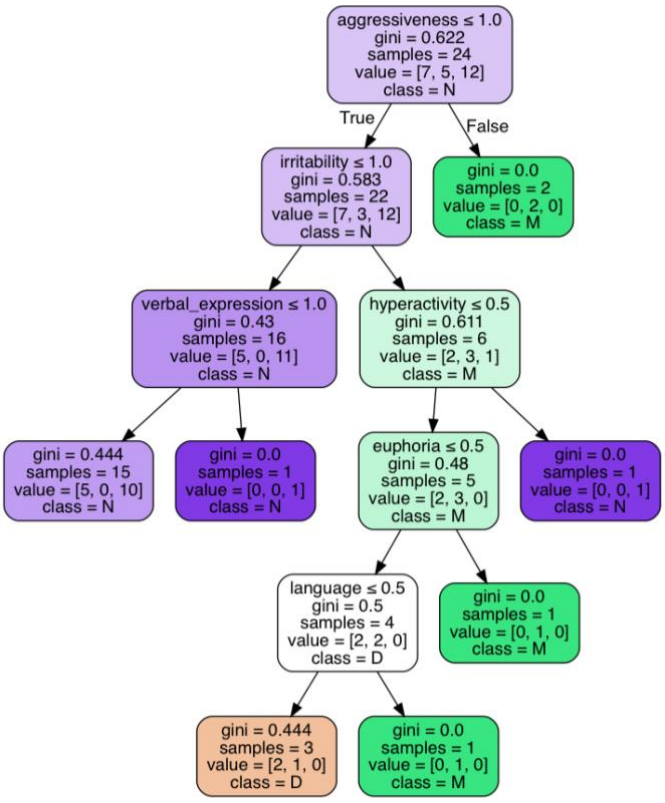

Figure 7.6. Decision Tree graph of YMRS data

There are three different variables shown in each node:

- **gini:** this variable represents the Gini impurity, which measures how often the node element would be incorrectly labelled if it was labelled randomly.
- **samples:** this variable measures the number of samples in the node.
- **value:** this variable represents the number of samples of the node that fall into each category (D, M or N), so if the values are, for example [5, 1, 9], it means that five samples were captured as D (Depression), one sample was captured as M (Mania) and the rest of the samples were captured as N (Euthymic). These values add up to the number shown in the samples variable.
- **class:** target in which the node is classified into (Depression, Mania or Euthymic). It is obtained with the highest value in the value array ([D, M, N]).

We could see that the most important split for this tree was aggressiveness lower or equal than 1 and that the tree continued by dividing the left branch with irritability values lower than or equal to 1 and the right branch with hyperactivity values lower than or equal to 0'5. This showed that if a patient had a value of aggressiveness greater than 1, he or she would automatically be classified as having a Mania episode, because the splits were not equally distributed, which didn't make much sense, as an aggressiveness level of 1 is quite low (it ranges from 0 to 8).

7.1.2. HDRS

With the HDRS data set, the process was exactly the same as with the YMRS data set. After obtaining training and testing data sets, we used the *DecisionTreeClassifier* to get an accuracy score. The hyper-parameters were set to the same values as the YMRS data set (max_depth=5 and min_samples_leaf=1).

An accuracy of 78% was obtained with the HDRS data set, which was too high for a data set with such a small amount of data. This most surely meant that the model was overfitted, so the testing set was probably very similar to the training set. In order to see how the algorithm made the decisions, we plotted the Decision Tree like we did with the YMRS data set (see *Figure 7.7*).

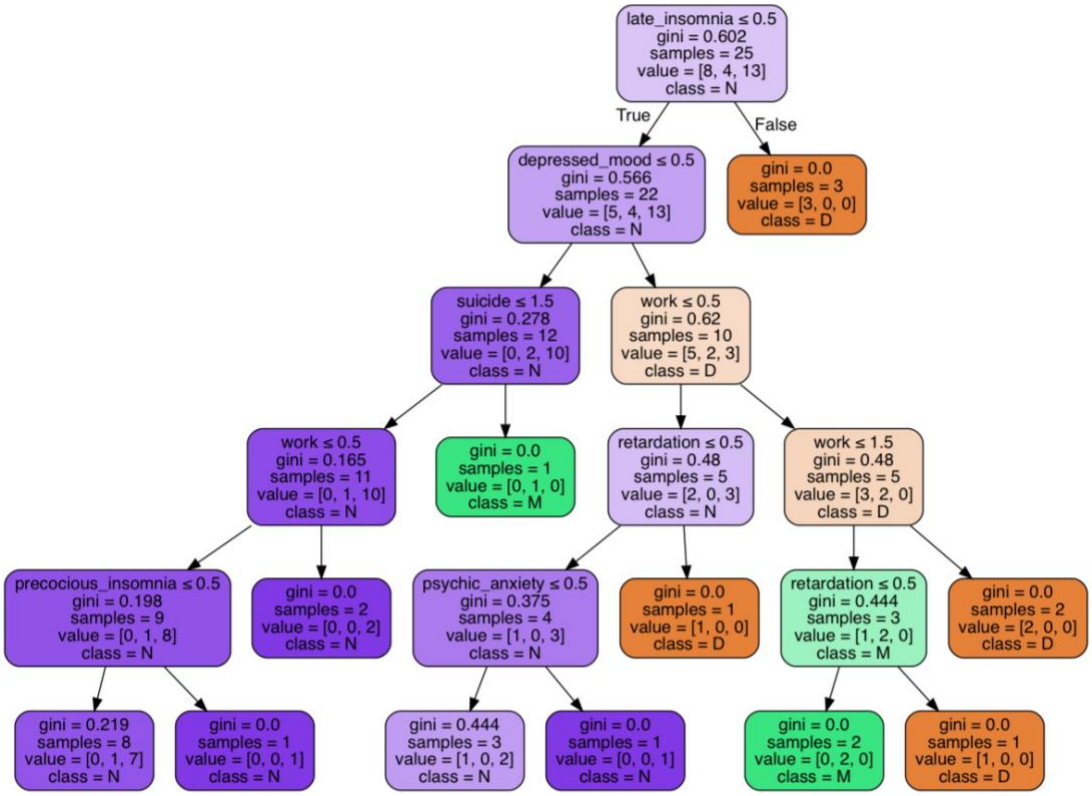

Figure 7.7. Decision Tree graph of HDRS data

The most important split was late_insomnia lower or equal than 0'5. Again, if the patient had a value for late insomnia that was higher than 0'5, he or she would be classified as having a Depression episode, which might or might not be a very representative value and should be further observed in future applications of this algorithm on similar problems.

The second most important split was depressed_mood lower or equal than 0'5, and all the samples classified as Depression (class=D) were represented on the right branch (depressed_mood greater than 0'5), which made sense because the higher the level of this variable, the more depressed the patient will feel, and it could indicate the start of a Depression episode in the patient.

7.1.3. Interviews

With the Interview data set we followed the same process as with the previous data sets. The accuracy obtained for this data set was 70%, which was a quite reasonable score having in mind that the data set had almost 650 entries. In this case, max_depth was set to 10 and min_samples_leaf was set to 15, because the amount of data that could be grouped was bigger than in the previous data sets. The tree graph (see *Figure 7.8*) was much deeper than with the previous data sets because of the size of the Interview data set.

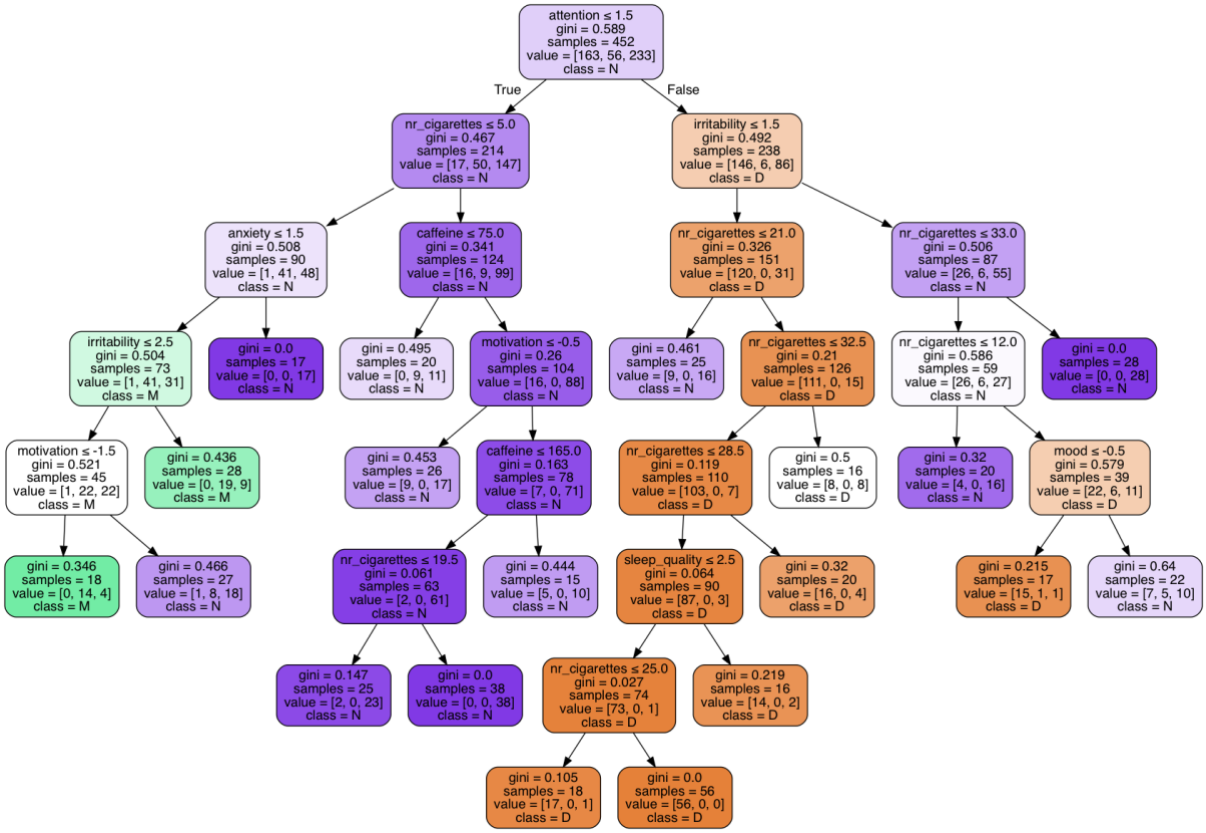

Figure 7.8. Decision Tree graph of Interview data

In this tree, the most important split was an attention value lower or greater than 1'5. All the Mania episodes were classified on the left side of this split and all the Depression episodes on the right side, which could mean that attention is an important variable in this kind of classification. The nr_cigarettes variable was present in eight different splits in the tree, which could indicate that this variable is important for this type of diagnosis.

It was interesting to follow the splits as a whole, as they presented very concrete conclusions on the decision making of the algorithm. For example, as seen in the first branch, a patient with a low attention level that doesn't smoke a lot and is quite anxious and not that irritable but feels very unmotivated will be labelled as having a Mania episode.

7.1.4. Interventions

The accuracy of the Decision Tree model obtained with this data set was 44%, which is really low. It had only two features that the algorithm could use for the decision making and that is probably the reason why it had such a bad performance. In the tree graph (see *Figure 7.9*) we can observe that the splits were very random and that there was no clear pattern on how the algorithm made the decisions.

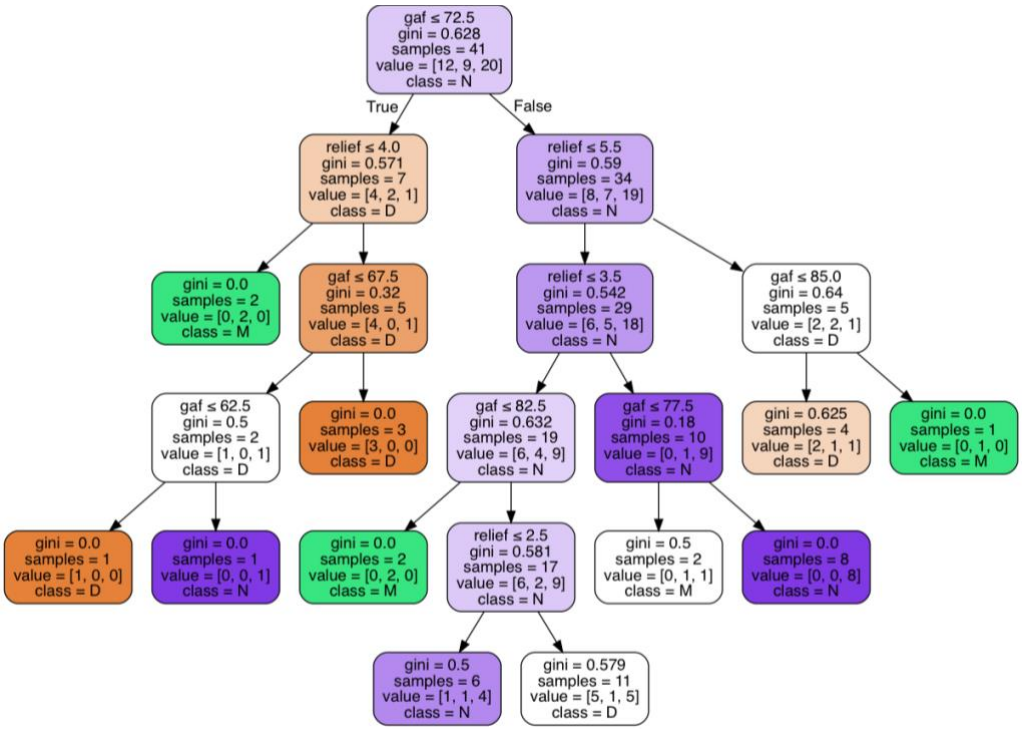

Figure 7.9. Decision Tree graph of Intervention data

7.1.5. YMRS-HDRS

This model returned an accuracy of 63% which was not so bad but, again, the amount of data (41 entries) didn't give a lot of confidence in the model. In the graph (see *Figure 7.10*) we can see the features used by the algorithm to make the decisions.

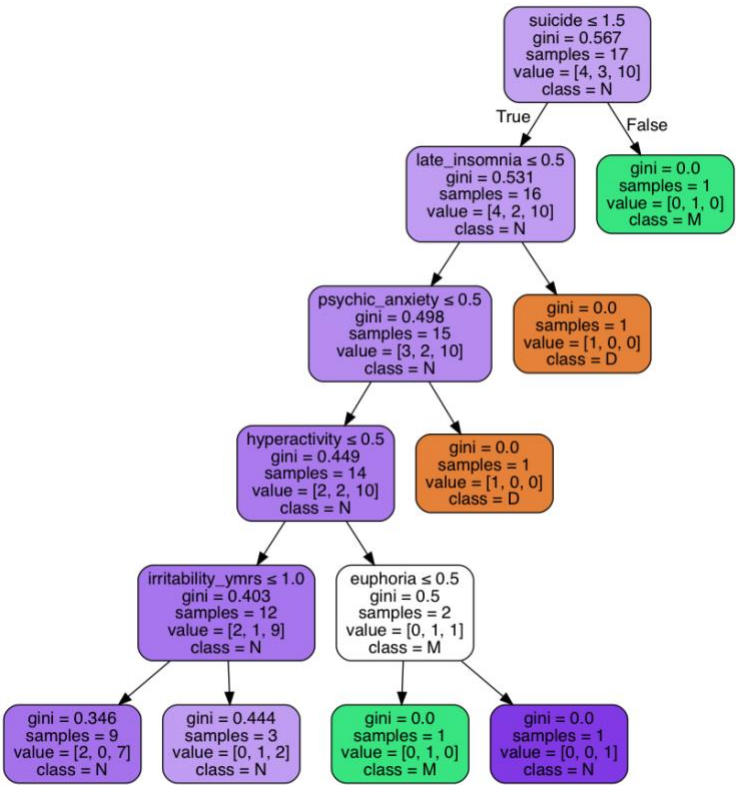

Figure 7.10. Decision Tree graph of YMRS-HDRS data

There were clear splits that defined a patient having a Depression episode, such as a level of late_insomnia and psychic_anxiety higher than 0'5. The most important split was a value of suicide lower or equal than 1'5, which means that if the patient has suicidal thoughts, tries to commit suicide or thinks that his or her life is not worth living (see HDRS values), this classifier would label him or her as having a Mania episode rather than a Depression episode, because it is clear that this patient could not be in a euthymic state.

7.1.6. Interviews-Interventions

This data set had 14 entries and, yet, the model had an accuracy of 67%, which means that it was clearly overfitted and the testing data was either very similar to the training data or the test sample was too small. In this case, max_depth was set to 5 and min_samples_leaf was set to 1, which generated a tree that was not very complex (see *Figure 7.11*). This classifier used the number of cigarettes (nr_cigarettes ≤ 22'5) as the most important split.

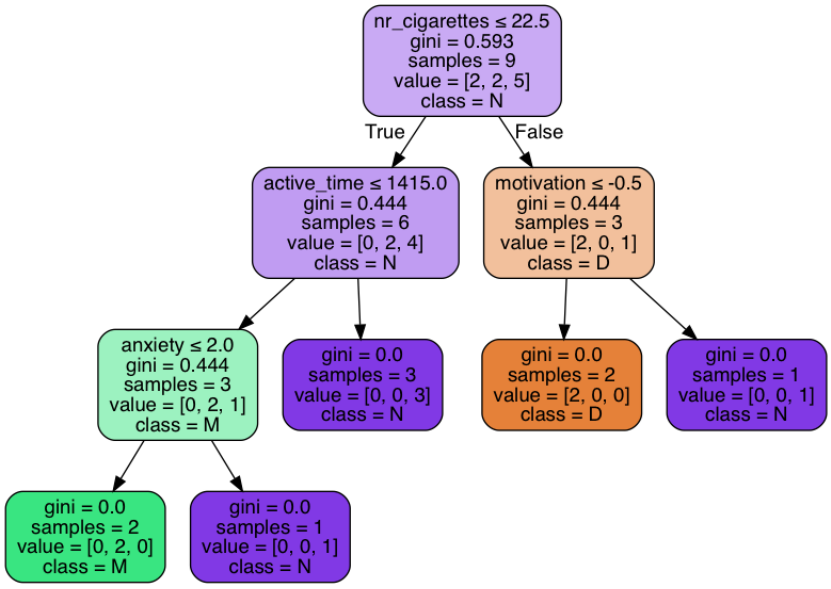

Figure 7.11. Decision Tree graph of Interviews-Interventions data

7.1.7. Prediction accuracy comparison

As shown in *Figure 7.12*, the Decision Tree classifier had the best accuracy with the HDRS data set which, as we stated before, probably was because the model was overfitted. We can say that the best scoring data set, in relation with the amount of data, was the Interview data set.

Dataset	Accuracy
YMRS	36%
HDRS	78%
Interviews	70%
Interventions	44%
YMRS-HDRS	63%
Interviews-Interventions	67%
Average	60%

Figure 7.12. Prediction accuracies of the Decision Tree algorithm

7.2. Random Forest

The Random Forest algorithm consists in an array of tree predictors in which every tree is depending on values of a random vector that is sampled with the exact same distribution for all the trees [23]. Like the Decision Tree algorithm, the decision criteria consists in looking at each feature in the training set individually and choosing the ones that help the most in giving the model an optimal fit.

The Random Forest algorithm is implemented with the *RandomForestClassifier* from the scikit-learn library, and the process of implementation was very similar to the *DecisionTreeClassifier*, as seen in *Figure 7.13*.

```
clf = RandomForestClassifier(n_jobs=-1)
clf.fit(X_train, y_train)
```

Figure 7.13. Model training with Random Forest classifier

The n_jobs hyper-parameter represents the number of jobs running in parallel for both fitting and predicting. The -1 value indicates that the number of jobs is equal to the number of cores of the processor.

This algorithm has been chosen as one of the options for this project because of the speed and scalability it presents on huge data sets and because it can be combined with programming paradigms like Map Reduce [31], which is made for processing huge amounts of data. This is quite useful for future implementations and the continuity of the project, which will hopefully include much larger data sets.

7.2.1. Prediction accuracy comparison

The prediction accuracy matrix (see *Figure 7.14*) shows that the Random Forest algorithm performed very poorly on both the YMRS data set and the combination of the Interviews and Interventions, with 38% and 17% respectively, the latter showing a very bad performance.

The accuracy obtained with the HDRS data did not show overfitting of the model but was lower than the best accuracies obtained. The combination of the YMRS and HDRS data set resulted in the same accuracy as the one obtained with the Decision Tree algorithm. The only data set that resulted in a high prediction accuracy with this algorithm was the Interview data set, with a score of 72%.

Dataset	Accuracy
YMRS	38%
HDRS	55%
Interviews	72%
Interventions	65%
YMRS-HDRS	63%
Interviews-Interventions	17%
Average	52%

Figure 7.14. Prediction accuracies of the Random Forest algorithm

7.3. SVM

Support Vector Machines (SVM) aim to maximize the distance between the targets in a hyperplane, in order to find a hyperplane that is optimal for classification. The support vectors are the vectors that define this hyperplane, which can be seen in the basic diagram of *Figure 7.16*.

SVM maps the input vectors into a high-dimensional feature space. In this space, the optimal separating hyperplane is constructed [32]. Given the distance (confidence) from each point in the training set to the decision hyperplane

$$d(x, L) = w \cdot x + b$$

where w is the vector that defines the optimal hyperplane and x are the coordinates of the point, SVM finds this optimal hyperplane with the optimization problem [32]

$$\max_{w,\gamma} \gamma$$

$$s.t. \forall i, y_i(w \cdot x_i + b) \geq \gamma$$

which finds γ, the largest possible margin, such that, for every point in the training set, the margin is at least γ, which implicates finding the w that achieves this margin (see *Figure 7.15*). This is achieved by multiplying the distance of each point in the training set by the class of the prediction.

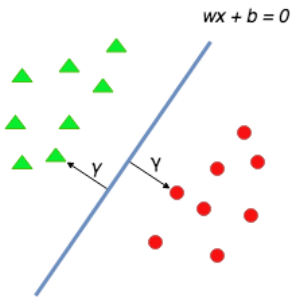

Figure 7.15. Finding the largest possible margin to the hyperplane

The main goal of SVM is to produce a predictive model with the training data that will correctly classify the target values (output) of the test data given only the test data attributes [24].

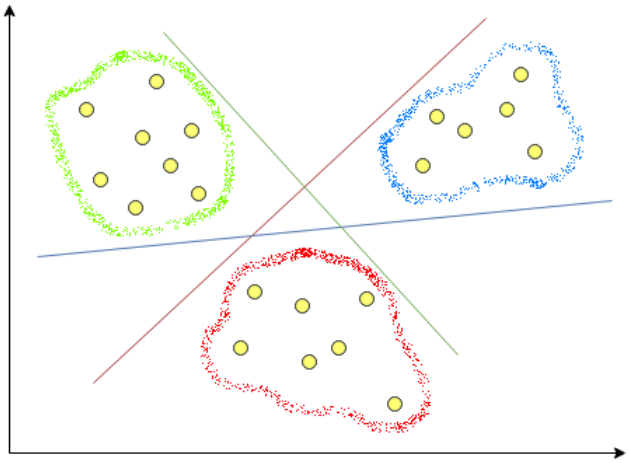

Figure 7.16. Basic diagram of the SVM hyperplanes

The reason why this algorithm has been chosen for this project is the good generalization that it generally provides even if there are biased values in the data [33], which is something very useful as we don't really know if there could be any bias in our data.

7.3.1. Prediction accuracy comparison

The accuracy matrix for the SVM algorithm, as seen in *Figure 7.17*, showed that the accuracy obtained with the YMRS data set was 38%, which was the same as the accuracy obtained with the Random Forest algorithm and, again, was very low. The consequence would probably be a very poor model performance. Moreover, the accuracy obtained with the HDRS data set was 36%, which was even lower than with the YMRS and would also incur in a very bad performance of the model.

The Interview data set returned a score of 68% with the SVM algorithm, which was lower than the accuracy obtained with the two previously studied algorithms (Decision Tree and Random Forest), but we considered that it could still be interesting to use this algorithm for prediction so that we could check the performance it offered on real data.

The combined data set of YMRS and HDRS contained only 25 entries, as we mentioned above, which tells us that the accuracy of 70% was clearly biased and that the model was probably overfitted. The accuracy of the model with the other combination of data sets (Interviews and Interventions) was 42%. The use of SVM on this data set would clearly result in quite a bad performance. The Intervention data set returned an accuracy of 57%, which is a quite good performance and should be considered for future implementations of this algorithm on the same kind of problem.

Dataset	Accuracy
YMRS	38%
HDRS	36%
Interviews	68%
Interventions	57%
YMRS-HDRS	70%
Interviews-Interventions	42%
Average	52%

Figure 7.17. Prediction accuracies of the SVM algorithm

7.4. Logistic Regression

The mathematical concept behind a Logistic Regression model is the natural logarithm of an odds ratio between two variables, also called logit [25]. The Logistic Regression is explained with the Logistic Function, which takes an input and outputs a number between 0 and 1, and is defined as:

$$\sigma(t) = \frac{e^t}{e^t + 1}$$

In this function, t is a linear function that can be represented as $t = \beta_0 + \beta_1 x$ and, therefore, the Logistic Function can be expressed as:

$$\pi(x) = \frac{e^{\beta_0 + \beta_1 x}}{e^{\beta_0 + \beta_1 x} + 1}$$

The logit, which is the equivalent to the Linear Regression expression and, as mentioned above, represents the concept behind the whole model, is the inverse of the Logistic Function [34], i.e.:

$$g(x) = ln\left[\frac{\pi(x)}{1 - \pi(x)}\right] = \beta_0 + \beta_1 x$$

Traditionally, what characterizes the Logistic Regression algorithm is that the outcome variable is binary, which is not the case in this problem, as the possible outcomes are three (D, M or N). Nevertheless, the scikit-learn Python library [12] allows the application of a Multi-class Logistic Regression by default, fitting a binary problem for each label (one vs. all classification) by calculating the probability of each possible outcome separately. A basic diagram of how this algorithm works can be seen in *Figure 7.18*.

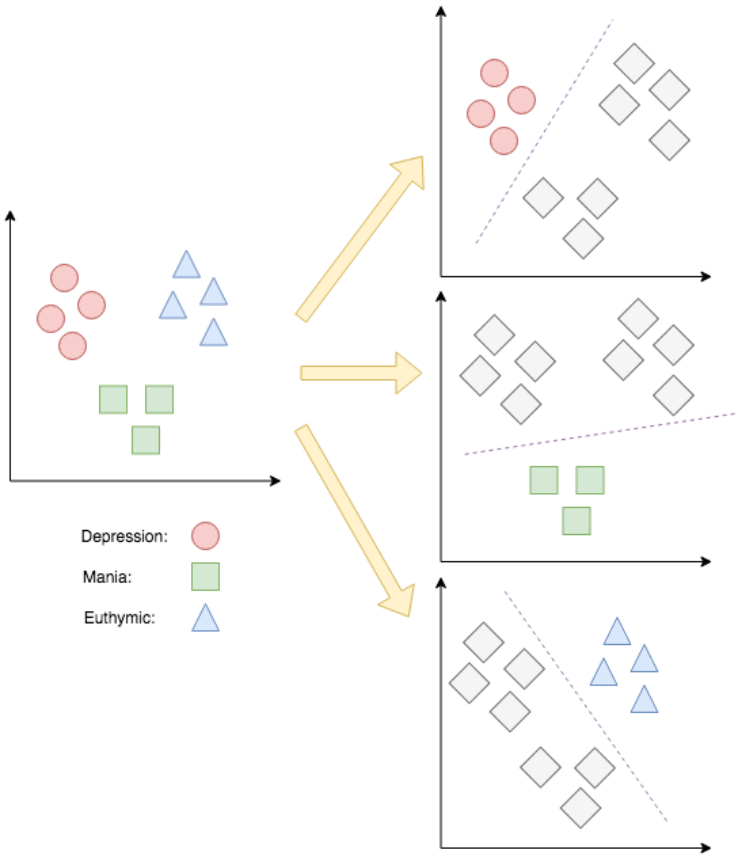

Figure 7.18. Basic diagram of one vs. all Multi-class Logistic Regression classification

The reason why this algorithm has been selected among the others for this study is its ease of implementation and flexibility.

7.4.1. Prediction accuracy comparison

Taking a look at the prediction accuracy matrix (see *Figure 7.19*) we were able to see which data sets that the Logistic Regression algorithm performed the best with. The accuracy obtained with the YMRS data set was 76%, which is too high for such a small data set. This data set clearly didn't perform quite well with any of the algorithms used, but it shouldn't be discarded for future implementations with more data.

The HDRS data set returned an accuracy of 62%, which was quite good for a data set that small. It could be interesting to test the performance with more data, the same way as the YMRS data set. The Interview data set returned a score of 67% which is an acceptable performance if we consider that it has much more data than the rest of the data sets.

The accuracy of the Logistic Regression classifier with the Intervention data set was 51%, which is lower than the score obtained with the rest of the algorithms used. The algorithm performed very poorly with the combination of YMRS and HDRS, as it returned an accuracy of only 33%. The model was clearly underfitted. The combination of Interviews and Interventions had an accuracy of 83%, which should also be considered for future implementations but was discarded for testing on randomized data because of the small size of the data set.

Dataset	Accuracy
YMRS	76%
HDRS	62%
Interviews	67%
Interventions	51%
YMRS-HDRS	33%
Interviews-Interventions	83%
Average	52%

Figure 7.19. Prediction accuracies of the Logistic Regression algorithm

7.5. Tests on randomized data

In order to check if the predictions made by the different models obtained with the algorithms made any sense, we proceeded to randomize some patient data to test the models on. This was done by randomizing the values of the different items in the Interview data set (see *Figure 7.20*).

Every item has a different range so, for example, the mood values that we generated had to be between -3 and 3, the range in which those values were in the interviews performed by the psychiatrist. We did this with a loop that generated interview data for fifty random patients in order to be able to see which algorithms made the most reasonable prediction.

```
variables = {}
variables['mood'] = randint(-3, 3)
variables['motivation'] = randint(-3, 3)
variables['attention'] = randint(0, 4)
variables['irritability'] = randint(0, 4)
variables['anxiety'] = randint(0, 4)
variables['sleep_quality'] = randint(0, 4)
variables['nr_cigarettes'] = randint(0, 40)
variables['caffeine'] = randint(0, 250)
variables['active_time'] = randint(200, 1800)

test = np.array([
        variables['mood'], variables['motivation'],
        variables['attention'], variables['irritability'],
        variables['anxiety'], variables['sleep_quality'],
        variables['nr_cigarettes'], variables['caffeine'],
        variables['active_time']
])
```

Figure 7.20. Random patient interview data generation

The Random Forest classifier seemed to make quite reasonable predictions. In the case of a patient that feels down and sleeps a lot, this algorithm predicted that he or she could be tending towards a Depression episode, as seen in *Figure 7.21*.

```
-----------------------------------------
PATIENT  7
-----------------------------------------
Mood:  -3
Motivation:  1
Attention:  2
Irritability:  4
Anxiety:  2
Sleep quality:  4
Number of cigarettes:  27
Caffeine:  27
Active time:  551
-----------------------------------------
PREDICTIONS
-----------------------------------------
- Decision Tree prediction:  The patient is in a Euthymic state
- Random Forest prediction:  The patient could be tending towards a
Depression episode
- Logistic Regression prediction:  The patient is in a Euthymic state
- SVM prediction:  The patient is in a Euthymic state
```

Figure 7.21. Prediction of a Depression episode on a patient with the lowest mood value

Another example of a precise prediction was the one made by both Logistic Regression and Random Forest on a patient feeling unmotivated and down and with a high level of anxiety, as can be seen in *Figure 7.22*.

```
-----------------------------------------
PATIENT  46
-----------------------------------------
Mood:  -2
Motivation:  -2
Attention:  4
Irritability:  2
Anxiety:  3
Sleep quality:  2
Number of cigarettes:  15
Caffeine:  211
Active time:  804
-----------------------------------------
PREDICTIONS
-----------------------------------------
- Decision Tree prediction:  The patient is in a Euthymic state
- Random Forest prediction:  The patient could be tending towards a
Depression episode
- Logistic Regression prediction:  The patient could be tending towards
a Depression episode
- SVM prediction:  The patient is in a Euthymic state
```

Figure 7.22. Prediction of Depression episode on a patient with low mood and motivation values

In other cases, if the patient was feeling down but was quite active and was both irritable and anxious, the Random Forest algorithm predicted that this patient could be tending towards a Mania episode (see *Figure 7.23*) when the rest of the algorithms predicted that he or she was in a Euthymic state.

```
-----------------------------------------
PATIENT  35
-----------------------------------------
Mood:  -3
Motivation:  0
Attention:  0
Irritability:  3
Anxiety:  3
Sleep quality:  1
Number of cigarettes:  15
Caffeine:  66
Active time:  1239
-----------------------------------------
PREDICTIONS
-----------------------------------------
- Decision Tree prediction:  The patient is in a Euthymic state
- Random Forest prediction:  The patient could be tending towards a
Mania episode
- Logistic Regression prediction:  The patient is in a Euthymic state
- SVM prediction:  The patient is in a Euthymic state
```

Figure 7.23. Prediction of a Mania episode on an active patient with high anxiety and irritability levels

If the patient had a normal behaviour, i.e. good mood and motivated, with normal anxiety and irritability levels, all the algorithms predicted that he or she was in a euthymic state, as seen in *Figure 7.24*.

```
-----------------------------------------
PATIENT  39
-----------------------------------------
Mood:  3
Motivation:  3
Attention:  2
Irritability:  0
Anxiety:  2
Sleep quality:  2
Number of cigarettes:  12
Caffeine:  117
Active time:  571
-----------------------------------------
PREDICTIONS
-----------------------------------------
- Decision Tree prediction:  The patient is in a Euthymic state
- Random Forest prediction:  The patient is in a Euthymic state
- Logistic Regression prediction:  The patient is in a Euthymic state
- SVM prediction:  The patient is in a Euthymic state
```

Figure 7.24. Prediction of a Euthymic state on a patient with normal behaviour

Some of the predictions were not reasonable at all. For example, the state of a patient with a depressed mood and a quite low motivation level with a lot of sleeping hours was predicted as Euthymic, as seen in *Figure 7.25*, when the doctor would probably label it as tending towards a Depression.

```
-----------------------------------------
PATIENT  23
-----------------------------------------
Mood:  -3
Motivation:  -1
Attention:  2
Irritability:  2
Anxiety:  1
Sleep quality:  2
Number of cigarettes:  37
Caffeine:  64
Active time:  612
-----------------------------------------
PREDICTIONS
-----------------------------------------
- Decision Tree prediction:  The patient is in a Euthymic state
- Random Forest prediction:  The patient is in a Euthymic state
- Logistic Regression prediction:  The patient is in a Euthymic state
- SVM prediction:  The patient is in a Euthymic state
```

Figure 7.25. Prediction of a Euthymic state on a patient with clear depression symptoms

7.6. Tests on real data

As an example of how to use these algorithms on real patients, we implemented a small Python program which imports the Random Forest classifier and makes predictions based on a specific input from a terminal. This input contained all the variables that could be present in an interview made by the doctor. The tests weren't run on real patients, but the input data used in them corresponded to different states that the patient could be in.

The program returns the prediction in a human readable format and colours the output messages depending on the severity of the diagnosis. If the model predicts that the patient could be tending towards a Mania or Depression episode, the messages are shown in yellow (see *Figure 7.26* and *Figure 7.27*), and if the prediction states that the patient will stay in a Euthymic state, the message will be blue (see *Figure 7.28*), as everything should be normal.

```
(analysis) Axels-MacBook-Pro:TFG axeljunes$ python state_prediction.py


---------------------------------------
|  BIPOLAR DISORDER CRISIS PREDICTION  |
---------------------------------------
Author: Axel Junestrand
License: Apache 2.0


Please indicate each of the values gathered during the interview:
* Mood (-3 to 3)? -2
* Motivation (-3 to 3)? -2
* Attention (0 to 4)? 4
* Irritability (0 to 4)? 3
* Anxiety (0 to 4)? 3
* Sleep quality (0 to 4)? 0
* Number of cigarettes smoked? 30
* Amount of caffeine ingested? 200
* When did the patient wake up (hh:mm)? 05:00
* When did the patient go to bed (hh:mm)? 20:30


The patient could be tending towards a DEPRESSION episode
```

Figure 7.26. Prediction of a Depression episode

```
(analysis) Axels-MacBook-Pro:TFG axeljunes$ python state_prediction.py


----------------------------------------
|  BIPOLAR DISORDER CRISIS PREDICTION  |
----------------------------------------
Author: Axel Junestrand
License: Apache 2.0


Please indicate each of the values gathered during the interview:
* Mood (-3 to 3)? 2
* Motivation (-3 to 3)? -2
* Attention (0 to 4)? 1
* Irritability (0 to 4)? 2
* Anxiety (0 to 4)? 1
* Sleep quality (0 to 4)? 3
* Number of cigarettes smoked? 20
* Amount of caffeine ingested? 0
* When did the patient wake up (hh:mm)? 10:00
* When did the patient go to bed (hh:mm)? 01:00


The patient could be tending towards a MANIA episode
```

Figure 7.27. Prediction of a Mania episode

```
(analysis) Axels-MacBook-Pro:BDCP axeljunes$ python state_prediction.py


----------------------------------------
|  BIPOLAR DISORDER CRISIS PREDICTION  |
----------------------------------------
Author: Axel Junestrand
License: Apache 2.0


Please indicate each of the values gathered during the interview:
* Mood (-3 to 3)? 1
* Motivation (-3 to 3)? 3
* Attention (0 to 4)? 2
* Irritability (0 to 4)? 1
* Anxiety (0 to 4)? 4
* Sleep quality (0 to 4)? 2
* Number of cigarettes smoked? 5
* Amount of caffeine ingested? 100
* When did the patient wake up (hh:mm)? 09:30
* When did the patient go to bed (hh:mm)? 23:40


The patient is staying in a EUTHYMIC state
```

Figure 7.28. Prediction of a Euthymic state

8. Results

The results obtained during this project are separated in different sections. Each section contains a summary of the most important results that were obtained in the corresponding part of the project.

8.1. Results of the data gathering

The data gathering is a small part of the project but one of the most important, as it lays the ground for the rest of the project. With the help of the results obtained, we could observe that the data sets were generally really small, as seen in *Figure 8.1*, except for the Interview data set which had an acceptable size in comparison with the others.

```
print "Number of rows in HDRS dataframe:", len(hamilton.index)
print "Number of rows in YDRS dataframe:", len(young.index)
print "Number of rows in interview dataframe:", len(interviews.index)
print "Number of rows in intervention dataframe:", len(interventions.index)
```

```
Number of rows in HDRS dataframe: 49
Number of rows in YDRS dataframe: 47
Number of rows in interview dataframe: 727
Number of rows in intervention dataframe: 91
```

Figure 8.1. Length of the data sets after importing the data from the files

The fact that the data sets had such a small amount of data has clearly been the most challenging part of the project, because the rest of the sections depended on the amount of data that was available.

8.2. Results of the data cleaning

In the data cleaning part, there were very few uncomplete values in the data sets, and the ones that were empty did not present a very big challenge as the distribution was very similar in the data set and they could easily be filled.

The whole process of data cleaning returned a clean data set for each data set that was gathered: Episode data set (*Figure 8.2*), YMRS data set (*Figure 8.3*), HDRS data set (*Figure 8.4*), Interview data set (*Figure 8.5*) and Intervention data set (*Figure 8.6*).

	patient	start	end	episode
0	D	2017-07-01 00:00:00	2017-07-24 00:00:00	D
1	D	2017-08-15 00:00:00	2917-09-11 00:00:00	D
2	G	2017-07-24 00:00:00	2017-08-07 00:00:00	D
3	G	2017-09-04 00:00:00	2017-11-01 00:00:00	M
5	M	2017-06-07 00:00:00	2017-07-01 00:00:00	M
6	M	2017-07-14 00:00:00	2017-07-30 00:00:00	D
7	M	2017-09-25 00:00:00	2017-10-10 00:00:00	D

Figure 8.2. Sample of clean Episode data set

code	date	euphoria	hyperactivity	sexual_impulse	sleep	irritability	verbal_expression	language	thought	aggressiveness	appearance	illness_awareness
G	2017-06-06 00:00:00	0	1	0	0	2	2	0	0	0	0	0
G	2017-06-16 00:00:00	0	0	0	0	0	2	0	0	0	0	0
M	2017-06-21 00:00:00	0	1	0	0	0	2	0	0	2	0	0
M	2017-06-21 00:00:00	0	1	0	0	0	2	0	0	2	0	0

Figure 8.3. Sample of clean YMRS data set

code	date	depressed_mood	guilt	suicide	precocious_insomnia	medium_insomnia	verbal_expression	language	thought	...	retardation	agitation
G	2017-06-06 00:00:00	0	0	0	0	0	0	0	0	...	0	0
G	2017-06-16 00:00:00	0	0	0	0	0	0	0	0	...	0	0
M	2017-06-21 00:00:00	0	0	0	0	0	0	0	0	...	0	0
M	2017-06-21 00:00:00	0	0	0	0	0	0	0	0	...	0	0
D	2017-06-26 00:00:00	0	0	0	0	0	0	0	0	...	0	0

Figure 8.4. Sample of clean HDRS data set

mood	motivation	attention	irritability	anxiety	sleep_quality	nr_cigarettes	caffeine	alcohol	other_drugs	patient	date	active_time
0	1	1	1	1	2	0	90	0	0	G	2007-07-17 00:00:00	1470
-1	-1	2	1	1	3	24	120	0	0	D	2017-01-02 00:00:00	1790
0	-1	2	1	1	1	24	90	0	0	D	2017-01-15 00:00:00	1545
2	2	3	3	3	3	34	150	0	0	D	2017-06-01 00:00:00	1710
2	2	3	3	3	3	38	150	0	0	D	2017-06-02 00:00:00	1770

Figure 8.5. Sample of clean Interview data set

	code	date	gaf	relief
0	D	2017-06-01 00:00:00	80	5
1	D	2017-06-01 00:00:00	80	5
20	G	2017-06-06 00:00:00	80	1
68	M	2017-06-07 00:00:00	60	3
21	G	2017-06-12 00:00:00	70	5

Figure 8.6. Sample of clean intervention data set

8.3. Results of the Exploratory Data Analysis

The Exploratory Data Analysis showed some very interesting relationships between certain features in the data sets that we visualized. Particularly, the Heatmap of feature correlations in the HDRS data set (*see Figure 5.12*), showed a possible correlation between the work/activities performed by the patient and a depressed mood, which we confirmed with the marginal plot (*see Figure 5.13*) and 2D kernel density plot (see *Figure 5.14*) that we plotted afterwards. Like we mentioned in the corresponding section, this is relationship that makes a lot of sense because when a patient feels depressed, he or she usually dedicates less time to work and other activities.

In the 2D kernel density plot (see *Figure 5.31*) obtained from the mood and motivation features from the Interview data set we could see a linear relationship between them, which means that the patient almost always gives the same answers for these two variables in the interviews with the doctor.

In general, this analysis helped us understand the data much better and visualize the most important relationships between features in all the data sets gathered for the project with the help of different types of plots and graphs.

8.4. Results of the data combination

As a whole, the data combination resulted in some very small data sets, like the combination of Interviews and Interventions, which only had 14 rows, as seen in *Figure 8.7*. The rest of the combinations did not have a lot of entries either, but they were interesting for future implementations, as they contained a lot of features.

```
print "Length of the combined data set (interviews and interventions): ",
len(interviews_interventions)
```

```
Length of the combined data set (interviews and interventions):
14
```

Figure 8.7. Length of the combination of Interviews and Interventions

The size of the data sets made it possible to select the ones that we could use for the algorithm application and comparison, which in the end were the YMRS, HDRS, Interviews and Interventions data sets, as well as the combination of YMRS and HDRS and the combination of Interviews and Interventions.

8.5. Results of the algorithm implementation

With the help of the study of the different algorithms and their implementations, we were able to compare the data sets in order to see which ones performed better and could be further applied for predicting randomized test data. The Interview data set performed very well, accompanied by a prediction accuracy above 65% with all the algorithms used. The rest of the data sets did not perform as good as the Interview data set, which we might conclude is because of the lack of data. In future implementations of these algorithms, the prediction accuracies will most likely be higher thanks to new data.

The best way to compare the accuracies obtained by the different algorithms with all the data sets was to make an algorithm performance matrix, which is shown in *Figure 8.8.*

Algorithms/Datasets	Decision Tree	Random Forest	SVM	Logistic Regression	Average
YMRS	36%	38%	38%	76%	47%
HDRS	78%	55%	36%	62%	58%
Interviews	70%	72%	68%	67%	69%
Interventions	44%	65%	57%	51%	54%
YMRS-HDRS	63%	63%	70%	33%	57%
Interviews-Interventions	67%	17%	42%	83%	52%
Average	60%	52%	52%	62%	

Figure 8.8. Algorithm performance matrix

This matrix showed that, in average, the data set that returned the best prediction accuracy (69%) was the Interview data set, as seen on the right column. The algorithm that performed the best, also in average (62%), was the Logistic Regression algorithm, as seen on the row farthest down.

Even though the algorithm that had the best accuracy in average was the Logistic Regression, we stated that the Random Forest algorithm made the most accurate predictions, in the sense that they were very reasonable given the behaviour of the patients, which we tested with randomized data. These predictions made possible the implementation of a small program with the Random Forest classifier that we obtained, which can be used by the doctors with real patients.

9. Conclusions and future work

9.1. Conclusions

As a conclusion of this project we can state that having a deep understanding of the data we are working with is essential in any Machine Learning project focused on a branch of medical science like psychiatry, where knowing which behaviours are normal and abnormal in the patients can help us create much more precise prediction models.

We can also affirm that the amount of data used in a project is a very important factor because with a larger amount of data we will be able to get prediction accuracies with a much higher level of confidence.

In the same way that understanding the data is important, having a deep perception of the theory behind each algorithm used, as well as their different implementations, is crucial in order to get the models to perform in the best way possible.

9.2. Future work

The most immediate use of the results obtained in this project would be to train the same algorithms used with larger amounts of data, in order to see if they perform in a similar way.

Gathering objective data from devices like mobile phones or wristbands is something that can be accomplished quite easily. The goal of this would be to compare the performance of different algorithms on the objective data gathered from these devices with the performance results obtained on the subjective data used in this project.

As for other more indirect applications of the results obtained during this project, the implementation of a drug recommending system for patients with Bipolar Disorder could be made by predicting the states in which the patients are in during a certain period of time. These predictions could be stored in a database which also contains the medicine that these patients have been prescribed with during the same period of time, thus providing the possibility of checking which drugs are better for the patient.

10. Conclusiones y trabajo futuro

10.1. Conclusiones

Como conclusión de este trabajo, podemos afirmar que entender los datos con los que estamos trabajando en profundidad es esencial en cualquier proyecto de *Machine Learning* relacionado con ramas de la medicina como la psiquiatría, donde entender qué comportamientos son normales y anormales en los pacientes nos puede ayudar a crear modelos de predicción mucho más precisos.

También podemos concluir que la cantidad de datos utilizados en un proyecto es un factor muy importante ya que con una mayor cantidad de datos podremos obtener predicciones con un nivel de confianza mucho más alto.

Del mismo modo que la comprensión de los datos es importante, tener una percepción profunda de la teoría detrás de cada algoritmo utilizado, así como de sus diferentes implementaciones, es crucial para que los modelos funcionen de la mejor manera posible.

10.2. Trabajo futuro

El uso más inmediato de los resultados obtenidos en este trabajo sería entrenar los mismos algoritmos utilizados con mayores cantidades de datos, para ver si funcionan de manera similar.

Recopilar datos objetivos de dispositivos como teléfonos móviles o pulseras es algo que se puede lograr con relativa facilidad. El objetivo sería comparar el rendimiento de los diferentes algoritmos conseguido con los datos objetivos recopilados mediante estos dispositivos con los resultados obtenidos a partir de los datos subjetivos utilizados en este trabajo.

En cuanto a otras aplicaciones más indirectas de los resultados obtenidos en este proyecto, se podría hacer un sistema de recomendación de medicamentos para pacientes con Trastorno Bipolar, prediciendo el estado en el que se encuentran estos pacientes durante un determinado período de tiempo. Estas predicciones pueden almacenarse en una base de datos que también contiene el medicamento con el que se prescribió a estos pacientes durante el mismo período de tiempo, posibilitando así verificar qué medicamentos son mejores para el paciente.

11. Bibliography

[1] M. Bonsall et al., Bipolar disorder dynamics: affective instabilities, relaxation oscillations and noise, vol. 12, Journal of The Royal Society Interface, The Royal Society, 2015.

[2] www.bip4cast.org (Last accessed: 23/05/2018).

[3] V. López et al., Specification of a CAD Prediction System for Bipolar Disorder, Conference on Uncertainty Modelling in Knowledge Engineering and Decision Making, 2016, pp. 162-167.

[4] V. Mariscal, Integración de Datos y Análisis Predictivo en Tratamiento de Drogodependencia, Trabajo de Fin de Master, Madrid: E-Prints Universidad Complutense de Madrid, 2016.

[5] A. Terceño, Análisis de un modelo predictivo basado en Google Cloud y TensorFlow, Trabajo de Fin de Doble Grado, Madrid: E-Prints Universidad Complutense de Madrid, 2017.

[6] J. Anchiraico, Diseño de una Arquitectura Big Data para la Predicción de Crisis en el Trastorno Bipolar, Trabajo de Fin de Master, E-Prints Universidad Complutense de Madrid, 2016.

[7] A. Martínez, Introduction to Big Data and First Steps in a Big Data Project, Trabajo de Fin de Grado, Madrid: E-Prints Universidad Complutense de Madrid, 2016.

[8] Python Software Foundation, Python Language Reference, version 2.7 (http://www.python.org).

[9] F. Pérez and B.E. Granger, IPython: A System for Interactive Scientific Computing, Computing in Science & Engineering, 2007.

[10] W. McKinney, Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 2010.

[11] J. D. Hunter, Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 2007.

[12] F. Pedregosa et al., Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 2011.

[13] A. Grünerbl et al., Smart-Phone Based Recognition of States and State Changes in Bipolar Disorder Patients, vol. 19, IEEE Journal of Biomedical and Health Informatics, 2014, pp. 140-148.

[14] J. Nicholas et al., Mobile Apps for Bipolar Disorder: A Systematic Review of Features and Content Quality, vol. 17, Journal of medical Internet research, 2015, p. 198.

[15] A. Doryab et al., Impact factor analysis: combining prediction with parameter ranking to reveal the impact of behaviour on health outcome, vol. 19, Personal and Ubiquitous Computing, 2014, pp. 355-365.

[16] M. Faurholt-Jepsen et al., Smartphone data as objective measures of bipolar disorder symptoms, vol. 217, Psychiatry Research, 2014, pp. 124-127.

[17] P. Grof et al., Mood Charting and Technology: New Approach to Monitoring Patients with Mood Disorders, vol. 2, Current Psychiatry Reviews, 2006, pp. 423-429.

[18] R. C. Young et al., A rating scale for mania: reliability, validity and sensitivity, British Journal of Psychiatry, 1978, pp. 429-435.

[19] M. Hamilton, A rating scale for depression, Journal of Neurology, Neurosurgery, and Psychiatry, 1960, pp. 56-62.

[20] J. W. Tukey, Exploratory Data Analysis, vol. 23, London: Addison-Wesley Publishing Company Reading, 1977, pp. 413-414.

[21] Garima, H. Gulati and P. K. Singh, Clustering techniques in data mining: A comparison, New Delhi: 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015, pp. 410-415.

[22] J. R. Quinlan, Induction of Decision Trees, vol. 1, Hingham, Massachusetts: Kluwer Academic Publishers, 1986, pp. 81-106.

[23] L. Breiman, Random Forests, vol. 45, Berkeley, California: Machine Learning, 2001, pp. 5-32.

[24] C. Hsu, C. Chang and C. Lin, A Practical Guide to Support Vector Classification, Taipei: National Taiwan University, 2003.

[25] C. Peng, K. Lee and G. Ingersoll, An Introduction to Logistic Regression Analysis and Reporting, vol. 96, Bloomington, Indiana: The Journal of Educational Research, 2002, pp. 3-14.

[26] T. Patil and S. Sherekar, Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification, vol. 6, International Journal Of Computer Science And Applications, 2013, pp. 256-261.

[27] P. Bühlmann and T. Hothorn, Boosting Algorithms: Regularization, Prediction and Model Fitting, vol. 22, Zürich: Seminar für Statistik, ETH Zürich, 2007, pp. 477-505.

[28] M. Stone, Cross-validatory choice and assessment of statistical predictions, vol. 36, London: Journal of the Royal Statistical Society, Royal Statistical Society, 1974, pp. 111-147.

[29] P. Refaeilzadeh, L. Tang and H. Lui, Cross-Validation, Encyclopedia of Database Systems, Springer US, 2008, p. 532–538.

[30] AT&T Labs Research, GraphViz: Graph Visualization Software.

[31] B. Li et al., Scalable Random Forests for Massive Data, vol. 7301, Berlin: PAKDD 2012: Advances in Knowledge Discovery and Data Mining, Springer, 2012, pp. 135-146.

[32] V. Vapnik, The nature of statistical learning, vol. 6, New York: Springer-Verlag, 1995.

[33] L. Auria and R. Moro, Support Vector Machines (SVM) as a Technique for Solvency Analysis, vol. 811, Berlin: Deutsches Institut für Wirtschaftsforschung, 2008.

[34] D. Hosmer and S. Lemeshow, Applied logistic regression, vol. 10, New York: Wiley, 1989, pp. 1162-1163.