A PROJECT REPORT

On

Data Mining Techniques to predict Depression for Mental Health Awareness

BY

Muskaan Kumar 2020AAPS2188H

Kushal Mishra 2020A7PS2083H

Rya Sanovar 2020AAPS0306H

Shreya Senapaty 2020AAPS0309H

Under the supervision of

Dr. Apurba Das

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(May, 2022)

CONTENTS

Acknowledgements…………………………………………………………………………...3

Abstract……………………………………………………………………………………...4

Problem Definition………………………………………………………………………….5

State of the Art……………………………………………………………………………..5-6

Novelty……………………………………………………………………………………..…6

Dataset Description…………………………………………………………………………7

Code and Details of techniques used……………………….……………………………8-22

Conclusion and future work………………………..……………… ……………….22-23

Individual Contributions…………………………………………………………………….24

References…………………………………………………………………………………..9

ACKNOWLEDGMENTS

We would like to thank Dr. Apurba Das for giving us the opportunity to apply our knowledge practically for a good cause. He has always been understanding and has motivated us to push our limits and expand our horizon in a field that we are so eager to learn more about. He provided valuable insight during the project and was always present to guide us on the way forward while helping us tackle issues that were very new to us. Lastly, we would like to thank each other, our team who worked tirelessly to make this project a success.

ABSTRACT

A 2012 survey conducted by the World Health Organization (WHO) suggests that depression is a major public health issue that affects 350 million individuals globally. Depression when undiagnosed and untreated, is detrimental to physical health, social relations and quality of life and can also lead to self harm and suicide. Depression is a leading cause of disability around the world and contributes greatly to the global burden of disease. The effects of depression can be long-lasting or recurrent and can dramatically affect a person's ability to function and live a rewarding life. It is associated with disrupted biological rhythms caused by environmental disturbance like seasonal change in daylight, alteration of social rhythms due to for instance shift-work or longitude traveling; besides linked to lifestyles associated with diurnal rhythms inconsistent with the natural daylight cycle. Before depression can be effectively treated and it has to be rightfully recognised. Recognising depression in patients is an ongoing battle due to the complexity of several features used to correctly classify depression. Moreover, the research on identifying depression through motion sensing data is relatively new. In this project we seek to develop various classifiers on a dataset of user information collected through a general survey and motor activity data collected through a smartwatch to find the best algorithm that results in the most accurate classification. Our results will be really valuable in addressing key concerns in countering depression.

[For the purpose of this project, we would rely on the following definition of Depression: *A mental health disorder characterized by persistent feelings of sadness or loss of interest in activities, causing significant impairment in daily life. Possible causes include a combination of biological, psychological and social sources of distress.*]

PROBLEM DEFINITION

Our aim is to utilize survey data regarding a patient’s general details and also their motor-activity and sleep circadian schedule (tracked using an actigraph watch) to develop a system that is capable of automatically detecting depression states based on sensor data. In our experiments we’d like to explore this aspect and other analyses like MADRS score prediction based on motor activity data and sleep pattern analysis of depressed v.s. Non-depressed participants.

STATE OF THE ART

The current progress made so far in mental health prediction (including but not limited to Depression) is credible through these citations of previous works in this field:

U. S. Reddy et al. have applied various algorithms to find the most accurate one and compared the relationship between various parameters in the dataset.

M. P. Dooshima et al. have used demographic, biological, psychological and environmental factors for prediction. Different mental health experts were consulted to validate the obtained parameters.

M. Srividya et al. have used a questionnaire to obtain values for different attributes that can be helpful for prediction of mental health. The motive of this paper was to analyze different algorithms and predict the most accurate one. Various classification algorithms such as Decision Tree, Naïve Bayes as well as SVM were used in this paper. The labels from the data collected were used to compute a MOS. The above algorithms were then applied to find the most accurate one. The paper concluded that Support Vector Machine, K-Nearest Neighbor and Random Forest are the most accurate algorithms with similar accuracy results.

D.Filip & C. Jesus. have used Neural Networks to predict the psychological conditions of humans such as depression, PTSD, anxiety etc. They also studied the effect of concussion or injuries on sportspersons.

S. G. Alonso et al. have conducted extensive review of different algorithms used for mental health prediction. Different techniques such as Association Rule Mining and Randomization were studied and their predictions were noted for our project. This paper also reviewed other algorithms such as SVM, Decision tree, KNN, ANN, Naïve Bayes.

NOVELTY

- There had been little known research on the connection between human rest/activity cycles and their inclination to Depression states.
- A lot of current ongoing research on similar ideas is based on physiological data such as ecg and eeg data requiring a lot of pre-processing as well. However, we've used sensor data which is more accurate in gauging a person's states and sensitivity to factors compared to physiological data.
- In this project we aim to explore this aspect using a comparative study of several data mining techniques such as Support Vector Machines, Decision Trees and the Naïve Bayes Algorithm.
- We've also done a thorough analysis through exploratory data analysis and data visualization to show the correlations between the data and the important factors involved.

DATASET DESCRIPTION

The dataset we'll use in this project contains two folders, where one contains the data for the controls and one for the condition group.
For each patient a CSV file has been provided containing the actigraph data collected over time. The columns are:

- ➢ timestamp (one minute intervals)
- ➢ date (date of measurement)
- ➢ activity (activity measurement from the actigraph watch).

In addition, the MADRS scores are provided in the file "scores.csv".
It contains the following columns:

- ➢ number (patient identifier)
- ➢ days (number of days of measurements)
- ➢ gender (1 or 2 for female or male)
- ➢ age (age in age groups)
- ➢ afftype (1: bipolar II, 2: unipolar depressive, 3: bipolar I)
- ➢ melanch (1: melancholia, 2: no melancholia)
- ➢ inpatient (1: inpatient, 2: outpatient)
- ➢ edu (education grouped in years)
- ➢ marriage (1: married or cohabiting, 2: single)
- ➢ work (1: working or studying, 2: unemployed/sick leave/pension)
- ➢ madrs1 (MADRS score when measurement started)
- ➢ madrs2 (MADRS when measurement stopped).

This publicly available dataset can be found on Kaggle at:
https://www.kaggle.com/arashnic/the-depression-dataset

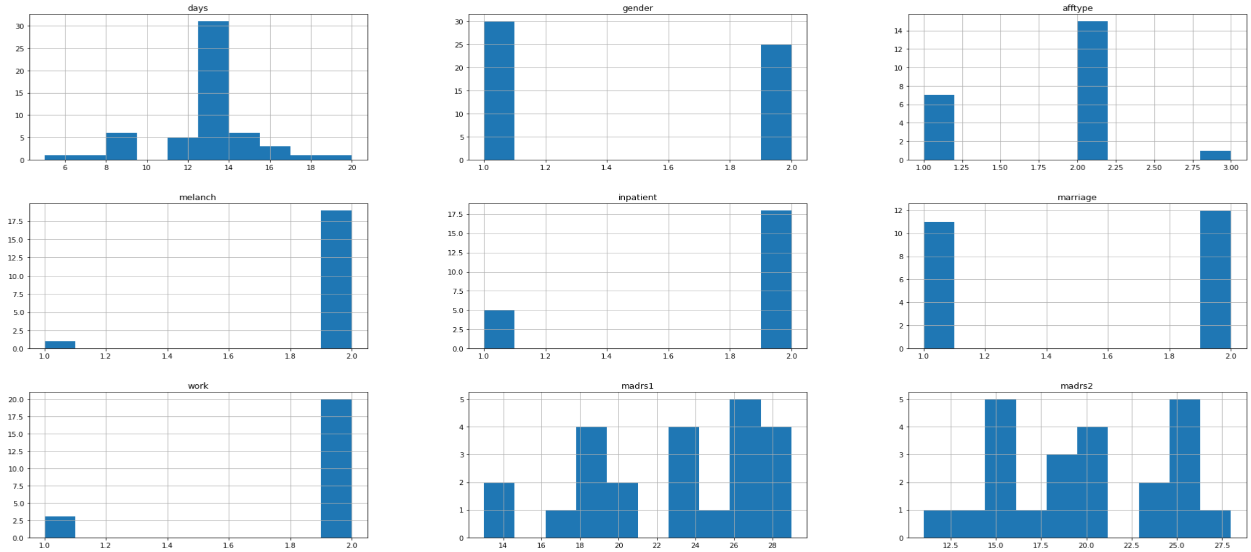
CODE AND DETAILS OF TECHNIQUES USED

EXPLORATORY DATA ANALYSIS and DATA VISUALIZATION

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypotheses and to check assumptions with the help of summary statistics and graphical representations. Below is the comprehensive data exploration we performed on the depression dataset.

This is generally done through data visualization the act of arranging the data in a visual manner to make it easier to understanding conduct exploratory data analysis.

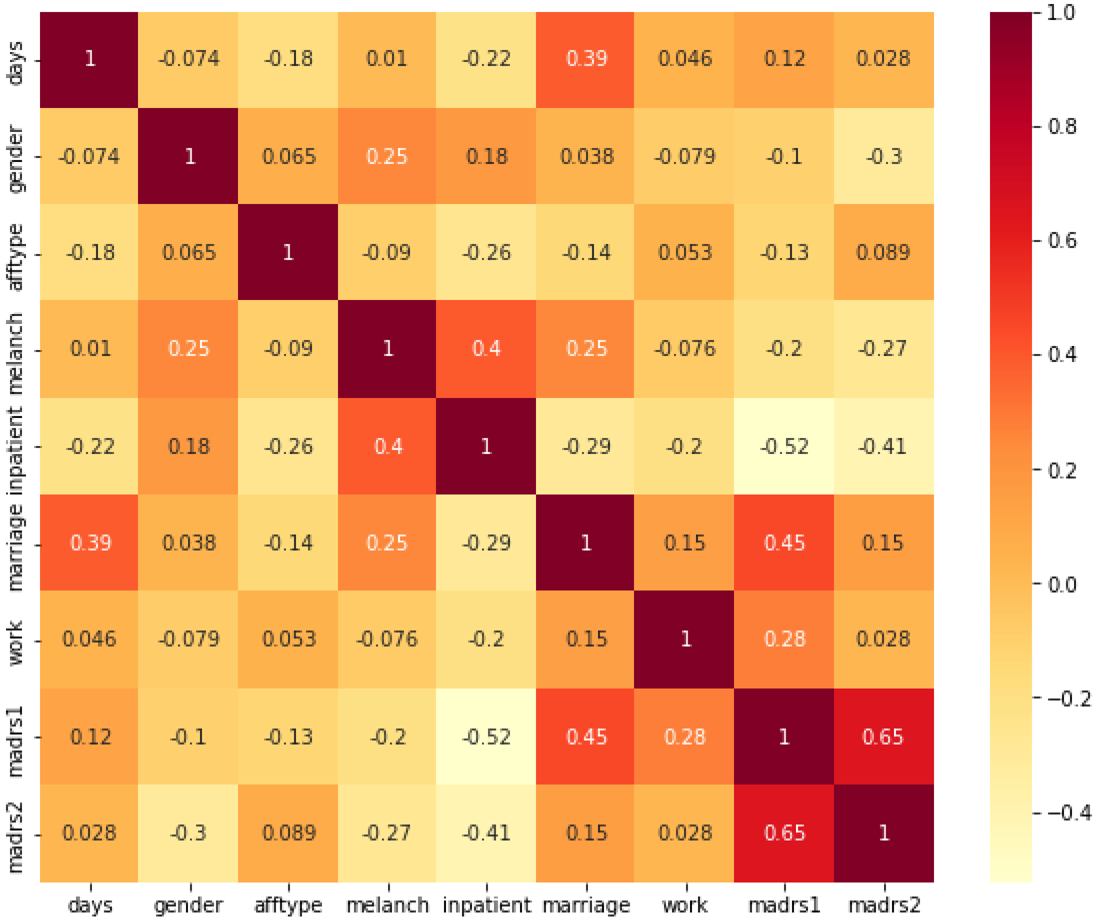
A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables.



To use linear regression for modeling, it's necessary to remove correlated variables to improve your model. One can find correlations using pandas ".corr()" function and can visualize the correlation matrix using a heatmap in seaborn.Here we have done the same for the dataset.

From the heatmap there are a few major correlation points that can be deduced.

- While madrs 1 and 2 are highly related as expected, something of note is that marriage and madrs1 hads a good amount of correlation meaning it's likely that the state of marriage can have an impact on the condition and indicators of depression.
- Similarly while work is less correlated it’s the second highest correlated factor for madrs 1 and hence the workplace environment has an effect on the condition and indicators of depression.

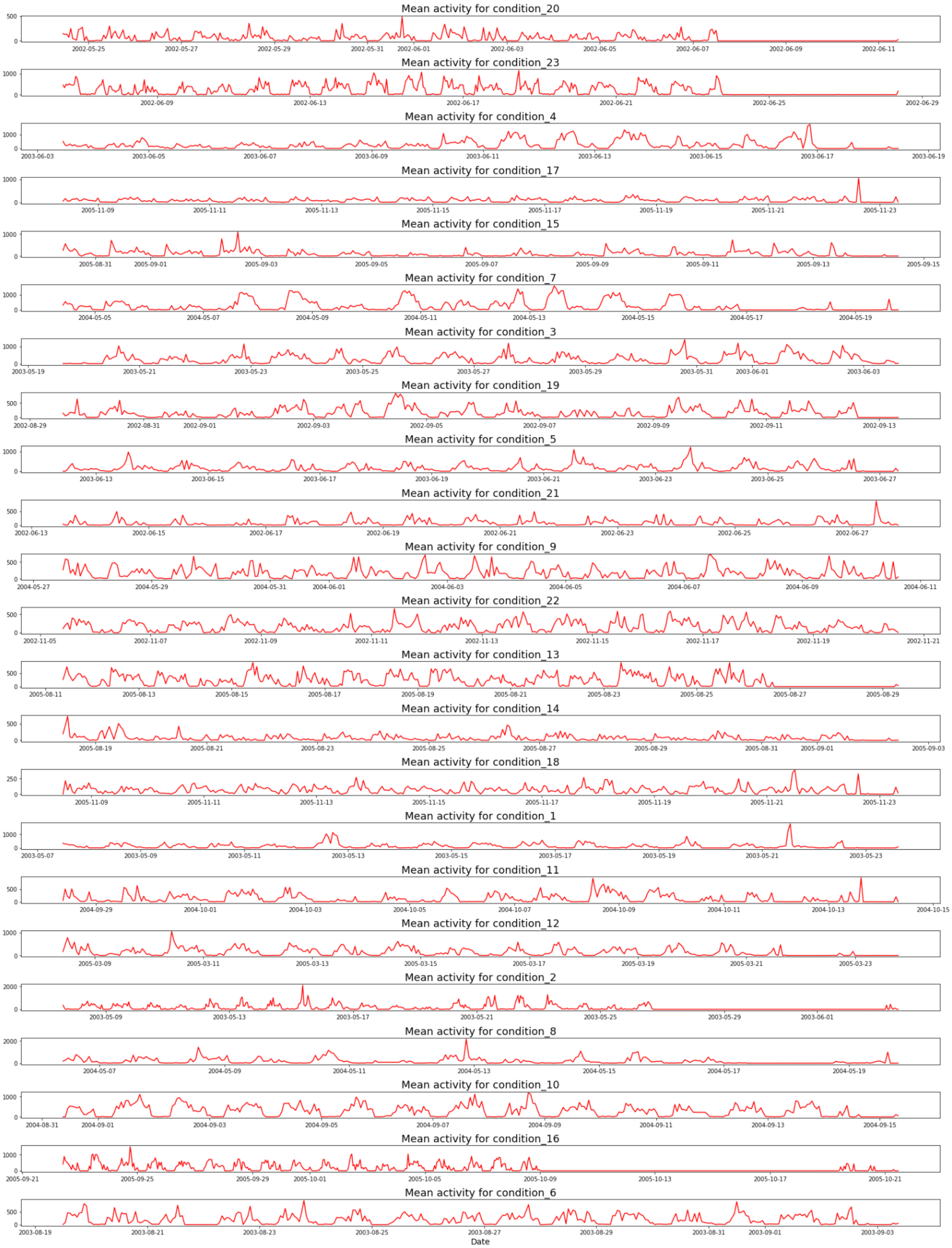


TIME SERIES ANALYSIS:

Time series analysis is a specific way of analyzing a sequence of data points collected over an interval of time. In time series analysis, analysts record data points at consistent intervals over a set period of time rather than just recording the data points intermittently or randomly. However, this type of analysis is not merely the act of collecting data over time.

What sets time series data apart from other data is that the analysis can show how variables change over time.

A very useful tool to help segregate and look at time and date based data, we looked at this through multiple different graphs. One can see the mean activity control in the form of a line graph in terms of the date and time of the data collected for each entry. Similarly the box plots all provide an outlook into comparison of zero activity.



Mean activity for condition_20

Mean activity for condition_23

Mean activity for condition_4

Mean activity for condition_17

Mean activity for condition_15

Mean activity for condition_7

Mean activity for condition_3

Mean activity for condition_19

Mean activity for condition_5

Mean activity for condition_21

Mean activity for condition_9

Mean activity for condition_22

Mean activity for condition_13

Mean activity for condition_14

Mean activity for condition_18

Mean activity for condition_1

Mean activity for condition_11

Mean activity for condition_12

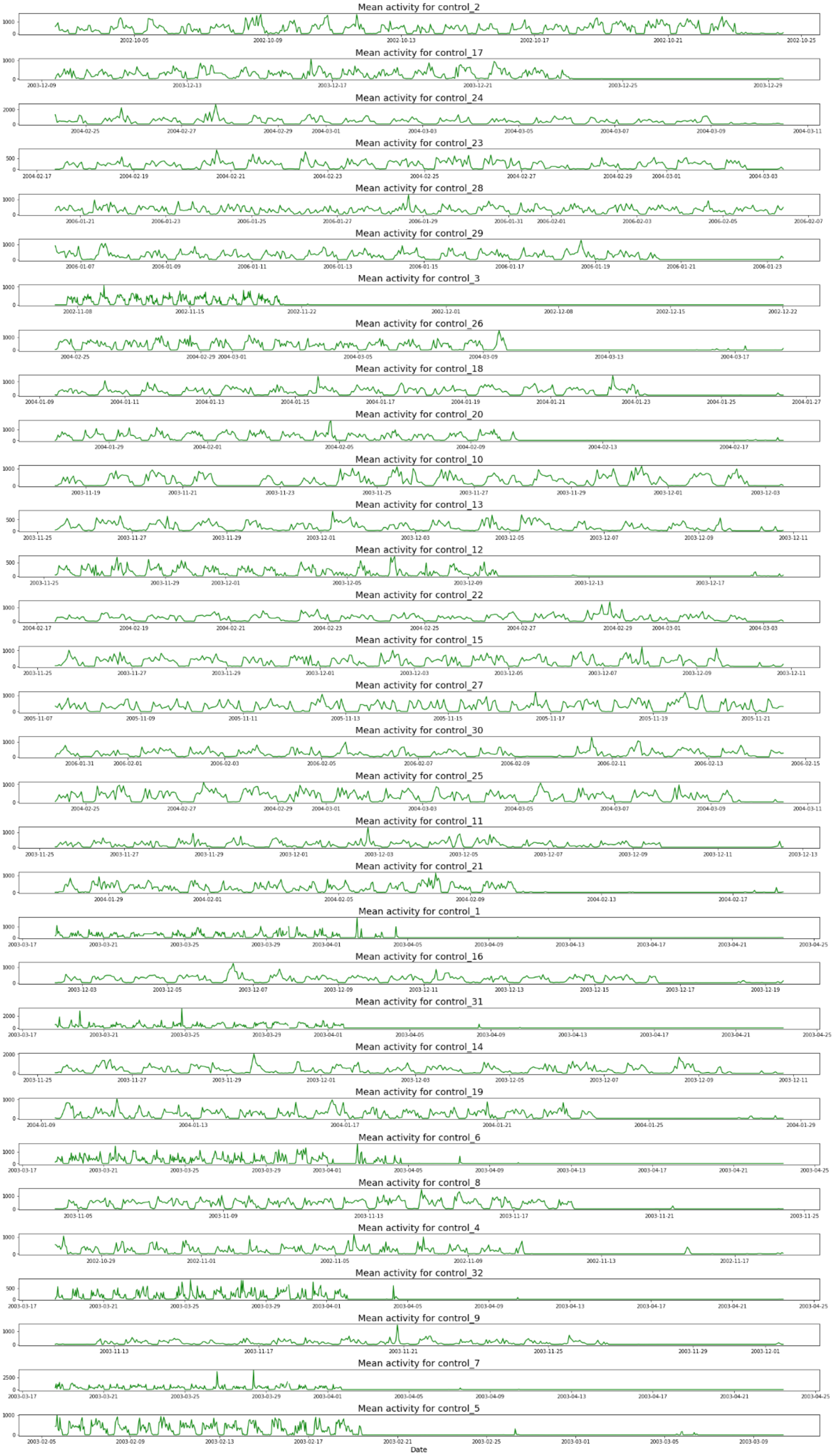
Mean activity for condition_2

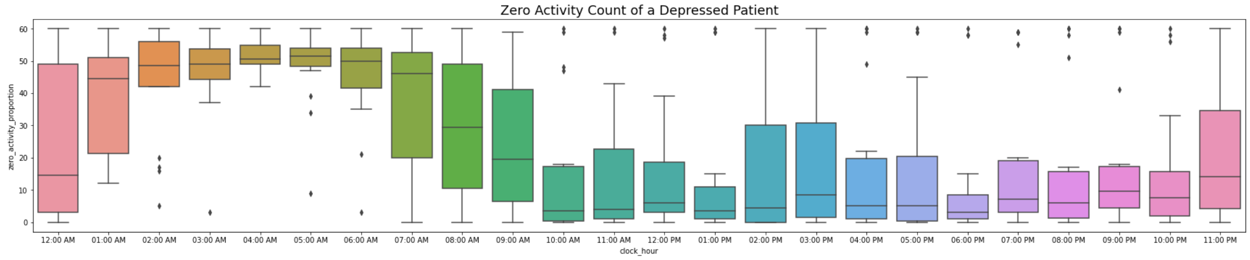
Mean activity for condition_8

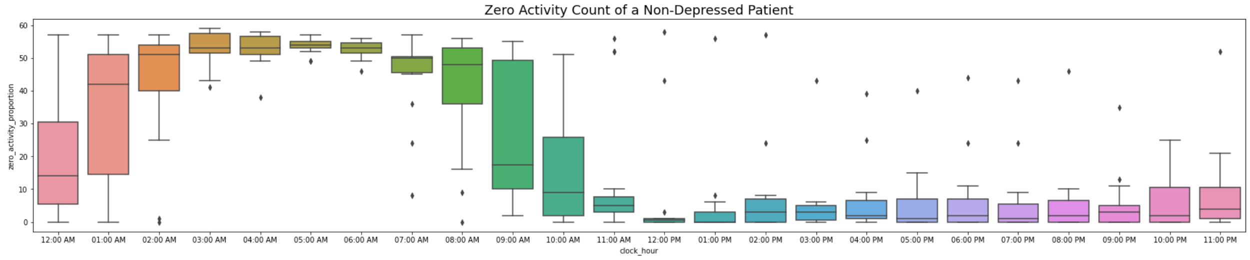
Mean activity for condition_10

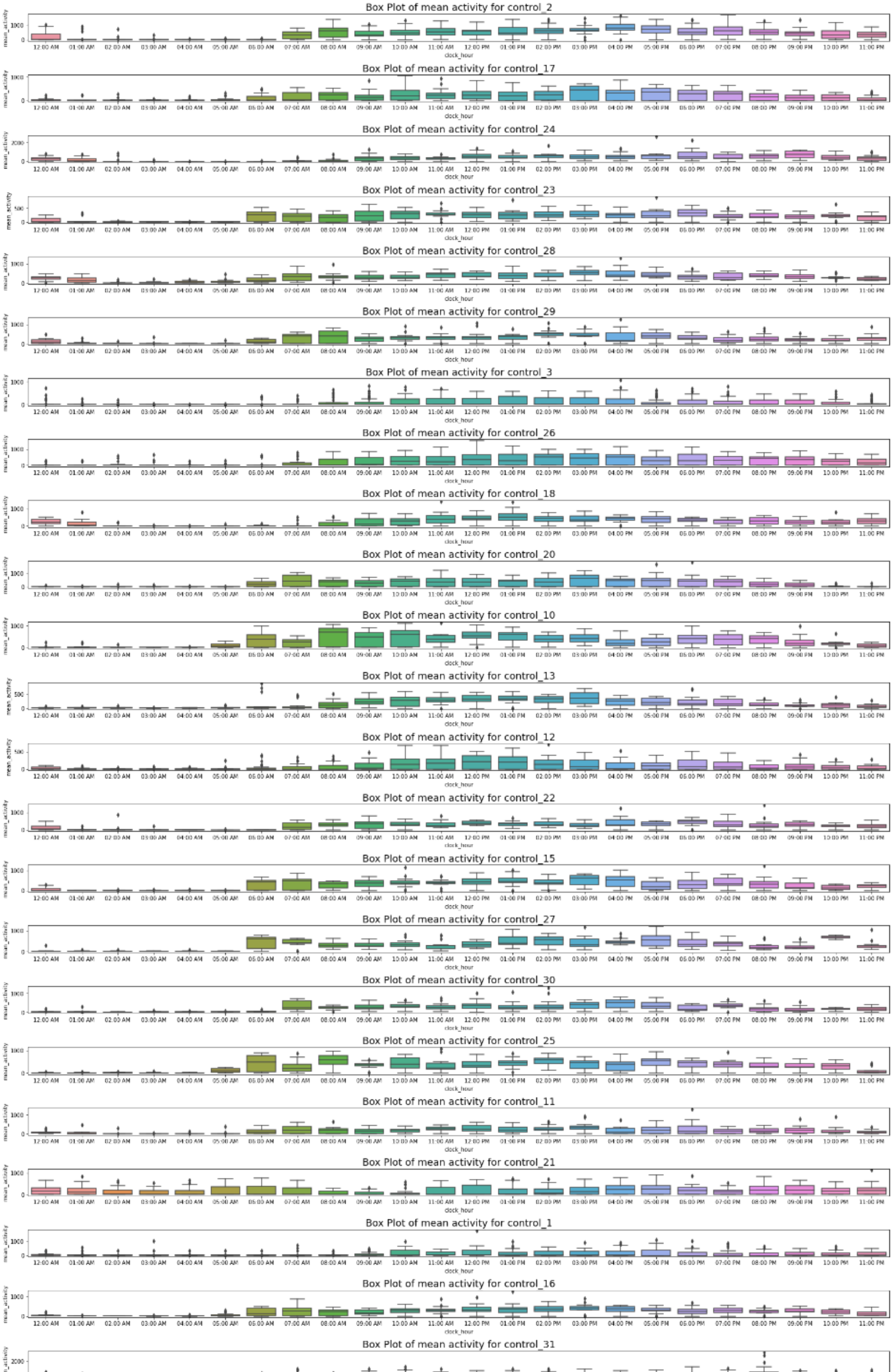
Mean activity for condition_16

Mean activity for condition_6


Mean activity for control_2
Mean activity for control_17
Mean activity for control_24
Mean activity for control_23
Mean activity for control_28
Mean activity for control_29
Mean activity for control_3
Mean activity for control_26
Mean activity for control_18
Mean activity for control_20
Mean activity for control_10
Mean activity for control_13
Mean activity for control_12
Mean activity for control_22
Mean activity for control_15
Mean activity for control_27
Mean activity for control_30
Mean activity for control_25
Mean activity for control_11
Mean activity for control_21
Mean activity for control_1
Mean activity for control_16
Mean activity for control_31
Mean activity for control_14
Mean activity for control_19
Mean activity for control_6
Mean activity for control_8
Mean activity for control_4
Mean activity for control_32
Mean activity for control_9
Mean activity for control_7
Mean activity for control_5
Date


Zero Activity Count of a Depressed Patient
zero_activity_proportion
0
10
20
30
40
50
60
12:00 AM
01:00 AM
02:00 AM
03:00 AM
04:00 AM
05:00 AM
06:00 AM
07:00 AM
08:00 AM
09:00 AM
10:00 AM
11:00 AM
12:00 PM
01:00 PM
02:00 PM
03:00 PM
04:00 PM
05:00 PM
06:00 PM
07:00 PM
08:00 PM
09:00 PM
10:00 PM
11:00 PM
clock_hour


Zero Activity Count of a Non-Depressed Patient
zero_activity_proportion
0
10
20
30
40
50
60
12:00 AM
01:00 AM
02:00 AM
03:00 AM
04:00 AM
05:00 AM
06:00 AM
07:00 AM
08:00 AM
09:00 AM
10:00 AM
11:00 AM
12:00 PM
01:00 PM
02:00 PM
03:00 PM
04:00 PM
05:00 PM
06:00 PM
07:00 PM
08:00 PM
09:00 PM
10:00 PM
11:00 PM
clock_hour



Box Plot of mean activity for control_2

Box Plot of mean activity for control_17

Box Plot of mean activity for control_24

Box Plot of mean activity for control_23

Box Plot of mean activity for control_28

Box Plot of mean activity for control_29

Box Plot of mean activity for control_3

Box Plot of mean activity for control_26

Box Plot of mean activity for control_18

Box Plot of mean activity for control_20

Box Plot of mean activity for control_10

Box Plot of mean activity for control_13

Box Plot of mean activity for control_12

Box Plot of mean activity for control_22

Box Plot of mean activity for control_15

Box Plot of mean activity for control_27

Box Plot of mean activity for control_30

Box Plot of mean activity for control_25

Box Plot of mean activity for control_11

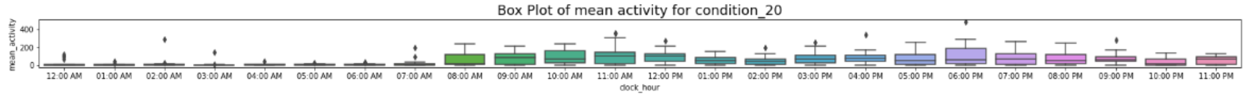
Box Plot of mean activity for control_21

Box Plot of mean activity for control_1

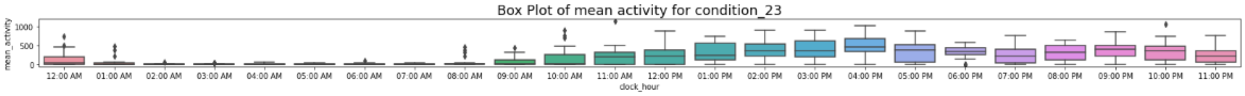
Box Plot of mean activity for control_16

Box Plot of mean activity for control_31

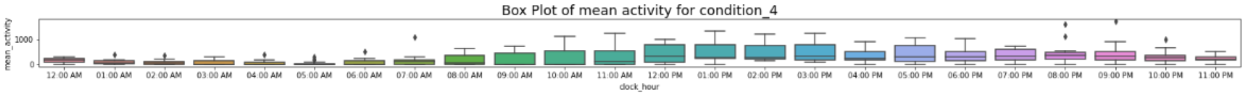
Box Plot of mean activity for condition_20



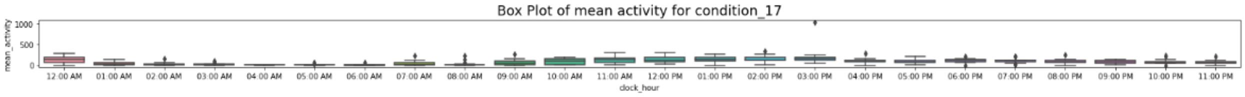
Box Plot of mean activity for condition_23



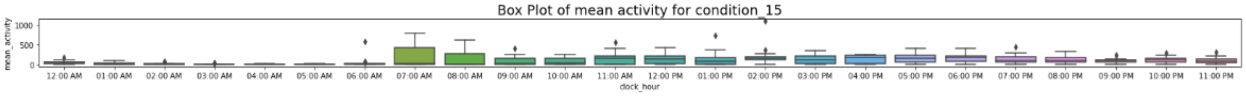
Box Plot of mean activity for condition_4



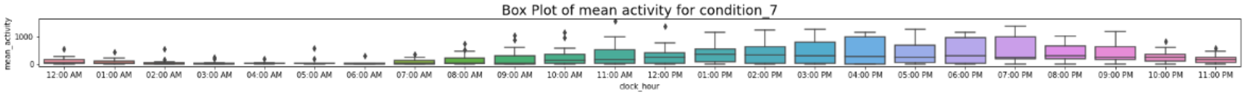
Box Plot of mean activity for condition_17



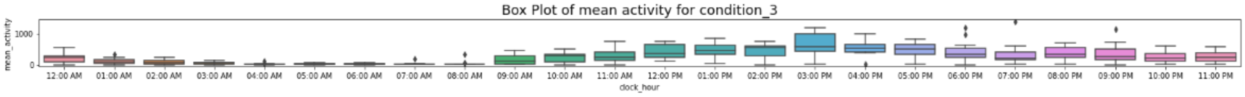
Box Plot of mean activity for condition_15



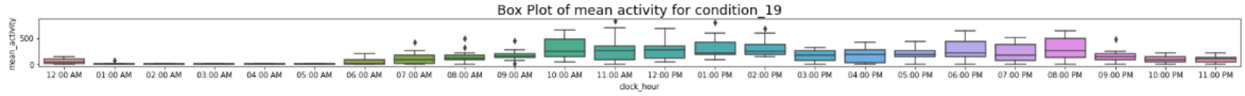
Box Plot of mean activity for condition_7



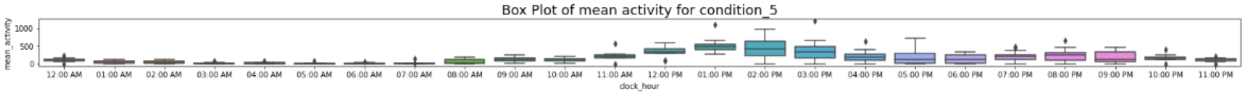
Box Plot of mean activity for condition_3



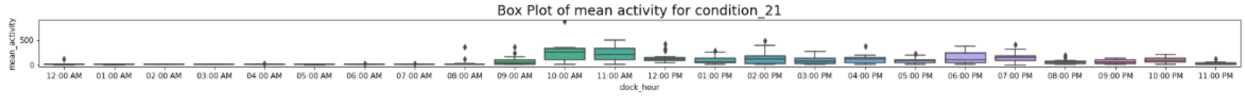
Box Plot of mean activity for condition_19



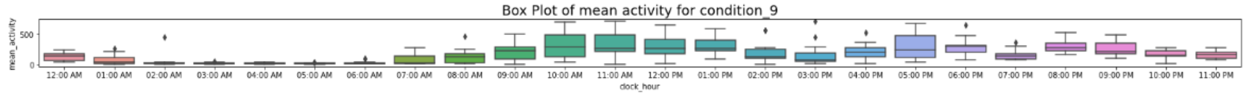
Box Plot of mean activity for condition_5



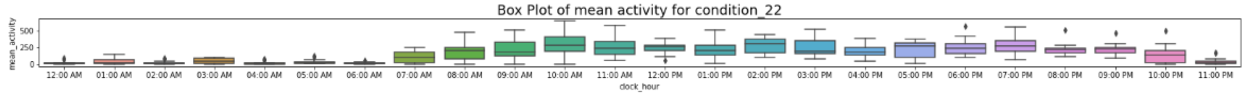
Box Plot of mean activity for condition_21



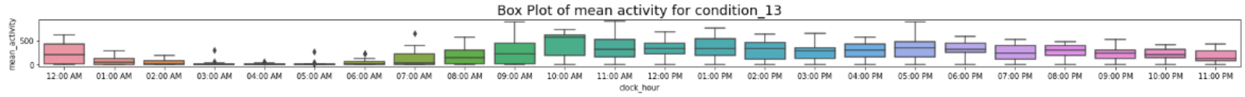
Box Plot of mean activity for condition_9



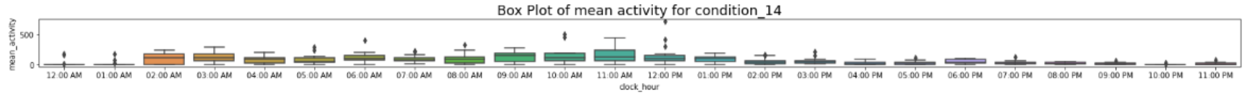
Box Plot of mean activity for condition_22



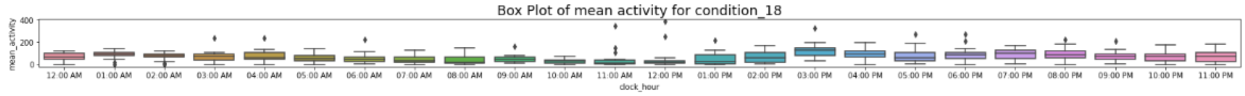
Box Plot of mean activity for condition_13



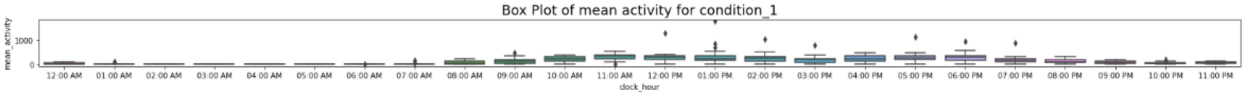
Box Plot of mean activity for condition_14



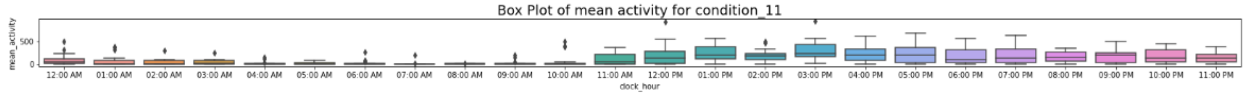
Box Plot of mean activity for condition_18



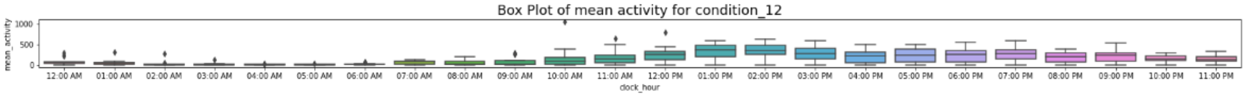
Box Plot of mean activity for condition_1



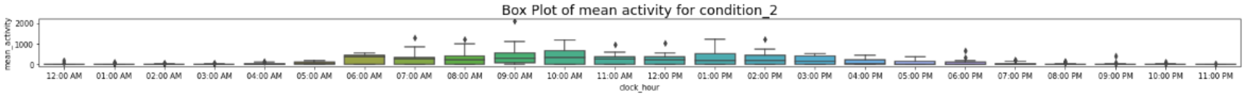
Box Plot of mean activity for condition_11



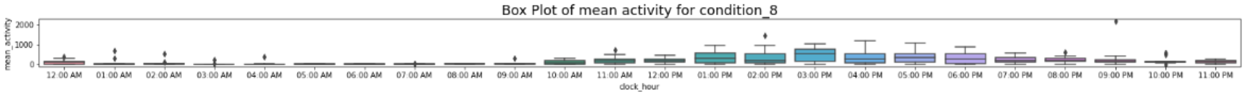
Box Plot of mean activity for condition_12



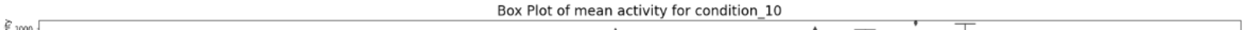
Box Plot of mean activity for condition_2



Box Plot of mean activity for condition_8



Box Plot of mean activity for condition_10



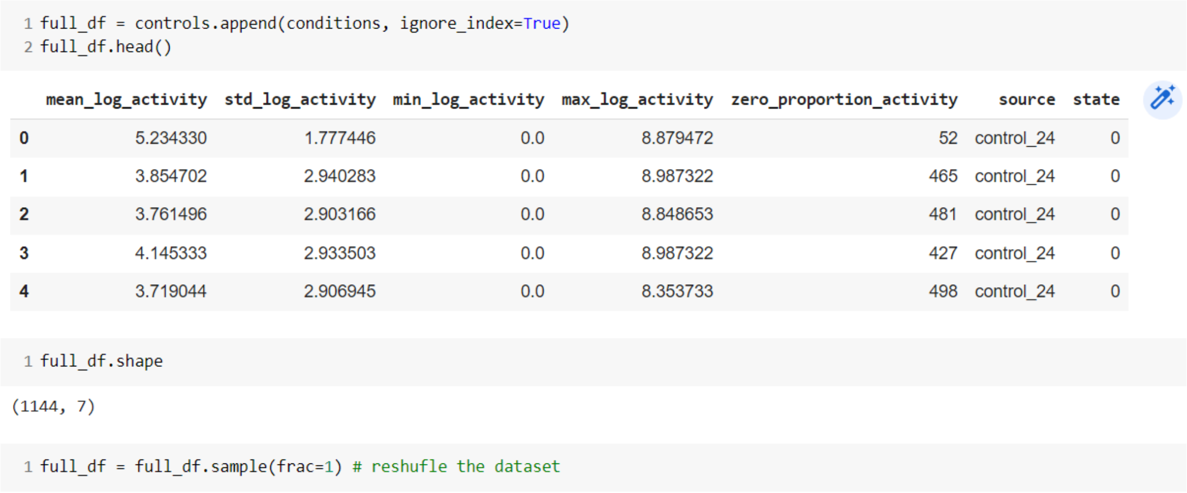
NAIVE BAYES CLASSIFIER:

Naive Bayes Classifier is a statistical classifier that predicts class membership probabilities such as the probability that a given tuple belongs to a particular class. It is based on Bayes theorem. The naive assumption of class-conditional independencies made. This presumes that the attributes' values are conditionally independent of one another, given the class label of the tuple. It is observed that accuracy improves as the size of the dataset increases. The naive bayes classifier we've used here has been written completely from scratch.

The work on this classifier has been divided into 2 parts:

1. Where we use only sensor data to predict depressed or healthy patients.
2. Here we use the scores file to predict MADRS score after the measurements.

After the time series analysis it's time to model the dataset. The ExtractData() function when called with a path of the Conditions files, extracts them all and returns a dataframe with features like mean_log_activity etc which have been calculated for each day of the experiment using functions nextday(), zero_count(), extractfeatures(). The clinically depressed patients whose data had been recorded in the conditions file have been assigned a state of 1. The same code has been implemented on the control file data. The healthy patients whose data had been recorded in the conditions file have been assigned a state of 0.



```
full_df = controls.append(conditions, ignore_index=True)
full_df.head()
```

	mean_log_activity	std_log_activity	min_log_activity	max_log_activity	zero_proportion_activity	source	state
0	5.234330	1.777446	0.0	8.879472	52	control_24	0
1	3.854702	2.940283	0.0	8.987322	465	control_24	0
2	3.761496	2.903166	0.0	8.848653	481	control_24	0
3	4.145333	2.933503	0.0	8.987322	427	control_24	0
4	3.719044	2.906945	0.0	8.353733	498	control_24	0

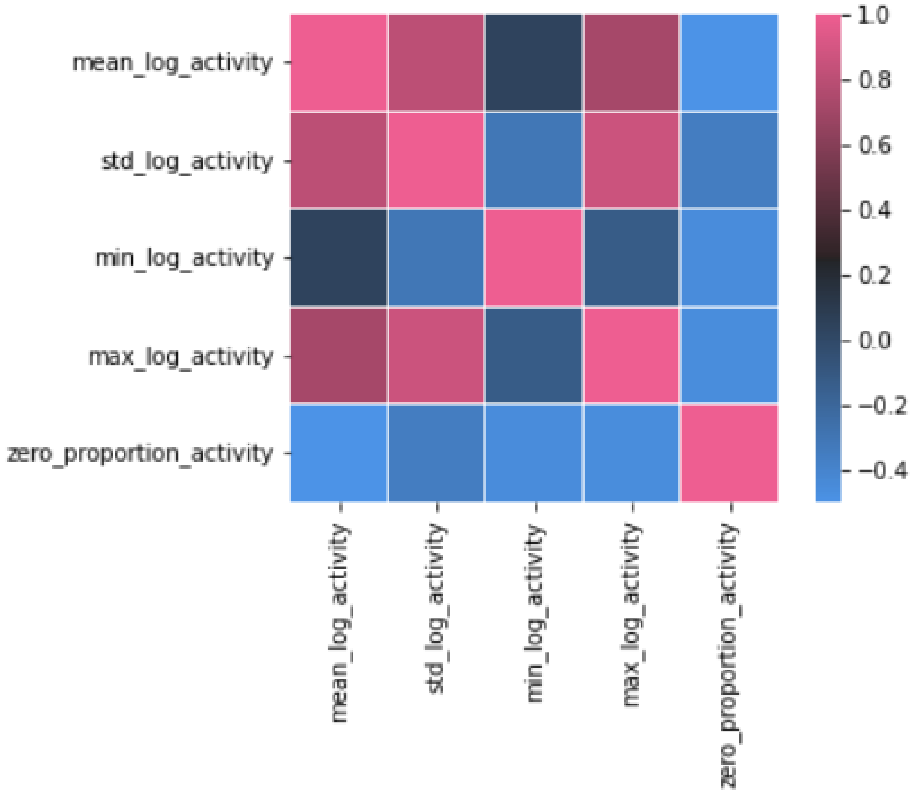
```
full_df.shape
```

```
(1144, 7)
```

```
full_df = full_df.sample(frac=1) # reshufle the dataset
```

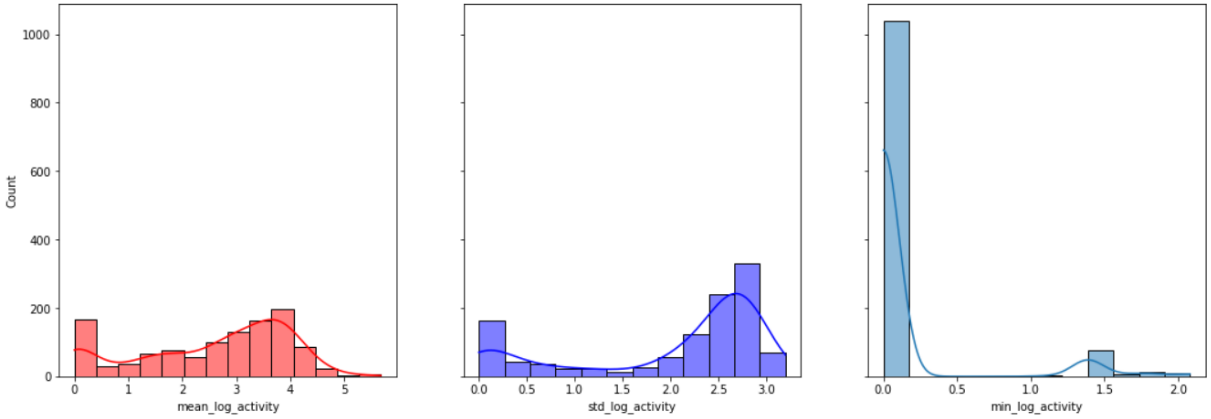
A generated heat map shown below reveals to us that dropping other features except standard, mean and minimum log activity due to high correlation can satisfy our assumption that features are linearly independent.

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0797702d0>
```



Next, we've plotted individual histograms (as seen below) for features: mean_log_activity, std_log_activity, and min_log_activity to see how they fit under the Gaussian distribution. We can see that mean and standard deviation of log activity fills pretty well, but the minimum log activity doesn't. We will still use Gaussian distribution for it, for simplicity's sake.

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe06b63ba50>
```



In the dataframe full_df, the attributes are mean_log_activity, std_log_activtiy, min_log_activity of the collected sensor data.
The value of Y to be predicted is in binary form i.e 2 options: state 0 for healthy patients and state 1 for depressed patients.

We calculate conditional probability P(x/y) using Gaussian distribution and the assumption that the features are independent of each other.

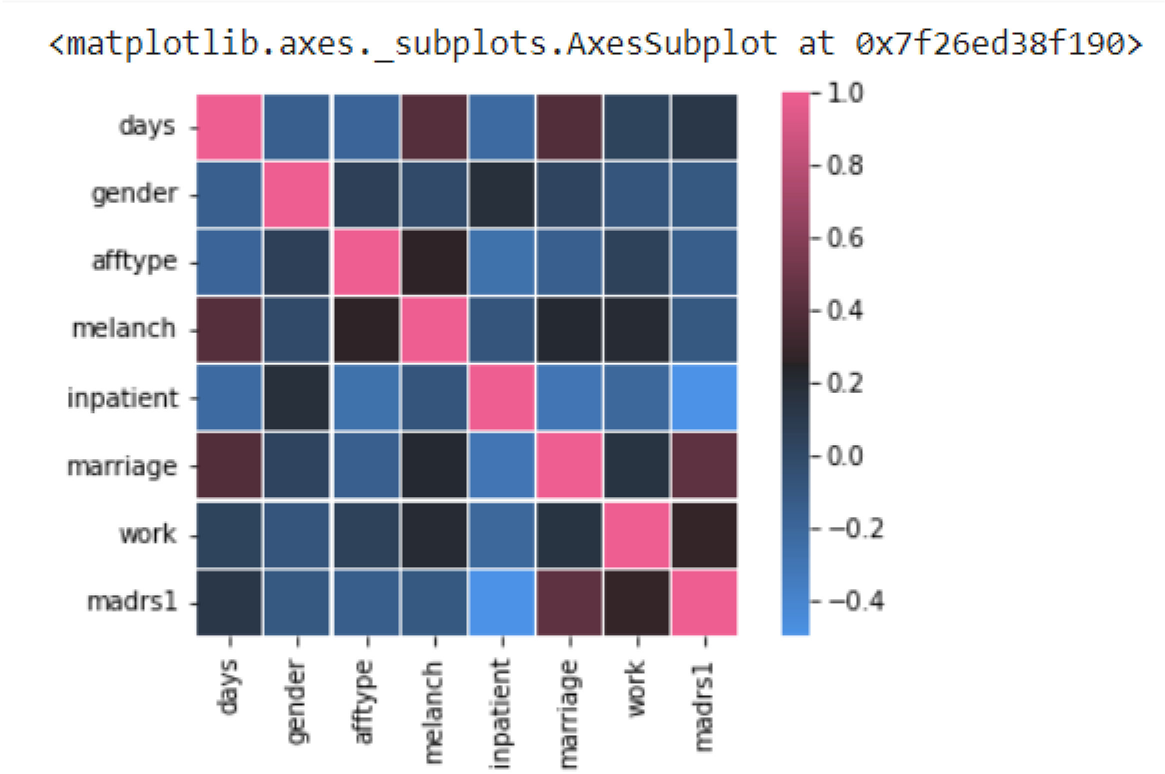
The performance measures of classifiers measure the decision making capability of the classifier. The measures used to determine the performance are: accuracy, precision, recall, F-score. The overall accuracy measure is rarely considered adequate for a classifier.

Precision is in simple words positive predictive value. Recall on the other hand is the true positive rate. A model could have perfect recall but low precision and accuracy hence why we need to look at both factors and not just any one. We've used f1 score as a metric of baseline performance of this classification task. It provides the harmonic mean of precision and recall.

The f1 score we've got from applying the sensor dataset on this classifier is 0.518.

Next, we use the scores file to predict MADRS scores after measurement. Before that, we implement some basic data preprocessing where we replace the range of age and the range of years of education with a mean value.

The generated heat map for this dataset is shown below.



```
<matplotlib.axes._subplots.AxesSubplot at 0x7f26ed38f190>
```

The final features we've selected are gender, age, afftype, melanch, edu, inpatient, marriage, work, mards1 and the value to be predicted is madrs2.

Before we start applying the algorithm, we need to convert continuous integral data of age, education years and madrs1 and 2 columns to categorical data by dividing them into separate bins of 0, 1 and 2.
We calculate the likelihood P(X=x/Y=y) categorically.

Finally applying the Naive Bayes algorithm to this dataset gives us an f1 score of 0.67.

Conclusion for naive bayes:
Naïve Bayes does not give a good performance as the assumption of class conditional independence will work only for a huge dataset.

The code for this algorithm is available on the ipynb file shared along with this report.

SVM CLASSIFIER:

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well, it's best suited for classification. The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

SVM solves the problem of binary classification using the convex optimization method.

The model takes into account the features:

- Gender
- Age
- Melanch
- Inpatient status
- Education level
- Relationship status
- Employment status

And uses the above to predict MADRS1 and MADRS2 scores

In order to make the predictions, there are two sets of dependent and independent variables made, namely y_1, y_2, X_1 and X_2 respectively. The independent variables in X_i use the other (3 - i)th MADRS column as a feature.

The model makes use of sci-kit learn’s implementation of the SVM via the support vector classifier sub-module. A ‘linear’ kernel function is used for the SVC.

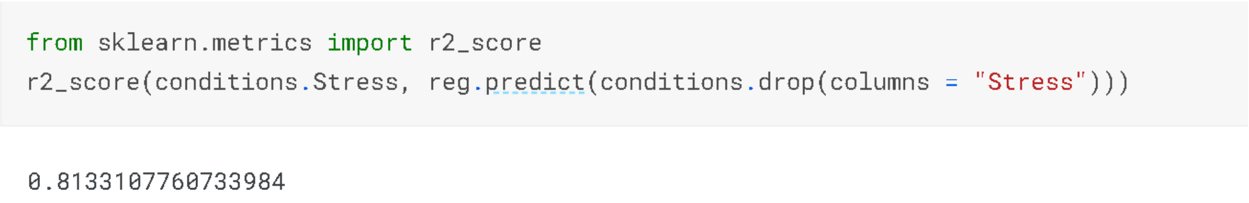
The model achieves an accuracy of 50%. This can be improved by utilising a larger dataset or using k-fold cross validation with a high ‘k’-value given the relatively small size of the dataset.

RANDOM FOREST CLASSIFIER

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, a random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

It also helps give you an idea of precision and recall both which as explained earlier helps give a more accurate picture of the model's performance.

In this code, we have implemented random forest with the help of sklearn library. The steps included preprocessing the data using StandardScaler so that it fit the function. Using sklearn metrics we measured the stress factor.

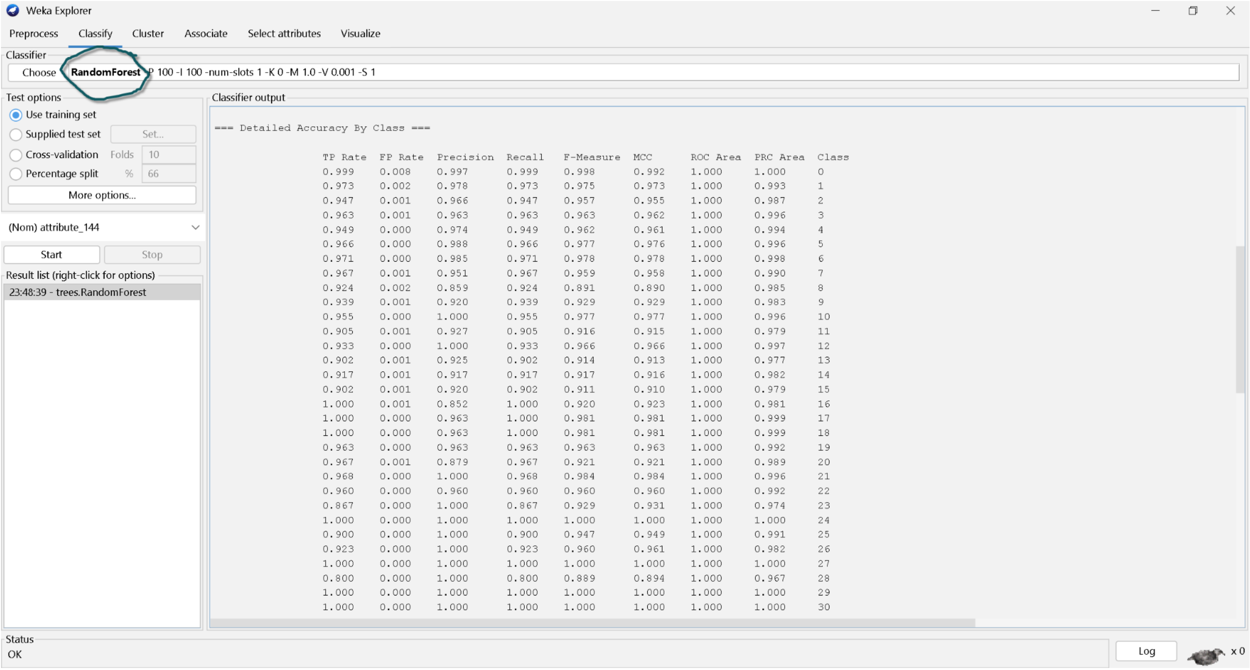


```
from sklearn.metrics import r2_score
r2_score(conditions.Stress, reg.predict(conditions.drop(columns = "Stress")))
```

```
0.8133107760733984
```

VALIDATING SCORES USING WEKA

We tried exploring the data mining tool- WEKA
Weka is a collection of machine learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization.
Here we used Random Forest



Classifier: RandomForest P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options
- ● Use training set
- ○ Supplied test set
- ○ Cross-validation Folds 10
- ○ Percentage split % 66

(Nom) attribute_144

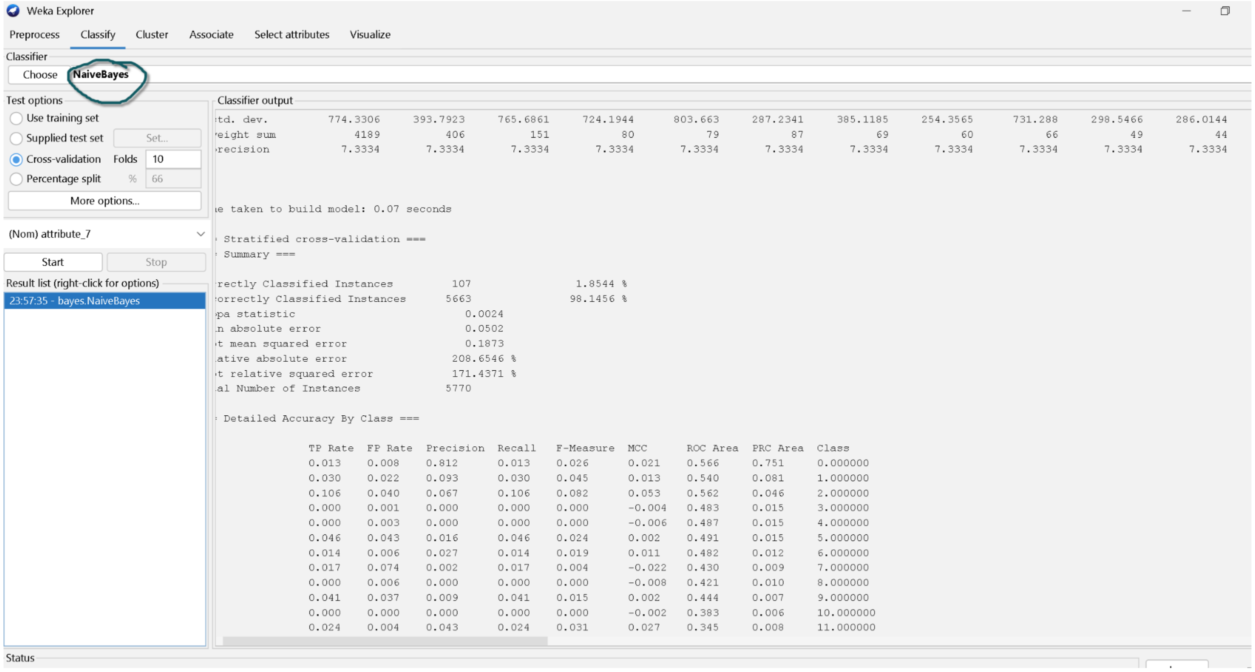
Result list (right-click for options)
23:48:39 - trees.RandomForest

Status
OK

Classifier output

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.999    0.008    0.997      0.999    0.998      0.992    1.000     1.000     0
                 0.973    0.002    0.978      0.973    0.975      0.973    1.000     0.993     1
                 0.947    0.001    0.966      0.947    0.957      0.955    1.000     0.987     2
                 0.963    0.001    0.963      0.963    0.963      0.962    1.000     0.996     3
                 0.949    0.000    0.974      0.949    0.962      0.961    1.000     0.994     4
                 0.966    0.000    0.988      0.966    0.977      0.976    1.000     0.996     5
                 0.971    0.000    0.985      0.971    0.978      0.978    1.000     0.998     6
                 0.967    0.001    0.951      0.967    0.959      0.958    1.000     0.990     7
                 0.924    0.002    0.859      0.924    0.891      0.890    1.000     0.985     8
                 0.939    0.001    0.920      0.939    0.929      0.929    1.000     0.983     9
                 0.955    0.000    1.000      0.955    0.977      0.977    1.000     0.996     10
                 0.905    0.001    0.927      0.905    0.916      0.915    1.000     0.979     11
                 0.933    0.000    1.000      0.933    0.966      0.966    1.000     0.997     12
                 0.902    0.001    0.925      0.902    0.914      0.913    1.000     0.977     13
                 0.917    0.001    0.917      0.917    0.917      0.916    1.000     0.982     14
                 0.902    0.001    0.920      0.902    0.911      0.910    1.000     0.979     15
                 1.000    0.001    0.852      1.000    0.920      0.923    1.000     0.981     16
                 1.000    0.000    0.963      1.000    0.981      0.981    1.000     0.999     17
                 1.000    0.000    0.963      1.000    0.981      0.981    1.000     0.999     18
                 0.963    0.000    0.963      0.963    0.963      0.963    1.000     0.992     19
                 0.967    0.001    0.879      0.967    0.921      0.921    1.000     0.989     20
                 0.968    0.000    1.000      0.968    0.984      0.984    1.000     0.996     21
                 0.960    0.000    0.960      0.960    0.960      0.960    1.000     0.992     22
                 0.867    0.000    1.000      0.867    0.929      0.931    1.000     0.974     23
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     24
                 0.900    0.000    1.000      0.900    0.947      0.949    1.000     0.991     25
                 0.923    0.000    1.000      0.923    0.960      0.961    1.000     0.982     26
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     27
                 0.800    0.000    1.000      0.800    0.889      0.894    1.000     0.967     28
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     29
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     30
```

Naive Bayes



Classifier: NaiveBayes

Test options
- ○ Use training set
- ○ Supplied test set
- ● Cross-validation Folds 10
- ○ Percentage split % 66

(Nom) attribute_7

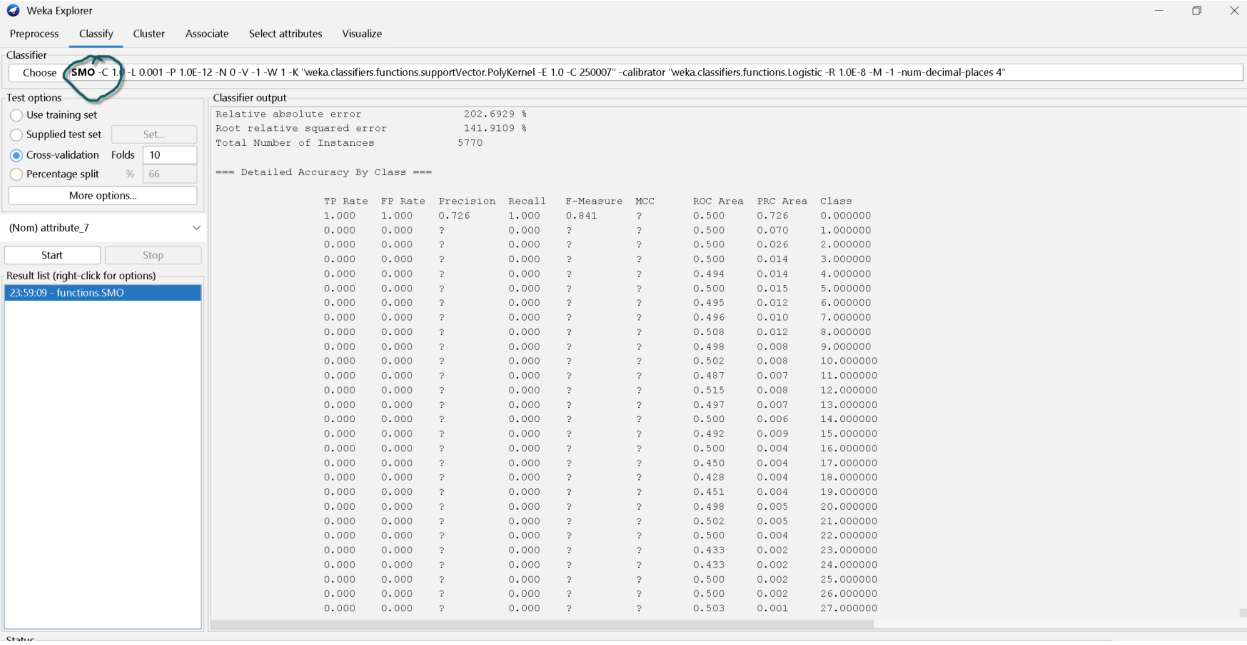
Result list (right-click for options)
23:57:35 - bayes.NaiveBayes

Status

Classifier output

```
td. dev.        774.3306    393.7923    765.6861    724.1944     803.663    287.2341    385.1185    254.3565     731.288    298.5466    286.0144
eight sum           4189         406         151          80          79          87          69          60          66          49          44
recision          7.3334      7.3334      7.3334      7.3334      7.3334      7.3334      7.3334      7.3334      7.3334      7.3334      7.3334



e taken to build model: 0.07 seconds

 Stratified cross-validation ===
 Summary ===

rectly Classified Instances         107                1.8544 %
orrectly Classified Instances      5663               98.1456 %
pa statistic                          0.0024
n absolute error                      0.0502
t mean squared error                  0.1873
ative absolute error                208.6546 %
t relative squared error            171.4371 %
al Number of Instances             5770

 Detailed Accuracy By Class ===

              TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
              0.013    0.008    0.812      0.013    0.026      0.021    0.566     0.751     0.000000
              0.030    0.022    0.093      0.030    0.045      0.013    0.540     0.081     1.000000
              0.106    0.040    0.067      0.106    0.082      0.053    0.562     0.046     2.000000
              0.000    0.001    0.000      0.000    0.000      -0.004   0.483     0.015     3.000000
              0.000    0.003    0.000      0.000    0.000      -0.006   0.487     0.015     4.000000
              0.046    0.043    0.016      0.046    0.024      0.002    0.491     0.015     5.000000
              0.014    0.006    0.027      0.014    0.019      0.011    0.482     0.012     6.000000
              0.017    0.074    0.002      0.017    0.004      -0.022   0.430     0.009     7.000000
              0.000    0.006    0.000      0.000    0.000      -0.008   0.421     0.010     8.000000
              0.041    0.037    0.009      0.041    0.015      0.002    0.444     0.007     9.000000
              0.000    0.000    0.000      0.000    0.000      -0.002   0.383     0.006     10.000000
              0.024    0.004    0.043      0.024    0.031      0.027    0.345     0.008     11.000000
```

SVM using SMOTE function



Classifier: SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -calibrator "weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4"

Test options

○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) attribute_7

Result list (right-click for options)

23:59:09 - functions.SMO

Classifier output

```
Relative absolute error                202.6929 %
Root relative squared error            141.9109 %
Total Number of Instances             5770

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.726      1.000    0.841      ?        0.500     0.726     0.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.070     1.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.026     2.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.014     3.000000
                 0.000    0.000    ?          0.000    ?          ?        0.494     0.014     4.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.015     5.000000
                 0.000    0.000    ?          0.000    ?          ?        0.495     0.012     6.000000
                 0.000    0.000    ?          0.000    ?          ?        0.496     0.010     7.000000
                 0.000    0.000    ?          0.000    ?          ?        0.508     0.012     8.000000
                 0.000    0.000    ?          0.000    ?          ?        0.498     0.008     9.000000
                 0.000    0.000    ?          0.000    ?          ?        0.502     0.008     10.000000
                 0.000    0.000    ?          0.000    ?          ?        0.487     0.007     11.000000
                 0.000    0.000    ?          0.000    ?          ?        0.515     0.008     12.000000
                 0.000    0.000    ?          0.000    ?          ?        0.497     0.007     13.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.006     14.000000
                 0.000    0.000    ?          0.000    ?          ?        0.492     0.009     15.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.004     16.000000
                 0.000    0.000    ?          0.000    ?          ?        0.450     0.004     17.000000
                 0.000    0.000    ?          0.000    ?          ?        0.428     0.004     18.000000
                 0.000    0.000    ?          0.000    ?          ?        0.451     0.004     19.000000
                 0.000    0.000    ?          0.000    ?          ?        0.498     0.005     20.000000
                 0.000    0.000    ?          0.000    ?          ?        0.502     0.005     21.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.004     22.000000
                 0.000    0.000    ?          0.000    ?          ?        0.433     0.002     23.000000
                 0.000    0.000    ?          0.000    ?          ?        0.433     0.002     24.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.002     25.000000
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.002     26.000000
                 0.000    0.000    ?          0.000    ?          ?        0.503     0.001     27.000000
```

Conclusion and Future Work:

As the world begins to understand the importance of mental health and its effect on the various aspects of our lives, we hope more data will be gathered regarding the same in the coming years, with more unique attributes combined with a non-restricting sample size. Our models were able to make predictions with what little data was given to them with a satisfactory level of accuracy.

We chose this niche field because although there was not much data available on the same, the drive to bring awareness to this issue more than outweighed the lack of sufficient data.

We successfully managed to apply three data mining techniques to the same which have been explained in detail above. Of the three techniques, the decision tree classifier could not overcome some of the inconsistencies in the data even after having analyzed it carefully. It would be interesting to know the results of the decision tree classifier, as it seems that the particular method might render better results given a larger and more viable dataset.

We also did a comparison of data mining techniques using WEKA which confirmed that SVM had the most accuracy.

Finally, we hope our work motivates others to take a closer look into this niche field and find new information that might have evaded us, by applying the latest techniques of the future.

References:

Research on student mental health based on data mining algorithms by Mengjun Luo:
https://www.hindawi.com/journals/jhe/2021/1382559/

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3561661

https://link.springer.com/article/10.1007/s10916-018-1018-2

TEAM CONTRIBUTIONS:

Rya Sanovar -

- Implementation of Naive Bayes Algorithm
- Coding the Time Series Analysis and Dataset analysis
- Data preprocessing of both datasets
- Dataset modeling
- Abstract, Problem Definition, Dataset description, Naive Bayes algorithm in the Project report

Kushal Mishra -

- Implementation of SVM algorithm
- SVM Algorithm description,
- Conclusion and Future Work in the project report
- Debugging

Shreya Senapaty -

- Data Visualization
- Data Pre-processing
- Implementation of Random Forest Classifier
- Novelty, Conclusion and Future Work, EDA analysis, Data Visualization, Time Series Analysis and Random Forest Classifier in Project Report
- Debugging

Muskaan Kumar -

- Data Preprocessing
- EDA Analysis
- Implementation of Decision trees/ random forest classifier.

- Explanation of EDA, random forest and WEKA in project report
- Debugging
- Used WEKA to cross check accuracy between different models implemented.