Mental Health Predictions

Using Machine Learning Algorithms

CS725: Foundations of Machine Learning
Autumn Semester, 2022
Project Report

Bhawna Chelani	22M0751
Nilava Sarkar	22M0753
Bhumika Khetan	22M0754
Tathagata Dey	22M0765
Saurabh Kumar	22M0802

Contents

1. Abstract

'Epidemiological studies suggest that approximately 20% of the Indian population suffer from some form of mental disorder.'[13] The early detection of anxiety, depression and other form of mental disorder can help to start the treatment at an early stage and thus curing and reducing it. It can prevent various side-effects of these disorders including heart diseases and suicidal instincts thus improving the quality of life. The objective of this project is to determine the mental state of an individual on the basis of its mental and physical well-being and its environment, and indicate whether professional help is required or not. For this project, we used various machine learning algorithms including Decision trees, Stacking, Logistic Regression, SVM, Random Forest, Neural Networks, Gradient Boost and Adaboost. The resulting outcomes are compared and used for selecting the appropriate model for the predictions.

2. Introduction

Mental health of a person is defined in-terms of various aspects including emotional health, psychological health, and social well-being. Our mental health influences our thoughts, emotions and behaviours. A mentally healthy person differs from a mentally unfit one in ways they think, behave, take decisions, handle stress and interact with others. Not being mentally well, in some way, affects the general health of a person too. According to WHO reports, 'In 2019, 1 in every 8 people, or 970 million people around the world were living with a mental disorder. In 2020, the number of people living with anxiety and depressive disorders rose significantly because of the COVID-19 pandemic. Initial estimates show a 26% and 28% increase respectively for anxiety and major depressive disorders in just one year [**?**]. Mental disorders include anxiety and depression, the most common ones, and other disorders such bipolar disorder, post-traumatic stress disorder, schizophrenia, eating disorders, disruptive behaviours, social disorders and neuro-developmental disorders.

A person can get mental health disorders at any age group belonging to any class, lifestyle or community. The effects of mental health issues are severe. Along with the emotional effects, it can cause gain or loss in weight, fatigue, gastritis, acid reflux, palpitation, hypersomnia or insomnia. Depression and anxiety may be fatal. They may cause suicidal instincts, hypertension, diabetes and several heart diseases too.

Major mental health disorders develop over time. They start with minute effect which can be mostly treated in this phase. If left untreated or undetected, with time, it grows and can cause many problems. Therefore, detection of mental disorders in initial phase when they start developing, plays a vital role in curing them. With huge population, it is either infeasible to perform screening tests and diagnostic tests or people resist to take part in the tests die to social stigma.

With the advent of technology, the smart devices and the internet, data can be collected easily. In recent years, machine learning has helped a lot in areas like bio-informatics, disease predictions etc., which may cost a lot without machine learning. With the large amount of data available for analysing mental health, we can use this data and create a model which can predict mental disorder with good accuracy. We used an open source dataset collected from tech world employees [2].

3. Methodology

3.1 Architecture Diagram

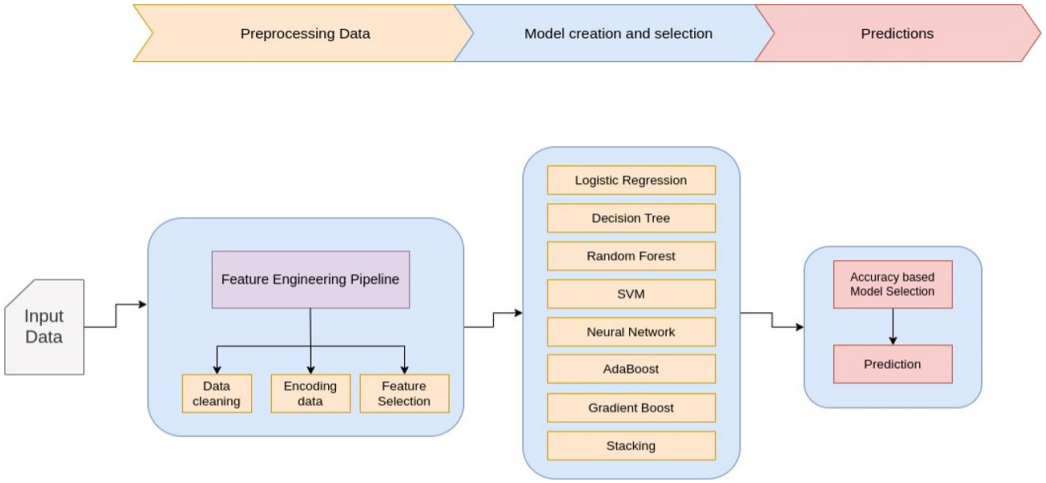

Figure 1: Architecture Diagram

A high level methodology is shown in figure. The input dataset is preprocessed. For preprocessing, data cleaning, data encoding and feature selection are used. After the data is preprocessed, it is used to train various models which we have used, including logistic regression, decision trees, random forests, stacking, SVM, Neural network and more. Based on the accuracy, a model is selected and predictions are made using the selected model.

3.2 Preprocessing

The dataset contained a lot of unstructured information in 'Gender' column and a lot of outliers in 'Age' column. As part of **Data Cleaning**, we dropped the irrelevant features like timestamp, comments, obs_consequence etc. Gender feature had more than 40 different values. So, we classified those into three main categories. Age had many outliers so we scaled the outliers to point to median of the dataset.

We also categorically mapped the unstructured information into defined features. Therefore, we implemented textbfLabel Encoding and textbfOne Hot Encoding.

We used PCA for feature selection and feature scaling is doen suing Min-max Scaler and Standard Scaler. The Figure 2 represent the unstructured original data of Gender feature and the result of structuring it.

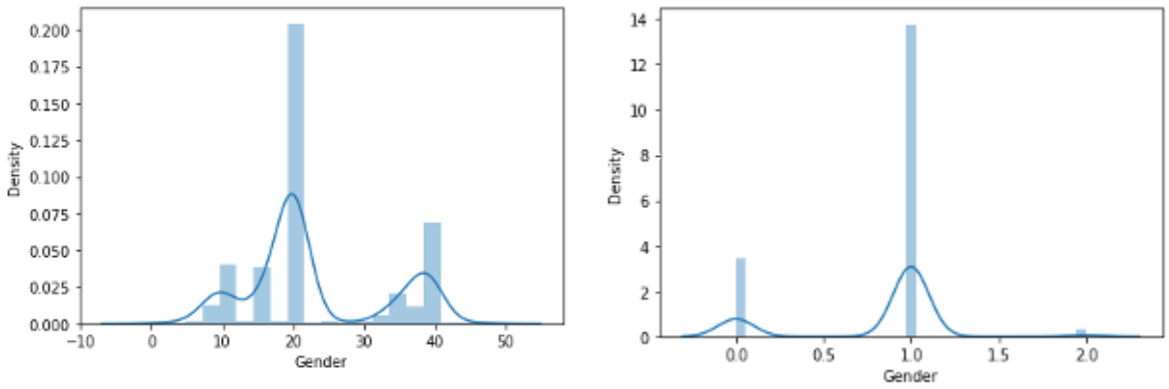

Figure 2: Gender vs Density

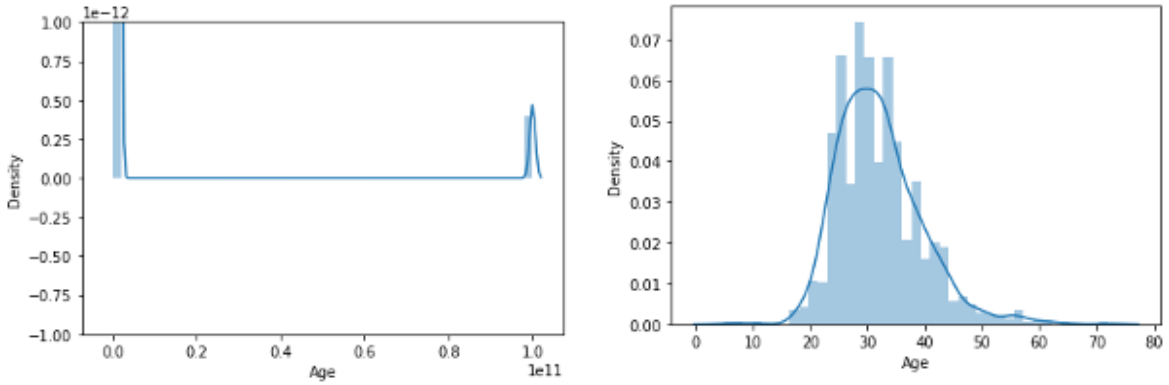

Figure 3: Age vs Density

The Figure 3 demonstrate the outliers present in Age data and mapping of those data into median.

3.3 Logistic Regression

The problem that we have addressed in this project is a simple binary classification problem. So, logistic regression is the first candidate for a model to be used in this case. It gives us a probabilistic output which is between 0 and 1 instead of concrete 0 and 1. We have used "Hold One out" approach with the data division between test set and train set in the ratio 25:75. The penalty used is "L1 regularization" and solver used is "liblinear". At first, we implemented Logistic Regression with Label encoded data and the accuracy obtained was around 76%. Also, the ROC curve is shown in figure 4. Then we used one-hot encoded dataset and found that our model performed surprisingly well as compared to its previous performance. The results are shown in table 6. Also, the ROC curve is shown in figure 5.

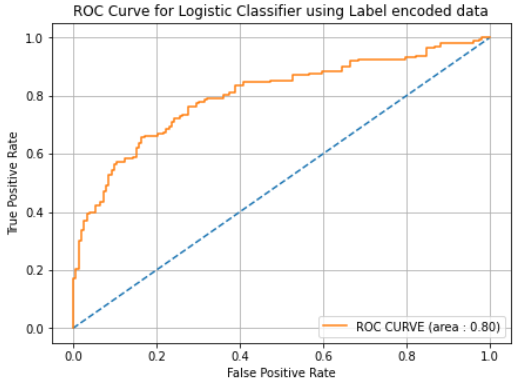

Figure 4: ROC Curve with Label Encoded Data

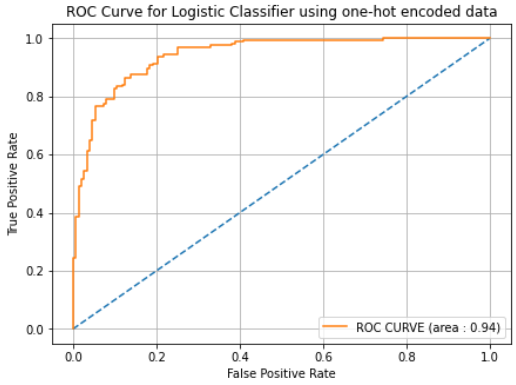

Figure 5: ROC Curve with One Hot Encoded Data

3.4 Decision Tree

Another natural candidate for a classification problem is Decision trees which give the capability where entire dataset and features can be used. If we know which features are more important, then DTs acquire high accuracy with low computation power. We have used decision trees with the data division of 35:65 for train and test dataset. The hyper-parameters we used were max_height 3 and random_state 10. Setting maximum height a small constant value avoids over-fitting as otherwise, decision tree would go down until there are only pure nodes. The feature at the root node came out to be benefits_don't know and it is splitting the data almost equally which is [408, 410]. The decision tree visualization is shown in figure 6 which shows the feature of split, entropy and split size at each level.

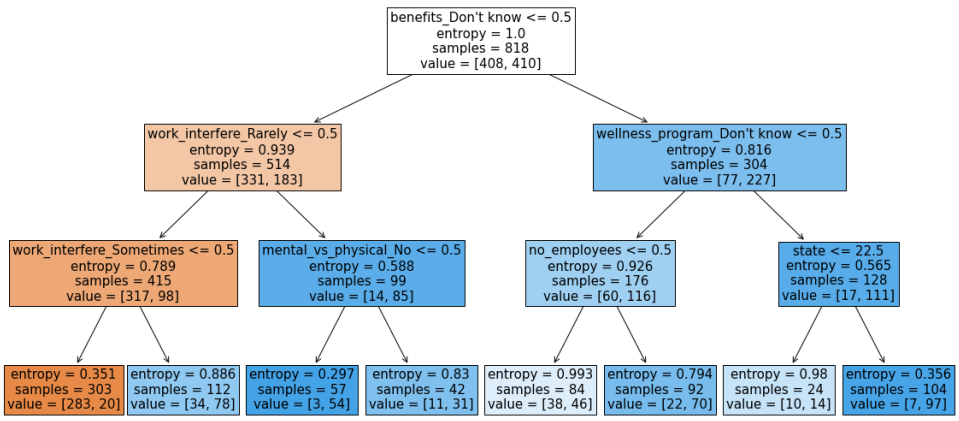

Figure 6: Decision Tree Model

3.5 Neural Network

Neural Network is widely used and popular algorithm of deep learning. We implemented a model with 2 hidden layers with 20 and 10 neurons while the input dimensions are 53. The layers are fully connected and all the parameters are initialised randomly.

However, the results were inadequate and excessively poor. Highest accuracy achieved was around 62.9%. The underperformance was due to the poor structuring of dataset and inadequate quantity. Deep neural networks require humongous amount of data which made it fail miserably. Synthetic data generation is also tried, but it didn't improve the results. Therefore, we moved ahead to implement some other models.

3.6 Ensemble Learning

In an effort to increase the accuracy we were attaining using logistic and decision trees, we turned to ensemble learning. We experimented with random forest, AdaBoost, gradient boost, and stacking algorithms.

3.6.1 Random Forest

We began by using a random forest with 300 estimators, leaving all other parameters constant. The training dataset was found to be overfitting. Test accuracy was observed to be 81.4%, F1 score was 81%, whereas training accuracy was 100%. The confusion matrix is show in table 1.

149	30
40	159

Table 1: Confusion matrix for Random Forest

3.6.2 AdaBoost

We reasoned that perhaps using adaboost might help us overcome the issue of overfitting and obtain the required accuracy. Decision stumps were used as weak learners. Test dataset accuracy was 82%, F1 score was around 82%, and training set accuracy was 84%. AdaBoost did not overfit the training data, but this model did not show any substantial improvement either. Find the confusion matrix below in table 2.

140	39
28	171

Table 2: Confusion matrix for AdaBoost

3.6.3 GradientBoost

Following our unsuccessful trials with previous ensemble learning techniques, we chose to try gradient boost. Gradient boost has recently gained a lot of popularity for its use in getting good predictions while maintaining prediction speed. Decision trees are typically used by them as their learners. In contrast to adaboost, they are not limited to using decision stumps and can typically use decision trees with a depth of 4 to 32. In several ways, it is extremely similar to Adaboost, such as the fact that both of them build models based on the earlier model. With gradient boost, all learners are scaled with the same value as opposed to Adaboost, which assigns various weak learners varying weights based on their error rates.

Gradient Boost is based on the idea that the previous model will unquestionably improve when the best next model is chosen. This is comparable to making a better prediction and moving towards the right path. Given that we are advancing toward the ideal model (decreasing the prediction error rate) while making small steps, this is the main justification for the name gradient Boost. After applying gradient boost, we got similar kind of accuracy as that of Adaboost. Even though we observed that training accuracy significantly improved to 91% but the test accuracy did not show any significant change staying at 83.3% and having F1 score as 83.2%. The confusion matrix is shown below in table 3.

147	32
31	168

Table 3: Confusion matrix for GradientBoost

3.6.4 Stacking

Stacking is used as in our regression task as it can combine the strengths of multiple well performing machine learning algorithms and outperform any single one of them in terms of accuracy of prediction. In our case we have used Kneighbours , GaussianNB , SVM as underlying learning algorithms from where we have fed the output as an input to logistic regression which acts here as the final learning model. With the use of stacking we achieved a highest accuracy of approximately 87% and having an F1 score of approximately 88% . The confusion matrix is shown below.

147	36
14	181

Table 4: Confusion matrix for GradientBoost

3.7 SVM

Support Vector Machines is a classifier to distinguish linearly seperable classes of data points. It optimizes the margin to draw a decision boundary between different classes. We used, radial basis function kernel to enhance the seperability based on gaussian distances between data points. The train-test split used was 8:2 and kernel co-efficient was set to one upon number of features. The accuracy using label encoded data was around 76%, however enhancing the model with one hot encoded data led us to the best ever results. The highest accuracy we achieved came as 89% with confusion matrix as table 5 and all other metrics are given in table 6.

921	23
3	134

Table 5: Confusion matrix for SVM

4. Results

Classifier	Accuracy(%)	Precision	Recall	F1 Score
Logistic Regression	86	0.832	0.914	0.871
Decision Trees	84	0.86	0.85	0.84
AdaBoost	82	0.81	0.85	0.82
Random Forest	81	0.84	0.79	0.81
GradientBoost	83	0.84	0.84	0.83
Stacking	86.7	0.83	0.92	0.87
SVM	89	0.85	0.97	0.91

Table 6: Result metric of all the models

The results of all the models ares shown in table 6 with Accuracy on test set, precision, recall and f1 scores. The models are executed on multiple iterations to achieve the best possible results. The most sound results were found from Support Vector Machines with an accuracy of 89% whereas other models aren't far behind.

Similar works have been executed on the same dataset by Vaishnavi et al., [14] where they achieved highest accuracy of 81.75% using Stacking. Compared to that, our model identified different classes in a more accurate manner. Also, gold medal winning model in Kaggle competition on this dataset achieved an accuracy of 85.63% using XGBoost [1]. So, we can ensure the improvement in our implementations. The better performance is primarily caused due to sound data cleaning and one hot encoding.

5. Conclusion

There are lot of machine learning models available. Each model fits in a different scenario depending on the dataset. Comparing the prediction accuracy of some the major suitable models is important as we will know the model which suits the situation best. We compared the accuracy of four models which are logistic regression, random forest, decision trees, SVM, Stacking, Gradient Boost and adaboost and found out that SVM is giving better predictions. Prediction of diseases, including mental health disorders, can help in curing it and can avoid other diseases and risks which come with it. The size of dataset we used is small and so it is upscaled. Also, we can see how important data cleaning and preprocessing can be while dealing with small data. Our model can be expanded with more real world data and the model accuracy can be improved further. The documentation of the python libraries we used are cited here , [5], [6] [7], [8], [9], [10], [11], [12]. Also, the Code for the project and the presentations are linked here respectively [3], [4].

Bibliography

[1] Kaggle competition. https://www.kaggle.com/code/aditimulye/mental-health-at-workplace/notebook.

[2] Dataset source. https://osmihelp.org/research.

[3] Presentation. https://iitbacin-my.sharepoint.com/:p:/g/personal/22m0802_iitb_ac_in/Ef35Pi-psQ5CpGfknZhyTHABq3u9aoXMhBBl19kt4S9cyA?e=ykRdEc.

[4] Code. https://github.com/tathaiitb/Mental-Health-Prediction.

[5] Documentation. https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing.

[6] Documentation. https://scikit-learn.org/stable/supervised_learning.html#supervised-learning.

[7] Documentation. https://www.tensorflow.org/api_docs.

[8] Documentation. https://keras.io/api/.

[9] Documentation. https://numpy.org/doc/stable/user/basics.html.

[10] Documentation. https://pandas.pydata.org/docs/user_guide/index.html.

[11] Documentation. https://matplotlib.org/stable/index.html.

[12] Documentation. https://seaborn.pydata.org/tutorial.html.

[13] Suresh Bada Math and Ravindra Srinivasaraju. Indian psychiatric epidemiological studies: Learning from the past. *Indian journal of psychiatry*, 52(Suppl1):S95, 2010.

[14] Konda Vaishnavi, U Nikhitha Kamath, B Ashwath Rao, and NV Subba Reddy. Predicting mental health illness using machine learning algorithms. In *Journal of Physics: Conference Series*, volume 2161, page 012021. IOP Publishing, 2022.