ORIE 4741: Final Report
Mental Health Analysis in Tech: Final Report

Donghao Huong(dh734)
Lihe Cao(lc977)
Sukriti Sudhakar(ss2742)

December 2020

I. Introduction(Problem Definition)

In current society, people suffering from abnormal mental conditions tend to conceal the fact of their abnormal condition and refuse to seek professional help. In the long term, patients, companies, and even psychotherapists are negatively affected by this atmosphere. Mental illness, when not treated can cause severe emotional, behavioral, and physical disabilities.[1] To better tackle this problem, we think it is important to find out the factor that stops people from asking for help from the psychotherapist. This project aims to find out what kind of environment stops people from seeking mental health treatment. In this project, we try to build a model which will predict whether a person will seek mental treatment in the given environment or not.

II. About the Dataset

IIA. Data description

The dataset we used comprises of 1524 observations and 78 attributes. In our dataset, seventh column is the responses which represent whether the people go to ask for professional help or not. We use that column as the column to be predicted. Among all the features, 16 of them are already numerical, which we directly used in the model, 18 features contained a high ratio(85%) of missing value, 4 features contain geographic information which has little to do with our model selection. We get rid of these 22 features. Finally, we got the following features where 12 features contain boolean data, 9 features contain nominal data, 11 features contain ordinal data, and 9 features contain text data.

IIB. Handling missing values

As mentioned above, our data consists of numerical data, boolean data, nominal data, ordinal data, and text data. We realized that by simply getting rid of the rows or columns containing missing values, we will lose a lot of information about the dataset. Therefore, we tried to fill up the missing values. We used different methods to fill up different values from different data types. For the continuous numerical data, we used the mean value of the column to fill up the missing value. For the discontinuous numerical data, we used prediction generated by probability proportional to the each kind of value and the size of the column to fill up the missing value. We used similar methods to fill up the missing values for boolean and ordinal data. For the nominal data, we treated the missing entry as a type of value.

IIC. Data Encoding

We used real encoding method for the Boolean value. For Yes/No answers, we set Yes to 1 and No to 0. We then used one-hot encoding for the nominal data and Boolean encoding to encode the ordinal data. For the text features, we used word embedding to transform them into a matrix which can be received as input, we used SentenceTransformer package to do this task. After the text embedding, we get a matrix with 6912 columns. Since our original dataset has only 1524 observations, directly using the matrix would have cause overfitting. To avoid overfitting, we used PCA to reduce the dimension. We needed to find a dimension large enough to cover the principal component of the text data. We computed the explained variance ratio and found 50 as the smallest dimension when the variance ratio reached 90%(Figure1), which means the resulting text matrix can cover approximately 90% information of the original text data. After we have encoded all the data, we concatenated the data matrices into one to be our final feature matrix.

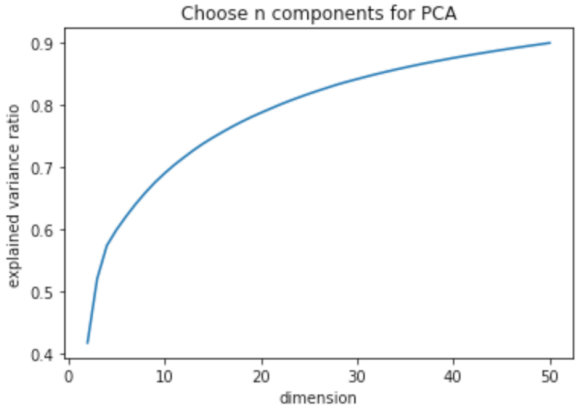

Figure 1: *Variance ratio against dimension*

IID. Data Normalization

After filling the missing values and encoding the data, we found that among the 177 features, all feature have value 0 or 1 except for the age and the 50 columns from the word embedding. When we did cross-validation, it took quite long time for some of the models to run and the logistic model could not converge using some solvers. Therefore, we decided to do the normalization for the age data and text data that we got from word embedding. After normalization, all the models converged under the maximum iteration of 1000 iterations and the training time decreased largely. In addition, it also raised the accuracy for some classifiers.

III. Model

In this project, we choose the following classification models to do the prediction, the logistic classifier, soft-margin SVM, K Nearest Neighbor, Decision Tree, Random forest, and XGBoost. As a thumb of rule, we shuffle the data and then separate it into training and test set. Then we use different parameters to train the models on the training set and compare the results on the test set using the chosen parameters.

IIIA. CROSS VALIDATION

First, we do cross validation for each model to choose the suitable hyper-parameters. The following are the hyper-parameters we tuned for each model and the range of the hyper-parameters when doing cross validation.

model	hyper-parameter		
logistic regression	loss function	C value	solver
soft SVM	kernel type	C value	gamma value
KNN	n nearst neighbors	distance function	
decision tree	criterion to split	max depth of tree	
random forest	n estimators	max features	max depth
xgboost	max_depth	min_child_weight	

Figure 2: *Hyperparameters tuned for each model*

After we did cross validation for the first round, we have found the best magnitude for each hyper-parameter. Then we do a second round of cross validation. In this round, we narrow the range of hyper-parameters around the best magnitude we get in the first round. For example, the highest C value for rbf kernel is C =10, when the cross-validation score is 0.861. Then we run cross validation again for C = [5, 6, 7, . . . , 14, 15], and find the best C value to fit the test data. For example, the figure below is graph of accuracy of SVM with rbf kernel and C in range (5,15) in the second round of cross-validation.

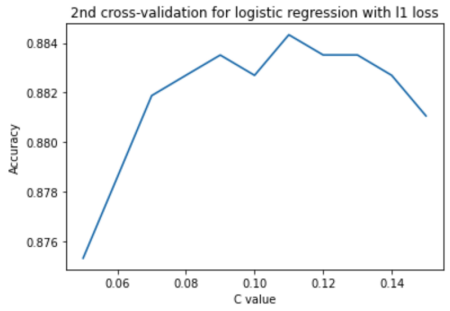

Figure 3: *Accuracy versus the C-value graph for SVM*

IIIB. Model Fitting

Now we use the best hyperparameters for each model chosen from the cross-validation to fit the test data. The following table shows the result of all the models, we only pick the hyperparameters with the highest accuracy for each model.

Model	Accuracy	Hyper-parameter		
Logistic	86.2%	loss function = l1	C value =0.12	Solver = liblinear
Logistic	84.6%	loss function = l2	C value = 0.01	Solver = lbfgs
KNN	84.3%	number of neighbours = 59	Euclidean Distance	
SVM	84.9%	kernal = rbf	C value = 7	
Decision Tree	80.1%	criterion = entropy	max_depth = 13	
Random Forest	86%	n_estimators=1800	max_depth=420	max_features='auto'
Xgboost	87.4%	max_depth=9	min_child_weight = 5	

Figure 4: *Accuracy and value of the tuned hyperparameter*

III.IIIB.A. Random Forest Classifier

Random forest is a machine learning algorithm used for classification. A random forest comprises multiple individual decision trees and each tree outputs a prediction and the prediction with the majority votes becomes the output of the algorithm., or in other words it outputs the class that is the mode of the different trees or the mean of different decision trees.[3]
We ran the random forest classification on our data. The accuracy of the model before tuning the hyperparameters was 84.26% .
We then used Randomized Search Cross validation to select the best value of hyperparameters. We chose 3 hyperparameters, n_estimators, max_features, and max_depth. High numbers of hyperparameters increase the running time and hence, we chose to consider these three hyperparameters.[2] After tuning the hyperparameters, there was an increase in accuracy and the new accuracy was 86%.
The values of tuned hyperparameters that were used for 2nd cross validation.

Model	Accuracy	Hyper-parameter		
Random Forest	86%	n_estimators=1800	max_depth=420	max_features='auto'

Figure 5: *The values of tuned hyperparameters that were used for 2nd cross validation*

The confusion matrix for random forest classification model is as follows:

Model	Accuracy	Precision weighted _avg	Recall weighted _avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
Random Forest	86%	86%	86%	86%	87%	83%	90%	80%

Figure 6: *Confusion matrix for random forest after tuning hyperparameters*

III.IIIB.B. Logistic Regression

Logistic regression is a machine learning algorithm used for classification problems.Logistic Regression uses the log odds ratio rather than probabilities and an iterative maximum likelihood method rather than a least squares to fit the final model.[4]
We train the model using the training examples, different regularizer terms, and different solvers and get the accuracy rate using both the training and test set(Figure5)(After using cross validation).

The first round cross validation results(Figure 5), accuracy and valu of tuned hyperparameters(Figure 6), and accuracy(Figure 7) all have been listed below.

solver \ C value	0.01	0.1	1	10
saga	0.863	0.866	0.868	0.868
newton-cg	0.875	0.862	0.861	0.838
lbfgs	0.875	0.863	0.860	0.841

Figure 7: *Results for logistic regression from cross validation*

Model	Accuracy	Hype-parameter		
Logistic	86.2%	loss function = l1	C value =0.12	Solver = liblinear
Logistic	84.6%	loss function = l2	C value = 0.01	Solver = lbfgs

Figure 8: *The values of tuned hyperparameters that were used for 2nd cross validation*

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
Logistic_l1	86.2%	87%	86%	86%	87%	85%	91%	80%
Logistic _l2	84.6%	85%	85%	85%	86%	82%	78%	89%

Figure 9: *Accuracy for logistic regression model*

III.IIIB.C. Support Vector Machine

Support Vector Machine (or the SVM) is again a machine learning algorithm that can be used for classification problem. The best part about using SVM is that there is lesser risk of overfitting.[5]. The results are shown below:

kernel \ C value	0.01	0.1	1	10	100
linear	0.862	0.852	0.850	0.847	0.842
polynomial	0.610	0.610	0.839	0.861	0.858
rbf	0.610	0.829	0.847	0.861	0.839
sigmoid	0.610	0.837	0.865	0.849	0.844

Figure 10: *Results for soft SVM from cross validation*

Accuracy and value of tuned hyperparametrs:

Model	Accuracy	Hype-parameter	
SVM	84.9%	kernal = rbf	C value = 7

Figure 11: *The values of tuned hyperparameters that were used for 2nd cross validation*

Confusion Matrix for the SVM model is shown below:

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
SVM	84.9%	86%	85%	85%	89%	82%	93%	75%

Figure 12: *Accuracy for SVM model*

III.IIIB.D. K-NearestNeighbor Classification

K-nearest neighbors(KNN) is a supervised machine learning algorithm that can be used for both regression and classification problems. This algorithm works on assumption that similar things are near each other.[7]. This algorithm outputs the. class that is the most common class amongst k of its nearest neighbors.[8]. The results obtained through this model have been shown below:
Accuracy and value of hyperparameters tuned:

Model	Accuracy	Hyper-parameter	
KNN	84.3%	number of neighbours = 59	Euclidean Distance

Figure 13: *The values of tuned hyperparameters that were used for 2nd cross validation*

Accuracy of KNN model:

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
KNN	84.3%	84%	84%	85%	87%	79%	87%	79%

Figure 14: *Accuracy for KNN model*

III.IIIB.E. Decision Tree

A decision tree is a supervised learning method where the model continuously split based on a certain parameter. This algorithm is based on the divide and conquer approach, which keeps on splitting data into smaller subsets and goes on until the algorithm decides if the data is sufficiently homogeneous. It is a very fast method for classifying unknown records and it doesn't not include unknown features.[9] The results obtained are shown below:
Accuracy and value of hyperparameters tuned:

Model	Accuracy	Hyper-parameter	
Decision Tree	80.1%	criterion = entropy	max_depth = 13

Figure 15: *The values of tuned hyperparameters that were used for 2nd cross validation*

Accuracy of the model:

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
Decision Tree	80.1%	80%	79%	79%	80%	77%	85%	70%

Figure 16: *Accuracy for Decision tree model*

III.IIIB.F. XGBoost Model

XGBoost is a decision tree based model that uses gradient boosting framework. XGBoost often gives the best accuracy and can be executed in least amount of time.[10] The results obtained from this model are shown below.
Accuracy and value of hyperparameters tuned:

Model	Accuracy	Hyper-parameter	
Xgboost	87.4%	max_depth=9	min_child_weight = 5

Figure 17: *The values of tuned hyperparameters that were used for 2nd cross validation*

Accuracy of the model:

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
Xgboost	87.4%	87%	87%	87%	85%	89%	93%	78%

Figure 18: *Accuracy for XGBoost Model*

IV. Result Analysis

Model	Accuracy	Precision weighted_avg	Recall weighted_avg	F_1 score	"1" Recall	"0" Recall	"1" Precision	"0" Precision
Logistic_l1	86.2%	87%	86%	86%	87%	85%	91%	80%
Logistic _l2	84.6%	85%	85%	85%	86%	82%	78%	89%
KNN	84.3%	84%	84%	85%	87%	79%	87%	79%
SVM	84.9%	86%	85%	85%	89%	82%	93%	75%
Decision Tree	80.1%	80%	79%	79%	80%	77%	85%	70%
Random Forest	86%	86%	86%	86%	87%	83%	90%	80%
Xgboost	87.4%	87%	87%	87%	85%	89%	93%	78%

Figure 19: *Final result table*

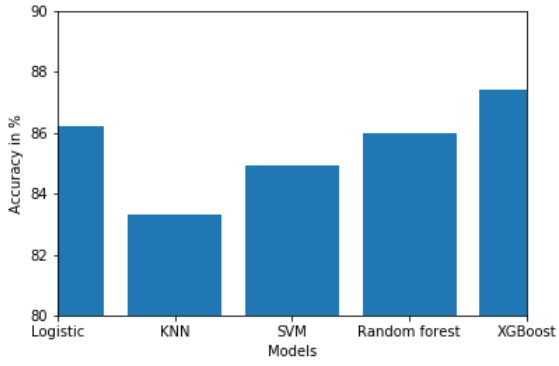

Figure 20: *Accuracy bar chart*

In the above table, class "1" represents people who are willing to seek treatment when suffering from mental problems and class "0" otherwise. From the table we can see that among all the models we use, the Xgboost model gives us the best overall accuracy. However, the high accuracy

does not always mean the model is good. Our goal of the project is to build a model which can help us find the patients with psychological problems but do not seek professional medical help. Therefore, we should focus on increasing the accuracy in predicting people who suffer from mental problems but tend not to seek medical help. Considering these, we should not only take into consider the accuracy, but also calculate the precision and recall for the two classes. Here, $prediction = \frac{trueexample}{goldenset}$. If we predict 100 positive class and 80 among them are indeed positive, precision =80%. $recall = \frac{trueexample}{retrievedset}$, which shows how precisely the data is being predicted. If we have 100 positive example in the test set and we predict 40 of them, recall =40%. In our project, we want to have high precision for class "1" and high recall for class "0". The former ensures that we don't mistakenly predict patients who tend not to seek mental treatment as they do. The latter ensures that our models will cover most patients who do not seek treatment.
We can observe from the table above that XGBoost has both the highest "0" recall and "1" precision which are 89% and 93% respectively. It also the highest overall accuracy of 87.4%. Therefore, XGBoost is the best model to achieve our goal in this project.

v. Conclusion

The purpose of this report was to build a model to predict whether the people in technology having mental problems will seek help or not. The data we chose consisted of 1524 observations and 78 features. We implemented data encoding, filling missing data, and data normalization at the data prepossessing stage. After all the feature engineering, we implemented six different models on the training set and noted their accuracies on the testing set. To avoid the problem of overfitting and in order to find the optimal parameters for each model, we conducted two rounds of cross validation. Then we use the use the model with hyperparameter and predicted the responses for the test data. Considering the models' performances on overall accuracy, precision, and recall on each class, we believe that XGBoost is the optimal model for the problem.

A weapon of Math Destruction(WMD)is a predictive model whose predictions can have a negative impact consequences and whose outcome is not easily measurable.[11]. We believe that our model can be a weapon of math destruction. The wrong prediction of our model may have a huge negative impact on people. For example, if a person who is actually suffering from mental health disorders might be predicted to not seek help and that will be very harmful. At the same time, if a person is not expected to see a doctor but is predicted to seek help might get some unfair treatment by the employer.

References

[1] "Mental Illness." Mayo Clinic, Mayo Foundation for Medical Education and Research, 8 June 2019, www.mayoclinic.org/diseases-conditions/mental-illness/symptoms-causes/syc-20374968.

[2] Huneycutt, Jake. "Implementing a Random Forest Classification Model in Python." Medium.com, medium.com/@hjhuney/implementing-a-random-forest-classification-model-in-python-583891c99652.

[3] "Random Forest." Wikipedia, Wikimedia Foundation, 9 Dec. 2020, en.wikipedia.org/wiki/Random_forest.

[4] "Logistic Regression Analysis." Logistic Regression Analysis - an Overview | ScienceDirect Topics, www.sciencedirect.com/topics/medicine-and-dentistry/logistic-regression-analysis.

[5] "204.6.8 SVM : Advantages Disadvantages and Applications." Statinfer.com, statinfer.com/204-6-8-svm-advantages-disadvantages-applications/.

[6] Brownlee, J., 2020. Classification Accuracy Is Not Enough: More Performance Measures You Can Use. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/> [Accessed 13 December 2020].

[7] Harrison, Onel. "Machine Learning Basics with the K-Nearest Neighbors Algorithm." Medium, Towards Data Science, 14 July 2019, towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.

[8] "K-Nearest Neighbors Algorithm." Wikipedia, Wikimedia Foundation, 29 Nov. 2020, en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.

[9] Chakure, Afroz. "Decision Tree Classification." Medium, The Startup, 6 Nov. 2020, medium.com/swlh/decision-tree-classification-de64fc4d5aac.

[10] Morde, Vishal. "XGBoost Algorithm: Long May She Reign!" Medium, Towards Data Science, 8 Apr. 2019, towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d.

[11] "204.6.8 SVM : Advantages Disadvantages and Applications." Statinfer.com, statinfer.com/204-6-8-svm-advantages-disadvantages-applications/.