Summer 2022

Modeling document classification to automate mental health diagnosis

William M. Tadlock

Modeling Document Classification to Automate Mental Health Diagnosis

---

A Thesis

Presented To

Eastern Washington University

Cheney, Washington

In Partial Fulfillment of the Requirements

for the Degree

Master of Science in Computer Science

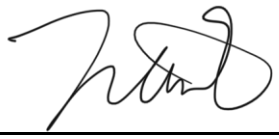
By

William M. Tadlock

Summer 2022

THESIS OF WILLIAM TADLOCK APPROVED BY

DAN LI, GRADUATE STUDY COMMITTEE DATE 8/17/22



TONY TIAN, GRADUATE STUDY COMMITTEE DATE 8/17/22

Abstract

The objective of this study is to determine if diagnosis documents can be used with document classification to automatically diagnose mental health conditions. Document classification allows text documents to be analyzed and organized into their appropriate classes based on the features and words presented in the text. One application of this is within the medical field to automatically classify different patient diagnosis based on medical or patient notes. This research applied mental health diagnosis documents to automatically diagnose a group of patients with a mental health condition based on text-based survey data. This classification was approached through several feature engineering and machine learning models to determine the optimal methods for diagnosis classification. A model was created that successfully classified diagnosis documents to their appropriate mental health condition, but due to limitation in the patient dataset, no model successfully classified patient diagnoses.

Table of contents

1 Introduction

With the number of text documents online, automatic text classification has become a useful tool with many applications in various fields of study, specifically the medical field. The main goal of text classification is to categorize electronic documents based on the information within the document. Text classification is commonly used for built in spam checkers or email filters to determine the type of email that was received [4], and can also be used to categorize news articles as they are published through a news classifier. The type of text classification that this thesis is concerned with is for medical document classification, where medical documents are received as inputs and are classified to an appropriate diagnosis.

Within the medical field patients are diagnosed based on criteria for each disease or disorder. For mental disorders, the Diagnostics and Statistics Manual for Mental Disorders (DSM-5) is used [1]. Since this manual has been used successfully within the field to diagnosis patients, a natural step is to determine if this same manual can be used to help automatically find a diagnosis for a patient based on text inputs. The goal of this thesis is to create a mental health diagnosis system trained from the documents within the DSM-5.

To classify the diagnosis documents within the DSM-5, this project will use a system built with feature engineering combined with predictive algorithms. Five feature engineering methods were used, including Term Frequency Inverse Document Frequency (TF-IDF), Recursive Feature Elimination (RFE), Chi Squared Score, F-score, and

a custom built portable feature set. Four machine learning pattern classification algorithms were implemented including Support Vector Machines (SVM), Naïve Bayes, K Nearest Neighbor (KNN), and finally Recurrent Neural Networks (RNN).

The structure of the rest of this paper is as follows: Section 2 explores the related works within the field of text classification and medical document classification. Section 3 briefly covers the methods that are used for feature engineering and classification. The data preprocessing and collection are in Section 4 along with the implementation and results from the program. The Section 4 is split into three parts, each representing a stage in the development and experimentation process. These subsections include fitting the diagnosis documents, predicting the diagnoses of patient data using feature-based approaches, and predicting diagnoses using deep learning. Finally, the analysis and concluding statements are provided in Section Five.

2 Related Works

While little work exists relating to the automatic diagnosis of patients that this thesis focuses on, there is a lot of work relating to components that are used within this thesis: text processing through feature selection and extraction, classification methods through use of predictive models, and patient diagnosis. A large part of classification methods relies on determining the best way to preprocess the text for the context of a given classification problem. Similarly, there does exist work that diagnoses patients based on a classification model [10].

2.1 Feature Engineering

Before natural language can be processed by a classification algorithm, the text needs to be preprocessed so the computer is able to read and recognize the data. An important piece of preprocessing for classification algorithms is feature engineering which consists of feature selection and feature extraction. The goal is to find features within the text documents (feature selection) and find the most relevant features within each document (feature extraction). Some existing feature selection methods include TF-IDF, Chi-Square, F-score, and Gini index [18], and some feature extraction methods commonly used with text classification include Principal Component Analysis, Latent Semantic Indexing [18], Particle Swarm Optimization[11], and Recursive Feature Elimination [3]. Another useful feature engineering method involves portable feature sets [15] with the use of WordNet [17] [8]. All of these feature engineering methods work by themselves or combined with others to produce the appropriate input for the remainder of the text classification.

Foram P. Shah et al. [18] explains the general text classification process and the roles that feature select and extraction play. Each document is preprocessed in order to simplify the text for more efficient use in later steps of the classification process. Next, feature selection occurs by "selecting the subset from the original feature set on the basis of importance of features" [18]. These features are initially derived from words or phrases within the initial document and are a key aspect of classification, as most predictive models rely on the set of features provided to determine the final

classification. Different types of feature selection include filter methods, which evaluate each feature based on a statistical metric and select the top k features, and wrapper methods, which use a learning model to evaluate each feature and how it interacts with the data set before selecting a subset of the features. After the features are selected, feature extraction occurs to reduce, transform, or add to the set of features based on the structure of the set and the needs of the classification model. Both feature selection and extraction can serve the same purpose of defining the proper feature set to be used for classification. Due to this similarity, the term feature engineering will be used when describing the process that can be completed by selection of extraction.

Other works have found unique ways to optimize specific feature extraction methods. Abdollahi et al.[11] developed a method of document classification by focusing on providing more meaningful features for classification, deriving features that will have a greater impact on the classification results. The model they used has two stages. The first stage uses an ontology of terms from the related document domain to narrow down the text features, and the second stage uses a Particle Swarm Optimization (PSO) algorithm to further optimize the features.

Robert Dzisevič et al. [7] compared three methods of feature extraction with a neural network classification model. The findings of this paper show that TF-IDF is overall a very useful and efficient method for feature selection, regardless the size of the dataset. The documents for this thesis comprise of a relatively small dataset, inferencing Dzisevič's use of TF-IDF with Latent Semantic Analysis (LSA) and being cautious of overfitting.

Anasari et al. [3] compared different feature selection methods created by combining filter methods of feature selection using the metric of Chi squared and F-score with wrapper methods of Particle Swarm Optimization (PSO) and Recursive feature elimination (RFE). They used three supervised learning classification methods: Naïve Bayes, support vector machine, and logistic regression. The classification is based on sentiment analysis with the goal of two classes, positive or negative. Anasari et al. focused on optimizing the feature extraction element of classification based on the comparing and analyzing the classification results of each machine learning algorithm. Though these methods of feature selection are useful in combination, the datasets being used do not relate to the distinct test and train set that this thesis aims to use.

Abeed Sarker et al. [15] showed an approach of feature engineering to handle text classification using three distinct corpora, similar to the two distinct corpora used for testing and training within this thesis. Portable feature extraction was used to extract a feature set that can be used in multiple domains. In order for cross training to be supported among data sets, Sarker et al worked under the assumption that the data sets were compatible, which may not be the case within the two data sets in this project. In order to use multiple data sets, it is important to create portable feature sets to attempt an increase in compatibility. The creation of a portable dataset includes adding preprocessing steps that increase the usability of text before transforming it into a feature set. One of the steps is negating terms with a tool called NegEx, which could be very useful with phrases that begin with “not” without using “not” as a feature. Another tool used was MetaMap that identified semantic types and context IDs within

the text based a medical language database. The concern is finding a database for useful language within the domain of the being used which is not always possible. One of the most relevant work is the step including set expansion through WordNet, which is an online network that stores relationships between words. After these steps are completed, the feature sets are vectorized through TF-IDF and implemented with a classification algorithm, but were also tested in combination with other datasets for improved accuracy. Out of all ideas of this portable feature set, WordNet is the only element that could reliably be used within this thesis.

As described above, WordNet is an online collection of words and is composed of SynSets containing terms that are synonyms. Other notable relations include antonyms, hypernyms and hyponyms. The latter two are relating terms to a more general or specific term respectively. This can be useful when dealing with the specific terminology within a domain or the broad terminology within another. Among other feature engineering techniques Sam Scott et al. tested the use of WordNet with a text classification model [17]. The idea was to aid classification by “a feature engineering method that mapped words with low information gain to common hypernyms that yield a high information gain.” Scott et al. used WordNet to look up hypernyms and synonyms of each noun and verb, and then used the resulting Synsets to create feature vectors of the new representation. Unfortunately, the experiment showed that the WordNet set performed worse than the others, but WordNet could still be applied in combination with other feature extraction methods to create a more portable feature set which would make up for the lower accuracy.

2.2 Classification Models

After finding a method of feature engineering through either feature selection or extraction, it is common to compare those with several different classification algorithms to see which best fits the model and the problem [13][3]. It can also be useful to use a single classification method and optimize it to receive a certain outcome with the test data [10]. The most effective classification algorithms typically utilize a supervised machine learning technique to train the computer what to expect when given a certain input. The algorithm will train with a set of data that has already been classified and runs on multiple iterations to find the best fit for the data. Once the training data has been trained to a certain point within a chosen metric such as accuracy, classified test data that is independent of the training data is run through the algorithm and metrics are evaluated and analyzed. There are cases when the model is trained closely to a specific dataset, which causes the model to overfit and not provide accurate results when tested. The goal is always to find the best combination of features and predictive models based on the specific dataset and type of data being processed.

It is useful when working with document classification to compare the results of feature extraction with the classification algorithms. Bekir Partak et al. [13] focuses on analyzing the results of document classification of two datasets of medical documents using a combination of two feature selection methods and two text classification methods. The documents are classified by different diagnoses, and they removed any multi point documents from the datasets in order to focus their models on only

classifying one diagnosis per document. The methods used by Partak et al. were Gini Index and Distinguishing Feature Selection for feature selection and Bayesian Networks and Decision Tree for classification, with all methods using the same text preprocessing of removing stop words and stemming. Stop words are often simple and common words in the language that typically have little impact on the overall meaning of a statement, such as “the”, “and”, or “every”. Due to the low value that these words present within classification, they are often removed with the intent of improving the accuracy of a model. Stemming is cutting off the beginning or end of a word based on common prefixes or suffixes. They found that the best combination for their datasets was selecting features with Distinguishing Feature Selection and classifying with Bayesian Networks.

There has been more work of classification done in the medical field, both using medical documents [13] and using medical records. Jamaluddin et al. [10] used existing medical records with the goal of creating an application to classify patients with a diagnosis. They used support vector machines (SVM) with three different kernel functions on the text data from electronic medical records after finding features using TF-IDF. They also tested what impact the removal of stop words would have on the testing results and discovered that the results without stop words provided more accuracy. This is one of the few classification models used directly on patient data to find a diagnosis.

2.3 Medical Document Classification

The goal of this thesis is to recommend a diagnosis to an individual patient from text data provided, similar to the works shown in [10]. There have also been works related to recommender systems using natural language documents outside the medical field [11], which take a different approach to the problem than the classifier models discussed. Another method used keywords to match similar diagnoses [4], and another used deep learning to screen for a specific diagnosis [6]. Each of these focuses on a different aspect of classification and recommendation systems that could be used in finding a diagnosis for a patient.

Cataldo Musto et al. [11] focused on creating a recommender system that uses a multiple criteria item-based filtering framework to recommend products for users to purchase. The user is compared to other similar user profiles to create a list of recommended products for the user. This could be applied to recommending a diagnosis based on other patients who have been diagnosed. The model presented by Musto et al. thrived from having multiple items reviewed in each user profile and had the ability to dynamically find aspects from text in an unsupervised environment. They took a list of aspects that each user represented in a review and compared it to all other users to find a similarity score between users. Similarly, the aspects and products were compared with a similarity score. Each user was recommended a product based on the scores of other users and products. Though this model was created with product recommendation in mind, it has many aspects that could be applied to a diagnoses

recommender system. If applied to a medical diagnosis system, a dataset would be needed that has multiple documents for each patient that could each be diagnosed separately. Unfortunately, no such dataset was able to be discovered for this thesis, but it remains an interesting concept if the resources are provided for a proper implementation.

Ghassan Azar et al. [4] created an architecture to match patient symptoms with a diagnosis using machine learning methods, including a genetic algorithm and k-means. Azar et al. used the Diagnostics and Statistics Manual for Mental Disorders to create a model for patient diagnosis. The later edition of this manual is the main training document being used within this thesis. They ran each patient through multiple generations, applying each keyword to a genetic algorithm and storing each result within a database. The diagnosis was found by applying a k-means algorithm to the stored results based on a text input. While this work is similar in many ways to this thesis, from motivation to a similar diagnosis source used, the inputs and training based on a keyword input differ from the document-based classification being explored in this thesis.

Another way of finding a diagnosis for a patient is to focus on a single diagnosis and population. This allows for a more detailed application of a diagnosis system since there is only one focus compared to many. Yong Chen et al. [6] uses sentiment analysis and a deep learning model of a collection of messages sent to extract an emotion to help diagnose depression in perinatal women. Unlike many other sources listed above, they took the focus away from a feature-based classification model in favor of exploring

sentiment analysis. This was in part due to the emotions displayed within the collection of messages used as the input data. They measure with the Edinburgh Postnatal Depression Scale to base the diagnosis, which is one of the unique benefits of using a single diagnosis within one model. This thesis attempts something similar on a larger scale by using diagnosis documents to generate several diagnoses, but it may not be as effective as building a model focused on the criteria of a single diagnosis. This article provides insight into using text classification in a setting of mental health in the medical field and introduces the concept of deep learning within a classification model.

Beyond the feature-based classification models that have been explored, deep learning models offer a new dimension for diagnosis classification that could overcome the shortcomings of focusing on TF-IDF within the context of the medical field. Due to the high dimensionality of medical texts, Li Qing et al. developed a neural network-based classification method which incorporates deep learning at both the word and sentence level [14]. They found the most difficult aspect of classifying medical text was accounting for the professional vocabulary and irregular grammar used, creating an issue of data sparsity which made classification difficult. Using multiple layers of various neural networks including convolutional neural networks (CNN) and recurrent neural networks (RNN), a hierarchical model was constructed to effectively combat the sparsity and dimensionality of the medical texts with a result more effective than baseline neural networks.

A similar approach was made by Julia Ive et al. by developing another hierarchical neural network model for classification [9]. The goal of this model was to

classify social media text based on mental health relations, which is a similar goal of this thesis. However, instead of using a diagnostics manual as the training data they used the social media dataset as the training and testing set through a deep learning model. A document level RNN was developed, with a hierarchical attention mechanism to allow detection of both words and sentences. The model builds the document by aggregating important words into sentences, and then the sentences into documents, and can distinguish specific words or sentences for classification decisions. The model successfully assignmened mental health classifications within their dataset and performed better than a CNN solution to the same problem. Several of the methods explored by the works of others will be used within this thesis and are described in more detail in the next section.

3 Methods

This thesis will utilize many of the methods introduced above to develop a classification model for patient diagnosis. The implementation of these methods is generally conducted in two separate stages, feature engineering and building models. First the data will enter the feature engineering stage to properly format and optimize it for classification. Then the newly engineered features will be used within a predictive model, both as a training set to fit the model and a test set to predict the diagnosis of each document. This section will list and briefly explain each of the methods of feature engineering and predictive modeling used within the experimentation of this thesis.

3.1 Feature Engineering

An important piece of text classification is the feature set used by the classifier algorithms, and the features used can greatly impact the accuracy of the model. That is where feature engineering comes in to find the best feature set for the documents given. A feature set is a list of relevant aspects found within a given set of documents, with each feature being represented by an n-gram where n is the number of words contained within the feature. This thesis will focus on finding and using 1- and 2-gram features. Feature engineering can be broken into feature extraction and feature selection, but for the purpose of this paper the general term "feature engineering" will be used since there can be some overlap between the functions of selection and extraction. According to Cataldo Musto et al., feature engineering "is based on (a) vector space model, where a text is viewed as a dot in a N-dimensional space. Each dimension of the dot represents one feature of the text in digital form" [11]. The words from the text are evaluated and assigned a weight based on the document and evaluation metric. Those words are then formed into a digital vector which becomes the feature vector of the text that is used in most predictive algorithms for classification.

Each feature engineering method used in this project works with a Term Frequency Inverse Document Frequency (TF-IDF) vector as the initial vectorization of the text. TF-IDF is one of the most widely used methods for finding feature vectors [4]. It works by weighing the frequency of a given term with the inverse document frequency. This is a simple and effective approach to feature vectorization, but it does have issues

with high dimensionality when working with large datasets [11]. For this reason, TF-IDF can be paired with another feature engineering method to help reduce the feature vector. Since the datasets used within this thesis are relatively small, some cases will be evaluated with TF-IDF alone.

When selecting features from a vector, filter methods and wrapper methods can be used. A filter method finds a specific metric to assign a score to each of the features, and then selects the k best features with the highest scores to include within the feature set. They do not involve any learning algorithm in removing irrelevant features, and thus are a high speed and computationally inexpensive option when compared to wrapper methods [18]. A limitation of filter methods is that the feature dependencies are not accounted for in the scoring [4]. Wrapper methods look at feature subsets to select features based on a learning algorithm. While providing optimal feature sets, wrapper methods are slow when applied to large feature spaces. This project uses one wrapper method of recursive feature elimination (RFE) and two filter methods of Chi Squares and F-score.

The wrapper method being used is RFE with the classifier being a support vector classifier. RFE works by scoring each group of features through the learning algorithm and recursively eliminating the weaker features through an iterative backward-forward elimination method [18]. The iterations continue until an appropriate feature set is found, but the drawback is that the optimal feature set size must be known in advance. If the optimal feature set is found, RFE can have higher accuracy than other feature

engineering approaches; however, the backward-forward nature does make it slower compared to other wrapper methods.

The design of filter methods requires a scoring metric to be applied to all features in the feature set independently, and then selecting the k best of those features. For text classifications, the metric being used should relate each feature to the document to give the best results when using the feature set in the classifier. Chi Squared achieves this by measuring the dependence of a given feature through measuring deviance of the observed and expected counts [4]. This removes the features that are most likely to be independent of the class and thus irrelevant for classification purposes. The F-score statistic measures the degree of linear dependency and the discriminating power of each feature. “Chi-square ranks the features based on their presence or the absence in a class, whereas F-score also considers feature strength for ranking the features” [18]. Though widely used, F-score has a major weakness due to only scoring each feature independently without regard to the larger feature set. This causes the overall accuracy of F-score to be lower with classification. Beyond the use of filters and wrappers, other methods can be used in other circumstances to find the correct set of features for a dataset.

Portable datasets provide a unique opportunity within feature engineering that can use the feature set of one dataset to help define the feature set of another. The goal of this method is to create a link between two distinct datasets to allow classification of both within the same model, resulting in a set of features modified for use within the model called a portable feature set [15]. This thesis will focus on

WordNet for developing a portable feature set, which is a thesaurus for the English language developed as a data processing resource [8]. Synsets, the building blocks of WordNet, contain a group of synonyms or words of similar significance. WordNet is built in a tree structure connecting every word through synsets and represents the levels of the tree through hypernyms or hyponyms. A hypernym relates one word to a more general word (water/beverage). A hyponym relates one word to a more specific concept and is the opposite of hypernym. The root of each tree within WordNet is the eventual hypernym of word within that tree. These concepts can be used to manipulate features within a set to better fit the need of classification.

Within the tree nature of WordNet, any two words will have a similarity score, and this score can be used to find connections between concepts to build a similarity feature set. The similarity score is determined by the longest path it takes to get from one word to another through synsets, hypernyms, and hyponyms. The higher the score, the more similar the two words are. When applying WordNet similarity to feature engineering, the typical use is for only words within a single synset [8], but an expanded search can be useful to develop more thorough feature sets. However, it is important to set a threshold for similarity scores as to have a consistent level of minimum similarity and to avoid over generalization. This method can be used in combination with any of the other feature engineering techniques discussed within this section to provide a more accurate result.

3.2 Predictive models

Once a feature set has been found for the documents within a dataset, the classification algorithm fits them for each class. The most common method for classification is through a machine learning predictive model. This project uses three commonly used models: Support Vector Machines (SVM), Naïve Bayes, and K Nearest Neighbor (KNN). Each of the three models receive a set of training features which are used to train the model and the model is then tested for accuracy with a set of testing features. The output is a classification of each document into a class based on the feature set of the document. Recurrent neural networks (RNN) are another method used and do not follow the norm of receiving feature for training. Each of these models utilize a different aspect of machine learning to predict and classify the data used.

SVM is a supervised learning algorithm utilized for classification and regression using the mathematical concept of hyperplanes for classification [18]. It works by separating the classes using hyperplanes, and the tuples that fall on the edges of the hyperplanes are called support vectors. An advantage of SVM is the ability to effectively work with high dimensional data such as natural language, which makes it suitable for this project.

Naïve Bayes classification applies a naïve solution to the bayes theorem of conditional probability by assuming the independence of features. That is, “one feature should be independent from another feature under known probability and class conditional probability” [4]. Naïve Bayes is widely used for easy implementation and fast

computation. However, the naïve assumption of independence of class can result in loss of accuracy.

Another learning algorithm used is KNN, which ranks the neighbors of a document and then compares the classes to the k most similar neighbors. The distance metric used to measure similarity for this project is Minkowski distance, a generalization of Euclidean and Manhattan distance. The similarity score is used as the weight on the classes of the k nearest of the neighbors [4]. The implementation of KNN is simple yet robust and is good with noisy training data; however, a challenge is finding the optimal value of k to use since the wrong value could lead to under or over fitting the model.

Introducing the concept of deep learning adds an extra dimension for classification that the previous predictive models lacked. Deep learning applies neural networks within multiple layers to achieve a higher level of learning and prediction with a structure modeled off the biological human brain [19]. An application of deep learning for classification is Recurrent Neural Network (RNN), utilizing neural connections in a directed cycle, using internal memory to sequence through the input and retain previous computations to calculate the output of each element. Figure 1 shows the depiction of a sequential RNN, with the image on the left depicting a network with cycles, and the right depicting a network with time steps, where each time step corresponds to one word [19]. A bidirectional RNN is a modification based on each element depending on the output from both the previous and the next element in a sequence. This can be accomplished by using long short term memory (LSTM) layers to increase the learning of long term dependencies. The ability to maintain the relationship

between elements makes it useful for tasks relating to natural language, as understanding how words relate to each other can help understand the overall document.

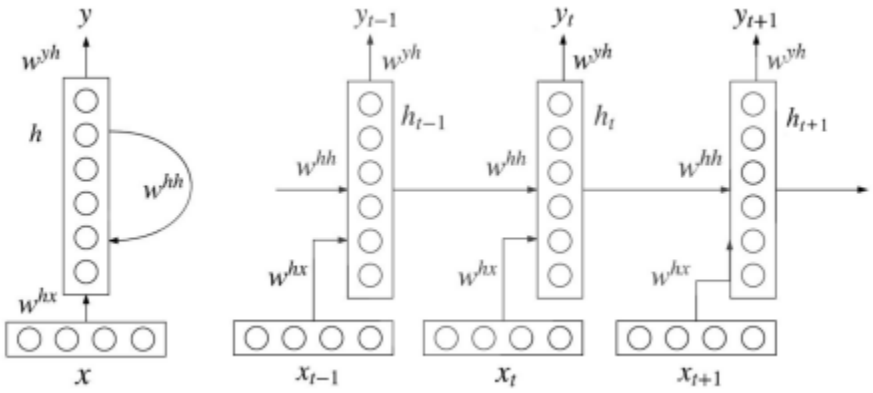


Figure 1: Model of RNN Structure [19]

An application that utilizes the connections between words within an RNN model is sentiment analysis. There are three main levels of sentiment analysis: document, sentence, and aspect. Each level is evaluated for either a positive or negative opinion, and is the sentiment is often targeted at a specific topic or entity [19]. Though this thesis does not utilize sentiment analysis directly since the classification labels of the model are already defined, the close connection RNN has with sentiment analysis makes understanding it an important part of utilizing the model.

4 Experimentation & Results

4.1 Fitting diagnosis documents

In order to classify an individual with a diagnosis, each of these diagnoses must be properly defined and recognized by the predictive models. This led to the first step of

fitting and classifying the diagnostic criteria. The goal of this step is to determine if the diagnoses are distinct enough to be classified individually based on the diagnosis documents being used. Once a classification method is successful with training and testing only among diagnosis documents, the next step is testing the fitted model on a patient dataset.

First, the diagnosis documents are collected and preprocessed as the dataset for use within the predictive models. Next, the training and testing is implemented into three different models with four different methods of feature engineering. Finally, results are presented, analyzed, and interpreted in the context of patient diagnosis.

4.1.1 Data Collection and Preprocessing

This project uses documents for medical diagnoses as described in the DSM-5 [6] and references the provided diagnosis categories as the classes for classification. The first step in creating the dataset is compiling each diagnosis into a separate text file. Each file is then placed in a folder with the name of its diagnosis category, and only categories with more than 10 diagnosis documents are kept for the dataset to have more accurate classification. Using R Studio, all the documents are combined into a CSV file with file contents, file name, and folder name to give the raw dataset used for classification.

Using Visual Studios Code and the Natural Language Toolkit (NLTK) library, the dataset prepares for classification by preprocessing and labeling the text. When preprocessing text, it is important to have each word represented by the root of the

word. Two ways of accomplishing this are stemming and lemmatization. Stemming is the simpler action; however, this can cause unnecessary cuts or create roots that do not match with the original word. That is where lemmatization comes in, considering the different forms of words by referencing a detailed dictionary to bring a word back to its lemma. This project adheres to the lemmatization approach using the libraries in NLTK due to the more precise nature compared to stemming.

The next step in preprocessing is stop word and punctuation removal. Again, NLTK is used, providing a list of stop words that the document text was compared against. This is also completed with punctuation and new line symbols, so the result of the document contents is simplified text that only contains words; those words are only the more important words and thus more relevant for classification.

Before the text is ready to run through feature engineering and classification, each of the documents need a label for the classification model to utilize. Anxiety Disorders are labeled 0, Depressive Disorders labeled 1, Neurocognitive Disorders labeled 2, Neurodevelopmental Disorders labeled 3, Obsessive Compulsive Disorders labeled 4, Paraphilic Disorders labeled 5, Personality Disorders labeled 6, Schizophrenia Spectrum Disorders labeled 7, Sexual Dysfunctions labeled 8, and Sleep Wake Disorders labeled 9. With the labels added, the dataset is pruned of unnecessary columns and is ready to be classified.

4.1.2 Implementation

Throughout the rest of the implementation, the libraries from Sci Kit Learn are used. The first step is the test train split. The dataset is initially split into one training and testing pair, but in order to more accurately represent the full dataset, stratified 10-fold cross validation is then applied. This creates 10 test train pairs split equally distributed based on the classes. Each of these is then ran through every part of the model, and the mean of all pairs is used in evaluation for a given model and feature method.

For feature engineering, the dataset is transformed and fit into a TF-IDF vector. The parameters for TF-IDF gives 300 features with a maximum document frequency of 10 features per document. From this feature set, three separate feature sets are created with a reduced 150 features. The final four feature sets include the following: TF-IDF, TF-IDF with RFE using SVC, TF-IDF with Chi Squared, and TF-IDF with F-score.

Finally, these feature sets are then fitted and tested with each of the classification models. Before the models are used on the dataset, a random search and grid search for optimal hyperparameters are conducted. Of these hyperparameters, the value of k for KNN is selected as 1. Each fold of the dataset is used to train the SVM, Naïve Bayes, and KNN model separately, and is then tested for accuracy.

4.1.3 Results

The results shown above illustrate that KNN is the best model for classifying this dataset, having the four highest test set accuracy among the four feature sets used with KNN. Overall, Naïve Bayes was the least accurate classification method. F-score is shown

to be the least effective feature engineering method, being the lowest scoring accuracy for each of the three models. While RFE is the highest scoring overall while using KNN, the other classifiers show the standard TF-IDF as the superior feature set. This could be due to the TF-IDF having twice as many features as the other three since features were removed during feature extraction for the others. This means there were more features to train from, which can often times lead to a higher accuracy.

Figure 2 Accuracy of Classification Models

	Model	Training Set Accuracy	Test Set Accuracy
5	rfe_KNN	0.850450	0.829435
6	chi_KNN	0.839863	0.816633
4	KNN	0.850450	0.816532
7	f_KNN	0.832806	0.810383
0	SVM	0.850450	0.810081
1	rfe_SVM	0.841277	0.797681
8	NB	0.832812	0.791532
2	chi_SVM	0.788720	0.705141
9	rfe_NB	0.779896	0.702419
3	f_SVM	0.743218	0.689617
10	chi_NB	0.746397	0.673891
11	f_NB	0.699485	0.638911

An interesting result can be seen on the confusion matrix of each run through the model in figure 3. For every run, there is a significant number of false positives shown under the class for Sleep Wake Disorders. This shows an uneven distribution within the dataset which displays a need for better data distribution and distribution analysis in the future.

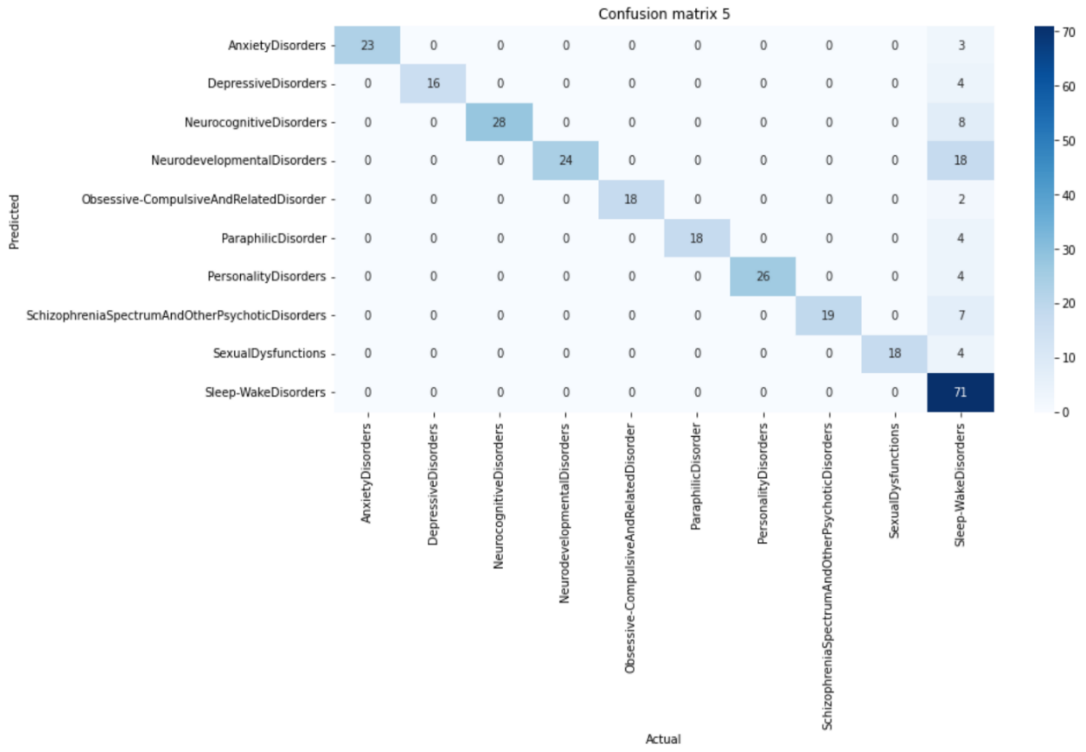


Figure 3: Confusion Matrix of DSM-5 Classification

Since this model was tested and trained on documents from the same diagnosis dataset, these results are not representative of patient diagnosis. Since the language within the DSM-5 is very specific to each classification used, the hope is that patient text input will be recognized in each unique class. With the results shown, most of the features within each class are distinct enough to be classified as the appropriate diagnoses. When looking at patient input, the features being used for classification are a key component of a proper diagnosis with these methods.

Another takeaway of this experiment is noting which setup for the classification model is most effective for this domain. Moving forward it is known that recursive feature elimination with k nearest neighbor works best, so when testing a patient

dataset that configuration should be used first. This is due to the training set being identical, so if the model was properly fit for DSM-5 testing and training, keeping the model fit in a similar way should provide better results.

4.2 Predicting Diagnoses of patient survey data using TF-IDF

In the previous subsection, a model was created to evaluate diagnosis documents within the DSM-5 and classify them into the appropriate diagnosis. The next piece of this thesis requires using that classification model on a set of patient documents to diagnose each patient. After searching for an appropriate patient dataset to use, the results from the Beck Depression Inventory (BDI) are settled upon. However, the BDI only monitors depression, which is only a single diagnosis from our previous model. This simplifies the classification to binary, either the patient has depression or does not have depression.

Since there are now two separate datasets being used, BDI and DSM-5, the model relies heavily on their feature sets and respective TF-IDF vectors; a method is then developed to combine the features. This is to allow for the model to classify both datasets even if they are initially dissimilar. This creation of a portable dataset is what the bulk of this section contains. What follows is an explanation of the data collection and preprocessing for patient documents, then the implementation of creating a portable dataset and integrating it into the existing classification model. To conclude is an explanation and interpretation of the results of this experiment.

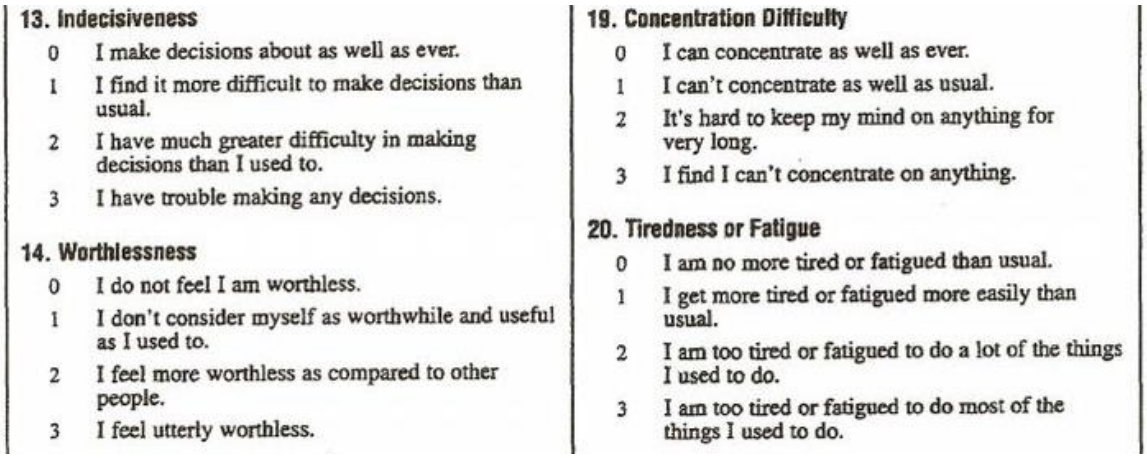
4.2.1 Data Collection and Preprocessing

The patient dataset is the test dataset within the model, and each document in the dataset represents one patient and their diagnosis. To achieve this purpose, a dataset needed to be found that had preassigned diagnoses for natural language inputs relating to each patient. An ideal dataset uses the natural language from medical records, but most medical notes that have patient diagnoses are not available for public use due to privacy concerns.

The alternative is to look at anonymous health surveys that list diagnoses for the survey users/patients and use the survey questions as the natural language input. The National Health Information Survey (NHIS) conducted by the Center for Disease Control was the dataset initially selected to match these criteria. The NHIS is a multiple-choice survey and contains questions that identifies several diagnoses; however, a problem arises when trying to collect the survey questions and answers from the survey manual. All survey answers are provided in a CSV file, but in order to obtain the natural language, each question needs to be individually extracted from the PDF of the survey manual, which is not formatted in a way to accomplish this easily. Even if all questions are extracted as natural language documents, the number and content of multiple-choice answers to each of the questions is inconsistent, with some questions giving five answers representing the best answer to a question and others giving seven answers that representing the patient's feeling of a question. Overall, the complications of extracting the natural language leads to abandoning the NHIS in search of a dataset that is easier to use.

Trying to find a survey or other dataset with multiple labeled diagnoses proved to be a challenge, so the search was narrowed to a dataset that had many single diagnoses labeled. With the new search criteria, a diagnostic tool for depression was found called the Beck Depression Inventory (BDI). The BDI was developed in 1979 by A.T. Beck and is still in use within the medical field today to determine whether a patient could be diagnosed with depression. It consists of 21 sets of four statements numbered 0 through 3, and the patient selects one of the four statements from each set that they feel represents them the most. Once complete, the scores of each set are added up, and based on the sum it is determined whether the patient has depression and the severity of the depression. For the sake of this project and the binary nature of the classification, severity is not taken into consideration only if the patient has the diagnosis of depression.

The dataset being used is from 242 McGill University students filling out the BDI, collected by Prof. David Zuroff. The response dataset was imported though R Studio and saved in a CSV file. The PDF of the BDI questionnaire was converted into a CSV file in R



13. Indecisiveness

0 I make decisions about as well as ever.

1 I find it more difficult to make decisions than usual.

2 I have much greater difficulty in making decisions than I used to.

3 I have trouble making any decisions.

14. Worthlessness

0 I do not feel I am worthless.

1 I don't consider myself as worthwhile and useful as I used to.

2 I feel more worthless as compared to other people.

3 I feel utterly worthless.

19. Concentration Difficulty

0 I can concentrate as well as ever.

1 I can't concentrate as well as usual.

2 It's hard to keep my mind on anything for very long.

3 I find I can't concentrate on anything.

20. Tiredness or Fatigue

0 I am no more tired or fatigued than usual.

1 I get more tired or fatigued more easily than usual.

2 I am too tired or fatigued to do a lot of the things I used to do.

3 I am too tired or fatigued to do most of the things I used to do.

Figure 4: Sample of BDI Questions [5]

Studio with the score and statement in each row, ordered by set. Next, a patient document was generated for each response. The CSV files were loaded into Python and one statement for each set was appended to the document based on the patient score for that set. This resulted in a document for each patient containing 21 sentences. Then, the scores were added up for each document, and a label of “Depression” or “Not Depression” was assigned to the document according the evaluation of the BDI criteria. In this dataset there were seven labeled “Depression” and 235 labeled “Not Depression.”

After the dataset was created, text preprocessing occurs in order to make the documents easier to classify. During this preprocessing, a common step of stop word removal creates a problem with this specific dataset. Within the BDI statements, negating words such as “not” are frequently used to have great impact on the understanding of that statement. However, “not” is a common stop word to be removed, so if stop words are removed like a typical preprocessing, the meaning of the statements would be lost and could potentially cause misclassification. This is because the classification algorithm being used is a feature-based model, and each feature with “not” attached to it should be negated to get the correct use of that feature within a given document. For this reason, stop words are not removed and the DSM-5 dataset is preprocessed again without stop word removal for the sake of consistency.

Another modification to the DSM-5 dataset regards the labels. Within section 4.1 there were 10 different diagnosis labels, but now with the change to binary classification of depression this dataset is not compatible for classification. Any

diagnosis label that is not “Depression” is changed to “Not Depression” to account for any of the patients that are not classified as “Depression” from the BDI. With the preprocessed BDI and updated DSM-5 dataset, both datasets are now ready to run through a classification algorithm.

4.2.2 Creating a Portable Feature Set (Implementation)

To begin the classification process for the BDI dataset, a tf-idf vector is fitted and generated in the same way as the DSM-5 tf-idf vector; these two datasets are disjointed. Not only are there only 166 features generated for BDI compared to the 300 generated for DSM-5, none of the generated features of BDI match up with those from the DSM-5 feature set. This leads to two approaches, combining the datasets and generating a tf-idf vector and feature set based on the combined set for testing and training, or find a way to connect the two disjointed sets through word similarity.

The simpler of the two approaches is to combine the sets. This involves taking the two preprocessed data frames within Python and concatenating them together, thus creating one larger data frame with both BDI and DSM documents. The resulting dataset is then transformed and fit into a TF-IDF vector of 300 features. The feature set for the combined dataset is not identical to the DSM-5 feature set, so the BDI dataset within the combined set does impact the transforming and fitting of the TF-IDF vector. To follow, the combined set is split into a 5-fold stratified cross validation set for training and testing. The set is then run through the top three classification models determined within section 4.1 with the DSM-5 classification. This ultimately does not have the BDI as

the test set, but rather incorporates into both the test and training sets in a hope to achieve a more accurate result.

The second approach to integrate the BDI into the classification model involves keeping the BDI as the entire test set and the DSM-5 as the entire training set. Since the model relies on having a consistent feature set for the TF-IDF vectors, the goal is to transform the BDI vectors into the feature space of the DSM. This incorporates a concept called portable datasets [15] and involves creating a method to make one dataset usable within the domain of another separate dataset. To accomplish this, the 166 features of the BDI is then compared for similarity with the 300 features of the DSM-5. Once all features of both sets are compared, the highest similarity score is used in the final TF-IDF for the BDI.

The method being used to create a portable dataset from the BDI is implementing WordNet Synonym sets known as SynSets and comparing each word within the SynSet to get a similarity score or SimScore. Each feature within the BDI and the DSM-5 have a corresponding SynSet generated for them, but since the feature sets are generated for 1-gram and 2-gram features, features cannot initially have a SynSet. For 2-gram features, the feature is split into two separate words that each receive their own SynSet, and the SimScore results in an average of the score of each of the two words individually. The pseudocode for the SimScore algorithm is shown in figure 5.

Each feature is iterated through on both BDI and DSM-5, split in the case where it is a 2-gram feature, and the features from each set are compared to one another and

Algorithm 1 SimScore Algorithm

```
for each feature b in BDI do
    if f is n-gram then split b into n features
    for each feature d in DSM do
        if f is n-gram then split d into n features
        for each word x in SynSet of b do
            for each word y in SynSet of d do
                s = WordNet Similarity(b,d)
        if f is n-gram then s = average of s from each split
        Append (d,b,s) to SimList
```

Figure 5 Pseudocode for SimScore Algorithm

generate a SimScore for that pair. To maximize similarity potential, every synonym generated in a word's SynSet is individually compared to every synonym in the SynSet for the other word in the comparison pair. The maximum SimScore from all SynSet comparisons is then used as the SimScore for that pair. As mentioned above, for 2-gram features the feature comparison pair used the average SimScore from the comparison pairs of the two words that compose the 2-gram feature. Once the features have been compared, the feature comparison pair with the highest SimScore for each BDI feature is appended to a list. In the case of a SimScore tie, multiple feature pairs are added. Now that the each BDI feature has a list of the most similar DSM-5 feature(s), the list must be narrowed in order to only get meaningful pairs.

After analyzing the similarity score statistic amongst the similarity score list created by the Figure 5 algorithm, a threshold is found and applied to the list to create a final list of pairs to extract a new TF-IDF vector for the BDI set. The SimScore threshold is determined by finding that under .3, specifically at .25, is where most SimScores seem to land. Having a large number of pairs saturates the dataset with pairs that do not carry

enough meaning and have less value to the classification. Above .3 is where more distinct values for SimScores started to appear, and by increasing the threshold, there are too few pairs to create a full TF-IDF vector. The value of .3 is chosen as a balance spot between too few and too many pairs, and the resulting list consists of 8,000 feature comparison pairs out of the initially generated 24,000 pairs left to create an updated BDI TF-IDF vector.

Since the DSM-5 has the largest and fullest TF-IDF vector and is being used for the training set, that vector remains unchanged and contains the feature list being used by the new BDI TF-IDF vector. The initial run of the SimList algorithm attempted to create a new feature list based on the highest common features that resulted. However, this led to only 28 features being represented within the TF-IDF vector which was not enough to accurately represent the classification on either the testing or training data. Even with decreasing the similarity score threshold, the feature set is too small for the classification algorithm.

This led to the idea of mapping each feature within the DSM-5 feature set to a BDI feature and setting the TF-IDF value of that feature to the value of initial BDI TF-IDF vector that the BDI feature corresponded to. If no BDI feature corresponds to a DSM feature, the value on the vector for that feature is set to zero. Additionally, there are cases where a single DSM-5 feature is paired to multiple BDI features with the same SimScore. The goal in selecting the BDI feature for a DSM feature is to have an equal distribution of BDI features and not select them in a way that would reduce the number of BDI features chosen.

Several methods were attempted to select the optimal feature to pair with each DSM-5 feature, but this became difficult since some BDI features could potentially match equally with the same several DSM-5 features. When trying to filter features to only allow the minimum DSM-5 features for each BDI feature, there began to be less features selected due to limiting the matches. After trying to modify the selection algorithm to maintain the same features, it was determined that the best solution was to simply select the first BDI feature listed for each DSM-5 feature.

The TF-IDF vector is created for the BDI dataset from the structure of the DSM-5 TF-IDF vector, and then both vectors are used within the classification algorithm developed in section 4.1.B. The DSM-5 dataset is used as the training data by training and fitting with no changes to the TF-IDF vector. The difference with this classification model is that none of the feature engineering techniques, such as Chi-Squared filter or RFE, are implemented since the creation of a portable feature set for the BDI is the main method of feature engineering used and adding another could risk the efficiency. The test set is made up entirely of the 255 BDI entries represented in the updated TF-IDF vector of the BDI dataset. With a vector now matching the training set, the BDI set can be used within the classification model for prediction. The BDI set has been classified both as the test set and while integrated into both the training and the test set.

4.2.3 Results and Discussion

With two different approaches of classifying a distinct patient dataset with the DSM-5 dataset, the results were evaluated using the same predicative algorithms to obtain the results. Though the models used were the same, the results would need to

be interpreted differently since the test sets were different. The first set contained a mix of DSM-5 and BDI, but since the model already proved successful with the DSM-5 dataset in section 4.1, this experiment only focused on the results from the part of the test set that contained BDI entries.

The combined test set was evaluated by the fitted classification models for a predicted classified result. Unsurprisingly, the best model from 4.1, recursive feature elimination with K-nearest neighbors, was also the best with this test set, as the bulk of the predicted test documents were from the DSM-5. The accuracy and confusion matrix of this predicted model may seem promising from a glance. However, a deeper look at the specific documents being classified will show every single BDI document classified as "Not Depression," even in the cases where the document should be labeled "Depression." This result shows that the overwhelming features were coming from the DSM-5 and because of this not enough impact was given to the BDI documents during the training phase. To combat this problem, the second method of creating a portable dataset was used, transferring the importance on each feature onto the BDI feature set.

After training the model on the entire DSM-5 dataset, the updated BDI TF-IDF vector was used to predict the classification of the BDI Documents. As mentioned in section 4.2.B, there were multiple iterations of the TF-IDF vector to attempt optimal classification. For each attempt, the classification model was evaluated and produced the same results; so while the final TF-IDF vector was created to be the best in theory, it provided no functional improvement over the other attempts. Overall, the prediction of the portable dataset was always to classify every document as "Not Depressed."

Through several attempts at modifying the TF-IDF vector were used for the prediction, it became apparent that creating a new vector with similarity score might not be enough to relate the BDI set to the DSM-5 set.

With neither of the classification approaches of the BDI set giving any meaningful results, it is important to evaluate the reasons for the failure, the takeaways from this experiment, and the next steps to see improvements when diagnosing a patient from the BDI. When integrating the datasets, it was already mentioned that the disparity between the two datasets caused the model to only fit to the DSM-5 side of the training data, which resulted in not having any BDI documents diagnosed as depression in the test set. The portable dataset did not provide results for a similar reason. The model was fit only to the DSM-5 and there was not a strong enough similarity connection able to be made from the transformed BDI set.

When looking at why the BDI set did not work even after modifying the TF-IDF vector, the use of portable datasets for this problem should be evaluated. The idea of a portable dataset is to strengthen the relationship between two datasets in order to have the datasets work together for classification by modifying the TF-IDF vector used as the input for the classification. However, this is still looking at classification based on a set of features, and though connections can be drawn between the features of two sets, as was done between the DSM-5 and BDI, sometimes the features themselves are structured differently within the documents to create a different meaning.

That is to say that even if all features are matches, two different datasets may have documents where the features are used in different ways with different relationships that dictate the classification. With the BDI, the documents are built differently than the DSM-5, giving short sentences about feeling a certain way about a certain thing. Even if the DSM-5 describes those same feelings about things, the documents have longer, more detailed sentences which talk about the feelings in a different way. This uses different words which show up as features that are irrelevant to the same feeling in the BDI and throw off the classification. By focusing less on the features alone and more on the relationships between the features, the BDI may have a better chance of being classified by the DSM-5.

To properly diagnose patients, it is best to move away from classification and predictive models that are based on TF-IDF. The restriction of focusing on specific feature sets make it difficult to classify anything that may focus on how a person feels, especially since those same feelings can be described in many ways. Since the focus is on mental health diagnosis, feelings are a key aspect that show up in several forms, so the focus should be on the general idea or sentiment of a document rather than the raw features a document contains. The idea is to treat the problem in a similar way to how sentiment analysis classifications are handled.

The next section will cover an experiment using a recurrent neural network classification model to analyze the documents for classification.

4.3 Predicting Diagnoses of patient survey data using deep learning

The discoveries of the previous sections led to taking a new approach to the classification problem. In order to classify patient documents based on diagnosis documents, the next method attempted requires using recurrent neural networks (RNN) to evaluate each document based on the relationship between words. This method has been used in sentiment analysis in which a document is evaluated for the overall sentiment, usually either positive or negative. By treating this problem closer to a sentiment analysis problem, the hope is to achieve results that can properly classify the BDI dataset as "Depression." This section uses the same data collected and preprocessed within sections 4.2 and 4.1 and begins with an explanation of the motivations for this experiment with RNN for classifications. Then, the model being used will be described, followed by the presentation and discussion of the results.

4.3.1 Motivation

As described in the previous section, the use of TF-IDF in predictive algorithms may not be suitable for this classification problem. This is due to the difficulty of finding the exact same type and frequency of features used in the DSM-5 within shorter patient documents that need to be diagnosed. This sparked the idea of focusing less on the features contained in documents and more on the sentiment and relationship between words within the document. With this new focus, the hope is to give a deeper understanding of what each document represents, thus leading to a more accurate classification or diagnosis.

Sentiment analysis is a form of classification that determines the feeling that a user has based on the text in a document. This is typically accomplished by looking at how specific words interact with others, sometimes forming dependency relationships between words or concepts within a text [19]. A common form of sentiment analysis is aspect based, where aspects or features are used as the elements that the sentiment is directed towards. With the discovery of the problematic nature of relying heavily on features within these datasets, an aspect-based approach would most likely not yield appropriate results. A more general approach to sentiment analysis would be more beneficial, using the concept of word relationships to analyze the sentiment of a document, and then relating that to the document label to aid in classification.

The classification approach of RNN has been primarily utilized for sentiment analysis due to the how the use of Long Short-Term Memory (LSTM) utilizes the relationships between words. Though this can be useful to determine sentiment, it can also be used for general classification. Within the context of mental health patient diagnosis, it makes sense to use an approach that can factor in sentiment. When looking at the BDI documents, understanding each of them requires looking at how the words connect together within each sentence. As mentioned in section 4.2.A, stop words do not seem useful in classifying these documents, and the use of RNN could be a benefit in helping the computer interpret the sentences. The deep learning nature of RNN is another addition to helping interpret each sentence.

Deep learning is important to consider within this problem since it provides a built-in automation of feature extraction within the layers of the neural network during

classification. Even though the previous TF-IDF method with predictive algorithms also focused on feature extraction, the deep learning algorithm learns from the previous layers and iterations and can modify features during the classification process. This can provide a more useful resulting feature set after classification, as opposed to having a set list of features for the entire classification which would need to change only with separate runs of the algorithm.

The benefits of using RNN make it a useful alternative for this classification problem, overcoming the feature-based classification issues that arose with previous experiments. Adding the use of deep learning provides a more reliable system for feature extraction, and the parallels to sentiment analysis techniques allow for a well-rounded view of the documents. These improvements address the issues this patient classification problem has faced, and implementing the RNN provides hope to find the proper patient diagnosis from the BDI.

4.3.2 Implementation

This section will explain the process of the development of using RNN for classification with patient diagnosis, and when starting any classification model, the environment and libraries are a key feature to start with. The implementation of the RNN classification model is set up differently than the other predictive models introduced in 4.1.B, utilizing the Tensorflow library in Python 3.9 instead of SciKitLearn. Due to a change in library, the first piece to address is the datasets that were created in sections 4.1 and 4.2, which need to be made compatible with Tensorflow.

All datasets used with a predictive algorithm from Tensorflow must be in a Tensorflow dataset data type. The two ways of getting the BDI and DSM-5 datasets into that data type are either building the datasets from their initial files from scratch into a Tensorflow dataset, or converting the pandas dataframe to the Tensorflow dataset. The initial consideration was building them from scratch and importing the files, but the issue came from the BDI set. Since each document from BDI was created within Python directly into a pandas dataframe, the documents do not exist as files and thus cannot be easily imported to Tensorflow. This leaves the option of converting from pandas to Tensorflow. The unprocessed text and the labels are both created as separate tensor slices as type string and int respectively. These tensor slices are then combined into a tensorflow dataset. This is done for both the BDI and the DSM-5 datasets, resulting in two datasets, the testing and the training.

With the data properly imported, each dataset is batched and the training set is shuffled to prepare for use with the RNN. The vocabulary is then encoded with a size of 1000 and generated based on the training set. This vocabulary list contains all words that the RNN will recognize, and all others get coded as "[UNK]." The first classification that will be executed is the sequential model, built through Keras and from the vocabulary just created, using one layer of LSTM. The model is fit to the training set and initially set with the testing set as the validation set. A section of the training set was also tested as the validation set on another run of the model for higher accuracy. The model was set to run for 10 epochs during the fitting, and after the completion of the 10^{th} epoch, the test set containing the BDI was predicted through the model.

A new model was created with additional layers of LSTM, created with other parameters the same as the sequential model. Initially two layers of LSTM were tested, and then three. These additional layers use the bidirectional feature of RNNs, passing the result from one layer to another. Another modification between different runs was the number of Epochs, running with 2, 5, and 15 on top of the original 10. This was to account for potential over or under fitting of the model. No other tests were run on the RNN model, with a total of 24 different possible configurations attempted.

4.3.3 Results and Discussion

Despite the many configurations of the RNN model that were tested, each prediction on the BDI test set created the same resulting classification. Just like the experiments in section 4.2, all documents were classified as “Not Depressed” despite the use of LSTM and RNN. Each subsequent configuration had slight changes in the accuracy and loss during the fitting of the training set, but it was a trivial change considering the failure to properly diagnose “Depression” within the test set.

The issue previously encountered in section 4.2.B of the dissimilarity between the BDI and DSM-5 sets can be seen again during the fitting of the training data. When the BDI test data was used as the validation set, the accuracy of the fitting was around 0.06 when first using the sequential model. However, the RNN did improve this with multiple layers, giving around 0.8 fitting accuracy. If the initial dissimilarity was due to features being disjoint as with section 4.2, the hypothesis was correct that the deep learning represented by the multiple layers allowed the model to find greater similarities between the two data sets.

One thing to note is that in every single run of the RNN model; all documents were always on the same classification, meaning there were no false positives. This is interesting because it brings into consideration that the documents within BDI are too similar for classification, which follows from the method of creating the data set. Since there was little deviation between phrases of the BDI, coupled with 20 other phrases within the same document that suffer from the same redundancy issue, it is clear that the setup for this dataset was not properly balanced for classification.

5 Conclusion & Future Work

After attempting to classify patient documents with two different approaches, the various models were unable to properly diagnose the mental health condition of depression. Despite this, ten different mental health conditions were identified through several classification methods with varying levels of success. Throughout the process, many issues were able to be identified with both the datasets and the initial setup of the problem.

From multiple classification models, document classification can be used successfully with diagnosis documents to classify other diagnosis documents. Due to the small size of this dataset, there is a risk of overfitting. When training and fitting to a single fold and testing on each of the ten folds, the accuracy level was significantly lower on the folds that were not trained on. Along with the false positives for a single class, the need for a larger or different dataset is apparent. However, the accuracy for the

majority of the classes was promising enough to attempt classification with a patient dataset.

With the goal of diagnosing from patient text data, it was difficult to acquire a useful and relevant dataset, but the BDI set was found and ran as the test set. Though the BDI set seemed to fit the need for this thesis, it was not the optimal choice for several reasons.

First, there was only a single diagnosis contained within the dataset, which may have reduced the effectiveness of the multiclass training set. Since the classification of "Not Depression" did not necessarily coincide with any of the other diagnoses that were also classified as "Not Depression," there was no true connection within the training set that represented a neutral patient diagnosis. This made the single diagnosis set ultimately incompatible with a training set consisting of only diagnoses.

Second, the structure of each phrase within the documents were repetitive and gave little variety between each document, regardless of the diagnosis assigned. This also was exacerbated by the training set which contained diverse and unique documents for each diagnosis. Despite the attempts to find similar elements through RNN and deep learning, the generic structure of the BDI prevented true classification from the diverse training set.

Third, the dataset was not large. Though there was no research done on if a larger dataset would have provided results, a larger set would have made it easier to depict and understand the accuracy of the models in a more efficient manner. The

survey at the base of the BDI could have been used to artificially create a larger dataset, but for the purposes of this thesis, authenticity to actual patient diagnosis was important. It would not be a useful feature if an artificially created document was classified, as that would not represent any real diagnostic use this thesis might provide.

Finally, the feature set generated from the BDI set was not only small but also disjointed from the training feature set. This caused two separate approaches in section 4.2.B in order to use the test set with the training set. The features within the BDI were also not meaningful enough to provide a diagnosis when the portable feature set was created. Not having enough features and not having features relevant to the key components of the training set made it impossible to classify the test set with any of the feature based predictive algorithms.

If the goal for this thesis were to be attempted again, it is clear that the test data must be carefully selected to reduce the flaws seen within the BDI. The test data needs to contain more organically written documents rather than results generated from a survey. There also needs to be multiple diagnosis represented to match with the training set that this thesis is based on. The most important thing is finding access to such a dataset that does not include personal health information that would violate any privacy policy. This is the key reason why a suitable dataset was not able to be acquired for the purpose of this thesis. But even if all of those points were met, there is little chance of proper diagnosis unless natural language and features within the testing set is compatible with the training set.

The DSM-5 being used as the training set was the main foundation of this thesis. The idea was to create a model that could diagnose a patient in a similar way and use the same tool(s) that a mental health professional might use. While this concept seemed simple, it did not account of the complexity of the diagnosis documents within the DSM-5. While the documents did contain information pertaining to each diagnosis, the presentation and features contained within were not accessible to common natural language that a patient might use. The DSM-5 was able to be classified among itself when split into both training and testing datasets, but when brought in front of an outside patient dataset, it struggled to fit the models in a meaningful way for the patient dataset to be properly classified. Even though there were issues with the BDI test set as enumerated above, there were also faults identified within the training set that need to be addressed in any future mental health diagnosis project.

It may be the case that the raw documents from DSM-5 as a foundation of classifying patient diagnosis may not be an optimal solution, but there are improvements that could be made to have a true automatic patient diagnosis system. The first improvement was explored by Yong Chen et al. [6] where a specific diagnosis was evaluated for classification by directly applying elements of diagnostic surveys to the training data. By custom building the training set based on the information found within the DSM-5, a more user-friendly training set could be developed and could prove to be compatible with a wider range of natural language within patient datasets. The main drawback of this method is the requirement of having a mental health professional present during the creation of the training set. This is to ensure that all updated

diagnosis data is properly translated from the DSM-5 and still represents the key diagnostic traits of each diagnosis. This new training set should be able to find diagnosis results with any predictive algorithms if the set was developed properly, but further testing should find the model that best fits the new set.

Though all the classification methods equally failed to classify the BDI within the experiments in this thesis, some did perform better than others. All of the feature-based classification algorithms that required a TF-IDF input were all equally shown to not perform well. This is due to the specific nature of mental health diagnosis where the sentiment matters more than the actual features represented. That is why RNN is predicted to be the best model for patient diagnosis. Not only does RNN utilize LSTM to analyze the relationships between words in a given document, but the deep learning also allows for a custom generated feature set that better fits the classification needs for the given datasets. Even though RNN is the best fit out of all methods used in this thesis, there are still more classification methods and strategies that were not attempted and could potentially provide diagnosis classification results if given appropriate training and testing datasets.

Several classification methods using machine learning techniques have been developed, and each have their own strengths. The methods used in experiments within this thesis were chosen based on the strengths that fit the problem; however there are still some beneficial methods that were not used. Cloud integration [16] was a method that was considered and ultimately not used due to the limited customization of the machine learning system. This could have implemented several techniques including

SVM and Naïve Bayes, but the cloud system was too rigid for the design this thesis required. Another deep learning method called Transformers [12] was also considered but was not used due to time constraints. There is no way of claiming that these methods would have provided results, but based on the analysis of the datasets, it is doubtful the additional properties in these classification methods would have made a noticeable difference.

Overall, there was no way found to automatically diagnose a patient using classification algorithms based on the DSM-5. Despite the lack of results, many ideas were explored throughout the process of updating the model in the hope of creating the proper diagnosis. After many failed attempts, each step taken was understood and analyzed to open the door for more successful automatic diagnosis systems.

References

[1] American Psychiatric Association, “Diagnostic and statistical manual of mental disorders, 5th ed.”, *American Psychiatric Publishing*, 2013

[2] M. Abdollahi, X. Gao, Y. Mei, S. Ghosh, and J. Li, “An Ontology-based Two-Stage Approach to Medical Text Classification with Feature Selection by Particle Swarm Optimisation”, *2019 IEEE Congress on Evolutionary Computation,* 2019

[3] G. Anasari, T. Ahmad, and M. N. Doja, “Hybrid Filter–Wrapper Feature Selection Method for Sentiment Classifcation”, *2019 Arabian Journal for science and engineering,* 2019

[4] G. Azar, N. El-Bathy, Su Yu, R. H. Neela, and K. Alfarwati, “Intelligent Mental Health Diagnosis Architecture using Data Mining and Machine Learning”, *International Conference of Scientific Computer 2016,* 2017

[5] A. T. Beck, C. H. Ward, M. Mendelson, J. Mock, and J. Erbaugh, “An inventory for measuring depression”, *Archives of general psychiatry*, 1961

[6] Y. Chen, B. Zhou, W. Zhang, W. Gong, and G. Sun, “Sentiment Analysis Based on Deep Learning and its Application in Screening for Perinatal Depression”, *2018 IEE Third International Conference on Data Science in Cyberspace,* 2018

[7] R. Dzisevič and D. Šešok, “Text Classification using Different Feature Extraction Approaches”, *2019 Open Conference of Electrical, Electronic and Information Sciences,* 2019

[8] Z. Elberrichi, A. Rahmoun, and M. A. Bentaalah. "Using WordNet for Text Categorization." *International Arab Journal of Information Technology (IAJIT) 5*, 2008

[9] J. Ive, G. Gkotsis, R. Dutta, R. Stewart, S. Velupillai, “Hierarchical neural model with attention mechanisms for the classification of social media text related to mental health”, *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic,* 2018

[10] M. Jamaluddin and A. D. Wibawa, “Patient Diagnosis Classification based on Electronic Medical Record using Text Mining and Support Vector Machine”, *2021 International Seminar on Application for Technology of Information and Communication,* 2021

[11] C. Musto, M. Gemmis, G. Semeraro, and P. Lops, **“**A Multi-criteria Recommender System Exploiting Aspect-based Sentiment Analysis of Users’ Reviews”, *Proceedings of Rec-Sys’17,* 2017

[12] H. Park, Y. Vyas, and K. ShahEfficient, “Classification of Long Documents Using Transformers”, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2022

[13] B. Partak and A. K. Uysal, “The impact of feature selection on medical document classification”, *2016 11th Iberian Conference on Information Systems and Technologies,* 2016

[14] L. Qing, W. Linhong, and D. Xuehai, “A Novel Neural Network-Based Method for Medical Text Classification”, *Future Internet*, 2019

[15] A. Sarker and G. Gonzalez, “Portable automatic text classification for adverse drug reaction detection via multi-corpus training”, *Journal of Biomedical Informatics Volume 53*, 2015

[16] G. Satyanarayana, J. Bhuvana, and M. Balamurugan, “Sentimental Analysis on voice using AWS Comprehend”, *2020 International Conference on Computer Communication and Informatics,* 2020

[17] S. Scott, and S. Matwin, “Feature engineering for text classification”, *InICML 1999,* 1999

[18] F. P. Shah and V. Patel, “A Review on Feature Selection and Feature Extraction for Text Classification”, *IEEE WiSPNET 2016 Conference,* 2016

[19] L. Zhang, S. Wang, B. Liu “Deep Learning for Sentiment Analysis: A Survey”, *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery,* 2018

VITA

Author:	William Tadlock
Place of Birth:	Redmond, Washington
Undergraduate School Attended:	Whitworth University
Degrees Awarded:	Bachelor of Science Computer Science, 2018, Whitworth University
	Bachelor of Science Mathematics, 2018, Whitworth University
Honors and Awards:	Graduate Assistantship, Eastern Washington University, 2018-2019 and 2021-2022, Eastern Washington University
Professional Experience:	Internship, The Software Revolution Inc., Kirkland, Washington, 2015
	Internship, NextIT, Spokane Valley, Washington, 2016-2017