Sentiment Classification for Mental Health Status *

Russell Loniello

Determining an individuals mental health status based on text alone is a complex problem for machines and people[1]. In this document I explore the use of sentiment classification to classify a particular text string into a category of various mental health statuses that represent a wide range of normal and abnormal mental health states. I show a proof of concept workflow with the dataset Sentiment Analysis for Mental Health using Create ML and standard R workflow proceedures. Discussion continues with the sentiment classification of public forms, interpersonal messages, news articles and more using similar methods.

Keywords: Sentiment Classification, Create ML, RStudio, TM, Parallel Computing, Mental Health

Introduction

Identifying an individual's emotional and psychological state by text alone is a difficult problem to solve. Generally, at any given moment people can be classified in to several classes or types of mental health well-being and ailments.

Assuming the true nature of one's internal mental status is reflected in their speech we can use machine learning and natural language processing to determine which classification the individual's speech belongs. This article explores an somewhat easier problem; The classification of sentiment of text as an indicator of mental health status. Without the use of intonation, and other speech related aspects the problem space is reduced to a system that is easier to analyze and predict. Given enough readings of this sort, we can arrive at a statistical likelihood for mental health status.

Using general practices for sentiment classification of plain text I show how to use R to import, clean and separate data into a training set and testing sets. Then, using `caret` and `Create ML` to generate models for comparison. Both models yielded acceptable performance (~0.80%) for basic analysis with Create ML being substantially faster to create and deploy.

Cleaning Data

The data being used has 53043 observations and 3 variables from multiple data sets. Per the author, "The dataset amalgamates raw data from multiple sources, cleaned and compiled to create a robust resource for developing chatbots and performing sentiment analysis." For more information see the link provided above in the abstract section.

The dataset includes seven mental health statuses (called *status*) that will be used as the classes to be analyzed: Normal, Anxiety, Bipolar, Depression, Suicidal, Stress, and Personality Disorder. Notably, the majority of instances are categorized as Normal (31%) and Depression (29%). This distribution may influence the model's results. However, if the dataset accurately reflects the

*See the author's Github account (http://github.com/rloniello). **Revision**: September 06, 2024;

[1]Zhang, T., Schoene, A.M., Ji, S. et al. Natural language processing applied to mental illness detection: a narrative review. npj Digit. Med. 5, 46 (2022). https://doi.org/10.1038/s41746-022-00589-7

broader population this imbalance should not significantly affect the overall findings in real world applications. The dataset also includes a string value called *statement* which is used to train the model for a target *status*.

Import Data and Libraries

```
library("dplyr")
library("tm")
library("caret")
library("e1071")
library("parallel")
library("doParallel")

mentalHealthData <- read.csv2("Combined Data Original.csv", sep = ",")
```

The file *Combined Data Original.csv* is available on the authors github file packaged along with this document. It is just a renamed version of the existing data available on Kaggle.com from the link above.

Basic Cleaning

Several basic functions where used to clean the data. An ideal text input is a basic string of multiple English words, no punctuation, numbers, or special characters. This is normally how text classification and sentiment analysis is performed, in the initial review of the data there did not appear to a need to change standard workflow proceedures for this dataset. For the status field, a simple check is ensuring every observation has one of a predetermined valid status:

```
isValidStatus <- function(status) {

    validStatuses <- c("Anxiety", "Normal", "Depression",
                       "Suicidal", "Stress", "Bipolar",
                       "Personality disorder")

    status %in% validStatuses
}

healthData <- filter(mentalHealthData, isValidStatus(mentalHealthData$status))
```

Next, we can remove punctuation and numbers with `gsub()`.

```
healthData$statement <- gsub("[0-9.]", "", healthData$statement)
healthData$statement <- gsub("[[:punct:]]", "", healthData$statement)
```

To get the best results from a sentiment classification model, we need the input text to contain only English words. So, we should remove non-English words from the "statement" field before creating a corpus (bag-of-words). We can use the standard dictionary on macOS located in /usr/share/dict/, which includes connectives and proper names. Since connectives and proper names are usually dropped from the document corpus matrix, we don't need to check for them. Since this process can be time-consuming, we can speed it up by using the parallel and `doParallel` packages to perform the operation in `parallel`.[2]:

[2]This task may take several minutes to complete depending on hardware. The task is O^2 because sentences are converted to lists (array) in which every element is parsed against every element in the dictionary until it is found or not. In the worse case, all list words are not in the dictionary.

```
dictionary <- tolower(readLines("./dict/web2"))
checkStatement <- function(statement) {
    trimmedText <- trimws(statement)
    words <- unlist(strsplit(tolower(trimmedText), "\\s+"))
    validWords <- words[words %in% dictionary]
    cleanedText <- paste(validWords, collapse = " ")
    return(cleanedText)
}

cores <- detectCores()

statements <- healthData$statement

cluster <- makeCluster(cores -1)

clusterExport(cluster, list("statements", "checkStatement", "dictionary"))

result <- parLapply(cluster, statements, checkStatement)

healthData$statement <- result
```

Finally, since some observations only contain numbers, punctuation like "…", or other malformed text, cleaning these rows can leave the "statement" field empty. We should remove these empty rows from the dataset. This is also a good time to drop the "X" field, which simply represents the observation row number.

```
healthData <- filter(healthData, healthData$statement != "")

healthData <- select(healthData, -X)
```

You ought to be left with 52303 Observations with 2 Variables. Now, we can split the data into training and testing data in the next section.

Analysis with XCode's Create ML

Create ML is Apple's suite for generating high-performance machine learning models that can be integrated into Apple devices. For our purposes, we'll use Create ML's preview functionality to showcase the model in action, as well as, compare to results in R with caret.

First, we need to prepare our data by splitting it into training and testing sets. After splitting, we'll relabel the data to fit Create ML's text classification format. Specifically, we'll convert our data columns to "text" and "label," which correspond to the "statement" and "status" columns in our dataset.

The next steps are to split the data, export it, and rename the columns accordingly. Renaming can be done in R or simply open the file in a text editior and change the field names at the begining of the document.

```
set.seed(1337)
trainIndex <- createDataPartition(healthData$status, p = 0.7, list = FALSE)
trainingData <- healthData[trainIndex, ]
testingData <- healthData[-trainIndex, ]

# Use a temp data frame (df) for text conversion.
# 36,615 Observations
df <- apply(trainingData,2,as.character)
write.csv(trainingData, "TrainingData.csv", row.names = FALSE)
```

```r
# 15,688 Observations
df <- apply(testingData,2,as.character)
write.csv(testingData, "TestingData.csv", row.names = FALSE)
```

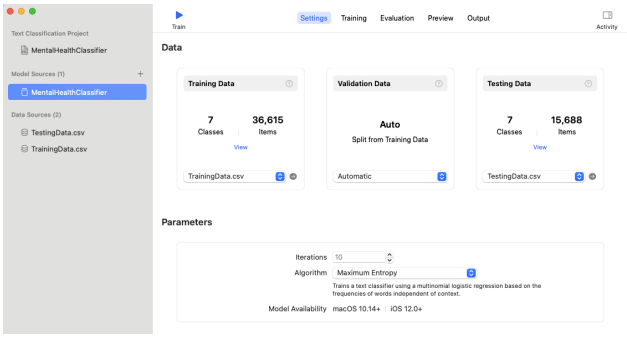

Figure 1: Image of Create ML's Interface, default settings used.

This model may take several minutes to train depending on hardware, selecting the Preview tab. Type something into the text box to try out the model.

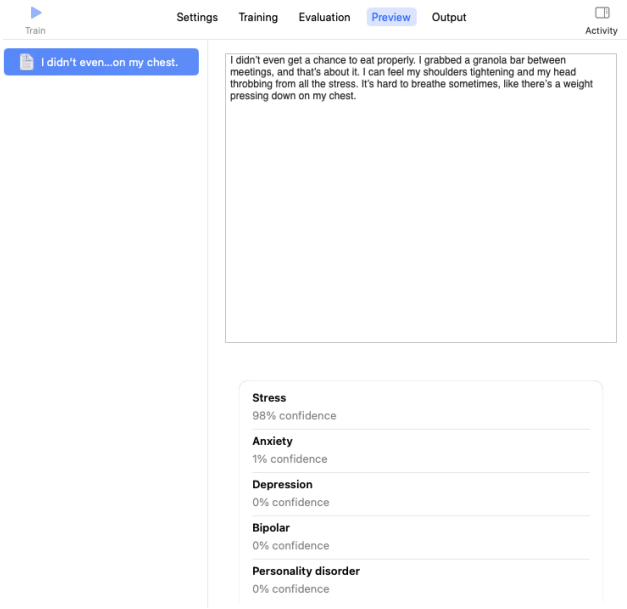

Figure 2: Example Text from a "stressful" journal entry.

Analysis with Caret using a SVM

Previous cross-validation concluded that SVM was best suited for this analysis. Although, other types of models such as Naive Bayes is also suitable for some Sentiment Classification. In the case presented uses 640 features and 7 classes, this was determined to be too advanced for simple Naive Bayes.

Using the same data partition above we can train a SVM with Caret in R. To do this, we need to convert the data frame to bag-of-words representation. Then, eliminate common stop-words only keeping those words that carry the most significance. Again, we can use parallel and doParallel to help speed up this task:

Warning: This is a time consuming task that may take several hours to complete.

```
# See cluster creation above.
registerDoParallel(cluster)

corpus <- Corpus(VectorSource(healthData$statement))
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removeWords, stopwords("en"))
corpus <- tm_map(corpus, stripWhitespace)

dtm <- DocumentTermMatrix(corpus)
dtmc <- removeSparseTerms(dtm, 0.99)

matrix <- as.matrix(dtmc)
finalHealthData <- as.data.frame(matrix)

finalHealthData$status <- as.factor(healthData$status)

library(e1071)
set.seed(1337)
trainIndex <- createDataPartition(finalHealthData$status, p = 0.7, list = FALSE)
trainingData <- finalHealthData[trainIndex, ]
testingData <- finalHealthData[-trainIndex, ]

trainCntrl <- trainControl(method = "cv", number = 5)

svmLinearModel <- train(x = trainingData, y = trainingData$status,
                        method = "svmLinear", trControl = trainCntrl)

predictions <- predict(model, newdata = testingData)
confusionMatrix(predictions, testingData$status)
```

Support Vector Machine with Linear Kernel

Attribute	Value
Number of samples	36,615
Number of predictors	640
Number of classes	7
Class names	Anxiety, Bipolar, Depression, Normal, Personality disorder, Stress, Suicidal
Pre-processing	None
Resampling method	Cross-Validated (5 fold)
Summary of sample sizes	29,292; 29,292; 29,293; 29,293; 29,290
Accuracy	0.7085351
Kappa	0.6167174
Tuning parameter 'C'	Held constant at a value of 1

Logistic Regression with Create ML

Class	Count	Precision	Recall	F1 Score
Normal	10636	91%	80%	0.85
Bipolar	1856	86%	78%	0.82
Anxiety	2551	79%	85%	0.82
Depression	10226	77%	83%	0.80
Stress	1727	61%	75%	0.68
Personality Disorder	721	74%	71%	0.72

Concusions and Discussion

In this exploration of the available data, we saw two different models. A Support Vector Machine (SVM) with a linear kernel and a Logistic Regression model built using Create ML. Both were trained on about 36k observations, and demonstrated moderate to high performance. The Logistic Regression model demonstrates a strong performance across most classes, with particularly high precision for the Normal class (91%) and relatively balanced recall across other classes. The F1 scores reflect a generally good balance between precision and recall, indicating reliable performance for practical applications. However, the model shows lower performance for Stress and Personality Disorder classes, with F1 scores of 0.68 and 0.72, respectively, suggesting potential areas for further improvement.

While both models operate virtually the same with the same input Create ML (MaxEnt) is trained much faster with more iterations. Using this type of analysis as a built in app for mobile devices can help detect and address mental health changes over time. If an individual suddenly develops a change in perspective it can be an indicator of a serious health issue[3]. This demonstrates that such analysis can be reliable given good data. In the near future, our devices can alert us of changes to our mental health depending upon the messages we type, things we search and, more broadly how we communicate with each other. Using sentiment classification models on mobile devices for detecting mental health status changes offers promising benefits, including early intervention and accessibility.

[3]See "Personality and Behavior Changes" by ByMichael B. First, MD, Columbia University,