Homework 8

Alex Hyman

11/29/2018

## Introduction

Machine learning has often been used to detect sentiment of texts, but some people believe that machine learning on text can be used to detect whether a piece of text is telling a lie. To test this theory, we have been provided with labeled yelp reviews that state whether a review is being truthful or not, and whether the sentiment of the text is positive or negative. A SVM classifier and a Naïve Bayes classifier will be trained for each classification problem, and various performance metrics will be reported for each classification task, and each model.

## Reading the data

The data was provided in a csv file with the first two columns indicating whether the review was truthful or not, and whether the review had a positive or negative sentiment. The reviews were the remaining columns in the csv file, but the reviews were broken up due to the use of commas in the review. To fix this problem, I decided to read in the file by the line, so that the sentiment, lie status, and review were essentially on character column. To convert the data into a record format, the fist position on every line was saved into character vector as the lie, the third position of the line was saved as the sentiment of the review, and the sixth position of the line through the length of the line minus one was saved as the review. The headers were deleted, and the vectors were stored into a data frame, and the sentiment and lie status of the reviews were converted into factors. An ID column was also created to keep track of the records for future processing.

## Text Processing

After the data had been stored into a record table, the text needed to be standardized and tokenized. First the reviews were tokenized and converted into lowercase. After the tokenization step, the words were stemmed, so the suffixes would not affect the meaning of the same root word. Stopwords and any punctuation in the reviews were then removed from the list of tokens, and tokens were then converted into a tf-idf data frame.

## Sentiment

The first classification task completed was the classification of sentiment of the review, or whether the review was positive or negative. The training set was chosen to be 75% of all the reviews, with the test set being the remaining 20%. All performance metrics reported are the model’s performance on the test set.

### SVM

Support vector machines were first used to classify sentiment. Some additional preprocessing was conducted to help the classifier better read the data. The id and lie status of the review was removed from the data frame, and the tf-idf values were normalized. A polynomial kernel and a linear kernel were both tested, and it was determined that the linear kernel performed better. The cost hyper parameter chosen for the model was 0.1 as it performed the best under the cross-validation training.

#Making a data frame for only sentiment analysis  
sentiment\_df <- review\_words\_dtm %>%   
 dplyr::select(-id, -lie)  
#Replacing na with 0  
sentiment\_df[,2:ncol(sentiment\_df)] <- scale(sentiment\_df[,2:ncol(sentiment\_df)])  
#train test split  
set.seed(1000)  
train\_ix <- createDataPartition(sentiment\_df$sentiment, p = 0.75, list = F)  
sentiment.train <- sentiment\_df[train\_ix, ]  
sentiment.test <- sentiment\_df[-train\_ix, ]  
#train control  
tc <- trainControl(method = "cv", number = 10)  
  
#SVM cost grid  
linear.grid = expand.grid(C = c(1, 0.9, 0.8, 0.5, 0.25, 0.1))  
#training linear svm  
svm.sentiment.linear <- suppressWarnings(train(sentiment ~ ., data = sentiment.train, trControl = tc,   
 method = "svmLinear", tuneGrid = linear.grid))  
preds.linear <- predict(svm.sentiment.linear, sentiment.test)  
svm.confmat.linear <- confusionMatrix(preds.linear, sentiment.test$sentiment)  
svm.confmat.linear$table

## Reference  
## Prediction n p  
## n 7 2  
## p 4 9

### Naive Bayes

To train a naïve bayes model on the term data, a new data frame was created to make the term values in the data frame binary, to indicate where a term was present in the review, or not present in the review. The Naïve Bayes model was created to classify sentiment for the review with multiple tunes, and it was determine to be the best tuned model with a Laplace correction value of 1 and to not use the kernel density estimate. A Laplace factor of 1 was likely the correct choice as the matrix is sparse, and having too large of a LaPlace factor could take out meaning of some specific terms.

#Make only binary factors  
sentiment.nb <- review\_words\_dtm %>% dplyr::select(-id, -lie)  
sentiment.nb[sentiment.nb > 0] <- "Yes"

## Warning in Ops.factor(left, right): '>' not meaningful for factors

sentiment.nb[sentiment.nb == 0] <-"No"  
words <- colnames(sentiment.nb)[2:ncol(sentiment.nb)]  
sentiment.nb[words] <- lapply(sentiment.nb[words], factor)  
  
  
#Train and test split  
train.nb <- sentiment.nb[train\_ix,]  
test.nb <- sentiment.nb[-train\_ix,]  
#Training model  
nb.binary <- suppressWarnings(NaiveBayes(sentiment ~ ., data=train.nb, fL = 1, usekernel = F))  
preds <- suppressWarnings(predict(nb.binary, test.nb[,-1]))  
nb.sentiment.confmat <- confusionMatrix(preds$class, test.nb$sentiment)  
nb.sentiment.confmat$table

## Reference  
## Prediction n p  
## n 1 0  
## p 10 11

### Sentiment Results

The two models were evaluated for precision, recall, and accuracy and it was determined that the SVM model performed much better than the Naïve Bayes model with the small sample size. The results and the tuning parameters of the model are as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameter Setting** | **Overall Accuracy** | **Precision for Pos. Sentiment** | **Recall for Pos. Sentiment** | **Precision for Neg. Sentiment** | **Recall for Neg. Sentiment** |
| **SVM**  Cost = 0.1  kernel = linear | 0.7273 | 0.692 | 0.818 | 0.778 | 0.636 |
| **Naïve Bayes**  fL = 1  usekernel = False | 0.5455 | 0.523 | 1.00 | 1.00 | 0.091 |

The performance metrics show that the SVM algorithm performed significantly better than the Naïve Bayes learning algorithm when classifying the sentiment of a review. While the performance was not great for either of model, adding new features to the model could increase performance.

## Lies

The SVM and Naïve Bayes learning algorithms were also applied to determine the truthfulness of a review. The same reviews chosen in the training set and test set for the sentiment analysis were again chosen for the training and test set for the lie analysis. Some general preprocessing was conducted on the original tf-idf data frame to remove the sentiment label and the id of the review.

### SVM

The SVM model for the truthfulness of a review used the scaled tf-idf data frame to classify the truthfulness of the a review. All of the terms that were not present in the data frame were once again changed from NA to 0 before the scaling. The data was scaled because SVM is a distance based algorithm. The training of the model determined that the linear kernel performed better than the polynomial kernel, and that the best cost parameter was determined to be 0.1 by a 10-fold cross validation set.

#Making a data frame for only sentiment analysis  
lies\_df <- review\_words\_dtm %>%   
 dplyr::select(-id, -sentiment)  
#Replacing na with 0  
lies\_df[is.na(lies\_df)] <- 0  
lies\_df[,2:ncol(lies\_df)] <- scale(lies\_df[,2:ncol(lies\_df)])  
#train test split  
set.seed(1000)  
lies.train <- lies\_df[train\_ix, ]  
lies.test <- lies\_df[-train\_ix, ]  
#train control  
tc <- trainControl(method = "cv", number = 10)  
  
#SVM cost grid  
linear.grid <- expand.grid(C = c(1, 0.9, 0.8, 0.5, 0.25, 0.1))  
#training linear svm  
svm.lies.linear <- suppressWarnings(train(lie ~ ., data = lies.train, trControl = tc,   
 method = "svmLinear", tuneGrid = linear.grid))  
preds.linear <- predict(svm.lies.linear, lies.test)  
svm.confmat.linear <- confusionMatrix(preds.linear, lies.test$lie)  
svm.confmat.linear$table

## Reference  
## Prediction f t  
## f 9 6  
## t 2 5

### Naive Bayes

The Naïve Bayes learning algorithm was once again trained to predict the truthfulness of a review. The model was trained on a data frame that stated whether a term was present in a document or not. This was chosen as the data because text documents are so sparse, that a kernel would have difficult determining the proper distribution. Multiple Laplace correction factors were tested for their effectiveness on the model, but because the terms are so sparse and the data was essentially binary, only a Laplace factor of 1 was able to work.

#Binary  
#Make only binary factors  
lies.nb <- review\_words\_dtm %>% dplyr::select(-id, -sentiment)  
lies.nb[lies.nb > 0] <- "Yes"

## Warning in Ops.factor(left, right): '>' not meaningful for factors

lies.nb[lies.nb == 0] <-"No"  
words <- colnames(lies.nb)[2:ncol(lies.nb)]  
lies.nb[words] <- lapply(lies.nb[words], factor)  
  
  
#Train and test split  
train.nb <- lies.nb[train\_ix,]  
test.nb <- lies.nb[-train\_ix,]  
#Training model  
nb.binary <- suppressWarnings(NaiveBayes(lie ~ ., data=train.nb, fL = 1, usekernel = T))  
preds <- suppressWarnings(predict(nb.binary, test.nb[,-1]))  
nb.lies.confmat <- confusionMatrix(preds$class, test.nb$lie)  
nb.lies.confmat$table

## Reference  
## Prediction f t  
## f 6 2  
## t 5 9

### Lie Results

The two models were evaluated for precision, recall, and accuracy and it was determined that the Naïve Bayes model performed slightly better than the SVM model when it comes to detecting lies. The results and the tuning parameters of the models are as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameter Setting** | **Overall Accuracy** | **Precision for Lie** | **Recall for Lie** | **Precision for Truth** | **Recall for Truth** |
| **SVM**  Cost = 0.1  kernel = linear | 0.6364 | 0.600 | 0.818 | 0.714 | 0.455 |
| **Naïve Bayes**  fL = 1  usekernel = True | 0.6818 | 0.75 | 0.545 | 0.818 | 0.642 |

Overall the two models performed similarly, but the Naïve Bayes models was more biased to classify reviews as being truthful, and the SVM model was more biased to classify reviews as being fake. The accuracy of the Naïve Bayes models was slightly better than the SVM model, but depending on what the goal is, it may be better to prioritize the classification of fake reviews.

### Sentiment vs. Lie Detection

Comparing the two tasks, it is apparent that it is a more difficult task to determine whether a review is a lie than to detect the sentiment of a review. I determined this primarily due to the success the SVM model was able to classify the sentiment of the reviews to how both of the models classified lies in terms of accuracy. This intuitively makes sense as there are a lot of different features that go into detecting a lie such as tone, body language, and heart rate, that would not be available when classifying the truthfulness of a document. On the other hand, sentiment is very much dependent on the content of a statement. There are more telling terms to know when someone is being negative or positive compared to someone telling the truth or a lie.

### Variable Importance

To see which terms the were the most deterministic in the sentiment or truthfulness of a review, I used the gain ratio function to see which terms had the greatest effect on the sentiment of truthfulness of the reviews. The binary data frame that was created for both tasks was used as the data for the gain.ratio function, and the top 20 words for log entropy were chosen as having the most impact on the sentiment. The top 20 words in determining sentiment were: "amaz", "terribl", "alwai", "worst", "consist", "eat", "bland", "drink", "atmospher", "feel", "ingredi", "smile", "sushi", "half", "pm", "review", "awesom", "cafe", "celebr", and "chocol". The top 20 terms for sentiment determined by the chi squared function were as follows: "amaz", "terribl", "minut", "alwai", "wait", "friendli" "worst", "found", "eat", "bad", "befor", "consist", "nice", "bland", "drink", "tabl", "wa", "atmospher", "feel", and "ingredi".

The same method was used to determine the most important stemmed terms when detecting lies. The top 20 terms in terms of the truthfulness of a review were: "ic", "cold", "drink", "enter", "tea", "add", "cafe", "dirti", "offer", "onli", "pack", "averag", "celebr", "chocol", "enjoi", "everyon", "expens", "fly", "horribl", and "kitchen". The top 20 terms for truthfulness determined by the chi squared function were as follows: "ic", "cold", "drink", "enter", "tea", "add", "cafe", "dirti", "offer", "onli", "pack", "averag", "celebr", "chocol", "enjoi", "everyon", "expens", "fly", "horribl", and "kitchen".

When comparing the list of the words that have the greatest impact on sentiment versus the terms that have the greatest impact on truthfulness, it is apparent why machine learning is used for sentiment analysis, but not detecting lies. The terms impacting sentiment generally have a positive or negative meaning, that can only be used to mean one thing. The words determined to have the most impact on the truthfulness of a review were more generic and could be used in many different ways. Additionally, the values provided for the gain ratio of the top 20 sentiment words were higher than the gain ratio provided for the top 20 truth detecting words.