Spam Filter

Jeffrey Hunt, Anmoldeep

2022-09-24

Problem Statement

The problem statement for this assignment is to build a spam email detector to reduce the error traffic and improve security of a computer network.

Naive Bayes Algorithm

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Load the required Packages

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(e1071)
library(tibble)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(caTools)
```

Data setup and Expoloration

The spam data set has three columns: An Index Column, a body column (Body of email) and a Label(0 = ham and 1 = spam).

```
spam <- read_csv("completeSpamAssassin.csv")
```

```
## New names:
## Rows: 6046 Columns: 3
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Body dbl (2): ...1, Label
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
glimpse(spam)
```

```
## Rows: 6,046
## Columns: 3
## $ ...1  <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
## $ Body  <chr> "\nSave up to 70% on Life Insurance.\nWhy Spend More Than You Ha~
## $ Label <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

Prepare Training and Testing Sets

```
set.seed(123)

#create a sample of the email dataset that only includes the body texts of the email
#we split the dataset into 80% training and 20% testing
sample_set <- sample(1:dim(spam)[1], dim(spam)[1]*0.80)
spam_train <- spam[sample_set,] #80% section
spam_test <- spam[-sample_set,] #20% section
```

Build the Model

WE build are model, Our dependent variable is Label because we want to know based of the email will the Label be ham or spam. Laplace is a nice add on to the naive Bayes model that will prevent "empty" emails from being added to evaluating the email.

```
model <- naiveBayes(Label ~ ., data = spam_train, laplace = 1)
head(model, 1)
```

```
## $apriori
## Y
##    0    1
## 3308 1528
```

Make predictions

```
pred <- predict(model, newdata = spam_test, type="class")
head(pred)
```

```
## [1] 1 1 1 1 1 1
## Levels: 0 1
```

With the prediction model we can determine whether an email is spam or not. We can see that given the values with either a 0(ham) or 1(spam) it predicts the spam.

create confusion matrix

```
t1 <- table(spam_test$Label, pred)
t1
```

```
##    pred
##       0   1
##   0 839   3
##   1   0 368
```

- When it predicted it was a 0(ham) it was perfect.
- When it predicted it was a 1(spam) it was nearly perfect only making the wrong prediction 3 times.

State Model Accuracy

```
accuracy <- sum(diag(t1)) / nrow(spam_test)
cat("Accuracy: ", accuracy)
```

```
## Accuracy:  0.9975207
```

```
error_rate <- 1-accuracy
cat("\nError Rate: ", error_rate)
```

```
##
## Error Rate:  0.002479339
```

The accuracy of our model is high at 99.8%. As we can see from the confusion matrix it is nearly perfect with it only making a wrong prediction in the Test set 3 times.

This is for verification of Model

```
set.seed(123)

spam2 <- read_csv("lingSpam.csv")
```

```
## New names:
## Rows: 2605 Columns: 3
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Body dbl (2): ...1, Label
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
glimpse(spam2)
```

```
## Rows: 2,605
## Columns: 3
## $ ...1  <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
## $ Body  <chr> "Subject: great part-time or summer job !\n \n * * * * * * * * *~
## $ Label <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

A common way of verifying a model is to test it on another set of data. The lingSpam Data set uses the same column format as the last one so all we need to do is predict the model.

this is for prediction of the lingSpam dataset

```
pred2 <- predict(model, newdata = spam2)
head(pred2)
```

```
## [1] 1 1 1 1 1 1
## Levels: 0 1
```

For Printing the Confusion Matrix

```
t2 <- table(spam2$Label, pred2)
t2
```

```
##    pred2
##        0    1
##   0  708 1464
##   1    0  433
```

Stating the model accuracy

```r
accuracy <- sum(diag(t2)) / nrow(spam_test)
cat("Accuracy: ", accuracy)
```

```
## Accuracy:  0.9429752
```

```r
error_rate <- 1-accuracy
cat("\nError Rate: ", error_rate)
```

```
##
## Error Rate:  0.05702479
```

Conclusion

The accuracy of the 2nd model is 95.3%. The confusion matrix shows perfection on predicting ham words but it does not do a good job in predicting if it is a spam getting only 433/1464 correct. In conclusion we can probably say that the first data set skews the results. For example a word may appear 80% in all non-spam emails in the first data set but in the second data set that same word only appears 15% of the time. A way that we can combat this is to use an even larger database Although it does a poor job in predicting spam emails we can can conclude that this is a conclusive model with a 99.8% accuracy and a 95.3% accuracy.

References

https://padlet.com/isac_artzi/k6zxeuwz2aon5vv https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset