Lab 3 – Part 4

Group 1: Laxman Panthi & Zubin Shah

Overview:

This part (4) of the Lab 3 focuses on the online news popularity dataset. The dataset tells us whether an online news is popular or not. For the same, we try to create an indicator variable based on the mean value of shares and decide whether the news is popular or not. We would first create a predictive model specifically KNN which to determine and check if the accuracy has improved when compared to Lab 1 & 2.

Let's start with importing all the libraries that we would require for performing the analysis.

```r
# load libraries
library(tidyverse)

library(cluster)

library(NbClust)

library(rpart.plot)

library(RColorBrewer)
library(rpart)
library(rattle)
```

Online News Dataset: R Code

Step 1:

First we will load the data and create a new variable based on shares to indicate whether the news is popular or not.

```r
newsShort <- read_csv("OnlineNewsPopularity.csv")%>%
  select("n_tokens_title", "n_tokens_content", "n_unique_tokens",
"n_non_stop_words", "num_hrefs", "num_imgs", "num_videos",
"average_token_length", "num_keywords", "kw_max_max",
"global_sentiment_polarity", "avg_positive_polarity", "title_subjectivity",
"title_sentiment_polarity", "abs_title_subjectivity",
"abs_title_sentiment_polarity", "shares")

## Parsed with column specification:
## cols(
##   .default = col_double(),
##   url = col_character()
## )

## See spec(...) for full column specifications.

newsShort <- newsShort %>% mutate(popular=if_else((shares >= 1400),2,1)) %>%
select(-shares)
newsShort$popular <- as.factor(newsShort$popular)
glimpse(newsShort)

## Observations: 39,644
## Variables: 17
## $ n_tokens_title               <dbl> 12, 9, 9, 9, 13, 10, 8, 12, 11, 1...
## $ n_tokens_content             <dbl> 219, 255, 211, 531, 1072, 370, 96...
## $ n_unique_tokens              <dbl> 0.6635945, 0.6047431, 0.5751295, ...
## $ n_non_stop_words             <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ num_hrefs                    <dbl> 4, 3, 3, 9, 19, 2, 21, 20, 2, 4, ...
## $ num_imgs                     <dbl> 1, 1, 1, 1, 20, 0, 20, 20, 0, 1, ...
## $ num_videos                   <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, ...
## $ average_token_length         <dbl> 4.680365, 4.913725, 4.393365, 4.4...
## $ num_keywords                 <dbl> 5, 4, 6, 7, 7, 9, 10, 9, 7, 5, 8,...
## $ kw_max_max                   <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ global_sentiment_polarity    <dbl> 0.09256198, 0.14894781, 0.3233333...
## $ avg_positive_polarity        <dbl> 0.3786364, 0.2869146, 0.4958333, ...
## $ title_subjectivity           <dbl> 0.5000000, 0.0000000, 0.0000000, ...
## $ title_sentiment_polarity     <dbl> -0.1875000, 0.0000000, 0.0000000,...
## $ abs_title_subjectivity       <dbl> 0.00000000, 0.50000000, 0.5000000...
## $ abs_title_sentiment_polarity <dbl> 0.1875000, 0.0000000, 0.0000000, ...
## $ popular                      <fct> 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 2, ...
```

Step 2:

We will now randomize the data and create training and testing set.

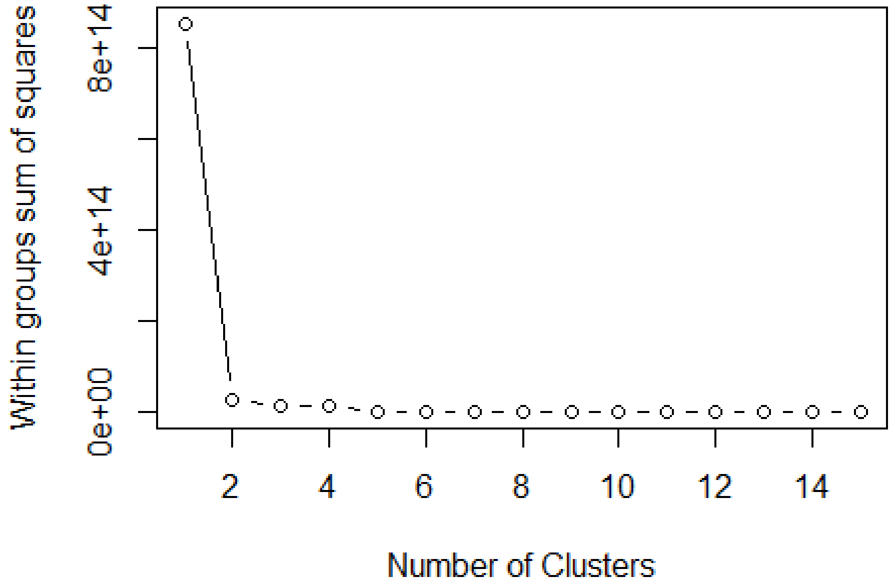
```
news_rand <- newsShort[order(runif(10000)), ]
set.seed(12345)

#Split the data into training and test datasets
news_train <- news_rand[1:9000, ]
news_test <- news_rand[9001:10000, ]
```

Step 3:

Create a “wss” function to plot and identify the optimal number of clusters for the dataset

```
df<- data.frame(news_rand [,-17])
wssplot <- function(data, nc = 15, seed = 1234) {
  wss <- (nrow(data) - 1) * sum(apply(data, 2, var))
  for (i in 2:nc) { set.seed(seed)
```

```
    wss[i] <- sum(kmeans(data, centers = i)$withinss)
    }
  plot(1:nc, wss, type = "b", xlab = "Number of Clusters", ylab = "Within
groups sum of squares")
  }
wssplot(df)
```



Step 4:

From the above, there should be 2 clusters for the dataset. Using the k=2 we will create a model based on the training data and predict the results on the test data.

```
library(class)

## Warning: package 'class' was built under R version 3.5.3

knn_model<- knn(train = news_train, test = news_test, cl=news_train$popular,
k = 2)
table(news_test$popular, knn_model)

##    knn_model
##       1   2
##   1 215 215
##   2 243 327
```

Conclusion & Summary:

From the above, we can see that the accuracy of the model is 54.2% which is comparatively lesser that Naïve Bayes classification.