Comparison of 5 models on a news popularity dataset

Contents

Introduction to the dataset and data preparation for the model

The assignment is to apply the 5 classification methods on the Online News Popularity data set from the UCI Machine Learning Repository. I need to conduct a binary classification (popular/unpopular) using the 58 predictive variables in the dataset or by using a subset of them. Before applying any method, we first need to do some data preparation steps. This includes collecting the data and exploring it. It also include pre-processing the data such as removal of unwanted variables, scaling and setting up the binary classifier. It further involves splitting the data into train and test for the model. Each step is performed as below.

Collecting and exploring the dataset

```
onlinepop <-
read.csv("C:/Users/inatara/Downloads/OnlineNewsPopularity.csv")
str(onlinepop)
```

```
## 'data.frame':    39644 obs. of  61 variables:
##  $ url                          : Factor w/ 39644 levels
"http://mashable.com/2013/01/07/amazon-instant-video-browser/",..: 1 2
3 4 5 6 7 8 9 10 ...
##  $ timedelta                    : num  731 731 731 731 731 731 731
731 731 731 ...
##  $ n_tokens_title               : num  12 9 9 9 13 10 8 12 11 10
...
##  $ n_tokens_content             : num  219 255 211 531 1072 ...
##  $ n_unique_tokens              : num  0.664 0.605 0.575 0.504
0.416 ...
##  $ n_non_stop_words             : num  1 1 1 1 1 ...
##  $ n_non_stop_unique_tokens     : num  0.815 0.792 0.664 0.666
0.541 ...
##  $ num_hrefs                    : num  4 3 3 9 19 2 21 20 2 4 ...
##  $ num_self_hrefs               : num  2 1 1 0 19 2 20 20 0 1 ...
##  $ num_imgs                     : num  1 1 1 1 20 0 20 20 0 1 ...
##  $ num_videos                   : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ average_token_length         : num  4.68 4.91 4.39 4.4 4.68 ...
```

```
##  $ num_keywords                 : num  5 4 6 7 7 9 10 9 7 5 ...
##  $ data_channel_is_lifestyle    : num  0 0 0 0 0 0 1 0 0 0 ...
##  $ data_channel_is_entertainment: num  1 0 0 1 0 0 0 0 0 0 ...
##  $ data_channel_is_bus          : num  0 1 1 0 0 0 0 0 0 0 ...
##  $ data_channel_is_socmed       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ data_channel_is_tech         : num  0 0 0 0 1 1 0 1 1 0 ...
##  $ data_channel_is_world        : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ kw_min_min                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_max_min                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_avg_min                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_min_max                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_max_max                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_avg_max                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_min_avg                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_max_avg                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ kw_avg_avg                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ self_reference_min_shares    : num  496 0 918 0 545 8500 545 545
0 0 ...
##  $ self_reference_max_shares    : num  496 0 918 0 16000 8500 16000
16000 0 0 ...
##  $ self_reference_avg_sharess   : num  496 0 918 0 3151 ...
##  $ weekday_is_monday            : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ weekday_is_tuesday           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday_is_wednesday         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday_is_thursday          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday_is_friday            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday_is_saturday          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ weekday_is_sunday            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ is_weekend                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ LDA_00                       : num  0.5003 0.7998 0.2178 0.0286
0.0286 ...
##  $ LDA_01                       : num  0.3783 0.05 0.0333 0.4193
0.0288 ...
##  $ LDA_02                       : num  0.04 0.0501 0.0334 0.4947
0.0286 ...
##  $ LDA_03                       : num  0.0413 0.0501 0.0333 0.0289
0.0286 ...
##  $ LDA_04                       : num  0.0401 0.05 0.6822 0.0286
0.8854 ...
##  $ global_subjectivity          : num  0.522 0.341 0.702 0.43 0.514
...
##  $ global_sentiment_polarity    : num  0.0926 0.1489 0.3233 0.1007
0.281 ...
##  $ global_rate_positive_words   : num  0.0457 0.0431 0.0569 0.0414
0.0746 ...
##  $ global_rate_negative_words   : num  0.0137 0.01569 0.00948
```

```
0.02072 0.01213 ...
##  $ rate_positive_words          : num  0.769 0.733 0.857 0.667 0.86
...
##  $ rate_negative_words          : num  0.231 0.267 0.143 0.333 0.14
...
##  $ avg_positive_polarity        : num  0.379 0.287 0.496 0.386
0.411 ...
##  $ min_positive_polarity        : num  0.1 0.0333 0.1 0.1364 0.0333
...
##  $ max_positive_polarity        : num  0.7 0.7 1 0.8 1 0.6 1 1 0.8
0.5 ...
##  $ avg_negative_polarity        : num  -0.35 -0.119 -0.467 -0.37 -
0.22 ...
##  $ min_negative_polarity        : num  -0.6 -0.125 -0.8 -0.6 -0.5 -
0.4 -0.5 -0.5 -0.125 -0.5 ...
##  $ max_negative_polarity        : num  -0.2 -0.1 -0.133 -0.167 -
0.05 ...
##  $ title_subjectivity           : num  0.5 0 0 0 0.455 ...
##  $ title_sentiment_polarity     : num  -0.188 0 0 0 0.136 ...
##  $ abs_title_subjectivity       : num  0 0.5 0.5 0.5 0.0455 ...
##  $ abs_title_sentiment_polarity : num  0.188 0 0 0 0.136 ...
##  $ shares                       : int  593 711 1500 1200 505 855
556 891 3600 710 ...
```

```r
summary(onlinepop)
```

```
##                                                                 url
##  http://mashable.com/2013/01/07/amazon-instant-video-browser/  :
1
##  http://mashable.com/2013/01/07/ap-samsung-sponsored-tweets/   :
1
##  http://mashable.com/2013/01/07/apple-40-billion-app-downloads/:
1
##  http://mashable.com/2013/01/07/astronaut-notre-dame-bcs/      :
1
##  http://mashable.com/2013/01/07/att-u-verse-apps/              :
1
##  http://mashable.com/2013/01/07/beewi-smart-toys/              :
1
##  (Other)
:39638
##    timedelta     n_tokens_title n_tokens_content n_unique_tokens
##  Min.   :  8.0   Min.   : 2.0   Min.   :   0.0   Min.   :  0.0000
##  1st Qu.:164.0   1st Qu.: 9.0   1st Qu.: 246.0   1st Qu.:  0.4709
##  Median :339.0   Median :10.0   Median : 409.0   Median :  0.5392
##  Mean   :354.5   Mean   :10.4   Mean   : 546.5   Mean   :  0.5482
```

```
##  3rd Qu.:542.0   3rd Qu.:12.0   3rd Qu.: 716.0   3rd Qu.:  0.6087
##  Max.   :731.0   Max.   :23.0   Max.   :8474.0   Max.   :701.0000
##
##  n_non_stop_words    n_non_stop_unique_tokens   num_hrefs
##  Min.   :   0.0000   Min.   :  0.0000         Min.   :  0.00
##  1st Qu.:   1.0000   1st Qu.:  0.6257         1st Qu.:  4.00
##  Median :   1.0000   Median :  0.6905         Median :  8.00
##  Mean   :   0.9965   Mean   :  0.6892         Mean   : 10.88
##  3rd Qu.:   1.0000   3rd Qu.:  0.7546         3rd Qu.: 14.00
##  Max.   :1042.0000   Max.   :650.0000         Max.   :304.00
##
##  num_self_hrefs       num_imgs         num_videos
average_token_length
##  Min.   :  0.000   Min.   :  0.000   Min.   : 0.00   Min.   :0.000
##  1st Qu.:  1.000   1st Qu.:  1.000   1st Qu.: 0.00   1st Qu.:4.478
##  Median :  3.000   Median :  1.000   Median : 0.00   Median :4.664
##  Mean   :  3.294   Mean   :  4.544   Mean   : 1.25   Mean   :4.548
##  3rd Qu.:  4.000   3rd Qu.:  4.000   3rd Qu.: 1.00   3rd Qu.:4.855
##  Max.   :116.000   Max.   :128.000   Max.   :91.00   Max.   :8.042
##
##   num_keywords    data_channel_is_lifestyle
data_channel_is_entertainment
##  Min.   : 1.000   Min.   :0.00000           Min.   :0.000
##  1st Qu.: 6.000   1st Qu.:0.00000           1st Qu.:0.000
##  Median : 7.000   Median :0.00000           Median :0.000
##  Mean   : 7.224   Mean   :0.05295           Mean   :0.178
##  3rd Qu.: 9.000   3rd Qu.:0.00000           3rd Qu.:0.000
##  Max.   :10.000   Max.   :1.00000           Max.   :1.000
##
##  data_channel_is_bus data_channel_is_socmed data_channel_is_tech
##  Min.   :0.0000      Min.   :0.0000         Min.   :0.0000
##  1st Qu.:0.0000      1st Qu.:0.0000         1st Qu.:0.0000
##  Median :0.0000      Median :0.0000         Median :0.0000
##  Mean   :0.1579      Mean   :0.0586         Mean   :0.1853
##  3rd Qu.:0.0000      3rd Qu.:0.0000         3rd Qu.:0.0000
##  Max.   :1.0000      Max.   :1.0000         Max.   :1.0000
##
##  data_channel_is_world   kw_min_min       kw_max_min
kw_avg_min
##  Min.   :0.0000        Min.   : -1.00   Min.   :     0   Min.   :
-1.0
##  1st Qu.:0.0000        1st Qu.: -1.00   1st Qu.:   445   1st Qu.:
141.8
##  Median :0.0000        Median : -1.00   Median :   660   Median :
235.5
##  Mean   :0.2126        Mean   : 26.11   Mean   :  1154   Mean   :
```

```
312.4
##  3rd Qu.:0.0000        3rd Qu.:  4.00   3rd Qu.:  1000   3rd Qu.:
357.0
##  Max.   :1.0000        Max.   :377.00   Max.   :298400   Max.
:42827.9
##
##    kw_min_max       kw_max_max       kw_avg_max       kw_min_avg
##  Min.   :     0   Min.   :     0   Min.   :     0   Min.   :  -1
##  1st Qu.:     0   1st Qu.:843300   1st Qu.:172847   1st Qu.:   0
##  Median :  1400   Median :843300   Median :244572   Median :1024
##  Mean   : 13612   Mean   :752324   Mean   :259282   Mean   :1117
##  3rd Qu.:  7900   3rd Qu.:843300   3rd Qu.:330980   3rd Qu.:2057
##  Max.   :843300   Max.   :843300   Max.   :843300   Max.   :3613
##
##    kw_max_avg       kw_avg_avg    self_reference_min_shares
##  Min.   :     0   Min.   :    0   Min.   :     0
##  1st Qu.:  3562   1st Qu.: 2382   1st Qu.:   639
##  Median :  4356   Median : 2870   Median :  1200
##  Mean   :  5657   Mean   : 3136   Mean   :  3999
##  3rd Qu.:  6020   3rd Qu.: 3600   3rd Qu.:  2600
##  Max.   :298400   Max.   :43568   Max.   :843300
##
##  self_reference_max_shares self_reference_avg_sharess
weekday_is_monday
##  Min.   :     0            Min.   :     0.0           Min.   :0.000
##  1st Qu.:  1100            1st Qu.:   981.2           1st Qu.:0.000
##  Median :  2800            Median :  2200.0           Median :0.000
##  Mean   : 10329            Mean   :  6401.7           Mean   :0.168
##  3rd Qu.:  8000            3rd Qu.:  5200.0           3rd Qu.:0.000
##  Max.   :843300            Max.   :843300.0           Max.   :1.000
##
##  weekday_is_tuesday weekday_is_wednesday weekday_is_thursday
##  Min.   :0.0000     Min.   :0.0000       Min.   :0.0000
##  1st Qu.:0.0000     1st Qu.:0.0000       1st Qu.:0.0000
##  Median :0.0000     Median :0.0000       Median :0.0000
##  Mean   :0.1864     Mean   :0.1875       Mean   :0.1833
##  3rd Qu.:0.0000     3rd Qu.:0.0000       3rd Qu.:0.0000
##  Max.   :1.0000     Max.   :1.0000       Max.   :1.0000
##
##  weekday_is_friday weekday_is_saturday weekday_is_sunday
is_weekend
##  Min.   :0.0000    Min.   :0.00000     Min.   :0.00000   Min.
:0.0000
##  1st Qu.:0.0000    1st Qu.:0.00000     1st Qu.:0.00000   1st
Qu.:0.0000
##  Median :0.0000    Median :0.00000     Median :0.00000   Median
```

```
:0.0000
##  Mean   :0.1438    Mean   :0.06188     Mean   :0.06904   Mean
:0.1309
##  3rd Qu.:0.0000    3rd Qu.:0.00000     3rd Qu.:0.00000   3rd
Qu.:0.0000
##  Max.   :1.0000    Max.   :1.00000     Max.   :1.00000   Max.
:1.0000
##
##      LDA_00            LDA_01            LDA_02            LDA_03
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.
:0.00000
##  1st Qu.:0.02505   1st Qu.:0.02501   1st Qu.:0.02857   1st
Qu.:0.02857
##  Median :0.03339   Median :0.03334   Median :0.04000   Median
:0.04000
##  Mean   :0.18460   Mean   :0.14126   Mean   :0.21632   Mean
:0.22377
##  3rd Qu.:0.24096   3rd Qu.:0.15083   3rd Qu.:0.33422   3rd
Qu.:0.37576
##  Max.   :0.92699   Max.   :0.92595   Max.   :0.92000   Max.
:0.92653
##
##      LDA_04        global_subjectivity global_sentiment_polarity
##  Min.   :0.00000   Min.   :0.0000      Min.   :-0.39375
##  1st Qu.:0.02857   1st Qu.:0.3962      1st Qu.: 0.05776
##  Median :0.04073   Median :0.4535      Median : 0.11912
##  Mean   :0.23403   Mean   :0.4434      Mean   : 0.11931
##  3rd Qu.:0.39999   3rd Qu.:0.5083      3rd Qu.: 0.17783
##  Max.   :0.92719   Max.   :1.0000      Max.   : 0.72784
##
##  global_rate_positive_words global_rate_negative_words
rate_positive_words
##  Min.   :0.00000            Min.   :0.000000           Min.
:0.0000
##  1st Qu.:0.02838            1st Qu.:0.009615           1st
Qu.:0.6000
##  Median :0.03902            Median :0.015337           Median
:0.7105
##  Mean   :0.03962            Mean   :0.016612           Mean
:0.6822
##  3rd Qu.:0.05028            3rd Qu.:0.021739           3rd
Qu.:0.8000
##  Max.   :0.15549            Max.   :0.184932           Max.
:1.0000
##
##  rate_negative_words avg_positive_polarity min_positive_polarity
```

```
##  Min.   :0.0000      Min.   :0.0000        Min.   :0.00000
##  1st Qu.:0.1852      1st Qu.:0.3062        1st Qu.:0.05000
##  Median :0.2800      Median :0.3588        Median :0.10000
##  Mean   :0.2879      Mean   :0.3538        Mean   :0.09545
##  3rd Qu.:0.3846      3rd Qu.:0.4114        3rd Qu.:0.10000
##  Max.   :1.0000      Max.   :1.0000        Max.   :1.00000
##
##  max_positive_polarity avg_negative_polarity min_negative_polarity
##  Min.   :0.0000        Min.   :-1.0000       Min.   :-1.0000
##  1st Qu.:0.6000        1st Qu.:-0.3284       1st Qu.:-0.7000
##  Median :0.8000        Median :-0.2533       Median :-0.5000
##  Mean   :0.7567        Mean   :-0.2595       Mean   :-0.5219
##  3rd Qu.:1.0000        3rd Qu.:-0.1869       3rd Qu.:-0.3000
##  Max.   :1.0000        Max.   : 0.0000       Max.   : 0.0000
##
##  max_negative_polarity title_subjectivity title_sentiment_polarity
##  Min.   :-1.0000       Min.   :0.0000     Min.   :-1.00000
##  1st Qu.:-0.1250       1st Qu.:0.0000     1st Qu.: 0.00000
##  Median :-0.1000       Median :0.1500     Median : 0.00000
##  Mean   :-0.1075       Mean   :0.2824     Mean   : 0.07143
##  3rd Qu.:-0.0500       3rd Qu.:0.5000     3rd Qu.: 0.15000
##  Max.   : 0.0000       Max.   :1.0000     Max.   : 1.00000
##
##  abs_title_subjectivity abs_title_sentiment_polarity     shares
##  Min.   :0.0000         Min.   :0.0000               Min.   :     1
##  1st Qu.:0.1667         1st Qu.:0.0000               1st Qu.:   946
##  Median :0.5000         Median :0.0000               Median :  1400
##  Mean   :0.3418         Mean   :0.1561               Mean   :  3395
##  3rd Qu.:0.5000         3rd Qu.:0.2500               3rd Qu.:  2800
##  Max.   :0.5000         Max.   :1.0000               Max.   :843300
##
```

```
nrow(onlinepop)
```

```
## [1] 39644
```

```
#Median value of shares
summary(onlinepop$shares)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1     946    1400    3395    2800  843300
```

Pre-processing the data

Deleting url and timedelta variables as these are the 2 non-predictive variables

```
onlinepop1 <-subset(onlinepop, select = -c(url, timedelta ) )
ncol(onlinepop1)
```

```
## [1] 59
```

From the summary, we see that the all the variables are numeric but have very different scales. Hence, we need to normalize the data. I use the scale function available in R and apply to it all columns except the goal field called shares. The modification for shares is discussed in the next step.

```
for(i in ncol(onlinepop1)-1){
  onlinepop1[,i]<-scale(onlinepop1[,i], center =TRUE, scale =TRUE)
}
```

In order to conduct a binary classification (popular/unpopular) we define articles with shares greater than 1400 (the median) as popular articles. Using this definition, I create the binary variable shares as below.

```
onlinepop_c <-onlinepop1
onlinepop_c$shares <-ifelse(onlinepop_c$shares >1400,1,0)
onlinepop_c$shares <-as.factor(onlinepop_c$shares)
table(onlinepop_c$shares)
```

```
##
##     0     1
## 20082 19562
```

```
summary(onlinepop_c$n_tokens_title)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     9.0    10.0    10.4    12.0    23.0
```

Split the data

Before splitting the data into train and test, I randomize the data using runif function in R. The checks confirm that randomizing did not make any substantive changes to the data. This is done by comparing the summary of n_tokens_title with the previous dataset and by comparing the distribution of the binary variable shares with the previous dataset. I then do a split of 75% of the data for the training dataset and 25% of the data for the test dataset. The percentage of the 'shares' variable between the training and test dataset is close. Hence the randomization went well.

```
set.seed(1234)
onlinepop_rand <-onlinepop_c[order(runif(39644)),]

#Checks to see if randomization affected the data
table(onlinepop_rand$shares)

##
##     0     1
## 20082 19562

summary(onlinepop_rand$n_tokens_title)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     9.0    10.0    10.4    12.0    23.0

#Actual split
onlinepop_train <-onlinepop_rand[1:29733,]
onlinepop_test <-onlinepop_rand[29734:nrow(onlinepop_c),]
prop.table(table(onlinepop_train$shares))

##
##         0         1
## 0.5054989 0.4945011

prop.table(table(onlinepop_test$shares))

##
##         0         1
## 0.5097367 0.4902633
```

Method 1: Tree based classification

Training a model on the data

Fernandes K. et. al. grouped the attributes of this dataset into 5 categories - Natural Language Processing, Words, Links, Digital Media, Time and Keywords in his paper. I tried to run tree-based classification for all the 58 features of the dataset but it took too much processing time and hence had to be aborted. Therefore, I split the features as mentioned in Fernandes K. et. al. and built 5 different models.

```
library(C50)
```

```
## Warning: package 'C50' was built under R version 3.3.3
```

```
#Natural Language Processing
onlinepop_tc <-C5.0.default(x = onlinepop_train[,38:58], y =
onlinepop_train$shares)
onlinepop_tc
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, 38:58], y =
onlinepop_train$shares)
##
## Classification Tree
## Number of samples: 29733
## Number of predictors: 21
##
## Tree size: 62
##
## Non-standard options: attempt to group attributes
```

```
summary(onlinepop_tc)
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, 38:58], y =
onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:57:49 2017
## -------------------------------
```

```
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (22 attributes) from undefined.data
##
## Decision tree:
##
## LDA_02 > 0.5490565:
## :...min_positive_polarity > 0.03333334:
## :   :...title_sentiment_polarity <= 0.575: 0 (3992/1183)
## :   :   title_sentiment_polarity > 0.575:
## :   :   :...abs_title_subjectivity <= 0.425: 0 (75/31)
## :   :       abs_title_subjectivity > 0.425: 1 (35/12)
## :   min_positive_polarity <= 0.03333334:
## :   :...LDA_03 > 0.2094295: 1 (36/9)
## :       LDA_03 <= 0.2094295:
## :       :...min_negative_polarity > -0.625:
## :           :...LDA_00 <= 0.2000307: 0 (535/224)
## :           :   LDA_00 > 0.2000307: 1 (43/12)
## :           min_negative_polarity <= -0.625:
## :           :...max_negative_polarity <= -0.07692308: 0 (148/38)
## :               max_negative_polarity > -0.07692308:
## :               :...LDA_02 <= 0.8666621: 0 (126/42)
## :                   LDA_02 > 0.8666621: 1 (36/12)
## LDA_02 <= 0.5490565:
## :...LDA_01 > 0.03358202:
##     :...LDA_01 > 0.4825372:
##     :   :...LDA_04 > 0.02500035: 0 (2456/873)
##     :   :   LDA_04 <= 0.02500035:
##     :   :   :...LDA_03 <= 0.02523642: 0 (267/110)
##     :   :       LDA_03 > 0.02523642: 1 (351/149)
##     :   LDA_01 <= 0.4825372:
##     :   :...title_subjectivity > 0.845: 1 (779/313)
##     :       title_subjectivity <= 0.845:
##     :       :...min_positive_polarity <= 0.03333334:
##     :           :...LDA_00 <= 0.2528693: 0 (880/430)
##     :           :   LDA_00 > 0.2528693:
##     :           :   :...rate_negative_words > 0.1052632: 1
(862/302)
##     :           :       rate_negative_words <= 0.1052632:
##     :           :       :...abs_title_subjectivity <= 0.002727273:
0 (26/4)
##     :           :           abs_title_subjectivity > 0.002727273:
##     :           :           :...avg_positive_polarity <= 0.4320833:
1 (132/57)
##     :           :               avg_positive_polarity > 0.4320833:
```

```
0 (11)
##     :           min_positive_polarity > 0.03333334:
##     :           :...global_subjectivity <= 0.3881313: 0 (1137/438)
##     :               global_subjectivity > 0.3881313:
##     :               :...avg_positive_polarity > 0.5720644: 1
(98/30)
##     :                   avg_positive_polarity <= 0.5720644:
##     :                   :...min_positive_polarity <= 0.0625:
##     :                       :...LDA_04 <= 0.4094816: 0 (848/330)
##     :                       :   LDA_04 > 0.4094816: 1 (290/127)
##     :                       min_positive_polarity > 0.0625:
##     :                       :...LDA_02 > 0.03333358: 0 (2647/1258)
##     :                           LDA_02 <= 0.03333358:
##     :                           :...global_rate_positive_words <=
0.04392206: 1 (737/292)
##     :                               global_rate_positive_words >
0.04392206:
##     :                               :...abs_title_subjectivity <=
0.1291667:
##     :                                   :...title_subjectivity <=
0.59375: 0 (177/62)
##     :                                   :   title_subjectivity >
0.59375: 1 (20/6)
##     :                                   abs_title_subjectivity >
0.1291667:
##     :                                   :...max_positive_polarity
<= 0.65: 0 (75/30)
##     :                                       max_positive_polarity >
0.65: 1 (369/164)
##     LDA_01 <= 0.03358202:
##     :...LDA_00 > 0.9199756: 0 (35/6)
##         LDA_00 <= 0.9199756:
##         :...min_positive_polarity <= 0.03333334:
##             :...LDA_03 > 0.4989006:
##             :   :...min_positive_polarity <= 0: 1 (371/147)
##             :   :   min_positive_polarity > 0:
##             :   :   :...max_positive_polarity > 0.7: 1 (252/101)
##             :   :       max_positive_polarity <= 0.7:
##             :   :       :...title_sentiment_polarity <= -0.1818182:
0 (16/2)
##             :   :           title_sentiment_polarity > -0.1818182:
##             :   :           :...LDA_03 <= 0.8851883: 1 (74/26)
##             :   :               LDA_03 > 0.8851883: 0 (33/9)
##             :   LDA_03 <= 0.4989006:
##             :   :...rate_positive_words <= 0.9142857: 1 (2091/600)
##             :       rate_positive_words > 0.9142857:
```

```
##             :       :...LDA_04 > 0.2858283:
##             :           :...title_sentiment_polarity <= -0.0625: 0
(6/1)
##             :           :   title_sentiment_polarity > -0.0625: 1
(59/14)
##             :           LDA_04 <= 0.2858283:
##             :           :...abs_title_subjectivity <= 0.275: 0
(42/9)
##             :               abs_title_subjectivity > 0.275:
##             :               :...abs_title_subjectivity <=
0.4666667: 1 (10/1)
##             :                   abs_title_subjectivity > 0.4666667:
##             :                   :...avg_positive_polarity <=
0.2665318: 0 (8)
##             :                       avg_positive_polarity >
0.2665318:
##             :                       :...LDA_00 <= 0.866574: 1
(22/5)
##             :                           LDA_00 > 0.866574: 0 (12/2)
##             min_positive_polarity > 0.03333334:
##             :...LDA_02 > 0.339902:
##                 :...title_sentiment_polarity <= -0.475: 0 (42/9)
##                 :   title_sentiment_polarity > -0.475:
##                 :   :...max_positive_polarity <= 0.85:
##                 :       :...abs_title_sentiment_polarity <=
1.426127: 0 (772/315)
##                 :       :   abs_title_sentiment_polarity >
1.426127:
##                 :       :   :...global_sentiment_polarity <=
0.09550505: 0 (25/9)
##                 :       :       global_sentiment_polarity >
0.09550505: 1 (31/6)
##                 :       max_positive_polarity > 0.85:
##                 :       :...max_negative_polarity <= -0.08333334: 0
(200/94)
##                 :           max_negative_polarity > -0.08333334:
##                 :           :...abs_title_subjectivity <= 0.0375: 0
(16/5)
##                 :               abs_title_subjectivity > 0.0375: 1
(164/60)
##                 LDA_02 <= 0.339902:
##                 :...max_negative_polarity > -0.008333334:
##                     :...min_positive_polarity <= 0.45: 0 (347/154)
##                     :   min_positive_polarity > 0.45: 1 (16/2)
##                     max_negative_polarity <= -0.008333334:
##                     :...global_subjectivity > 0.451284: 1
```

```
(4916/1891)
##                           global_subjectivity <= 0.451284:
##                           :...abs_title_sentiment_polarity >
0.9373557: 1 (403/148)
##                               abs_title_sentiment_polarity <=
0.9373557:
##                               :...LDA_00 > 0.4031859:
##                                   :...min_negative_polarity <= -0.6:
1 (144/61)
##                                   :   min_negative_polarity > -0.6: 0
(472/185)
##                                   LDA_00 <= 0.4031859:
##                                   :...max_positive_polarity <= 0.35:
0 (49/16)
##                                       max_positive_polarity > 0.35:
[S1]
##
## SubTree [S1]
##
## global_rate_positive_words <= 0.06118143: 1 (1803/803)
## global_rate_positive_words > 0.06118143:
## :...max_negative_polarity > -0.03333334: 1 (5)
##     max_negative_polarity <= -0.03333334:
##     :...abs_title_sentiment_polarity > 0.3046315: 1 (16/6)
##         abs_title_sentiment_polarity <= 0.3046315:
##         :...max_positive_polarity <= 0.95: 0 (64/12)
##             max_positive_polarity > 0.95:
##             :...global_sentiment_polarity <= 0.1931566: 1 (14/3)
##                 global_sentiment_polarity > 0.1931566: 0 (15/2)
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##      62 11242(37.8%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##    9671  5359    (a): class 0
##    5883  8820    (b): class 1
##
##
```

```
##  Attribute usage:
##
##  100.00% LDA_02
##   86.92% min_positive_polarity
##   83.10% LDA_01
##   50.56% LDA_00
##   48.09% global_subjectivity
##   30.57% title_subjectivity
##   30.11% max_negative_polarity
##   18.63% title_sentiment_polarity
##   18.32% avg_positive_polarity
##   15.26% LDA_03
##   14.70% LDA_04
##   13.43% max_positive_polarity
##   12.82% abs_title_sentiment_polarity
##   11.08% global_rate_positive_words
##    7.57% rate_positive_words
##    5.06% min_negative_polarity
##    4.02% abs_title_subjectivity
##    3.47% rate_negative_words
##    0.29% global_sentiment_polarity
##
##
## Time: 1.8 secs
```

```
#Error :37.8%

#Words
#onlinepop_tc1 <- C5.0.default(x = onlinepop_train[,c(1:5,10)], y =
onlinepop_train$shares)
#summary(onlinepop_tc1)
#Too much processing time - crashes

#Links
onlinepop_tc2 <-C5.0.default(x = onlinepop_train[,c(6:7,27:29)], y =
onlinepop_train$shares)
summary(onlinepop_tc2)
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, c(6:7, 27:29)], y
##  = onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:57:52 2017
## -------------------------------
```

```
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (6 attributes) from undefined.data
##
## Decision tree:
##
## self_reference_min_shares <= 1600:
## :...self_reference_max_shares <= 12100:
## :   :...self_reference_min_shares <= 40:
## :   :   :...num_self_hrefs > 9: 1 (27/5)
## :   :   :   num_self_hrefs <= 9:
## :   :   :   :...num_hrefs > 4:
## :   :   :       :...num_hrefs <= 31: 0 (3308/1431)
## :   :   :       :   num_hrefs > 31: 1 (64/25)
## :   :   :       num_hrefs <= 4:
## :   :   :       :...num_hrefs > 3: 1 (304/128)
## :   :   :           num_hrefs <= 3:
## :   :   :           :...num_self_hrefs <= 0: 1 (1364/628)
## :   :   :               num_self_hrefs > 0: 0 (355/161)
## :   :   self_reference_min_shares > 40:
## :   :   :...num_hrefs > 14:
## :   :       :...self_reference_min_shares > 925: 1 (1311/596)
## :   :       :   self_reference_min_shares <= 925:
## :   :       :   :...num_hrefs <= 55: 0 (1105/461)
## :   :       :       num_hrefs > 55: 1 (44/14)
## :   :       num_hrefs <= 14:
## :   :       :...self_reference_min_shares <= 946: 0 (3937/1163)
## :   :           self_reference_min_shares > 946:
## :   :           :...num_self_hrefs <= 6: 0 (4335/1726)
## :   :               num_self_hrefs > 6:
## :   :               :...self_reference_max_shares <= 2100: 0
(57/16)
## :   :                   self_reference_max_shares > 2100: 1
(270/108)
## :   self_reference_max_shares > 12100:
## :   :...self_reference_max_shares > 298400: 1 (32/4)
## :       self_reference_max_shares <= 298400:
## :       :...num_hrefs <= 7: 0 (527/236)
## :           num_hrefs > 7:
## :           :...self_reference_min_shares > 809: 1 (858/329)
## :               self_reference_min_shares <= 809:
## :               :...num_self_hrefs > 31: 1 (29/7)
## :                   num_self_hrefs <= 31:
## :                   :...self_reference_max_shares <= 15200: 1
(86/29)
```

```
## :                       self_reference_max_shares > 15200: 0
(455/201)
## self_reference_min_shares > 1600:
## :...num_hrefs > 10: 1 (3862/1196)
##     num_hrefs <= 10:
##     :...self_reference_max_shares > 10900: 1 (2101/764)
##         self_reference_max_shares <= 10900:
##         :...self_reference_max_shares <= 2000: 0 (674/305)
##             self_reference_max_shares > 2000:
##             :...num_hrefs <= 5: 1 (2664/1286)
##                 num_hrefs > 5:
##                 :...num_self_hrefs > 2: 1 (1208/506)
##                     num_self_hrefs <= 2:
##                     :...self_reference_max_shares > 6700: 1
(158/55)
##                         self_reference_max_shares <= 6700:
##                         :...self_reference_avg_sharess <= 4175: 1
(412/183)
##                             self_reference_avg_sharess > 4175: 0
(186/72)
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##      27 11635(39.1%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##    9167  5863    (a): class 0
##    5772  8931    (b): class 1
##
##
##  Attribute usage:
##
##  100.00% self_reference_min_shares
##   99.80% num_hrefs
##   87.01% self_reference_max_shares
##   42.44% num_self_hrefs
##    2.01% self_reference_avg_sharess
##
```

```
##
## Time: 0.2 secs

#Error :39.1%

#Digital Media
onlinepop_tc3 <-C5.0.default(x = onlinepop_train[,8:9], y =
onlinepop_train$shares)
summary(onlinepop_tc3)

##
## Call:
## C5.0.default(x = onlinepop_train[, 8:9], y =
onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:57:52 2017
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (3 attributes) from undefined.data
##
## Decision tree:
##
## num_imgs > 5: 1 (6925/2846)
## num_imgs <= 5:
## :...num_videos <= 0: 0 (14262/6391)
##     num_videos > 0:
##     :...num_videos > 22: 0 (251/86)
##         num_videos <= 22:
##         :...num_videos > 4: 1 (1230/556)
##             num_videos <= 4:
##             :...num_imgs <= 2: 0 (6416/3120)
##                 num_imgs > 2: 1 (649/296)
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##       6 13295(44.7%)   <<
##
##
```

```
##     (a)   (b)    <-classified as
##    ----  ----
##   11332  3698    (a): class 0
##    9597  5106    (b): class 1
##
##
##  Attribute usage:
##
##  100.00% num_imgs
##   76.71% num_videos
##
##
## Time: 0.0 secs
```

```
#Error :44.7%

#Time
onlinepop_tc4 <-C5.0.default(x = onlinepop_train[,30:37], y =
onlinepop_train$shares)
summary(onlinepop_tc4)
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, 30:37], y =
onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:57:53 2017
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (9 attributes) from undefined.data
##
## Decision tree:
##
## weekday_is_saturday > 0: 1 (1836/537)
## weekday_is_saturday <= 0:
## :...weekday_is_sunday <= 0: 0 (25860/12102)
##     weekday_is_sunday > 0: 1 (2037/735)
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
```

```
##    Size      Errors
##
##       3 13374(45.0%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##   13758  1272    (a): class 0
##   12102  2601    (b): class 1
##
##
##  Attribute usage:
##
##  100.00% weekday_is_saturday
##   93.83% weekday_is_sunday
##
##
## Time: 0.1 secs
```

```
#Error :45%

#Keywords
onlinepop_tc5 <-C5.0.default(x = onlinepop_train[,11:26], y =
onlinepop_train$shares)
summary(onlinepop_tc5)
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, 11:26], y =
onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:57:55 2017
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (17 attributes) from undefined.data
##
## Decision tree:
##
## data_channel_is_socmed > 0:
## :...kw_max_avg <= 3697.155: 0 (112/37)
## :   kw_max_avg > 3697.155:
## :   :...kw_min_max <= 9700: 1 (1232/337)
## :       kw_min_max > 9700:
```

```
## :       :...kw_min_avg <= 3507.763: 1 (359/54)
## :           kw_min_avg > 3507.763:
## :           :...kw_avg_max <= 378500: 1 (12)
## :               kw_avg_max > 378500: 0 (27/10)
## data_channel_is_socmed <= 0:
## :...kw_avg_avg > 2892.734:
##     :...data_channel_is_entertainment > 0:
##     :   :...kw_avg_avg > 4020.608:
##     :   :   :...kw_max_max > 690400: 1 (676/304)
##     :   :   :   kw_max_max <= 690400:
##     :   :   :   :...kw_max_avg <= 6497.588: 1 (5)
##     :   :   :       kw_max_avg > 6497.588:
##     :   :   :       :...num_keywords <= 8: 0 (43/14)
##     :   :   :           num_keywords > 8: 1 (17/6)
##     :   :   kw_avg_avg <= 4020.608:
##     :   :   :...kw_max_max > 690400:
##     :   :       :...kw_min_min > 0: 0 (345/124)
##     :   :       :   kw_min_min <= 0:
##     :   :       :   :...num_keywords <= 8: 0 (1226/423)
##     :   :       :       num_keywords > 8: 1 (305/147)
##     :   :       kw_max_max <= 690400:
##     :   :       :...kw_min_avg <= 149:
##     :   :           :...kw_max_max <= 73100: 0 (13/5)
##     :   :           :   kw_max_max > 73100: 1 (102/37)
##     :   :           kw_min_avg > 149:
##     :   :           :...kw_min_avg <= 2297.536: 0 (125/43)
##     :   :               kw_min_avg > 2297.536:
##     :   :               :...kw_min_min <= 98: 1 (57/20)
##     :   :                   kw_min_min > 98:
##     :   :                   :...kw_min_avg <= 2484.396: 1 (8)
##     :   :                       kw_min_avg > 2484.396: 0 (7/1)
##     :   data_channel_is_entertainment <= 0:
##     :   :...data_channel_is_world > 0:
##     :       :...num_keywords > 9: 1 (314/133)
##     :       :   num_keywords <= 9:
##     :       :   :...kw_avg_max <= 362400: 0 (807/352)
##     :       :       kw_avg_max > 362400: 1 (164/64)
##     :       data_channel_is_world <= 0:
##     :       :...kw_max_max <= 57600:
##     :           :...data_channel_is_lifestyle > 0: 1 (3)
##     :           :   data_channel_is_lifestyle <= 0:
##     :           :   :...kw_avg_min <= 360.8571: 1 (5)
##     :           :       kw_avg_min > 360.8571:
##     :           :       :...kw_max_avg <= 10177.06: 0 (28/4)
##     :           :           kw_max_avg > 10177.06: 1 (7/2)
##     :           kw_max_max > 57600:
```

```
##     :           :...data_channel_is_tech > 0: 1 (2023/605)
##     :               data_channel_is_tech <= 0:
##     :               :...kw_max_min <= 245:
##     :                   :...kw_min_min <= 0: 1 (538/253)
##     :                   :   kw_min_min > 0: 0 (151/59)
##     :                   kw_max_min > 245:
##     :                   :...kw_avg_max <= 398175: 1 (4383/1509)
##     :                       kw_avg_max > 398175:
##     :                       :...kw_min_min <= 0:
##     :                           :...kw_avg_avg > 3342.877: 1
(1375/499)
##     :                           :   kw_avg_avg <= 3342.877:
##     :                           :   :...kw_avg_avg <= 2926.491: 1
(8/1)
##     :                           :       kw_avg_avg > 2926.491: 0
(163/72)
##     :                           kw_min_min > 0:
##     :                           :...num_keywords > 7: 0 (139/56)
##     :                               num_keywords <= 7:
##     :                               :...data_channel_is_bus > 0:
##     :                                   :...kw_avg_avg <= 3108.288:
0 (32/7)
##     :                                   :   kw_avg_avg > 3108.288:
1 (119/51)
##     :                                   data_channel_is_bus <= 0:
##     :                                   :...kw_avg_max > 652900: 0
(19/5)
##     :                                       kw_avg_max <= 652900:
##     :                                       :...kw_max_avg <=
4698.701: 0 (4)
##     :                                           kw_max_avg >
4698.701: 1 (271/113)
##     kw_avg_avg <= 2892.734:
##     :...data_channel_is_tech > 0:
##         :...kw_min_min > 98:
##         :   :...kw_max_max > 28000: 1 (570/203)
##         :   :   kw_max_max <= 28000:
##         :   :   :...kw_avg_min <= 262.6667: 0 (8)
##         :   :       kw_avg_min > 262.6667:
##         :   :       :...kw_max_max <= 18200: 0 (12/4)
##         :   :           kw_max_max > 18200: 1 (22/8)
##         :   kw_min_min <= 98:
##         :   :...kw_min_min <= -1:
##         :       :...kw_avg_avg > 2542.014: 1 (742/293)
##         :       :   kw_avg_avg <= 2542.014:
##         :       :   :...kw_min_max <= 1000: 1 (614/289)
```

```
##          :       :       kw_min_max > 1000:
##          :       :       :...num_keywords > 7: 0 (73/20)
##          :       :           num_keywords <= 7:
##          :       :           :...kw_min_avg <= 909.5: 1 (3)
##          :       :               kw_min_avg > 909.5:
##          :       :               :...kw_max_avg <= 3465.509: 1
(12/4)
##          :       :                   kw_max_avg > 3465.509: 0 (28/8)
##          :       kw_min_min > -1:
##          :       :...kw_min_avg <= 1313.857: 0 (1077/471)
##          :           kw_min_avg > 1313.857:
##          :           :...kw_max_max <= 617900: 1 (18/4)
##          :               kw_max_max > 617900:
##          :               :...num_keywords <= 5: 0 (35/13)
##          :                   num_keywords > 5:
##          :                   :...kw_max_max > 690400: 1 (194/81)
##          :                       kw_max_max <= 690400:
##          :                       :...kw_max_avg <= 3346.833: 0
(18/3)
##          :                           kw_max_avg > 3346.833:
##          :                           :...kw_avg_min <= 291: 0 (4)
##          :                               kw_avg_min > 291: 1 (43/12)
##          data_channel_is_tech <= 0:
##          :...kw_max_max <= 617900:
##              :...data_channel_is_lifestyle > 0: 1 (260/119)
##              :   data_channel_is_lifestyle <= 0:
##              :   :...kw_max_avg <= 2815.5: 0 (453/174)
##              :       kw_max_avg > 2815.5:
##              :       :...kw_min_max <= 665: 1 (923/435)
##              :           kw_min_max > 665:
##              :           :...kw_min_avg <= 1518.273: 0 (224/75)
##              :               kw_min_avg > 1518.273:
##              :               :...data_channel_is_entertainment <= 0:
##              :                   :...kw_min_max <= 23300: 1 (146/57)
##              :                   :   kw_min_max > 23300: 0 (19/4)
##              :                   data_channel_is_entertainment > 0:
##              :                   :...num_keywords <= 9: 0 (53/19)
##              :                       num_keywords > 9: 1 (3)
##              kw_max_max > 617900:
##              :...kw_max_avg <= 3645.058:
##                  :...data_channel_is_lifestyle <= 0: 0 (5144/1333)
##                  :   data_channel_is_lifestyle > 0:
##                  :   :...kw_max_max > 690400: 0 (27/6)
##                  :       kw_max_max <= 690400:
##                  :       :...kw_avg_avg <= 1898.883: 1 (5)
##                  :           kw_avg_avg > 1898.883:
```

```
##                  :           :...kw_max_avg <= 3409.864: 0 (9/1)
##                  :               kw_max_avg > 3409.864: 1 (18/6)
##                 kw_max_avg > 3645.058:
##                 :...kw_min_max > 383: 0 (1154/345)
##                     kw_min_max <= 383:
##                     :...data_channel_is_bus > 0:
##                         :...kw_avg_avg > 2372.579: 1 (364/153)
##                         :   kw_avg_avg <= 2372.579:
##                         :   :...kw_avg_max <= 341988.9: 0 (151/54)
##                         :       kw_avg_max > 341988.9: 1 (23/7)
##                         data_channel_is_bus <= 0:
##                         :...data_channel_is_entertainment <= 0: 0
(1253/544)
##                             data_channel_is_entertainment > 0:
##                             :...kw_max_max > 690400: 0 (695/233)
##                                 kw_max_max <= 690400:
##                                 :...num_keywords > 7: 0 (58/19)
##                                     num_keywords <= 7:
##                                     :...kw_avg_min <= 161.5714: 0
(7)
##                                         kw_avg_min > 161.5714:
##                                         :...kw_avg_min <= 220.918:
1 (10/2)
##                                             kw_avg_min > 220.918:
##                                             :...kw_max_avg > 4780:
0 (7/1)
##                                                 kw_max_avg <= 4780:
##                                                 :...kw_avg_min <=
284.6: 0 (10/3)
##                                                     kw_avg_min >
284.6: 1 (20/4)
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##      80 10354(34.8%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##    9218  5812    (a): class 0
##    4542 10161    (b): class 1
```

```
##
##
##  Attribute usage:
##
##  100.00% data_channel_is_socmed
##   94.14% kw_avg_avg
##   81.25% kw_max_max
##   79.82% data_channel_is_tech
##   53.02% data_channel_is_entertainment
##   43.72% kw_max_avg
##   35.49% data_channel_is_world
##   27.71% kw_min_min
##   25.89% kw_avg_max
##   25.16% kw_min_max
##   24.64% data_channel_is_lifestyle
##   24.22% kw_max_min
##   13.58% num_keywords
##   10.23% data_channel_is_bus
##    8.70% kw_min_avg
##    0.62% kw_avg_min
##
##
## Time: 0.8 secs

#Error :34.8%
```

I choose the tree model using the 'Keywords' category as it has the least misclassification error. Based on the evaluation above for this 'Keywords' model output the algorithm properly classified 9218 records as (a), 10161 records as (b) and the remaining 10354 (5812+4542) records create a 34.8% (misclassification) error in training.

Evaluating Model Performance

Going forward with 'Keywords' category model, we still need to use our test set to evaluate/validate the model's overall performance. To do this we'll use the predict() command as follows:

```
onlinepop_pred <-predict(onlinepop_tc5, onlinepop_test)
```

Last, we'll use the gmodels package to create a confusion table looking at the predicted and actual values using the training and test sets for our credit data.

```
library(gmodels)
```

```
## Warning: package 'gmodels' was built under R version 3.3.3
```

```
CrossTable(onlinepop_test$shares, onlinepop_pred, prop.chisq =FALSE,
prop.c =FALSE, prop.r =FALSE, dnn =c('actual popular', 'predicted
popular'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  9911
##
##
##                | predicted popular
## actual popular |         0 |         1 | Row Total |
## ---------------|-----------|-----------|-----------|
##              0 |      3024 |      2028 |      5052 |
##                |     0.305 |     0.205 |           |
## ---------------|-----------|-----------|-----------|
##              1 |      1577 |      3282 |      4859 |
##                |     0.159 |     0.331 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |      4601 |      5310 |      9911 |
## ---------------|-----------|-----------|-----------|
##
##
```

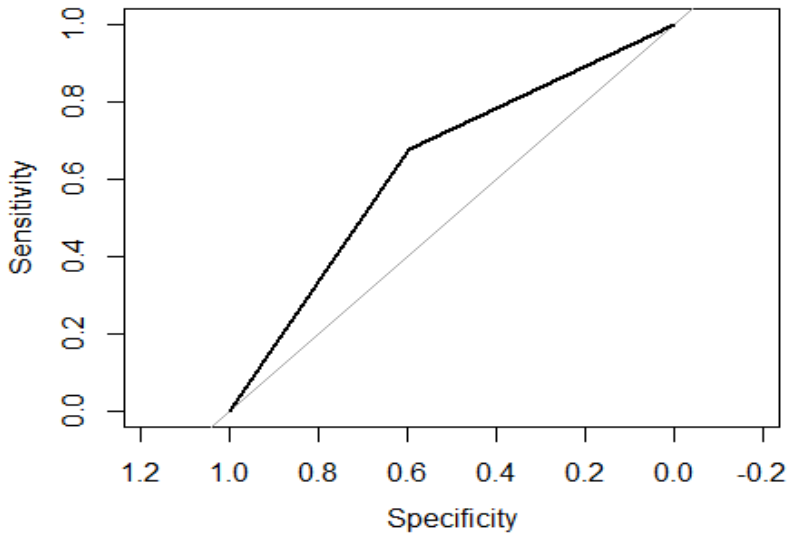
```
percent_accuracy_tc <-(3024+3282)/9911
percent_accuracy_tc
```

```
## [1] 0.6362627
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

```
## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.3.3

precision_tc <-posPredValue(onlinepop_pred, onlinepop_test$shares,
positive="1")
precision_tc

## [1] 0.6180791

recall_tc <-sensitivity(onlinepop_pred,
onlinepop_test$shares,positive="1")
recall_tc

## [1] 0.6754476

F1_tc <-(2 *precision_tc *recall_tc) /(precision_tc +recall_tc)
F1_tc

## [1] 0.6454912

library(pROC)

## Warning: package 'pROC' was built under R version 3.3.3

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following object is masked from 'package:gmodels':
##
##     ci

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

onlinepop_pred1 <-as.numeric(onlinepop_pred)
roc_obj_tc <-roc(onlinepop_test$shares,onlinepop_pred1)
library(pROC)
roc_obj_tc$auc

## Area under the curve: 0.637

plot(roc_obj_tc)
```



The table indicates that for the 9911 records in our test set 2028 non-popular articles were misclassified, i.e. false negatives or a Type II error, and 1577 actual popular articles were misclassified as not popular, i.e. false positives or a Type I error. The accuracy of the model is 63.6%.

Feature selection and Model building

The accuracy is better than random classification (50%) but perhaps could be improved if we could use features from all categories. In He Ren's paper, he mentioned that when PCA (Principal Component Analysis) is used for dimensionality reduction, it actually makes the model perform worse. This is because the original feature set is well-designed and correlated information between features is limited.

He Ren used Fisher's criterion to aid in feature selection. Based on this approach, he selected 20 features that have the highest Fisher scores in feature selection. I built the tree classification model using the 20 features selected in He Ren's paper.

```
onlinepop_tc6 <-C5.0.default(x = onlinepop_train[,c(6, 13, 15, 16, 17,
18, 20, 23, 26, 35, 36, 37, 38, 40, 41, 42, 43, 44, 48, 55)], y =
onlinepop_train$shares)
summary(onlinepop_tc6)
```

```
##
## Call:
## C5.0.default(x = onlinepop_train[, c(6, 13, 15, 16, 17, 18, 20, 23,
##  26, 35, 36, 37, 38, 40, 41, 42, 43, 44, 48, 55)], y
##  = onlinepop_train$shares)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Dec 23 14:58:01 2017
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 29733 cases (21 attributes) from undefined.data
##
## Decision tree:
##
## data_channel_is_socmed > 0:
## :...kw_avg_max > 638150: 0 (33/7)
## :   kw_avg_max <= 638150:
## :   :...LDA_00 > 0.1205231: 1 (1281/267)
## :       LDA_00 <= 0.1205231:
## :       :...weekday_is_sunday <= 0:
## :           :...LDA_00 <= 0.03043473: 1 (182/61)
## :           :   LDA_00 > 0.03043473: 0 (221/96)
## :           weekday_is_sunday > 0:
## :           :...LDA_00 <= 0.05000034: 1 (22/1)
## :               LDA_00 > 0.05000034: 0 (3)
## data_channel_is_socmed <= 0:
## :...weekday_is_saturday > 0:
##     :...LDA_02 > 0.4619689:
##     :   :...data_channel_is_entertainment <= 0: 1 (345/160)
##     :   :   data_channel_is_entertainment > 0:
##     :   :   :...title_subjectivity <= 0.2296296: 0 (15/2)
##     :   :       title_subjectivity > 0.2296296:
##     :   :       :...LDA_03 <= 0.04081338: 1 (4)
##     :   :           LDA_03 > 0.04081338: 0 (2)
##     :   LDA_02 <= 0.4619689:
##     :   :...data_channel_is_entertainment > 0: 1 (257/107)
##     :       data_channel_is_entertainment <= 0:
##     :       :...LDA_00 > 0.3672832: 1 (220/22)
##     :           LDA_00 <= 0.3672832:
```

```
##     :           :...kw_min_min > 4:
##     :               :...LDA_02 <= 0.1700026: 1 (86/5)
##     :               :   LDA_02 > 0.1700026:
##     :               :   :...LDA_04 > 0.5211969: 0 (4)
##     :               :       LDA_04 <= 0.5211969:
##     :               :       :...num_hrefs <= 2: 0 (2)
##     :               :           num_hrefs > 2: 1 (11)
##     :               kw_min_min <= 4:
##     :               :...kw_min_min <= 0: 1 (444/89)
##     :                   kw_min_min > 0:
##     :                   :...data_channel_is_tech > 0: 1 (143/41)
##     :                       data_channel_is_tech <= 0:
##     :                       :...title_subjectivity <= 0.03333334:
##     :                           :...global_subjectivity <=
0.5432823: 1 (45/7)
##     :                           :   global_subjectivity >
0.5432823:
##     :                           :   :...LDA_02 <= 0.02857196: 0
(9/1)
##     :                           :       LDA_02 > 0.02857196: 1
(7/1)
##     :                           title_subjectivity > 0.03333334:
##     :                           :...kw_avg_min <= 173.1475:
##     :                               :...num_hrefs <= 27: 0 (33/8)
##     :                               :   num_hrefs > 27: 1 (2)
##     :                               kw_avg_min > 173.1475:
##     :                               :...LDA_04 <= 0.02222281: 0 (5)
##     :                                   LDA_04 > 0.02222281: 1
(66/23)
##     weekday_is_saturday <= 0:
##     :...kw_avg_avg > 2892.734:
##         :...data_channel_is_entertainment > 0:
##         :   :...weekday_is_sunday > 0: 1 (262/98)
##         :   :   weekday_is_sunday <= 0:
##         :   :   :...kw_avg_avg > 4054.305:
##         :   :       :...kw_min_min <= 0: 1 (514/245)
##         :   :       :   kw_min_min > 0:
##         :   :       :   :...kw_min_min <= 98: 1 (88/42)
##         :   :       :       kw_min_min > 98:
##         :   :       :       :...title_subjectivity <= 0.1: 1 (3)
##         :   :       :           title_subjectivity > 0.1:
##         :   :       :           :...kw_avg_max <= 46387.5: 1 (2)
##         :   :       :               kw_avg_max > 46387.5: 0 (10)
##         :   :       kw_avg_avg <= 4054.305:
##         :   :       :...kw_avg_max > 179150:
##         :   :           :...LDA_02 > 0.03666735:
```

```
##         :   :           :   :...kw_min_min <= 0: 0 (586/164)
##         :   :           :   :   kw_min_min > 0:
##         :   :           :   :   :...title_subjectivity > 0.2166667:
0 (73/14)
##         :   :           :   :       title_subjectivity <=
0.2166667:
##         :   :           :   :       :...LDA_04 <= 0.02857219: 0
(10)
##         :   :           :   :           LDA_04 > 0.02857219: 1
(56/22)
##         :   :           :   LDA_02 <= 0.03666735:
##         :   :           :   :...num_hrefs <= 6:
##         :   :           :       :...kw_min_min <= 0: 0 (357/109)
##         :   :           :       :   kw_min_min > 0:
##         :   :           :       :   :...num_hrefs <= 2: 1 (9/1)
##         :   :           :       :       num_hrefs > 2: 0 (69/22)
##         :   :           :       num_hrefs > 6:
##         :   :           :       :...kw_min_min <= 0: 0 (313/143)
##         :   :           :           kw_min_min > 0:
##         :   :           :           :...rate_negative_words <=
0.3777778:
##         :   :           :               :...LDA_00 <= 0.02222433: 0
(24/3)
##         :   :           :               :   LDA_00 > 0.02222433: 1
(66/26)
##         :   :           :               rate_negative_words >
0.3777778:
##         :   :           :               :...title_subjectivity >
0.4772727: 0 (14)
##         :   :           :                   title_subjectivity <=
0.4772727: [S1]
##         :   :          kw_avg_max <= 179150:
##         :   :          :...title_subjectivity > 0.5361111: 1
(52/17)
##         :   :              title_subjectivity <= 0.5361111:
##         :   :              :...kw_min_min > 98: 1 (59/28)
##         :   :                  kw_min_min <= 98:
##         :   :                  :...kw_min_min > 0: 0 (118/50)
##         :   :                      kw_min_min <= 0:
##         :   :                      :...kw_avg_max <= 148271.4: 1
(7)
##         :   :                          kw_avg_max > 148271.4:
##         :   :                          :...rate_negative_words <=
0.1126761: 1 (5)
##         :   :                              rate_negative_words >
0.1126761:
```

```
##         :   :                                  :...LDA_00 > 0.1784056:
0 (4)
##         :   :                                      LDA_00 <=
0.1784056:
##         :   :                                      :...LDA_03 <=
0.02406099: 1 (4)
##         :   :                                          LDA_03 >
0.02406099:
##         :   :                                          :...LDA_03 <=
0.2417656: 0 (11)
##         :   :                                              LDA_03 >
0.2417656: [S2]
##         :   data_channel_is_entertainment <= 0:
##         :   :...LDA_02 > 0.3290989:
##         :       :...weekday_is_sunday > 0: 1 (94/39)
##         :       :   weekday_is_sunday <= 0:
##         :       :   :...LDA_02 <= 0.575861:
##         :       :       :...kw_min_min > 0: 1 (158/68)
##         :       :       :   kw_min_min <= 0:
##         :       :       :   :...data_channel_is_tech <= 0:
##         :       :       :       :...data_channel_is_world <= 0: 0
(119/57)
##         :       :       :       :   data_channel_is_world > 0: 1
(308/152)
##         :       :       :       data_channel_is_tech > 0:
##         :       :       :       :...global_sentiment_polarity <=
0.1996149: 1 (69/27)
##         :       :       :           global_sentiment_polarity >
0.1996149: 0 (9)
##         :       :       LDA_02 > 0.575861:
##         :       :       :...num_hrefs <= 14: 0 (447/145)
##         :       :           num_hrefs > 14:
##         :       :           :...kw_avg_max > 350966.7: 1 (8)
##         :       :               kw_avg_max <= 350966.7:
##         :       :               :...num_hrefs > 44: 1 (6)
##         :       :                   num_hrefs <= 44:
##         :       :                   :...kw_min_min > 0: 0 (22/7)
##         :       :                       kw_min_min <= 0:
##         :       :                       :...kw_avg_max <= 204088.9:
1 (19/5)
##         :       :                           kw_avg_max > 204088.9:
##         :       :                           :...title_subjectivity
> 0.1666667: [S3]
##         :       :                               title_subjectivity
<= 0.1666667:
##         :       :                               :...kw_avg_avg >
```

```
3112.945: 0 (15)
##         :       :                                    kw_avg_avg <=
3112.945:
##         :       :                                    :...LDA_04 <=
0.02783793: 0 (3)
##         :       :                                        LDA_04 >
0.02783793: 1 (6/1)
##         :       LDA_02 <= 0.3290989:
##         :       :...weekday_is_sunday > 0:
##         :           :...global_sentiment_polarity > 0.2724263:
##         :           :   :...title_subjectivity > 0.8666667: 1
(13/1)
##         :           :   :   title_subjectivity <= 0.8666667:
##         :           :   :   :...kw_avg_min > 719.5: 1 (4)
##         :           :   :       kw_avg_min <= 719.5:
##         :           :   :       :...kw_min_min <= 0:
##         :           :   :           :...num_hrefs <= 39: 1 (24/9)
##         :           :   :           :   num_hrefs > 39: 0 (4)
##         :           :   :           kw_min_min > 0:
##         :           :   :           :...kw_avg_avg <= 4613.954: 0
(34/5)
##         :           :   :               kw_avg_avg > 4613.954: 1
(2)
##         :           :   global_sentiment_polarity <= 0.2724263:
##         :           :   :...LDA_04 > 0.3431782: 1 (183/22)
##         :           :       LDA_04 <= 0.3431782:
##         :           :       :...LDA_00 > 0.8036497: 1 (65/6)
##         :           :           LDA_00 <= 0.8036497:
##         :           :           :...kw_min_min <= 0: 1 (240/61)
##         :           :               kw_min_min > 0:
##         :           :               :...kw_min_min <= 98:
##         :           :                   :...LDA_03 <= 0.1158141: 0
(46/16)
##         :           :                   :   LDA_03 > 0.1158141: 1
(94/28)
##         :           :                   kw_min_min > 98: [S4]
##         :           weekday_is_sunday <= 0:
##         :           :...num_hrefs > 12:
##         :               :...LDA_00 > 0.8999569: 1 (47)
##         :               :   LDA_00 <= 0.8999569:
##         :               :   :...kw_avg_avg > 4921.325:
##         :               :       :...LDA_04 > 0.02222258:
##         :               :       :   :...kw_min_min <= 0: 1
(486/116)
##         :               :       :   :   kw_min_min > 0:
##         :               :       :   :   :...kw_avg_min <= 145.7778:
```

```
0 (12/3)
##         :                :       :   :       kw_avg_min > 145.7778:
1 (157/38)
##         :                :       :   LDA_04 <= 0.02222258:
##         :                :       :   :...kw_avg_max > 291300: 0
(72/30)
##         :                :       :       kw_avg_max <= 291300:
##         :                :       :       :...LDA_02 > 0.02000355: 1
(24/1)
##         :                :       :           LDA_02 <= 0.02000355:
##         :                :       :           :...kw_avg_max <=
239242.9: 1 (6)
##         :                :       :               kw_avg_max >
239242.9:
##         :                :       :               :...kw_avg_avg <=
5057.702: 1 (2)
##         :                :       :                   kw_avg_avg >
5057.702: 0 (8)
##         :                :       kw_avg_avg <= 4921.325:
##         :                :       :...data_channel_is_world <= 0: 1
(1899/657)
##         :                :           data_channel_is_world > 0:
##         :                :           :...LDA_03 > 0.7585189: 0 (4)
##         :                :               LDA_03 <= 0.7585189:
##         :                :               :...kw_min_min > 0: 1 (17)
##         :                :                   kw_min_min <= 0: [S5]
##         :                num_hrefs <= 12:
##         :                :...data_channel_is_tech > 0: 1 (1227/422)
##         :                    data_channel_is_tech <= 0:
##         :                    :...LDA_00 <= 0.6799576:
##         :                        :...kw_avg_avg > 4704.033: 1
(825/297)
##         :                        :   kw_avg_avg <= 4704.033:
##         :                        :   :...data_channel_is_world > 0:
##         :                        :       :...kw_avg_min <= 53.65572:
0 (6)
##         :                        :       :   kw_avg_min > 53.65572:
1 (119/42)
##         :                        :       data_channel_is_world <= 0:
##         :                        :       :...kw_min_min <= 45:
##         :                        :           :...kw_min_min <= 0: 1
(1095/541)
##         :                        :           :   kw_min_min > 0: 0
(699/340)
##         :                        :           kw_min_min > 45:
##         :                        :           :...LDA_03 > 0.4400999:
```

```
0 (101/46)
##         :                       :               LDA_03 <=
0.4400999:
##         :                       :               :...kw_avg_min <=
1469.429: 1 (76/17)
##         :                       :                   kw_avg_min >
1469.429: 0 (5/1)
##         :                       LDA_00 > 0.6799576:
##         :                       :...global_subjectivity <=
0.3845238:
##         :                           :...num_hrefs <= 2: 1 (19/5)
##         :                           :   num_hrefs > 2: 0 (145/57)
##         :                           global_subjectivity >
0.3845238:
##         :                           :...title_subjectivity >
0.7133333: 1 (71/11)
##         :                               title_subjectivity <=
0.7133333:
##         :                               :...LDA_04 <= 0.0401671: 1
(359/103)
##         :                                   LDA_04 > 0.0401671:
[S6]
##         kw_avg_avg <= 2892.734:
##         :...data_channel_is_tech > 0:
##             :...weekday_is_sunday > 0: 1 (171/50)
##             :   weekday_is_sunday <= 0:
##             :   :...kw_min_min > 98: 1 (534/214)
##             :       kw_min_min <= 98:
##             :       :...num_hrefs > 13:
##             :           :...kw_avg_avg <= 2324.05: 0 (132/62)
##             :           :   kw_avg_avg > 2324.05: 1 (246/85)
##             :           num_hrefs <= 13:
##             :           :...kw_min_min <= 0: 1 (1131/553)
##             :               kw_min_min > 0:
##             :               :...title_subjectivity > 0.95: 1
(42/16)
##             :                   title_subjectivity <= 0.95:
##             :                   :...rate_negative_words <=
0.05555556: 0 (81/17)
##             :                       rate_negative_words >
0.05555556:
##             :                       :...kw_avg_min <= 335.4: 0
(622/232)
##             :                           kw_avg_min > 335.4:
##             :                           :...num_hrefs <= 5: 0
(106/46)
```

```
##             :                                num_hrefs > 5:
##             :                                :...LDA_00 <=
0.02002179: 0 (18/3)
##             :                                    LDA_00 >
0.02002179: 1 (167/66)
##             data_channel_is_tech <= 0:
##             :...weekday_is_sunday > 0:
##                 :...num_hrefs > 13:
##                 :   :...data_channel_is_entertainment <= 0: 1
(139/51)
##                 :   :   data_channel_is_entertainment > 0:
##                 :   :   :...kw_min_min > 98: 0 (5)
##                 :   :       kw_min_min <= 98:
##                 :   :       :...kw_avg_min <= 103.8: 0 (7/2)
##                 :   :           kw_avg_min > 103.8: 1 (25/6)
##                 :   num_hrefs <= 13:
##                 :   :...LDA_02 > 0.8182245: 0 (97/21)
##                 :       LDA_02 <= 0.8182245:
##                 :       :...kw_min_min > 98: 1 (87/30)
##                 :           kw_min_min <= 98:
##                 :           :...kw_min_min > 0:
##                 :               :...data_channel_is_world <= 0:
##                 :               :   :...LDA_04 <= 0.4411417: 1
(56/22)
##                 :               :   :   LDA_04 > 0.4411417:
##                 :               :   :   :...num_hrefs <= 10: 0
(15/1)
##                 :               :   :       num_hrefs > 10: 1 (4/1)
##                 :               :   data_channel_is_world > 0:
##                 :               :   :...kw_avg_max > 331570: 0 (3)
##                 :               :       kw_avg_max <= 331570:
##                 :               :       :...kw_avg_min <= 461.625:
1 (17/2)
##                 :               :           kw_avg_min > 461.625: 0
(2)
##                 :               kw_min_min <= 0:
##                 :               :...data_channel_is_entertainment
<= 0:
##                 :                   :...data_channel_is_world <= 0:
##                 :                   :   :...LDA_02 > 0.2870134: 0
(3)
##                 :                   :   :   LDA_02 <= 0.2870134:
##                 :                   :   :   :...LDA_04 <=
0.3405596: 1 (28/9)
##                 :                   :   :       LDA_04 > 0.3405596:
0 (3)
```

```
##                  :                    :   data_channel_is_world > 0:
##                  :                    :   :...kw_avg_min > 158.8975:
1 (81/35)
##                  :                    :       kw_avg_min <= 158.8975:
##                  :                    :       :...LDA_03 <=
0.1084445: 0 (47/11)
##                  :                    :           LDA_03 > 0.1084445:
1 (4/1)
##                  :                   data_channel_is_entertainment >
0:
##                  :                   :...LDA_03 > 0.2054794: 0
(21/4)
##                  :                       LDA_03 <= 0.2054794:
##                  :                       :...LDA_03 > 0.1046986: 1
(2)
##                  :                           LDA_03 <= 0.1046986:
[S7]
##                 weekday_is_sunday <= 0:
##                 :...num_hrefs > 28:
##                     :...LDA_00 <= 0.5458954: 0 (325/146)
##                     :   LDA_00 > 0.5458954: 1 (40/9)
##                     num_hrefs <= 28:
##                     :...kw_min_min > 201:
##                         :...data_channel_is_world <= 0: 0
(1261/577)
##                         :   data_channel_is_world > 0:
##                         :   :...title_subjectivity > 0.725: 1
(44/16)
##                         :       title_subjectivity <= 0.725:
##                         :       :...LDA_02 <= 0.4754567: 1 (49/18)
##                         :           LDA_02 > 0.4754567: 0 (252/87)
##                         kw_min_min <= 201:
##                         :...kw_avg_avg <= 2198.56:
##                             :...global_subjectivity <= 0.02222222:
##                             :   :...data_channel_is_entertainment >
0: 0 (3)
##                             :   :   data_channel_is_entertainment
<= 0:
##                             :   :   :...kw_min_min <= 0: 1 (44/19)
##                             :   :       kw_min_min > 0: 0 (5/1)
##                             :   global_subjectivity > 0.02222222:
##                             :   :...LDA_04 <= 0.5320873: 0
(2287/485)
##                             :       LDA_04 > 0.5320873:
##                             :       :...global_subjectivity <=
0.5630686: 0 (145/50)
```

```
##                                 :           global_subjectivity >
0.5630686: 1 (9)
##                                kw_avg_avg > 2198.56:
##                                :...data_channel_is_entertainment > 0:
0 (1350/338)
##                                    data_channel_is_entertainment <= 0:
##                                    :...LDA_03 > 0.490064:
##                                        :...data_channel_is_world > 0:
0 (14/5)
##                                        :   data_channel_is_world <= 0:
[S8]
##                                        LDA_03 <= 0.490064:
##                                        :...LDA_02 > 0.6265502: 0
(1629/444)
##                                            LDA_02 <= 0.6265502:
##                                            :...global_subjectivity >
0.4713611:
##                                                :...LDA_00 <=
0.8666523:
##                                                :   :...kw_min_min <=
0: 0 (354/142)
##                                                :   :   kw_min_min > 0:
##                                                :   :   :...LDA_02 <=
0.04000054: 0 (128/45)
##                                                :   :       LDA_02 >
0.04000054: 1 (117/49)
##                                                :   LDA_00 > 0.8666523:
[S9]
##                                                global_subjectivity <=
0.4713611:
##                                                :...num_hrefs <= 14: 0
(1485/489)
##                                                    num_hrefs > 14:
##                                                    :...kw_avg_max >
231542.9: [S10]
##                                                        kw_avg_max <=
231542.9: [S11]
##
## SubTree [S1]
##
## rate_negative_words <= 0.53125: 0 (32/6)
## rate_negative_words > 0.53125: 1 (4)
##
## SubTree [S2]
##
## LDA_04 <= 0.02002305: 0 (6)
```

```
## LDA_04 > 0.02002305: 1 (12/4)
##
## SubTree [S3]
##
## title_subjectivity <= 0.3907408: 1 (5)
## title_subjectivity > 0.3907408: 0 (15/5)
##
## SubTree [S4]
##
## global_sentiment_polarity > 0.1490319: 1 (12)
## global_sentiment_polarity <= 0.1490319:
## :...LDA_00 <= 0.03340276: 0 (15/5)
##     LDA_00 > 0.03340276: 1 (6)
##
## SubTree [S5]
##
## global_subjectivity > 0.5155384: 1 (11)
## global_subjectivity <= 0.5155384:
## :...kw_avg_min > 247.625: 1 (10/1)
##     kw_avg_min <= 247.625:
##     :...kw_avg_avg <= 3759.998: 0 (15/3)
##         kw_avg_avg > 3759.998: 1 (6/1)
##
## SubTree [S6]
##
## rate_negative_words <= 0.2295082: 1 (97/35)
## rate_negative_words > 0.2295082:
## :...kw_min_min <= 0: 0 (79/33)
##     kw_min_min > 0:
##     :...num_hrefs > 7: 0 (8)
##         num_hrefs <= 7:
##         :...title_subjectivity > 0.08333334: 1 (5)
##             title_subjectivity <= 0.08333334:
##             :...kw_avg_avg <= 3116.684: 0 (5)
##                 kw_avg_avg > 3116.684:
##                 :...global_sentiment_polarity <= 0.06593164: 0 (2)
##                     global_sentiment_polarity > 0.06593164: 1 (7)
##
## SubTree [S7]
##
## global_sentiment_polarity <= 0.1701727: 0 (18/7)
## global_sentiment_polarity > 0.1701727: 1 (4)
##
## SubTree [S8]
##
## global_sentiment_polarity <= 0.04970364: 0 (9/1)
```

```
## global_sentiment_polarity > 0.04970364:
## :...rate_negative_words <= 0.07272727: 0 (3)
##     rate_negative_words > 0.07272727:
##     :...num_hrefs > 3: 1 (36/5)
##         num_hrefs <= 3:
##         :...LDA_04 <= 0.0288543: 0 (4)
##             LDA_04 > 0.0288543: 1 (2)
##
## SubTree [S9]
##
## rate_negative_words <= 0.1041667: 0 (4)
## rate_negative_words > 0.1041667:
## :...global_subjectivity > 0.5253867: 1 (17)
##     global_subjectivity <= 0.5253867:
##     :...kw_avg_min > 407: 1 (10)
##         kw_avg_min <= 407:
##         :...kw_avg_min <= 207: 1 (8)
##             kw_avg_min > 207: 0 (14/4)
##
## SubTree [S10]
##
## kw_min_min <= 0: 0 (73/20)
## kw_min_min > 0:
## :...LDA_04 <= 0.02364285: 0 (9)
##     LDA_04 > 0.02364285:
##     :...title_subjectivity <= 0.3666667: 1 (17/3)
##         title_subjectivity > 0.3666667:
##         :...kw_avg_min <= 139.2857: 1 (3)
##             kw_avg_min > 139.2857: 0 (14/2)
##
## SubTree [S11]
##
## data_channel_is_world <= 0:
## :...kw_min_min <= 0: 1 (10/1)
## :   kw_min_min > 0:
## :   :...num_hrefs <= 22: 1 (20/4)
## :       num_hrefs > 22: 0 (4)
## data_channel_is_world > 0:
## :...kw_min_min > 0: 0 (11/5)
##     kw_min_min <= 0:
##     :...num_hrefs > 19: 1 (6)
##         num_hrefs <= 19:
##         :...global_sentiment_polarity <= 0.08492064: 0 (9/1)
##             global_sentiment_polarity > 0.08492064:
##             :...global_sentiment_polarity <= 0.1775253: 1 (6)
##                 global_sentiment_polarity > 0.1775253: 0 (2)
```

```
##
##
## Evaluation on training data (29733 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##     195 9864(33.2%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##    9792  5238    (a): class 0
##    4626 10077    (b): class 1
##
##
##  Attribute usage:
##
##  100.00% data_channel_is_socmed
##   94.14% weekday_is_saturday
##   89.86% weekday_is_sunday
##   88.42% kw_avg_avg
##   74.94% num_hrefs
##   68.31% kw_min_min
##   67.43% data_channel_is_entertainment
##   65.63% data_channel_is_tech
##   59.96% LDA_02
##   37.31% LDA_00
##   21.84% data_channel_is_world
##   19.22% global_subjectivity
##   16.03% LDA_04
##   15.26% LDA_03
##   13.66% kw_avg_max
##    9.56% title_subjectivity
##    5.80% kw_avg_min
##    4.97% rate_negative_words
##    3.10% global_sentiment_polarity
##
##
## Time: 1.7 secs
```

Based on the evaluation above for this model output the algorithm properly classified 9792 records as (a), 10077 records as (b) and the remaining 9864 (5238+4626) records create a

33.2% (misclassification) error in training.Thus, we see that the error reduces from 34.8% to 33.2% in this model compared to the 'Keywords' category model.

Evaluating Model Performance (Top 20 Features)

```
onlinepop_pred_t20 <-predict(onlinepop_tc6, onlinepop_test)
```

Last, we'll use the gmodels package to create a confusion table looking at the predicted and actual values using the training and test sets for our credit data.

```
library(gmodels)
CrossTable(onlinepop_test$shares, onlinepop_pred_t20, prop.chisq
=FALSE, prop.c =FALSE, prop.r =FALSE, dnn =c('actual popular',
'predicted popular'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  9911
##
##
##                 | predicted popular
## actual popular |         0 |         1 | Row Total |
## ---------------|-----------|-----------|-----------|
##              0 |      3099 |      1953 |      5052 |
##                |     0.313 |     0.197 |           |
## ---------------|-----------|-----------|-----------|
##              1 |      1703 |      3156 |      4859 |
##                |     0.172 |     0.318 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |      4802 |      5109 |      9911 |
## ---------------|-----------|-----------|-----------|
##
##
```

```
percent_accuracy_t20 <-(3099+3156)/9911
percent_accuracy_t20
```

```
## [1] 0.6311169
```

The table indicates that for the 9911 records in our test set 1953 non-popular articles were misclassified, i.e. false negatives or a Type II error, and 1703 actual popular articles were misclassified as not popular, i.e. false positives or a Type I error. The accuracy of the model is 63.1%.

From this, we see that the accuracy of the 'Keywords' category model and the Top 20 features model on the test dataset is very close (63.6% vs. 63.1%). This is because in the Top 20 features model Type II error reduces but Type I error increases as compared to 'Keywords' category model. This makes the overall accuracy similar for both models.

Method 2: KNN Algorithm

Training a model on the data

Since I am using the same dataset for all the methods, I can use the cleaned, preprocessed datasets split into train and test obtained previously for training the KNN algorithm model on the data. In order to ensure there isn't a tie in the classification I will use an odd number for k. I will use k=173 because 173 is roughly the square root of the number of training records. I will take all the 58 features in the dataset for this method as this method takes less computation time than the first method. The first method took too much computation time when we took all the features and hence had to be aborted.

```
library(class)
n <-as.numeric(nrow(onlinepop_train))
k <-sqrt(n)
k
```

```
## [1] 172.4326

onlinepop_train_labels <-onlinepop_rand[1:29733,59]
onlinepop_test_labels <-onlinepop_rand[29734:nrow(onlinepop_c),59]

onlinepop_test_pred <-knn(train = onlinepop_train, test =
onlinepop_test,cl=onlinepop_train_labels,k =173)
summary(onlinepop_test_pred)

##    0    1
## 5661 4250
```

Evaluating Model Performance

As before we'll create a confusion table to evaluate our model. We'll set up the CrossTable()

command a bit differently and have a bit different output as follows:

```
CrossTable(x = onlinepop_test_labels, y = onlinepop_test_pred)

##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  9911
##
##
##                      | onlinepop_test_pred
## onlinepop_test_labels |         0 |         1 | Row Total |
## ----------------------|-----------|-----------|-----------|
##                     0 |      3241 |      1811 |      5052 |
##                       |    43.767 |    58.298 |           |
##                       |     0.642 |     0.358 |     0.510 |
##                       |     0.573 |     0.426 |           |
##                       |     0.327 |     0.183 |           |
## ----------------------|-----------|-----------|-----------|
##                     1 |      2420 |      2439 |      4859 |
##                       |    45.506 |    60.614 |           |
```

```
##                       |     0.498 |     0.502 |     0.490 |
##                       |     0.427 |     0.574 |           |
##                       |     0.244 |     0.246 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |      5661 |      4250 |      9911 |
##                       |     0.571 |     0.429 |           |
## ----------------------|-----------|-----------|-----------|
##
##
```

```
percent_accuracy_knn <-(3241+2439)/9911
percent_accuracy_knn
```

```
## [1] 0.5731006
```

```
library(caret)
precision_knn <-posPredValue(onlinepop_test_pred,
onlinepop_test$shares, positive="1")
precision_knn
```

```
## [1] 0.5738824
```

```
recall_knn <-sensitivity(onlinepop_test_pred,
onlinepop_test$shares,positive="1")
recall_knn
```

```
## [1] 0.5019551
```

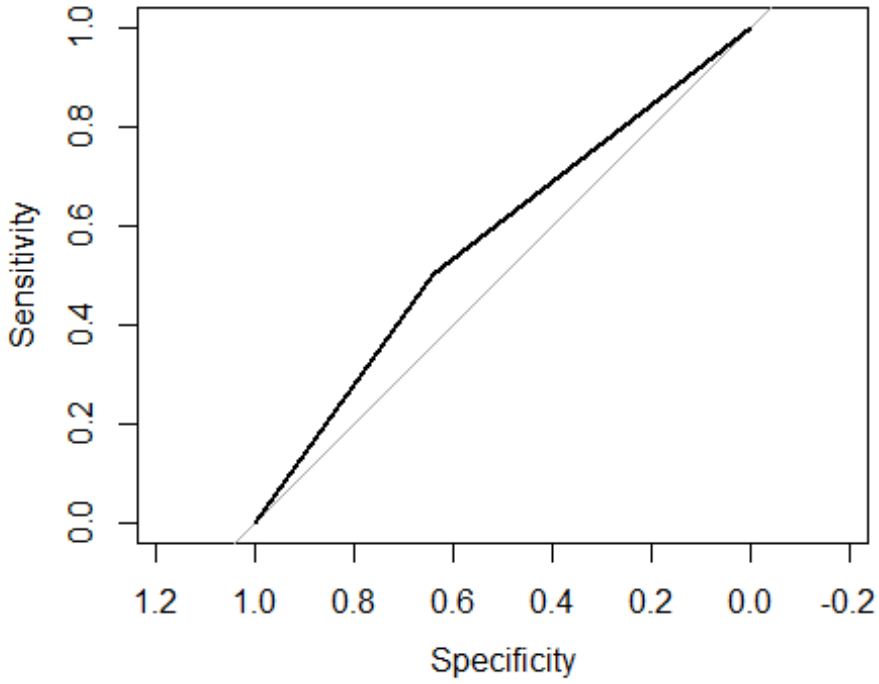
```
F1_knn <-(2 *precision_knn *recall_knn) /(precision_knn +recall_knn)
F1_knn
```

```
## [1] 0.5355143
```

```
library(pROC)
onlinepop_pred1 <-as.numeric(onlinepop_test_pred)
roc_obj_knn <-roc(onlinepop_test$shares,onlinepop_pred1)
library(pROC)
roc_obj_knn$auc
```

```
## Area under the curve: 0.5717
```

```
plot(roc_obj_knn)
```



This tells us that the model correctly classified 3241 articles that were not popular as not popular and 2439 articles that were popular as popular.The accuracy of the model is 57.3%.The accuracy obtained here is less than the first method (57.3% vs. 63%). I believe the value of k affects the accuracy rate. So I followed the same steps above with k=171. This time the model correctly classified 3249 articles that were not popular as not popular and 2445 articles that were popular as popular. The overall accuracy of the model marginally increases from 57.3% to 57.45%.

Method 3: Support vector machines

Training a model on the data

Again, I will use the same cleaned, pre-processed datasets for training the SVM model on the data. I use the kvsm() command by setting up a syntax like fn(y ~ x) on the train dataset. I also use the "vanilladot" kernel for this analysis.

```
library(kernlab)
```

```
## Warning: package 'kernlab' was built under R version 3.3.2
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
article_classifier <-ksvm(shares ~., data = onlinepop_train, kernel
="vanilladot")
```

```
##  Setting default kernel parameters
```

```
article_classifier
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 22979
##
## Objective Function Value : -22934.63
## Training error : 0.355968
```

The ksvm command runs for about 10 mins when I take the entire dataset with the 58 features. By entering the object name I used for the output from the ksvm() command at the command prompt, we see that we have an initial training error of 35.59%.

Evaluating Model Performance

I use the predict function on the test dataset to obtain predictions from the model. Again, I develop a confusion table to see how well the SVM model performed overall.

```
article_predictions <-predict(article_classifier, onlinepop_test)
summary(article_predictions)
```

```
##    0    1
## 5058 4853
```

```
ct <-table(article_predictions, onlinepop_test$shares)
addmargins(ct) # adding row and column totals
```

```
##
## article_predictions    0    1  Sum
##                   0   3279 1779 5058
##                   1   1773 3080 4853
##                   Sum 5052 4859 9911
```

```
agreement <-article_predictions ==onlinepop_test$shares
table(agreement)
```

```
## agreement
## FALSE  TRUE
##  3552  6359
```

```
#accuracy
percent_accuracy_svm <-(3279+3080)/9911
percent_accuracy_svm
```

```
## [1] 0.6416103
```

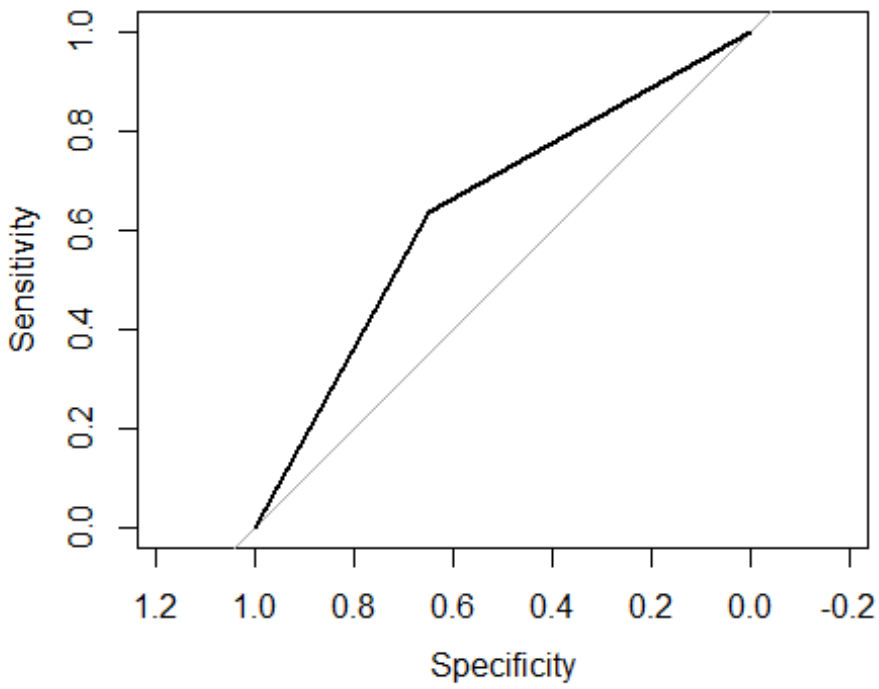
```
library(caret)
precision_svm <-posPredValue(article_predictions,
onlinepop_test$shares, positive="1")
precision_svm
```

```
## [1] 0.634659
```

```
recall_svm <-sensitivity(article_predictions,
onlinepop_test$shares,positive="1")
recall_svm
```

```
## [1] 0.6338753
```

```
F1_svm <-(2 *precision_svm *recall_svm) /(precision_svm +recall_svm)
F1_svm

## [1] 0.6342669

library(pROC)
onlinepop_pred1 <-as.numeric(article_predictions)
roc_obj_svm <-roc(onlinepop_test$shares,onlinepop_pred1)
library(pROC)
roc_obj_svm$auc

## Area under the curve: 0.6415

plot(roc_obj_svm)
```



The confusion table tells us that the model correctly classified 3279 articles that were not popular as not popular and 3080 articles that were popular as popular. This means out of our 9911 observations, 6359 observations were classified correctly. The accuracy of the model is 64.16%. From this we see that the accuracy obtained for this SVM model is higher than the previous methods.

Method 4: Naïve Bayes Algorithm

Training a model on the data

Proceeding to the next method using the same cleaned and pre-processed datasets, I use the e1071 package to apply the Naive Bayes Algorithm on the data.

```
library(e1071)

## Warning: package 'e1071' was built under R version 3.3.3

article_classifier_nb <-naiveBayes(onlinepop_train,
onlinepop_train$shares)
class(article_classifier_nb)

## [1] "naiveBayes"
```

Evaluating Model Performance

Now we can evaluate the performance of our model. Again we'll use the predict() function and look at the output in a confusion table.

```
article_test_pred <-predict(article_classifier_nb,
newdata=onlinepop_test)
t <-table(article_test_pred, onlinepop_test$shares)
addmargins(t) # adding row and column totals

##
## article_test_pred    0    1  Sum
##               0   4857 3451 8308
##               1    195 1408 1603
##               Sum 5052 4859 9911

#accuracy
np <-4857/5052
np

## [1] 0.9614014

p <-1408/4859
p

## [1] 0.2897716
```

```r
percent_accuracy_nb <-(4857+1408)/9911
percent_accuracy_nb
```

```
## [1] 0.6321259
```

```r
library(caret)
precision_nb <-posPredValue(article_test_pred, onlinepop_test$shares,
positive="1")
precision_nb
```

```
## [1] 0.8783531
```

```r
recall_nb <-sensitivity(article_test_pred,
onlinepop_test$shares,positive="1")
recall_nb
```

```
## [1] 0.2897716
```

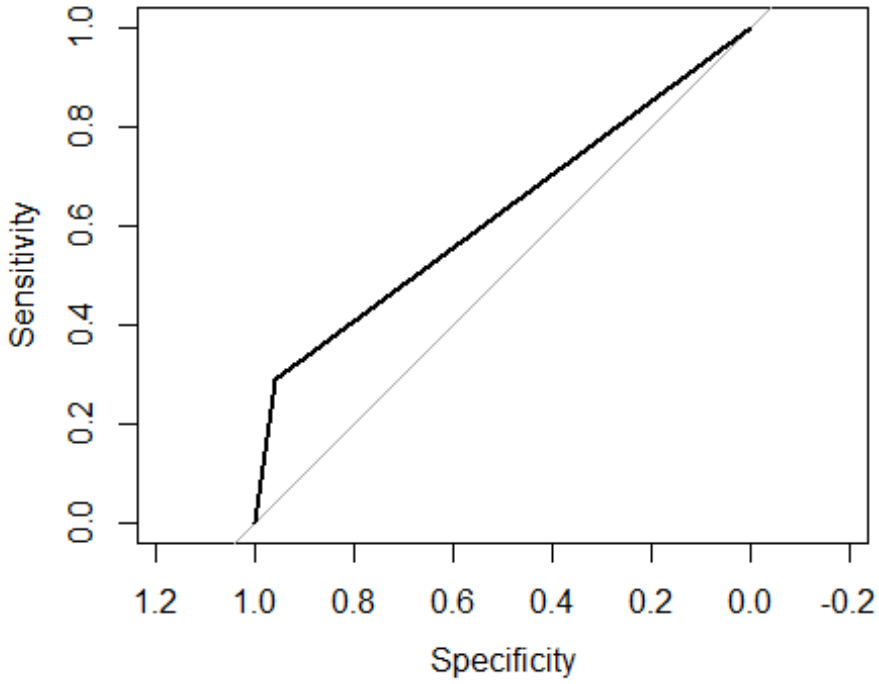
```r
F1_nb <-(2 *precision_nb *recall_nb) /(precision_nb +recall_nb)
F1_nb
```

```
## [1] 0.4357784
```

```r
library(pROC)
onlinepop_pred1 <-as.numeric(article_test_pred)
roc_obj_nb <-roc(onlinepop_test$shares,onlinepop_pred1)
library(pROC)
roc_obj_nb$auc
```

```
## Area under the curve: 0.6256
```

```r
plot(roc_obj_nb)
```



The confusion table tells us that the model correctly classified 4857 articles that were not popular as not popular and 1408 articles that were popular as popular. This means that the Naive Bayes filter correctly classified non-popular articles 4857 times out of 5052 observations (96.14%) and correctly classified popular articles 1408 times out of 4859 observations (28.97%). Thus, Type I error is much higher than Type II error for this model. The overall accuracy of the model is 63.21%. The overall accuracy of this model is similar to that of the first two methods - Tree based classification and KNN Algorithm.

Method 5: Adding regression to trees

Training a Model on the Data

As done before, I will use the cleaned and pre-processed train and test datasets obtained previously for building the model. To create this model I use the rpart package. "rpart"

stands for recursive partitioning. Using the rpart function I set-up the command with the formula syntax and the factor variable shares as below:

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 3.3.3
```

```
m.rpart <-rpart(shares ~., data=onlinepop_train)
m.rpart
```

```
## n= 29733
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 29733 14703 0 (0.5054989 0.4945011)
##   2) kw_avg_avg< 2910.297 15364  6178 0 (0.5978912 0.4021088)
##     4) data_channel_is_tech< 0.5 11821  4290 0 (0.6370865
0.3629135)
##       8) data_channel_is_socmed< 0.5 11201  3871 0 (0.6544059
0.3455941) *
##       9) data_channel_is_socmed>=0.5 620   201 1 (0.3241935
0.6758065) *
##     5) data_channel_is_tech>=0.5 3543  1655 1 (0.4671183 0.5328817)
*
##   3) kw_avg_avg>=2910.297 14369  5844 1 (0.4067089 0.5932911)
##     6) data_channel_is_entertainment>=0.5 2862  1241 0 (0.5663871
0.4336129) *
##     7) data_channel_is_entertainment< 0.5 11507  4223 1 (0.3669940
0.6330060) *
```

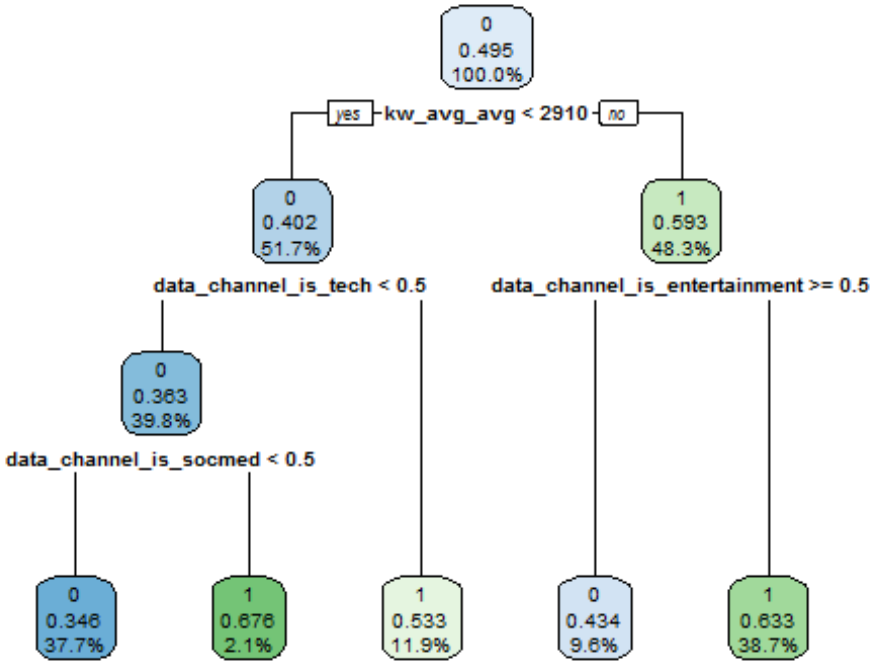
The basic information about the tree is obtained by typing the object name m.rpart.Note that the rpart function automatically determined that the most important predictor was the variable "kw_avg_avg". The root split between <2910.297 and >=2910.297 as shown above. Nodes with an * are terminal or leaf nodes.

I then use the rpart.plot package to actually plot the tree.

```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.3.3
```

```
rpart.plot(m.rpart, digits=3)
```



Here each node shows:

- the predicted class
- the predicted probability of becoming popular
- the percentage of observations in the node.

The highest probability of being popular is in the left leaf of the tree – 0.67 but has only 2.1% of the observations in the node. In contrast, the right has a lesser probability of 0.63 but has 38.7% of the observations in the node.

```
summary(m.rpart)
```

```
## Call:
## rpart(formula = shares ~ ., data = onlinepop_train)
##   n= 29733
##
##           CP nsplit rel error    xerror        xstd
## 1 0.18234374      0 1.0000000 1.0000000 0.005863503
## 2 0.02584507      1 0.8176563 0.8210569 0.005759330
```

```
## 3 0.01584711      2 0.7918112 0.7891587 0.005720826
## 4 0.01482691      3 0.7759641 0.7855540 0.005716082
## 5 0.01000000      4 0.7611372 0.7716112 0.005696973
##
## Variable importance
##                    kw_avg_avg                    kw_max_avg
##                            24                            15
##                    kw_min_avg                        LDA_03
##                            10                             9
##                    kw_min_max data_channel_is_entertainment
##                             9                             8
##                    kw_avg_max          data_channel_is_tech
##                             8                             7
##        data_channel_is_socmed                        LDA_04
##                             6                             4
##                        LDA_01
##                             2
##
## Node number 1: 29733 observations,    complexity param=0.1823437
##   predicted class=0  expected loss=0.4945011  P(node) =1
##     class counts: 15030 14703
##    probabilities: 0.505 0.495
##   left son=2 (15364 obs) right son=3 (14369 obs)
##   Primary splits:
##       kw_avg_avg                 < 2910.297   to the left,
improve=542.7720, (0 missing)
##       kw_max_avg                 < 3751.873   to the left,
improve=481.8572, (0 missing)
##       self_reference_min_shares  < 1650       to the left,
improve=426.4478, (0 missing)
##       self_reference_avg_sharess < 2948.667   to the left,
improve=386.6414, (0 missing)
##       LDA_02                     < 0.549098   to the right,
improve=356.1277, (0 missing)
##   Surrogate splits:
##       kw_max_avg < 4334.585   to the left,  agree=0.825, adj=0.637,
(0 split)
##       kw_min_avg < 1692.644   to the left,  agree=0.723, adj=0.426,
(0 split)
##       LDA_03     < 0.05105884 to the left,  agree=0.692, adj=0.363,
(0 split)
##       kw_min_max < 2950       to the left,  agree=0.692, adj=0.363,
(0 split)
##       kw_avg_max < 283307.1   to the left,  agree=0.675, adj=0.328,
(0 split)
##
```

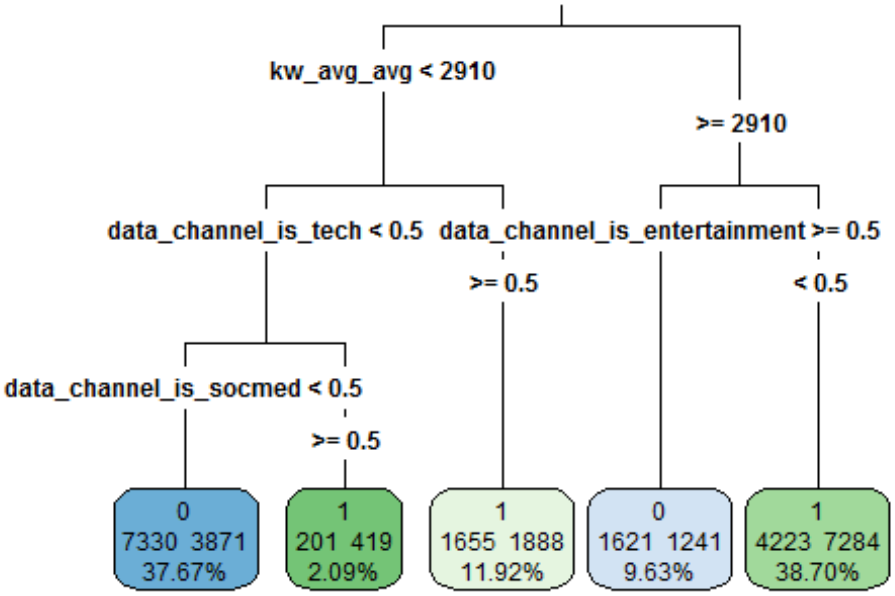
```
## Node number 2: 15364 observations,    complexity param=0.01584711
##   predicted class=0  expected loss=0.4021088  P(node) =0.5167323
##     class counts:  9186  6178
##    probabilities: 0.598 0.402
##   left son=4 (11821 obs) right son=5 (3543 obs)
##   Primary splits:
##       data_channel_is_tech       < 0.5        to the left,
improve=157.5023, (0 missing)
##       is_weekend                 < 0.5        to the left,
improve=149.6727, (0 missing)
##       self_reference_avg_sharess < 1995.273   to the left,
improve=138.0419, (0 missing)
##       self_reference_min_shares  < 1650       to the left,
improve=133.1357, (0 missing)
##       LDA_04                     < 0.3228501  to the left,
improve=132.7131, (0 missing)
##   Surrogate splits:
##       LDA_04                    < 0.5105669  to the left,
agree=0.896, adj=0.550, (0 split)
##       num_self_hrefs            < 13.5       to the left,
agree=0.772, adj=0.013, (0 split)
##       kw_avg_max                < 561685     to the left,
agree=0.770, adj=0.001, (0 split)
##       kw_min_avg                < 2599.111   to the left,
agree=0.770, adj=0.001, (0 split)
##       self_reference_min_shares < 640750     to the left,
agree=0.770, adj=0.001, (0 split)
##
## Node number 3: 14369 observations,    complexity param=0.02584507
##   predicted class=1  expected loss=0.4067089  P(node) =0.4832677
##     class counts:  5844  8525
##    probabilities: 0.407 0.593
##   left son=6 (2862 obs) right son=7 (11507 obs)
##   Primary splits:
##       data_channel_is_entertainment < 0.5        to the right,
improve=182.24500, (0 missing)
##       self_reference_min_shares     < 1650       to the left,
improve=148.33920, (0 missing)
##       self_reference_avg_sharess    < 2888.167   to the left,
improve=115.87460, (0 missing)
##       is_weekend                    < 0.5        to the left,
improve= 91.94284, (0 missing)
##       self_reference_max_shares     < 2950       to the left,
improve= 84.12785, (0 missing)
##   Surrogate splits:
##       LDA_01     < 0.4824625  to the right, agree=0.856, adj=0.279,
```

```
(0 split)
##       num_videos < 23.5       to the right, agree=0.804, adj=0.017,
(0 split)
##       num_imgs   < 48.5       to the right, agree=0.803, adj=0.009,
(0 split)
##       num_hrefs  < 179        to the right, agree=0.801, adj=0.001,
(0 split)
##       LDA_02     < 0.01818215 to the left,  agree=0.801, adj=0.001,
(0 split)
##
## Node number 4: 11821 observations,    complexity param=0.01482691
##   predicted class=0  expected loss=0.3629135  P(node) =0.3975717
##     class counts:  7531  4290
##    probabilities: 0.637 0.363
##   left son=8 (11201 obs) right son=9 (620 obs)
##   Primary splits:
##       data_channel_is_socmed    < 0.5        to the left,
improve=128.1182, (0 missing)
##       kw_avg_max                < 142493.1   to the right,
improve=112.6429, (0 missing)
##       kw_max_avg                < 3645.021   to the left,
improve=112.2319, (0 missing)
##       kw_max_max                < 654150     to the right,
improve=103.0073, (0 missing)
##       self_reference_min_shares < 1550       to the left,
improve=102.1694, (0 missing)
##   Surrogate splits:
##       num_self_hrefs < 40.5       to the left,  agree=0.948,
adj=0.010, (0 split)
##       num_keywords   < 2.5        to the right, agree=0.948,
adj=0.005, (0 split)
##
## Node number 5: 3543 observations
##   predicted class=1  expected loss=0.4671183  P(node) =0.1191605
##     class counts:  1655  1888
##    probabilities: 0.467 0.533
##
## Node number 6: 2862 observations
##   predicted class=0  expected loss=0.4336129  P(node) =0.09625668
##     class counts:  1621  1241
##    probabilities: 0.566 0.434
##
## Node number 7: 11507 observations
##   predicted class=1  expected loss=0.366994  P(node) =0.3870111
##     class counts:  4223  7284
##    probabilities: 0.367 0.633
```

```
##
## Node number 8: 11201 observations
##   predicted class=0  expected loss=0.3455941  P(node) =0.3767195
##     class counts:  7330  3871
##    probabilities: 0.654 0.346
##
## Node number 9: 620 observations
##   predicted class=1  expected loss=0.3241935  P(node) =0.02085225
##     class counts:   201   419
##    probabilities: 0.324 0.676
```

Another way to visualize the tree is using the command:

```
rpart.plot(m.rpart, digits=4, fallen.leaves =TRUE, type =3, extra
=101)
```



Evaluating Model Performance

Since we are doing a binary classification, that is our target variable is a binary variable (popular/not popular), we need to build a confusion table to determine how well the model works.

```r
article_test_pred_rt <-predict(m.rpart,
newdata=onlinepop_test,type="class")
t <-table(onlinepop_test$shares,article_test_pred_rt)
rownames(t) <-paste("Actual", rownames(t), sep =":")
colnames(t) <-paste("Pred", colnames(t), sep =":")
addmargins(t)
```

```
##           article_test_pred_rt
##            Pred:0 Pred:1  Sum
##   Actual:0   2986   2066 5052
##   Actual:1   1673   3186 4859
##   Sum        4659   5252 9911
```

```r
#accuracy
np <-2986/5052
np
```

```
## [1] 0.591053
```

```r
p <-3186/4859
p
```

```
## [1] 0.6556905
```

```r
percent_accuracy_rt <-(2986+3186)/9911
percent_accuracy_rt
```

```
## [1] 0.6227424
```

```r
library(caret)
precision_rt <-posPredValue(article_test_pred_rt,
onlinepop_test$shares, positive="1")
precision_rt
```

```
## [1] 0.606626
```

```r
recall_rt <-sensitivity(article_test_pred_rt,
onlinepop_test$shares,positive="1")
recall_rt
```

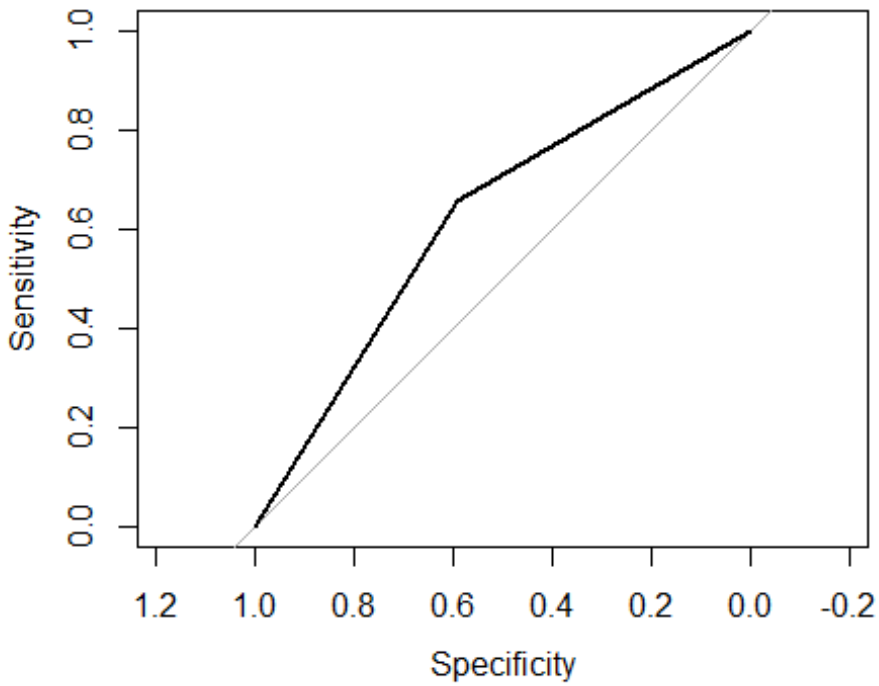
```
## [1] 0.6556905
```

```r
F1_rt <-(2 *precision_rt *recall_rt) /(precision_rt +recall_rt)
F1_rt
```

```
## [1] 0.6302047
```

```r
library(pROC)
onlinepop_pred1 <-as.numeric(article_test_pred_rt)
roc_obj_rt <-roc(onlinepop_test$shares,onlinepop_pred1)
```

```
library(pROC)
roc_obj_rt$auc

## Area under the curve: 0.6234

plot(roc_obj_rt)
```



The confusion table tells us that the model correctly classified 2986 articles that were not popular as not popular and 3186 articles that were popular as popular. This means that the regression tree model correctly classified non-popular articles 2986 times out of 5052 observations (59.10%) and correctly classified popular articles 3186 times out of 4859 observations (65.57%). The overall accuracy of the model is 62.27%. Thus, this method has the least overall accuracy amongst the 5 methods used on this dataset.

Conclusion

When using a single dataset for all the 5 methods we see that that some models performed better than others. However, when we look at overall accuracy of each method, we see that it is pretty close for all methods except KNN Algorithm which has significantly less accuracy.

```
ctable <-
matrix(c(percent_accuracy_tc,precision_tc,recall_tc,F1_tc,roc_obj_tc$auc,percent_accuracy_knn,precision_knn,recall_knn,F1_knn,roc_obj_knn$auc,percent_accuracy_svm,precision_svm,recall_svm,F1_svm,roc_obj_svm$auc,percent_accuracy_nb,precision_nb,recall_nb,F1_nb,roc_obj_nb$auc,percent_accuracy_rt,precision_rt,recall_rt,F1_tc,roc_obj_rt$auc),byrow = T,nrow =5,ncol =5)

dimnames(ctable) <-list(c("Tree based classification","KNN Algorithm","Support vector machines","Naive Bayes Algorithm","Adding regression to trees"),c("Accuracy","Precision","Recall","F1","AUC"))

ctable
```

```
##                              Accuracy Precision    Recall        F1
## Tree based classification  0.6362627 0.6180791 0.6754476 0.6454912
## KNN Algorithm              0.5731006 0.5738824 0.5019551 0.5355143
## Support vector machines    0.6416103 0.6346590 0.6338753 0.6342669
## Naive Bayes Algorithm      0.6321259 0.8783531 0.2897716 0.4357784
## Adding regression to trees 0.6227424 0.6066260 0.6556905 0.6454912
##                                 AUC
## Tree based classification  0.6370112
## KNN Algorithm              0.5717416
## Support vector machines    0.6414626
## Naive Bayes Algorithm      0.6255865
## Adding regression to trees 0.6233718
```

** The best values are highlighted in bold*

The table above summarizes the key parameters used to measure how each model performs. When we look at the overall Accuracy of the model, Support vector machines (SVM) are the best and KNN Algorithm is the worst.

The precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances, while recall (also known as sensitivity) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Looking at the Precision parameter of the 5 models, the Naive Bayes Algorithm has the highest precision of 0.87 followed by the SVM model which has a precision value of 0.63. However, when we look at the Recall parameter, we see that the Naive Bayes Algorithm has least Recall value of 0.28. Tree based classification has the highest recall value of 0.67.

The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0. Both Tree based classification and Adding regression to trees have highest F-Score of 0.64. Naive Bayes Algorithm has least F1-Score of 0.43.

From a random classifier you can expect as many true positives as false positives. AUC score for the case is 0.5. A score for a perfect classifier would be 1. Most often you get something in between. SVM has the highest AUC score of 0.64.

I see that SVM strikes a balance in all the 5 parameters and hence is the best model for classifying this news popularity dataset. This is in contrast to the Naive Bayes Algorithm which has the highest precision but very low Recall value and F-Score. All the parameters of the SVM model have values around 0.6 which is greater than a random classifier value of 0.5. Hence, I would choose the SVM model as the best model amongst the 5 models.

References

- Fernandes, K., Vinagre, P., & Cortez, P. (2015). A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. Progress in Artificial Intelligence Lecture Notes in Computer Science, 535-546. doi:10.1007/978-3-319-23485-4_53
- Ren, H., & Yang, Q. (n.d.). Predicting and Evaluating the Popularity of Online News.
- Z., Z. (n.d.). FastML. Retrieved December 23, 2017, from http://fastml.com/what-you-wanted-to-know-about-auc
- Precision and recall. (2017, December 13). Retrieved December 23, 2017, from https://en.wikipedia.org/wiki/Precision_and_recall
- Yedidia, A. (2016). Against the F-score. Against the F-score. Retrieved from https://adamyedidia.files.wordpress.com/2014/11/f_score.pdf.
- Milborrow, S. (2016). Plotting rpart trees with the rpart.plot package.