Capstone Project-Social Media Shares Prediction

Shiva Chaitanya

Loading Libraries:

```
library(dplyr)
library(caret)
library(corrplot)
library(ggplot2)
library(esquisse)
```

Reading the data:

```
Data<- read.csv("Data.csv")
```

Data Cleaning:

```
#Check for null values:

null_values<-Data%>%is.na()%>%colMeans()*100
head(null_values,61)
```

```
##                            url                      timedelta
##                              0                              0
##                 n_tokens_title                n_tokens_content
##                              0                              0
##                n_unique_tokens                n_non_stop_words
##                              0                              0
##      n_non_stop_unique_tokens                      num_hrefs
##                              0                              0
##                 num_self_hrefs                       num_imgs
##                              0                              0
##                     num_videos            average_token_length
##                              0                              0
##                   num_keywords       data_channel_is_lifestyle
##                              0                              0
## data_channel_is_entertainment             data_channel_is_bus
##                              0                              0
##         data_channel_is_socmed            data_channel_is_tech
##                              0                              0
##          data_channel_is_world                      kw_min_min
##                              0                              0
##                     kw_max_min                      kw_avg_min
##                              0                              0
##                     kw_min_max                      kw_max_max
##                              0                              0
```

```
##                   kw_avg_max                   kw_min_avg
##                            0                            0
##                   kw_max_avg                   kw_avg_avg
##                            0                            0
##     self_reference_min_shares     self_reference_max_shares
##                            0                            0
##    self_reference_avg_sharess            weekday_is_monday
##                            0                            0
##           weekday_is_tuesday          weekday_is_wednesday
##                            0                            0
##          weekday_is_thursday             weekday_is_friday
##                            0                            0
##          weekday_is_saturday             weekday_is_sunday
##                            0                            0
##                  is_weekend                       LDA_00
##                            0                            0
##                       LDA_01                       LDA_02
##                            0                            0
##                       LDA_03                       LDA_04
##                            0                            0
##          global_subjectivity     global_sentiment_polarity
##                            0                            0
##    global_rate_positive_words    global_rate_negative_words
##                            0                            0
##          rate_positive_words           rate_negative_words
##                            0                            0
##        avg_positive_polarity        min_positive_polarity
##                            0                            0
##        max_positive_polarity        avg_negative_polarity
##                            0                            0
##        min_negative_polarity        max_negative_polarity
##                            0                            0
##           title_subjectivity      title_sentiment_polarity
##                            0                            0
##       abs_title_subjectivity  abs_title_sentiment_polarity
##                            0                            0
##                      shares
##                            0
```

```
#Removing irrelavant variables:

Data1<- Data[,-c(1,2)]

#There are 59 variables in total after removing irrelavant variables.

#Removing rows where number of words in the article are zero:

dim(Data1)
```

```
## [1] 39644    59
```

```
mydata = Data1 %>% filter(n_tokens_content!=0)
dim(mydata)
```

```
## [1] 38463    59
```

```
#After eliminating rows in which there are no words in the articles, there are 38463 rows and 59 variab

#Creating a Preprocessing Model to eliminate variables with high correlation and near zero variance:

preProcessModel <- preProcess(mydata[,-c(59)], method = c("nzv", "corr"))
Preprocessed_data <- predict(preProcessModel, mydata)

preProcessModel
```

```
## Created from 38463 samples and 5 variables
##
## Pre-processing:
##   - ignored (0)
##   - removed (5)
```

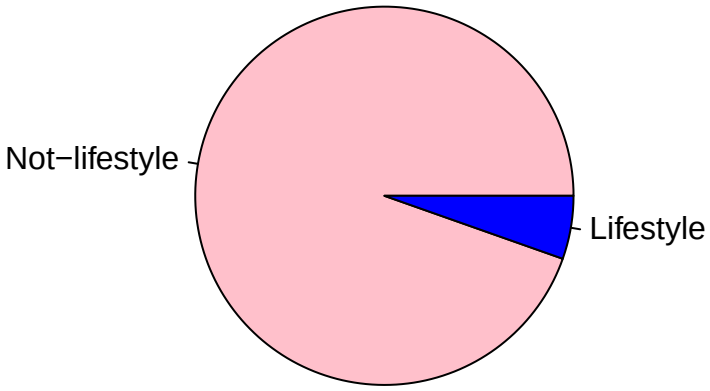
```
New_data<- Preprocessed_data
```

Exploratory Data Analysis:

```
lifestyle<- New_data %>% select(data_channel_is_lifestyle) %>% group_by(data_channel_is_lifestyle) %>% s

head(lifestyle)
```

```
## # A tibble: 2 x 3
##   data_channel_is_lifestyle count percentage
##                       <int> <int>      <dbl>
## 1                         0 36386      94.6
## 2                         1  2077       5.40
```

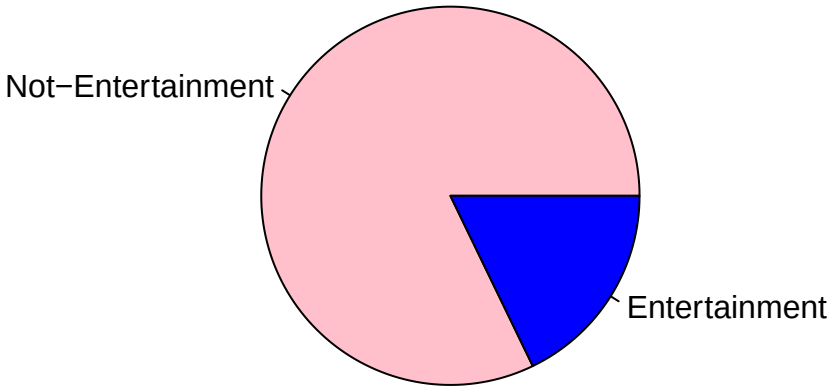
```
pie(lifestyle$percentage,
    labels = c("Not-lifestyle","Lifestyle"),
    col=c("pink","blue"))
```




```
Entertainment<- New_data %>% select(data_channel_is_entertainment) %>% group_by(data_channel_is_entertai

head(Entertainment)
```

```
## # A tibble: 2 x 3
##   data_channel_is_entertainment count percentage
##                           <int> <int>      <dbl>
## 1                             0 31607       82.2
## 2                             1  6856       17.8
```

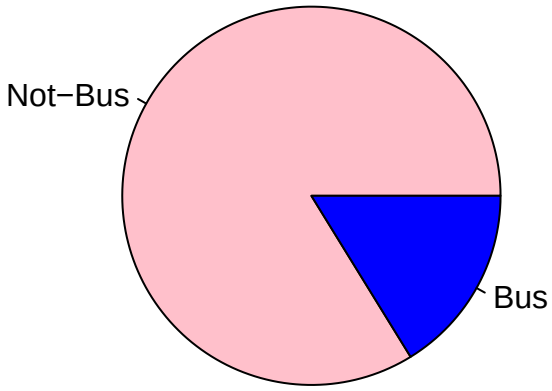
```
pie(Entertainment$percentage,
    labels = c("Not-Entertainment","Entertainment"),
    col=c("pink","blue"))
```




```
Bus<- New_data %>% select(data_channel_is_bus) %>% group_by(data_channel_is_bus) %>% summarise(count=n()

head(Bus)
```

```
## # A tibble: 2 x 3
##   data_channel_is_bus count percentage
##                 <int> <int>      <dbl>
## 1                   0 32228       83.8
## 2                   1  6235       16.2
```

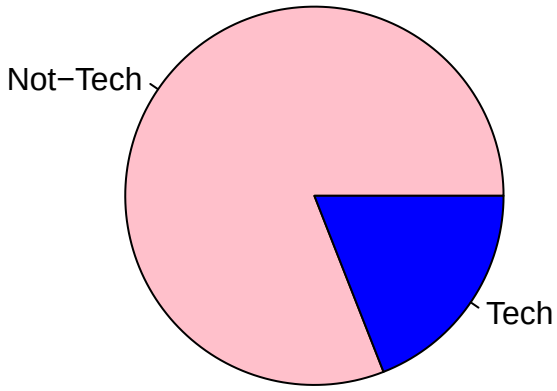
```
pie(Bus$percentage,
    labels = c("Not-Bus","Bus"),
    col=c("pink","blue"))
```




```
Tech<- New_data %>% select(data_channel_is_tech) %>% group_by(data_channel_is_tech) %>% summarise(count=

head(Tech)
```

```
## # A tibble: 2 x 3
##   data_channel_is_tech count percentage
##                  <int> <int>      <dbl>
## 1                    0 31138       81.0
## 2                    1  7325       19.0
```

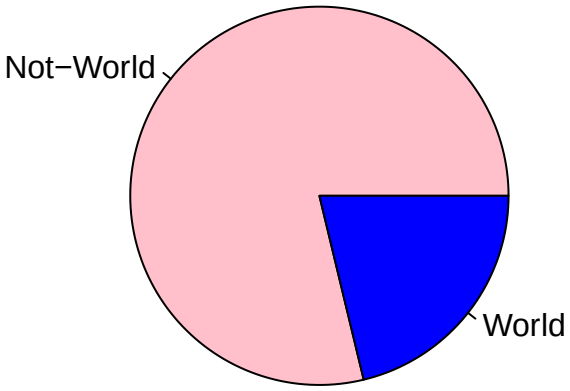
```
pie(Tech$percentage,
    labels = c("Not-Tech","Tech"),
    col=c("pink","blue"))
```




```
World<- New_data %>% select(data_channel_is_world) %>% group_by(data_channel_is_world) %>% summarise(co

head(World)
```

```
## # A tibble: 2 x 3
##   data_channel_is_world count percentage
##                   <int> <int>      <dbl>
## 1                     0 30295       78.8
## 2                     1  8168       21.2
```

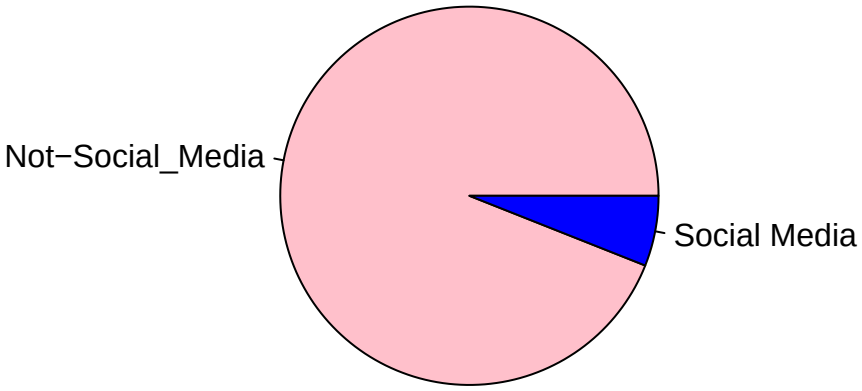
```
pie(World$percentage,
    labels = c("Not-World","World"),
    col=c("pink","blue"))
```



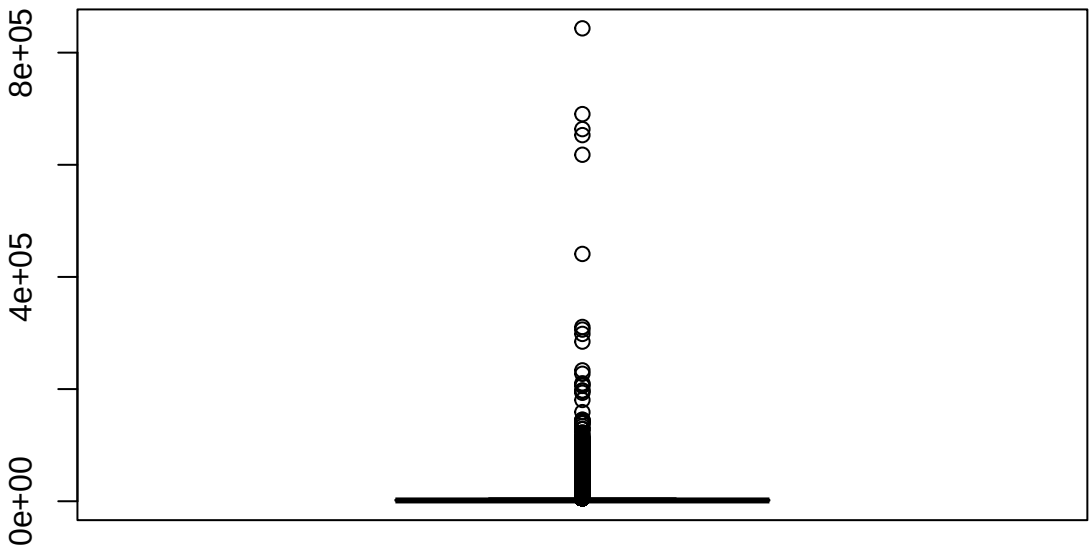

```
Social_Media<- New_data %>% select(data_channel_is_socmed) %>% group_by(data_channel_is_socmed) %>% summ

head(Social_Media)
```

```
## # A tibble: 2 x 3
##   data_channel_is_socmed count percentage
##                    <int> <int>      <dbl>
## 1                      0 36152      94.0
## 2                      1  2311       6.01
```

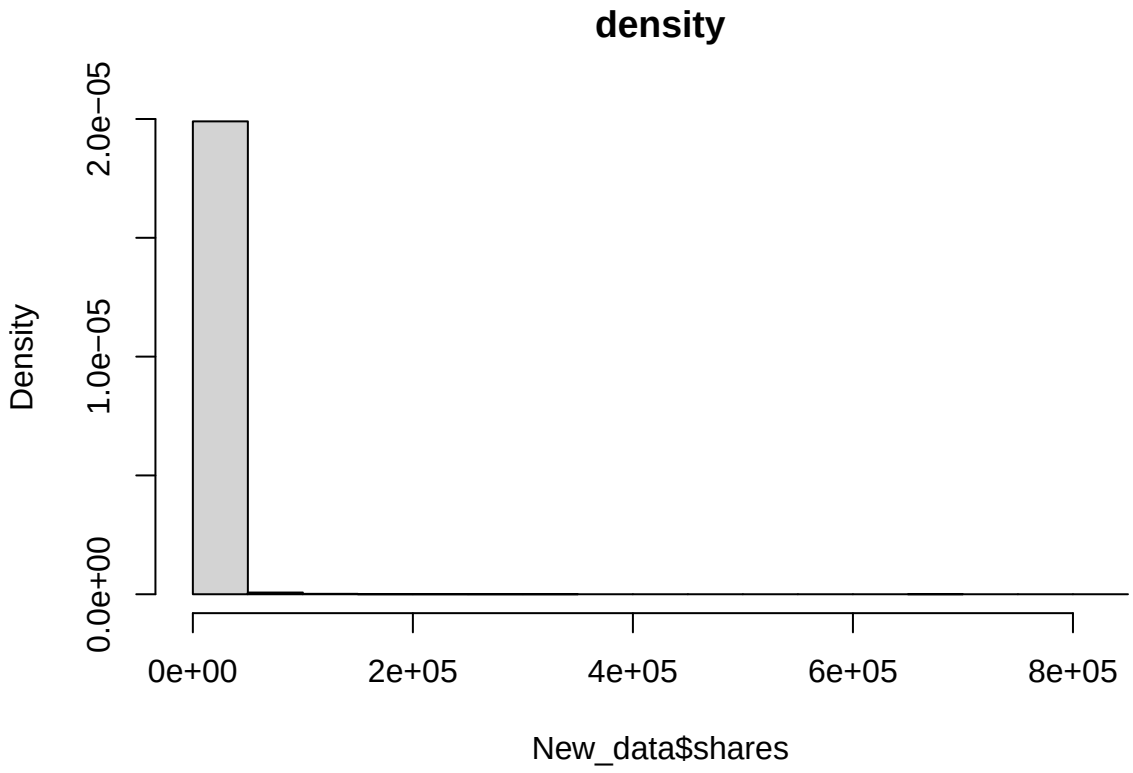
```
pie(Social_Media$percentage,
    labels = c("Not-Social_Media","Social Media"),
    col=c("pink","blue"))
```



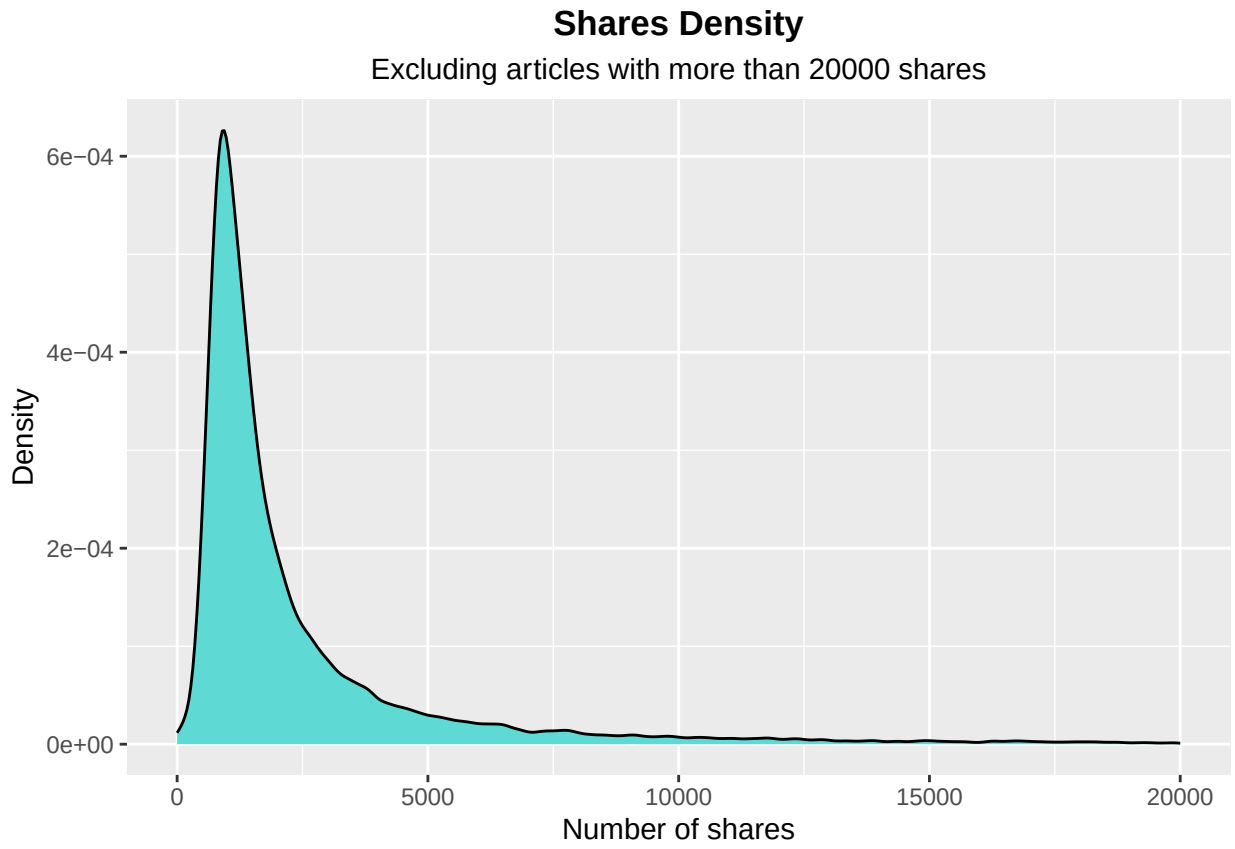

```
boxplot(New_data$shares)
```




```
hist(New_data$shares, freq = FALSE, main = "density")
```




```
ggplot(New_data) + aes(x=shares) + geom_density(fill="#5fd9d3") + xlim(0,20000)+xlab("Number of shares"
 theme(plot.title = element_text(hjust = 0.5,face="bold"),plot.subtitle = element_text(hjust = 0.5))
```



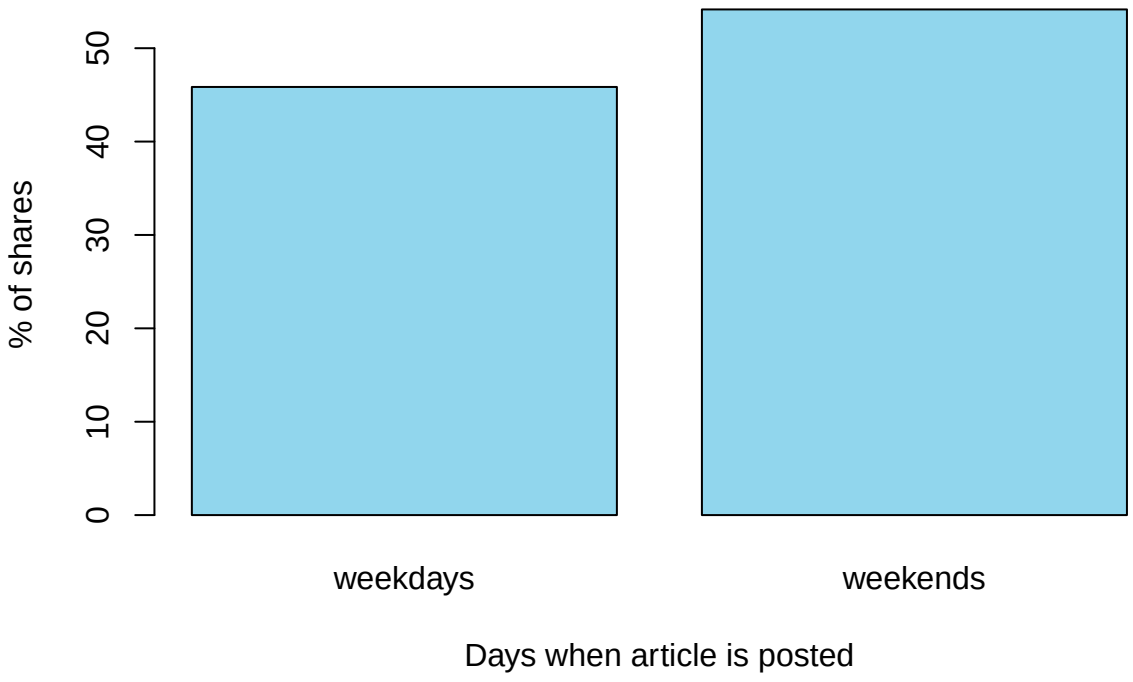

```
Weekend<- New_data %>% select(is_weekend) %>% group_by(is_weekend) %>% summarise(count=n()) %>% mutate(
```

```
Weekend_shares<- New_data %>% select(is_weekend,shares) %>% group_by(is_weekend) %>% summarise(avg_share
```

```
Weekend_shares
```

```
## # A tibble: 2 x 3
##   is_weekend avg_shares percentage
##        <int>      <dbl>      <dbl>
## 1          0      3278.       45.8
## 2          1      3871.       54.2
```

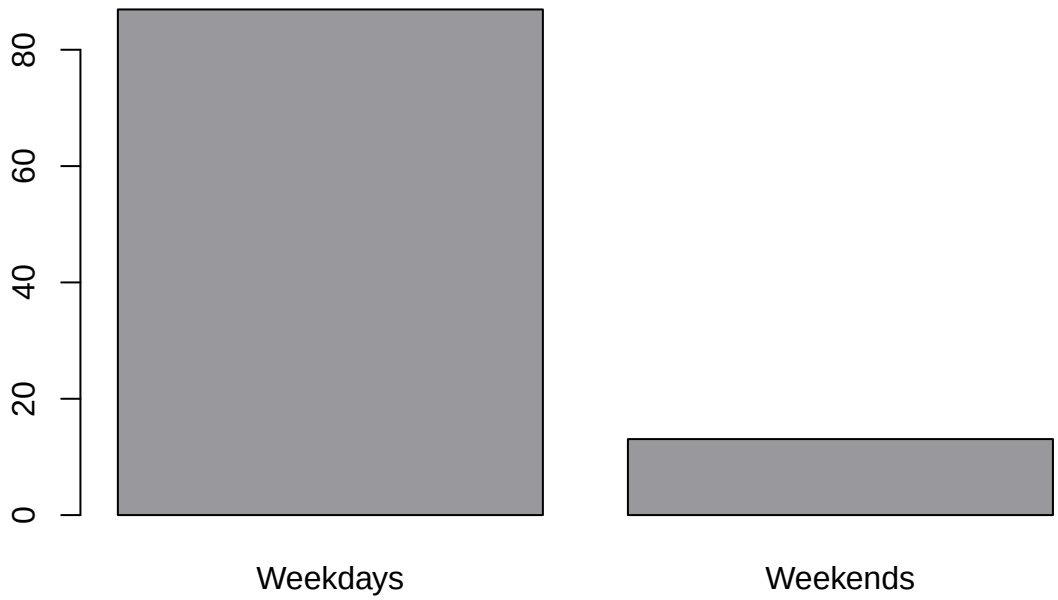
```
barplot(Weekend_shares$percentage, names=c("weekdays","weekends"),col="#91d6ed",xlab="Days when article
```




```
head(Weekend)
```

```
## # A tibble: 2 x 3
##   is_weekend count percentage
##        <int> <int>      <dbl>
## 1          0 33437       86.9
## 2          1  5026       13.1
```

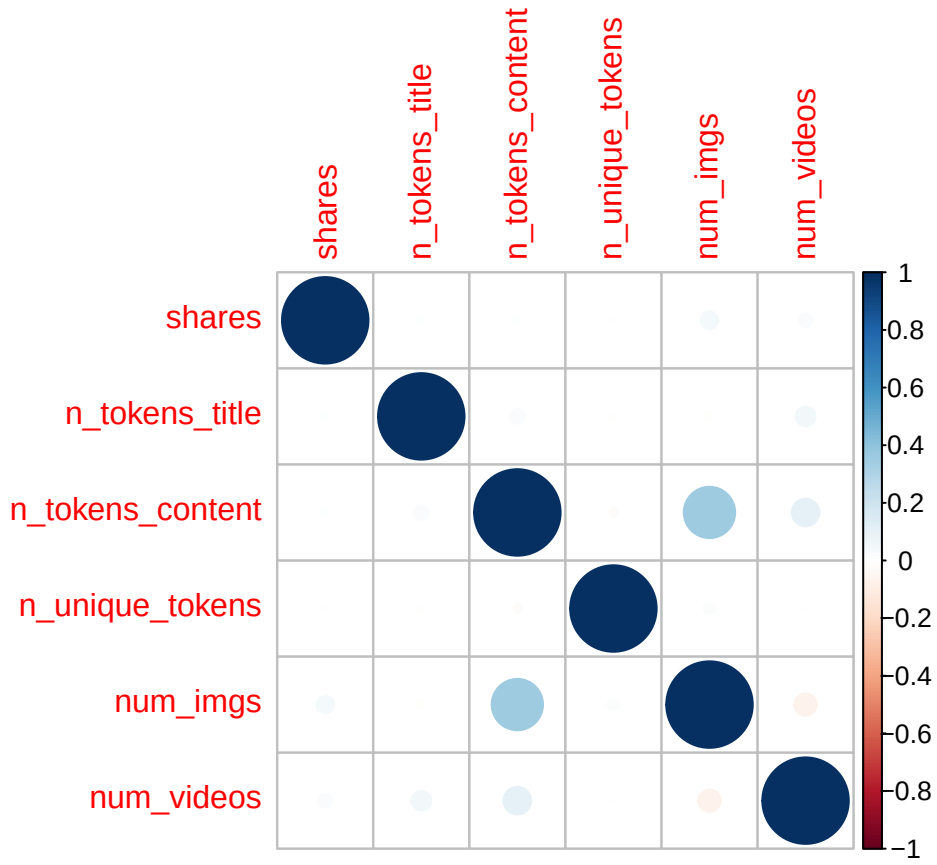
```
barplot(Weekend$percentage,names=c("Weekdays","Weekends"),col="#99989c")
```




```
#Checking the correlations between few important variables with number of shares:

Correlation<-cor(New_data[,c("shares","n_tokens_title","n_tokens_content","n_unique_tokens","num_imgs",'

Correlation
```

```
##                      shares n_tokens_title n_tokens_content n_unique_tokens
## shares           1.000000000    0.006208955      0.006713658     0.001368099
## n_tokens_title   0.006208955    1.000000000      0.028124037    -0.004180165
## n_tokens_content 0.006713658    0.028124037      1.000000000    -0.010506403
## n_unique_tokens  0.001368099   -0.004180165     -0.010506403     1.000000000
## num_imgs         0.041294287   -0.006617612      0.353096761     0.018767055
## num_videos       0.024712981    0.052470297      0.102032955    -0.001126779
##                      num_imgs   num_videos
## shares            0.041294287  0.024712981
## n_tokens_title   -0.006617612  0.052470297
## n_tokens_content  0.353096761  0.102032955
## n_unique_tokens   0.018767055 -0.001126779
## num_imgs          1.000000000 -0.066592705
## num_videos       -0.066592705  1.000000000
```

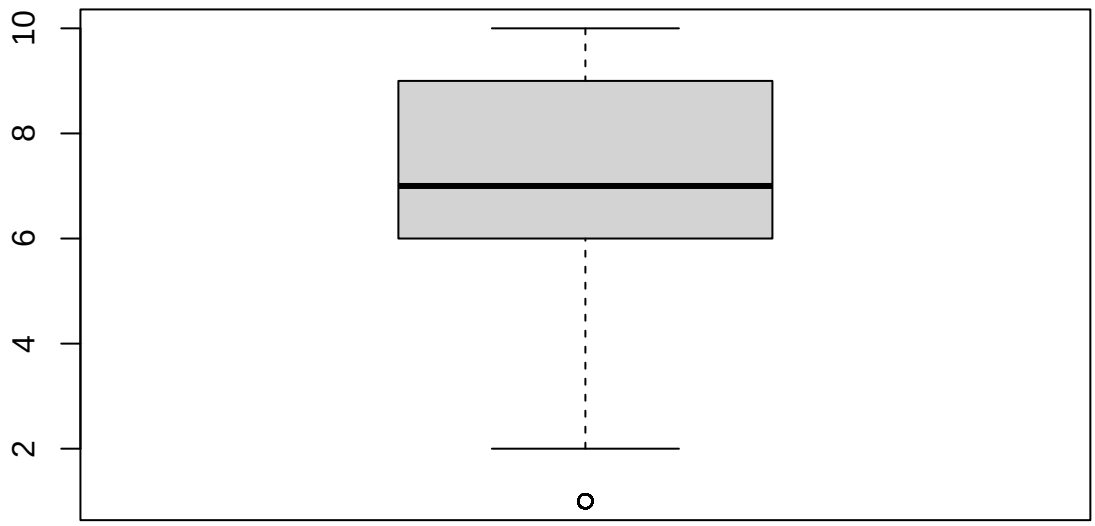
```
corrplot(Correlation)
```




```
New_data%>%select(num_imgs,num_videos,shares)%>%mutate(Graphics=num_imgs+num_videos)%>%group_by(Graphics
```

```
## # A tibble: 6 x 2
##   Graphics mean_shares
##      <int>       <dbl>
## 1       57        629
## 2       72        796
## 3       69        888
## 4       67       1026.
## 5       85       1100
## 6       77       1228.
```

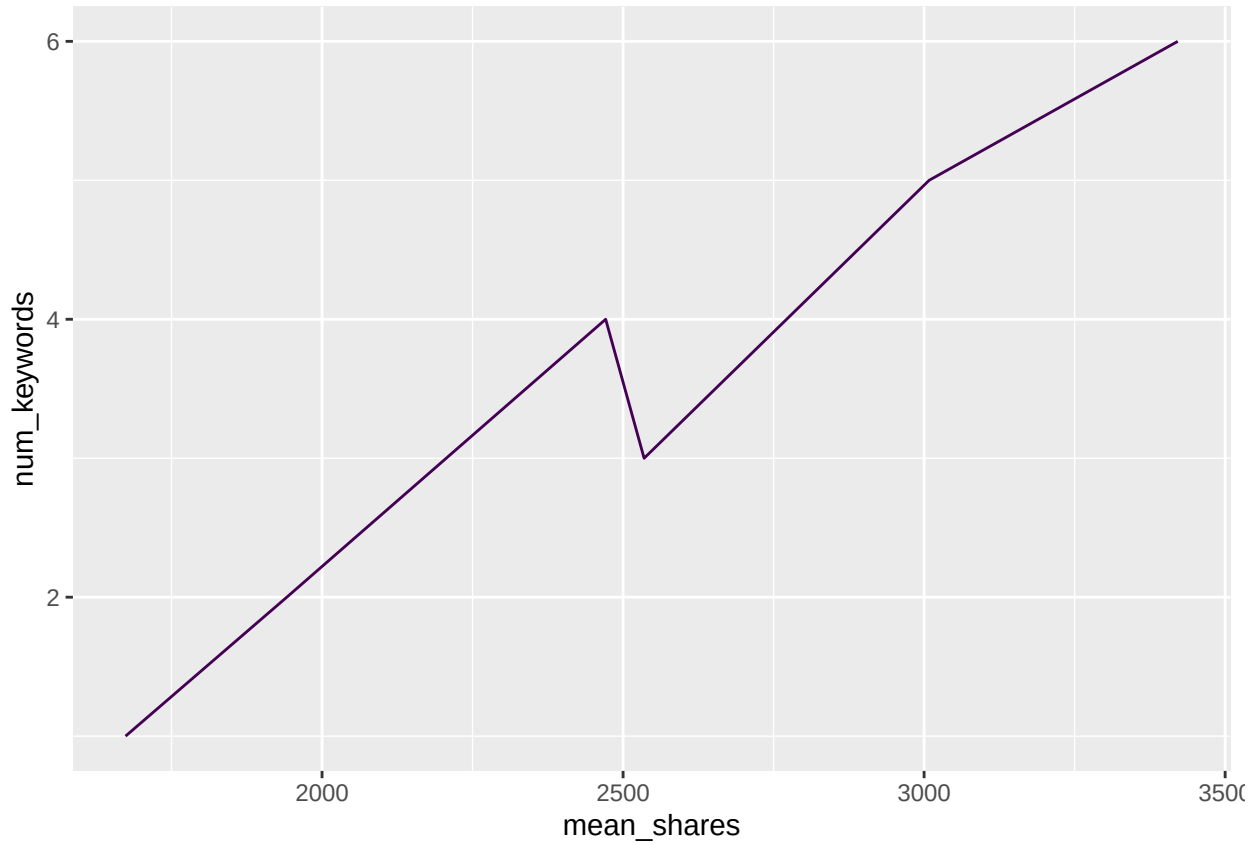
```
boxplot(New_data$num_keywords)
```



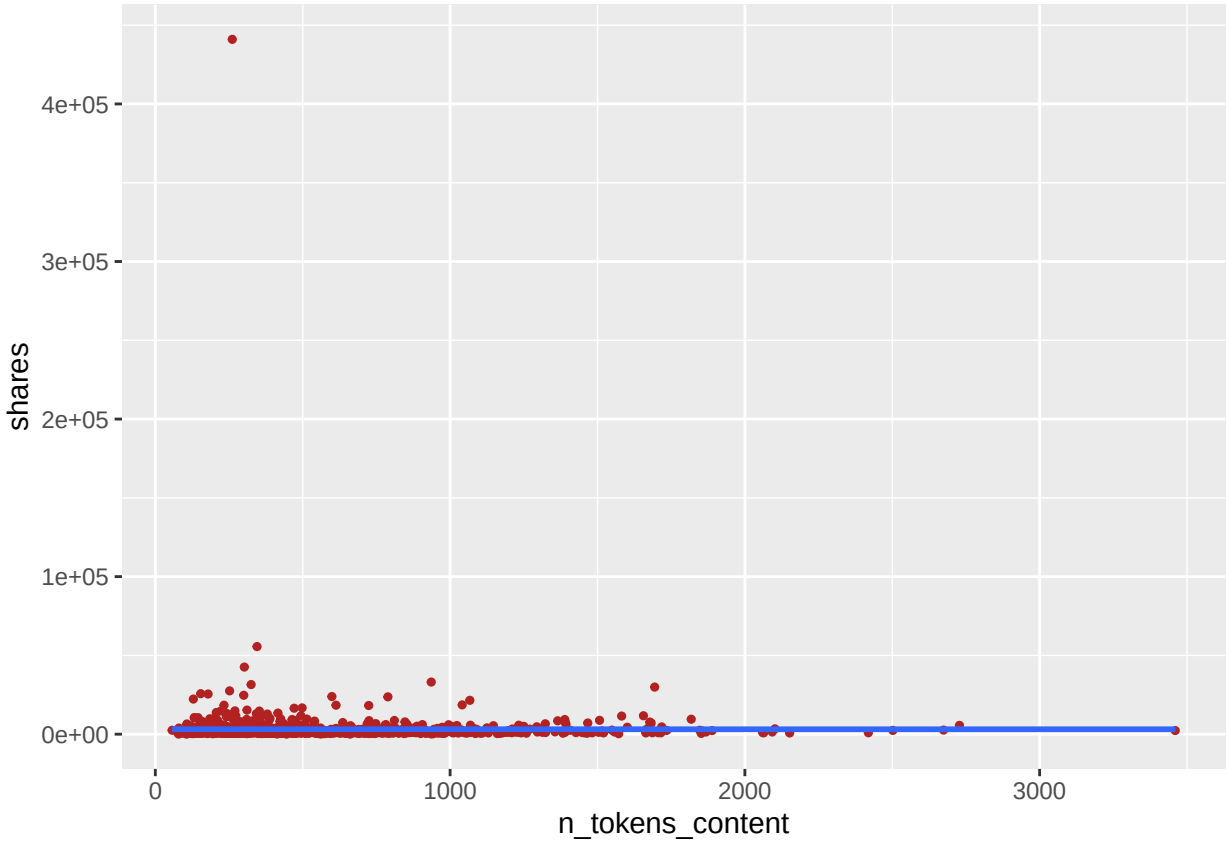

```
keywords_shares<- New_data%>%select(num_keywords,shares)%>%group_by(num_keywords)%>%summarise(mean_shar

keywords_shares
```

```
## # A tibble: 6 x 2
##   num_keywords mean_shares
##          <int>       <dbl>
## 1            1       1674.
## 2            2       1941.
## 3            4       2471.
## 4            3       2535.
## 5            5       3008.
## 6            6       3421.
```

```
ggplot(keywords_shares) +
  aes(x = mean_shares, y = num_keywords) +
  geom_line(colour = "#440154") +
  theme_gray()
```




```
temp = sample_n(tbl = New_data, size = 1000)


ggplot(temp) +
  aes(x = n_tokens_content, y = shares) +
  geom_point(
    shape = "circle",
    size = 0.95,
    colour = "#B22222"
  ) +
  geom_smooth(span = 0.75) +
  theme_gray()
```



Data Preparation:

```
#Variable Selection:

Model_variables<- New_data[,-c(10:15,26:33,23,24,29)]
dim(Model_variables)
```

```
## [1] 38463    38
```

```
View(Model_variables)

#Data Partition:

set.seed(123)
samples=createDataPartition(Model_variables$shares,p=0.7,list=FALSE,times=1)  #Training And Testing
training = Model_variables[samples, ]
test = Model_variables[-samples,]

dim(training)
```

```
## [1] 26925    38
```

```
dim(test)
```

```
## [1] 11538    38
```

```r
#Normalization:

#Normalization <- preProcess(training[,-(38)],method = c("center","scale"))

Normalization <- preProcess(training,method = c("center","scale"))
Normalized_data = predict(Normalization,training)

Normalized_test = predict(Normalization,test)
```

Model Building:

```r
#Regression:

Model<-lm(log(shares)~.,data=Normalized_data)
summary(Model)
```

```
##
## Call:
## lm(formula = log(shares) ~ ., data = Normalized_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6788 -0.9320  0.1085  1.0311  5.7722
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -1.63258    0.02309 -70.706  < 2e-16 ***
## n_tokens_title                0.06327    0.02166   2.921  0.00350 **
## n_tokens_content              0.02747    0.03202   0.858  0.39098
## n_unique_tokens               2.62596    1.35485   1.938  0.05265 .
## num_hrefs                     0.05877    0.02163   2.718  0.00660 **
## num_self_hrefs               -0.06891    0.02196  -3.138  0.00171 **
## num_imgs                      0.02573    0.02257   1.140  0.25435
## num_videos                    0.01687    0.02046   0.824  0.40975
## average_token_length         -0.02976    0.02247  -1.324  0.18546
## num_keywords                 -0.05845    0.02516  -2.323  0.02024 *
## kw_min_min                    0.06228    0.03804   1.637  0.10166
## kw_avg_min                   -0.01575    0.02200  -0.716  0.47403
## kw_max_max                    0.08894    0.04288   2.074  0.03809 *
## kw_avg_max                   -0.05741    0.03278  -1.751  0.07993 .
## kw_min_avg                   -0.10335    0.02605  -3.967 7.36e-05 ***
## kw_max_avg                   -0.10515    0.04570  -2.301  0.02142 *
## kw_avg_avg                    0.25969    0.05526   4.699 2.67e-06 ***
## self_reference_avg_sharess    0.02299    0.01637   1.404  0.16028
## LDA_00                      113.92147   58.91543   1.934  0.05321 .
## LDA_01                       95.24439   49.21864   1.935  0.05303 .
## LDA_02                      121.24098   62.66881   1.935  0.05309 .
## LDA_03                      124.61316   64.36677   1.936  0.05292 .
## LDA_04                      125.25125   64.74759   1.934  0.05311 .
## global_subjectivity           0.04369    0.02646   1.651  0.09878 .
## global_sentiment_polarity     0.10044    0.05381   1.866  0.06204 .
## global_rate_positive_words   -0.07524    0.03995  -1.883  0.05972 .
## global_rate_negative_words    0.03779    0.04845   0.780  0.43545
```

```
## rate_negative_words            0.02950    0.05929   0.497  0.61890
## avg_positive_polarity         -0.05431    0.04069  -1.335  0.18201
## min_positive_polarity          0.03107    0.02765   1.124  0.26114
## max_positive_polarity          0.01855    0.03270   0.567  0.57059
## avg_negative_polarity         -0.06193    0.05236  -1.183  0.23701
## min_negative_polarity         -0.03903    0.04510  -0.866  0.38678
## max_negative_polarity          0.03757    0.03320   1.132  0.25781
## title_subjectivity            -0.01544    0.02973  -0.519  0.60345
## title_sentiment_polarity      -0.02397    0.02337  -1.026  0.30514
## abs_title_subjectivity         0.02432    0.02395   1.015  0.30996
## abs_title_sentiment_polarity   0.07087    0.03028   2.341  0.01927 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.527 on 5404 degrees of freedom
##   (21483 observations deleted due to missingness)
## Multiple R-squared:  0.05248,    Adjusted R-squared:  0.046
## F-statistic:  8.09 on 37 and 5404 DF,  p-value: < 2.2e-16
```

R square of this model is just 5%. R square indicates the proportion of variability explained by the independent variable in predicting dependent variable.

P-value is used to understand the significant relationship each independent variable has with the dependent variable. Considering threshold for p-value to be 0.5 and all variables with p-value less than 0.5 will be selected to create a new model.

```
#Deploying the Regression model on Test data:

Testing<- predict(Model, Normalized_test)
View(Testing)

RMSE(Normalized_test$shares,Testing)
```

```
## [1] 1.919897
```

```
MAE(Normalized_test$shares,Testing)
```

```
## [1] 1.636684
```

2. Implementing Decision Trees:

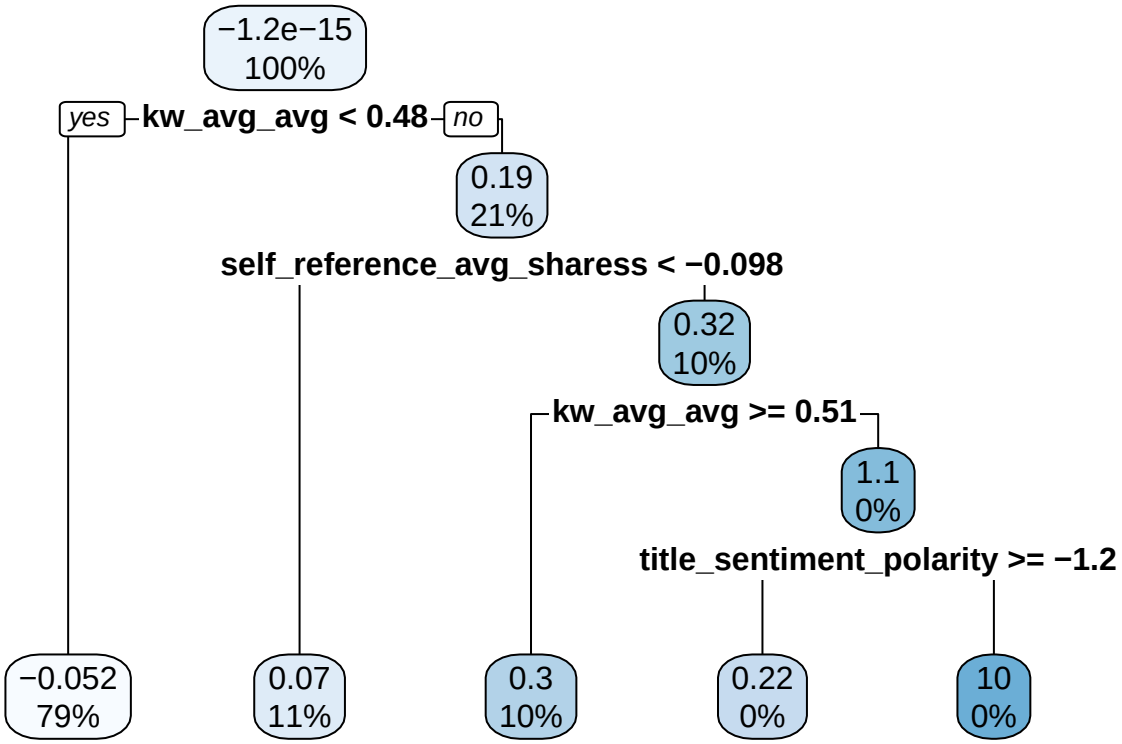
```
library(rpart)
library(rpart.plot)

DT_Model<- rpart(Normalized_data$shares~.,data=Normalized_data,method = "anova")

summary(DT_Model)
```

```
## Call:
## rpart(formula = Normalized_data$shares ~ ., data = Normalized_data,
##     method = "anova")
##   n= 26925
```

```
##
##           CP nsplit rel error   xerror      xstd
## 1 0.01016825      0 1.0000000 1.000049 0.2852842
## 2 0.01007462      1 0.9898318 1.055993 0.2855698
## 3 0.01000000      4 0.9596079 1.053996 0.2848968
##
## Variable importance
##   title_sentiment_polarity                 kw_avg_avg
##                         44                         29
##                kw_max_avg self_reference_avg_sharess
##                          9                          6
##                kw_min_avg                     LDA_03
##                          5                          3
##                kw_avg_max             num_self_hrefs
##                          1                          1
##                kw_avg_min
##                          1
##
## Node number 1: 26925 observations,    complexity param=0.01016825
##   mean=-1.172184e-15, MSE=0.9999629
##   left son=2 (21238 obs) right son=3 (5687 obs)
##   Primary splits:
##       kw_avg_avg                 < 0.4789966   to the left,  improve=0.010168250, (0 missing)
##       self_reference_avg_sharess < 0.05820575  to the left,  improve=0.009136286, (0 missing)
##       kw_max_avg                 < -0.08924886 to the left,  improve=0.008885773, (0 missing)
##       LDA_03                     < 0.4853166   to the left,  improve=0.006925981, (0 missing)
##       num_hrefs                  < 0.2889801   to the left,  improve=0.004675041, (0 missing)
##   Surrogate splits:
##       kw_max_avg < 0.2044047   to the left,  agree=0.894, adj=0.497, (0 split)
##       kw_min_avg < 1.434858    to the left,  agree=0.843, adj=0.256, (0 split)
##       LDA_03     < 0.7845483   to the left,  agree=0.827, adj=0.179, (0 split)
##       kw_avg_max < 1.611529    to the left,  agree=0.800, adj=0.055, (0 split)
##       kw_avg_min < 1.581208    to the left,  agree=0.796, adj=0.036, (0 split)
##
## Node number 2: 21238 observations
##   mean=-0.05217949, MSE=0.597424
##
## Node number 3: 5687 observations,    complexity param=0.01007462
##   mean=0.1948634, MSE=2.455097
##   left son=6 (2900 obs) right son=7 (2787 obs)
##   Primary splits:
##       self_reference_avg_sharess < -0.09819786 to the left,  improve=0.006582502, (0 missing)
##       kw_avg_avg                 < 1.476441    to the left,  improve=0.003584726, (0 missing)
##       num_hrefs                  < 0.2889801   to the left,  improve=0.002792616, (0 missing)
##       avg_negative_polarity      < -0.6752942  to the right, improve=0.002443455, (0 missing)
##       num_imgs                   < 1.058174    to the left,  improve=0.002355079, (0 missing)
##   Surrogate splits:
##       num_self_hrefs < -0.2284394  to the left,  agree=0.584, adj=0.151, (0 split)
##       kw_avg_avg     < 1.078817    to the left,  agree=0.566, adj=0.114, (0 split)
##       num_imgs       < -0.01262243 to the left,  agree=0.550, adj=0.081, (0 split)
##       num_hrefs      < 0.3769279   to the left,  agree=0.546, adj=0.074, (0 split)
##       num_videos     < -0.182801   to the left,  agree=0.545, adj=0.072, (0 split)
##
## Node number 6: 2900 observations
```

```
##   mean=0.07024004, MSE=0.6110762
##
## Node number 7: 2787 observations,    complexity param=0.01007462
##   mean=0.3245396, MSE=4.340908
##   left son=14 (2707 obs) right son=15 (80 obs)
##   Primary splits:
##       kw_avg_avg            < 0.506972    to the right, improve=0.004284960, (0 missing)
##       avg_negative_polarity < -0.6752942  to the right, improve=0.003566803, (0 missing)
##       kw_max_avg            < -0.07764797 to the right, improve=0.003341113, (0 missing)
##       kw_avg_min            < 2.422737    to the left,  improve=0.003178640, (0 missing)
##       num_hrefs             < 0.4648758   to the left,  improve=0.003106866, (0 missing)
##
## Node number 14: 2707 observations
##   mean=0.3010938, MSE=2.507201
##
## Node number 15: 80 observations,    complexity param=0.01007462
##   mean=1.117886, MSE=65.741
##   left son=30 (73 obs) right son=31 (7 obs)
##   Primary splits:
##       title_sentiment_polarity < -1.218044   to the right, improve=0.12739410, (0 missing)
##       num_imgs                 < 0.9391969   to the left,  improve=0.12640440, (0 missing)
##       num_hrefs                < 1.388328    to the left,  improve=0.12093170, (0 missing)
##       kw_avg_avg               < 0.5040441   to the left,  improve=0.12026480, (0 missing)
##       kw_max_max               < 0.07282981  to the right, improve=0.07214455, (0 missing)
##   Surrogate splits:
##       kw_avg_avg < 0.5059507   to the left,  agree=0.925, adj=0.143, (0 split)
##
## Node number 30: 73 observations
##   mean=0.2217366, MSE=1.880452
##
## Node number 31: 7 observations
##   mean=10.46344, MSE=636.0008
```

```
rpart.plot(DT_Model)
```




```
#Making Predictions:

predictions<-predict(DT_Model,Normalized_test)
```

Evaluating performance of Decision Trees Model:

```
MAE(Normalized_test$shares,predictions)
```

```
## [1] 0.263379
```

```
RMSE(Normalized_test$shares,predictions)
```

```
## [1] 0.9805839
```

3. Implementing Random Forest Model:

```
library(randomForest)

rf_model <- randomForest(shares ~ ., data = Normalized_data,ntree=150, mtry=3,maxnodes=60 )

rf_model
```

```
##
```

```
## Call:
##  randomForest(formula = shares ~ ., data = Normalized_data, ntree = 150,      mtry = 3, maxnodes = 6
##                Type of random forest: regression
##                      Number of trees: 150
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.9884413
##                     % Var explained: 1.15
```

```
Predictions_rf<-predict(rf_model,Normalized_test)
```

Evaluating the performance of Random Forest Model:

```
RMSE(Normalized_test$shares,Predictions_rf)
```

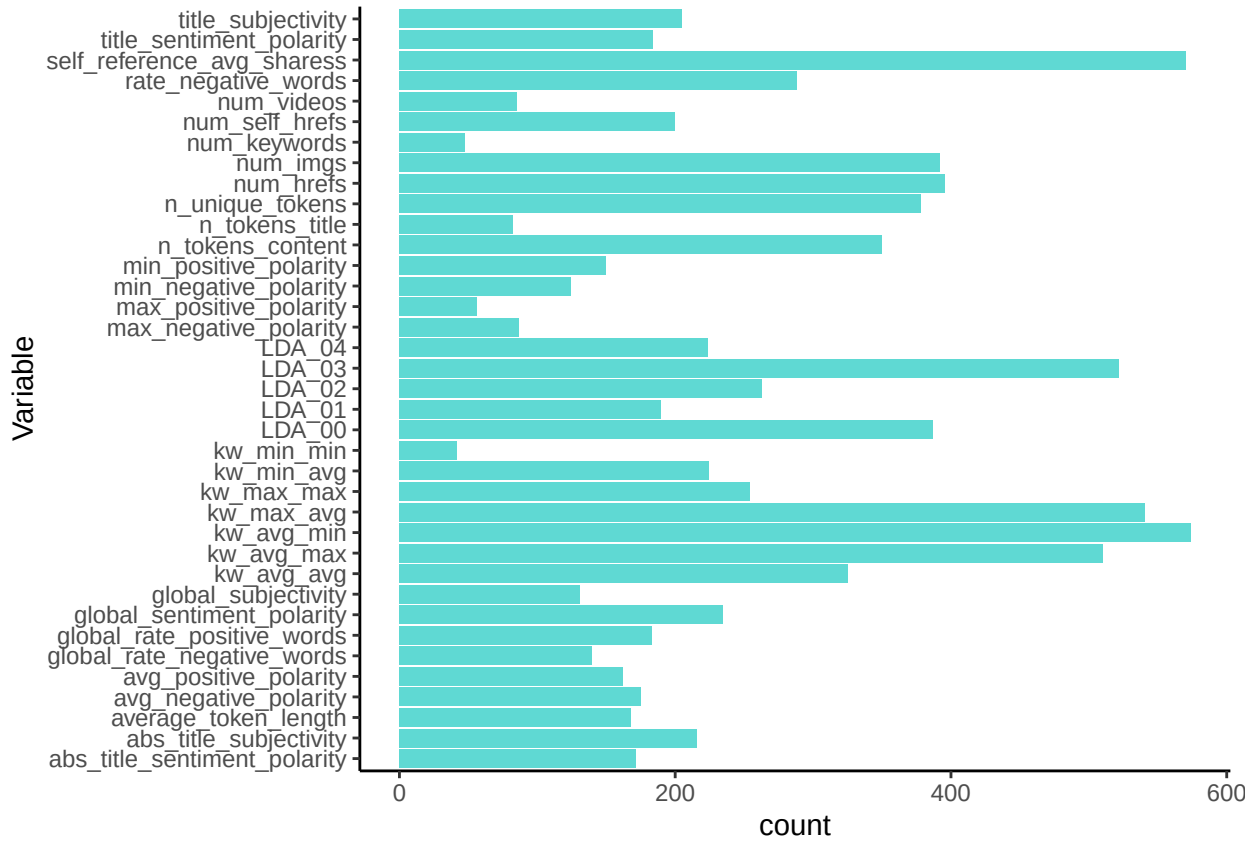
```
## [1] 0.9777411
```

```
MAE(Normalized_test$shares,Predictions_rf)
```

```
## [1] 0.26046
```

Visualizing the variable importance:

```
var_imp <- as.data.frame(importance(rf_model))
var_imp$Variable <- rownames(var_imp)

var_imp_sort<- var_imp  %>% arrange(var_imp$IncNodePurity)

View(var_imp_sort)

ggplot(var_imp_sort) +
  aes(x = Variable, weight = IncNodePurity) +
  geom_bar(fill = "#5fd9d3") +
  coord_flip() +
  theme_classic()
```



Interpreting results of all models:

- **Based on the above results, it can be concluded that Decision Trees improved the performance of the model significantly compared to Regression model.**
- **A drastic decrease in the MAE was observed when Decision Trees are used over Regression.**
- **Even though Random Forest model is expected to reduce overfitting, in comparison to Decision Trees, they do not show much of an improvement.**
- **Hence, we conclude that Decision Trees algorithm works best in predicting the number of shares variable in our dataset.**