Configuration Manual

Chizoba Ezegbu

x19109661

1 Brief Overview

This configuration manual itemizes the technical requirements and highlights the steps required in conducting this research study - *Predicting Key Success Factors for Selecting Crowdfunding Campaigns/Projects using Machine Learning Techniques: A Case Study of the Reward Based Crowdfunding Platform.*

2 System Requirements

2.1 Hardware

- Windows Operating System Version 10 – 64bit
- Processor: Intel® Core™ i5 processor; Windows 10 Pro 64; 512 GB
- CPU @ 1.60GHz, 1800 Mhz, RAM: 8GB memory HDD: 500GB

2.2 Software

- R programming Language and R studio – Version 3.5.2 - This analysis conducted via R.

3 Data Collection

Publicly available Kickstarter crowdfunding past projects data is sourced from Webrobots website. This data covers 8 project categories for the period 2009 to 2018.

3.1 Metadata

S/N	Variable	Description
1	State	The outcome of the project
2	Goal	The amount of funding set to achieve/raise for the project
3	Backers Count	Number of investors supporting the project
4	Spotlight	Updates on the project
5	Staff Pick	Projects recommended by Kickstarter staff for funding
6	Country	This is the location of the creator/project initiator
7	Duration	Period of funding
8	Category	Project area

4 Libraries

S/N	Name of Library	Version	Purpose
1	library(corrplot)	0.84	To graphically show the correlation graph/matrix
2	library(metrics)	0.1.4	Evaluation metrics (used when calculating the AUC)
3	library(dplyr)	1.0.0	Grammar for data manipulation
4	library(ggplot2)	3.3.2	For data visualization/plotting graphs
5	library(e1071)	1.7-3	For training in SVM
6	library(naivebayes)	0.9.7	Implementing Naïve Bayes Algorithm
7	library(tidyverse)	0.2.5	Used for text mining
8	library(data.table)	1.12.8	An extension of data.frame
9	library(tm)	0.7.7	Used for text mining
10	library(caTools)	1.18.0	For AUC
11	library(readr)	1.3.1	Used in reading data
12	library(readxl)	1.3.1	Used in reading excel files
13	library(stringr)	1.4.0	For string operations
14	library(caret)	6.0-86	A set of functions for creating predictive models (classification)
15	library(Boruta)	7.0.0	Used for feature selection

16	library(randomForest)	4.6.14	Used for solving classification problem (random forest algorithm)
17	library(SentimentAnalysis)	1.3.3	Used for sentiment analysis
18	library(sentimentr)	2.7.1	For text mining/sentiment
19	library(wordcloud)	2.6	Used in text mining, analyzes text and visualizes keywords
20	library(reshape2)	1.4.4	Reshaping data
21	library(rpart.plot)	3.0.8	Plotting decision tree
22	library(tokenizers)	0.2.1	Used in text mining
23	library(tidyr)	1.1.0	Used in text mining to tidy the data
24	library(tidytext)	0.2.5	Used in text mining
25	library(rpart.plot)	3.0.8	Used in plotting the decision tree
26	library(NLP)	0.2-0	Used for text mining
27	library(splitstackshape)	1.4.8	Text mining (splitting)
28	library(MASS)	7.3-51.6	Used in statistical analysis

5 Architectural Design

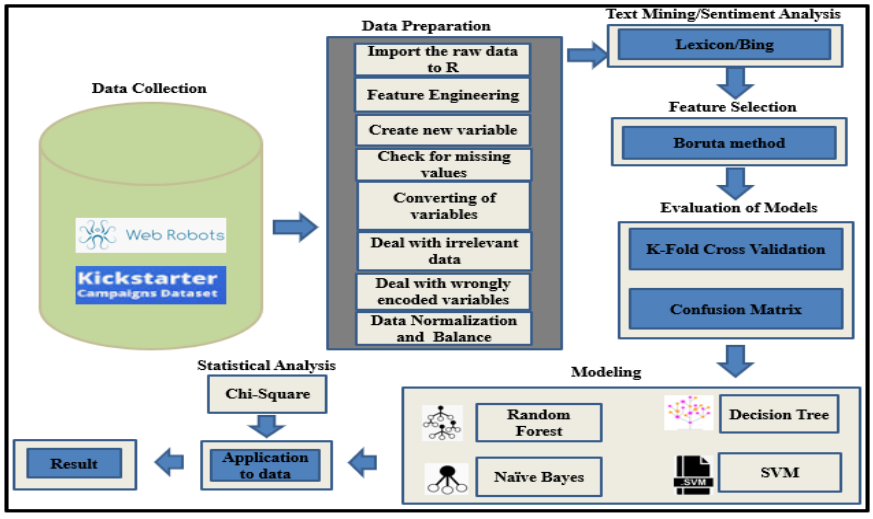

Figure 1 - Research Design

6 R Codes for Analysis

Step 1

Installation of relevant R packages for the analysis using "**install.package**"

Step 2

Import dataset using **library(readxl)** function

```
getwd()
setwd('R')
library(readxl)
NEWJanuary_2018_kickstarter_CFnew <-
read_excel("C:/Users/mypc/Desktop/NEWJanuary 2018 kickstarter CFnew.xlsx")
kcs <- NEWJanuary_2018_kickstarter_CFnew
View(kcs)
```

Step 3

Data variables is re-arranged to have the dependent variable as the first column

```
kcs <- kcs[, c(27,1:26, 28:37)]
KSdata<-kcs
View(KSdata)
```

Step 4

Data Cleaning and Transformation

```
## Changing the date column
date_cols <- c("deadline","state_changed_at","created_at","launched_at")

## Converting time stamp into date format
KSdata[date_cols] <-
lapply(lapply(KSdata[date_cols],as.POSIXlt.POSIXct),as.Date)

View(KSdata)

#Creating new variable - Duration
KSdata$duration<-as.numeric(KSdata$deadline - KSdata$launched_at)

View(KSdata)

#checking for duplicate data
KSdata<-KSdata[!duplicated(KSdata$id), ]
sum(duplicated(KSdata))
```

```
##Summary of Missing values in the data
sum(is.na(KSdata))
colSums(is.na(KSdata))


##Removing columns irrelevant to the analysis
KSdata <- KSdata[, -c(28:37)]
KSdata <- KSdata[, -c(5,8,10,15,20,21,23,24, 25)]


str(KSdata)

## Converting variables
KSdata$deadline <- as.Date(KSdata$deadline)

KSdata$launched_at <- as.Date(KSdata$launched_at)

KSdata$created_at <- as.Date(KSdata$created_at)

KSdata$state<-as.factor(KSdata$state)

KSdata$spotlight<-as.factor(KSdata$spotlight)
levels(KSdata$spotlight) <- c("0", "1")

KSdata$currency_trailing_code<-as.factor(KSdata$currency_trailing_code)
levels(KSdata$currency_trailing_code) <- c("0", "1")

KSdata$staff_pick<-as.factor(KSdata$staff_pick)
levels(KSdata$staff_pick) <- c("0", "1")

KSdata$disable_communication<-as.factor(KSdata$disable_communication)
levels(KSdata$disable_communication) <- c("0", "1")

KSdata$is_starrable<-as.factor(KSdata$is_starrable)

levels(KSdata$is_starrable) <- c("0", "1")

#currency
levels(KSdata$currency)
KSdata$currency<-as.character(KSdata$currency)
KSdata$currency[KSdata$currency %in% c("JPY", "HKD", "SGD", "CHF", "NOK",
"DKK", "MXN", "NZD",  "SEK")] <- "Other"
KSdata$currency <- as.factor(KSdata$currency)
levels(KSdata$currency)


#country
levels(KSdata$country)
KSdata$country<-as.character(KSdata$country)
KSdata$country[KSdata$country %in% c("JP", "LU", "AT", "HK", "SG", "BE",
"CH", "IE", "NO", "DK",  "MX", "NZ", "SE", "ES", "IT", "NL", "FR", "DE")] <-
"Other"
KSdata$country <- as.factor(KSdata$country)
levels(KSdata$country)

str(KSdata)
```

```
#Encoding Variables
#STATE
levels(KSdata$state)
levels(KSdata$state) <- c("failed", "failed",
"successful","successful","failed")
levels(KSdata$state) <- c("0", "1")

#Plotting levels
ggplot(KSdata, aes(state)) +
  geom_bar() +
  ylab("# of Projects") + xlab("Final State") +
  ggtitle("Final state of Kickstarter campaigns")
```

Step 5

Normalizing the Data and Checking Data Balancing

```
normalize<- function(x) {   return ((x - min(x)) / (max(x) - min(x))) }
KSdata$goal<-normalize(KSdata$goal)
KSdata$pledged<-normalize(KSdata$pledged)
KSdata$backers_count<-normalize(KSdata$backers_count)
KSdata$duration<-normalize(KSdata$duration)


#data balancing
prop.table(table(KSdata$state))

View(KSdata)
```

Step 6

Check for Correlating Variables

```
#Checking for correlation of variables #numeric variables only
corMat <- cor(KSdata[, c(2,12,16)])
corrplot.mixed(corMat,tl.pos = "lt")
View(KSdata)

#removing correlated variable #pledged

KSdata <- KSdata[, -c(16)]
```

Step 7

Text Mining/Sentiment Analysis (Bing – Lexicon Approach)

```
#Text Mining
kick_data<-KSdata
kick_data<-kick_data[,-c(2,4:18)]

kick_data<-cSplit(kick_data,"blurb", sep =" ", direction ="long")

# separating special characters
kick_data<-cSplit(kick_data,"blurb", sep ="/ ", direction ="long")
kick_data

names(kick_data)<-c("state","word")

#cleaning of special characters
kick_data$word<-gsub("$","", kick_data$word)
kick_data$word<-gsub(":","", kick_data$word,fixed = TRUE)

kick_databing<-kick_data%>%
  inner_join((get_sentiments("bing")))

kick_databing %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D","#00BFC4"),
                   max.words = 500)

kick_datacount <- kick_databing%>% count(word,sentiment,sort = TRUE)

kick_datacount %>%
  group_by(sentiment) %>%
  top_n(20) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()

#remove irrelevant column #blurb
KSdata <- KSdata[, -c(3)]

KSdata <- KSdata[, -c(5,9,14)]
```

Step 8

Feature Selection using Boruta

```
set.seed(111)

boruta<- Boruta(state~.,data = KSdata,doTrace = 2)

print(boruta)
```

```
plot(boruta)

view(KSdata)

##Removing columns irrelevant to the analysis #boruta suggestion
KSdata <- KSdata[, -c(6,7,10)]

str(KSdata)
```

Step 9

```
Splitting Data into Train and Test

##Splitting the data


set.seed(123)

index <- createDataPartition(y = KSdata$state, p = 0.7, list = F)

train_KSdata <- KSdata[index, ]

test_KSdata<- KSdata[-index, ]

#data balancing for train data
prop.table(table(train_KSdata$state))
```

Step 10

Model Evaluation (Cross Validation and Confusion Matrix) and Modeling

Models

- Random Forest
- Support Vector Machine (SVM) Linear Kernel
- Decision Tree
- Naïve Bayes

```
#k-fold cross validation

set.seed(123)

ctrl<-trainControl(method = "repeatedcv",
                   number = 10,
                   savePredictions = TRUE)
mod_fit<-train(state ~ goal+backers_count+duration+staff_pick,
```

```
                method ="rf",
                data=train_KSdata,
                metric ="Accuracy",
                tuneGrid =NULL,
                trControl = ctrl)


#test data

pred = predict(mod_fit,newdata=test_KSdata)

confusionMatrix(pred,test_KSdata$state)

#AUC
auc(test_KSdata$state,pred)
```

Random Forest

```
#SVM #Support Vector Machine with Linear Kernel

#getting SVM Linear Output

ctrl<-trainControl(method = "repeatedcv",
                   number = 10,
                   savePredictions = TRUE)

svm_Linear <- train(state ~ goal+backers_count+duration+staff_pick,
                    data = train_KSdata,
                    method = "svmLinear",
                    trControl=ctrl,
                    PreProcess = c("center", "scale"),
                    tuneLength = 10)

print(svm_Linear)

#getting test output

test_pred <- predict(svm_Linear, newdata = test_KSdata)

test_pred

confusionMatrix(table(test_pred, test_KSdata$state))

#AUC
auc(test_KSdata$state,test_pred)
```

Decision Tree

```
prop.table(table(train_KSdata$state))
prop.table(table(test_KSdata$state))

ctrl<-trainControl(method = "repeatedcv",
                   number = 10,
```

```
                    savePredictions = TRUE)

rpart.cv <- train(state ~ goal+backers_count+duration+staff_pick,
                  data = train_KSdata,
                  method = "rpart",
                  trControl = ctrl,
                  tuneLength = 15)

model<-rpart(state~goal+backers_count+duration+staff_pick,
             data= train_KSdata, method="class")

rpart.plot(model)

#test data

m<-predict(model,test_KSdata,type="class")


tm<-table(test_KSdata$state,m)

tm

accuracy<-sum(diag(tm))/sum(tm)

print(paste("Percentage Accuracy",accuracy))

confusionMatrix(data = m,test_KSdata$state)

#AUC
auc(test_KSdata$state,m)
```

Naïve Bayes

```
Ctrl<-trainControl(method = "repeatedcv",
                   number = 10,
                   savePredictions = TRUE)
model1<-naive_bayes(state~goal+backers_count+duration+staff_pick,
                    data=train_KSdata,
                    usekernel=T)

#model1 ##prediction

#train data
p1<-predict(model1,type = "prob")

N_pred<-predict(model1,train_KSdata)

tab1<-table(N_pred,train_KSdata$state)

1-sum(diag(tab1))/sum(tab1)

confusionMatrix(data = N_pred,train_KSdata$state)
```

```
#test data

Actual_pred<-predict(model1,test_KSdata)

(tab2<-table(Actual_pred,test_KSdata$state))

1-sum(diag(tab2))/sum(tab2)

confusionMatrix(data = Actual_pred,test_KSdata$state)

#AUC
auc(test_KSdata$state,Actual_pred)
```

Step 11

Statistical Distribution and Analysis

```
#checking relationship of variables using Pearson's Chi-squared test

chisq.test(KSdata$state,KSdata$duration)
#p-value < 0.05 #significance level

chisq.test(KSdata$state,KSdata$backers_count)
#p-value < 0.05 #significance level

chisq.test(KSdata$state,KSdata$staff_pick)
#p-value < 0.05 #significance level

chisq.test(KSdata$state,KSdata$goal)
#p-value < 0.05 #significance level


#Descriptive Statistics

KSdata<- rnorm(6525, mean = 0, sd = 1)
mean(KSdata)
sd(KSdata)
descriptives <- function(KSdata) {
  u <- length(unique(KSdata))
  missing <- sum(is.na(KSdata)) / length(KSdata) * 100
  quantiles <- quantile(KSdata, na.rm = T)
  av <- mean(KSdata, na.rm = T)
  stdev <- sd(KSdata, na.rm = T)
  df <- data.frame(u, missing, quantiles[1], quantiles[2],
                   quantiles[3], quantiles[4], quantiles[5], av, stdev)
  names(df) <- c("no. unique", "% missing", "min", "first quartile",
                 "median", "third quartile", "max", "mean", "sd")
  rownames(df) <- NULL
  return(df)
}
df <- descriptives(KSdata)


#exploration
```

```
#histogram
hist(KSdata)

#distribution curve
hist(KSdata,freq=FALSE,col="Gray",xlab="KSdata",main="Distribtion Curve")
curve(dnorm(x,mean=mean(KSdata),sd=sd(KSdata)),add=TRUE,col="red")
```