**SCRIPT : -**

**# Calling librares**

**> library(twitteR)**

**> library(ggplot2)**

**> library(plyr)**

**> library(stringr)**

**> library(tm)**

**> library(RColorBrewer)**

**> library(ROAuth)**

**> library(RCurl)**

**> options(RCurlOptions = list(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl")))**

**> require(twitteR)**

**> requestURL <- "https://api.twitter.com/oauth/request\_token"**

**> accessURL <- "https://api.twitter.com/oauth/access\_token"**

**> authURL <- "https://api.twitter.com/oauth/authorize"**

**> consumerKey <- #<consumer key obtained from twitter api when you create your app>**

**> consumerSecret <- #<consumer secret key obtained from twitter api when you create your app>**

**> cred <- OAuthFactory$new(consumerKey = consumerKey, consumerSecret = consumerSecret, requestURL = requestURL, accessURL = accessURL, authURL = authURL)**

**> accessToken <- #< accessToken obtained from twitter api when you create your app>**

**> accessTokenSecret <- #<access token secret key obtained from twitter api when you create your app>**

**> setup\_twitter\_oauth(consumerKey, consumerSecret, accessToken, accessTokenSecret)**

**> baltimoreU <- searchTwitter('#BaltimoreUprising', n = 2000) #Reading tweets from twitter**

**> bu\_df <- twListToDF(baltimoreU)**

**> write.csv(bu\_df, file="bu.csv", row.names=FALSE)**

**> library(maptools)**

**> install.packages("classInt")**

**> colors <- brewer.pal(9, "YlOrRd")**

**> baltimore.shp <- readShapePoly("nhood\_2010.shp") #Reading Shapefile**

**> nbr\_desc <- baltimore.shp@data$LABEL**

**#Reading excel files into R.**

**> crime\_b <- read.csv(file="Crime\_by\_Neighborhood.csv", header=TRUE)**

**> crime\_nbr <- crime\_b[,c(1:4,10)] #Extracting only selected columns**

**# Using ddply function from library 'plyr' to group by data**

**> crime\_nbr2 <- ddply(crime\_nbr, c("CrimeYear", "Neighborhood"), summarize, count = sum(!is.na(c(CrimeYear,Neighborhood))))**

**> crime\_nbr1 <- ddply(crime\_nbr, c("CrimeMonthYear", "Neighborhood"), summarize, count = sum(!is.na(c(CrimeMonthYear,Neighborhood)))**

**> crime\_nbr <- ddply(crime\_b, c("CrimeMonthYear", "Neighborhood"), summarize, count = sum(!is.na(Neighborhood))))**

**> crime\_victim <- read.csv(file="BPD\_Part\_1\_Victim\_Based\_Crime\_Data.csv", header = TRUE)**

**> write.csv(nbr\_desc, file="nbr\_desc.csv", row.names=F)**

**> nbr\_desc <- unique(crime\_victim[,9]) # Selecting uniques neighborhood names**

**> nbr\_desc <- nbr\_desc[order(nbr\_desc)] # Sorting names by A-Z**

**> nbr\_desc <- data.frame(nbr\_desc)**

**> nbr\_desc<- nbr\_desc[-1, ]**

**> baltimore\_crm <- ddply(crime\_b, c("Neighborhood"), summarize, count = sum(!is.na(Neighborhood)))**

**> baltimore\_crm <- baltimore\_crm[-1, ]**

**# Renaming names of all the columns in crime\_nbr2**

**# subsetting crm\_nbr2 by year**

**# Join created tables from crm\_nbr2 and joining with nbr\_desc to maintain consisteny of data**

**> names(crime\_nbr2) <- c("YEAR", "LABEL", "COUNT")**

**> crm\_2010 <- subset(crime\_nbr2, YEAR == 2010 & LABEL !="", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2010 <- merge(x = crm\_2010, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**> crm\_2011 <- subset(crime\_nbr2, YEAR == 2011 & LABEL !="", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2011 <- merge(x = crm\_2011, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**> crm\_2012 <- subset(crime\_nbr2, YEAR == 2012 & LABEL !="", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2012 <- merge(x = crm\_2012, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**> crm\_2013 <- subset(crime\_nbr2, YEAR == 2013 & LABEL !="", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2013 <- merge(x = crm\_2013, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**> crm\_2014 <- subset(crime\_nbr2, YEAR == 2014 & LABEL !="", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2014 <- merge(x = crm\_2014, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**> crm\_2015 <- subset(crime\_nbr2, YEAR == 2015 & LABEL != "", select=c("YEAR", "LABEL", "COUNT"))**

**> crm\_2015 <- merge(x = crm\_2015, y = nbr\_desc, by = "LABEL", all.y=TRUE)**

**# Generating multiple copies of shapefile for each year**

**# Merging data of each year in to individual shape file for generating map plots**

**> bshp\_2010 <- baltimore.shp**

**> bshp\_2010@data <- data.frame(bshp\_2010@data, crm\_2010[match(bshp\_2010@data$LABEL, crm\_2010$LABEL),])**

**> bshp\_2011 <- baltimore.shp**

**> bshp\_2011@data <- data.frame(bshp\_2011@data, crm\_2011[match(bshp\_2011@data$LABEL, crm\_2011$LABEL),])**

**> bshp\_2012 <- baltimore.shp**

**> bshp\_2012@data <- data.frame(bshp\_2012@data, crm\_2012[match(bshp\_2012@data$LABEL, crm\_2012$LABEL),])**

**> bshp\_2013 <- baltimore.shp**

**> bshp\_2013@data <- data.frame(bshp\_2013@data, crm\_2013[match(bshp\_2013@data$LABEL, crm\_2013$LABEL),])**

**> bshp\_2014 <- baltimore.shp**

**> bshp\_2014@data <- data.frame(bshp\_2014@data, crm\_2014[match(bshp\_2014@data$LABEL, crm\_2014$LABEL),])**

**> bshp\_2015 <- baltimore.shp**

**> bshp\_2015@data <- data.frame(bshp\_2015@data, crm\_2015[match(bshp\_2015@data$LABEL, crm\_2015$LABEL),])**

**> bshp <- baltimore.shp**

**> names(baltimore\_crm) <- c("LABEL", "COUNT", "PERCENTILE", "Rank")**

**> bshp@data <- data.frame(bshp@data, baltimore\_crm[match(bshp@data$LABEL, baltimore\_crm$LABEL),])**

**# Writing files into csv**

**> write.csv(crm\_2010, file="crm\_2010.csv")**

**> write.csv(crm\_2011, file="crm\_2011.csv")**

**> write.csv(crm\_2012, file="crm\_2012.csv")**

**> write.csv(crm\_2013, file="crm\_2013.csv")**

**> write.csv(crm\_2014, file="crm\_2014.csv")**

**> write.csv(crm\_2015, file="crm\_2015.csv")**

**> write.csv(baltimore\_crm, file="baltimore\_crm.csv")**

**# reading th csv files into R**

**> crm\_2010 <- read.csv(file = "crm\_2010.csv", header = TRUE)**

**> crm\_2011 <- read.csv(file = "crm\_2011.csv", header = TRUE)**

**> crm\_2012 <- read.csv(file = "crm\_2012.csv", header = TRUE)**

**> crm\_2013 <- read.csv(file = "crm\_2013.csv", header = TRUE)**

**> crm\_2014 <- read.csv(file = "crm\_2014.csv", header = TRUE)**

**> crm\_2015 <- read.csv(file = "crm\_2015.csv", header = TRUE)**

**> baltimore\_crm <- read.csv(file = "baltimore\_crm.csv", header = TRUE)**

**# Creating class intervals**

**> brks2010<-classIntervals(crm\_2010$Rank, n=6, style="quantile")**

**> brks2011<-classIntervals(crm\_2011$Rank, n=6, style="quantile")**

**> brks2012<-classIntervals(crm\_2012$Rank, n=6, style="quantile")**

**> brks2013<-classIntervals(crm\_2013$Rank, n=6, style="quantile")**

**> brks2014<-classIntervals(crm\_2013$Rank, n=6, style="quantile")**

**> brks2015<-classIntervals(crm\_2015$Rank, n=6, style="quantile")**

**> brks<-classIntervals(baltimore\_crm$Rank, n=5, style="quantile")**

**> brks2010 <- brks2010$brks**

**> brks2011 <- brks2011$brks**

**> brks2012 <- brks2012$brks**

**> brks2013 <- brks2013$brks**

**> brks2014 <- brks2014$brks**

**> brks2015 <- brks2015$brks**

**> brks <- brks$brks**

**# Plotting crime data for each year**

**> plot(bshp, col=colors[findInterval(bshp@data$Rank, brks,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2010-2015"))**

**> legend("bottomleft", legend = unique(bshp@data$Rank[order(bshp@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2010, col=colors[findInterval(bshp\_2010@data$Rank, brks2010,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2010"))**

**> legend("bottomleft", legend = unique(bshp\_2010@data$Rank[order(bshp\_2010@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2011, col=colors[findInterval(bshp\_2011@data$Rank, brks2011,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2011"))**

**> legend("bottomleft", legend = unique(bshp\_2011@data$Rank[order(bshp\_2011@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2012, col=colors[findInterval(bshp\_2012@data$Rank, brks2012,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2012"))**

**> legend("bottomleft", legend = unique(bshp\_2012@data$Rank[order(bshp\_2012@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2013, col=colors[findInterval(bshp\_2013@data$Rank, brks2013,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2013"))**

**> legend("bottomleft", legend = unique(bshp\_2013@data$Rank[order(bshp\_2013@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2014, col=colors[findInterval(bshp\_2014@data$Rank, brks2014,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2014"))**

**> legend("bottomleft", legend = unique(bshp\_2014@data$Rank[order(bshp\_2014@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**> plot(bshp\_2015, col=colors[findInterval(bshp\_2015@data$Rank, brks2015,all.inside=TRUE)], axes=F)**

**> title(paste ("Baltimore Crime Classification 2015"))**

**> legend("bottomleft", legend = unique(bshp\_2015@data$Rank[order(bshp\_2015@data$Rank)]), title = "Rank", fill = colors, cex = .90, bty = "n")**

**# Crating score sentiment function for sentiment analysis**

**>score.sentiment = function(sentences, pos.words, neg.words, .progress='none')**

**{**

**require(plyr)**

**require(stringr)**

**# we got a vector of sentences. plyr will handle a list**

**# or a vector as an "l" for us**

**# we want a simple array ("a") of scores back, so we use**

**# "l" + "a" + "ply" = "laply":**

**scores = laply(sentences, function(sentence, pos.words, neg.words) {**

**# clean up sentences with R's regex-driven global substitute, gsub():**

**sentence = gsub('[[:punct:]]', '', sentence)**

**sentence = gsub('[[:cntrl:]]', '', sentence)**

**sentence = gsub('\\d+', '', sentence)**

**# and convert to lower case:**

**sentence = tolower(sentence)**

**# split into words. str\_split is in the stringr package**

**word.list = str\_split(sentence, '\\s+')**

**# sometimes a list() is one level of hierarchy too much**

**words = unlist(word.list)**

**# compare our words to the dictionaries of positive & negative terms**

**pos.matches = match(words, pos.words)**

**neg.matches = match(words, neg.words)**

**# match() returns the position of the matched term or NA**

**# we just want a TRUE/FALSE:**

**pos.matches = !is.na(pos.matches)**

**neg.matches = !is.na(neg.matches)**

**# and conveniently enough, TRUE/FALSE will be treated as 1/0 by sum():**

**score = sum(pos.matches) - sum(neg.matches)**

**return(score)**

**}, pos.words, neg.words, .progress=.progress )**

**scores.df = data.frame(score=scores, text=sentences)**

**return(scores.df)**

**}**

**# load positive and negative words for sentiment analysis**

**> pos <- scan('positive\_words.txt', what='character', comment.char = ';')**

**Read 2006 items**

**> neg <- scan('negative\_words.txt', what='character', comment.char = ';')**

**Read 4783 items**

**# add words to a list**

**> pos.words <- c(pos, 'upgrade')**

**> neg.words <- c(neg, 'wtf', 'wait', 'waiting', 'epicfall', 'mechanical')**

**# import csv file**

**> ds\_bu <- read.csv(file = "bu.csv", header = TRUE)**

**> ds\_bu$text <- as.factor(ds\_bu$text)**

**# Score all tweets**

**> n.tweets <- length(baltimoreU)**

**> bu.scores <- score.sentiment(ds\_bu$text, pos.words, neg.words, .progress = 'text')**

**> write.csv(bu.scores, file=paste("buScores.csv", sep = ""), row.names=T)**

**# plotting using qplot**

**> hist(bu.scores$score, xlab = "Score of Tweets", col=brewer.pal(9, "Set3"))**

**> qplot(bu.scores$score, xlab = "Score of Tweets")**

**# text analysis via data mining**

**> library(tm)**

**> install.packages("wordcloud")**

**> library(wordcloud)**

**> myCorpus <- Corpus(VectorSource(bu\_df$text))**

**> myCorpus <- tm\_map(myCorpus, stripWhitespace)**

**> myCorpus <- tm\_map(myCorpus, removePunctuation)**

**> myCorpus <- tm\_map(myCorpus, removeNumbers)**

**> myCorpus <- tm\_map(myCorpus, tolower)**

**> myCorpus <- tm\_map(myCorpus, removeWords, myStopwords)**

**> myCorpus <- tm\_map(myCorpus, stemDocument)**

**> myCorpus <- tm\_map(myCorpus, stemDocument)**

**> myCorpus <- tm\_map(myCorpus, PlainTextDocument)**

**> tdm <- TermDocumentMatrix(myCorpus, control = list(wordLengths = c(1, Inf)))**

**> freq.terms1 <- findFreqTerms(tdm, lowfreq = 15)**

**> term.freq1 <- rowSums(as.matrix(tdm))**

**> term.freq1 <- subset(term.freq1, term.freq1>=15)**

**> df1 <- data.frame(term = names(term.freq1), freq = term.freq1)**

**> library(ggplot2)**

**> ggplot(df1, aes(x = term, y = freq)) + geom\_bar(stat = "identity") + xlab("Terms") + ylab("Count") + coord\_flip()**

**> findAssocs(tdm, "baltimoreuprising", 0.2)**

**baltimoreuprising.deray**

**0.22**

**> findAssocs(tdm, "protestors", 0.2)**

**protestors**

**lovemyopinions 0.94**

**hall 0.84**

**httptcoxgwqsyvdiu 0.82**

**demanding 0.69**

**released 0.66**

**city 0.48**

**demonstration 0.47**

**httptcok… 0.47**

**lnonblonde 0.47**

**dviyer 0.33**

**now 0.33**

**petition 0.33**

**signed 0.33**

**turn 0.33**

**vs 0.33**

**# Downloading libraries 'graph', 'Rgraphviz', 'Rcpp', 'wordcloud', 'RColorBrewer'**

**> source("http://bioconductor.org/biocLite.R")**

**> biocLite("graph")**

**> biocLite("Rgraphviz")**

**> plot(tdm, term = freq.terms1, corThreshold = 0.12, weighting = T)**

**> library(wordcloud)**

**> install.packages("Rcpp")**

**> library(Rcpp)**

**> install.packages("wordcloud")**

**> library(wordcloud)**

**> install.packages("RColorBrewer")**

**> library(RColorBrewer)**

**> m <- as.matrix(tdm)**

**> word.freq <- sort(rowSums(m), decreasing = T) # Sorting words**

**# Generating wordcloud**

**> wordcloud(words = names(word.freq), freq = word.freq, min.freq = 3, random.order = F)**

**> library(tm)**

**> tdm2 <- removeSparseTerms(tdm, sparse = 0.95)**

**> m2 <- as.matrix(tdm2)**

**> distMatrix <- dist(scale(m2))**

**> fit <- hclust(distMatrix, method = "ward")**

**The "ward" method has been renamed to "ward.D"; note new "ward.D2"**

**> fit <- hclust(distMatrix, method = "ward.D")**

**> plot(fit)**

**> rect.hclust(fit, k = 6)**

**> m3 <- t(m2)**

**> set.seed(22)**

**> k <- 6**

**> install.packages("fpc")**

**> library(fpc)**

**> pamResult <- pamk(m3, metric="manhattan")**

**> k <- pamResult$ncpam**

**> pamResult <- pamResult$pamobject**

**> layout(matrix(c(1,2), 1, 2))**

**> plot(pamResult, col.p = pamResult$clustering)**

**> install.packages("topicmodels")**

**> library(topicmodels)**

**> lda <- LDA(dtm, k = 8)**

**> term <- terms(lda, 4)**

**> term**

**Topic 1 Topic 2 Topic 3 Topic 4 Topic 5**

**[1,] "rt" "rt" "rt" "baltimoreuprising" "baltimoreuprising"**

**[2,] "baltimoreuprising" "deray" "baltimoreuprising" "baltimoreriots" "rt"**

**[3,] "baltimore" "baltimoreuprising" "aimnplease" "rt" "baltimore"**

**[4,] "state" "httptcosyxujdhsj" "image" "blacklivesmatter" "baltimoreupris"**

**Topic 6 Topic 7 Topic 8**

**[1,] "rt" "baltimoreuprising" "rt"**

**[2,] "baltimoreuprising" "rt" "baltimoreuprising"**

**[3,] "deray" "freddiegray" "america"**

**[4,] "baltimorepolice" "protest" "bipartisanism"**

**> term <- apply(term, MARGIN = 2, paste, collapse = ", ")**

**> term**

**Topic 1**

**"rt, baltimoreuprising, baltimore, state"**

**Topic 2**

**"rt, deray, baltimoreuprising, httptcosyxujdhsj"**

**Topic 3**

**"rt, baltimoreuprising, aimnplease, image"**

**Topic 4**

**"baltimoreuprising, baltimoreriots, rt, blacklivesmatter"**

**Topic 5**

**"baltimoreuprising, rt, baltimore, baltimoreupris"**

**Topic 6**

**"rt, baltimoreuprising, deray, baltimorepolice"**

**Topic 7**

**"baltimoreuprising, rt, freddiegray, protest"**

**Topic 8**

**"rt, baltimoreuprising, america, bipartisanism"**

**> topic <- topics(lda, 1)**

**> topics <- data.frame(date = as.IDate(bu\_df$created), topic)**

**> library(ggplot2)**

**> qplot(date, ..count.., data = topics, geom="density", fill=term[topic], position="stack")**

**> mapdata <- read.csv(file="Officer\_Involved\_Shooting\_View.csv", header =TRUE)**

**> mapdata <- ddply(off\_shoot, c("CrimeYear", "Neighborhood"), summarize, count = sum(!is.na(c(CrimeYear,Neighborhood))))**

**> library(plyr)**

**> off\_shoot1 <- ddply(off\_shoot, c("Year", "X..LONG", "Y..LAT"), summarize, count = sum(!is.na(c(Year,X..LONG, Y..LAT))))**

**> install.packages("rworldmap")**

**> library(rworldmap)**

**> newmap <- getMap(resolution = "low")**

**> plot(newmap)**

**> plot(newmap, xlim = range(off\_shoot1$X..LONG.), ylim = range(off\_shoot1$Y..LAT.), asp = 1)**

**> library("maps")**

**# Creating ggplot**

**> ggplot() + geom\_point(data=off\_shoot1, aes(x=X..LONG., y=Y..LAT., color = "red"))**

**> ggplot() + geom\_polygon(data=baltimore.shp, aes(x=long, y=lat, group=group))**

**Regions defined for each Polygons**

**> ggplot()+geom\_polygon(data=baltimore.shp, aes(x=long, y=lat, group=group)) + geom\_point(data=off\_shoot1, aes(x=X..LONG., y=Y..LAT.), color="red")**

**# Renaming specific fields of mapdata**

**> names(mapdata)[names(mapdata)=="X..LONG."]<-"x"**

**> names(mapdata)[names(mapdata)=="Y..LAT."]<-"y"**

**> ggplot() +geom\_polygon(data=baltimore.shp, aes(x=long, y=lat, group=group))+ geom\_point(data=mapdata, aes(x=x, y=y), color="red")**

**> library(rgdal)**

**> neighborhoods <- readOGR(".", "nhood\_2010")**

**> neighborhoods <- spTransform(neighborhoods, CRS("+proj=longlat +datum=WGS84"))**

**> library(ggplot2)**

**> neighborhoods <- fortify(neighborhoods)**

**# Using Rgoogle maps to plot offcier shooting for years 2013-2015 in baltimore**

**> install.packages("RgoogleMaps")**

**> library(ggmap)**

**Google Maps API Terms of Service: http://developers.google.com/maps/terms.**

**Please cite ggmap if you use it: see citation('ggmap') for details.**

**> library(RgoogleMaps)**

**> CenterOfMap <- geocode("Baltimore, MD")**

**> CenterOfMap <- geocode(" 39.299768,-76.614929")**

**> Baltimore <- get\_map(c(lon=CenterOfMap$lon, lat=CenterOfMap$lat),zoom = 12, maptype = "terrain", source = "google")**

**> Baltimore <- get\_map(c(lon=CenterOfMap$lon, lat=CenterOfMap$lat),zoom = 12, maptype = "toner", source = "stamen")**

**> BaltimoreMap <- ggmap(Baltimore)**

**> BaltimoreMap**

**> Baltimore <- get\_map(c(lon=CenterOfMap$lon, lat=CenterOfMap$lat),zoom = 12,source = "osm")**

**> BaltimoreMap <- ggmap(Baltimore)**

**> BaltimoreMap**

**> BaltimoreMap <- BaltimoreMap + geom\_polygon(aes(x=long, y=lat, group=group), fill='grey', size=.2,color='black', data=neighborhoods, alpha=0) + geom\_point(data=mapdata, aes(x=x, y=y), colour="Red", fill="Red",pch=21, size=9, alpha=I(0.7))**