Finance

library(mongolite)

library(twitteR)

library(stringr)

library(tm)

library(plyr)

library(rjson)

library(httr)

library(RJSONIO)

users <- mongo(collection = "users", db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

user\_names <- mongo(collection = "user\_info", db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

currentDate<-Sys.Date()

month<-format(currentDate,"%m")

year<-format(currentDate,"%Y")

#month <- 10

#year <- 2016

###########################################Financial data###############################################

update\_bank\_summary <- function(name) {

account <- mongo(collection = name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

each\_bank\_balance <- account$aggregate(

'[{"$match":{"account\_number": { "$exists": true}}},{"$group":{"\_id":"$bank\_name", "balance": {"$last":"$balance"}}}]'

)

print(each\_bank\_balance)

if(dim(each\_bank\_balance)[1] > 0 && dim(each\_bank\_balance)[2] > 0)

{

sum = 0

for(balance in each\_bank\_balance[,2]) {

sum <- sum + balance

}

no\_of\_accounts <- dim(each\_bank\_balance)[1]

print(no\_of\_accounts)

users$update(query = paste0('{"name":"', name,'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"no\_of\_accounts": ', no\_of\_accounts, '}}'))

}

each\_accounts\_balance <- account$aggregate(

'[{"$match":{"account\_number": { "$exists": true}}},{"$group":{"\_id":"$account\_number", "balance": {"$last":"$balance"}}}]'

)

if(dim(each\_accounts\_balance)[1] > 0 && dim(each\_accounts\_balance)[2] > 0)

{

sum = 0

for(balance in each\_accounts\_balance[,2]) {

sum <- sum + balance

}

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"total\_bank\_balance": ', sum, '}}'))

no\_of\_accounts <- dim(each\_accounts\_balance)[1]

print(no\_of\_accounts)

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"no\_of\_accounts": ', no\_of\_accounts, '}}'))

}

}

update\_banks <- function(name) {

account <- mongo(collection = name , db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

each\_account\_credit\_summary <- account$aggregate(

paste0('[ {"$match": {"account\_number": { "$exists": true}, "type":"credit", "month" : ', as.integer(month) , ',"year" :', year, '}},

{"$group":{"\_id":{"account\_number":"$account\_number"}, "amount": {"$sum":"$amount"}, "number\_of\_transcations": {"$sum":1} }}]')

)

each\_account\_debit\_summary <- account$aggregate(

paste0('[ {"$match": {"account\_number": { "$exists": true}, "type":"debit", "month" : ', as.integer(month) , ',"year" :', year, '}},

{"$group":{"\_id":{"account\_number":"$account\_number"}, "amount": {"$sum":"$amount"}, "number\_of\_transcations": {"$sum":1} }}]')

)

each\_account\_balance\_bankname <- account$aggregate(

'

[ {"$match": {"account\_number": { "$exists": true}}},

{"$group":{"\_id":{"account\_number":"$account\_number"}, "balance": {"$last":"$balance"},"bank\_name": {"$last":"$bank\_name"} }}]

'

)

bank\_collection\_name <- paste(name,"\_summary",sep="")

summary\_col <- mongo(collection = bank\_collection\_name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

#print(each\_account\_balance\_bankname)

#print(each\_account\_credit\_summary)

#print(each\_account\_debit\_summary)

nrows <- nrow(each\_account\_balance\_bankname)

for(i in 1:nrows){

account\_number <- each\_account\_balance\_bankname[i,1]

balance <- each\_account\_balance\_bankname[i,2]

bank\_name <- each\_account\_balance\_bankname[i,3]

summary\_col$update(query = paste0('{"account\_number":"', account\_number[[1]],'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"type" : "account", "balance": ', balance , ',"bank\_name":"', bank\_name,'"}}'), upsert = TRUE)

}

nrows <- nrow(each\_account\_credit\_summary)

if(nrows > 0) {

for(i in 1:nrows){

account\_number <- each\_account\_credit\_summary[i, 1]

credit\_amount <- each\_account\_credit\_summary[i,2]

credit\_no\_of\_transactions <- each\_account\_credit\_summary[i,3]

#print(account\_number)

#print(credit\_no\_of\_transactions)

summary\_col$update(query = paste0('{"account\_number":"', account\_number[[1]],'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"total\_credit\_amount": ', credit\_amount , ',"total\_credit\_transactions":', credit\_no\_of\_transactions,'}}'), upsert = TRUE)

}

}

nrows <- nrow(each\_account\_debit\_summary)

if(nrows > 0) {

for(i in 1:nrows){

account\_number <- each\_account\_debit\_summary[i, 1]

debit\_amount <- each\_account\_debit\_summary[i,2]

debit\_no\_of\_transactions <- each\_account\_debit\_summary[i,3]

#print(account\_number)

#print(credit\_no\_of\_transactions)

summary\_col$update(query = paste0('{"account\_number":"', account\_number[[1]],'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"total\_debit\_amount": ', debit\_amount , ',"total\_debit\_transactions":', debit\_no\_of\_transactions,'}}'), upsert = TRUE)

}

}

}

update\_finance\_credit\_debit <- function(name, twitter\_handle) {

#users$update( query = paste0('{"name":"', name,'"}'), update = paste0('{"$set":{"tweets\_sentiment": ', get\_tweets\_sentiment(twitter\_handle), '}}'))

account <- mongo(collection = name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

transactions\_info\_credit <- account$aggregate(

paste0( '[{"$match":{"type":"credit", "month" : ', as.integer(month) , ',"year" :', year, '}},{"$group":{"\_id":"$type", "count": {"$sum":1}, "amount":{"$sum":"$amount"},"average":{"$avg":"$amount"}}}]' )

)

#, "month" : "10", "year" : "2016"

transactions\_info\_debit <- account$aggregate(

paste0( '[{"$match":{"type":"debit", "month" : ', as.integer(month) , ',"year" :', year, '}},{"$group":{"\_id":"$type", "count": {"$sum":1}, "amount":{"$sum":"$amount"},"average":{"$avg":"$amount"}}}]' )

)

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"twitter\_handle":"', twitter\_handle, '"}}'), upsert = TRUE)

if(dim(transactions\_info\_credit)[1] > 0 && dim(transactions\_info\_credit)[2] > 0)

{

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"number\_transactions\_credit": ', transactions\_info\_credit[2], '}}'), upsert = TRUE)

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"totalamount\_transactions\_credit": ', transactions\_info\_credit[3], '}}'), upsert = TRUE)

users$update(query = paste0('{"name":"', name, '","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"averageamount\_transactions\_credit": ', transactions\_info\_credit[4], '}}'), upsert = TRUE)

}

if(dim(transactions\_info\_debit)[1] > 0 && dim(transactions\_info\_debit)[2] > 0)

{

users$update(query = paste0('{"name":"', name,'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"averageamount\_transactions\_debit": ', transactions\_info\_debit[4], '}}'), upsert = TRUE)

users$update(query = paste0('{"name":"', name,'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"number\_transactions\_debit": ', transactions\_info\_debit[2], '}}'), upsert = TRUE)

users$update(query = paste0('{"name":"', name,'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"totalamount\_transactions\_debit": ', transactions\_info\_debit[3], '}}'), upsert = TRUE)

}

}

update\_location\_data <- function(name) {

account <- mongo(collection = name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

city\_not\_fetched <- account$aggregate(

'

[{"$match":{"type":"location", "country": {"$exists": false}}}]

'

)

if(dim(city\_not\_fetched)[1] > 0 && dim(city\_not\_fetched)[2] > 0)

{

nrows <- nrow(city\_not\_fetched)

for(i in 1:nrows){

print(city\_not\_fetched)

latitude <- city\_not\_fetched[i,3]

longitude <- city\_not\_fetched[i,4]

latlon <- paste(latitude,',',longitude,sep="")

print(latlon)

API\_Key <- "AIzaSyAZI9adLQIuSi2-YESdSwlqJj087KH3DrU"

#Google Maps API key skylinelabs account AIzaSyAZI9adLQIuSi2-YESdSwlqJj087KH3DrU

connectStr <- paste("https://maps.google.com/maps/api/geocode/json?key=",API\_Key,"&latlng=",latlon, sep="")

con <- url(connectStr)

data.json <- fromJSON(paste(readLines(con), collapse=""))

close(con)

#print(data.json)

country <- ""

city <- ""

i <- 0

nrows <- lengths(data.json)

for(i in 1:nrows){

type <- data.json$results[[i]]$types[1]

if(identical(type,"country")){

country <- data.json$results[[i]]$address\_components[[1]]$long\_name[1]

print(country)

}

if(identical(type,"locality")){

city <- data.json$results[[i]]$address\_components[[2]]$long\_name[1]

print(city)

}

if(identical(city,"")){

city <- data.json$results[[i]]$address\_components[[3]]$long\_name[1]

print(city)

}

}

query <- account$update(query = paste0('{"latitude":', latitude,',"longitude":',longitude,'}'), update = paste0('{"$set":{"city":"', city, '","country":"',country,'"}}'), upsert=TRUE, multiple=TRUE )

#m <- mongo(collection = bank\_collection\_name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

#m$update(query = paste0('{"account\_number":"', account\_number,'","type":"',transaction\_type,'"}'), update = paste0('{"$set":{"amount": ', amount, ',"transactions\_number":', numer\_of\_transactions, ',"balance":', balance, ',"bank\_name":"', bank,'"}}'),upsert = TRUE)

}

}

}

update\_summary\_location\_data <- function (name) {

collection\_name <- paste(name,"\_summary",sep="")

summary\_collection <- mongo(collection = collection\_name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

name\_collection <- mongo(collection = name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

city\_wise\_location <- name\_collection$aggregate(

paste0('[ {"$match": {"type":"location", "month" : ', as.integer(month) , ',"year" :', year, '}},

{"$group":{"\_id":{"city":"$city", "country":"$country"}, "number": {"$sum":1} }}]')

)

print(city\_wise\_location)

nrows <- nrow(city\_wise\_location)

if(nrows > 0) {

for(i in 1:nrows){

city <- city\_wise\_location[i,1][1]

country <- city\_wise\_location[i,1][2]

number <- city\_wise\_location[i,2]

print(city[[1]])

print(country[[1]])

print(number)

summary\_collection$update(query = paste0('{"city":"', city[[1]],'","country":"',country[[1]],'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"type" : "location", "number": ', number ,'}}'), upsert = TRUE)

}

}

}

update\_loan\_payment\_data <- function(name) {

usercol <- mongo(collection = "users", db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

payment\_collection\_name <- paste(name,"\_payments",sep="")

pay\_col <- mongo(collection = payment\_collection\_name, db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

payments\_info <- pay\_col$aggregate(

'[{"$match":{"type":"EMI" }},{"$group":{"\_id":"$type","sum\_default": {"$sum": "$default\_payment"}, "sum\_timely":{"$sum":"$timely\_payment"}}}]'

)

print(payments\_info)

nrows <- nrow(payments\_info)

if(nrows == 1) {

ndefaults <- payments\_info[1,2]

ntimely <- payments\_info[1,3]

loan\_hist\_score <- as.integer(ntimely) / (as.integer(ntimely) + as.integer(ndefaults)) \* 100

print(loan\_hist\_score)

usercol$update(query = paste0('{"name":"', name,'","month" : ', as.integer(month) , ',"year" :', year, '}'), update = paste0('{"$set":{"loan\_history\_score": ', loan\_hist\_score, '}}'), upsert = TRUE)

}

}

print\_userdb <- function(){

users\_info <- users$find()

print(users\_info)

}

update\_all\_data <- function() {

users\_info <- user\_names$find()

for (i in 1:nrow(users\_info)) {

name <- users\_info[i, "name"]

twitter\_handle <- users\_info[i, "twitter\_handle"]

update\_finance\_credit\_debit(name, twitter\_handle)

update\_bank\_summary(name)

update\_banks(name)

update\_location\_data(name)

update\_summary\_location\_data(name)

update\_loan\_payment\_data(name)

}

}

update\_all\_data()

#print\_userdb()

###########################################Financial data###############################################

Twitter

library(mongolite)

library(twitteR)

library(stringr)

library(tm)

require('sentR')

library(plyr)

library(rjson)

####################################Twitter Auth#################################################

api\_key = 'RUmPyAIYfK9eMzIavtopngC21'

api\_secret = 'pHIFNMYpzqzIgxnHAXdrXAU75fCg1foTruHCV3lkM8BGwTyBBM'

access\_token = '2405586890-T11ug4N2p1qBFNBuNHLnc63zEsOw5vQUmqtcFAP'

access\_token\_secret = '59Qd4Ew50ssNeYcNfrT6HxvXUQ3ATI4A56PGLFpeMTFkz'

setup\_twitter\_oauth(api\_key,api\_secret,access\_token,access\_token\_secret)

####################################Twitter Auth#################################################

####################################Load sentiments word file#################################################

projectDir = getwd()

dataDir = file.path(projectDir, 'data')

hu.liu.pos = scan(file.path(dataDir, 'opinion-lexicon-English', 'positive-words.txt'), what='character', comment.char=';')

hu.liu.neg = scan(file.path(dataDir, 'opinion-lexicon-English', 'negative-words.txt'), what='character', comment.char=';')

pos.words = c(hu.liu.pos, 'upgrade')

neg.words = c(hu.liu.neg, 'wtf', 'wait', 'waiting', 'epicfail', 'mechanical')

####################################Load sentiments word file#################################################

currentDate<-Sys.Date()

month<-format(currentDate,"%m")

year<-format(currentDate,"%Y")

users\_summary <- mongo(collection = "user\_info", db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

users<- mongo(collection = "users", db = "r\_db", url = "mongodb://localhost", verbose = TRUE)

########################################Twitter Update##################################################

update\_tweets\_sentiment <- function() {

users\_info <- users\_summary$find()

for(twitter\_handle in users\_info[[2]]) {

print(twitter\_handle)

users$update( query = paste0('{"twitter\_handle":"', twitter\_handle,'","month" : ', as.integer(month) , ',"year" :', year,'}'), update = paste0('{"$set":{"tweets\_sentiment": ', get\_tweets\_sentiment(twitter\_handle), '}}'))

}

}

get\_tweets\_sentiment <- function(twitter\_handle) {

name\_mentioned <- searchTwitter(paste0("from:", twitter\_handle))

print(name\_mentioned)

tweets.text <- sapply(name\_mentioned, function(x) x$getText())

tweets.text <- gsub("rt", "", tweets.text)

tweets.text <- gsub("[[:punct:]]", "", tweets.text)

tweets.text <- gsub("[ |\t]{2,}", "", tweets.text)

tweets.text <- gsub("^ ", "", tweets.text)

tweets.text <- gsub(" $", "", tweets.text)

tweets.text <- tolower(tweets.text)

tweets.text.corpus <- Corpus(VectorSource(tweets.text))

tweets.text.corpus <- tm\_map(tweets.text.corpus, function(x)removeWords(x,stopwords()))

length <- length (tweets.text)

sentiment\_score <-0

for (i in 1:length){

all\_tweets\_string <- tweets.text[[i]]

word.list = str\_split(all\_tweets\_string, '\\s+')

words = unlist(word.list)

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

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

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

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

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

sentiment\_score <- sentiment\_score + score

print(score)

}

final <- sentiment\_score/length

}

update\_friends\_followers\_count <- function() {

users\_info <- users$find()

for(twitter\_handle in users\_info[[2]]) {

users$update(query = paste0('{"twitter\_handle":"', twitter\_handle,'","month" : ', as.integer(month) , ',"year" :', year,'}'), update = paste0('{"$set":{"twitter\_followers\_count": ', get\_followers\_count(twitter\_handle), '}}'))

users$update(query = paste0('{"twitter\_handle":"', twitter\_handle,'","month" : ', as.integer(month) , ',"year" :', year,'}'), update = paste0('{"$set":{"twitter\_friends\_count": ', get\_friends\_count(twitter\_handle), '}}'))

}

}

get\_followers\_count <- function(twitter\_handle) {

tweet <- getUser(twitter\_handle)

twitter\_number\_of\_followers <-tweet$getFollowersCount()

}

get\_friends\_count <- function(twitter\_handle) {

tweet <- getUser(twitter\_handle)

twitter\_number\_of\_friends<-tweet$getFriendsCount()

}

update\_tweets\_sentiment()

update\_friends\_followers\_count()

########################################Twitter Update##################################################

users\_info <- users$find()

print(users\_info)