**“SYNTAX COMPILATION”**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**PYTHON SYNTAX**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Installing Twint**

\*\*\*========================\*\*\*

***#install program***

pip install twint

\*\*\*========================\*\*\*

**Create Scraper Program**

\*\*\*========================\*\*\*

***#import program***

import twint

\*\*\*==\*\*\*

***#configuration***

config = twint.Config()

config.Search = "omnibuslaw"

config.Lang = "id"

config.Limit = 5000

config.Since = "2020–01–05"

config.Until = "2020–01–12"

config.Store\_csv = True

config.Output = "data1.csv"

\*\*\*==\*\*\*

***#running search***

twint.run.Search(config)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**R SYNTAX**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Pre-processing Data**

\*\*\*===========================\*\*\*

***#install packages***

install.packages("textclean")

install.packages("devtools")

install.packages(“githubinstall”)

install\_github("nurandi/katadasaR")

***#ketika install github package devtools harus aktif***

install.packages("tokenizers")

install.packages("stopwords")

install.packages("wordcloud")

instal.packages("tm")

\*\*\*==\*\*\*

***#loading package***

library(textclean)

library(katadasaR)

library(tokenizers)

library(wordcloud)

library(dplyr)

library(tidyverse)

library(ggplot2)

library(readr)

library(tidytext)

library(stopwords)

library(tm)

\*\*\*==\*\*\*

***#import data***

data1 <- read\_csv("omnibuslawD14.csv")

data2 <- read\_csv("omnibuslawD15.csv")

data3 <- read\_csv("omnibuslawD16.csv")

data4 <- read\_csv("omnibuslawD17.csv")

data5 <- read\_csv("omnibuslawD18.csv")

data6 <- read\_csv("omnibuslawD19.csv")

data7 <- read\_csv("omnibuslawD20.csv")

data8 <- read\_csv("omnibuslawD21.csv")

data9 <- read\_csv("omnibuslawD22.csv")

data10 <- read\_csv("omnibuslaw13.csv")

data11 <- read\_csv("omnibuslaw12.csv")

data12 <- read\_csv("omnibuslaw11.csv")

data13 <- read\_csv("omnibuslaw10.csv")

data14 <- read\_csv("omnibuslaw9.csv")

data15 <- read\_csv("omnibuslaw8.csv")

data16 <- read\_csv("omnibuslaw7.csv")

data17 <- read\_csv("omnibuslaw6.csv")

data18 <- read\_csv("omnibuslaw5.csv")

data19 <- read\_csv("omnibuslaw4.csv")

data20 <- read\_csv("omnibuslaw3.csv")

data21 <- read\_csv("omnibuslaw2.csv")

data22 <- read\_csv("omnibuslaw1.csv")

\*\*\*==\*\*\*

***#lakukan merge***

data <- bind\_rows(data1,

data2,

data3,

data4,

data5,

data6,

data7,

data8,

data9,

data10,

data11,

data12,

data13,

data14,

data15,

data16,

data17,

data18,

data19,

data20,

data21,

data22

)

\*\*\*==\*\*\*

***#hapus data set yang tidak diperlukan***

rm(data1,

data2,

data3,

data4,

data5,

data6,

data7,

data8,

data9,

data10,

data11,

data12,

data13,

data14,

data15,

data16,

data17,

data18,

data19,

data20,

data21,

data22

)

\*\*\*==\*\*\*

***#cek data***

dim(data)

\*\*\*==\*\*\*

***#hilangkan redundant data***

data\_NoRed <- data %>%

distinct(id, .keep\_all = T)

glimpse(data\_NoRed)

\*\*\*==\*\*\*

***#pilih atribut yang dianalsis***

data\_selected <- data\_NoRed %>%

select(id, username, tweet, replies\_count, retweets\_count, likes\_count, hashtags)

\*\*\*==\*\*\*

***#mengubah data dalam bentuk corpus***

data\_corpus <- VCorpus(VectorSource(data\_selected$tweet))

\*\*\*==\*\*\*

***#atur meta untuk data corpus***

meta(data\_corpus, "retweets\_count") <- data\_selected$retweets\_count

meta(data\_corpus, "likes\_count") <- data\_selected$likes\_count

\*\*\*==\*\*\*

***#import stopwords***

file <- "https://raw.githubusercontent.com/masdevid/ID-Stopwords/master/id.stopwords.02.01.2016.txt"

stop\_words <- read.delim(url(file))

names(stop\_words) <- "word"

stop\_words <- as\_tibble(stop\_words)

stop\_words <-stop\_words$word

\*\*\*==\*\*\*

***#menghilangkan kata tidak baku***

urlfile = "https://raw.githubusercontent.com/nasalsabila/kamus-alay/master/colloquial-indonesian-lexicon.csv"

Kamusnonbaku <- read\_csv(url(urlfile))

Kamusnonbaku <- Kamusnonbaku %>%

select(slang)

Kamusnonbaku <- Kamusnonbaku$slang

Kamusnonbaku <- Kamusnonbaku[sample(length(Kamusnonbaku), 700)]

kamusManual <- c("ada", "yg", "ga", "1", "gak", "co", "terkait", "utk", "bikin", "kalo", "sih", "dgn", "2","3","dlm", "mari","biar","html","4","5","dg","krn","sdh","tp","jd","jg","yuk","dll","lu","gw","msh","sbg","sby","m","kek", "aja", "kalo", "tdk","via", "tau", "nih", "dg", "krn", "isi", "null", "jgn", "klo", "ttg", "com", "tuh", "si", "no", "md", "ngga", "sah", "air", "pd", "lg", "ya", "nya", "detik")

\*\*\*==\*\*\*

***#cleaning data***

cleaning <- function(data) {

data <- tm\_map(data, content\_transformer(function(x) gsub("\\.|\\,", " ", x, perl=T)))

data <- tm\_map(data, content\_transformer(function(x) gsub("pic.twitter.com/\\S+", "", x, perl=T)))

data <- tm\_map(data, content\_transformer(function(x) gsub("(f|ht)tp(s?)://\\S+", "", x, perl=T)))

data <- tm\_map(data, removePunctuation)

data <- tm\_map(data, content\_transformer(tolower))

data <- tm\_map(data, content\_transformer(function(x) gsub(x, pattern = "@([A-Za-z0-9\_]+)",replacement="")))

data <- tm\_map(data, content\_transformer(function(x) gsub(x, pattern = "#([A-Za-z0-9\_]+)",replacement="")))

data <- tm\_map(data, removeNumbers)

data <- tm\_map(data, removeWords, stop\_words)

data <- tm\_map(data, removeWords, Kamusnonbaku)

data <- tm\_map(data, removeWords, kamusManual)

data <- tm\_map(data, stripWhitespace)

return(data)

}

clean\_corpus <- cleaning(data\_corpus)

\*\*\*==\*\*\*

***# kembalikan menjadi bentuk tibble***

tidy\_data <- tidy(clean\_corpus)

head(tidy\_data)

\*\*\*==\*\*\*

***# rapikan data***

tidy\_data <- tidy\_data %>%

select(id, text)

data\_rapih <- data\_selected

data\_rapih$tweet <- tidy\_data$text

\*\*\*==\*\*\*

***# lakukan tokenizing pada atribut tweet***

data\_rapih <- data\_rapih %>%

unnest\_tokens(word, tweet)

\*\*\*==\*\*\*

***#lakukan stemming [OPTONAL]***

***#jika Obs > 10000 lebih baik tidak perlu***

kata <- data\_rapih$word

kata\_rapih <- lapply(kata, katadasaR)

\*\*\*==\*\*\*

***#cek token kata yang dihasilkan***

count\_word<- data\_NoRed %>%

count(word, sort = T)

view(count\_word)

\*\*\*==\*\*\*

***#visualisasi***

***#note: sesuaikan freq dengan data kalian***

wordcloud(words = count\_word$word, freq = count\_word$n,

min.freq = 500,

max.words=1000,

random.order=FALSE, rot.per=0.35,

colors=brewer.pal(8, "Dark2"))

\*\*\*==\*\*\*

***#eksport data ke csv setelah data bersih***

write.csv(data\_rapih, "C:\\Users\\Aldi Rochman N\\Documents\\Tingkat IV\\Semester 7\\Data mining\\Data\_Aldi2.csv",

row.names=F)

\*\*\*===========================\*\*\*

**Sampling**

\*\*\*===========================\*\*\*

***#loading packages***

library(textclean)

library(katadasaR)

library(tokenizers)

library(wordcloud)

library(dplyr)

library(tidyverse)

library(ggplot2)

library(readr)

library(tidytext)

library(stopwords)

library(tm)

\*\*\*==\*\*\*

***#import dataset***

data1 <- read\_csv("Data\_Aldi2.csv")

data2 <- read\_csv("Data\_Amal2.csv")

data3 <- read\_csv("Data\_Amran2.csv")

data4 <- read\_csv("Data\_Febi2.csv")

data5 <- read\_csv("Data\_Najia2.csv")

data6 <- read\_csv("Data\_Nia2.csv")

\*\*\*==\*\*\*

***#penyederhana dataset : 1%***

sederhana <- function(a,b,c,d,e,f, presentase) {

a <- a[sample(nrow(a), round(nrow(a)\*presentase)),]

b <- b[sample(nrow(b), round(nrow(b)\*presentase)),]

c <- c[sample(nrow(c), round(nrow(c)\*presentase)),]

d <- d[sample(nrow(d), round(nrow(d)\*presentase)),]

e <- e[sample(nrow(e), round(nrow(e)\*presentase)),]

f <- f[sample(nrow(f), round(nrow(f)\*presentase)),]

small\_dataset <- bind\_rows(a,b,c,d,e,f)

return(small\_dataset)

}

data\_small <- sederhana(

data1,

data2,

data3,

data4,

data5,

data6,

0.1

)

\*\*\*==\*\*\*

***#cleaning***

***#hilangkan redundant data***

data\_paket\_hemat <- data\_small %>%

distinct(id, .keep\_all = T)

dim(data\_paket\_hemat)

\*\*\*==\*\*\*

***#count word small***

count\_word\_small <- data\_small %>%

select(-date) %>%

count(word, sort = T)

view(count\_word\_small)

\*\*\*==\*\*\*

***#world cloud small***

wordcloud(words = count\_word\_small$word, freq = count\_word\_small$n,

min.freq = 20,

max.words=1000,

random.order=FALSE, rot.per=0.35,

colors=brewer.pal(8, "Dark2"))

\*\*\*==\*\*\*

***#export data***

write.csv(data\_paket\_hemat,

"C:\\Users\\Aldi Rochman N\\Documents\\Tingkat IV\\Semester 7\\Data mining\\Data\_PaketHemat\_10 persen.csv",

row.names=F)

\*\*\*==\*\*\*

***#penyederhaan data 10%***

data\_high <- sederhana(

data1,

data2,

data3,

data4,

data5,

data6,

0.5

)

dim(data\_high)

\*\*\*==\*\*\*

***#count word middle***

count\_word\_middle <- data\_middle %>%

select(-date) %>%

count(word, sort = T)

view(count\_word\_middle)

\*\*\*==\*\*\*

***#world cloud small***

wordcloud(words = count\_word\_middle$word, freq = count\_word\_middle$n,

min.freq = 200,

max.words=1000,

random.order=FALSE, rot.per=0.35,

colors=brewer.pal(8, "Dark2"))

\*\*\*==\*\*\*

***#export data***

write.csv(data\_high,

"C:\\Users\\Aldi Rochman N\\Documents\\Tingkat IV\\Semester 7\\Data mining\\Data\_paketSULTAN\_50 persen.csv",

row.names=F)

\*\*\*===========================\*\*\*

**Eksplorasi Data (Visualisasi)**

\*\*\*===========================\*\*\*

***#loading packages***

library(SnowballC)

library(tm)

library(NLP)

library(SentimentAnalysis)

library(plyr)

library(ggplot2)

library(RColorBrewer)

library(wordcloud)

library(textclean)

library(katadasaR)

library(tokenizers)

library(wordcloud)

library(dplyr)

library(tidyverse)

library(readr)

library(tidytext)

library(stopwords)

library(tm)

library(syuzhet)

library(caret)

library(RTextTools)

library(stringr)

\*\*\*==\*\*\*

***#import data***

library(readxl)

data <- read\_excel("D:/niaa/Untitled spreadsheet - Data\_PaketHemat\_10 persen.xlsx")

***#data <- read\_excel("D:/niaa/Data\_paketGonjangGanjing\_30 persen.xlsx")***

class(data)

\*\*\*==\*\*\*

***#cek data***

dim(data)

head(data)

\*\*\*==\*\*\*

***#menampilkan grafik kata yang paling banyak muncul***

head(data\_rapih$word)

data\_rapih %>%

count(word, sort = TRUE) %>%

top\_n(20) %>%

mutate(word= reorder(word, n)) %>%

ggplot(aes(x=word, y=n)) +

geom\_col() +

coord\_flip() +

theme\_classic() +

labs(x="Word",

y="Count",

tittle = " Top 20 Kata Teratas ")

\*\*\*==\*\*\*

***#membuat plot sentiment Polarity***

pos = scan('D:/niaa/positive-words.txt', what='character')

neg = scan('D:/niaa/negative-words.txt', what='character')

pos.words = c(pos, "setuju", "Indonesiabutuhkerja","dukung","optimis")

neg.words = c(neg, "tolakomnibuslaw", "tolakruuciptakerja", "tolak", "cancel", "hentikan")

getSentimentScore = function(sentences, pos.words, neg.words, .progress = "none")

{

require(plyr)

require(stringr)

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

***#remove digit, punctuation, dan special/control character:***

sentence = gsub("[[:cntrl:]]", "", gsub("[[:punct:]]", "", gsub("\\d+", "", sentence)))

***#convert semua teks menjadi lowercase:***

sentence = tolower(sentence)

***#pisahkan setiap kalimat menggunakan spasi (space delimiter):***

words = unlist(str\_split(sentence, "\\s+"))

***#lakukan boolean match dari setiap kata-kata menggunakan pos &amp;amp;amp; neg opinion-lexicon:***

pos.matches = !is.na(match(words, pos.words))

neg.matches = !is.na(match(words, neg.words))

***#score sentimen = total positive sentiment - total negative:***

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

return(score)

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

***#return data frame berisi kalimat beserta sentimennya:***

return(data.frame(tweet= sentences, score = scores))

}

Result = getSentimentScore(data$tweet ,pos.words, neg.words)

head(Result)

New.Result <- Result %>%

filter(score != 0)

df <- data.frame(Sentiment=convertToDirection(New.Result$score))

df

my\_data <- cbind(New.Result$tweet, df)

count\_polarity<- my\_data %>%

count(Sentiment, sort = T)

view(count\_polarity)

ggplot( data = count\_polarity, aes(x = Sentiment, y = n )) +

geom\_bar(aes(fill= Sentiment), stat = "identity") +

theme(legend.position = "none") +

xlab("sentiment") + ylab("Score") + ggtitle("Sentiment Analysis : Polarity")

\*\*\*==\*\*\*

***#membuat plot sentiment Emotions***

***#pakai data b.ing***

***#import data***

library(readxl)

data1 <- read\_excel("D:/niaa/tweet\_english.xlsx")

head(data1)

review <- as.character(data1$tweet\_en)

library(SentimentAnalysis)

my\_sentiment <- get\_nrc\_sentiment(review)

\*\*\*==\*\*\*

***#memfilter emosi anger fear joy sadness surprise***

my\_sentiment\_filter <- my\_sentiment %>%

select(anger, fear, joy, sadness, surprise)

SentimentScore <- data.frame(colSums(my\_sentiment\_filter[,]))

names(SentimentScore) <- "Score"

SentimentScore <- cbind("sentiment" = rownames(SentimentScore), SentimentScore)

rownames(SentimentScore) <- NULL

view(SentimentScore)

\*\*\*==\*\*\*

***#membuat plot sentimen emosi***

ggplot(data = SentimentScore, aes(x = sentiment, y = Score)) +

geom\_bar(aes(fill= sentiment), stat = "identity") +

theme(legend.position = "none") +

xlab("sentiment") + ylab("Score") + ggtitle("Sentiment Analysis : Emotions")

\*\*\*===========================\*\*\***Model Naïve Bayes**

\*\*\*===========================\*\*\*

***#analisis sentiment positif dan negatif***

data <- Data\_10\_persen

View(data)

library(stringr)

library(plyr)

pos = scan('positive-words.txt', what='character')

neg = scan('negative-words.txt', what='character')

pos.words = c(pos, "setuju", "Indonesiabutuhkerja","dukung","optimis")

neg.words = c(neg, "tolakomnibuslaw", "tolakruuciptakerja", "tolak", "cancel", "hentikan")

getSentimentScore = function(sentences, pos.words, neg.words, .progress = "none")

{

require(plyr)

require(stringr)

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

***#remove digit, punctuation, dan special/control character:***

sentence = gsub("[[:cntrl:]]", "", gsub("[[:punct:]]", "", gsub("\\d+", "", sentence)))

***#convert semua teks menjadi lowercase:***

sentence = tolower(sentence)

***#pisahkan setiap kalimat menggunakan spasi (space delimiter):***

words = unlist(str\_split(sentence, "\\s+"))

***#lakukan boolean match dari setiap kata-kata menggunakan pos &amp;amp;amp; neg opinion-lexicon:***

pos.matches = !is.na(match(words, pos.words))

neg.matches = !is.na(match(words, neg.words))

***#score sentimen = total positive sentiment - total negative:***

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

return(score)

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

***#return data frame berisi kalimat beserta sentimennya:***

return(data.frame(tweet= sentences, score = scores))

}

Result = getSentimentScore(data$tweet,pos.words, neg.words)

New.Result <- Result %>%

filter(score != 0)

New.Result %>%

count(score)

library(plyr)

New.Result$score <- as.factor(New.Result$score)

New.Result$sentiment = revalue(New.Result$score,c('1'="Positif",'2'="Positif",'3'="Positif",'4'="Positif",'5'="Positif", '6'="Positif",'7'="Positif",'8'="Positif",'9'="Positif",'10'="Positif",'-1'="Negatif",'-2'="Negatif",'-3'="Negatif",'-4'="Negatif",'-5'="Negatif",'-6'="Negatif",'-7'="Negatif",'-8'="Negatif",'-9'="Negatif",'-10'="Negatif"))

New.Result %>%

count(sentiment, sort = T)

\*\*\*==\*\*\*

***#naive bayes***

***#load required libraries***

library(tm)

library(RTextTools)

library(e1071)

library(dplyr)

library(caret)

\*\*\*==\*\*\*

***#library for parallel processing***

list\_of\_values <- c("Positif", "Negatif")

New.Result <- filter(New.Result, sentiment %in% list\_of\_values)

New.Reslt %>%

count(sentiment, sort = T)

New.Result$sentiment <- as.factor(New.Result$sentiment)

data\_corpus <- VCorpus(VectorSource(New.Result$tweet))

\*\*\*==\*\*\*

***#inspect the corpus***

data\_corpus

dtm <- DocumentTermMatrix(data\_corpus)

dim(dtm)

\*\*\*==\*\*\*

***#SAMPLE***

Sample <- sample(1:5845, 1000)

df.test <- New.Result[Sample,]

df.train <- New.Result[-Sample,]

dtm.train <- dtm[-Sample,]

dtm.test <- dtm[Sample,]

corpus.train <- data\_corpus[-Sample]

corpus.test <- data\_corpus[Sample]

\*\*\*==\*\*\*

***#FEATURE SELECTION***

fivefreq <- findFreqTerms(dtm.train, 5)

length((fivefreq))

## [1] 2089

\*\*\*==\*\*\*

***#use only 5 most frequent words (fivefreq) to build the DTM***

dtm.train.nb <- DocumentTermMatrix(corpus.train, control=list(dictionary = fivefreq))

dim(dtm.train.nb)

## [1] 6000 2089

dtm.test.nb <- DocumentTermMatrix(corpus.test, control=list(dictionary = fivefreq))

dim(dtm.train.nb)

## [1] 6000 2089

\*\*\*==\*\*\*

***#function to convert the word frequencies to yes (presence) and no (absence) labels***

convert\_count <- function(x) {

y <- ifelse(x > 0, 1,0)

y <- factor(y, levels=c(0,1), labels=c("No", "Yes"))

y

}

***# Apply the convert\_count function to get final training and testing DTMs***

trainNB <- apply(dtm.train.nb, 2, convert\_count)

testNB <- apply(dtm.test.nb, 2, convert\_count)

***# Train the classifier***

classifier <- naiveBayes(trainNB,df.train$sentiment , laplace = 1)

***# Use the NB classifier we built to make predictions on the test set.***

pred <- predict(classifier, newdata= testNB)

***# Create a truth table by tabulating the predicted class labels with the actual class labels***

table("Predictions"= pred, "Actual" = df.test$sentiment )

***# Prepare the confusion matrix***

conf.mat <- confusionMatrix(pred, df.test$sentiment)

conf.mat

conf.mat$byClass

conf.mat$overall

conf.mat$overall['Accuracy']

\*\*\*===========================\*\*\*

**Asosiasi Kata (Word Embedding)**

\*\*\*===========================\*\*\*

***#install and loading packages***

install.packages("h2o")

library(h2o)

library(tibble)

library(tidytext)

library(dplyr)

library(readr)

library(data.table)

library(bit64)

library(reshape2)

library(tidyverse)

\*\*\*==\*\*\*

***#import data***

data <- read\_csv("Data\_10\_persen.csv",)

head(data, 5)

\*\*\*==\*\*\*

***#proses ke bentuk h2o***

h2o.init()

h2o\_object <- as.h2o(data)

\*\*\*==\*\*\*

***#preprocess menggunakan h2o***

tokenize <- function(sentences) {

tokenized <- h2o.tokenize(sentences, "\\\\W+")

tokenized.lower <- h2o.tolower(tokenized)

tokenized.lengths <- h2o.nchar(tokenized.lower)

tokenized.filtered <- tokenized.lower[is.na(tokenized.lengths) || tokenized.lengths >= 2,]

tokenized.words <- tokenized.filtered[h2o.grep("[0-9]", tokenized.filtered, invert = TRUE, output.logical = TRUE),]

}

words <- tokenize(h2o\_object$tweet)

\*\*\*==\*\*\*

***#pembuatan model***

word2vec\_model <- h2o.word2vec(words, min\_word\_freq = 100, epochs = 10)

\*\*\*==\*\*\*

***#asosiasi kata***

***#-omnibuslaw***

h2o.findSynonyms(word2vec\_model, "omnibuslaw")

***#-ruu***

h2o.findSynonyms(word2vec\_model, "ruu")

***#-phk***

h2o.findSynonyms(word2vec\_model, "phk")

***#-buruh***

h2o.findSynonyms(word2vec\_model, "buruh")

***#-investor***

h2o.findSynonyms(word2vec\_model, "investor")

***#-pengusaha***

h2o.findSynonyms(word2vec\_model, "pengusaha")

\*\*\*==\*\*\*

words <- h2o.tokenize(h2o\_object$tweet, "\\\\W+")

words <- h2o.tolower(words)

words <- h2o.char(words)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*