TUGAS III DATA MINING

Text Mining dan *Web Scrapping*

Diajukan Untuk Melengkapi Tugas Mata Kuliah Data Mining

Disusun Oleh :

Thalita Safa Azzahra (140610190053)

Dosen : Sinta Septi P, S.Si., M.Stat

PROGRAM STUDI S-1 STATISTIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

JATINANGOR

2021

1. Persiapkan data yang anda perlukan yaitu Data Berita Train.csv, buat plot wordcloud pada data tersebut, kemudian lakukan analisis asosiasi kata.

Berikut langkah – langkah analisis *wordcloud* dan asosiasi kata yang dilakukan dengan bantuan *software* R :

a. Install Package

```
> library(tm)
> library(NLP)
> library(tm)
> library(SnowballC)
> library(wordcloud)
> library(RColorBrewer)
> library(stringr)
```

b. Import Data ke R

```
> berita<-read.csv("D:/THALITA/Materi Kuliah/SMT 5/DATA MNING/Data Train
Berita.csv", header=TRUE, sep = ",",quote ="",
+                  stringsAsFactors=FALSE)
> attach(berita)
The following objects are masked from berita (pos = 3):

    Articles, Category, Text
> str(berita)
'data.frame':40 obs. of  3 variables:
 $ Articles: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Text    : chr  "howard  truanted to play snooker  conservative leader
michael howard has admitted he used to play truant to spe"| __truncated__
"wales silent on grand slam talk rhys williams says wales are still not
thinking of winning the grand slam despi"| __truncated__ "blair rejects
iraq advice calls tony blair has rejected calls for the publication of
advice on the legality of "| __truncated__ "ireland 21-19 argentina an
injury-time dropped goal by ronan o gara stole victory for ireland from
underneath t"| __truncated__ ...
 $ Category: chr  "politics" "sport" "politics" "sport" ...
```

Dari perintah "str(berita) didapat informasi bahwa data tersebut mempunyai tipe "data.frame" dengan tiga kolom utama yaitu "Articles", "Text", dan "Category".

c. Buat Corpus untuk dokumen tersebut

```
> beritacorpus<-VCorpus(VectorSource(berita$Text))
> beritacorpus
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 40
```

Jadi, kamus (corpus) telah terbuat dengan berisi 40 dokumen.

d. Buat Matriks Kata-Kata

```
> DTMberita<-DocumentTermMatrix(beritacorpus, control=list(tolower=TRUE,
+ removeNumbers=TRUE, stopwords=TRUE, removePunctuation=TRUE,
+ stemming=TRUE))
> DTMberita
<<DocumentTermMatrix (documents: 40, terms: 2314)>>
Non-/sparse entries: 5097/87463
Sparsity           : 94%
Maximal term length: 17
Weighting          : term frequency (tf)
```

Matrix telah berhasil kita buat dengan informasi seperti berikut, dengan diseting semua menjadi huruf kecil (*tolower*), data berupa angka dihilangkan (*removeNumber)*, kata-kata yang tidak bermakna dihilangkan (*stopwords*), menghilangkan tanda baca (*removePunctuation*), serta dipilih hanya kata dasar saja (*stemming*). Berdasarkan output tersebut, didapat informasi bahwa dari 40 dokumen terdapat 2314 kata yang berbeda.

e. Panggil kata – kata dalam dokumen

Daftar kata-kata diurutkan secara abjad atau lexical dan dianggap sebagai kolom, pada perintah inspect hanya diperlihatkan 10 kata-kata pertama saja.

```
> inspect(DTMberita)
<<DocumentTermMatrix (documents: 40, terms: 2314)>>
Non-/sparse entries: 5097/87463
Sparsity           : 94%
Maximal term length: 17
```

```
Weighting          : term frequency (tf)
Sample             :
    Terms
Docs also first game open play said time two will win
  11    0     5    0   13    5    2    0   3    4   2
  26    0     0    0    0    0   12    1   0    1   0
  3     0     1    0    0    0    9    4   1    0   0
  32    2     1    0    0    0    1    1   0   11   0
  33    1     2    0    0    0    6    1   0    4   0
  36    3     0    1    0    0    8    3   0    0   0
  37    5     5    6    0    0    0    2   2    0   5
  39    1     2    0    1    1    1    0   4    2   1
  4     0     2    1    1    2    0    1   4    0   0
  40    1     0    0    0    0   10    0   2    3   0
> str(DTMberita)
List of 6
 $ i       : int [1:5097] 1 1 1 1 1 1 1 1 1 1 …
 $ j       : int [1:5097] 20 27 36 63 93 144 174 297 299 331 …
 $ v       : num [1:5097] 2 2 1 1 2 1 2 1 1 1 …
 $ nrow    : int 40
 $ ncol    : int 2314
 $ dimnames:List of 2
  ..$ Docs : chr [1:40] "1" "2" "3" "4" …
  ..$ Terms: chr [1:2314] "â£bn" "â£m" "abi" "abil" …
 - attr(*, "class")= chr [1:2] "DocumentTermMatrix"
"simple_triplet_matrix"
 - attr(*, "weighting")= chr [1:2] "term frequency" "tf"
```

Nampak pada output "`inspect(DTMberita)`" dalam matriks tersebut hanya diperlihatkan 10 kata (kolom) pertama saja dan disusun secara abjad dari keseluruhan kata yang ada pada dokumen. Kemudian jika ingin dilihat kata apa saja yang didapatkan dapat menggunakan perintah berikut,

```
# kalo mau panggil semua kata - katanya
> DTMberita$dimnames$Terms
```

f. Membuat matriks

```
> # Membuat matrix
```

```
> dokDTM<-TermDocumentMatrix(beritacorpus) ; dokDTM
<<TermDocumentMatrix (terms: 3426, documents: 40)>>
Non-/sparse entries: 6930/130110
Sparsity           : 95%
Maximal term length: 21
Weighting          : term frequency (tf)
> ma<-as.matrix(dokDTM)
> vec<-sort(rowSums(ma),decreasing = TRUE)
```

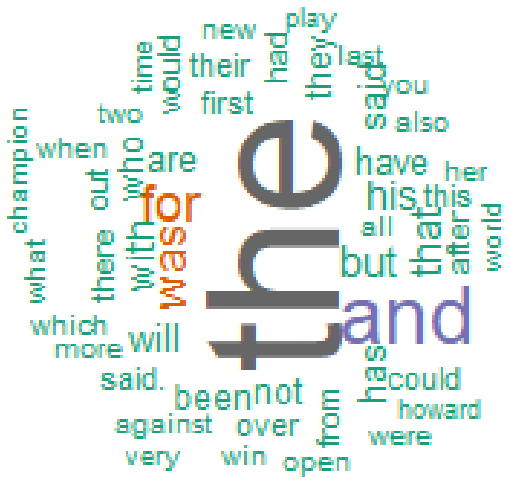
g. Melihat kata dengan frekuensi terbanyak

```
> dec<-data.frame(word=names(vec),freq=vec)
> head(dec)
     word freq
the   the  797
and   and  291
for   for  143
was   was  119
but   but   96
that that   93
> str(dec)
'data.frame':3426 obs. of  2 variables:
 $ word: chr  "the" "and" "for" "was" ...
 $ freq: num  797 291 143 119 96 93 88 81 75 72 ...
> dec$word #panggil kata
> dec$freq #lihat frekuensi dari setiap kata
```

Berdasarkan output diatas, dapat disimpulkan bahwa 3 kata yang memiliki frekuensi terbanyak adalah “the” sebanyak 797, “and” sebanyak 291, dan “for” sebanyak 143.

h. Membuat *Wordcloud*

```
> wordcloud(words = dec$word, freq = dec$freq,
+           min.freq = 1, max.words=50, random.order=FALSE,
+           rot.per=0.35, colors=brewer.pal(8, "Dark2"))
```



Berdasarkan output gambar *wordcloud* yang dihasilkan, semakin besar ukuran kata pada *wordcloud* maka semakin tinggi frekuensi kata tersebut, artinya kata tersebut sering digunakan. Kata "the", "and", dan "for" memiliki ukuran yang cukup besar artinya 3 kata tersebut menjadi kata yang sering digunakan dalam dokumen berita.

i. Analisis Asosiasi kata

```
> veec<-as.list(findAssocs(dokDTM, terms =c("the", "and", "for"),
+                          corlimit =c(0.15,0.15,0.15,0.15,0.15,0.15)))
> head(veec$the)
   year.      who becoming     year   seemed     with
    0.72     0.64     0.63     0.63     0.62     0.61
> head(veec$and)
    long becoming      who    year.    rival  rivalry
    0.68     0.66     0.61     0.61     0.59     0.59
> head(veec$for)
    less   muslim outdated   leader   suggested   although
    0.72     0.58     0.58     0.57        0.57        0.55
```

Berdasarkan output diatas, maka dapat diambil kesimpulan bahwa, misal kata "the" mempunyai hubungan asosiasi yang erat dengan kata "year" sebesar 0.72, kata "and" mempunyai hubungan asosiasi yang erat dengan kata "long" sebesar 0.68, dan kata "for" mempunyai hubungan yang erat dengan kata "less" sebesar 0.72. Serta bisa juga diinterpretasikan dengan asosiasi kata yang lainnya,

2. Gunakan data: Data Berita Train.csv sebagai data training dan Data Berita Test.csv sebagai data testing. Lakukan preprocessing pada data dengan menggunakan corpus sehingga kata terstandasisasi. Setelah itu convert dokumen dengan Document Term Matrix, setelah itu gunakan algoritma Naïve Bayes untuk mengklasifikasikan berita. Bagaimana hasil prediksinya? Berapa nilai akurasi yang dihasilkan metode Naïve Bayes berdasarkan confusion matrix nya?

Naive Bayes classifier merupakan salah satu metoda pemelajaran mesin yang memanfaatkan perhitungan probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi probabilitas pada masa depan berdasarkan pengalaman pada masa sebelumnya. Data yang digunakan untuk membuat model yaitu "Data Train Berita.csv" yang terdiri dari 40 artikel berita dan kategorinya. Data yang digunakan untuk menguji model yaitu "Data Test Berita.csv" yang terdiri dari 19 artikel berita. Berikut langkah – langkah untuk melakukan klasifikasi menggunakan pendekatan *Naïve Bayes* dengan bantuan *software R* :

1. Install Packages

```
> library(NLP)
> library(tm)
> library(e1071)
> library(naivebayes)
```

2. Import Data Train

```
> berita<-read.csv("D:/THALITA/Materi Kuliah/SMT 5/DATA MNING/Data Train
Berita.csv", header=TRUE, sep = ",",quote ="",
+                  stringsAsFactors=FALSE)
> attach(berita)
The following objects are masked from berita (pos = 4):

    Articles, Category, Text
> str(berita)
'data.frame': 40 obs. of  3 variables:
 $ Articles: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Text    : chr  "howard  truanted to play snooker  conservative leader
michael howard has admitted he used to play truant to spe"| __truncated__
"wales silent on grand slam talk rhys williams says wales are still not
thinking of winning the grand slam despi"| __truncated__ "blair rejects
iraq advice calls tony blair has rejected calls for the publication of
advice on the legality of "| __truncated__ "ireland 21-19 argentina an
```

```
injury-time dropped goal by ronan o gara stole victory for ireland from
underneath t"| __truncated__ ...
 $ Category: chr  "politics" "sport" "politics" "sport" ...
> berita$Articles
> berita$Text
> berita$Category
```

Dari perintah "str(berita) didapat informasi bahwa data tersebut mempunyai tipe "data.frame" dengan tiga kolom utama yaitu "Articles", "Text", dan "Category".

3. Kolom *Category* dijadikan *factor*

```
> berita$Category<-factor(berita$Category)
```

4. *Pre-processing* dengan corpus

```
> beritacorpus<-VCorpus(VectorSource(berita$Text))
> beritacorpus
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 40
> str(beritacorpus)
```

5. Buat matriks kata dokumen atau *Document Term Matrix*

```
> DTMberita<-DocumentTermMatrix(beritacorpus, control=list(tolower=TRUE,
+                                                          removeNumbers=TRUE,
stopwords=TRUE, removePunctuation=TRUE,
+                                                              stemming=TRUE))
> DTMberita
<<DocumentTermMatrix (documents: 40, terms: 2314)>>
Non-/sparse entries: 5097/87463
Sparsity           : 94%
Maximal term length: 17
Weighting          : term frequency (tf)
```

Matrix telah berhasil kita buat dengan informasi seperti berikut, dengan diseting semua menjadi huruf kecil (tolower), data berupa angka dihilangkan (removeNumber), kata-kata yang tidak bermakna dihilangkan (stopwords), menghilangkan tanda baca (removePunctuation), serta dipilih

hanya kata dasar saja (stemming). Berdasarkan output tersebut, didapat informasi bahwa dari 40 dokumen terdapat 2314 kata yang berbeda.

6. Panggil kata – kata dalam dokumen

   Daftar kata-kata diurutkan secara abjad atau lexical dan dianggap sebagai kolom, pada perintah inspect hanya diperlihatkan 10 kata-kata pertama saja.

```
> inspect(DTMberita)
<<DocumentTermMatrix (documents: 40, terms: 2314)>>
Non-/sparse entries: 5097/87463
Sparsity           : 94%
Maximal term length: 17
Weighting          : term frequency (tf)
Sample             :
    Terms
Docs also first game open play said time two will win
  11    0     5    0   13    5    2    0   3    4   2
  26    0     0    0    0    0   12    1   0    1   0
  3     0     1    0    0    0    9    4   1    0   0
  32    2     1    0    0    0    1    1   0   11   0
  33    1     2    0    0    0    6    1   0    4   0
  36    3     0    1    0    0    8    3   0    0   0
  37    5     5    6    0    0    0    2   2    0   5
  39    1     2    0    1    1    1    0   4    2   1
  4     0     2    1    1    2    0    1   4    0   0
  40    1     0    0    0    0   10    0   2    3   0
> str(DTMberita)
List of 6
 $ i       : int [1:5097] 1 1 1 1 1 1 1 1 1 1 ...
 $ j       : int [1:5097] 20 27 36 63 93 144 174 297 299 331 ...
 $ v       : num [1:5097] 2 2 1 1 2 1 2 1 1 1 ...
 $ nrow    : int 40
 $ ncol    : int 2314
 $ dimnames:List of 2
  ..$ Docs : chr [1:40] "1" "2" "3" "4" ...
  ..$ Terms: chr [1:2314] "â£bn" "â£m" "abi" "abil" ...
 -     attr(*,     "class")=     chr     [1:2]     "DocumentTermMatrix"
"simple_triplet_matrix"
```

```
 - attr(*, "weighting")= chr [1:2] "term frequency" "tf
```

Nampak pada output "inspect(DTMberita)" dalam matriks tersebut hanya diperlihatkan 10 kata (kolom) pertama saja dan disusun secara abjad dari keseluruhan kata yang ada pada dokumen. Kemudian jika ingin dilihat kata apa saja yang didapatkan dapat menggunakan perintah berikut,

```
> DTMberita$dimnames$Terms
> beritafreq<-findFreqTerms(DTMberita,1) ; beritafreq
> DTMberita.freq<-DTMberita[,beritafreq] ; DTMberita.freq
```

7. Penambahan fungsi (*convert_counts*)

   Kemudian dengan sedikit penambahan sebuah fungsi tersebut yang akan mengubah nilai 0 menjadi "no" dan 1 menjadi "yes" untuk mempersiapkan data di training dalam Naïve Bayes

```
> convert_counts<- function(x) {x<-ifelse(x>0, "yes","no")}
> berita.train<-apply(DTMberita.freq, MARGIN=2, convert_counts)
```

8. Melihat probabilitas untuk masing – masing kategori

```
> berita.klas <- naiveBayes(berita.train,berita.Category,laplace=1)
> berita.klas$apriori
berita.Category
politics    sport
      13       27
> politics = 13/40 ; politics
[1] 0.325
> sport = 27/40 ; sport
[1] 0.675
```

Nampak bahwa probabilitas prior untuk politik adalah 0.325 dan sport adalah 0.675. Juga pada output diatas ditunjukkan hasil untuk probabilitas bersyarat pada masing – masing kata.

9. Import Data Test

   Melakukan prediksi klasifikasi sebuah berita menggunakan Data Test.

```
>  tanya<-read.csv("D:/THALITA/Materi  Kuliah/SMT  5/DATA  MNING/Data  Test
Berita.csv", header=TRUE,quote ="",
+                stringsAsFactors=FALSE)
> tanyacorpus <- VCorpus(VectorSource(tanya$Text)) ; tanyacorpus
```

```
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 19
> tanyaDTM <-DocumentTermMatrix(tanyacorpus,control=list(tolower=TRUE,
+                                                        removeNumbers=TRUE,
stopwords=TRUE, removePunctuation=TRUE,
+                                                          stemming=TRUE))
> tanyaDTM
<<DocumentTermMatrix (documents: 19, terms: 1490)>>
Non-/sparse entries: 2252/26058
Sparsity           : 92%
Maximal term length: 15
Weighting          : term frequency (tf)
> tanyafreq<-findFreqTerms(tanyaDTM,1)
> tanyaDTMfreq <- tanyaDTM[,tanyafreq] ; tanyaDTMfreq
<<DocumentTermMatrix (documents: 19, terms: 1490)>>
Non-/sparse entries: 2252/26058
Sparsity           : 92%
Maximal term length: 15
Weighting          : term frequency (tf)
> convert_counts<-function(y){y<-ifelse(y>0,"yes","no")}
> tanyatest<-apply(tanyaDTMfreq,MARGIN=2,convert_counts)
```

10. Prediksi

```
> hasilc<-predict(berita.klas,tanyatest,type="class") ; hasilc
 [1] politics politics sport   politics sport   politics politics politics
 [9]  politics  politics  politics  politics  politics  politics  politics
politics
[17] politics politics politics
Levels: politics sport
> hasilr<-predict(berita.klas,tanyatest,type="raw") ; hasilr
        politics        sport
 [1,] 1.00000000 3.267966e-09
 [2,] 1.00000000 9.002139e-36
 [3,] 0.01706988 9.829301e-01
 [4,] 1.00000000 2.660047e-22
 [5,] 0.01521966 9.847803e-01
 [6,] 1.00000000 1.304904e-10
```

```
 [7,] 1.00000000 1.622301e-62
 [8,] 1.00000000 1.611220e-32
 [9,] 0.99999999 1.351683e-08
[10,] 1.00000000 2.655392e-24
[11,] 1.00000000 3.218536e-16
[12,] 1.00000000 3.557745e-21
[13,] 1.00000000 4.983378e-34
[14,] 1.00000000 4.139873e-14
[15,] 0.99999975 2.479594e-07
[16,] 1.00000000 6.945830e-12
[17,] 1.00000000 3.400330e-47
[18,] 1.00000000 3.343821e-25
[19,] 0.99702094 2.979060e-03
```

Jika hanya sebuah berita atau sebuah kalimat maka berita tersebut terpisah per kata. Namun jika >1 maka akan menjadi tiap kalimat. Berdasarkan output prediksi diatas, dapat disimpulkan bahwa berita pertama masuk dalam kategori "Politik" dengan tingkat probabilitas yang cukup tinggi yakni 1, berita kedua masuk dalam kategori "Politik" dengan tingkat probabilitas yang cukup tinggi yakni 1, berita ketiga masuk dalam kategori "Sport" dengan tingkat probabilitas yang cukup tinggi yakni 0.98, dan begitupun seterusnya.

11. Akurasi

Untuk melihat akurasi, kita akan menggunakan Confusion Matrix yang akan membandingkan hasil prediksi dengan model dan data aktualnya. Namun, karena Data Test Berita tidak mengandung category, maka kita akan mengkategorikan data test tersebut secara manual.

```
> hasil=confusionMatrix(table(hasilc,tanyaa$Category)) ; hasil
Confusion Matrix and Statistics


hasilc     politics sport
  politics       11     6
  sport           0     2

               Accuracy : 0.6842
                 95% CI : (0.4345, 0.8742)
```

```
    No Information Rate : 0.5789
    P-Value [Acc > NIR] : 0.24554

                  Kappa : 0.2785

 Mcnemar's Test P-Value : 0.04123

            Sensitivity : 1.0000
            Specificity : 0.2500
         Pos Pred Value : 0.6471
         Neg Pred Value : 1.0000
             Prevalence : 0.5789
         Detection Rate : 0.5789
   Detection Prevalence : 0.8947
      Balanced Accuracy : 0.6250

       'Positive' Class : politics
```

Berdasarkan output diatas, terlihat bahwa didapat nilai akurasi sebesar 0.6842 yang artinya prediksi yang dilakukan cukup akurat. Jika dilihat dari *confusion matrix* dapat disimpulkan bahwa terdapat 6 data prediksi yang tidak sesuai dengan data aktualnya.

3. Bandingkan analisis point 2 dengan algoritma Support Vector Machine. Bagaimana perbandingan akurasinya?

Support Vector Machine Algorithm merupakan sebuah algoritma klasifikasi untuk data linear dan non-linear. Untuk analisis dengan software R, kita akan menggunakan library RTextTools. Data yang digunakan sama dengan data sebelumnya.

1. Install Packages

```
> library(RTextTools)
```

2. Data dimasukkan ke dalam matrix dan membuat model

```
> # Configure the training data
> container <- create_container(DTMberita,berita$Category, trainSize=1:40,
virgin=FALSE)
> # train a SVM Model
```

```
> model <- train_model(container, "SVM", kernel="linear", cost=1)
> summary(model)

Call:
svm.default(x = container@training_matrix, y = container@training_codes,
    kernel = kernel, cost = cost, cross = cross, probability = TRUE,
    method = method)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  36

 ( 12 24 )


Number of Classes:  2

Levels:
 politics sport
```

3. Prediksi

```
> # new data ( anggap gapakai data test dulu biar tagnya udah ada )
> predictionData <- tanya$Text
> # create a prediction document term matrix
> predMatrix <- create_matrix(predictionData, originalMatrix=DTMberita)
> # create the corresponding container
> predSize = length(predictionData);
>        predictionContainer        <-        create_container(predMatrix,
labels=rep(0,predSize), testSize=1:predSize, virgin=FALSE)
> # predict
> results <- classify_model(predictionContainer, model)
> results
   SVM_LABEL  SVM_PROB
```

```
1       sport 0.8750627
2       sport 0.6158833
3       sport 0.9499904
4       sport 0.8246861
5       sport 0.9146354
6       sport 0.8173504
7    politics 0.9866297
8       sport 0.8169541
9       sport 0.7645902
10      sport 0.6667232
11      sport 0.6575999
12   politics 0.5680835
13   politics 0.6697700
14      sport 0.7873291
15      sport 0.9061611
16      sport 0.8897039
17   politics 0.9091010
18      sport 0.6570008
19      sport 0.9378362
```

4. Akurasi

```
> hasil
Confusion Matrix and Statistics


          politics sport
  politics        4     0
  sport           7     8

               Accuracy : 0.6316
                 95% CI : (0.3836, 0.8371)
    No Information Rate : 0.5789
    P-Value [Acc > NIR] : 0.41317

                  Kappa : 0.3249

 Mcnemar's Test P-Value : 0.02334
```

```
            Sensitivity : 0.3636
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.5333
             Prevalence : 0.5789
         Detection Rate : 0.2105
   Detection Prevalence : 0.2105
      Balanced Accuracy : 0.6818

       'Positive' Class : politics
```

Berdasarkan output diatas, terlihat bahwa didapat nilai akurasi sebesar 0.6316 yang artinya prediksi yang dilakukan cukup akurat.

KESIMPULAN

Stelah dilakukan prediksi data berita menggunakan dua metode yang berbeda yaitu *Naïve bayes* dan *Support Vector Machine* diperoleh nilai akurasi untuk masing – masing metode adalah 0.6842 dan 0.6316. Oleh karena itu, dapat disimpulkan bahwa metode yang paling cocok untuk melakukan prediksi dalam data ini adalah metode *Naïve Bayes* karena memiliki nilai akurasi yang lebih tinggi.