43

Text Mining: Primer, Illustration, and TXTDM Software

Appendix 43.A Loading Corpus TEXT Dataset

```
%let num_vars=10;
%let max_varlen=$25.;

data TEXT;
infile datalines dlm = ' ' missover;
input ID (c01-c&num_vars) (:&max_varlen);
datalines;
1 GenIQModel machine-learning genetic-programming
2 GenIQModel no-assumptions nonparametric black-box
3 GenIQModel no-coefficients uninterpretable
4 GenIQModel data-defining no-fitting-the-data
5 GenIQModel machine-learning alt-regression
6 GenIQModel data-mining new-variables
7 GenIQModel new-variables
8 GenIQModel variable-selection new-variables
9 GenIQModel no-data-prep
10 GenIQModel optimizes cumlift decile-table
11 GenIQModel uninterpretable no-coefficients
12 OLS-Logistic benchmarks newer-prediction
13 OLS-Logistic equations yes-coefficients
```

```
14 OLS-Logistic interpretable yes-coefficients
15 OLS-Logistic reliable accurate interpretable
16 OLS-Logistic not-black-box yes-equations
17 OLS-Logistic variable-selection
18 OLS-Logistic data-prep time-consuming
19 OLS-Logistic data-prep yes-coefficients not-black-box
20 OLS-Logistic yes-coefficients not-black-box
21 OLS-Logistic data-prep yes-equations not-black-box
;
PROC PRINT data = TEXT;
title2‘ TEXT Dataset ’;
run;
```

Appendix 43.B Intermediate Step Creating Binary Words

```
PROC TRANSPOSE data=TEXT out=TEXT_transp;
var c01-c&num_vars;
by ID;
run;

data TEXT;
set TEXT_transp;
_COL1 = COL1;
run;

PROC TRANSREG data=TEXT DESIGN;
model class (_COL1 / ZERO=‘x’);
output out = TEXT (drop = Intercept _NAME_ _TYPE_ );
id ID;
run;

PROC SQL noprint;
select trim(name)||‘=’||substr(trim(name),6)
into: varlist separated by ‘ ’
```

```
from dictionary.columns
where libname eq "WORK" and memname eq "TEXT";
quit;
%put &varlist;
```

Appendix 43.C Creating the Final Binary Words

```
%let varlist=
_COL1GenIQModel=GenIQModel _COL1OLS_Logistic=OLS_Logistic
_COL1accurate=accurate
_COL1alt_regression=alt_regression _COL1benchmarks=benchmarks
_COL1black_box=black_box
_COL1cumlift=cumlift _COL1data_defining=data_defining
_COL1data_mining=data_mining
_COL1data_prep=data_prep _COL1decile_table=decile_table _COL1equations=equations
_COL1genetic_programming=genetic_programming _COL1interpretable=interpretable
_COL1machine_learning=machine_learning _COL1new_variables=new_variables
_COL1newer_prediction=newer_prediction _COL1no_assumptions=no_assumptions
_COL1no_coefficients=no_coefficients _COL1no_data_prep=no_data_prep
_COL1no_fitting_the_data=no_fitting_the_data _COL1nonparametric=nonparametric
_COL1not_black_box=not_black_box _COL1optimizes=optimizes _COL1reliable=reliable
_COL1time_consuming=time_consuming _COL1uninterpretable=uninterpretable
_COL1variable_selection=variable_selection _COL1yes_coefficients=yes_coefficients
_COL1yes_equations=yes_equations;

PROC DATASETS library=work nolist;
modify TEXT;
rename &varlist;
quit;

PROC CONTENTS data=TEXT
out = vars (keep = name type)
noprint; run;
```

```
PROC SQL noprint;
select name into: varlist separated by ' '
from vars;
quit;
%put _global_;
```

Appendix 43.D Calculate Statistics TF, DF, NUM_DOCS, and N (=Num of Words)

```
libname tm 'c://0-tm';

PROC SORT data=TEXT; by ID;
run;


PROC SUMMARY data=TEXT;
var _numeric_;
output out=sums (drop=ID) sum=;
by ID;
run;

data tm.WORDS;
retain ID;
set sums;
ID+1;
drop _TYPE_ _FREQ_;
run;

PROC SUMMARY data=tm.WORDS;
var _numeric_;
output out=sums (drop=ID) sum=;
by ID;
run;
```

```
data tm.WORDS;
retain ID;
set sums;
ID+1;
drop _TYPE_ _FREQ_;
run;

PROC PRINT data=tm.WORDS noobs;
title2‘ WORD FREQUENCY - TF (Zero-One WORD dataset) ’;
run;

PROC SUMMARY data=tm.WORDS;
var _numeric_;
output out=tm.DF (drop= ID _TYPE_ _FREQ_) sum=;
run;

PROC PRINT data=tm.df noobs;
title2‘ DOCUMENT FREQUENCY for given Word - DF ’;
run;

PROC SUMMARY data=tm.words;
var ID;
output out=tm.NUM_DOCS (drop= _TYPE_ _FREQ_) max=NUM_DOCS;
run;

PROC PRINT data=tm.NUM_DOCS;
title2‘ NUMBER of DOCUMENTS ’;
run;

data tm.TOT_WORDS;
set tm.WORDS;
drop ID;
if _n_=1;
array nums(*) _numeric_;
TOT_WORDS=dim(nums)-1;
keep TOT_WORDS;
run;
```

```
PROC PRINT data=tm.TOT_WORDS;
title2 ' N - NUMBER of WORDS in ALL DOCUMENTS ';
run;
```

Appendix 43.E Append GenIQ_FAVORED to WORDS Dataset

```
libname tm 'c://0-tm';

data GenIQ_FAVORED;
infile datalines dlm = ' ' missover;
input ID GenIQ_FAVORED;
datalines;
1 1
2 1
3 1
4 1
5 1
6 1
7 1
8 1
9 1
10 1
11 1
12 0
13 0
14 0
15 0
16 0
17 0
18 0
19 0
20 0
21 0
```

```
;
run;

PROC SORT data=GenIQ_FAVORED; by ID;
PROC SORT data=tm.WORDS; by ID;
run;

data tm.WORD_GenIQ_FAVORED;
merge tm.WORDS GenIQ_FAVORED;
by ID;
wt=1;
run;

PROC PRINT data=tm.WORD_RESP;
run;
```

Appendix 43.F Logistic GenIQ_FAVORED Model

```
%let varlist=
   data_prep time_consuming accurate yes_coefficients not_black_box interpretable equations
   benchmarks machine_learning no_assumptions no_data_prep no_coefficients
   data_defining data_mining variable_selection new_variables alt_regression cumlift;

PROC LOGISTIC data= tm.WORD_GenIQ_FAVORED nosimple des outest=coef;
model GenIQ_FAVORED = &varlist;
run;

PROC SCORE data=tm.WORD_GenIQ_FAVORED predict type=parms
        score=coef out=score;
var &varlist;
run;
```

```
data score;
set score;
logit=GenIQ_FAVORED2;
prob_GenIQ_FAVORED=exp(logit)/(1+ exp(logit));

PROC SORT data=score; by descending prob_GenIQ_FAVORED;
run;

PROC PRINT data=score noobs;
var &varlist ID GenIQ_FAVORED prob_GenIQ_FAVORED;
format prob_GenIQ_FAVORED 5.4;
run;
```

Appendix 43.G Average Correlation among Words

```
libname tm "c://0-tm";
title " AVG_CORR of WORDS ";

%let varlist=
   data_prep time_consuming accurate yes_coefficients not_black_box interpretable equations
   benchmarks machine_learning no_assumptions no_data_prep no_coefficients
   data_defining data_mining variable_selection new_variables alt_regression cumlift;

data num_vars;
set tm.WORDS;
keep &varlist;
data num_vars;
set num_vars;
if _n_=1;
array nums(*) _numeric_;
num_vars=dim(nums);
keep num_vars;
call symputx ('num_vars',num_vars);
run;
```

```
%put &num_vars;
PROC PRINT data=num_vars;
title2 ‘ num_vars ’;
run;

PROC CORR data=tm.WORDS out=out noprint;
var &varlist;
run;

data out1;
set out;
if _type_=‘MEAN’ or _type_=‘STD’ or _type_=‘N’ then delete;
drop _type_;
array vars (&num_vars) &varlist;
array pos (&num_vars) x1 - x&num_vars;
do i= 1 to &num_vars;
pos(i)=abs(vars(i));
end;
drop &varlist i;
run;

data out2;
set out1;
array poss (&num_vars) x1- x&num_vars;
do i= 1 to &num_vars;
if poss(i) =1 then poss(i)=.;
drop i;
end;
run;

PROC MEANS data=out2 sum noprint;
output out=out3 sum=;
run;

data out4;
set out3;
sum_=sum(of x1-x&num_vars);
```

```
sum_div2= sum_/2;
bot= ((_freq_*_freq_)-_freq_)/2;
avg_corr= sum_div2/bot;
run;

data AVG_CORR;
set out4;
keep AVG_CORR;
proc print data=AVG_CORR;
run;
```

Appendix 43.H Creating TF–IDF

```
title2‘ creating tf_idf ’;
libname tm ‘c://0-tm’;
options mprint symbolgen;

data tm.TOT_WORDS;
set tm.WORDS;
drop ID;
if _n_=1;
array nums(*) _numeric_;
TOT_WORDS=dim(nums)-1;
* minus 1 because of TOT_WORDS;
keep TOT_WORDS;
call symputx (‘TOT_WORDS’,TOT_WORDS);
run;
%put &TOT_WORDS;

%let varlist=
GenIQModel OLS accurate classification computer_program cumlift
decile_table interpretable logistic machine_learning no_coefficient no_equation
prediction regression reliable specifies statistical workhorses;
```

```
PROC PRINT data=tm.NUM_DOCS;
title‘ NUM_DOCS ’;
run;

PROC SUMMARY data=tm.WORDS;
var _numeric_;
output out=tm.df (drop= ID _TYPE_ _FREQ_) sum=;
run;

data tm.df;
set tm.df;
array words(&tot_words) &varlist;
array df(&tot_words) df1-df&tot_words;
do i=1 to &tot_words;
df(i)=words(i);
drop i &varlist;
end;
run;

PROC PRINT data=tm.df noobs;
title‘ words_inrows - df‘;
run;

data tm.tf;
set tm.WORDS;
array words(&tot_words) &varlist;
array tf(&tot_words) tf1-tf&tot_words;
do i=1 to &tot_words;
tf(i)=words(i);
drop i &varlist;
end;
run;

PROC PRINT data=tm.tf;
title‘ tf ’;
run;
```

```
data tm.tf;
set tm.tf; m=1;

data tm.df;
set tm.df; m=1;

data tm.num_docs;
set tm.num_docs; m=1;
run;

data tm.tf_idf;
merge tm.tf tm.df tm.num_docs;
by m;
drop m;
run;

PROC PRINT data=tm.tf_idf;
title‘ tf_idf ’;
run;

data tm.tf_idf;
set tm.tf_idf;
array tf(&tot_words) tf1-tf&tot_words;
array df(&tot_words) df1-df&tot_words;
array idf(&tot_words) idf1 - idf&tot_words;
array tf_idf(&tot_words) tf_idf1 - tf_idf&tot_words;

do i=1 to dim(df);
if df(i)=0 then
idf(i)= log(num_docs);else
idf(i)= log(num_docs/(df(i)));
tf_idf(i)=tf(i)*idf(i);

keep ID tf_idf1 - tf_idf&tot_words;
end;
run;
```

```
PROC PRINT data=tm.tf_idf;
title‘ tf_idf ’;
run;
```

Appendix 43.I WORD_TF–IDF Weights by Concat of WORDS and TF–IDF

```
libname tm "c://0-tm";
options symbolgen mprint;

data tm.tot_words;
set tm.WORDS;
drop ID;
if _n_=1;
array nums(*) _numeric_;
TOT_WORDS=dim(nums)-1;
keep TOT_WORDS;
call symputx (‘TOT_WORDS’,TOT_WORDS);
run;

%put &TOT_WORDS;
%let varlist=
   GenIQModel OLS accurate classification computer_program cumlift
   decile_table interpretable logistic machine_learning no_coefficient no_equation prediction
   regression reliable specifies statistical workhorses;

data _null_;
set tm.tf_idf;
array word(&tot_words) &varlist;
array tf_idf(&tot_words) tf_idf1- tf_idf&tot_words;
do i=1 to &tot_words;
 call symputx(‘word’|| left(put(i,2.)),vname(word(i)));
 call symputx(‘tf_idf’ || left(put(i,2.)),vname(tf_idf(i)));
```

```
end;
run;

%macro concat_word_tf_idf;
data tm.concat_word_tf_idf;
set tm.tf_idf;
rename %do i=1 %to &tot_words;
          &&tf_idf&i=&&word&i.._&&tf_idf&i
       %end;;
run;

%mend concat_word_tf_idf;
%concat_word_tf_idf

PROC CONTENTS data=tm.concat_word_tf_idf
run;

PROC CONTENTS data=tm.concat_word_tf_idf
out = vars (keep = name type);
run;

PROC SQL noprint;
select name into: varlist separated by ' '
from vars;
quit;
%put _global_;
```

Appendix 43.J WORD_RESP WORD_TF–IDF RESP

```
libname tm 'c://0-tm';
options pageno=1;
PROC SORT data=tm.concat_word_tf_idf; by ID;
PROC SORT data=tm.word_resp; by ID;
run;
```

```
data tm.word_word_tf_idf_resp;
retain ID;
merge
tm.concat_word_tf_idf
tm.word_resp; by ID;
run;

PROC PRINT data=tm.word_word_tf_idf_resp;
run;
```

Appendix 43.K Stemming

```
data tm.word_resp;
set tm.word_resp;
interpretable=sum(of interpretable:);
uninterpretable=sum(of uninterpretable:);
machine_learning=sum(of machine_learning:);
not_black_box=sum(of not_black_box:);
new_variables=sum(of new_variables:);
no_coefficient=sum(of no_coefficient:);
yes_coefficient=sum(of yes_coefficient:);
yes_equation=sum(of yes_equation:);
wt=1;
run;

PROC CONTENTS data=tm.word_resp;
run;
```

Appendix 43.L WORD Times TF–IDF

```
%let varlist=
   GenIQModel OLS accurate classification computer_program cumlift
   decile_table interpretable logistic machine_learning no_coefficient no_equation prediction
   regression reliable specifies statistical workhorses;

PROC SORT data=tm.tf_idf; by ID;
PROC SORT data=tm.word_resp; by ID;
run;

data tm.word_tf_idf_resp;
retain ID;
merge
tm.tf_idf tm.word_resp; by ID;
run;

data tm.word_tf_idf_resp;
set tm.word_tf_idf_resp;
array words(*) &varlist;
array tf_idf(*) tf_idf1-tf_idf&tot_words;
array word_wted(*) wted1- wted&tot_words;
do i = 1 to dim(words);
word_wted(i)=words(i)*tf_idf(i);
drop i;
end;
run;

PROC PRINT;
run;
```

Appendix 43.M Dataset Weighted with Words for Profile

```
%let varlist=
   GenIQModel OLS accurate classification computer_program cumlift
   decile_table interpretable logistic machine_learning no_coefficient no_equation prediction
   regression reliable specifies statistical workhorses;

data _null_;
set tm.word_tf_idf_resp;
array word(&tot_words) &varlist;
array word_wted(*) wted1 - wted&tot_words;
do i=1 to &tot_words;
call symputx('word'|| left(put(i,2.)),vname(word(i)));
call symputx('word_wted' || left(put(i,2.)),vname(word_wted(i)));
end;
run;

%macro word_wted;
data tm.word_wted;
set tm.word_tf_idf_resp;
rename %do i=1 %to &tot_words;
          &&word_wted&i=&&word&i.._&&word_wted&i
       %end;;
drop tf_idf1-tf_idf&tot_words;
run;

%mend word_wted;
%word_wted

PROC PRINT data=tm.WORD_WTED;
run;

PROC CONTENTS data= tm.WORD_WTED
out = vars (keep = name type);
run;
```

```
PROC SQL noprint;
select name into: varlist separated by ' '
from vars;
quit;
%put _global_ ;
```

Appendix 43.N VARCLUS for Two-Class Solution

```
libname tm 'c://0-tm';
options pageno=1;
title' ';
title ' VARCLUS - 2-Cluster Solution ';

ods listing;
ods html;
%let varlist=
data_prep yes_coefficients not_black_box data_mining new_variables
machine_learning alt_regression;

data tm.WORDS_NLBS;
set tm.WORDS;
attrib _ALL_ label=' ';
run;

PROC VARCLUS data= tm.WORDS_NLBS MINC=2 MAXC=2 simple outstat=coef;
ods select Rsquare;
var &varlist;
run;
ods html close;
```

Appendix 43.O Scoring VARCLUS for Two-Cluster Solution

```
%let clusoltn=2;
title2 "The &clusoltn Cluster Solution";

%let varlist=
   data_prep yes_coefficients not_black_box data_mining new_variables
   machine_learning alt_regression;

data Coef&clusoltn;
set Coef;
if _ncl_ = . or _ncl_ = &clusoltn;
drop _ncl_;
run;

PROC SCORE data=tm.words_nlbs score=Coef&clusoltn out=scored;
var &varlist;
run;

PROC SORT data=scored; by descending GenIQModel;

PROC PRINT data=scored noobs;var &varlist clus1-clus2 ID GenIQModel;
format clus1 clus2 5.2;
run;

* Assigning the Individual to the Classified Cluster-Segment;
data scored_classified;
set scored ;
temp=max(clus1, clus2);
      if clus1 = temp then predictd = clus1;
else if clus2 = temp then predictd = clus2;
run;

data tm.scored_classified (drop=temp);
set scored_classified;
temp=max(clus1, clus2);
      if clus1 = temp then SEGMENT = 'clus1';
```

```
else if clus2 = temp then SEGMENT = ‘clus2’;
run;

PROC PRINT data=tm.scored_classified noobs;
var &varlist SEGMENT ID GenIQModel;
run;
```

Appendix 43.P Direction of Words with Its Cluster 1

```
%let Clus=Clus1;
title " r of &Clus with Correlates ";

%let varlistclus1=
data_prep yes_coefficients not_black_box
data_mining new_variables;

PROC CORR data=tm.scored_classified rank
outp=out noprint;
var &varlistclus1;
with &Clus;
run;

PROC PRINT data=out;
title‘ out ’;
run;

data out1;
set out;
if _TYPE_=‘MEAN’ then delete;
if _TYPE_=‘STD’ then delete;
drop _NAME_;
run;

PROC PRINT data=out1;
title‘ out1 ’; run;
```

```
PROC TRANSPOSE data=out1
out=out2 (rename=(_1=n _2=Corr_&Clus ) ) prefix=_;
run;

data out2;
set out2;
word=_NAME_;
run;

data out3;
set out2;

PROC SORT data=out3; by descending Corr_&clus;
run;

data words;
set out3;
Rank+1;
keep Rank word Corr_&clus;
run;

PROC PRINT data=words noobs;var Rank word Corr_&clus ;
format corr_&clus 6.4;
run;
```

Appendix 43.Q Performance of GenIQ Model versus Chance Model

```
libname tm 'c://0-tm';
options pageno=1;
title' ';

PROC FREQ data=tm.scored_classified;
table GenIQModel*SEGMENT /chisq sparse out=D;
run;
```

```
PROC TRANSPOSE data=D out=transp;
run;

data IMPROV;
retain GenIQMODEL_TCCR;
set transp;
drop _LABEL_;
if _NAME_="GenIQModel" then delete;
if _NAME_="SEGMENT" then delete;
if _NAME_="PERCENT" then CHANCE_TCCR=( ((col1+col2)**2)+((col3+col4)**2))/10000;
if _NAME_="COUNT" then GenIQMODEL_TCCR=((col1+col4)/sum(of col1-col4))/1;
GAIN_OVER_CHANCE= ((GenIQMODEL_TCCR- CHANCE_TCCR)/CHANCE_TCCR);
if GAIN_OVER_CHANCE=. then delete;
run;

PROC PRINT data=IMPROV;
var GenIQMODEL_TCCR CHANCE_TCCR GAIN_OVER_CHANCE;
format CHANCE_TCCR GenIQMODEL_TCCR GAIN_OVER_CHANCE percent8.2;
run;
```

Appendix 43.R Performance of Liberal-Cluster Model versus Chance Model

```
data Liberal_Cluster;
input GenIQModel SEGMENT Count @@;
datalines;
0 1 10 0 2 0
1 1 3 1 2 8
;
PROC FREQ data=Liberal_Cluster;
table GenIQModel*SEGMENT /chisq sparse out=D;
weight count;
run;
```

```
PROC TRANSPOSE data=D out=transp;
run;

data tccr;
retain GenIQMODEL_TCCR;
set transp;
drop _LABEL_;
if _NAME_="GenIQModel" then delete;
if _NAME_="SEGMENT" then delete;
if _NAME_="PERCENT" then CHANCE_TCCR=( ((col1+col2)**2)+((col3+col4)**2))/10000;
if _NAME_="COUNT" then GenIQMODEL_TCCR=((col1+col4)/sum(of col1-col4))/1;
GAIN_OVER_CHANCE= ((GenIQMODEL_TCCR- CHANCE_TCCR)/CHANCE_TCCR);
if GAIN_OVER_CHANCE=. then delete;
run;

PROC PRINT data=tccr;
var GenIQMODEL_TCCR CHANCE_TCCR GAIN_OVER_CHANCE;
format CHANCE_TCCR GenIQMODEL_TCCR GAIN_OVER_CHANCE percent8.2;
run;
```