**Data Minig Lab Experiment**

Name: Om Ashish Mishra

Registration Number: 16BCE0789

Slot: G2

**Apriori Algorithm**

Question:

1) Apply apriori algorithm to perform Market Basket Analysis Model for the

following :

|  |  |
| --- | --- |
| T\_ID | Itemsets |
| T1 | M,O,N,K,E,Y |
| T2 | D,O,N,K,E,Y |
| T3 | M,A,K,E |
| T4 | M,U,C,K,Y |
| T5 | C,O,O,K,E |

Find the Association Rule using minimum support of 60% and minimum

confidence of 80%.

Answer:

|  |
| --- |
| > mydata = read.csv("apriori\_1.csv")  > mydata  T\_ID Itemsets  1  2 T1 M,O,N,K,E,Y  3 T2 D,O,N,K,E,Y  4 T3 M,A,K,E  5 T4 M,U,C,K,Y  6 T5 C,O,O,K,E  > d<-data.frame(sapply(mydata,as.factor))  > rules<-apriori(d)  Apriori  Parameter specification:  confidence minval smax arem aval  0.8 0.1 1 none FALSE  originalSupport maxtime support minlen maxlen  TRUE 5 0.1 1 10  target ext  rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 0  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [12 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 2 done [0.00s].  writing ... [12 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > rules<-apriori(d,parameter=list(support=0.01,confidence=0.1))  Apriori  Parameter specification:  confidence minval smax arem aval  0.1 0.1 1 none FALSE  originalSupport maxtime support minlen maxlen  TRUE 5 0.01 1 10  target ext  rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 0  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [12 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 2 done [0.00s].  writing ... [24 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > rules<-apriori(d,parameter=list(support=0.6,confidence=0.8))  Apriori  Parameter specification:  confidence minval smax arem aval  0.8 0.1 1 none FALSE  originalSupport maxtime support minlen maxlen  TRUE 5 0.6 1 10  target ext  rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 3  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > inspect(head(sort(rules, by="lift"),3)); |
| |  | | --- | | > | |
|  |

2) Create a dataset with minimum 30 transactions with the purchase list of

combination made out of 10 Items. Apply Apriori Algorithm to generate

the association rule with

10 Items are

1 = Tea

2 = Coffee

3 = Chips

4 = Toffee

5 = Cold drinks

6 = Biscuits

7 = Ice Creams

8 = Sweets

9 = Cakes

10 = Fruits



a) Minimum confidence – 50% and Minimum support -50%

|  |
| --- |
| > d1 = read.csv(file.choose(),header = T)  > d1  TID Itemsets  1 T1 1,6,4,9  2 T2 2,6  3 T3 8,1,10,5,7  4 T4 3,5,9,2  5 T5 6,2,8,10  6 T6 4,2,8,6,10  7 T7 9,5,2,1  8 T8 2,5,7,1  9 T9 3,7,9,4  10 T10 7,4,5,2  11 T11 9,8,7,10  12 T12 3,2,6,2  13 T13 4,7,2,3  14 T14 6,4,9,2  15 T15 3,6,5,8  16 T16 5,3  17 T17 7,2,1,10  18 T18 8,3,6,1  19 T19 6,9,1  20 T20 5,8,3,2,1  21 T21 10,9,4,3  22 T22 1,7,4,8  23 T23 9,6,4,10  24 T24 3,8,5,10  25 T25 9,10,8,3  26 T26 1,4,7,3  27 T27 1,9,5,3  28 T28 3,6,7,8  29 T29 9,4,5,1  30 T30 9,4,6,2  > rules<-apriori(d1)  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.8 0.1 1 none FALSE TRUE 5 0.1 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 3  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[60 item(s), 30 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > rules<-apriori(d,parameter=list(support=0.5,confidence=0.5))  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.5 0.1 1 none FALSE TRUE 5 0.5 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 3  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > inspect(head(sort(rules,by="lift"),3)); |
| |  | | --- | | > | |
|  |

b) Minimum confidence – 20% and Minimum support -80%

|  |
| --- |
| > rules<-apriori(d1)  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.8 0.1 1 none FALSE TRUE 5 0.1 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 3  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[60 item(s), 30 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > rules<-apriori(d,parameter=list(support=0.8,confidence=0.2))  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.2 0.1 1 none FALSE TRUE 5 0.8 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 4  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > inspect(head(sort(rules,by="lift"),3)); |
| |  | | --- | | > | |
|  |

c) Minimum confidence – 80% and Minimum support -20%

|  |
| --- |
| > rules<-apriori(d1)  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.8 0.1 1 none FALSE TRUE 5 0.1 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 3  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[60 item(s), 30 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > rules<-apriori(d,parameter=list(support=0.2,confidence=0.8))  Apriori  Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.8 0.1 1 none FALSE TRUE 5 0.2 1 10 rules FALSE  Algorithmic control:  filter tree heap memopt load sort verbose  0.1 TRUE TRUE FALSE TRUE 2 TRUE  Absolute minimum support count: 1  set item appearances ...[0 item(s)] done [0.00s].  set transactions ...[12 item(s), 6 transaction(s)] done [0.00s].  sorting and recoding items ... [0 item(s)] done [0.00s].  creating transaction tree ... done [0.00s].  checking subsets of size 1 done [0.00s].  writing ... [0 rule(s)] done [0.00s].  creating S4 object ... done [0.00s].  > inspect(head(sort(rules,by="lift"),3)); |
| |  | | --- | | > | |
|  |

**Decision Tree**

> library(caret)

Loading required package: lattice

Loading required package: ggplot2

> library(rpart.plot)

Loading required package: rpart

> car\_df=read.csv("dataset.csv")

> car\_df

vhigh vhigh.1 X2 X2.1 small low unacc

1 vhigh vhigh 2 2 small med unacc

2 vhigh vhigh 2 2 small high unacc

3 vhigh vhigh 2 2 med low unacc

4 vhigh vhigh 2 2 med med unacc

5 vhigh vhigh 2 2 med high unacc

6 vhigh vhigh 2 2 big low unacc

7 vhigh vhigh 2 2 big med unacc

8 vhigh vhigh 2 2 big high unacc

9 vhigh vhigh 2 4 small low unacc

10 vhigh vhigh 2 4 small med unacc

11 vhigh vhigh 2 4 small high unacc

12 vhigh vhigh 2 4 med low unacc

13 vhigh vhigh 2 4 med med unacc

14 vhigh vhigh 2 4 med high unacc

15 vhigh vhigh 2 4 big low unacc

16 vhigh vhigh 2 4 big med unacc

17 vhigh vhigh 2 4 big high unacc

18 vhigh vhigh 2 more small low unacc

19 vhigh vhigh 2 more small med unacc

20 vhigh vhigh 2 more small high unacc

21 vhigh vhigh 2 more med low unacc

22 vhigh vhigh 2 more med med unacc

23 vhigh vhigh 2 more med high unacc

24 vhigh vhigh 2 more big low unacc

25 vhigh vhigh 2 more big med unacc

26 vhigh vhigh 2 more big high unacc

27 vhigh vhigh 3 2 small low unacc

28 vhigh vhigh 3 2 small med unacc

29 vhigh vhigh 3 2 small high unacc

30 vhigh vhigh 3 2 med low unacc

31 vhigh vhigh 3 2 med med unacc

32 vhigh vhigh 3 2 med high unacc

33 vhigh vhigh 3 2 big low unacc

34 vhigh vhigh 3 2 big med unacc

35 vhigh vhigh 3 2 big high unacc

36 vhigh vhigh 3 4 small low unacc

37 vhigh vhigh 3 4 small med unacc

38 vhigh vhigh 3 4 small high unacc

39 vhigh vhigh 3 4 med low unacc

40 vhigh vhigh 3 4 med med unacc

41 vhigh vhigh 3 4 med high unacc

42 vhigh vhigh 3 4 big low unacc

43 vhigh vhigh 3 4 big med unacc

44 vhigh vhigh 3 4 big high unacc

45 vhigh vhigh 3 more small low unacc

46 vhigh vhigh 3 more small med unacc

47 vhigh vhigh 3 more small high unacc

48 vhigh vhigh 3 more med low unacc

49 vhigh vhigh 3 more med med unacc

50 vhigh vhigh 3 more med high unacc

51 vhigh vhigh 3 more big low unacc

52 vhigh vhigh 3 more big med unacc

53 vhigh vhigh 3 more big high unacc

54 vhigh vhigh 4 2 small low unacc

55 vhigh vhigh 4 2 small med unacc

56 vhigh vhigh 4 2 small high unacc

57 vhigh vhigh 4 2 med low unacc

58 vhigh vhigh 4 2 med med unacc

59 vhigh vhigh 4 2 med high unacc

60 vhigh vhigh 4 2 big low unacc

61 vhigh vhigh 4 2 big med unacc

62 vhigh vhigh 4 2 big high unacc

63 vhigh vhigh 4 4 small low unacc

64 vhigh vhigh 4 4 small med unacc

65 vhigh vhigh 4 4 small high unacc

66 vhigh vhigh 4 4 med low unacc

67 vhigh vhigh 4 4 med med unacc

68 vhigh vhigh 4 4 med high unacc

69 vhigh vhigh 4 4 big low unacc

70 vhigh vhigh 4 4 big med unacc

71 vhigh vhigh 4 4 big high unacc

72 vhigh vhigh 4 more small low unacc

73 vhigh vhigh 4 more small med unacc

74 vhigh vhigh 4 more small high unacc

75 vhigh vhigh 4 more med low unacc

76 vhigh vhigh 4 more med med unacc

77 vhigh vhigh 4 more med high unacc

78 vhigh vhigh 4 more big low unacc

79 vhigh vhigh 4 more big med unacc

80 vhigh vhigh 4 more big high unacc

81 vhigh vhigh 5more 2 small low unacc

82 vhigh vhigh 5more 2 small med unacc

83 vhigh vhigh 5more 2 small high unacc

84 vhigh vhigh 5more 2 med low unacc

85 vhigh vhigh 5more 2 med med unacc

86 vhigh vhigh 5more 2 med high unacc

87 vhigh vhigh 5more 2 big low unacc

88 vhigh vhigh 5more 2 big med unacc

89 vhigh vhigh 5more 2 big high unacc

90 vhigh vhigh 5more 4 small low unacc

91 vhigh vhigh 5more 4 small med unacc

92 vhigh vhigh 5more 4 small high unacc

93 vhigh vhigh 5more 4 med low unacc

94 vhigh vhigh 5more 4 med med unacc

95 vhigh vhigh 5more 4 med high unacc

96 vhigh vhigh 5more 4 big low unacc

97 vhigh vhigh 5more 4 big med unacc

98 vhigh vhigh 5more 4 big high unacc

99 vhigh vhigh 5more more small low unacc

100 vhigh vhigh 5more more small med unacc

101 vhigh vhigh 5more more small high unacc

102 vhigh vhigh 5more more med low unacc

103 vhigh vhigh 5more more med med unacc

104 vhigh vhigh 5more more med high unacc

105 vhigh vhigh 5more more big low unacc

106 vhigh vhigh 5more more big med unacc

107 vhigh vhigh 5more more big high unacc

108 vhigh high 2 2 small low unacc

109 vhigh high 2 2 small med unacc

110 vhigh high 2 2 small high unacc

111 vhigh high 2 2 med low unacc

112 vhigh high 2 2 med med unacc

113 vhigh high 2 2 med high unacc

114 vhigh high 2 2 big low unacc

115 vhigh high 2 2 big med unacc

116 vhigh high 2 2 big high unacc

117 vhigh high 2 4 small low unacc

118 vhigh high 2 4 small med unacc

119 vhigh high 2 4 small high unacc

120 vhigh high 2 4 med low unacc

121 vhigh high 2 4 med med unacc

122 vhigh high 2 4 med high unacc

123 vhigh high 2 4 big low unacc

124 vhigh high 2 4 big med unacc

125 vhigh high 2 4 big high unacc

126 vhigh high 2 more small low unacc

127 vhigh high 2 more small med unacc

128 vhigh high 2 more small high unacc

129 vhigh high 2 more med low unacc

130 vhigh high 2 more med med unacc

131 vhigh high 2 more med high unacc

132 vhigh high 2 more big low unacc

133 vhigh high 2 more big med unacc

134 vhigh high 2 more big high unacc

135 vhigh high 3 2 small low unacc

136 vhigh high 3 2 small med unacc

137 vhigh high 3 2 small high unacc

138 vhigh high 3 2 med low unacc

139 vhigh high 3 2 med med unacc

140 vhigh high 3 2 med high unacc

141 vhigh high 3 2 big low unacc

142 vhigh high 3 2 big med unacc

[ reached getOption("max.print") -- omitted 1585 rows ]

> set.seed(3033)

> car\_df=read.csv("dataset.csv",sep=',',header=FALSE)

> str(car\_df)

'data.frame': 1728 obs. of 7 variables:

$ V1: Factor w/ 4 levels "high","low","med",..: 4 4 4 4 4 4 4 4 4 4 ...

$ V2: Factor w/ 4 levels "high","low","med",..: 4 4 4 4 4 4 4 4 4 4 ...

$ V3: Factor w/ 4 levels "2","3","4","5more": 1 1 1 1 1 1 1 1 1 1 ...

$ V4: Factor w/ 3 levels "2","4","more": 1 1 1 1 1 1 1 1 1 2 ...

$ V5: Factor w/ 3 levels "big","med","small": 3 3 3 2 2 2 1 1 1 3 ...

$ V6: Factor w/ 3 levels "high","low","med": 2 3 1 2 3 1 2 3 1 2 ...

$ V7: Factor w/ 4 levels "acc","good","unacc",..: 3 3 3 3 3 3 3 3 3 3 ...

> head(car\_df)

V1 V2 V3 V4 V5 V6 V7

1 vhigh vhigh 2 2 small low unacc

2 vhigh vhigh 2 2 small med unacc

3 vhigh vhigh 2 2 small high unacc

4 vhigh vhigh 2 2 med low unacc

5 vhigh vhigh 2 2 med med unacc

6 vhigh vhigh 2 2 med high unacc

> set.seed(3033)

> intrain=createDataPartition(y=car\_df$V7,p=0.7,list=FALSE)

> testing=car\_df[-itrain,]

Error in `[.data.frame`(car\_df, -itrain, ) : object 'itrain' not found

> testing=car\_df[-intrain,]

> dim(training)

[1] 1211 7

> dim(testing);

[1] 517 7

> anyNA(car\_df)

[1] FALSE

> summary(car\_df)

V1 V2 V3 V4

high :432 high :432 2 :432 2 :576

low :432 low :432 3 :432 4 :576

med :432 med :432 4 :432 more:576

vhigh:432 vhigh:432 5more:432

V5 V6 V7

big :576 high:576 acc : 384

med :576 low :576 good : 69

small:576 med :576 unacc:1210

vgood: 65

> trctrl=trainControl(method="repeatedcv",number=10,repeats=3)

> set.seed(3333)

>dtree\_fir=train(V7~.,data=training,method="rpart",parms=list(split="information"),trControl=trctrl,tuneLength=10)

Error: package e1071 is required

> install.packages("e1071")

Installing package into ‘/home/likewise-open/VITUNIVERSITY/16bce0752/R/x86\_64-pc-linux-gnu-library/3.0’

(as ‘lib’ is unspecified)

trying URL 'https://cran.rstudio.com/src/contrib/e1071\_1.6-8.tar.gz'

Content type 'application/x-gzip' length 581513 bytes (567 KB)

==================================================

downloaded 567 KB

\* installing \*source\* package ‘e1071’ ...

\*\* package ‘e1071’ successfully unpacked and MD5 sums checked

checking for C++ compiler default output file name... a.out

checking whether the C++ compiler works... yes

checking whether we are cross compiling... no

checking for suffix of executables...

checking for suffix of object files... o

checking whether we are using the GNU C++ compiler... yes

checking whether g++ accepts -g... yes

\*\* libs

gcc -std=gnu99 -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c Rsvm.c -o Rsvm.o

gcc -std=gnu99 -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c cmeans.c -o cmeans.o

gcc -std=gnu99 -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c cshell.c -o cshell.o

gcc -std=gnu99 -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c floyd.c -o floyd.o

gcc -std=gnu99 -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c init.c -o init.o

g++ -I/usr/share/R/include -DNDEBUG -fpic -g -O2 -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -D\_FORTIFY\_SOURCE=2 -g -c svm.cpp -o svm.o

svm.cpp: In function ‘svm\_model\* svm\_load\_model(const char\*)’:

svm.cpp:2788:24: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%80s",cmd);

^

svm.cpp:2792:25: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%80s",cmd);

^

svm.cpp:2817:25: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%80s",cmd);

^

svm.cpp:2841:33: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%d",&param.degree);

^

svm.cpp:2843:33: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%lf",&param.gamma);

^

svm.cpp:2845:33: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%lf",&param.coef0);

^

svm.cpp:2847:36: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%d",&model->nr\_class);

^

svm.cpp:2849:29: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%d",&model->l);

^

svm.cpp:2855:36: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%lf",&model->rho[i]);

^

svm.cpp:2862:37: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%d",&model->label[i]);

^

svm.cpp:2869:38: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%lf",&model->probA[i]);

^

svm.cpp:2876:38: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%lf",&model->probB[i]);

^

svm.cpp:2883:35: warning: ignoring return value of ‘int fscanf(FILE\*, const char\*, ...)’, declared with attribute warn\_unused\_result [-Wunused-result]

fscanf(fp,"%d",&model->nSV[i]);

^

g++ -shared -L/usr/lib/R/lib -Wl,-Bsymbolic-functions -Wl,-z,relro -o e1071.so Rsvm.o cmeans.o cshell.o floyd.o init.o svm.o -L/usr/lib/R/lib -lR

installing to /home/likewise-open/VITUNIVERSITY/16bce0752/R/x86\_64-pc-linux-gnu-library/3.0/e1071/libs

\*\* R

\*\* inst

\*\* preparing package for lazy loading

\*\* help

\*\*\* installing help indices

\*\* building package indices

\*\* installing vignettes

\*\* testing if installed package can be loaded

\* DONE (e1071)

The downloaded source packages are in

‘/tmp/RtmpmbPVGT/downloaded\_packages’

> dtree\_fir=train(V7~.,data=training,method="rpart",parms=list(split="information"),trControl=trctrl,tuneLength=10)

> dtree\_fir

CART

1211 samples

6 predictors

4 classes: 'acc', 'good', 'unacc', 'vgood'

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 3 times)

Summary of sample sizes: 1090, 1090, 1091, 1090, 1089, 1090, ...

Resampling results across tuning parameters:

cp Accuracy Kappa

0.005494505 0.8693149 0.7159620

0.006181319 0.8712320 0.7181902

0.009340659 0.8632449 0.7021444

0.010989011 0.8591103 0.6926996

0.015384615 0.8214062 0.6080717

0.018543956 0.8225150 0.6101472

0.019230769 0.8225150 0.6101472

0.027472527 0.8150194 0.5823527

0.054945055 0.7968301 0.5530351

0.071428571 0.7479121 0.3655432

Accuracy was used to select the optimal model

using the largest value.

The final value used for the model was cp

= 0.006181319.

> testing[1,]

V1 V2 V3 V4 V5 V6 V7

2 vhigh vhigh 2 2 small med unacc

> predict(dtree\_fir,newdata=testing[1,])

[1] unacc

Levels: acc good unacc vgood

> testing[1,]

V1 V2 V3 V4 V5 V6 V7

2 vhigh vhigh 2 2 small med unacc

> predict(dtree\_fir,newdata=testing[1,])

[1] unacc

Levels: acc good unacc vgood

> test\_pred=predict(dtree\_fir,newdata=testing)

> confusionMatrix(test\_pred,testing$V7)

Confusion Matrix and Statistics

Reference

Prediction acc good unacc vgood

acc 82 11 28 8

good 4 3 0 0

unacc 25 3 335 0

vgood 4 3 0 11

Overall Statistics

Accuracy : 0.8337

95% CI : (0.7987, 0.8647)

No Information Rate : 0.7021

P-Value [Acc > NIR] : 3.659e-12

Kappa : 0.6301

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: acc Class: good

Sensitivity 0.7130 0.150000

Specificity 0.8831 0.991952

Pos Pred Value 0.6357 0.428571

Neg Pred Value 0.9149 0.966667

Prevalence 0.2224 0.038685

Detection Rate 0.1586 0.005803

Detection Prevalence 0.2495 0.013540

Balanced Accuracy 0.7981 0.570976

Class: unacc Class: vgood

Sensitivity 0.9229 0.57895

Specificity 0.8182 0.98594

Pos Pred Value 0.9229 0.61111

Neg Pred Value 0.8182 0.98397

Prevalence 0.7021 0.03675

Detection Rate 0.6480 0.02128

Detection Prevalence 0.7021 0.03482

Balanced Accuracy 0.8705 0.78245

> set.seed(3333)

> dtree\_fir=train(V7~.,data=training,method="rpart",parms=list(split="gini"),trControl=trctrl,tuneLength=10)

> dtree\_fir=train(V7~.,data=training,method="rpart",parms=list(split="information"),trControl=trctrl,tuneLength=10)

> dtree\_fit\_gini=train(V7~.,data=training,method="rpart",parms=list(split="gini"),trControl=trctrl,tuneLength=10)

> dtree\_fit\_gini

CART

1211 samples

6 predictors

4 classes: 'acc', 'good', 'unacc', 'vgood'

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 3 times)

Summary of sample sizes: 1089, 1090, 1089, 1089, 1090, 1089, ...

Resampling results across tuning parameters:

cp Accuracy Kappa

0.005494505 0.8698165 0.7208267

0.006181319 0.8728333 0.7251870

0.009340659 0.8720163 0.7217385

0.010989011 0.8670710 0.7104458

0.015384615 0.8359740 0.6400357

0.018543956 0.8255028 0.6177590

0.019230769 0.8227433 0.6119036

0.027472527 0.8172220 0.5956661

0.054945055 0.8012776 0.5737683

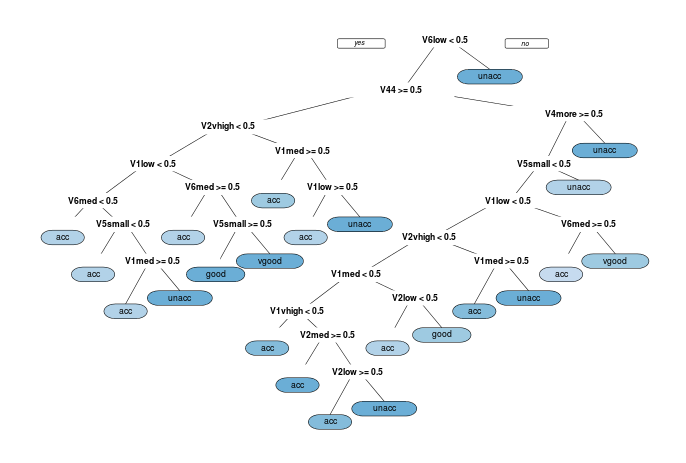
0.071428571 0.7569321 0.3918510

Accuracy was used to select the optimal model

using the largest value.

The final value used for the model was cp

= 0.006181319.



> dtree\_fit\_gini

CART

1211 samples

6 predictors

4 classes: 'acc', 'good', 'unacc', 'vgood'

No pre-processing

Resampling: Cross-Validated (10 fold, repeated 3 times)

Summary of sample sizes: 1089, 1090, 1089, 1089, 1090, 1089, ...

Resampling results across tuning parameters:

cp Accuracy Kappa

0.005494505 0.8698165 0.7208267

0.006181319 0.8728333 0.7251870

0.009340659 0.8720163 0.7217385

0.010989011 0.8670710 0.7104458

0.015384615 0.8359740 0.6400357

0.018543956 0.8255028 0.6177590

0.019230769 0.8227433 0.6119036

0.027472527 0.8172220 0.5956661

0.054945055 0.8012776 0.5737683

0.071428571 0.7569321 0.3918510

Accuracy was used to select the optimal model

using the largest value.

The final value used for the model was cp

= 0.006181319.

> prp(dtree\_fit\_gini$finalModel,box.palette="Blues",tweak=1.2)

Error in prp(dtree\_fit\_gini$finalModel, box.palette = "Blues", tweak = 1.2) :

could not find function "prp"

> library(rpart.plot)

Loading required package: rpart

> prp(dtree\_fit\_gini$finalModel,box.palette="Blues",tweak=1.2)

> test\_pred\_gini=predict(dtree\_fit\_gini,newdata=testing)

