BIG DATA ANALYTICS

A PRACTICAL REPORT

ON

BIG DATA ANALYTICS

SUBMITTED BY  
Mr. **FARHAN SAYED**

Roll No:**[IT21015]**

UNDER THE GUIDANCE OF

PROF. DIPIKA MANKAR

Submitted in fulfillment of the requirements for qualifying

MSc. IT Part I Semester - II Examination 2021-2022

University of Mumbai

Department of Information Technology

R.D. & S.H National College of Arts, Commerce & W.A. Science College Bandra (West), Mumbai – 400 050



R. D. & S.H National & W. A. Science College

**Bandra (W), Mumbai – 400050*.***

**Department of Information Technology**

**M.Sc. (IT)**

**Certificate**

*This is to certify that* ***Big Data Analytics*** *performed at* ***R.D & S.H National & W.A. Science College*** *by Mr.****Farhan Sayed*** *holding Seat No.* ***IT21015*** *studying Master of Science in Information Technology Semester – II has been satisfactorily completed as prescribed by the University of Mumbai, during the year 2021 – 2022.*

|  |  |  |
| --- | --- | --- |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Lecturer In charge** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **External Examiner** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Head of Department** |

**College Stamp**

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No** | **Date** | **Practical** | **Page No.** | **Sign** |
| 1 |  | K means clustering. |  |  |
| 2 |  | Apriori algorithm |  |  |
| 3 |  | Linear regression |  |  |
|  |  | a) Simple Linear regression |  |  |
|  |  | b) Logistic regression |  |  |
| 4 |  | a) Decision Tree |  |  |
|  |  | b) Naïve Bayes Classification |  |  |
| 5 |  | Text Analysis |  |  |

**Practical No: 1 – K means clustering**

**Aim: Read a datafile grades\_km\_input.csv and apply k-means clustering.**

**Source Code:**

# install required packages

install.packages("plyr")

install.packages("ggplot2")

install.packages("cluster")

install.packages("lattice")

install.packages("grid")

install.packages("gridExtra")

# Load the package

library(plyr)

library(ggplot2)

library(cluster)

library(lattice)

library(grid)

library(gridExtra)

# A data frame is a two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column.

grade\_input=as.data.frame(read.csv("C:\\Users\\zende\\Downloads\\grades\_km\_input.csv"))

kmdata\_orig=as.matrix(grade\_input[, c ("Student","English","Math","Science")]) kmdata=kmdata\_orig[,2:4] kmdata[1:10,]

# the k-means algorithm is used to identify clusters for k = 1, 2, .. . , 15. For each value of k, the WSS is calculated.

wss=numeric(15)

# the option n start=25 specifies that the k-means algorithm will be repeated 25 times, each starting with k random initial centroids

for(k in 1:15)wss[k]=sum(kmeans(kmdata,centers=k,nstart=25)$withinss)

plot(1:15,wss,type="b",xlab="Number of Clusters",ylab="Within sum of square")

#As can be seen, the WSS is greatly reduced when k increases from one to two. Another substantial reduction in WSS occurs at k = 3. However, the improvement in WSS is fairly linear fork > 3.

km = kmeans(kmdata,3,nstart=25) km

c( wss[3] , sum(km$withinss))

df=as.data.frame(kmdata\_orig[,2:4])

df$cluster=factor(km$cluster)

centers=as.data.frame(km$centers)

g1=ggplot(data=df, aes(x=English, y=Math, color=cluster )) + geom\_point() + theme(legend.position="right") + geom\_point(data=centers,aes(x=English,y=Math, color=as.factor(c(1,2,3))),size=10, alpha=.3, show.legend =FALSE)

g2=ggplot(data=df, aes(x=English, y=Science, color=cluster )) + geom\_point () +geom\_point(data=centers,aes(x=English,y=Science, color=as.factor(c(1,2,3))),size=10, alpha=.3, show.legend=FALSE)

g3 = ggplot(data=df, aes(x=Math, y=Science, color=cluster )) + geom\_point () + geom\_point(data=centers,aes(x=Math,y=Science, color=as.factor(c(1,2,3))),size=10, alpha=.3, show.legend=FALSE) tmp=ggplot\_gtable(ggplot\_build(g1))

grid.arrange(arrangeGrob(g1 + theme(legend.position="none"),g2 + theme(legend.position="none"),g3 + theme(legend.position="none"),top ="High School Student Cluster Analysis" ,ncol=1))

**OUTPUT:**

**install.packages("plyr")**

**Installing package into ‘C:/Users/rohit/AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**--- Please select a CRAN mirror for use in this session ---**

**trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/plyr\_1.8.7.zip'**

**Content type 'application/zip' length 1153549 bytes (1.1 MB)**

**downloaded 1.1 MB**

**package ‘plyr’ successfully unpacked and MD5 sums checked**

**The downloaded binary packages are in**

**C:\Users\ rohit \AppData\Local\Temp\RtmpENVl66\downloaded\_packages**

**> install.packages("ggplot2")**

**Installing package into ‘C:/Users/ rohit /AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/ggplot2\_3.3.6.zip'**

**Content type 'application/zip' length 4116541 bytes (3.9 MB)**

**downloaded 3.9 MB**

**package ‘ggplot2’ successfully unpacked and MD5 sums checked**

**The downloaded binary packages are in**

**C:\Users\ rohit \AppData\Local\Temp\RtmpENVl66\downloaded\_packages**

**> install.packages("cluster")**

**Installing package into ‘C:/Users/ rohit /AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/cluster\_2.1.3.zip'**

**Content type 'application/zip' length 578016 bytes (564 KB)**

**downloaded 564 KB**

**package ‘cluster’ successfully unpacked and MD5 sums checked**

**The downloaded binary packages are in**

**C:\Users\ rohit \AppData\Local\Temp\RtmpENVl66\downloaded\_packages**

**> install.packages("lattice")**

**Installing package into ‘C:/Users/ rohit /AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/lattice\_0.20-45.zip'**

**Content type 'application/zip' length 1171580 bytes (1.1 MB)**

**downloaded 1.1 MB**

**package ‘lattice’ successfully unpacked and MD5 sums checked**

**The downloaded binary packages are in**

**C:\Users\ rohit \AppData\Local\Temp\RtmpENVl66\downloaded\_packages**

**> install.packages("grid")**

**Installing package into ‘C:/Users/ rohit /AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**Warning message:**

**package ‘grid’ is a base package, and should not be updated**

**> install.packages("gridExtra")**

**Installing package into ‘C:/Users/ rohit /AppData/Local/R/win-library/4.2’**

**(as ‘lib’ is unspecified)**

**trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/gridExtra\_2.3.zip'**

**Content type 'application/zip' length 1109222 bytes (1.1 MB)**

**downloaded 1.1 MB**

**package ‘gridExtra’ successfully unpacked and MD5 sums checked**

**The downloaded binary packages are in**

**C:\Users\ rohit \AppData\Local\Temp\RtmpENVl66\downloaded\_packages**

**>**

**> library(plyr)**

**> library(ggplot2)**

**> library(cluster)**

**> library(lattice)**

**> library(grid)**

**> library(gridExtra)**

**>**

**> grade\_input=as.data.frame(read.csv("C:\\Users\\ rohit \\Downloads\\grades\_km\_input.csv"))**

**> kmdata\_orig=as.matrix(grade\_input[, c ("Student","English","Math","Science")])**

**> kmdata=kmdata\_orig[,2:4]**

**> kmdata[1:10,]**

**English Math Science**

**[1,] 99 96 97**

**[2,] 99 96 97**

**[3,] 98 97 97**

**[4,] 95 100 95**

**[5,] 95 96 96**

**[6,] 96 97 96**

**[7,] 100 96 97**

**[8,] 95 98 98**

**[9,] 98 96 96**

**[10,] 99 99 95**

**>**

**> wss=numeric(15)**

**>**

**for(k in 1:15)wss[k]=sum(kmeans(kmdata,centers=k,nstart=25)$withinss)**

**> plot(1:15,wss,type="b",xlab="Number of Clusters",ylab="Within sum of square")**

**>**

**>**

**>**

**> km = kmeans(kmdata,3,nstart=25)**

**> km**

**K-means clustering with 3 clusters of sizes 218, 244, 158**

**Cluster means:**

**English Math Science**

**1 73.22018 64.62844 65.84862**

**2 85.84426 79.68033 81.50820**

**3 97.21519 93.37342 94.86076**

**Clustering vector:**

**[1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3**

**[37] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3**

**[73] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3**

**[109] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3**

**[145] 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 3 3 2**

**[181] 2 3 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2**

**[217] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2**

**[253] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2**

**[289] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2**

**[325] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2**

**[361] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 1 2 1 2 2 2 1 1 1 1 2 2**

**[397] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

**[433] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1**

**[469] 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

**[505] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

**[541] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1**

**[577] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 2 2 2 2 3 3 2 2**

**[613] 2 1 1 2 1 2 2 2**

**Within cluster sum of squares by cluster:**

**[1] 34806.339 22984.131 6692.589**

**(between\_SS / total\_SS = 76.5 %)**

**Available components:**

**[1] "cluster" "centers" "totss" "withinss"**

**[5] "tot.withinss" "betweenss" "size" "iter"**

**[9] "ifault"**

**> c( wss[3] , sum(km$withinss))**

**[1] 64483.06 64483.06**

**> df=as.data.frame(kmdata\_orig[,2:4])**

**> df$cluster=factor(km$cluster)**

**> centers=as.data.frame(km$centers)**

**>**

**>**

**> g1=ggplot(data=df, aes(x=English, y=Math, color=cluster )) +**

**+ geom\_point() + theme(legend.position="right") +**

**+ geom\_point(data=centers,aes(x=English,y=Math, color=as.factor(c(1,2,3))),size=10, alpha=.3,**

**+ show.legend =FALSE)**

**>**

**> g2=ggplot(data=df, aes(x=English, y=Science, color=cluster )) +**

**+ geom\_point () +**

**+ geom\_point(data=centers,aes(x=English,y=Science, color=as.factor(c(1,2,3))),size=10, alpha=.3,**

**+ show.legend=FALSE)**

**>**

**>**

**> g3 = ggplot(data=df, aes(x=Math, y=Science, color=cluster )) +**

**+ geom\_point () +**

**+ geom\_point(data=centers,aes(x=Math,y=Science, color=as.factor(c(1,2,3))),size=10, alpha=.3,**

**+ show.legend=FALSE)**

**>**

**>**

**> tmp=ggplot\_gtable(ggplot\_build(g1))**

**>**

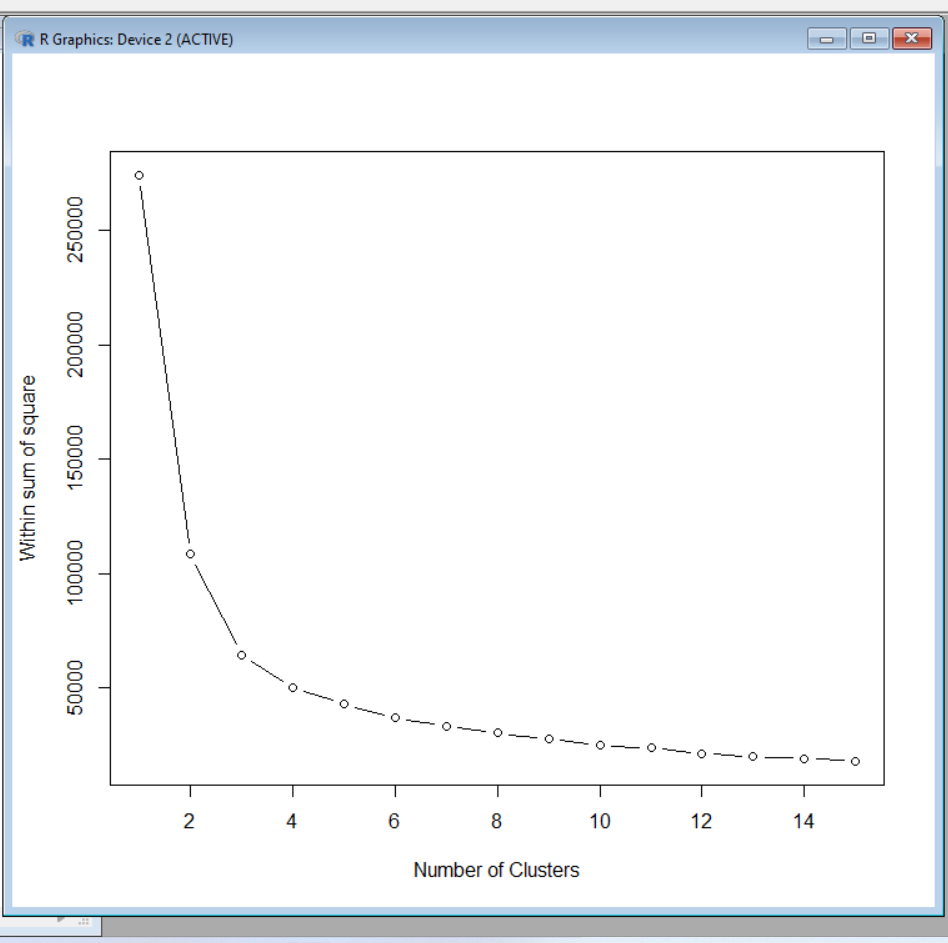
**>**

**> grid.arrange(arrangeGrob(g1 + theme(legend.position="none"),g2 +**

**+ theme(legend.position="none"),g3 + theme(legend.position="none"),**

**+ top ="High School Student Cluster Analysis" ,ncol=1))**

**>**





**Practical no 2: Apriori algorithm**

**Aim: Perform Apriori algorithm using Groceries dataset from the R arules package.**

**Source Code:**

install.packages("arules")

install.packages("arulesViz")

install.packages("RColorBrewer")

# Loading Libraries

library(arules)

library(arulesViz)

library(RColorBrewer)

# import dataset

data(Groceries)

Groceries

summary(Groceries)

class(Groceries)

# using apriori() function

rules = apriori(Groceries, parameter = list(supp = 0.02, conf = 0.2))

summary (rules)

# using inspect() function

inspect(rules[1:10])

# using itemFrequencyPlot() function

arules::itemFrequencyPlot(Groceries, topN = 20, col = brewer.pal(8, 'Pastel2'), main = 'Relative Item Frequency Plot', type = "relative", ylab = "Item Frequency (Relative)")

itemsets = apriori(Groceries, parameter = list(minlen=2, maxlen=2,support=0.02, target="frequent itemsets"))

summary(itemsets)

# using inspect() function

inspect(itemsets[1:10])

itemsets\_3 = apriori(Groceries, parameter = list(minlen=3, maxlen=3,support=0.02, target="frequent itemsets"))

summary(itemsets\_3)

inspect(itemsets\_3)

**OUTPUT:**

install.packages("RColorBrewer")

Installing package into ‘C:/Users/rohit/AppData/Local/R/win-library/4.2’

(as ‘lib’ is unspecified)

trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/RColorBrewer\_1.1-3.zip'

Content type 'application/zip' length 55837 bytes (54 KB)

downloaded 54 KB

package ‘RColorBrewer’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\ rohit \AppData\Local\Temp\RtmpCeWOaU\downloaded\_packages

>

> library(arules)

Loading required package: Matrix

Attaching package: ‘arules’

The following objects are masked from ‘package:base’:

abbreviate, write

> library(arulesViz)

> library(RColorBrewer)

>

>

> data(Groceries)

> Groceries

transactions in sparse format with

9835 transactions (rows) and

169 items (columns)

>

>

> summary(Groceries)

transactions as itemMatrix in sparse format with

9835 rows (elements/itemsets/transactions) and

169 columns (items) and a density of 0.02609146

most frequent items:

whole milk other vegetables rolls/buns soda

2513 1903 1809 1715

yogurt (Other)

1372 34055

element (itemset/transaction) length distribution:

sizes

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

2159 1643 1299 1005 855 645 545 438 350 246 182 117 78 77 55

16 17 18 19 20 21 22 23 24 26 27 28 29 32

46 29 14 14 9 11 4 6 1 1 1 1 3 1

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.000 2.000 3.000 4.409 6.000 32.000

includes extended item information - examples:

labels level2 level1

1 frankfurter sausage meat and sausage

2 sausage sausage meat and sausage

3 liver loaf sausage meat and sausage

> class(Groceries)

[1] "transactions"

attr(,"package")

[1] "arules"

>

>

> rules = apriori(Groceries,parameter=list(supp = 0.02, conf= 0.2))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen

0.2 0.1 1 none FALSE TRUE 5 0.02 1

maxlen target ext

10 rules TRUE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 196

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].

sorting and recoding items ... [59 item(s)] done [0.00s].

creating transaction tree ... done [0.01s].

checking subsets of size 1 2 3 done [0.00s].

writing ... [73 rule(s)] done [0.00s].

creating S4 object ... done [0.00s].

> summary (rules)

set of 73 rules

rule length distribution (lhs + rhs):sizes

1 2 3

1 66 6

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.000 2.000 2.000 2.068 2.000 3.000

summary of quality measures:

support confidence coverage lift

Min. :0.02003 Min. :0.2006 Min. :0.04342 Min. :0.8991

1st Qu.:0.02257 1st Qu.:0.2369 1st Qu.:0.07168 1st Qu.:1.3112

Median :0.02664 Median :0.3079 Median :0.09395 Median :1.5570

Mean :0.03424 Mean :0.3187 Mean :0.11739 Mean :1.6061

3rd Qu.:0.03589 3rd Qu.:0.3868 3rd Qu.:0.11052 3rd Qu.:1.8502

Max. :0.25552 Max. :0.5129 Max. :1.00000 Max. :2.8421

count

Min. : 197.0

1st Qu.: 222.0

Median : 262.0

Mean : 336.8

3rd Qu.: 353.0

Max. :2513.0

mining info:

data ntransactions support confidence

Groceries 9835 0.02 0.2

call

apriori(data = Groceries, parameter = list(supp = 0.02, conf = 0.2))

>

> inspect(rules[1:10])

lhs rhs support confidence

[1] {} => {whole milk} 0.25551601 0.2555160

[2] {frozen vegetables} => {whole milk} 0.02043721 0.4249471

[3] {beef} => {whole milk} 0.02125064 0.4050388

[4] {curd} => {whole milk} 0.02613116 0.4904580

[5] {pork} => {other vegetables} 0.02165735 0.3756614

[6] {pork} => {whole milk} 0.02216573 0.3844797

[7] {frankfurter} => {whole milk} 0.02053889 0.3482759

[8] {bottled beer} => {whole milk} 0.02043721 0.2537879

[9] {brown bread} => {whole milk} 0.02521607 0.3887147

[10] {margarine} => {whole milk} 0.02419929 0.4131944

coverage lift count

[1] 1.00000000 1.0000000 2513

[2] 0.04809354 1.6630940 201

[3] 0.05246568 1.5851795 209

[4] 0.05327911 1.9194805 257

[5] 0.05765125 1.9414764 213

[6] 0.05765125 1.5047187 218

[7] 0.05897306 1.3630295 202

[8] 0.08052872 0.9932367 201

[9] 0.06487036 1.5212930 248

[10] 0.05856634 1.6170980 238

>

>

> arules::itemFrequencyPlot(Groceries,topN = 20,

+ col = brewer.pal(8,'Pastel2'),

+ main = 'Relative Item Frequency Plot',

+ type = "relative",

+ ylab = "Item Frequency (Relative)")

>

> itemsets= apriori(Groceries, parameter = list(minlen=2,maxlen=2, support=0.02,

+ target="frequent itemsets"))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen

NA 0.1 1 none FALSE TRUE 5 0.02 2

maxlen target ext

2 frequent itemsets TRUE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 196

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].

sorting and recoding items ... [59 item(s)] done [0.00s].

creating transaction tree ... done [0.01s].

checking subsets of size 1 2 done [0.00s].

sorting transactions ... done [0.00s].

writing ... [61 set(s)] done [0.00s].

creating S4 object ... done [0.00s].

Warning message:

In apriori(Groceries, parameter = list(minlen = 2, maxlen = 2, support = 0.02, :

Mining stopped (maxlen reached). Only patterns up to a length of 2 returned!

> summary(itemsets)

set of 61 itemsets

most frequent items:

whole milk other vegetables yogurt rolls/buns

25 17 9 9

soda (Other)

9 53

element (itemset/transaction) length distribution:sizes

2

61

Min. 1st Qu. Median Mean 3rd Qu. Max.

2 2 2 2 2 2

summary of quality measures:

support count

Min. :0.02003 Min. :197.0

1st Qu.:0.02227 1st Qu.:219.0

Median :0.02613 Median :257.0

Mean :0.02951 Mean :290.3

3rd Qu.:0.03223 3rd Qu.:317.0

Max. :0.07483 Max. :736.0

includes transaction ID lists: FALSE

mining info:

data ntransactions support confidence

Groceries 9835 0.02 1

call

apriori(data = Groceries, parameter = list(minlen = 2, maxlen = 2, support = 0.02, target = "frequent itemsets"))

>

> inspect(itemsets[1:10])

items support count

[1] {whole milk, frozen vegetables} 0.02043721 201

[2] {beef, whole milk} 0.02125064 209

[3] {whole milk, curd} 0.02613116 257

[4] {pork, other vegetables} 0.02165735 213

[5] {pork, whole milk} 0.02216573 218

[6] {frankfurter, whole milk} 0.02053889 202

[7] {whole milk, bottled beer} 0.02043721 201

[8] {whole milk, brown bread} 0.02521607 248

[9] {whole milk, margarine} 0.02419929 238

[10] {other vegetables, butter} 0.02003050 197

>

> itemsets\_3= apriori(Groceries, parameter = list(minlen=2,maxlen=2, support=0.02,

+ target="frequent itemsets"))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen

NA 0.1 1 none FALSE TRUE 5 0.02 2

maxlen target ext

2 frequent itemsets TRUE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 196

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].

sorting and recoding items ... [59 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 done [0.00s].

sorting transactions ... done [0.00s].

writing ... [61 set(s)] done [0.00s].

creating S4 object ... done [0.00s].

Warning message:

In apriori(Groceries, parameter = list(minlen = 2, maxlen = 2, support = 0.02, :

Mining stopped (maxlen reached). Only patterns up to a length of 2 returned!

> summary(itemsets\_3)

set of 61 itemsets

most frequent items:

whole milk other vegetables yogurt rolls/buns

25 17 9 9

soda (Other)

9 53

element (itemset/transaction) length distribution:sizes

2

61

Min. 1st Qu. Median Mean 3rd Qu. Max.

2 2 2 2 2 2

summary of quality measures:

support count

Min. :0.02003 Min. :197.0

1st Qu.:0.02227 1st Qu.:219.0

Median :0.02613 Median :257.0

Mean :0.02951 Mean :290.3

3rd Qu.:0.03223 3rd Qu.:317.0

Max. :0.07483 Max. :736.0

includes transaction ID lists: FALSE

mining info:

data ntransactions support confidence

Groceries 9835 0.02 1

call

apriori(data = Groceries, parameter = list(minlen = 2, maxlen = 2, support = 0.02, target = "frequent itemsets"))

>

> inspect(itemsets\_3)

items support count

[1] {whole milk, frozen vegetables} 0.02043721 201

[2] {beef, whole milk} 0.02125064 209

[3] {whole milk, curd} 0.02613116 257

[4] {pork, other vegetables} 0.02165735 213

[5] {pork, whole milk} 0.02216573 218

[6] {frankfurter, whole milk} 0.02053889 202

[7] {whole milk, bottled beer} 0.02043721 201

[8] {whole milk, brown bread} 0.02521607 248

[9] {whole milk, margarine} 0.02419929 238

[10] {other vegetables, butter} 0.02003050 197

[11] {whole milk, butter} 0.02755465 271

[12] {whole milk, newspapers} 0.02735130 269

[13] {other vegetables, domestic eggs} 0.02226741 219

[14] {whole milk, domestic eggs} 0.02999492 295

[15] {other vegetables, fruit/vegetable juice} 0.02104728 207

[16] {whole milk, fruit/vegetable juice} 0.02663955 262

[17] {yogurt, whipped/sour cream} 0.02074225 204

[18] {other vegetables, whipped/sour cream} 0.02887646 284

[19] {whole milk, whipped/sour cream} 0.03223183 317

[20] {tropical fruit, pip fruit} 0.02043721 201

[21] {pip fruit, other vegetables} 0.02613116 257

[22] {pip fruit, whole milk} 0.03009659 296

[23] {pastry, soda} 0.02104728 207

[24] {rolls/buns, pastry} 0.02094560 206

[25] {other vegetables, pastry} 0.02257245 222

[26] {whole milk, pastry} 0.03324860 327

[27] {citrus fruit, yogurt} 0.02165735 213

[28] {citrus fruit, other vegetables} 0.02887646 284

[29] {citrus fruit, whole milk} 0.03050330 300

[30] {soda, shopping bags} 0.02460600 242

[31] {other vegetables, shopping bags} 0.02318251 228

[32] {whole milk, shopping bags} 0.02450432 241

[33] {sausage, soda} 0.02430097 239

[34] {sausage, rolls/buns} 0.03060498 301

[35] {sausage, other vegetables} 0.02694459 265

[36] {sausage, whole milk} 0.02989324 294

[37] {bottled water, soda} 0.02897814 285

[38] {yogurt, bottled water} 0.02297916 226

[39] {rolls/buns, bottled water} 0.02419929 238

[40] {other vegetables, bottled water} 0.02480935 244

[41] {whole milk, bottled water} 0.03436706 338

[42] {tropical fruit, root vegetables} 0.02104728 207

[43] {tropical fruit, soda} 0.02084392 205

[44] {tropical fruit, yogurt} 0.02928317 288

[45] {tropical fruit, rolls/buns} 0.02460600 242

[46] {tropical fruit, other vegetables} 0.03589222 353

[47] {tropical fruit, whole milk} 0.04229792 416

[48] {root vegetables, yogurt} 0.02582613 254

[49] {root vegetables, rolls/buns} 0.02430097 239

[50] {root vegetables, other vegetables} 0.04738180 466

[51] {root vegetables, whole milk} 0.04890696 481

[52] {yogurt, soda} 0.02735130 269

[53] {rolls/buns, soda} 0.03833249 377

[54] {other vegetables, soda} 0.03274021 322

[55] {whole milk, soda} 0.04006101 394

[56] {yogurt, rolls/buns} 0.03436706 338

[57] {other vegetables, yogurt} 0.04341637 427

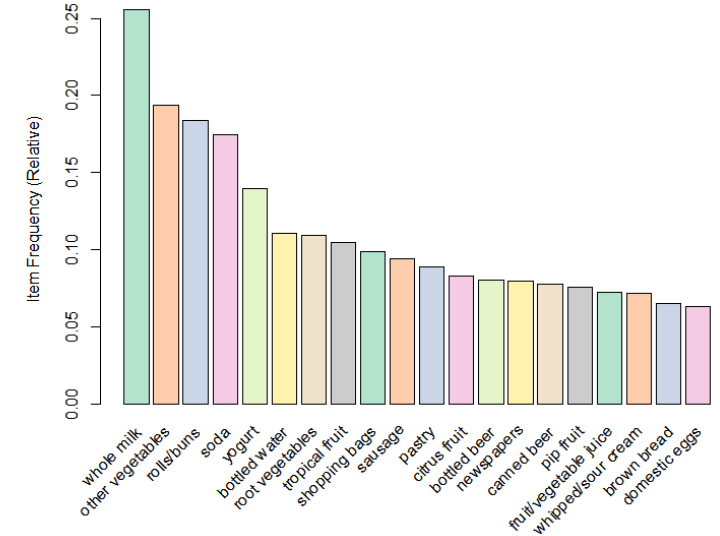
[58] {whole milk, yogurt} 0.05602440 551

[59] {other vegetables, rolls/buns} 0.04260295 419

[60] {whole milk, rolls/buns} 0.05663447 557

[61] {other vegetables, whole milk} 0.07483477 736

>



**Practical no: 3 Linear regressions**

**a) Simple Linear regression**

**Aim: Create your own data for years of experience and salary in lakhs and apply linear regression model to predict the salary.**

**Source Code:**

years\_of\_exp = c(7,5,1,3)

salary\_in\_lakhs = c(21,13,6,8)

#employee.data = data.frame(satisfaction\_score, years\_of\_exp, salary\_in\_lakhs)

employee.data = data.frame(years\_of\_exp, salary\_in\_lakhs)

employee.data

# Estimation of the salary of an employee, based on his year of experience and satisfaction score in his company.

model <- lm(salary\_in\_lakhs ~ years\_of\_exp, data = employee.data)

summary(model)

# The formula of Regression becomes

# Y = 2 + 2.5\*year\_of\_Exp

# Visualization of Regression

plot(salary\_in\_lakhs ~ years\_of\_exp, data = employee.data) abline(model)

output:

years\_of\_exp salary\_in\_lakhs

1 7 21

2 5 13

3 1 6

4 3 8

Residuals:

1. 2 3 4
   1. -1.5 1.5 -1.5

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2.0000 2.1737 0.92 0.4547

years\_of\_exp 2.5000 0.4743 5.27 0.0342 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.121 on 2 degrees of freedom

Multiple R-squared: 0.9328, Adjusted R-squared: 0.8993

F-statistic: 27.78 on 1 and 2 DF, p-value: 0.03417

**OUTPUT:**

years\_of\_exp= c(7,5,1,3)

> salary\_in\_lakhs= c(21,13,6,8)

> employee.data = data.frame(years\_of\_exp,salary\_in\_lakhs)

> employee.data

years\_of\_exp salary\_in\_lakhs

1 7 21

2 5 13

3 1 6

4 3 8

> model <- lm(salary\_in\_lakhs~ years\_of\_exp,data = employee.data)

> summary(model)

Call:

lm(formula = salary\_in\_lakhs ~ years\_of\_exp, data = employee.data)

Residuals:

1 2 3 4

1.5 -1.5 1.5 -1.5

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2.0000 2.1737 0.92 0.4547

years\_of\_exp 2.5000 0.4743 5.27 0.0342 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.121 on 2 degrees of freedom

Multiple R-squared: 0.9328, Adjusted R-squared: 0.8993

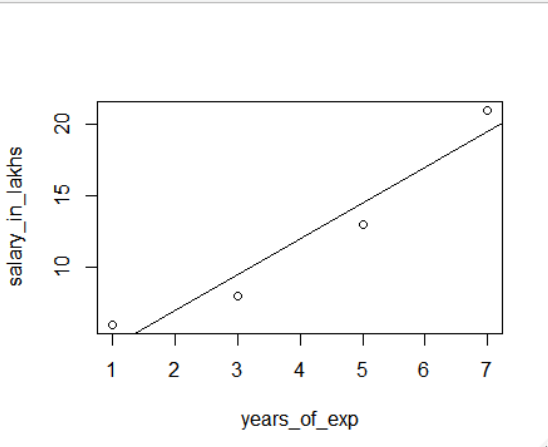
F-statistic: 27.78 on 1 and 2 DF, p-value: 0.03417

>

> plot(salary\_in\_lakhs~ years\_of\_exp, data=employee.data)

> abline(model)

>



**Practical no: 3 Linear regressions**

**b) Logistic regression**

**Aim: Take the in-built data from ISLR package and apply generalized logistic regression to find whether a person would be defaulter or not; considering input as student, income and balance.**

**Source code:**

install.packages("ISLR")

library(ISLR)

#load dataset

data <- ISLR::Default

print (head(ISLR::Default))

#view summary of dataset summary(data)

#find total observations in dataset

nrow(data)

#Create Training and Test Samples

#split the dataset into a training set to train the model on and a testing set to test the model set.seed(1)

#Use 70% of dataset as training set and remaining 30% as testing set sample <- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.7,0.3))

print (sample)

train <- data[sample, ]

test <- data[!sample, ]

nrow(train)

nrow(test)

# Fit the Logistic Regression Model

# use the glm (general linear model) function and specify family="binomial"

#so that R fits a logistic regression model to the dataset

model <- glm(default~student+balance+income, family="binomial", data=train)

#view model summary

summary(model)

#Model Diagnostics

install.packages("InformationValue")

library(InformationValue)

predicted <- predict(model, test, type="response")

confusionMatrix(test$default, predicted)

**Output:**

install.packages("ISLR")

Installing package into ‘C:/Users/zende/AppData/Local/R/win-library/4.2’

(as ‘lib’ is unspecified)

--- Please select a CRAN mirror for use in this session ---

trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/ISLR\_1.4.zip'

Content type 'application/zip' length 2924120 bytes (2.8 MB)

downloaded 2.8 MB

package ‘ISLR’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\zende\AppData\Local\Temp\RtmpCCOj2f\downloaded\_packages

> library(ISLR)

>

> data <- ISLR::Default

>

> print(head(ISLR::Default))

default student balance income

1 No No 729.5265 44361.625

2 No Yes 817.1804 12106.135

3 No No 1073.5492 31767.139

4 No No 529.2506 35704.494

5 No No 785.6559 38463.496

6 No Yes 919.5885 7491.559

> summary(data)

default student balance income

No :9667 No :7056 Min. : 0.0 Min. : 772

Yes: 333 Yes:2944 1st Qu.: 481.7 1st Qu.:21340

Median : 823.6 Median :34553

Mean : 835.4 Mean :33517

3rd Qu.:1166.3 3rd Qu.:43808

Max. :2654.3 Max. :73554

>

> nrow(data)

[1] 10000

>

> set.seed(1)

>

> sample<- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.7,0.3))

> print(sample)

[1] TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[12] TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE FALSE TRUE

[23] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[34] TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE

[45] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[56] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[67] TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE

[78] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[89] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE

[100] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[111] FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[122] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[133] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[144] TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[155] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[166] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[177] TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[188] FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[199] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[210] FALSE FALSE TRUE FALSE FALSE FALSE TRUE TRUE FALSE FALSE TRUE

[221] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[232] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[243] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[254] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[265] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[276] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[287] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[298] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[309] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE

[320] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE

[331] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[342] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[353] FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE

[364] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[375] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[386] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[397] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE

[408] FALSE TRUE FALSE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[419] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[430] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[441] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[452] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[463] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[474] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[485] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[496] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[507] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[518] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE FALSE

[529] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[540] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[551] FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[562] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[573] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[584] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[595] FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE

[606] FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[617] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[628] TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[639] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE

[650] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[661] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[672] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[683] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[694] TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[705] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE

[716] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE

[727] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE

[738] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[749] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[760] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE

[771] TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[782] FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[793] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[804] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[815] TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE FALSE

[826] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE

[837] TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[848] TRUE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE

[859] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[870] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[881] TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[892] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[903] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[914] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE

[925] FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[936] FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[947] TRUE FALSE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[958] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[969] FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[980] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[991] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[1002] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[1013] TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[1024] FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[1035] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[1046] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[1057] FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[1068] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[1079] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[1090] TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE

[1101] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1112] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[1123] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1134] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[1145] FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE

[1156] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1167] FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[1178] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[1189] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[1200] TRUE TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE

[1211] TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[1222] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[1233] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[1244] TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE

[1255] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[1266] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[1277] TRUE FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[1288] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[1299] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[1310] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[1321] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1332] FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[1343] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[1354] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[1365] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[1376] FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[1387] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[1398] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[1409] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[1420] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[1431] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[1442] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1453] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[1464] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[1475] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[1486] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[1497] FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[1508] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE

[1519] TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE FALSE TRUE

[1530] FALSE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[1541] TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[1552] TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE

[1563] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[1574] FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE FALSE FALSE FALSE

[1585] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[1596] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[1607] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE

[1618] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE

[1629] TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[1640] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[1651] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[1662] TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[1673] FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[1684] FALSE FALSE FALSE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE

[1695] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1706] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[1717] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE

[1728] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1739] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1750] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[1761] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[1772] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[1783] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[1794] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[1805] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[1816] TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[1827] TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE TRUE TRUE

[1838] TRUE FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[1849] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[1860] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1871] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[1882] FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1893] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[1904] FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[1915] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[1926] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[1937] FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[1948] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[1959] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[1970] FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[1981] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[1992] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[2003] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[2014] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[2025] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[2036] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[2047] FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[2058] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2069] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE

[2080] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[2091] FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[2102] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[2113] FALSE FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE

[2124] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[2135] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2146] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2157] FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE

[2168] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[2179] TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[2190] TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[2201] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[2212] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[2223] FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[2234] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2245] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[2256] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[2267] TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE

[2278] TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[2289] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[2300] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE

[2311] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2322] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[2333] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2344] FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE

[2355] FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2366] FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[2377] FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[2388] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2399] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE

[2410] TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[2421] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[2432] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE

[2443] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[2454] FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[2465] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2476] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[2487] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[2498] FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE

[2509] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE

[2520] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[2531] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE

[2542] TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[2553] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[2564] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2575] FALSE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2586] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[2597] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2608] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[2619] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[2630] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE

[2641] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[2652] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2663] TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE

[2674] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2685] TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[2696] FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[2707] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2718] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE

[2729] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[2740] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE

[2751] TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[2762] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE

[2773] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[2784] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[2795] FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[2806] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[2817] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[2828] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[2839] FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[2850] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[2861] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE

[2872] FALSE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[2883] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[2894] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[2905] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[2916] FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[2927] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE

[2938] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[2949] FALSE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[2960] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[2971] FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[2982] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE

[2993] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3004] TRUE FALSE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE

[3015] FALSE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[3026] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[3037] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3048] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[3059] FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE FALSE

[3070] FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE

[3081] FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[3092] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[3103] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[3114] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[3125] TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[3136] TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[3147] TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[3158] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[3169] FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[3180] FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[3191] TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[3202] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[3213] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[3224] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[3235] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[3246] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[3257] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3268] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[3279] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[3290] FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE

[3301] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[3312] FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE

[3323] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[3334] TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[3345] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[3356] TRUE FALSE FALSE FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[3367] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[3378] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[3389] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3400] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[3411] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[3422] FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE

[3433] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[3444] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE

[3455] FALSE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[3466] TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[3477] TRUE TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE FALSE

[3488] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[3499] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[3510] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[3521] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[3532] FALSE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[3543] TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[3554] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[3565] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[3576] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[3587] FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE

[3598] TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE

[3609] TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[3620] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[3631] FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[3642] TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE FALSE

[3653] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[3664] TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE FALSE FALSE

[3675] TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE

[3686] TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE

[3697] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE TRUE FALSE

[3708] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[3719] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[3730] TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[3741] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[3752] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[3763] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3774] TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[3785] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE

[3796] TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE

[3807] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[3818] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[3829] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[3840] FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[3851] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[3862] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[3873] FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[3884] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[3895] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[3906] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[3917] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[3928] TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[3939] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[3950] FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[3961] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[3972] TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[3983] FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[3994] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4005] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[4016] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[4027] TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[4038] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[4049] FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[4060] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[4071] FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[4082] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[4093] TRUE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE TRUE

[4104] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4115] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4126] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4137] TRUE FALSE FALSE TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE

[4148] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[4159] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[4170] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[4181] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4192] FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[4203] TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE

[4214] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[4225] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[4236] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[4247] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[4258] TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[4269] FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[4280] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE FALSE

[4291] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4302] TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[4313] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[4324] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE

[4335] TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[4346] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[4357] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4368] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4379] TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[4390] TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[4401] FALSE TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[4412] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4423] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[4434] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[4445] TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[4456] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[4467] TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[4478] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[4489] TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[4500] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[4511] TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[4522] FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE

[4533] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[4544] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4555] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE TRUE TRUE

[4566] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[4577] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[4588] FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[4599] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[4610] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[4621] FALSE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[4632] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4643] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[4654] TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[4665] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[4676] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE FALSE

[4687] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[4698] FALSE FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE

[4709] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[4720] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE

[4731] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE

[4742] FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[4753] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[4764] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[4775] FALSE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE TRUE

[4786] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[4797] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[4808] FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE

[4819] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4830] TRUE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4841] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4852] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[4863] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4874] FALSE TRUE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE

[4885] TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE

[4896] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[4907] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[4918] TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE TRUE TRUE FALSE

[4929] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[4940] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[4951] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE

[4962] TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[4973] FALSE FALSE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[4984] TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE

[4995] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[5006] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[5017] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[5028] TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[5039] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[5050] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[5061] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[5072] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE

[5083] FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE

[5094] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[5105] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5116] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[5127] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[5138] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[5149] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[5160] FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[5171] TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[5182] FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[5193] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE

[5204] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[5215] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[5226] FALSE FALSE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE

[5237] TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[5248] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[5259] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5270] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5281] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[5292] TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE TRUE

[5303] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[5314] TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE

[5325] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5336] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5347] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE

[5358] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[5369] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE

[5380] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[5391] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[5402] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[5413] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5424] TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[5435] FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE

[5446] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5457] FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE

[5468] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[5479] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[5490] TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE TRUE FALSE

[5501] TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5512] TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE

[5523] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5534] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[5545] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[5556] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE

[5567] FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE TRUE FALSE

[5578] FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[5589] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE

[5600] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE FALSE

[5611] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[5622] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[5633] TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5644] TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE

[5655] FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE

[5666] TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[5677] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[5688] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5699] FALSE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE

[5710] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[5721] TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE

[5732] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[5743] FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[5754] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[5765] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[5776] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[5787] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[5798] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5809] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[5820] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE

[5831] TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[5842] TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[5853] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[5864] TRUE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[5875] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[5886] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[5897] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[5908] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[5919] FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[5930] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[5941] FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5952] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[5963] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[5974] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[5985] TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE

[5996] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[6007] TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE

[6018] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[6029] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[6040] TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[6051] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[6062] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6073] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[6084] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[6095] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[6106] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[6117] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[6128] FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE FALSE

[6139] FALSE FALSE FALSE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[6150] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE

[6161] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE

[6172] TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[6183] TRUE TRUE FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE

[6194] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6205] FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[6216] TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[6227] FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE FALSE TRUE

[6238] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6249] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[6260] TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[6271] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[6282] FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[6293] FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[6304] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[6315] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6326] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[6337] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[6348] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6359] FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE

[6370] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[6381] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6392] TRUE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE

[6403] FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[6414] TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[6425] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[6436] FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[6447] FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6458] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[6469] TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE

[6480] FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE

[6491] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE

[6502] TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE

[6513] FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[6524] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[6535] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[6546] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[6557] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6568] TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[6579] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[6590] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE

[6601] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[6612] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE FALSE

[6623] FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE

[6634] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[6645] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[6656] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[6667] TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[6678] FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[6689] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[6700] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[6711] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6722] TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE

[6733] TRUE FALSE FALSE TRUE FALSE FALSE TRUE TRUE FALSE FALSE FALSE

[6744] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[6755] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[6766] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE

[6777] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[6788] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6799] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE

[6810] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[6821] FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[6832] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE

[6843] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[6854] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[6865] TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE

[6876] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[6887] TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[6898] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[6909] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6920] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[6931] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[6942] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[6953] TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE

[6964] FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[6975] TRUE TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE TRUE

[6986] TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE FALSE

[6997] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[7008] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[7019] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[7030] FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[7041] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7052] FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7063] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE

[7074] TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[7085] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[7096] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[7107] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[7118] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[7129] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[7140] TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[7151] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[7162] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE

[7173] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[7184] TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[7195] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE

[7206] FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[7217] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE

[7228] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7239] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[7250] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[7261] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[7272] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7283] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[7294] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[7305] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[7316] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[7327] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE TRUE

[7338] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[7349] FALSE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[7360] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[7371] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[7382] FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[7393] TRUE FALSE FALSE FALSE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[7404] FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE

[7415] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7426] FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE

[7437] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE

[7448] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[7459] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[7470] TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE

[7481] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE

[7492] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[7503] TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[7514] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7525] TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[7536] FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE

[7547] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[7558] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE

[7569] FALSE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[7580] FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[7591] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7602] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[7613] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[7624] FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE

[7635] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[7646] FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE

[7657] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[7668] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[7679] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[7690] TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE FALSE

[7701] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE

[7712] TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[7723] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[7734] TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE

[7745] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7756] TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[7767] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[7778] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE

[7789] TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[7800] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE

[7811] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[7822] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7833] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[7844] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[7855] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7866] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[7877] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE

[7888] TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE FALSE

[7899] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[7910] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[7921] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE FALSE

[7932] TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[7943] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[7954] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[7965] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[7976] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[7987] FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[7998] FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[8009] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE FALSE

[8020] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8031] TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[8042] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8053] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[8064] TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[8075] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[8086] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[8097] FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[8108] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[8119] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8130] TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[8141] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[8152] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[8163] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[8174] FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[8185] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[8196] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[8207] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE

[8218] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE

[8229] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[8240] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[8251] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8262] TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE FALSE TRUE TRUE

[8273] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[8284] FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE FALSE FALSE

[8295] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE

[8306] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[8317] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[8328] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[8339] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[8350] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[8361] TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[8372] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[8383] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[8394] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE

[8405] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[8416] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE FALSE

[8427] FALSE FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE

[8438] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8449] TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[8460] FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[8471] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[8482] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[8493] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[8504] TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8515] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[8526] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[8537] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[8548] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[8559] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[8570] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8581] FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[8592] FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[8603] FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE

[8614] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE

[8625] TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE

[8636] FALSE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE

[8647] FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE

[8658] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[8669] FALSE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE

[8680] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[8691] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE FALSE

[8702] FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE

[8713] TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[8724] TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE

[8735] FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[8746] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8757] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[8768] FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[8779] FALSE TRUE FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[8790] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8801] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[8812] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[8823] TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE TRUE TRUE

[8834] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[8845] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE

[8856] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[8867] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[8878] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[8889] FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE

[8900] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE

[8911] FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE

[8922] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE

[8933] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8944] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[8955] TRUE FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE TRUE TRUE

[8966] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE

[8977] TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE

[8988] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[8999] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[9010] TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[9021] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[9032] FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE

[9043] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9054] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9065] TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE

[9076] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9087] FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE

[9098] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE FALSE

[9109] FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE

[9120] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[9131] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[9142] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE FALSE TRUE FALSE

[9153] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[9164] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[9175] FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[9186] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE

[9197] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[9208] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[9219] FALSE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE

[9230] FALSE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE

[9241] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE

[9252] TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[9263] TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE

[9274] TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE

[9285] TRUE TRUE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[9296] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[9307] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[9318] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[9329] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[9340] TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE

[9351] TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE TRUE

[9362] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[9373] TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE

[9384] TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE FALSE FALSE FALSE

[9395] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE

[9406] TRUE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE

[9417] TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE FALSE

[9428] TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[9439] TRUE FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE

[9450] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[9461] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[9472] FALSE FALSE TRUE TRUE FALSE FALSE TRUE FALSE FALSE TRUE FALSE

[9483] FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE

[9494] TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[9505] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9516] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[9527] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE

[9538] TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[9549] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[9560] TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9571] FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[9582] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[9593] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9604] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[9615] FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE

[9626] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE

[9637] TRUE TRUE TRUE FALSE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[9648] FALSE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[9659] TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE

[9670] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9681] FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE FALSE

[9692] TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE

[9703] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE

[9714] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE TRUE

[9725] TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE TRUE

[9736] FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE

[9747] FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE

[9758] FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE

[9769] TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE

[9780] TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[9791] TRUE TRUE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE

[9802] TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE

[9813] TRUE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE TRUE

[9824] TRUE TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE TRUE TRUE

[9835] TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE

[9846] TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE

[9857] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE

[9868] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE

[9879] TRUE FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE

[9890] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[9901] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE

[9912] TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE

[9923] FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[9934] TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE FALSE TRUE

[9945] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE

[9956] TRUE FALSE TRUE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE

[9967] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[9978] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE TRUE

[9989] TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE TRUE TRUE TRUE

[10000] TRUE

>

>

> train <- data[sample,]

> test <- data[!sample,]

>

> nrow(train)

[1] 6964

> nrow(test)

[1] 3036

>

> model <- glm(default~student + balance + income,

+ family = "binomial", data = train)

>

> summary(model)

Call:

glm(formula = default ~ student + balance + income, family = "binomial",

data = train)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.5586 -0.1353 -0.0519 -0.0177 3.7973

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -1.148e+01 6.234e-01 -18.412 <2e-16 \*\*\*

studentYes -4.933e-01 2.857e-01 -1.726 0.0843 .

balance 5.988e-03 2.938e-04 20.384 <2e-16 \*\*\*

income 7.857e-06 9.965e-06 0.788 0.4304

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 2021.1 on 6963 degrees of freedom

Residual deviance: 1065.4 on 6960 degrees of freedom

AIC: 1073.4

Number of Fisher Scoring iterations: 8

> install.packages("InformationValue")

Installing package into ‘C:/Users/rohit/AppData/Local/R/win-library/4.2’

(as ‘lib’ is unspecified)

trying URL 'https://cloud.r-project.org/bin/windows/contrib/4.2/InformationValue\_1.2.3.zip'

Content type 'application/zip' length 213044 bytes (208 KB)

downloaded 208 KB

package ‘InformationValue’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\ rohit \AppData\Local\Temp\RtmpCCOj2f\downloaded\_packages

> library(InformationValue)

> predicted <- predict(model,test,type="response")

>

> confusionMatrix(test$default,predicted)

No Yes

0 2912 64

1 21 39

>

**Practical 4 a Decision Tree**

**Source Code:**

# Importing the dataset

dataset = read.csv(‘D:\Msc.it part 1 (sem 2)\Big Data Analytics\\Social\_Network\_Ads.csv')

dataset = dataset[3:5]

# Encoding the target feature as factor

dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))

# Splitting the dataset into the Training set and Test set

install.packages('caTools')

library(caTools)

set.seed(123)

split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training\_set = subset(dataset, split == TRUE)

# Feature Scaling

training\_set[-3] = scale(training\_set[-3])

test\_set[-3] = scale(test\_set[-3])

# Fitting Decision Tree Classification to the Training set

install.packages('rpart')

library(rpart)

classifier = rpart(formula = Purchased ~ ., data = training\_set)

# Predicting the Test set results

y\_pred = predict(classifier, newdata = test\_set[-3], type = 'class')

# Making the Confusion Matrix

cm = table(test\_set[, 3], y\_pred)

# Visualising the Training set results

install.packages("ElemStatLearn")

library(ElemStatLearn)

set = training\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set, type = 'class')

plot(set[, -3],

main = 'Decision Tree Classification (Training set)', xlab = 'Age', ylab = 'Estimated Salary', xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

# Visualising the Test set results

library(ElemStatLearn)

set = test\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set, type = 'class')

plot(set[, -3], main = 'Decision Tree Classification (Test set)', xlab = 'Age', ylab = 'Estimated Salary', xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

# Plotting the tree

plot(classifier)

text(classifier)

input: Social\_Network\_Ads.csv

User ID Gender Age EstimatedSalary Purchased

15624510 Male 19 19000 0

15810944 Male 35 20000 0

15668575 Female 26 43000 0

15603246 Female 27 57000 0

15804002 Male 19 76000 0

15728773 Male 27 58000 0

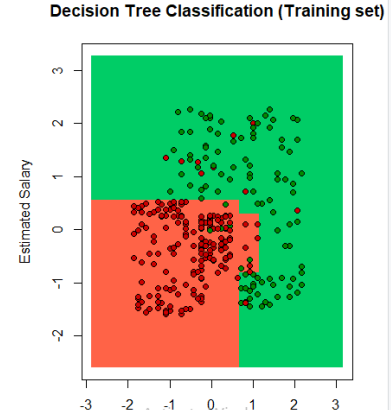
15598044 Female 27 84000 0

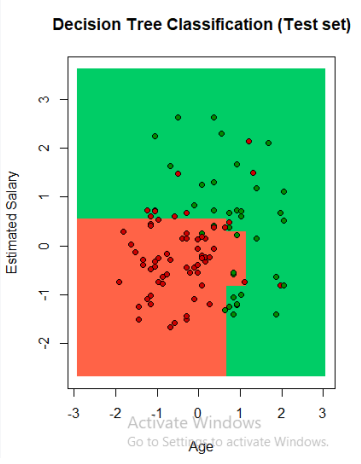
15694829 Female 32 150000 1

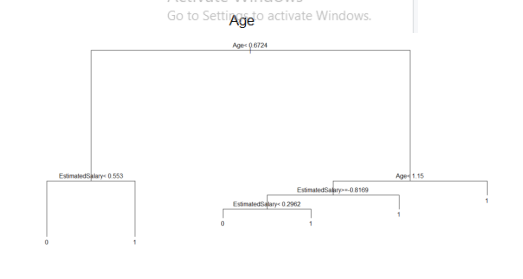
15600575 Male 25 33000 0

15727311 Female 35 65000 0

**OUTPUT:**







**Practical no: 4b Naïve Bayes Classification**

**Source Code:**

# Naive Bayes

# Importing the dataset dataset = read.csv(‘D:\Msc.it part 1 (sem 2)\Big Data Analytics\\Social\_Network\_Ads.csv')

dataset = dataset[3:5]

# Encoding the target feature as factor

dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))

# Splitting the dataset into the Training set and Test set

#install.packages('caTools')

library(caTools)

set.seed(123)

split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training\_set = subset(dataset, split == TRUE)

test\_set = subset(dataset, split == FALSE)

# Feature Scaling

training\_set[-3] = scale(training\_set[-3])

test\_set[-3] = scale(test\_set[-3])

# Fitting Naive Bayes to the Training set

install.packages('e1071')

library(e1071)

classifier = naiveBayes(x = training\_set[-3], y = training\_set$Purchased)

# Predicting the Test set results

y\_pred = predict(classifier, newdata = test\_set[-3])

# Making the Confusion Matrix

cm = table(test\_set[, 3], y\_pred)

print(cm)

# Visualising the Training set results

install.packages("ElemStatLearn")

library(ElemStatLearn)

set = training\_set

print(set)

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set)

plot(set[, -3], main = 'Naive Bayes (Training set)',

xlab = 'Age', ylab = 'Estimated Salary', xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

# Visualising the Test set results

library(ElemStatLearn)

set = test\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set)

plot(set[, -3], main = 'NaiveBayes (Test set)', xlab = 'Age', ylab = 'Estimated Salary', xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

input: Social\_Network\_Ads.csv

User ID Gender Age EstimatedSalary Purchased

15624510 Male 19 19000 0

15810944 Male 35 20000 0

15668575 Female 26 43000 0

15603246 Female 27 57000 0

15804002 Male 19 76000 0

15728773 Male 27 58000 0

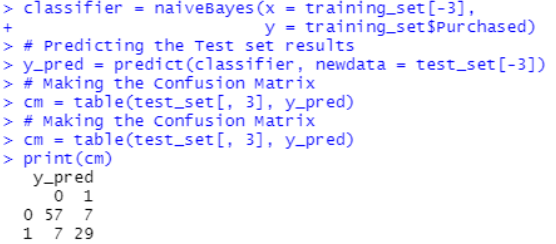
15598044 Female 27 84000 0

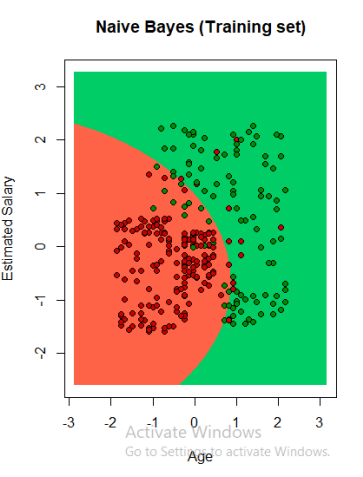
15694829 Female 32 150000 1

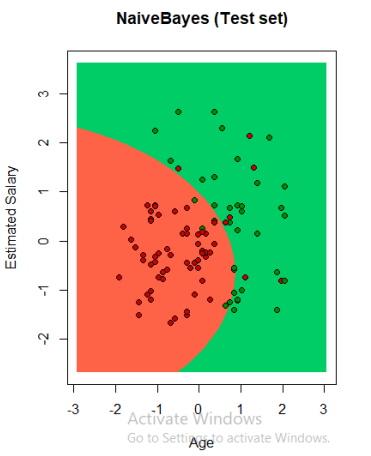
15600575 Male 25 33000 0

15727311 Female 35 65000 0

**Output:**







**Practical 5: Text Analysis**

**Source Code:**

Natural Language Processing

# Importing the dataset

dataset\_original = read.delim('D:\\2020\\Big Data Analytics\\Practical\\P6 NLP\\Restaurant\_Reviews.tsv', quote = '', stringsAsFactors = FALSE)

# Cleaning the texts

install.packages('tm')

install.packages('SnowballC')

library(tm) library(SnowballC)

corpus = VCorpus(VectorSource(dataset\_original$Review))

corpus = tm\_map(corpus, content\_transformer(tolower))

corpus = tm\_map(corpus, removeNumbers)

corpus = tm\_map(corpus, removePunctuation)

corpus = tm\_map(corpus, removeWords, stopwords())

corpus = tm\_map(corpus, stemDocument)

corpus = tm\_map(corpus, stripWhitespace)

# Creating the Bag of Words model

dtm = DocumentTermMatrix(corpus)

dtm = removeSparseTerms(dtm, 0.999)

dataset = as.data.frame(as.matrix(dtm))

dataset$Liked = dataset\_original$Liked

print(dataset$Liked)

# Encoding the target feature as factor

dataset$Liked = factor(dataset$Liked, levels = c(0, 1))

# Splitting the dataset into the Training set and Test set

install.packages('caTools')

library(caTools)

set.seed(123)

split = sample.split(dataset$Liked, SplitRatio = 0.8)

training\_set = subset(dataset, split == TRUE)

test\_set = subset(dataset, split == FALSE)

# Fitting Random Forest Classification to the Training set

install.packages('randomForest')

library(randomForest)

classifier = randomForest(x = training\_set[-692], y = training\_set$Liked, ntree = 10)

# Predicting the Test set results

y\_pred = predict(classifier, newdata = test\_set[-692])

# Making the Confusion Matrix

cm = table(test\_set[, 692], y\_pred)

print(cm)

input

Review Liked

Wow... Loved this place. 1

Crust is not good. 0

Not tasty and the texture was just nasty. 0

Stopped by during the late May bank holiday off Rick Steve recommendation and loved it. 1

The selection on the menu was great and so were the prices. 1 :

:

Overall I was not impressed and would not go back. 0

The whole experience was underwhelming, and I think we'll just go to Ninja Sushi next time. 0 Then, as if I hadn't wasted enough of my life there, they poured salt in the wound by drawing out the time it took to bring the check. 0

**Output:**

