Homework 2 - Lukasz Grzybek

1a

```
#reads BankData
library(readr)
BankData <- read_csv("rr/BankData.csv")
```

```
## New names:
## Rows: 690 Columns: 13
## ── Column specification
## ──────────────────────────────────────────────────────── Delimiter: "," chr
## (1): approval dbl (9): ...1, cont1, cont2, cont3, cont4, cont5, cont6,
## credit.score, ages lgl (3): bool1, bool2, bool3
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

```
summary(BankData)
```

```
##       ...1           cont1           cont2            cont3
##  Min.   :  1.0   Min.   :13.75   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:173.2   1st Qu.:22.60   1st Qu.: 1.000   1st Qu.: 0.165
##  Median :345.5   Median :28.46   Median : 2.750   Median : 1.000
##  Mean   :345.5   Mean   :31.57   Mean   : 4.759   Mean   : 2.223
##  3rd Qu.:517.8   3rd Qu.:38.23   3rd Qu.: 7.207   3rd Qu.: 2.625
##  Max.   :690.0   Max.   :80.25   Max.   :28.000   Max.   :28.500
##                  NA's   :12
##    bool1           bool2             cont4        bool3             cont5
##  Mode :logical   Mode :logical   Min.   : 0.0   Mode :logical   Min.   :   0
##  FALSE:329       FALSE:395       1st Qu.: 0.0   FALSE:374       1st Qu.:  75
##  TRUE :361       TRUE :295       Median : 0.0   TRUE :316       Median : 160
##                                  Mean   : 2.4                   Mean   : 184
##                                  3rd Qu.: 3.0                   3rd Qu.: 276
##                                  Max.   :67.0                   Max.   :2000
##                                                                 NA's   :13
##      cont6            approval          credit.score        ages
##  Min.   :     0.0   Length:690         Min.   :583.7   Min.   :11.00
##  1st Qu.:     0.0   Class :character   1st Qu.:666.7   1st Qu.:31.00
##  Median :     5.0   Mode  :character   Median :697.3   Median :38.00
##  Mean   :  1017.4                      Mean   :696.4   Mean   :39.67
##  3rd Qu.:   395.5                      3rd Qu.:726.4   3rd Qu.:48.00
##  Max.   :100000.0                      Max.   :806.0   Max.   :84.00
##
```

```
library(ggplot2)

library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.2.1      ✓ stringr 1.4.1
## ✓ purrr   0.3.4      ✓ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
```

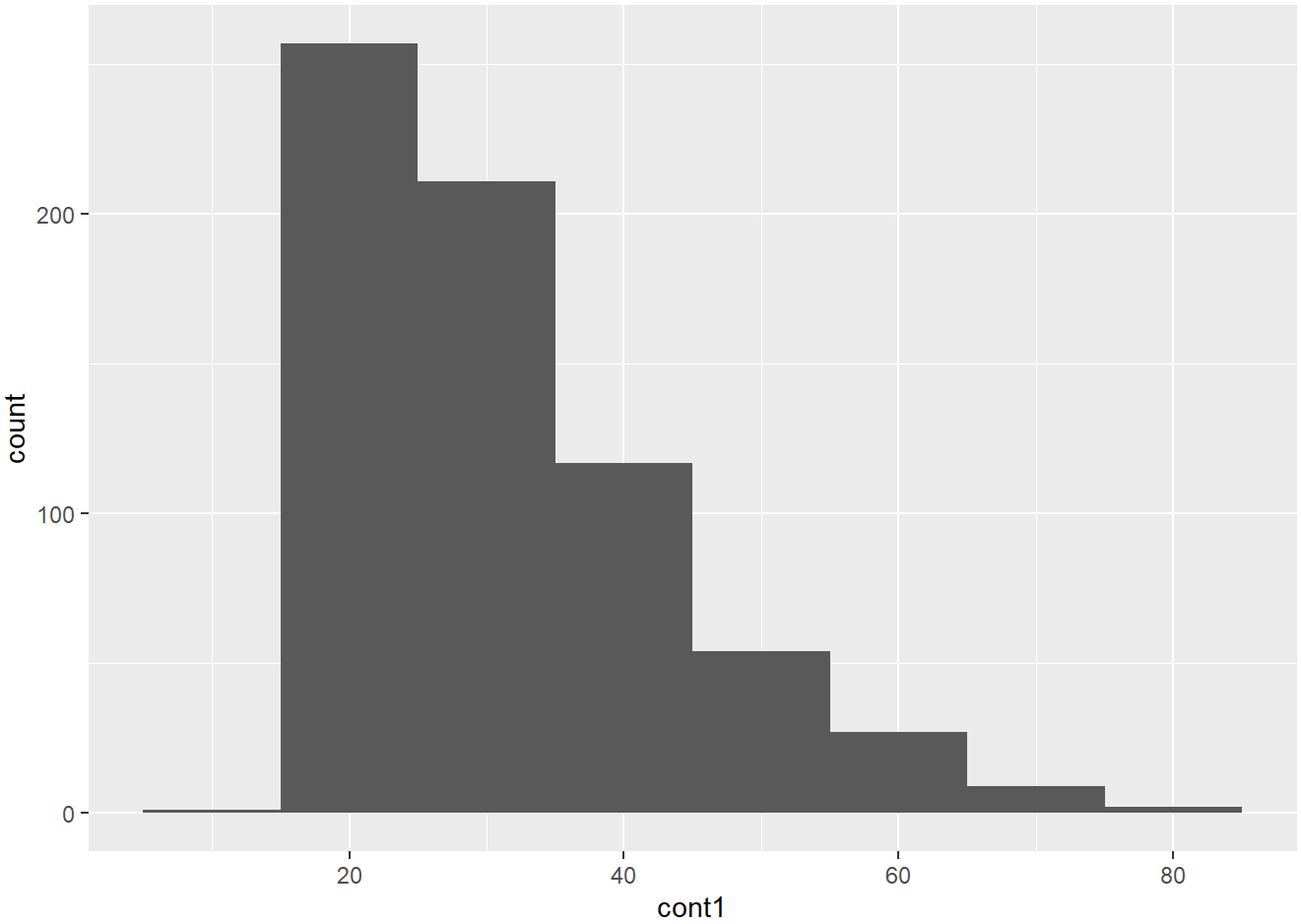
```
#creates dataframe
df <- data.frame(BankData)
head(df)
```

	...1 <dbl>	cont1 <dbl>	cont2 <dbl>	cont3 <dbl>	bool1 <lgl>	bool2 <lgl>	cont4 <dbl>	bool3 <lgl>	cont5 <dbl>
1	1	30.83	0.000	1.25	TRUE	TRUE	1	FALSE	202
2	2	58.67	4.460	3.04	TRUE	TRUE	6	FALSE	43
3	3	24.50	0.500	1.50	TRUE	FALSE	0	FALSE	280
4	4	27.83	1.540	3.75	TRUE	TRUE	5	TRUE	100
5	5	20.17	5.625	1.71	TRUE	FALSE	0	FALSE	120
6	6	32.08	4.000	2.50	TRUE	FALSE	0	TRUE	360

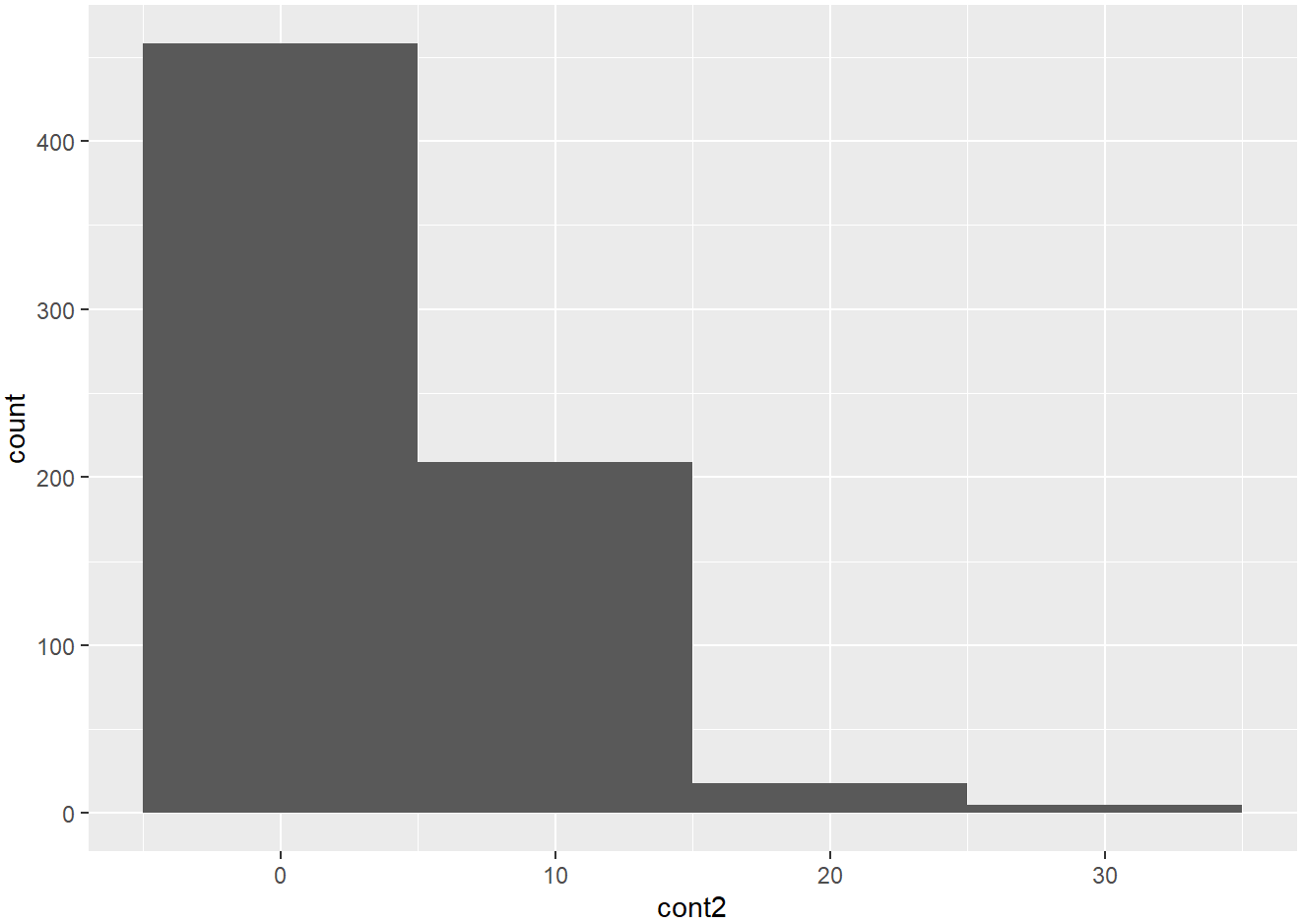
6 rows | 1-10 of 14 columns

```
#plots cont1 variable using a histogram
cont1_hist <- ggplot(df, aes(cont1)) + geom_histogram(binwidth = 10)
cont1_hist
```

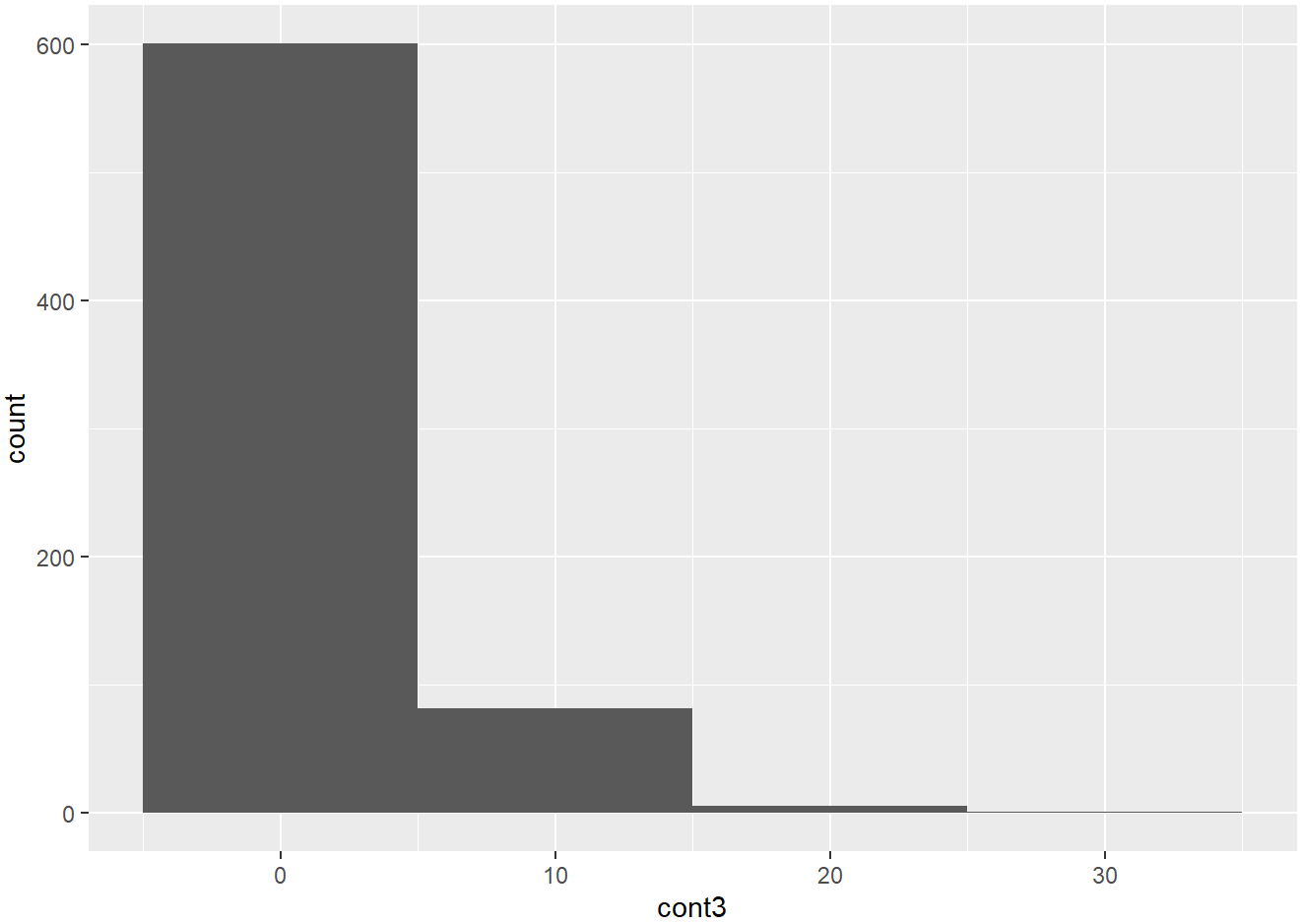
```
## Warning: Removed 12 rows containing non-finite values (stat_bin).
```



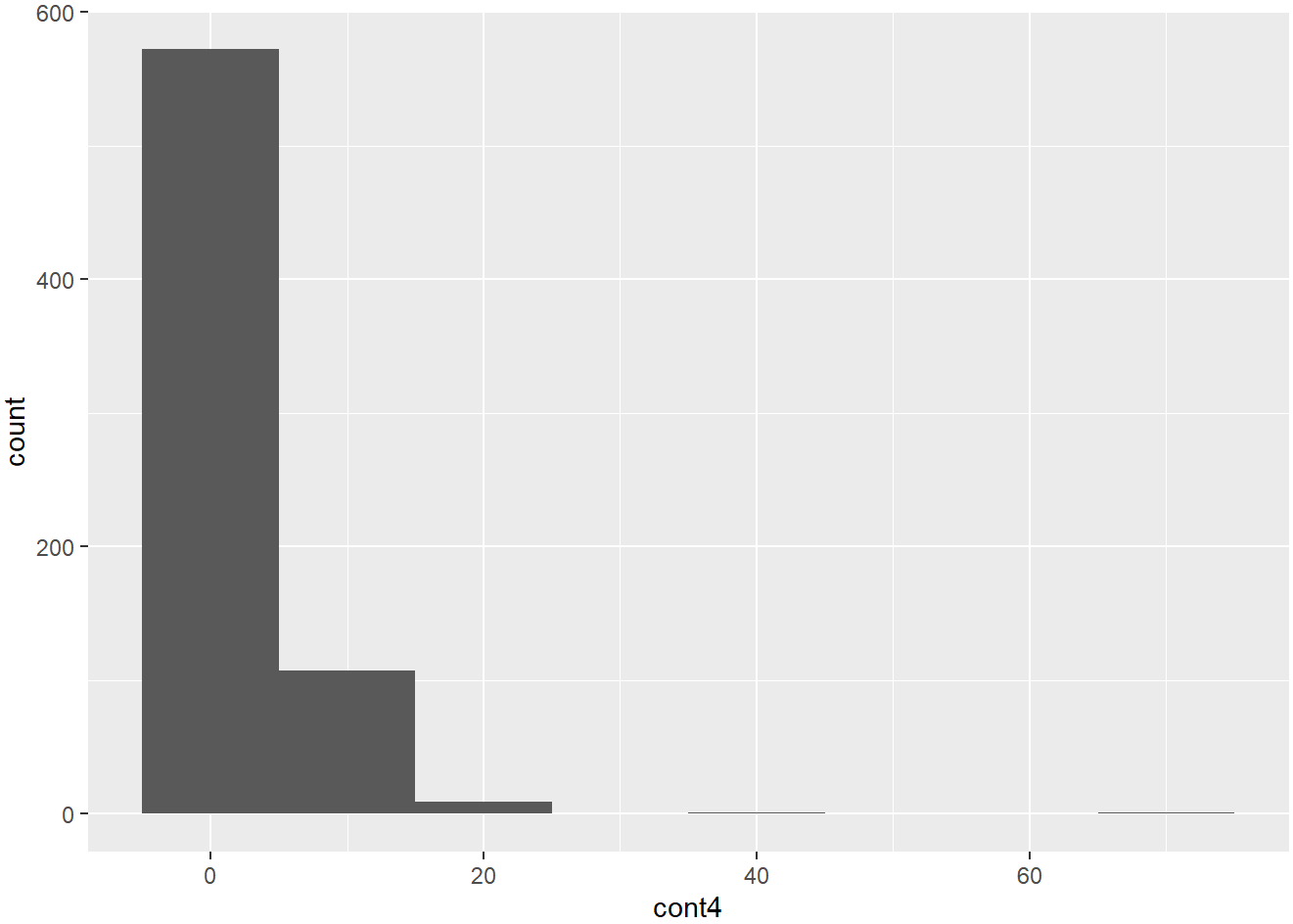

```
#plots cont2 variable using a histogram
cont2_hist <- ggplot(df, aes(cont2)) + geom_histogram(binwidth = 10)
cont2_hist
```




```
#plots cont3 variable using a histogram
cont3_hist <- ggplot(df, aes(cont3)) + geom_histogram(binwidth = 10)
cont3_hist
```

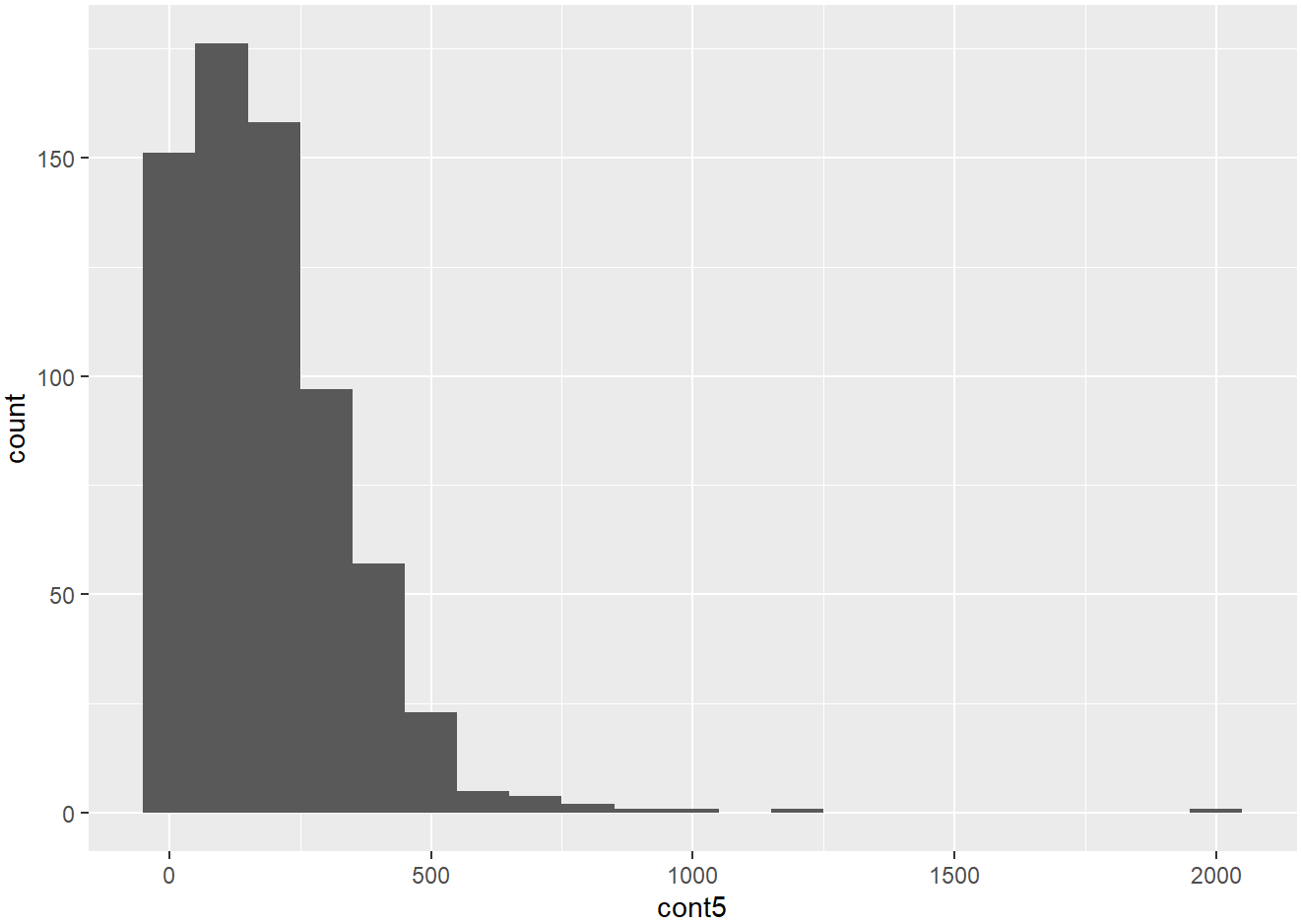



```
#plots cont4 variable using a histogram
cont4_hist <- ggplot(df, aes(cont4)) + geom_histogram(binwidth = 10)
cont4_hist
```

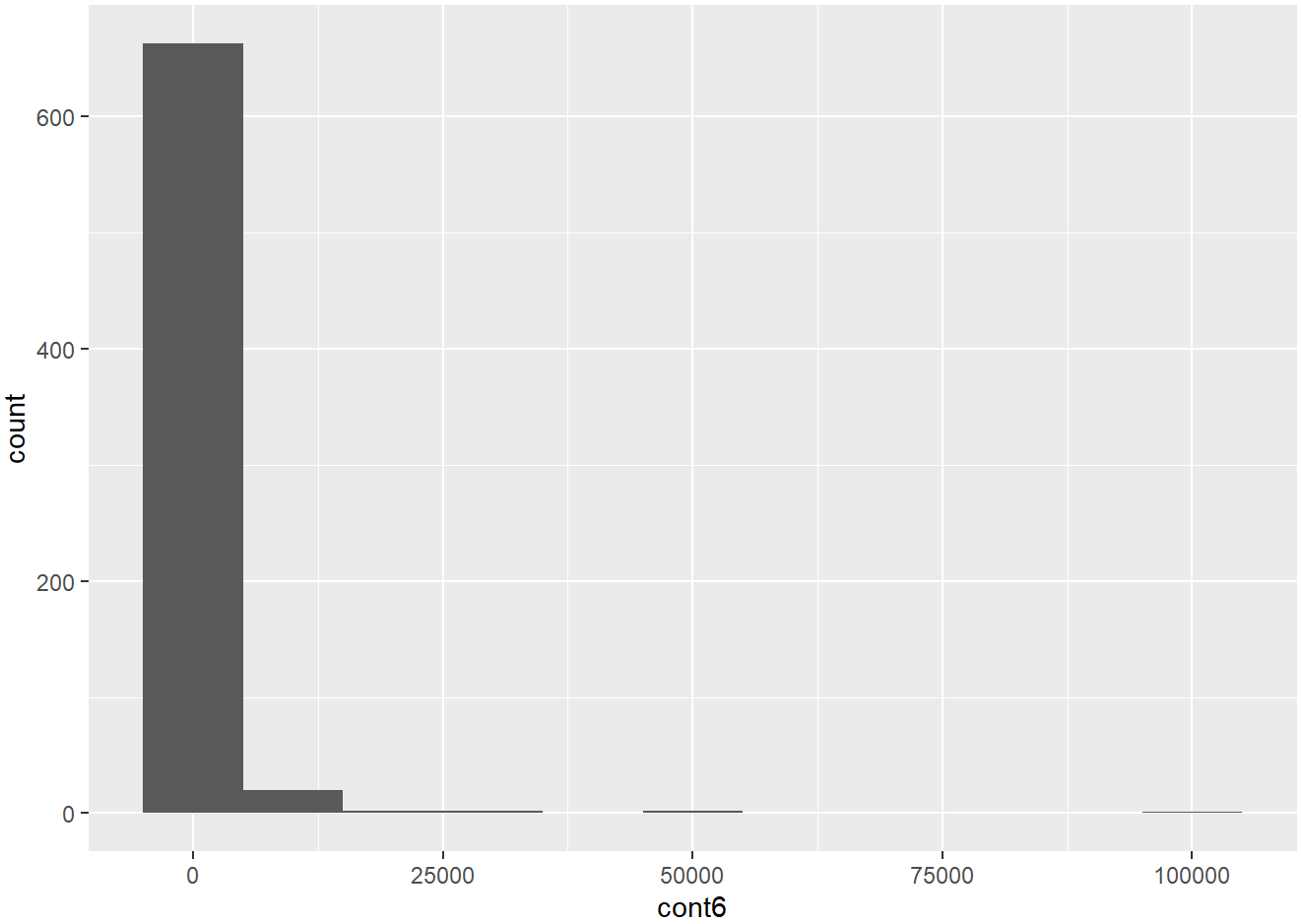



```
#plots cont5 variable using a histogram
cont5_hist <- ggplot(df, aes(cont5)) + geom_histogram(binwidth = 100)
cont5_hist
```

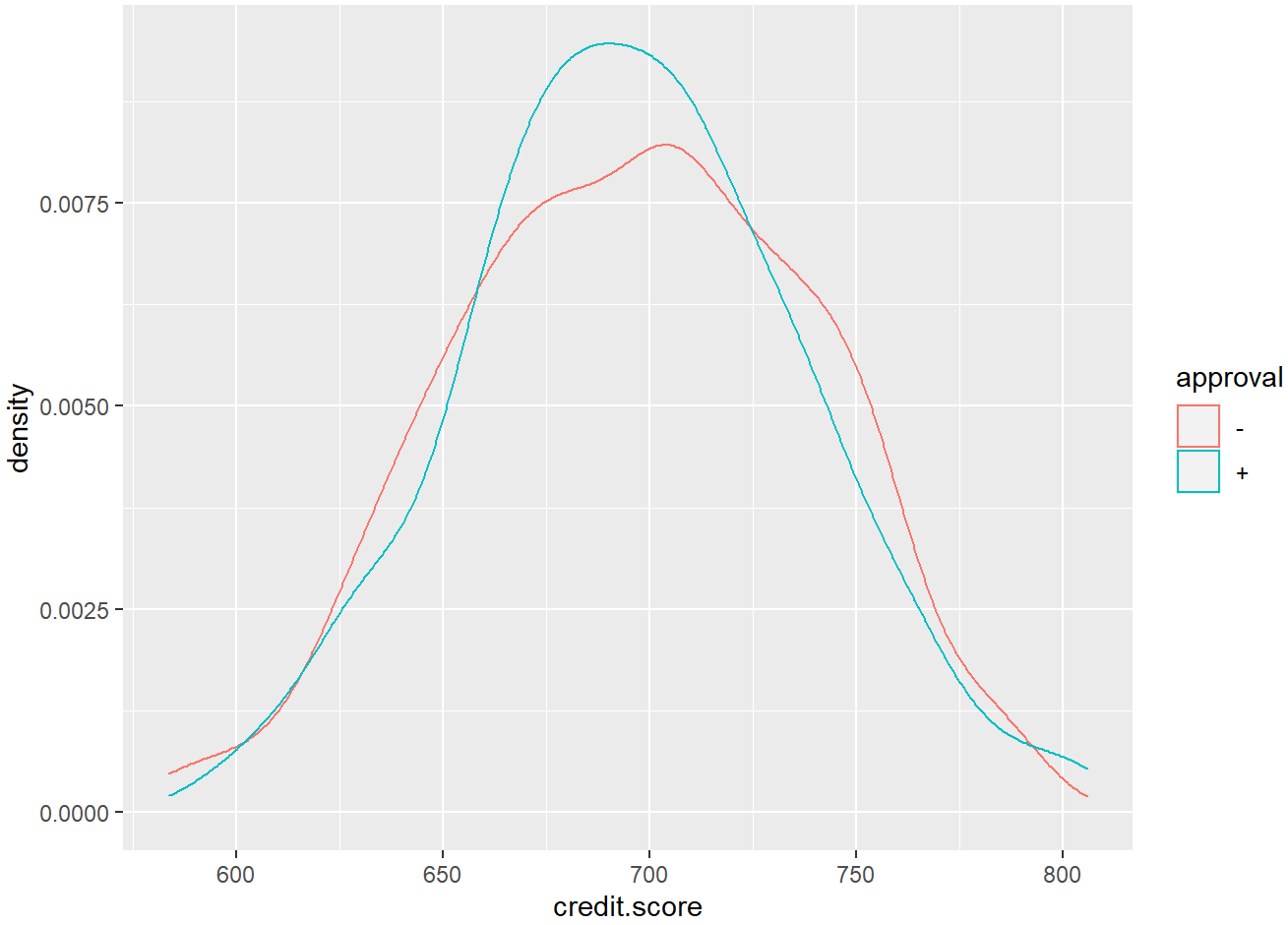
```
## Warning: Removed 13 rows containing non-finite values (stat_bin).
```



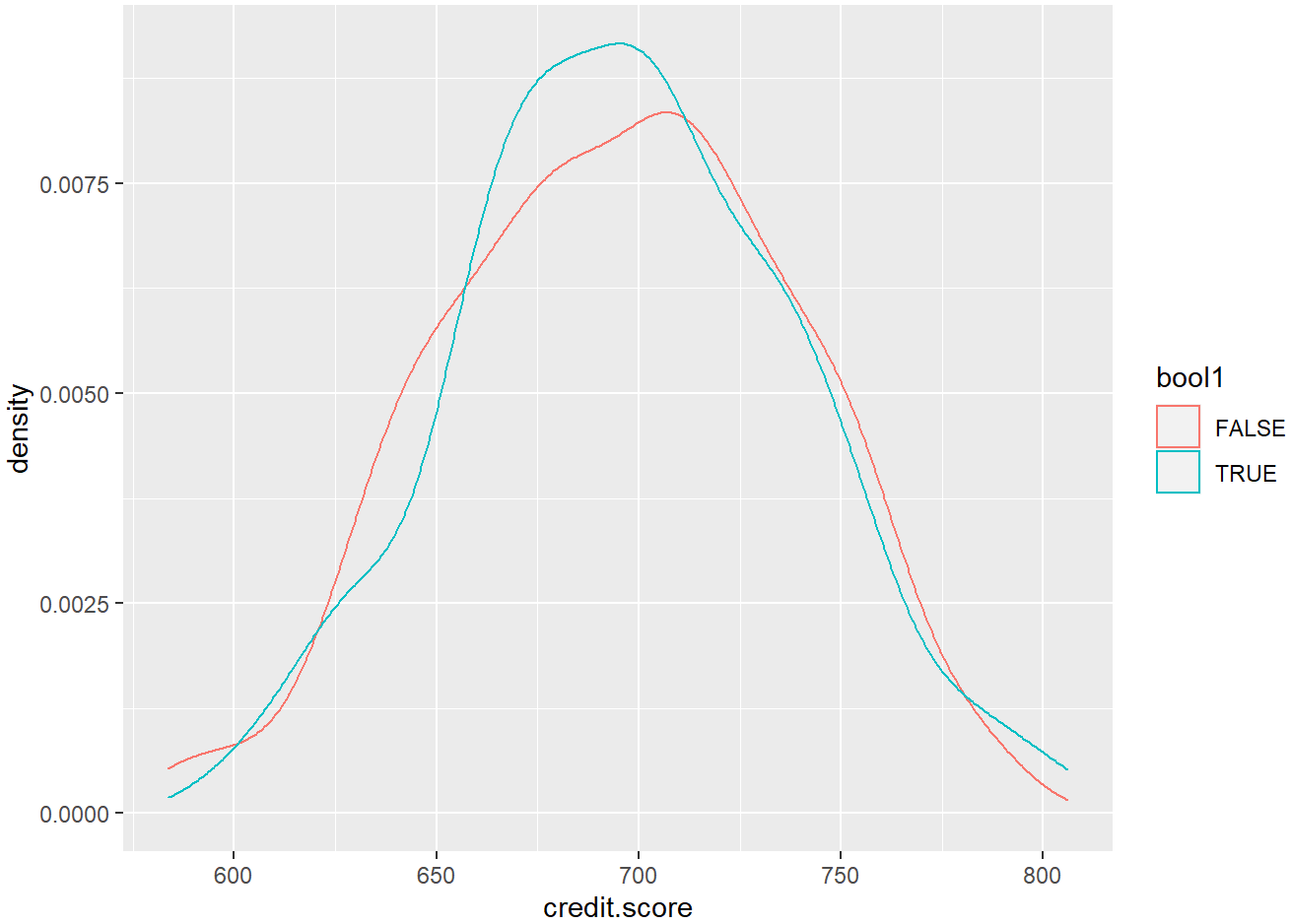

```
#plots cont6 variable using a histogram
cont6_hist <- ggplot(df, aes(cont6)) + geom_histogram(binwidth = 10000)
cont6_hist
```



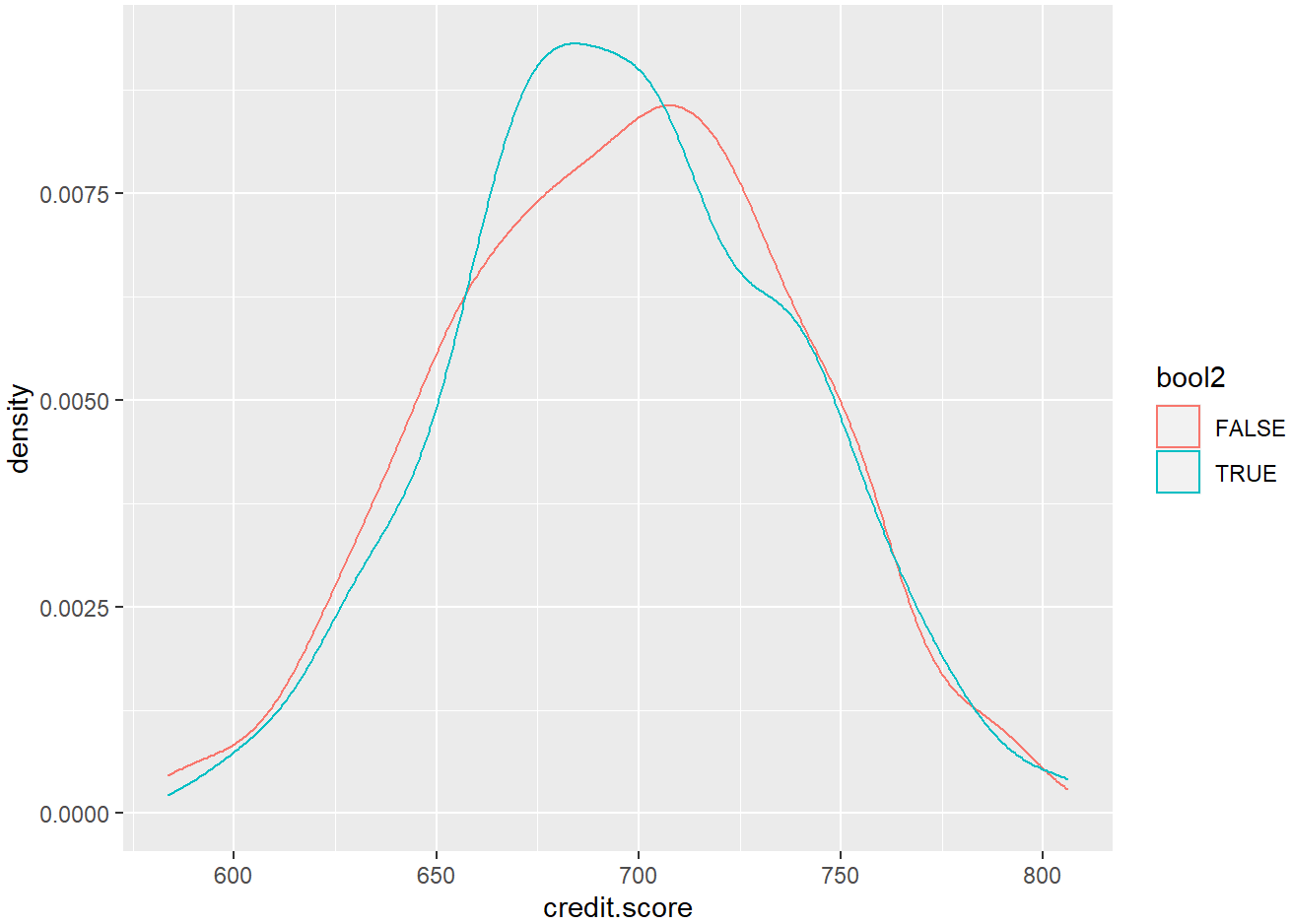

```
#plots densityplot of credit.score variable and the approval variable is used for color
denplot1 <- ggplot(df, aes(x=credit.score, color = approval)) +
  geom_density()

denplot1
```



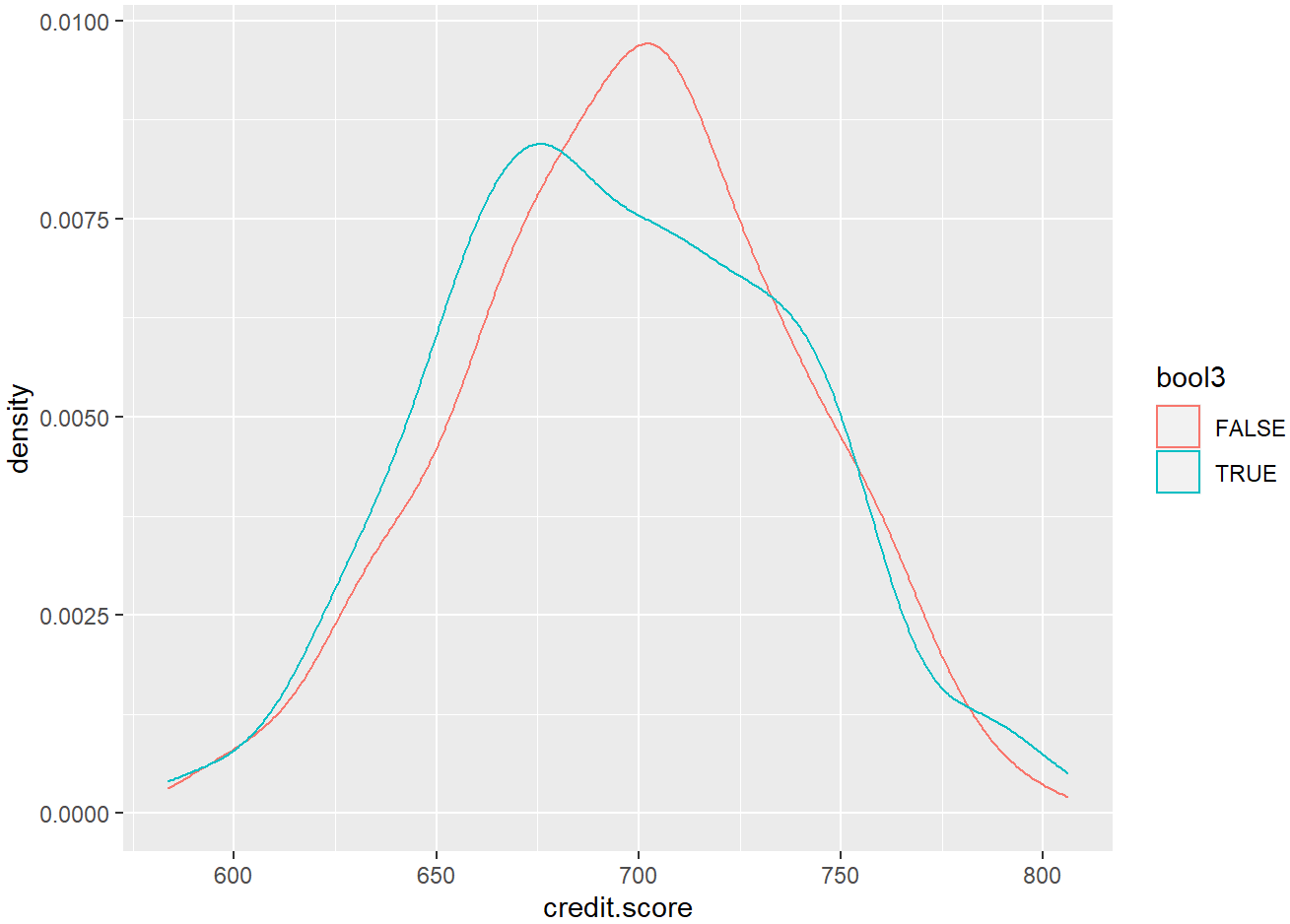

```
#plots densityplot of credit.score variable and the bool1 variable is used for color
denplot2 <- ggplot(df, aes(x=credit.score, color = bool1)) +
  geom_density()

denplot2
```



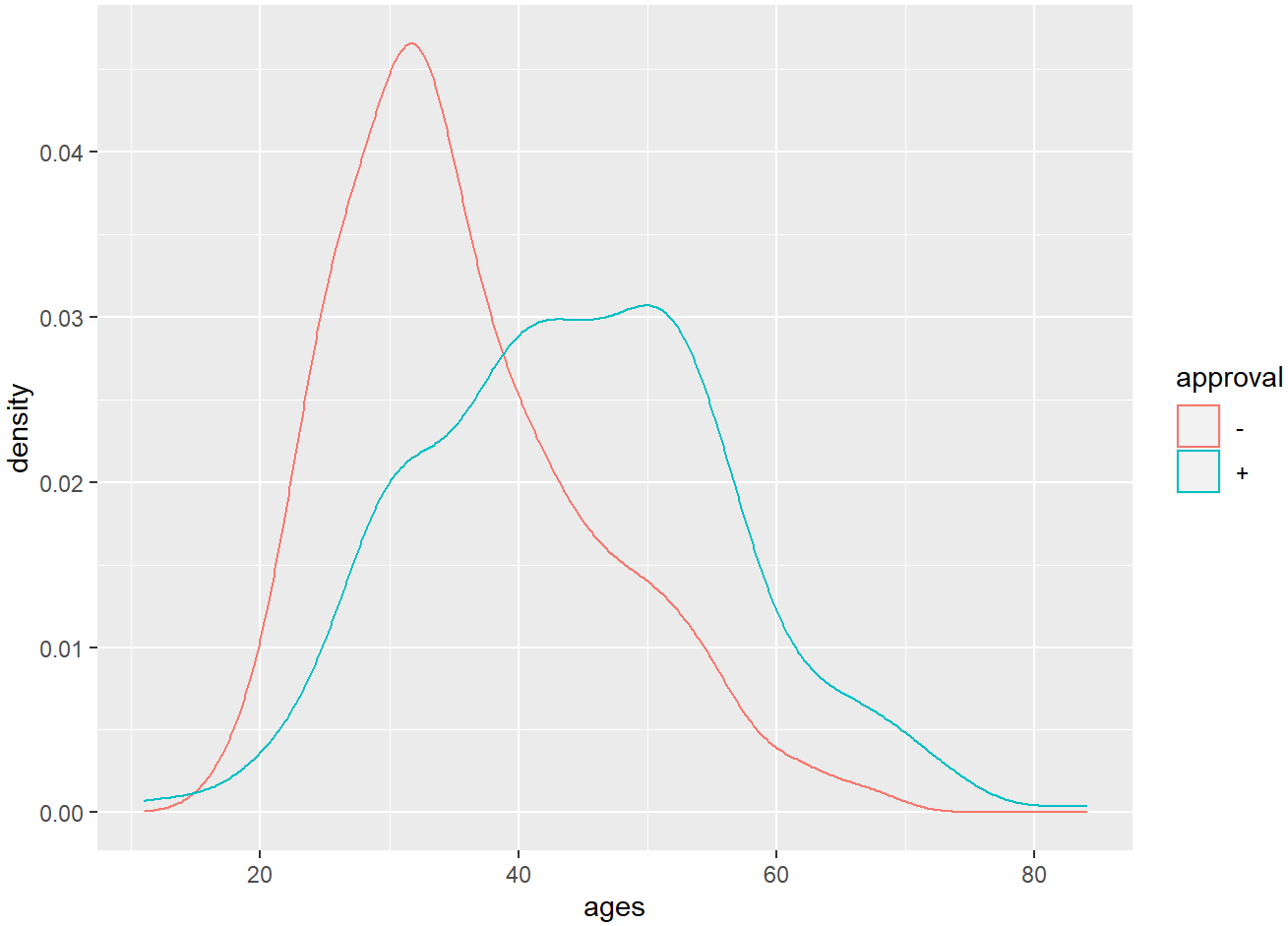

```
#plots densityplot of credit.score variable and the bool2 variable is used for color
denplot3 <- ggplot(df, aes(x=credit.score, color = bool2)) +
  geom_density()

denplot3
```




```
#plots densityplot of credit.score variable and the bool3 variable is used for color
denplot4 <- ggplot(df, aes(x=credit.score, color = bool3)) +
  geom_density()

denplot4
```




```
#plots densityplot of ages variable and the approval variable is used for color
denplot5 <- ggplot(df, aes(x=ages, color = approval)) +
  geom_density()

denplot5
```




```
#1. b and c

library(caret)
```

```
## Loading required package: lattice
```

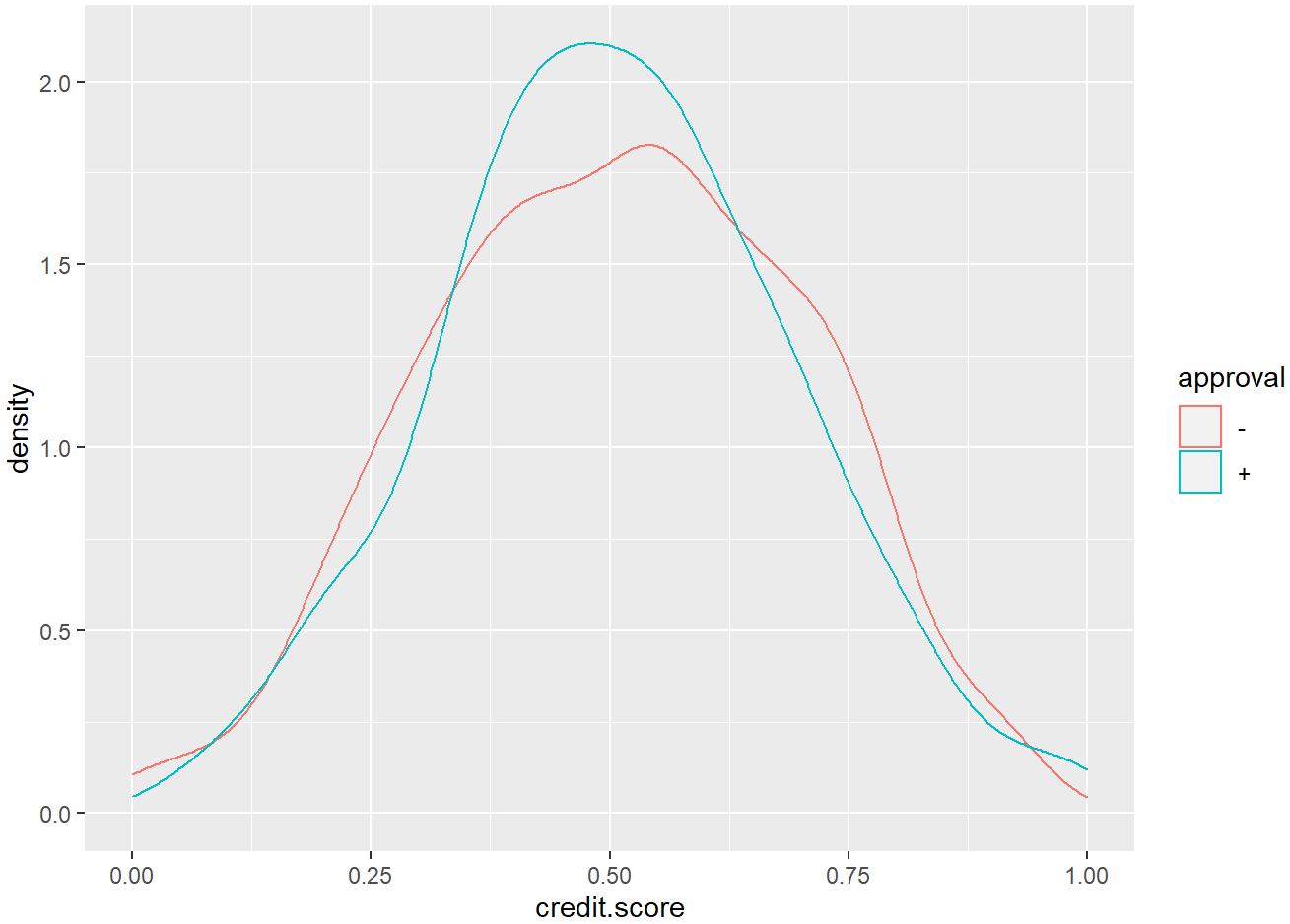
```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
#minmax normalization used on the dataframe
minmax <- preProcess(df, method=c("range"))
norm1 <- predict(minmax, df)

summary(norm1)
```

```
##       ...1          cont1            cont2             cont3
##  Min.   :0.00   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.25   1st Qu.:0.1331   1st Qu.:0.03571   1st Qu.:0.00579
##  Median :0.50   Median :0.2212   Median :0.09821   Median :0.03509
##  Mean   :0.50   Mean   :0.2679   Mean   :0.16995   Mean   :0.07801
##  3rd Qu.:0.75   3rd Qu.:0.3681   3rd Qu.:0.25741   3rd Qu.:0.09211
##  Max.   :1.00   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
##                 NA's   :12
##    bool1           bool2             cont4           bool3
##  Mode :logical   Mode :logical   Min.   :0.00000   Mode :logical
##  FALSE:329       FALSE:395       1st Qu.:0.00000   FALSE:374
##  TRUE :361       TRUE :295       Median :0.00000   TRUE :316
##                                  Mean   :0.03582
##                                  3rd Qu.:0.04478
##                                  Max.   :1.00000
##
##      cont5             cont6            approval          credit.score
##  Min.   :0.00000   Min.   :0.000000   Length:690         Min.   :0.0000
##  1st Qu.:0.03750   1st Qu.:0.000000   Class :character   1st Qu.:0.3737
##  Median :0.08000   Median :0.000050   Mode  :character   Median :0.5110
##  Mean   :0.09201   Mean   :0.010174                      Mean   :0.5070
##  3rd Qu.:0.13800   3rd Qu.:0.003955                      3rd Qu.:0.6418
##  Max.   :1.00000   Max.   :1.000000                      Max.   :1.0000
##  NA's   :13
##       ages
##  Min.   :0.0000
##  1st Qu.:0.2740
##  Median :0.3699
##  Mean   :0.3928
##  3rd Qu.:0.5068
##  Max.   :1.0000
##
```

```
#plots normalized densityplot of credit.score variable and the approval variable is used for color

denplotminmax <- ggplot(norm1, aes(x=credit.score, color = approval)) +
  geom_density()

denplotminmax
```

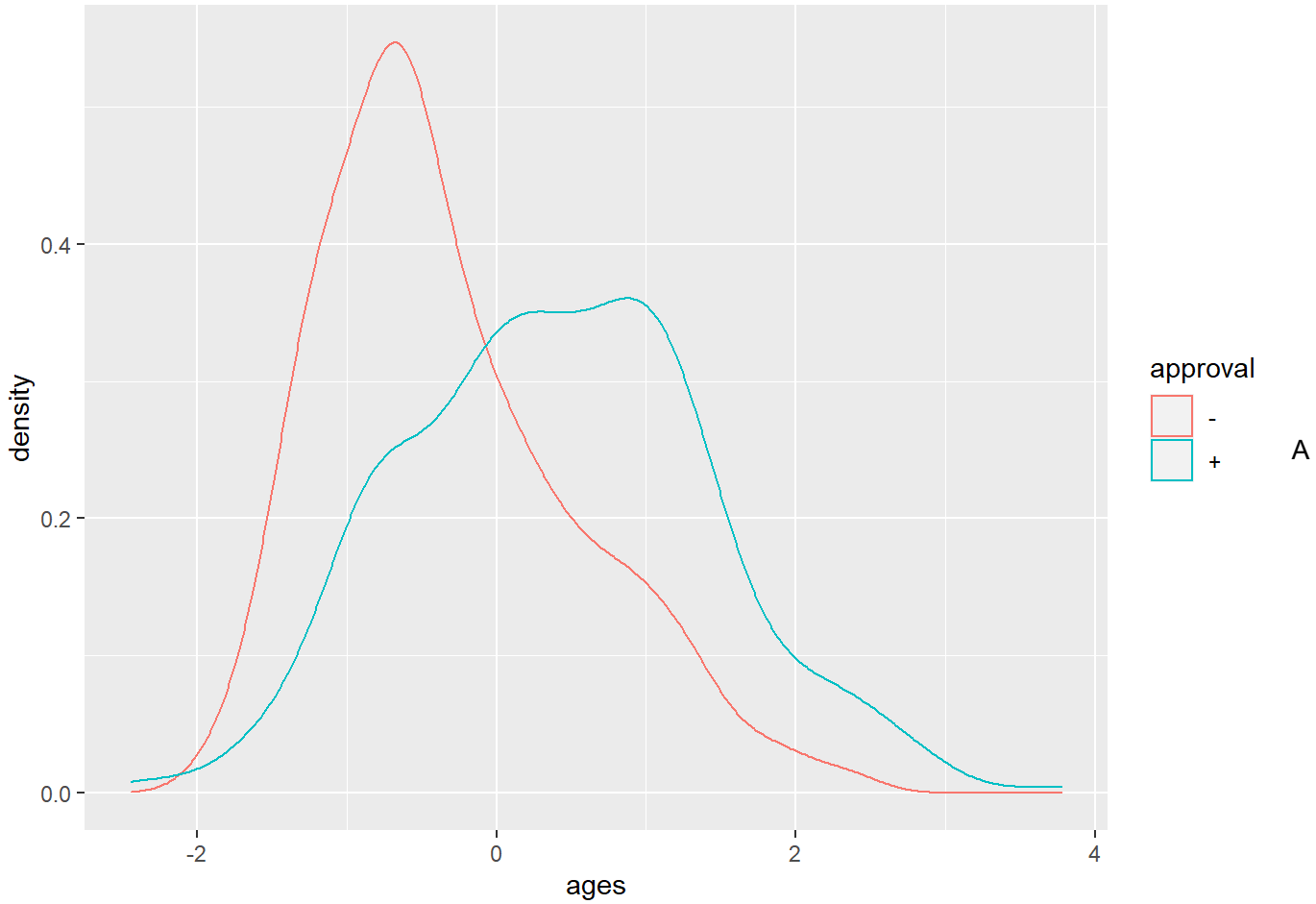


The credit.score variable was normalized using min-max normalization, and the result plotted on a density plot. Min-max normalization changed the credit.score's scale ranging from 583.7 to 806.0, to a more normalized scale ranging from 0 to 1 (as seen on the plot). The original scale that described how high a person's credit score was, was very arbitrary. The arbitrary scale makes it more difficult to discern how high someone's credit score is in comparison to other individuals in the data. However, knowing that the max score in this data is 1.0, and lowest becomes 0.0, we can quickly understand if an individual's particular credit score is more towards the higher end of the spectrum of this data, or towards the lower end.

```
#z-score normalization performed on dataframe
zscore <- preProcess(df, method=c("center", "scale"))
norm2 <- predict(zscore, df)
summary(norm2)
```

```
##       ...1             cont1             cont2             cont3        
##  Min.   :-1.7283   Min.   :-1.4901   Min.   :-0.9559   Min.   :-0.6644  
##  1st Qu.:-0.8641   1st Qu.:-0.7498   1st Qu.:-0.7550   1st Qu.:-0.6151  
##  Median : 0.0000   Median :-0.2599   Median :-0.4035   Median :-0.3656  
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.8641   3rd Qu.: 0.5571   3rd Qu.: 0.4919   3rd Qu.: 0.1200  
##  Max.   : 1.7283   Max.   : 4.0711   Max.   : 4.6686   Max.   : 7.8519  
##                    NA's   :12                                           
##    bool1            bool2             cont4            bool3        
##  Mode :logical   Mode :logical   Min.   :-0.4935   Mode :logical  
##  FALSE:329       FALSE:395       1st Qu.:-0.4935   FALSE:374      
##  TRUE :361       TRUE :295       Median :-0.4935   TRUE :316      
##                                  Mean   : 0.0000                  
##                                  3rd Qu.: 0.1234                  
##                                  Max.   :13.2841                  
##                                                                   
##      cont5             cont6           approval          credit.score      
##  Min.   :-1.0587   Min.   :-0.1953   Length:690         Min.   :-2.68864  
##  1st Qu.:-0.6272   1st Qu.:-0.1953   Class :character   1st Qu.:-0.70690  
##  Median :-0.1382   Median :-0.1943   Mode  :character   Median : 0.02146  
##  Mean   : 0.0000   Mean   : 0.0000                      Mean   : 0.00000  
##  3rd Qu.: 0.5292   3rd Qu.:-0.1194                      3rd Qu.: 0.71482  
##  Max.   :10.4483   Max.   :18.9982                      Max.   : 2.61446  
##  NA's   :13                                                               
##       ages        
##  Min.   :-2.4434  
##  1st Qu.:-0.7391  
##  Median :-0.1426  
##  Mean   : 0.0000  
##  3rd Qu.: 0.7095  
##  Max.   : 3.7772  
##
```

```
#plots normalized densityplot of ages variable and the approval variable is used for color

zscoreplot <- ggplot(norm2, aes(x=ages, color = approval)) +
  geom_density()

zscoreplot
```

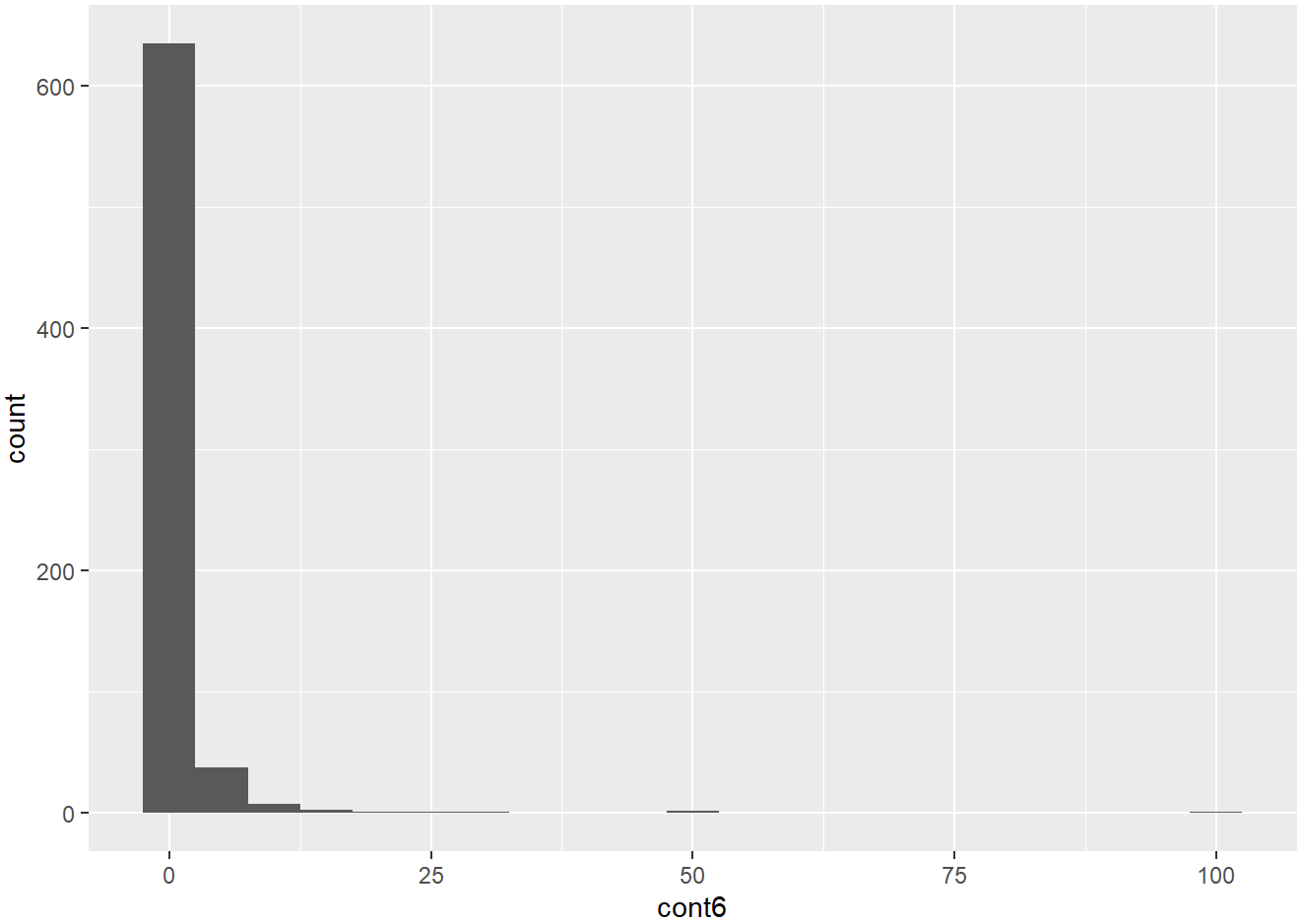


z-score normalization was used for the ages variable and the results checked using a density plot. Following normalization, the mean becomes set to 0 (originally set to 39.67 years). The goal is to therefore better see how far someone's age differed from the mean and the extent of which this effected that individual's approval for a loan. Following normalization, we can better see where the mean lies on the plot and discover a new piece of info - starting with right before the mean age, individuals are more likely to be approved for a loan.

```
#decimal scaling performed on dataframe
norm3 <- df %>% select(-c( "bool1", "bool2", "bool3", "approval"))
norm3 <- norm3/1000
summary(norm3)
```

```
##       ...1            cont1             cont2              cont3
##  Min.   :0.0010   Min.   :0.01375   Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.1732   1st Qu.:0.02260   1st Qu.:0.001000   1st Qu.:0.000165
##  Median :0.3455   Median :0.02846   Median :0.002750   Median :0.001000
##  Mean   :0.3455   Mean   :0.03157   Mean   :0.004759   Mean   :0.002223
##  3rd Qu.:0.5178   3rd Qu.:0.03823   3rd Qu.:0.007208   3rd Qu.:0.002625
##  Max.   :0.6900   Max.   :0.08025   Max.   :0.028000   Max.   :0.028500
##                   NA's   :12
##      cont4            cont5           cont6           credit.score
##  Min.   :0.0000   Min.   :0.000   Min.   :  0.0000   Min.   :0.5837
##  1st Qu.:0.0000   1st Qu.:0.075   1st Qu.:  0.0000   1st Qu.:0.6667
##  Median :0.0000   Median :0.160   Median :  0.0050   Median :0.6973
##  Mean   :0.0024   Mean   :0.184   Mean   :  1.0174   Mean   :0.6964
##  3rd Qu.:0.0030   3rd Qu.:0.276   3rd Qu.:  0.3955   3rd Qu.:0.7264
##  Max.   :0.0670   Max.   :2.000   Max.   :100.0000   Max.   :0.8060
##                   NA's   :13
##       ages
##  Min.   :0.01100
##  1st Qu.:0.03100
##  Median :0.03800
##  Mean   :0.03967
##  3rd Qu.:0.04800
##  Max.   :0.08400
##
```

```
#plots normalized histogram of cont6 variable

cont6decmalscaling <- ggplot(norm3, aes(cont6)) + geom_histogram(binwidth = 5)
cont6decmalscaling
```



Decimal scaling was applied to the cont6 variable and the results checked using a histogram. Following normalization, the massive scale of 0 to 100000 was reduced to something more manageable, 0 to 100. The goal with this was be able to better see the distribution of the data while using a much smaller binwidth. In this case only 5 after normalization.

1d

```
library(dplyr)
#new column created for v, v set equal to credit.score column
norm1 <- norm1 %>%
  mutate(v= credit.score)

#new column created with normalized credit.score data being binned into 4 different ranges of values
norm1 <- norm1 %>%
mutate(v_bins = cut(credit.score,
breaks=c(0.00, 0.25, 0.50, 0.75, 1.00),
labels=c("firstquadrant","secondquadrant","thirdquadrant","fourthquadrant"))) %>%
head()
```

``

I used the normalized version of the credit.score variable as the normalized version is much easier to divide into equal sized ranges of values. I chose to divide it into 4 quadrants - first quadrant (0.00 to 0.25), second quadrant (0.25 to 0.50), third quadrant (0.50 to 0.75), and fourth quadrant (0.75 to 1.00). Each quadrant groups

similarly ranked credit scores. The higher the quadrant, the better the credit scores. I chose four ranges as the original normalized range (0 to 1) is perfectly divisible by 4.

1e

```
#displays relevant columns of v and v_bins
df2 <- norm1 %>% select(-c( "cont1", "cont2", "cont3", "cont4", "cont5", "cont6", "bool1", "b
ool2", "bool3", "approval", "ages"))
df2
```

	...1 <dbl>	credit.score <dbl>	v <dbl>	v_bins <fct>
1	0.000000000	0.3640371	0.3640371	secondquadrant
2	0.001451379	0.4957273	0.4957273	secondquadrant
3	0.002902758	0.1716290	0.1716290	firstquadrant
4	0.004354136	0.3162274	0.3162274	secondquadrant
5	0.005805515	0.3894936	0.3894936	secondquadrant
6	0.007256894	0.3980390	0.3980390	secondquadrant

6 rows

```
#values in v replaced my the means of their respective ranges
firstquadrant <- df2 %>%
  filter(v_bins == 'firstquadrant') %>%
  mutate(v = mean(v, na.rm = T))
secondquadrant <- df2 %>%
  filter(v_bins == 'secondquadrant') %>%
  mutate(v = mean(v, na.rm = T))
thirdquadrant <- df2 %>%
  filter(v_bins == 'thirdquadrant') %>%
  mutate(v = mean(v, na.rm = T))
fourthquadrant <- df2 %>%
  filter(v_bins == 'fourthquadrant') %>%
  mutate(v = mean(v, na.rm = T))

bind_rows(list(firstquadrant, secondquadrant, thirdquadrant, fourthquadrant))
```

...1 <dbl>	credit.score <dbl>	v <dbl>	v_bins <fct>
0.002902758	0.1716290	0.1716290	firstquadrant
0.000000000	0.3640371	0.3927049	secondquadrant
0.001451379	0.4957273	0.3927049	secondquadrant
0.004354136	0.3162274	0.3927049	secondquadrant

...1	credit.score	v	v_bins
<dbl>	<dbl>	<dbl>	<fct>
0.005805515	0.3894936	0.3927049	secondquadrant
0.007256894	0.3980390	0.3927049	secondquadrant

6 rows

`The data was smoothed in a way where each value (v) in each particular quadrant, was replaced by that quadrant's mean. Now there are 4 different values present in v, each depicting one of the 4 quadrant's means. These mean values replace the individual credit scores.

2.a.

```
dfnew <- data.frame(BankData)
summary(dfnew)
```

```
##       ...1           cont1           cont2            cont3
##  Min.   :  1.0   Min.   :13.75   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:173.2   1st Qu.:22.60   1st Qu.: 1.000   1st Qu.: 0.165
##  Median :345.5   Median :28.46   Median : 2.750   Median : 1.000
##  Mean   :345.5   Mean   :31.57   Mean   : 4.759   Mean   : 2.223
##  3rd Qu.:517.8   3rd Qu.:38.23   3rd Qu.: 7.207   3rd Qu.: 2.625
##  Max.   :690.0   Max.   :80.25   Max.   :28.000   Max.   :28.500
##                  NA's   :12
##    bool1           bool2             cont4        bool3             cont5
##  Mode :logical   Mode :logical   Min.   : 0.0   Mode :logical   Min.   :   0
##  FALSE:329       FALSE:395       1st Qu.: 0.0   FALSE:374       1st Qu.:  75
##  TRUE :361       TRUE :295       Median : 0.0   TRUE :316       Median : 160
##                                  Mean   : 2.4                   Mean   : 184
##                                  3rd Qu.: 3.0                   3rd Qu.: 276
##                                  Max.   :67.0                   Max.   :2000
##                                                                 NA's   :13
##      cont6            approval          credit.score        ages
##  Min.   :     0.0   Length:690         Min.   :583.7   Min.   :11.00
##  1st Qu.:     0.0   Class :character   1st Qu.:666.7   1st Qu.:31.00
##  Median :     5.0   Mode  :character   Median :697.3   Median :38.00
##  Mean   :  1017.4                      Mean   :696.4   Mean   :39.67
##  3rd Qu.:   395.5                      3rd Qu.:726.4   3rd Qu.:48.00
##  Max.   :100000.0                      Max.   :806.0   Max.   :84.00
##
```

```
#missing values discovered
summary(dfnew$...1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0   173.2   345.5   345.5   517.8   690.0
```

```
summary(dfnew$cont1) #missing value
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   13.75   22.60   28.46   31.57   38.23   80.25      12
```

```
summary(dfnew$cont2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.750   4.759   7.207  28.000
```

```
summary(dfnew$cont3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.165   1.000   2.223   2.625  28.500
```

```
summary(dfnew$bool1)
```

```
##    Mode   FALSE    TRUE
## logical     329     361
```

```
summary(dfnew$bool2)
```

```
##    Mode   FALSE    TRUE
## logical     395     295
```

```
summary(dfnew$cont4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0     0.0     0.0     2.4     3.0    67.0
```

```
summary(dfnew$bool3)
```

```
##    Mode   FALSE    TRUE
## logical     374     316
```

```
summary(dfnew$cont5) #missing value
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##       0      75     160     184     276    2000      13
```

```
summary(dfnew$cont6)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      0.0      0.0      5.0   1017.4    395.5 100000.0
```

```
summary(dfnew$approval)
```

```
##    Length     Class      Mode
##       690 character character
```

```
summary(dfnew$credit.score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   583.7   666.7   697.3   696.4   726.4   806.0
```

```
summary(dfnew$ages)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   11.00   31.00   38.00   39.67   48.00   84.00
```

```
library(dbplyr)
```

```
##
## Attaching package: 'dbplyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
```

```
#missing values removed
dfnew <- dfnew %>% drop_na(cont1)
summary(dfnew$cont1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.75   22.60   28.46   31.57   38.23   80.25
```

```
#missing values removed
dfnew <- dfnew %>% drop_na(cont5)
summary(dfnew$cont5)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   75.25  160.00  182.12  271.00 2000.00
```

```
#categorical columns other than "approval" removed
#dfnew <- dfnew %>% select(-c( "bool1", "bool2", "bool3"))
str(dfnew)
```

```
## 'data.frame':    666 obs. of  13 variables:
##  $ ...1        : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ cont1       : num  30.8 58.7 24.5 27.8 20.2 ...
##  $ cont2       : num  0 4.46 0.5 1.54 5.62 ...
##  $ cont3       : num  1.25 3.04 1.5 3.75 1.71 ...
##  $ bool1       : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
##  $ bool2       : logi  TRUE TRUE FALSE TRUE FALSE FALSE ...
##  $ cont4       : num  1 6 0 5 0 0 0 0 0 0 ...
##  $ bool3       : logi  FALSE FALSE FALSE TRUE FALSE TRUE ...
##  $ cont5       : num  202 43 280 100 120 360 164 80 180 52 ...
##  $ cont6       : num  0 560 824 3 0 ...
##  $ approval    : chr  "+" "+" "+" "+" ...
##  $ credit.score: num  665 694 622 654 670 ...
##  $ ages        : num  42 54 29 58 65 61 50 41 30 35 ...
```

```
library(caret)
library(e1071)
```

```
#svm performed
train_control_cv = trainControl(method = "cv", number = 10)
svm_cv <- train(approval ~., data = dfnew, method = "svmLinear",
              trControl = train_control_cv)
svm_cv
```

```
## Support Vector Machines with Linear Kernel
##
## 666 samples
##  12 predictor
##   2 classes: '-', '+'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 599, 599, 600, 599, 599, 599, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8634102  0.7284524
##
## Tuning parameter 'C' was held constant at a value of 1
```

Using 10-fold cross validation, the accuracy for predicting the approval is 0.8154229, with a kappa of 0.6249221

2b

```
#Grid search performed
grid <- expand.grid(C = 10^seq(-5,2,0.5))
svm_grid <- train(approval ~., data = dfnew, method = "svmLinear",
              trControl = train_control_cv, tuneGrid = grid)
svm_grid
```

```
## Support Vector Machines with Linear Kernel
##
## 666 samples
##  12 predictor
##   2 classes: '-', '+'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 600, 599, 599, 599, 599, 600, ...
## Resampling results across tuning parameters:
##
##   C             Accuracy   Kappa
##   1.000000e-05  0.5510403  0.0000000
##   3.162278e-05  0.5510403  0.0000000
##   1.000000e-04  0.5510403  0.0000000
##   3.162278e-04  0.5721167  0.0510088
##   1.000000e-03  0.8364993  0.6673270
##   3.162278e-03  0.8679783  0.7376161
##   1.000000e-02  0.8634781  0.7286683
##   3.162278e-02  0.8634781  0.7286683
##   1.000000e-01  0.8634781  0.7286683
##   3.162278e-01  0.8634781  0.7286683
##   1.000000e+00  0.8634781  0.7286683
##   3.162278e+00  0.8634781  0.7286683
##   1.000000e+01  0.8634781  0.7286683
##   3.162278e+01  0.8634781  0.7286683
##   1.000000e+02  0.8634781  0.7286683
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.003162278.
```

The final value for C was 0.1. The accuracy was 0.8137637 with a kappa of 0.62068211

2c

Even if the grid of parameters in (b) includes the default value of C = 1, the accuracy result will sometimes be different for this value of C due to class imbalances. Some of the training test sets could end up with little or none of the less common classes. The lack of presence of these classes during training can result in inaccurate accuracy results. Stratified class validation is performed to help with this issue.

3a

```
 library(dplyr)
head(starwars)
```

name <chr>	height <int>	m… <dbl>	hair_color <chr>	skin_color <chr>	eye_color <chr>	birth_year <dbl>	sex <chr>
Luke Skywalker	172	77	blond	fair	blue	19.0	male
C-3PO	167	75	*NA*	gold	yellow	112.0	none
R2-D2	96	32	*NA*	white, blue	red	33.0	none
Darth Vader	202	136	none	white	yellow	41.9	male
Leia Organa	150	49	brown	light	brown	19.0	female
Owen Lars	178	120	brown, grey	light	blue	52.0	male

6 rows | 1-8 of 14 columns

```
#variables not needed remjoved
df3 <- starwars %>% select(-c("vehicles", "films", "name", "starships"))
```

```
head(df3)
```

height <int>	m… <dbl>	hair_color <chr>	skin_color <chr>	eye_color <chr>	birth_year <dbl>	sex <chr>	gender <chr>
172	77	blond	fair	blue	19.0	male	masculine
167	75	*NA*	gold	yellow	112.0	none	masculine
96	32	*NA*	white, blue	red	33.0	none	masculine
202	136	none	white	yellow	41.9	male	masculine
150	49	brown	light	brown	19.0	female	feminine
178	120	brown, grey	light	blue	52.0	male	masculine

6 rows | 1-8 of 10 columns

```
#missing values discovered
summary(df3$height) #missing value
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    66.0   167.0   180.0   174.4   191.0   264.0       6
```

```
summary(df3$mass) #missing value
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   15.00   55.60   79.00   97.31   84.50 1358.00      28
```

```
summary(df3$hair_color)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$skin_color)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$eye_color)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$birth_year) #missing value
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    8.00   35.00   52.00   87.57   72.00  896.00      44
```

```
summary(df3$sex)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$gender)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$homeworld)
```

```
##    Length     Class      Mode
##        87 character character
```

```
summary(df3$species)
```

```
##    Length     Class      Mode
##        87 character character
```

```
#missing values removed
df3 <- df3 %>% drop_na(height)
summary(df3$height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    66.0   167.0   180.0   174.4   191.0   264.0
```

```
#missing values removed
df3 <- df3 %>% drop_na(mass)
summary(df3$mass)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.00   55.60   79.00   97.31   84.50 1358.00
```

```
#missing values removed
df3 <- df3 %>% drop_na(birth_year)
summary(df3$birth_year)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.00   31.38   46.50   91.09   67.50  896.00
```

```
summary(df3)
```

```
##      height           mass         hair_color         skin_color
##  Min.   : 66.0   Min.   :  17.0   Length:36          Length:36
##  1st Qu.:170.0   1st Qu.:  72.0   Class :character   Class :character
##  Median :178.0   Median :  79.0   Mode  :character   Mode  :character
##  Mean   :173.1   Mean   : 112.2
##  3rd Qu.:188.0   3rd Qu.:  84.0
##  Max.   :228.0   Max.   :1358.0
##   eye_color           birth_year         sex                gender
##  Length:36          Min.   :  8.00   Length:36          Length:36
##  Class :character   1st Qu.: 31.38   Class :character   Class :character
##  Mode  :character   Median : 46.50   Mode  :character   Mode  :character
##                     Mean   : 91.09
##                     3rd Qu.: 67.50
##                     Max.   :896.00
##   homeworld           species
##  Length:36          Length:36
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
#dummy variables created to change categorical columns to numerical
dummy <- dummyVars(gender ~ ., data = df3)
dummiesdf4 <- as.data.frame(predict(dummy, newdata = df3))
```

```
head(dummiesdf4)
```

	height <dbl>	m… <dbl>	hair_colorauburn, white <dbl>	hair_colorblack <dbl>	hair_colorblond <dbl>
1	172	77	0	0	1
2	167	75	*NA*	*NA*	*NA*
3	96	32	*NA*	*NA*	*NA*
4	202	136	0	0	0
5	150	49	0	0	0
6	178	120	0	0	0

6 rows | 1-6 of 79 columns

3b

```
#gender column added back to dummy data
dummiesdf4$gender <- df3$gender
```

```
head(dummiesdf4)
```

	height <dbl>	m… <dbl>	hair_colorauburn, white <dbl>	hair_colorblack <dbl>	hair_colorblond <dbl>
1	172	77	0	0	1
2	167	75	*NA*	*NA*	*NA*
3	96	32	*NA*	*NA*	*NA*
4	202	136	0	0	0
5	150	49	0	0	0
6	178	120	0	0	0

6 rows | 1-6 of 80 columns

```
#missing values removed
dummiesdf4 <- na.omit(dummiesdf4)
```

```
head(dummiesdf4)
```

	height <dbl>	m… <dbl>	hair_colorauburn, white <dbl>	hair_colorblack <dbl>	hair_colorblond <dbl>
1	172	77	0	0	1
4	202	136	0	0	0
5	150	49	0	0	0
6	178	120	0	0	0
7	165	75	0	0	0
8	183	84	0	1	0

6 rows | 1-6 of 80 columns

```
#svm performed
svm_starwar <- train(gender ~., data = dummiesdf4, method = "svmLinear")
```

```
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
```

```
svm_starwar
```

```
## Support Vector Machines with Linear Kernel
##
## 29 samples
## 78 predictors
##  2 classes: 'feminine', 'masculine'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 29, 29, 29, 29, 29, 29, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8454848  0.5618793
##
## Tuning parameter 'C' was held constant at a value of 1
```

Accuracy = 0.8859573
Kappa = 0.6297646

3c

```
#gender variable removed from dummy data and saved to new dataframe
dummiesdf5 <- dummiesdf4 %>% select(-c("gender"))
dummiesdf5 <- dummiesdf4$gender
dummiesdf5 <- dummiesdf4 %>% select(-c("gender"))
```

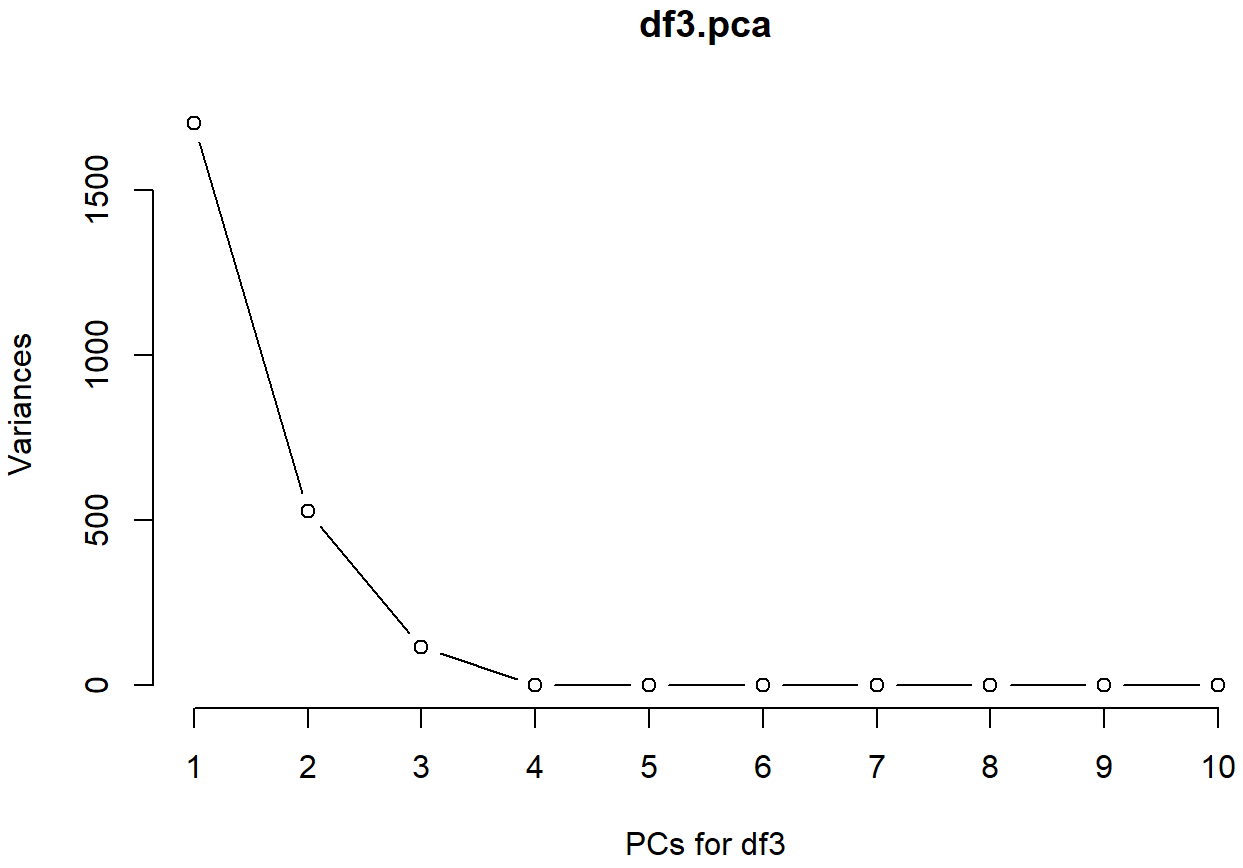
```
#remove near zero variance predictors
nzv <- nearZeroVar(dummiesdf5)
length(nzv)
```

```
## [1] 51
```

```
#gets pca object
df3.pca <- prcomp(dummiesdf5)
summary(df3.pca)
```

```
## Importance of components:
##                             PC1     PC2      PC3     PC4     PC5     PC6     PC7
## Standard deviation     41.2498 22.9472 10.69955 0.78075 0.73976 0.58724 0.57943
## Proportion of Variance  0.7252  0.2244  0.04879 0.00026 0.00023 0.00015 0.00014
## Cumulative Proportion   0.7252  0.9496  0.99838 0.99864 0.99887 0.99902 0.99916
##                             PC8    PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.51580 0.4766 0.41954 0.38652 0.37506 0.37045 0.35266
## Proportion of Variance 0.00011 0.0001 0.00008 0.00006 0.00006 0.00006 0.00005
## Cumulative Proportion  0.99928 0.9994 0.99945 0.99951 0.99957 0.99963 0.99968
##                            PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.34046 0.32506 0.31306 0.29384 0.27264 0.25462 0.24142
## Proportion of Variance 0.00005 0.00005 0.00004 0.00004 0.00003 0.00003 0.00002
## Cumulative Proportion  0.99973 0.99978 0.99982 0.99986 0.99989 0.99991 0.99994
##                            PC22    PC23    PC24    PC25    PC26   PC27    PC28
## Standard deviation     0.20136 0.18976 0.16411 0.14962 0.11579 0.0376 0.02181
## Proportion of Variance 0.00002 0.00002 0.00001 0.00001 0.00001 0.0000 0.00000
## Cumulative Proportion  0.99996 0.99997 0.99998 0.99999 1.00000 1.0000 1.00000
##                             PC29
## Standard deviation     3.822e-15
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```
#scree plot created to see how many pc's needed
screeplot(df3.pca, type = "l") + title(xlab = "PCs for df3")
```




```
## integer(0)
```

Based on the scree plot and summary of PC components, we capture most of the variance by using 3 principal components at +95% variance (99.838%).

```
#Unfortunately ran into the following traceback error: "Error in prcomp.default(x[, method$pc
a, drop = FALSE], scale = TRUE, retx = FALSE) : cannot rescale a constant/zero column to unit
variance" when trying to use 3 PCs to model data. The following line of code was found on htt
ps://stackoverflow.com/questions/15068981/removal-of-constant-columns-in-r , the comments on
thi website advised to remove some of the columns which were causing these problems. This is
what I did in order for the error to not appear. However, I'm not sure why this error appeare
d in the first place or if removing these columns causes massive changes to do the data.


names(dummiesdf5[, sapply(dummiesdf5, function(v) var(v, na.rm=TRUE)==0)])
```

```
##  [1] "skin_colorgold"            "skin_colorgreen-tan, brown"
##  [3] "skin_colormetal"           "skin_colorwhite, blue"
##  [5] "sexhermaphroditic"         "sexnone"
##  [7] "homeworldNal Hutta"        "homeworldRodia"
##  [9] "speciesDroid"              "speciesHutt"
## [11] "speciesRodian"             "speciesYoda's species"
```

```
dummiesdf7 <- dummiesdf5
dummiesdf7$gender <- dummiesdf4$gender #gender added back to dummy data and saved to new data frame
dummiesdf7 <- dummiesdf7 %>% select(-c("skin_colorgold","skin_colorgreen-tan, brown", "skin_colormetal", "skin_colorwhite, blue", "sexhermaphroditic", "sexnone", "homeworldNal Hutta", "homeworldRodia", "speciesDroid", "speciesHutt", "speciesRodian", "speciesYoda's species"))
```

`

```
#new dataset with the PCs created instead of the predictors.
target <- df3 %>% dplyr::select(gender)

preProc <- preProcess(dummiesdf7, method="pca", pcaComp=3)
df3.pc <- predict(preProc, dummiesdf7)
df3.pc$gender <- dummiesdf7$gender
head(df3.pc)
```

	gender <chr>	PC1 <dbl>	PC2 <dbl>	PC3 <dbl>
1	masculine	-0.1842293	1.7888683	-0.12884747
4	masculine	2.6287944	0.7247147	0.04277518
5	feminine	-3.5412999	0.3049210	-0.53174004
6	masculine	0.4172034	1.7928384	0.73884639
7	feminine	-2.1186809	0.6821707	1.12733245
8	masculine	-0.7075658	1.4212027	-0.47516161

6 rows

3d

```
#svm done on the pca data
svm_starwarpca <- train(gender ~., data = df3.pc, method = "svmLinear")
svm_starwarpca
```

```
## Support Vector Machines with Linear Kernel
##
## 29 samples
##  3 predictor
##  2 classes: 'feminine', 'masculine'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 29, 29, 29, 29, 29, 29, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9097716  0.7077554
##
## Tuning parameter 'C' was held constant at a value of 1
```

3e The Accuracy increased from 0.8859573 with a kappa of 0.6297646 to 0.9361978 with a kappa of 0.791099. The smaller representation in the pca version suggests that it is more efficient. PCA alters the complexity of the model by making it simpler and helps prevent overfitting.