ADA Group Assignment_Clustering

Group26

2024-02-21

Checking quality of data

```
# Read a file: loan_data_ADA_assignment.xlsx
data <- read_excel("loan_data_ADA_assignment.xlsx")
```

```
# Check Structure and Summary from raw data frame
str(data)
```

```
## tibble [50,000 x 53] (S3: tbl_df/tbl/data.frame)
##  $ id                          : num [1:50000] 3296446 3286412 3286406 3296434 3286395 ...
##  $ member_id                   : num [1:50000] 4068857 4058853 4058848 4068843 4058836 ...
##  $ loan_amnt                   : num [1:50000] 11200 10000 8000 16000 4000 15000 8000 19800 4000 1440
##  $ funded_amnt                 : num [1:50000] 11200 10000 8000 16000 4000 15000 8000 19800 4000 1440
##  $ funded_amnt_inv             : num [1:50000] 11200 10000 8000 15950 4000 ...
##  $ term                        : num [1:50000] 36 36 36 36 36 36 36 60 36 36 ...
##  $ int_rate                    : num [1:50000] 6.62 11.14 16.29 7.9 7.9 ...
##  $ installment                 : num [1:50000] 344 328 282 501 125 ...
##  $ grade                       : chr [1:50000] "A" "B" "C" "A" ...
##  $ sub_grade                   : chr [1:50000] "A2" "B2" "C4" "A4" ...
##  $ emp_title                   : chr [1:50000] "Nokia Siemens Network" "creative financial group" "Te
##  $ emp_length                  : num [1:50000] 10 2 7 10 10 10 10 10 NA 3 ...
##  $ home_ownership              : chr [1:50000] "OWN" "MORTGAGE" "RENT" "MORTGAGE" ...
##  $ annual_inc                  : num [1:50000] 108000 65000 35000 110000 155000 ...
##  $ verification_status         : chr [1:50000] "Not Verified" "Not Verified" "Not Verified" "Verified
##  $ issue_d                     : POSIXct[1:50000], format: "2013-02-01" "2013-02-01" ...
##  $ loan_status                 : chr [1:50000] "Current" "Charged Off" "Current" "Fully Paid" ...
##  $ pymnt_plan                  : chr [1:50000] "n" "n" "n" "n" ...
##  $ desc                        : chr [1:50000] "Borrower added on 01/27/13 > Credit Card Refinancing<
##  $ purpose                     : chr [1:50000] "credit_card" "credit_card" "debt_consolidation" "debt_
##  $ title                       : chr [1:50000] "Credit Card" "my lending club Loan" "All in One" "Deb
##  $ zip_code                    : chr [1:50000] "750xx" "085xx" "440xx" "060xx" ...
##  $ addr_state                  : chr [1:50000] "TX" "NJ" "OH" "CT" ...
##  $ dti                         : num [1:50000] 12.52 9.58 27.84 28.87 17.87 ...
##  $ delinq_2yrs                 : num [1:50000] 0 0 0 0 0 0 1 0 0 0 ...
##  $ earliest_cr_line            : POSIXct[1:50000], format: "2002-10-01" "2000-03-01" ...
##  $ inq_last_6mths              : num [1:50000] 0 0 2 0 0 2 0 1 0 1 ...
##  $ mths_since_last_delinq      : num [1:50000] NA NA NA NA NA 67 19 NA NA NA ...
##  $ mths_since_last_record      : num [1:50000] NA NA NA NA NA NA NA NA NA NA ...
##  $ open_acc                    : num [1:50000] 9 9 12 21 7 9 7 18 9 10 ...
##  $ pub_rec                     : num [1:50000] 0 0 0 0 0 0 0 0 0 0 ...
##  $ revol_bal                   : num [1:50000] 37822 16623 17938 23691 43945 ...
##  $ revol_util                  : num [1:50000] 0.662 0.742 0.72 0.752 0.955 0.681 0.476 0.767 0.873 0
```

```
##  $ total_acc                 : num [1:50000] 21 11 17 56 21 19 30 26 14 29 ...
##  $ total_pymnt               : num [1:50000] 11676 4620 9602 16768 4252 ...
##  $ total_pymnt_inv           : num [1:50000] 11676 4620 9602 16716 4252 ...
##  $ total_rec_prncp           : num [1:50000] 10505 2711 7447 16000 3749 ...
##  $ total_rec_int             : num [1:50000] 1172 898 2155 768 503 ...
##  $ total_rec_late_fee        : num [1:50000] 0 0 0 0 0 0 0 0 0 0 ...
##  $ recoveries                : num [1:50000] 0 1012 0 0 0 ...
##  $ collection_recovery_fee   : num [1:50000] 0 10.1 0 0 0 ...
##  $ last_pymnt_d              : POSIXct[1:50000], format: "2015-12-01" "2014-01-01" ...
##  $ last_pymnt_amnt           : num [1:50000] 344 328 282 13269 125 ...
##  $ next_pymnt_d              : POSIXct[1:50000], format: "2016-01-01" NA ...
##  $ last_credit_pull_d        : POSIXct[1:50000], format: "2015-12-01" "2014-01-01" ...
##  $ collections_12_mths_ex_med : num [1:50000] 0 0 0 0 0 0 0 0 0 0 ...
##  $ mths_since_last_major_derog: num [1:50000] NA NA NA NA NA 67 19 NA NA NA ...
##  $ policy_code               : num [1:50000] 1 1 1 1 1 1 1 1 1 1 ...
##  $ acc_now_delinq            : num [1:50000] 0 0 0 0 0 0 0 0 0 0 ...
##  $ tot_coll_amt              : num [1:50000] 0 0 0 0 0 52 0 0 90 0 ...
##  $ tot_cur_bal               : num [1:50000] 187717 16623 17938 372771 331205 ...
##  $ total_credit_rv           : num [1:50000] 66400 22400 24900 31500 46000 27100 31000 20800 13800 4
##  $ loan_is_bad               : logi [1:50000] FALSE TRUE FALSE FALSE FALSE FALSE ...
```

```
summary(data)
```

```
##        id            member_id         loan_amnt      funded_amnt
##  Min.   :  58524   Min.   : 149512   Min.   : 1000   Min.   : 1000
##  1st Qu.:1443048   1st Qu.:1695278   1st Qu.: 8000   1st Qu.: 8000
##  Median :1587758   Median :1857296   Median :12000   Median :12000
##  Mean   :1918444   Mean   :2283786   Mean   :13901   Mean   :13896
##  3rd Qu.:2311939   3rd Qu.:2744578   3rd Qu.:19200   3rd Qu.:19200
##  Max.   :3304574   Max.   :4076727   Max.   :35000   Max.   :35000
##
##  funded_amnt_inv      term          int_rate      installment
##  Min.   :  950   Min.   :36.00   Min.   : 6.00   Min.   :  25.81
##  1st Qu.: 7950   1st Qu.:36.00   1st Qu.:11.14   1st Qu.: 255.66
##  Median :12000   Median :36.00   Median :14.09   Median : 399.26
##  Mean   :13878   Mean   :40.49   Mean   :14.00   Mean   : 436.95
##  3rd Qu.:19175   3rd Qu.:36.00   3rd Qu.:17.27   3rd Qu.: 567.04
##  Max.   :35000   Max.   :60.00   Max.   :24.89   Max.   :1388.45
##
##     grade            sub_grade          emp_title           emp_length
##  Length:50000       Length:50000       Length:50000       Min.   : 1.000
##  Class :character   Class :character   Class :character   1st Qu.: 3.000
##  Mode  :character   Mode  :character   Mode  :character   Median : 6.000
##                                                           Mean   : 5.993
##                                                           3rd Qu.:10.000
##                                                           Max.   :10.000
##                                                           NA's   :1802
##  home_ownership       annual_inc      verification_status
##  Length:50000       Min.   :   5000   Length:50000
##  Class :character   1st Qu.:  45000   Class :character
##  Mode  :character   Median :  60000   Mode  :character
##                     Mean   :  71317
##                     3rd Qu.:  85000
##                     Max.   :7141778
##
```

```
##     issue_d                       loan_status         pymnt_plan
##  Min.   :2012-05-01 00:00:00.00   Length:50000       Length:50000
##  1st Qu.:2012-08-01 00:00:00.00   Class :character   Class :character
##  Median :2012-10-01 00:00:00.00   Mode  :character   Mode  :character
##  Mean   :2012-09-29 03:53:13.33
##  3rd Qu.:2012-12-01 00:00:00.00
##  Max.   :2013-02-01 00:00:00.00
##
##      desc             purpose             title             zip_code
##  Length:50000       Length:50000       Length:50000       Length:50000
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   addr_state             dti         delinq_2yrs
##  Length:50000       Min.   : 0.00   Min.   : 0.0000
##  Class :character   1st Qu.:11.51   1st Qu.: 0.0000
##  Mode  :character   Median :17.16   Median : 0.0000
##                     Mean   :17.37   Mean   : 0.2244
##                     3rd Qu.:23.05   3rd Qu.: 0.0000
##                     Max.   :34.99   Max.   :18.0000
##
##  earliest_cr_line                 inq_last_6mths   mths_since_last_delinq
##  Min.   :1951-12-01 00:00:00.000   Min.   :0.0000   Min.   :  0.00
##  1st Qu.:1994-05-01 00:00:00.000   1st Qu.:0.0000   1st Qu.: 18.00
##  Median :1999-01-01 00:00:00.000   Median :1.0000   Median : 33.00
##  Mean   :1997-09-29 09:34:28.416   Mean   :0.8389   Mean   : 36.08
##  3rd Qu.:2002-05-01 00:00:00.000   3rd Qu.:1.0000   3rd Qu.: 52.00
##  Max.   :2009-12-01 00:00:00.000   Max.   :8.0000   Max.   :152.00
##                                                     NA's   :28126
##  mths_since_last_record    open_acc        pub_rec          revol_bal
##  Min.   :  2.0          Min.   : 0.00   Min.   :0.00000   Min.   :      0
##  1st Qu.: 76.0          1st Qu.: 8.00   1st Qu.:0.00000   1st Qu.:   7102
##  Median : 93.0          Median :10.00   Median :0.00000   Median :  12368
##  Mean   : 87.7          Mean   :11.01   Mean   :0.05648   Mean   :  16011
##  3rd Qu.:106.0          3rd Qu.:14.00   3rd Qu.:0.00000   3rd Qu.:  20515
##  Max.   :119.0          Max.   :53.00   Max.   :8.00000   Max.   :1743266
##  NA's   :47468
##    revol_util       total_acc      total_pymnt    total_pymnt_inv
##  Min.   :0.0000   Min.   : 2.00   Min.   :    0   Min.   :    0
##  1st Qu.:0.4310   1st Qu.:16.00   1st Qu.: 7614   1st Qu.: 7601
##  Median :0.6150   Median :23.00   Median :12858   Median :12842
##  Mean   :0.5885   Mean   :24.31   Mean   :14828   Mean   :14808
##  3rd Qu.:0.7750   3rd Qu.:31.00   3rd Qu.:20051   3rd Qu.:20024
##  Max.   :1.1390   Max.   :99.00   Max.   :57778   Max.   :57778
##  NA's   :31
##  total_rec_prncp total_rec_int   total_rec_late_fee   recoveries
##  Min.   :    0   Min.   :    0   Min.   :  0.0000   Min.   :    0.0
##  1st Qu.: 6000   1st Qu.: 1058   1st Qu.:  0.0000   1st Qu.:    0.0
##  Median :10000   Median : 2047   Median :  0.0000   Median :    0.0
##  Mean   :11611   Mean   : 3071   Mean   :  0.8419   Mean   :  144.2
##  3rd Qu.:15479   3rd Qu.: 3737   3rd Qu.:  0.0000   3rd Qu.:    0.0
```

```
##  Max.   :35000   Max.   :22778   Max.   :286.7476   Max.   :33520.3
##
##  collection_recovery_fee  last_pymnt_d                    last_pymnt_amnt
##  Min.   :   0.00         Min.   :2012-06-01 00:00:00.00   Min.   :    0.0
##  1st Qu.:   0.00         1st Qu.:2014-03-01 00:00:00.00   1st Qu.:  353.1
##  Median :   0.00         Median :2015-03-01 00:00:00.00   Median :  723.6
##  Mean   :  10.66         Mean   :2014-11-26 07:40:19.91   Mean   : 3569.0
##  3rd Qu.:   0.00         3rd Qu.:2015-10-01 00:00:00.00   3rd Qu.: 4675.9
##  Max.   :3896.24         Max.   :2015-12-01 00:00:00.00   Max.   :35683.2
##                          NA's   :43
##   next_pymnt_d                    last_credit_pull_d
##  Min.   :2016-01-01 00:00:00.00   Min.   :2012-05-01 00:00:00.00
##  1st Qu.:2016-01-01 00:00:00.00   1st Qu.:2015-03-01 00:00:00.00
##  Median :2016-01-01 00:00:00.00   Median :2015-11-01 00:00:00.00
##  Mean   :2016-01-06 08:08:08.33   Mean   :2015-06-01 13:41:50.21
##  3rd Qu.:2016-01-01 00:00:00.00   3rd Qu.:2015-12-01 00:00:00.00
##  Max.   :2016-02-01 00:00:00.00   Max.   :2015-12-01 00:00:00.00
##  NA's   :42864
##  collections_12_mths_ex_med mths_since_last_major_derog  policy_code
##  Min.   :0.00000            Min.   :  0.00              Min.   :1
##  1st Qu.:0.00000            1st Qu.: 25.00              1st Qu.:1
##  Median :0.00000            Median : 40.00              Median :1
##  Mean   :0.00114            Mean   : 42.31              Mean   :1
##  3rd Qu.:0.00000            3rd Qu.: 59.00              3rd Qu.:1
##  Max.   :2.00000            Max.   :152.00              Max.   :1
##                             NA's   :42880
##  acc_now_delinq     tot_coll_amt    tot_cur_bal      total_credit_rv
##  Min.   :0.00000   Min.   :    0   Min.   :      0   Min.   :      0
##  1st Qu.:0.00000   1st Qu.:    0   1st Qu.:  26298   1st Qu.:  14000
##  Median :0.00000   Median :    0   Median :  72117   Median :  22800
##  Mean   :0.00082   Mean   :   52   Mean   : 133594   Mean   :  29300
##  3rd Qu.:0.00000   3rd Qu.:    0   3rd Qu.: 202362   3rd Qu.:  36600
##  Max.   :4.00000   Max.   :55009   Max.   :8000078   Max.   :2013133
##                    NA's   :14618   NA's   :14618     NA's   :14618
##  loan_is_bad
##  Mode :logical
##  FALSE:42186
##  TRUE :7814
##
##
##
##
```

#Data Preperation Filtering data

```
#Filter data

data_filtered <- select(data, loan_amnt, sub_grade, emp_length, annual_inc, loan_status, dti, delinq_2y
```

Factoring categorical variable and change them into numeric.

```
# Change categorical into factor variable
data_filtered$sub_grade <-  factor(data_filtered$sub_grade, levels=c("A1", "A2", "A3", "A4", "A5","B1",
data_filtered$sub_grade <-  as.numeric(data_filtered$sub_grade)

data_filtered$loan_status <-  factor(data_filtered$loan_status, levels=c("Fully Paid", "Current", "In G
```

```r
data_filtered$loan_status <-  as.numeric(data_filtered$loan_status)
```

As we will focus on 500 data from raw data, we will remove NA values as the remaining data without NA values is still more than 500 records.

```r
# Remove missing values
data_filtered <- na.omit(data_filtered)
```

We will sampling 500 observations from data frame for future analysis.

```r
# Sampling 500 data by random sampling

set.seed(123)
RndSampledData <- sample_n(data_filtered, 500)

# Check data after random sampling
summary(RndSampledData)
```

```
##    loan_amnt       sub_grade       emp_length       annual_inc
##  Min.   : 1000   Min.   : 1.00   Min.   : 1.000   Min.   :  18000
##  1st Qu.: 8188   1st Qu.: 7.00   1st Qu.: 3.000   1st Qu.:  45000
##  Median :12000   Median :10.00   Median : 6.000   Median :  60000
##  Mean   :13960   Mean   :11.22   Mean   : 6.054   Mean   :  72742
##  3rd Qu.:20000   3rd Qu.:15.00   3rd Qu.:10.000   3rd Qu.:  85000
##  Max.   :35000   Max.   :32.00   Max.   :10.000   Max.   :1250000
##   loan_status         dti          delinq_2yrs       open_acc
##  Min.   :1.000   Min.   : 0.40   Min.   :0.000   Min.   : 2.00
##  1st Qu.:1.000   1st Qu.:11.61   1st Qu.:0.000   1st Qu.: 8.00
##  Median :1.000   Median :17.16   Median :0.000   Median :10.00
##  Mean   :1.862   Mean   :17.54   Mean   :0.236   Mean   :10.87
##  3rd Qu.:2.000   3rd Qu.:23.73   3rd Qu.:0.000   3rd Qu.:13.00
##  Max.   :7.000   Max.   :34.92   Max.   :5.000   Max.   :34.00
##    revol_util      total_pymnt     tot_coll_amt      tot_cur_bal
##  Min.   :0.0000   Min.   : 1025   Min.   :   0.00   Min.   :   2486
##  1st Qu.:0.4860   1st Qu.: 8763   1st Qu.:   0.00   1st Qu.:  25481
##  Median :0.6685   Median :12833   Median :   0.00   Median :  76380
##  Mean   :0.6278   Mean   :15384   Mean   :  29.46   Mean   : 134577
##  3rd Qu.:0.8013   3rd Qu.:21021   3rd Qu.:   0.00   3rd Qu.: 198360
##  Max.   :0.9890   Max.   :55145   Max.   :5491.00   Max.   :1103872
```

```r
str(RndSampledData)
```

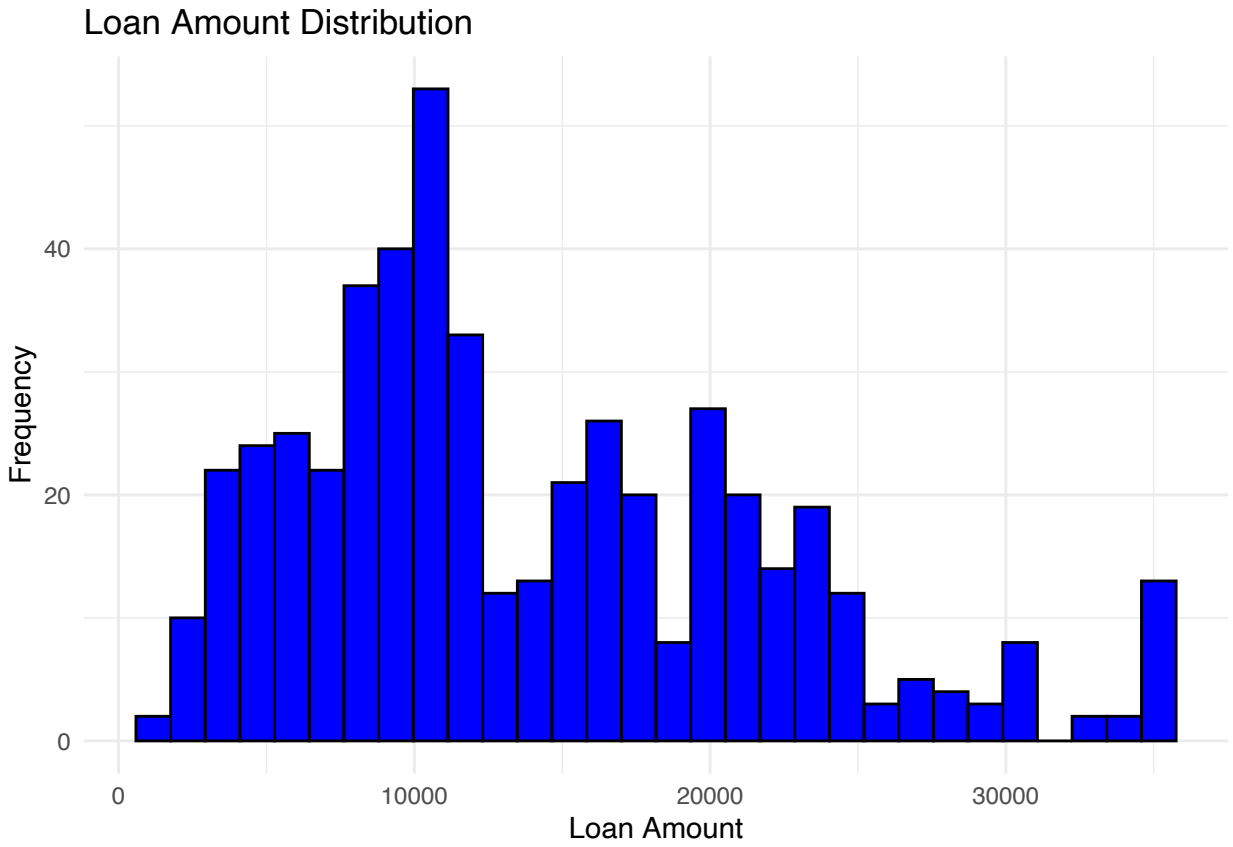
```
## tibble [500 x 12] (S3: tbl_df/tbl/data.frame)
##  $ loan_amnt   : num [1:500] 15000 21000 9600 15000 29175 ...
##  $ sub_grade   : num [1:500] 3 17 12 7 11 8 16 13 3 8 ...
##  $ emp_length  : num [1:500] 10 10 1 10 10 1 4 5 5 10 ...
##  $ annual_inc  : num [1:500] 99671 135000 104000 75000 65000 ...
##  $ loan_status : num [1:500] 1 2 1 3 1 1 7 1 1 1 ...
##  $ dti         : num [1:500] 20.81 25.24 7.64 17.06 23.72 ...
##  $ delinq_2yrs : num [1:500] 0 2 1 0 1 0 0 0 0 0 ...
##  $ open_acc    : num [1:500] 17 12 9 10 8 6 4 10 9 8 ...
##  $ revol_util  : num [1:500] 0.575 0.758 0.07 0.845 0.781 0.744 0.805 0.526 0.524 0.294 ...
##  $ total_pymnt : num [1:500] 16657 21015 10081 16253 32559 ...
##  $ tot_coll_amt: num [1:500] 0 0 0 0 0 0 0 0 0 0 ...
##  $ tot_cur_bal : num [1:500] 225980 336439 37833 186049 213413 ...
##  - attr(*, "na.action")= 'omit' Named int [1:15948] 9 21 61 202 262 326 337 341 379 395 ...
```

```
##   ..- attr(*, "names")= chr [1:15948] "9" "21" "61" "202" ...
```

As our data has 12 variables and 500 observations, we consider this data frame as big sample size data since the number of observations/variables is 41.66 (more than 10).

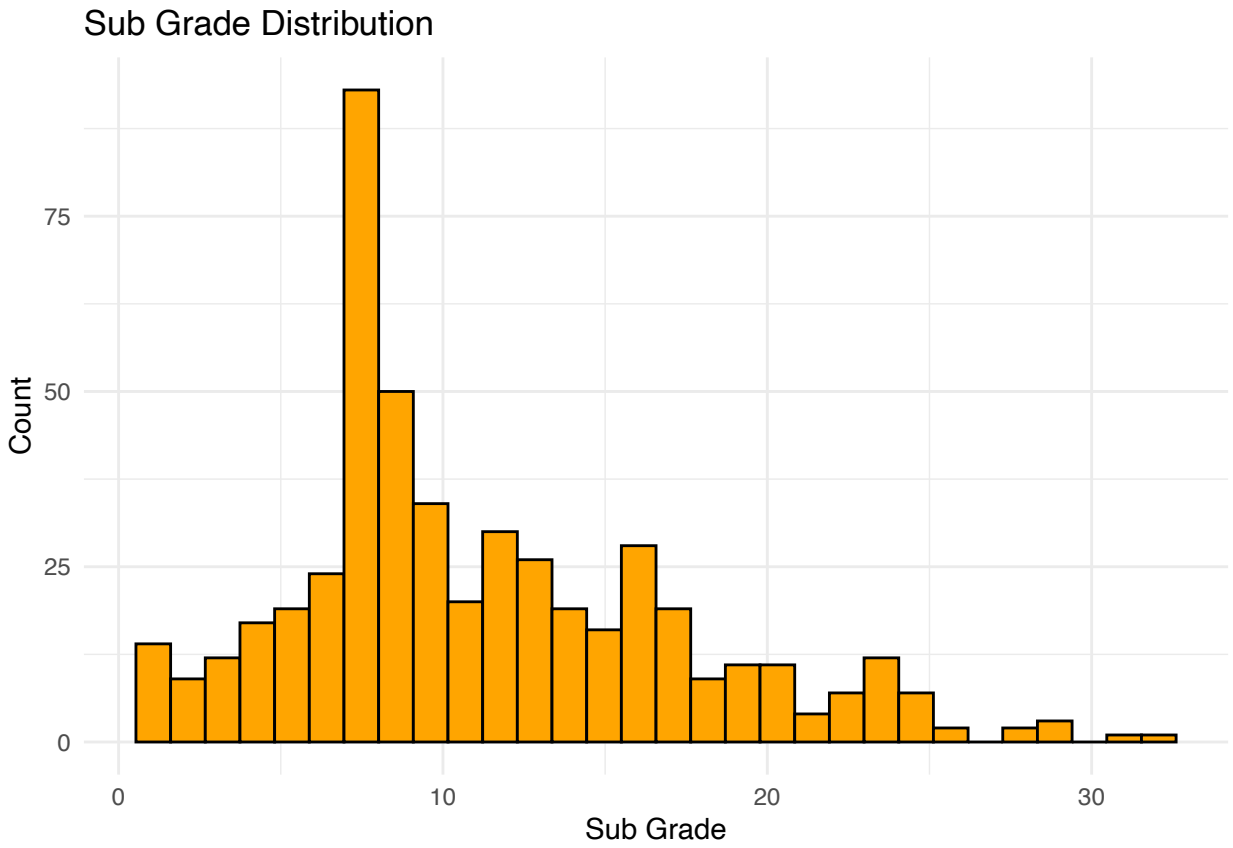
We will plot our new dataset to see the distribution of each variable.

```
# Histograms for Numeric Variables
RndSampledData %>% ggplot(aes(x = loan_amnt)) +
  geom_histogram(bins = 30, fill = "blue", color = "black") +
  theme_minimal() +
  labs(title = "Loan Amount Distribution", x = "Loan Amount", y = "Frequency")
```

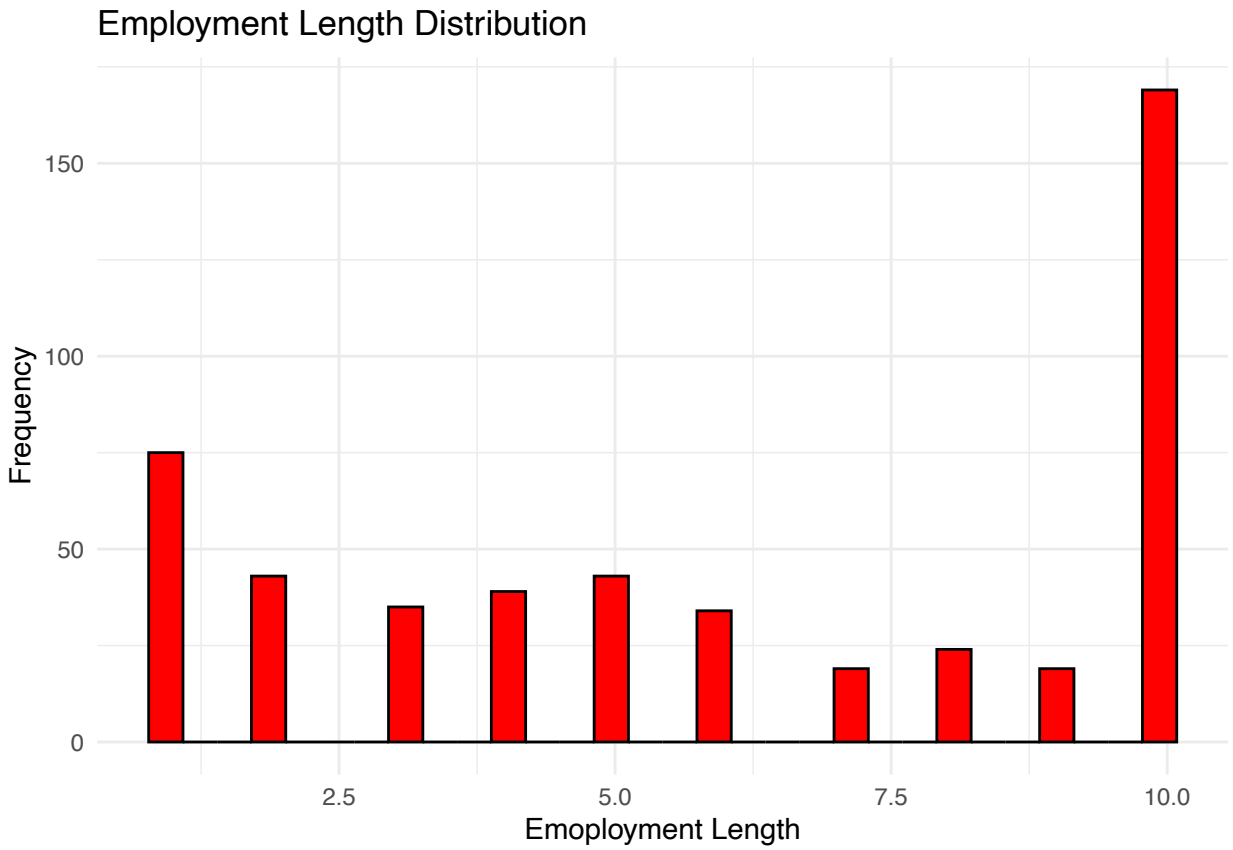



```
RndSampledData %>% ggplot(aes(x = sub_grade)) +
  geom_histogram(fill = "orange", color = "black") +
  theme_minimal() +
  labs(title = "Sub Grade Distribution", x = "Sub Grade", y = "Count")
```

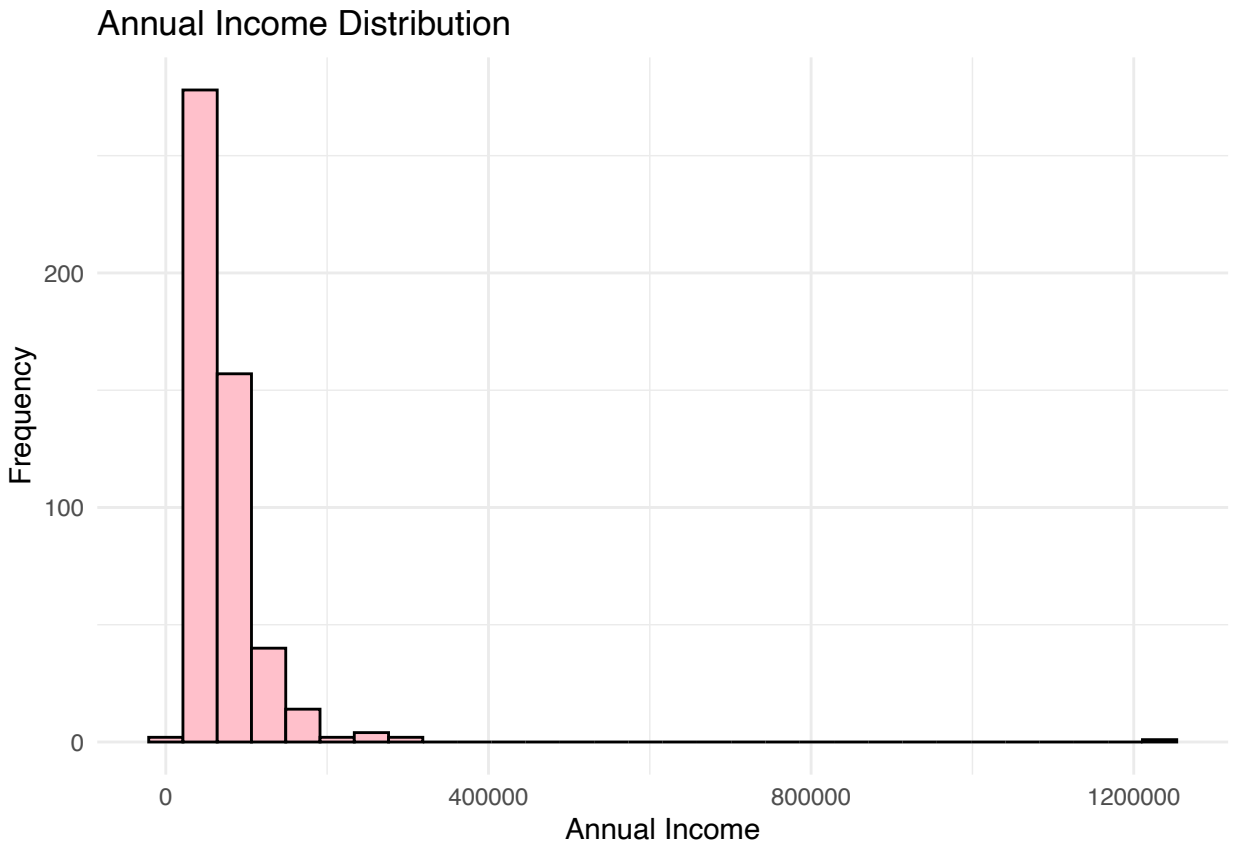
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
RndSampledData %>% ggplot(aes(x = emp_length)) +
  geom_histogram(bins = 30, fill = "red", color = "black") +
  theme_minimal() +
  labs(title = "Employment Length Distribution", x = "Emoployment Length", y = "Frequency")
```

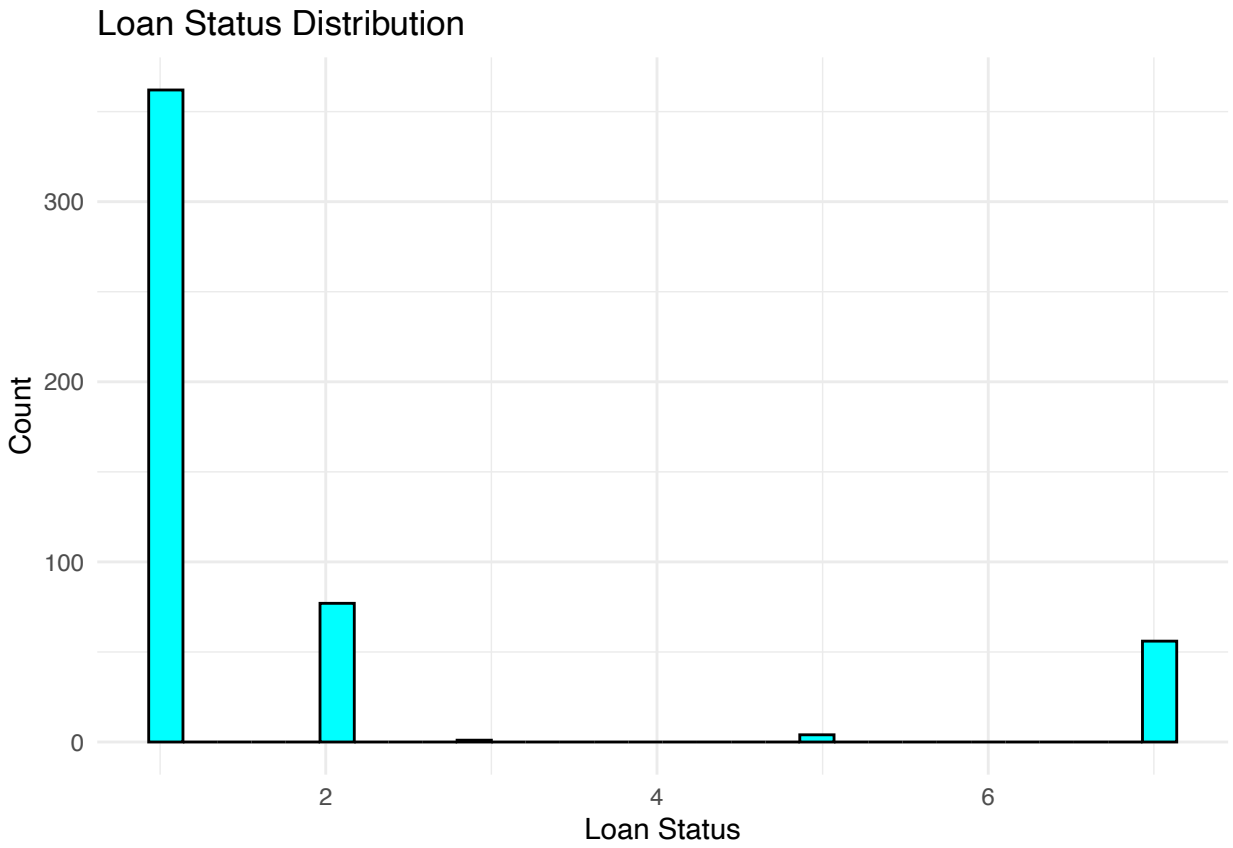



```
RndSampledData %>% ggplot(aes(x = annual_inc)) +
  geom_histogram(bins = 30, fill = "pink", color = "black") +
  theme_minimal() +
  labs(title = "Annual Income Distribution", x = "Annual Income", y = "Frequency")
```

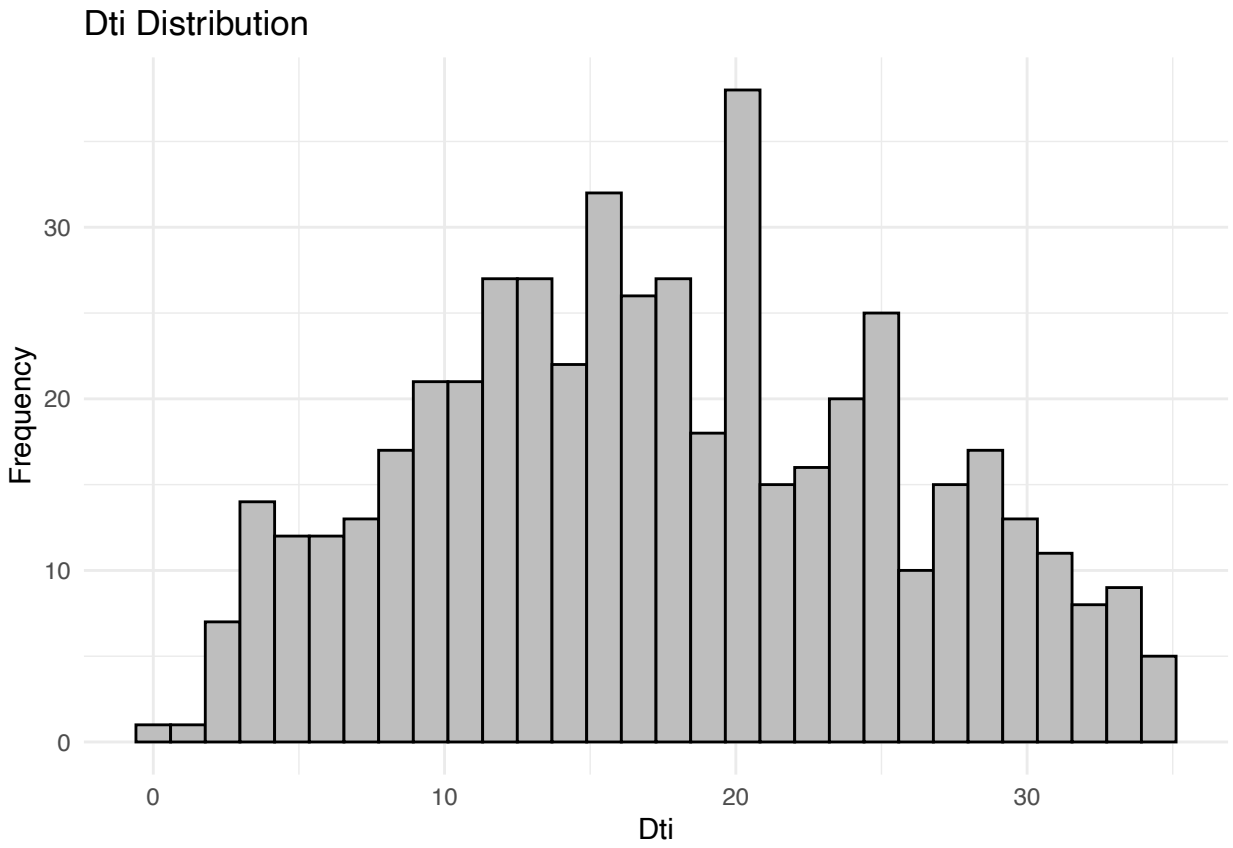



```
RndSampledData %>% ggplot(aes(x = loan_status)) +
  geom_histogram(fill = "cyan", color = "black") +
  theme_minimal() +
  labs(title = "Loan Status Distribution", x = "Loan Status", y = "Count")
```

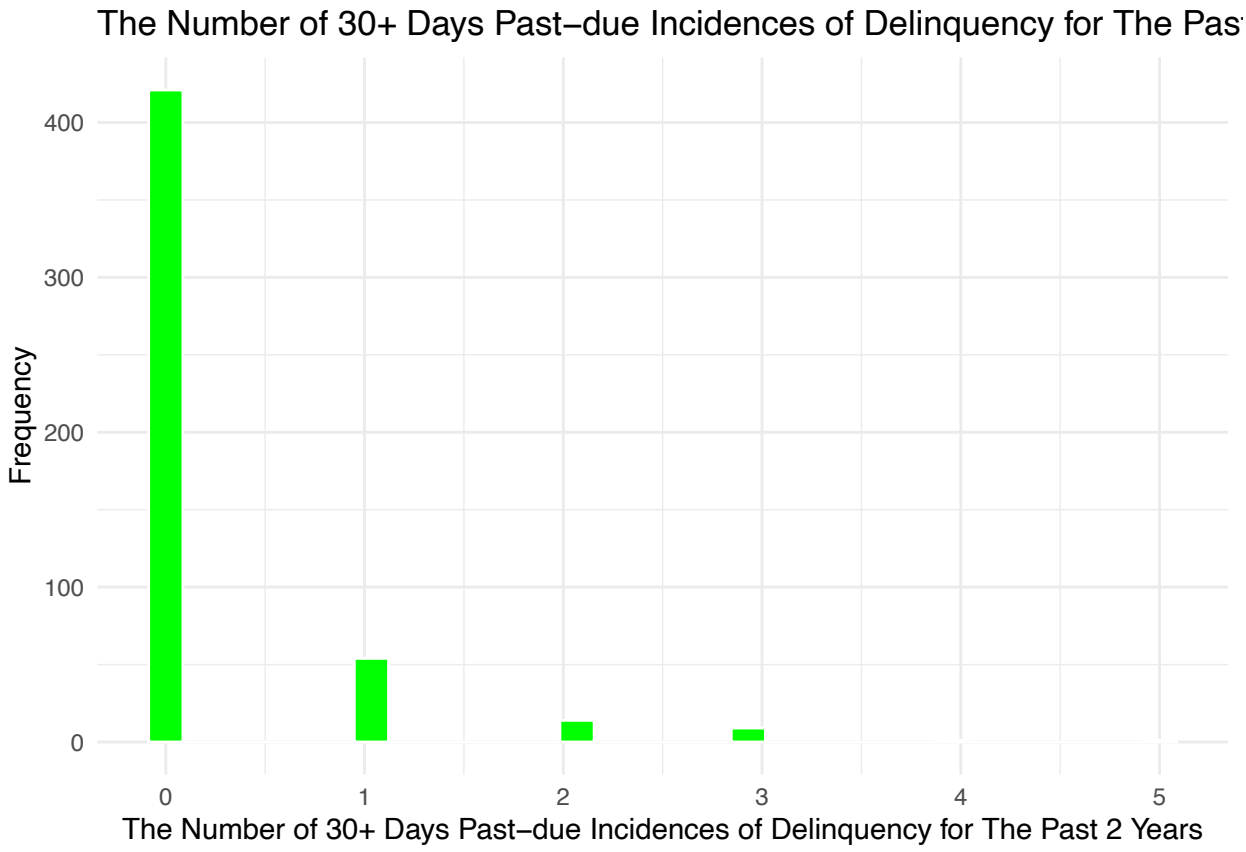
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



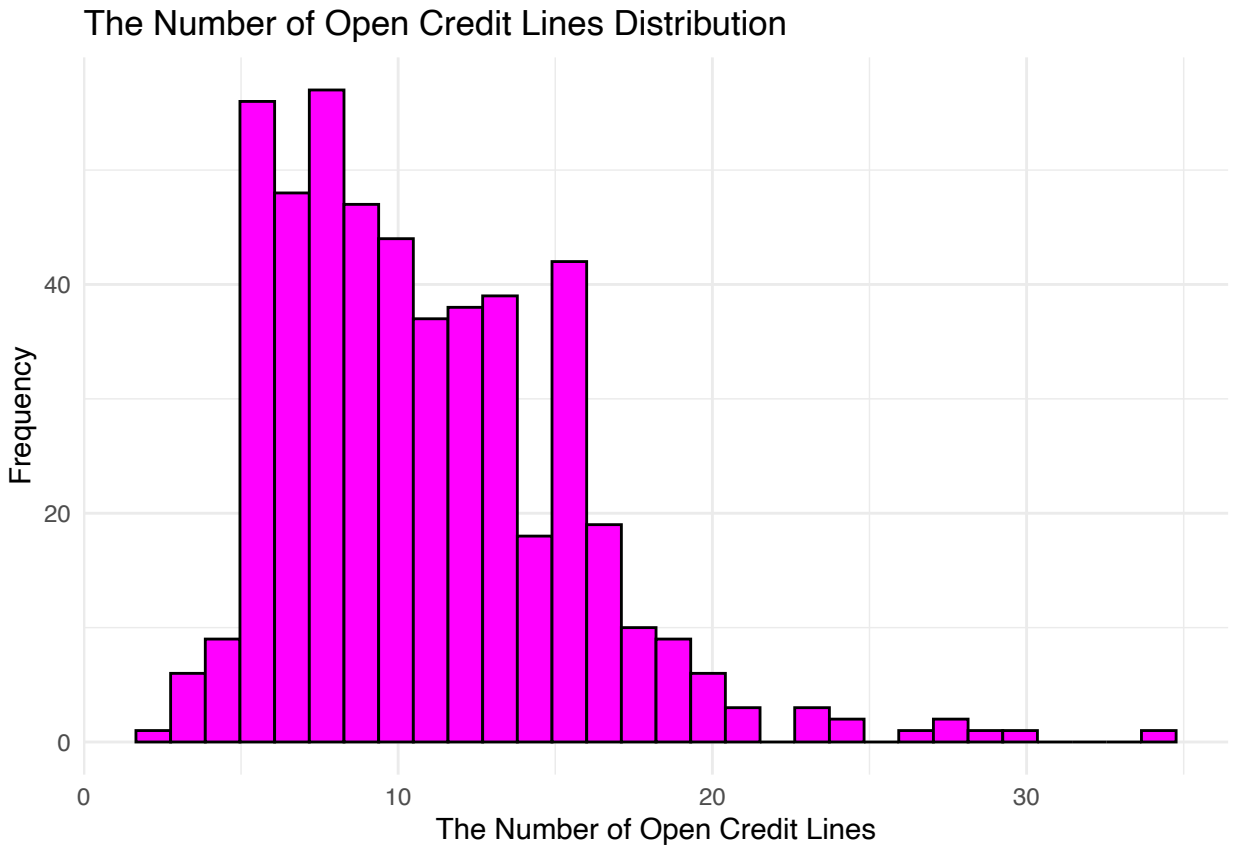

```
RndSampledData %>% ggplot(aes(x = dti)) +
  geom_histogram(bins = 30, fill = "grey", color = "black") +
  theme_minimal() +
  labs(title = "Dti Distribution", x = "Dti", y = "Frequency")
```



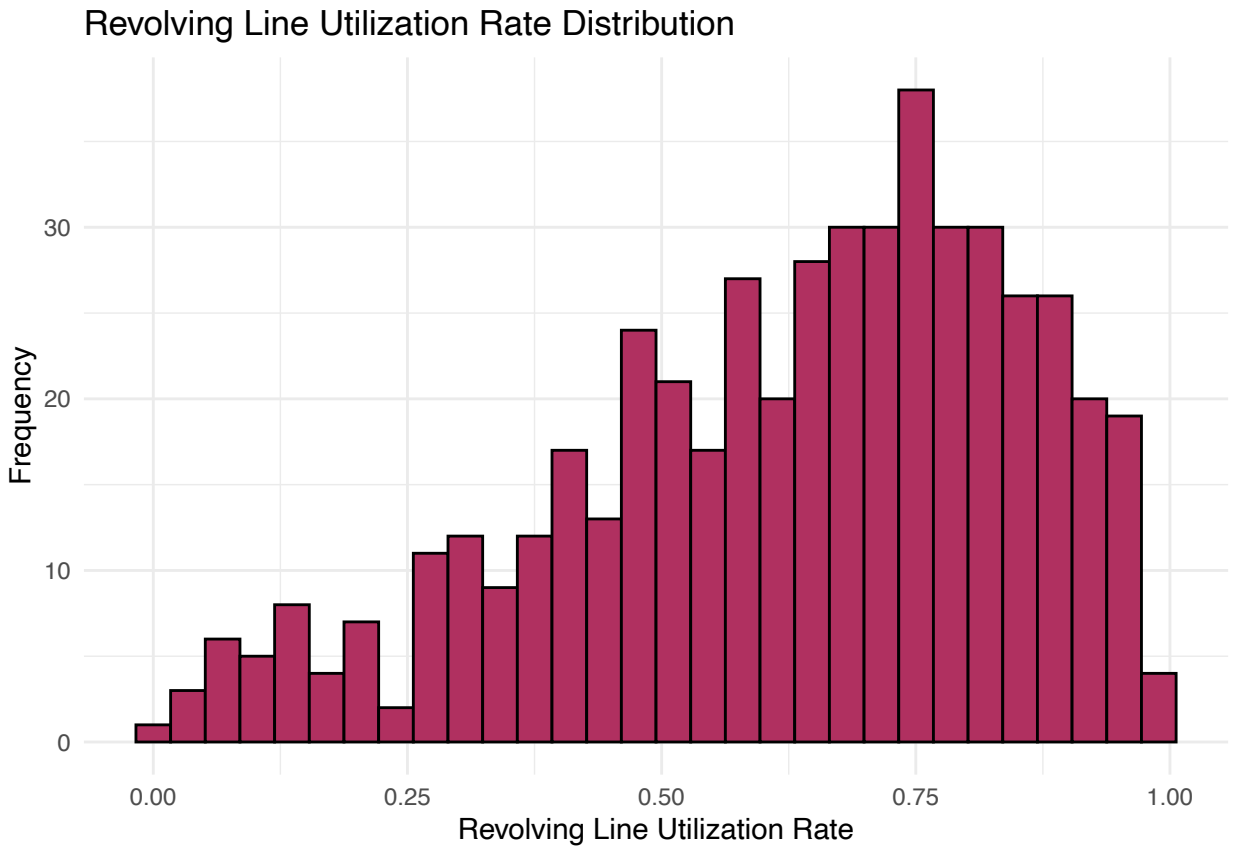

```
RndSampledData %>% ggplot(aes(x = delinq_2yrs)) +
  geom_histogram(bins = 30, fill = "green", color = "white") +
  theme_minimal() +
  labs(title = "The Number of 30+ Days Past-due Incidences of Delinquency for The Past 2 Years Distribu
```



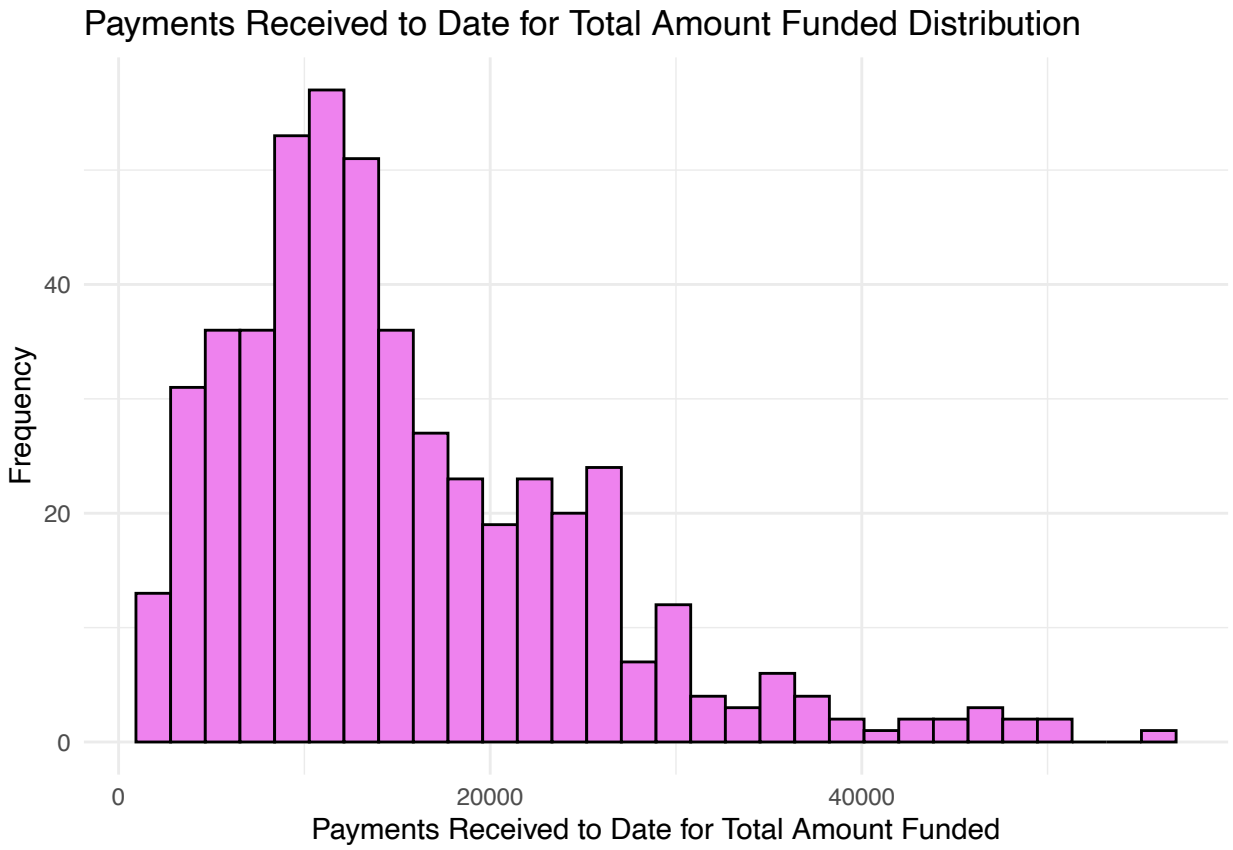

```
RndSampledData %>% ggplot(aes(x = open_acc)) +
  geom_histogram(bins = 30, fill = "magenta", color = "black") +
  theme_minimal() +
  labs(title = "The Number of Open Credit Lines Distribution", x = "The Number of Open Credit Lines", y
```



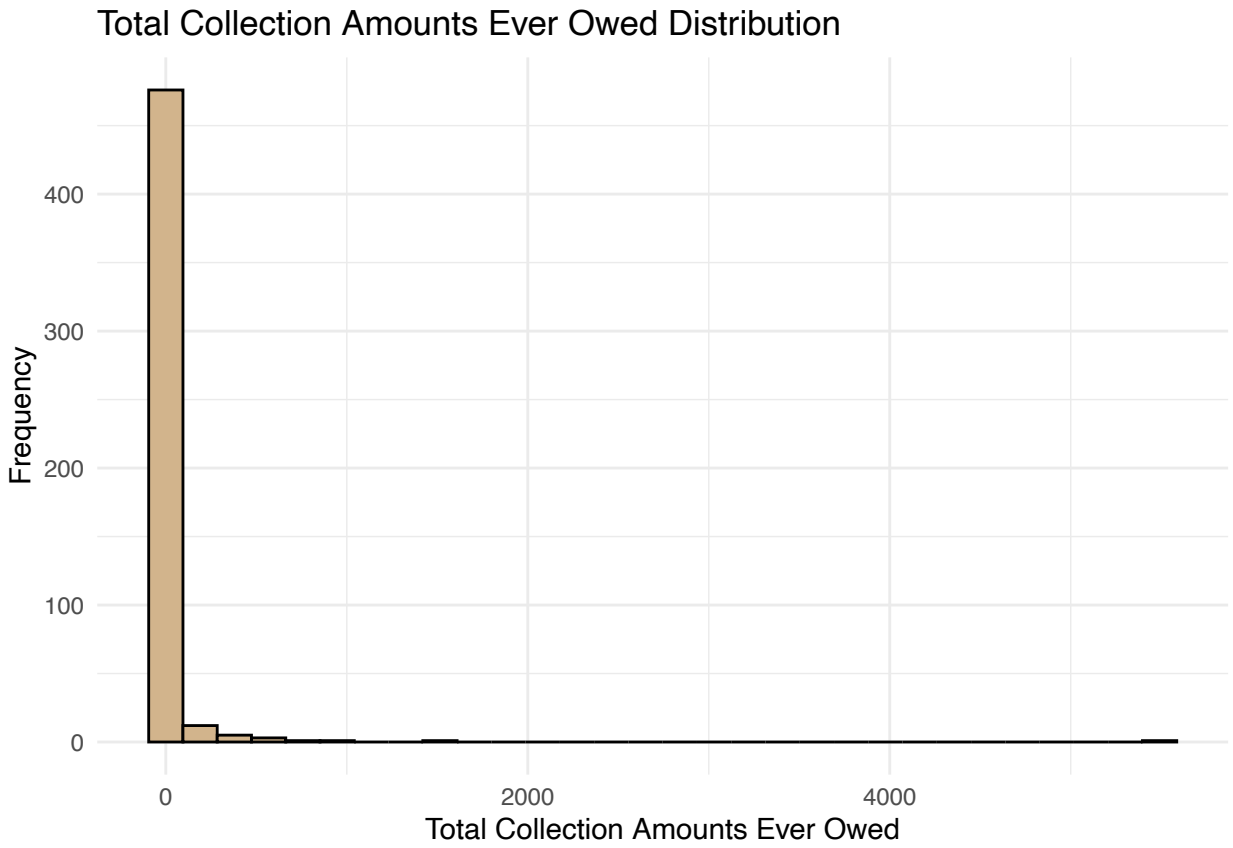

```
RndSampledData %>% ggplot(aes(x = revol_util)) +
  geom_histogram(bins = 30, fill = "maroon", color = "black") +
  theme_minimal() +
  labs(title = "Revolving Line Utilization Rate Distribution", x = "Revolving Line Utilization Rate", y
```



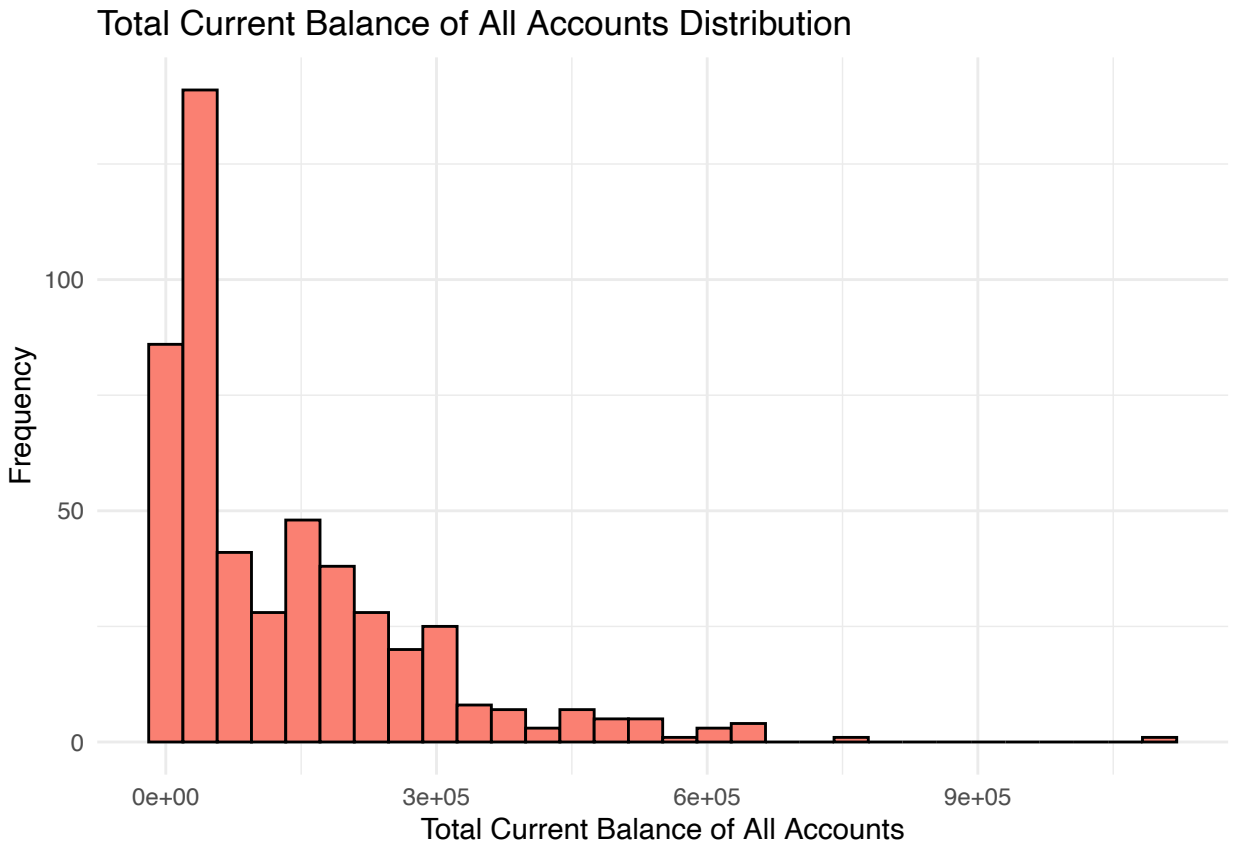

```
RndSampledData %>% ggplot(aes(x = total_pymnt)) +
  geom_histogram(bins = 30, fill = "violet", color = "black") +
  theme_minimal() +
  labs(title = "Payments Received to Date for Total Amount Funded Distribution", x = "Payments Received
```




```
RndSampledData %>% ggplot(aes(x = tot_coll_amt)) +
  geom_histogram(bins = 30, fill = "tan", color = "black") +
  theme_minimal() +
  labs(title = "Total Collection Amounts Ever Owed Distribution", x = "Total Collection Amounts Ever Ow
```




```
RndSampledData %>% ggplot(aes(x = tot_cur_bal)) +
  geom_histogram(bins = 30, fill = "salmon", color = "black") +
  theme_minimal() +
  labs(title = "Total Current Balance of All Accounts Distribution", x = "Total Current Balance of All
```



There are some data that present far from the other observations. We will examine the dataset to see whether there are potential outlier as part of pre-analysis as cluster analysis is sensitive to outliers.

1. Univariate Detection Methods by Z-scores.

```
# We will check for potential outliers by using Z scores.
RndSampledData_zscore = mutate(RndSampledData,
                        loan_amnt_Z_score = (loan_amnt - mean(loan_amnt))/sd(loan_amnt),
                        sub_grade_Z_score = (sub_grade - mean(sub_grade))/sd(sub_grade),
                        emp_length_Z_score = (emp_length - mean(emp_length))/sd(emp_length),
                        annual_inc_Z_score = (annual_inc - mean(annual_inc))/sd(annual_inc),
                        loan_status_Z_score = (loan_status - mean(loan_status))/sd(loan_status),
                        dti_Z_score = (dti - mean(dti))/sd(dti),
                        delinq_2yrs_Z_score = (delinq_2yrs - mean(delinq_2yrs))/sd(delinq_2yrs),
                        open_acc_Z_score = (open_acc - mean(open_acc))/sd(open_acc),
                        revol_util_Z_score = (revol_util - mean(revol_util))/sd(revol_util),
                        total_pymnt_Z_score = (total_pymnt - mean(total_pymnt))/sd(total_pymnt),
                        tot_coll_amt_Z_score = (tot_coll_amt - mean(tot_coll_amt))/sd(tot_coll_amt),
                        tot_cur_bal_Z_score = (tot_cur_bal - mean(tot_cur_bal))/sd(tot_cur_bal)
)

#Check how many observations with z-scores more than 4 (since our data frame is big sampled data, we wi
zscore_more4 <- apply(RndSampledData_zscore[, 13:24], 1, function(row) any(row > 4))
data_zscore4 <- RndSampledData_zscore[zscore_more4, ]
print(sum(sum(apply(data_zscore4 > 4, MARGIN = 1, any))))
```

```
## [1] 18
```

There are 18 potential outliers from univariate detection methods with z-scores. We will continue to check on

datset by multi-dimension detection methods with Mahalanobis Distance and p-value for each distance.

Calculate Mahalanobis distance to identify potential outliers.

```
Maha <- mahalanobis(RndSampledData,colMeans(RndSampledData),cov(RndSampledData))
print(sum(Maha>44))
```

```
## [1] 7
```

Based on the results, there are 10 of the distances are much higher than others. Next, we will identify any of the distances that are statistically significant by calculating p-values. The p-value for each distance is calculated as the Chi-Square statistic of the Mahalanobis distance with k-1 degrees of freedom, where k is the number of variables in this dataset, which is 12. So, the degrees of freedom is 11.

```
MahaPvalue <-pchisq(Maha,df=11,lower.tail = FALSE)
print(sum(MahaPvalue<0.001))
```

```
## [1] 16
```

From the result, there are 16 observations with p values less than 0.001 that are considered to be outliers. We consider dropping it for the cluster analysis.

We then add the Mahalanobis distance and its p values into the data frame to identify which these cases are.

```
RndSampledData_zscore$MahaDistance <- mahalanobis(RndSampledData,colMeans(RndSampledData),cov(RndSample
RndSampledData_zscore$MahaPvalue <- pchisq(Maha, df = 11, lower.tail = FALSE)
```

```
data_no_outliers_Maha = RndSampledData_zscore[!(RndSampledData_zscore$MahaPvalue < 0.001), ]

data_no_outliers_Maha_Zscore = data_no_outliers_Maha[!(apply(data_no_outliers_Maha[, 13:24], 1, functio
```

```
data_no_outliers <- select(data_no_outliers_Maha_Zscore, loan_amnt, sub_grade, emp_length, annual_inc,
```

```
summary(data_no_outliers)
```

```
##    loan_amnt       sub_grade    emp_length       annual_inc      loan_status
##  Min.   : 1000   Min.   : 1   Min.   : 1.000   Min.   : 18000   Min.   :1.000
##  1st Qu.: 8250   1st Qu.: 7   1st Qu.: 3.000   1st Qu.: 45000   1st Qu.:1.000
##  Median :12000   Median :10   Median : 6.000   Median : 60000   Median :1.000
##  Mean   :13959   Mean   :11   Mean   : 6.019   Mean   : 69160   Mean   :1.811
##  3rd Qu.:20000   3rd Qu.:15   3rd Qu.:10.000   3rd Qu.: 84000   3rd Qu.:2.000
##  Max.   :35000   Max.   :29   Max.   :10.000   Max.   :310000   Max.   :7.000
##       dti         delinq_2yrs        open_acc       revol_util
##  Min.   : 0.40   Min.   :0.0000   Min.   : 2.00   Min.   :0.0000
##  1st Qu.:11.62   1st Qu.:0.0000   1st Qu.: 8.00   1st Qu.:0.4770
##  Median :17.18   Median :0.0000   Median :10.00   Median :0.6680
##  Mean   :17.54   Mean   :0.1677   Mean   :10.65   Mean   :0.6255
##  3rd Qu.:23.78   3rd Qu.:0.0000   3rd Qu.:13.00   3rd Qu.:0.7980
##  Max.   :34.92   Max.   :2.0000   Max.   :28.00   Max.   :0.9890
##   total_pymnt     tot_coll_amt     tot_cur_bal
##  Min.   : 1025   Min.   :  0.00   Min.   :  2486
##  1st Qu.: 8765   1st Qu.:  0.00   1st Qu.: 24545
##  Median :12973   Median :  0.00   Median : 68855
##  Mean   :15468   Mean   : 16.21   Mean   :127128
##  3rd Qu.:21122   3rd Qu.:  0.00   3rd Qu.:193740
##  Max.   :51019   Max.   :854.00   Max.   :659629
```

For variable total_rec_late_fee, after removing outliers, all the observations value turn to 0, so we will drop this variable as it has no more information for the variable.

```
# Standardised data
data_no_outliers_std <- data_no_outliers %>% mutate_all(~scale(.) %>% as.vector)
```

As can see below, the mean for each column is nearly 0 and standard deviation of 1, so all the variables are standarized.

```
#Observe overall data
describe(data_no_outliers_std)
```

```
##              vars   n mean sd median trimmed  mad   min   max range  skew
## loan_amnt       1 477    0  1  -0.25   -0.08 0.99 -1.67  2.71  4.37  0.69
## sub_grade       2 477    0  1  -0.17   -0.07 0.77 -1.73  3.11  4.83  0.68
## emp_length      3 477    0  1  -0.01    0.04 1.70 -1.44  1.14  2.58 -0.12
## annual_inc      4 477    0  1  -0.24   -0.14 0.75 -1.32  6.22  7.54  2.33
## loan_status     5 477    0  1  -0.44   -0.29 0.00 -0.44  2.84  3.28  2.33
## dti             6 477    0  1  -0.04   -0.01 1.10 -2.13  2.16  4.29  0.11
## delinq_2yrs     7 477    0  1  -0.38   -0.26 0.00 -0.38  4.11  4.49  2.70
## open_acc        8 477    0  1  -0.15   -0.07 1.04 -2.02  4.04  6.06  0.75
## revol_util      9 477    0  1   0.19    0.08 1.01 -2.76  1.60  4.36 -0.64
## total_pymnt    10 477    0  1  -0.26   -0.11 0.85 -1.51  3.72  5.23  1.10
## tot_coll_amt   11 477    0  1  -0.20   -0.20 0.00 -0.20 10.51 10.71  6.50
## tot_cur_bal    12 477    0  1  -0.44   -0.17 0.63 -0.93  3.98  4.92  1.47
##              kurtosis   se
## loan_amnt       -0.16 0.05
## sub_grade        0.15 0.05
## emp_length      -1.55 0.05
## annual_inc       8.71 0.05
## loan_status      3.75 0.05
## dti             -0.77 0.05
## delinq_2yrs      6.72 0.05
## open_acc         0.54 0.05
## revol_util      -0.29 0.05
## total_pymnt      1.22 0.05
## tot_coll_amt    48.38 0.05
## tot_cur_bal      1.96 0.05
```

Checking multicolinearity

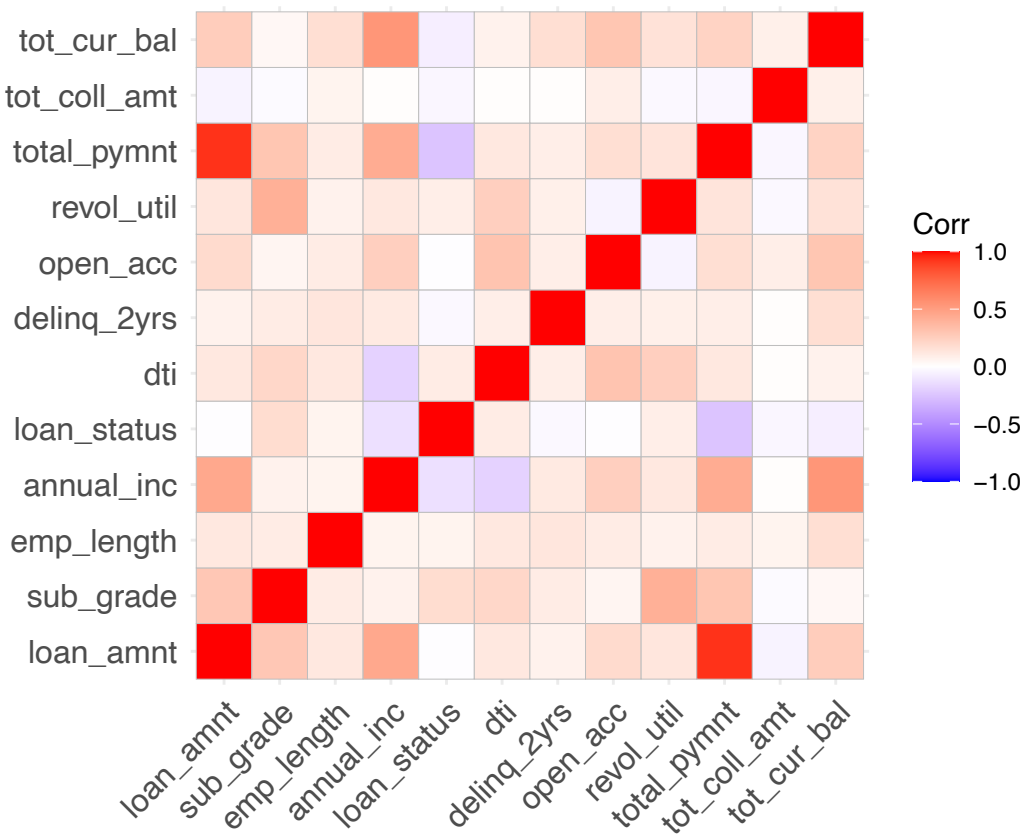
1. Pairwise correlation

```
#Create matrix to determine the correlation between each variable
Loan_Matrix<-cor(data_no_outliers_std)
round(Loan_Matrix, 2)
```

```
##              loan_amnt sub_grade emp_length annual_inc loan_status   dti
## loan_amnt         1.00      0.29       0.12       0.45       -0.01  0.12
## sub_grade         0.29      1.00       0.10       0.07        0.18  0.21
## emp_length        0.12      0.10       1.00       0.06        0.06  0.12
## annual_inc        0.45      0.07       0.06       1.00       -0.13 -0.19
## loan_status      -0.01      0.18       0.06      -0.13        1.00  0.10
## dti               0.12      0.21       0.12      -0.19        0.10  1.00
## delinq_2yrs       0.07      0.10       0.13       0.11       -0.03  0.09
## open_acc          0.19      0.05       0.10       0.25       -0.01  0.31
## revol_util        0.13      0.41       0.07       0.12        0.09  0.25
```

```
## total_pymnt       0.93      0.30       0.10       0.43       -0.25  0.12
## tot_coll_amt     -0.05     -0.02       0.06       0.01       -0.04  0.01
## tot_cur_bal       0.26      0.04       0.17       0.54       -0.07  0.07
##              delinq_2yrs open_acc revol_util total_pymnt tot_coll_amt
## loan_amnt           0.07     0.19       0.13        0.93        -0.05
## sub_grade           0.10     0.05       0.41        0.30        -0.02
## emp_length          0.13     0.10       0.07        0.10         0.06
## annual_inc          0.11     0.25       0.12        0.43         0.01
## loan_status        -0.03    -0.01       0.09       -0.25        -0.04
## dti                 0.09     0.31       0.25        0.12         0.01
## delinq_2yrs         1.00     0.09       0.08        0.09         0.01
## open_acc            0.09     1.00      -0.05        0.17         0.09
## revol_util          0.08    -0.05       1.00        0.14        -0.03
## total_pymnt         0.09     0.17       0.14        1.00        -0.04
## tot_coll_amt        0.01     0.09      -0.03       -0.04         1.00
## tot_cur_bal         0.17     0.30       0.15        0.23         0.08
##              tot_cur_bal
## loan_amnt           0.26
## sub_grade           0.04
## emp_length          0.17
## annual_inc          0.54
## loan_status        -0.07
## dti                 0.07
## delinq_2yrs         0.17
## open_acc            0.30
## revol_util          0.15
## total_pymnt         0.23
## tot_coll_amt        0.08
## tot_cur_bal         1.00
```

```
ggcorrplot(Loan_Matrix)
```



As a result, we can interpret that the data has multicollinearity which is needed to manage in order to prepare for cluster analysis

```
#Full correlation metric might hard to see, generate correlation by using lowerCor might be easier to s
lowerCor(data_no_outliers_std)
```

```
##              ln_mn sb_gr emp_l annl_ ln_st dti   dln_2 opn_c rvl_t ttl_p tt_cl_
## loan_amnt     1.00
## sub_grade     0.29  1.00
## emp_length    0.12  0.10  1.00
## annual_inc    0.45  0.07  0.06  1.00
## loan_status  -0.01  0.18  0.06 -0.13  1.00
## dti           0.12  0.21  0.12 -0.19  0.10  1.00
## delinq_2yrs   0.07  0.10  0.13  0.11 -0.03  0.09  1.00
## open_acc      0.19  0.05  0.10  0.25 -0.01  0.31  0.09  1.00
## revol_util    0.13  0.41  0.07  0.12  0.09  0.25  0.08 -0.05  1.00
## total_pymnt   0.93  0.30  0.10  0.43 -0.25  0.12  0.09  0.17  0.14  1.00
## tot_coll_amt -0.05 -0.02  0.06  0.01 -0.04  0.01  0.01  0.09 -0.03 -0.04  1.00
## tot_cur_bal   0.26  0.04  0.17  0.54 -0.07  0.07  0.17  0.30  0.15  0.23  0.08
## [1]  1.00
```

We can see that the result is under the condition of 1. at least 1 pairwise > 0.8 or 2. Many sufficient correlations are found (Correlation > 0.3)

2. KMO

```
#Using KMO to check sampling adequacy and correlation
#Looking for KMO greater than 0.5
```

```
KMO(data_no_outliers_std)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = data_no_outliers_std)
## Overall MSA =  0.51
## MSA for each item =
##    loan_amnt    sub_grade   emp_length   annual_inc  loan_status          dti
##         0.50         0.64         0.76         0.65         0.13         0.44
##  delinq_2yrs     open_acc   revol_util  total_pymnt tot_coll_amt  tot_cur_bal
##         0.75         0.57         0.53         0.49         0.63         0.66
```

Overall KMO is 0.51, more than than 0.5, meaning that overall variables are highly correlated. However, loan_status, dti, and total_pymnt have KMO less than 0.5, we will still using this variables and check the analysis to manage this uncorrelated variables.

3. Bartlett's test

```
#Using Bartlett's to check statistical significant of data
#Looking for P-value<0.05
cortest.bartlett(data_no_outliers_std)
```

```
## R was not square, finding R from data
```

```
## $chisq
## [1] 1971.555
##
## $p.value
## [1] 0
##
## $df
## [1] 66
```

```
cortest.bartlett(data_no_outliers_std, n=500)
```

```
## R was not square, finding R from data
```

```
## $chisq
## [1] 1971.555
##
## $p.value
## [1] 0
##
## $df
## [1] 66
```

We check the assumptions to see whether the data are suitable for PCA:

1. Pairwise Correlation

1.1 Rule of thumb: If at least one pairwise correlation > 0.8, then we conclude that these variables are highly correlated. 1.2 a reasonable number of pairwise correlations that are > 0.3

2. KMO is used for assessing sampling adequacy and evaluates the correlations and partial correlations to determine if the data are likely to coalesce on components (i.e. some items highly correlated, some not) If KMO > 0.5 then we conclude that some variables are highly correlated The Kaiser-Meyer-Olkin (KMO) test is a standard to assess the suitability of a data set for PCA. We are looking for a KMO value of 0.5 or more. Here it is 0.51, so we are good.

3. Bartlett Hypothesis: The Bartlett's test evaluates whether or not our correlation matrix is an identity matrix (1 on the diagonal & 0 on the off-diagonal). p-value < 0.05 then we conclude that the correlation matrix is different from an identity matrix

Conclusion: the three methods confirmed that PCA can be implemented.

#Performing PCA

```
#Do PCA
PCA_Model<-principal(data_no_outliers_std, 12, rotate="none", weights=TRUE, scores=TRUE)
print(PCA_Model)
```

```
## Principal Components Analysis
## Call: principal(r = data_no_outliers_std, nfactors = 12, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10  PC11
## loan_amnt     0.84 -0.07 -0.35 -0.22  0.02 -0.01  0.22  0.14  0.18  0.03 -0.08
## sub_grade     0.43  0.59 -0.25  0.14  0.03  0.19  0.01  0.12 -0.49 -0.28  0.06
## emp_length    0.26  0.19  0.35  0.14 -0.26 -0.11  0.68 -0.44 -0.08  0.07  0.02
## annual_inc    0.67 -0.44  0.05  0.33  0.26 -0.04 -0.07  0.01 -0.05  0.14  0.39
## loan_status  -0.11  0.54  0.07  0.17  0.54 -0.23  0.35  0.39  0.21  0.05 -0.01
## dti           0.25  0.60  0.30 -0.49 -0.10 -0.05 -0.23 -0.14  0.28 -0.13  0.25
## delinq_2yrs   0.26  0.09  0.30  0.32 -0.64 -0.24 -0.12  0.49  0.04  0.06 -0.01
## open_acc      0.42 -0.03  0.56 -0.43  0.24 -0.16 -0.16  0.09 -0.36  0.24 -0.14
## revol_util    0.34  0.56 -0.13  0.39  0.03  0.22 -0.36 -0.29  0.12  0.33 -0.13
## total_pymnt   0.83 -0.14 -0.38 -0.26 -0.14  0.07  0.11  0.05  0.09  0.02 -0.08
## tot_coll_amt  0.02 -0.07  0.44 -0.03  0.00  0.84  0.18  0.23  0.09  0.03  0.01
## tot_cur_bal   0.58 -0.23  0.43  0.32  0.21 -0.08 -0.16 -0.16  0.17 -0.39 -0.19
##                PC12 h2       u2 com
## loan_amnt    -0.13  1 -2.2e-16 2.0
## sub_grade    -0.01  1 -1.6e-15 4.3
## emp_length    0.00  1 -2.2e-15 3.6
## annual_inc    0.00  1  1.0e-15 3.6
## loan_status   0.04  1  7.8e-16 4.8
## dti           0.00  1  1.3e-15 4.6
## delinq_2yrs   0.00  1 -8.9e-16 3.9
## open_acc      0.00  1 -4.4e-16 5.2
## revol_util    0.00  1 -4.4e-16 5.8
## total_pymnt   0.14  1  2.9e-15 2.0
## tot_coll_amt  0.00  1 -4.4e-16 1.8
## tot_cur_bal   0.00  1  2.2e-16 5.3
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11
## SS loadings           2.86 1.63 1.36 1.08 0.97 0.96 0.93 0.81 0.59 0.45 0.31
## Proportion Var        0.24 0.14 0.11 0.09 0.08 0.08 0.08 0.07 0.05 0.04 0.03
## Cumulative Var        0.24 0.37 0.49 0.58 0.66 0.74 0.82 0.88 0.93 0.97 1.00
## Proportion Explained  0.24 0.14 0.11 0.09 0.08 0.08 0.08 0.07 0.05 0.04 0.03
## Cumulative Proportion 0.24 0.37 0.49 0.58 0.66 0.74 0.82 0.88 0.93 0.97 1.00
##                       PC12
## SS loadings           0.04
## Proportion Var        0.00
## Cumulative Var        1.00
## Proportion Explained  0.00
## Cumulative Proportion 1.00
##
```

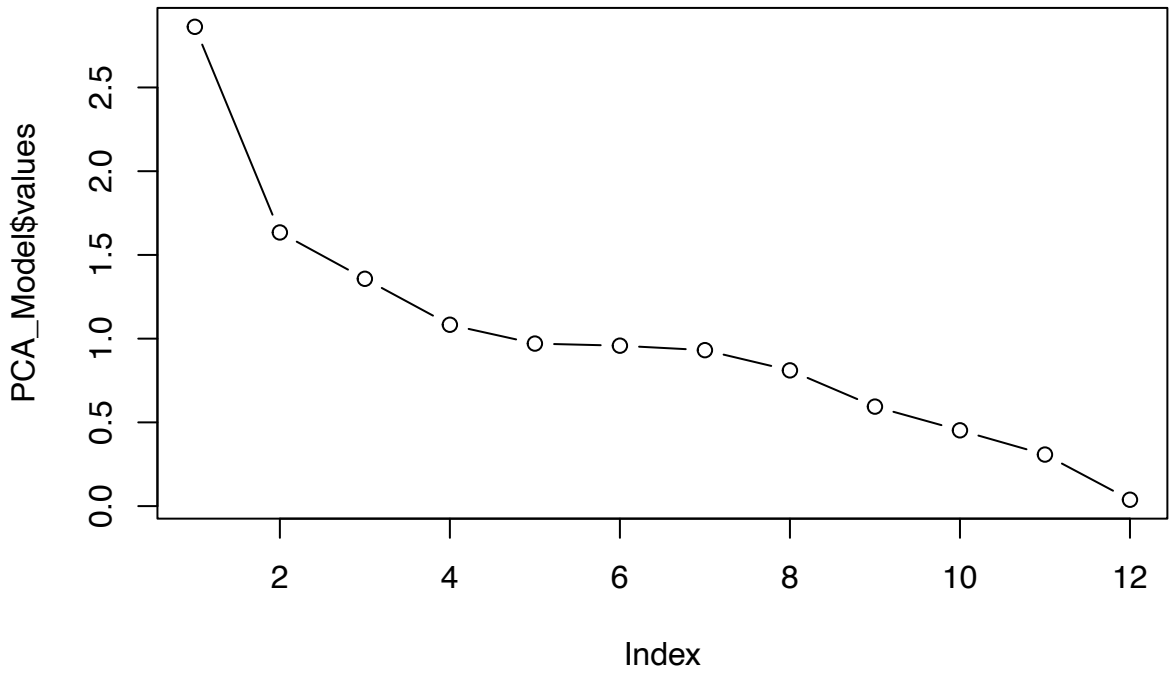
```
## Mean item complexity =  3.9
## Test of the hypothesis that 12 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

```
print.psych(PCA_Model, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_no_outliers_std, nfactors = 12, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##              item   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## loan_amnt       1  0.84       -0.35
## total_pymnt    10  0.83       -0.38
## annual_inc      4  0.67 -0.44        0.33
## tot_cur_bal    12  0.58        0.43  0.32                              -0.39
## dti             6        0.60       -0.49
## sub_grade       2  0.43  0.59                                   -0.49
## revol_util      9  0.34  0.56        0.39             -0.36               0.33
## open_acc        8  0.42        0.56 -0.43                        -0.36
## delinq_2yrs     7              0.30  0.32 -0.64              0.49
## loan_status     5        0.54              0.54        0.35  0.39
## tot_coll_amt   11              0.44              0.84
## emp_length      3              0.35                    0.68 -0.44
##              PC11  PC12 h2       u2 com
## loan_amnt                1 -2.2e-16 2.0
## total_pymnt              1  2.9e-15 2.0
## annual_inc    0.39       1  1.0e-15 3.6
## tot_cur_bal              1  2.2e-16 5.3
## dti                      1  1.3e-15 4.6
## sub_grade                1 -1.6e-15 4.3
## revol_util               1 -4.4e-16 5.8
## open_acc                 1 -4.4e-16 5.2
## delinq_2yrs              1 -8.9e-16 3.9
## loan_status              1  7.8e-16 4.8
## tot_coll_amt             1 -4.4e-16 1.8
## emp_length               1 -2.2e-15 3.6
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11
## SS loadings           2.86 1.63 1.36 1.08 0.97 0.96 0.93 0.81 0.59 0.45 0.31
## Proportion Var        0.24 0.14 0.11 0.09 0.08 0.08 0.08 0.07 0.05 0.04 0.03
## Cumulative Var        0.24 0.37 0.49 0.58 0.66 0.74 0.82 0.88 0.93 0.97 1.00
## Proportion Explained  0.24 0.14 0.11 0.09 0.08 0.08 0.08 0.07 0.05 0.04 0.03
## Cumulative Proportion 0.24 0.37 0.49 0.58 0.66 0.74 0.82 0.88 0.93 0.97 1.00
##                       PC12
## SS loadings           0.04
## Proportion Var        0.00
## Cumulative Var        1.00
## Proportion Explained  0.00
## Cumulative Proportion 1.00
##
## Mean item complexity =  3.9
```

```
## Test of the hypothesis that 12 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

```
#Plotting scree plot
plot(PCA_Model$values, type="b")
```



Our goal is to rid multicollinearity of the data set.

Results

We may want to keep first several PCs that explain certain amount of information (usually 90% or higher) contained in original variables. According to cumulative Variance: it can be PC 9

In summary, we will use PC9 to do further analysis

In terms of cumulative variance, PC9 stands out as the optimal variable, which it contributes to 93% of the cumulative variance, which is sufficient. Therefore, it is appropriate to select PC9 in further analysis

Trying PC9 without rotation

```
#Try 9 PCs
PCA_Model9wo<-principal(data_no_outliers_std, 9, rotate="none", weights=TRUE, scores=TRUE)
print(PCA_Model9wo, cut = 0.4)
```

```
## Principal Components Analysis
## Call: principal(r = data_no_outliers_std, nfactors = 9, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9   h2      u2
## loan_amnt     0.84                                                  0.97 0.02571
## sub_grade     0.43  0.59                                      -0.49 0.92 0.08383
## emp_length                                        0.68 -0.44       0.99 0.00559
```

```
## annual_inc    0.67 -0.44                                          0.83 0.17297
## loan_status         0.54              0.54                      1.00 0.00443
## dti                 0.60       -0.49                            0.92 0.08034
## delinq_2yrs                          -0.64              0.49     1.00 0.00314
## open_acc      0.42        0.56 -0.43                            0.92 0.07829
## revol_util          0.56                                        0.87 0.12843
## total_pymnt   0.83                                              0.97 0.02590
## tot_coll_amt              0.44              0.84                1.00 0.00083
## tot_cur_bal   0.58        0.43                                  0.81 0.18917
##               com
## loan_amnt     1.9
## sub_grade     3.7
## emp_length    3.5
## annual_inc    2.7
## loan_status   4.7
## dti           4.0
## delinq_2yrs   3.9
## open_acc      4.5
## revol_util    4.7
## total_pymnt   1.9
## tot_coll_amt  1.8
## tot_cur_bal   4.0
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9
## SS loadings           2.86 1.63 1.36 1.08 0.97 0.96 0.93 0.81 0.59
## Proportion Var        0.24 0.14 0.11 0.09 0.08 0.08 0.08 0.07 0.05
## Cumulative Var        0.24 0.37 0.49 0.58 0.66 0.74 0.82 0.88 0.93
## Proportion Explained  0.26 0.15 0.12 0.10 0.09 0.09 0.08 0.07 0.05
## Cumulative Proportion 0.26 0.40 0.52 0.62 0.71 0.79 0.87 0.95 1.00
##
## Mean item complexity =  3.4
## Test of the hypothesis that 9 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.04
##  with the empirical chi square  94.4  with prob <  NA
##
## Fit based upon off diagonal values = 0.97
```

There are 9 cross-loadings, and since the variables mention above with KMO less than 0.5 still has correlation with other variables, we keep those variables for further analysis.

We now proceed to FA.

Factor Analysis

PC extraction with Oblique rotation - 9 factors solution

```
#Oblique Rotation
pcModel9ob<-principal(data_no_outliers_std, 9, rotate="oblimin")
print.psych(pcModel9ob, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_no_outliers_std, nfactors = 9, rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
```

```
##              item   TC1   TC2   TC9   TC4   TC3   TC8   TC7   TC5   TC6   h2
## loan_amnt       1  0.99                                                  0.97
## total_pymnt    10  0.95                                                  0.97
## tot_cur_bal    12        0.84                                            0.81
## annual_inc      4        0.69       -0.36                                0.83
## sub_grade       2              0.95                                      0.92
## revol_util      9        0.45  0.51  0.37 -0.38                          0.87
## dti             6                    0.93                                0.92
## open_acc        8                          0.92                          0.92
## loan_status     5                                0.99                    1.00
## emp_length      3                                      1.00              0.99
## delinq_2yrs     7                                            1.00        1.00
## tot_coll_amt   11                                                  1.00 1.00
##                   u2 com
## loan_amnt    0.02571 1.0
## total_pymnt  0.02590 1.1
## tot_cur_bal  0.18917 1.2
## annual_inc   0.17297 2.0
## sub_grade    0.08383 1.1
## revol_util   0.12843 3.9
## dti          0.08034 1.1
## open_acc     0.07829 1.1
## loan_status  0.00443 1.0
## emp_length   0.00559 1.0
## delinq_2yrs  0.00314 1.0
## tot_coll_amt 0.00083 1.0
##
##                        TC1  TC2  TC9  TC4  TC3  TC8  TC7  TC5  TC6
## SS loadings           2.05 1.50 1.26 1.20 1.11 1.04 1.02 1.02 1.01
## Proportion Var        0.17 0.12 0.10 0.10 0.09 0.09 0.08 0.08 0.08
## Cumulative Var        0.17 0.30 0.40 0.50 0.59 0.68 0.76 0.85 0.93
## Proportion Explained  0.18 0.13 0.11 0.11 0.10 0.09 0.09 0.09 0.09
## Cumulative Proportion 0.18 0.32 0.43 0.54 0.64 0.73 0.82 0.91 1.00
##
##  With component correlations of
##       TC1   TC2   TC9   TC4   TC3   TC8  TC7   TC5   TC6
## TC1  1.00  0.25  0.21 -0.01  0.19 -0.12 0.10  0.09 -0.05
## TC2  0.25  1.00  0.15  0.03  0.12 -0.07 0.09  0.12  0.03
## TC9  0.21  0.15  1.00  0.24 -0.12  0.12 0.05  0.06 -0.03
## TC4 -0.01  0.03  0.24  1.00  0.06  0.08 0.08  0.05  0.01
## TC3  0.19  0.12 -0.12  0.06  1.00  0.00 0.09  0.08  0.09
## TC8 -0.12 -0.07  0.12  0.08  0.00  1.00 0.07 -0.03 -0.04
## TC7  0.10  0.09  0.05  0.08  0.09  0.07 1.00  0.14  0.07
## TC5  0.09  0.12  0.06  0.05  0.08 -0.03 0.14  1.00  0.02
## TC6 -0.05  0.03 -0.03  0.01  0.09 -0.04 0.07  0.02  1.00
##
## Mean item complexity =  1.4
## Test of the hypothesis that 9 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.04
##  with the empirical chi square  94.4  with prob <  NA
##
## Fit based upon off diagonal values = 0.97
```

1 cross-loadings

PC extraction with Orthogonal rotation - 9 factors

```
pcModel9or<-principal(data_no_outliers_std, 9, rotate="quartimax")
print.psych(pcModel9or, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_no_outliers_std, nfactors = 9, rotate = "quartimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##              item   RC1   RC4   RC9   RC2   RC3   RC8   RC7   RC5   RC6   h2
## loan_amnt       1  0.97                                                  0.97
## total_pymnt    10  0.96                                                  0.97
## tot_cur_bal    12        0.85                                            0.81
## annual_inc      4  0.41  0.73       -0.32                                0.83
## sub_grade       2              0.89                                      0.92
## revol_util      9        0.36  0.68  0.40 -0.35                          0.87
## dti             6                    0.92                                0.92
## open_acc        8                          0.90                          0.92
## loan_status     5                                0.99                    1.00
## emp_length      3                                      0.99              0.99
## delinq_2yrs     7                                            0.99        1.00
## tot_coll_amt   11                                                  1.00 1.00
##                    u2 com
## loan_amnt    0.02571 1.1
## total_pymnt  0.02590 1.1
## tot_cur_bal  0.18917 1.2
## annual_inc   0.17297 2.2
## sub_grade    0.08383 1.3
## revol_util   0.12843 2.8
## dti          0.08034 1.2
## open_acc     0.07829 1.3
## loan_status  0.00443 1.0
## emp_length   0.00559 1.0
## delinq_2yrs  0.00314 1.0
## tot_coll_amt 0.00083 1.0
##
##                        RC1  RC4  RC9  RC2  RC3  RC8  RC7  RC5  RC6
## SS loadings           2.14 1.49 1.30 1.16 1.06 1.04 1.01 1.01 1.00
## Proportion Var        0.18 0.12 0.11 0.10 0.09 0.09 0.08 0.08 0.08
## Cumulative Var        0.18 0.30 0.41 0.51 0.60 0.68 0.77 0.85 0.93
## Proportion Explained  0.19 0.13 0.12 0.10 0.09 0.09 0.09 0.09 0.09
## Cumulative Proportion 0.19 0.32 0.44 0.54 0.64 0.73 0.82 0.91 1.00
##
## Mean item complexity =  1.4
## Test of the hypothesis that 9 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.04
##  with the empirical chi square  94.4  with prob <  NA
##
## Fit based upon off diagonal values = 0.97
```

2 Cross-loadings

So, we will use 9 factors - PC extraction with Oblique rotation since it has less cross-loadings.

```r
pcModel9ob_score <- pcModel9ob$scores
```

We finished the PCA and FA, next is Cluster Analysis.

Performing Cluster Analysis

```r
# Defining linkage methods
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")
```

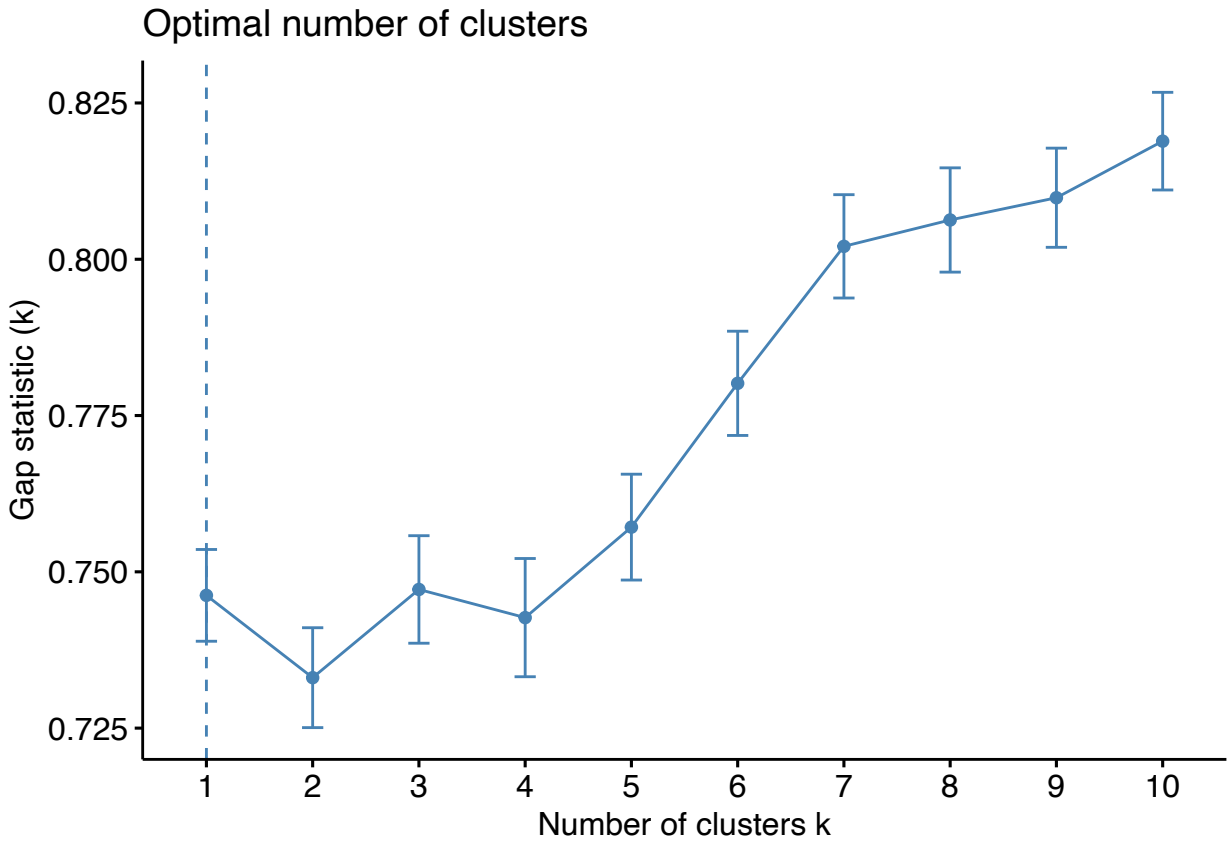
Clustering analysis after PCA & FA

```r
#Function to compute agglomerative coefficient

ac_model9ob <- function(x) {
  agnes(pcModel9ob_score, method = x)$ac
}
```

```r
# Calculate agglomerative coefficient for each clustering linkage method
sapply(m, ac_model9ob)
```

```
##   average    single  complete      ward
## 0.8252392 0.7366227 0.8935709 0.9554026
```

```r
# Calculate gap statistic for each number of clusters (up to 10 clusters)
gap_stat_h_9ob_score <- clusGap(pcModel9ob_score, FUN = hcut, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(gap_stat_h_9ob_score)
```

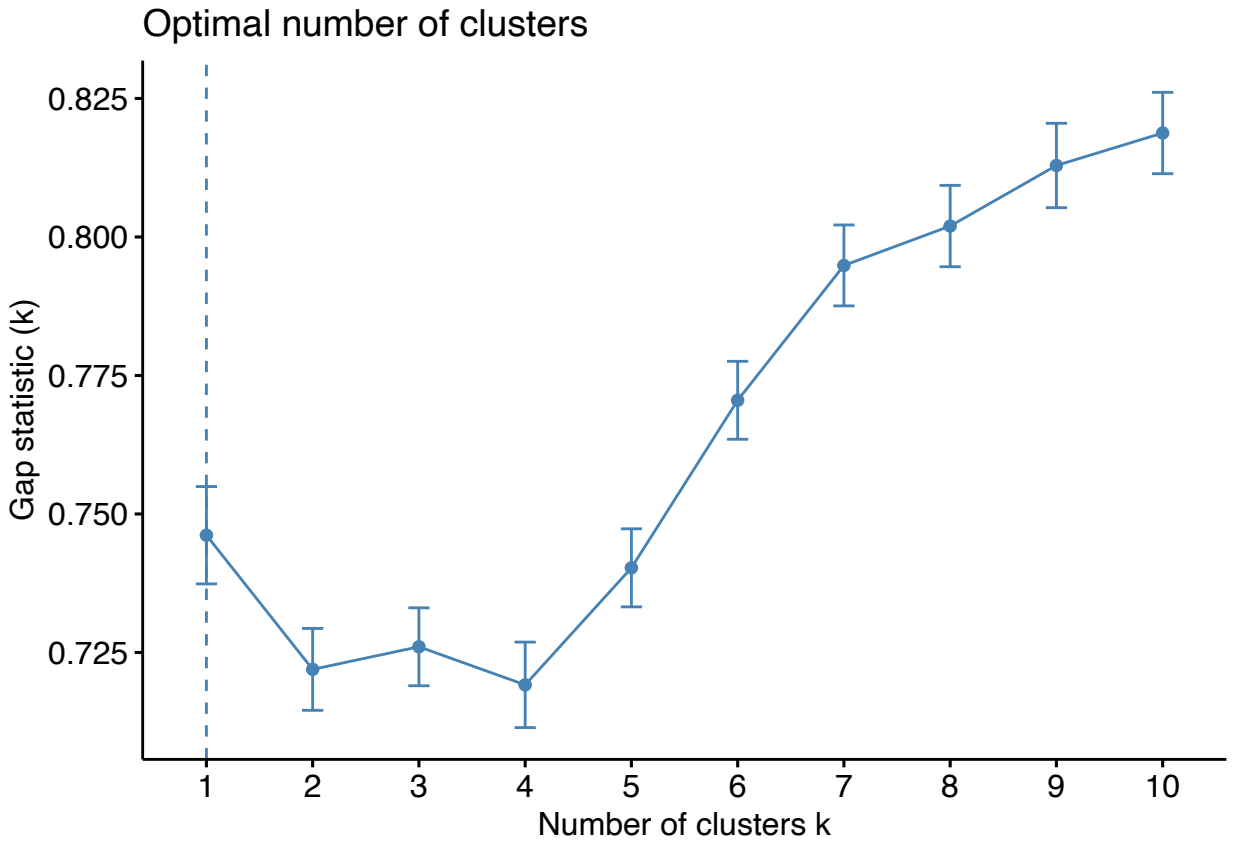



```
# From the plot we can see that the gap statistic is high at k = 3 and 7 clusters. Thus, we'll choose t
```

```
# Checking k means
# Need to check different values for nstart, kmax and B and explain them

gap_stat_k_9ob_score <- clusGap(pcModel9ob_score, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
```

```
## Warning: did not converge in 10 iterations
```

```
fviz_gap_stat(gap_stat_k_9ob_score)
```




```
# From the plot we can see that the gap statistic is high at k = 4 and 7 clusters. Thus, we'll choose t
```

We use combination of h and k-means on this analysis, we will use h to check potential number of cluster, and run the clustering with K-means. Therefore, we will proceed with number of cluster is 3.

K-means clustering

```
set.seed(55)
k_cl <- kmeans(pcModel9ob_score,3,nstart=25)
k_cl
```

```
## K-means clustering with 3 clusters of sizes 146, 53, 278
##
## Cluster means:
##          TC1        TC2        TC9        TC4        TC3        TC8        TC7
## 1  0.8383971  0.7080520  0.4543340  0.1413972  0.4930245 -0.3139381  0.3623219
## 2 -0.4071669 -0.2342459  0.3837065  0.2294968 -0.1027954  2.7397592  0.1324533
## 3 -0.3626839 -0.3271963 -0.3117597 -0.1180120 -0.2393289 -0.3574542 -0.2155360
##           TC5         TC6
## 1  0.59945602  0.23128047
## 2 -0.05119209 -0.11651142
## 3 -0.30506258 -0.09925124
##
## Clustering vector:
##   [1] 1 1 3 3 1 3 2 3 3 3 3 1 3 3 2 1 3 1 3 3 3 1 3 3 1 1 2 3 3 1 3 3 3 1 3 2 3
##  [38] 2 1 1 1 3 3 3 1 3 1 1 3 1 3 3 2 3 3 3 3 3 1 3 3 1 3 1 3 1 3 3 1 3 1 2 3 3
##  [75] 1 3 3 1 1 3 3 2 1 3 1 3 3 1 3 3 3 3 3 2 1 2 1 3 2 1 1 3 1 3 3 3 2 2 1 1 2
```

```
## [112] 2 1 3 3 1 3 2 1 1 3 3 3 3 1 3 3 3 2 1 3 3 3 3 1 3 3 3 3 2 1 3 3 1 3 1 3 3
## [149] 3 3 2 2 1 3 1 3 3 1 3 3 2 1 3 1 1 3 3 1 3 1 1 1 1 1 3 3 3 3 3 3 3 3 2 3 1
## [186] 2 3 2 1 1 3 3 2 3 3 3 1 3 3 2 3 3 2 2 3 3 1 2 3 1 1 3 1 1 3 3 3 1 3 3 1 1
## [223] 1 3 3 3 3 1 3 3 2 1 3 3 1 3 1 3 3 1 3 2 3 2 3 3 3 3 3 3 3 1 3 3 2 3 3 2 3
## [260] 3 2 3 3 3 3 2 1 3 3 1 1 3 2 3 3 2 3 3 1 1 1 1 3 3 3 3 1 3 2 3 1 3 1 3 3 3
## [297] 3 3 3 1 2 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 1 1 3 1 2 3 3 1 3 3 2 1 3 1
## [334] 3 3 1 3 2 3 2 1 1 3 1 1 3 3 3 1 3 1 3 3 3 3 2 2 1 1 3 3 1 2 3 1 2 3 3 3 1
## [371] 3 3 3 1 1 3 1 3 3 2 3 3 3 1 1 1 1 3 3 3 3 3 1 1 3 3 1 3 3 1 3 3 2 3 3 3 1
## [408] 1 3 3 3 1 3 3 1 1 1 3 1 3 3 1 1 1 3 1 3 3 1 1 3 3 3 1 3 3 3 3 1 1 2 2 3 1
## [445] 1 3 3 1 1 3 1 3 3 2 3 3 3 3 3 3 1 3 3 3 3 1 1 1 3 3 3 1 3 3 3 3 3
##
## Within cluster sum of squares by cluster:
## [1] 1610.0351  268.5535 1454.5548
##  (between_SS / total_SS =  22.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Finding distance matrix

```
distance_mat_9ob_score <- dist(pcModel9ob_score, method = "manhattan")
#print(distance_mat_9ob_score)
```

Fitting Hierarchical clustering Model to dataset

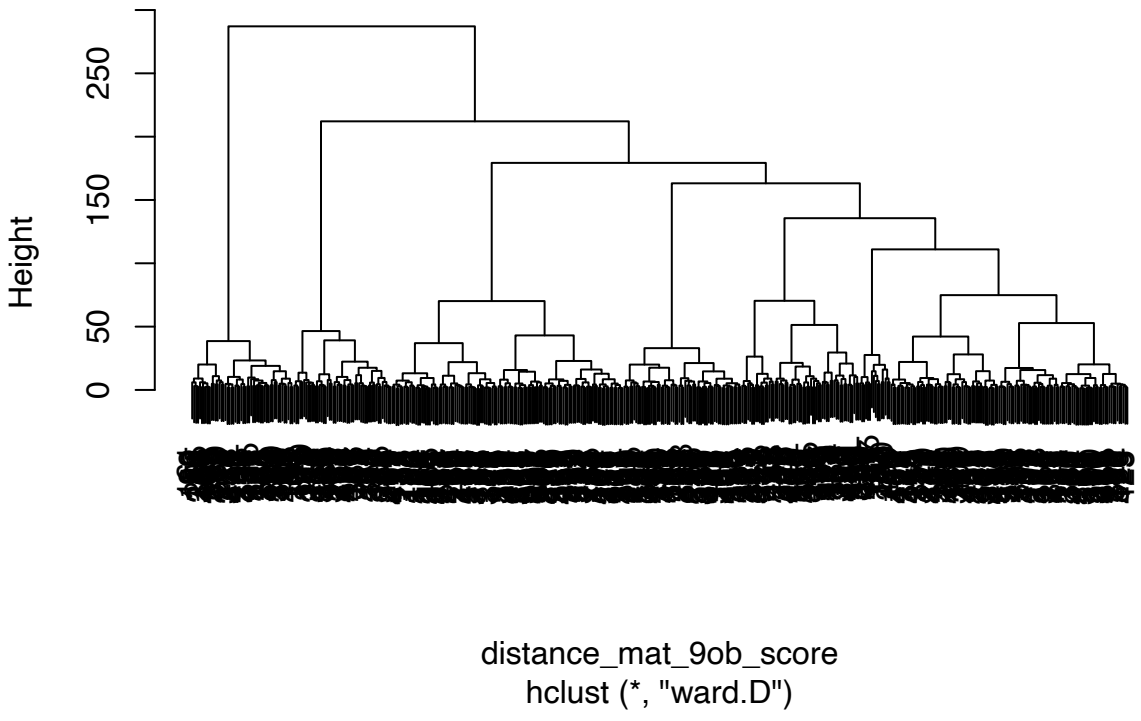
```
set.seed(240)  # Setting seed
Hierar_cl_9ob_score <- hclust(distance_mat_9ob_score, method = "ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
Hierar_cl_9ob_score
```

```
##
## Call:
## hclust(d = distance_mat_9ob_score, method = "ward")
##
## Cluster method   : ward.D
## Distance         : manhattan
## Number of objects: 477
```

```
#Plotting dendrogram
plot(Hierar_cl_9ob_score)
```



Choosing no. of clusters

```
#Cutting tree by no. of clusters
fit <- cutree(Hierar_cl_9ob_score, k = 3 )
fit
```

```
##   [1] 1 2 2 1 2 1 3 1 1 1 1 1 1 1 3 2 1 2 1 1 1 1 2 1 1 1 3 1 1 1 1 1 1 2 1 3 1
##  [38] 3 2 1 2 1 1 1 2 1 1 1 1 1 2 1 3 1 1 1 1 1 1 2 1 1 2 1 2 1 2 2 1 2 1 1 3 1 1
##  [75] 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 3 1 3 1 1 3 1 2 1 1 1 1 1 3 3 1 1 3
## [112] 3 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 3 2 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1
## [149] 1 1 3 3 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 3 1 1
## [186] 3 1 3 1 2 1 1 3 1 1 1 1 1 1 3 1 1 3 3 1 1 1 3 1 1 1 1 2 1 1 1 1 2 1 1 2 1
## [223] 1 1 1 2 1 1 1 1 3 2 1 1 1 1 1 1 1 1 1 3 1 3 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1
## [260] 1 3 1 1 1 1 3 1 2 1 1 1 1 3 1 1 3 1 1 1 2 2 1 1 1 1 1 2 1 3 1 1 1 1 1 1 1
## [297] 1 1 1 2 3 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 3 1 1 1 1 1 3 1 1 2
## [334] 1 1 1 1 3 1 3 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 1 1 1 1 1 3 1 1 3 1 1 1 2
## [371] 1 1 1 1 2 1 2 1 1 3 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1
## [408] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 3 3 1 1
## [445] 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 2 1 1 2 1 2 1 1 1 1 1 1 1 1 1
```

```
#Find number of observations in each cluster
table(fit)
```

```
## fit
##   1   2   3
## 375  50  52
```

```
pcModel9ob<-principal(data_no_outliers_std, 9, rotate="oblimin", scores = TRUE)
```

```
final_data_pc9ob <-cbind(pcModel9ob_score, cluster = fit)
```

```
hcentres<-aggregate(x=final_data_pc9ob, by=list(cluster=fit), FUN="mean")
print(hcentres)
```

```
##   cluster        TC1          TC2         TC9         TC4          TC3
## 1       1  0.0244516  0.033942940 -0.08075009 -0.05989392 -0.003856792
## 2       2  0.2511834 -0.006948996  0.19622873  0.22200010  0.131433514
## 3       3 -0.4178561 -0.238099092  0.39365094  0.21846568 -0.098564974
##          TC8         TC7         TC5         TC6 cluster
## 1 -0.3302450 -0.05866702 -0.31997640  0.03177184       1
## 2 -0.3948257  0.31459250  2.44494417 -0.11958986       2
## 3  2.7612149  0.12058666 -0.04338576 -0.11413362       3
```

```
#Assigning index
final_data_pc9ob <- cbind(Member_ID = 1:nrow(final_data_pc9ob), final_data_pc9ob)
```

Internal Validation

```
# Assigning result to data before std
# data_no_outliers2$cluster <- final_dat_pc3$cluster
```

```
#Assigning index in validation model
data_no_outliers <- cbind(Member_ID = 1:nrow(data_no_outliers), data_no_outliers)
```

We will be sampling 100 observations from data frame with no outliers for internal validation.

```
# Sampling 100 data by random sampling
set.seed(8)
RndSampledData_withindex_int_val <- sample_n(data_no_outliers, 100)
```

```
#Extract index out from dataframe
ID_int_val <- RndSampledData_withindex_int_val[, "Member_ID", drop = FALSE]

# Create a data frame for using in validation
RndSampledData_int_val <- RndSampledData_withindex_int_val[, -which(names(RndSampledData_withindex_int_
```

```
# Check data after random sampling
summary(RndSampledData_int_val)
```

```
##    loan_amnt       sub_grade       emp_length      annual_inc
##  Min.   : 1800   Min.   : 1.00   Min.   : 1.00   Min.   : 21500
##  1st Qu.: 7200   1st Qu.: 7.00   1st Qu.: 2.00   1st Qu.: 44822
##  Median :10000   Median : 9.00   Median : 6.00   Median : 53500
##  Mean   :13070   Mean   :11.04   Mean   : 5.78   Mean   : 60341
##  3rd Qu.:18694   3rd Qu.:16.00   3rd Qu.:10.00   3rd Qu.: 75000
##  Max.   :30000   Max.   :26.00   Max.   :10.00   Max.   :140000
##   loan_status        dti         delinq_2yrs      open_acc       revol_util
##  Min.   :1.00   Min.   : 0.67   Min.   :0.00   Min.   : 3.00   Min.   :0.0680
##  1st Qu.:1.00   1st Qu.:11.98   1st Qu.:0.00   1st Qu.: 7.00   1st Qu.:0.4320
##  Median :1.00   Median :16.58   Median :0.00   Median :10.00   Median :0.6260
##  Mean   :2.06   Mean   :16.91   Mean   :0.19   Mean   :10.23   Mean   :0.5859
##  3rd Qu.:2.00   3rd Qu.:22.29   3rd Qu.:0.00   3rd Qu.:13.00   3rd Qu.:0.7640
##  Max.   :7.00   Max.   :33.34   Max.   :2.00   Max.   :23.00   Max.   :0.9500
##   total_pymnt     tot_coll_amt     tot_cur_bal
```

```
##  Min.   : 1044   Min.   :  0.00   Min.   :  4494
##  1st Qu.: 6834   1st Qu.:  0.00   1st Qu.: 20178
##  Median :12024   Median :  0.00   Median : 51468
##  Mean   :14286   Mean   :  9.37   Mean   :114172
##  3rd Qu.:18357   3rd Qu.:  0.00   3rd Qu.:169883
##  Max.   :46059   Max.   :668.00   Max.   :659155
```

```
str(RndSampledData_int_val)
```

```
## 'data.frame':    100 obs. of  12 variables:
##  $ loan_amnt   : num  8000 13850 9250 10000 30000 ...
##  $ sub_grade   : num  8 7 7 21 26 20 25 10 19 1 ...
##  $ emp_length  : num  1 5 6 2 10 10 6 3 3 10 ...
##  $ annual_inc  : num  35000 45000 82000 50000 80000 95000 84000 60000 50500 90000 ...
##  $ loan_status : num  1 1 7 1 1 1 1 1 7 1 ...
##  $ dti         : num  16.5 14.8 16.7 12.2 24.4 ...
##  $ delinq_2yrs : num  0 0 0 1 0 0 0 1 0 0 ...
##  $ open_acc    : num  9 9 16 7 7 15 10 16 6 5 ...
##  $ revol_util  : num  0.318 0.48 0.497 0.95 0.868 0.596 0.491 0.667 0.727 0.068 ...
##  $ total_pymnt : num  9576 15448 1935 12779 46059 ...
##  $ tot_coll_amt: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ tot_cur_bal : num  9258 18080 39883 23345 66475 ...
```

Standardisation of data from the data frame without outliers

```
# Standardised data

data_int_val_std <- RndSampledData_int_val %>% mutate_all(~scale(.) %>% as.vector)

# Check data summary after standardisation
summary(data_int_val_std)
```

```
##    loan_amnt         sub_grade         emp_length         annual_inc
##  Min.   :-1.4822   Min.   :-1.6005   Min.   :-1.34459   Min.   :-1.6127
##  1st Qu.:-0.7720   1st Qu.:-0.6440   1st Qu.:-1.06329   1st Qu.:-0.6443
##  Median :-0.4038   Median :-0.3252   Median : 0.06188   Median :-0.2840
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.7396   3rd Qu.: 0.7907   3rd Qu.: 1.18706   3rd Qu.: 0.6086
##  Max.   : 2.2266   Max.   : 2.3848   Max.   : 1.18706   Max.   : 3.3074
##   loan_status            dti           delinq_2yrs        open_acc
##  Min.   :-0.51486   Min.   :-2.1322   Min.   :-0.4088   Min.   :-1.86633
##  1st Qu.:-0.51486   1st Qu.:-0.6472   1st Qu.:-0.4088   1st Qu.:-0.83378
##  Median :-0.51486   Median :-0.0428   Median :-0.4088   Median :-0.05937
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.:-0.02914   3rd Qu.: 0.7074   3rd Qu.:-0.4088   3rd Qu.: 0.71504
##  Max.   : 2.39942   Max.   : 2.1583   Max.   : 3.8940   Max.   : 3.29641
##    revol_util       total_pymnt       tot_coll_amt      tot_cur_bal
##  Min.   :-2.2199   Min.   :-1.3252   Min.   :-0.1342   Min.   :-0.8407
##  1st Qu.:-0.6596   1st Qu.:-0.7458   1st Qu.:-0.1342   1st Qu.:-0.7205
##  Median : 0.1720   Median :-0.2264   Median :-0.1342   Median :-0.4806
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.7635   3rd Qu.: 0.4074   3rd Qu.:-0.1342   3rd Qu.: 0.4270
##  Max.   : 1.5608   Max.   : 3.1795   Max.   : 9.4346   Max.   : 4.1772
```

```
str(data_int_val_std)
```

```
## 'data.frame':    100 obs. of  12 variables:
```

```
##  $ loan_amnt   : num  -0.667 0.103 -0.502 -0.404 2.227 ...
##  $ sub_grade   : num  -0.485 -0.644 -0.644 1.588 2.385 ...
##  $ emp_length  : num  -1.3446 -0.2194 0.0619 -1.0633 1.1871 ...
##  $ annual_inc  : num  -1.052 -0.637 0.899 -0.429 0.816 ...
##  $ loan_status : num  -0.515 -0.515 2.399 -0.515 -0.515 ...
##  $ dti         : num  -0.0494 -0.27 -0.0244 -0.618 0.9776 ...
##  $ delinq_2yrs : num  -0.409 -0.409 -0.409 1.743 -0.409 ...
##  $ open_acc    : num  -0.318 -0.318 1.489 -0.834 -0.834 ...
##  $ revol_util  : num  -1.148 -0.454 -0.381 1.561 1.209 ...
##  $ total_pymnt : num  -0.471 0.116 -1.236 -0.151 3.179 ...
##  $ tot_coll_amt: num  -0.134 -0.134 -0.134 -0.134 -0.134 ...
##  $ tot_cur_bal : num  -0.804 -0.737 -0.569 -0.696 -0.366 ...
```

```
# Checking whether the standarization is done
(column_means <- colMeans(data_int_val_std, na.rm = TRUE))
```

```
##     loan_amnt     sub_grade    emp_length    annual_inc   loan_status
##  1.283695e-18  1.379452e-16 -6.855627e-17  7.389922e-18 -8.659740e-17
##           dti   delinq_2yrs      open_acc    revol_util   total_pymnt
##  1.270165e-16  1.554312e-17 -8.631984e-17  1.964011e-16  8.893927e-17
##  tot_coll_amt   tot_cur_bal
## -4.996004e-18 -1.658396e-17
```

```
(column_std_dev <- sapply(data_int_val_std, sd))
```

```
##    loan_amnt    sub_grade   emp_length   annual_inc  loan_status          dti
##            1            1            1            1            1            1
##  delinq_2yrs     open_acc   revol_util  total_pymnt tot_coll_amt  tot_cur_bal
##            1            1            1            1            1            1
```

As we can see above, the mean for each column is nearly 0 and standard deviation of 1, so all the variables are standarized.

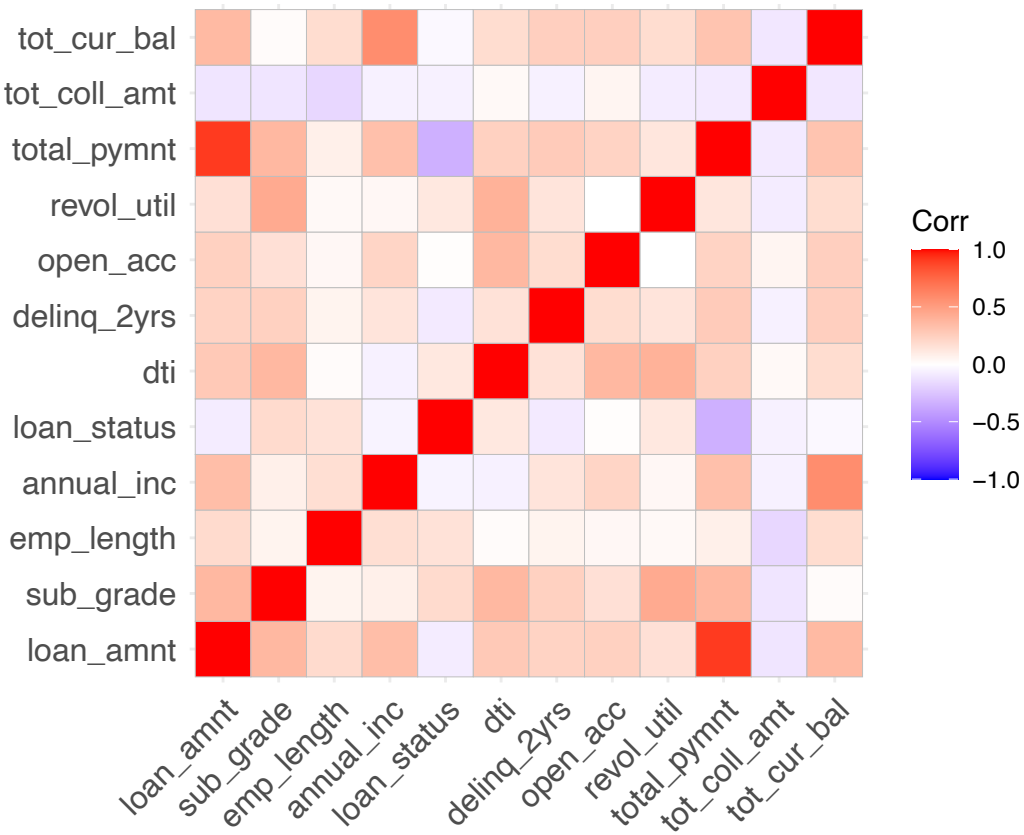
#Checking multicolinearity We check the assumptions to see whether the data are suitable for PCA: ## 1. Pairwise correlation

```
#Create matrix to determine the correlation between each variable
Loan_Matrix_int_val <-cor (data_int_val_std)
round(Loan_Matrix_int_val, 2)
```

```
##              loan_amnt sub_grade emp_length annual_inc loan_status   dti
## loan_amnt         1.00      0.37       0.19       0.34       -0.08  0.28
## sub_grade         0.37      1.00       0.06       0.08        0.19  0.37
## emp_length        0.19      0.06       1.00       0.17        0.15  0.02
## annual_inc        0.34      0.08       0.17       1.00       -0.05 -0.06
## loan_status      -0.08      0.19       0.15      -0.05        1.00  0.12
## dti               0.28      0.37       0.02      -0.06        0.12  1.00
## delinq_2yrs       0.23      0.24       0.06       0.14       -0.09  0.15
## open_acc          0.24      0.16       0.04       0.22        0.01  0.37
## revol_util        0.16      0.44       0.03       0.04        0.12  0.40
## total_pymnt       0.91      0.37       0.08       0.33       -0.34  0.24
## tot_coll_amt     -0.11     -0.11      -0.17      -0.06       -0.06  0.03
## tot_cur_bal       0.36      0.02       0.18       0.58       -0.03  0.18
##              delinq_2yrs open_acc revol_util total_pymnt tot_coll_amt
## loan_amnt           0.23     0.24       0.16        0.91        -0.11
## sub_grade           0.24     0.16       0.44        0.37        -0.11
## emp_length          0.06     0.04       0.03        0.08        -0.17
```

```
## annual_inc          0.14      0.22       0.04       0.33       -0.06
## loan_status        -0.09      0.01       0.12      -0.34       -0.06
## dti                 0.15      0.37       0.40       0.24        0.03
## delinq_2yrs         1.00      0.18       0.14       0.27       -0.06
## open_acc            0.18      1.00       0.00       0.23        0.05
## revol_util          0.14      0.00       1.00       0.13       -0.08
## total_pymnt         0.27      0.23       0.13       1.00       -0.09
## tot_coll_amt       -0.06      0.05      -0.08      -0.09        1.00
## tot_cur_bal         0.25      0.25       0.18       0.31       -0.10
##               tot_cur_bal
## loan_amnt            0.36
## sub_grade            0.02
## emp_length           0.18
## annual_inc           0.58
## loan_status         -0.03
## dti                  0.18
## delinq_2yrs          0.25
## open_acc             0.25
## revol_util           0.18
## total_pymnt          0.31
## tot_coll_amt        -0.10
## tot_cur_bal          1.00
```

```
ggcorrplot(Loan_Matrix_int_val)
```




```
#Full correlation metric might hard to see, generate correlation by using lowerCor might be easier to s
lowerCor(data_int_val_std)
```

```
##              ln_mn sb_gr emp_l annl_ ln_st dti   dln_2 opn_c rvl_t ttl_p tt_cl_
## loan_amnt     1.00
## sub_grade     0.37  1.00
## emp_length    0.19  0.06  1.00
## annual_inc    0.34  0.08  0.17  1.00
## loan_status  -0.08  0.19  0.15 -0.05  1.00
## dti           0.28  0.37  0.02 -0.06  0.12  1.00
## delinq_2yrs   0.23  0.24  0.06  0.14 -0.09  0.15  1.00
## open_acc      0.24  0.16  0.04  0.22  0.01  0.37  0.18  1.00
## revol_util    0.16  0.44  0.03  0.04  0.12  0.40  0.14  0.00  1.00
## total_pymnt   0.91  0.37  0.08  0.33 -0.34  0.24  0.27  0.23  0.13  1.00
## tot_coll_amt -0.11 -0.11 -0.17 -0.06 -0.06  0.03 -0.06  0.05 -0.08 -0.09  1.00
## tot_cur_bal   0.36  0.02  0.18  0.58 -0.03  0.18  0.25  0.25  0.18  0.31 -0.10
## [1]  1.00
```

We can see that the result is under the condition of 1. at least 1 pairwise > 0.8 or 2. Many sufficient correlations are found (Correlation > 0.3)

2. KMO

```
#Using KMO to check sampling adequacy and correlation
#Looking for KMO greater than 0.5
KMO(data_int_val_std)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = data_int_val_std)
## Overall MSA =  0.56
## MSA for each item =
##    loan_amnt    sub_grade   emp_length   annual_inc  loan_status          dti
##         0.55         0.62         0.62         0.61         0.21         0.63
##  delinq_2yrs     open_acc   revol_util  total_pymnt tot_coll_amt  tot_cur_bal
##         0.79         0.65         0.61         0.52         0.64         0.62
```

```
#Full correlation metric might hard to see, generate correlation by using lowerCor might be easier to s
lowerCor(data_int_val_std)
```

```
##              ln_mn sb_gr emp_l annl_ ln_st dti   dln_2 opn_c rvl_t ttl_p tt_cl_
## loan_amnt     1.00
## sub_grade     0.37  1.00
## emp_length    0.19  0.06  1.00
## annual_inc    0.34  0.08  0.17  1.00
## loan_status  -0.08  0.19  0.15 -0.05  1.00
## dti           0.28  0.37  0.02 -0.06  0.12  1.00
## delinq_2yrs   0.23  0.24  0.06  0.14 -0.09  0.15  1.00
## open_acc      0.24  0.16  0.04  0.22  0.01  0.37  0.18  1.00
## revol_util    0.16  0.44  0.03  0.04  0.12  0.40  0.14  0.00  1.00
## total_pymnt   0.91  0.37  0.08  0.33 -0.34  0.24  0.27  0.23  0.13  1.00
## tot_coll_amt -0.11 -0.11 -0.17 -0.06 -0.06  0.03 -0.06  0.05 -0.08 -0.09  1.00
## tot_cur_bal   0.36  0.02  0.18  0.58 -0.03  0.18  0.25  0.25  0.18  0.31 -0.10
## [1]  1.00
```

3. Bartlett's Test

```
#Using Bartlett's to check statistical significant of data
#Looking for P-value<0.05
```

```
cortest.bartlett(data_int_val_std)
```

```
## R was not square, finding R from data
```

```
## $chisq
## [1] 418.9042
##
## $p.value
## [1] 9.166778e-53
##
## $df
## [1] 66
```

```
cortest.bartlett(data_int_val_std, n=100)
```

```
## R was not square, finding R from data
```

```
## $chisq
## [1] 418.9042
##
## $p.value
## [1] 9.166778e-53
##
## $df
## [1] 66
```

Performing PCA

```
#Do PCA
PCA_Model_in_val<-principal(data_int_val_std, 12, rotate="none", weights=TRUE, scores=TRUE)
print(PCA_Model_in_val)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 12, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10  PC11
## loan_amnt     0.83 -0.15 -0.14 -0.24  0.30  0.14  0.06  0.08 -0.23  0.14  0.04
## sub_grade     0.56  0.55 -0.06 -0.23  0.02  0.03  0.14  0.31  0.32 -0.23 -0.23
## emp_length    0.24 -0.07  0.63 -0.13  0.37 -0.11  0.35 -0.47  0.17 -0.02 -0.01
## annual_inc    0.52 -0.48  0.34  0.21 -0.19  0.27  0.00  0.26  0.23 -0.16  0.28
## loan_status  -0.07  0.54  0.57  0.19  0.20  0.08  0.13  0.40 -0.31  0.11  0.05
## dti           0.50  0.55 -0.17  0.30  0.10 -0.03 -0.20 -0.32 -0.21 -0.31  0.17
## delinq_2yrs   0.46  0.00 -0.08  0.03 -0.41 -0.65  0.40  0.08 -0.11  0.02  0.09
## open_acc      0.46  0.01 -0.08  0.61  0.31 -0.32 -0.27  0.07  0.28  0.23 -0.04
## revol_util    0.40  0.58  0.05 -0.10 -0.46  0.30 -0.03 -0.25  0.16  0.31  0.08
## total_pymnt   0.81 -0.25 -0.34 -0.29  0.19  0.10  0.02  0.03 -0.08  0.07  0.02
## tot_coll_amt -0.17 -0.01 -0.44  0.55  0.06  0.35  0.59 -0.06  0.01  0.01 -0.04
## tot_cur_bal   0.60 -0.33  0.34  0.32 -0.33  0.13 -0.11 -0.12 -0.24 -0.05 -0.33
##                PC12 h2       u2 com
## loan_amnt    -0.14  1  1.1e-16 2.0
## sub_grade    -0.01  1  2.3e-15 4.6
## emp_length    0.01  1  1.4e-15 4.1
## annual_inc    0.00  1  1.1e-15 6.1
## loan_status   0.05  1 -2.0e-15 4.3
```

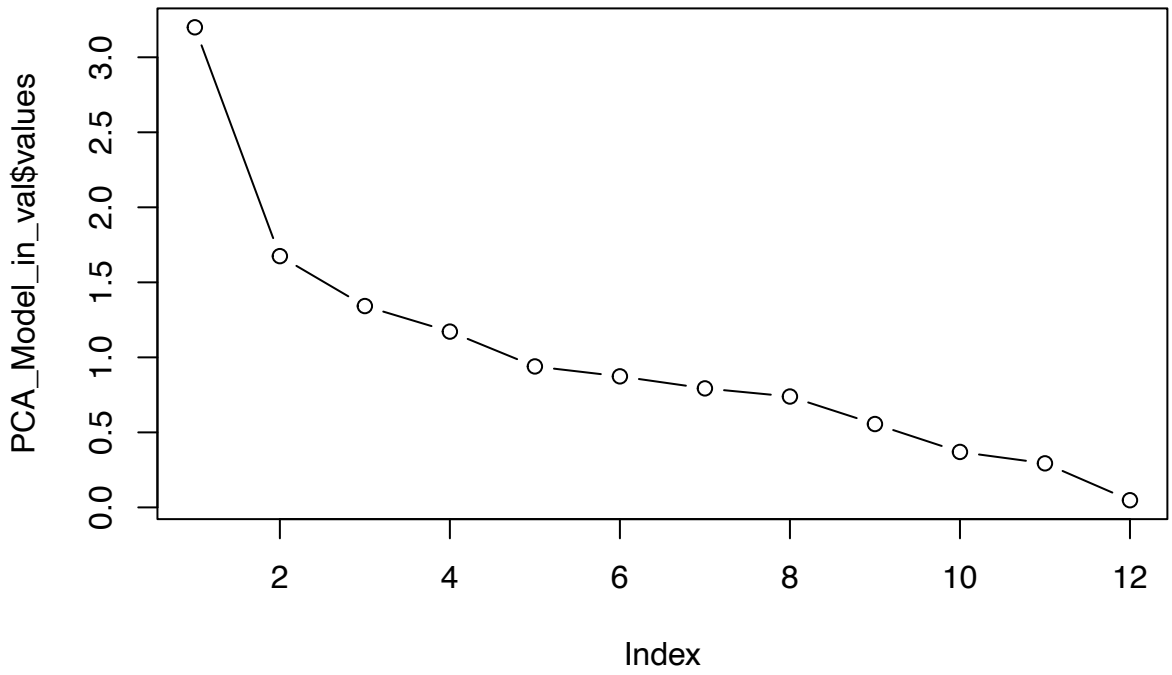
```
## dti           0.00  1  7.8e-16 5.3
## delinq_2yrs   0.00  1  2.2e-16 3.6
## open_acc      0.00  1  6.7e-16 4.6
## revol_util    0.00  1  1.0e-15 4.8
## total_pymnt   0.16  1  4.1e-15 2.2
## tot_coll_amt  0.00  1  1.1e-15 3.8
## tot_cur_bal   0.01  1  1.0e-15 5.2
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11
## SS loadings           3.20 1.67 1.34 1.17 0.94 0.87 0.79 0.74 0.56 0.37 0.29
## Proportion Var        0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Var        0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
## Proportion Explained  0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Proportion 0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
##                       PC12
## SS loadings           0.05
## Proportion Var        0.00
## Cumulative Var        1.00
## Proportion Explained  0.00
## Cumulative Proportion 1.00
##
## Mean item complexity =  4.2
## Test of the hypothesis that 12 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

```
print.psych(PCA_Model_in_val, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 12, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##              item   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## loan_amnt       1  0.83
## total_pymnt    10  0.81       -0.34
## tot_cur_bal    12  0.60 -0.33  0.34  0.32 -0.33
## sub_grade       2  0.56  0.55                                0.31  0.32
## annual_inc      4  0.52 -0.48  0.34
## revol_util      9  0.40  0.58             -0.46                       0.31
## dti             6  0.50  0.55                                -0.32       -0.31
## emp_length      3              0.63        0.37        0.35 -0.47
## loan_status     5        0.54  0.57                            0.40 -0.31
## open_acc        8  0.46              0.61  0.31 -0.32
## delinq_2yrs     7  0.46                   -0.41 -0.65  0.40
## tot_coll_amt   11             -0.44  0.55        0.35  0.59
##                PC11  PC12 h2       u2 com
## loan_amnt                1  1.1e-16 2.0
## total_pymnt              1  4.1e-15 2.2
## tot_cur_bal  -0.33       1  1.0e-15 5.2
## sub_grade                1  2.3e-15 4.6
## annual_inc               1  1.1e-15 6.1
## revol_util               1  1.0e-15 4.8
```

```
## dti                         1  7.8e-16 5.3
## emp_length                  1  1.4e-15 4.1
## loan_status                 1 -2.0e-15 4.3
## open_acc                    1  6.7e-16 4.6
## delinq_2yrs                 1  2.2e-16 3.6
## tot_coll_amt                1  1.1e-15 3.8
##
##                         PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11
## SS loadings            3.20 1.67 1.34 1.17 0.94 0.87 0.79 0.74 0.56 0.37 0.29
## Proportion Var         0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Var         0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
## Proportion Explained   0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Proportion  0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
##                        PC12
## SS loadings            0.05
## Proportion Var         0.00
## Cumulative Var         1.00
## Proportion Explained   0.00
## Cumulative Proportion  1.00
##
## Mean item complexity =  4.2
## Test of the hypothesis that 12 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

```
#Plot scree plot
plot(PCA_Model_in_val$values, type="b")
```



Our goal is to rid multicollinearity of the data set.

Results

We may want to keep first several PCs that explain certain amount of information (usually 90% or higher) contained in original variables. According to cumulative Variance: it can be PC 9

In summary, we will use PC9 to do further analysis

In terms of cumulative variance, PC9 stands out as the optimal variable, which it contributes to 92% of the cumulative variance, which is sufficient. Therefore, it is appropriate to select PC9 in further analysis

Trying PC9 without rotation

```
#Try 9 PCs
PCA_Model9wo_in<-principal(data_int_val_std, 9, rotate="none", weights=TRUE, scores=TRUE)
print(PCA_Model9wo_in, cut = 0.4)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 9, rotate = "none",
##     scores = TRUE, weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9   h2      u2
## loan_amnt     0.83                                                  0.96 0.04094
## sub_grade     0.56  0.55                                            0.89 0.10777
## emp_length                0.63                        -0.47       1.00 0.00044
## annual_inc    0.52 -0.48                                            0.89 0.10579
## loan_status         0.54  0.57                         0.40        0.98 0.01769
## dti           0.50  0.55                                            0.87 0.12636
## delinq_2yrs   0.46                    -0.41 -0.65                  0.99 0.00808
## open_acc      0.46              0.61                               0.95 0.05477
## revol_util    0.40  0.58              -0.46                        0.90 0.10470
## total_pymnt   0.81                                                  0.97 0.02994
## tot_coll_amt             -0.44  0.55              0.59             1.00 0.00169
## tot_cur_bal   0.60                                                  0.89 0.11298
##               com
## loan_amnt     1.9
## sub_grade     3.7
## emp_length    4.1
## annual_inc    5.1
## loan_status   4.2
## dti           4.2
## delinq_2yrs   3.5
## open_acc      4.1
## revol_util    4.1
## total_pymnt   2.1
## tot_coll_amt  3.8
## tot_cur_bal   4.4
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9
## SS loadings           3.20 1.67 1.34 1.17 0.94 0.87 0.79 0.74 0.56
## Proportion Var        0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05
## Cumulative Var        0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94
## Proportion Explained  0.28 0.15 0.12 0.10 0.08 0.08 0.07 0.07 0.05
## Cumulative Proportion 0.28 0.43 0.55 0.65 0.74 0.82 0.89 0.95 1.00
##
## Mean item complexity =  3.8
## Test of the hypothesis that 9 components are sufficient.
##
```

```
## The root mean square of the residuals (RMSR) is  0.03
##  with the empirical chi square  15.67  with prob <  NA
##
## Fit based upon off diagonal values = 0.98
```

There are 9 cross-loadings, and since the variables mention above with KMO less than 0.5 still has correlation with other variables, we keep those variables for further analysis.

We now proceed to FA.

Factor Analysis

PC extraction with Oblique rotation - 9 factors solution

```
#Oblique Rotation
pcModel9ob_in<-principal(data_int_val_std, 9, rotate="oblimin")
print.psych(pcModel9ob_in, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 9, rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
##              item   TC1   TC2   TC5   TC4   TC8   TC6   TC7   TC3   TC9   h2
## loan_amnt       1  1.00                                                  0.96
## total_pymnt    10  0.94                                                  0.97
## revol_util      9        0.99                                            0.90
## dti             6        0.47 -0.41  0.42                                0.87
## annual_inc      4              0.90                                      0.89
## open_acc        8                    1.00                                0.95
## loan_status     5                          0.99                          0.98
## delinq_2yrs     7                                1.01                    0.99
## tot_coll_amt   11                                      1.00              1.00
## emp_length      3                                            1.00       1.00
## sub_grade       2        0.44                                     -0.61 0.89
## tot_cur_bal    12              0.53                                0.57 0.89
##                    u2 com
## loan_amnt    0.04094 1.0
## total_pymnt  0.02994 1.1
## revol_util   0.10470 1.1
## dti          0.12636 4.3
## annual_inc   0.10579 1.1
## open_acc     0.05477 1.1
## loan_status  0.01769 1.0
## delinq_2yrs  0.00808 1.0
## tot_coll_amt 0.00169 1.0
## emp_length   0.00044 1.0
## sub_grade    0.10777 2.7
## tot_cur_bal  0.11298 2.6
##
##                        TC1  TC2  TC5  TC4  TC8  TC6  TC7  TC3  TC9
## SS loadings           2.13 1.50 1.34 1.26 1.11 1.10 1.02 1.02 0.82
## Proportion Var        0.18 0.12 0.11 0.10 0.09 0.09 0.08 0.08 0.07
## Cumulative Var        0.18 0.30 0.41 0.52 0.61 0.70 0.79 0.87 0.94
## Proportion Explained  0.19 0.13 0.12 0.11 0.10 0.10 0.09 0.09 0.07
## Cumulative Proportion 0.19 0.32 0.44 0.55 0.65 0.75 0.84 0.93 1.00
```

```
##
##  With component correlations of
##       TC1   TC2   TC5  TC4   TC8   TC6   TC7   TC3   TC9
## TC1  1.00  0.28  0.22 0.30 -0.07  0.28 -0.10  0.13  0.00
## TC2  0.28  1.00 -0.02 0.18  0.21  0.19 -0.07  0.02  0.00
## TC5  0.22 -0.02  1.00 0.06 -0.05  0.14 -0.09  0.15  0.12
## TC4  0.30  0.18  0.06 1.00  0.11  0.20  0.05  0.03  0.11
## TC8 -0.07  0.21 -0.05 0.11  1.00 -0.02 -0.06  0.15 -0.05
## TC6  0.28  0.19  0.14 0.20 -0.02  1.00 -0.08  0.06  0.01
## TC7 -0.10 -0.07 -0.09 0.05 -0.06 -0.08  1.00 -0.17  0.03
## TC3  0.13  0.02  0.15 0.03  0.15  0.06 -0.17  1.00  0.06
## TC9  0.00  0.00  0.12 0.11 -0.05  0.01  0.03  0.06  1.00
##
## Mean item complexity =  1.6
## Test of the hypothesis that 9 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.03
##  with the empirical chi square  15.67  with prob <  NA
##
## Fit based upon off diagonal values = 0.98
```

1 cross-loadings

PC extraction with Orthogonal rotation - 9 factors

```
pcModel9or_in<-principal(data_int_val_std, 9, rotate="quartimax")
print.psych(pcModel9or_in, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 9, rotate = "quartimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##              item   RC1   RC2   RC5   RC4   RC8   RC6   RC7   RC3   RC9   h2
## loan_amnt       1  0.95                                                 0.96
## total_pymnt    10  0.94                                                 0.97
## revol_util      9        0.92                                           0.90
## dti             6        0.63        0.49                               0.87
## annual_inc      4              0.90                                     0.89
## tot_cur_bal    12              0.75                           -0.41 0.89
## open_acc        8                    0.94                               0.95
## loan_status     5                          0.97                         0.98
## delinq_2yrs     7                                0.97                   0.99
## tot_coll_amt   11                                      0.99             1.00
## emp_length      3                                            0.99       1.00
## sub_grade       2  0.38  0.47                                      0.66 0.89
##                   u2 com
## loan_amnt    0.04094 1.1
## total_pymnt  0.02994 1.2
## revol_util   0.10470 1.1
## dti          0.12636 3.3
## annual_inc   0.10579 1.2
## tot_cur_bal  0.11298 2.2
## open_acc     0.05477 1.1
## loan_status  0.01769 1.1
## delinq_2yrs  0.00808 1.1
```

```
## tot_coll_amt 0.00169 1.0
## emp_length   0.00044 1.1
## sub_grade    0.10777 3.1
##
##                         RC1  RC2  RC5  RC4  RC8  RC6  RC7  RC3  RC9
## SS loadings            2.18 1.53 1.52 1.20 1.09 1.03 1.01 1.00 0.73
## Proportion Var         0.18 0.13 0.13 0.10 0.09 0.09 0.08 0.08 0.06
## Cumulative Var         0.18 0.31 0.44 0.54 0.63 0.71 0.80 0.88 0.94
## Proportion Explained   0.19 0.14 0.13 0.11 0.10 0.09 0.09 0.09 0.06
## Cumulative Proportion 0.19 0.33 0.46 0.57 0.67 0.76 0.85 0.94 1.00
##
## Mean item complexity =  1.6
## Test of the hypothesis that 9 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.03
##  with the empirical chi square  15.67  with prob <  NA
##
## Fit based upon off diagonal values = 0.98
```

1 Cross-loadings

So, we will use 9 factors - PC extraction with Oblique rotation since it has minimum cross-loadings with correlation more than 0.4 for all variables.

```
pcModel9ob_in_score <- pcModel9ob_in$scores
```

We finished the PCA and FA, next is Cluster Analysis.

Performing Cluster Analysis

```
# Defining linkage methods
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")
```

Clustering after PCA & FA

```
#Function to compute agglomerative coefficient

ac <- function(x) {
  agnes(pcModel9ob_in_score, method = x)$ac
}
```

```
# Calculating agglomerative coefficient for each clustering linkage method
sapply(m, ac)
```

```
##   average    single  complete      ward
## 0.8128179 0.7462503 0.8283755 0.8600994
```

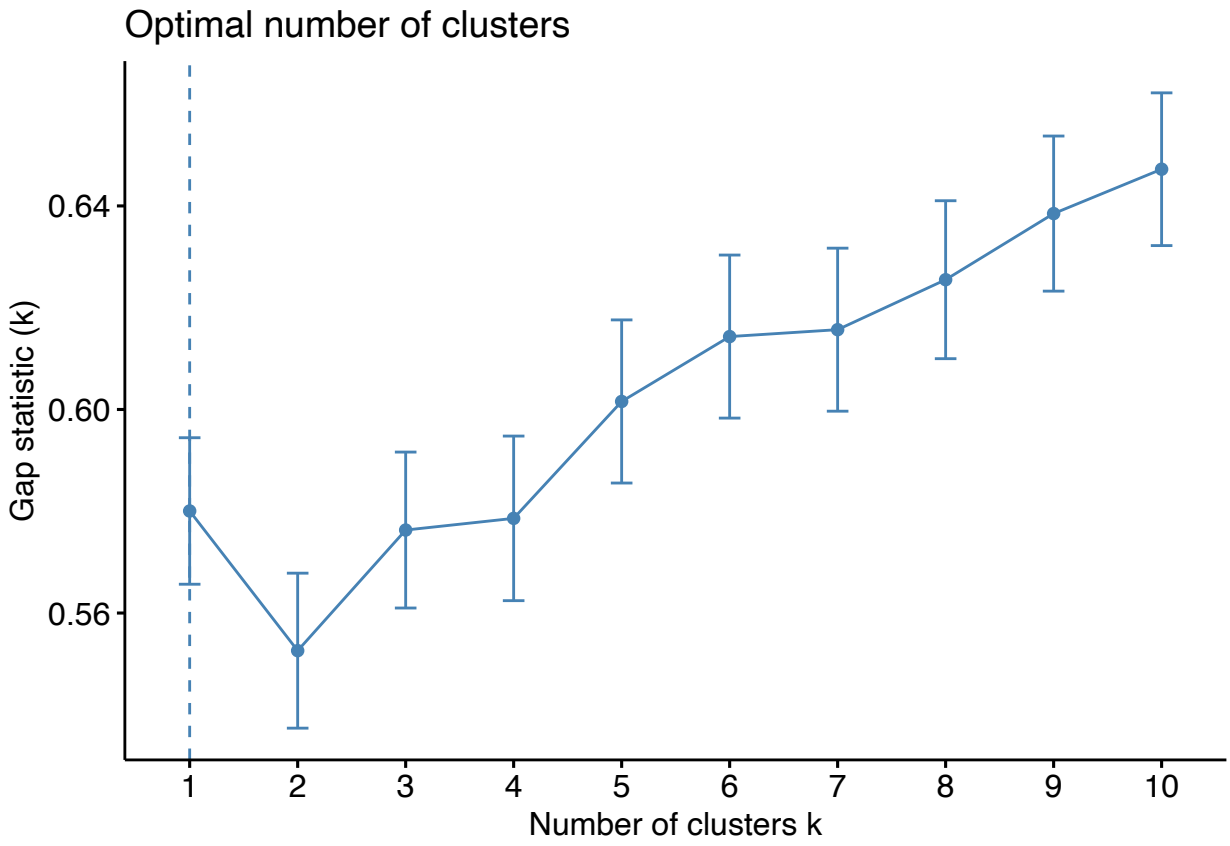
```
#Try 9 PCs
PCA_Model9wo_in<-principal(data_int_val_std, 12, rotate="none", weights=TRUE, scores=TRUE)
print(PCA_Model9wo_in)
```

```
## Principal Components Analysis
## Call: principal(r = data_int_val_std, nfactors = 12, rotate = "none",
##     scores = TRUE, weights = TRUE)
```

```
## Standardized loadings (pattern matrix) based upon correlation matrix
##                PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10  PC11
## loan_amnt     0.83 -0.15 -0.14 -0.24  0.30  0.14  0.06  0.08 -0.23  0.14  0.04
## sub_grade     0.56  0.55 -0.06 -0.23  0.02  0.03  0.14  0.31  0.32 -0.23 -0.23
## emp_length    0.24 -0.07  0.63 -0.13  0.37 -0.11  0.35 -0.47  0.17 -0.02 -0.01
## annual_inc    0.52 -0.48  0.34  0.21 -0.19  0.27  0.00  0.26  0.23 -0.16  0.28
## loan_status  -0.07  0.54  0.57  0.19  0.20  0.08  0.13  0.40 -0.31  0.11  0.05
## dti           0.50  0.55 -0.17  0.30  0.10 -0.03 -0.20 -0.32 -0.21 -0.31  0.17
## delinq_2yrs   0.46  0.00 -0.08  0.03 -0.41 -0.65  0.40  0.08 -0.11  0.02  0.09
## open_acc      0.46  0.01 -0.08  0.61  0.31 -0.32 -0.27  0.07  0.28  0.23 -0.04
## revol_util    0.40  0.58  0.05 -0.10 -0.46  0.30 -0.03 -0.25  0.16  0.31  0.08
## total_pymnt   0.81 -0.25 -0.34 -0.29  0.19  0.10  0.02  0.03 -0.08  0.07  0.02
## tot_coll_amt -0.17 -0.01 -0.44  0.55  0.06  0.35  0.59 -0.06  0.01  0.01 -0.04
## tot_cur_bal   0.60 -0.33  0.34  0.32 -0.33  0.13 -0.11 -0.12 -0.24 -0.05 -0.33
##                PC12 h2       u2 com
## loan_amnt    -0.14  1  1.1e-16 2.0
## sub_grade    -0.01  1  2.3e-15 4.6
## emp_length    0.01  1  1.4e-15 4.1
## annual_inc    0.00  1  1.1e-15 6.1
## loan_status   0.05  1 -2.0e-15 4.3
## dti           0.00  1  7.8e-16 5.3
## delinq_2yrs   0.00  1  2.2e-16 3.6
## open_acc      0.00  1  6.7e-16 4.6
## revol_util    0.00  1  1.0e-15 4.8
## total_pymnt   0.16  1  4.1e-15 2.2
## tot_coll_amt  0.00  1  1.1e-15 3.8
## tot_cur_bal   0.01  1  1.0e-15 5.2
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11
## SS loadings           3.20 1.67 1.34 1.17 0.94 0.87 0.79 0.74 0.56 0.37 0.29
## Proportion Var        0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Var        0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
## Proportion Explained  0.27 0.14 0.11 0.10 0.08 0.07 0.07 0.06 0.05 0.03 0.02
## Cumulative Proportion 0.27 0.41 0.52 0.62 0.69 0.77 0.83 0.89 0.94 0.97 1.00
##                       PC12
## SS loadings           0.05
## Proportion Var        0.00
## Cumulative Var        1.00
## Proportion Explained  0.00
## Cumulative Proportion 1.00
##
## Mean item complexity =  4.2
## Test of the hypothesis that 12 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

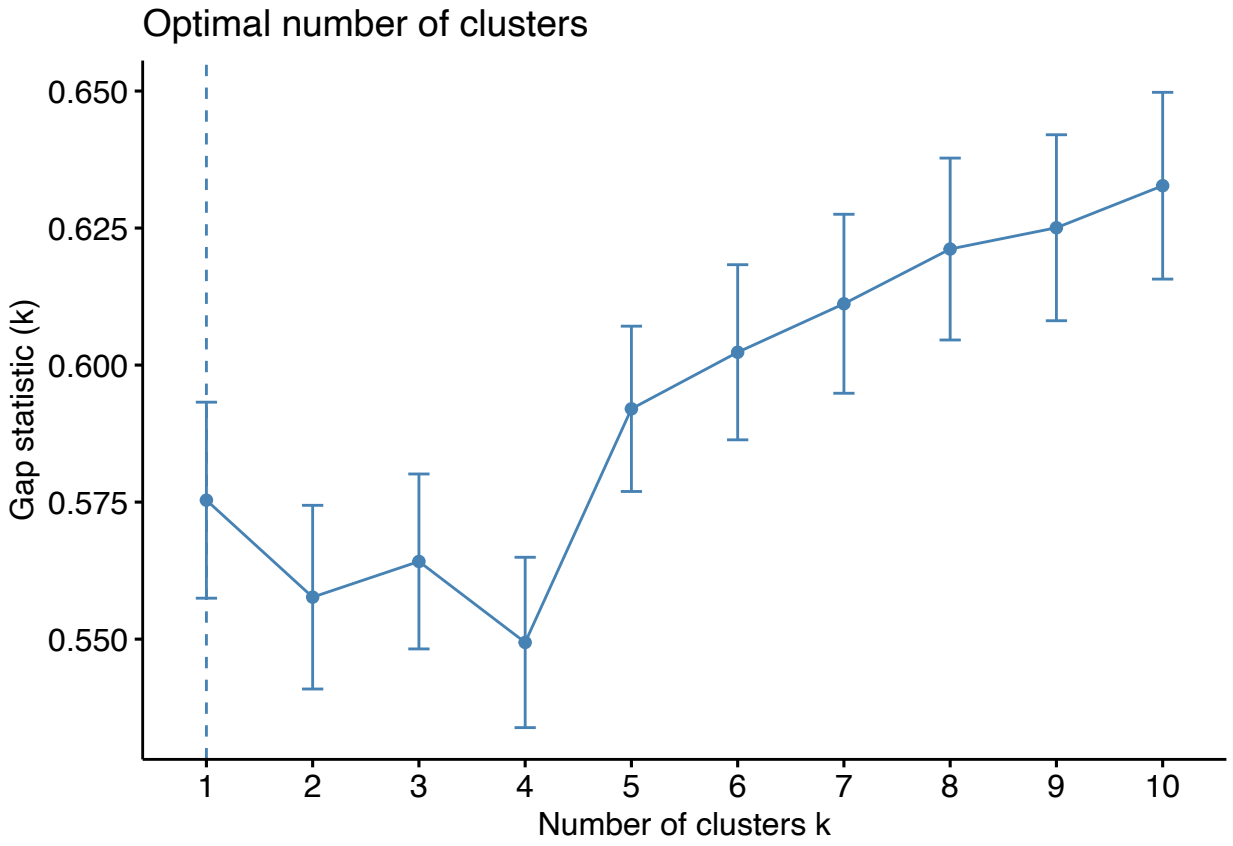
Calculating gap statistic for each number of clusters (up to 10 clusters)

```
in_gap_stat_h_9ob_score <- clusGap(pcModel9ob_in_score, FUN = hcut, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(in_gap_stat_h_9ob_score)
```




```
# From the plot we can see that the gap statistic is high at k = 3 and 7 clusters. Thus, we'll choose t
```

```
# checking k means
# need to check different values for nstart, kmax and B and explain them

in_gap_stat_k_9ob_score <- clusGap(pcModel9ob_in_score, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
fviz_gap_stat(in_gap_stat_k_9ob_score)
```




```
#From the plot we can see that the gap statistic is high at k = 4 and 7 clusters. Thus, we'll choose t
```

K-means clustering

```
set.seed(55)
k_cl_val <- kmeans(pcModel9ob_in_score,3,nstart=25)
k_cl_val
```

```
## K-means clustering with 3 clusters of sizes 1, 63, 36
##
## Cluster means:
##          TC1        TC2        TC5        TC4        TC8        TC6         TC7
## 1 -0.7719427 -0.5587162 -0.6805182  0.6498600 -0.5026699 -0.4578740  9.40613320
## 2 -0.3020385 -0.2688060 -0.1894601 -0.4281283 -0.2511645 -0.2944260 -0.06878908
## 3  0.5500102  0.4859303  0.3504585  0.7311728  0.4535010  0.5279641 -0.14090059
##         TC3        TC9
## 1 -1.341214  0.3221110
## 2 -0.377664 -0.1117065
## 3  0.698168  0.1865388
##
## Clustering vector:
##   [1] 2 2 3 2 3 3 3 3 2 2 1 2 2 2 2 2 2 2 3 2 3 3 2 2 3 2 3 2 2 2 2 3 3 2 2 2 2
##  [38] 2 2 2 3 2 2 3 2 3 3 3 3 3 2 2 3 2 3 3 2 3 2 3 3 2 2 2 3 3 2 2 2 3 2 2 3 3
##  [75] 3 2 2 2 2 2 2 2 2 2 2 2 2 3 3 2 3 3 2 3 2 2 2 2 2 2
##
## Within cluster sum of squares by cluster:
```

```
## [1]   0.0000 354.6772 320.0626
##  (between_SS / total_SS =  24.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Finding distance matrix

```
in_distance_mat_9ob_score <- dist(pcModel9ob_in_score, method = "manhattan")
#print(in_distance_mat_3or_score)
```

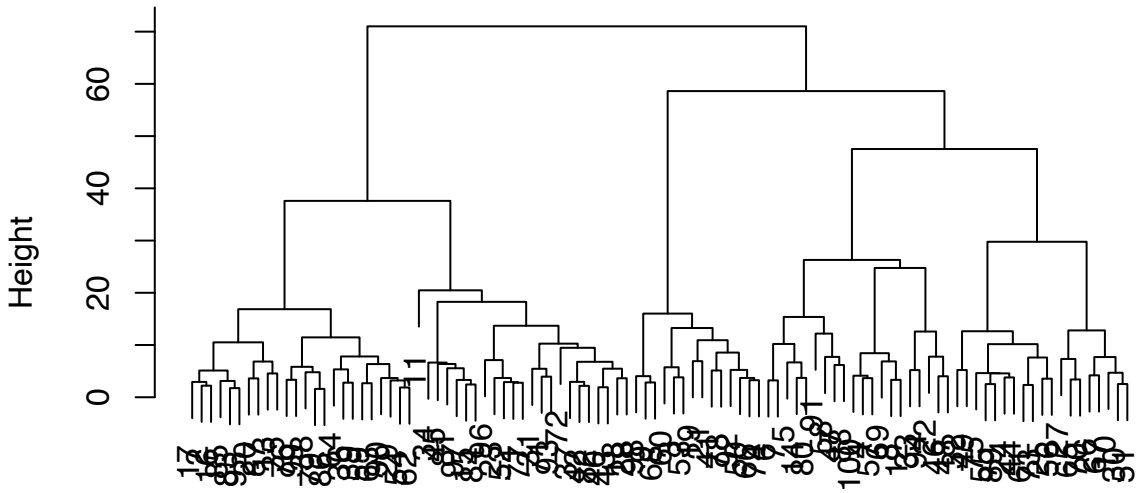
```
#Fitting Hierarchical clustering Model to dataset

set.seed(240)  # Setting seed
Hierar_cl_9ob_score_in <- hclust(in_distance_mat_9ob_score, method = "ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
Hierar_cl_9ob_score_in
```

```
##
## Call:
## hclust(d = in_distance_mat_9ob_score, method = "ward")
##
## Cluster method   : ward.D
## Distance         : manhattan
## Number of objects: 100
```

```
#Plotting dendrogram
plot(Hierar_cl_9ob_score_in)
```



##

Choosing no. of clusters

```
#Cutting tree by no. of clusters
fit_in <- cutree(Hierar_cl_9ob_score_in, k = 3 )
fit_in
```

```
##   [1] 1 1 2 3 3 3 3 3 2 3 1 3 1 3 1 1 1 1 2 1 2 3 1 3 3 3 3 1 1 3 1 3 3 1 1 1 1
##  [38] 1 1 1 2 1 1 3 1 3 3 3 3 2 3 1 2 1 3 3 3 3 1 2 3 1 1 1 3 2 3 2 3 3 1 1 3 2
##  [75] 3 1 1 2 1 3 1 1 1 1 1 1 1 3 3 1 3 2 1 3 1 1 1 2 1 3
```

```
#Find number of observations in each cluster
table(fit_in)
```

```
## fit_in
##  1  2  3
## 47 14 39
```

```
final_data_in <-cbind(data_int_val_std, cluster = fit_in)
```

We can then append the cluster labels of each child back to the original dataset:

Append cluster labels to original data

Find mean values for each cluster

```
hcentres_in_val<-aggregate(x=final_data_in, by=list(cluster=fit_in), FUN="mean")
print(hcentres_in_val)
```

```
##   cluster  loan_amnt  sub_grade emp_length annual_inc loan_status        dti
## 1       1 -0.4387716 -0.4235572 -0.4647942 -0.4177629  -0.3391719 -0.3277248
## 2       2 -0.0378913  0.6084928  0.4637340 -0.1319382   2.2606441  0.3320469
## 3       3  0.5423780  0.2920074  0.3936680  0.5508203  -0.4027676  0.2757540
##   delinq_2yrs    open_acc revol_util total_pymnt tot_coll_amt tot_cur_bal
```

```
## 1  -0.4087576 -0.27357050 -0.2745548  -0.3093541    0.1513562 -0.41373898
## 2  -0.1014211  0.08813541  0.3618098  -0.6937477   -0.1342216 -0.08290361
## 3   0.5290128  0.29804917  0.2009933   0.6218489   -0.1342216  0.52836878
##   cluster
## 1       1
## 2       2
## 3       3
```

```
final_data_in_val <-cbind(pcModel9ob_in_score, cluster = fit_in)
```

```
#Calculate mean for each cluster
hcentres_mean_in_val<-aggregate(x=final_data_in_val, by=list(cluster=fit_in), FUN="mean")
print(hcentres_mean_in_val)
```

```
##   cluster        TC1        TC2        TC5        TC4        TC8         TC6
## 1       1 -0.4152204 -0.3530810 -0.3289576 -0.3252452 -0.4055501 -0.43744148
## 2       2 -0.2674507  0.3625376 -0.1481554  0.1352274  2.2445265 -0.07797605
## 3       3  0.5964018  0.2953662  0.4496201  0.3434190 -0.3169876  0.55516446
##          TC7        TC3         TC9 cluster
## 1  0.1590465 -0.4617293 -0.02646895       1
## 2 -0.1352291  0.4715041 -0.26901348       2
## 3 -0.1431276  0.3871851  0.12846742       3
```

```
#Set Member_id back in the dataframe
final_data_in_val <- cbind(ID_int_val, final_data_in_val)
```

```
#Filter the model only the ID that provide in validation model
filtered_id <- unique(final_data_in_val$Member_ID)
final_data_pc9ob <- as.data.frame(final_data_pc9ob)
filtered_model <- final_data_pc9ob %>%
  filter(Member_ID %in% final_data_in_val$Member_ID)
```

```
#Count observations in each cluster in the model after filter 100 observations
table(filtered_model$cluster)
```

```
##
##  1  2  3
## 72 12 16
```

```
Combined_model <- merge(filtered_model, final_data_in_val, by = "Member_ID", suffixes = c("_model", "_va
```

```
counts_cluster <- table(Combined_model$cluster_model, Combined_model$cluster_validation_model)
print(counts_cluster)
```

```
##
##      1  2  3
##   1 45  0 27
##   2  0  0 12
##   3  2 14  0
```

```
filtered_model$cluster <- ifelse(filtered_model$cluster == 1, 1,
                ifelse(filtered_model$cluster == 2, 3,
                       ifelse(filtered_model$cluster == 3, 2, filtered_model$cluster)))
```

```
#Count observations in each cluster in the model after filter 100 observations
table(filtered_model$cluster)
```

```
##
##  1  2  3
```

```
## 72 16 12
```

```
# Merge the two dataframes based on the "ID" column
Combined_model_adj_cluster <- merge(filtered_model, final_data_in_val, by = "Member_ID", suffixes = c("

# Count the number of IDs with the same and different groups
same_cluster_count <- sum(Combined_model_adj_cluster$cluster_model == Combined_model_adj_cluster$cluste
different_cluster_count <- sum(Combined_model_adj_cluster$cluster_model != Combined_model_adj_cluster$cl

#result in ratio
calculate_ratio <- function(different_cluster_count, same_cluster_count) {
  result <- different_cluster_count / (different_cluster_count + same_cluster_count)
  return(result)
}
ratio_result <- calculate_ratio(different_cluster_count, same_cluster_count)

# Output the results
print(paste("Number of Member_ID with the same cluster:", same_cluster_count))
```

```
## [1] "Number of Member_ID with the same cluster: 71"
```

```
print(paste("Number of Member_ID with different cluster:", different_cluster_count))
```

```
## [1] "Number of Member_ID with different cluster: 29"
```

```
print(paste("Inaccuracy Rate:", ratio_result))
```

```
## [1] "Inaccuracy Rate: 0.29"
```

Interal validation using clValid package

```
data_int_validation <- as.data.frame(data_no_outliers_std)

internal_val <- clValid(data_int_validation, 3:8, clMethods=c("hierarchical","kmeans"), validation="int
```

```
## Warning in clValid(data_int_validation, 3:8, clMethods = c("hierarchical", :
## rownames for data not specified, using 1:nrow(data)
```

```
summary(internal_val)
```

```
##
## Clustering Methods:
##  hierarchical kmeans
##
## Cluster sizes:
##  3 4 5 6 7 8
##
## Validation Measures:
##                                  3        4        5        6        7        8
##
## hierarchical Connectivity   12.9925  49.3694  52.0984  53.7734  53.9734  54.8845
##              Dunn            0.3496   0.2192   0.2192   0.2192   0.2192   0.2192
##              Silhouette      0.4170   0.3232   0.2669   0.2532   0.2426   0.2434
## kmeans       Connectivity  159.4353 170.5238 162.9921 217.9540 281.6782 279.4948
##              Dunn            0.0986   0.0986   0.1053   0.1078   0.0911   0.0911
##              Silhouette      0.2074   0.2043   0.1879   0.1709   0.1279   0.1305
```

```
##
## Optimal Scores:
##
##              Score   Method       Clusters
## Connectivity 12.9925 hierarchical 3
## Dunn          0.3496 hierarchical 3
## Silhouette    0.4170 hierarchical 3
```