Contents Page

Executive Summary

We analysed Lending Club's loan dataset, through selecting relevant features, then using K-means and hierarchical clustering algorithms to classify borrowers into groups based on similar combinations of features. Four distinct customer segments are revealed, each with unique credit profiles. We then provide recommendations in customising loan products and business strategies for each target group. The 45% within-sample accuracy rate means that our customer analysis may not be generalised to the population. We advise employing a larger sample size in the future to enhance robustness and representativeness.

1. Introduction

Lending Club is a US-based peer-to-peer lending company. The dataset covers loans issued from 2007 to 2015, encompassing borrower information such as credit scores, address details, and loan status.

Utilising cluster analysis on its extensive loan dataset, the company aims to uncover patterns and gain a deeper understanding of customer behaviour and delineate distinct borrower segments. Traditionally, clustering concentrates only on either quantitative or qualitative data at a time; however, since credit applicants are characterised by mixed personal features, a cluster analysis specific for mixed data can discover particularly informative patterns (Caruso et al., 2021).

2. Data Understanding and Preparation

In preparation for the analysis, the dataset has been meticulously refined to prioritise variables directly revealing customer characteristics, ensuring cluster analysis yields the most actionable insights into borrower behaviour.

The dataset includes 50,000 observations and 54 variables about customer's loans, each entry within the dataset is uniquely identified by '*id*' and *'member_id'*.

a. Single Variate Outliers:

   The distribution of data revealed that there are several outliers within *'annual_inc'* variable. These outliers are retained because they could potentially form a distinct cluster of high-earning customers.

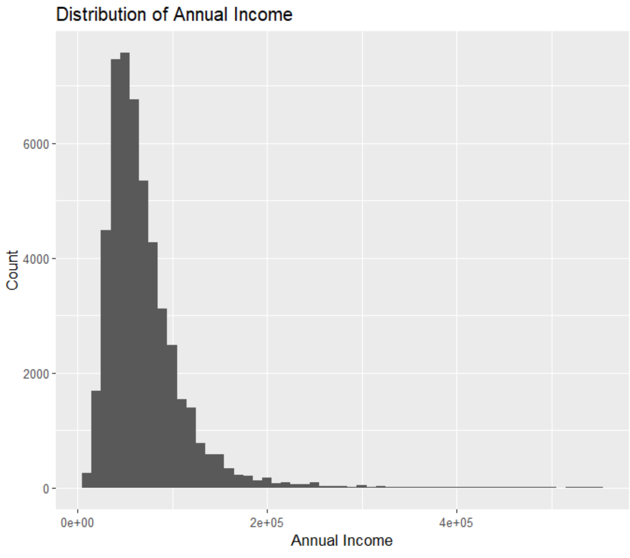


Figure 2.1: Annual income histogram

b. Missing Values:

Either treatments was implemented for missing values: dropping columns or imputation with mean. Specifically, the columns *'mths_since_last_record'* and *'mths_since_last_major_derog'* were dropped due to the high proportion of missing values (>85%). Meanwhile, missing values within the columns *'mths_since_last_delinq', 'revol_util', 'emp_length', 'tot_coll_amt', 'tot_cur_bal',* and *'total_credit_rv'* were imputed using their respective mean values.

c. Feature Selection:

After analysing the data, several variables are removed from the dataset (Appendix 2.1) because they are irrelevant or are nominal categorical variables. Out of the total 54 variables, a subset of 17 variables that best provide information pertaining to customer borrowing patterns were identified based on the information in the data dictionary (Appendix 2.2). A new numeric variable, *'months_since_last_credit_pull'*, is calculated based on the time interval between *'issue_d'* to *'last_credit_pull_d'* .

d. Integer Encoding:

Most clustering approaches are exclusively limited to a single data type, so it is usual to convert mixed data types into a single data type, such as transforming categorical variables into numerical variables (Caruso et. al, 2021).

Two ordinal categorical variables *'sub_grade'* and *'loan_status'* were transformed into integer representations. We assume that the difference between each *'loan_status'* is the same, although in reality, it might not be equidistant.

e. Correlation and Multicollinearity:

Variables with high correlation (more than 0.6), including *'installment'*, *'mths_since_last_delinq'*, *'total_pymnt'*, *'open_acc'*, and *'loan_amnt'*, were dropped from further analysis. (Correlation plot in Appendix 2.3)

f. Sampling:

We conducted random sampling for 600 data points from the dataset, so that we have a sufficient sample size of nearly 500 to work with, after the removal of multivariate outliers.

g. Normalisation:

Sample data is normalised to facilitate further clustering analysis.

h. Multivariate Outliers:

By using Mahalanobi's distance method, 58 records were identified as outliers. We analyse the sample including outliers and excluding outliers to see their impact to the clusters.

We decide to remove the outliers for our subsequent analysis, because they only represent small 9.67% of the sample size and is a non-representative group of observations as compared to the others. As cluster analysis is sensitive to outliers, we decide to remove outliers to avoid skewing our cluster analysis. (See Appendix 2.4)

3. Modelling

3.1. Principle Component Analysis (PCA) and Factor Analysis (FA)

PCA and FA are performed on 9 variables which have correlations more than 0.3. (See Appendix 2.5 & 2.6). Though, after considering the difficulty to interpret PCA and FA, we did not use them in cluster analysis.

3.2 Determining Suitable Number of Clusters (k)

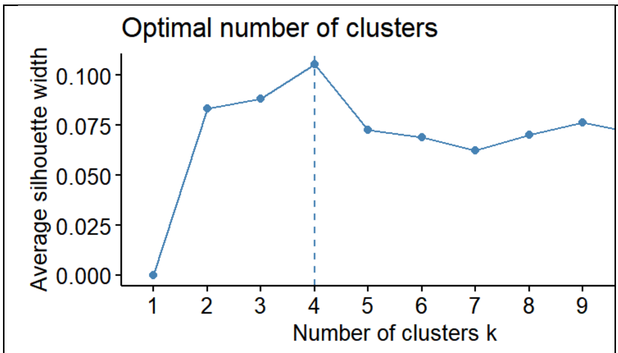


Figure 3.1: Hierarchical Clustering Silhouette plot

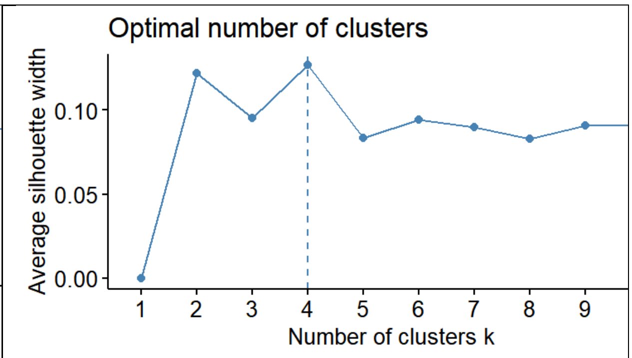


Figure 3.2: K-means Clustering Silhouette plot

Gap statistic and elbow plots are common tools to determine k. (University of Cincinnati, n.d.) However, these methods are not very useful in our case. The gap statistic plot displays a consistent upward trend without a peak that higher than its neighbours that indicates optimal k (Appendix 3). Similarly, the elbow plots for both clustering methods does not have a sharp turn (Appendix 4).

Another method to determine optimal k is silhouette plots. Silhouette plots in figures 3.1 and 3.2 indicate that optimal k = 4, where silhouette width is at a higher peak than its neighbours, so there are better defined and well-separated clusters when k = 4.

3.3 Cluster Analysis Comparison

To determine the most suitable clustering algorithm for our analysis, we compare Hierarchical and Non-hierarchical models, then analysed which clusters have the highest and lowest centroid values for each variable, in order to discover the unique characteristics of customers in each cluster.

3.3.1 Hierarchical Clustering

We opted for an agglomerative "bottom-up" hierarchical approach rather than a divisive method. Our preliminary analysis did not suggest the presence of one or several large clusters that naturally should be subdivided.

In selecting the specific linkage criterion for our agglomerative clustering, we settled on Ward's method because it has the highest agglomerative coefficient, scoring at 0.934, as compared to other linkage criteria, indicating a robust clustering structure.

Cutting the dendrogram at k = 4, we obtained 4 cluster sizes of 169, 281, 24, and 68.

3.3.2 Non-hierarchical Clustering

We ran k-means algorithm for k = 4. The clusters obtained are of sizes 134, 219, 120 and 69 respectively.

3.4 Evaluation of Best Model

In order to select the best model between hierarchical and k-means clustering, we analysed the centroid values over the 17 variables we used in cluster analysis.

We segment the variables into those traditionally used for creditworthiness analysis (such as annual income and credit history-related variables) (Xue, 2022) – highlighted in red (bad) or green (good), and neutral variables (such as funded loan amount) that do not provide information on creditworthiness – highlighted in orange (high) or blue (low). For instance, high annual income usually suggests a good customer who has better financial ability to repay loan, whereas the magnitude of loan amount does not necessarily imply a customer's creditworthiness.

For hierarchical clustering, a deeper dive into these clusters' characteristics revealed clear and distinct patterns. Cluster 1 demonstrated more creditworthy customers, exhibiting 7 favourable traits, and only one negative trait. Cluster 2 scored highest on 6 of the neutral variables, while cluster 3 scored lowest on 5 of the neutral variables. Conversely, the fourth cluster consists of less promising customers, with 4 unfavourable traits and only 1 positive trait. Hence, the clusters produced from hierarchical clustering are more interpretable.

		Neutral			Neutral							Neutral		Neutral	Neutral	Neutral	Neutral	Neutral	Better	Worse	Neutra	Neutral		Sample size = 542
Hierarchical	cluster	funded_amnt	int_rate	sub_grade	emp_length	annual_inc	loan_status	dti	delinq_2yrs	inq_last_6mths	pub_rec	revol_bal	revol_util	total_acc	tot_coll_amt	tot_cur_bal	total_credit_rv	months_since_last_credit_pull	Green	Red	High	Low	Cluster size	Percentage
k =4	1	-0.52999518	-0.7465802	0.71516716	-0.19146655	-0.2962732	0.3676746	-0.2805815	-0.3958163	-0.34412426	-0.2365206	-0.3459943	-0.4300851	-0.35707522	-0.0843575	-0.24960107	-0.16197962	0.23366964	7	1	1	2	169	31.18%
	2	0.21959238	0.27831	-0.23645478	0.13878676	0.0263141	0.349575	0.2259107	0.1721685	0.06637375	-0.2365206	0.0882827	0.330619	0.13239957	-0.11819165	0.03173603	-0.03429777	0.03997812	2	3	6	0	281	51.85%
	3	-0.3964019	0.1143276	-0.06745994	-0.2928007	-0.1239197	0.1928647	-0.468285	-0.272681	-0.15673128	3.5989479	-0.4785661	-0.1298586	-0.42205461	-0.14303035	-0.05577798	-0.46924885	0.08426041	1	1	0	5	24	4.43%
	4	0.03889089	0.4236321	-0.43772409	0.03573127	-0.2347818	-2.3023762	0.1471515	-0.2642389	0.13791212	-0.2365206	-0.1005356	0.1823019	-0.06398478	-0.05999734	-0.10508874	-0.11487195	-0.58848504	1	4	1	1	68	12.55%

Figure 3.3: Hierarchical Clustering

On the other hand, k-means clustering produced clusters that do not have as evident and well-defined patterns. This can be seen from the small difference in count between favourable and unfavourable variables, and, between highest and lowest neutral variables.

		Neutral			Neutral							Neutral		Neutral	Neutral	Neutral	Neutral	Neutral	Better	Worse	Neutra	Neutral		Sample size = 542
Non- Hierarchical		funded_amnt	int_rate	sub_grade	emp_length	annual_inc	loan_status	dti	delinq_2yrs	inq_last_6mths	pub_rec	revol_bal	revol_util	total_acc	tot_coll_amt	tot_cur_bal	total_credit_rv	months_since_last_credit_pull	Green	Red	High	Low	Cluster size	Percentage
k =4	**1**	-0.49066071	-0.9558863	0.88985442	-0.39538442	-0.1335981	0.2244792	-0.5244323	-0.12059655	-0.12296858	-0.007537363	-0.4564623	-1.0171181	-0.13838008	-0.08671889	-0.1135136	-0.05733001	-0.1356836	5	0	0	3	134	24.72%
	2	-0.369776	0.1564132	-0.09292751	-0.12845664	-0.3920395	0.408194	0.1401578	-0.08591319	-0.16558491	0.00866921	-0.25962155	0.5209983	-0.48711955	-0.08248795	-0.3623109	-0.38380484	0.2774444	2	3	2	3	219	40.41%
	3	0.90475571	0.3301468	-0.31711484	0.65820435	0.4730026	0.3991074	0.2770747	-0.01435163	0.09371561	-0.236520554	0.58350695	0.2443156	0.70646071	-0.15547138	0.5364614	0.38431796	0.1262042	2	1	6	1	120	22.14%
	4	0.04965222	0.5478342	-0.547832	0.06047444	-0.2118357	-2.3033741	0.3066738	-0.09172565	0.11022473	-0.125347556	-0.05775889	0.3243015	0.04094377	-0.09621828	-0.1676508	-0.15171259	-0.6520861	0	5	0	1	69	12.73%

Figure 3.4: K-means Clustering

4. Internal Validation

To validate the effectiveness of our Hierarchical Clustering Model, we randomly sampled 200 data points from the original sample of size 542. Then, we reiterate the same steps taken in the hierarchical model to ensure consistency in cluster formation.

During the validation process, we identified 4 clusters of sizes 95, 48, 13, and 44 from the validation sample. A key challenge was matching these clusters to those identified in the original sample, given that cluster labels are not inherently consistent across different samples.

Therefore, we compared clusters from the original and validation samples by finding similar patterns in the same variables. For instance, if Cluster 1 in the original sample had the highest variable means in ten variables and the lowest in one variable, we sought the cluster with a similar pattern/profile in the validation sample. Hence, based on their attribute profiles, clusters from both samples are matched as the same.

Subsequently, we calculated an accuracy rate to assess how consistently the data points are assigned to the same cluster in both validation and original samples. The resulting accuracy rate of approximately 45% suggests some instability in the model, which may be attributed to noise within the dataset.

5. Interpretation and Recommendations

These are the interpretations made on our findings for k = 4.

Cluster 1:

This cluster possesses individuals with notable characteristics that a lending institution usually considers. Considering their employment length, the people in this cluster could be in their late 20's and mid 30's. Despite earning the least among all clusters, their noteworthy public record and lower interest rates display excellent quality of creditworthiness. In addition, a remarkably low track record of delinquency over the past 2 years demonstrates consistent timely payments and minimal defaults. The frequency of recent credit inquiries and a very low revolving utilisation depict that they are not too dependent on the loan and utilise them responsibly. The customers in this category can be targeted for more customised loan products, providing them with a favourable interest rate thus rewarding their responsible financial behaviour.

Cluster 2:

This cluster comprises of people who earn the most compared to the rest of the clusters. Based on their employment length and annual income, they are probably business owners or managers. Despite maintaining good bank balance, they have a high debt-to-income ratio (DTI) and often miss payment deadlines, leading to delinquencies in the past 2 years. In addition, they carry significant revolving credit balances and utilisation, showing a heavy reliance on credit, which raises concerns about their ability to repay loans. Moreover, they tend to request the largest loan amounts and face the highest interest rates compared to other groups which could contribute to difficulties in repaying loans. The lending institutions should assess more on the customers in this segment before they decide to offer them loans.

Cluster 3:

This group consists of individuals with less work experience compared to the rest of the groups indicating that they are likely younger and in the early stages of their careers. Factors like DTI and revolving utilisation position them as financially responsible customers. However, they have a higher incidence of public records making them a riskier candidate for loan approval. Also, they possess fewer credit lines which leads to limited availability of credit history for lenders to assess. As a result, the company should conduct more thorough background checks on these customers before making decisions.

Cluster 4:

Individuals within this group are encountering the highest interest rates when seeking loans. It’s worth noting that they have made the greatest number of loan inquiries in the past six months compared to their counterparts, suggesting an urgent need for financial assistance. Nevertheless, their public records reflect positively, and they maintain a reasonably decent DTI, along with a favourable delinquency record over the last two years. These factors collectively position them as suitable candidates for loan eligibility considering they are more financially responsible and are likely to repay loans.

Any new customer belonging to clusters 1 and 4 shall be provided loans with favourable interest rates based on their credit history while those belonging to clusters 2 and 3 shall be thoroughly examined before offering them a credit. The company should consider increasing the interest rates for the latter, provided they have poorer creditworthiness.

6. Conclusion

The use of hierarchical clustering helped us to achieve the business objective of customer segmentation by giving insights on the main customer profiles that we are serving. Based on these insights, we gave recommendations on business and marketing strategies catering to each profile to optimise risk and return. The 45% accuracy rate for our sample means that our customer analysis may not be generalised to the population, so we may consider using a larger sample in the future.

References

Caruso, G., Gattone, S. A., Fortuna, F., & Di Battista, T. (2021). Cluster Analysis for mixed data: An application to credit risk evaluation. Socio-Economic Planning Sciences, 73, 100850. https://doi.org/10.1016/j.seps.2020.100850

University of Cincinnati. (no date) K-means Cluster Analysis · UC Business Analytics R Programming Guide. Available at: https://uc-r.github.io/kmeans_clustering

Xue, R. (2022). Segmentation for Financial Loan Company's Customers Data Based on K-means. 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), 291–296. https://doi.org/10.1109/IWECAI55315.2022.00062

Yoshino, N., Taghizadeh-Hesary, F., Charoensivakorn, P., & Niraula, B. (2016). Small and Medium Sized Enterprise (SME) Credit Risk Analysis Using Bank Lending Data: An Analysis of Thai SMEs. Journal of Comparative Asian Development, 15(3), 383–406. https://doi.org/10.1080/15339114.2016.1233821

Zuo, Y. (2015). *CLUSTERING ANALYSIS TO SUPPORT LENDER'S DECISION-MAKING IN P2P LENDING - Bondora case study: borrower's creditworthiness classification*. https://doi.org/10.13140/RG.2.2.11598.48965

Appendix

Appendix 1 – Members' Contributions & meeting of the group

No	Time of meeting	Agenda
1	Thursday, 15 February 2024	Introduction Decide the routine meeting schedule Decide the workplan Identify the potential variable Sampling Divide the task: 1. 5583010: descriptive analysis (histogram) 2. 5506618: Data normalization 3. 2182698: Integer encoding 4. 5590002: Multicollinearity 5. 5504008 & 5585530: Mahalanobi's distance
2	Sunday, 18 February 2024	Checking the progress
3	Thursday, 22 February 2024	Checking the progress
4	Thursday, 29 February 2024	Checking the progress Divide the task: 1. 2182698 & 5506618: PCA & FA 2. 5504008 & 5585530: Hierarchical clustering 3. 5583010 & 5590002: K-Means clustering
5	Thursday, 7 March 2024	Checking PCA & FA
6	Thursday, 14 March 2024	Checking clustering Divide the task: 1. 5506618: write the report about introduction 2. 2182698: write the report about data preparation 3. 5583010 & 5504008: do clustering & validation again 4. 5590002 & 5585530: do interpretation and suggestion
7	Saturday, 16 March 2024	Discuss about interpretation and write the report
8	Sunday, 17 March, 2024	Run through and finalise the report

Appendix 2.1 - Variables Removed

No	Variables Name	Reason to delete
1	id	Identification for loan listing, it is not useful
2	member_id	Identification for borrower, it is not useful
3	loan_amnt	Highly correlated with funded_amnt (Appendix 1.3)
4	funded_amnt_inv	Highly correlated with funded_amnt (Appendix 1.3)
5	term	Just consist of 2 nominal categories, 36 and 60 months
6	installment	High correlation with funded_amnt (Appendix 1.3)
7	grade	Highly correlated with sub_grade, and sub_grade captures more information than grade
8	emp_title	very messy and highly unstructured textual data

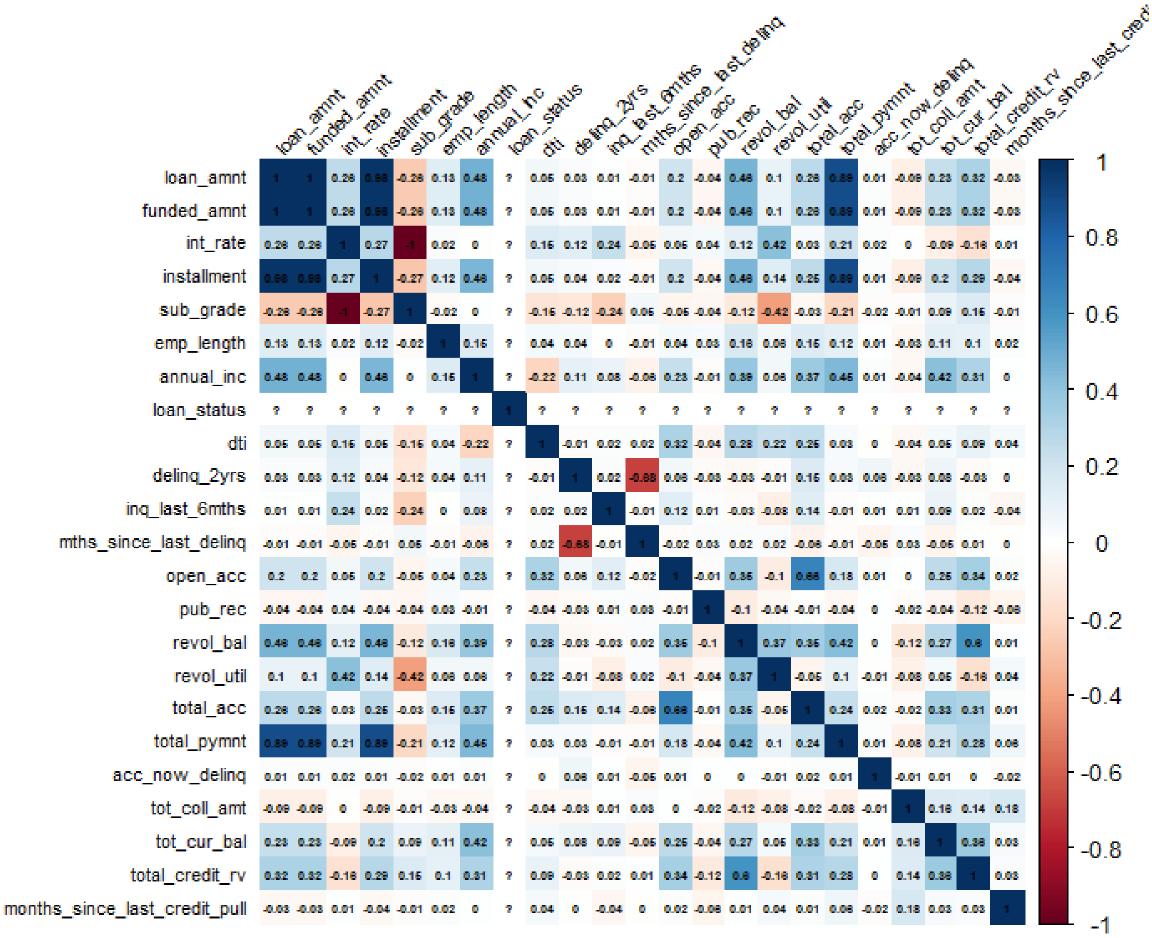
No	Variables Name	Reason to delete
9	home_ownership	Nominal categorical data
10	verification_status	Nominal categorical data
11	issue_d	Date type, not useful for clustering
12	pymnt_plan	Not useful because only 3 customer have payment plan (‘yes’), the rest are ‘no’; nominal categorical data
13	desc	Textual unstructured data provided by lender
14	purpose	Nominal categorical data
15	title	Textual unstructured data provided by borrower, not useful
16	zip_code	Nominal categorical data
17	addr_state	Nominal categorical data
18	earliest_cr_line	Not useful, irrelevant to creditworthiness
19	mths_since_last_delinq	High correlation with delinq_2_yrs (Appendix 1.3)
20	mths_since_last_record	percentage of missing data is too high (94.9%)
21	open_acc	High correlation with total_acc (Appendix 1.3)
22	total_pymnt	High correlation with loan_amnt, funded_amnt and installment (Appendix 1.3)
23	total_pymnt_inv	High correlation with total payment
24	total_rec_prncp	A component used to calculate total payment
25	total_rec_int	A component used to calculate total payment
26	total_rec_late_fee	A component used to calculate total payment
27	recoveries	A component used to calculate total payment
28	collection_recovery_fee	Not useful because related with recoveries
29	last_pymnt_d	most of these are in the future: 2014 or 2015 (current dataset is 2012-2023 data)
30	last_pymnt_amnt	most of these are in the future: 2014 or 2015 (current dataset is 2012-2023 data)
31	next_pymnt_d	Not useful because this is a date variable
32	last_credit_pull_d	We already made the calculation variable months_since_last_credit_pull (interval between issue_d to last_credit_pull_d)
33	collections_12_mths_ex_med	Majority are 0
34	mths_since_last_major_derog	Percentage of missing data is high (85.8%)
35	policy_code	Just have one value, not useful
36	acc_now_delinq	Majority are 0

Appendix 2.2 - Variables Used

No	Variables Name	Description
1	funded_amnt	The total amount committed to that loan at that point in time.
2	int_rate	Interest Rate on the loan
3	sub_grade	The subgrade of the customer

No	Variables Name	Description
4	emp_length	Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.
5	annual_inc	The self-reported annual income provided by the borrower during registration.
6	loan_status	Current status of the loan, consist of "Charged Off"=1,"Late (31-120 days)"=2,"Late (16-30 days)"=3,"In Grace Period"=4, "Fully Paid"=5,"Current"=6
7	dti	A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower’s self-reported monthly income.
8	delinq_2yrs	The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years
9	inq_last_6mths	The number of inquiries in past 6 months (excluding auto and mortgage inquiries)
10	pub_rec	Number of derogatory public records
11	revol_bal	Total credit revolving balance
12	revol_util	Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit.
13	total_acc	The total number of credit lines currently in the borrower's credit file
14	tot_coll_amt	Total collection amounts ever owed
15	tot_cur_bal	Total current balance of all accounts
16	total_credit_rv	Total revolving credit
17	months_since_last_credit_pull	New calculated variable (time interval, in months, between issue_d to last_credit_pull_d)

Appendix 2.3 - Correlation plot



Appendix 2.4 – Cluster visualization with VS without multivariate outliers

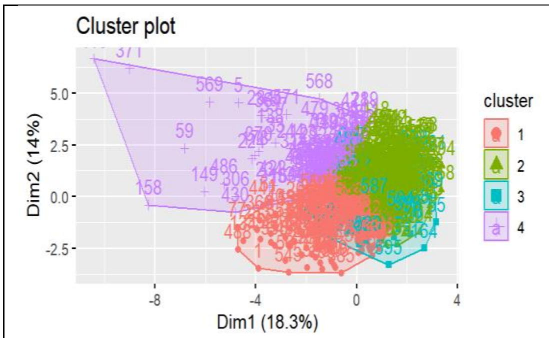


Figure 1: K-means Clustering visualisation with outliers

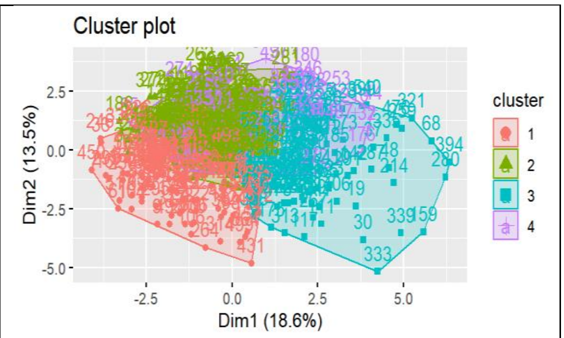
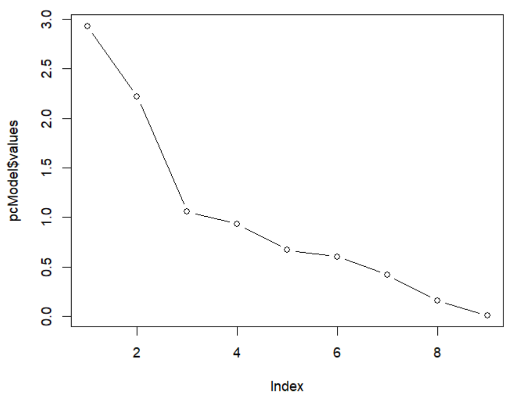


Figure 2: K-means Clustering visualisation without outliers

Appendix 2.5 – PCA Result

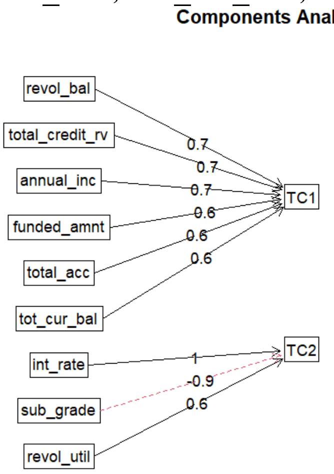
PCA is performed on 9 variables which have pairwise correlation coefficients more than 0.3 (*'funded_amnt', 'int_rate', 'sub_grade', 'annual_inc', 'revol_bal', 'revol_util', 'total_acc', 'tot_cur_bal', 'total_credit_rv'*).




```
Principal Components Analysis
Call: principal(r = sample_pca, nfactors = 3, rotate = "none")
Standardized loadings (pattern matrix) based upon correlation matrix
                  PC1   PC2   PC3   h2    u2 com
funded_amnt      0.75  0.03  0.08 0.58 0.423 1.0
int_rate         0.53 -0.79 -0.11 0.91 0.085 1.8
sub_grade       -0.55  0.77  0.10 0.90 0.100 1.8
annual_inc       0.61  0.33  0.48 0.71 0.289 2.5
revol_bal        0.75  0.22 -0.31 0.70 0.297 1.5
revol_util       0.37 -0.52  0.25 0.46 0.536 2.3
total_acc        0.49  0.40 -0.14 0.42 0.580 2.1
tot_cur_bal      0.45  0.39  0.56 0.67 0.330 2.7
total_credit_rv  0.52  0.52 -0.55 0.84 0.156 3.0

                       PC1  PC2  PC3
SS loadings           2.93 2.22 1.05
Proportion Var        0.33 0.25 0.12
Cumulative Var        0.33 0.57 0.69
Proportion Explained  0.47 0.36 0.17
Cumulative Proportion 0.47 0.83 1.00

Mean item complexity =  2.1
Test of the hypothesis that 3 components are sufficient.

The root mean square of the residuals (RMSR) is  0.1
 with the empirical chi square  413.58  with prob <  5e-81

Fit based upon off diagonal values = 0.9>
```

Appendix 2.6 – FA Result

FA is performed on 9 variables which have pairwise correlation coefficients more than 0.3(*'funded_amnt', 'int_rate', 'sub_grade', 'annual_inc', 'revol_bal', 'revol_util', 'total_acc', 'tot_cur_bal', 'total_credit_rv'*).




```
Call: principal(r = sample_pca, nfactors = 2, rotate = "oblimin")
Standardized loadings (pattern matrix) based upon correlation matrix
                item  TC1   TC2   h2    u2 com
revol_bal          5 0.74        0.61 0.390 1.1
total_credit_rv    9 0.73        0.54 0.460 1.2
annual_inc         4 0.69        0.48 0.521 1.0
funded_amnt        1 0.64  0.33 0.57 0.429 1.5
total_acc          7 0.63        0.40 0.601 1.1
tot_cur_bal        8 0.60        0.35 0.646 1.1
int_rate           2       0.95 0.90 0.096 1.0
sub_grade          3      -0.94 0.89 0.110 1.0
revol_util         6       0.63 0.40 0.597 1.0

                       TC1  TC2
SS loadings           2.73 2.42
Proportion Var        0.30 0.27
Cumulative Var        0.30 0.57
Proportion Explained  0.53 0.47
Cumulative Proportion 0.53 1.00

 With component correlations of
     TC1  TC2
TC1 1.00 0.12
TC2 0.12 1.00

Mean item complexity =  1.1
Test of the hypothesis that 2 components are sufficient.

The root mean square of the residuals (RMSR) is  0.12
 with the empirical chi square  555.33  with prob <  1.3e-105

Fit based upon off diagonal values = 0.86
```

Appendix 3 – Gap Statistic Plots

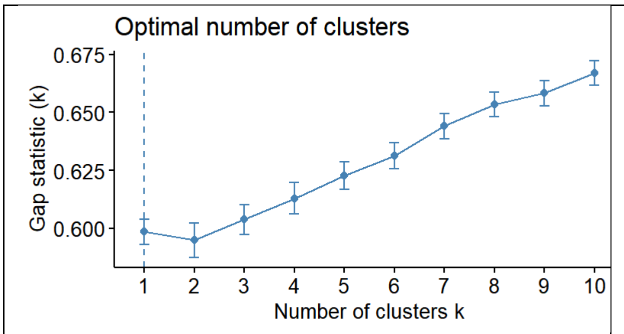


Figure X: Gap Statistic Plot for Hierarchical Clustering

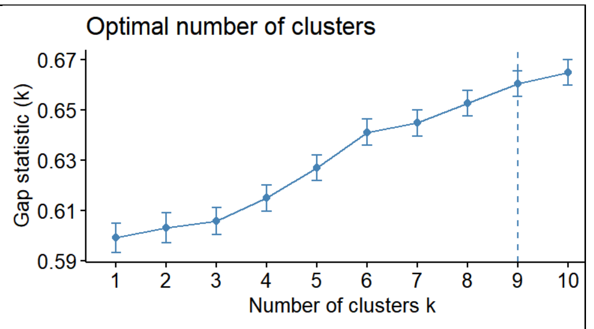


Figure X: Gap Statistic Plot for K-means Clustering

Appendix 4 – Elbow Plots

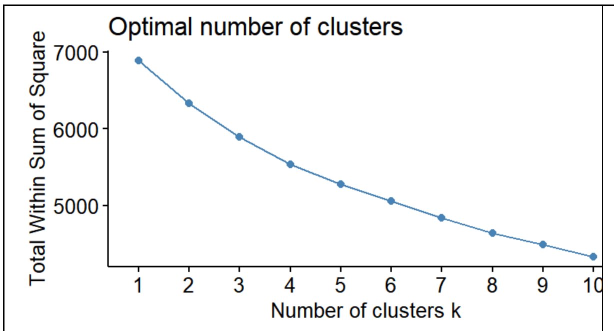


Figure X: Elbow Plot for Hierarchical Clustering

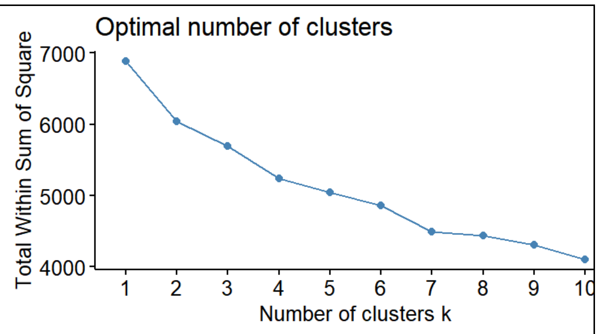


Figure X: Elbow Plot for K-means Clustering

ADA Group Assignment 1

Group 31

2024-02-28

Contents

Data Preparation

```r
# Import data
df <- read_xlsx("loan_data_ADA_assignment.xlsx",sheet="in")
```

```r
# Check structure and summary
str(df)
```

```
## 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 ...
```

```r
summary(df)
```

```
##        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
##
##
##
##
```

```r
# Check if each customer id is unique
n_distinct(df$member_id)
```

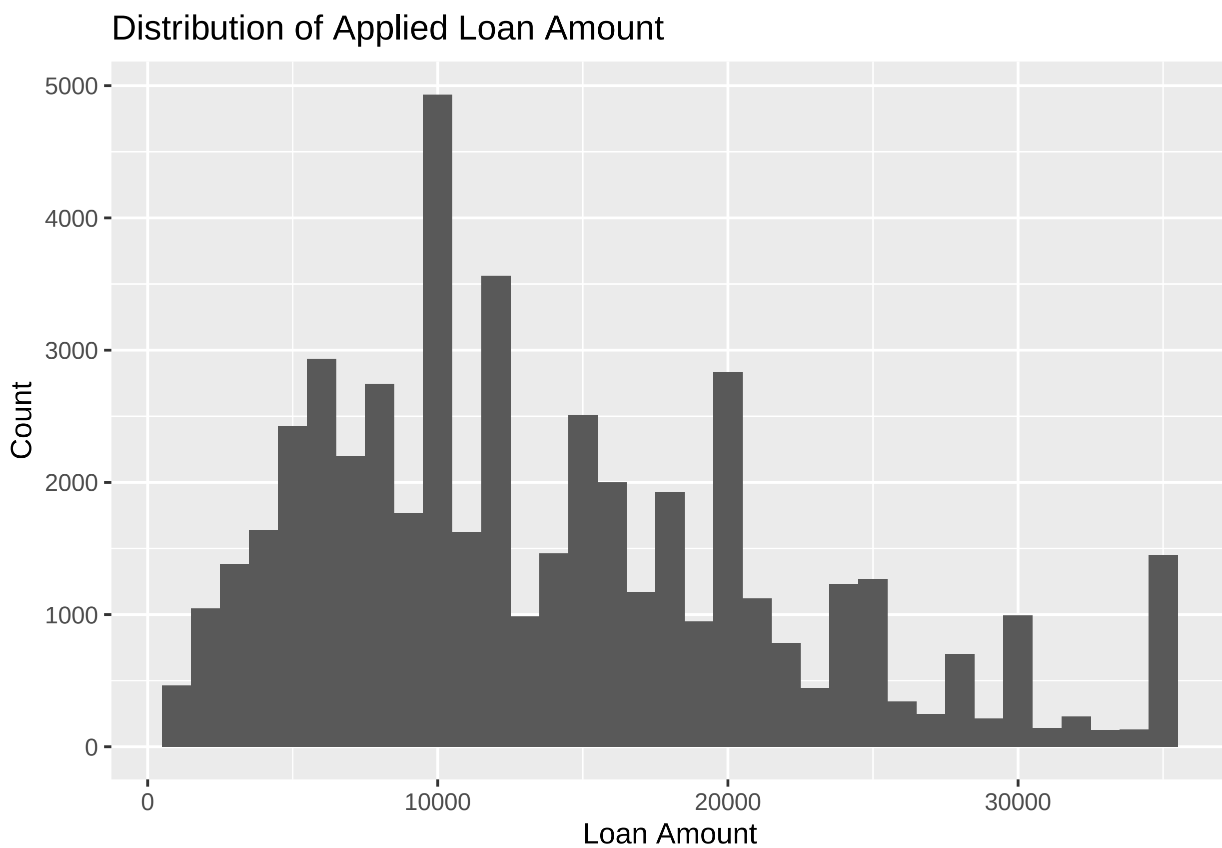
```
## [1] 50000
```

There are no repeat customers, and each loan is taken by a unique customer.

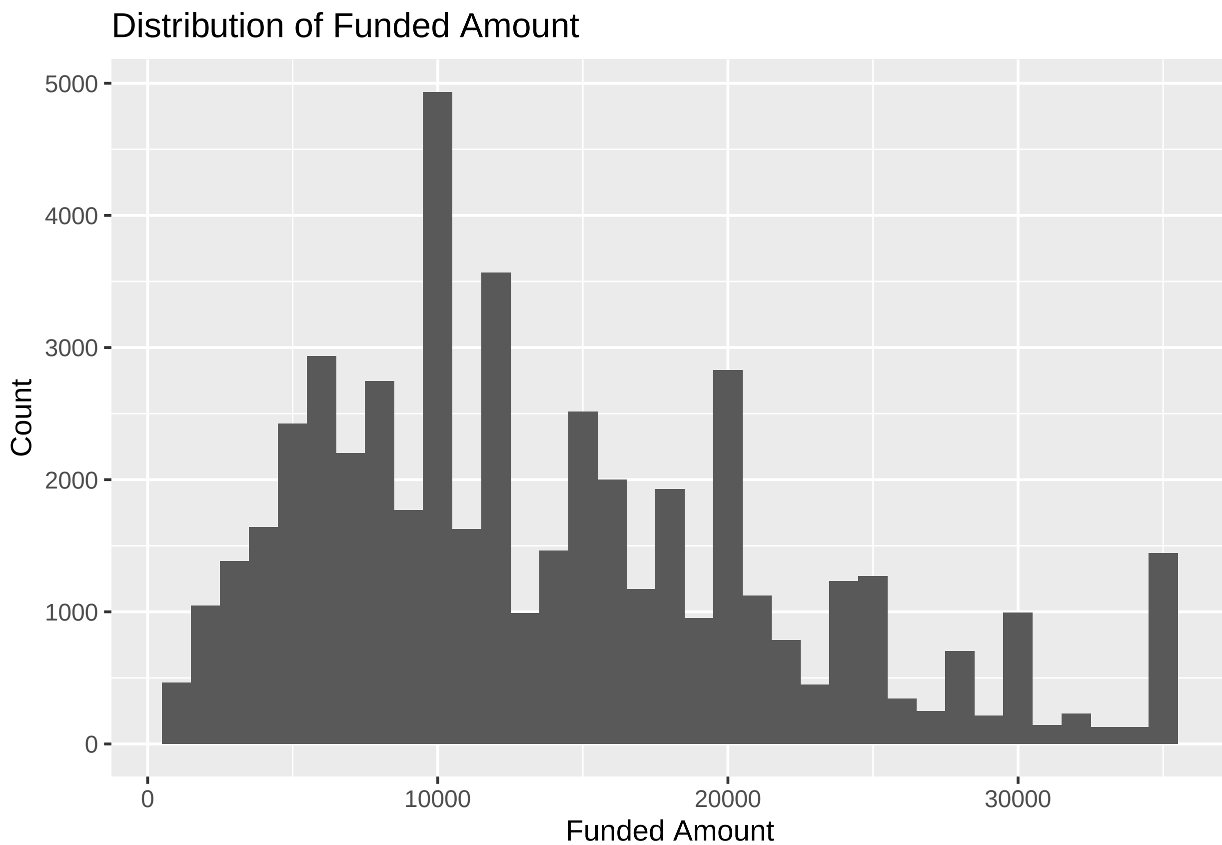
Check Distribution

Loan amount and funded amount

```r
# Check the distribution of loan amount (numeric continuous variable) in histogram
ggplot(df) + geom_histogram(aes(loan_amnt), binwidth=1000) +
  labs(y = "Count", x = "Loan Amount", title = "Distribution of Applied Loan Amount")
```



```r
# Check the distribution of funded amount (numeric continuous variable) in histogram
ggplot(df) +
  geom_histogram(aes(funded_amnt), binwidth=1000) +
  labs(y = "Count", x = "Funded Amount", title = "Distribution of Funded Amount")
```

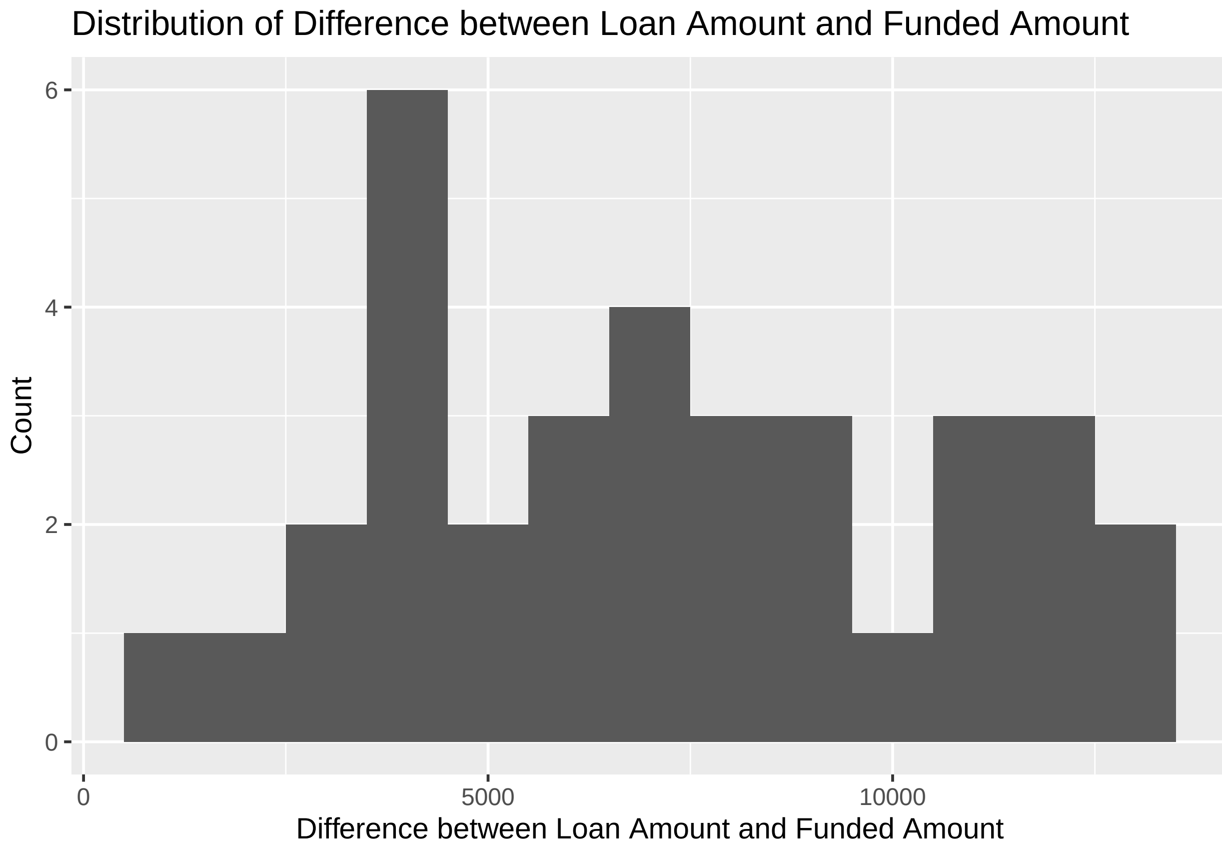


We can see there is not much difference in the distribution between loan_amnt and funded_amnt. Hence, most customers receive the exact loan amount they applied for except for a small number of exceptions. Hence, we choose to use funded_amnt.

```r
# Compare the value of loan_amnt to funded_amnt
sum(df$funded_amnt < df$loan_amnt)
```

```
## [1] 34
```

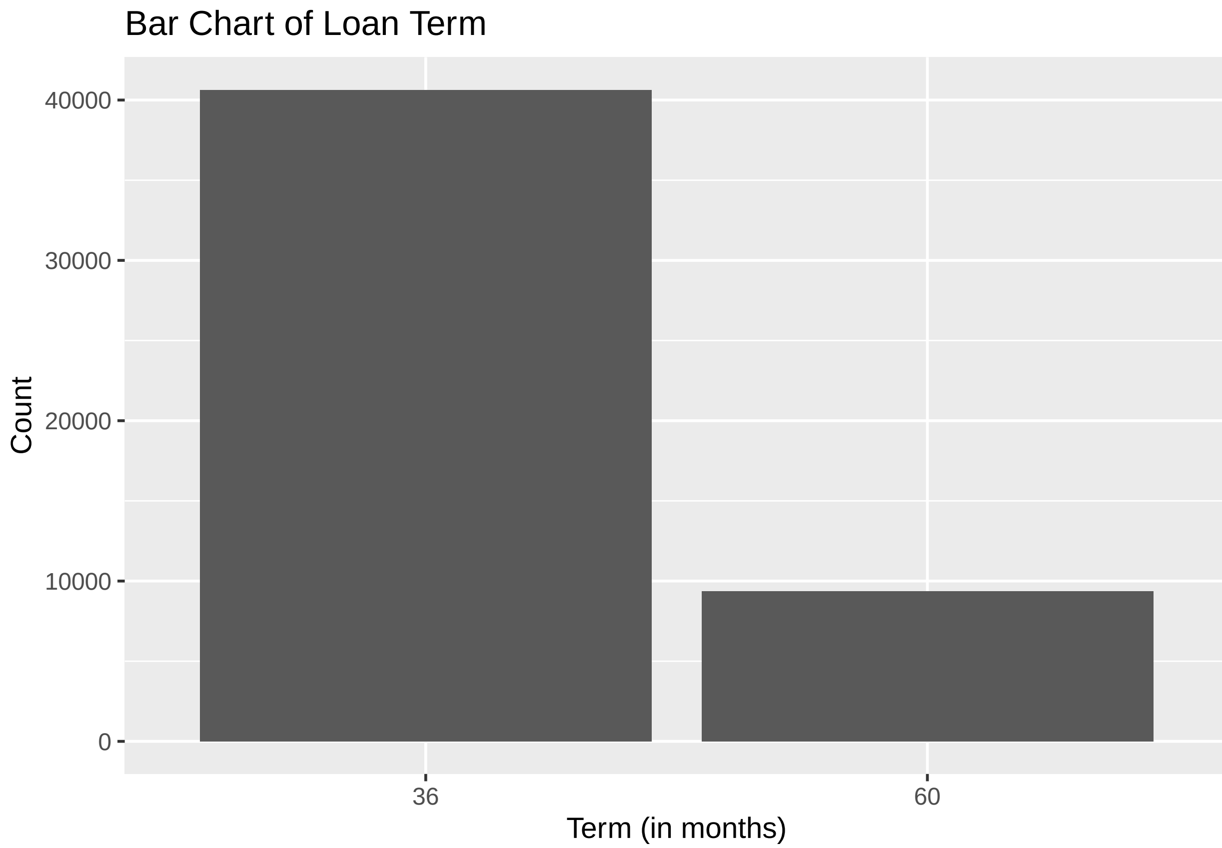
```r
# 34 cases where actual funded amount is smaller than loan amount (really small percentage)

df_diff <- filter(df, funded_amnt < loan_amnt)
df_diff <- mutate(df_diff, difference = loan_amnt-funded_amnt)
ggplot(df_diff) +
  geom_histogram(aes(difference), binwidth=1000) +
  labs(y = "Count", x = "Difference between Loan Amount and Funded Amount", title = "Distribution of Di
```



Despite there is only 34 cases, the difference between applied loan amount and funded amount is not negligible amount, the magnitude of the difference is quite significant. Hence, we will be including the calculated difference between applied loan amount and actual funded amount.

Term

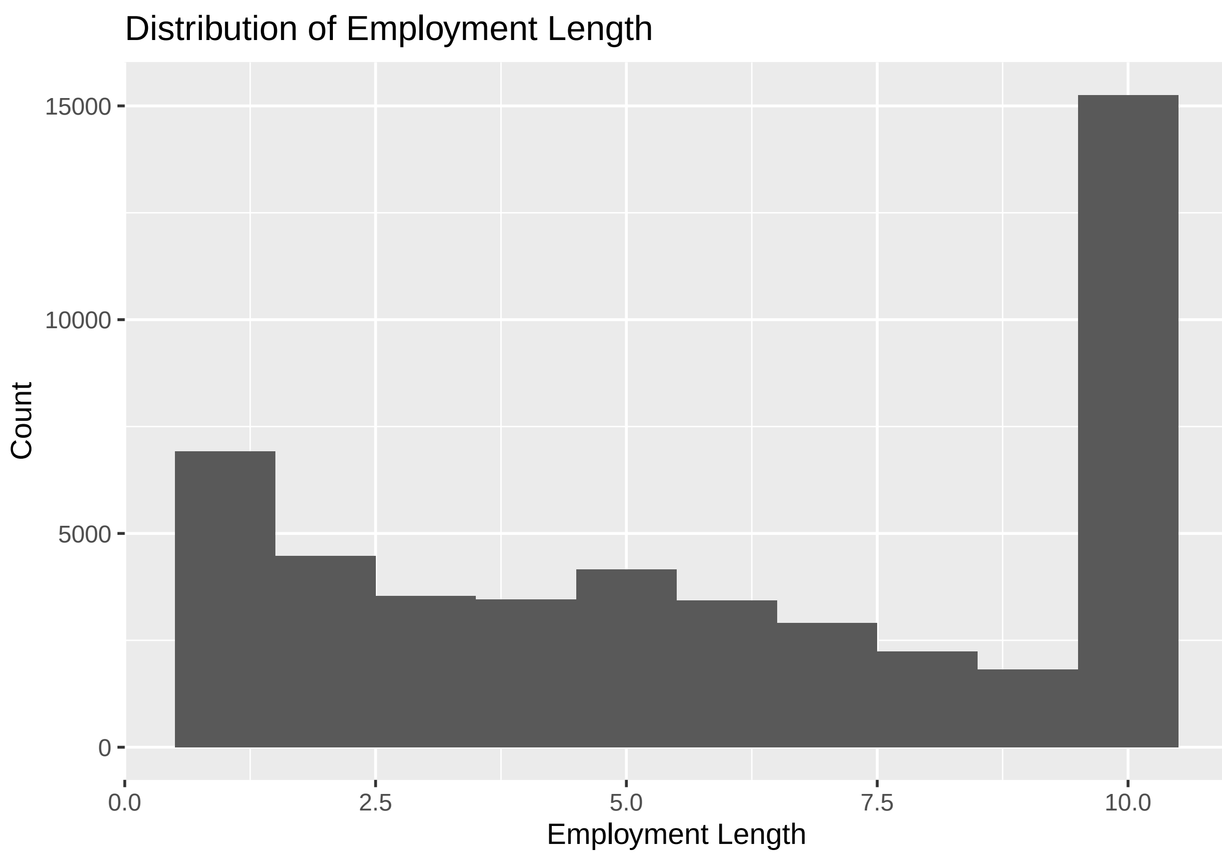
```
# Check the distribution of term (categorical variable) in bar chart
df$term <- as.factor(df$term)

ggplot(df,aes(x = term)) + geom_bar() +
  labs(y = "Count", x = "Term (in months)", title = "Bar Chart of Loan Term")
```



Employment length

```
# Check the distribution of employment length
ggplot(df) + geom_histogram(aes(emp_length), binwidth=1) +
  labs(y = "Count", x = "Employment Length", title = "Distribution of Employment Length")
```

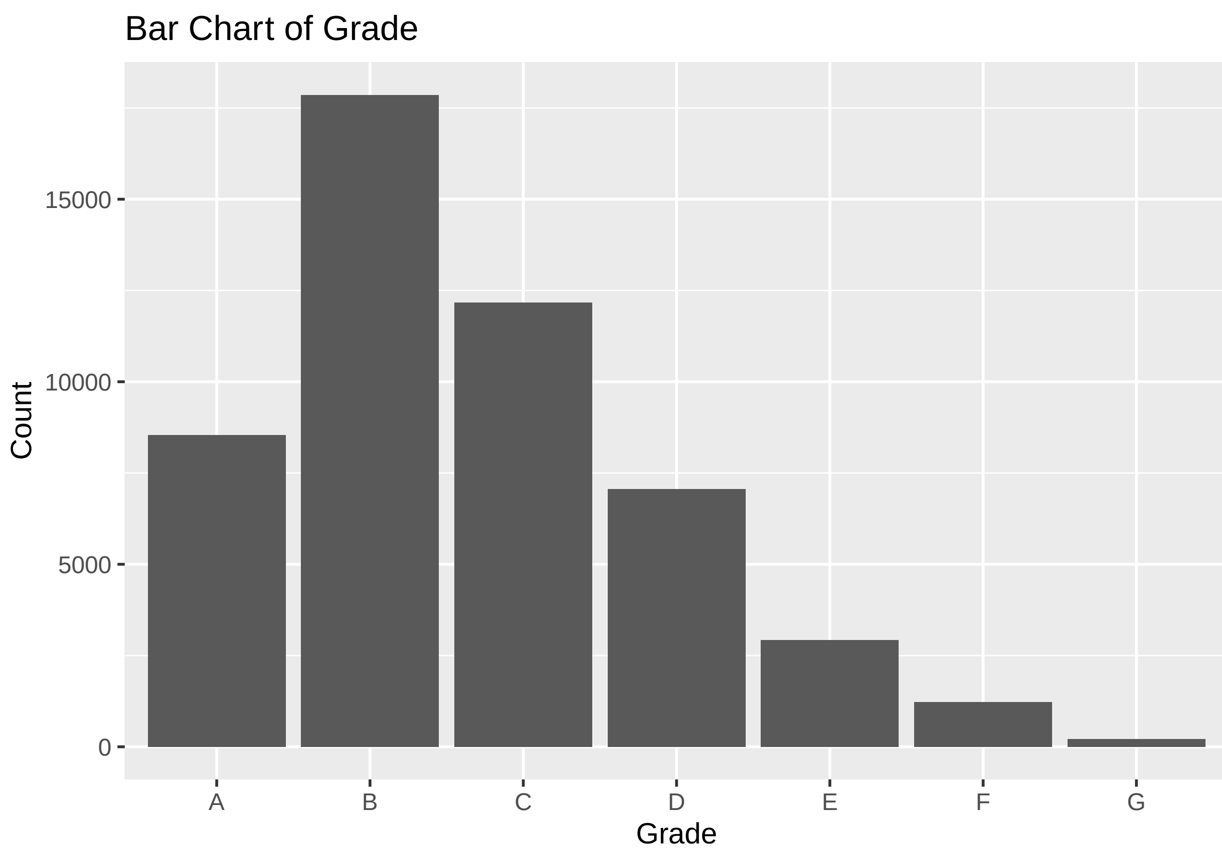
```
## Warning: Removed 1802 rows containing non-finite values (`stat_bin()`).
```



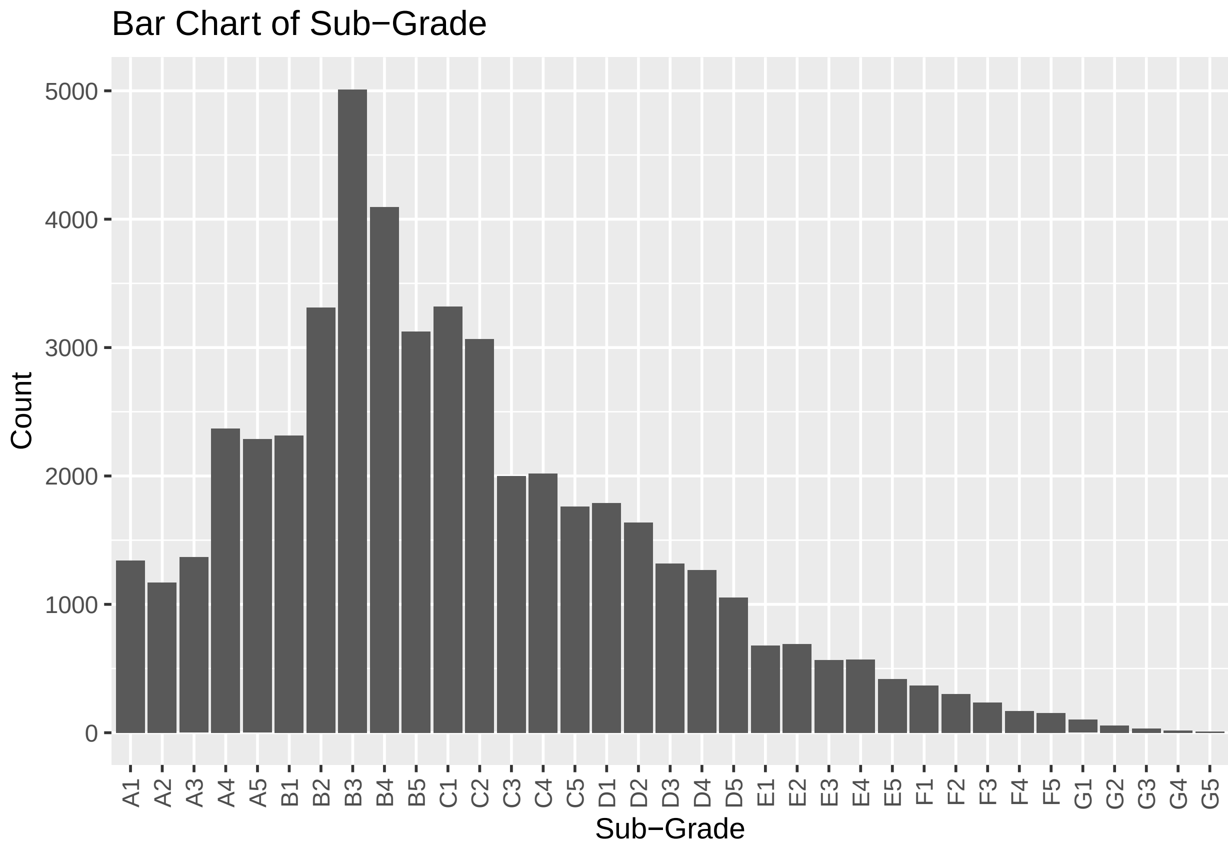
Most customers are employed for around 10 years.

Grade and sub-grade

```
# Check the distribution of grade and sub-grade
df$grade <- as.factor(df$grade)

ggplot(df,aes(x = grade)) + geom_bar() +
  labs(y = "Count", x = "Grade", title = "Bar Chart of Grade")
```



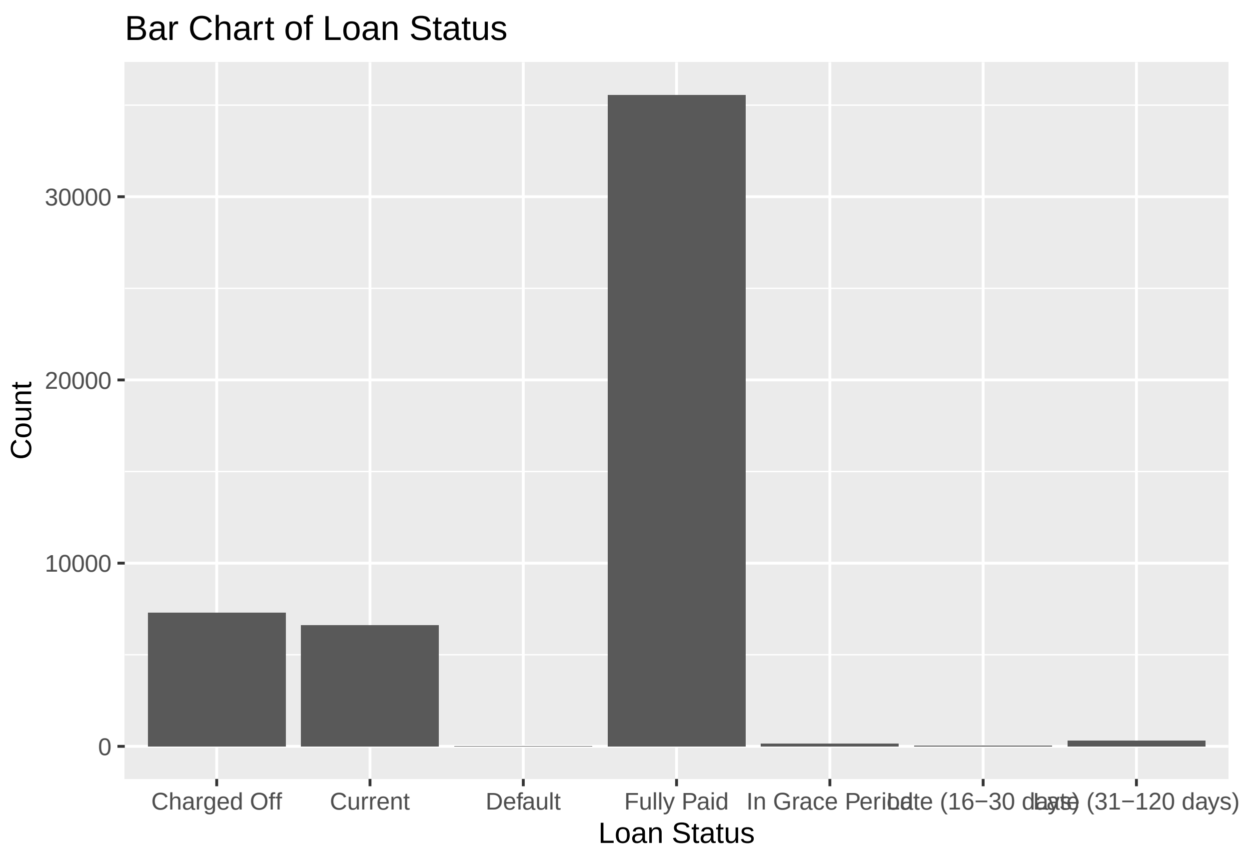

```
df$sub_grade <- as.factor(df$sub_grade)

ggplot(df,aes(x = sub_grade)) + geom_bar() +
  labs(y = "Count", x = "Sub-Grade", title = "Bar Chart of Sub-Grade") +
    scale_x_discrete(guide = guide_axis(angle = 90))
```



Most customers are centered around low A to high C range, with the highest number of customers in grade B.

Loan status

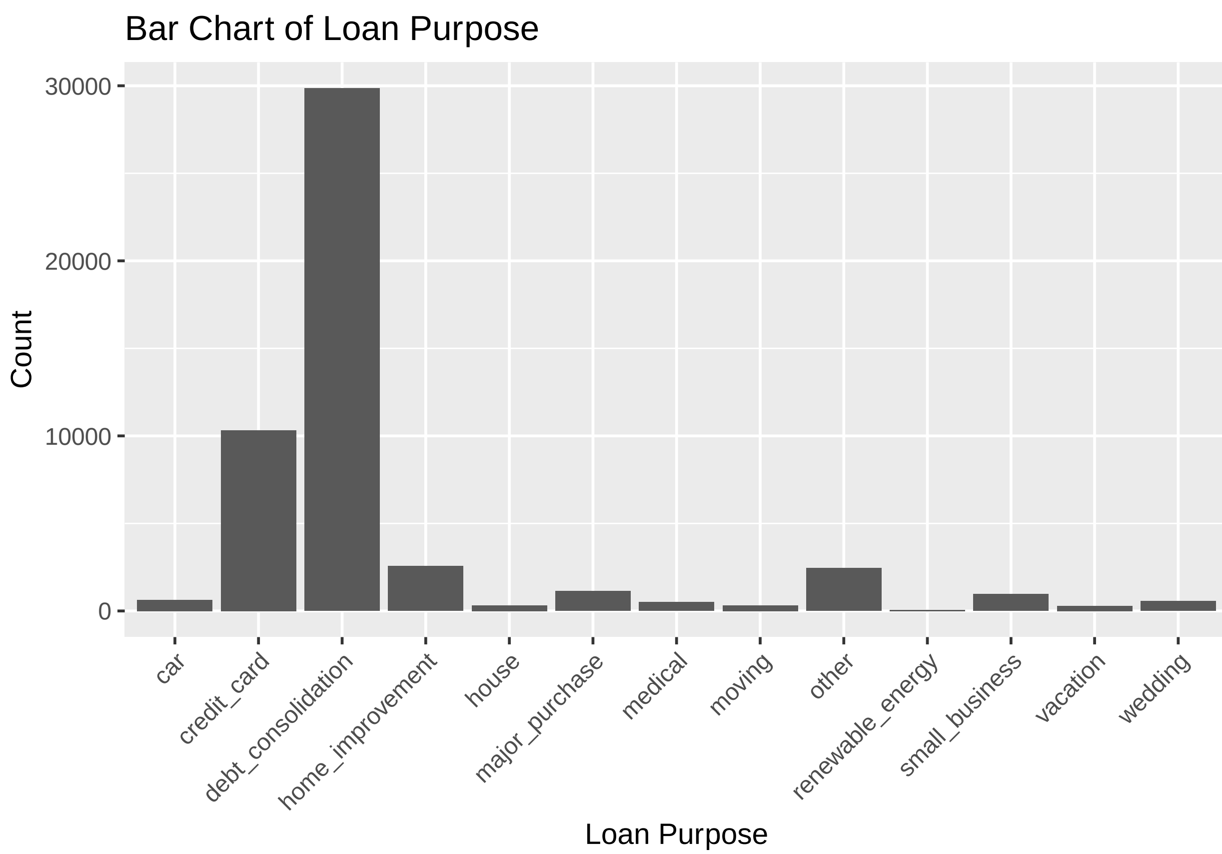
```
# Check the distribution of loan status
df$loan_status <- as.factor(df$loan_status)

ggplot(df,aes(x = loan_status)) + geom_bar() +
  labs(y = "Count", x = "Loan Status", title = "Bar Chart of Loan Status")
```



Most customers are past customers with a small number of 'current' customers. Majority fall under the 'fully paid' category. 'Good debt' consists of 'current' and 'fully paid' customers.'bad debt' consists of 'charged off', 'default', 'in grace period' and 'late' customers, taking up 15.6% of the population.

Loan purpose

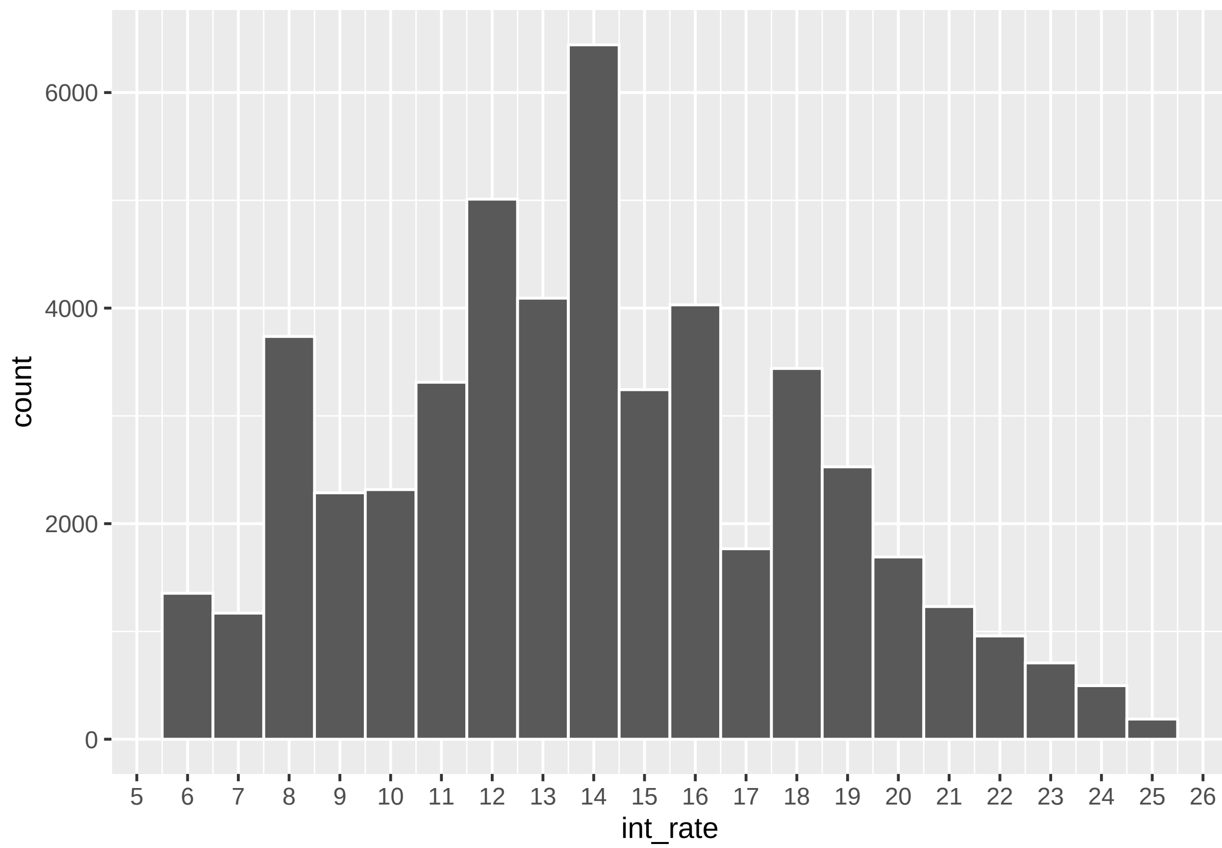
```
# Check the distribution of loan purpose
df$purpose <- as.factor(df$purpose)

ggplot(df,aes(x = purpose)) + geom_bar() +
  labs(y = "Count", x = "Loan Purpose", title = "Bar Chart of Loan Purpose") +
  scale_x_discrete(guide = guide_axis(angle = 45))
```



Most customers take loans for debt consolidation (highest number) or to repay credit card debt (runner-up).

Interest rate

```
# Check the distribution of interest rate
ggplot(df) + geom_histogram(aes(int_rate), binwidth=1, col="white") +
  scale_x_continuous(breaks=seq(5,26,1))
```



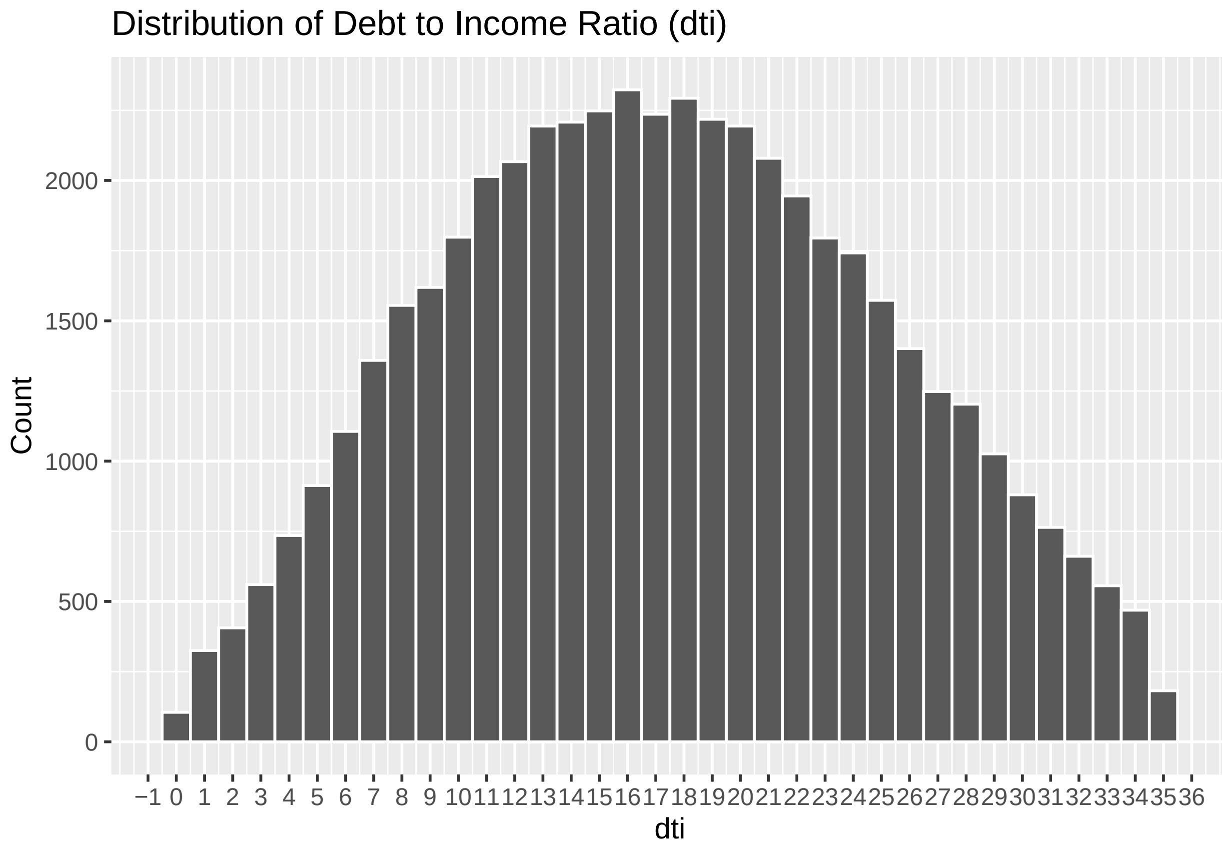

```
  labs(y = "Count", x = "Interest Rate", title = "Distribution of Interest Rate")
```

```
## $y
## [1] "Count"
##
## $x
## [1] "Interest Rate"
##
## $title
## [1] "Distribution of Interest Rate"
##
## attr(,"class")
## [1] "labels"
```

Interest rate ranges from 6% to 25%, with the highest count around 13-14%.

Dti (debt to income ratio)

```
# Check the distribution of debt to income ratio
ggplot(df) +geom_histogram(aes(dti), binwidth=1, col="white") +
  scale_x_continuous(breaks=seq(-1,36,1)) +
  labs(y = "Count", x = "dti", title = "Distribution of Debt to Income Ratio (dti)")
```

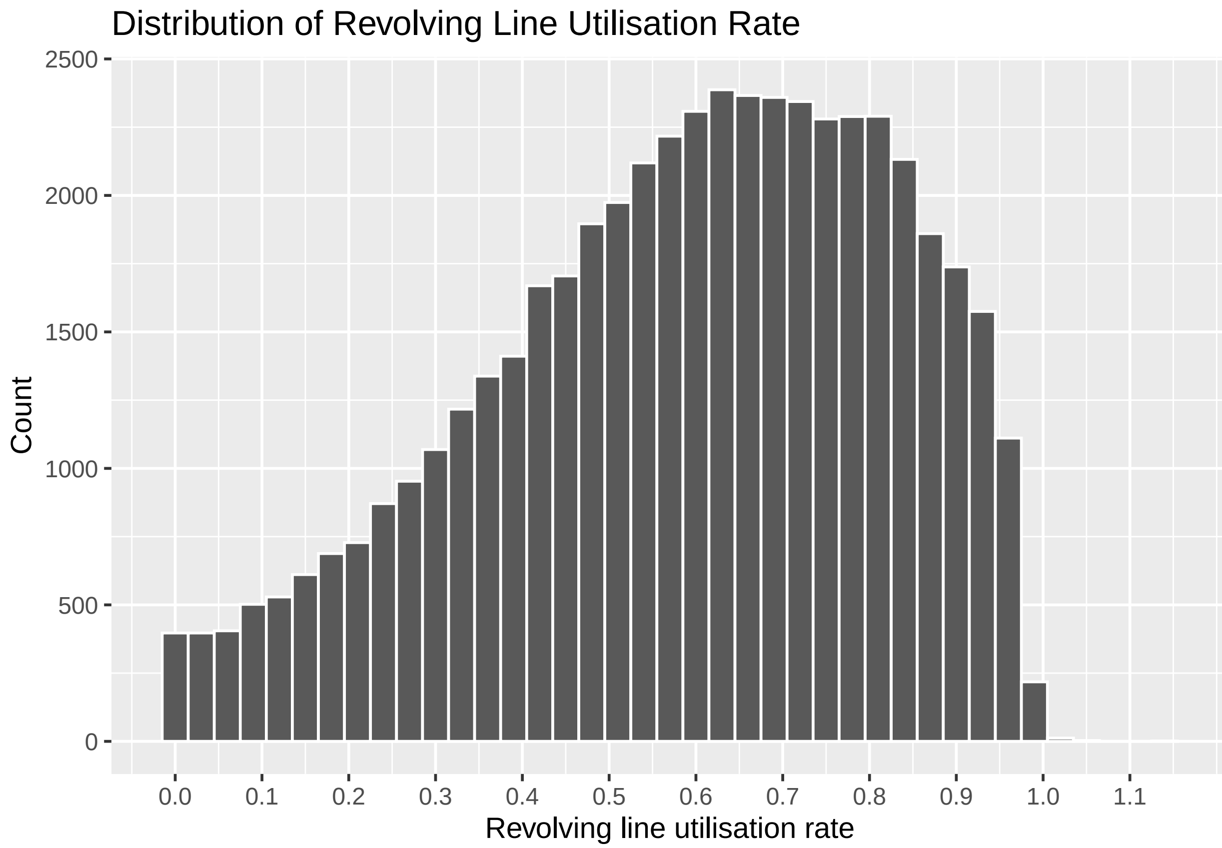


Distribution is symmetrical and clustered around 17. This means that on average, borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, is 17 times of their monthly income.

Revolving line utilization rate

```
# Check the distribution of revolving line utilization rate
ggplot(df) + geom_histogram(aes(revol_util), binwidth=0.03,col="white") +
  scale_x_continuous(breaks=seq(0,1.1, 0.1)) +
  labs(y = "Count", x = "Revolving line utilisation rate", title = "Distribution of Revolving Line Utili
```

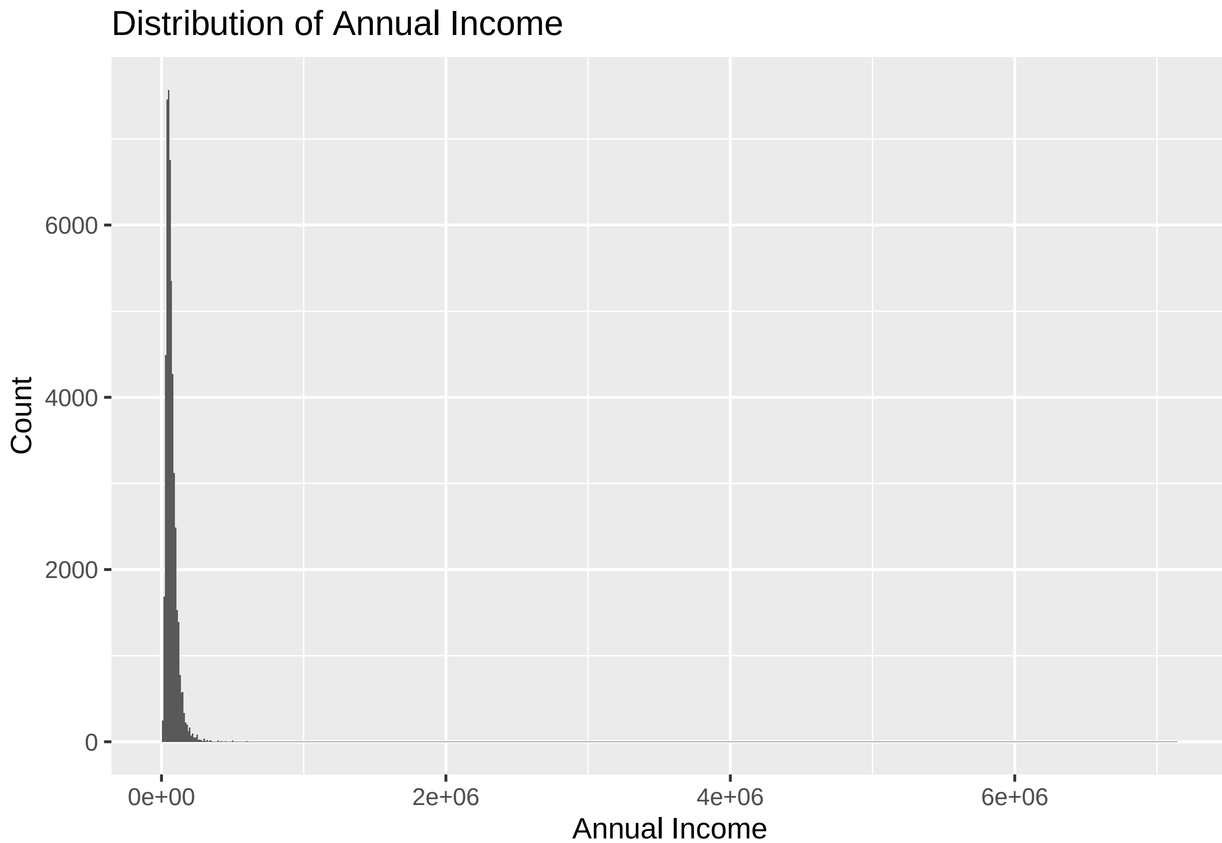
```
## Warning: Removed 31 rows containing non-finite values (`stat_bin()`).
```



Histogram is right-skewed towards 1, suggesting the most customers spend a relatively high % of the credit lines they have taken

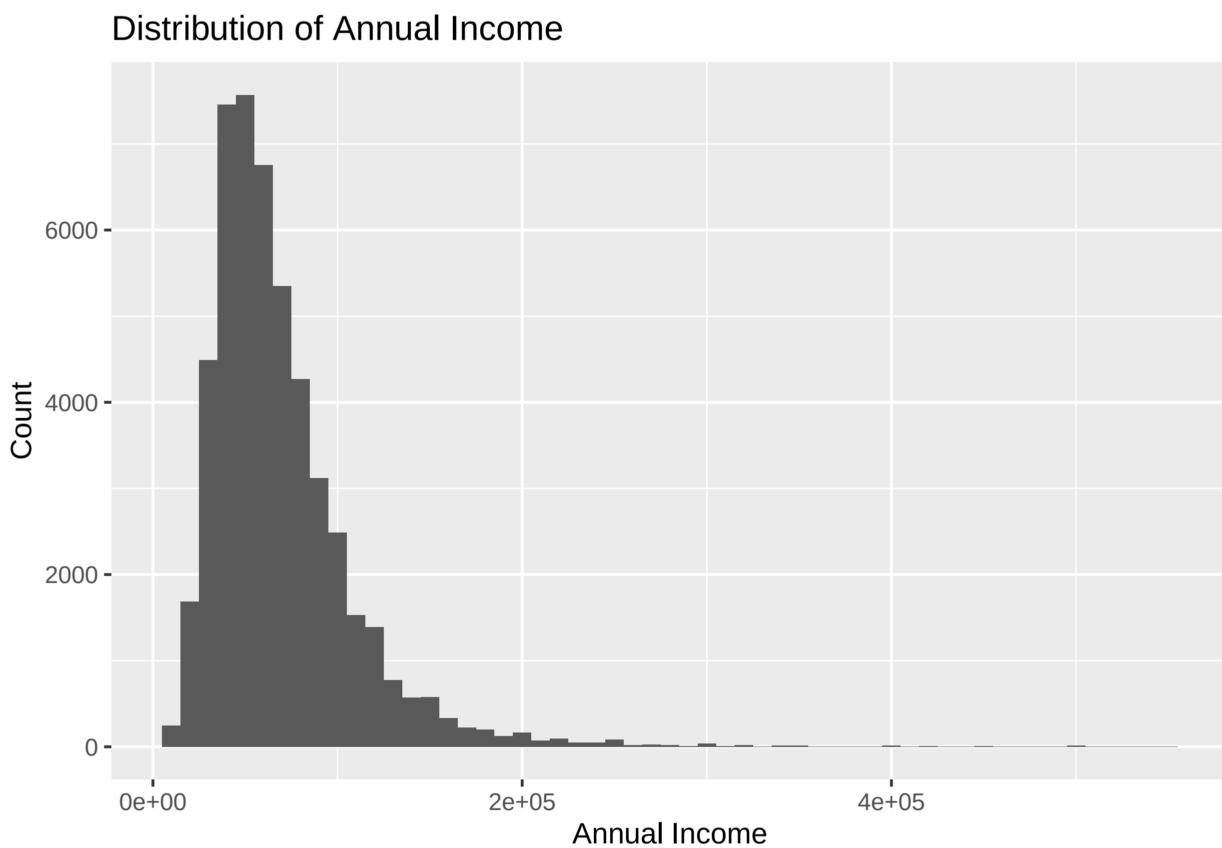
Annual income

```
# Check the distribution of annual income (numeric continuous variable) in histogram
ggplot(df) + geom_histogram(aes(annual_inc), binwidth=10000) +
  labs(y = "Count", x = "Annual Income", title = "Distribution of Annual Income")
```




```
top_income <- top_n(df, 50, df$annual_inc)
```

```
# Check the distribution of annual income without outliers
matching_rows <- df$id %in% top_income$id

# Remove the subset rows from the population
without_outliers <- df[!matching_rows, ]

# Distribution of annual income without outliers
ggplot(without_outliers) + geom_histogram(aes(annual_inc), binwidth=10000) +
  labs(y = "Count", x = "Annual Income", title = "Distribution of Annual Income")
```



Interesting thing to note is there are a very small number (around 7) of people with extremely high annual income above 1000k, that are outliers, but their income are verified. The top customers with the highest income tend to be verified. Hence, we do not remove these outliers as their income are reliable and they could be a potential minority group of customers.

Missing Value

```
# Detect NA value
sum(is.na(df))
```

```
## [1] 228899
```

```
# There are 228899 empty cells in df

sum(rowSums(is.na(df)) > 0)
```

```
## [1] 49941
```

```
# There are 49941 rows with at least an empty cell in its row

sum(rowSums(is.na(df)) > 0)/nrow(df)
```

```
## [1] 0.99882
```

```
# 99.8% of rows have at least an empty cell in its row
```

mths_since_last_delinq

```r
# Check the percentage of missing data of the column mths_since_last_delinq
sum(is.na(df$mths_since_last_delinq))/nrow(df)
```

```
## [1] 0.56252
```

```r
df$mths_since_last_delinq[is.na(df$mths_since_last_delinq)] <- mean(df$mths_since_last_delinq, na.rm = T
```

Because the percentage of missing data is not very high (56.2%), we do not drop the column and instead replace NA values with the mean.

mths_since_last_record

```r
# Check the percentage of missing data of the column mths_since_last_record
sum(is.na(df$mths_since_last_record))/nrow(df)
```

```
## [1] 0.94936
```

Because the percentage of missing data is high (94.9%), we drop the column.

mths_since_last_major_derog

```r
# Check the percentage of missing data of the column mths_since_last_major_derog
sum(is.na(df$mths_since_last_major_derog))/nrow(df)
```

```
## [1] 0.8576
```

Because the percentage of missing data is high (85.8%), we drop the column.

emp_length

```r
# Check the percentage of missing data of the column emp_length
sum(is.na(df$emp_length))/nrow(df)
```

```
## [1] 0.03604
```

```r
df$emp_length[is.na(df$emp_length)] <- mean(df$emp_length, na.rm = TRUE)
```

Because the percentage of missing data is low (3.6%), we do not drop the column and instead replace NA values with the mean.

revol_util

```r
# Check the percentage of missing data of the column revol_util
sum(is.na(df$revol_util))/nrow(df)
```

```
## [1] 0.00062
```

```r
df$revol_util[is.na(df$revol_util)] <- mean(df$revol_util, na.rm = TRUE)
```

Because the percentage of missing data is very low (0.062%), we do not drop the column and instead replace NA values with the mean.

tot_coll_amt, tot_cur_bal and total_credit_rv

```r
# Check the percentage of missing data of the columns tot_coll_amt, tot_cur_bal and total_credit_rv
sum(is.na(df$tot_coll_amt))/nrow(df)
```

```
## [1] 0.29236
```

```r
sum(is.na(df$tot_cur_bal))/nrow(df)
```

```
## [1] 0.29236
```

```r
sum(is.na(df$total_credit_rv))/nrow(df)
```

```
## [1] 0.29236
```

There are the same number of missing data rows for tot_coll_amt, tot_cur_bal and total_credit_rv. Upon closer inspection, we realise that they all belong to the same rows.

```r
df$tot_coll_amt[is.na(df$tot_coll_amt)] <- mean(df$tot_coll_amt, na.rm = TRUE)
df$tot_cur_bal[is.na(df$tot_cur_bal)] <- mean(df$tot_cur_bal, na.rm = TRUE)
df$total_credit_rv[is.na(df$total_credit_rv)] <- mean(df$total_credit_rv, na.rm = TRUE)
```

Because the percentage of missing data is not very high (29.2%), we do not drop the column and instead replace NA values with the mean of the remaining available data in that column.

Feature Selection

Create calculated fields

```r
# create no. of months since last credit pull (numeric variable)
df$months_since_last_credit_pull <- (interval((df$issue_d), (df$last_credit_pull_d)) %/% months(1))
```

Based on data dictionary, we select the variable which relevant to do clustering. We dropped 31 variables.

```r
# Made a list of column that need to drop
drop_column <- c('id','member_id','funded_amnt_inv','emp_title','issue_d','desc','title','zip_code','ea
```

```r
# Drop the columns
df2 <- df %>% select(-drop_column)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(drop_column)
##
##   # Now:
##   data %>% select(all_of(drop_column))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
# Drop missing data
df3 <- na.omit(df2)

sum(is.na(df3))
```

```
## [1] 0
```

```r
# All missing data has been filled, no NAs
```

Encoding

We transform the categorical data to be numerical with assumption that the distance between one class to another is similar.

```r
# Integer encoding for 'subgrade'
df3$sub_grade <- dplyr::recode(df3$sub_grade,"A1"=35,"A2"=34,"A3"=33,"A4"=32,"A5"=31,"B1"=30,"B2"=29,"B3
```

```r
# Loan Status. Check its level.
df3$loan_status <- as.factor(df3$loan_status)
```

```r
levels(df3$loan_status)
```

```
## [1] "Charged Off"        "Current"            "Default"
## [4] "Fully Paid"         "In Grace Period"    "Late (16-30 days)"
## [7] "Late (31-120 days)"
```

```r
# Level still based on assumption
df3$loan_status <- dplyr::recode(df3$loan_status,"Charged Off"=1,"Late (31-120 days)"=2,"Late (16-30 da
```

Correlation and Multicollinearity

Utilizing Spearman correlation with our assumption of normal data distribution, where "r" represents the correlation coefficient:

- $r = 1$ is a perfect positive correlation.
- $r = 0$ means no correlation.
- $r = -1$ means a perfect negative correlation.

```r
numeric_sample <- as.data.frame(lapply(df3, as.numeric))
correlation_matrix_spearman <- cor(numeric_sample, method = "spearman")
print(correlation_matrix_spearman)
```

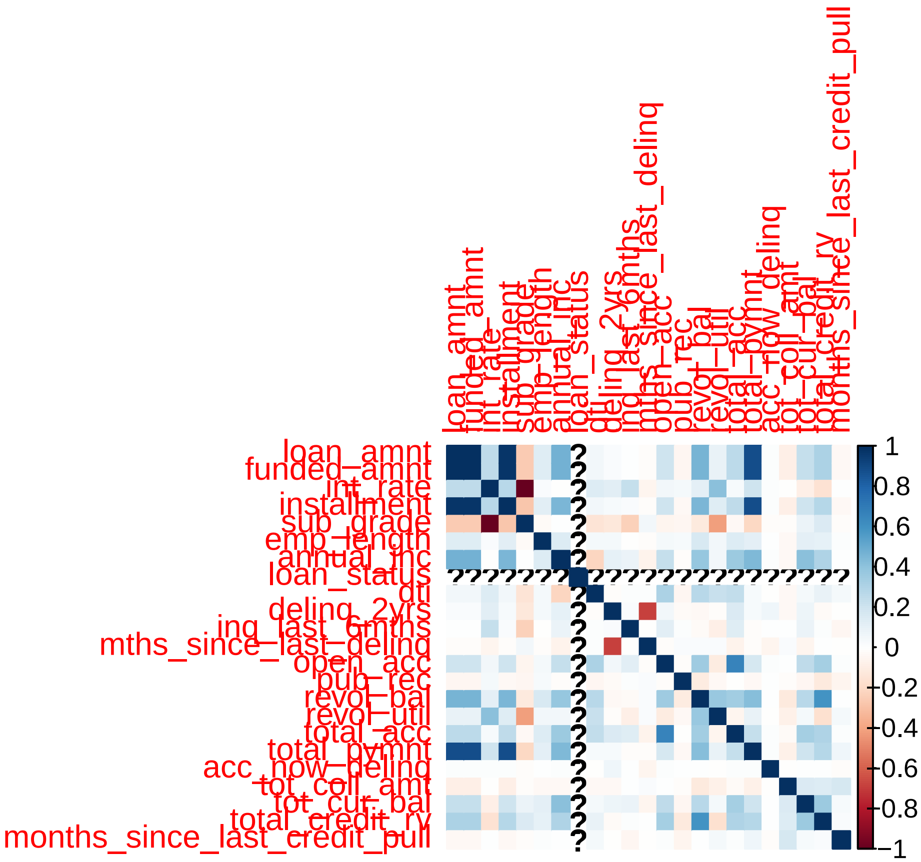
```
##                                 loan_amnt funded_amnt     int_rate installment
## loan_amnt                     1.000000000  0.99986251  0.258337505  0.97679117
## funded_amnt                   0.999862512  1.00000000  0.258260325  0.97693332
## int_rate                      0.258337505  0.25826032  1.000000000  0.27494452
## installment                   0.976791174  0.97693332  0.274944520  1.00000000
## sub_grade                    -0.258379614 -0.25827655 -0.998362617 -0.27429117
## emp_length                    0.131429342  0.13145462  0.024537614  0.12189391
## annual_inc                    0.477175107  0.47706662 -0.004011929  0.45989746
## loan_status                            NA          NA           NA          NA
## dti                           0.052568890  0.05251425  0.147509003  0.05327436
## delinq_2yrs                   0.025520547  0.02555914  0.122392140  0.03639896
## inq_last_6mths                0.008115187  0.00804998  0.237817181  0.02002947
## mths_since_last_delinq       -0.010833808 -0.01086396 -0.051248502 -0.01361056
## open_acc                      0.203110576  0.20303499  0.050823706  0.20094357
## pub_rec                      -0.043178644 -0.04309539  0.042614997 -0.03828899
## revol_bal                     0.460275563  0.46025613  0.118222107  0.45587113
## revol_util                    0.096761205  0.09669187  0.417200278  0.13501640
## total_acc                     0.264629376  0.26452354  0.032221701  0.25137548
## total_pymnt                   0.889036703  0.88919047  0.205599677  0.88741789
## acc_now_delinq                0.012062998  0.01207722  0.017421873  0.01267840
## tot_coll_amt                 -0.086099050 -0.08664127 -0.001150042 -0.08573837
## tot_cur_bal                   0.234730177  0.23466805 -0.088740691  0.20371570
## total_credit_rv               0.319623120  0.31956742 -0.156020451  0.28891375
## months_since_last_credit_pull -0.030883334 -0.03095741  0.009450027 -0.03923698
##                                 sub_grade   emp_length   annual_inc
## loan_amnt                    -0.258379614  0.131429342  0.477175107
## funded_amnt                  -0.258276552  0.131454624  0.477066621
## int_rate                     -0.998362617  0.024537614 -0.004011929
## installment                  -0.274291168  0.121893906  0.459897463
## sub_grade                     1.000000000 -0.023839540  0.004428529
## emp_length                   -0.023839540  1.000000000  0.145214704
## annual_inc                    0.004428529  0.145214704  1.000000000
## loan_status                            NA           NA           NA
```

```
## dti                          -0.146602918  0.043531048 -0.218989268
## delinq_2yrs                  -0.121772988  0.044827614  0.109225753
## inq_last_6mths               -0.238142752 -0.002882133  0.080137232
## mths_since_last_delinq        0.050899463 -0.011589950 -0.063045795
## open_acc                     -0.050207087  0.043464097  0.234389366
## pub_rec                      -0.041428978  0.031067268 -0.014905473
## revol_bal                    -0.117103726  0.163467914  0.387595034
## revol_util                   -0.417107617  0.059197829  0.058077989
## total_acc                    -0.031354316  0.145857249  0.365027127
## total_pymnt                  -0.205355640  0.118103178  0.446655918
## acc_now_delinq               -0.017128397  0.007515591  0.012165413
## tot_coll_amt                 -0.011897477 -0.031871277 -0.037297124
## tot_cur_bal                   0.085142202  0.110686482  0.416550499
## total_credit_rv               0.151741450  0.100681940  0.307506363
## months_since_last_credit_pull -0.013954288  0.018327006  0.003045817
##                              loan_status          dti  delinq_2yrs
## loan_amnt                             NA  0.052568890  0.025520547
## funded_amnt                           NA  0.052514251  0.025559140
## int_rate                              NA  0.147509003  0.122392140
## installment                           NA  0.053274362  0.036398956
## sub_grade                             NA -0.146602918 -0.121772988
## emp_length                            NA  0.043531048  0.044827614
## annual_inc                            NA -0.218989268  0.109225753
## loan_status                            1           NA           NA
## dti                                   NA  1.000000000 -0.012183737
## delinq_2yrs                           NA -0.012183737  1.000000000
## inq_last_6mths                        NA  0.019430352  0.020957463
## mths_since_last_delinq                NA  0.019762618 -0.682814708
## open_acc                              NA  0.316467432  0.059964275
## pub_rec                               NA -0.040174812 -0.029661323
## revol_bal                             NA  0.275159484 -0.032236212
## revol_util                            NA  0.223969860 -0.014916689
## total_acc                             NA  0.245006548  0.153290262
## total_pymnt                           NA  0.031763166  0.033119361
## acc_now_delinq                        NA -0.001389079  0.064331302
## tot_coll_amt                          NA -0.037967024 -0.033353047
## tot_cur_bal                           NA  0.045662002  0.078617317
## total_credit_rv                       NA  0.092881556 -0.029644398
## months_since_last_credit_pull         NA  0.040799453 -0.001667593
##                              inq_last_6mths mths_since_last_delinq
## loan_amnt                       0.008115187           -0.010833808
## funded_amnt                     0.008049980           -0.010863957
## int_rate                        0.237817181           -0.051248502
## installment                     0.020029473           -0.013610562
## sub_grade                      -0.238142752            0.050899463
## emp_length                     -0.002882133           -0.011589950
## annual_inc                      0.080137232           -0.063045795
## loan_status                              NA                     NA
## dti                             0.019430352            0.019762618
## delinq_2yrs                     0.020957463           -0.682814708
## inq_last_6mths                  1.000000000           -0.009797634
## mths_since_last_delinq         -0.009797634            1.000000000
## open_acc                        0.124263601           -0.018032637
## pub_rec                         0.013007007            0.026129610
```

```
## revol_bal                       -0.029971250          0.021113686
## revol_util                      -0.083647643          0.020364903
## total_acc                        0.138875917         -0.063011819
## total_pymnt                     -0.010190204         -0.013525986
## acc_now_delinq                   0.005573601         -0.050479051
## tot_coll_amt                     0.005268626          0.026886345
## tot_cur_bal                      0.087006958         -0.051409789
## total_credit_rv                  0.015209919          0.005259618
## months_since_last_credit_pull   -0.040420955         -0.002541929
##                                     open_acc      pub_rec    revol_bal
## loan_amnt                      0.2031105760 -0.043178644  0.460275563
## funded_amnt                    0.2030349890 -0.043095391  0.460256131
## int_rate                       0.0508237059  0.042614997  0.118222107
## installment                    0.2009435725 -0.038288994  0.455871135
## sub_grade                     -0.0502070875 -0.041428978 -0.117103726
## emp_length                     0.0434640972  0.031067268  0.163467914
## annual_inc                     0.2343893662 -0.014905473  0.387595034
## loan_status                              NA           NA           NA
## dti                            0.3164674322 -0.040174812  0.275159484
## delinq_2yrs                    0.0599642753 -0.029661323 -0.032236212
## inq_last_6mths                 0.1242636012  0.013007007 -0.029971250
## mths_since_last_delinq        -0.0180326372  0.026129610  0.021113686
## open_acc                       1.0000000000 -0.012341186  0.354944155
## pub_rec                       -0.0123411864  1.000000000 -0.104688974
## revol_bal                      0.3549441546 -0.104688974  1.000000000
## revol_util                    -0.1028301482 -0.037853828  0.370504423
## total_acc                      0.6645472805 -0.007771288  0.345718004
## total_pymnt                    0.1791419527 -0.036207527  0.423682389
## acc_now_delinq                 0.0144294125  0.004376214  0.003015193
## tot_coll_amt                   0.0001339259 -0.018036339 -0.115717947
## tot_cur_bal                    0.2508281789 -0.041704689  0.272826793
## total_credit_rv                0.3385360824 -0.117611374  0.596382670
## months_since_last_credit_pull  0.0186585142 -0.058419699  0.008105816
##                                  revol_util    total_acc total_pymnt
## loan_amnt                      0.096761205  0.264629376  0.88903670
## funded_amnt                    0.096691869  0.264523536  0.88919047
## int_rate                       0.417200278  0.032221701  0.20559968
## installment                    0.135016402  0.251375479  0.88741789
## sub_grade                     -0.417107617 -0.031354316 -0.20535564
## emp_length                     0.059197829  0.145857249  0.11810318
## annual_inc                     0.058077989  0.365027127  0.44665592
## loan_status                             NA           NA          NA
## dti                            0.223969860  0.245006548  0.03176317
## delinq_2yrs                   -0.014916689  0.153290262  0.03311936
## inq_last_6mths                -0.083647643  0.138875917 -0.01019020
## mths_since_last_delinq         0.020364903 -0.063011819 -0.01352599
## open_acc                      -0.102830148  0.664547280  0.17914195
## pub_rec                       -0.037853828 -0.007771288 -0.03620753
## revol_bal                      0.370504423  0.345718004  0.42368239
## revol_util                     1.000000000 -0.052303697  0.09929284
## total_acc                     -0.052303697  1.000000000  0.23742827
## total_pymnt                    0.099292835  0.237428267  1.00000000
## acc_now_delinq                -0.007611036  0.015189135  0.01046365
## tot_coll_amt                  -0.076956058 -0.023481482 -0.07997797
```

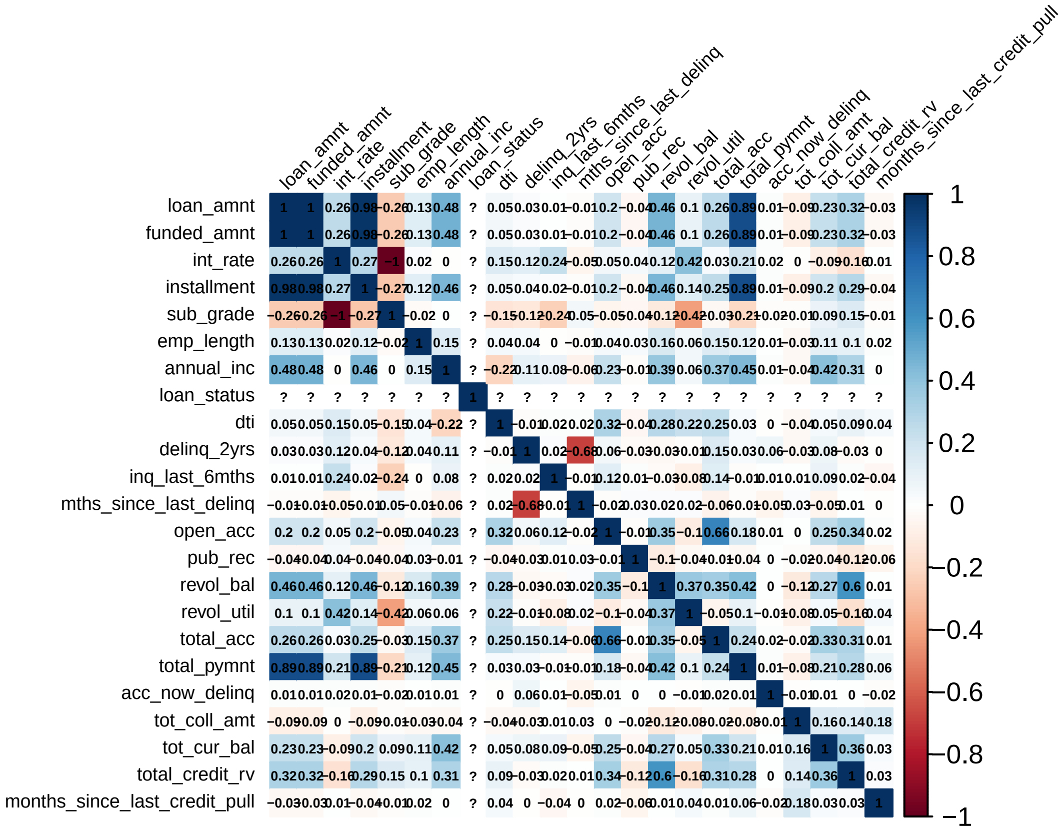
```
## tot_cur_bal                  0.045726293  0.334744825  0.20571044
## total_credit_rv             -0.163634151  0.307789029  0.28262227
## months_since_last_credit_pull  0.044781591  0.010068529  0.06129657
##                             acc_now_delinq  tot_coll_amt  tot_cur_bal
## loan_amnt                     0.0120629981 -0.0860990498  0.234730177
## funded_amnt                   0.0120772191 -0.0866412711  0.234668047
## int_rate                      0.0174218727 -0.0011500415 -0.088740691
## installment                   0.0126783960 -0.0857383722  0.203715697
## sub_grade                    -0.0171283966 -0.0118974768  0.085142202
## emp_length                    0.0075155907 -0.0318712769  0.110686482
## annual_inc                    0.0121654128 -0.0372971236  0.416550499
## loan_status                             NA            NA           NA
## dti                          -0.0013890791 -0.0379670237  0.045662002
## delinq_2yrs                   0.0643313020 -0.0333530470  0.078617317
## inq_last_6mths                0.0055736012  0.0052686262  0.087006958
## mths_since_last_delinq       -0.0504790514  0.0268863447 -0.051409789
## open_acc                      0.0144294125  0.0001339259  0.250828179
## pub_rec                       0.0043762143 -0.0180363394 -0.041704689
## revol_bal                     0.0030151926 -0.1157179474  0.272826793
## revol_util                   -0.0076110362 -0.0769560581  0.045726293
## total_acc                     0.0151891355 -0.0234814824  0.334744825
## total_pymnt                   0.0104636503 -0.0799779671  0.205710435
## acc_now_delinq                1.0000000000 -0.0062992944  0.008216218
## tot_coll_amt                 -0.0062992944  1.0000000000  0.156346028
## tot_cur_bal                   0.0082162184  0.1563460283  1.000000000
## total_credit_rv               0.0004949821  0.1402553525  0.360097945
## months_since_last_credit_pull -0.0158896199  0.1760207684  0.030941920
##                             total_credit_rv months_since_last_credit_pull
## loan_amnt                      0.3196231195                  -0.030883334
## funded_amnt                    0.3195674182                  -0.030957411
## int_rate                      -0.1560204511                   0.009450027
## installment                    0.2889137543                  -0.039236984
## sub_grade                      0.1517414497                  -0.013954288
## emp_length                     0.1006819402                   0.018327006
## annual_inc                     0.3075063632                   0.003045817
## loan_status                              NA                            NA
## dti                            0.0928815563                   0.040799453
## delinq_2yrs                   -0.0296443981                  -0.001667593
## inq_last_6mths                 0.0152099191                  -0.040420955
## mths_since_last_delinq         0.0052596179                  -0.002541929
## open_acc                       0.3385360824                   0.018658514
## pub_rec                       -0.1176113738                  -0.058419699
## revol_bal                      0.5963826701                   0.008105816
## revol_util                    -0.1636341513                   0.044781591
## total_acc                      0.3077890286                   0.010068529
## total_pymnt                    0.2826222750                   0.061296574
## acc_now_delinq                 0.0004949821                  -0.015889620
## tot_coll_amt                   0.1402553525                   0.176020768
## tot_cur_bal                    0.3600979446                   0.030941920
## total_credit_rv                1.0000000000                   0.029383905
## months_since_last_credit_pull  0.0293839054                   1.000000000
```

```r
# Visualize Spearman correlation matrix using corrplot
corrplot(correlation_matrix_spearman, method = "color")
```




```
corrplot(correlation_matrix_spearman, method = "color", addCoef.col = "black",
         tl.col = "black", tl.srt = 45, tl.cex = 0.6, number.cex = 0.4, mar = c(0,0,1,0),width = 30, he
```

```
## Warning in text.default(PosNA[, 1], PosNA[, 2], font = number.font, col =
## na.label.col, : "width" is not a graphical parameter

## Warning in text.default(PosNA[, 1], PosNA[, 2], font = number.font, col =
## na.label.col, : "height" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "width" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "height" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "width" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "height" is not a graphical parameter

## Warning in title(title, ...): "width" is not a graphical parameter

## Warning in title(title, ...): "height" is not a graphical parameter
```



Based on the correlation plot, we drop 5 columns which have value more than 0.6.

```
# Drop the columns that are highly correlated
drop_col_d3 <- c('installment', 'mths_since_last_delinq', 'total_pymnt', 'open_acc', 'loan_amnt') # NOT
df4 <- df3 %>% select(-drop_col_d3)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(drop_col_d3)
##
##   # Now:
##   data %>% select(all_of(drop_col_d3))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# Check for correlation after removing the correlated columns
numeric_sample <- as.data.frame(lapply(df4, as.numeric))
correlation_matrix_spearman <- cor(numeric_sample, method = "spearman")
print(correlation_matrix_spearman)
```

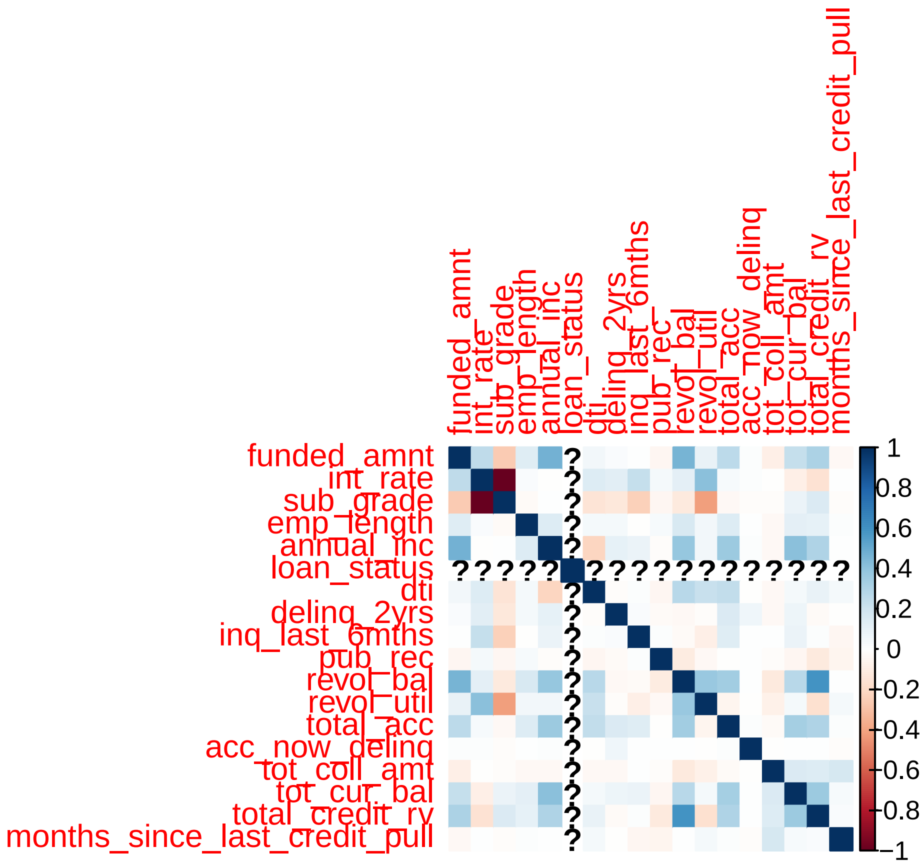
```
##                              funded_amnt     int_rate    sub_grade
## funded_amnt                   1.00000000  0.258260325 -0.258276552
## int_rate                      0.25826032  1.000000000 -0.998362617
## sub_grade                    -0.25827655 -0.998362617  1.000000000
```

```
## emp_length                     0.13145462  0.024537614 -0.023839540
## annual_inc                     0.47706662 -0.004011929  0.004428529
## loan_status                            NA           NA           NA
## dti                            0.05251425  0.147509003 -0.146602918
## delinq_2yrs                    0.02555914  0.122392140 -0.121772988
## inq_last_6mths                 0.00804998  0.237817181 -0.238142752
## pub_rec                       -0.04309539  0.042614997 -0.041428978
## revol_bal                      0.46025613  0.118222107 -0.117103726
## revol_util                     0.09669187  0.417200278 -0.417107617
## total_acc                      0.26452354  0.032221701 -0.031354316
## acc_now_delinq                 0.01207722  0.017421873 -0.017128397
## tot_coll_amt                  -0.08664127 -0.001150042 -0.011897477
## tot_cur_bal                    0.23466805 -0.088740691  0.085142202
## total_credit_rv                0.31956742 -0.156020451  0.151741450
## months_since_last_credit_pull -0.03095741  0.009450027 -0.013954288
##                                 emp_length   annual_inc loan_status
## funded_amnt                    0.131454624  0.477066621          NA
## int_rate                       0.024537614 -0.004011929          NA
## sub_grade                     -0.023839540  0.004428529          NA
## emp_length                     1.000000000  0.145214704          NA
## annual_inc                     0.145214704  1.000000000          NA
## loan_status                             NA           NA           1
## dti                            0.043531048 -0.218989268          NA
## delinq_2yrs                    0.044827614  0.109225753          NA
## inq_last_6mths                -0.002882133  0.080137232          NA
## pub_rec                        0.031067268 -0.014905473          NA
## revol_bal                      0.163467914  0.387595034          NA
## revol_util                     0.059197829  0.058077989          NA
## total_acc                      0.145857249  0.365027127          NA
## acc_now_delinq                 0.007515591  0.012165413          NA
## tot_coll_amt                  -0.031871277 -0.037297124          NA
## tot_cur_bal                    0.110686482  0.416550499          NA
## total_credit_rv                0.100681940  0.307506363          NA
## months_since_last_credit_pull  0.018327006  0.003045817          NA
##                                        dti  delinq_2yrs inq_last_6mths
## funded_amnt                    0.052514251  0.025559140    0.008049980
## int_rate                       0.147509003  0.122392140    0.237817181
## sub_grade                     -0.146602918 -0.121772988   -0.238142752
## emp_length                     0.043531048  0.044827614   -0.002882133
## annual_inc                    -0.218989268  0.109225753    0.080137232
## loan_status                             NA           NA             NA
## dti                            1.000000000 -0.012183737    0.019430352
## delinq_2yrs                   -0.012183737  1.000000000    0.020957463
## inq_last_6mths                 0.019430352  0.020957463    1.000000000
## pub_rec                       -0.040174812 -0.029661323    0.013007007
## revol_bal                      0.275159484 -0.032236212   -0.029971250
## revol_util                     0.223969860 -0.014916689   -0.083647643
## total_acc                      0.245006548  0.153290262    0.138875917
## acc_now_delinq                -0.001389079  0.064331302    0.005573601
## tot_coll_amt                  -0.037967024 -0.033353047    0.005268626
## tot_cur_bal                    0.045662002  0.078617317    0.087006958
## total_credit_rv                0.092881556 -0.029644398    0.015209919
## months_since_last_credit_pull  0.040799453 -0.001667593   -0.040420955
##                                    pub_rec    revol_bal   revol_util
```

```
## funded_amnt                  -0.043095391  0.460256131  0.096691869
## int_rate                      0.042614997  0.118222107  0.417200278
## sub_grade                    -0.041428978 -0.117103726 -0.417107617
## emp_length                    0.031067268  0.163467914  0.059197829
## annual_inc                   -0.014905473  0.387595034  0.058077989
## loan_status                            NA           NA           NA
## dti                          -0.040174812  0.275159484  0.223969860
## delinq_2yrs                  -0.029661323 -0.032236212 -0.014916689
## inq_last_6mths                0.013007007 -0.029971250 -0.083647643
## pub_rec                       1.000000000 -0.104688974 -0.037853828
## revol_bal                    -0.104688974  1.000000000  0.370504423
## revol_util                   -0.037853828  0.370504423  1.000000000
## total_acc                    -0.007771288  0.345718004 -0.052303697
## acc_now_delinq                0.004376214  0.003015193 -0.007611036
## tot_coll_amt                 -0.018036339 -0.115717947 -0.076956058
## tot_cur_bal                  -0.041704689  0.272826793  0.045726293
## total_credit_rv              -0.117611374  0.596382670 -0.163634151
## months_since_last_credit_pull -0.058419699  0.008105816  0.044781591
##                                  total_acc acc_now_delinq tot_coll_amt
## funded_amnt                   0.264523536   0.0120772191 -0.086641271
## int_rate                      0.032221701   0.0174218727 -0.001150042
## sub_grade                    -0.031354316  -0.0171283966 -0.011897477
## emp_length                    0.145857249   0.0075155907 -0.031871277
## annual_inc                    0.365027127   0.0121654128 -0.037297124
## loan_status                            NA             NA           NA
## dti                           0.245006548  -0.0013890791 -0.037967024
## delinq_2yrs                   0.153290262   0.0643313020 -0.033353047
## inq_last_6mths                0.138875917   0.0055736012  0.005268626
## pub_rec                      -0.007771288   0.0043762143 -0.018036339
## revol_bal                     0.345718004   0.0030151926 -0.115717947
## revol_util                   -0.052303697  -0.0076110362 -0.076956058
## total_acc                     1.000000000   0.0151891355 -0.023481482
## acc_now_delinq                0.015189135   1.0000000000 -0.006299294
## tot_coll_amt                 -0.023481482  -0.0062992944  1.000000000
## tot_cur_bal                   0.334744825   0.0082162184  0.156346028
## total_credit_rv               0.307789029   0.0004949821  0.140255353
## months_since_last_credit_pull  0.010068529  -0.0158896199  0.176020768
##                               tot_cur_bal total_credit_rv
## funded_amnt                   0.234668047    0.3195674182
## int_rate                     -0.088740691   -0.1560204511
## sub_grade                     0.085142202    0.1517414497
## emp_length                    0.110686482    0.1006819402
## annual_inc                    0.416550499    0.3075063632
## loan_status                            NA              NA
## dti                           0.045662002    0.0928815563
## delinq_2yrs                   0.078617317   -0.0296443981
## inq_last_6mths                0.087006958    0.0152099191
## pub_rec                      -0.041704689   -0.1176113738
## revol_bal                     0.272826793    0.5963826701
## revol_util                    0.045726293   -0.1636341513
## total_acc                     0.334744825    0.3077890286
## acc_now_delinq                0.008216218    0.0004949821
## tot_coll_amt                  0.156346028    0.1402553525
## tot_cur_bal                   1.000000000    0.3600979446
```

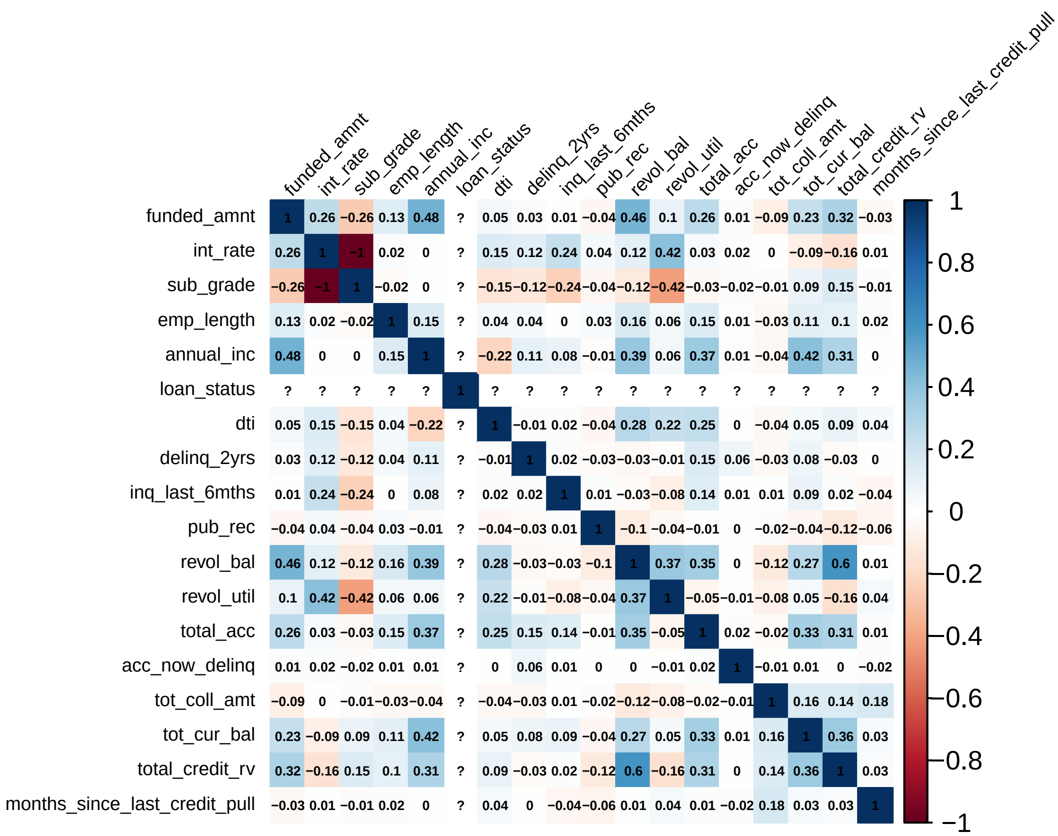
```
## total_credit_rv                    0.360097945    1.0000000000
## months_since_last_credit_pull  0.030941920    0.0293839054
##                                  months_since_last_credit_pull
## funded_amnt                                       -0.030957411
## int_rate                                           0.009450027
## sub_grade                                         -0.013954288
## emp_length                                         0.018327006
## annual_inc                                         0.003045817
## loan_status                                                 NA
## dti                                                0.040799453
## delinq_2yrs                                       -0.001667593
## inq_last_6mths                                    -0.040420955
## pub_rec                                           -0.058419699
## revol_bal                                          0.008105816
## revol_util                                         0.044781591
## total_acc                                          0.010068529
## acc_now_delinq                                    -0.015889620
## tot_coll_amt                                       0.176020768
## tot_cur_bal                                        0.030941920
## total_credit_rv                                    0.029383905
## months_since_last_credit_pull                      1.000000000
```

```
# Visualize Spearman correlation matrix using corrplot
corrplot(correlation_matrix_spearman, method = "color")
```




```
corrplot(correlation_matrix_spearman, method = "color", addCoef.col = "black",
         tl.col = "black", tl.srt = 45, tl.cex = 0.6,number.cex = 0.4, mar = c(0,0,1,0), width = 30, hei
```

```
## Warning in text.default(PosNA[, 1], PosNA[, 2], font = number.font, col =
## na.label.col, : "width" is not a graphical parameter

## Warning in text.default(PosNA[, 1], PosNA[, 2], font = number.font, col =
## na.label.col, : "height" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "width" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "height" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "width" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "height" is not a graphical parameter

## Warning in title(title, ...): "width" is not a graphical parameter

## Warning in title(title, ...): "height" is not a graphical parameter
```



Sampling

```
# Make a sample 600
set.seed(100)
sample <- sample_n(df4, 600)
head(sample)
```

```
## # A tibble: 6 x 18
##   funded_amnt int_rate sub_grade emp_length annual_inc loan_status   dti
##         <dbl>    <dbl>     <dbl>      <dbl>      <dbl>       <dbl> <dbl>
## 1       30000     24.7         5          7      93600           1  8.1
## 2       10000     10.2        30          5      73509           5  4.75
## 3       12000     18.5        19          9      56478.          5 21.5
## 4        7575     16.3        22          5      48000           5 28.3
## 5       20000      7.9        32          2     400000           5  5.5
## 6        9000     18.8        18          1      27000           5 26.9
## # i 11 more variables: delinq_2yrs <dbl>, inq_last_6mths <dbl>, pub_rec <dbl>,
## #   revol_bal <dbl>, revol_util <dbl>, total_acc <dbl>, acc_now_delinq <dbl>,
## #   tot_coll_amt <dbl>, tot_cur_bal <dbl>, total_credit_rv <dbl>,
## #   months_since_last_credit_pull <dbl>
```

Normalization

```r
#Normalize the data
sample_norm <- as.data.frame(scale(sample))
summary(sample_norm)
```

```
##   funded_amnt        int_rate          sub_grade         emp_length
##  Min.   :-1.6262   Min.   :-1.93058   Min.   :-3.0952   Min.   :-1.57598
##  1st Qu.:-0.7707   1st Qu.:-0.50325   1st Qu.:-0.5854   1st Qu.:-0.97213
##  Median :-0.2818   Median :-0.04153   Median : 0.2512   Median :-0.06636
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.6959   3rd Qu.: 0.70378   3rd Qu.: 0.5700   3rd Qu.: 1.14134
##  Max.   : 2.5291   Max.   : 2.44518   Max.   : 1.6854   Max.   : 1.14134
##    annual_inc       loan_status          dti            delinq_2yrs
##  Min.   :-1.4325   Min.   :-2.3712   Min.   :-2.13399   Min.   :-0.4162
##  1st Qu.:-0.6194   1st Qu.: 0.3043   1st Qu.:-0.75193   1st Qu.:-0.4162
##  Median :-0.2806   Median : 0.3043   Median : 0.02542   Median :-0.4162
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.3970   3rd Qu.: 0.3043   3rd Qu.: 0.76663   3rd Qu.:-0.4162
##  Max.   : 7.3988   Max.   : 0.9732   Max.   : 2.27260   Max.   : 6.4726
##  inq_last_6mths       pub_rec          revol_bal          revol_util
##  Min.   :-0.8030   Min.   :-0.2365   Min.   :-1.0884   Min.   :-2.4767
##  1st Qu.:-0.8030   1st Qu.:-0.2365   1st Qu.:-0.5812   1st Qu.:-0.6585
##  Median : 0.1664   Median :-0.2365   Median :-0.2283   Median : 0.1176
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1664   3rd Qu.:-0.2365   3rd Qu.: 0.2797   3rd Qu.: 0.7930
##  Max.   : 6.9527   Max.   :11.2699   Max.   :10.6803   Max.   : 1.7900
##    total_acc       acc_now_delinq      tot_coll_amt       tot_cur_bal
##  Min.   :-1.7595   Min.   :-0.04083   Min.   :-0.2473   Min.   :-1.05563
##  1st Qu.:-0.7185   1st Qu.:-0.04083   1st Qu.:-0.2473   1st Qu.:-0.73940
##  Median :-0.1979   Median :-0.04083   Median :-0.2473   Median :-0.06167
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.5828   3rd Qu.:-0.04083   3rd Qu.: 0.0525   3rd Qu.: 0.18064
##  Max.   : 3.3589   Max.   :24.45407   Max.   :12.5347   Max.   : 8.54562
##  total_credit_rv    months_since_last_credit_pull
##  Min.   :-1.32156   Min.   :-3.4111
##  1st Qu.:-0.53303   1st Qu.:-0.2793
##  Median :-0.02063   Median : 0.3918
##  Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.04257   3rd Qu.: 0.6155
```

```
##  Max.   :11.21531   Max.   : 1.1748
```

Multivariate Outliers

For sample:

```
Maha2 <- mahalanobis(sample, colMeans(sample), cov(sample))
print(Maha2) # prints Mahalanobis distance
```

```
##   [1]  37.034998  10.693559  14.997517   8.527786  80.756147   9.265092
##   [7]  12.197204   8.252436  12.533861   4.524654  19.957646  52.916000
##  [13]   7.674442  14.572603  18.517582  20.080699   7.730549  10.738028
##  [19]   8.359462   9.546727  13.414269  14.385267   9.083420  12.719638
##  [25]   6.377218  13.568849  13.031547  21.148305  11.378899  16.473622
##  [31]   6.360213   4.960951  14.509692   8.119099  16.186442  26.509636
##  [37]  28.579450   8.510023  14.384230  13.316746   8.195695   7.176494
##  [43]  12.363370  12.024505  10.460710  10.597282  15.520231  12.844790
##  [49]  20.644731  22.307848  20.608471  16.023247  14.450016  14.201850
##  [55] 135.117000  29.749692   6.568593  15.414380 598.001667   9.462104
##  [61]  20.346382   9.225843  34.380693   6.248870   5.273124  12.809725
##  [67]   5.531170  14.638146  28.837006   6.257398  31.343027  16.554337
##  [73]   6.316695  12.650531  48.570123   9.680824  21.122559   6.836101
##  [79]  11.997805  10.577028  17.351787   5.947491  10.799678   7.641882
##  [85]  30.700721  12.081487   8.109947   5.068778   7.823337  16.231118
##  [91]  15.339909  14.482182  12.472828  15.295217  19.170484  13.867215
##  [97]  19.444267   5.041763   5.167447  11.680120   7.671392 113.554351
## [103]   8.365672  18.887638   7.239966  26.167310   9.716957   8.693495
## [109]  22.717127   4.099142  13.035979  15.684188  13.922343   5.198144
## [115]  14.225655  11.117848  29.736784  15.695101  20.285691  17.541348
## [121]   8.092098   7.184650  20.532191  16.047267   7.086412  15.174877
## [127]  26.423230   9.189180  41.829165  18.454227  33.225818  21.575573
## [133]   9.256436  33.402406  11.077504  11.647322  11.252277  92.890926
## [139]   8.871205  14.570309  15.175511  12.385542  10.303151   8.465933
## [145]   3.425055  19.242456   7.307791   7.857081  37.826527  19.017157
## [151]   8.463174  11.830742  13.925185  15.429823   3.732565   9.106003
## [157]   9.360838 164.067584  14.076664   8.852594   5.207806  15.733088
## [163]   9.179983 163.297363   9.065949   7.486217  14.343143   5.393686
## [169]   7.130033   9.319984  13.479165  16.978066  19.891321  13.150807
## [175]   8.844655  20.372671  11.499629  15.573178   6.946929  26.843110
## [181]  28.516543  20.000537  25.908423   6.908989  20.431796   7.729124
## [187]  13.467821  16.571892   8.312097  17.317521  12.514478  14.506888
## [193]   7.241783   9.682868  20.269358  23.806016   8.589280   5.657030
## [199]  27.932322   6.531665  10.557537  15.724767  10.054020   9.422339
## [205]  10.558131   6.000555  10.593595  20.245649  21.527510  14.853501
## [211]  20.583211  19.255171  21.945137   9.964298  11.100093  10.287888
## [217]  19.392641  12.409340   9.890558  14.397101  23.894755   8.542078
## [223]  11.194273  37.408875   6.154731   7.971493   7.019298  21.642507
## [229]  22.238675  29.502636   7.297881  13.208834  11.616542  17.249183
## [235]   5.277067   9.787693   6.162044   4.078751   8.854906  19.802126
## [241]   9.335069  11.880773   4.773648  33.585799  12.608753  10.827942
## [247]  34.243833  14.406595  14.319073  25.540375  19.652831  14.355286
## [253]  14.185087   9.742475   6.762928  13.612113   8.001528  12.728506
## [259]  10.103845  24.235936  15.755286   9.440230  12.031454   6.285570
## [265]  11.613605   6.262593   6.362121  11.681549  13.725351  23.111612
## [271]   5.229348   9.104849  10.086223   8.856789  24.176116  15.580343
```

```
## [277]  12.693685   9.231625  68.604823  65.599103  26.848681   3.252662
## [283]  20.642657  30.574182   6.839731  15.233023  21.878438   5.471036
## [289]  19.993626   8.334928  16.043189  16.059509   7.084737  17.865421
## [295]  18.321689  20.438910  17.918327  22.858635  14.620688  16.708472
## [301]  45.268848   5.950843   7.275510   5.721787  10.023542  16.930317
## [307]  17.717357  25.499906   8.387505   9.427812   4.307589  12.025544
## [313]  19.073049  36.851338   8.650580  13.092476  13.222956   8.619800
## [319]   9.735595  31.002791   5.433673   5.123694  24.387004  11.279424
## [325]  16.326377   7.525780  18.686447   5.417768  17.041580  18.735992
## [331]   6.769975  18.054567  20.520457   8.956023  12.615112   8.225811
## [337]   9.398065   8.555043   7.385768  10.886538  78.402916  11.934189
## [343]  23.080658  43.342987  23.876963  28.592785   4.389984  17.183905
## [349]  31.292741   7.167398  16.824370  17.061856  20.134768   7.520864
## [355]  10.203648  11.517310  19.089821  11.747872  18.582669   7.697211
## [361]  16.144104  10.743525  20.705543  18.374861  13.319138  17.752649
## [367]  15.613577  19.780380  23.462509  18.943015 112.111317  21.488369
## [373]  17.443479  21.386769  20.673648   8.138955  24.142816  21.151668
## [379]  10.666776  13.793784   8.060969   5.709738  12.668907  21.112367
## [385]  22.478911   6.681663   4.669648   7.481036   8.201071   8.595963
## [391]   6.797637   6.899705  17.618441  14.457245   3.846746  14.780882
## [397]  21.191372  31.116221  13.770063   5.894393   9.884094  10.127495
## [403]  12.139794  10.981096  25.806367   7.619974  20.549282  45.346507
## [409]   5.567934   9.849664  20.742672  29.291188   8.357447  11.724996
## [415]  15.015867 187.733687   4.763529   6.225488   7.510301  11.566402
## [421]   7.135478  47.938817   9.850347  19.971789   6.643761   8.681248
## [427]  16.088843  20.920516  13.071542  28.488528  13.397809  16.573479
## [433]  30.548758   8.923178  13.421814  26.937222  24.757294  10.632277
## [439]  27.265381   5.216298  31.716186  15.037759  14.466922  10.353404
## [445]  27.643331  37.232281  19.946907   7.371919  17.879232   3.939070
## [451]   5.507944  14.691027  10.722939  19.486111   5.460843  16.979271
## [457]   9.161195   5.436618  10.850797   6.704334  17.273060   3.173986
## [463]  30.722111  10.102338   7.808327   4.206789  25.553933  13.675778
## [469]   5.632781  14.102390  27.786736  33.168083   4.324713  20.680572
## [475]   5.536132   3.934961  11.615160   7.980761  50.282006  17.743169
## [481]  28.843127   6.359200   6.160086   7.839435   4.790889  30.231510
## [487]   4.582581  35.412641   6.783972  16.991592  10.784144   7.112661
## [493]  11.002544  22.967661   8.189683  15.442010   8.207424  16.377623
## [499]  11.354829  23.426865  21.993263   5.561990  18.015251   7.164837
## [505]  10.562285  12.148577   6.260604  21.625142  10.226138   9.655296
## [511]   5.784580  18.276872  11.874979   8.277124   4.334791  11.809486
## [517]  16.838261  19.730716   5.446694 130.698262  16.336225   9.926976
## [523]  18.994002   7.587520  11.909523   8.181730   5.857184   5.728321
## [529]  12.209085   9.182189  29.774993  31.323234   7.465483  12.560523
## [535]   7.181128   7.939060   9.583566   8.345070  14.967399   8.457626
## [541]  24.397507  13.844149  43.550966  32.757660   9.422248  15.880920
## [547]  16.864478  15.418763   7.662400  29.112836  13.909154   9.382958
## [553]  32.046071  37.706994  18.350264  17.441429   6.056836  10.181288
## [559]   5.610803  21.032899  13.145535  14.047849   9.556100  19.886734
## [565]   9.470502  12.938635   5.083757  50.242376 104.438972  16.332695
## [571]  49.721640   8.681907   5.222887   6.679638   7.489419  11.149366
## [577]  21.958482   8.870417   6.126303   9.231432  10.907845   4.685075
## [583]  14.975884  32.352130  19.191120   9.584436  21.916856  12.016613
## [589]   9.240481  11.944986  13.373336  13.876579   9.650087  19.142726
## [595]  36.285163  49.476413  65.432791  26.545701  21.236027  11.462335
```

```r
MahaPvalue2 <-pchisq(Maha2, df=10,lower.tail = FALSE) # prints the p-value for each Mahalanobis distanc
sample_maha2 <- cbind(sample, Maha2, MahaPvalue2)
sample_maha2 <- sample_maha2 %>% select(-acc_now_delinq)
sample_maha_updated2 <- sample_maha2 %>% filter(MahaPvalue2 > 0.001) # only keep the rows which p-value
sample_maha_outlier2 <- sample_maha2 %>% filter(MahaPvalue2 < 0.001)
sample_maha_outlier2 <- sample_maha_outlier2[-c(18:19)]
sample_maha_updated2 <- sample_maha_updated2[-c(18:19)]
sample_maha2 <- sample_maha2[-c(18:19)]
```

Export outliers as csv to investigate further:

```r
write.csv(sample_maha_outlier2, "maha outliers.csv",row.names = FALSE)
```

For normalised sample:

```r
Maha <- mahalanobis(sample_norm, colMeans(sample_norm), cov(sample_norm))
print(Maha)# prints Mahalanobis distance
```

```
##   [1]  37.034998  10.693559  14.997517   8.527786  80.756147   9.265092
##   [7]  12.197204   8.252436  12.533861   4.524654  19.957646  52.916000
##  [13]   7.674442  14.572603  18.517582  20.080699   7.730549  10.738028
##  [19]   8.359462   9.546727  13.414269  14.385267   9.083420  12.719638
##  [25]   6.377218  13.568849  13.031547  21.148305  11.378899  16.473622
##  [31]   6.360213   4.960951  14.509692   8.119099  16.186442  26.509636
##  [37]  28.579450   8.510023  14.384230  13.316746   8.195695   7.176494
##  [43]  12.363370  12.024505  10.460710  10.597282  15.520231  12.844790
##  [49]  20.644731  22.307848  20.608471  16.023247  14.450016  14.201850
##  [55] 135.117000  29.749692   6.568593  15.414380 598.001667   9.462104
##  [61]  20.346382   9.225843  34.380693   6.248870   5.273124  12.809725
##  [67]   5.531170  14.638146  28.837006   6.257398  31.343027  16.554337
##  [73]   6.316695  12.650531  48.570123   9.680824  21.122559   6.836101
##  [79]  11.997805  10.577028  17.351787   5.947491  10.799678   7.641882
##  [85]  30.700721  12.081487   8.109947   5.068778   7.823337  16.231118
##  [91]  15.339909  14.482182  12.472828  15.295217  19.170484  13.867215
##  [97]  19.444267   5.041763   5.167447  11.680120   7.671392 113.554351
## [103]   8.365672  18.887638   7.239966  26.167310   9.716957   8.693495
## [109]  22.717127   4.099142  13.035979  15.684188  13.922343   5.198144
## [115]  14.225655  11.117848  29.736784  15.695101  20.285691  17.541348
## [121]   8.092098   7.184650  20.532191  16.047267   7.086412  15.174877
## [127]  26.423230   9.189180  41.829165  18.454227  33.225818  21.575573
## [133]   9.256436  33.402406  11.077504  11.647322  11.252277  92.890926
## [139]   8.871205  14.570309  15.175511  12.385542  10.303151   8.465933
## [145]   3.425055  19.242456   7.307791   7.857081  37.826527  19.017157
## [151]   8.463174  11.830742  13.925185  15.429823   3.732565   9.106003
## [157]   9.360838 164.067584  14.076664   8.852594   5.207806  15.733088
## [163]   9.179983 163.297363   9.065949   7.486217  14.343143   5.393686
## [169]   7.130033   9.319984  13.479165  16.978066  19.891321  13.150807
## [175]   8.844655  20.372671  11.499629  15.573178   6.946929  26.843110
## [181]  28.516543  20.000537  25.908423   6.908989  20.431796   7.729124
## [187]  13.467821  16.571892   8.312097  17.317521  12.514478  14.506888
## [193]   7.241783   9.682868  20.269358  23.806016   8.589280   5.657030
## [199]  27.932322   6.531665  10.557537  15.724767  10.054020   9.422339
## [205]  10.558131   6.000555  10.593595  20.245649  21.527510  14.853501
## [211]  20.583211  19.255171  21.945137   9.964298  11.100093  10.287888
## [217]  19.392641  12.409340   9.890558  14.397101  23.894755   8.542078
```

```
## [223]  11.194273  37.408875   6.154731   7.971493   7.019298  21.642507
## [229]  22.238675  29.502636   7.297881  13.208834  11.616542  17.249183
## [235]   5.277067   9.787693   6.162044   4.078751   8.854906  19.802126
## [241]   9.335069  11.880773   4.773648  33.585799  12.608753  10.827942
## [247]  34.243833  14.406595  14.319073  25.540375  19.652831  14.355286
## [253]  14.185087   9.742475   6.762928  13.612113   8.001528  12.728506
## [259]  10.103845  24.235936  15.755286   9.440230  12.031454   6.285570
## [265]  11.613605   6.262593   6.362121  11.681549  13.725351  23.111612
## [271]   5.229348   9.104849  10.086223   8.856789  24.176116  15.580343
## [277]  12.693685   9.231625  68.604823  65.599103  26.848681   3.252662
## [283]  20.642657  30.574182   6.839731  15.233023  21.878438   5.471036
## [289]  19.993626   8.334928  16.043189  16.059509   7.084737  17.865421
## [295]  18.321689  20.438910  17.918327  22.858635  14.620688  16.708472
## [301]  45.268848   5.950843   7.275510   5.721787  10.023542  16.930317
## [307]  17.717357  25.499906   8.387505   9.427812   4.307589  12.025544
## [313]  19.073049  36.851338   8.650580  13.092476  13.222956   8.619800
## [319]   9.735595  31.002791   5.433673   5.123694  24.387004  11.279424
## [325]  16.326377   7.525780  18.686447   5.417768  17.041580  18.735992
## [331]   6.769975  18.054567  20.520457   8.956023  12.615112   8.225811
## [337]   9.398065   8.555043   7.385768  10.886538  78.402916  11.934189
## [343]  23.080658  43.342987  23.876963  28.592785   4.389984  17.183905
## [349]  31.292741   7.167398  16.824370  17.061856  20.134768   7.520864
## [355]  10.203648  11.517310  19.089821  11.747872  18.582669   7.697211
## [361]  16.144104  10.743525  20.705543  18.374861  13.319138  17.752649
## [367]  15.613577  19.780380  23.462509  18.943015 112.111317  21.488369
## [373]  17.443479  21.386769  20.673648   8.138955  24.142816  21.151668
## [379]  10.666776  13.793784   8.060969   5.709738  12.668907  21.112367
## [385]  22.478911   6.681663   4.669648   7.481036   8.201071   8.595963
## [391]   6.797637   6.899705  17.618441  14.457245   3.846746  14.780882
## [397]  21.191372  31.116221  13.770063   5.894393   9.884094  10.127495
## [403]  12.139794  10.981096  25.806367   7.619974  20.549282  45.346507
## [409]   5.567934   9.849664  20.742672  29.291188   8.357447  11.724996
## [415]  15.015867 187.733687   4.763529   6.225488   7.510301  11.566402
## [421]   7.135478  47.938817   9.850347  19.971789   6.643761   8.681248
## [427]  16.088843  20.920516  13.071542  28.488528  13.397809  16.573479
## [433]  30.548758   8.923178  13.421814  26.937222  24.757294  10.632277
## [439]  27.265381   5.216298  31.716186  15.037759  14.466922  10.353404
## [445]  27.643331  37.232281  19.946907   7.371919  17.879232   3.939070
## [451]   5.507944  14.691027  10.722939  19.486111   5.460843  16.979271
## [457]   9.161195   5.436618  10.850797   6.704334  17.273060   3.173986
## [463]  30.722111  10.102338   7.808327   4.206789  25.553933  13.675778
## [469]   5.632781  14.102390  27.786736  33.168083   4.324713  20.680572
## [475]   5.536132   3.934961  11.615160   7.980761  50.282006  17.743169
## [481]  28.843127   6.359200   6.160086   7.839435   4.790889  30.231510
## [487]   4.582581  35.412641   6.783972  16.991592  10.784144   7.112661
## [493]  11.002544  22.967661   8.189683  15.442010   8.207424  16.377623
## [499]  11.354829  23.426865  21.993263   5.561990  18.015251   7.164837
## [505]  10.562285  12.148577   6.260604  21.625142  10.226138   9.655296
## [511]   5.784580  18.276872  11.874979   8.277124   4.334791  11.809486
## [517]  16.838261  19.730716   5.446694 130.698262  16.336225   9.926976
## [523]  18.994002   7.587520  11.909523   8.181730   5.857184   5.728321
## [529]  12.209085   9.182189  29.774993  31.323234   7.465483  12.560523
## [535]   7.181128   7.939060   9.583566   8.345070  14.967399   8.457626
## [541]  24.397507  13.844149  43.550966  32.757660   9.422248  15.880920
```

```
## [547]  16.864478  15.418763   7.662400  29.112836  13.909154   9.382958
## [553]  32.046071  37.706994  18.350264  17.441429   6.056836  10.181288
## [559]   5.610803  21.032899  13.145535  14.047849   9.556100  19.886734
## [565]   9.470502  12.938635   5.083757  50.242376 104.438972  16.332695
## [571]  49.721640   8.681907   5.222887   6.679638   7.489419  11.149366
## [577]  21.958482   8.870417   6.126303   9.231432  10.907845   4.685075
## [583]  14.975884  32.352130  19.191120   9.584436  21.916856  12.016613
## [589]   9.240481  11.944986  13.373336  13.876579   9.650087  19.142726
## [595]  36.285163  49.476413  65.432791  26.545701  21.236027  11.462335
```

```
MahaPvalue <-pchisq(Maha, df=10,lower.tail = FALSE)# prints the p-value for each Mahalanobis distance
sample_maha <- cbind(sample_norm, Maha, MahaPvalue)
sample_maha <- sample_maha %>% select(-acc_now_delinq)
sample_maha_outlier <- sample_maha %>% filter(MahaPvalue < 0.001)
sample_maha_updated <- sample_maha %>% filter(MahaPvalue > 0.001)
sample_maha_outlier <- sample_maha_outlier[-c(18:19)]
sample_maha_updated <- sample_maha_updated[-c(18:19)]
sample_maha <-sample_maha[-c(18:19)]
```

For normalised sample:

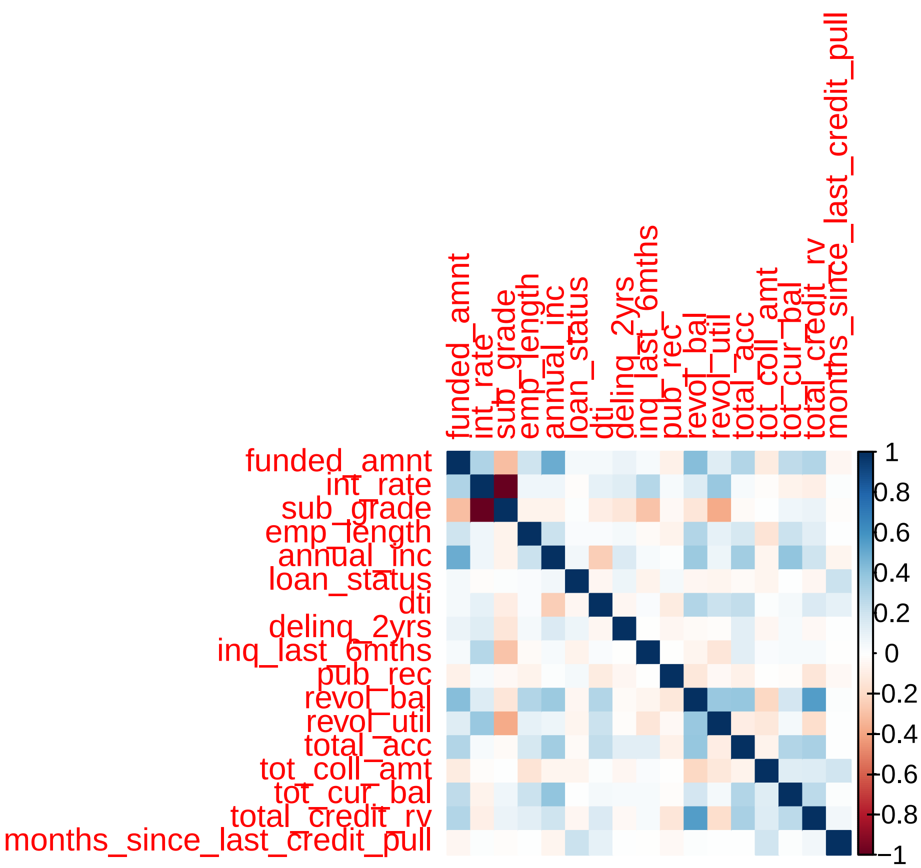
```
numeric_sample <- as.data.frame(lapply(sample_maha_updated, as.numeric))
correlation_matrix_spearman <- cor(numeric_sample, method = "spearman")
print(correlation_matrix_spearman)
```

```
##                              funded_amnt    int_rate     sub_grade
## funded_amnt                   1.00000000  0.30488068 -0.3035471489
## int_rate                      0.30488068  1.00000000 -0.9990692627
## sub_grade                    -0.30354715 -0.99906926  1.0000000000
## emp_length                    0.20193603  0.06254488 -0.0607630838
## annual_inc                    0.49347688  0.06712132 -0.0661686131
## loan_status                   0.04176071 -0.01333042  0.0148317725
## dti                           0.04605276  0.10097265 -0.0993360937
## delinq_2yrs                   0.08193364  0.13920683 -0.1367048323
## inq_last_6mths                0.03359265  0.28102850 -0.2804732632
## pub_rec                      -0.07579158  0.03417268 -0.0337213736
## revol_bal                     0.42809283  0.14386819 -0.1387802952
## revol_util                    0.13521098  0.37089535 -0.3737317290
## total_acc                     0.29149563  0.03256687 -0.0298578281
## tot_coll_amt                 -0.10034317 -0.01613846  0.0003222787
## tot_cur_bal                   0.25920024 -0.06445858  0.0604351574
## total_credit_rv               0.29304858 -0.08597750  0.0812234914
## months_since_last_credit_pull -0.04545931  0.01043475 -0.0159302344
##                                emp_length  annual_inc  loan_status         dti
## funded_amnt                   0.201936030  0.49347688  0.041760715  0.04605276
## int_rate                      0.062544877  0.06712132 -0.013330420  0.10097265
## sub_grade                    -0.060763084 -0.06616861  0.014831772 -0.09933609
## emp_length                    1.000000000  0.21254217  0.023652435  0.02777216
## annual_inc                    0.212542173  1.00000000  0.050133599 -0.24729327
## loan_status                   0.023652435  0.05013360  1.000000000 -0.04705744
## dti                           0.027772155 -0.24729327 -0.047057438  1.00000000
## delinq_2yrs                   0.047805449  0.15222337  0.071181358 -0.04835813
## inq_last_6mths               -0.021710588  0.03470918 -0.067210690  0.02507381
## pub_rec                      -0.065201054  0.01089335  0.044109173 -0.10971446
## revol_bal                     0.298225518  0.36316386 -0.042341483  0.29894731
```

```
## revol_util                   0.109617395  0.07733418 -0.052297587  0.21589184
## total_acc                    0.172227317  0.34268491 -0.028718354  0.24535524
## tot_coll_amt                -0.145223289 -0.05766885 -0.057578580  0.01498655
## tot_cur_bal                  0.213439716  0.39270751  0.004035506  0.04593813
## total_credit_rv              0.122587933  0.20608567 -0.041436811  0.15986872
## months_since_last_credit_pull  0.001868316 -0.05554885  0.216160209  0.10702256
##                              delinq_2yrs inq_last_6mths      pub_rec
## funded_amnt                  0.081933639    0.033592648 -0.075791578
## int_rate                     0.139206827    0.281028498  0.034172681
## sub_grade                   -0.136704832   -0.280473263 -0.033721374
## emp_length                   0.047805449   -0.021710588 -0.065201054
## annual_inc                   0.152223370    0.034709182  0.010893350
## loan_status                  0.071181358   -0.067210690  0.044109173
## dti                         -0.048358133    0.025073811 -0.109714458
## delinq_2yrs                  1.000000000   -0.006162875 -0.048303851
## inq_last_6mths              -0.006162875    1.000000000 -0.002871659
## pub_rec                     -0.048303851   -0.002871659  1.000000000
## revol_bal                   -0.022811029   -0.050306068 -0.128114767
## revol_util                  -0.015530886   -0.131177862 -0.039724339
## total_acc                    0.129783854    0.122257740 -0.079057893
## tot_coll_amt                -0.040439624    0.025005821 -0.007484896
## tot_cur_bal                  0.031292980    0.035519039 -0.011984924
## total_credit_rv             -0.032915779    0.033685994 -0.130147416
## months_since_last_credit_pull  0.007583384   -0.008138092 -0.030794733
##                                revol_bal  revol_util    total_acc
## funded_amnt                  0.42809283  0.135210978  0.291495632
## int_rate                     0.14386819  0.370895347  0.032566865
## sub_grade                   -0.13878030 -0.373731729 -0.029857828
## emp_length                   0.29822552  0.109617395  0.172227317
## annual_inc                   0.36316386  0.077334179  0.342684908
## loan_status                 -0.04234148 -0.052297587 -0.028718354
## dti                          0.29894731  0.215891836  0.245355236
## delinq_2yrs                 -0.02281103 -0.015530886  0.129783854
## inq_last_6mths              -0.05030607 -0.131177862  0.122257740
## pub_rec                     -0.12811477 -0.039724339 -0.079057893
## revol_bal                    1.00000000  0.374731233  0.384873736
## revol_util                   0.37473123  1.000000000 -0.096559498
## total_acc                    0.38487374 -0.096559498  1.000000000
## tot_coll_amt                -0.20903779 -0.125250616 -0.063396035
## tot_cur_bal                  0.18986986  0.042127534  0.295703766
## total_credit_rv              0.55012347 -0.174500974  0.326670352
## months_since_last_credit_pull  0.01666952  0.006141489  0.007006246
##                              tot_coll_amt  tot_cur_bal total_credit_rv
## funded_amnt                 -0.1003431666  0.259200242      0.29304858
## int_rate                    -0.0161384631 -0.064458583     -0.08597750
## sub_grade                    0.0003222787  0.060435157      0.08122349
## emp_length                  -0.1452232890  0.213439716      0.12258793
## annual_inc                  -0.0576688532  0.392707509      0.20608567
## loan_status                 -0.0575785795  0.004035506     -0.04143681
## dti                          0.0149865464  0.045938125      0.15986872
## delinq_2yrs                 -0.0404396242  0.031292980     -0.03291578
## inq_last_6mths               0.0250058211  0.035519039      0.03368599
## pub_rec                     -0.0074848959 -0.011984924     -0.13014742
## revol_bal                   -0.2090377910  0.189869865      0.55012347
```

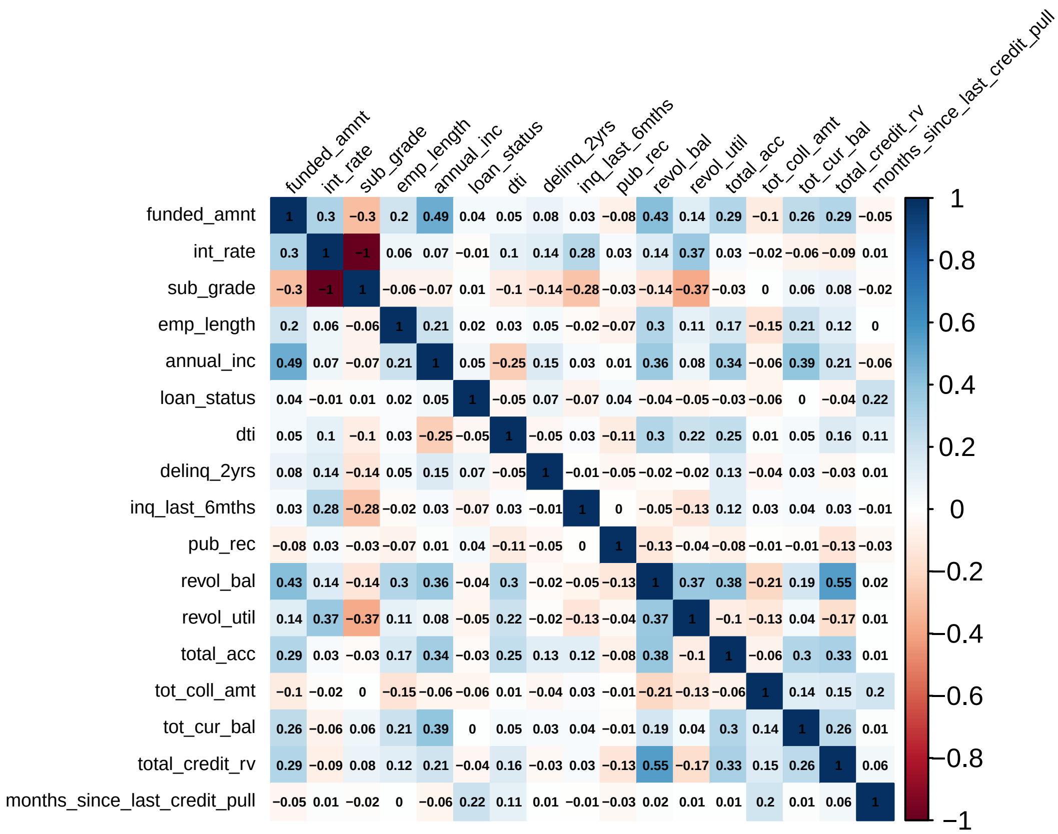
```
## revol_util                    -0.1252506155  0.042127534     -0.17450097
## total_acc                     -0.0633960346  0.295703766      0.32667035
## tot_coll_amt                   1.0000000000  0.136165868      0.14829209
## tot_cur_bal                    0.1361658683  1.000000000      0.26303075
## total_credit_rv                0.1482920868  0.263030747      1.00000000
## months_since_last_credit_pull  0.1958148717  0.010217534      0.05899894
##                              months_since_last_credit_pull
## funded_amnt                                   -0.045459311
## int_rate                                       0.010434748
## sub_grade                                     -0.015930234
## emp_length                                     0.001868316
## annual_inc                                    -0.055548851
## loan_status                                    0.216160209
## dti                                            0.107022558
## delinq_2yrs                                    0.007583384
## inq_last_6mths                                -0.008138092
## pub_rec                                       -0.030794733
## revol_bal                                      0.016669522
## revol_util                                     0.006141489
## total_acc                                      0.007006246
## tot_coll_amt                                   0.195814872
## tot_cur_bal                                    0.010217534
## total_credit_rv                                0.058998941
## months_since_last_credit_pull                  1.000000000
```

```r
# Visualize Spearman correlation matrix using corrplot
corrplot(correlation_matrix_spearman, method = "color")
```



```r
corrplot(correlation_matrix_spearman, method = "color", addCoef.col = "black",
         tl.col = "black", tl.srt = 45, tl.cex = 0.6,number.cex = 0.4, mar = c(0,0,1,0), width = 30, hei
```

```
## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "width" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "height" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "width" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "height" is not a graphical parameter

## Warning in title(title, ...): "width" is not a graphical parameter

## Warning in title(title, ...): "height" is not a graphical parameter
```



For sample:

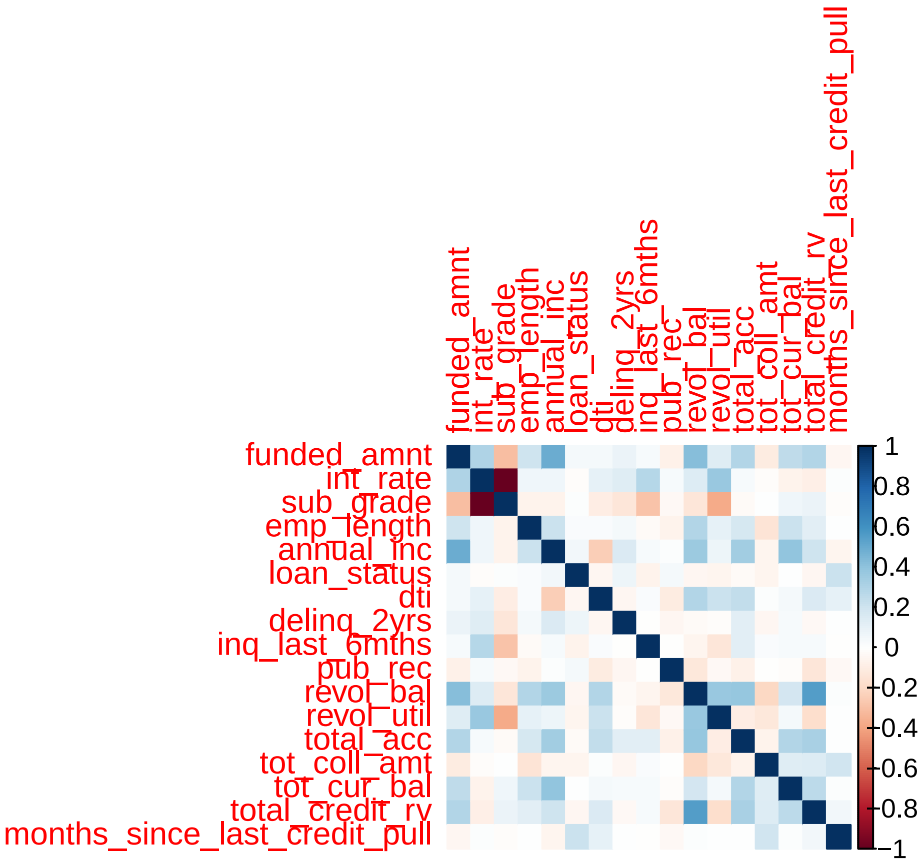
```r
numeric_sample <- as.data.frame(lapply(sample_maha_updated2, as.numeric))
correlation_matrix_spearman <- cor(numeric_sample, method = "spearman")
print(correlation_matrix_spearman)
```

```
##                               funded_amnt    int_rate     sub_grade
## funded_amnt                    1.00000000  0.30488068 -0.3035471489
## int_rate                       0.30488068  1.00000000 -0.9990692627
## sub_grade                     -0.30354715 -0.99906926  1.0000000000
```

```
## emp_length                    0.20193603  0.06254488 -0.0607630838
## annual_inc                    0.49347688  0.06712132 -0.0661686131
## loan_status                   0.04176071 -0.01333042  0.0148317725
## dti                           0.04605276  0.10097265 -0.0993360937
## delinq_2yrs                   0.08193364  0.13920683 -0.1367048323
## inq_last_6mths                0.03359265  0.28102850 -0.2804732632
## pub_rec                      -0.07579158  0.03417268 -0.0337213736
## revol_bal                     0.42809283  0.14386819 -0.1387802952
## revol_util                    0.13521098  0.37089535 -0.3737317290
## total_acc                     0.29149563  0.03256687 -0.0298578281
## tot_coll_amt                 -0.10034317 -0.01613846  0.0003222787
## tot_cur_bal                   0.25920024 -0.06445858  0.0604351574
## total_credit_rv               0.29304858 -0.08597750  0.0812234914
## months_since_last_credit_pull -0.04545931  0.01043475 -0.0159302344
##                                emp_length  annual_inc  loan_status         dti
## funded_amnt                   0.201936030  0.49347688  0.041760715  0.04605276
## int_rate                      0.062544877  0.06712132 -0.013330420  0.10097265
## sub_grade                    -0.060763084 -0.06616861  0.014831772 -0.09933609
## emp_length                    1.000000000  0.21254217  0.023652435  0.02777216
## annual_inc                    0.212542173  1.00000000  0.050133599 -0.24729327
## loan_status                   0.023652435  0.05013360  1.000000000 -0.04705744
## dti                           0.027772155 -0.24729327 -0.047057438  1.00000000
## delinq_2yrs                   0.047805449  0.15222337  0.071181358 -0.04835813
## inq_last_6mths               -0.021710588  0.03470918 -0.067210690  0.02507381
## pub_rec                      -0.065201054  0.01089335  0.044109173 -0.10971446
## revol_bal                     0.298225518  0.36316386 -0.042341483  0.29894731
## revol_util                    0.109617395  0.07733418 -0.052297587  0.21589184
## total_acc                     0.172227317  0.34268491 -0.028718354  0.24535524
## tot_coll_amt                 -0.145223289 -0.05766885 -0.057578580  0.01498655
## tot_cur_bal                   0.213439716  0.39270751  0.004035506  0.04593813
## total_credit_rv               0.122587933  0.20608567 -0.041436811  0.15986872
## months_since_last_credit_pull  0.001868316 -0.05554885  0.216160209  0.10702256
##                               delinq_2yrs inq_last_6mths      pub_rec
## funded_amnt                   0.081933639    0.033592648 -0.075791578
## int_rate                      0.139206827    0.281028498  0.034172681
## sub_grade                    -0.136704832   -0.280473263 -0.033721374
## emp_length                    0.047805449   -0.021710588 -0.065201054
## annual_inc                    0.152223370    0.034709182  0.010893350
## loan_status                   0.071181358   -0.067210690  0.044109173
## dti                          -0.048358133    0.025073811 -0.109714458
## delinq_2yrs                   1.000000000   -0.006162875 -0.048303851
## inq_last_6mths               -0.006162875    1.000000000 -0.002871659
## pub_rec                      -0.048303851   -0.002871659  1.000000000
## revol_bal                    -0.022811029   -0.050306068 -0.128114767
## revol_util                   -0.015530886   -0.131177862 -0.039724339
## total_acc                     0.129783854    0.122257740 -0.079057893
## tot_coll_amt                 -0.040439624    0.025005821 -0.007484896
## tot_cur_bal                   0.031292980    0.035519039 -0.011984924
## total_credit_rv              -0.032915779    0.033685994 -0.130147416
## months_since_last_credit_pull  0.007583384   -0.008138092 -0.030794733
##                                revol_bal  revol_util    total_acc
## funded_amnt                   0.42809283  0.135210978  0.291495632
## int_rate                      0.14386819  0.370895347  0.032566865
## sub_grade                    -0.13878030 -0.373731729 -0.029857828
```

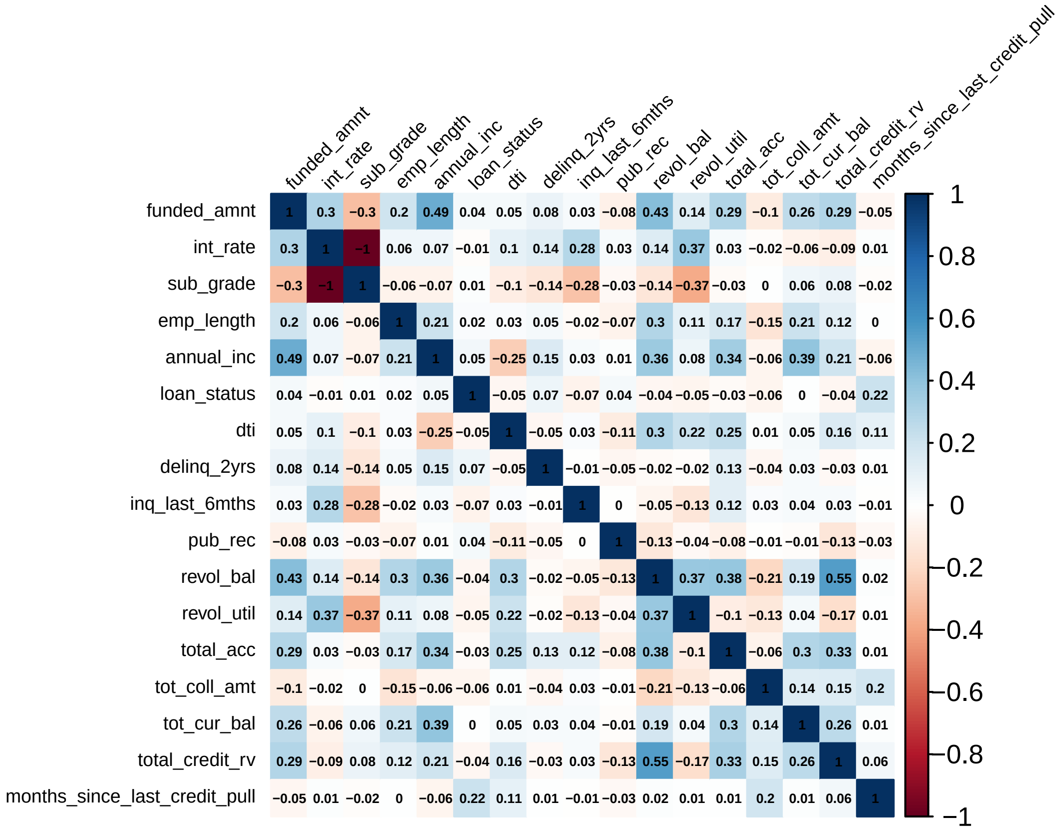
```
## emp_length                    0.29822552  0.109617395  0.172227317
## annual_inc                    0.36316386  0.077334179  0.342684908
## loan_status                  -0.04234148 -0.052297587 -0.028718354
## dti                           0.29894731  0.215891836  0.245355236
## delinq_2yrs                  -0.02281103 -0.015530886  0.129783854
## inq_last_6mths               -0.05030607 -0.131177862  0.122257740
## pub_rec                      -0.12811477 -0.039724339 -0.079057893
## revol_bal                     1.00000000  0.374731233  0.384873736
## revol_util                    0.37473123  1.000000000 -0.096559498
## total_acc                     0.38487374 -0.096559498  1.000000000
## tot_coll_amt                 -0.20903779 -0.125250616 -0.063396035
## tot_cur_bal                   0.18986986  0.042127534  0.295703766
## total_credit_rv               0.55012347 -0.174500974  0.326670352
## months_since_last_credit_pull  0.01666952  0.006141489  0.007006246
##                               tot_coll_amt  tot_cur_bal total_credit_rv
## funded_amnt                  -0.1003431666  0.259200242      0.29304858
## int_rate                     -0.0161384631 -0.064458583     -0.08597750
## sub_grade                     0.0003222787  0.060435157      0.08122349
## emp_length                   -0.1452232890  0.213439716      0.12258793
## annual_inc                   -0.0576688532  0.392707509      0.20608567
## loan_status                  -0.0575785795  0.004035506     -0.04143681
## dti                           0.0149865464  0.045938125      0.15986872
## delinq_2yrs                  -0.0404396242  0.031292980     -0.03291578
## inq_last_6mths                0.0250058211  0.035519039      0.03368599
## pub_rec                      -0.0074848959 -0.011984924     -0.13014742
## revol_bal                    -0.2090377910  0.189869865      0.55012347
## revol_util                   -0.1252506155  0.042127534     -0.17450097
## total_acc                    -0.0633960346  0.295703766      0.32667035
## tot_coll_amt                  1.0000000000  0.136165868      0.14829209
## tot_cur_bal                   0.1361658683  1.000000000      0.26303075
## total_credit_rv               0.1482920868  0.263030747      1.00000000
## months_since_last_credit_pull  0.1958148717  0.010217534      0.05899894
##                              months_since_last_credit_pull
## funded_amnt                                   -0.045459311
## int_rate                                       0.010434748
## sub_grade                                     -0.015930234
## emp_length                                     0.001868316
## annual_inc                                    -0.055548851
## loan_status                                    0.216160209
## dti                                            0.107022558
## delinq_2yrs                                    0.007583384
## inq_last_6mths                                -0.008138092
## pub_rec                                       -0.030794733
## revol_bal                                      0.016669522
## revol_util                                     0.006141489
## total_acc                                      0.007006246
## tot_coll_amt                                   0.195814872
## tot_cur_bal                                    0.010217534
## total_credit_rv                                0.058998941
## months_since_last_credit_pull                  1.000000000
```

```r
# Visualize Spearman correlation matrix using corrplot
corrplot(correlation_matrix_spearman, method = "color")
```




```r
corrplot(correlation_matrix_spearman, method = "color", addCoef.col = "black",
         tl.col = "black", tl.srt = 45, tl.cex = 0.6, number.cex = 0.4, mar = c(0,0,1,0), width = 30, he
```

```
## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "width" is not a graphical parameter

## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "height" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "width" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "height" is not a graphical parameter

## Warning in title(title, ...): "width" is not a graphical parameter

## Warning in title(title, ...): "height" is not a graphical parameter
```



We see that the correlation plots (after removal of outliers) for both unnormalised and normalised samples are the same, suggesting that the same outliers are removed irregardless of normalisation.

We decide to remove outliers because they only represent 9.67% of the sample size, and is a small and non-representative group of observations as compared to the others. As cluster analysis is sensitive to outliers, we decide to remove outliers to avoid skewing our clusters.

Modelling

PCA and FA

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

pairwise correlation

```r
#Choose the attribute that have pairwise correlation coefficients >0.3
sample_pca <- sample_maha_updated[, c(1, 2, 3, 5, 11, 12, 13, 15, 16 )]
lowerCor(sample_pca)
```

```
##                 fndd_ int_r sb_gr annl_ rvl_b rvl_t ttl_c tt_c_ ttl__
## funded_amnt      1.00
## int_rate         0.33  1.00
## sub_grade       -0.35 -0.99  1.00
## annual_inc       0.47  0.03 -0.05  1.00
## revol_bal        0.43  0.16 -0.16  0.36  1.00
## revol_util       0.14  0.41 -0.38  0.07  0.35  1.00
## total_acc        0.28  0.02 -0.02  0.32  0.33 -0.10  1.00
## tot_cur_bal      0.28 -0.06  0.04  0.45  0.22  0.06  0.26  1.00
```

```
## total_credit_rv  0.31 -0.06  0.04  0.21  0.65 -0.19  0.35  0.23  1.00
```

From the pairwise correlation coefficients of these attributes, we can see that many of them are above 0.3.

The Kaiser-Meyer-Olkin (KMO) test

```r
KMO(sample_pca)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = sample_pca)
## Overall MSA =  0.58
## MSA for each item =
##     funded_amnt        int_rate       sub_grade      annual_inc       revol_bal
##            0.86            0.56            0.56            0.68            0.52
##      revol_util       total_acc     tot_cur_bal total_credit_rv
##            0.41            0.84            0.68            0.47
```

KMO > 0.5 for most of these variables, then we conclude that they are highly correlated.

The Bartlett test

```r
cortest.bartlett(sample_pca)
```

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

## $chisq
## [1] 3327.47
##
## $p.value
## [1] 0
##
## $df
## [1] 36
```

The p value is < 0.05, hence there is sufficient for PCA.

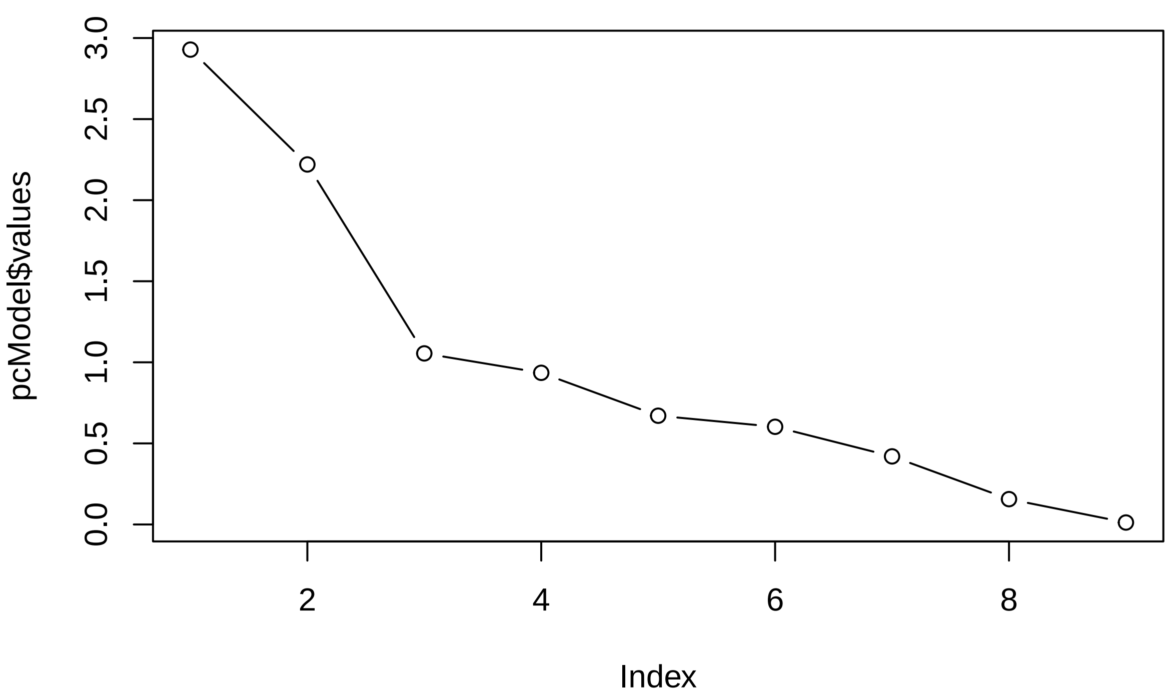
PCA

```r
# Do PCA with 8 principal components
pcModel<-principal(sample_pca, 8, rotate="none", weights=TRUE, scores=TRUE)
print.psych(pcModel, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = sample_pca, nfactors = 8, rotate = "none", scores = TRUE,
##     weights = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##                 item   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   h2
## funded_amnt        1  0.75                   -0.35        0.44       1.00
## revol_bal          5  0.75       -0.31  0.47                         1.00
## annual_inc         4  0.61  0.33  0.48              0.31 -0.41       1.00
## int_rate           2  0.53 -0.79                                     0.99
## sub_grade          3 -0.55  0.77                                     0.99
## tot_cur_bal        8  0.45  0.39  0.56             -0.56             1.00
## total_credit_rv    9  0.52  0.52 -0.55                               1.00
## revol_util         6  0.37 -0.52        0.65                         1.00
## total_acc          7  0.49  0.40       -0.35  0.66                   1.00
##                      u2 com
## funded_amnt     3.8e-07 2.6
```

```
## revol_bal       2.8e-06 2.6
## annual_inc      1.6e-06 4.2
## int_rate        6.1e-03 2.1
## sub_grade       5.9e-03 2.2
## tot_cur_bal     2.0e-06 3.8
## total_credit_rv 3.5e-06 4.1
## revol_util      1.8e-06 3.6
## total_acc       4.0e-07 3.5
##
##                        PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8
## SS loadings           2.93 2.22 1.05 0.94 0.67 0.60 0.42 0.16
## Proportion Var        0.33 0.25 0.12 0.10 0.07 0.07 0.05 0.02
## Cumulative Var        0.33 0.57 0.69 0.79 0.87 0.93 0.98 1.00
## Proportion Explained  0.33 0.25 0.12 0.10 0.07 0.07 0.05 0.02
## Cumulative Proportion 0.33 0.57 0.69 0.79 0.87 0.94 0.98 1.00
##
## Mean item complexity =  3.2
## Test of the hypothesis that 8 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0
##  with the empirical chi square  0.04  with prob <  NA
##
## Fit based upon off diagonal values = 1
```

```r
plot(pcModel$values, type="b")
```



From the graph, we sugest to choose 3 principal components

```r
pcModel3<-principal(sample_pca, 3, rotate="none")
print(pcModel3)
```

```
## Principal Components Analysis
## Call: principal(r = sample_pca, nfactors = 3, rotate = "none")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                   PC1   PC2   PC3   h2    u2 com
## funded_amnt      0.75  0.03  0.08 0.58 0.423 1.0
## int_rate         0.53 -0.79 -0.11 0.91 0.085 1.8
## sub_grade       -0.55  0.77  0.10 0.90 0.100 1.8
## annual_inc       0.61  0.33  0.48 0.71 0.289 2.5
## revol_bal        0.75  0.22 -0.31 0.70 0.297 1.5
## revol_util       0.37 -0.52  0.25 0.46 0.536 2.3
## total_acc        0.49  0.40 -0.14 0.42 0.580 2.1
## tot_cur_bal      0.45  0.39  0.56 0.67 0.330 2.7
## total_credit_rv  0.52  0.52 -0.55 0.84 0.156 3.0
##
##                        PC1  PC2  PC3
## SS loadings           2.93 2.22 1.05
## Proportion Var        0.33 0.25 0.12
## Cumulative Var        0.33 0.57 0.69
## Proportion Explained  0.47 0.36 0.17
## Cumulative Proportion 0.47 0.83 1.00
##
## Mean item complexity =  2.1
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.1
##  with the empirical chi square  413.58  with prob <  5e-81
##
## Fit based upon off diagonal values = 0.9
```

Three principal components can explain 69% of total variable

Factor Analysis

```r
# FA with 2 factor with pc
pcModel2o<-principal(sample_pca, 2, rotate="oblimin")
print.psych(pcModel2o, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = sample_pca, nfactors = 2, rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                 item   TC1   TC2   h2    u2 com
## revol_bal          5  0.74       0.61 0.390 1.1
## total_credit_rv    9  0.73       0.54 0.460 1.2
## annual_inc         4  0.69       0.48 0.521 1.0
## funded_amnt        1  0.64  0.33 0.57 0.429 1.5
## total_acc          7  0.63       0.40 0.601 1.1
## tot_cur_bal        8  0.60       0.35 0.646 1.1
## int_rate           2        0.95 0.90 0.096 1.0
## sub_grade          3       -0.94 0.89 0.110 1.0
## revol_util         6        0.63 0.40 0.597 1.0
##
```

```
##                          TC1  TC2
## SS loadings           2.73 2.42
## Proportion Var        0.30 0.27
## Cumulative Var        0.30 0.57
## Proportion Explained  0.53 0.47
## Cumulative Proportion 0.53 1.00
##
##  With component correlations of
##      TC1  TC2
## TC1 1.00 0.12
## TC2 0.12 1.00
##
## Mean item complexity =  1.1
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.12
##  with the empirical chi square  555.33  with prob <  1.3e-105
##
## Fit based upon off diagonal values = 0.86
```

```
# FA with 3 factor with pc
pcModel3o<-principal(sample_pca, 3, rotate="oblimin")
print.psych(pcModel3o, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = sample_pca, nfactors = 3, rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                 item   TC2   TC1   TC3   h2    u2 com
## int_rate           2  0.96              0.91 0.085 1.0
## sub_grade          3 -0.95              0.90 0.100 1.0
## revol_util         6  0.61              0.46 0.536 1.6
## total_credit_rv    9        0.94        0.84 0.156 1.0
## revol_bal          5        0.72        0.70 0.297 1.3
## total_acc          7        0.54        0.42 0.580 1.4
## tot_cur_bal        8              0.83 0.67 0.330 1.1
## annual_inc         4              0.82 0.71 0.289 1.0
## funded_amnt        1  0.37  0.33  0.44 0.58 0.423 2.8
##
##                        TC2  TC1  TC3
## SS loadings           2.44 1.96 1.80
## Proportion Var        0.27 0.22 0.20
## Cumulative Var        0.27 0.49 0.69
## Proportion Explained  0.39 0.32 0.29
## Cumulative Proportion 0.39 0.71 1.00
##
##  With component correlations of
##      TC2  TC1  TC3
## TC2 1.00 0.06 0.08
## TC1 0.06 1.00 0.31
## TC3 0.08 0.31 1.00
##
## Mean item complexity =  1.4
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.1
```

```
##  with the empirical chi square  413.58  with prob <  5e-81
##
## Fit based upon off diagonal values = 0.9
```

```r
# FA with 2 factor with ml
fa2<-(fa(sample_pca,2, fm="ml"))
print(fa2,cut=0.3,sort="TRUE")
```

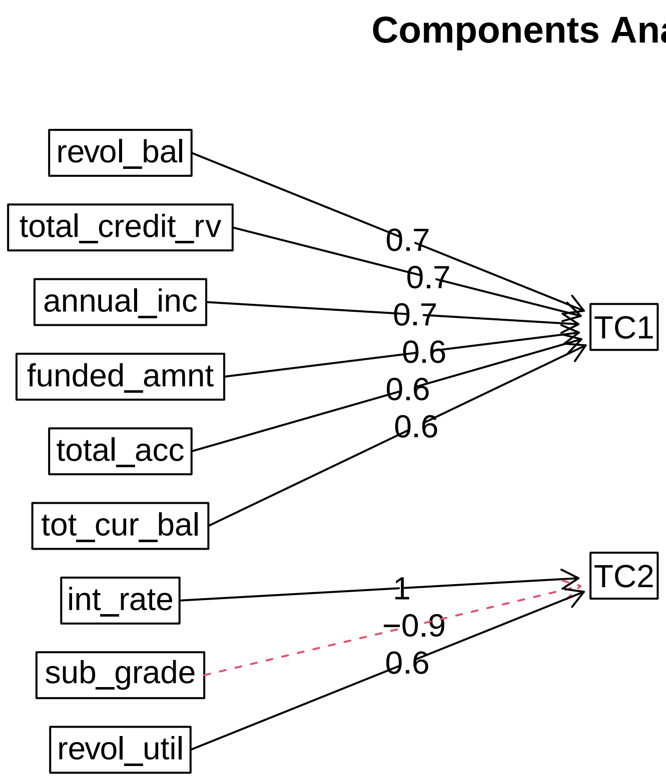
```
## Factor Analysis using method =  ml
## Call: fa(r = sample_pca, nfactors = 2, fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                 item   ML1   ML2   h2    u2 com
## int_rate           2  1.00       1.00 0.005 1.0
## sub_grade          3 -0.99       0.98 0.020 1.0
## revol_util         6  0.39       0.17 0.828 1.1
## revol_bal          5        0.79 0.65 0.352 1.0
## total_credit_rv    9        0.72 0.52 0.481 1.1
## funded_amnt        1        0.54 0.40 0.598 1.5
## annual_inc         4        0.52 0.27 0.734 1.0
## total_acc          7        0.48 0.23 0.774 1.0
## tot_cur_bal        8        0.41 0.17 0.831 1.1
##
##                        ML1  ML2
## SS loadings           2.25 2.13
## Proportion Var        0.25 0.24
## Cumulative Var        0.25 0.49
## Proportion Explained  0.51 0.49
## Cumulative Proportion 0.51 1.00
##
##  With factor correlations of
##      ML1  ML2
## ML1 1.00 0.13
## ML2 0.13 1.00
##
## Mean item complexity =  1.1
## Test of the hypothesis that 2 factors are sufficient.
##
## df null model =  36  with the objective function =  6.19 with Chi Square =  3327.47
## df of  the model are 19  and the objective function was  0.93
##
## The root mean square of the residuals (RMSR) is  0.09
## The df corrected root mean square of the residuals is  0.12
##
## The harmonic n.obs is  542 with the empirical chi square  315.76  with prob <  1.2e-55
## The total n.obs was  542  with Likelihood Chi Square =  500.48  with prob <  4.4e-94
##
## Tucker Lewis Index of factoring reliability =  0.722
## RMSEA index =  0.216  and the 90 % confidence intervals are  0.2 0.233
## BIC =  380.87
## Fit based upon off diagonal values = 0.92
## Measures of factor score adequacy
##                                                    ML1  ML2
## Correlation of (regression) scores with factors   1.00 0.90
## Multiple R square of scores with factors          1.00 0.81
## Minimum correlation of possible factor scores     0.99 0.62
```

```r
# FA with 2 factor with ml
pcModel3q<-principal(sample_pca, 3, rotate="quartimax")
print.psych(pcModel3q, cut=0.3, sort=TRUE)
```

```
## Principal Components Analysis
## Call: principal(r = sample_pca, nfactors = 3, rotate = "quartimax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                  item   RC2   RC1   RC3   h2    u2 com
## int_rate            2  0.95             0.91 0.085 1.0
## sub_grade           3 -0.95             0.90 0.100 1.0
## revol_util          6  0.62             0.46 0.536 1.4
## total_credit_rv     9        0.91       0.84 0.156 1.0
## revol_bal           5        0.75       0.70 0.297 1.5
## total_acc           7        0.58       0.42 0.580 1.5
## annual_inc          4              0.81 0.71 0.289 1.1
## tot_cur_bal         8              0.81 0.67 0.330 1.1
## funded_amnt         1  0.40  0.42  0.49 0.58 0.423 2.9
##
##                        RC2  RC1  RC3
## SS loadings           2.44 2.00 1.76
## Proportion Var        0.27 0.22 0.20
## Cumulative Var        0.27 0.49 0.69
## Proportion Explained  0.39 0.32 0.28
## Cumulative Proportion 0.39 0.72 1.00
##
## Mean item complexity =  1.4
## Test of the hypothesis that 3 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.1
##  with the empirical chi square  413.58  with prob <  5e-81
##
## Fit based upon off diagonal values = 0.9
```

Based on the results, we suggest to use 2 factor which can explain 58% of variable. Factor 1 can explain the `revol_bal`, `total_credit_rv`, `annual_inc`, `funded_amnt`, `tot_cur_bal`, `total_acc` adnd Factor 2 can explain about `int_rate`, `sub_grade`, `revol_util`.

```r
# Print the fac diagram
fa.diagram(pcModel2o)
```



Modelling

Define linkage methods

```
headTail(sample_maha_updated)
```

```
##     funded_amnt int_rate sub_grade emp_length annual_inc loan_status   dti
## 1         -0.53    -0.96      0.89      -0.37       0.02         0.3 -1.62
## 2         -0.28     0.99     -0.86       0.84      -0.36         0.3  0.55
## 3         -0.82     0.47     -0.39      -0.37      -0.55         0.3  1.42
## 4         -0.65     1.05     -1.02      -1.58      -1.03         0.3  1.24
## ...         ...      ...       ...        ...        ...         ...   ...
## 539       -0.89     0.01      0.09      -0.97      -0.42         0.3 -0.75
## 540        2.53     2.11     -2.46      -0.37       1.28         0.3 -0.59
## 541       -0.28    -0.96      0.89       0.54       0.24         0.3  0.36
## 542        0.45    -0.04      0.25       1.14      -0.37         0.3   0.1
##     delinq_2yrs inq_last_6mths pub_rec revol_bal revol_util total_acc
## 1         -0.42           -0.8   -0.24      -0.3      -1.23     -0.02
## 2          3.03           1.14   -0.24     -0.59       0.85     -0.28
## 3          1.31           0.17   -0.24      0.22      -0.34      0.06
## 4         -0.42           -0.8   -0.24     -0.65       1.36     -0.54
## ...         ...            ...     ...       ...        ...       ...
## 539       -0.42           -0.8     3.6     -0.19       1.44     -0.72
## 540       -0.42           -0.8   -0.24      0.48       1.39      0.67
## 541       -0.42           3.07   -0.24     -0.32      -1.73      1.62
```

```
## 542       -0.42           -0.8   -0.24      0.11      -0.58      2.06
##     tot_coll_amt tot_cur_bal total_credit_rv months_since_last_credit_pull
## 1          -0.25        0.82            0.47                         -1.96
## 2          -0.25       -0.72           -0.95                          0.39
## 3          -0.25        0.58            0.39                          0.62
## 4          -0.25       -0.42           -1.06                           0.5
## ...          ...         ...             ...                           ...
## 539        -0.25       -0.43           -0.72                         -0.06
## 540        -0.25        0.82           -0.21                         -2.85
## 541        -0.25       -0.52            1.62                          -0.5
## 542        -0.25        0.17            0.59                          0.17
```

```
m <- c("average", "single", "complete", "ward")
names(m) <- c("average", "single", "complete", "ward")
```

Function to compute agglomerative coefficient

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

Calculate agglomerative coefficient for each clustering linkage method

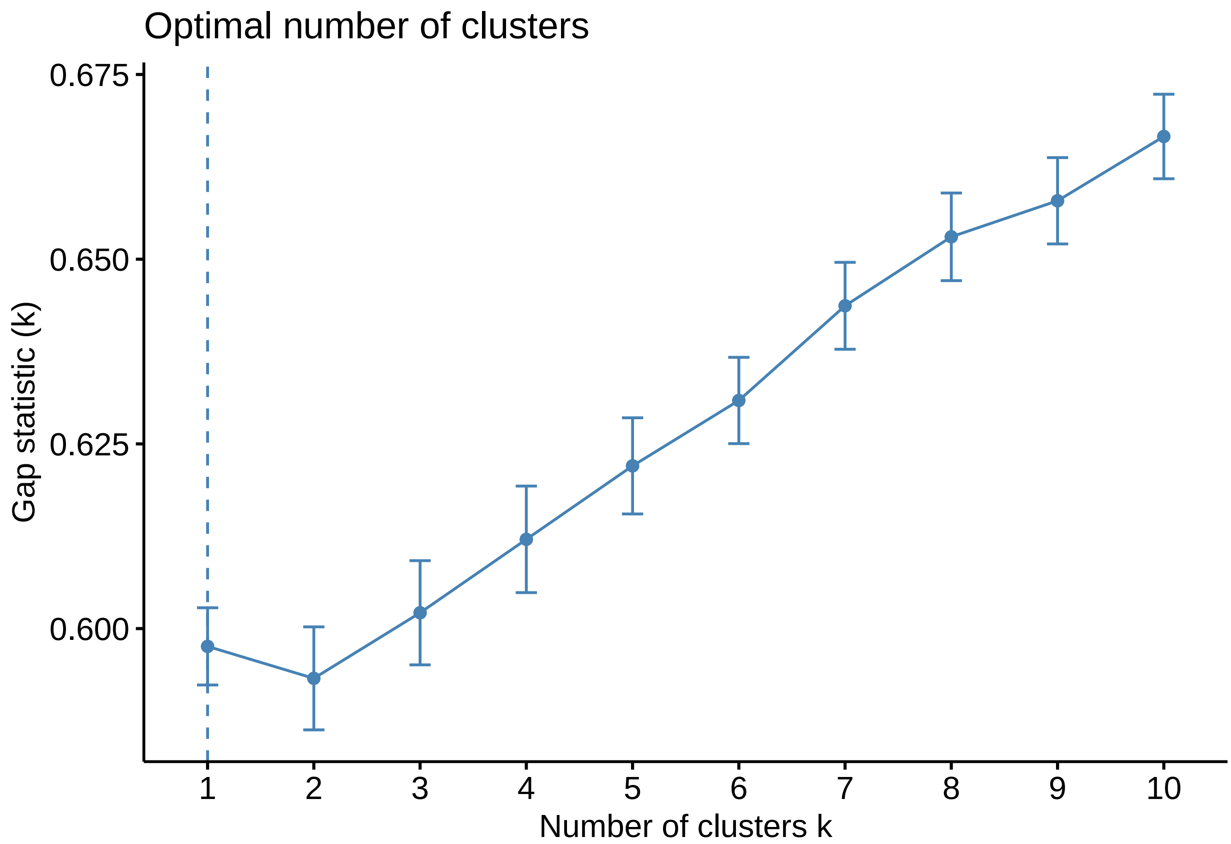
```
sapply(m, ac)
```

```
##   average    single  complete      ward
## 0.6563386 0.5034959 0.7876595 0.9338284
```

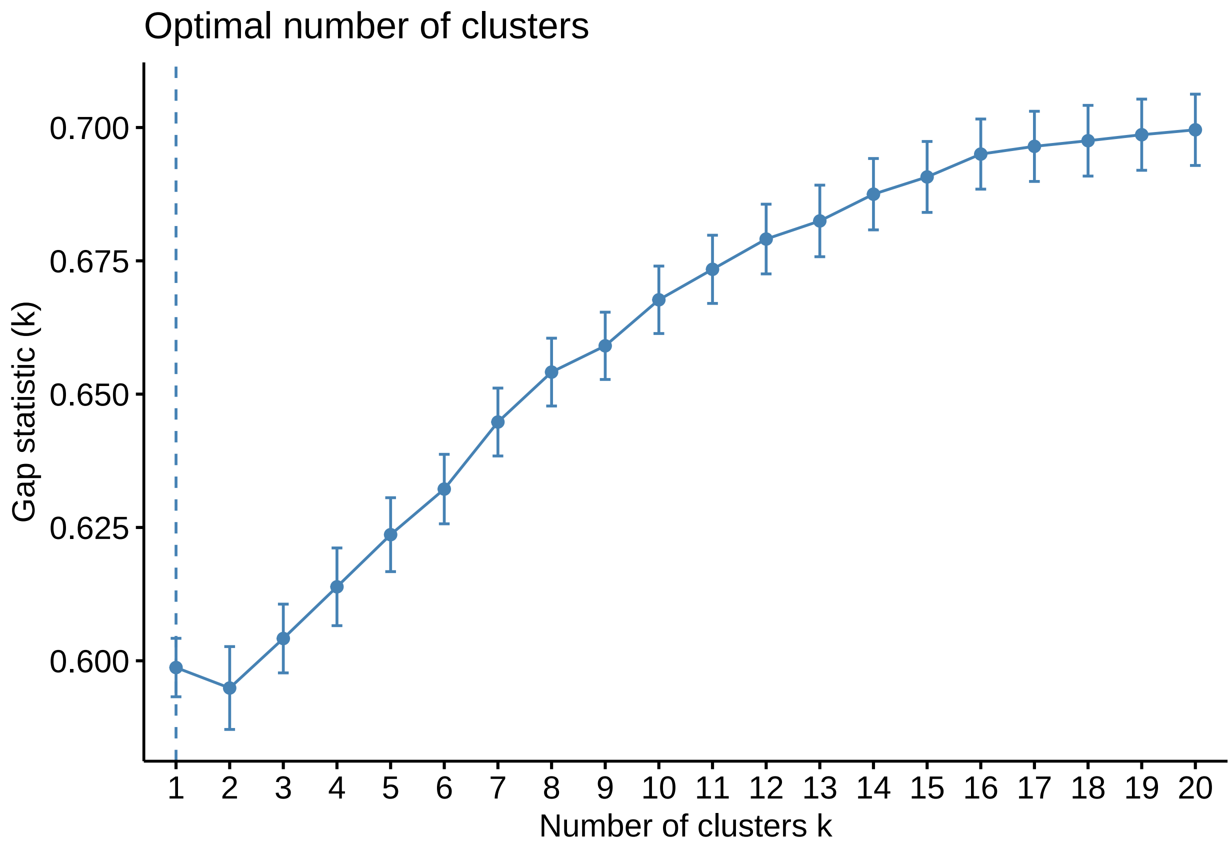
Based on the result, ward perform better with value 0.94

Determine the number of clusters

```
# Produce plot of clusters vs. gap statistic
gap_stat_h1 <- clusGap(sample_maha_updated, FUN = hcut, rstart = 25, K.max = 10, B = 50)
gap_stat_h2 <- clusGap(sample_maha_updated, FUN = hcut, rstart = 25, K.max = 20, B = 50)
fviz_gap_stat(gap_stat_h1)
```




```
fviz_gap_stat(gap_stat_h2)
```



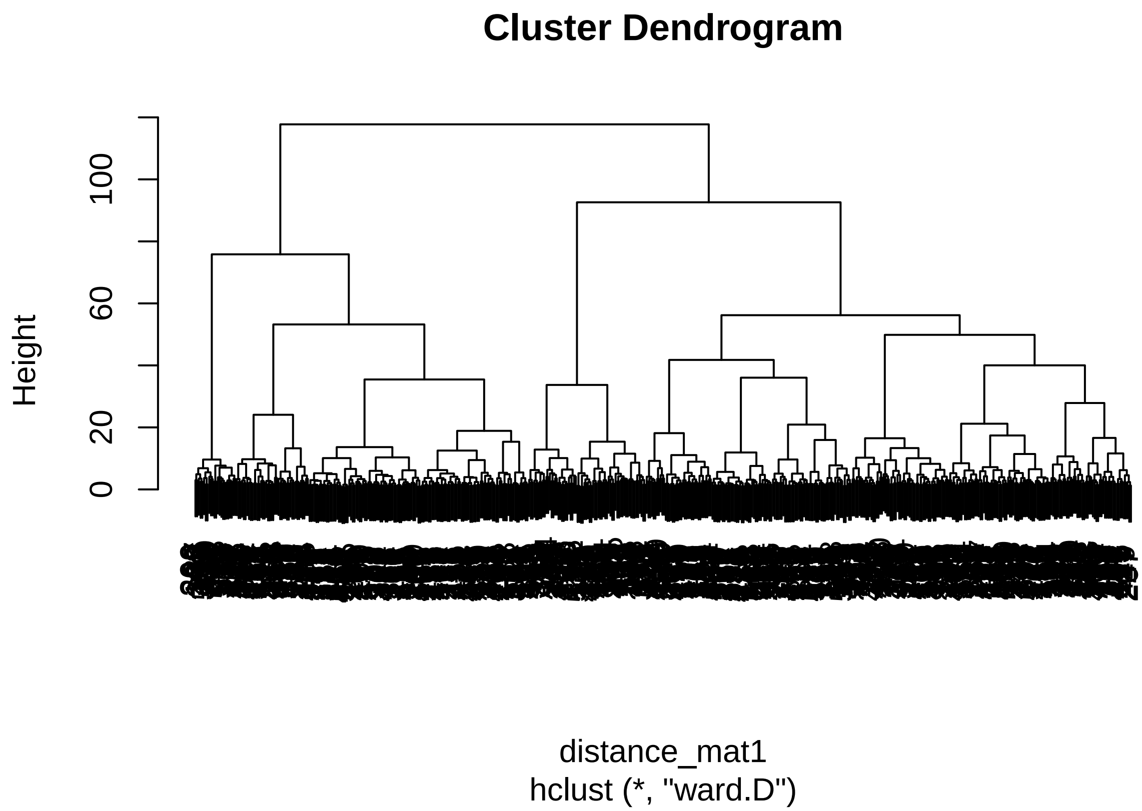

```r
# Plot dendrogram
# Finding distance matrix
distance_mat1 <- dist(sample_maha_updated, method = "euclidean")

set.seed(240)  # Setting seed
Hierar_cl1 <- hclust(distance_mat1, method = "ward")
```

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

```r
Hierar_cl1
```

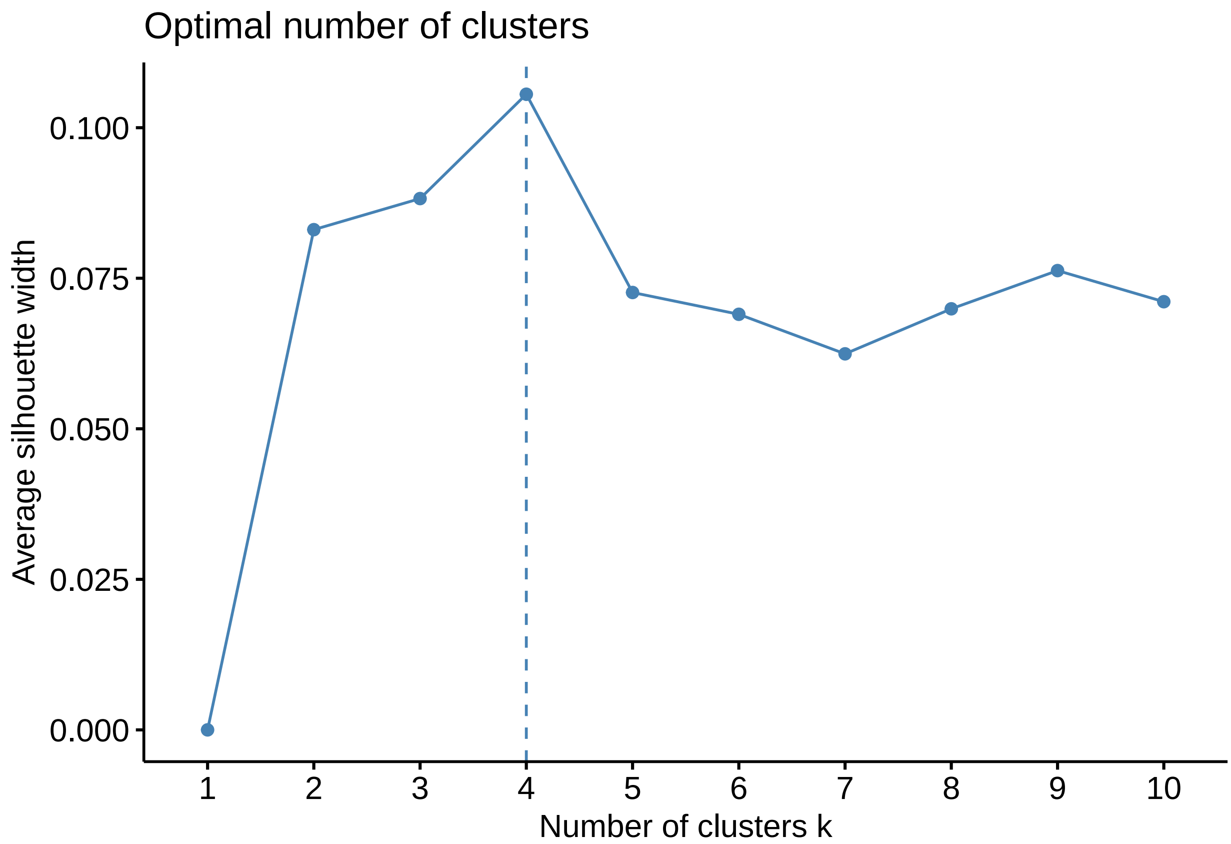
```
##
## Call:
## hclust(d = distance_mat1, method = "ward")
##
## Cluster method   : ward.D
## Distance         : euclidean
## Number of objects: 542
```

```r
plot(Hierar_cl1)
```



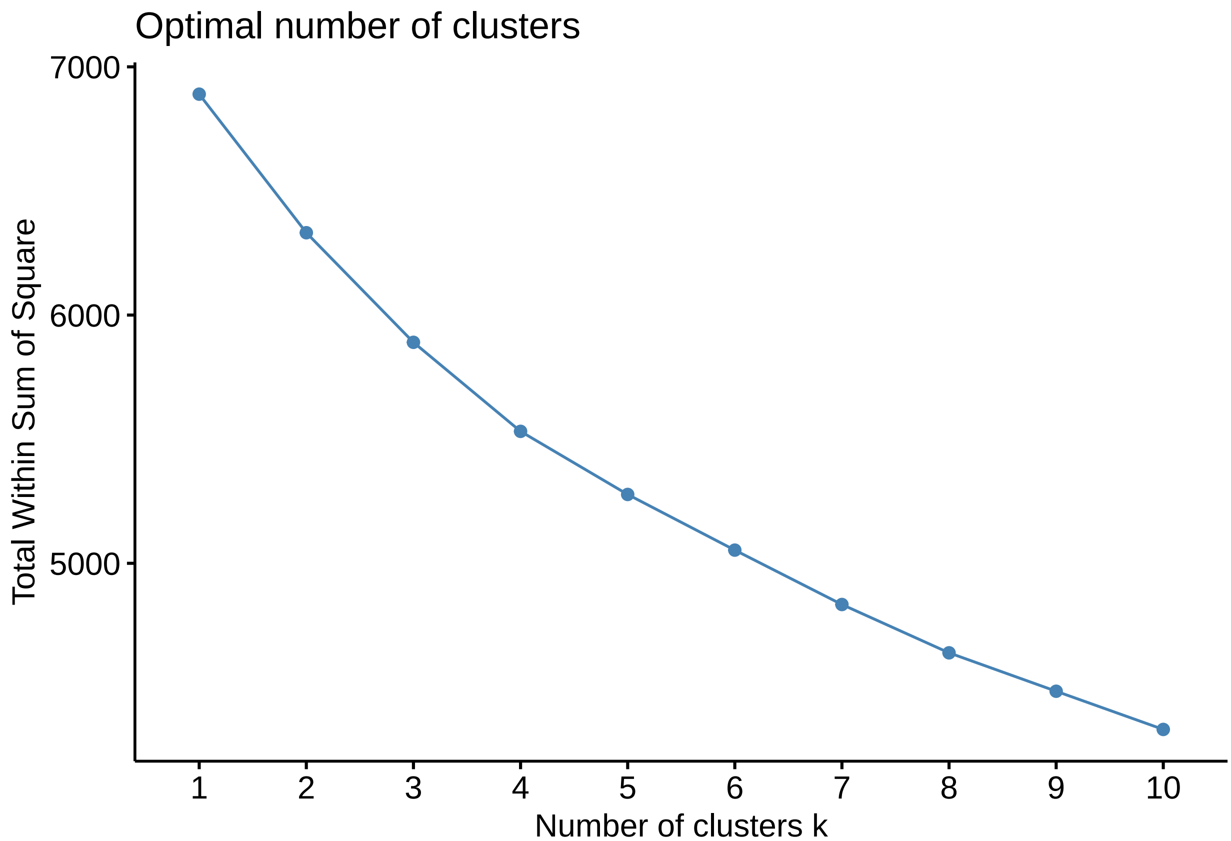
Silhouette plot for hcut

```
fviz_nbclust(sample_maha_updated, hcut, method = "silhouette")
```



Create elbow plot (without outliers)

```
fviz_nbclust(sample_maha_updated, hcut, method = "wss")
```



Based on the silhoutte plot, we decided the k (number of clusters) = 4

```
# Cutting tree by no. of clusters
fit1 <- cutree(Hierar_cl1, k = 4 )
fit1
```

```
##   [1] 1 2 2 2 1 1 2 1 1 2 1 3 2 2 2 2 2 1 2 2 1 1 2 2 2 2 1 1 1 2 2 4 1 4 1 1 2
##  [38] 2 2 2 2 1 2 4 2 2 4 4 2 4 4 1 4 2 2 2 2 1 4 1 1 3 1 2 2 1 1 4 1 4 2 4 1 1
##  [75] 2 2 1 1 1 1 2 4 4 2 4 2 3 1 1 1 2 2 2 2 4 2 1 2 2 2 4 2 1 1 2 1 2 2 2 1 2
## [112] 2 1 4 4 1 2 3 2 2 2 2 1 4 2 1 2 1 1 3 1 2 2 2 2 1 2 1 1 2 2 2 2 1 4 2 1 2
## [149] 1 2 2 2 2 2 4 1 2 2 2 2 2 3 1 2 1 3 2 1 2 1 2 2 2 1 1 2 1 1 2 4 2 2 2 1 2
## [186] 1 2 2 4 1 2 2 2 2 2 2 1 4 2 2 2 3 2 1 1 1 1 2 4 3 2 2 1 2 2 1 2 1 1 2 4 2 2
## [223] 1 2 1 2 2 2 4 1 2 2 2 2 1 2 2 3 2 1 4 2 2 2 1 1 1 1 2 2 2 2 1 2 4 2 4 3 2 2
## [260] 2 2 4 2 1 2 2 2 2 4 1 3 4 4 4 2 2 1 2 2 2 2 4 2 1 2 4 2 2 2 1 2 2 1 2 1 2
## [297] 2 2 1 1 2 2 2 1 4 1 2 2 1 2 1 1 1 1 3 3 1 2 2 4 2 2 1 1 1 4 2 1 1 2 2 2 2
## [334] 4 2 4 4 3 2 1 2 2 2 1 3 4 2 2 2 1 2 3 4 1 2 1 2 1 2 1 2 2 1 1 2 2 2 1 2 2
## [371] 1 1 1 2 1 2 1 2 1 2 2 2 1 2 4 2 2 2 1 2 2 1 4 2 4 2 2 2 2 1 4 3 1 4 2 2 2
## [408] 3 1 2 2 1 2 1 4 1 2 2 2 2 1 1 2 1 1 1 3 2 1 4 1 2 3 1 1 1 2 4 3 2 2 2 2 2
## [445] 2 4 4 1 2 1 1 2 1 1 2 2 1 1 1 1 1 2 2 2 1 2 2 4 1 2 1 1 4 4 2 2 2 2 1 2 1
## [482] 4 2 2 2 2 4 1 1 2 1 4 2 4 1 2 4 2 1 2 2 2 2 2 2 2 1 2 2 4 4 2 2 2 4 1 1 1
## [519] 2 1 2 2 2 2 1 1 2 2 1 2 2 3 4 2 1 1 2 1 3 2 2 2
```

```
table(fit1)
```

```
## fit1
##   1   2   3   4
## 169 281  24  68
```

Cluster Analysis - Hierarchical Clustering

```r
# Hierarchical Clustering without Factor Analysis
final_ca <- cbind(sample_maha_updated, cluster = fit1)
head(final_ca)
```

```
##   funded_amnt    int_rate   sub_grade emp_length  annual_inc loan_status
## 1  -0.5262529 -0.96262018  0.88866521 -0.3682816  0.02449554   0.3043473
## 2  -0.2818275  0.98971379 -0.86423090  0.8394129 -0.36016624   0.3043473
## 3  -0.8226186  0.47409137 -0.38616833 -0.3682816 -0.55166639   0.3043473
## 4  -0.6484655  1.05065099 -1.02358510 -1.5759761 -1.02598528   0.3043473
## 5  -1.3572991 -0.06496842  0.09189425  1.1413365 -0.39356009   0.3043473
## 6  -1.3817416 -0.73293383  0.72931102  0.2355656 -0.58696927   0.3043473
##            dti delinq_2yrs inq_last_6mths    pub_rec  revol_bal revol_util
## 1 -1.616398897  -0.4161973     -0.8030459 -0.2365206 -0.2954600 -1.2331561
## 2  0.548173278   3.0281943      1.1358979 -0.2365206 -0.5869740  0.8508702
## 3  1.420714000   1.3059985      0.1664260 -0.2365206  0.2207511 -0.3369390
## 4  1.241301041  -0.4161973     -0.8030459 -0.2365206 -0.6519309  1.3611565
## 5 -1.626724823  -0.4161973      0.1664260 -0.2365206 -0.6196246  0.8465821
## 6 -0.008135969  -0.4161973     -0.8030459 -0.2365206 -1.0008246  1.1982079
##     total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1 -0.02443550  -0.24728124  0.81792974      0.47076292
## 2 -0.28469528  -0.24728124 -0.71871428     -0.95186026
## 3  0.06231776  -0.24728124  0.58009947      0.39120833
## 4 -0.54495506  -0.24728124 -0.42495812     -1.05949294
## 5 -0.19794202   0.05249612 -0.06167051     -0.02062452
## 6 -0.63170832  -0.24728124 -0.98941498     -1.32155510
##   months_since_last_credit_pull cluster
## 1                    -1.9570039       1
## 2                     0.3918482       2
## 3                     0.6155484       2
## 4                     0.5036983       2
## 5                     0.1681480       1
## 6                    -1.0622031       1
```

```r
hcentres1 <- aggregate(x=final_ca, by=list(cluster=fit1), FUN="mean")
print(hcentres1)
```

```
##   cluster funded_amnt   int_rate   sub_grade  emp_length annual_inc loan_status
## 1       1 -0.52999518 -0.7465802  0.71516716 -0.19146655 -0.2962732   0.3676746
## 2       2  0.21959238  0.2783100 -0.23645478  0.13878676  0.0263141   0.3495750
## 3       3 -0.39640190  0.1143276 -0.06745994 -0.29280070 -0.1239197   0.1928647
## 4       4  0.03889089  0.4236321 -0.43772409  0.03573127 -0.2347818  -2.3023762
##          dti delinq_2yrs inq_last_6mths    pub_rec  revol_bal revol_util
## 1 -0.2805815  -0.3958163    -0.34412426 -0.2365206 -0.3459943 -0.4300851
## 2  0.2259107   0.1721685     0.06637375 -0.2365206  0.0882827  0.3306190
## 3 -0.4682850  -0.2726810    -0.15673128  3.5989479 -0.4785661 -0.1298586
## 4  0.1471515  -0.2642389     0.13791212 -0.2365206 -0.1005356  0.1823019
##     total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1 -0.35707522  -0.08435750 -0.24960107     -0.16197962
## 2  0.13239957  -0.11819165  0.03173603     -0.03429777
## 3 -0.42205461  -0.14303035 -0.05577798     -0.46924885
## 4 -0.06398478  -0.05999734 -0.10508874     -0.11487195
##   months_since_last_credit_pull cluster
## 1                    0.23366964       1
```

```
## 2                    0.03997812      2
## 3                    0.08426041      3
## 4                   -0.58848504      4
```

Cluster Analysis- Non-Hierarchical Clustering

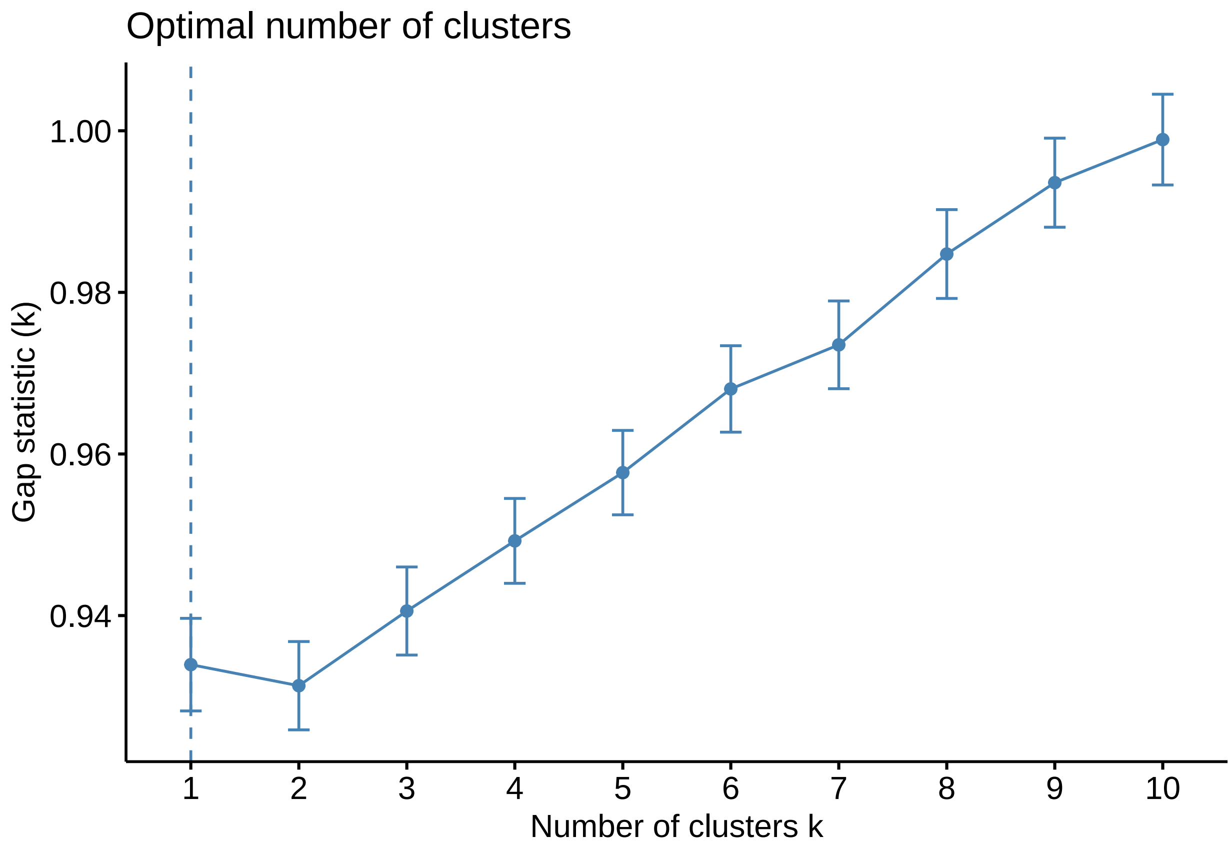
Cluster analysis including the outliers

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

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

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

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

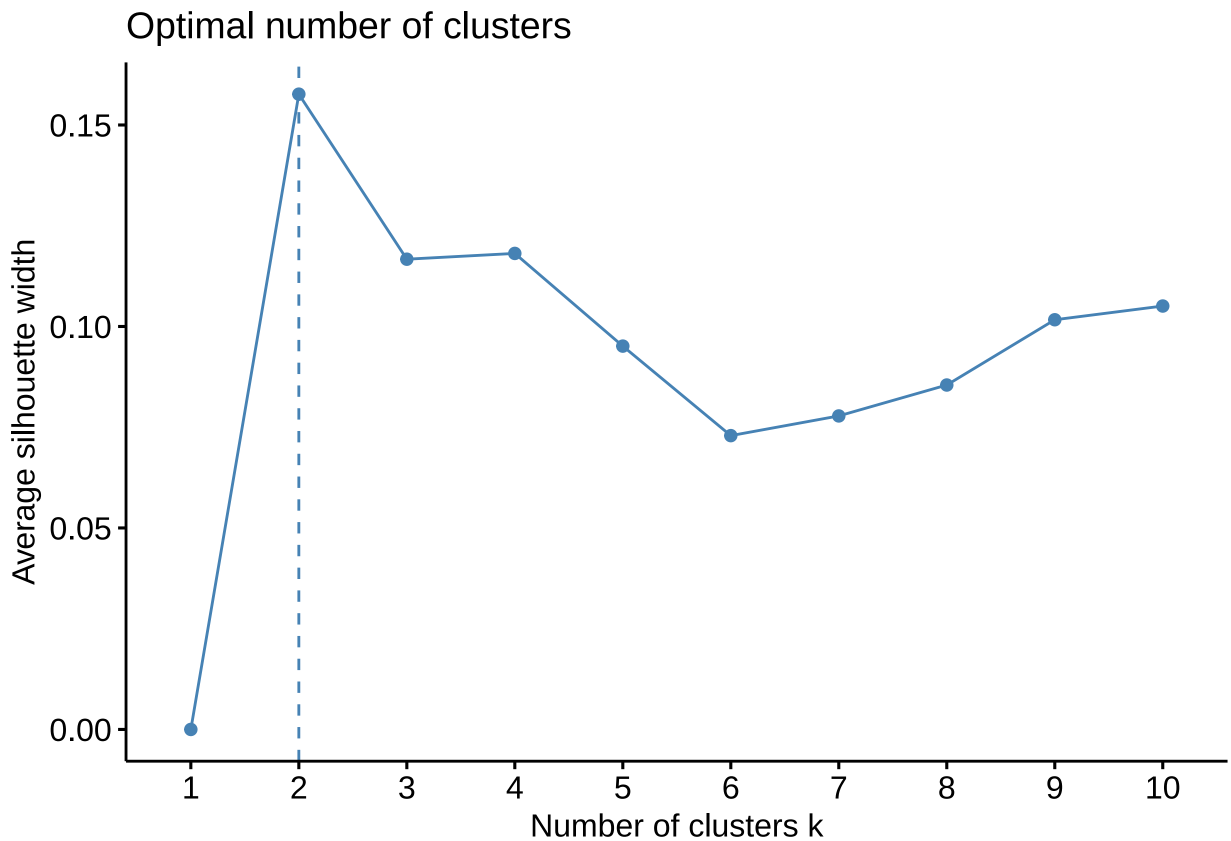
Produce Gap Statistic Plot

```
fviz_gap_stat(gap_stat_k)
```



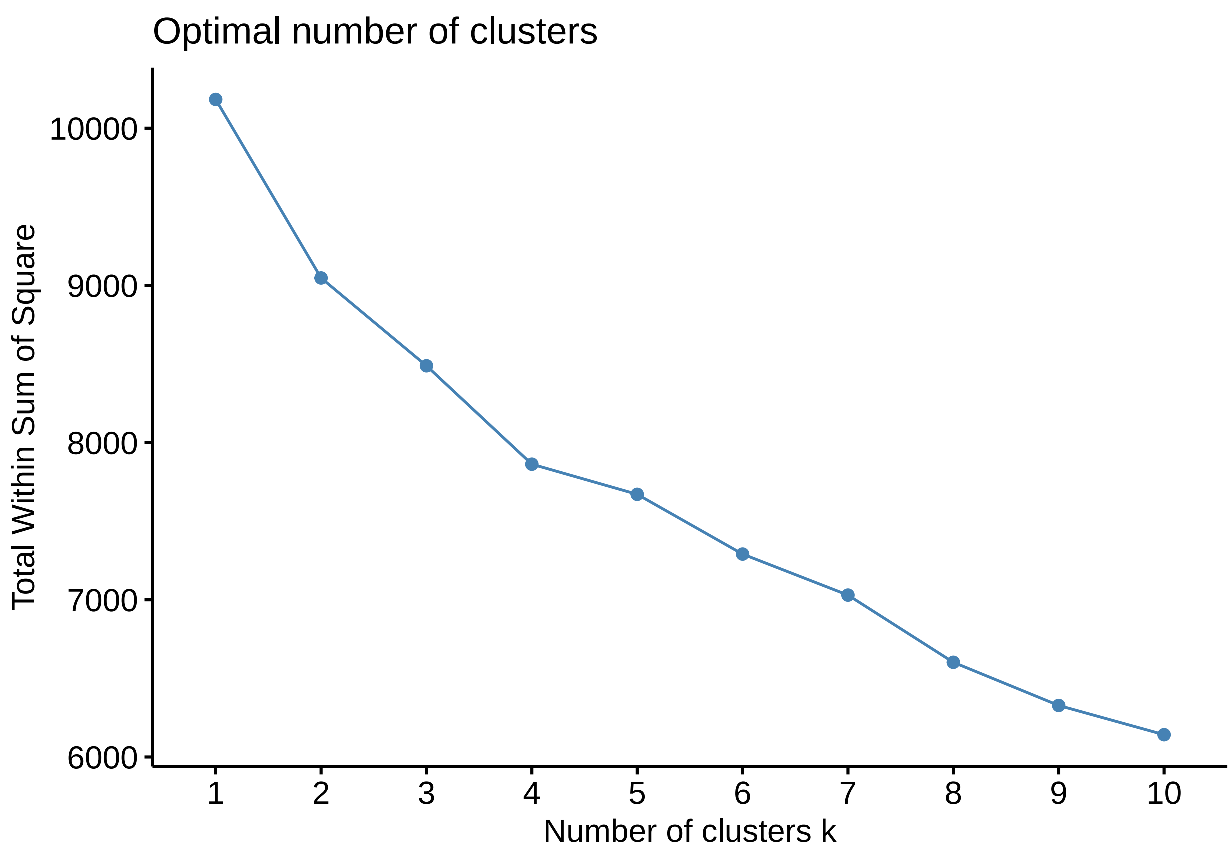
Silhouette plot

```
fviz_nbclust(sample_maha, kmeans, method = "silhouette") # with outliers
```



Create elbow plot (method 1)

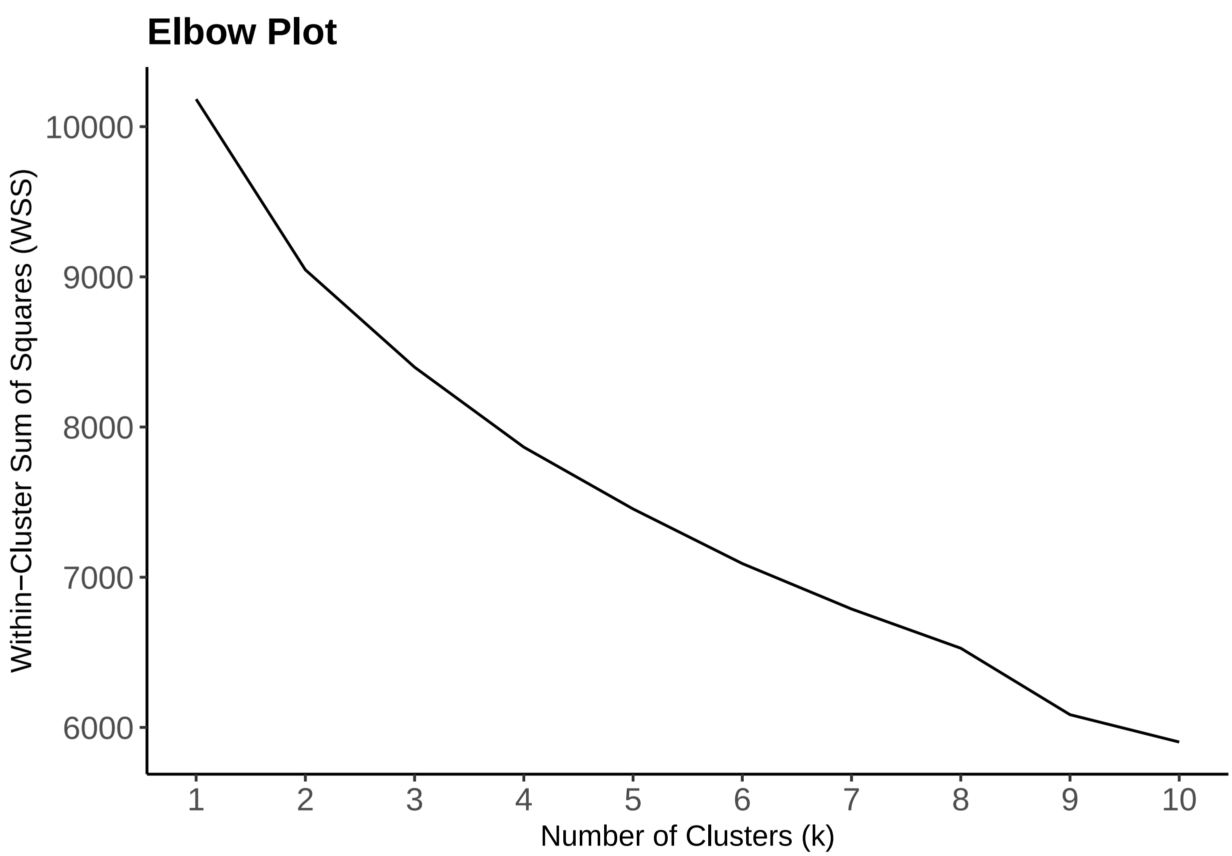
```
fviz_nbclust(sample_maha, kmeans, method = "wss") # with outliers
```



Elbow plot for kmeans (method 2)

```
# Define a range of k values to explore (adjust based on your data)
k_values <- 1:10

# Empty list to store WSS for different k values
wss_list <- list()

# Function to calculate WSS within a loop
calculate_wss <- function(data, k) {
  # Perform K-Means clustering with specific k
  kmeans_result <- kmeans(data, centers = k, nstart = 10)

  # Calculate within-cluster sum of squares
  wss <- sum(kmeans_result$withinss)
  return(wss)
}

# Loop through k values and calculate WSS
for (k in k_values) {
  wss_list[[k]] <- calculate_wss(sample_maha, k)
}

# Combine results into a data frame
wss <- unlist(wss_list)
elbow_data <- data.frame(k = k_values, wss = wss)
```

```r
# Create the ggplot for the elbow plot
ggplot(elbow_data, aes(x = k, y = wss)) +
  geom_line() +
  labs(title = "Elbow Plot", x = "Number of Clusters (k)", y = "Within-Cluster Sum of Squares (WSS)") +
  theme_classic() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12)
  ) +
  scale_x_continuous(breaks = seq(0, 10, by=1))
```



Cluster analysis without the outliers

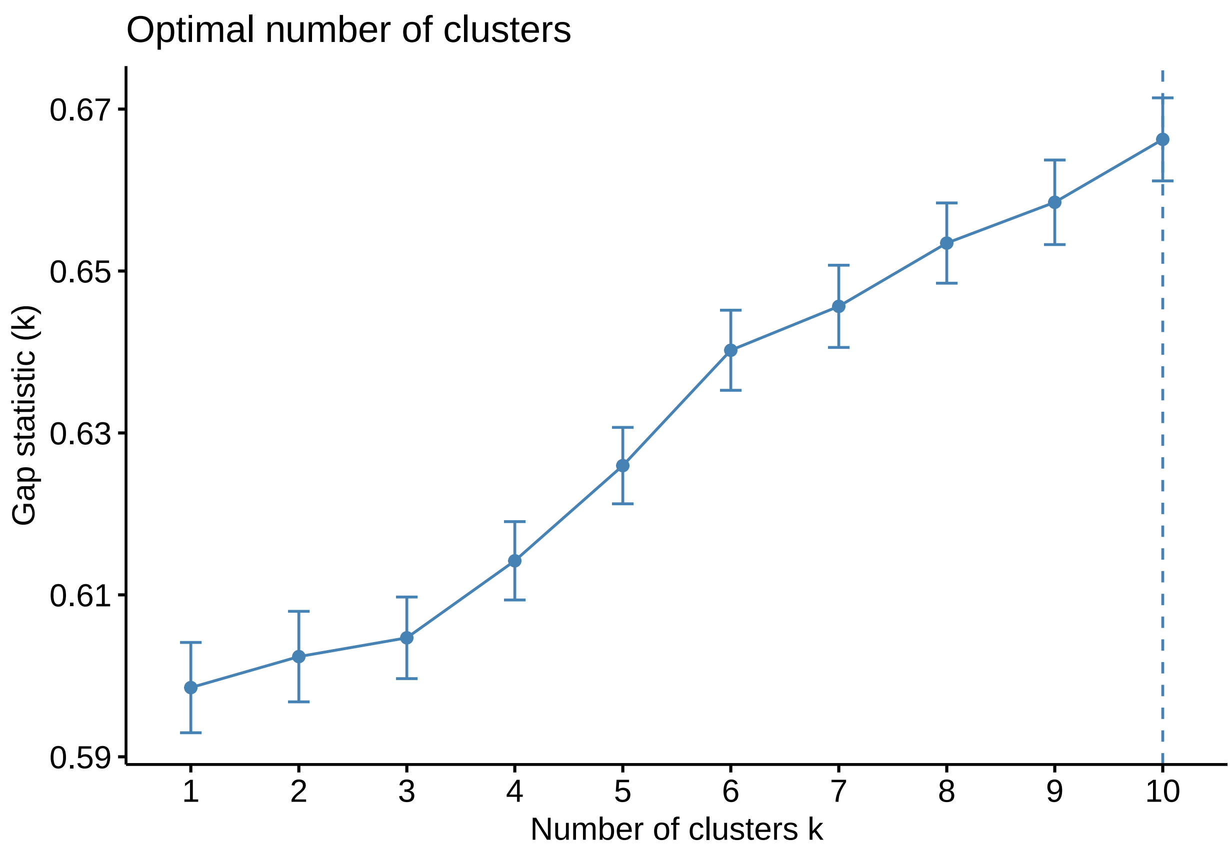
```r
#cluster analysis without outliers
gap_stat_k2 <- clusGap(sample_maha_updated, FUN = kmeans, nstart = 25, K.max = 10, B = 50)
```

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

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

## Warning: did not converge in 10 iterations

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

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

## Warning: did not converge in 10 iterations

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

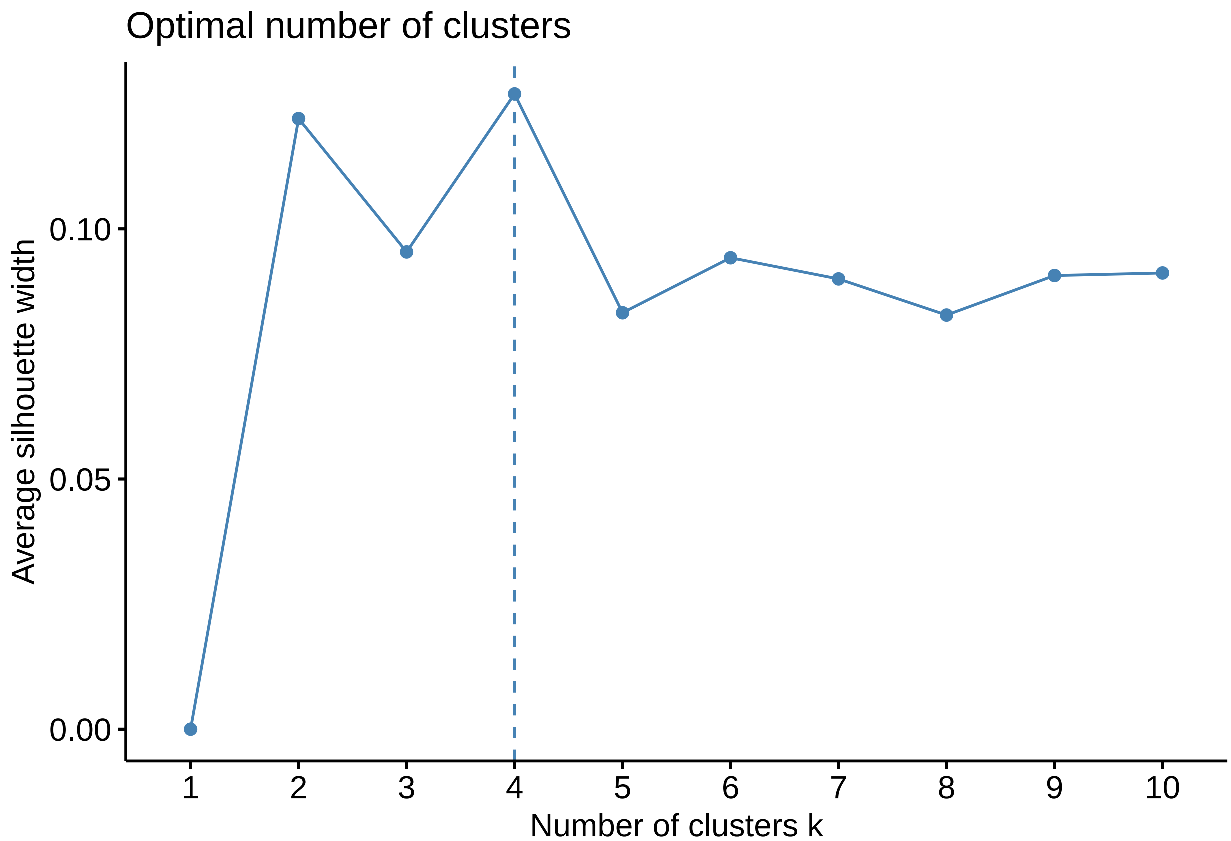
Produce Gap Statistic Plot

```
fviz_gap_stat(gap_stat_k2) # without outliers
```



Silhouette plot for kmeans

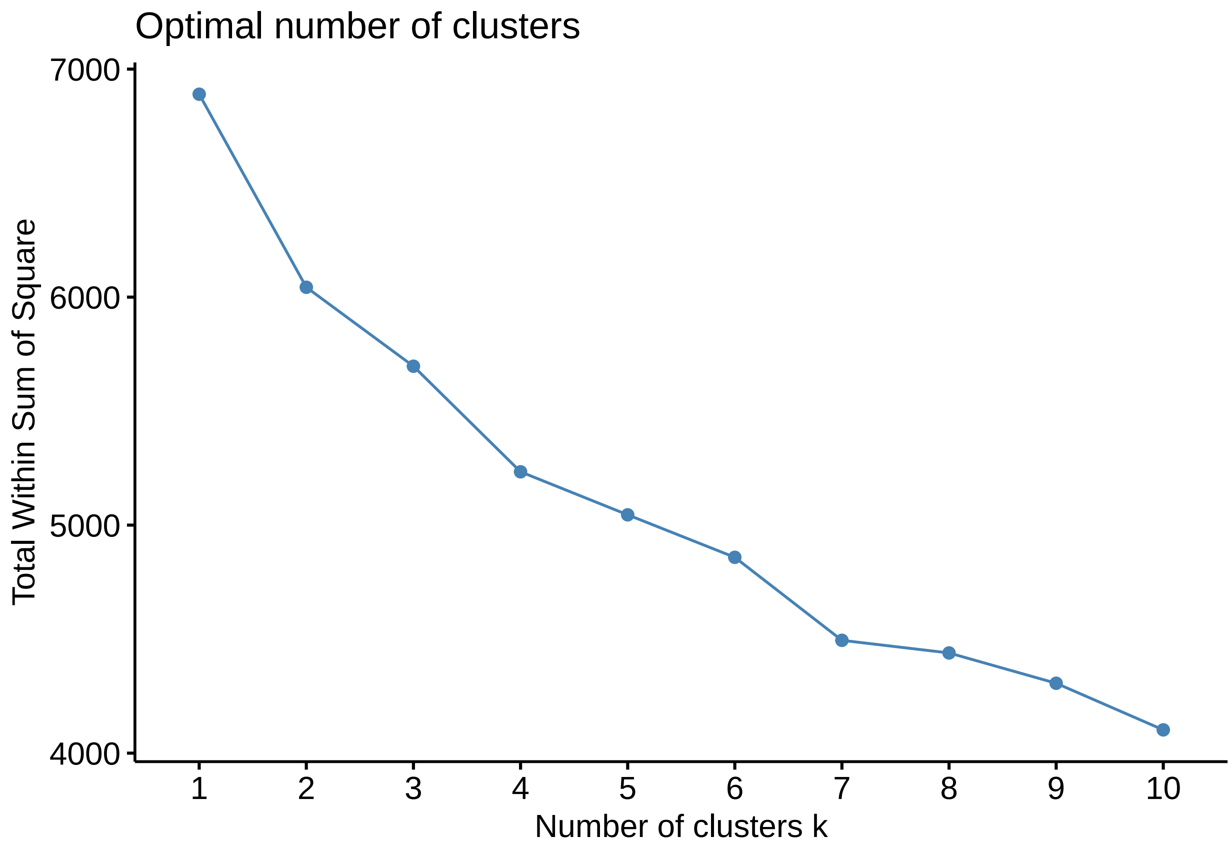
```
fviz_nbclust(sample_maha_updated, kmeans, method = "silhouette")
```




```
# without outliers
```

Elbow plot for kmeans (method 1)

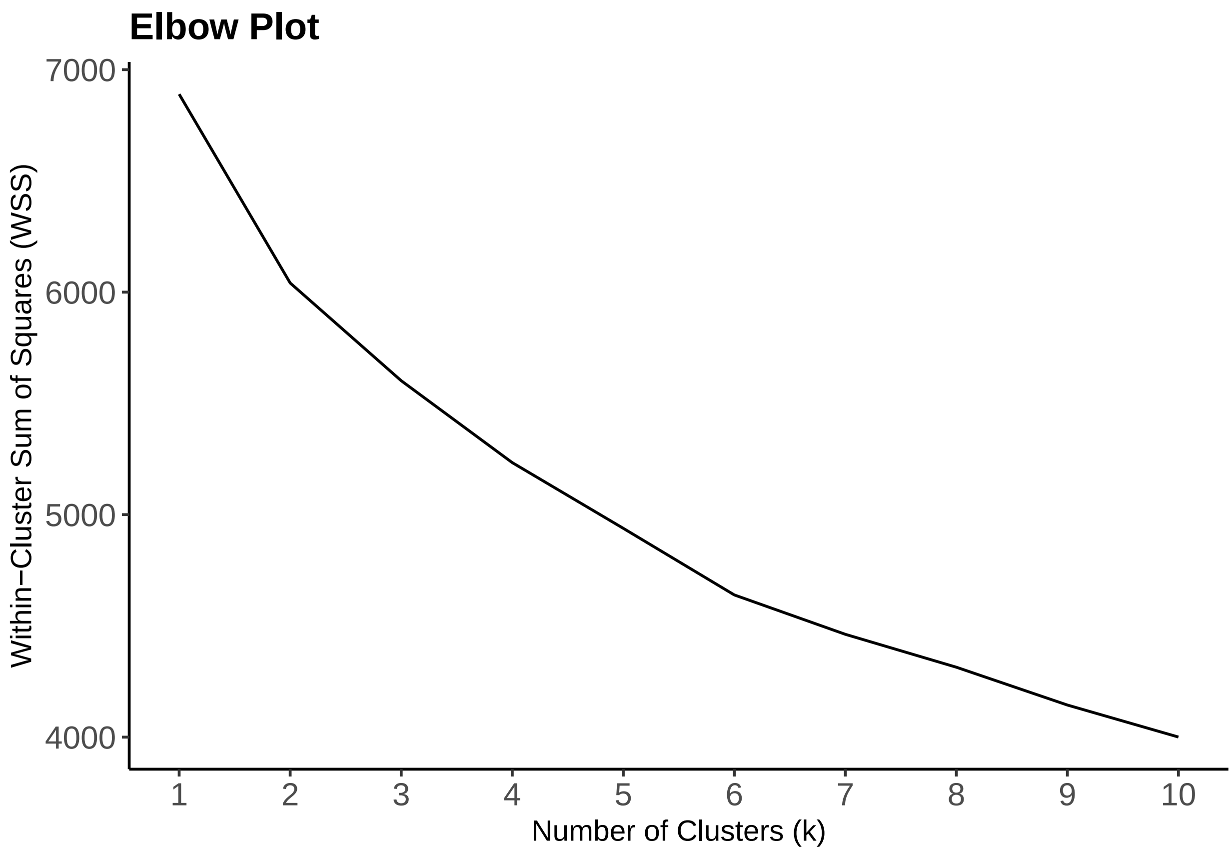
```
fviz_nbclust(sample_maha_updated, kmeans, method = "wss") # without outliers
```



Elbow plot for kmeans (method 2)

```
# Define a range of k values to explore (adjust based on your data)
k_values <- 1:10

# Empty list to store WSS for different k values
wss_list <- list()

# Function to calculate WSS within a loop
calculate_wss <- function(data, k) {
  # Perform K-Means clustering with specific k
  kmeans_result <- kmeans(data, centers = k, nstart = 10)

  # Calculate within-cluster sum of squares
  wss <- sum(kmeans_result$withinss)
  return(wss)
}

# Loop through k values and calculate WSS
for (k in k_values) {
  wss_list[[k]] <- calculate_wss(sample_maha_updated, k)
}

# Combine results into a data frame
wss <- unlist(wss_list)
elbow_data <- data.frame(k = k_values, wss = wss)
```

```r
# Create the ggplot for the elbow plot
ggplot(elbow_data, aes(x = k, y = wss)) +
  geom_line() +
  labs(title = "Elbow Plot", x = "Number of Clusters (k)", y = "Within-Cluster Sum of Squares (WSS)") +
  theme_classic() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12)
  ) +
  scale_x_continuous(breaks = seq(0, 10, by=1))
```



Run k-means

with outliers

```r
# k = 4
set.seed(55)
k_cl <- kmeans(sample_maha,4,nstart=25)
k_cl
```

```
## K-means clustering with 4 clusters of sizes 165, 300, 38, 97
##
## Cluster means:
##   funded_amnt   int_rate  sub_grade emp_length  annual_inc loan_status
## 1   0.5132215  1.0961905 -1.1278897  0.2117250  0.04334133  -0.4699495
## 2  -0.4913000 -0.4748414  0.4961227 -0.2375132 -0.37471943   0.1148270
```

```
## 3  -0.4360406  0.2152933 -0.1848788 -0.2093744 -0.11448854   0.2163347
## 4   0.8172991 -0.4804142  0.4566018  0.4564492  1.13005231   0.3595139
##           dti delinq_2yrs inq_last_6mths    pub_rec   revol_bal revol_util
## 1  0.19793665  0.12655530     0.41320066 -0.2365206  0.09612875  0.5012975
## 2 -0.08426386 -0.03731424    -0.23105746 -0.2365206 -0.33925003 -0.1239338
## 3 -0.36774312 -0.05362978    -0.03767334  3.4980145 -0.49461783 -0.3595081
## 4  0.06797781 -0.07886000     0.02650223 -0.2365206  1.07947673 -0.3285838
##    total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1  0.2179478  -0.06296939 -0.02512149     -0.08639599
## 2 -0.4119334  -0.09154655 -0.26755485     -0.26022780
## 3 -0.0952079   1.36460450 -0.25347245     -0.44447320
## 4  0.9405827  -0.14434081  0.96952014      1.12591402
##   months_since_last_credit_pull
## 1                   -0.17824838
## 2                    0.02572552
## 3                   -0.04377852
## 4                    0.24079289
##
## Clustering vector:
##   [1] 1 2 1 2 4 2 2 2 2 2 2 2 2 4 3 2 2 2 4 4 2 4 2 2 2 1 1 2 1 4 2 2 4 2 1 2 1
##  [38] 2 4 1 2 2 1 1 4 2 1 1 2 1 1 2 1 1 3 1 2 2 4 1 4 1 2 2 2 2 2 2 3 2 1 2 1 2
##  [75] 4 2 1 2 1 1 2 2 4 4 3 1 2 2 2 2 4 1 1 1 2 2 3 2 2 2 2 3 2 4 2 1 2 2 4 1 2
## [112] 2 1 2 2 1 1 2 2 2 2 2 4 1 2 1 1 2 3 4 1 3 2 1 2 1 2 4 2 1 1 2 1 2 2 3 2 2
## [149] 4 4 2 2 4 2 2 2 2 4 4 4 2 2 2 3 1 2 1 2 2 1 2 2 1 1 2 4 2 4 1 1 3 2 1 2 3
## [186] 2 2 2 2 4 2 1 2 4 1 4 2 4 1 4 2 2 2 1 2 1 2 2 2 2 4 2 1 1 2 2 1 4 1 2 3 1
## [223] 2 4 2 2 2 2 3 1 2 2 4 4 2 1 2 2 2 1 1 4 2 4 2 2 3 2 4 1 2 2 4 2 2 2 2 4 1
## [260] 3 1 2 2 2 1 1 2 2 2 1 1 4 1 2 1 1 2 1 2 1 3 2 1 4 2 2 1 1 4 1 1 4 1 2 2 3
## [297] 4 1 2 1 4 2 2 1 1 4 1 2 1 2 2 1 4 4 2 4 2 4 2 4 2 2 2 1 1 2 2 2 1 4 2 4 2
## [334] 2 1 4 2 1 2 2 3 4 4 3 3 3 2 2 1 2 1 1 1 2 2 2 2 1 2 2 2 1 1 4 1 1 1 2 3 4
## [371] 4 2 1 2 1 2 3 1 1 2 2 2 2 3 1 2 2 2 2 2 2 2 1 1 2 4 1 3 1 2 4 1 1 2 4 2 1
## [408] 3 2 4 2 2 2 2 4 4 1 2 1 2 2 4 2 2 2 2 4 2 2 4 1 4 4 1 2 1 2 2 3 2 1 2 1 4
## [445] 1 3 3 2 4 1 2 2 2 2 2 4 1 2 2 2 2 2 2 2 2 2 3 1 2 2 4 1 2 3 2 2 4 4 4 1 3
## [482] 2 4 2 2 4 2 1 2 2 1 2 2 2 2 2 2 2 1 2 4 2 2 2 2 1 2 2 2 2 2 1 2 1 2 2 1 1
## [519] 2 3 1 2 4 2 1 2 1 1 1 2 1 2 2 1 2 2 2 2 1 2 2 2 1 1 4 1 1 4 2 2 1 1 1 2 4
## [556] 1 1 2 1 4 1 1 1 2 4 1 2 4 4 2 4 2 2 2 1 2 4 2 2 2 2 1 2 1 2 2 3 2 1 2 2 1
## [593] 2 3 3 3 4 1 4 4
##
## Within cluster sum of squares by cluster:
## [1] 2113.5751 2890.7681  944.4208 1905.5843
##  (between_SS / total_SS =  22.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Show centroid values

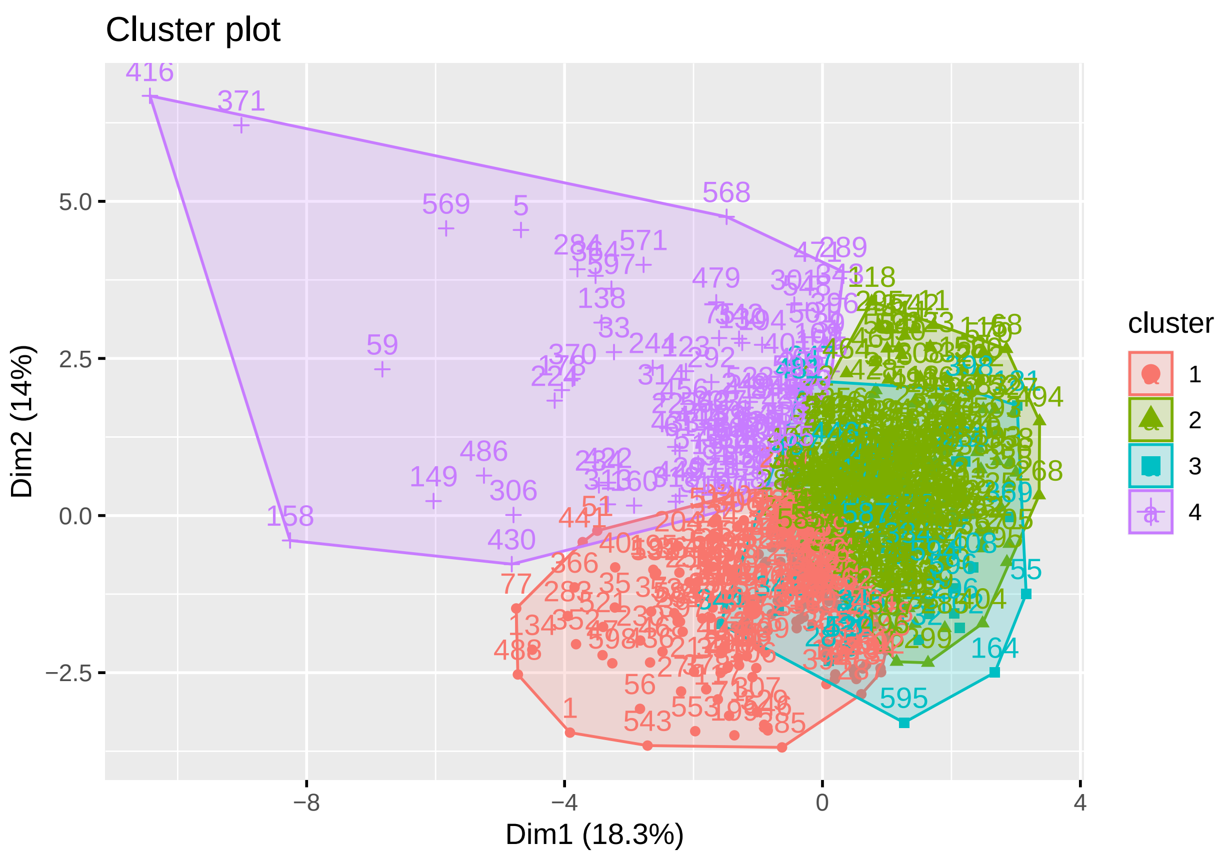
```r
k_cl$centers # with outliers, k = 4
```

```
##   funded_amnt   int_rate  sub_grade emp_length  annual_inc loan_status
## 1   0.5132215  1.0961905 -1.1278897  0.2117250  0.04334133  -0.4699495
## 2  -0.4913000 -0.4748414  0.4961227 -0.2375132 -0.37471943   0.1148270
## 3  -0.4360406  0.2152933 -0.1848788 -0.2093744 -0.11448854   0.2163347
## 4   0.8172991 -0.4804142  0.4566018  0.4564492  1.13005231   0.3595139
```

```
##           dti delinq_2yrs inq_last_6mths    pub_rec   revol_bal revol_util
## 1  0.19793665  0.12655530     0.41320066 -0.2365206  0.09612875  0.5012975
## 2 -0.08426386 -0.03731424    -0.23105746 -0.2365206 -0.33925003 -0.1239338
## 3 -0.36774312 -0.05362978    -0.03767334  3.4980145 -0.49461783 -0.3595081
## 4  0.06797781 -0.07886000     0.02650223 -0.2365206  1.07947673 -0.3285838
##    total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1  0.2179478  -0.06296939 -0.02512149     -0.08639599
## 2 -0.4119334  -0.09154655 -0.26755485     -0.26022780
## 3 -0.0952079   1.36460450 -0.25347245     -0.44447320
## 4  0.9405827  -0.14434081  0.96952014      1.12591402
##   months_since_last_credit_pull
## 1                   -0.17824838
## 2                    0.02572552
## 3                   -0.04377852
## 4                    0.24079289
```

Visualise clusters

```
fviz_cluster(k_cl, data = sample_maha) # with outliers, k = 4
```



without outliers

```
# k = 4
set.seed(55)
k_cl2 <- kmeans(sample_maha_updated,4,nstart=25) # without outliers
k_cl2
```

```
## K-means clustering with 4 clusters of sizes 134, 219, 120, 69
```

```
##
## Cluster means:
##   funded_amnt   int_rate   sub_grade  emp_length annual_inc loan_status
## 1 -0.49066071 -0.9558863  0.88985442 -0.39538442 -0.1335981   0.2244792
## 2 -0.36977600  0.1564132 -0.09292751 -0.12845664 -0.3920395   0.4081940
## 3  0.90475571  0.3301468 -0.31711484  0.65820435  0.4730026   0.3991074
## 4  0.04965222  0.5478342 -0.54783200  0.06047444 -0.2118357  -2.3033741
##          dti delinq_2yrs inq_last_6mths      pub_rec   revol_bal revol_util
## 1 -0.5244323 -0.12059655    -0.12296858 -0.007537363 -0.45646230 -1.0171181
## 2  0.1401578 -0.08591319    -0.16558491  0.008669210 -0.25962155  0.5209983
## 3  0.2770747 -0.01435163     0.09371561 -0.236520554  0.58350695  0.2443156
## 4  0.3066738 -0.09172565     0.11022473 -0.125347556 -0.05775889  0.3243015
##     total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1 -0.13838008  -0.08671889  -0.1135136     -0.05733001
## 2 -0.48711955  -0.08248795  -0.3623109     -0.38380484
## 3  0.70646071  -0.15547138   0.5364614      0.38431796
## 4  0.04094377  -0.09621828  -0.1676508     -0.15171259
##   months_since_last_credit_pull
## 1                    -0.1356836
## 2                     0.2774444
## 3                     0.1262042
## 4                    -0.6520861
##
## Clustering vector:
##   [1] 1 2 2 2 2 2 2 2 1 1 3 2 2 2 2 2 3 3 2 3 1 1 2 2 2 2 2 3 2 1 3 2 4 1 4 2 1 3
##  [38] 2 2 3 3 1 2 4 2 2 4 3 1 4 4 1 4 3 3 3 2 1 4 2 1 1 1 1 2 1 1 3 2 4 2 4 2 3
##  [75] 3 3 2 2 1 2 3 4 4 2 4 2 2 2 2 2 2 1 2 3 1 4 3 1 1 2 2 4 2 2 1 3 1 2 1 3 2 3
## [112] 2 2 4 4 1 3 2 2 2 2 2 2 2 4 3 2 3 2 2 2 2 2 3 2 2 3 2 2 1 1 3 3 2 1 4 2 2 3
## [149] 2 1 3 2 1 3 4 2 3 2 3 3 2 1 1 4 1 2 2 2 2 2 3 1 3 1 3 4 1 2 3 4 3 2 1 1 3
## [186] 2 3 1 1 2 2 3 1 3 3 2 1 4 3 3 1 4 2 1 1 2 2 4 1 4 2 2 3 3 2 2 2 1 2 4 2 3
## [223] 1 1 1 2 3 3 4 1 1 2 2 1 2 3 3 1 2 1 4 2 2 2 3 1 2 3 2 3 2 2 4 4 1 4 2 2 3
## [260] 2 2 2 2 1 3 3 3 2 1 1 2 4 4 4 2 2 1 3 2 3 2 1 2 2 2 4 3 1 3 2 3 2 1 2 1 3
## [297] 3 2 1 2 2 3 2 3 4 2 3 3 1 3 2 1 3 1 4 2 1 2 2 4 3 2 2 2 2 1 3 1 1 1 2 3 3
## [334] 4 3 4 4 1 3 1 3 3 4 2 2 4 2 2 2 2 1 2 4 1 2 1 2 1 2 1 2 2 2 1 3 2 1 3 2 3
## [371] 2 1 3 3 1 3 1 1 2 1 3 2 2 2 4 2 2 1 2 1 3 1 4 3 4 1 2 1 3 1 4 2 2 4 2 3 4
## [408] 2 2 3 3 1 2 2 4 2 3 3 2 2 1 1 2 1 1 1 1 3 1 4 1 2 1 2 2 3 3 4 1 2 3 2 2 2
## [445] 2 4 4 2 1 1 2 2 2 1 2 4 1 1 1 2 2 3 2 1 2 2 2 4 1 3 2 1 4 4 2 3 1 3 1 3 2
## [482] 4 3 2 2 2 4 1 2 2 2 4 1 1 1 3 4 3 1 2 2 3 2 3 2 2 1 2 3 4 4 3 2 3 4 2 1 2
## [519] 2 2 3 2 3 2 1 1 1 2 1 2 1 2 4 3 1 1 2 1 2 3 1 3
##
## Within cluster sum of squares by cluster:
## [1] 1369.6654 1809.2133 1305.2267  749.7183
##  (between_SS / total_SS =  24.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Show centroid values

```r
k_cl2$centers # without outliers, k=4
```

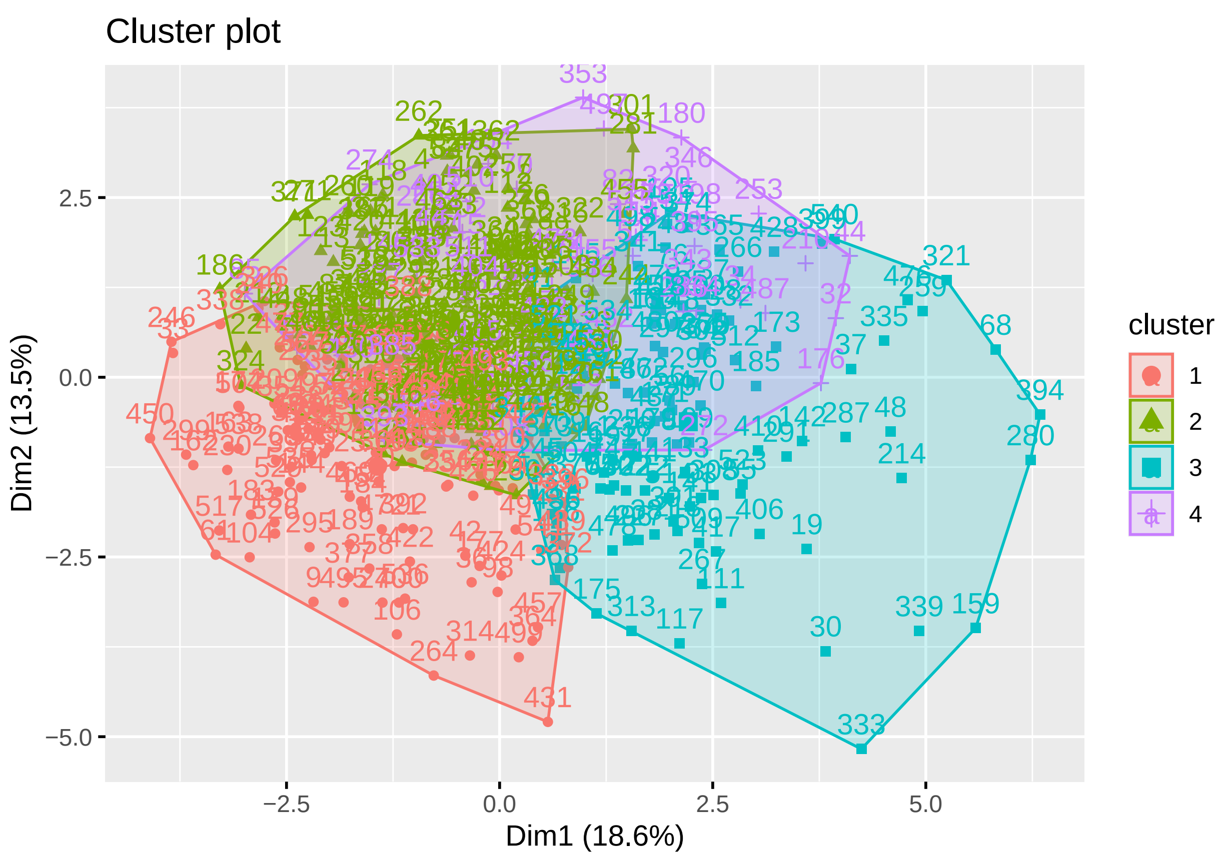
```
##   funded_amnt   int_rate   sub_grade  emp_length annual_inc loan_status
## 1 -0.49066071 -0.9558863  0.88985442 -0.39538442 -0.1335981   0.2244792
```

```
## 2 -0.36977600  0.1564132 -0.09292751 -0.12845664 -0.3920395   0.4081940
## 3  0.90475571  0.3301468 -0.31711484  0.65820435  0.4730026   0.3991074
## 4  0.04965222  0.5478342 -0.54783200  0.06047444 -0.2118357  -2.3033741
##          dti delinq_2yrs inq_last_6mths      pub_rec   revol_bal revol_util
## 1 -0.5244323 -0.12059655    -0.12296858 -0.007537363 -0.45646230 -1.0171181
## 2  0.1401578 -0.08591319    -0.16558491  0.008669210 -0.25962155  0.5209983
## 3  0.2770747 -0.01435163     0.09371561 -0.236520554  0.58350695  0.2443156
## 4  0.3066738 -0.09172565     0.11022473 -0.125347556 -0.05775889  0.3243015
##     total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1 -0.13838008  -0.08671889  -0.1135136     -0.05733001
## 2 -0.48711955  -0.08248795  -0.3623109     -0.38380484
## 3  0.70646071  -0.15547138   0.5364614      0.38431796
## 4  0.04094377  -0.09621828  -0.1676508     -0.15171259
##   months_since_last_credit_pull
## 1                    -0.1356836
## 2                     0.2774444
## 3                     0.1262042
## 4                    -0.6520861
```

```
k_cl_table<- k_cl2$centers
```

Visualise clusters

```
fviz_cluster(k_cl2, data = sample_maha_updated) # without outliers, k=4
```



Assign cluster labels to each data point in original un-normalised data

```
# Hierarchical clustering:
# without outliers, k=4
sample_maha_updated22 <- sample_maha_updated2
sample_maha_updated22$cluster <-fit1

# K-means clustering:
# with outliers, k=4
sample_maha2$cluster <- k_cl$cluster

# without outliers, k=4
sample_maha_updated2$cluster <- k_cl2$cluster
```

Left join labelled sample data with other variables in original population (df)

```
# Hierarchical clustering:
# without outliers, k=4
result_h_wo <- merge(sample_maha_updated22, df, by = c("funded_amnt", "int_rate", "emp_length", "annual_

n_distinct(result_h_wo$id) # double check that each and every row is unique
```

```
## [1] 542
```

```
write.csv(result_h_wo, "sample with outliers hcut k=4.csv",row.names = FALSE) # output as csv file for 

# K-means clustering:
# with outliers, k=4
result_k_w <- merge(sample_maha2, df, by = c("funded_amnt", "int_rate", "emp_length", "annual_inc", "dt

n_distinct(result_k_w$id) # double check that each and every row is unique
```

```
## [1] 600
```

```
write.csv(result_k_w, "sample with outliers kmeans k=4.csv",row.names = FALSE) # output as csv file for

# without outliers, k=4
result_k_wo <- merge(sample_maha_updated2, df, by = c("funded_amnt", "int_rate", "emp_length", "annual_

n_distinct(result_k_wo$id) # double check that each and every row is unique
```

```
## [1] 542
```

```
write.csv(result_k_wo, "sample without outliers kmeans k=4.csv",row.names = FALSE) # output as csv file
```

See cluster size

```
#Hierarchical Clustering:
# k=4, without outliers
table(fit1)
```

```
## fit1
##   1   2   3   4
## 169 281  24  68
```

```
# K-means clustering:
# k=4, with outliers
sample_maha2$cluster <- as.factor(sample_maha2$cluster)
sample_maha2 %>%
```

```r
  count(cluster)
```

```
##   cluster   n
## 1       1 165
## 2       2 300
## 3       3  38
## 4       4  97
```

```r
# k=4, without outliers
sample_maha_updated2$cluster <- as.factor(sample_maha_updated2$cluster)
sample_maha_updated2 %>%
  count(cluster)
```

```
##   cluster   n
## 1       1 134
## 2       2 219
## 3       3 120
## 4       4  69
```

#Validation ## Internal Validation for Sample without Outlier We choose hierarchical clustering over kmeans because hierarchical produces more distinct clusters in which we can better interpret the customer segments, by stringing the variables of interest together in a narrative to describe each customer segment.

Using domain knowledge, by looking at the centroid values, we can clearly identify 2 clusters consisting of high creditworthiness customers (using variables usually associated with creditworthiness, such as low interest rate, high credit grade,) and 2 clusters consisting of low creditworthiness customers.

Because the smallest cluster only has 24 data points, which is 4.43% of the sample size of 542, hence we decide to randomly take 200 data points for the validation sample in order to sample each cluster sufficiently.

Extract validation sample of size 200

```r
set.seed(240)
sample_validation <- sample_n(sample_maha_updated, 200)
```

Hierarchical Clustering for validation sample

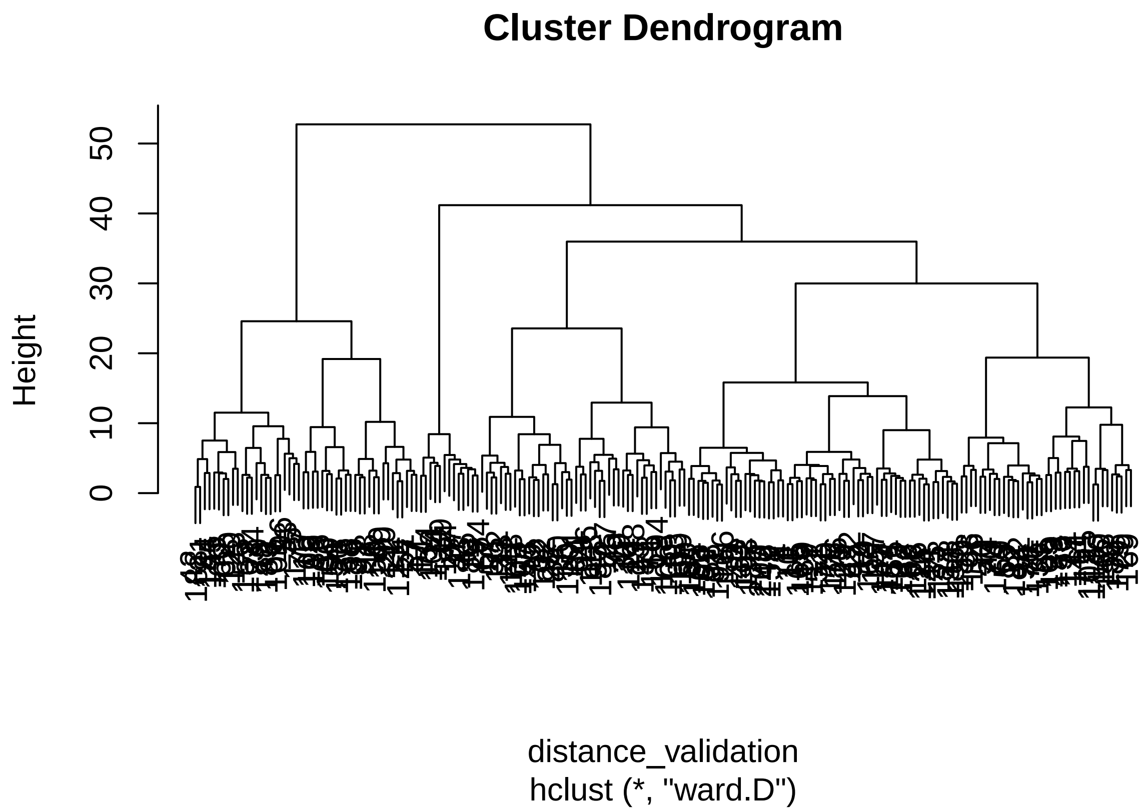
```r
distance_validation <- dist(sample_validation, method = "euclidean")

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

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

```r
Hierar_valid
```

```
##
## Call:
## hclust(d = distance_validation, method = "ward")
##
## Cluster method   : ward.D
## Distance         : euclidean
## Number of objects: 200
```

```r
plot(Hierar_valid)
```




```r
# Cutting tree by no. of clusters
fit2 <- cutree(Hierar_valid, k = 4 )
fit2
```

```
##   [1] 1 1 2 1 2 2 3 1 1 1 1 1 2 3 1 2 4 2 1 1 2 2 4 1 1 2 2 4 1 4 4 4 4 2 1 4 3
##  [38] 1 2 4 4 2 1 1 3 1 1 2 4 1 2 1 2 2 4 1 2 1 1 2 1 2 2 4 1 1 4 1 1 1 1 4 2 1
##  [75] 4 1 1 4 2 3 1 1 4 4 1 2 4 1 3 1 2 1 1 1 4 2 3 4 4 2 2 3 4 1 1 4 4 1 2 4 1
## [112] 4 1 4 2 1 1 4 3 2 2 1 2 1 1 1 4 4 1 2 1 1 1 3 1 2 1 1 1 1 1 1 3 3 2 4 1 4
## [149] 2 2 4 1 4 3 2 1 1 2 1 4 4 2 1 1 1 1 1 2 1 1 1 1 1 1 2 1 2 1 1 1 1 1 2 2 1
## [186] 1 4 2 1 2 4 4 1 4 1 1 1 4 4 1
```

```r
table(fit2)
```

```
## fit2
##  1  2  3  4
## 95 48 13 44
```

```r
final_ca2 <- cbind(sample_validation, validation_cluster = fit2)
head(final_ca2)
```

```
##   funded_amnt   int_rate   sub_grade  emp_length  annual_inc loan_status
## 1  -0.2818275  0.2444050 -0.06745994  0.53748925 -0.81141245   0.9732423
## 2  -0.8012314 -0.7329338  0.72931102 -0.06635798 -1.07115851   0.3043473
## 3   0.6958739  1.6904915 -1.97971025  1.14133648 -0.73235930  -1.0334429
## 4  -0.2818275  0.3592482 -0.22681413  0.53748925 -0.68718607   0.3043473
## 5   1.6735753  0.7037777 -0.70487671 -0.06635798  0.05817218  -2.3712330
## 6  -0.7706782 -0.7329338  0.72931102 -1.27405245  0.48731785   0.3043473
##          dti delinq_2yrs inq_last_6mths    pub_rec  revol_bal  revol_util
```

```
## 1  1.4749251  -0.4161973     -0.8030459 -0.2365206 -0.2962866  0.66648105
## 2  0.2422677  -0.4161973     -0.8030459 -0.2365206 -0.8741483 -0.64139555
## 3 -0.7193341  -0.4161973      0.1664260 -0.2365206 -0.2259568  0.48638001
## 4 -1.6357600  -0.4161973     -0.8030459 -0.2365206 -0.6302326  0.70078601
## 5 -0.6457619   1.3059985     -0.8030459 -0.2365206 -0.5360692 -0.04963499
## 6 -2.0113656  -0.4161973     -0.8030459 -0.2365206 -0.6201757 -0.59422623
##     total_acc tot_coll_amt tot_cur_bal total_credit_rv
## 1 -0.63170832  -0.24728124 -0.89816864     -0.65704033
## 2 -1.58599417  -0.24728124 -0.99996717     -1.05013358
## 3 -0.45820180   0.05249612 -0.06167051     -0.02062452
## 4 -1.67274743  -0.24728124 -0.18923531     -0.97057899
## 5  0.06231776   0.05249612 -0.06167051     -0.02062452
## 6 -1.41248765   0.05249612 -0.06167051     -0.02062452
##   months_since_last_credit_pull validation_cluster
## 1                     0.2799981                  1
## 2                     0.2799981                  1
## 3                     1.0629488                  2
## 4                     0.3918482                  1
## 5                    -2.0688540                  2
## 6                    -2.5162544                  2
```

```
hcentres2 <- aggregate(x=final_ca2, by=list(cluster=fit2), FUN="mean")
print(hcentres2)
```

```
##   cluster funded_amnt    int_rate   sub_grade  emp_length   annual_inc
## 1       1 -0.30617357 -0.40730224  0.42234452  0.00669355 -0.190702581
## 2       2  0.20575013  0.32780305 -0.35296954 -0.01608267 -0.164148529
## 3       3 -0.63741940 -0.02494459  0.04286219 -0.34505671 -0.146424314
## 4       4  0.08786582  0.61444934 -0.59984781 -0.10086510 -0.004452385
##   loan_status         dti delinq_2yrs inq_last_6mths    pub_rec   revol_bal
## 1   0.4240443  0.10506878  -0.3618122    -0.41525711 -0.2365206 -0.02010683
## 2  -1.4515023 -0.04914388  -0.2009228    -0.19712594 -0.2365206 -0.10196553
## 3   0.1499869 -0.48273140  -0.1512441    -0.05729827  3.5989479 -0.44706147
## 4   0.4107624  0.13598697   0.3274781     0.80539609 -0.2365206 -0.19678320
##   revol_util  total_acc tot_coll_amt  tot_cur_bal total_credit_rv
## 1  0.1501914 -0.2801293  -0.06583144 -0.115683673      -0.1089344
## 2  0.2148884 -0.2467407  -0.10939973 -0.112562117      -0.1241447
## 3 -0.1954311 -0.5516284  -0.15504205  0.060328954      -0.4258957
## 4  0.1048348  0.3836992  -0.04694002 -0.003014259      -0.1789522
##   months_since_last_credit_pull validation_cluster
## 1                    0.48486037                  1
## 2                   -1.21366678                  2
## 3                    0.02188247                  3
## 4                    0.17577412                  4
```

Compare original cluster number to validation cluster number

```
result_validation<- merge(final_ca2, final_ca, by = c("funded_amnt", "int_rate", "sub_grade", "emp_leng

# creates new column 'equal' that tells us whether original cluster number is equal to validation clust
result_validation <- result_validation %>% mutate(equal = cluster == validation_cluster)

# calculates the % of FALSE over total, which indicates the % if cases assigned to different cluster
sum(result_validation$equal == FALSE)/nrow(result_validation)
```

```
## [1] 0.555
```

55.5% of cases are assigned to a different cluster, suggesting that our solution is rather unstable.