Polytechnic University of Catalonia

Multivariate Analysis

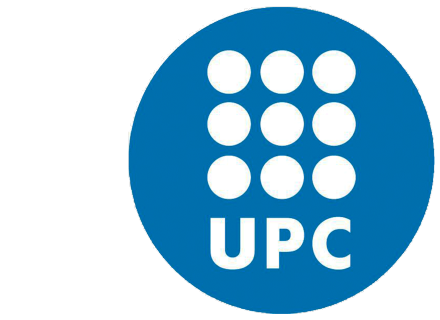
D3. Project development

Authors

Campeny, Eloi
Chriki, Fatima Zohra
Dai, Zhongkai
González, Victor
Moure, Ximena
Xu, Ange

Teachers

Conti, Dante
Gibert, Karina
Ramírez, Sergi



UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

October ◇ QT - 2022/23

Index

1. Motivation and problem definition

Google Play store is a digital distribution platform for mobile applications for devices with Android operating system, as well as an online store.
There are more than 2.5 million apps on the store and the service is used by millions of users on a daily basis. What's more, every day more than 2500 apps are added to it.

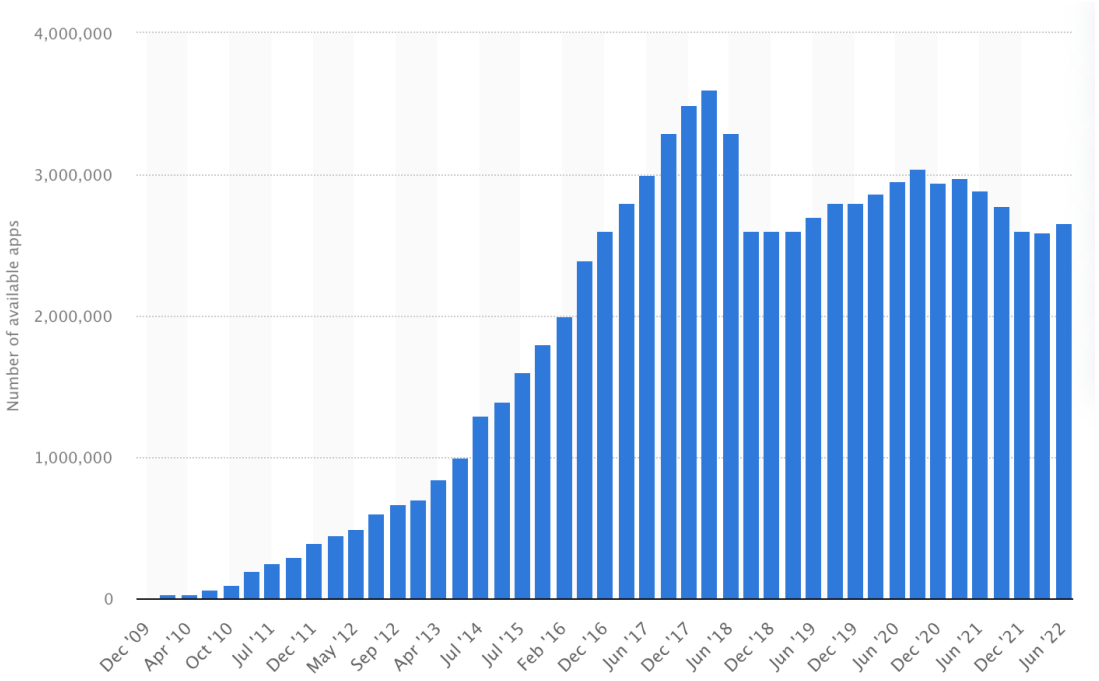


Figure 1: Number of available applications in the Google Play Store

The number of app downloads has been increasing throughout the years. In 2021, users worldwide downloaded approximately 111 billion mobile apps.
A vast majority of apps in the store are free to download so the monetization strategy may vary. This is why an application's success is often measured by the number of installations and the user reviews it has.
An application rating is based on voluntary feedback from users. This can lead to biased ratings due to insufficient or missing votes.
The aim of this study is to analyze which factors can influence the number of downloads and the rating of an app. This can potentially help developers to better understand the mobile application market.

2. Data source presentation

The dataset used in this project is taken from Kaggle and was collected in June 2021. It is publicly accessible and can be retrieved from the following url: https://www.kaggle.com/datasets/gauthamp10/google-playstore-apps

As the original dataset was quite large, containing 2312944 observations, we decided to reduce it to 20000 observations. The final dataset, as well as the script used to obtain it, can be found at the following url: https://drive.google.com/drive/folders/1B-6fwHjjXAG-JSsAIGmQOihYE80M2_3m?usp=sharing

3. Data structure and metadata description

3.1 Description of initial data matrix

The dataset contains 20000 entries and 24 variables, out of which 5 are numerical, 4 are binary and 15 are categorical, which accounts for 480000 data entries. Some of the categorical variables were transformed during the preprocessing phase into numerical ones (Installs, Size, Minimum.Android) in order to fulfill the requirement of having at least 7 numerical variables.

In total there are 10911 missing values in the dataset for a total of 480000 values. Therefore, 2.27 % of the values are missing.

Numerical variables

- Rating 344 missing 1.72%
- Rating.Count 344 missing 1.72%
- Minimum.Installs 1 missing 0.005%
- Maximum.Installs 0 missing 0%
- Price 0 missing 0%

Binary variables

- Free 0 missing 0%
- Ad.Supported 0 missing 0%
- In.App.Purchases 0 missing 0%
- Editors.Choice 0 missing 0%

Qualitative variables

- App.Name 0 missing 0%
- App.Id 0 missing 0%
- Category 0 missing 0%

● Installs	1 missing	0.005%
● Currency	1 missing	0.005%
● Size	0 missing	0%
● Minimum.Android	49 missing	0.245%
● Developer.Id	0 missing	0%
● Developer.Website	6097 missing	30.485%
● Developer.Email	0 missing	0%
● Released	750 missing	3.75%
● Last.Updated	0 missing	0%
● Content.Rating	0 missing	0%
● Privacy.Policy	3324 missing	16.62%
● Scraped.Time	0 missing	0%

Furthermore, the dataset contains some variables that do not provide useful information, such as the developer email and website, app identification, privacy policy.

3.2 Metadata table

URLs
- Original source
- Reduced dataset

Inclusion criteria: Google Play Store Apps collected in the month of June 2021.

No. of rows: 20000
No. of columns: 24

Variable	Modalities	Meaning	Type	Measuring unit	Missing code	Measuring procedure	Range	Role
App.Name		Name of the app	Categorical nominal (factor)					Explanatory
App.Id		Package name	Categorical nominal (factor)					Explanatory
Category	48 modalities in total	App category	Categorical nominal (factor)					Explanatory
Rating		Average rating	Numerical continuous (numeric)	Star			[0, 5]	Response
Rating.count		Number of ratings	Numerical discrete (integer)	Rating				Explanatory
Installs		Approximate install count	Categorical nominal (factor)					Explanatory
Minimum.Installs		Approximate minimum app install count	Numerical discrete (integer)	Install				Explanatory
Maximum.Installs		Approximate maximum app install count	Numerical discrete (integer)	Install				Explanatory
Free	True, False	Whether app is Free or Paid	Categorical binary [True, False] (factor)					Explanatory
Price		App price	Numerical continuous (numeric)	Currency				Explanatory
Currency		App currency	Categorical nominal (factor)					Explanatory
Size		Size of application package	Categorical nominal (factor)					Explanatory
Minimum.Android	32 in total	Minimum android version supported	Categorical nominal (factor)					Explanatory

Developer.Id		Developer Id in Google Playstore	Categorical nominal (factor)					Explanatory
Developer.Website		Website of the developer	Categorical nominal (factor)					Explanatory
Developer.Email		Email-id of developer	Categorical nominal (factor)					Explanatory
Released		App launch date on Google Playstore	Temporal (Date)					Explanatory
Last.Updated		Last app update date	Temporal (Date)					Explanatory
Content.Rating	Everyone, Teen, Mature 17+, Everyone 10+, Adults only 18+	Maturity level of app	Categorical nominal (factor)					Explanatory
Privacy.Policy		Privacy policy from developer	Categorical nominal (factor)					Explanatory
Ad.Supported	True, False	Ad support in app	Categorical binary [True, False] (factor)					Explanatory
In.App.Purchases	True, False	In-App purchases in app	Categorical binary [True, False] (factor)					Explanatory
Editors.Choice	True, False	Whether rated as Editor Choice	Categorical binary [True, False] (factor)					Explanatory
Scraped.Time		Scraped date-time in GMT	Temporal (Date)					Explanatory

3.3 Final scope of study

After a thoughtful consideration of which variables were adequate for our study, we decided to discard the following variables: `App.Id`, `Developer.Id`, `Developer.Website`, `Developer.Email`, `Installs` and `Privacy.Policy`. We consider that these variables do not provide any insight into what we are trying to accomplish. This was made during preprocessing.
Additionally, we created new variables and transformed other ones to numerical. All of this is explained in detail in the preprocessing section.
The cleansed dataset contains 12 variables, out of which 7 are numerical, 3 are categorical and 2 are binary.

4. Preprocessing

This section includes all the different preprocessing tasks carried out in this project together with the results obtained. Several preprocessing methods were applied in order to improve the quality of the dataset, which are:

- Feature selection
- Feature derivation
- Variable transformations
- Segmentation of population
- Identifying and handling missing data
- Detection of outliers
    - Univariate outliers
    - Multivariate outliers

In the following subsections, we present in detail each of the preprocessing methods that were carried out.

4.1 Feature selection

The aim of feature selection is filtering the uninteresting variables and to remove the noise in the dataset. After analyzing the `Playstore` dataset, many irrelevant variables were detected. The criteria followed to apply feature selection is the following:

- **Variables with unique value for each observation:** there were detected five variables in the dataset, which are `App.Id`, `Developer.Id`, `Developer.Website`, `Developer.Email` and `Privacy.Policy`. All these features were removed.
- **Variables with unique value for all observations (Constants):** one logical variable detected in the dataset, called `Editors.Choice`, was removed because 98% of the observations had the same value.
- **Variables highly correlated with other variables:** the dataset includes two columns with the same values, one called `Installs` and the other one `Minimum.Installs`, the difference between these columns is that the first one is categorical and the second one numerical. Since `Minimum.Installs` includes the same values as `Installs`, this last one was removed. After studying the correlation between the numerical data, the results show that `Minimum.Installs` is highly correlated with `Maximum.Installs` with a 0.98 correlation coefficient. In order to decide which column to keep and which one to remove, we studied the distribution of each column.

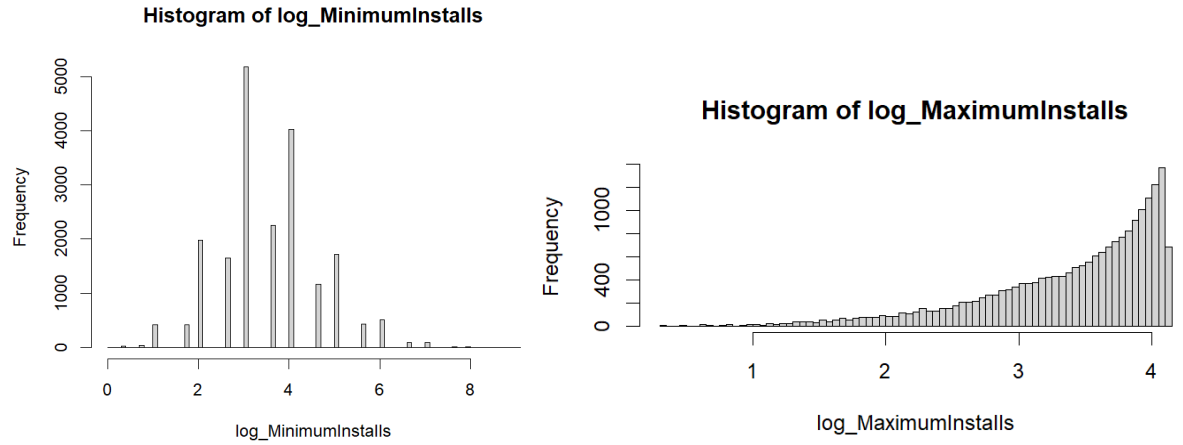


Figure 2: Distribution comparison

Comparing the distributions shown in Figure 2, `Minimum.Installs` includes dispersed values and distant ranges while `Maximum.Installs` seems to follow an exponential distribution. Therefore, `Minimum.Installs` was removed and `Maximum.Installs` was kept and named as `Installs`.

The unnecessary columns mentioned before were removed from the dataset, therefore, the dimension of the dataset was reduced from 24 to 15 columns. Throughout the following preprocessing process, the dimension of the dataset will be further reduced due to the transformation of some columns.

4.2 Feature derivation

We carried out some variable derivations in order to have more numerical variables in the dataset.

First, we decided to derive a new numerical variable called `DaysLastUpdate` from the variables `Scraped.Time` and `Last.Updated`. This new variable indicates the number of days passed since the last app update until the time it was scraped.

Afterwards, we derived from `Scraped.Time` and `Released` a new numerical variable called `ReleasedDays`. This other variable indicates the number of days since the app was released until the scrapped time.

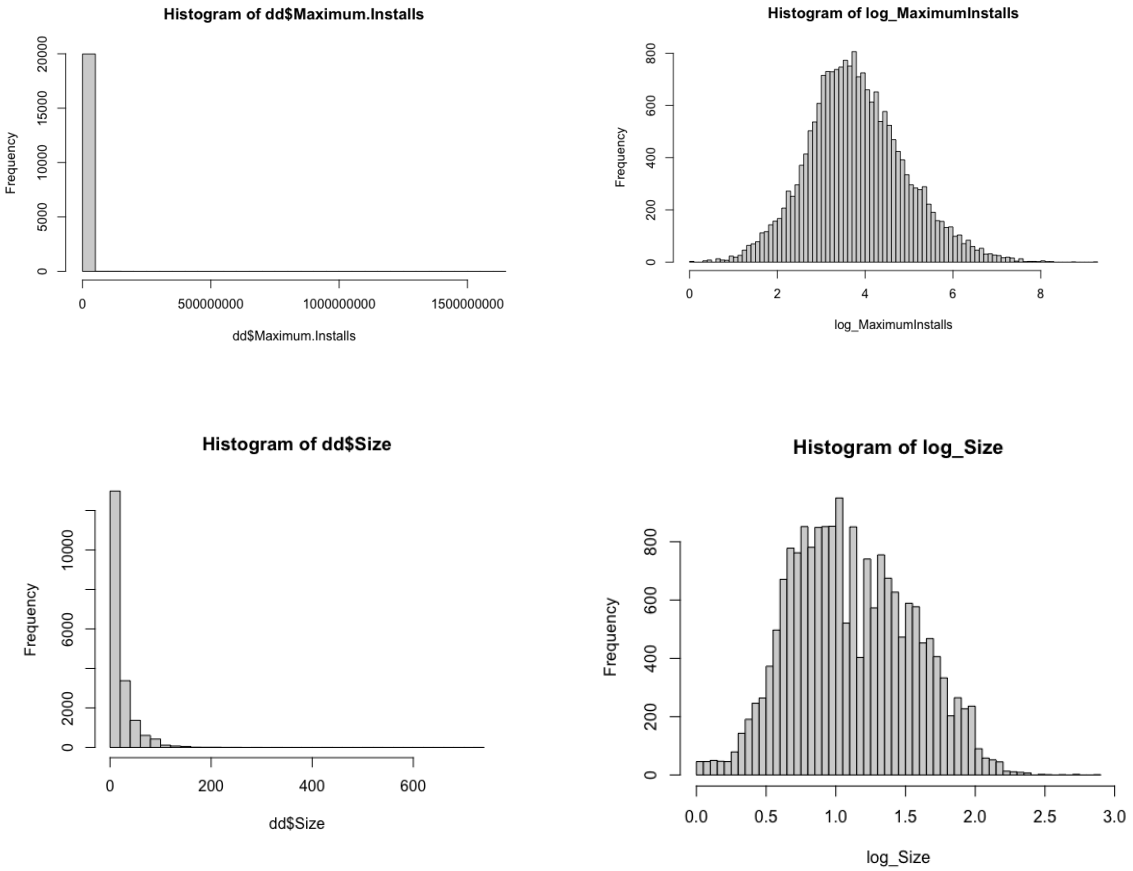
After the derivation of these variables, it did not make any sense to keep the variables `Released`, `Last.Updated` and `Scraped.Time` so we decided to remove them from the dataset.

Finally, we also researched whether an app's name has any impact on its visibility in the `Appstore`. We concluded that it certainly does. Therefore, we decided to do one more derivation which was to derive a new numeric variable called `AppNameLen` from the variable `App.Name` which accounts for the length of the name of the app.

4.3 Variable transformations

After analyzing the dataset we found that the variable called `Size` was a categorical one and it had different measures. The Size of data was in GB, MB and KB. So we converted all the data to MB and then we transformed it to a numerical variable.

Another decision we made, after looking at the histograms in Figure 3 to see the distribution of the values in the dataset, was to apply a logarithmic transformation to the variables `Size`, `Installs` and `Rating.Count`. The reason behind this is that the large range between their values could be a problem for some algorithms and that it is easier to visualize the data.



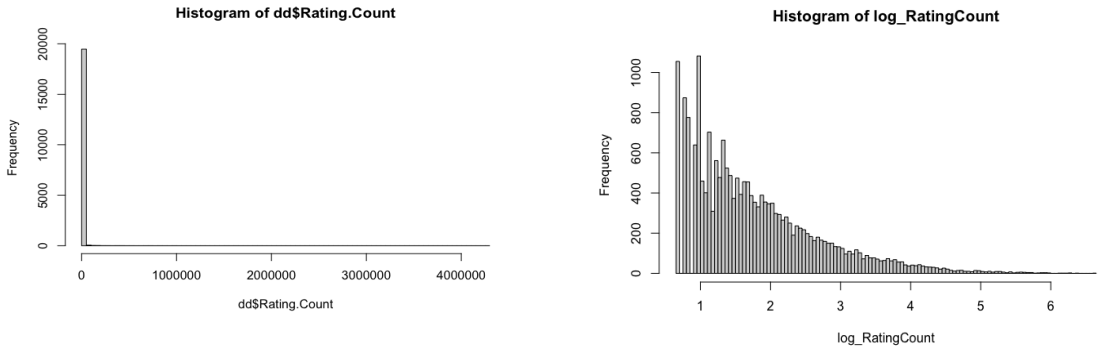


Figure 3: Histograms of the variables before and after applying a logarithmic transformation

Finally, the last transformation we did was to reduce the modalities of the categorical variables `Category` and `Minimum.Android`.
`Category` had 48 levels and we group them in 6 levels and `Minimum.Android` had 32 levels and we group them in 9 levels.

4.4 Segmentation of population

To assess if we should keep free and paid apps we performed a Kolmogrov-Smirnov test (KS test). This test is used to test whether two samples follow the same distribution or not. If they do not follow the same distribution they are two different populations.

In order to do this, we separated the data into two groups: one which contained the paid apps and another one with the free ones. After this, we performed the KS test for each numerical variable using both datasets.

As depicted in Figure 4, for almost every numerical variable, the KS test gave us a p-value less than 0.05. For this reason we concluded that there is more than one population (free and paid apps) and therefore we decided to analyze only free apps as it has the biggest population.

Following this decision, we eliminated from the dataset the two variables called `Currency` and `Price` as we no longer need them since we are going to analyze only free apps.

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$Rating and free$Rating
D = 0.061604, p-value = 0.08275
alternative hypothesis: two-sided
```

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$Rating.Count and free$Rating.Count
D = 0.12337, p-value = 0.000005664
alternative hypothesis: two-sided
```

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$Minimum.Installs and free$Minimum.Installs
D = 0.20459, p-value = 0.0000000000000009992
alternative hypothesis: two-sided
```

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$Maximum.Installs and free$Maximum.Installs
D = 0.2214, p-value < 0.00000000000000022
alternative hypothesis: two-sided
```

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$DaysLastUpdate and free$DaysLastUpdate
D = 0.19473, p-value = 0.00000000000002776
alternative hypothesis: two-sided
```

```
        Two-sample Kolmogorov-Smirnov test

data:  pay$Size and free$Size
D = 0.11604, p-value = 0.00005687
alternative hypothesis: two-sided
```

Figure 4: Test results for Kolmogorov-Smirnov Test

4.5 Identifying and handling of missing values

To obtain clean data it is necessary to deal with the missing values that could be present in it, so the first step for the treatment of missing data would be to identify the variables affected. Figure 5 shows the variables affected with missing values and the amount they have.

$Continuous

	label	var_type	n	missing_n	missing_percent	mean
Rating	Rating	<dbl>	19227	342	1.7	4.1
Rating.Count	Rating.Count	<int>	19227	342	1.7	3342.7
Size	Size	<dbl>	18646	923	4.7	20.3
DaysLastUpdate	DaysLastUpdate	<dbl>	19569	0	0.0	562.5
ReleasedDays	ReleasedDays	<dbl>	18829	740	3.8	1195.8
AppNameLen	AppNameLen	<int>	19569	0	0.0	23.1
Installs	Installs	<int>	19569	0	0.0	431313.5

$Categorical

	label	var_type	n	missing_n	missing_percent	levels_n
Category	Category	<fct>	19569	0	0.0	6
Minimum.Android	Minimum.Android	<fct>	18921	648	3.3	8
Content.Rating	Content.Rating	<fct>	19569	0	0.0	6
Ad.Supported	Ad.Supported	<fct>	19569	0	0.0	2
In.App.Purchases	In.App.Purchases	<fct>	19569	0	0.0	2

Figure 5: Variables with missing values

The next step would be to determine the type of missing data present. The little test was performed to determine if the data was missing completely at random, but as the results obtained in Figure 6 shows, that was not the case, so it was necessary to determine if the data was random missing or not.

A tibble: 1 × 4

	statistic	df	p.value	missing.patterns
	<dbl>	<dbl>	<dbl>	<int>
1	551.	29	0	7

Figure 6: Result of the Little test

Analyzing the Figures 7, 8, 9 and 10, the type of missing data can be determined.

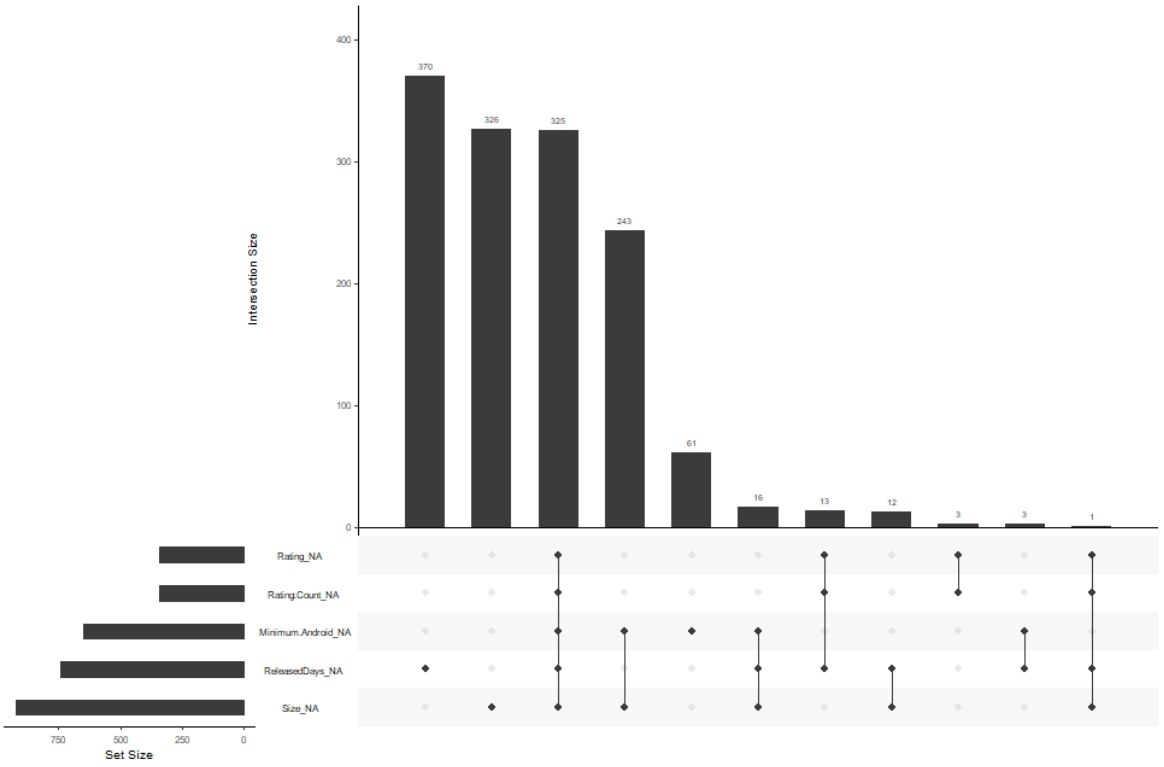


Figure 7: Amount of missing values for every pattern present in the data

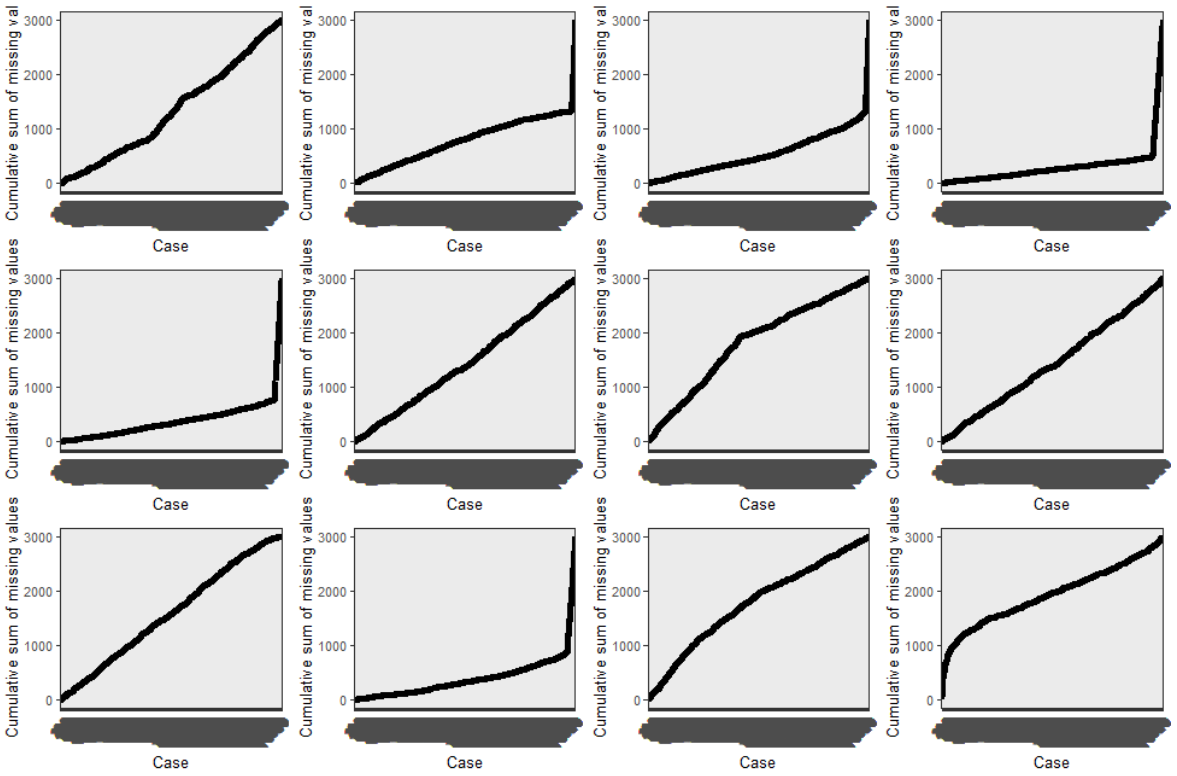


Figure 8: The graphs count the accumulative missing values for each variable ordered in ascending order: Category, Rating, Rating.Count, Size, Minimum.Android, Content.Rating, Add.Supported, In.App.Purchases, DaysLastUpdated, ReleasedDays, AppNameLen, Installs

Missing Data analysis				
Rating		Not missing	Missing	p-value
Content.Rating	only 18+	1	0	0297
	Everyone	16591	286	
	Everyone +10	332	10	
	Mature 17+	567	7	
	Teen	1735	39	
	Unrated	1	0	
In.App.Purchases	FALSE	16874	301	0.955
	TRUE	2353	41	
Installs	Mean	438640.9	19371	0.545
Rating.Count		Not missing	Missing	p-value
Content.Rating	only 18+	1	0	0.297
	Everyone	16591	286	
	Everyone +10	332	10	
	Mature 17+	567	7	
	Teen	1735	39	
	Unrated	1	0	
In.App.Purchases	FALSE	16874	301	0.955
	TRUE	2353	41	

Installs	Mean	438640.9	19371	0.545
Size		Not missing	Missing	p-value
Installs	Mean	399106.2	1081950.9	0.111
Minimum.Android		Not missing	Missing	p-value
Content.Rating	only 18+	1	0	0.511
	Everyone	16314	563	
	Everyone +10	328	14	
	Mature 17+	563	11	
	Teen	1714	60	
	Unrated	1	0	
Installs	Mean	399106.2	108195.,9	0.11
ReleasedDays		Not missing	Missing	p-value
Content.Rating	only 18+	1	0	0,833
	Everyone	16245	632	
	Everyone +10	327	15	
	Mature 17+	556	18	
	Teen	1699	75	
	Unrated	1	0	
Installs	Mean	429944.8	466139.5	0.939

Figure 9: Missing data analysis

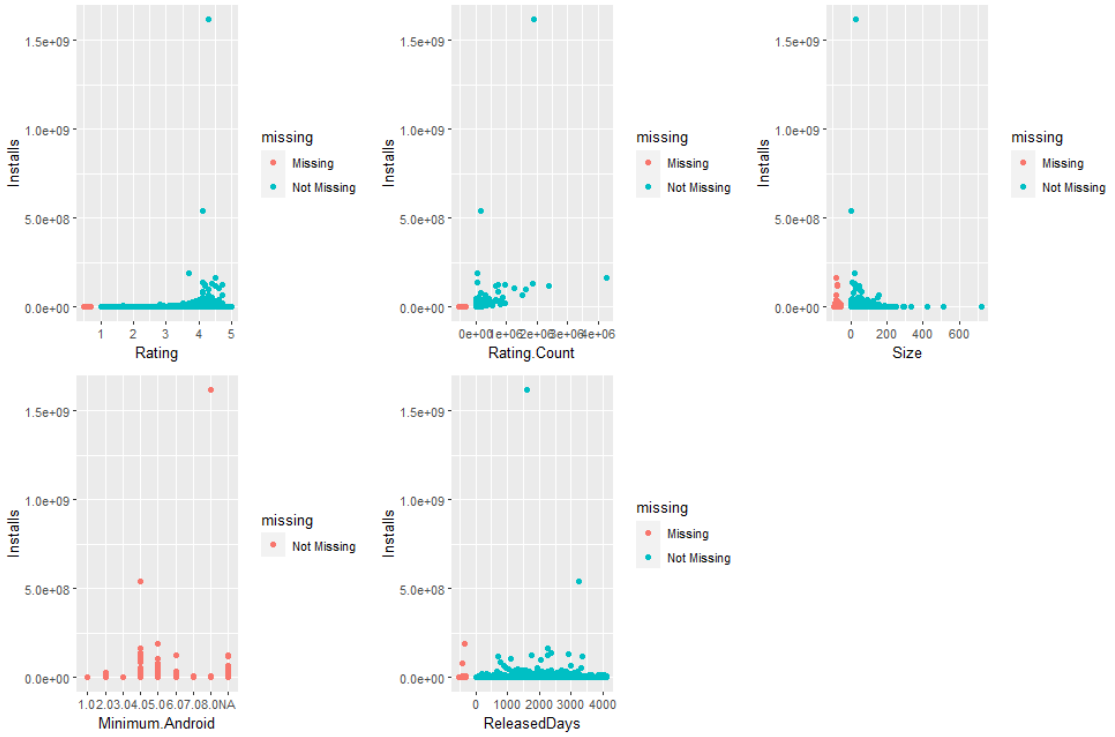


Figure 10: Missing values for every variable with missing data depending of number of installs

It can be appreciated that all the missing values have some dependency with the `Installs` variable, because for low values of `Installs` there is a concentration of missing values, mostly due to the concentration of rows following the pattern of 5 missing values per row. Because of this pattern of missing data can be explained by the number of installs (observed variable), having much less information for low installs, making the data to be missing at random.

To treat the missing values first the data is separated into numerical and categorical, then to perform the numerical imputation the MICE method is applied. In Figure 11 the frequency of the variables before and after the imputation can be observed.

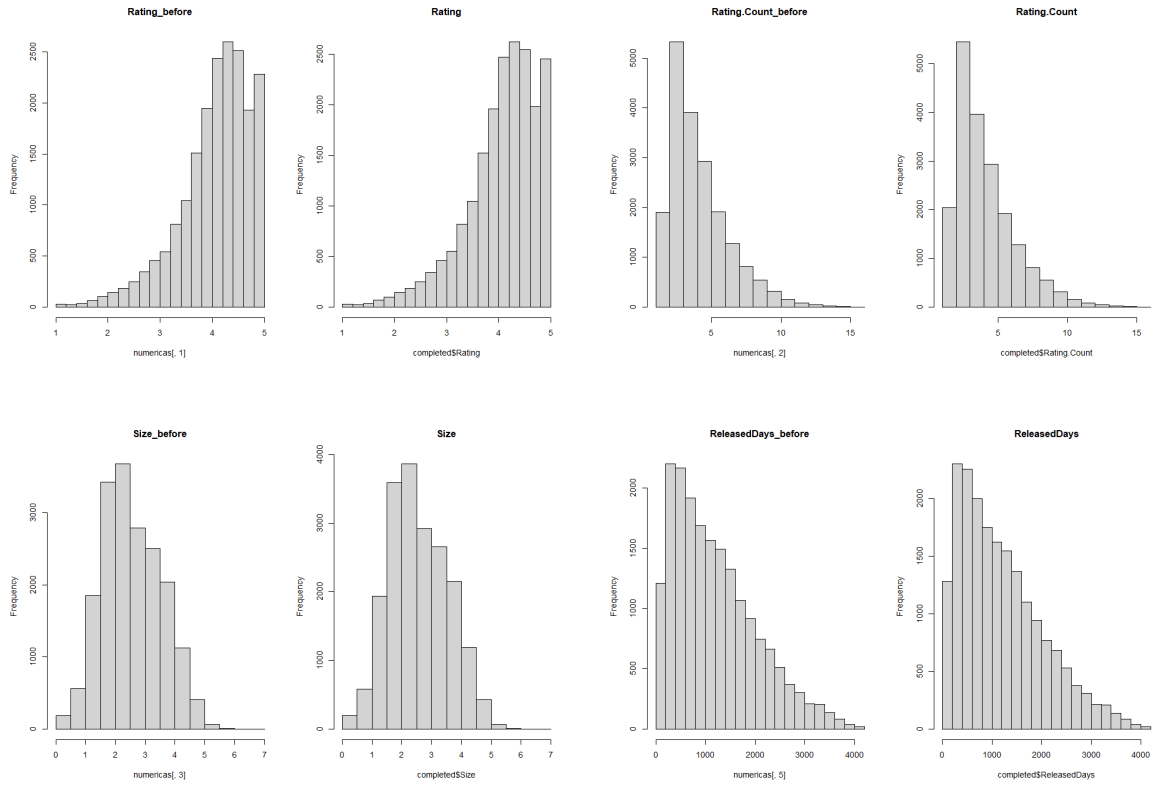


Figure 11: Histograms of Rating, Rating.Count, Size and ReleasedDays before and after the imputation

For the categorical values the MICE method is used and once all the data has been imputed then both tables are rejoined. Figure 11 shows the comparison before and after the imputation.

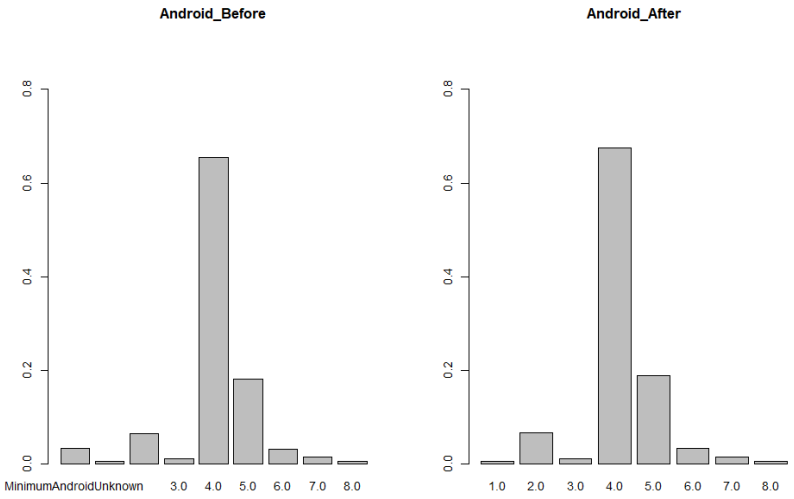


Figure 12. Comparison of `Minimum.Android` before and after de imputation

All the imputed variables have the same distribution after the imputation, which implies that the imputation has been done correctly.

4.6 Univariate detection of outliers

The aim of this section is to detect univariate outliers on numerical features and decide if it is better to keep outliers or just remove them from the main dataset and store them into the outlier dataset for further analysis.

Before moving to the analysis, it is better to point out that we only analyze the variables that we did not remove before (eg. `Minimum.Installs`) and if the variable was transformed we analyze its transformation.

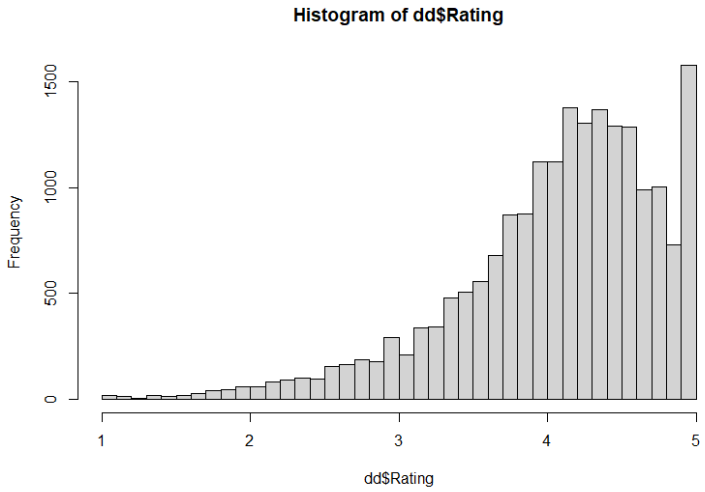


Figure 13: Histogram of `Rating`

First, we analyze the target value `Rating`, as the Figure 13 shows, `Rating` has a distribution that looks like an χ^2 except for the apps that are rated with 5. In this value there are more individuals than there should be if `Rating` follows this type of distribution.

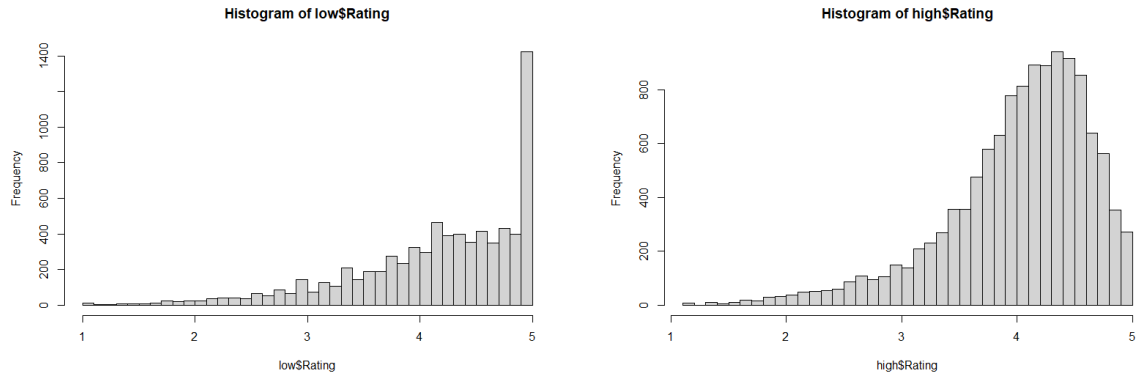


Figure 14: Histogram of Low Rating (left) and High Rating (Right)

We try to find out what was the reason behind this unusual behavior. We discovered that the apps with less than 20 votes have a completely different distribution, we checked this visually (Figure 14) and with Kolmogorov–Smirnov test. So we decided to separate these two populations, for our analysis we kept the apps with more or equal than 20 votes and the others were moved into the outlier dataset.

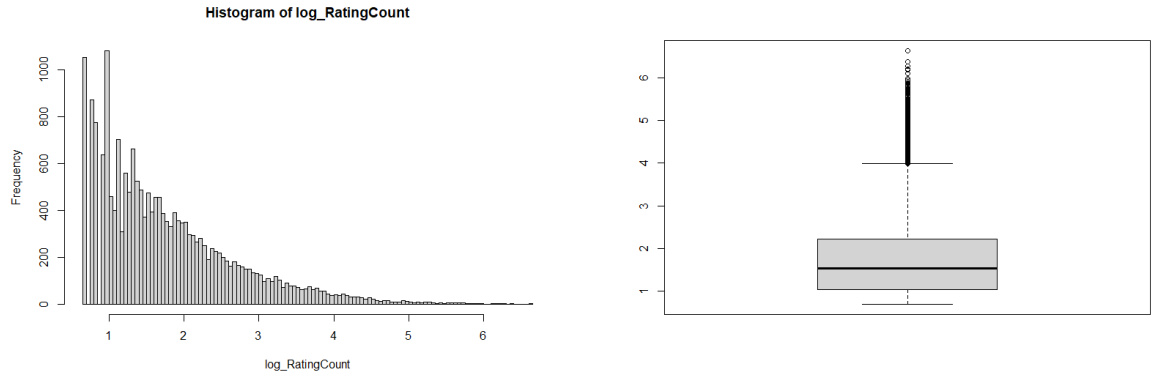


Figure 15: Histogram and box plot of the logarithm of `Rating.Count`

For analyzing the logarithm of the `Rating.Count`, we plotted an histogram and a box plot (Figure 15). In the first one we can see that it follows an exponential distribution, the only remarkable thing about the histogram is that it has some hollows but probably done by artifacts. Box plot has more interesting information, the first one is the tail that sticks out of the IQR method. This can be outliers, but it is important to remember that we have a large dataset of 20000 individuals. Since we

have this huge amount of individuals and they are close together, we decided to keep them. However, the second interesting thing is that on top of the tail there is a lonely individual. In this case we consider this individual as an outlier, and we changed its value for a missing value.

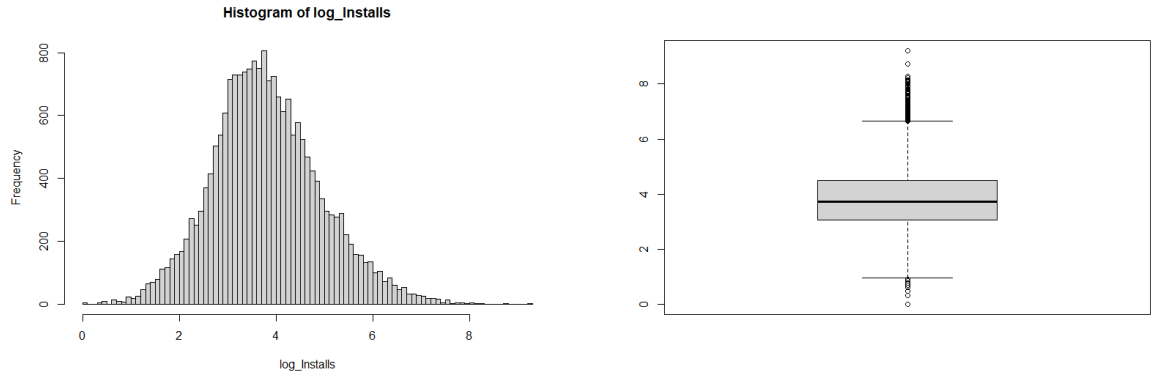


Figure 16: Histogram and box plot of the logarithm of `Installs`

In the analysis of the logarithm of the `Install`, we plotted a histogram and box plot (Figure 16). In the first one we can see that it follows a normal distribution, however the plot is not centered, this gives some clues about some outlier presence. Box plot has more interesting information, like the tails that stick out of the IQR method. This is the same case as the analysis of the logarithm of `Rating.Count`. Moreover, we have the same lonely individuals, in this case we have two on top and one on the bottom. We follow the same procedure as the case of the logarithm of `Rating.Count`, we erase the values and put a missing.

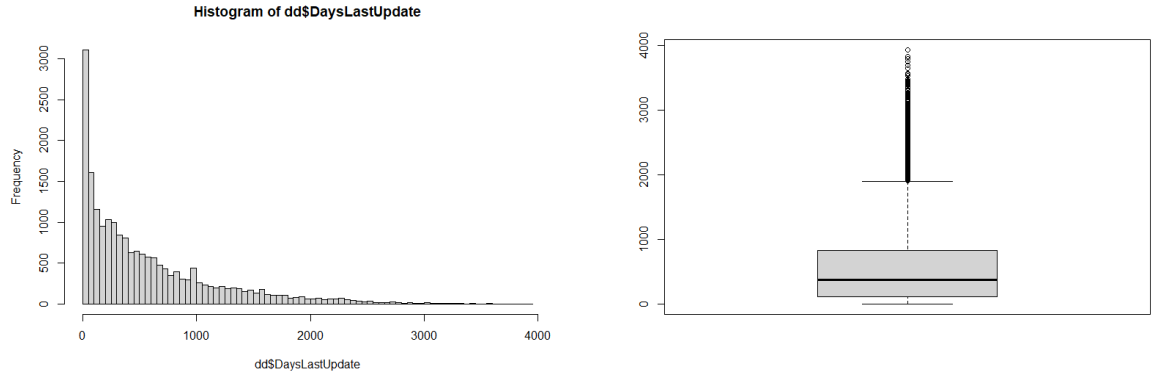


Figure 17: Histogram and box plot of `DaysLastUpdate`

In the analysis of `DaysLastUpdate`, we plotted a histogram and a box plot (Figure 17). In the first one, we can observe that it follows an exponential distribution. Box plot has the same tail that we have seen in other variables. Like in the other cases we kept the tail, however, in this case we did not erase the top value because the individuals are closer than in the other cases.

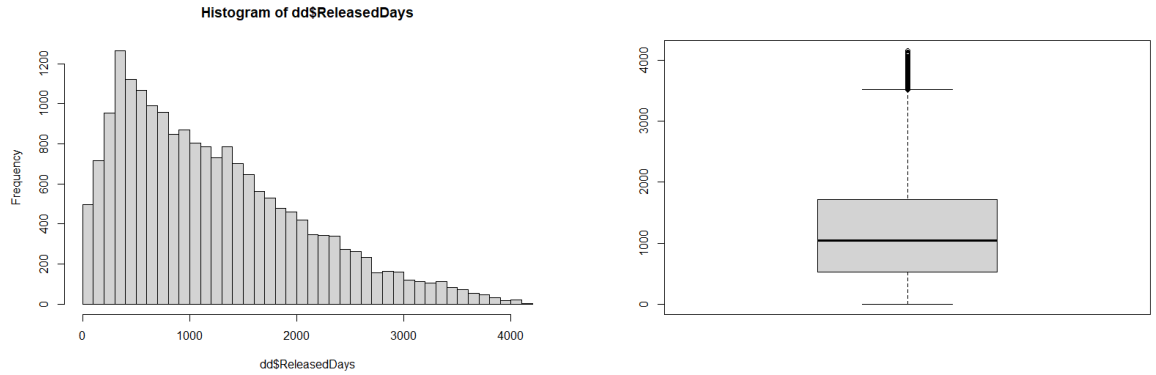


Figure 18: Histogram and box plot of `ReleasedDays`

The last variable that we analyze is `ReleasedDays`, as we have done with the other variables we make a histogram and box plot (Figure 18). The histogram shows an unusual distribution that does not look like any well known distribution. In addition, the box plot has a top tail like many other variables, so we kept these individuals on the main dataset.

4.7 Detection of multivariate outliers

In order to detect multivariate outliers we use the Mahalanobis distance. We use the r function of Mahalanobis distances for detecting and moving these individuals from the main dataset to the outliers dataset.

5. Basic initial descriptive statistics of preprocessed variables and conclusions

5.1 Univariate analysis

In this section, our cleaned and preprocessed dataset will be visualized by means of box plots, histograms and pie charts. This will help us to understand how the variables are distributed.

As a first step, we started by simply visualizing some basic information like mean, median, min, max for the numeric variables. We didn't notice any value out of the expected.

	Min	1st Q.	Median	Mean	3rd Q.	Max
Rating	1.600	3.800	4.200	4.078	4.500	5.000
Rating.Count	3.045	3.738	4.654	5.148	6.116	13.013
Size	0.01094	1.85630	2.56495	2.63017	3.36730	6.23637
DaysLastUpdate	0.0	91.0	325.0	534.4	787.0	3069.0
ReleasedDays	8	584	1146	1286	1837	4085
AppNameLen	1.00	14.00	22.00	24.18	31.00	50.00
Installs	3.664	8.661	9.863	10.048	11.303	18.165

Figure 19: Numerical variables summary

We then plotted the histograms and the box plot for all the numerical variables. From them we can see that people tend to not give a rating to apps. Another thing that results from these plots is that when they do tend to give a rating of 4+.

Most apps have a name with a length between 10 and 30 characters, but there are also apps with very long names according to the conventions of app naming.

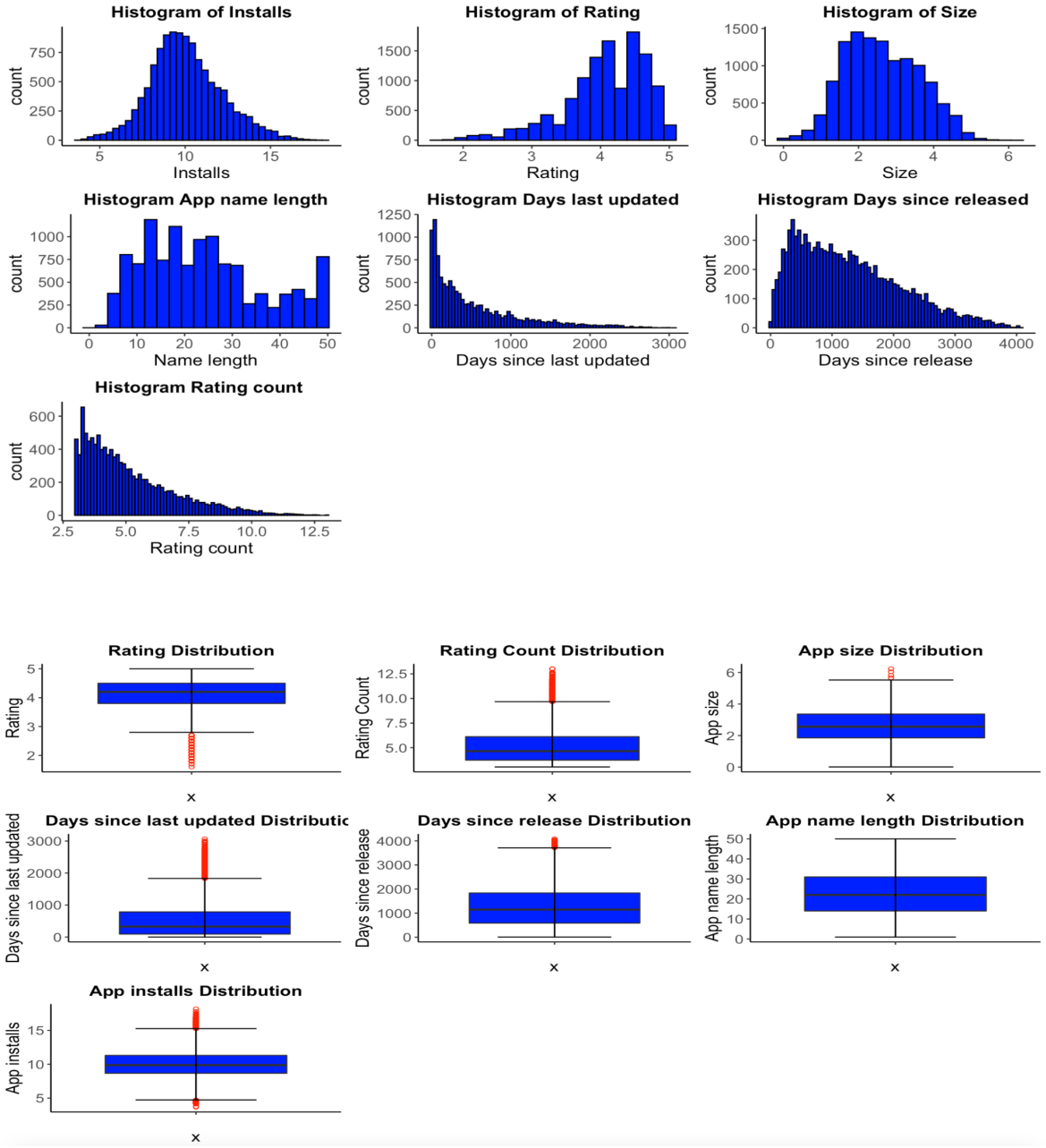


Figure 20: Histograms and box plot for numerical variables

If we look at Figure 21 it can be seen that the vast majority of apps have in-app purchases and that almost 60% of the apps support advertisements on them.

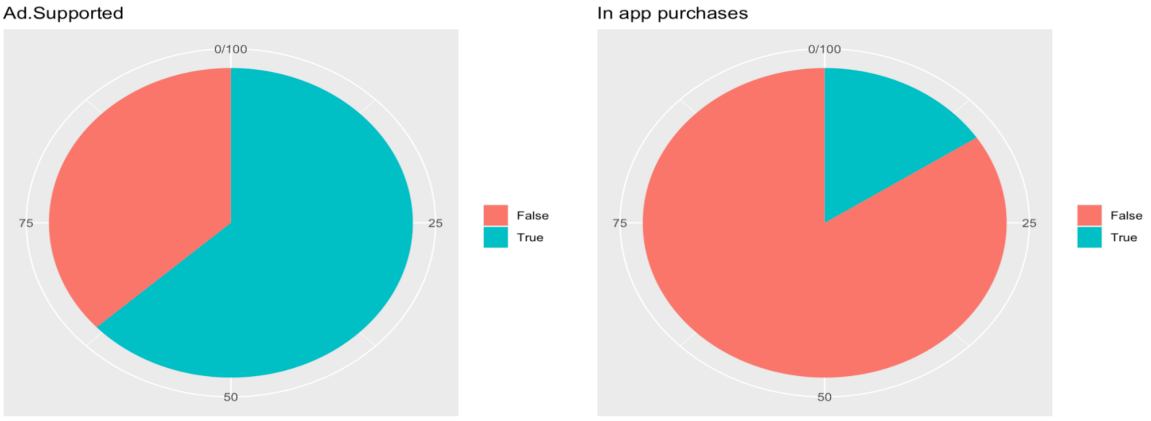


Figure 21: Ad supported and in app purchases pies

From Figure 22 we can conclude that most of the apps on the store belong to the Educational and the Lifestyle categories. Followed by the ones belonging to the Game and the Entertainment ones.

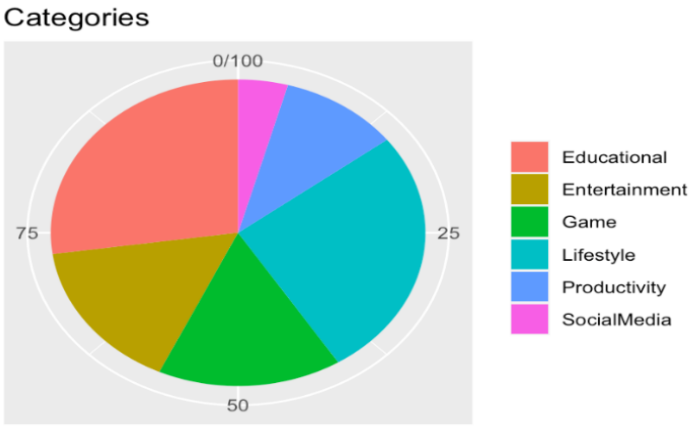


Figure 22: Pie chart for Category

As we can appreciate in figure 23 the vast majority of apps on the store, more than 80%, are suitable for every age, since their content rating is Everyone.

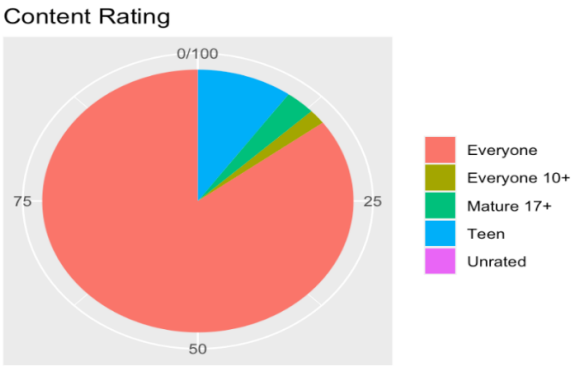


Figure 23: Pie chart for Content Rating

In **Figure 24**. we can see that the vast majority of the apps on the store require at least a version 4 of Android.

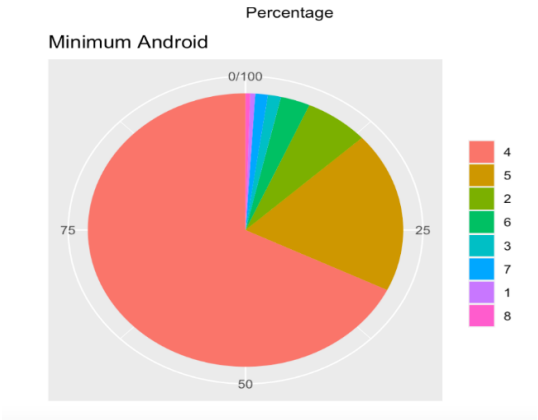


Figure 24: Pie chart for Content Rating

5.2 Bivariate analysis

In this section, we analyze the relationship between different pairs of variables in our data in order to determine if there are some patterns between the features of the different apps. The analysis was made mainly using visualization tools.

5.2.1 Correlation matrix

Firstly, in order to have a global scope of the relation of all variables of our data, we print the correlation matrix. From it we can establish that there is a significant positive correlation between `Rating.Count` and `Installs`. So the more `Installs` the more `Rating.Count`. There is also a light correlation between `ReleasedDays` and `DaysLastUpdate`, the “older” an app is, the longer the time since the last update.

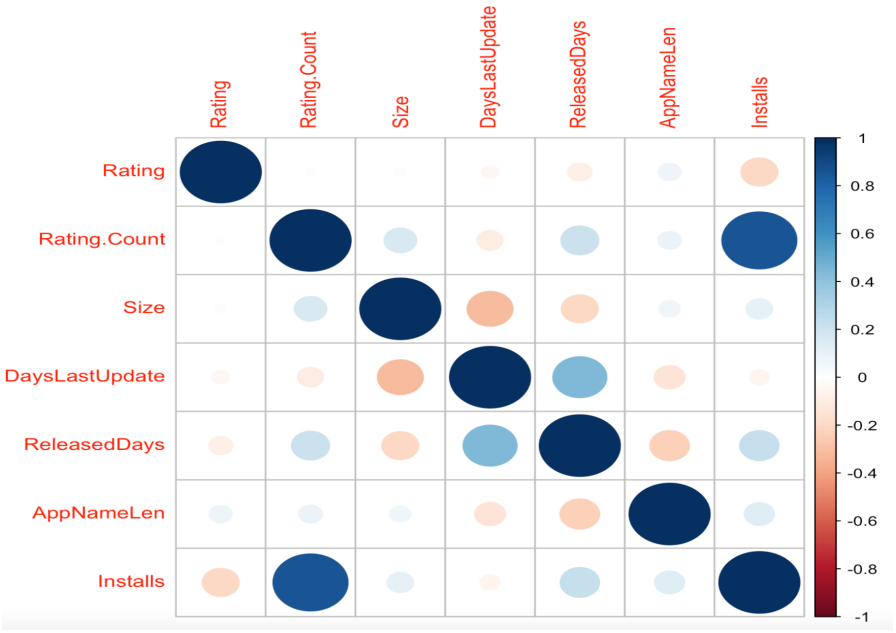


Figure 25: Correlation matrix

5.2.1.1 Rating vs Installs

To study the relationship between Rating and Installs, we print a scattershot using these two variables. From the plot we can see that an app that has a lower number of installs has a rating above 4. This can be explained by the fact that when an app has few downloads, it will generally have few reviews and they are usually good reviews, so the rating number is going to be high. Once an app starts getting more downloads it also starts having more reviews, so the rating is more dispersed. Finally, the apps that achieve more installs have higher scores than the ones in the center of installs.

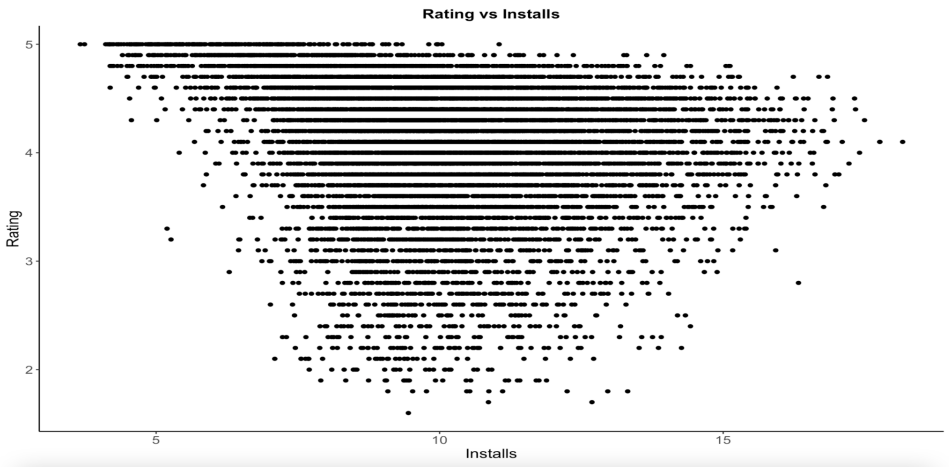


Figure 26: Rating vs Installs

5.2.1.2 Installs vs Size

We also tried to see if there is any relationship between `Installs` and `Size`. In this case, the results show, as we have already seen in the correlation matrix, that there is not a clear correlation between `Size` and `Installs`.

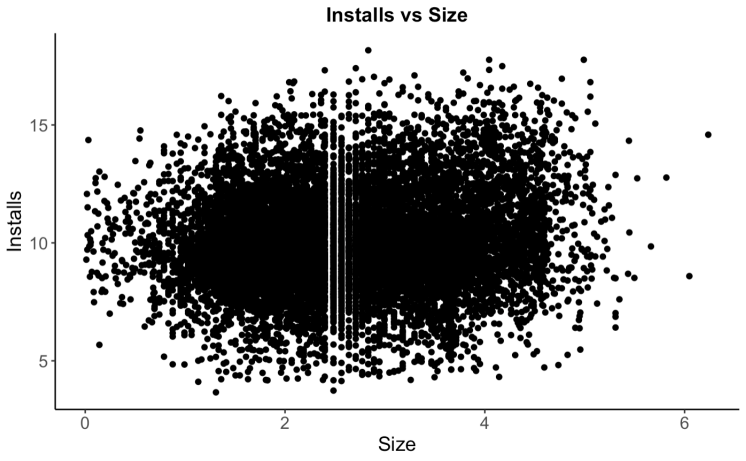


Figure 27: Installs vs Size

5.2.1.3 Installs vs Name Length

As we stated before, we did some research and we found out that there are some guidelines an app developer has to follow to name an app. Therefore, what we wanted to find out was if there is some relationship between the number of installs and the length of the app name.
But from Figure 28 we can see that the length of the app name does not affect the number of installs.

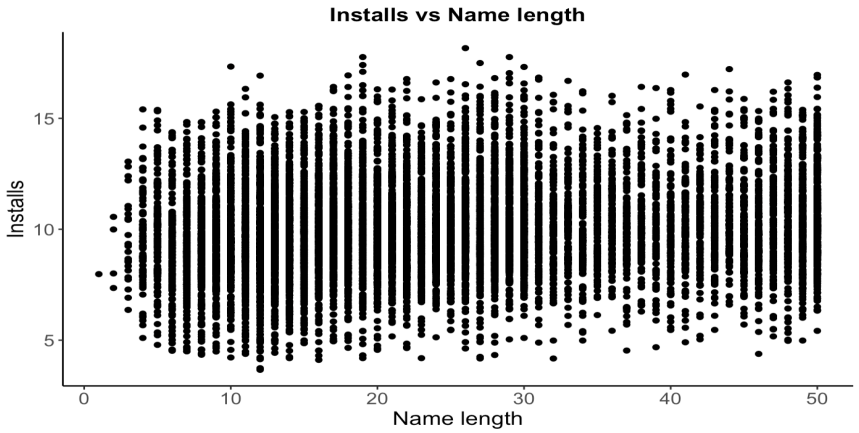


Figure 28: Installs vs Name length

5.2.1.4 Rating vs Category

We also were interested to know how the apps were rated according to its category. From Figure 29 we can establish that the distribution of apps rating for the different categories is quite similar.

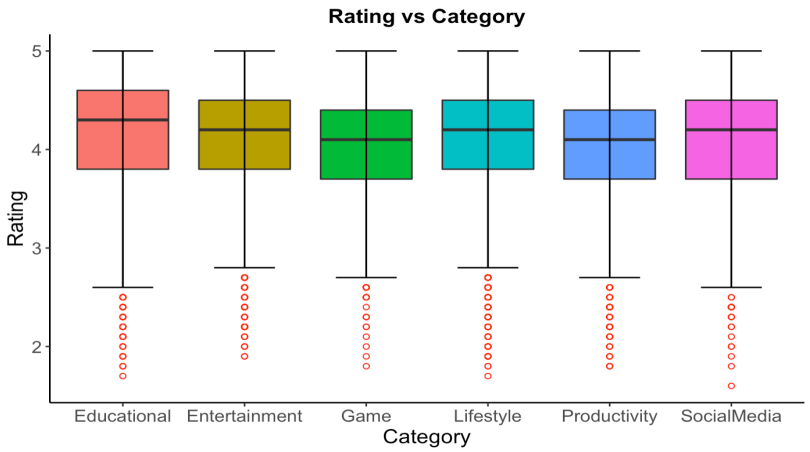


Figure 29: Rating vs Category

5.2.1.5 Size vs Category

We also tried to study if there is any relationship between the size of the apps according to their category Figure 30. The plot shows that apps that belong to the category Game are the ones with greater size. Which makes a lot of sense since game apps need more graphics among other things. While the apps with lower size are Productivity.

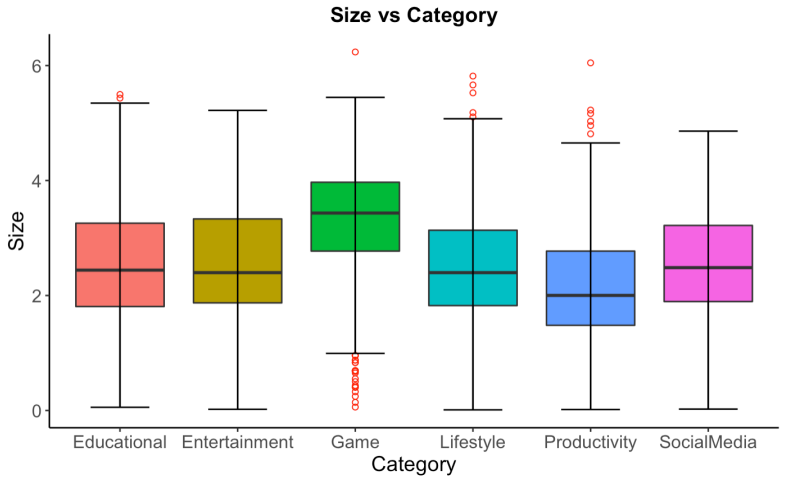


Figure 30: Size vs Category

6. PCA Analysis for numerical variables

PCA is a technique used in numerical data to reduce the dimensionality of the data by transforming it into a new coordinate system where most of the variation of the data can be described with fewer dimensions.

The aim of this section is to perform Principal Component Analysis (PCA) to summarize information in a dataset with multiple inter-correlated quantitative variables. Furthermore, to better visualize variation in a dataset with fewer variables than the original dataset.

6.1 Scree plot

Before starting PCA, the numerical variables need to be separated from the rest of the variables. Once this step is completed, the PCA process can be done.

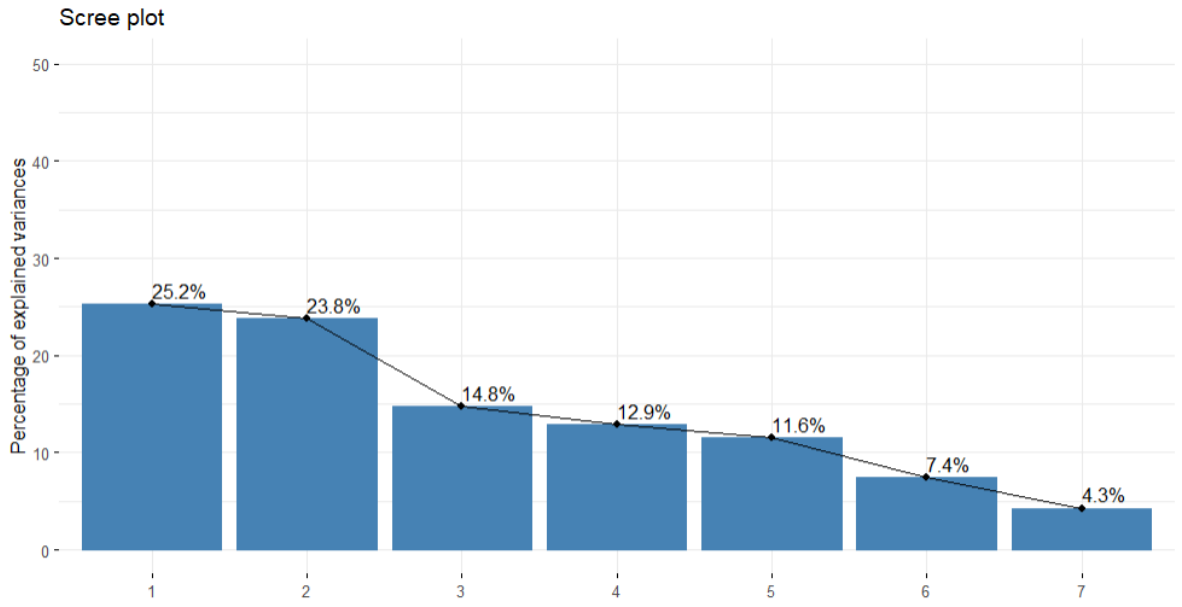


Figure 31: Scree plot

The proportion of variation explained by each dimension is given in the above Figure 31. As we can see, 25,2% of the variation is explained by the first dimension and 23,8% by the second one. As for the number of principal components to retain after PCA, in our analysis we only take the first three components that after adding up the successive proportions of variation, satisfied with 63.91% of the total variance, nearly two-thirds of the information in our dataset. From the plot above, we can see that after the point of dimension 3, all the remaining eigen-values decrease more slowly.

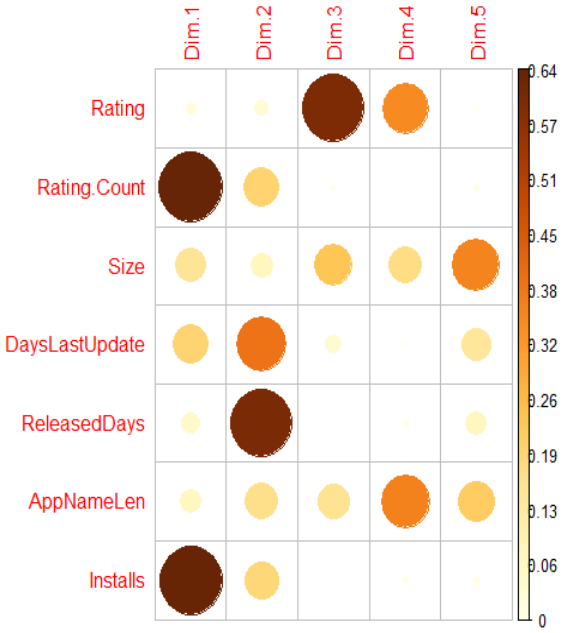


Figure 32: Correlation map

From Figure 32, it is clear to see that for both dimensions 3 and 4, we have the same list of most correlated variables. Due to this behavior, we can conclude that dimension 4 is just a reflection of dimension 3. We can also see that the variables `Rating.Count`, `DaysLastUpdate` and `Installs` are correlated with both PC1 and PC2, thus they are the most important in explaining the variability in the dataset.

From the figure 32, we can find the following highly correlated variables for each component:

- **Dim1:** `Rating.Count` and `Installs` → The meaning of this principal component can be interpreted as “Popularity”.
- **Dim2:** `DaysLastUpdate`, `ReleasedDays`→ Its meaning can be interpreted as “Active time && Update frequency”
- **Dim3:** `Rating`, `AppNameLen` and `Size` → Its meaning can be interpreted as “Rating && characteristics”

6.2 Factorial map visualization

6.2.1 Individuals projections

- Quality of representation of individuals on the factor map

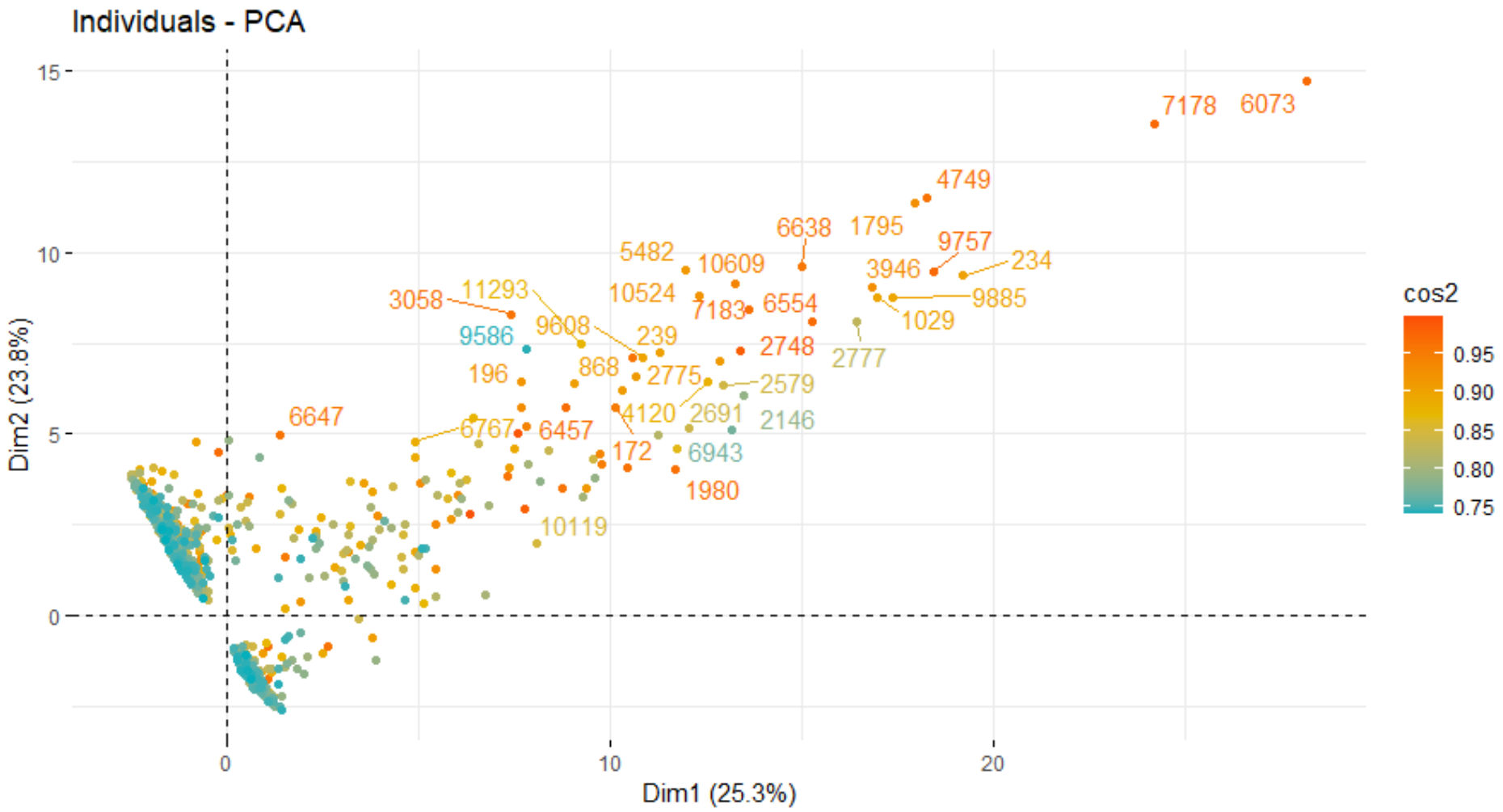


Figure 33: Quality of representation of the top 1000 individuals Dim1-2

From the Figure 33 of top 1000 individuals with better representation on the factor map we can clearly find three groups of individuals:

- Group 1: Apps with higher number of installs, votes, frequency of update and longer existing days, they are the ones with better quality of representation.
- Group 2: Apps with a poor or even zero number of installs and votes, but still exist in the market in which some release of updates were made recently.
- Group 3: Recently released or updated apps that haven't achieved much number of installs and votes

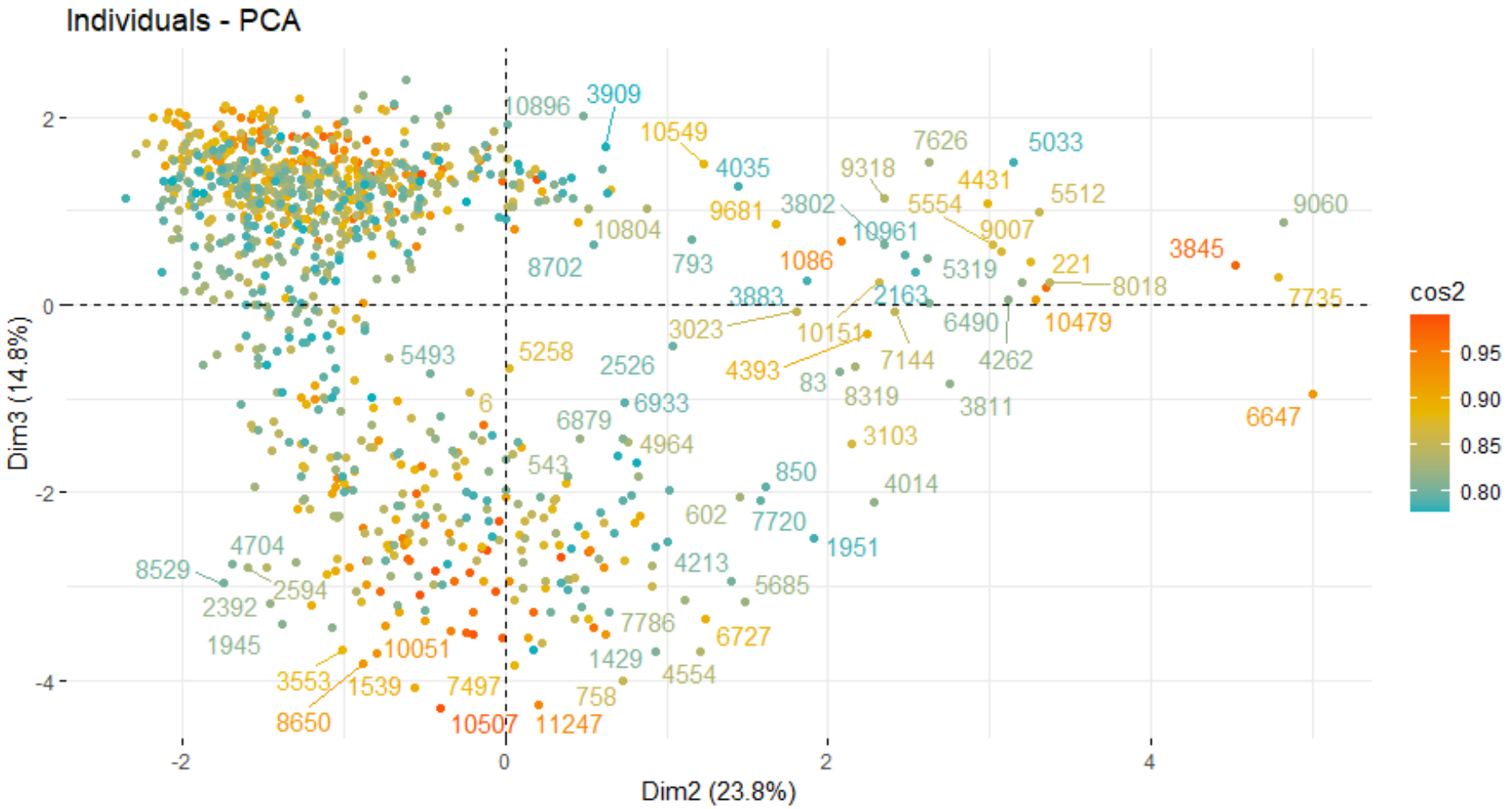


Figure 34: Quality of representation of top 1000 individuals Dim2-3

From Figure 34 we can see that `Rating` is not much correlated with number of existing days or frequency of update. Having a high `Rating` can't lead us to conclude that it is because of the longer existing days or frequency of update of an app. There are other factors that have more influence on the `Rating`.

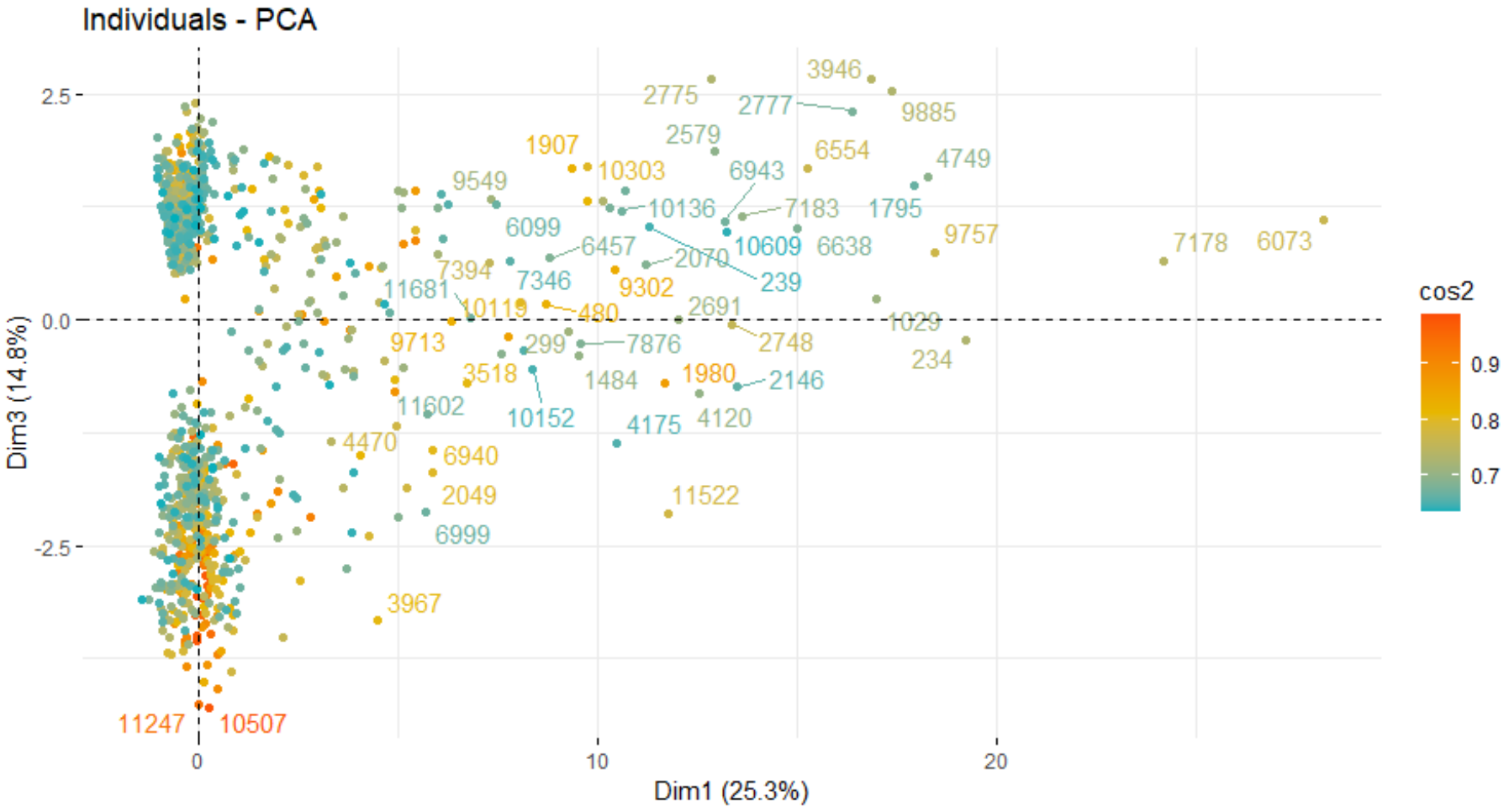


Figure 35: Quality of representation of top 1000 individuals Dim1-3

From figure 35, we can classify the individuals into the following groups:

- Group 1: Apps that have a higher number of installs and votes, in general, also have a good rating.
- Group 2: Apps with a poor number of installs and votes, but have an acceptable rating.
- Group 3: Apps with a poor number of installs and votes that receive bad ratings.
- Group 4: Apps with an acceptable number of installs and votes, but still their rating is bad.

- **Contribution of individuals to the first two principal components**

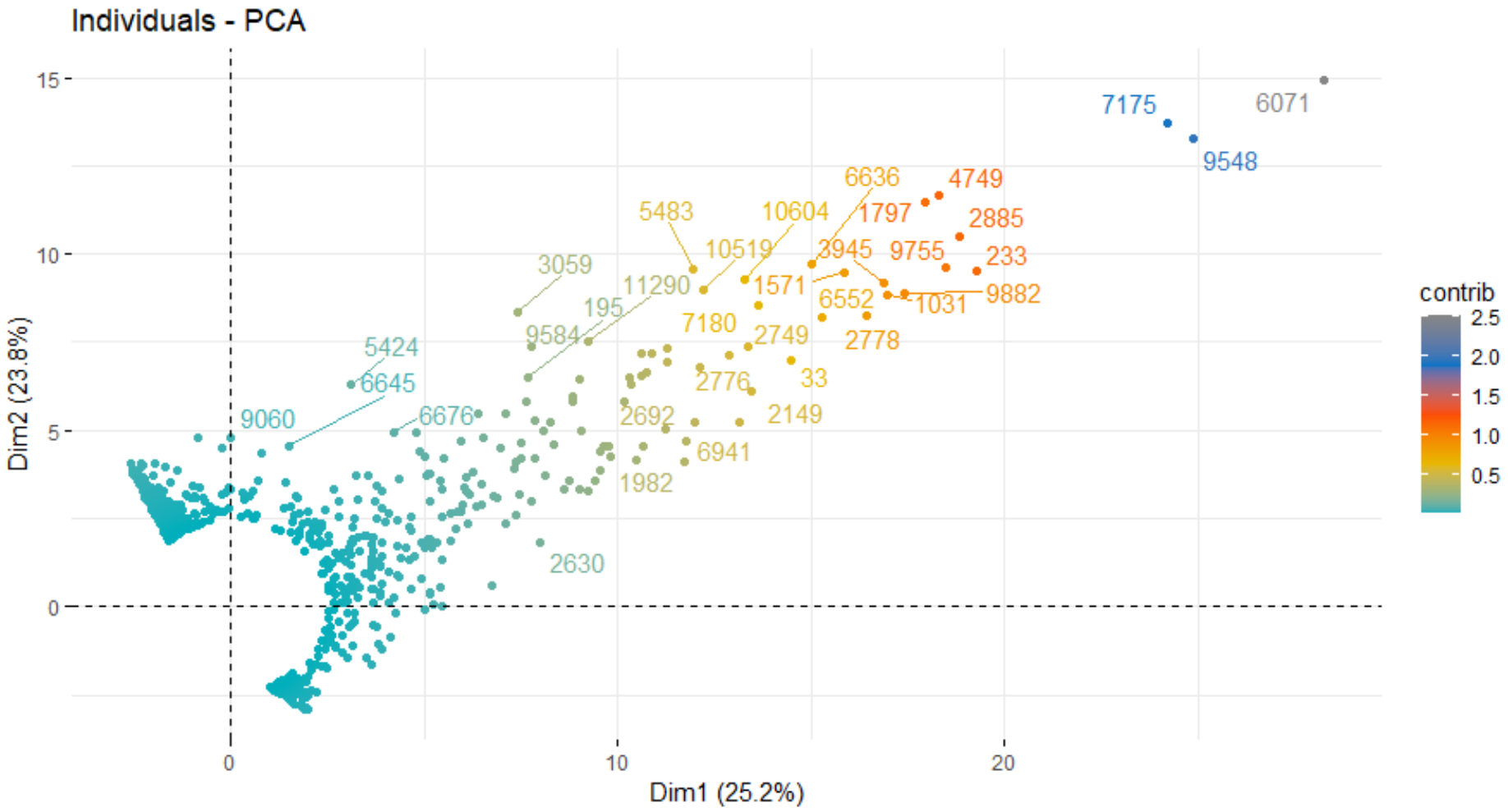


Figure 36: Contribution of individuals to Dim1-2

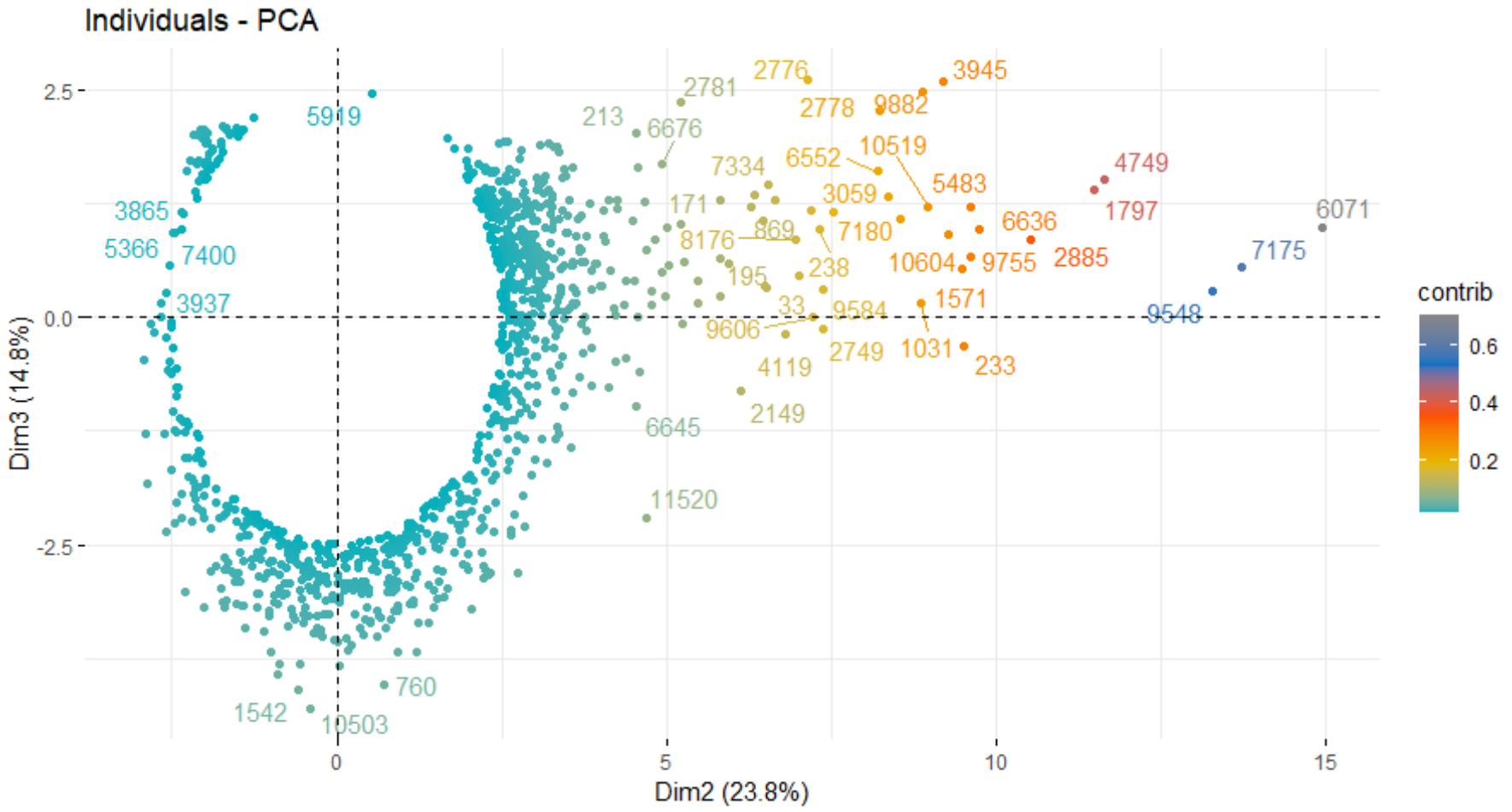


Figure 37: Contribution of individuals to Dim2-3

From figures 36 and 37 about the top most contributed 1000 individuals, we can see the same groups of individuals as before.

6.2.2 Common projection of numerical variables and modalities of qualitative variable

- Contribution of variables to PCs

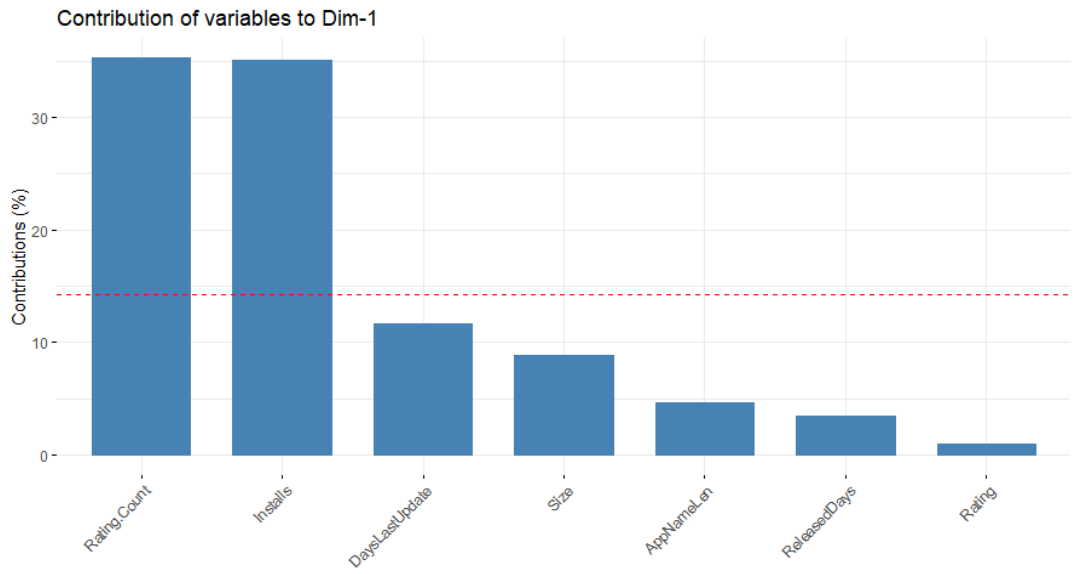


Figure 38: Contribution of variables to Dim1

From the above Figure 38 we can see that for dimension 1, the most contributed variables are `Rating.Count` and `Installs`.

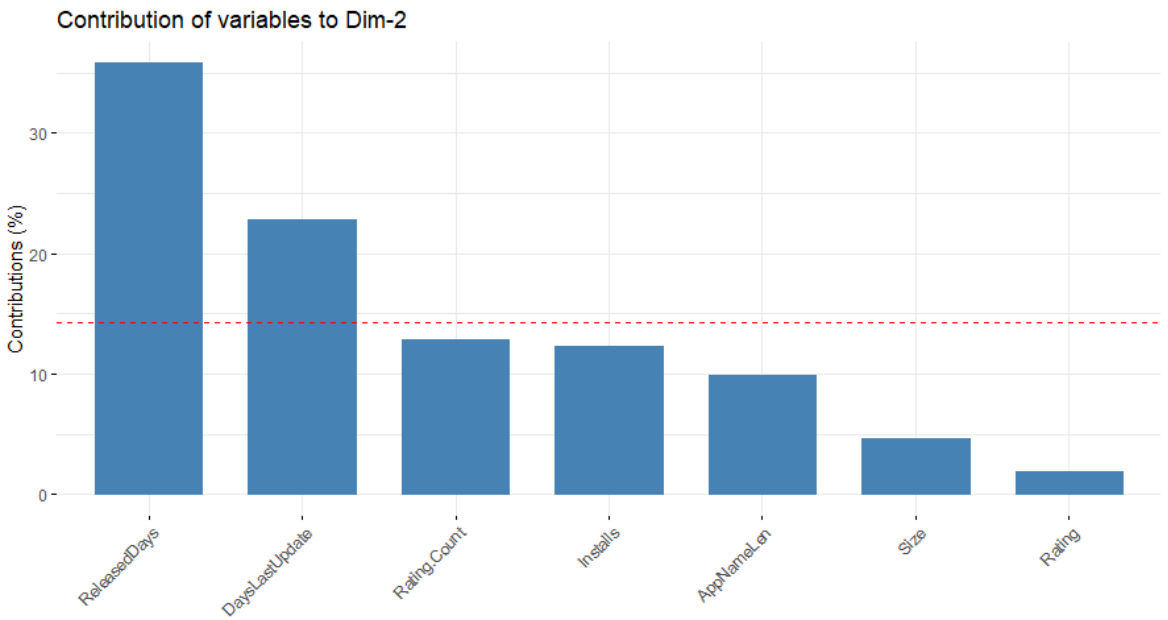


Figure 39: Contribution of variables to Dim 2

From the above Figure 39 we can see that `Released Days` and `DaysLastUpdate`, are the most contributed variables for dimension 2.

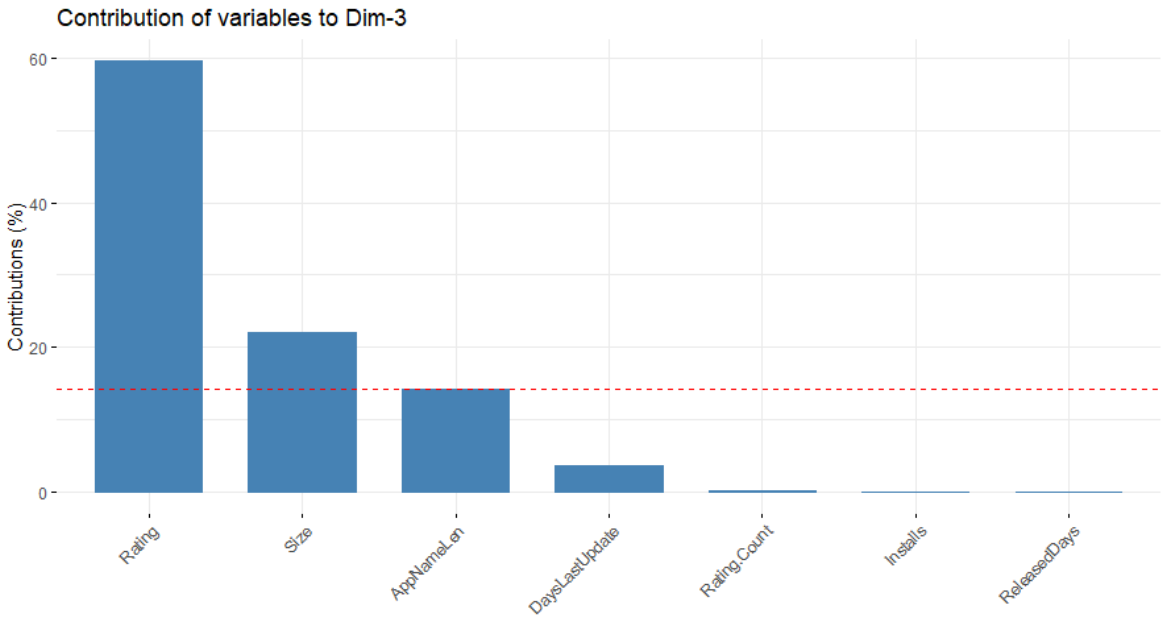


Figure 40: Contribution of variables to Dim 3

From the above Figure 40 we can see that `Rating`, `Size` and `AppNameLen` are the most contributed variables for dimensions 3.

- Biplots of individuals and variables

In the plot below (Figure 41) we use the variable `Category` as a supplementary qualitative variable and it is used for coloring individuals by groups.

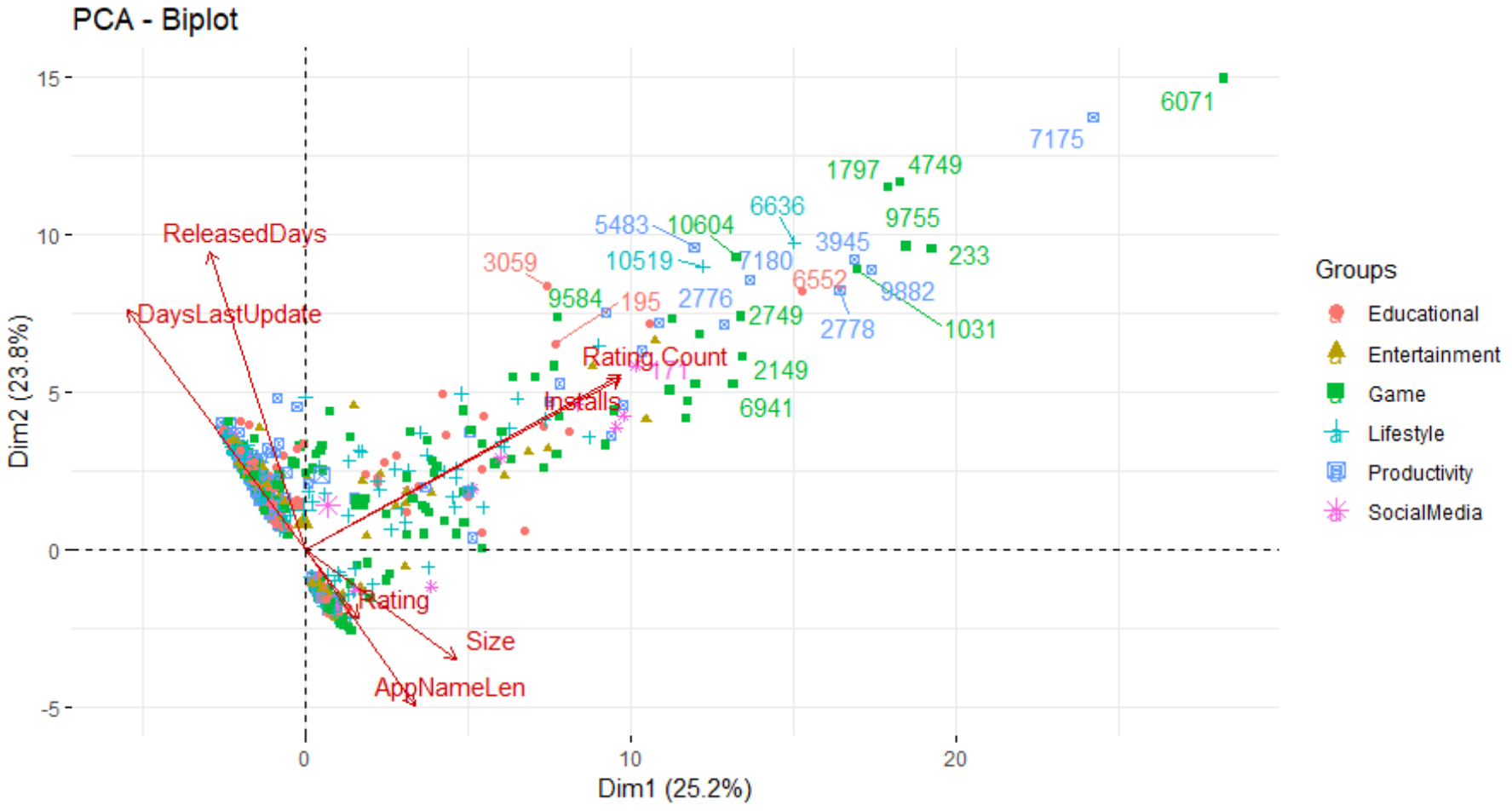


Figure 41: Biplot of individuals and variables Dim1-2

From Figure 41 we can observe that on the one hand, apps of categories `Game` and `Lifestyle` are always the ones with higher number of installs and votes, thus apps of these categories tend to be active longer and have a higher frequency of updates.

On the other hand, educational category tend to be less popular than other categories.

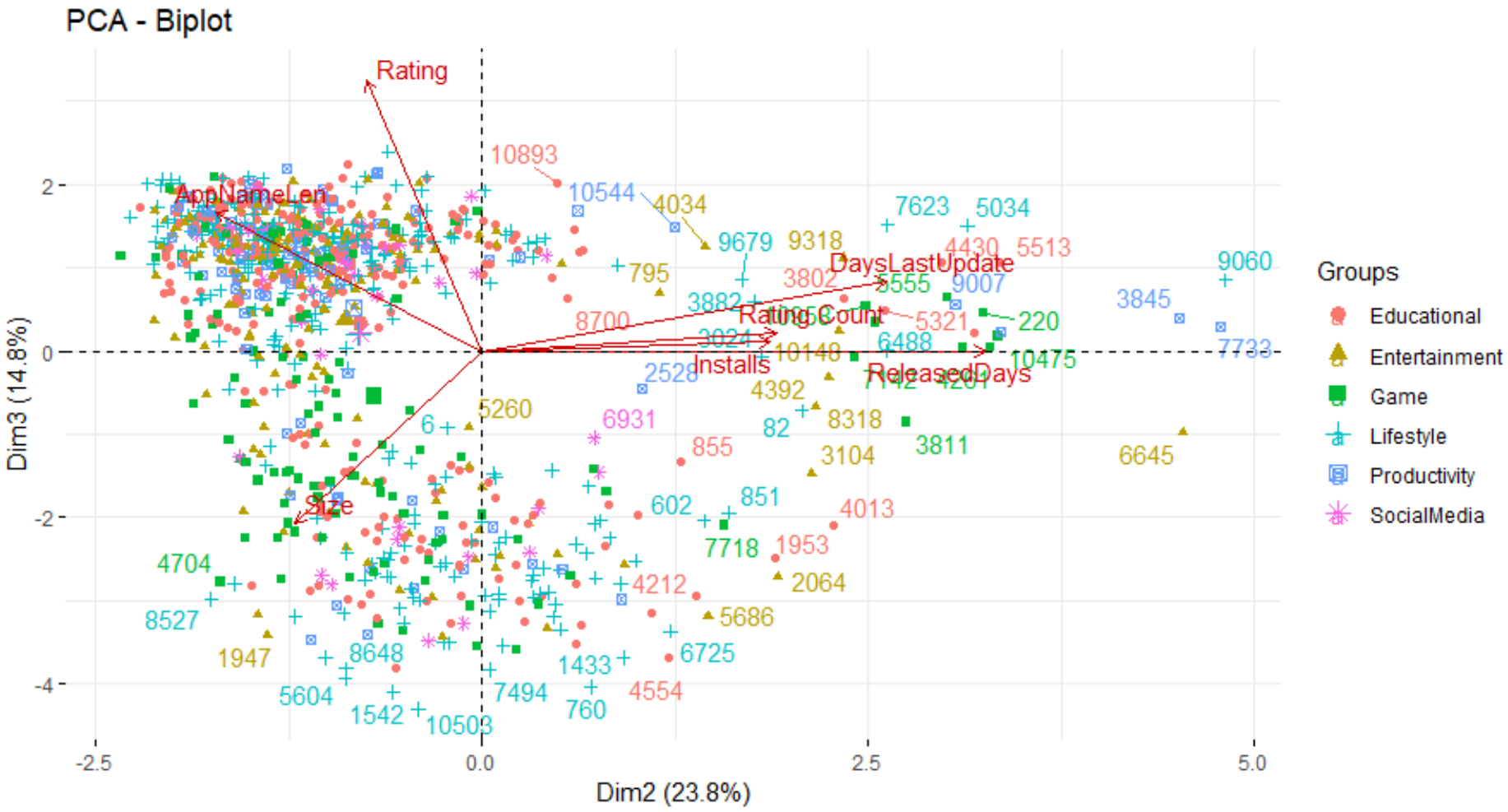


Figure 42: Biplot of individuals and variables Dim 2-3

From the Figure 42 we get the following interpretation:
For apps of categories `Game` and `Lifestyle` which are recently released to the market or have a low frequency of update, the probability of getting a not good or poor rating is relatively high.

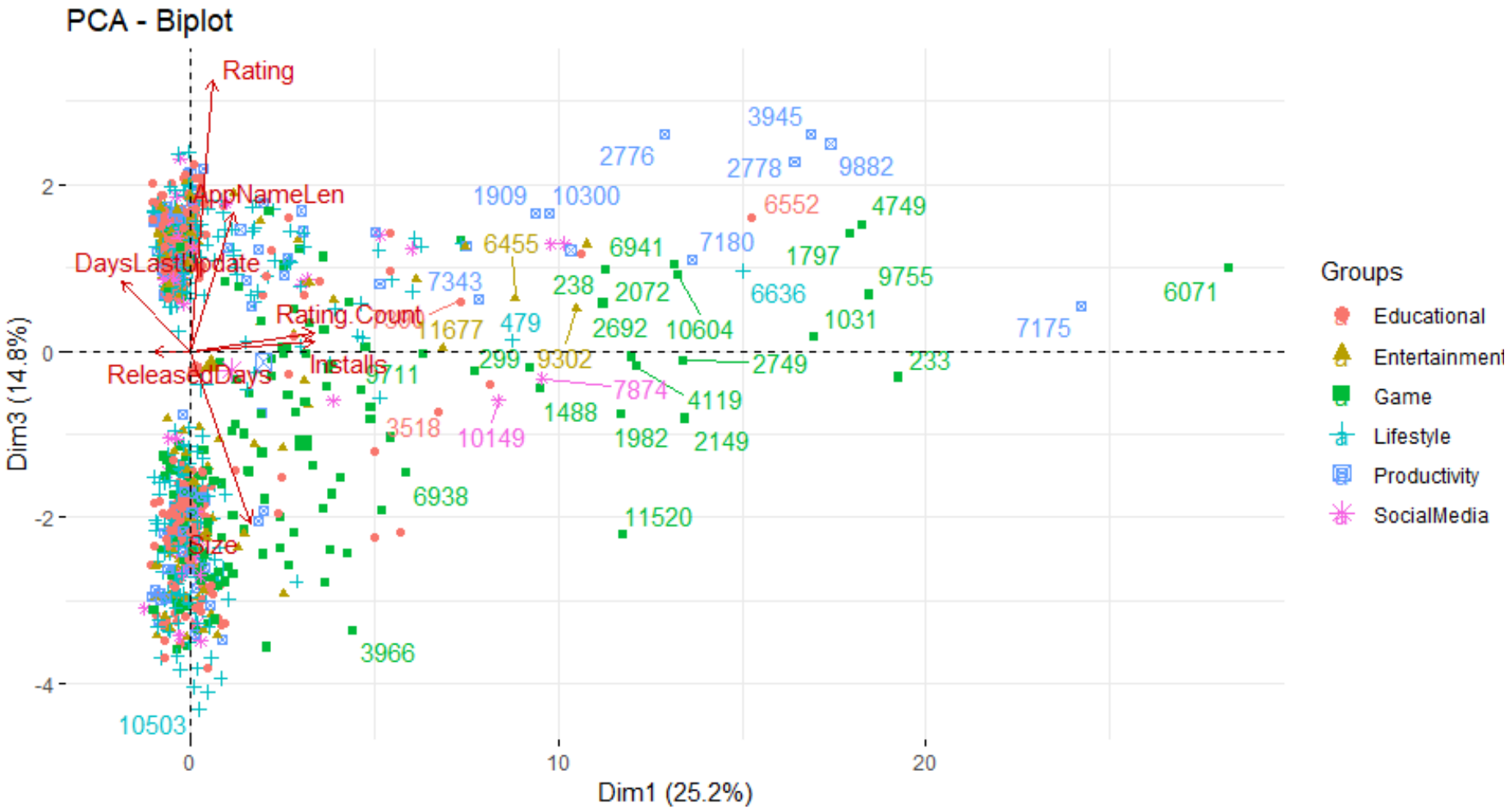


Figure 43: Biplot of individuals and variables Dim1-3

From the above Figure 43 we can also notice that although apps of category `Game` can always get a higher number of installs, it's not correlated with the rating. There are cases in which apps have higher popularity but poor ratings. What is noticeable is that unpopular `Game` apps tend to receive more bad ratings than good ratings.

6.2.3 Interpretation of relationships among variables observed

In order to understand the meaning of our principal components and to be able to assign a name to identify each component, it is necessary to identify those variables whose correlations are the highest with components.

The following plots can help us to better understand the relationship between all variables.

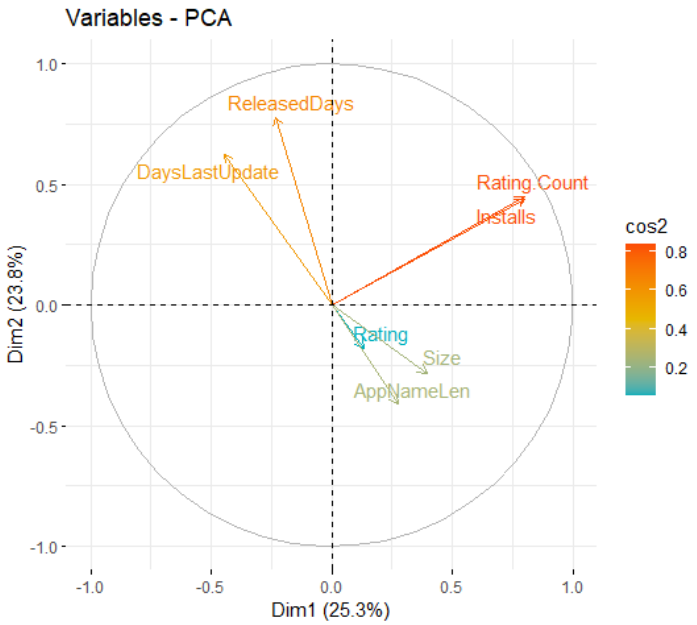


Figure 44: variable correlation Dim1-2

From the above variable correlation plot we observe the following behaviors:

- `Rating.Count` and `Installs` are positively correlated, because they are strongly grouped together. As the distance between the variables and the origin measures the quality of the variables on the factor map, and from the plot it's clear to see that they are far away from the origin, we can conclude that variables `Rating.Count` and Installs are well represented on the principal component.
- `ReleasedDays` and `DaysLastUpdate` are also positively correlated but not strongly grouped together. As `ReleasedDays` is closer to the circle of

correlation and the angle it forms with respect to the axis Dim2 is smaller, it has better representation on the factor map than `DaysLastUpdate`.

- `Rating`, `Size` and `AppNameLen`(Rating && characteristics) are negatively correlated with `ReleasedDays` and `DaysLastUpdate`(Active time && Update frequency), which means that "Rating && characteristics" has the opposite behavior than "Active time && Update frequency".

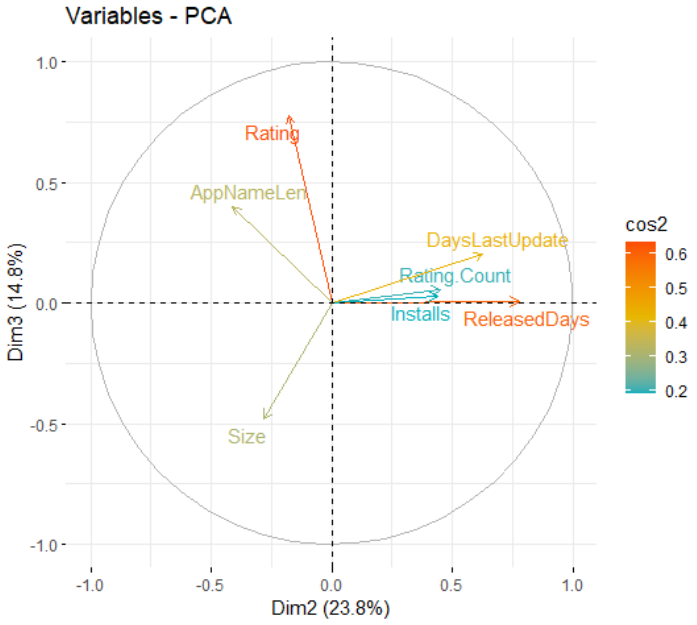


Figure 45: variable correlation Dim2-3

From the above variable correlation plot we observe the following behaviors:

- `Rating.Count` and `Installs` (Popularity) are positively correlated with `DaysLastUpdate` and `ReleasedDays` (Active time & update frequency) but as they are closer to the center of the circle, they are less important for the components 2-3, which means having a low quality of representation (cos2).
- `Rating` and `AppNameLen` are positively correlated, but `AppNameLen` contribution to the building of PC3 is less than `Rating` because it forms a bigger angle with respect to the axis dim3.

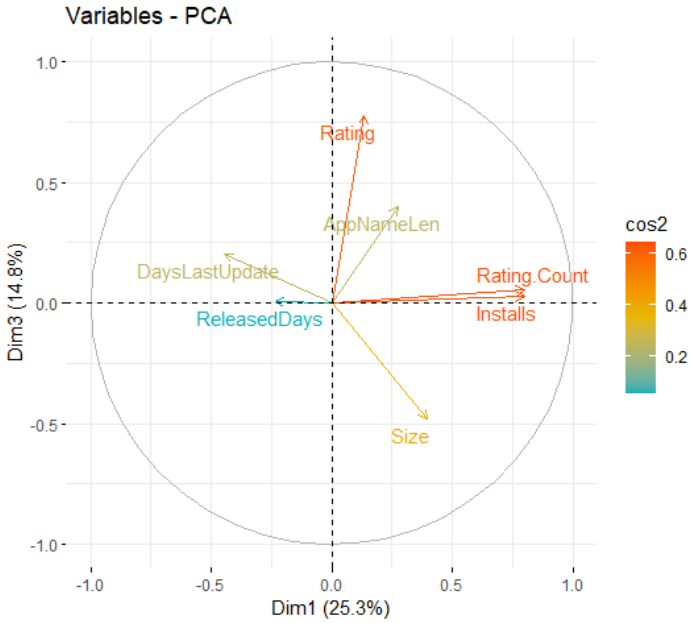


Figure 46: variable correlation Dim1-3

From the above variable correlation plot we observe the following behavior. `DaysLastUpdate` and `Size` are negatively correlated, this means that the more days without updating an app the smaller is its size, which makes sense because usually the larger is the size, the more bugs needed be fixed or new features to be included to the app, thus higher is the frequency of update.

As variables can be represented as points (coordinates) in components space by using their correlation with the components. We tried to apply the clustering algorithm Kmeans to classify variables into 3 groups. Thus, we get the same division of variables as before.

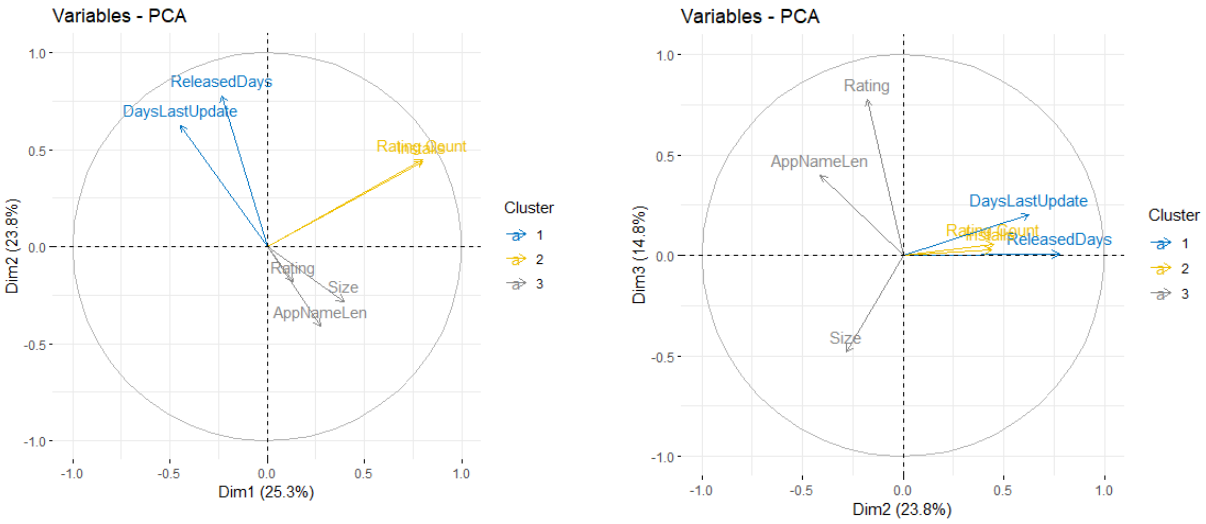


Figure 47: Correlation maps after applying Kmeans

6.3 Conclusions

After performing PCA and various types of analysis about variables and individuals behavior on factor map through plots, we get the following conclusions:

- Variables `Rating.Count`, `DaysLastUpdate` and `Installs` are correlated with both PC1 and PC2, therefore they are the most important in explaining the variability in the dataset
- The frequency of updates doesn't have any effect on the rating. We can consider in this case the possibility that not all apps information in this dataset was updated correctly or with the relevant information.
- There are apps with fewer votes but high ratings(on the figures of individuals we can observe this behavior), there seems to be a non-linear relationship between `Rating.Count` and `Rating`.
- `Game` and `Lifestyle` are the categories with the most number of installs.
- `Rating.Count` and Installs are highly correlated, the total variance would not be affected if we remove one of these variables when performing PCA.
- A higher number of installs or votes doesn't mean that the rating is also high.

7. MCA of multiple qualitative variables

MCA is a data analysis technique for categorical data, used to detect and represent underlying structures in a dataset. It does this by representing data as points in a low-dimensional Euclidean space.

7.1 Detection of low frequency variable categories

Before starting MCA, we need to identify variable categories with a very low frequency as these types of variables can distort the analysis. In Figure 48, we can see that the variables `Minimum.Android` and `Content.Rating` have low frequency categories.

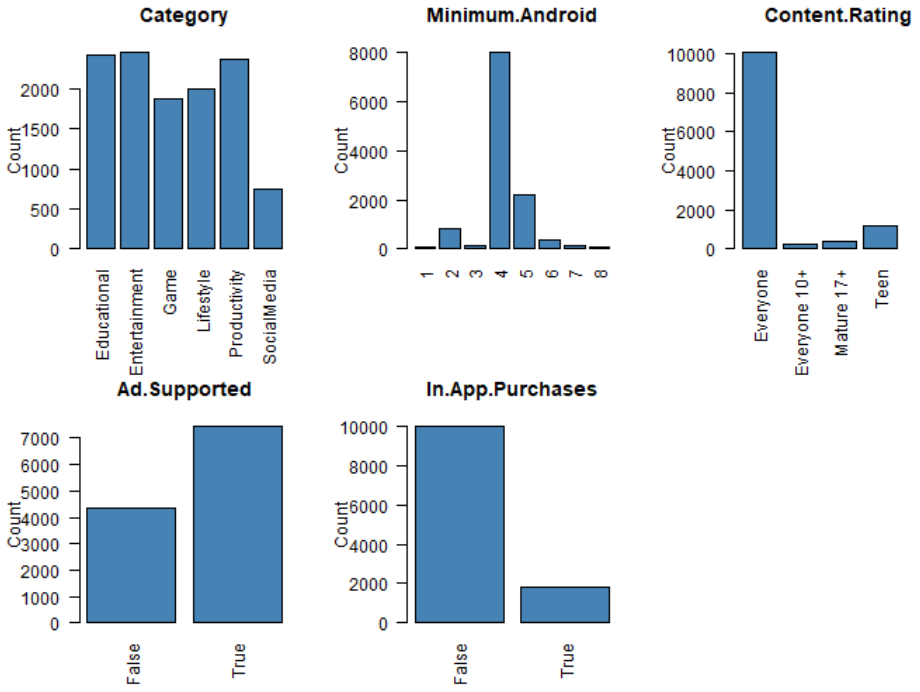


Figure 48: Distribution of the variable categories

For `Minimum.Android` and `Content.Rating` we aggregated the categories so that we have <4, 4, 5 and >5 and Everyone and AgeRestricted, respectively (Figure 49).

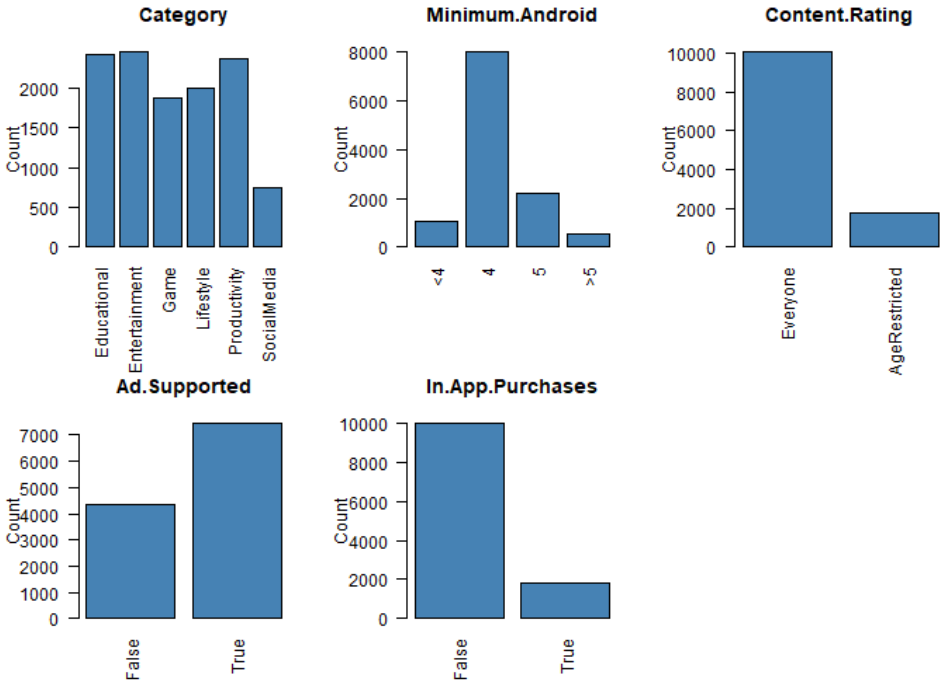


Figure 49: Distribution of the variable categories after aggregation of low frequency categories

7.2 Eigen values

After performing MCA using the **logical table**, we obtained the eigen values (Figure 50). We kept the dimensions that had an eigen value bigger than 1/p, where p is the number of categorical variables. In our case p=5, so 1/p = 0.2.
In the end, we decided to keep 5 dimensions.

	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	0.3226002	14.663646	14.66365
Dim.2	0.2530929	11.504222	26.16787
Dim.3	0.2278697	10.357712	36.52558
Dim.4	0.2084731	9.476049	46.00163
Dim.5	0.2022985	9.195386	55.19701
Dim.6	0.1976014	8.981880	64.17889
Dim.7	0.1962563	8.920743	73.09964
Dim.8	0.1750459	7.956633	81.05627
Dim.9	0.1521349	6.915224	87.97150
Dim.10	0.1380674	6.275793	94.24729
Dim.11	0.1265597	5.752712	100.00000

Figure 50: Eigen values of the dimensions

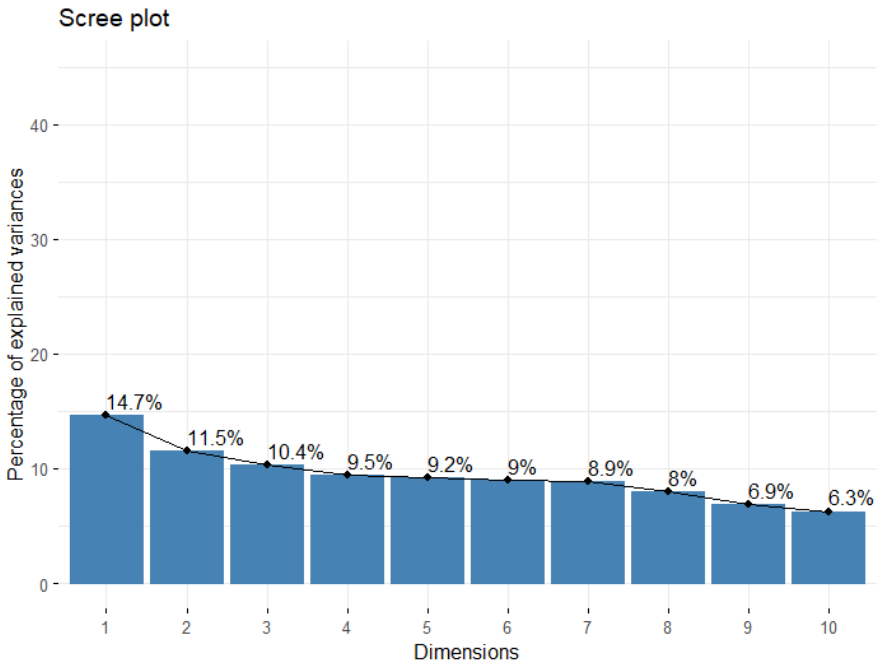


Figure 51: Scree plot of the dimensions

7.3 Biplots of individuals and variable categories

In this section, we can see the biplots of individuals and variable categories for each combination of dimensions.

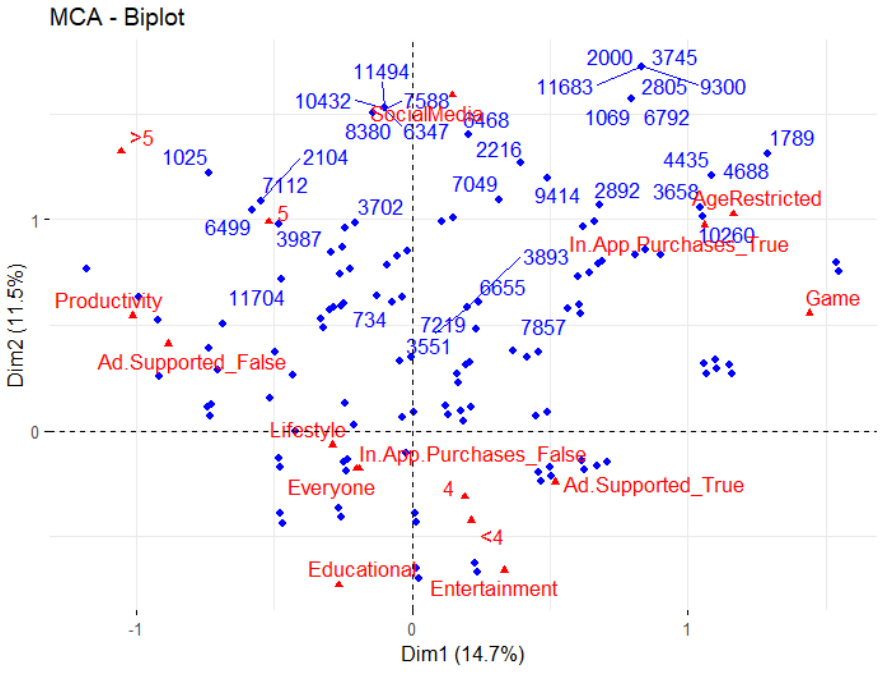


Figure 52: Biplot of individuals and variable categories in dimension 1-2

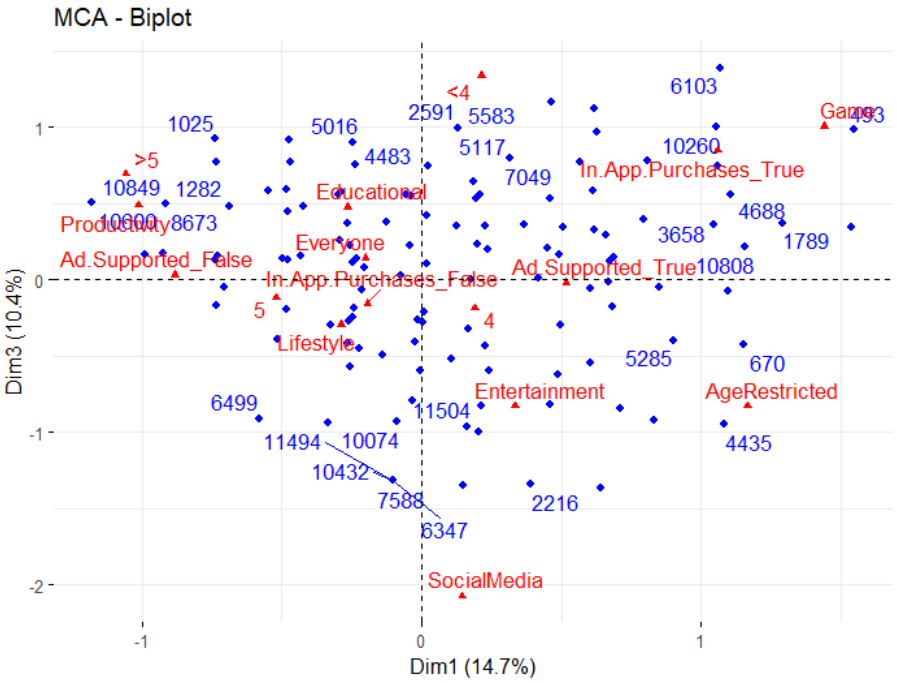


Figure 53: Biplot of individuals and variable categories in dimension 1-3

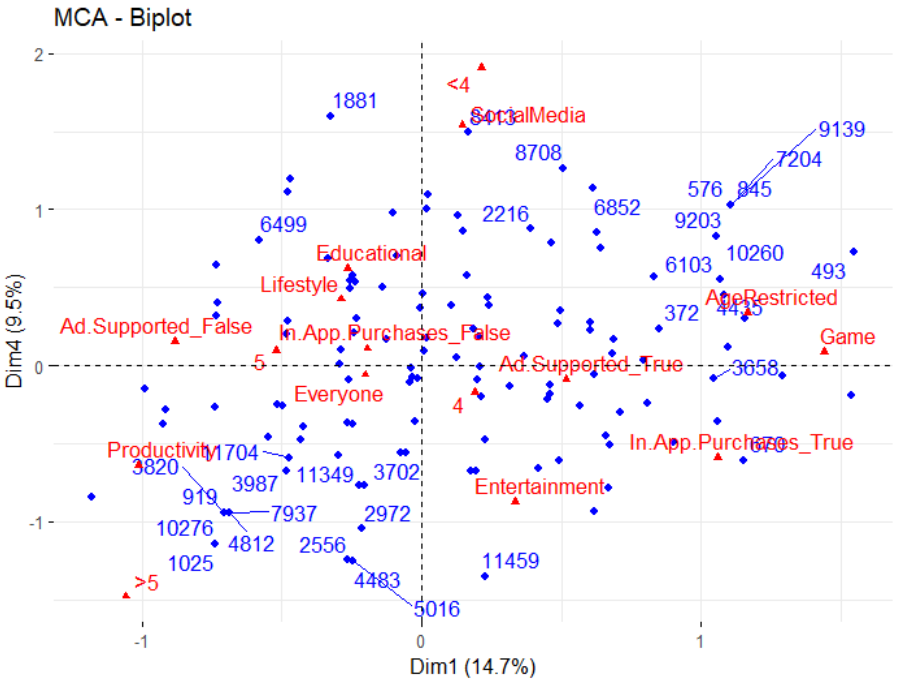


Figure 54: Biplot of individuals and variable categories in dimension 1-4

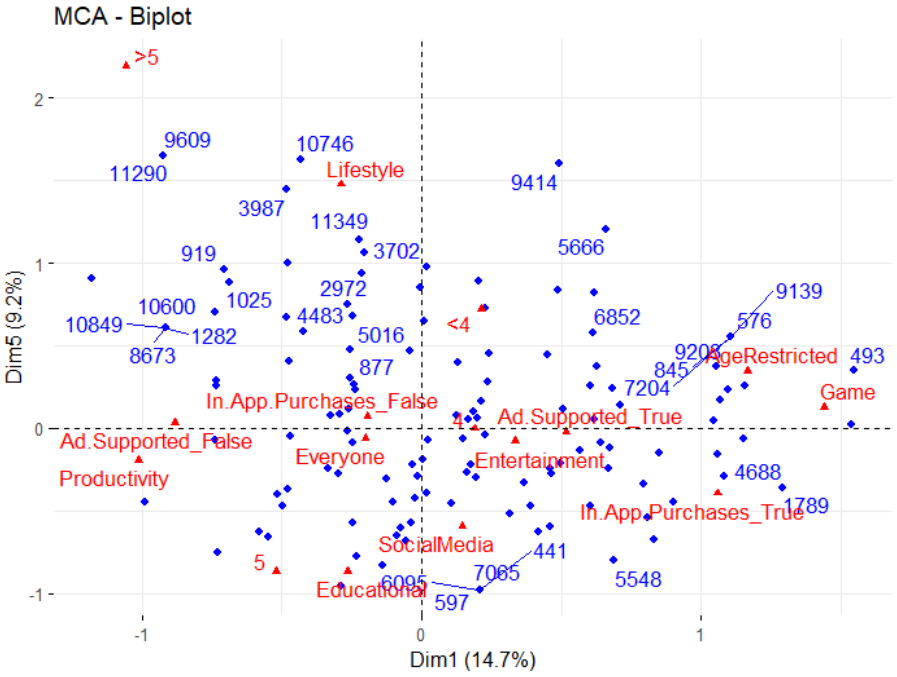


Figure 55: Biplot of individuals and variable categories in dimension 1-5

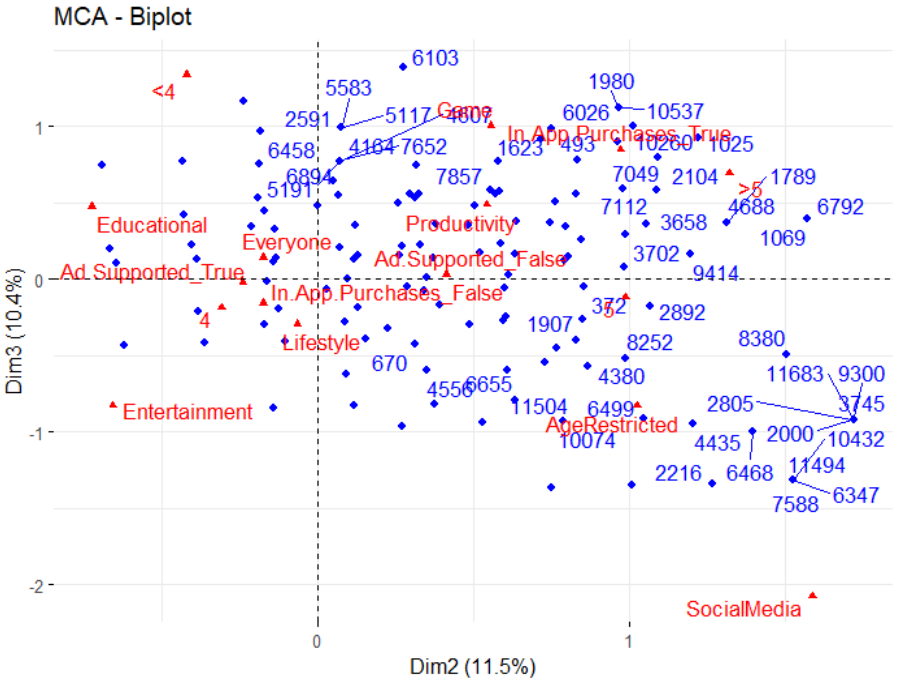


Figure 56: Biplot of individuals and variable categories in dimension 2-3

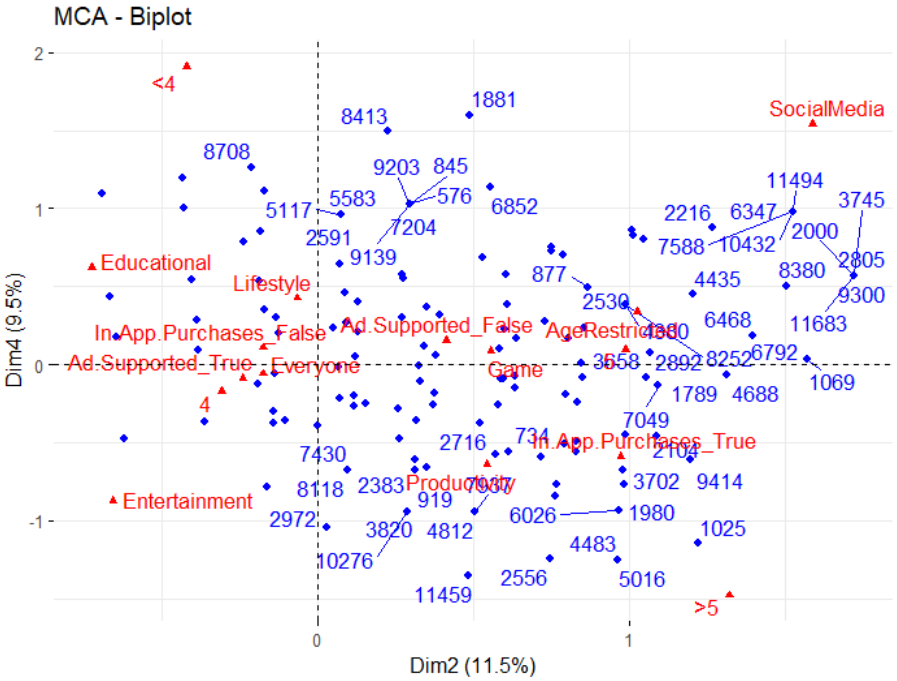


Figure 57: Biplot of individuals and variable categories in dimension 2-4

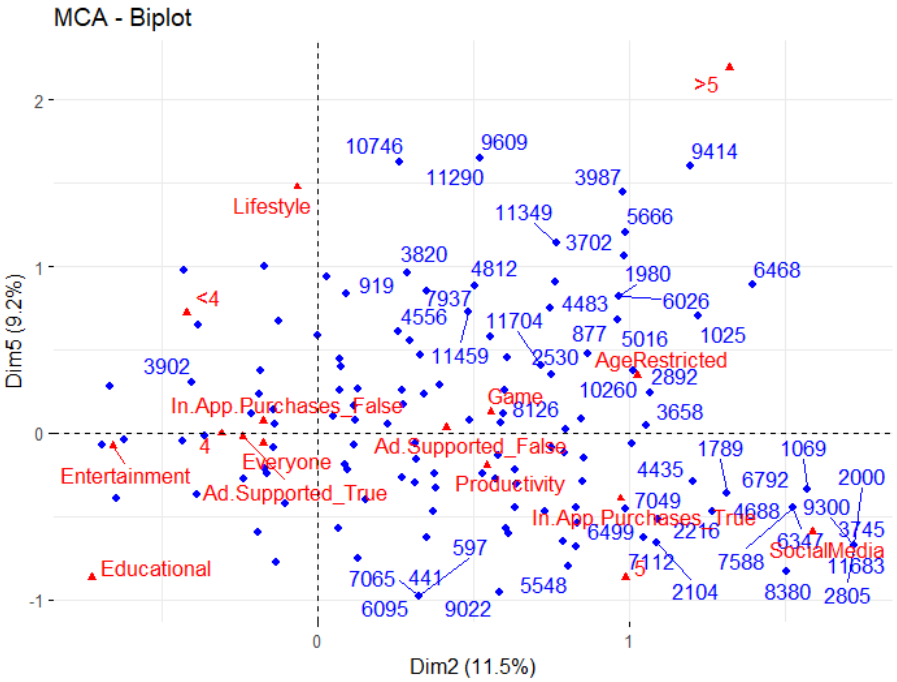


Figure 58: Biplot of individuals and variable categories in dimension 2-5

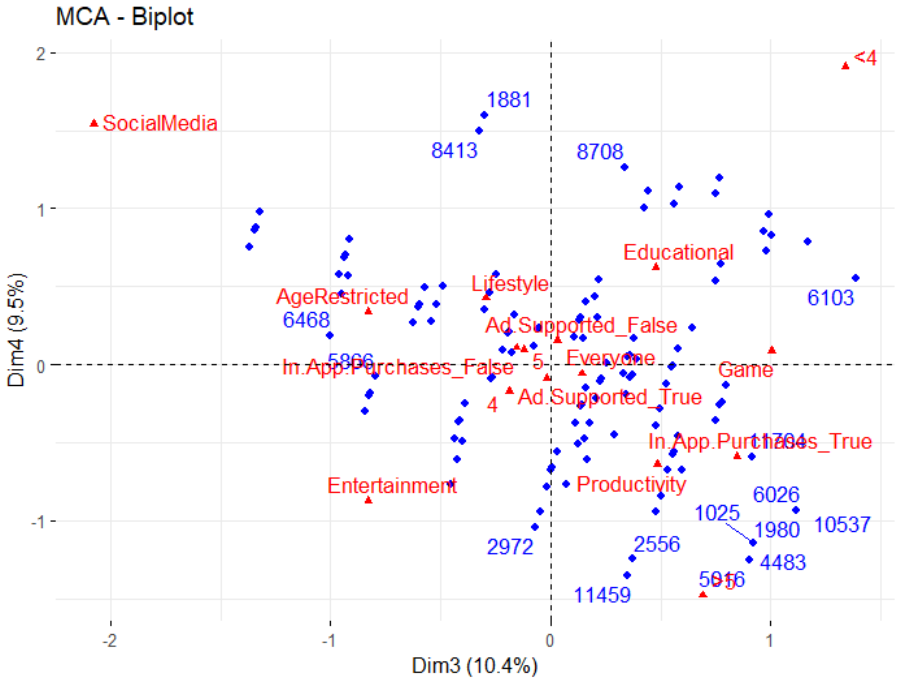


Figure 59: Biplot of individuals and variable categories in dimension 3-4

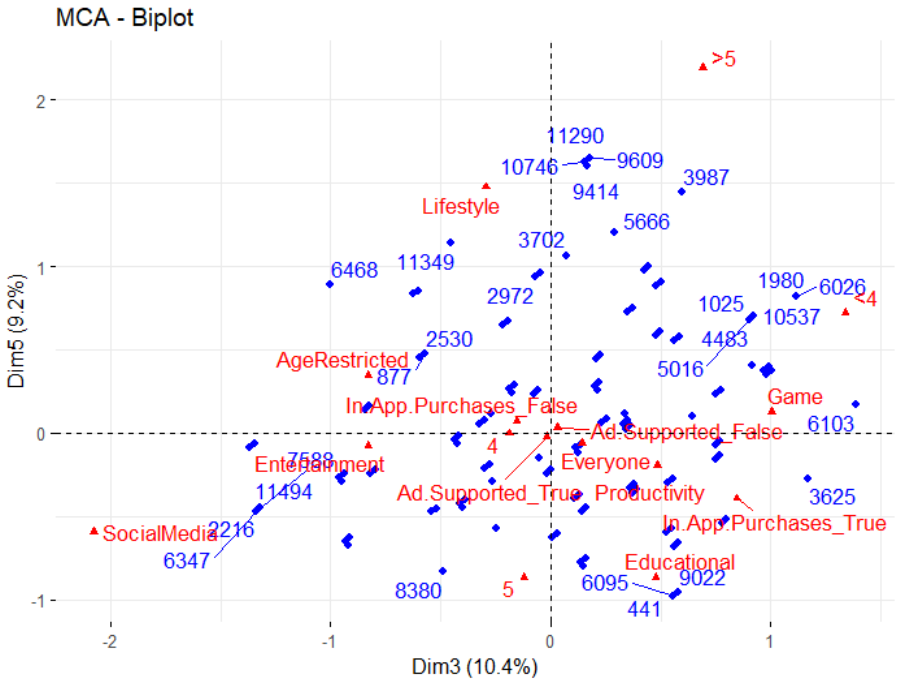


Figure 60: Biplot of individuals and variable categories in dimension 3-5

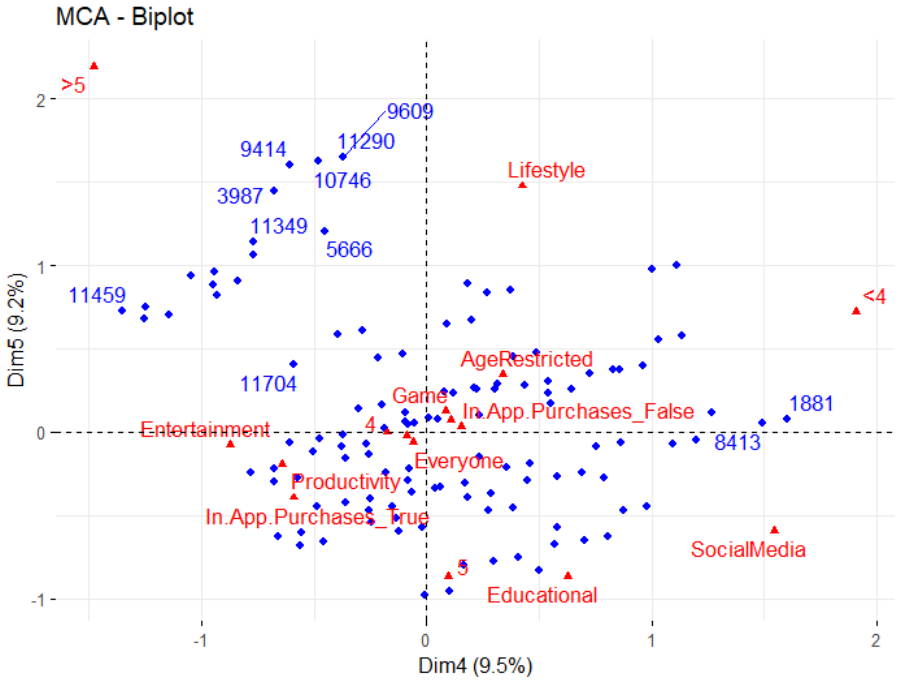


Figure 61: Biplot of individuals and variable categories in dimension 4-5

7.4 Correlation between variables and principal dimensions

In this section, we can see the correlation between variables and MCA principal dimensions for every combination of dimensions. The plots are quite similar because the selected dimensions have more or less the same inertia.

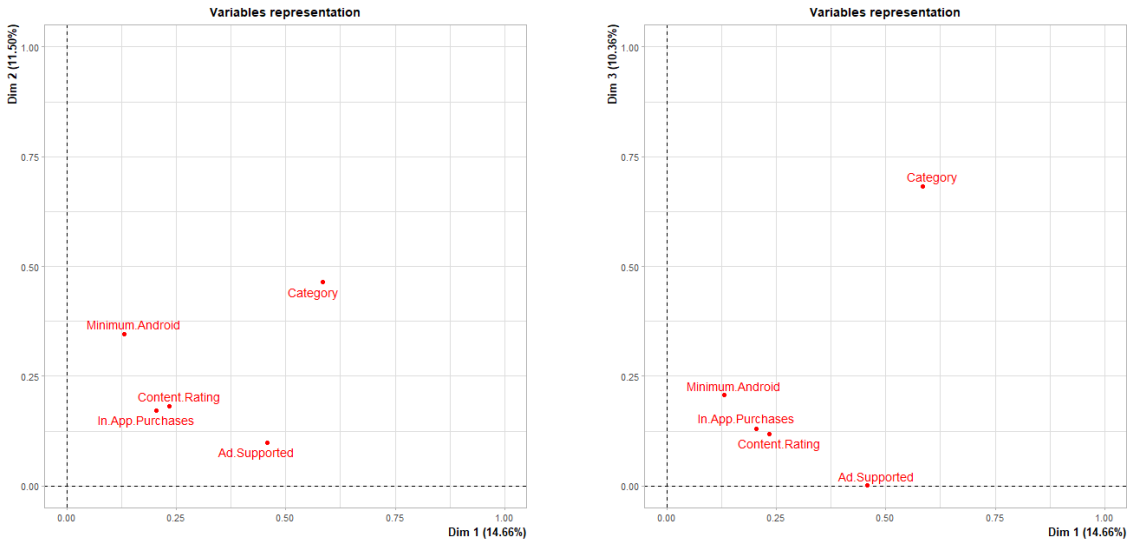


Figure 62: Correlation between variables and principal dimensions in dimension 1-2 and 1-3

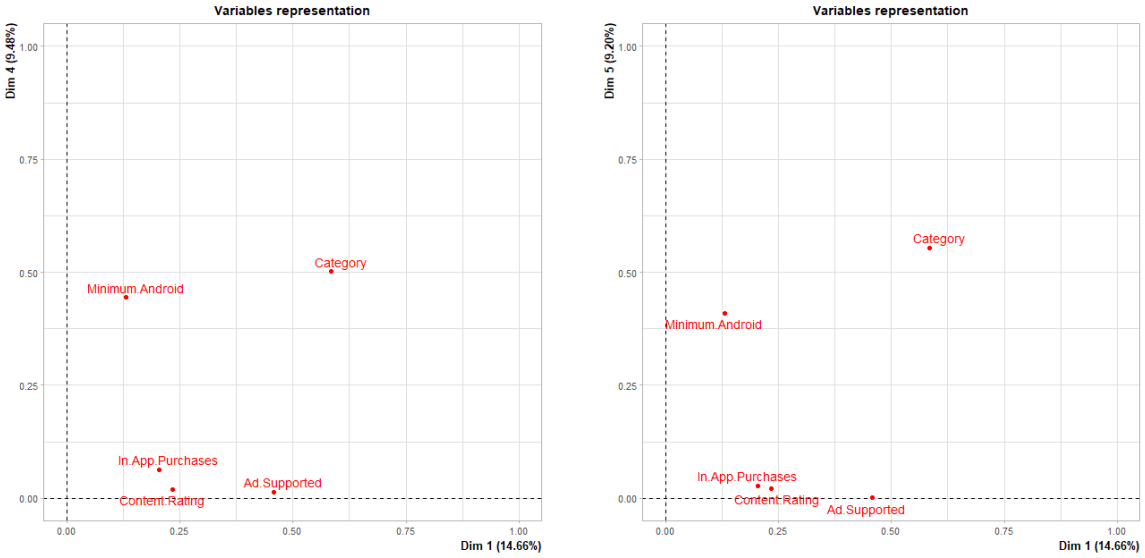


Figure 63: Correlation between variables and principal dimensions in dimension 1-4 and 1-5

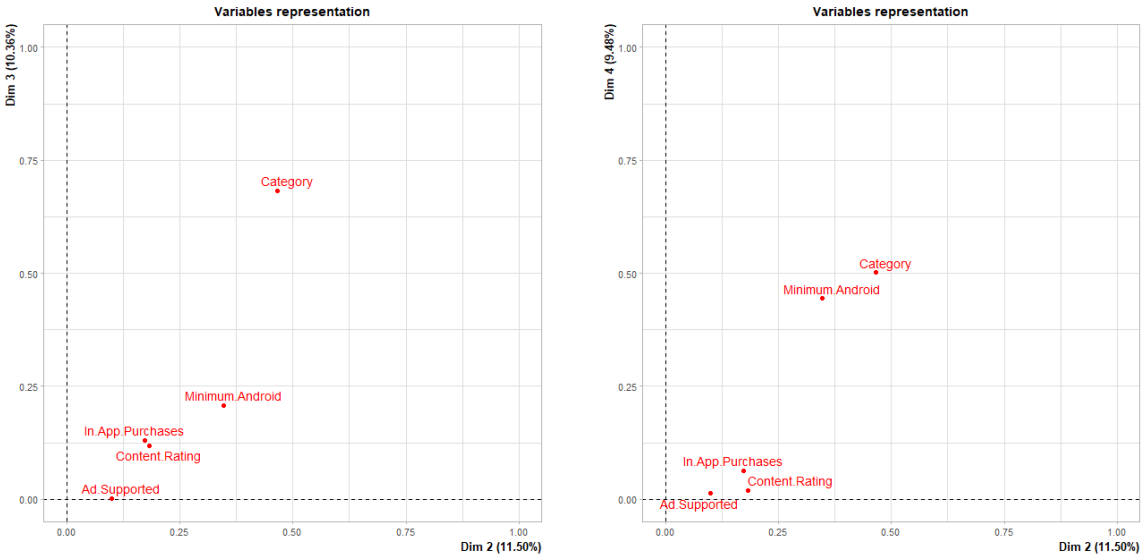


Figure 64: Correlation between variables and principal dimensions in dimension 2-3 and 2-4

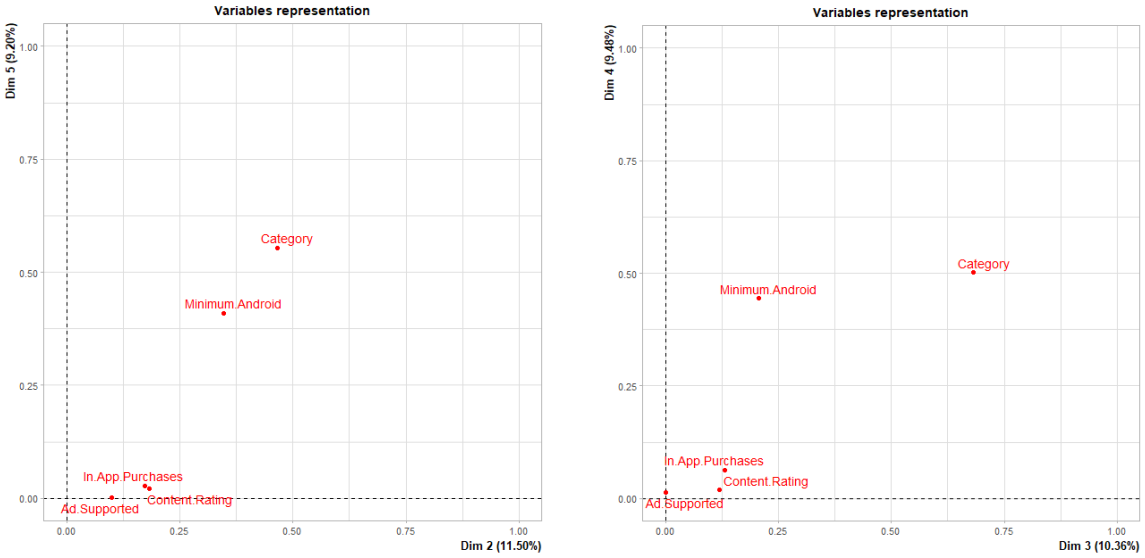


Figure 65: Correlation between variables and principal dimensions in dimension 2-5 and 3-4

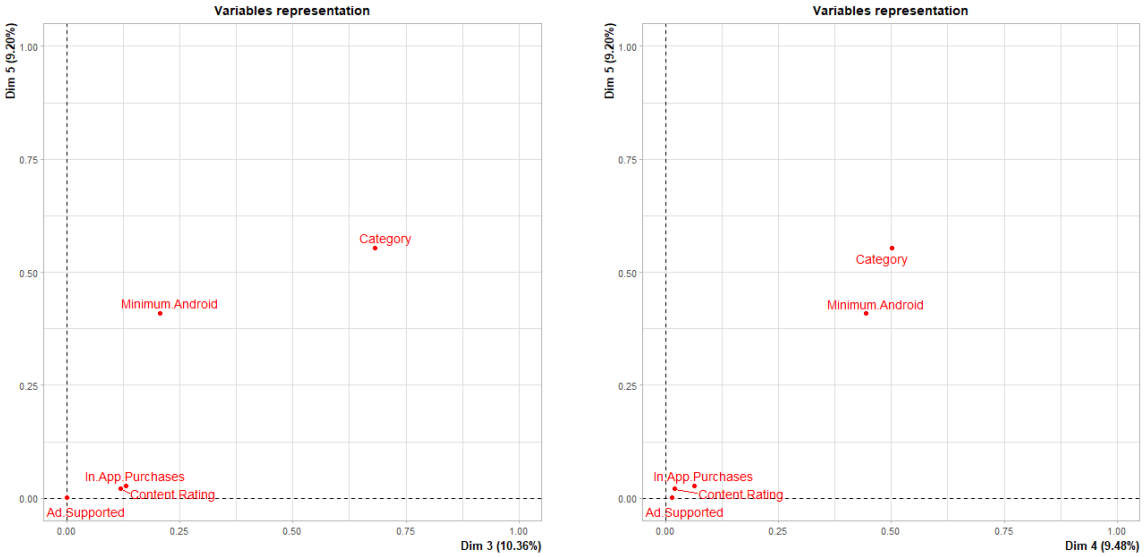


Figure 66: Correlation between variables and principal dimensions in dimension 3-5 and 4-5

7.5 Quality of representation of variable categories

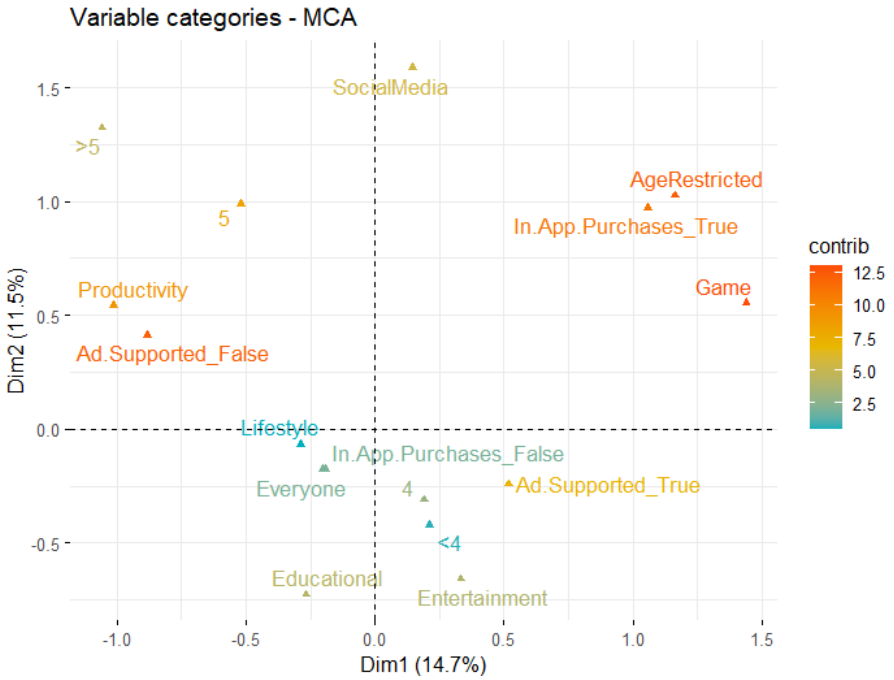


Figure 67: Quality of representation of variable categories in dimension 1-2

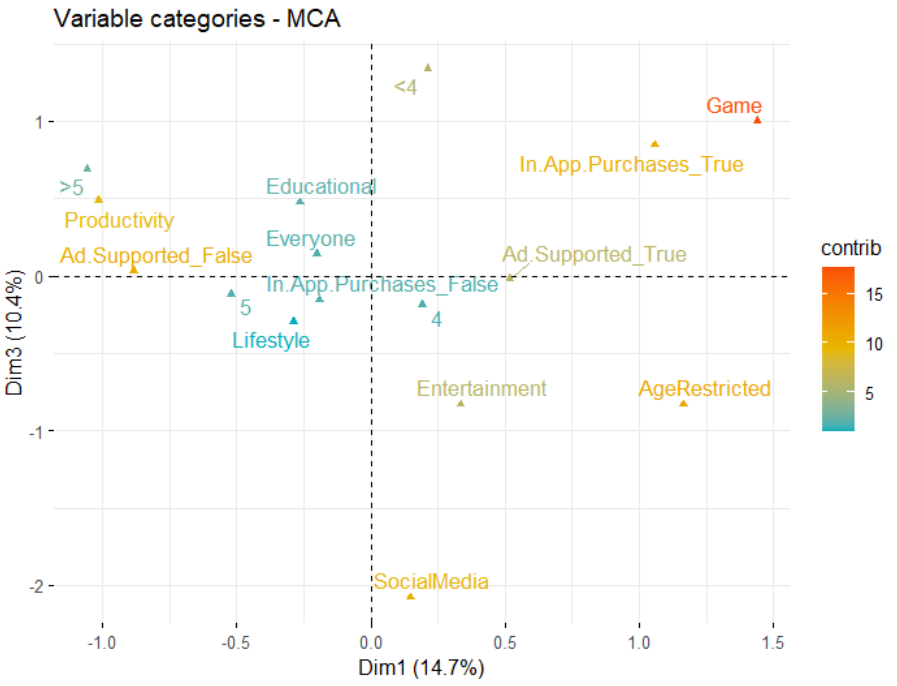


Figure 68: Quality of representation of variable categories in dimension 1-3

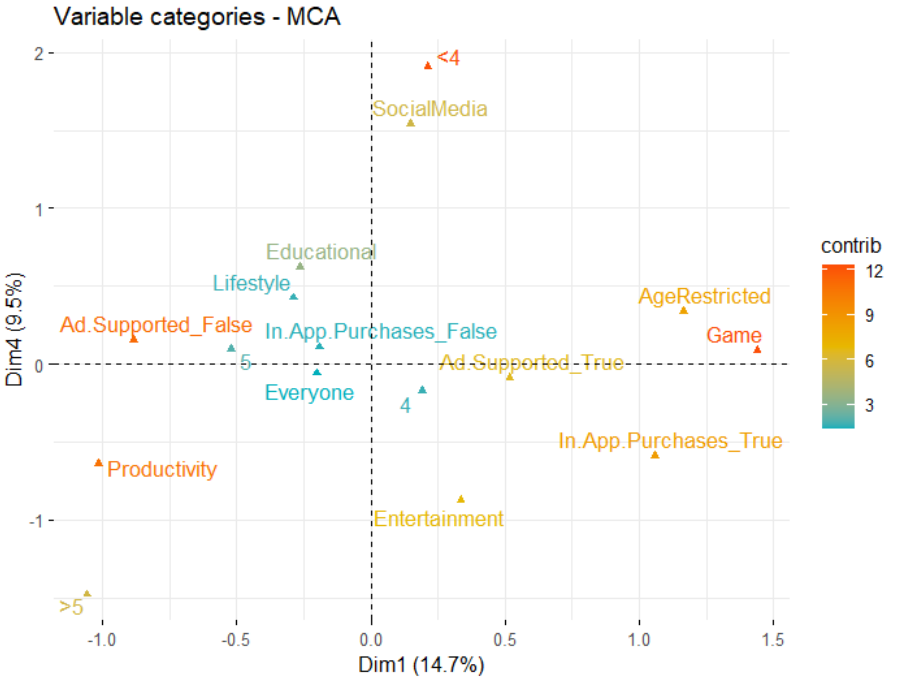


Figure 69: Quality of representation of variable categories in dimension 1-4

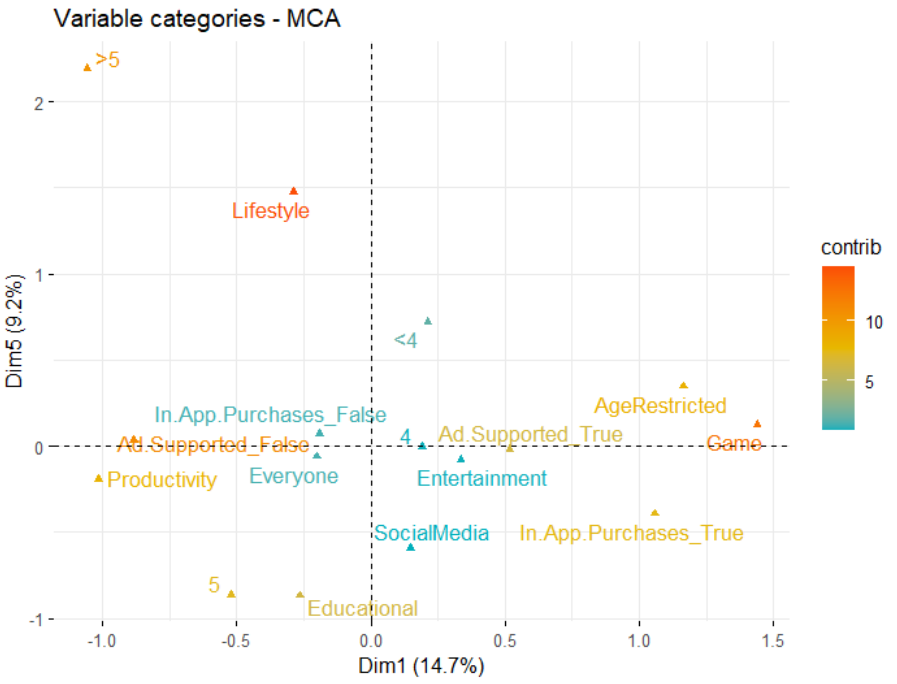


Figure 70: Quality of representation of variable categories in dimension 1-5

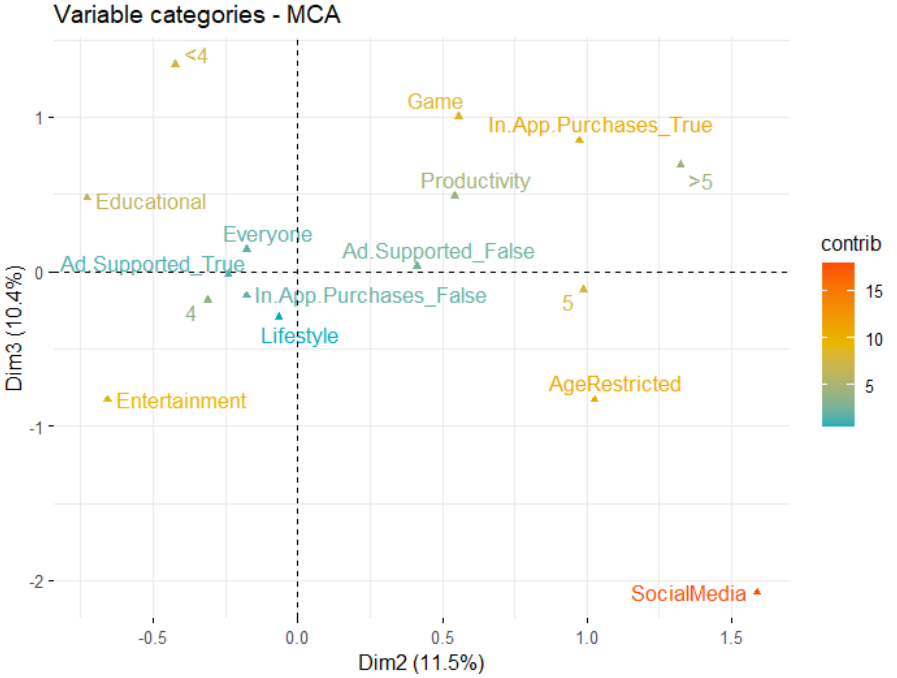


Figure 71: Quality of representation of variable categories in dimension 2-3

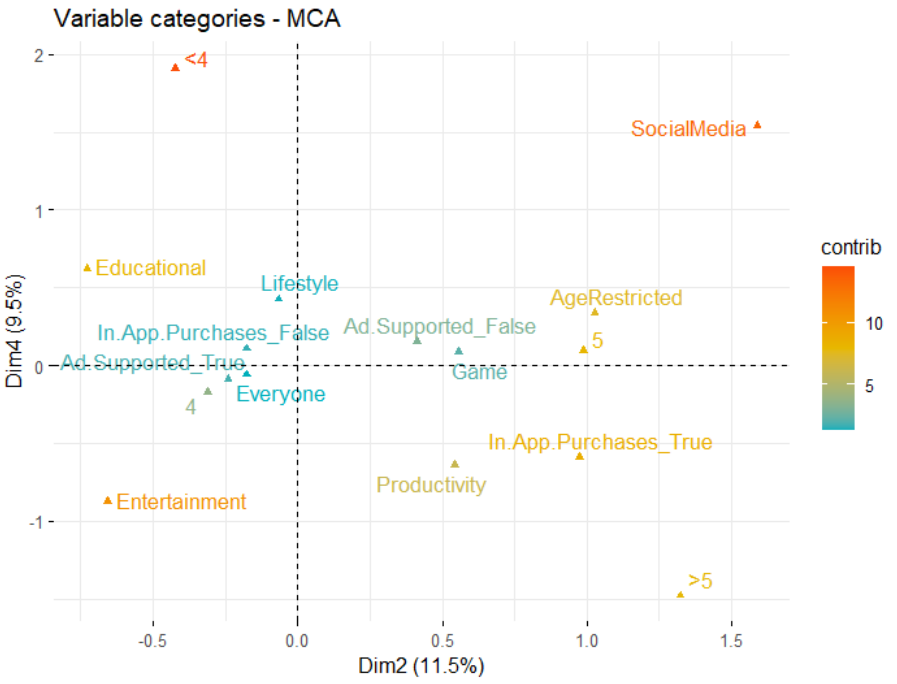


Figure 72: Quality of representation of variable categories in dimension 2-4

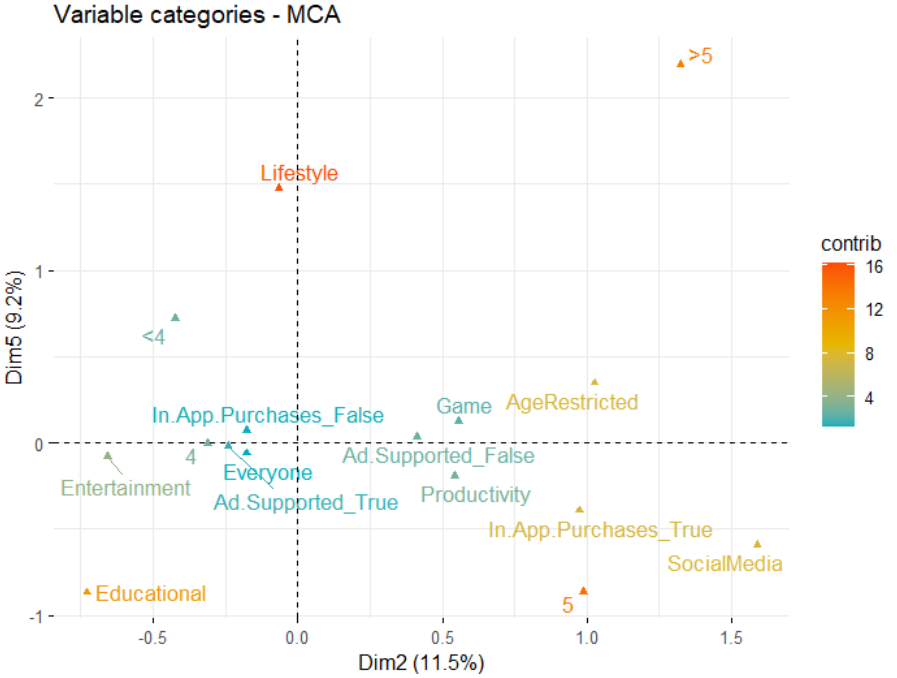


Figure 73: Quality of representation of variable categories in dimension 2-5

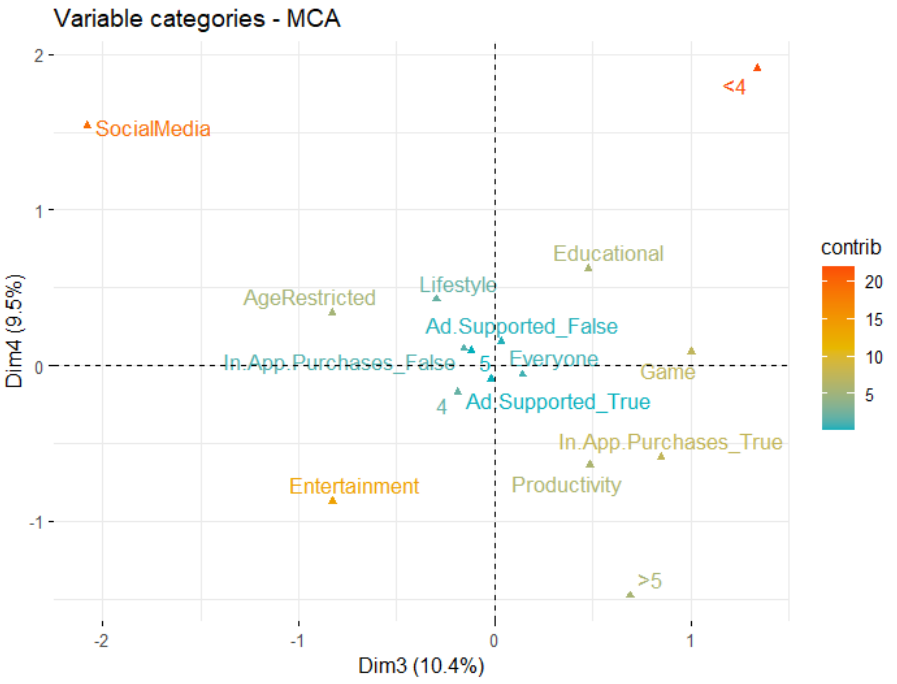


Figure 74: Quality of representation of variable categories in dimension 3-4

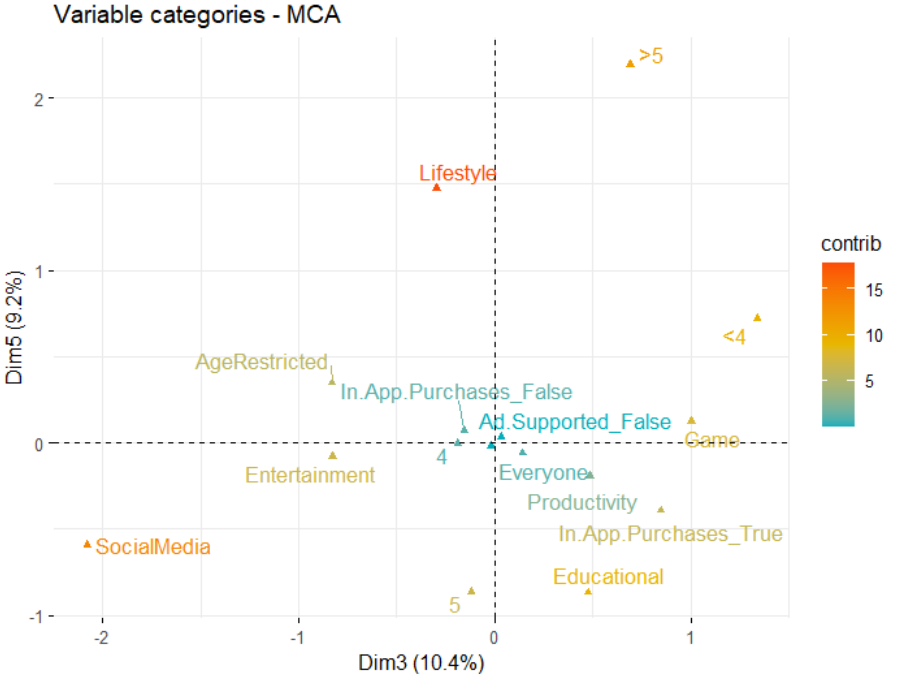


Figure 75: Quality of representation of variable categories in dimension 3-5

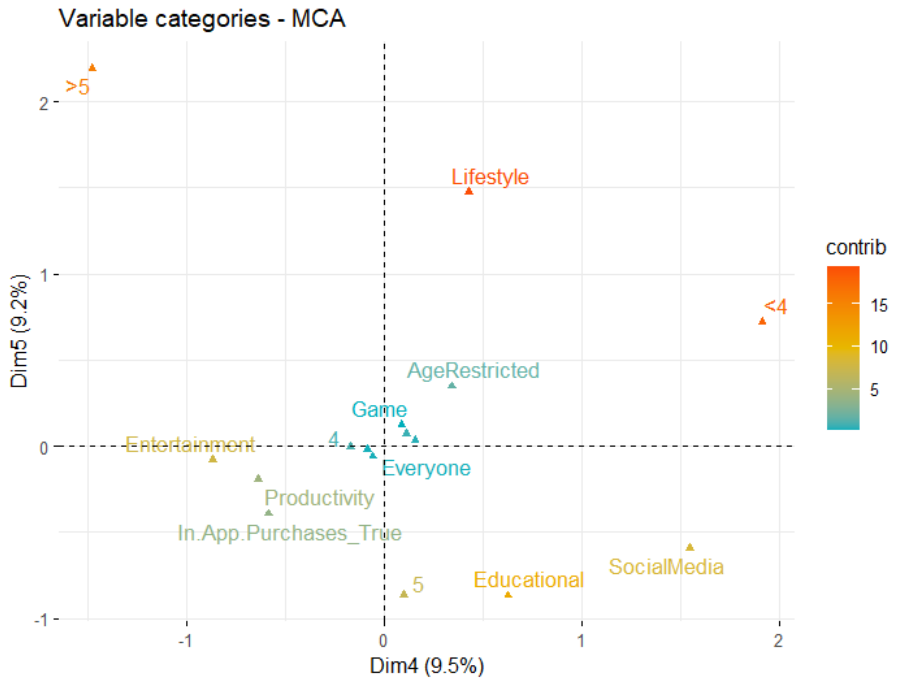


Figure 76: Quality of representation of variable categories in dimension 4-5

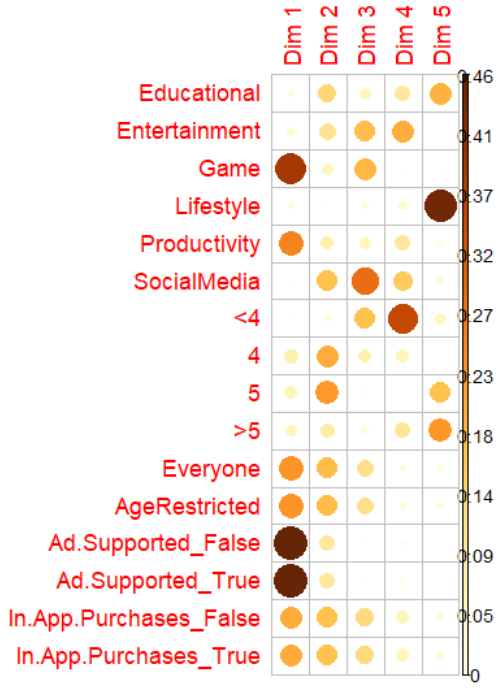


Figure 77: Correlation plot between variable categories and dimensions

Looking at the plots above, on the positive side of dimension 1 there are age restricted gaming apps with in-app purchases and ads. On the negative side of dimension 1 there are productivity apps for everyone with no in-app purchases and ads.

On the positive side of dimension 2 we have social media apps and on the negative side we have educational and entertainment apps.

On the positive side of dimension 3 we have gaming apps that require a minimum android version of less than 4, in other words, old apps and on the negative side we have social media apps.
On the positive side of dimension 4 we have old apps (minimum android version <4) and on the negative side we have new apps (minimum android version >5).
On the positive side of dimension 5 we have lifestyle apps and on the negative side we have educational apps.

7.6 Contribution of variable categories to the dimensions

The plots below show the contribution of variable categories for each dimension.

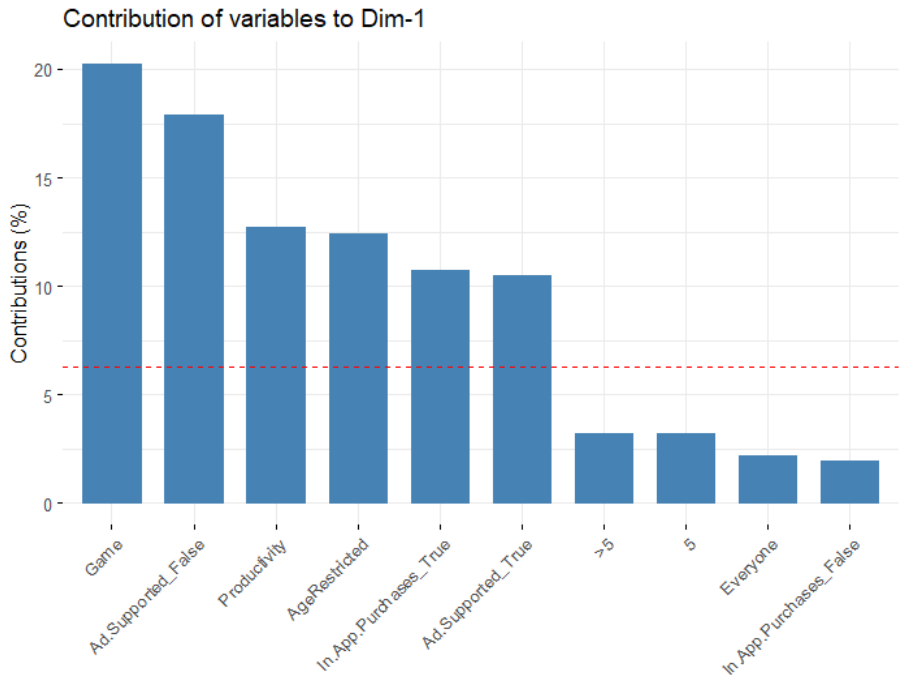


Figure 78: Contribution of variable categories in dimension 1

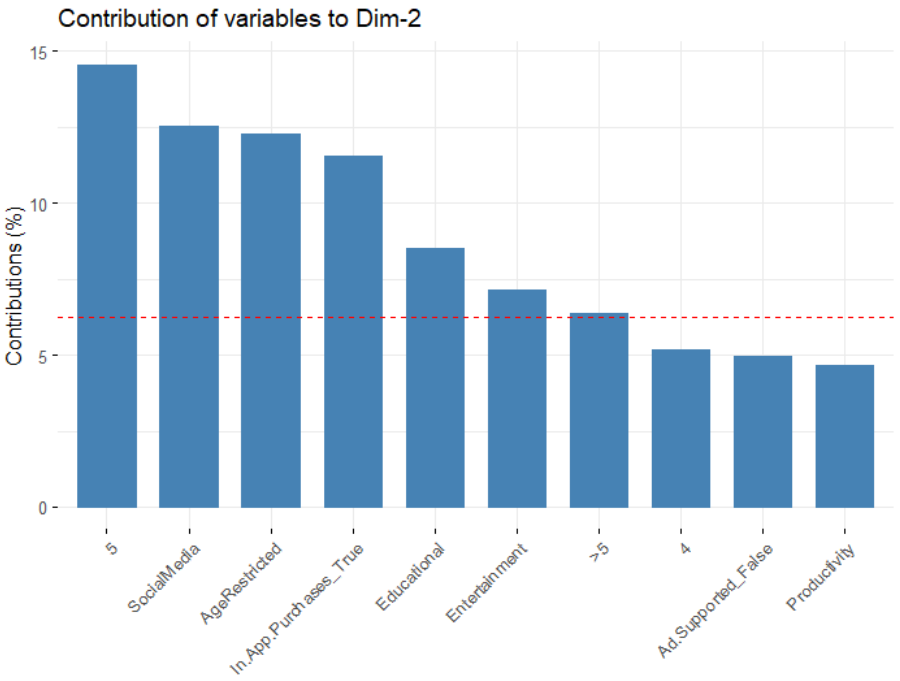


Figure 79: Contribution of variable categories in dimension 2

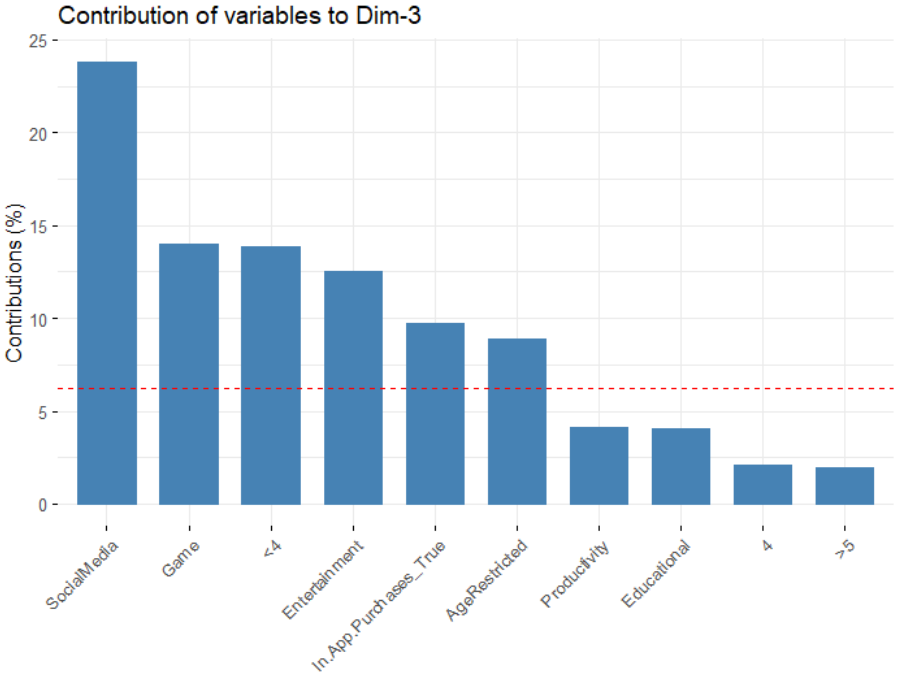


Figure 80: Contribution of variable categories in dimension 3

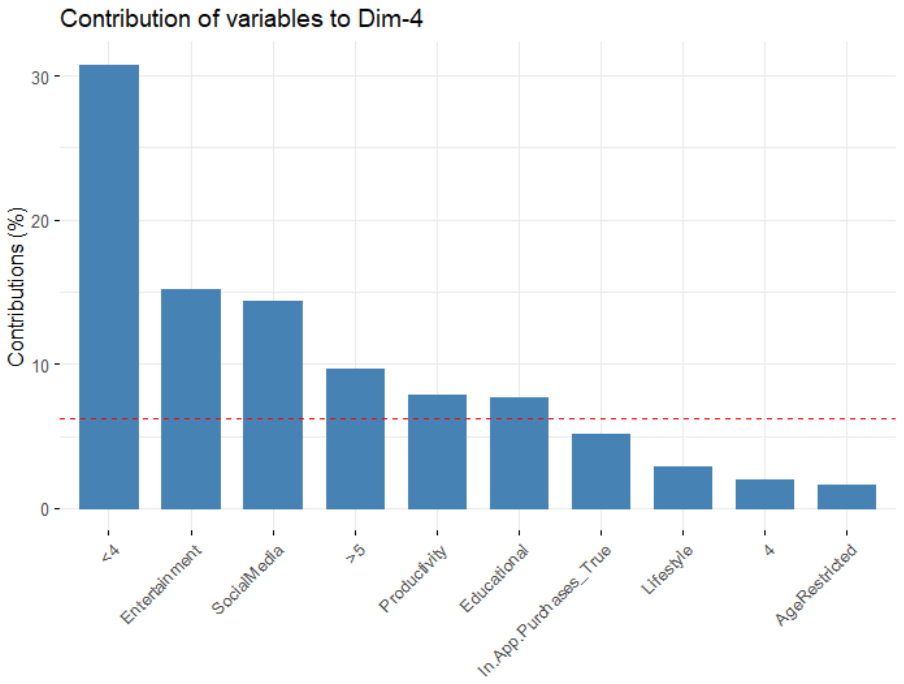


Figure 81: Contribution of variable categories in dimension 4

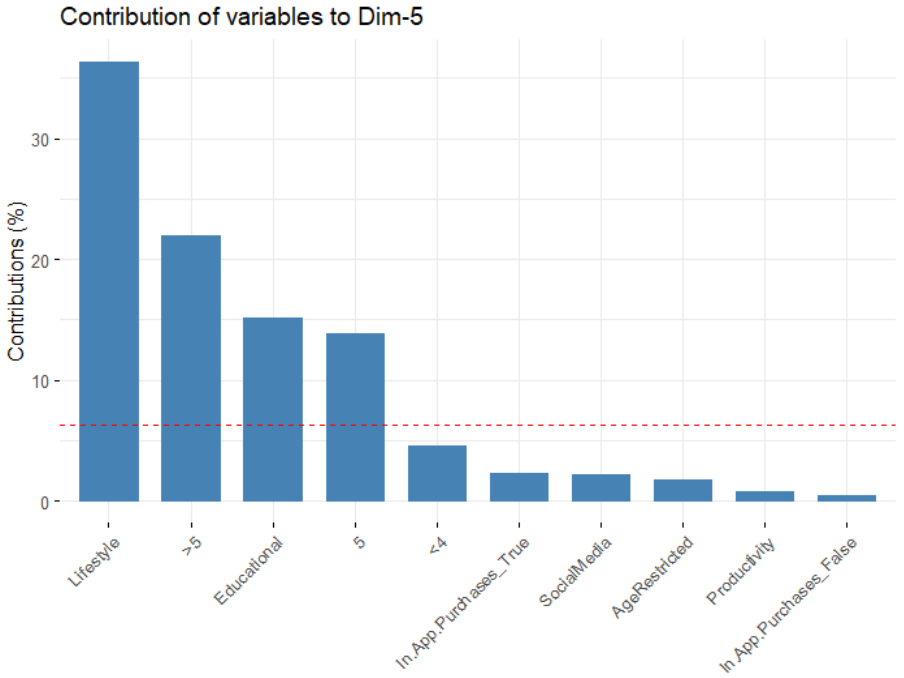


Figure 82: Contribution of variable categories in dimension 5

7.7 Color individuals by groups

Looking at the following plots we can differentiate individuals by groups if their ellipses are separated between them.
We can clearly see all the groups except for `Minimum.Android` in dimension 1-2 and `Ad.Supported` in dimensions 3-4, 3-5 and 4-5.

These plots can help us with the future clustering analysis, because we can find out the group of individuals which are similar to each other in the group but are different from the individuals in other groups.

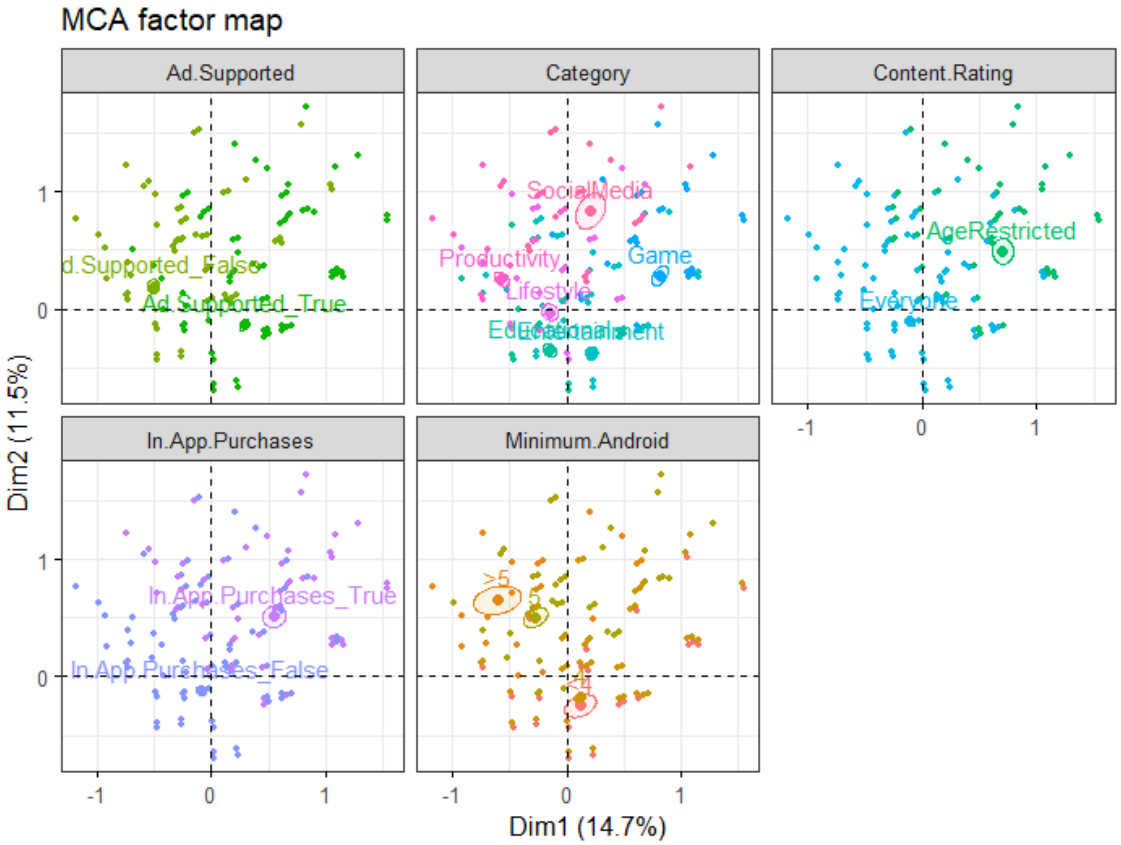


Figure 83: Groups of individuals for each variable in dimension 1-2

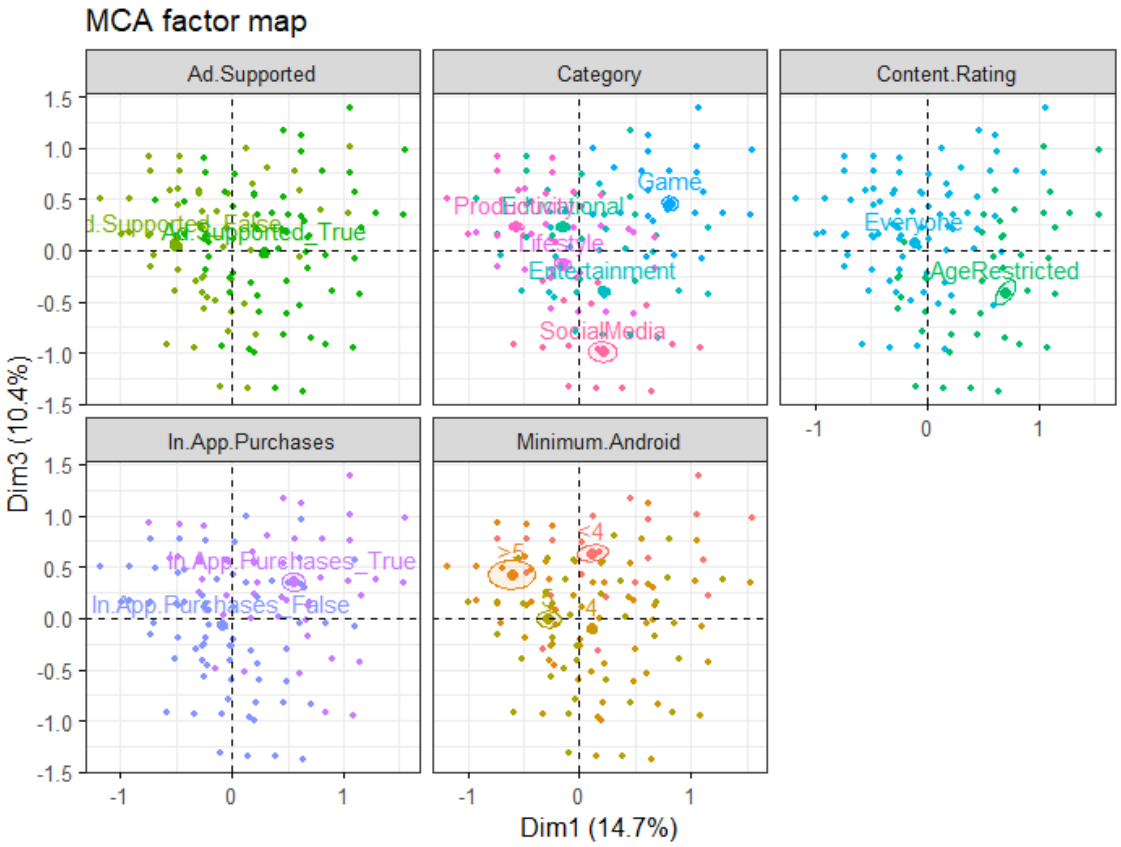


Figure 84: Groups of individuals for each variable in dimension 1-3

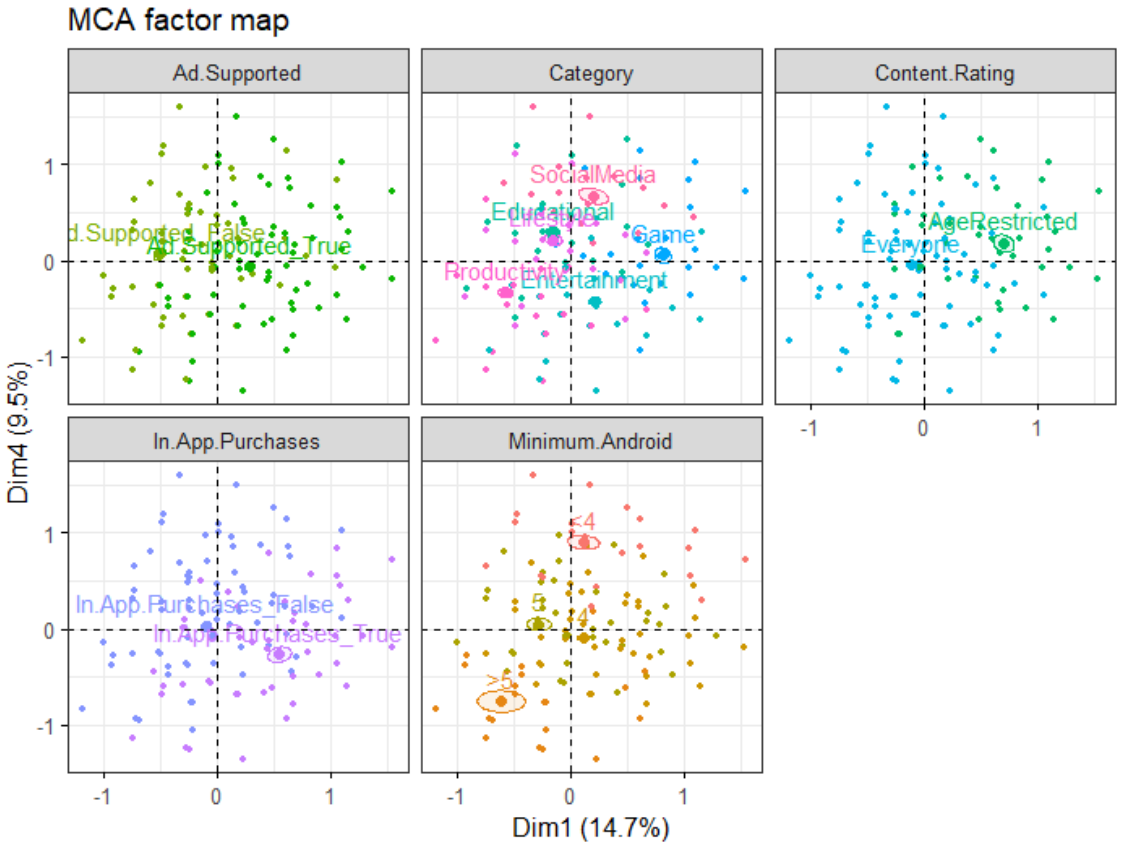


Figure 85: Groups of individuals for each variable in dimension 1-4

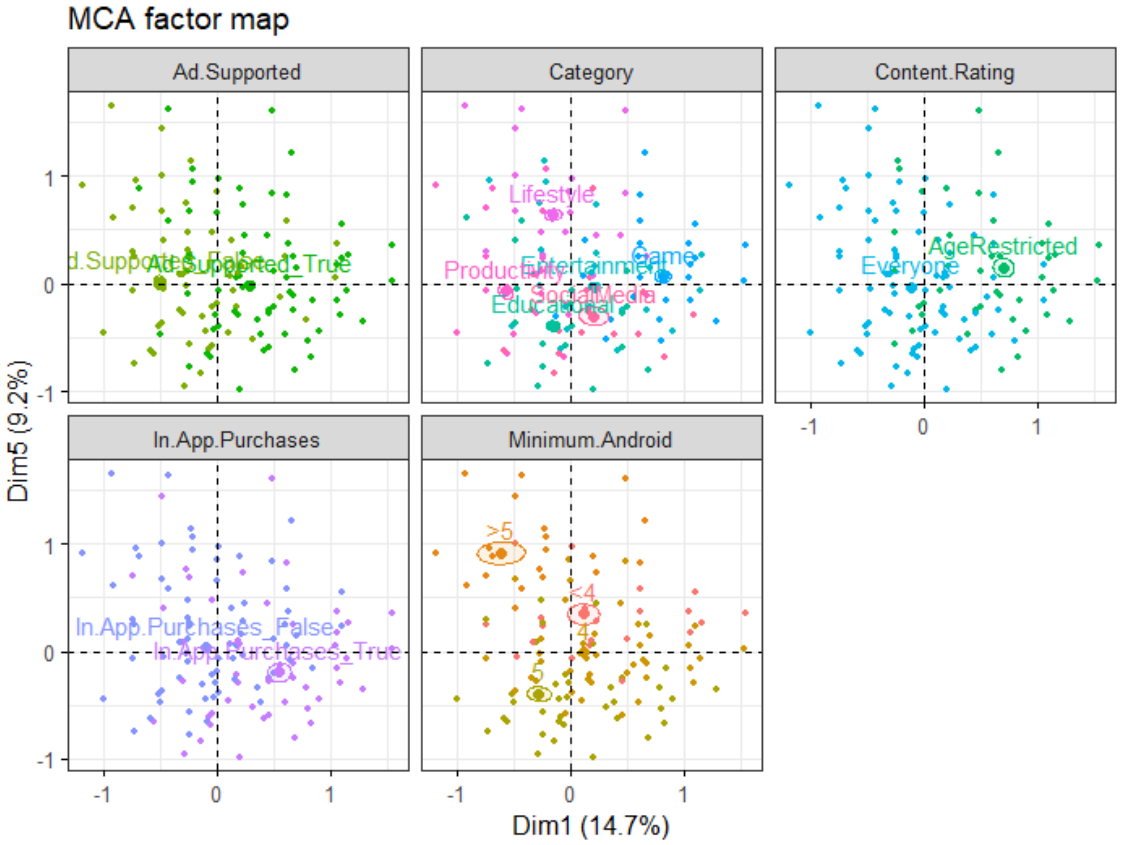


Figure 86: Groups of individuals for each variable in dimension 1-5

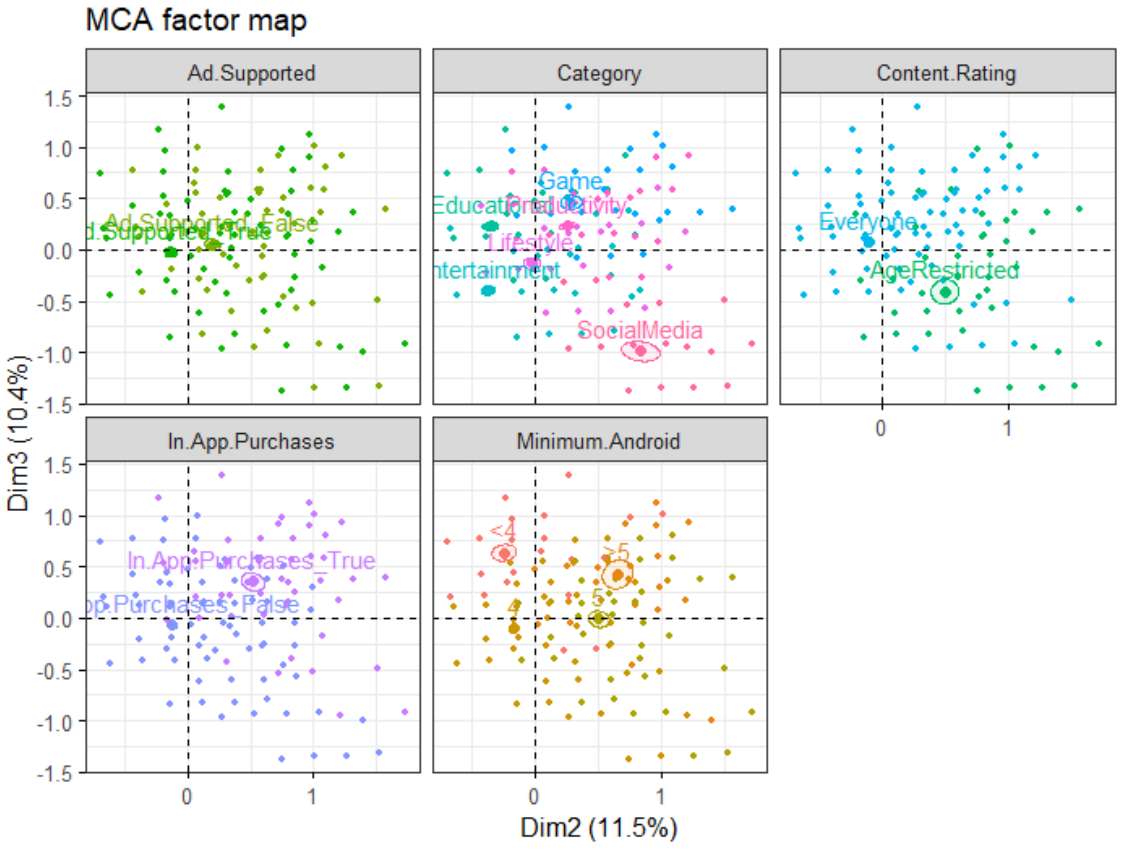


Figure 87: Groups of individuals for each variable in dimension 2-3

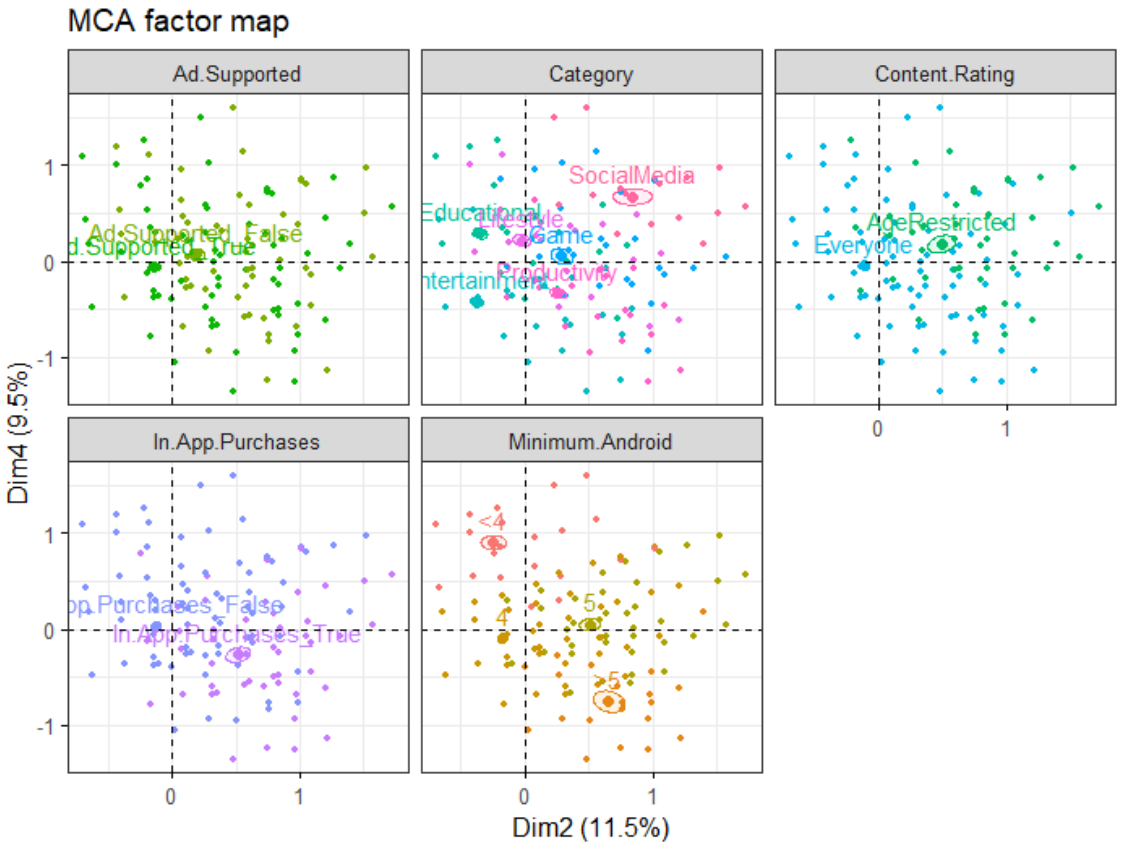


Figure 88: Groups of individuals for each variable in dimension 2-4

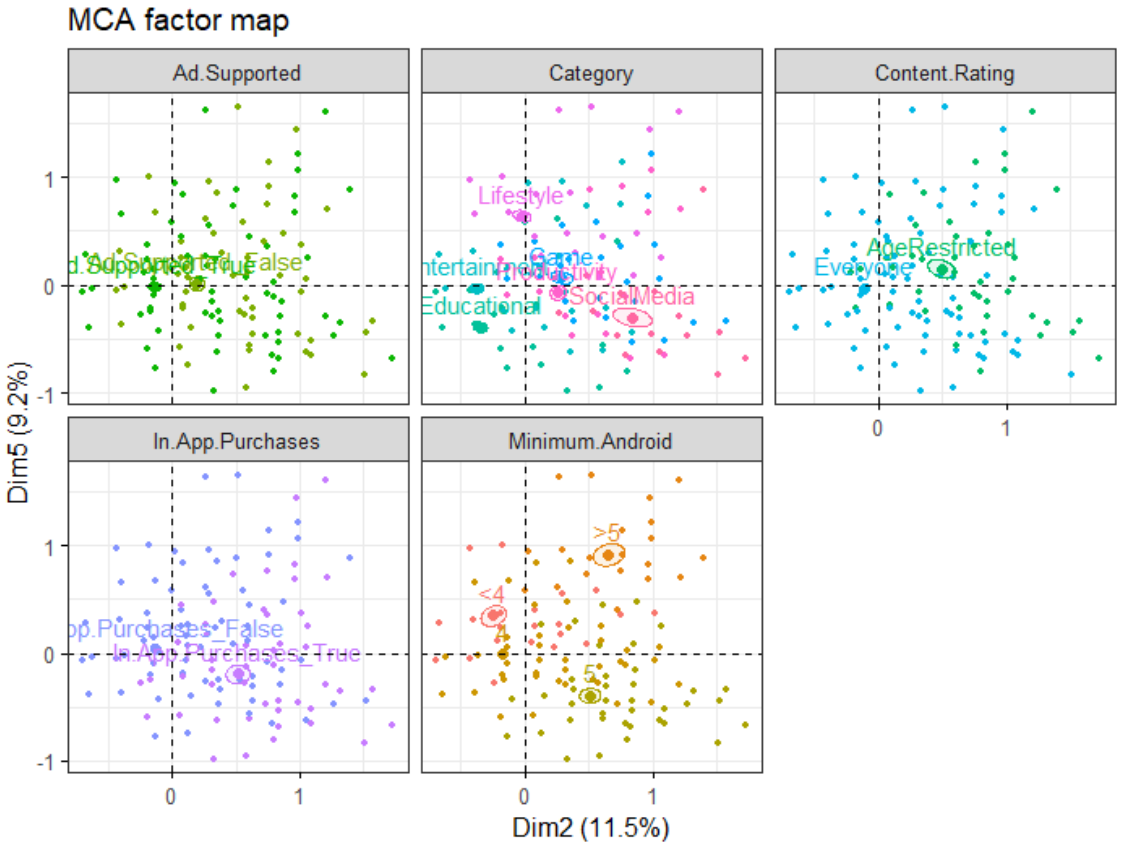


Figure 89: Groups of individuals for each variable in dimension 2-5

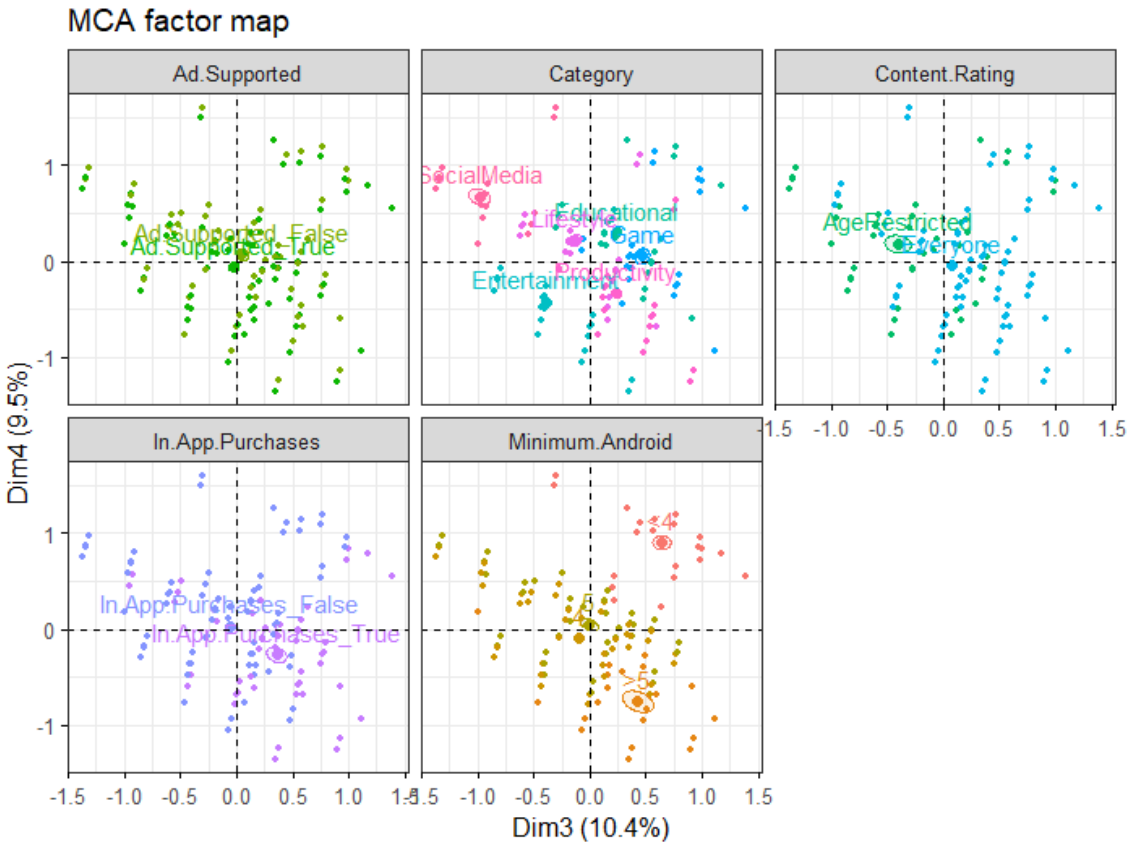


Figure 90: Groups of individuals for each variable in dimension 3-4

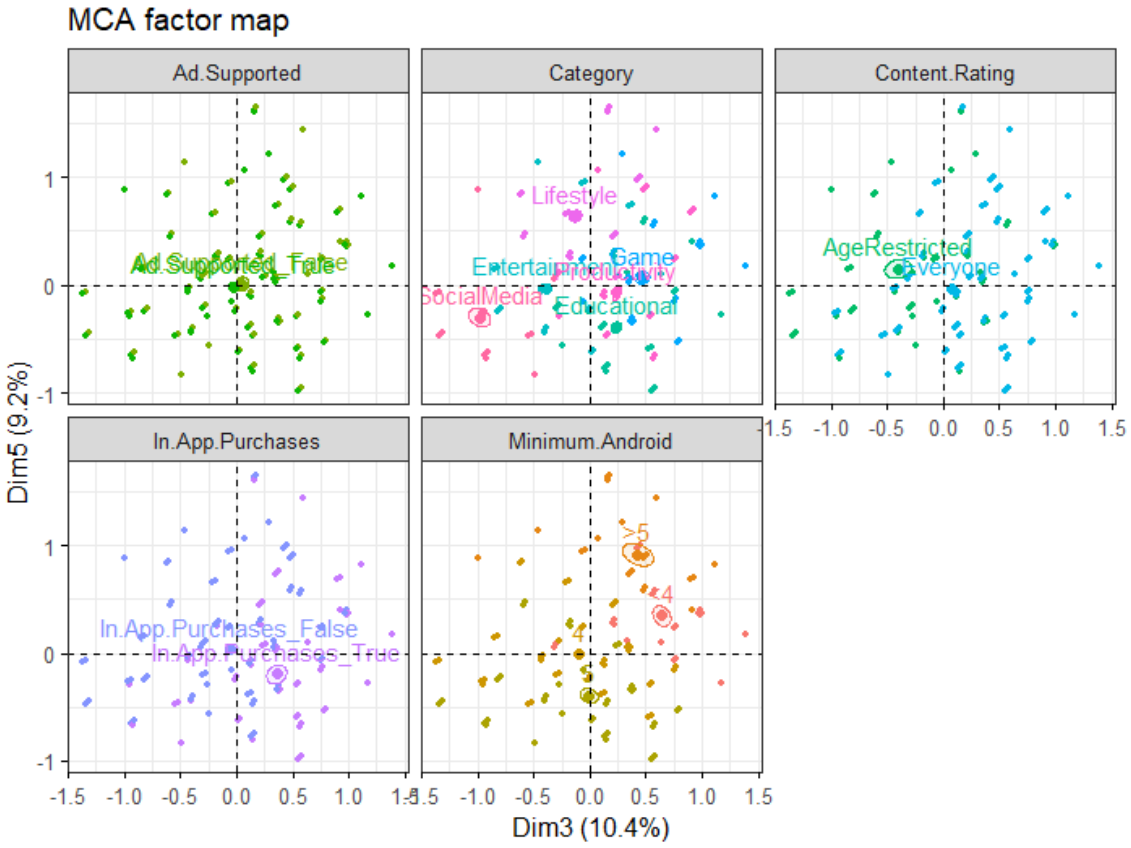


Figure 91: Groups of individuals for each variable in dimension 3-5

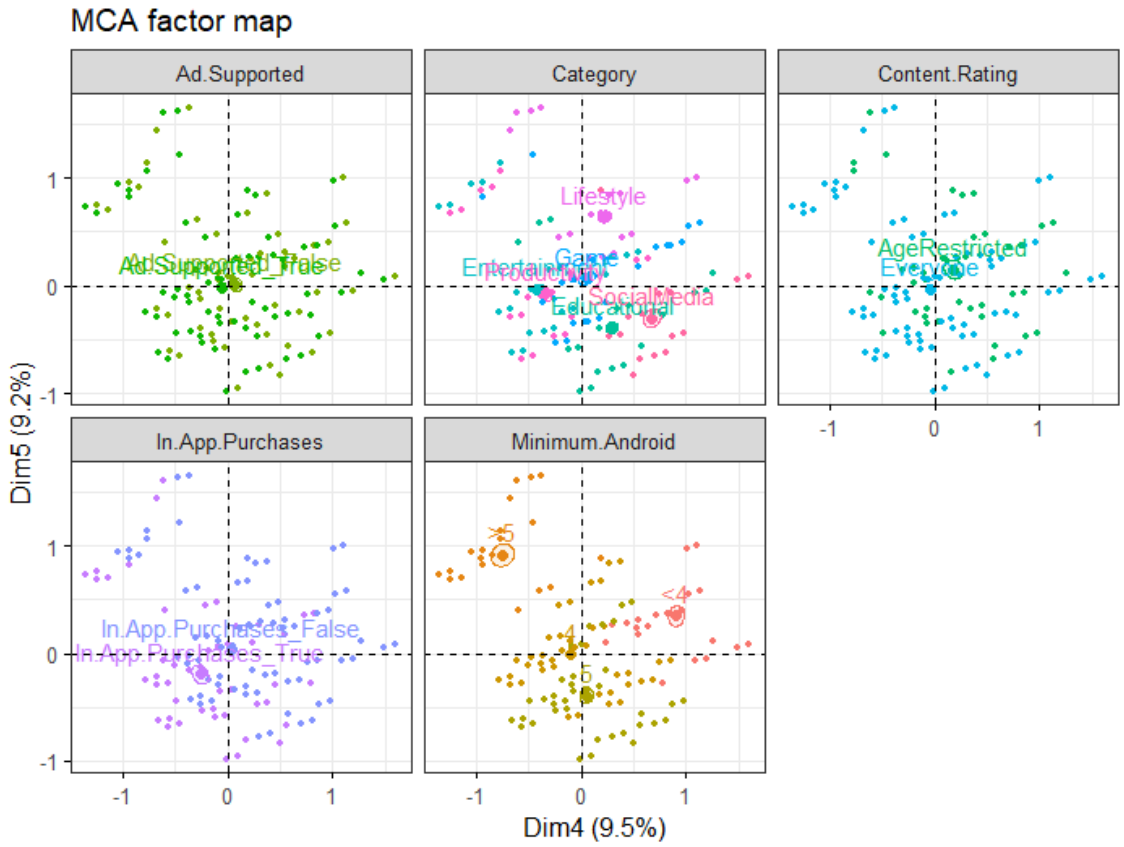


Figure 92: Groups of individuals for each variable in dimension 4-5

7.8 Conclusions

In this section, we are going to try to label our dimensions after performing MCA.
Dimension 1 can be the **level of entertainment** of an app, as on the positive side there are age restricted gaming apps with in-app purchases and ads and on the negative side there are productivity apps for everyone with no in-app purchases and ads.
Dimension 2 can be the **level of procrastination** of an app, as on the positive side we have social media apps and on the negative side we have educational and entertainment apps.
Dimension 3 can be the **level of companionship** of an app, as on the positive side we have gaming apps and on the negative side we have social media apps. Normally, we feel more connected to society using social media apps rather than gaming apps.
Dimension 4 can be the **longevity** of an app, as on the positive side we have old apps (minimum android version <4) and on the negative side we have new apps (minimum android version >5).
Finally, dimension 5 can be **the helpfulness in a person's lifestyle**, as on the positive side we have lifestyle apps (like diet and gym apps) and on the negative side we have educational apps.

8. Multiple Factor Analysis

Multiple Factor Analysis is a factorial method that allows analyzing data with mixed features typed classified in groups. This analysis was performed using the `FMA()` function offered by the `FactoMiner` Package. In order to perform a FAM analysis, we must group the features of the dataset in order to study the data. In total, there were five groups created that describe a specific aspect of the data:

- **Group 1 [Antiquity]:** describes the antiquity of the apps, includes two numerical variables, `DaysLastUpdate` and `ReleasedDays`.
- **Group 2 [Popularity]:** this group also includes numerical features, `Rating.Counts`, `Installs`, and `Rating`, and describes the popularity of the apps.
- **Group 3 [App Features]:** includes two features, `AppNameLen` and `Size`, which express the length of the apps' names and the size in memory.
- **Group 4 [Topic]:** this group includes factorial features, `Category`, `Content.Rating` and `Minimum.Android`, which state the categorization of the apps.
- **Group 5 [Monetization]:** specifies the monetization of the apps, includes two logical features, `Ad.Supported` and `In.App.Purchase`.

The first and fourth groups, `Antiquity` and `Topic`, were declared as supplementary groups, while the rest of the groups remain as active groups. Antiquity and the numeric features were scaled during the analysis in order to compare them in the same units and obtain meaningful results. As a result, we obtain 100% of accumulated variance with 7 dimensions, Figure 93.

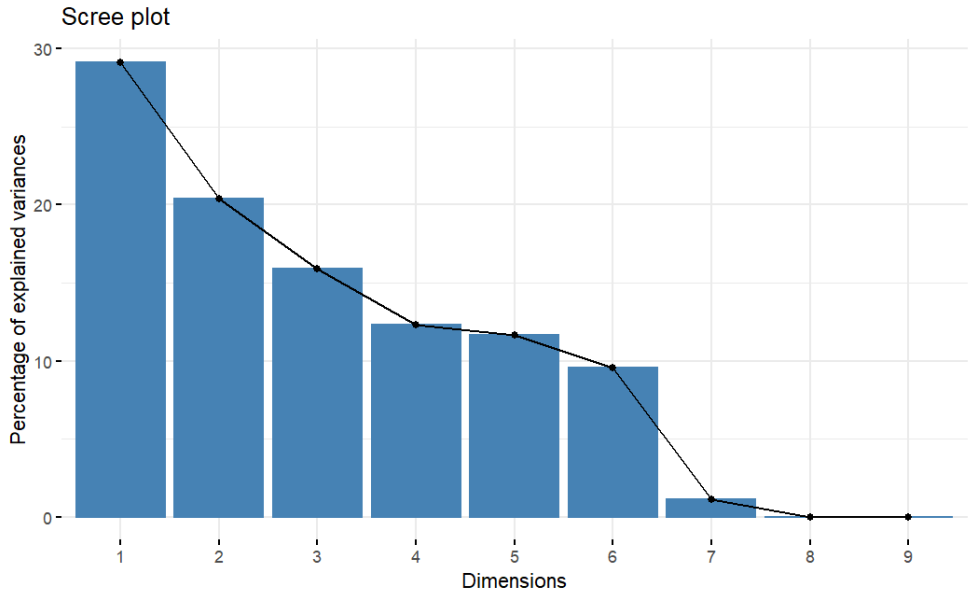


Figure 93: Percentage of explained variances

In the following section, we analyze the results taking the first three dimensions, which achieve 65.35% of cumulative variance.

8.1 Dimensions analysis

In this section, we study the contribution of each active group, and their variables, to each one of the three main dimensions. The plots below show the contribution of each group for each dimension. All the dimensions share the same groups, the differences reside in the contribution of each group in each dimension, but have different contributions. In the case of dim1, the group with the higher contribution is `Monetization`, in dim2 `App Features`, and dim3 `Popularity`. In all dimensions, we can see that the group `Topic` and `Antiquity` do not have a strong contribution compared with the other groups.

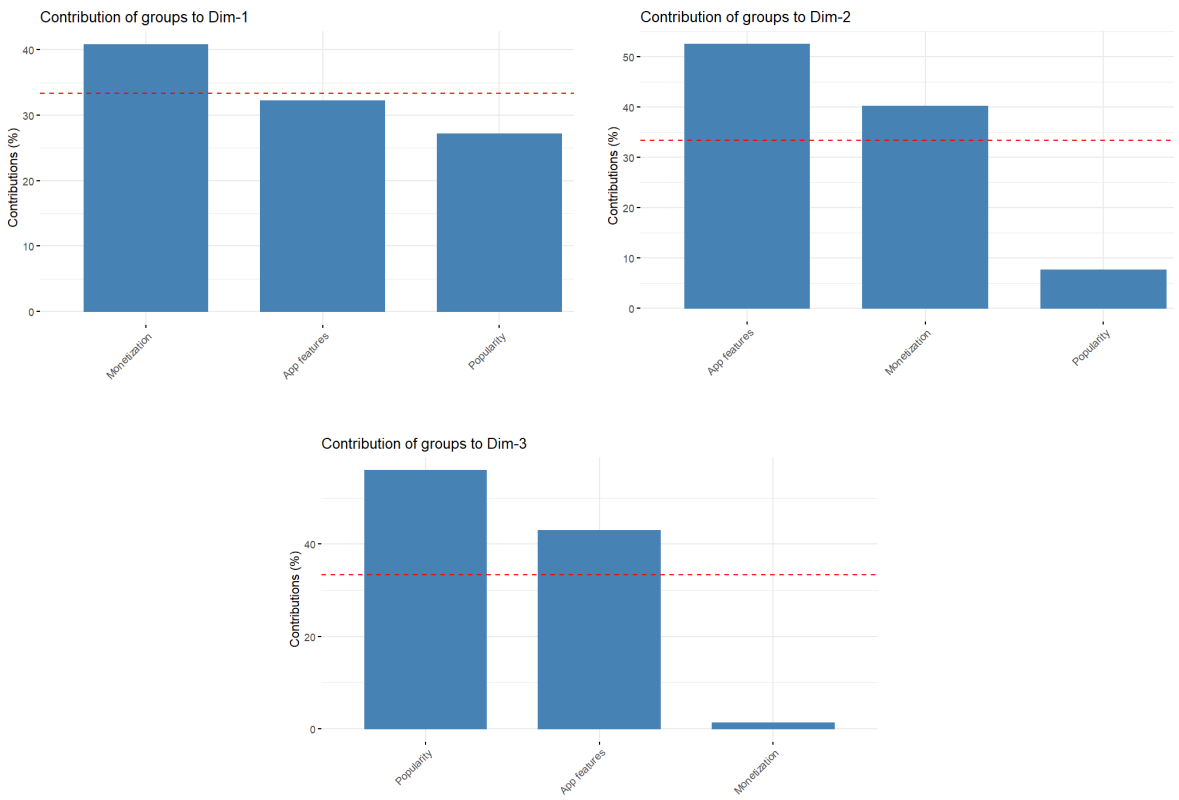


Figure 94: Groups contribution to dim1, dim2 and dim3

Meanwhile, the plots below show the contribution of each feature for each dimension. Variables that contribute the most to dimX are the most important in explaining the variability in the dataset. As we can see in Figure 95, there are variables that contribute more to a specific dimension and less to others. `AppNameLen` has the highest contribution in two dimensions: dim1 and dim2. Another important variable is `Installs`, which appears in dim1 and dim3 as the second main contributor. Another important variables are `Size` and `Rating.Count`, which appear in multiple dimensions as contributors.

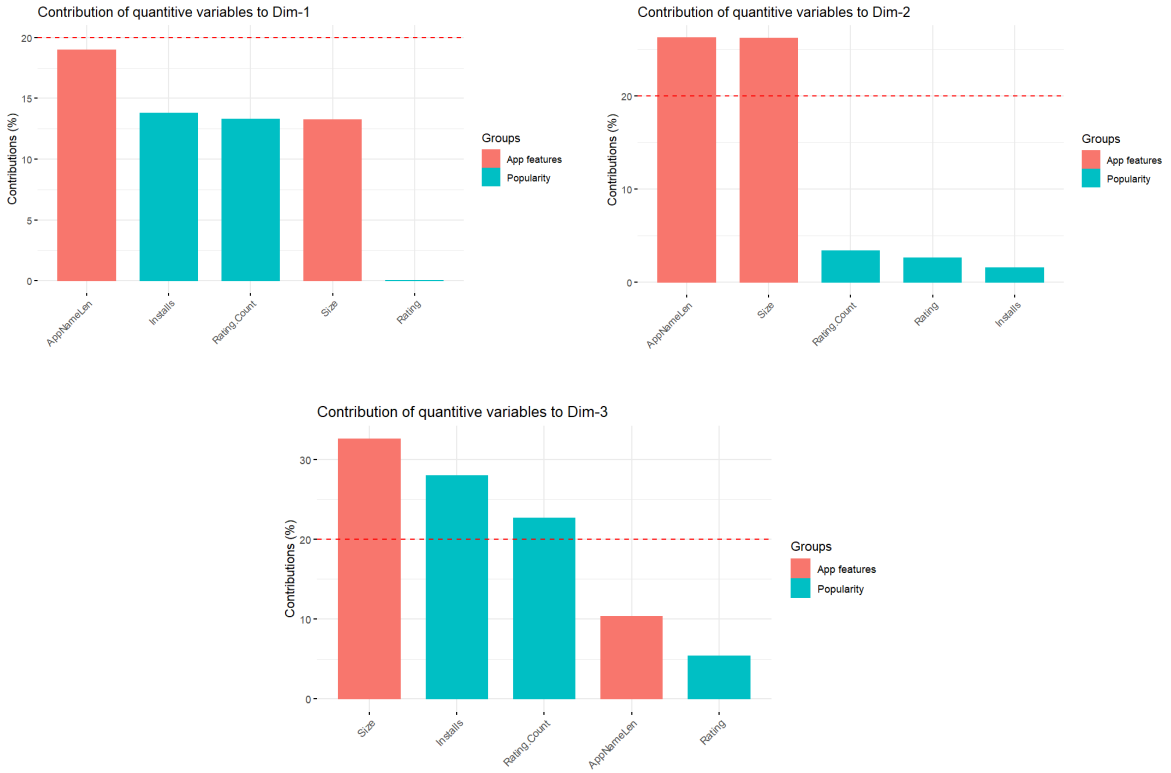


Figure 95: Variables contribution to dim1, dim2 and dim3.

After studying each dimension separately, next we will analyze dimensions by pairs: dim1 & dim2, dim1 & dim3, and dim2 & dim3. In the analysis, we make use of synthetic plots to compare the groups and variables contribution.

- **DIMENSION 1 & 2**

From the perspective of groups, the main contributors of dim1 are, in order of importance, `Monetization`, `App Features` and `Popularity`. While for dim2 are `App Features`, `Monetization` and `Popularity` Figure 96. From the perspective variables, we can see that both dim and dim2 have correlation with `AppNameLength`, as we saw before. Other main contributors for dim1 are `Install` and `Rating.Count`, while for dim2 is `Size`.

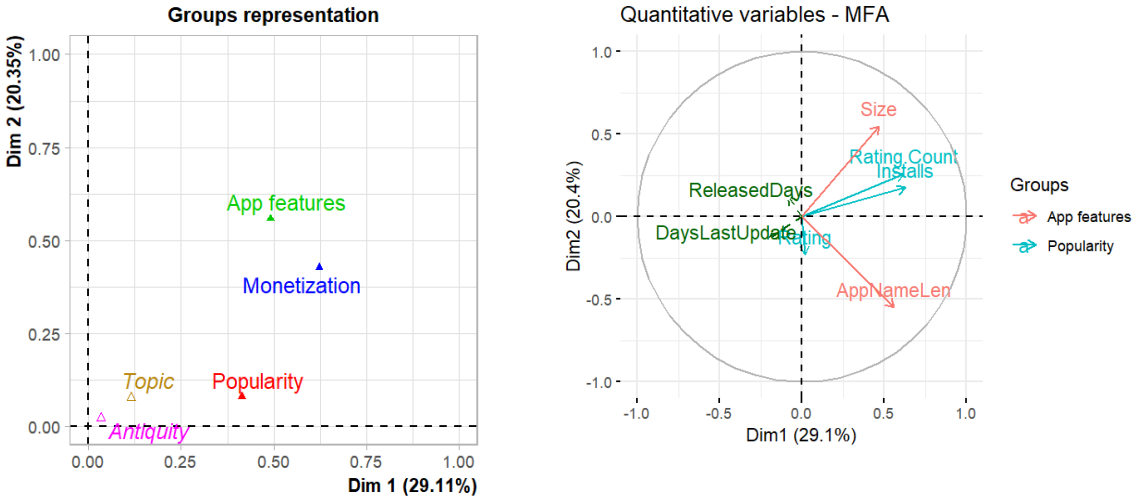


Figure 96: Groups and variables contribution to dim1 and dim2

The plot Figure 96 shows that there is a strong correlation between `Installs` and `Rating.Count`, this last variable also has a light correlation with `Size`. Moreover, `Size` and `AppNameLen` seem to be independent variables. So, the variables of the group `Popularity` are correlated, except `Rating`. And the variables of the `App Features` are independent.

Figure 97 shows the plot of the top 1500 individuals with the highest contribution to dim1 and dim2. We can see a dense cloud of points on the positive axis of dim1. Those individuals have a strong popularity and recent updates. We can also see small cloud points in the negative dim1 side, with a poor popularity and old updates. Other small clusters appear in dim2 negative axis, those are the individuals with higher ratings and middle-long names. This means that apps with shorter names are more popular but have lower ratings, while the unpopular ones have better ratings. This may be explained by the number of ratings. So, the rating does not completely define the popularity of an app, but the number of users does.

We also study how the individuals are seen by different groups. Our aim is to analyze if individuals are seen in the same way by different groups, or there are individuals specific to some groups. As we can see in the plot of Figure 98, the groups, specialty `App Features`, have a different perspective view in respect to the other groups.

The difference between Figure 97 and Figure 98, is that the first one plots the point that corresponds to the center of gravity of the partial points of the individual. That is, the individual viewed by all groups of variables. While Figure 98 plots how each app is viewed by each group and its barycenter.

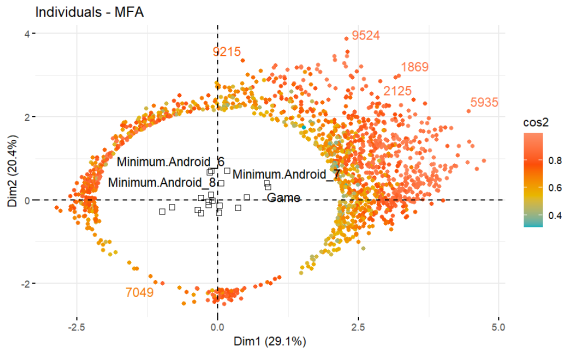


Figure 97: Individuals contribution to dim1&2

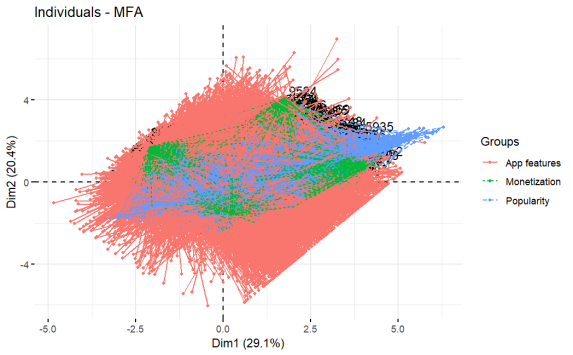


Figure 98: Partial points representation dim1&2

- **DIMENSION 1 & 3**

Figure 99 shows the group contribution for dim1 and dim3. On one hand, dim 3 is correlated mainly, as it does dim1, with `Popularity`, and `App Features`. The fact that `Popularity` and `App Features` are close to each other means that they have several dimensions in common. On the other hand, we have `Monetization`, which is highly correlated with dim1 but very poorly with dim3.
The second plot shows us the contribution of variables for dim1 and dim3. In this case, `Size`, `Installs` and `Rating.Count` have a considerable contribution for both dim1 and dim3. The difference remains in `AppNameLen`, wish has strong correlation with dim1 but no that much with dim3.

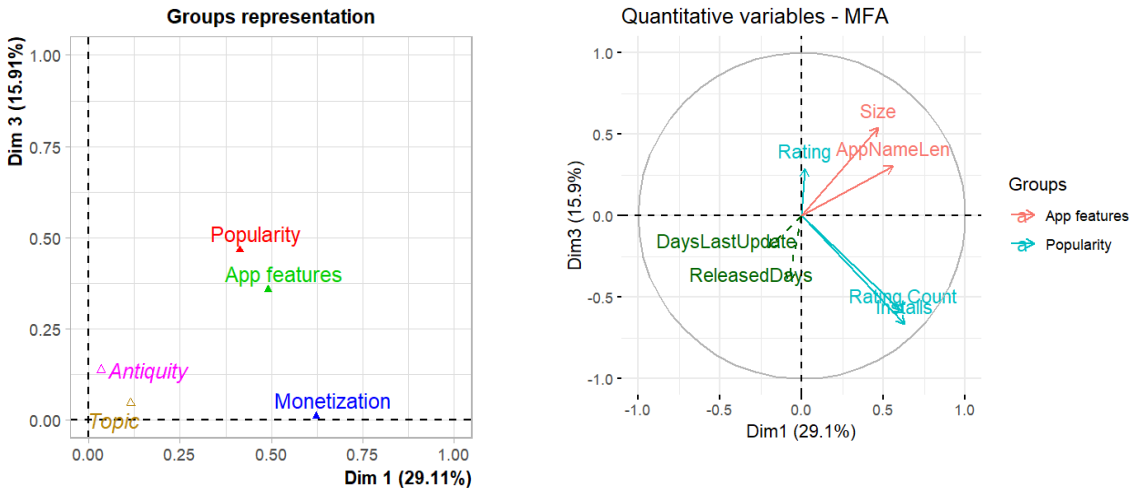


Figure 99: Groups and variables contribution to dim1 and dim3

In Figure 99 we plot the contribution of each variable with dim1 and dim3. This plot shows that dim3 is highly correlated mainly with the `Size`, `Installs` and `Rating.Counts` of the apps, these variables also have a considerable correlation with dim1 as we have seen before. From the plot, it seems that in this case `Size` and `AppNameLen` do have a positive correlation. So, there is a partial correlation between the variables of the `App Features` group.

The Figure 100 below plots the individuals, as the same case we saw before, there is not a clear group of clusters between the individuals, so we are going to summarize the individuals based on the axes. What we can see is that there is a difference in cloud points density between dim1 positive side and negative side. On the positive side of dim1, we have the popular apps (-dim3) and the apps with high size and name length (+dim3), while on the other side we have the unpopular apps (+dim3) and older apps (-dim3). This also indicates to us that the newer apps tend to have shorter names and lower sizes than the older ones.

In Figure 101 we plot the partial points on the individuals in dim1 and dim3. The first thing we can see is that `Monetization`, almost, has the same "view" to all the individuals. `App features` have higher "views" of individuals on axes (+dim1, +dim3) and (-dim1, -dim3).

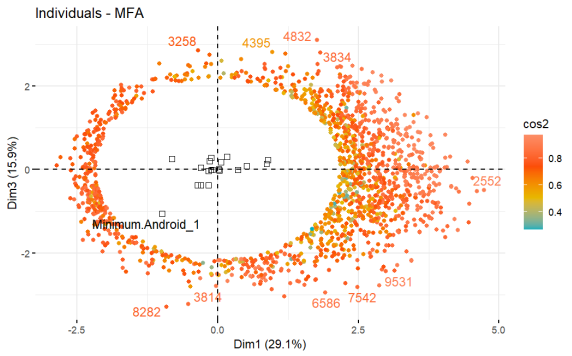


Figure 100: Individuals contribution to dim1&3

Figure 101: Partial points representation dim1&3

- **DIMENSION 2 & 3**

Once again, we plot the group and variable plots, but in this case we are comparing dim2 with dim3. The main difference between dim2 and dim3 is that while the main contributors of dim2, from highest to lowest, are `App Features`, `Monetization` and `Popularity`, while for dim3 is the order.

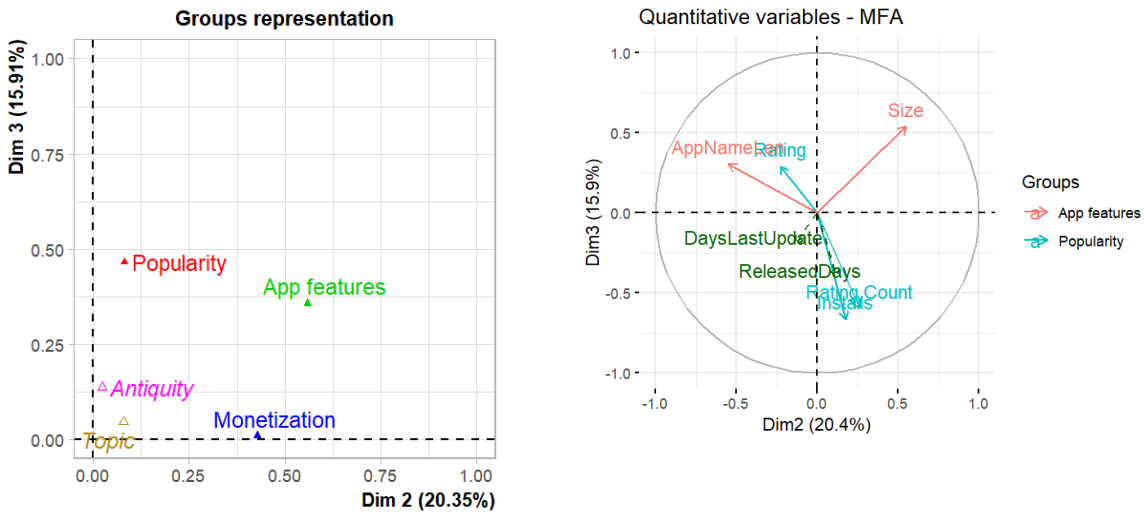


Figure 102: Groups and variables contribution to dim2 and dim3

On the other plot, we have a comparison of the variables of dim2 and dim3. In this case, we see the same scenario as we saw comparing dim1-dim2: the `AppNameLen` variable is independent of the `Size`. What is more is that in this case, `Rating` has an inverse correlation with `Rating.Count` and `Installs`.

Finally, plotting the individuals, we can see a similar behavior compared to what we have seen before. The individuals do not really have clusters groups, all the individuals are spread all over the four axes Figure 103. In the other plot Figure 104, we see how group `Population` and `App Feature` have “strong views” compared to `Monetization`.

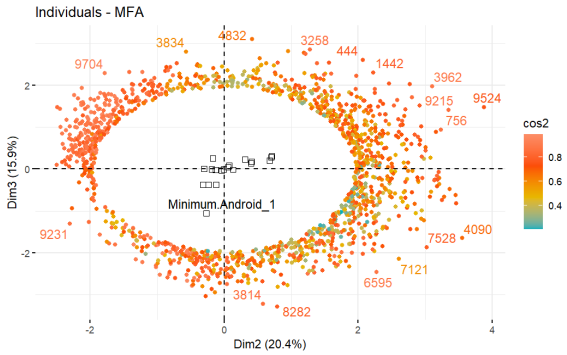


Figure 103: Individuals contribution to dim2&3

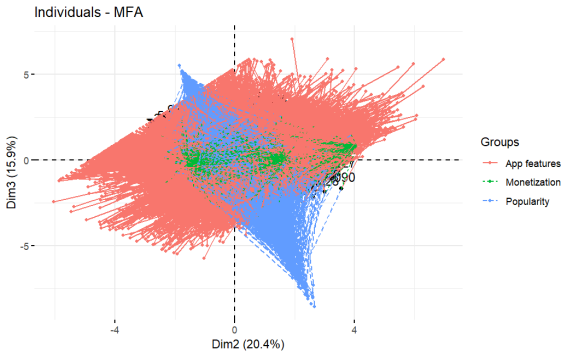


Figure 104: Partial points representation dim2&3

8.2 Conclusions

After the analysis conducted to our data with MFA, setting five groups: `Monetization`, `App Feature`, `Popularity`, `Topic` and `Antiquity`, we conclude the following:

- **About groups:**
  - The `App Feature` group contains more variety of apps compared to other groups, followed by `Monetization` and `Popularity`.
  - The groups are, mostly, not correlated between them.
  - `Monetization` is close to the mean configuration MFA, followed by `App Feature` and `Popularity`.
- **About variables:**
  - There is a relation between the number of installs and the number of ratings of an app, the number of ratings increases proportionally with the number of installs. So, the popularity of an app is defined by the number of installs and ratings.
  - In some cases, when the number of installs increases (and so the number of ratings), the `Rating` decreases, this is because there are unpopular apps that do have higher ratings that the popular ones. His means that rating should not be part of the group Popularity.
  - Usually, the size of an app is independent of the light length of an app name, but in some cases there is a positive correlation between them.
  - There is a light correlation between the days since the last release and days since the last update.
  - Usually, when older is an app, the more popular it is. We can also sometimes see that the newer apps tend to have less size and short names.
- **About individuals:**
  - There are no clear clusters of individuals in the data.
  - En general, not all individuals are seen the same by all the groups, there is a high difference, specially between `App Features` and `Popularity`.

9. Association rules mining analysis

Before we can start we did the same transformations for low frequency modalities as we had done in MCA. Moreover, in order to include numerical variables for our analysis we discretize the numerical values dividing the range of values approximately in 3 equally frequent ranges. The modalities names are the name of the feature plus low, mid or high.

9.1 Identification of the frequent itemsets and the extraction association

First, we identify the most frequent modalities that are listed in the table below.

Modality	Absolute frequency	Relative frequency
Content.Rating=Everyone	10080	0.852
In.App.Purchases=False	10001	0.846
Minimum.Android=4	8009	0.677
Ad.Supported=True	7459	0.631
RatingAPP=Rating High	4463	0.377

In order to give a comprehensive vision of the modalities we can take a look at Figure 105. There are a lot of modalities that have a frequency of 0.33, this is caused by the numerical features discretization.

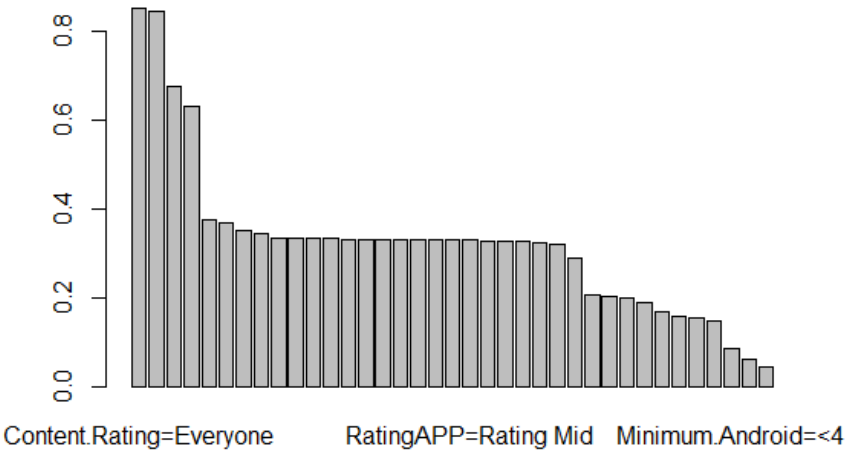


Figure 105: Barplot of modalities frequency

9.2 Rules from the dataset using Apriori

First we check the rules that have our response rule on the right side. As Figure 106 shows, there are few rules and they have poor confidence and low lift. So we can say that these rules are not reliable rules. Moreover all rules have Rating High on the right side and use only 3 modalities for the 4 rules.

```
    lhs                                rhs                          support confidence  coverage     lift count
[1] {Content.Rating=Everyone,
     In.App.Purchases=False,
     Installs=Installs Low}     => {RatingAPP=Rating High} 0.1489092  0.5535995 0.2689836 1.466921  1761
[2] {Content.Rating=Everyone,
     Installs=Installs Low}     => {RatingAPP=Rating High} 0.1608321  0.5478111 0.2935904 1.451583  1902
[3] {In.App.Purchases=False,
     Installs=Installs Low}     => {RatingAPP=Rating High} 0.1660748  0.5472276 0.3034838 1.450037  1964
[4] {Installs=Installs Low}     => {RatingAPP=Rating High} 0.1804499  0.5413496 0.3333333 1.434461  2134
```

Figure 106: Rules for `Rating`

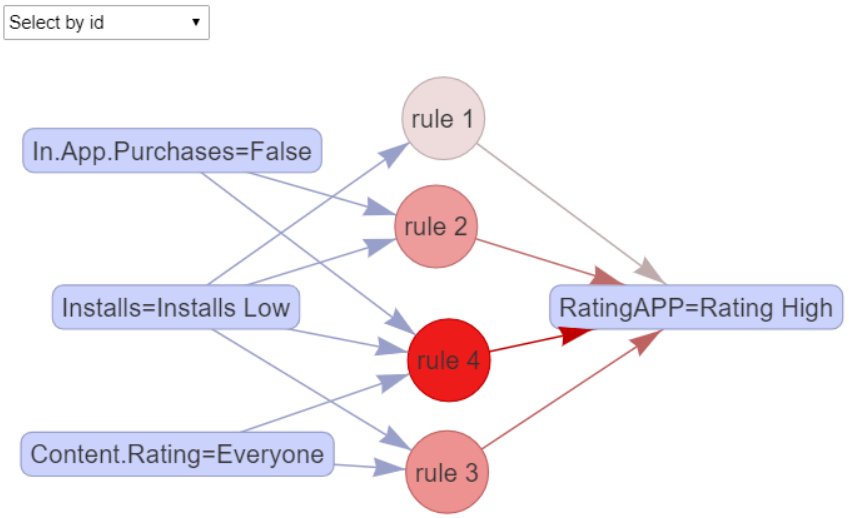


Figure 107: Graph of rules for `Rating`

The 4 rules have `Installs` low and `Rating` high. We have seen this behavior when we analyze the correlation between `Installs` and `Rating`.

An interesting variable for our analysis is `Installs`, due to that is like `Rating`, we want to extract knowledge for maximes these variables. So we checked the rules for this variable. As is shown in Figure 108 there are more rules and with higher confidence and lift. However, it is important to point out that on the left side of these rules there are always `Rating.Count`. As we have seen in previous sections there is a huge correlation between these two variables. Moreover if in future studies we want to predict the rating or installs before the app is released we do not have access to `Rating.Count`. For this reason we try to take more rules for `Installs`, but without using `Rating.Counts`.

	lhs		rhs	support	confidence	coverage	lift	count
[1]	{Rating.Count=Rating.Count High, ReleasedDays=ReleasedDays High}	=>	{Installs=Installs High}	0.1313208	0.8724719	0.1505158	2.617416	1553
[2]	{Rating.Count=Rating.Count High, Minimum.Android=4, Ad.Supported=True}	=>	{Installs=Installs High}	0.1456959	0.8606394	0.1692880	2.581918	1723
[3]	{Rating.Count=Rating.Count High, Content.Rating=Everyone, Ad.Supported=True}	=>	{Installs=Installs High}	0.1592254	0.8598174	0.1851852	2.579452	1883
[4]	{Rating.Count=Rating.Count High, Ad.Supported=True}	=>	{Installs=Installs High}	0.1998140	0.8580247	0.2328767	2.574074	2363
[5]	{Rating.Count=Rating.Count High, Minimum.Android=4}	=>	{Installs=Installs High}	0.1849315	0.8315589	0.2223913	2.494677	2187
[6]	{Rating.Count=Rating.Count High}	=>	{Installs=Installs High}	0.2753256	0.8249303	0.3337561	2.474791	3256
[7]	{Rating.Count=Rating.Count Low, Content.Rating=Everyone}	=>	{Installs=Installs Low}	0.2124979	0.7363024	0.2886014	2.208907	2513
[8]	{Rating.Count=Rating.Count Low, In.App.Purchases=False}	=>	{Installs=Installs Low}	0.2200237	0.7282396	0.3021309	2.184719	2602
[9]	{Rating.Count=Rating.Count Low}	=>	{Installs=Installs Low}	0.2400643	0.7279487	0.3297818	2.183846	2839

Figure 108: Rules for `Installs`

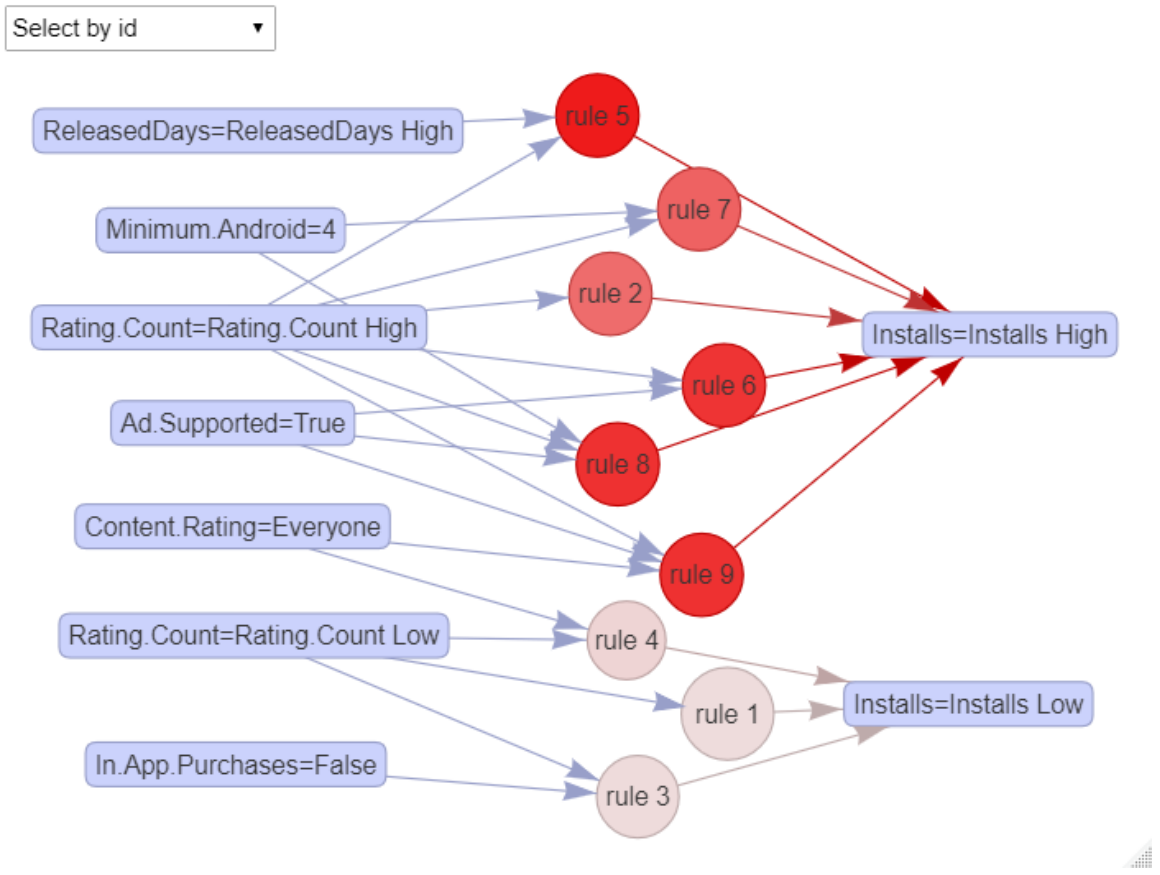


Figure 109: Graph of rules for `Installs`

Now that we have filtered the rules of Installs without using `Rating.Count` (Figure 110), there are fewer rules and with less accuracy and lift. The two rules have `In.App.Purchase` false and give low `Installs`.

```
    lhs                           rhs                      support confidence  coverage     lift count
[1] {Content.Rating=Everyone,
     In.App.Purchases=False,
     RatingAPP=Rating High}    => {Installs=Installs Low} 0.1489092  0.5222420 0.2851344 1.566726  1761
[2] {In.App.Purchases=False,
     RatingAPP=Rating High}    => {Installs=Installs Low} 0.1660748  0.5115916 0.3246237 1.534775  1964
```

Figure 110: Rules for `Installs` without `Rating.Count`

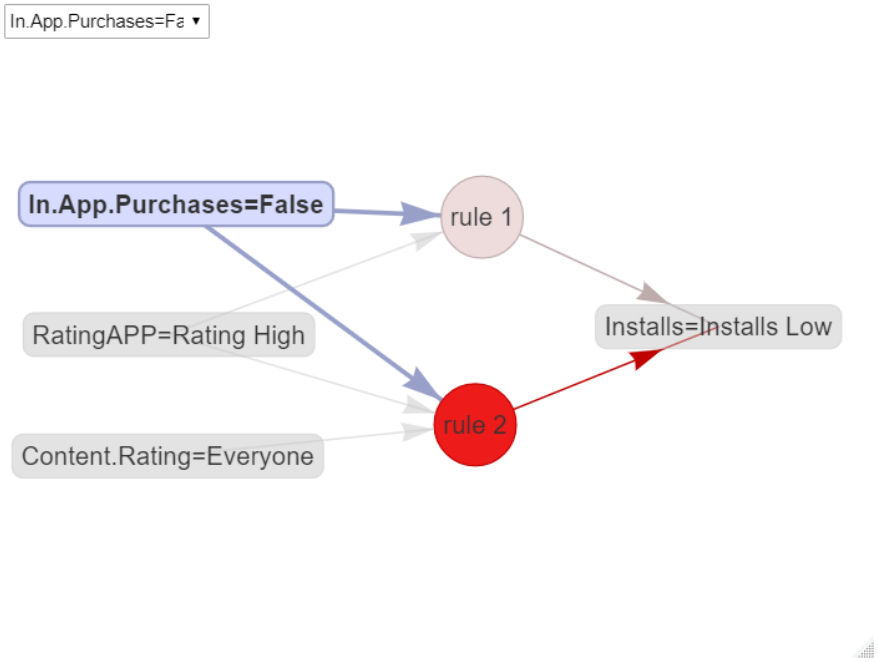


Figure 111: Graph of rules for `Installs` without `Rating.Count`

9.3 Top 20 rules explanation(sorted by decreasing confidence)

```
     lhs                                                rhs                                support confidence  coverage     lift count
[1]  {ReleasedDays=ReleasedDays High,
      Installs=Installs High}                        => {Rating.Count=Rating.Count High} 0.1313208  0.8849003 0.1484018 2.651338  1553
[2]  {Rating.Count=Rating.Count High,
      ReleasedDays=ReleasedDays High}                => {Installs=Installs High}         0.1313208  0.8724719 0.1505158 2.617416  1553
[3]  {Category=Game}                                => {Ad.Supported=True}              0.1367326  0.8679549 0.1575342 1.376114  1617
[4]  {Rating.Count=Rating.Count High,
      Minimum.Android=4,
      Ad.Supported=True}                            => {Installs=Installs High}         0.1456959  0.8606394 0.1692880 2.581918  1723
[5]  {Rating.Count=Rating.Count High,
      Content.Rating=Everyone,
      Ad.Supported=True}                            => {Installs=Installs High}         0.1592254  0.8598174 0.1851852 2.579452  1883
[6]  {Rating.Count=Rating.Count High,
      Ad.Supported=True}                            => {Installs=Installs High}         0.1998140  0.8580247 0.2328767 2.574074  2363
[7]  {Ad.Supported=True,
      In.App.Purchases=False,
      DaysLastUpdate=DaysLastUpdate Mid}            => {Minimum.Android=4}              0.1574497  0.8376068 0.1879756 1.236801  1862
[8]  {Minimum.Android=4,
      AppNameLen=AppNameLen High}                   => {Ad.Supported=True}              0.2120751  0.8373957 0.2532555 1.327663  2508
[9]  {Rating.Count=Rating.Count High,
      Minimum.Android=4}                            => {Installs=Installs High}         0.1849315  0.8315589 0.2223913 2.494677  2187
[10] {ReleasedDays=ReleasedDays Low,
      AppNameLen=AppNameLen High}                   => {Ad.Supported=True}              0.1302215  0.8315335 0.1566041 1.318369  1540
[11] {Ad.Supported=True,
      In.App.Purchases=False,
      ReleasedDays=ReleasedDays Mid}                => {Minimum.Android=4}              0.1437511  0.8308895 0.1730086 1.226882  1700
[12] {Content.Rating=Everyone,
      Ad.Supported=True,
      ReleasedDays=ReleasedDays Mid}                => {Minimum.Android=4}              0.1444275  0.8263183 0.1747844 1.220132  1708
[13] {Installs=Installs High}                       => {Rating.Count=Rating.Count High} 0.2753256  0.8259767 0.3333333 2.474791  3256
[14] {Ad.Supported=True,
      ReleasedDays=ReleasedDays Mid}                => {Minimum.Android=4}              0.1736005  0.8258246 0.2102148 1.219403  2053
[15] {Rating.Count=Rating.Count High}               => {Installs=Installs High}         0.2753256  0.8249303 0.3337561 2.474791  3256
[16] {Ad.Supported=True,
      DaysLastUpdate=DaysLastUpdate Mid}            => {Minimum.Android=4}              0.1840014  0.8220627 0.2238289 1.213849  2176
[17] {Category=Entertainment,
      Minimum.Android=4}                            => {Ad.Supported=True}              0.1258245  0.8144499 0.1544901 1.291284  1488
[18] {Minimum.Android=4,
      Installs=Installs High}                       => {Ad.Supported=True}              0.1815491  0.7987351 0.2272958 1.266368  2147
[19] {AppNameLen=AppNameLen High}                   => {Ad.Supported=True}              0.2766785  0.7882438 0.3510063 1.249735  3272
[20] {Category=Entertainment,
      In.App.Purchases=False}                       => {Ad.Supported=True}              0.1426518  0.7738532 0.1843396 1.226919  1687
```

Figure 112: Top 20 rules

The rules 1, 2, 4, 5, 9, 13 and 15 all have the correlation between `Installs` and `Rating.Count` that we have seen. The rules 3, 8, 10, 17, 18, 19 and 20 all have the `Ad.Supported` true, but all of them have a low lift. A quick summary of the more important rules are that the apps are games, have a long name or have a minimum android 4 have ad supported. The rules 7, 11, 12, 14 and 16 all have the `Minimum.Android` 4, but all of them have a low lift. A quick summary of the more important rules are that the apps are not in purchase, have ads or his release date is mid have minimum android 4.

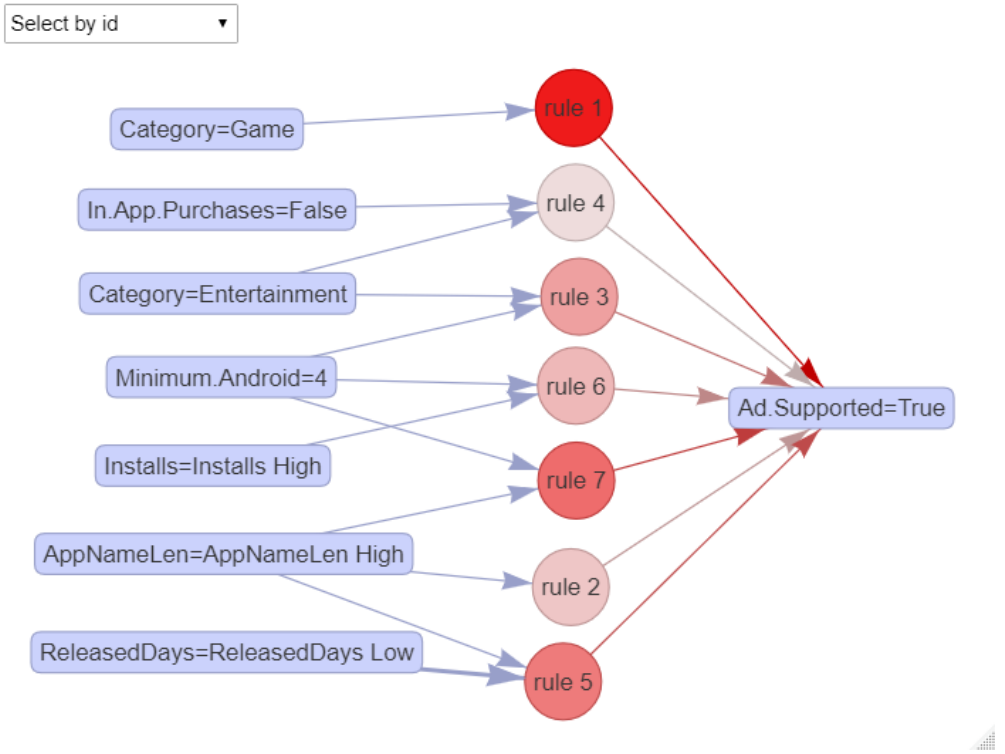


Figure 113: Graph of rules for `Ad.Supported`

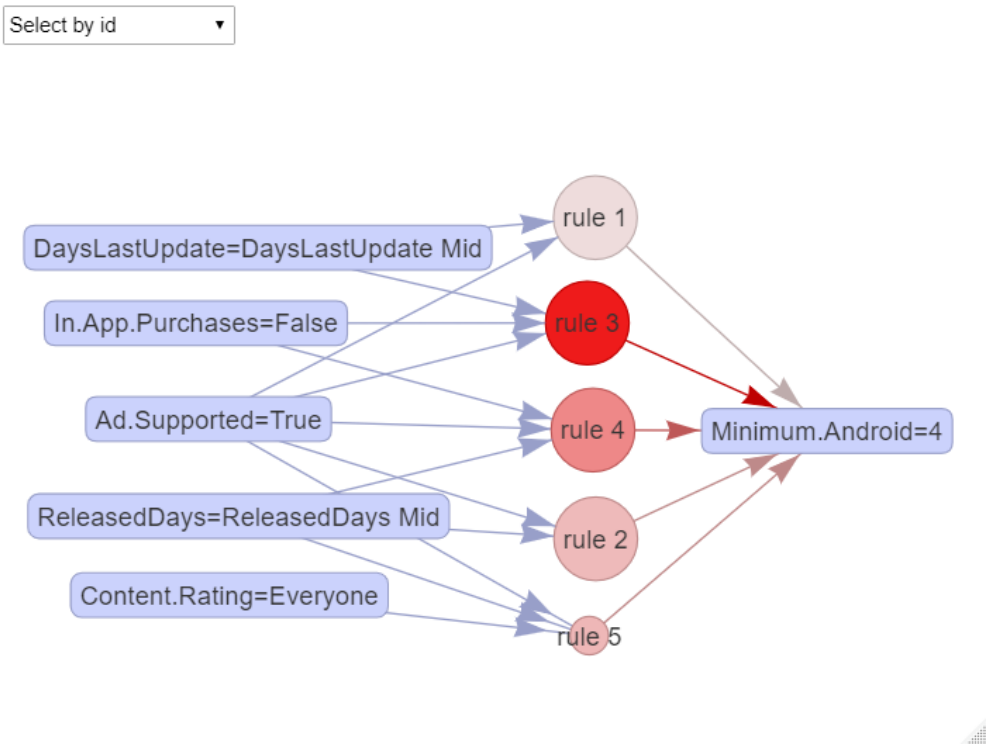


Figure 114: Graph of rules for `Minimum.Android`