Analysis on the Facebook Metrics Data Using Clustering

Introduction

Social media has already been an important platform for retail companies to keep themselves exposed in front of the public, interact with customers individually, and advertise their products. Past research (*Ling et al., 2005*) has proven the necessary needs for brands to maintain a social presence across social platforms. And most importantly, as mentioned by *Alsayat & El-Sayed (2016)* as well, it is also prevailed for retail companies to optimise their marketing strategies using social media for customer relationship management, branding, and risk prevention.

The motivation behind this research stems from recognizing the pivotal role of social media in the retail industry, where maintaining an active presence and optimizing marketing strategies are essential for success. And an easy way for doing this is to analyse the performance of social media and find the patterns of successful posts. We put our thoughts into practice by using *Facebook Metrics Data*, provided by *Moro et al. (2016)*, which is about posts published during the year of 2014 on the Facebook's page of a famous cosmetics company, and contains 500 rows and 18 features.

The main research question of this report is to explore engagement patterns within the *Facebook Metrics Data*. In addition, we will also try to identify key performance metrics that significantly impact the engagement and effectiveness of social media posts on Facebook. Specifically, we will conduct multivariate analysis and data mining by leveraging *Principal Component Analysis (PCA)* and clustering methods *(K-means* and *Hierarchical Clustering*) on *Facebook Metrics Data*, to understand the performance of social media posts in an informative way. Hopefully, with our analysis, we may discover valuable insights by identifying distinct clusters of posts with similar engagement patterns, and companies may optimize their Facebook content strategies and enhance the influence of their social media accounts.

We implement proposed methods on this data set and find out 4 clusters using *K-means* clustering and 3 clusters using *hierarchical clustering*. But only the hierarchical clustering with *Ward's* linkage demonstrates acceptable results. And we also identify *Engaged.Users* and *Post.Consumptions* potential key performance variables that can be useful for classification of high-performance posts. However, our clustering results require further analysis to assess their robustness and interpretation.".

Data

We download *Facebook Metrics Data* from the website of *UCI Machine Learning Repository* and it is freely accessible for the public. This data set provides information about Facebook posts made by cosmetic companies, with various metrics related to the engagement, time and types of these posts. After removing rows with missing values, we have 495 observations and 19 features with data types including continuous, discrete and categorical. Notably, the continuous data lacks consistency in scales.

Since *PCA* is only suitable for continuous data, we decide to discard categorical and discrete data, such as *Type*, *Category*, *Post.Month*, *Post.Weekday*, *Post.Hour* and *Paid*. By doing so, there are 13 features left in the data also demonstrating high correlation with each other, which further justifies our choice of using *PCA* (*Figure 1*). And this method also requires data scaled and centred, which means data needs to have mean equal to 0 and standard deviation equal to 1. The reason behind this is that we have to make sure that each original variable contributes the same weight to each principal component, and the principal components describe the direction of variance, otherwise the components might correspond more closely to the mean of the data.

dimension of the data, *PCA* can be helpful with both the clustering process and filtering out noise by keeping only several important components without losing a large amount of information.

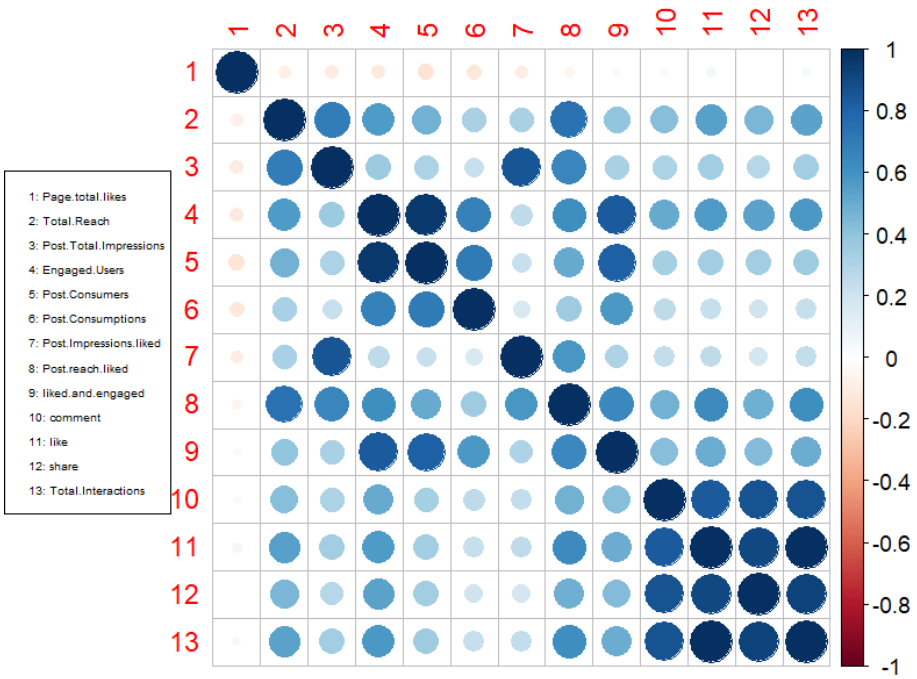


Figure 1: Correlation Plot
(The numbers represent name of variables and the shade of colour represent the level of correlation; several variables have significant positive correlation.)

Methods

Principal Component Analysis: It is a popular method for dimension reduction. The main base of PCA is that it selects the variables that contribute the largest variance to the whole dataset. Therefore, we only need the first few Principal Components (PCs) in the data analysis since they capture most of the variation in the original data set. Mathematically, *PCA* is equivalent to applying *Singular Value Decomposition (SVD)* on the covariance matrix of the data (*Yeung & Ruzzo, 2001*). By reducing the

K-means Clustering: As an unsupervised learning algorithm, *K-means* clustering is computationally efficient and suitable for exploratory analysis without any predefined labels. And in our case, we use this method to explore the engagement patterns of Facebook posts made by cosmetic companies without prior knowledge of specific clusters. This algorithm asks for a pre-specified number of clusters (denoted by *k*) and, based on this number, it will estimate the group membership of each observation and means of each cluster until the variance within each group, or *Within Cluster Sum of Square (WCSS)*, is minimized.

To implement this algorithm, our first job is to find the optimal number of clusters. Normally, we use *Silhouette score*, which evaluates the quality of clustering, to help us determine the optimal number of clusters. More specifically, it is the mean *Silhouette coefficient* over all the instances. And *Silhouette coefficient* can vary from -1 to 1. A coefficient more closed to 1 indicates the data point is well clustered, while a coefficient close to 0 means that it is close to a cluster boundary, and a score more closed to -1 may indicate a wrong clustering. Another method is to evaluate the ability of reducing *WCSS* at each number of clusters. In theory, when the number of clusters is one more than the optimal number, we would see a significant decrease in the amount of variation explained by the algorithm. We visualize the selection process by using an elbow plot, which plots the total *WCSS* against the number of clusters.

Hierarchical Clustering: The *Hierarchical Clustering* assumes the data has a hierarchical structure, in other words, individual observations are grouped into small clusters according to their similarities and small clusters are grouped into bigger clusters. The specific algorithm we applied to our data is *agglomerative hierarchical clustering*, and this approach produces hierarchical clustering by starting with each data point as a single cluster and then repeatedly merging the two closest clusters until all data points are in one group (*Gupta, 2014*). Furthermore, we choose *Euclidean distance* as the

measurement similarity between observations, and there exist many different linkage measures that define the distance between pairs of clusters. The three most common linkage measures are *single*, *complete*, and *average*, which define the distance between two clusters in terms of the minimum, maximum, and average of all distances between pairs of observations from different clusters, respectively (*Yim & Ramdeen, 2015*). Beside these three methods, we also introduce *Ward's* method to the algorithm. It computes *WCSS* for current clustering and merge the two clusters that lead to smallest increase in *WCSS*.

A *hierarchical clustering* is often displayed graphically using a tree-like diagram called a *dendrogram*, which displays both the cluster-subcluster relationships and the order in which the clusters were merged (*Gupta, 2014*). It is very helpful in determining where the hierarchical clustering should be stopped by looking at the height of the link, which means the distance between two data points or clusters. So, we apply dendrograms to the outputs and select best clustering from algorithms with different linkages.

Results

Principal Component Analysis: The *PCA* returns us a result that the first five principal components capture over 90% variation of the original data (*Figure 2*). However, we discover outliers when visualizing the principal components. *Xu & Wunsch (2005)* mentioned that *K-means* algorithm is sensitive to outliers, which may distort the cluster shapes. Therefore, we need to remove these outliers, at least part of them to reduce their effect on clustering to minimum.

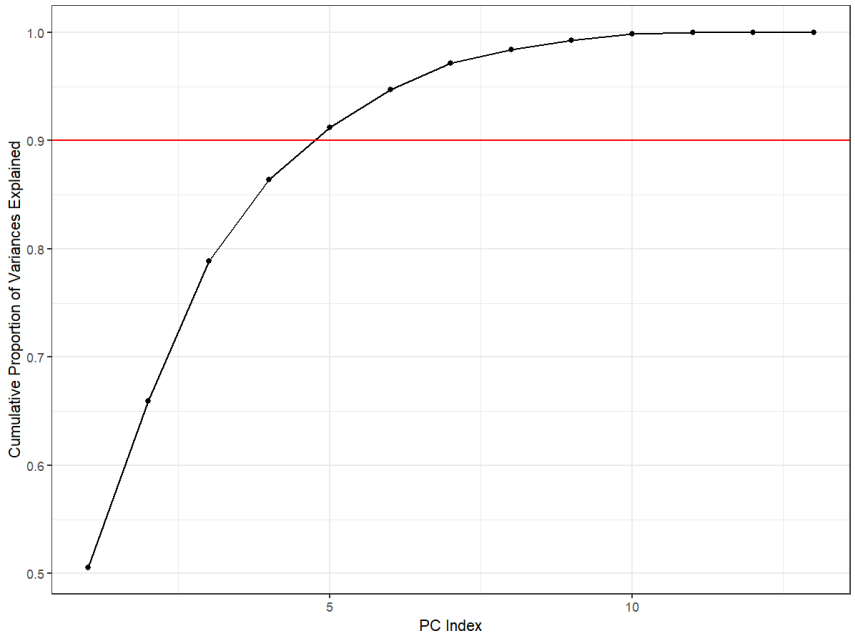


Figure 2: Cumulative Proportion of Variance Explained by Principal Components
(Red line is the threshold where 90% variances are explained; the first 5 PCs capture over 90% variances.)

K-means Clustering: By examining the *silhouette scores*, we find the optimal number of clusters for K-means is 4, but this result is *not* well-supported by the elbow plot in which there is an obvious decrease in *WCSS* explained by adding number of clusters after 3 (*Figure 3*). Because of the difference from two plots, we decide to plot the silhouette coefficients for outcomes from *K-means* with both 3 and 4 clusters.

The horizontal dashed lines in *Figure 4* represent the mean of the *silhouette coefficient* for all observations. We can see there is one cluster under this line in both $k = 3$ and $k = 4$, which means its members are too much close to other clusters. What's worse, we can also see some observations have negative *silhouette coefficients*, which indicates that they are allocated to wrong clustering. Overall, the performance of *K-means* clustering on our data is not satisfying.

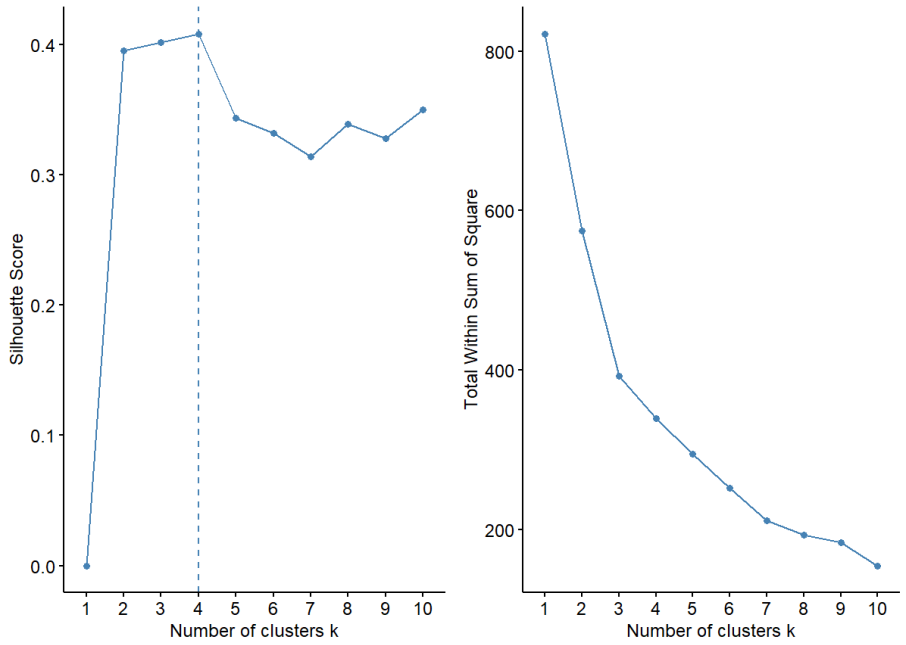


Figure 3: Optimal Number of Clusters for *K-means Clustering* with Different *k*
(The dashed line represents the optimal number of clusters; the slope in the elbow (right) plot has obvious reduction after $k = 3$; silhouette score(left) suggests 4 clusters, but elbow suggests 3 clusters.)

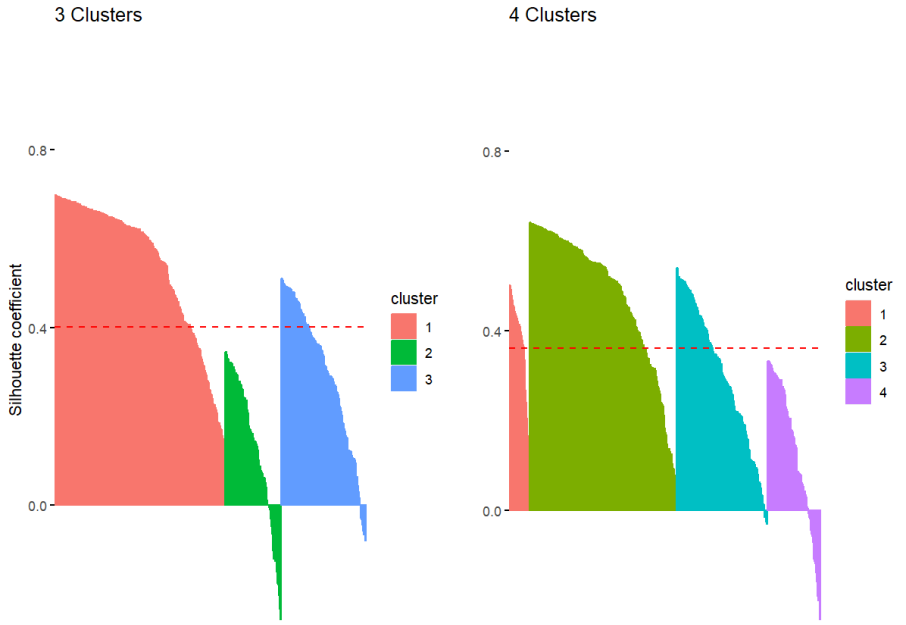


Figure 4: Distribution of Silhouette Coefficients for *K-means Clustering*
(The dashed lines represent average *silhouette coefficient*; both plots suggest bad performance of *K-means* clustering as there are negative values.)

Hierarchical Clustering: Again, we use *silhouette score* to choose the optimal number of clusters for hierarchical clustering with different linkages. We can see that the best numbers suggest by *Figure 5* are 2 for *complete*, *single* and *average* linkage, and 3 for *Ward's* linkage. Then we draw and cut their dendrograms to separate clusters corresponding to their optimal number of clusters, and discover that only the clustering using *Ward's* linkage has good performance (*Figure 6*). More importantly, its *dendrogram* suggests three clusters as well, and this can be explained by the height of the links, which

are noticeably high when there are 2 and 3 clusters. Therefore, considering both *silhouette score* and *dendrogram*, the number of clusters is decided to be 3.

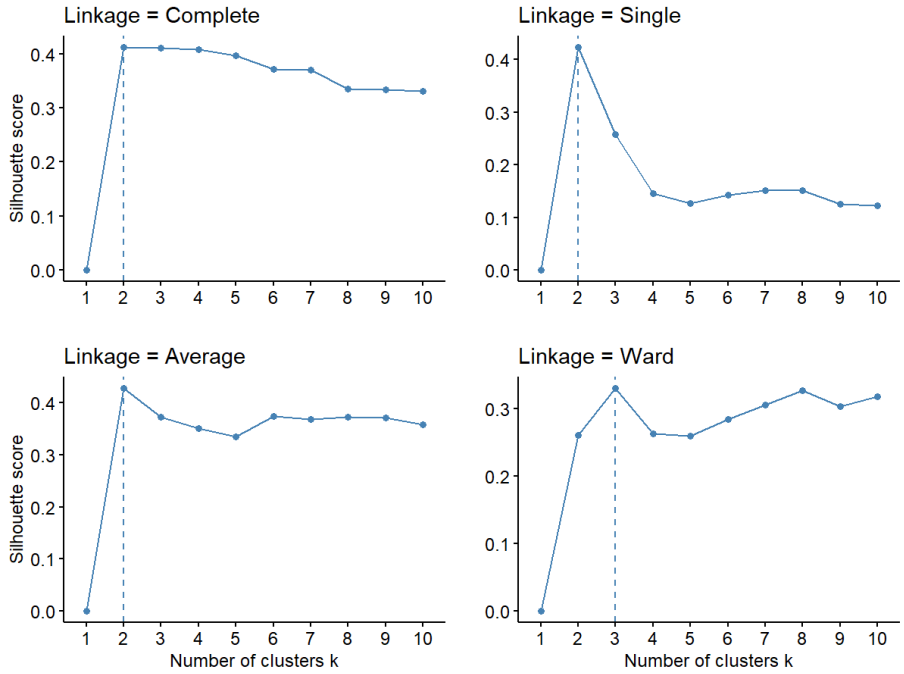


Figure 5: Optimal Number of Clusters for *Hierarchical Clustering* with Different Linkages
(The dashed lines represent optimal number of clusters; *hierarchical clusterings* with linkage of *complete*, *single* and *average* suggest 2 clusters, and *Ward's* linkage suggests 3 clusters)

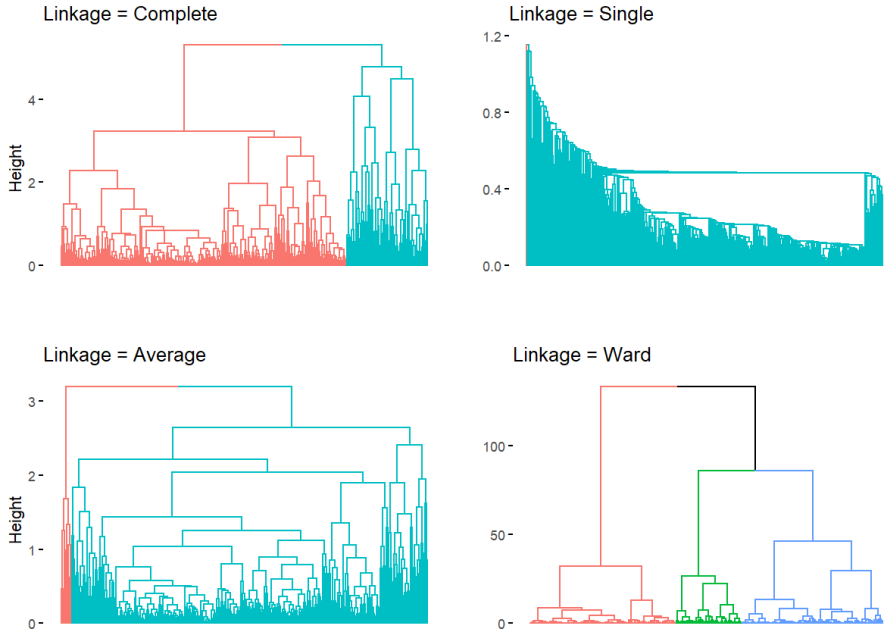


Figure 6: Dendrograms for Hierarchical Clustering with Different Linkages
(Different colour represents different cluster; only the Ward plot (bottom right) demonstrates acceptable clustering)

Now our best clustering result is from Hierarchical clustering with *Ward's* linkage, which divides the *Facebook Metrics Data* into 3 groups. Once the number of clusters has been decided, we can separate observations into three groups based on their membership.

We can see from *Table 1* that the means of the variables have proven that the second cluster has better performance than the other two. Perhaps we can recognize it as high-performance group. However, they also have larger variances, except for *Engaged.Users* and *Post.Consumptions*, which are not significantly larger than that of one of the other two. The two variables can be the key to the successful posts. Despite the fact that both of them are data after the post, which might not be useful for decision-making before the post, we can still use them as indicators for classification.

Table 1: Mean and Standard Deviation of Variables by Membership

Membership	1	2	3
Page.total.likes	135810 (2717)	129899 (8784)	112434 (12272)
Total.Reach	4303 (4071)	21328 (16061)	6429 (5806)
Post.Total.Impressions	8446 (9549)	38681 (33985)	10598 (8555)
Engaged.Users	375 (153)	1137 (378)	719 (344)
Post.Consumers	300 (124)	976 (427)	671 (335)
Post.Consumptions	470 (284)	1479 (631)	1151 (1011)
Post.Impressions.liked	5130 (4728)	18489 (9417)	7016 (4917)
Post.reach.liked	2618 (2187)	9611 (4327)	3932 (2308)
Liked.and.engaged	280 (107)	838 (375)	438 (146)
Comment	4 (6)	9 (10)	3 (4)
Like	102 (77)	224 (140)	89 (55)
Share	18 (13)	33 (21)	19 (13)
Total.Interactions	124 (90)	226 (157)	111 (68)

Note: the upper numbers are means; lower numbers with brackets are standard deviations.

Discussion

During data preprocessing, we discard categorical and discrete data, such as *Type* and *Category*. However, abandoning these features can be very costly, potentially resulting in the loss of crucial information that could significantly impact the outcomes we seek to achieve. The *PCA* works well for dimensional reduction, but some theoretical results show that the first few PCs may not contain more cluster information than other PCs (*Chang, 1983*). And PCs are not very useful for finding patterns in our case. Therefore, we show concerns about the outputs from hierarchical clustering that they might be less useful as it takes PCs as inputs, and we may need methods for diagnosis and validation. Another challenge in our analysis is the outliers, since we don't know if the outliers contain any valuable information. It is common on the social media that a post can lead to a phenomenon somehow, and we may not be able to catch such signals if we remove all outliers.

Conclusion

We have clustering results from *hierarchical clustering* with *Ward's* linkage. Three clusters are identified, and the second cluster can be labelled as high-performance data, which has larger mean in each variable involved in analysis. Besides, we also find that *Engaged.Users* and *Post.Consumptions* are potential key performance variables. However, we cannot visualize our clustering results in low dimension, so the trend and patterns of data distribution are still unclear to us. Most importantly, doubt and further justification should be placed on the robustness of our results to ensure their reliability both in this case and under other various conditions and analyses.

Reference

Alsayat, A. and El-Sayed, H., 2016, June. Social media analysis using optimized K-Means clustering. In *2016 IEEE 14th international conference on software engineering research, management and applications (SERA)* (pp. 61-66). IEEE.

Chang, W.C., 1983. On using principal components before separating a mixture of two multivariate normal distributions. Journal of the Royal Statistical Society: Series C (Applied Statistics), 32(3), pp.267-275.

Ding, C. and He, X., 2004, July. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning* (p. 29).

Géron, A., 2022. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc.".

Gupta, G.K., 2014. Introduction to data mining with case studies. PHI Learning Pvt. Ltd..

Ling, K., Beenen, G., Ludford, P., Wang, X., Chang, K., Li, X., Cosley, D., Frankowski, D., Terveen, L., Rashid, A.M. and Resnick, P., 2005. Using social psychology to motivate contributions to online communities. *Journal of Computer-Mediated Communication*, *10*(4), pp.00-00.

Moro,Srgio, Rita,Paulo, and Vala,Bernardo. 2016. Facebook Metrics. UCI Machine Learning Repository.

Xu, R. and Wunsch, D., 2005. Survey of clustering algorithms. IEEE Transactions on neural networks, 16(3), pp.645-678.

Yeung, K.Y. and Ruzzo, W.L., 2001. Principal component analysis for clustering gene expression data. Bioinformatics, 17(9), pp.763-774.

Yim, O. and Ramdeen, K.T., 2015. Hierarchical cluster analysis: comparison of three linkage measures and application to psychological data. The quantitative methods for psychology, 11(1), pp.8-21.

Appendix

Code

Ziteng 220009875 Group 1

2024-04-01

```
# Load packages
library(cluster)    # for clustering
library(corrplot)   # for correlation plot
library(tidyverse)  # for general plotting and data processing
library(factoextra) # for visualizing PCA
library(gridExtra)  # for arrangement of plots

# Read in data
df <- read.csv("dataset_Facebook.csv", sep = ";")

head(df)
```

```
##   Page.total.likes   Type Category Post.Month Post.Weekday Post.Hour Pa
id
## 1           139441  Photo        2         12            4         3
 0
## 2           139441 Status        2         12            3        10
 0
## 3           139441  Photo        3         12            3         3
 0
## 4           139441  Photo        2         12            2        10
 1
## 5           139441  Photo        2         12            2         3
 0
## 6           139441 Status        2         12            1         9
 0
##   Lifetime.Post.Total.Reach Lifetime.Post.Total.Impressions
## 1                      2752                            5091
## 2                     10460                           19057
## 3                      2413                            4373
## 4                     50128                           87991
## 5                      7244                           13594
## 6                     10472                           20849
##   Lifetime.Engaged.Users Lifetime.Post.Consumers Lifetime.Post.Consumpt
ions
## 1                    178                     109
 159
## 2                   1457                    1361
1674
## 3                    177                     113
 154
## 4                   2211                     790
1119
## 5                    671                     410
 580
```

```
## 6                    1191                      1073
1389
##   Lifetime.Post.Impressions.by.people.who.have.liked.your.Page
## 1                                                         3078
## 2                                                        11710
## 3                                                         2812
## 4                                                        61027
## 5                                                         6228
## 6                                                        16034
##   Lifetime.Post.reach.by.people.who.like.your.Page
## 1                                             1640
## 2                                             6112
## 3                                             1503
## 4                                            32048
## 5                                             3200
## 6                                             7852
##   Lifetime.People.who.have.liked.your.Page.and.engaged.with.your.post c
omment
## 1                                                                   119
     4
## 2                                                                  1108
     5
## 3                                                                   132
     0
## 4                                                                  1386
    58
## 5                                                                   396
    19
## 6                                                                  1016
     1
##   like share Total.Interactions
## 1   79    17                100
## 2  130    29                164
## 3   66    14                 80
## 4 1572   147               1777
## 5  325    49                393
## 6  152    33                186
```

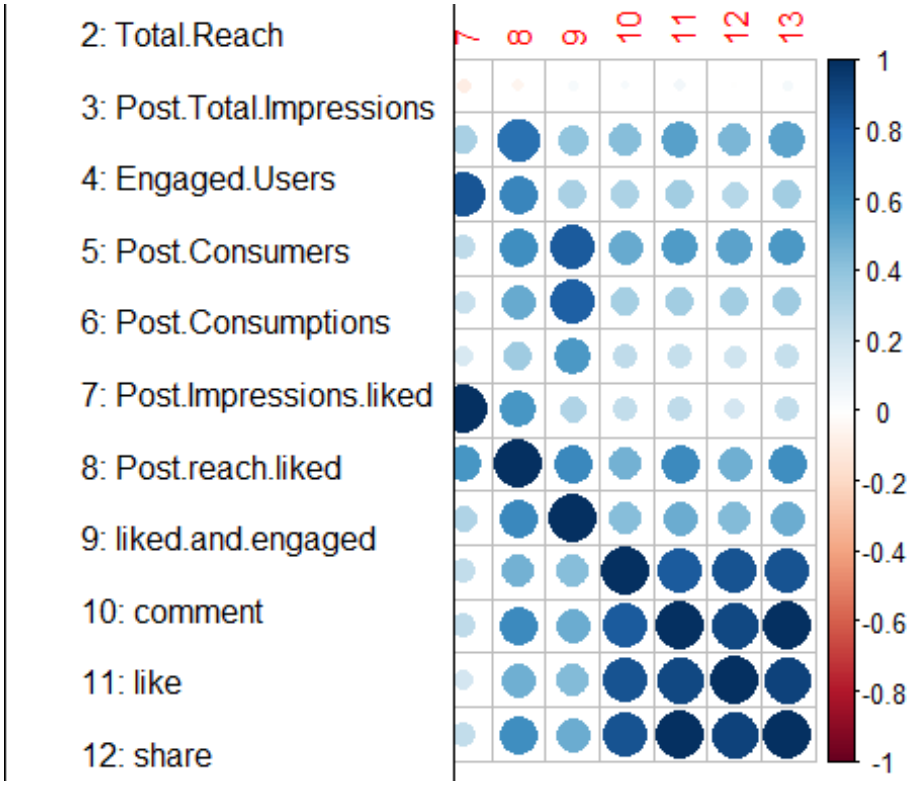
```
# Simplify column names
names(df)[names(df) == "Lifetime.Post.Total.Reach"] <- "Total.Reach"
names(df)[names(df) == "Lifetime.Post.Total.Impressions"] <- "Post.Total.I
mpressions"
names(df)[names(df) == "Lifetime.Engaged.Users"] <- "Engaged.Users"
names(df)[names(df) == "Lifetime.Post.Consumers"] <- "Post.Consumers"
names(df)[names(df) == "Lifetime.Post.Consumptions"] <- "Post.Consumptions
"
names(df)[names(df) == "Lifetime.Post.Impressions.by.people.who.have.like
d.your.Page"] <- "Post.Impressions.liked"
names(df)[names(df) == "Lifetime.Post.reach.by.people.who.like.your.Page"]
 <- "Post.reach.liked"
names(df)[names(df) == "Lifetime.People.who.have.liked.your.Page.and.engag
ed.with.your.post"] <- "liked.and.engaged"

names(df)
```

```
##  [1] "Page.total.likes"       "Type"                   "Category"

##  [4] "Post.Month"             "Post.Weekday"           "Post.Hour"

##  [7] "Paid"                   "Total.Reach"            "Post.Total.Impr
essions"
## [10] "Engaged.Users"          "Post.Consumers"         "Post.Consumptio
ns"
## [13] "Post.Impressions.liked" "Post.reach.liked"       "liked.and.engag
ed"
## [16] "comment"                "like"                   "share"

## [19] "Total.Interactions"
```

Data exploration and pre-process

```
# Drop rows with missing values
df <- na.omit(df)

# Check the values in "Type" column
unique(df$Type)

## [1] "Photo"  "Status" "Link"   "Video"

unique(df$Category)

## [1] 2 3 1

# Create new object to select continuous data
df_num <- df %>% select(-Type,                # categorical
                        -Category,            # discrete
                        -Post.Month,          # discrete
                        -Post.Weekday,        # discrete
                        -Post.Hour,           # discrete
                        -Paid)                # binary

# Copy data to aovid modification
df_num_copy <- df_num

# Change names of columns to numbers
colnames(df_num_copy) <- seq(from = 1, to = length(df_num), by = 1)

# Correlation matrix
df_num_corr <- cor(df_num_copy)

# Set legend
leg_text <- c("1: Page.total.likes",
              "2: Total.Reach",
              "3: Post.Total.Impressions",
              "4: Engaged.Users",
              "5: Post.Consumers",
              "6: Post.Consumptions",
              "7: Post.Impressions.liked",
              "8: Post.reach.liked",
              "9: liked.and.engaged",
```

```
              "10: comment",
              "11: like",
              "12: share",
              "13: Total.Interactions")

# Draw correlation plot
corrplot(df_num_corr)
legend("left", leg_text)
```




```
# Statistical summary
summary(df_num)
```

```
##  Page.total.likes  Total.Reach     Post.Total.Impressions Engaged.Users

##  Min.   : 81370   Min.   :   238   Min.   :    570        Min.   :
9.0
##  1st Qu.:112324   1st Qu.:  3331   1st Qu.:   5798        1st Qu.:  39
9.0
##  Median :129600   Median :  5290   Median :   9084        Median :  63
0.0
##  Mean   :123173   Mean   : 14028   Mean   :  29857        Mean   :  92
6.8
##  3rd Qu.:136393   3rd Qu.: 13248   3rd Qu.:  22503        3rd Qu.: 106
2.0
##  Max.   :139441   Max.   :180480   Max.   :1110282        Max.   :1145
2.0
##  Post.Consumers    Post.Consumptions Post.Impressions.liked Post.reach.
liked
##  Min.   :    9.0   Min.   :    9.0   Min.   :    567        Min.   :  2
36
##  1st Qu.:  335.0   1st Qu.:  512.5   1st Qu.:   4074        1st Qu.: 22
```

```
13
##  Median :  555.0   Median :  861.0   Median :   6282        Median : 34
78
##  Mean   :  804.2   Mean   : 1425.9   Mean   :  16916        Mean   : 66
41
##  3rd Qu.:  969.0   3rd Qu.: 1479.0   3rd Qu.:  15143        3rd Qu.: 80
18
##  Max.   :11328.0   Max.   :19779.0   Max.   :1107833        Max.   :514
56
##  liked.and.engaged    comment             like            share
##  Min.   :   9.0    Min.   :  0.000   Min.   :   0.0   Min.   :  0.00
##  1st Qu.: 297.5    1st Qu.:  1.000   1st Qu.:  57.0   1st Qu.: 10.00
##  Median : 416.0    Median :  3.000   Median : 101.0   Median : 19.00
##  Mean   : 614.1    Mean   :  7.558   Mean   : 179.1   Mean   : 27.26
##  3rd Qu.: 658.5    3rd Qu.:  7.000   3rd Qu.: 188.0   3rd Qu.: 32.50
##  Max.   :4376.0    Max.   :372.000   Max.   :5172.0   Max.   :790.00
##  Total.Interactions
##  Min.   :   0
##  1st Qu.:  72
##  Median : 125
##  Mean   : 214
##  3rd Qu.: 231
##  Max.   :6334
```

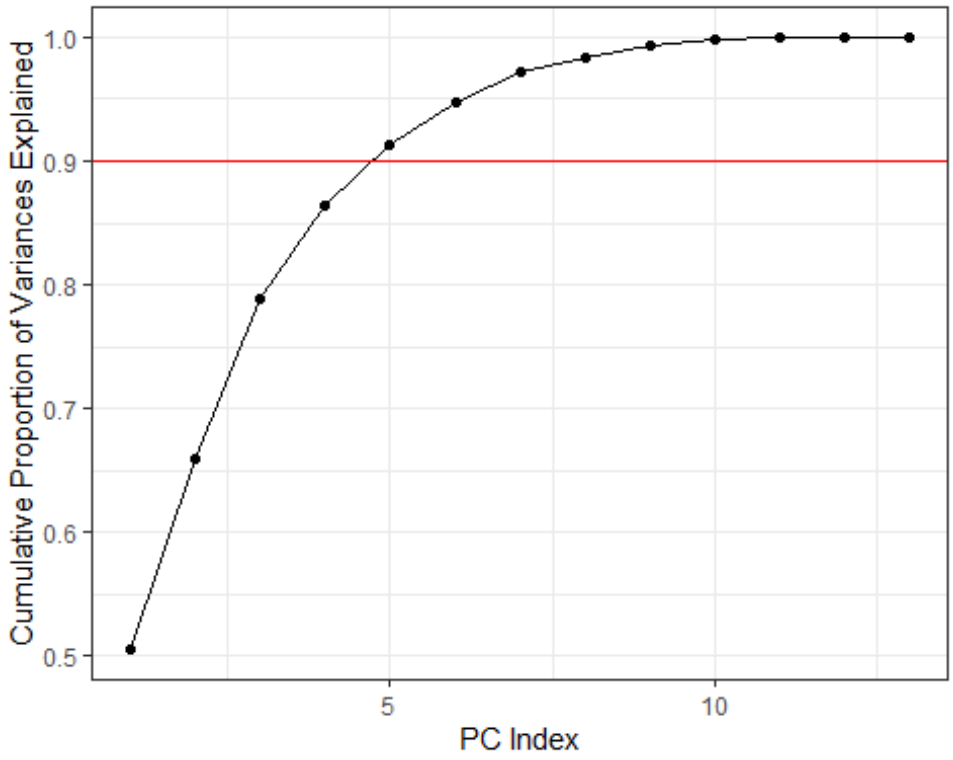
Principal Component Analysis

```
# Assign new object
df <- df_num

# Conduct PCA
pca <- prcomp(df, center = TRUE, scale. = TRUE)

# Scale data
df_scale <- scale(df, center = TRUE, scale = TRUE)

# Correlation matrix
corr_df <- cor(df_scale)

# Eigenpairs
eig <- eigen(corr_df)

# Calculations
pc_var <- eig$values
pc_prop_var <- pc_var / sum(pc_var)
pc_cum_prop_var <- cumsum(pc_prop_var)

# Create data frame to store values
pc_var_df <- data.frame("Variance" = pc_var,
                        "Proportion" = pc_prop_var,
                        "Cumulative" = pc_cum_prop_var)
pc_var_df
```

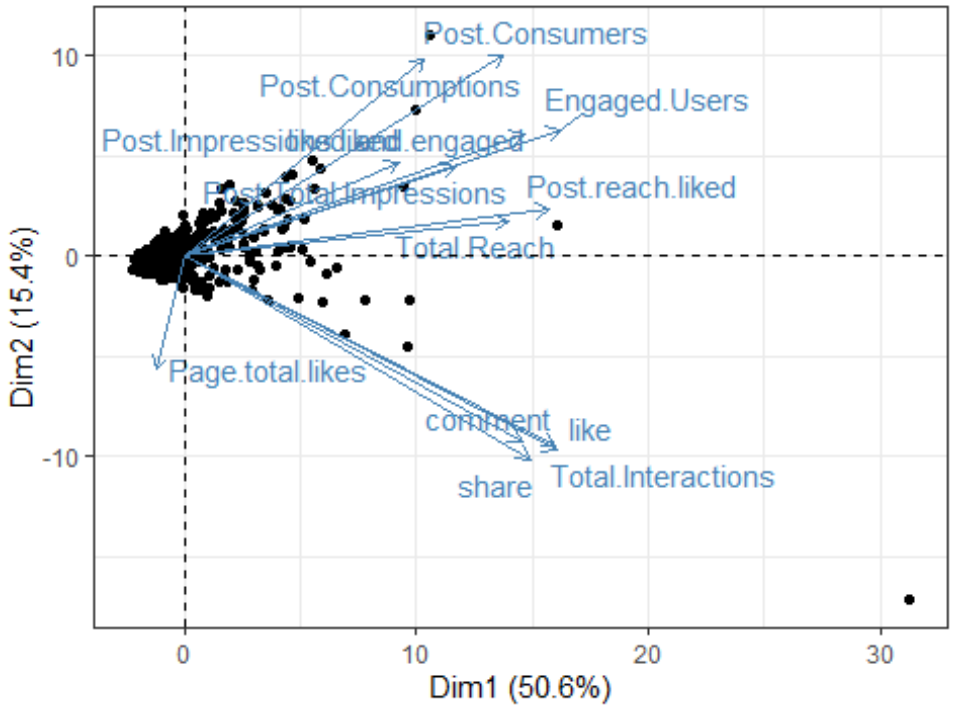
```
##         Variance    Proportion Cumulative
## 1   6.574561e+00  5.057355e-01  0.5057355
```

```
## 2   1.996740e+00  1.535954e-01  0.6593308
## 3   1.682108e+00  1.293929e-01  0.7887237
## 4   9.811161e-01  7.547047e-02  0.8641942
## 5   6.262434e-01  4.817257e-02  0.9123668
## 6   4.538301e-01  3.491001e-02  0.9472768
## 7   3.134590e-01  2.411223e-02  0.9713890
## 8   1.610745e-01  1.239034e-02  0.9837794
## 9   1.178409e-01  9.064687e-03  0.9928440
## 10  7.882310e-02  6.063315e-03  0.9989074
## 11  1.365994e-02  1.050765e-03  0.9999581
## 12  5.444125e-04  4.187789e-05  1.0000000
## 13 -1.743397e-16 -1.341075e-17  1.0000000
```

```r
# Plot cumulative proportion
ggplot(data = pc_var_df) +
  geom_line(aes(x = seq(1, length(Variance), 1), y = Cumulative)) +
  geom_point(aes(x = seq(1, length(Variance), 1), y = Cumulative)) +
  geom_hline(yintercept = 0.9, color = "red") + theme_bw() +
  xlab("PC Index") + ylab("Cumulative Proportion of Variances Explained")
```




```r
# Draw a biplot for first two PCs
fviz_pca_biplot(pca, label = "var", repel = TRUE, title = "") + theme_bw()
```

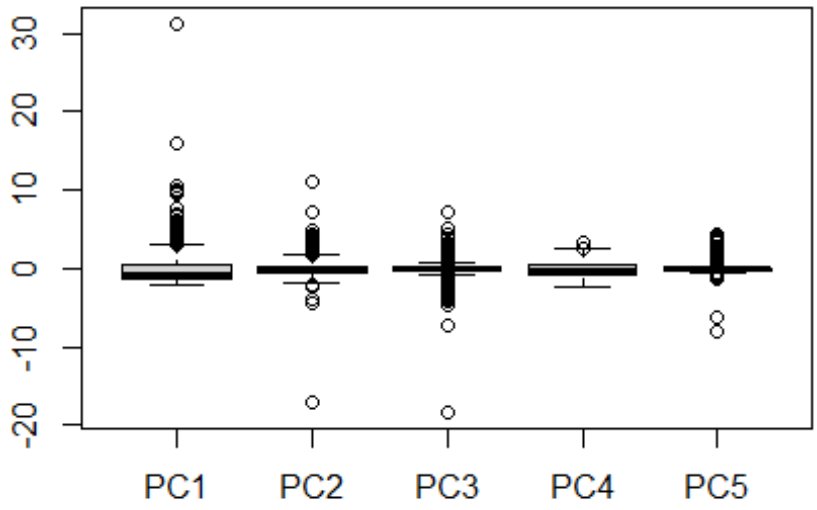



```
# Extract first 5 PCs
df_pca <- as.data.frame(pca$x[ , 1:5])

head(df_pca)
```

```
##          PC1        PC2        PC3        PC4          PC5
## 1 -1.6656662 -0.9071976 -0.2172729 -0.7560745 -0.037936869
## 2  0.3642079  0.4474522  0.7618548 -1.1429258 -0.145251714
## 3 -1.7797749 -0.7825328 -0.2208223 -0.7891346 -0.005288325
## 4  6.8931573 -3.8977473 -0.4940286 -0.6204243  0.435410776
## 5 -0.1276758 -1.4956877  0.1768322 -0.6504256 -0.217494017
## 6  0.2140330  0.1990814  0.4200946 -1.1301673 -0.073307661
```

Remove outliers

```
# Draw a box plot of first five PCs
boxplot(df_pca)
```

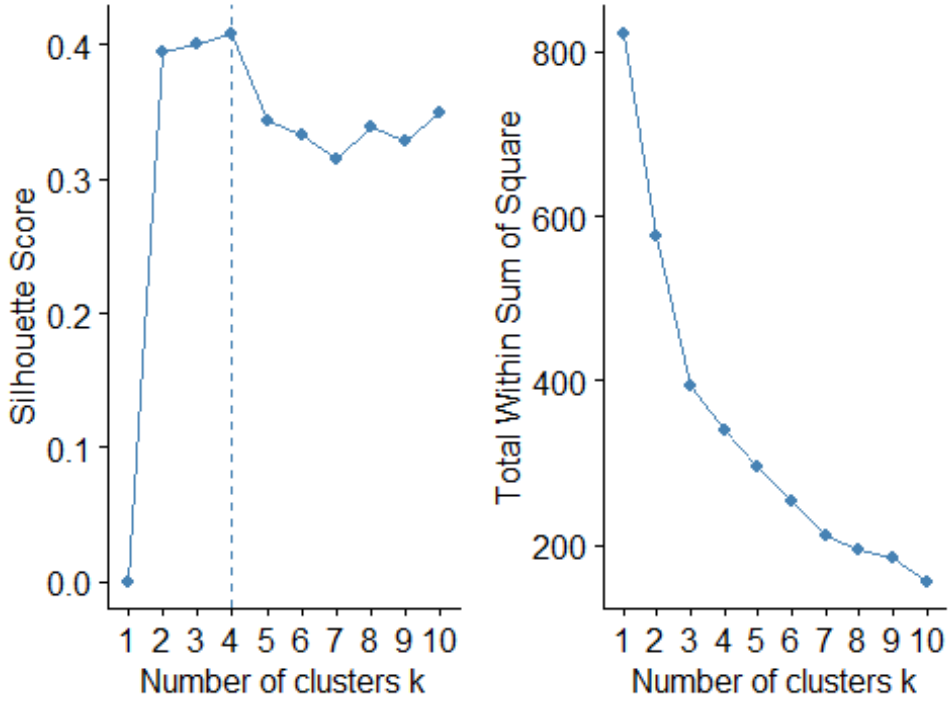



```
# Create unique values for each row
df_pca$ID <- rownames(df_pca)
df$ID <- rownames(df)

# Remove outliers in df_pca
df_pca <- subset(df_pca, PC1 <= 2 & PC1 >= -2 &
                         PC2 <= 2 & PC2 >= -2 &
                         PC3 <= 2 & PC3 >= -2 &
                         PC4 <= 2 & PC4 >= -2 &
                         PC5 <= 2 & PC5 >= -2)

# Keep obs in df same with df_pca
df <- semi_join(df, df_pca, "ID")

# Remove ID from df_pca
df_pca <- select(df_pca, -ID)
```

K-means Clustering

```
# Select number of clusters
m_sil <- fviz_nbclust(x = df_pca,
             FUNcluster = kmeans,
             method = "silhouette",
             k.max = 10) +
         ggtitle("") +
         ylab("Silhouette Score")

m_wss <- fviz_nbclust(x = df_pca,
             FUNcluster = kmeans,
             method = "wss",
```

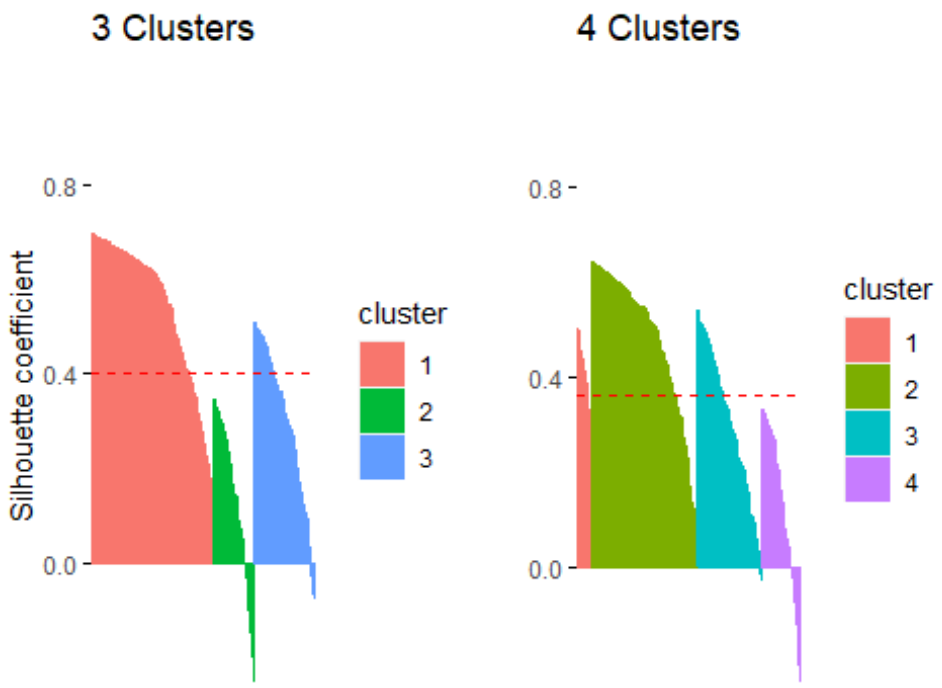
```
            k.max = 10) +
          ggtitle("")

grid.arrange(m_sil, m_wss, ncol = 2)
```




```
# K-means with 4 clusters
km3 <- kmeans(df_pca, centers = 3, nstart = 50)
km4 <- kmeans(df_pca, centers = 4, nstart = 50)

# Create silhouette object
sil3 <- silhouette(x = km3$cluster, dist = dist(df_pca))
sil4 <- silhouette(x = km4$cluster, dist = dist(df_pca))

# Silhouette plot
sil_plot3 <- fviz_silhouette(sil3) +
             ylab("Silhouette coefficient") +
             ggtitle("3 Clusters")

##   cluster size ave.sil.width
## 1       1  214          0.54
## 2       2   70          0.13
## 3       3  106          0.30

sil_plot4 <- fviz_silhouette(sil4) +
             ylab("") +
             ggtitle("4 Clusters")

##   cluster size ave.sil.width
## 1       1   26          0.37
## 2       2  184          0.48
```

```
## 3       3  114          0.30
## 4       4   66          0.13
```

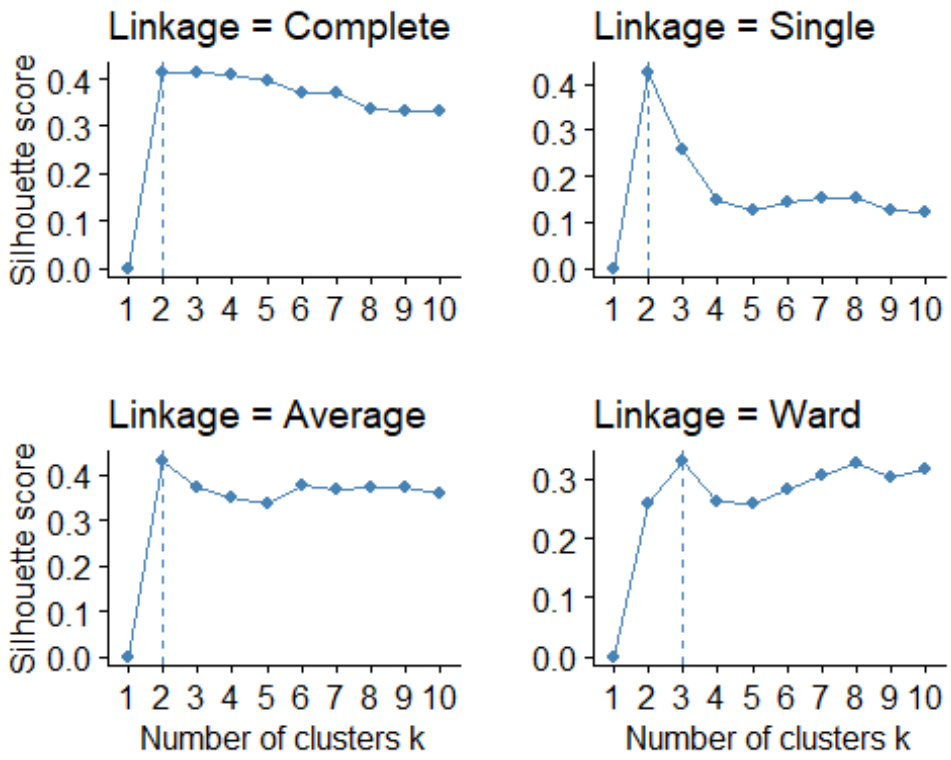
```
grid.arrange(sil_plot3, sil_plot4, ncol = 2)
```



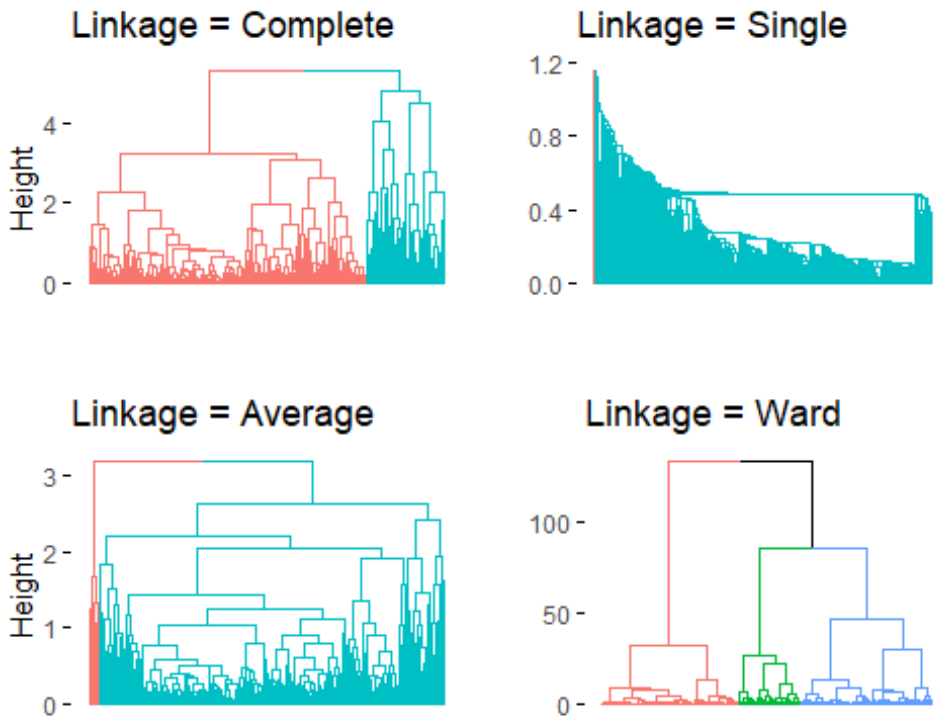
Hierachical clustering

```
# Distance matrix
D <- dist(df_pca)

# Hierachical clustering
hc1 <- hclust(D, method = "complete") # apply complete linkage
hc2 <- hclust(D, method = "single")   # apply single linkage
hc3 <- hclust(D, method = "average")  # apply average linkage
hc4 <- hclust(D, method = "ward.D")   # apply ward linkage

# Optimal number of clusters

# Function for cluster labels
hclust_custom1 <- function(x, k = 4) {
    hc <- hclust(dist(x), method = "complete")
    clust <- cutree(hc, k = k)
    return(list(cluster = clust))
}

hclust_custom2 <- function(x, k = 4) {
    hc <- hclust(dist(x), method = "single")
    clust <- cutree(hc, k = k)
    return(list(cluster = clust))
}
```

```
hclust_custom3 <- function(x, k = 4) {
    hc <- hclust(dist(x), method = "average")
    clust <- cutree(hc, k = k)
    return(list(cluster = clust))
}

hclust_custom4 <- function(x, k = 4) {
    hc <- hclust(dist(x), method = "ward.D")
    clust <- cutree(hc, k = k)
    return(list(cluster = clust))
}

# Plot optimal number using silhouette score
op_cls1 <- fviz_nbclust(x = df_pca, FUNcluster = hclust_custom1,
                        method = "silhouette", k.max = 10) +
           ggtitle("Linkage = Complete") + xlab("") + ylab("Silhouette sco
re")

op_cls2 <- fviz_nbclust(x = df_pca, FUNcluster = hclust_custom2,
                        method = "silhouette", k.max = 10) +
             ggtitle("Linkage = Single") + xlab("") + ylab("")

op_cls3 <- fviz_nbclust(x = df_pca, FUNcluster = hclust_custom3,
             method = "silhouette", k.max = 10) +
           ggtitle("Linkage = Average") + ylab("Silhouette score")

op_cls4 <- fviz_nbclust(x = df_pca, FUNcluster = hclust_custom4,
             method = "silhouette", k.max = 10) +
             ggtitle("Linkage = Ward") + ylab("")

grid.arrange(op_cls1, op_cls2, op_cls3, op_cls4, ncol = 2)
```

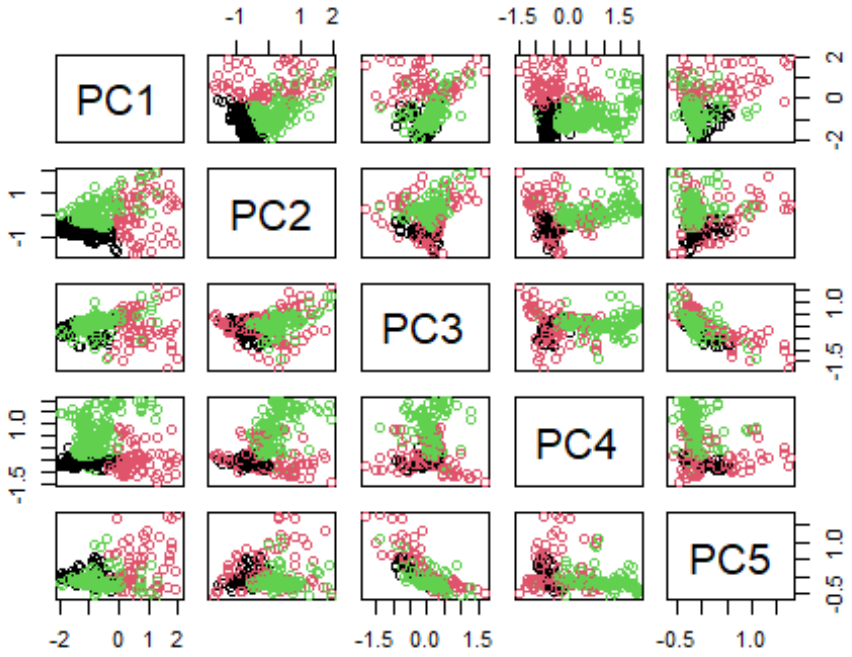



```
# Plot dendrogram
dend1 <- fviz_dend(hc1, show_labels = FALSE, k = 2) + xlab("") +
         ggtitle("Linkage = Complete")
dend2 <- fviz_dend(hc2, show_labels = FALSE, k = 2) +
         ggtitle("Linkage = Single") + xlab("") + ylab("")
dend3 <- fviz_dend(hc3, show_labels = FALSE, k = 2) +
         ggtitle("Linkage = Average")
dend4 <- fviz_dend(hc4, show_labels = FALSE, k = 3) +
         ggtitle("Linkage = Ward") + ylab("")

grid.arrange(dend1, dend2, dend3, dend4, ncol = 2)
```

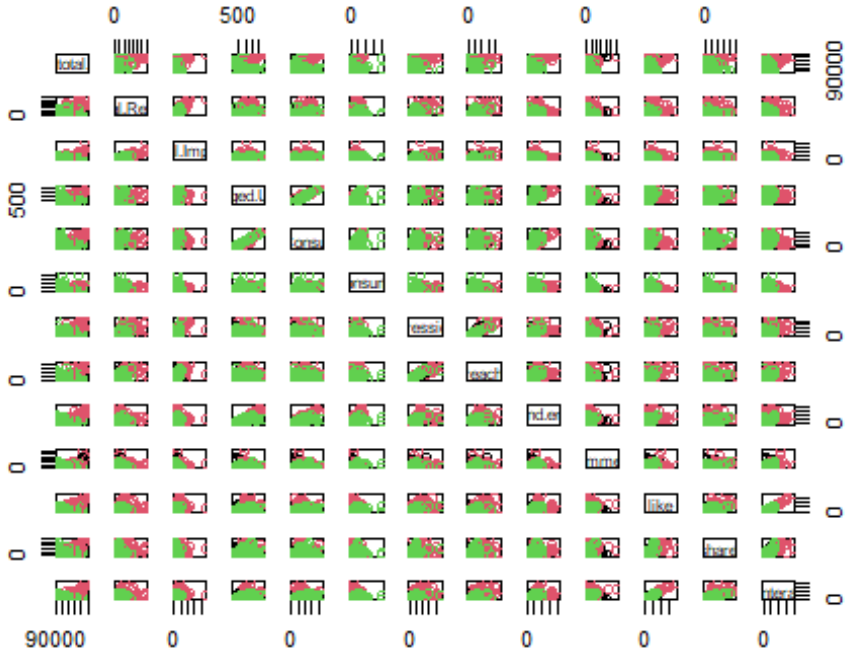


Result

```
# Remove ID from "df"
df <- select(df, -ID)

# Cut hc4 and return cluster labels
Membership <- cutree(hc4, k = 3)

# Pair plots for pca data by membership
pairs(df_pca, col=Membership)
```




```
# Pair plots for original data by membership
pairs(df, col = Membership)
```



```
# Output pdf
pdf("pairs.pdf")
pairs(df, cex=0.05, col=Membership)
dev.off()
```

```
## png
##   2

# Check the number of obs. in each cluster
table(Membership)

## Membership
##   1   2   3
## 161  73 156

# Compute mean and sd for each variable by membership
df$Membership <- Membership

med <- df %>%
  group_by(Membership) %>%
  summarise_all(~median(.))

mean <- df %>%
  group_by(Membership) %>%
  summarise_all(~mean(.)) %>%
  round(0)

sd <- df %>%
  group_by(Membership) %>%
  summarise_all(~sd(.)) %>%
  round(0)

t(rbind(mean, sd))

##                          [,1]   [,2]   [,3] [,4]  [,5]  [,6]
## Membership                  1      2      3    1     2     3
## Page.total.likes       135810 129899 112434 2717  8784 12272
## Total.Reach              4303  21328   6429 4071 16061  5806
## Post.Total.Impressions   8446  38681  10598 9549 33985  8555
## Engaged.Users             375   1137    719  153   378   344
## Post.Consumers            300    976    671  124   427   335
## Post.Consumptions         470   1479   1151  284   631  1011
## Post.Impressions.liked   5130  18489   7016 4728  9417  4917
## Post.reach.liked         2618   9611   3932 2187  4327  2308
## liked.and.engaged         280    838    438  107   375   146
## comment                     4      9      3    6    10     4
## like                      102    224     89   77   140    55
## share                      18     33     19   13    21    13
## Total.Interactions        124    266    111   90   157    68
```