Meghnath Reddy Challa- OUID (113508098)

IDA HOMEWORK-2

Marked in yellow=R CODE Output=Result when marked R code is executed

Required Packages:

1) Require (mlbench) Package to get glass dataset
2) Library (tidyverse) To install and load multiple tidyverse packages
3) Require (rgl) 3D visualization device package
4) Require (ggplot2)
5) Require (MASS) To load LDA
6) Require (Rtsne) FOR TSNE PACKAGE
7) Require (umap) FOR UMAP PACKAGE
8) Require(factoextra) #To draw bar plot of variable contribution

Problem 1) Glass Data

To remove the duplicated row:

```
unique_rows<-unique_rows[!duplicated(unique_rows),]
```

It finds and remove the duplicate row in this Glass data we have "1" duplicated

a) **Mathematics of PCA:**
   i) **correlational matrix**

```
my_num_data <- unique_rows[, sapply(unique_rows, is.numeric)] #remove non-numeric

corMat<-cor(my_num_data,use = "complete.obs") #To store matrix in corMat
```

```
> corMat
           RI          Na          Mg          Al          Si           K          Ca
RI  1.000000000 -0.19880214 -0.12752579 -0.40097326 -0.53899979 -0.28764513  0.8111829
Na -0.198802137  1.00000000 -0.27841998  0.16773502 -0.06488462 -0.26415757 -0.2781937
Mg -0.127525799 -0.27841998  1.00000000 -0.47957521 -0.16243673  0.00761676 -0.4461971
Al -0.400973259  0.16773502 -0.47957521  1.00000000 -0.01619465  0.32368277 -0.2580676
Si -0.538999788 -0.06488462 -0.16243673 -0.01619465  1.00000000 -0.19728091 -0.2071445
K  -0.287645126 -0.26415757  0.00761676  0.32368278 -0.19728091  1.00000000 -0.3170324
Ca  0.811182947 -0.27819366 -0.44619711 -0.25806764 -0.20714455 -0.31703243  1.0000000
Ba  0.001679071  0.32907979 -0.49181779  0.48064237 -0.10438937 -0.04365259 -0.1122076
Fe  0.147083050 -0.23937377  0.08542584 -0.08058344 -0.09771680 -0.00937223  0.1263143
           Ba          Fe
RI  0.001679071  0.147083050
Na  0.329079794 -0.239373767
Mg -0.491817790  0.085425844
Al  0.480642369 -0.080583442
Si -0.104389371 -0.097716800
K  -0.043652592 -0.009372226
Ca -0.112207559  0.126314271
Ba  1.000000000 -0.059729016
Fe -0.059729016  1.000000000
>
```

ii) Eigen Vector and Eigen Values:

eigen_vector<-eigen(corMat) #to get vector values

```
$vectors
           [,1]        [,2]        [,3]       [,4]        [,5]        [,6]        [,7]
 [1,]  0.5432231 -0.28911804 -0.08849541  0.1479796  0.07670808 -0.11455615 -0.08223530
 [2,] -0.2676141 -0.26909913  0.36710090  0.5010669 -0.14626769  0.55790564 -0.15419352
 [3,]  0.1093261  0.59215502 -0.02295318  0.3842440 -0.11610001 -0.30585293  0.20691746
 [4,] -0.4269512 -0.29636272 -0.32602906 -0.1488756 -0.01720068  0.02014091  0.69982052
 [5,] -0.2239232  0.15874450  0.47979931 -0.6394962 -0.01763694 -0.08850787 -0.20945417
 [6,] -0.2156587  0.15305116 -0.66349177 -0.0733491  0.30154622  0.24107648 -0.50515516
 [7,]  0.4924367 -0.34678973  0.01380151 -0.2743430  0.18431431  0.14957911  0.09984144
 [8,] -0.2516459 -0.48262056 -0.07649040  0.1299431 -0.24970936 -0.65986429 -0.35043794
 [9,]  0.1912640  0.06089167 -0.27223834 -0.2252596 -0.87828176  0.24066617 -0.07120579
            [,8]        [,9]
 [1,]  0.75177166 -0.02568051
 [2,]  0.12819398  0.31188932
 [3,]  0.07799332  0.57732740
 [4,]  0.27334224  0.19041178
 [5,]  0.38077660  0.29747147
 [6,]  0.11064442  0.26075531
 [7,] -0.39885229  0.57999243
 [8,] -0.14497643  0.19853265
 [9,]  0.01650505  0.01459278
```

eigen_values<-eigen_vector$values #to store eigen values

```
[1] 2.510152168 2.058169337 1.407484057 1.144693344 0.914768873 0.528593040 0.370262639
[8] 0.064267543 0.001608997
```

iii) Use of prcomp for PCA:

comp<-prcomp(Glass[,c(1:9)],scale. = TRUE,center = TRUE)

Since PCA works better with numerical value we avoid Type attribute

```
Standard deviations (1, ..., p=9):
[1] 1.58466518 1.43180731 1.18526115 1.07604017 0.95603465 0.72638502 0.60741950
[8] 0.25269141 0.04011007

Rotation (n x k) = (9 x 9):
           PC1         PC2           PC3          PC4           PC5          PC6          PC7
RI -0.5451766  0.28568318 -0.0869108293 -0.14738099  0.073542700 -0.11528772 -0.08186724
Na  0.2581256  0.27035007  0.3849196197 -0.49124204 -0.153683304  0.55811757 -0.14858006
Mg -0.1108810 -0.59355826 -0.0084179590 -0.37878577 -0.123509124 -0.30818598  0.20604537
Al  0.4287086  0.29521154 -0.3292371183  0.13750592 -0.014108879  0.01885731  0.69923557
Si  0.2288364 -0.15509891  0.4587088382  0.65253771 -0.008500117 -0.08609797 -0.21606658
K   0.2193440 -0.15397013 -0.6625741197  0.03853544  0.307039842  0.24363237 -0.50412141
Ca -0.4923061  0.34537980  0.0009847321  0.27644322  0.188187742  0.14866937  0.09913463
Ba  0.2503751  0.48470218 -0.0740547309 -0.13317545 -0.251334261 -0.65721884 -0.35178255
Fe -0.1858415 -0.06203879 -0.2844505524  0.23049202 -0.873264047  0.24304431 -0.07372136
           PC8         PC9
RI -0.75221590 -0.02573194
Na -0.12769315  0.31193718
Mg -0.07689061  0.57727335
Al -0.27444105  0.19222686
Si -0.37992298  0.29807321
K  -0.10981168  0.26050863
Ca  0.39870468  0.57932321
Ba  0.14493235  0.19822820
Fe -0.01627141  0.01466944
```

```
> summary(comp)
Importance of components:
                          PC1    PC2    PC3    PC4    PC5     PC6    PC7     PC8     PC9
Standard deviation     1.585 1.4318 1.1853 1.0760 0.9560 0.72639 0.6074 0.25269 0.04011
Proportion of Variance 0.279 0.2278 0.1561 0.1286 0.1016 0.05863 0.0410 0.00709 0.00018
Cumulative Proportion  0.279 0.5068 0.6629 0.7915 0.8931 0.95173 0.9927 0.99982 1.00000
```

iv) Comparisons of results ii) and iii):

The results are different. Eigenvectors help us determining the direction and Eigenvalues help in finding out the magnitude that is the variance of the data along the new feature axis. Whereas PCA helps us in understanding how each feature is related to other by covariance matrix which makes it easy in deciding on which Eigen vector to drop which we feel is unimportant.
Ex: RI feature from PC1 has -0.545

v) PC1 and PC2 are orthogonal:

Two vectors are said to be perpendicular if their inner product is zero.

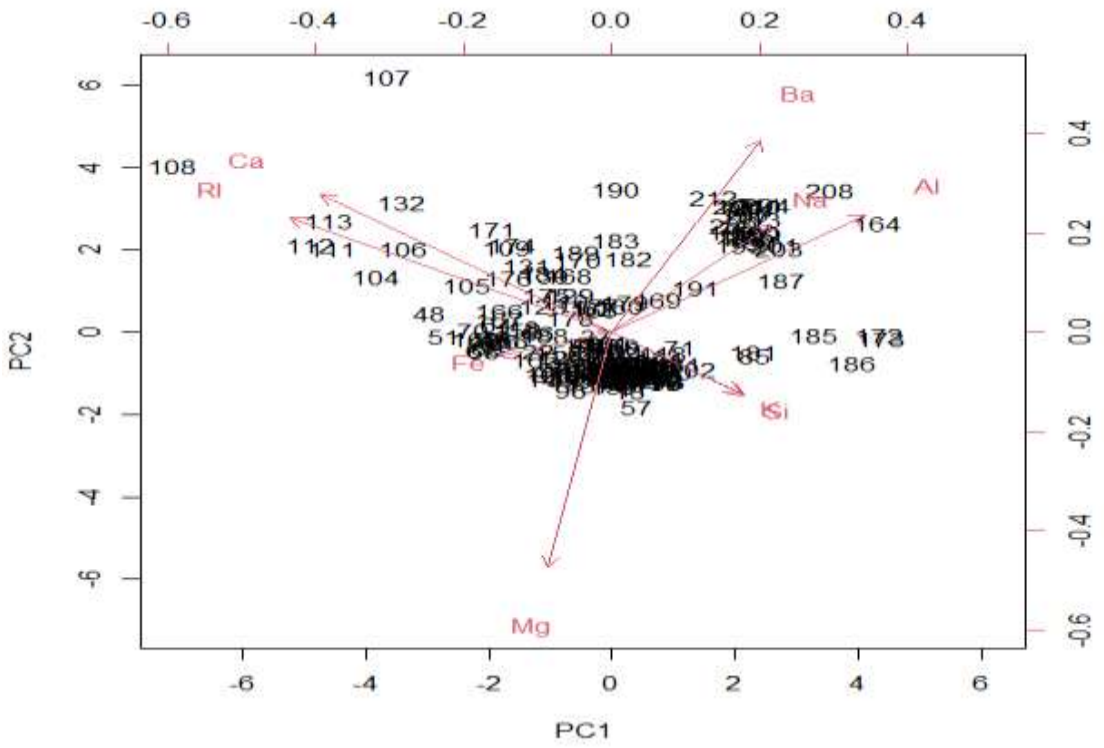
```
Z<-cor(comp$rotation)
Z<-data.frame(Z) #convert to data frame to perform calculations
w<-Z[(c("PC1"))] #to store PC1
x<-Z[c("PC2")] #to store PC2
sum((w)*x) #dot product
```

```
 sum((w)*x)
1] 0.0006000126
```

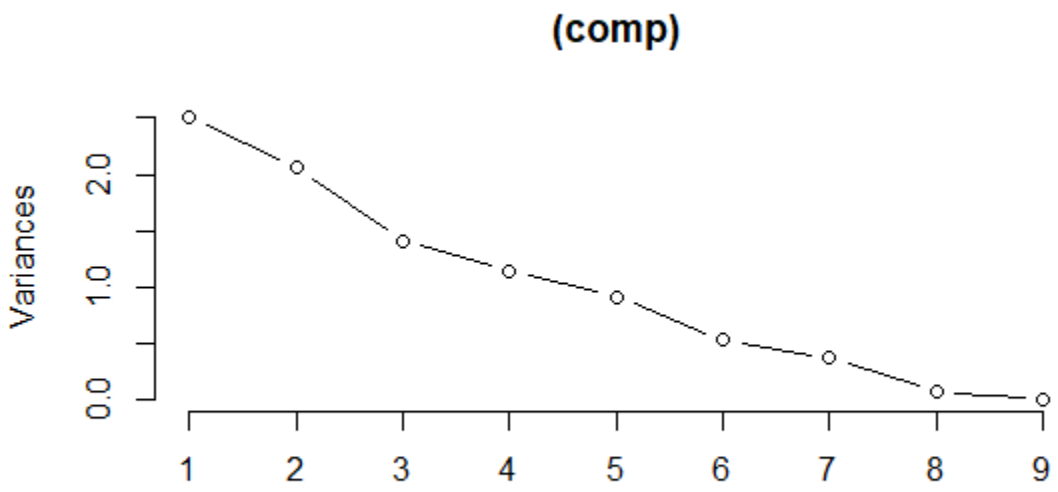
b) Application of PCA:

i) Biplot:

```
biplot(comp,group_by="Type",color="Type",scale = 0)
```



Insight: Red arrow indicates Eigenvector for each variable in the data frame. As the observation has higher value for PC1 it has lower value of Mg, Ca, Rl and higher Ba, Na and Al. Similarly, as the observation has higher value for PC2 it has lower value of Mg, slightly lower Fe and higher Ba.



plot((comp),type='l')

Insight: Most of the variability is accounted by the time we reach PC6.

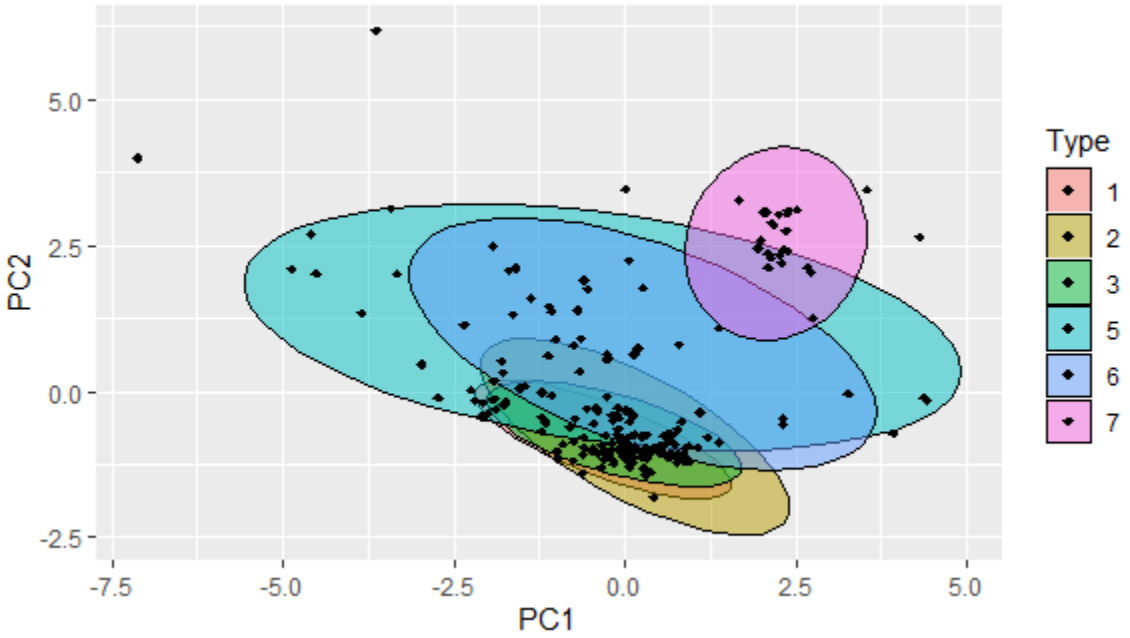
ii) Interpretation of PC1 and PC2:

str(comp) **To check all the objects stored in PCA**
glass_pca<-cbind(Glass,comp$x[,1:2]) **For the rows add PC1 and PC2**
head(glass_pca) **To check top rows**

	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type	PC1	PC2
1	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0	0.00	1	-1.1484468	-0.5282491
1	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0	0.00	1	0.5727942	-0.7580105
1	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0	0.00	1	0.9379605	-0.9276609
1	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0	0.00	1	0.1417509	-0.9594279
1	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0	0.00	1	0.3502710	-1.0886966
1	1.51596	12.79	3.61	1.62	72.97	0.64	8.07	0	0.26	1	0.2895876	-1.3209105

GGplot:

ggplot(glass_pca,aes(PC1,PC2,col=Type,fill=Type))+stat_ellipse(geom='polygon',col="black",alpha=0.5)+geom_point(shape=18,col="black")



Insight: We could not separate a lot between the Type of glass just from PC1 and PC2.

Correlation between VARS and PC1 and PC2:

correlation<-cor(Glass[,-10],glass_pca[,11:12])

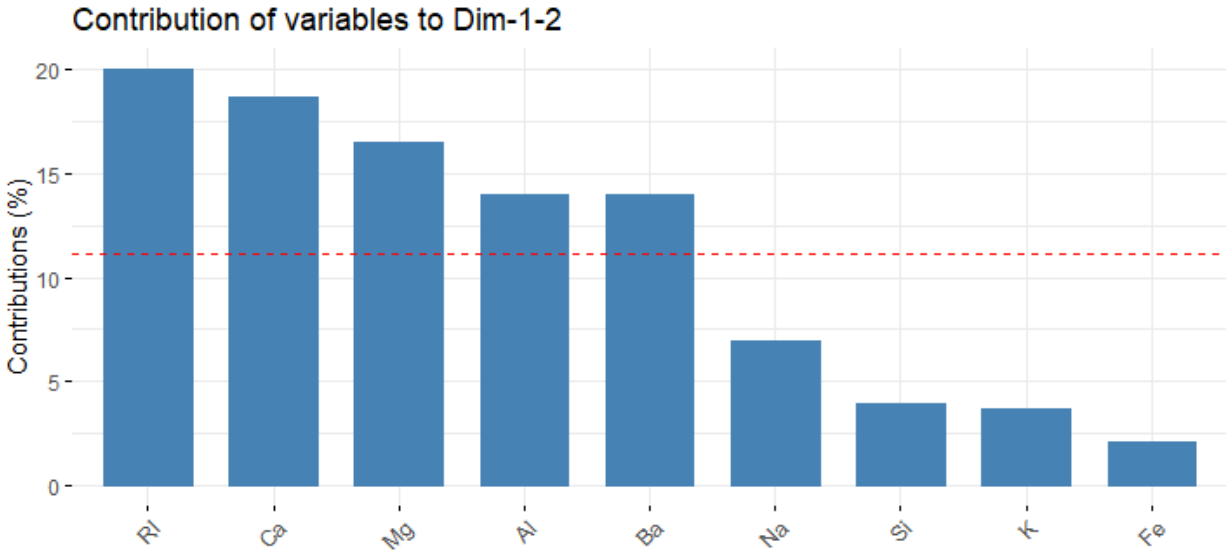
```
           PC1        PC2
RI -0.8639224  0.4090433
Na  0.4090426  0.3870892
Mg -0.1757092 -0.8498611
Al  0.6793596  0.4226860
Si  0.3626290 -0.2220718
K   0.3475869 -0.2204556
Ca -0.7801403  0.4945173
Ba  0.3967607  0.6940001
Fe -0.2944966 -0.0888276
```

Insight: Correlation between RI and PC1 is negative which indicates as the value of PC1 increases then RI decreases on the contrast as Al has positive correlation. For PC2 there is stronger negative correlation for Mg which tells that as PC2 increases Mg decreases. For the variable Ba there is positive correlation for PC2.

iii) Effectiveness:

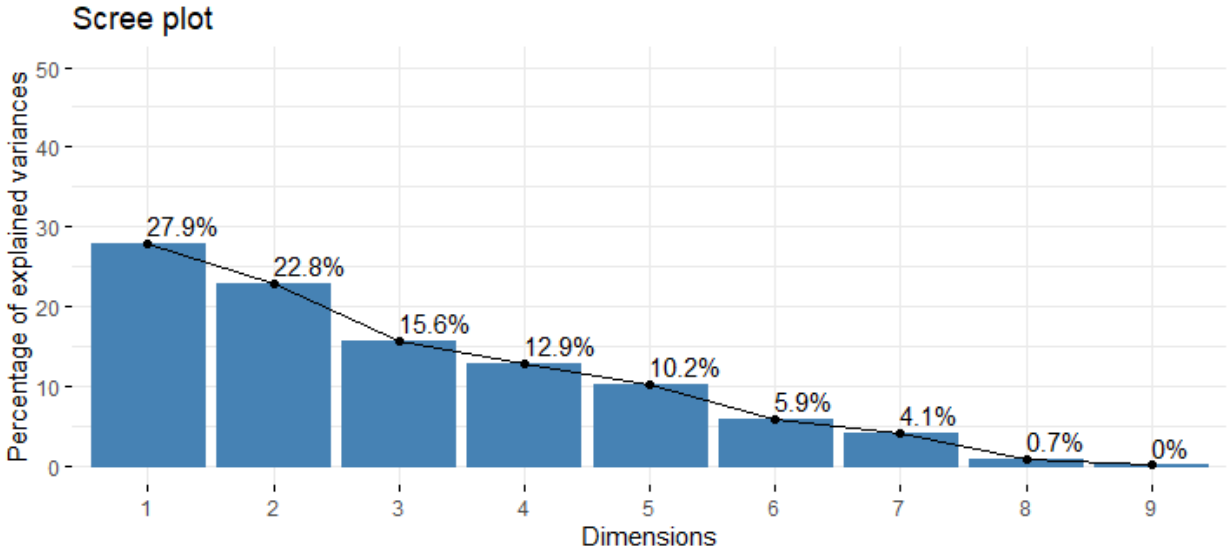
No, I feel PCA does not effectively reduce the dimensions of this Glass data. Most of the variations are not retained in the first few PC'S and they are ill-defined. (PC1&PC2), (PC3 &PC4) have similar variances and eigenvalues are almost similar. These results can be visualized through screeplot and barplot presented on the next page.

fviz_contrib(comp, choice = "var", axes = 1:2)



The plot explains contribution of variables to PC1 and PC2. For a given feature if it is above the cut-off red dashed line 10% are important in contributing to respective PC's.

fviz_eig(comp, addlabels = TRUE, ylim = c(0, 50))



c) Application of LDA:

i) LDA method:

glass_lda<-lda(Type~., data = Glass,scale=TRUE, center=TRUE)

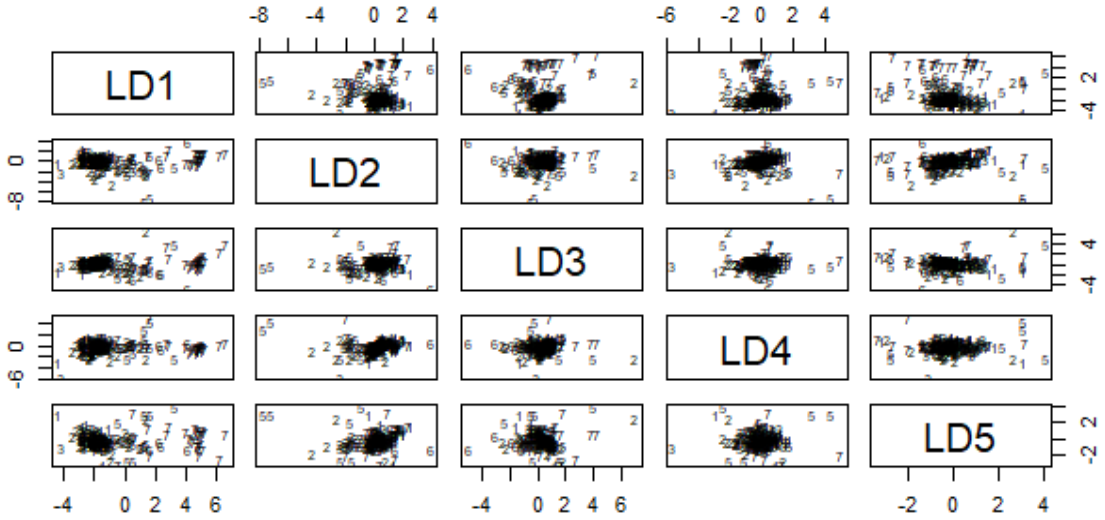
```
Prior probabilities of groups:
         1          2          3          5          6          7
0.32710280 0.35514019 0.07943925 0.06074766 0.04205607 0.13551402

Group means:
         RI       Na       Mg       Al       Si        K        Ca
1 1.518718 13.24229 3.5524286 1.163857 72.61914 0.4474286  8.797286
2 1.518619 13.11171 3.0021053 1.408158 72.59803 0.5210526  9.073684
3 1.517964 13.43706 3.5435294 1.201176 72.40471 0.4064706  8.782941
5 1.518928 12.82769 0.7738462 2.033846 72.36615 1.4700000 10.123846
6 1.517456 14.64667 1.3055556 1.366667 73.20667 0.0000000  9.356667
7 1.517116 14.44207 0.5382759 2.122759 72.96586 0.3251724  8.491379
           Ba          Fe
1 0.012714286 0.05700000
2 0.050263158 0.07973684
3 0.008823529 0.05705882
5 0.187692308 0.06076923
6 0.000000000 0.00000000
7 1.040000000 0.01344828

Coefficients of linear discriminants:
            LD1          LD2          LD3           LD4           LD5
RI 311.6912516 29.3910394 356.0188308 246.85720802 -804.6553938
Na   2.3812158  3.1650800   0.4596785   6.92435141    2.3987509
Mg   0.7403818  2.9858720   1.5728838   6.84983896    2.8002951
Al   3.3377416  1.7247396   2.2024668   6.41923638    0.9371345
Si   2.4516520  3.0063507   1.7026191   7.54220302    0.9562989
K    1.5714954  1.8620159   1.2861127   8.07611300    2.8209927
Ca   1.0063101  2.3729126   0.6475200   6.69663574    3.7110859
Ba   2.3140953  3.4431987   2.5964981   6.43849270    4.4077058
Fe  -0.5114573  0.2166388   1.2026071  -0.04474935   -1.3029207

Proportion of trace:
   LD1    LD2    LD3    LD4    LD5
0.8145 0.1169 0.0413 0.0163 0.0111
> |
```

plot(glass_lda)



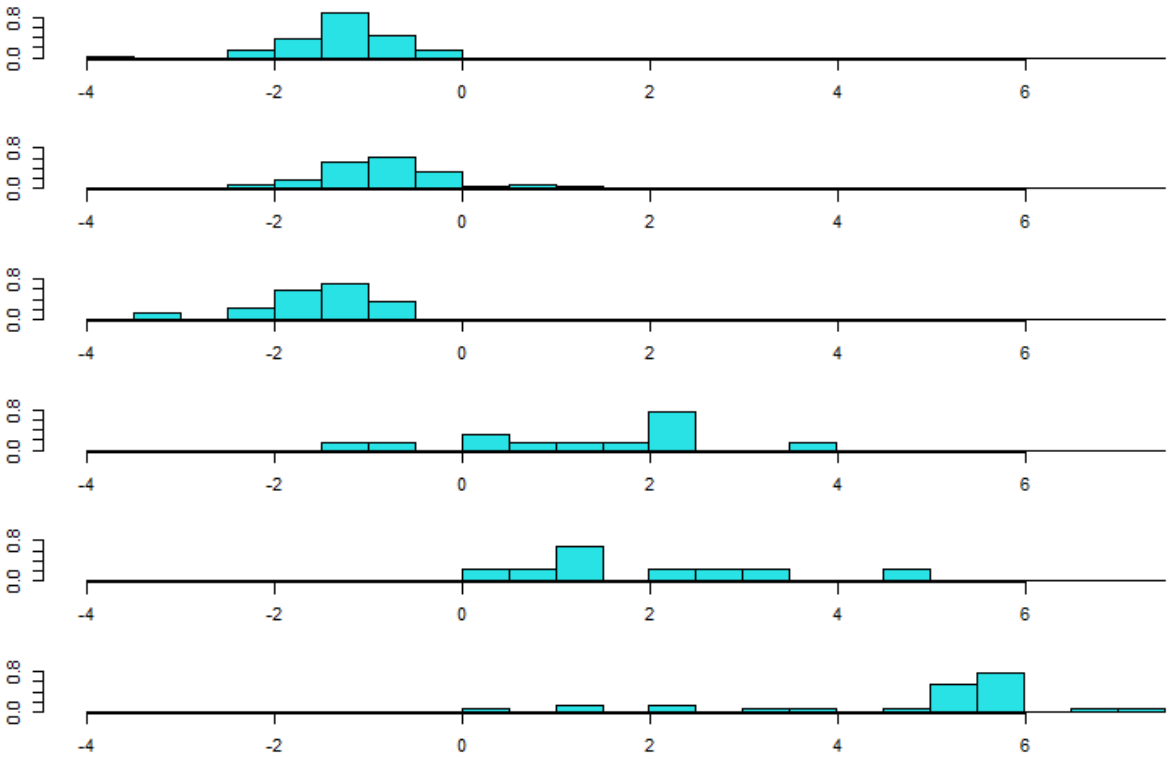
ii) Insight of LD1:

The linear discriminant function for LD1 is: (rounded)
311.69*RI+2.38*Na+0.74*Mg+3.33*Al+2.4*Si+K*1.57+Ca+2.3*Ba+Fe*-0.51

The percentage separation achieved by LD1 is 81.45% from proportion of Trace. It shows that LDA works very well for Glass data as we can achieve about 81% for the very first LD1 and about 93% by LD2.

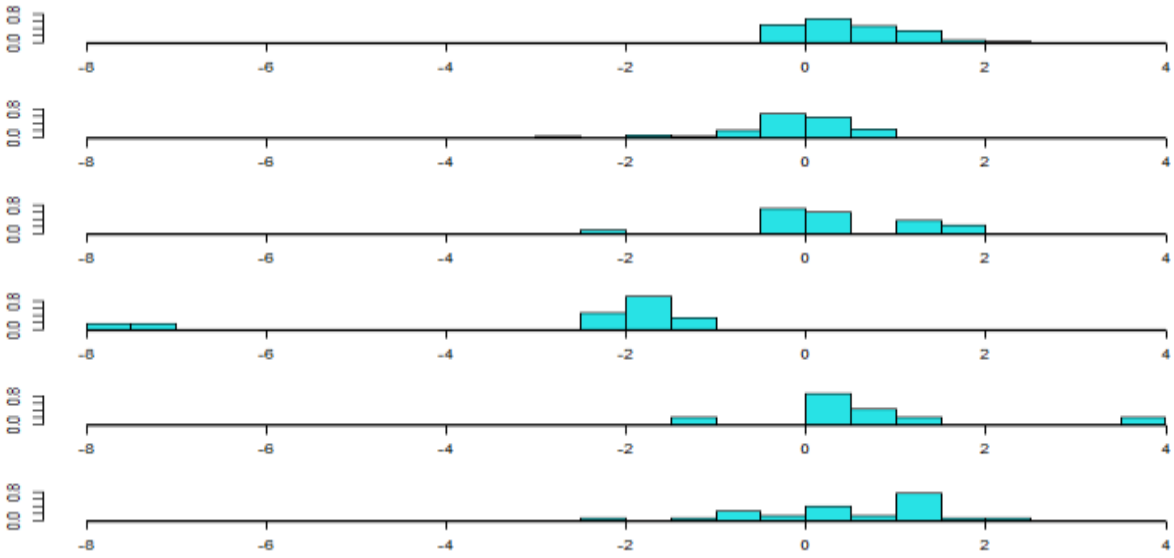
iii) LDA hist:

predict_values<-predict(glass_lda) #Make prediction values
ldahist(predict_values$x[,1],g=Glass$Type) #stacked histogram for LD1
ldahist(predict_values$x[,2],g=Glass$Type) #stacked histogram for LD2



Stacked Histogram for LD1

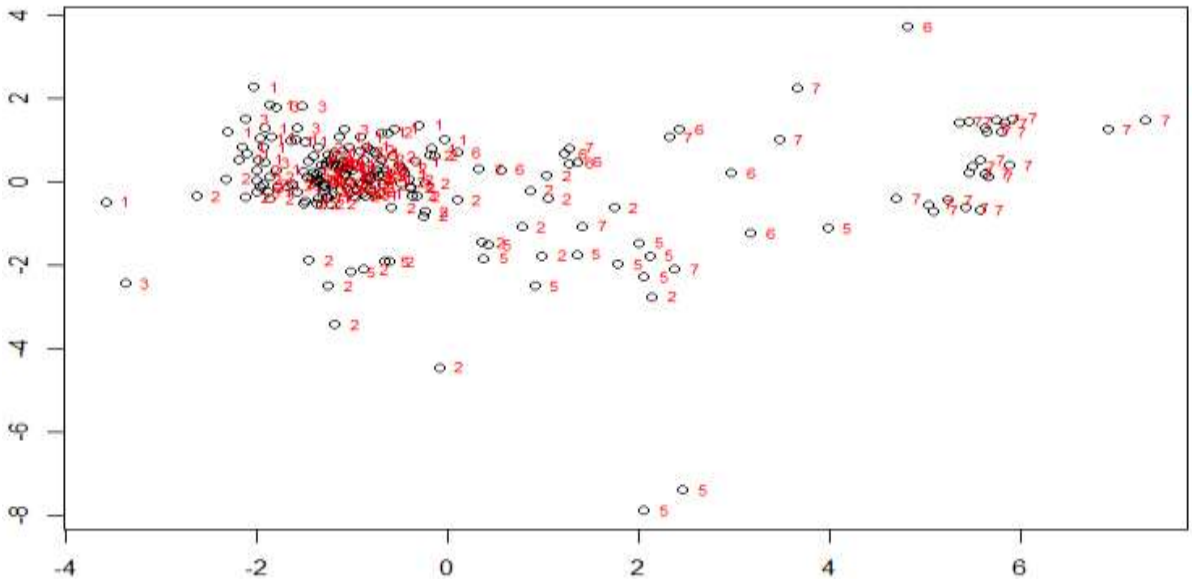
Insight: From the histogram we can see overall the distribution of LD1 is well separated between the values -4 to 6 Type 7 “headlamps” and Type 6 “tableware” is vastly spread-out between the values 0 to 6 with very less overlap. While Type5 between -2 to 4 with moderate overlap around 2. Similarly, type1 and type 3 between -4 and 0 with major overlap around 0 to -2 whereas type 2 from -4 to 2 with overlap around 0 to -2.



Stacked Histogram for LD2

Insight: From the histogram we can see overall the distribution of LD2 is separated between the values -8 to 4. Type 7 “headlamps” is spread-out between the values -2 to slightly above 2 with overlap mostly around “1” and Type 6 “tableware” is spread-out between the values -2 to 4 with overlap around 0 to 1. While Type5 is widely distributed between 0 to -8 with overlap around 1. Similarly, type1 and type 3 between -2 and 2 with overlap around 1 whereas type 2 from -4 to 2 with overlap around “-1” and “1”.

plot(predict_values$x[,1],predict_values$x[,2]) # make a scatterplot



Scatterplot to visualize the points made about LD1 and LD2 separation and overlap.

Problem 2) Facebook Metrics

a) PCA:

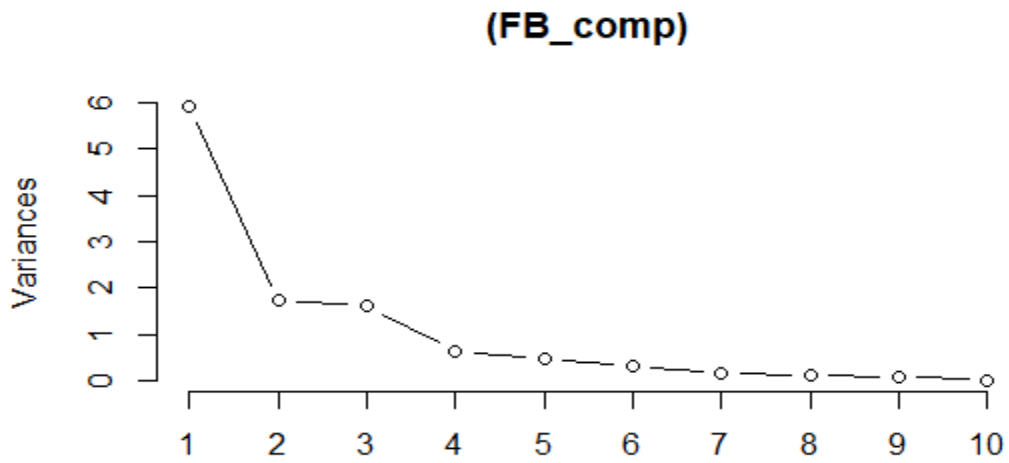
Promp:

FB_comp<-prcomp(FB_data[,c(8:18)],scale. = TRUE, center = TRUE) #on last 11 features with scaling

Summary(FB_comp) #to get the proportion of variance at each level

```
Importance of components:
                          PC1    PC2    PC3     PC4     PC5     PC6     PC7     PC8     PC9
Standard deviation     2.4300 1.3091 1.2707 0.79219 0.68482 0.55935 0.38343 0.35002 0.27073
Proportion of Variance 0.5368 0.1558 0.1468 0.05705 0.04263 0.02844 0.01337 0.01114 0.00666
Cumulative Proportion  0.5368 0.6926 0.8394 0.89646 0.93910 0.96754 0.98090 0.99204 0.99870
                          PC10    PC11
Standard deviation     0.11706 0.02340
Proportion of Variance 0.00125 0.00005
Cumulative Proportion  0.99995 1.00000
> |
```

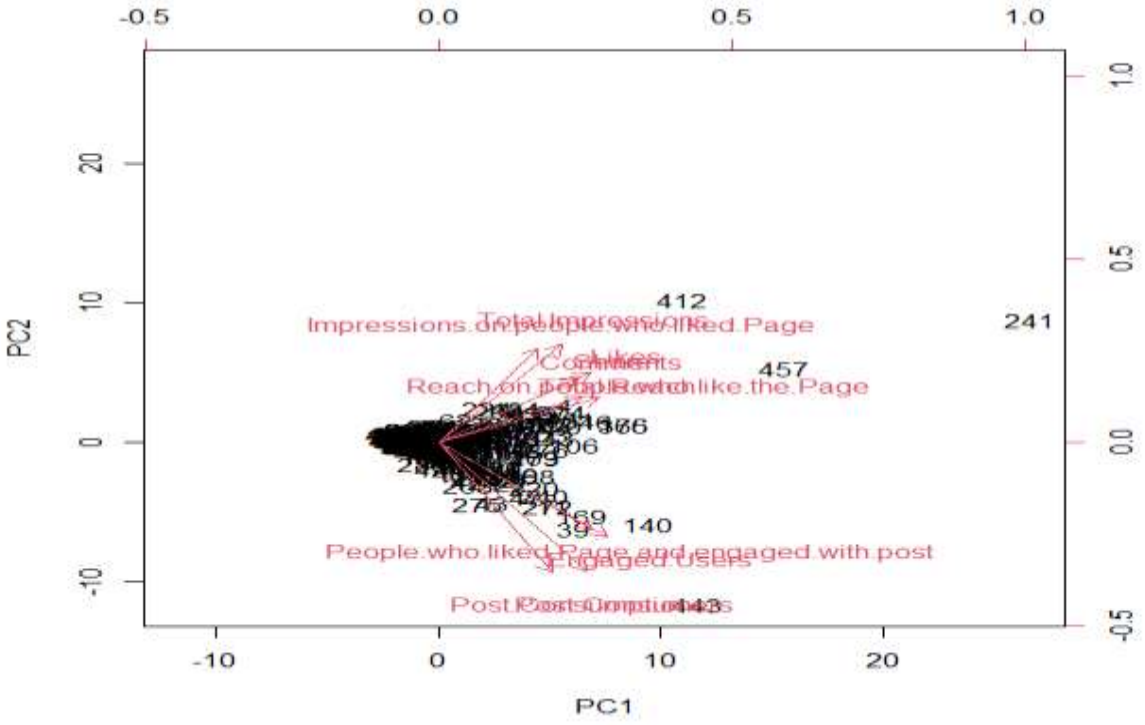
Insight: Almost 54% of variation is covered in the PC1 followed by 15.5% in PC2 which adds up to 60% of variance is captured. By the end of PC7 large chunk of variance is captured about 98%. The variance from one PC to next is well captured in below plot.



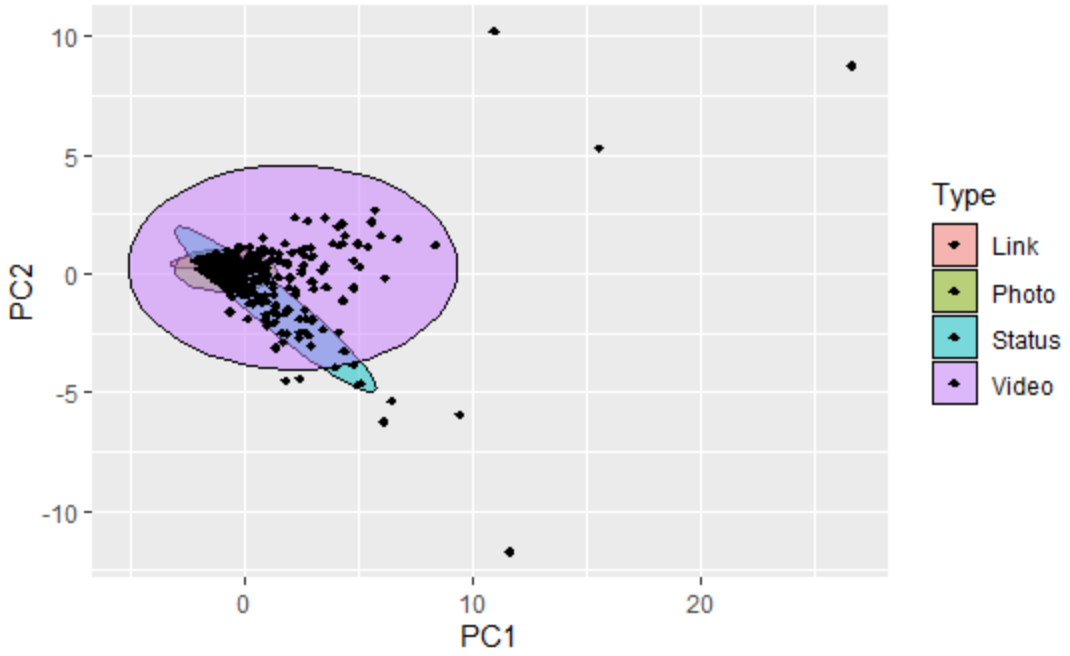
Biplot:

biplot(FB_comp,group_by=FB_data$Type,color=FB_data$Type,scale = 0)

Insight: As PC1 value increases there seem to be a positive increase in all the features where as PC2 increases Engaged users,Post consumer, Post consumptions and People who liked/not liked have a large decrease while Paid has negligible decrease with the rest features have an increase.



Interpretation of PC1 and PC2:



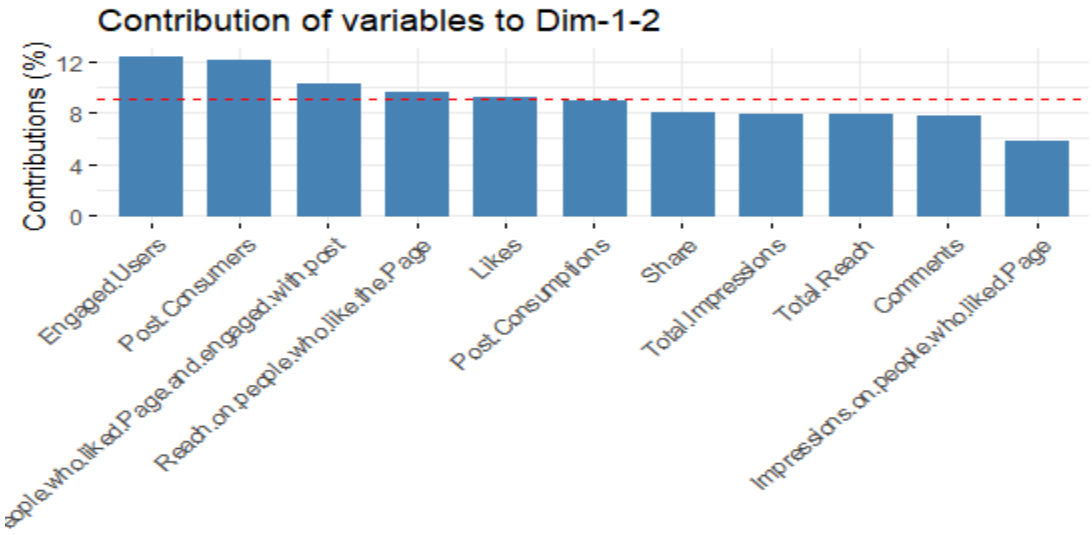
Insight: We can not see a lot of separation based on the "Type" feature. There is a lot of overlap between status, photo and link features while video occupies lot of chunk.

Correlation for PC1 and PC2:

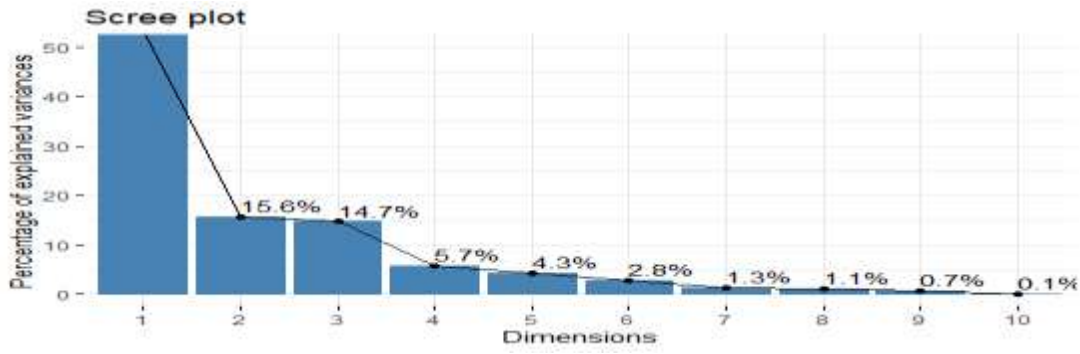
	PC1	PC2
Paid	0.1182652	-0.01351673
Total.Reach	0.7461734	0.20319948
Total.Impressions	0.6424995	0.43521114
Engaged.Users	0.8760239	-0.42461453
Post.Consumers	0.7656779	-0.57842789
Post.Consumptions	0.5847982	-0.58109743
Impressions.on.people.who.liked.Page	0.5137677	0.41774551
Reach.on.people.who.like.the.Page	0.8347720	0.19759396
People.who.liked.Page.and.engaged.with.post	0.7957610	-0.39343740
Comments	0.7154004	0.28638990
Likes	0.7808731	0.30948085
Share	0.7240445	0.29059687

Insight: PC1 has positive correlation with almost all features with Engaged users being highest where PC2 has negative correlation with 5 features.

Bar-plot: To represent contribution of each feature in PC1 and PC2 where red dash represents 10% and almost half of features are above this line.



Screeplot: To explain percentage of variance explored in respective PC's

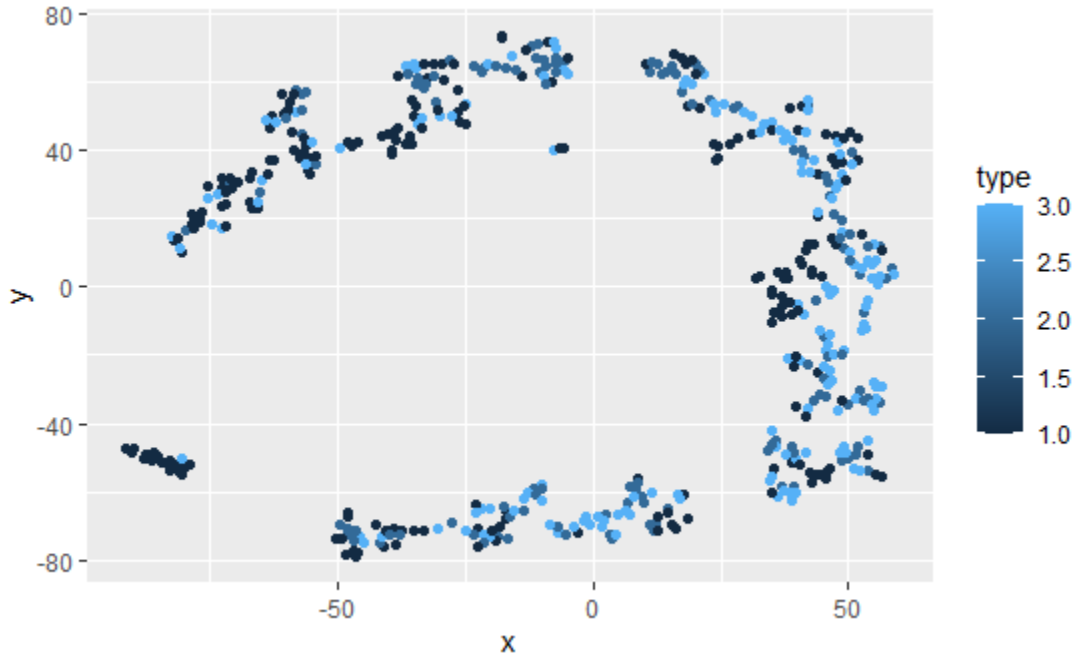


b) t-SNE:

```
tsne_out <- Rtsne(FB_data[,8:18],
                  perplexity=8,
                  theta=0.0,
                  max_iter = 3000) # Run TSNE
```

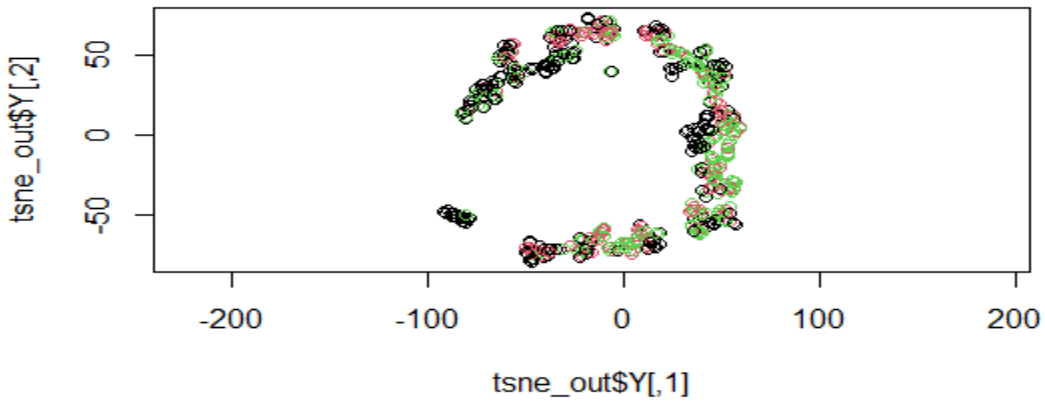
Visual plotting:

```
df<-data.frame(x=tsne_out$Y[,1],y=tsne_out$Y[,2], type=FB_data$Category)
ggplot(data=df,aes(x=x,y=y,group=type,color=type))+geom_point()
```



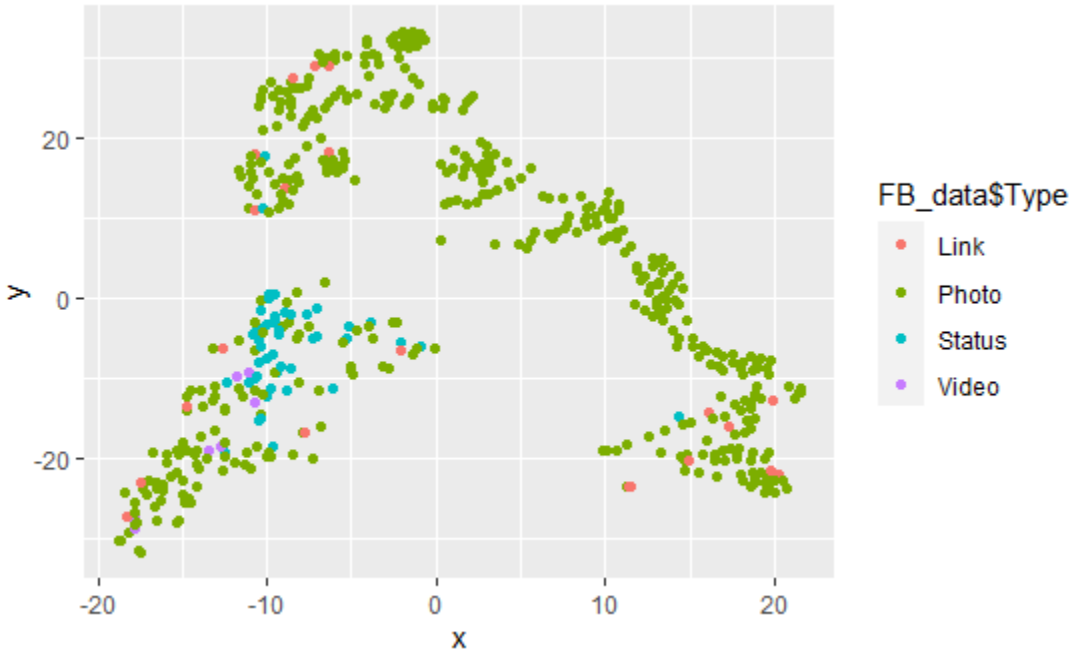
Insight: When we try to plot tSNE using ggplot we can not see a lot of separation between the types based on category.

Show the objects in the 2D tsne representation: Same thing can be observed in 2-D plot.



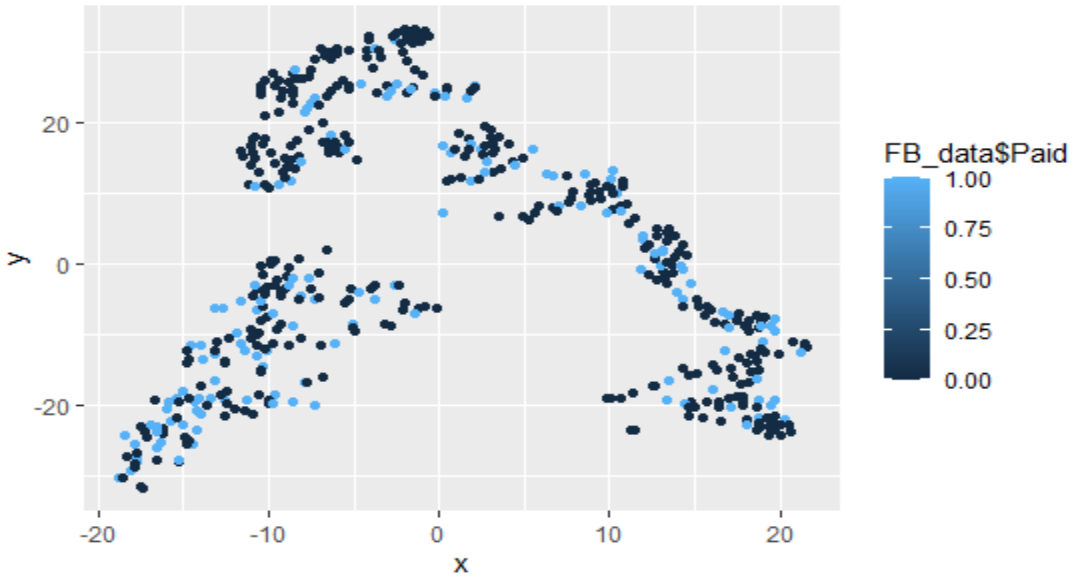
t-SNE on entire FB data:

type=Type

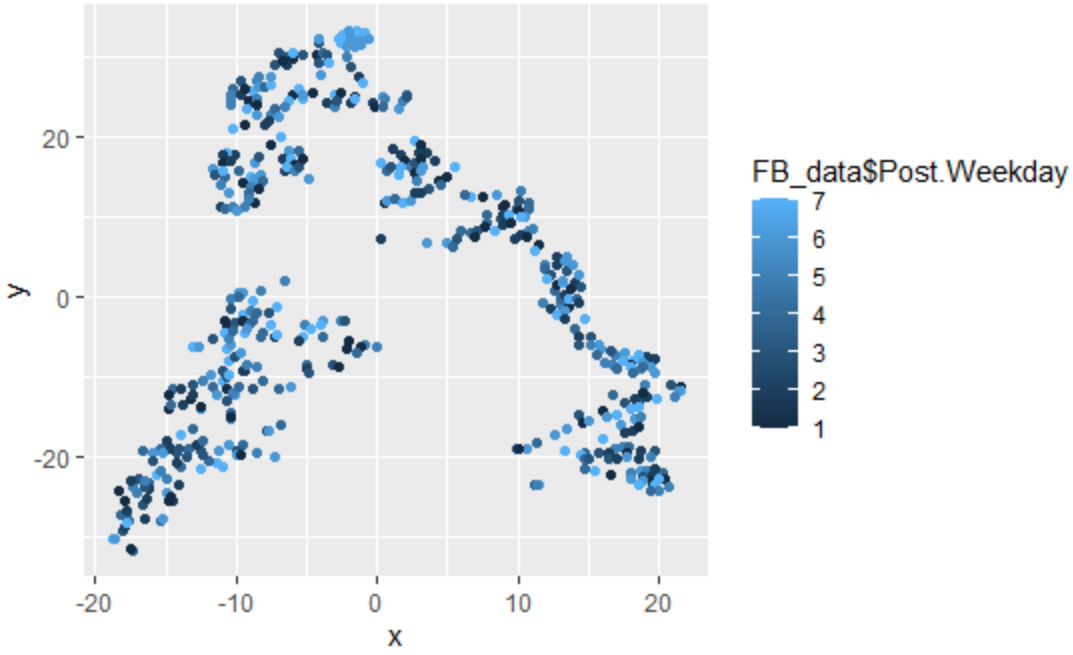


Insight: Photo type is scattered along the plot covering the major portion with slight overlap of other types. Interestingly this plot gives a good understanding on separation and differentiate better among the types when compared plotting with any other features.

Type=Paid- Clustered all over-Hard to differentiate



Type=Post.weekeday- Same clustered all over hard to separate.



Head of tsne_out-Y

```
            [,1]       [,2]
[1,] -11.515434 -59.00441
[2,]  17.473573  57.19456
[3,] -20.350535 -64.76347
[4,] -65.091898  27.80589
[5,]  41.932943  53.64917
[6,]  -7.507825  64.85179
```

Problem 3) UMAP

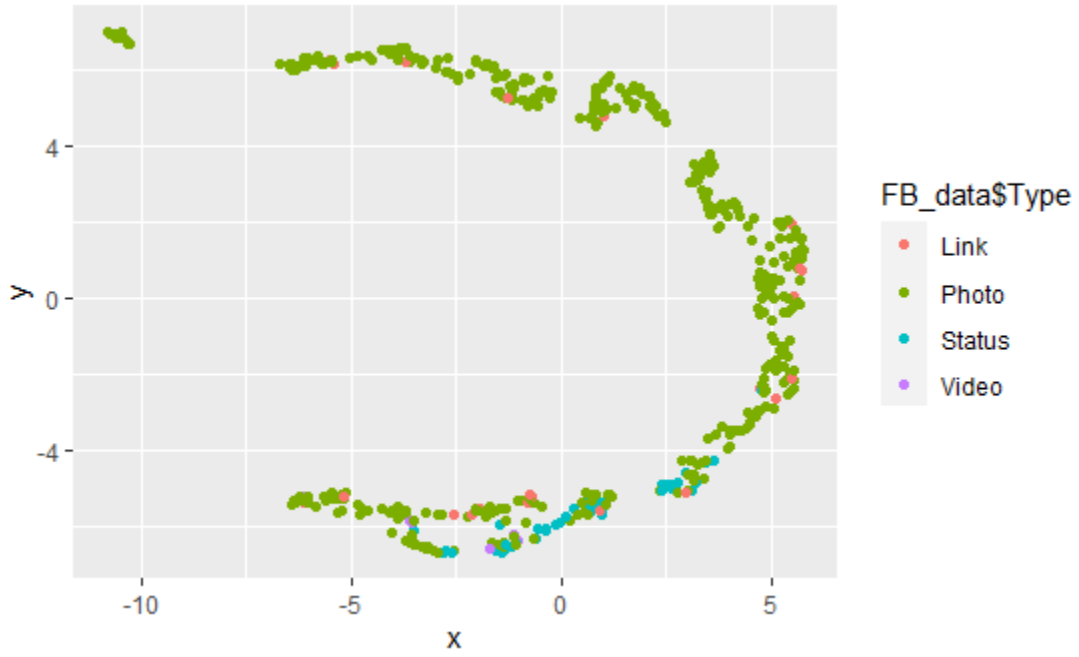
```
fb_data.frame<-FB_data[,c(8:18)]
fbdata_umap<-umap(fb_data.frame)
```

```
umap embedding of 495 items in 2 dimensions
object components: layout, data, knn, config
```

Main component of object components in layout.

```
            [,1]       [,2]
[1,] -2.5858776  5.874843
[2,]  2.9772679 -4.558935
[3,] -3.6206880  6.228290
[4,] -4.7941965 -5.700431
[5,]  4.7952774 -2.267333
[6,]  0.8605818 -5.314038
```

Visualize based on Type of post:



Comparison UMAS VS t-SNE: For the Type of post plot UMAS seems to outperform t-SNE with comparatively less amount of noise.