Project_Explained

Sofi Tiwari

31/08/2019

Importing the required libraries

Downloading and reading the data

```
#Downloaded the data from Synapse:Note, its required to create a synapse id
#https://www.synapse.org/#!Synapse:syn4303551
synLogin('sofitiwari','mlproject')
```

```
## Welcome, sofitiwari!
```

```
## NULL
```

```
# Obtain a pointer and download the data
syn4303551 = synGet(entity='syn4303551', downloadLocation=getwd())
df = read.table(file = 'unc.edu_PANCAN_IlluminaGA_RNASeqV2.geneExp.tsv',
                sep = '\t', header = TRUE)
```

Manipulating the data for use in further analysis

```
#Dropping the first 29 rows because the gene id's are not legible
df_mod = df[-c(1:29),]
rownames(df_mod)= c(1:nrow(df_mod))

#These ~20k genes are currently in the row. Transposing them to get as columns(features)
#and the rows will be the experiments
df_fin = transpose(df_mod)
rownames(df_fin) = colnames(df_mod)
colnames(df_fin) = df_fin[1,]
df_fin = df_fin[-1,]
df_fin = transpose(df_fin)
#Converting into a matrix for further use in plotting
df_matrix = as.matrix(sapply(df_fin,as.numeric))

#Checking for NA values. The function below counts the total NA values in each column
na_df = sapply(df_fin, function(x) sum(is.na(x)))

#no NA values
na_max = max(na_df)
summary(df_fin$V1)
```

```
##    Length     Class      Mode
##     20502 character character
```

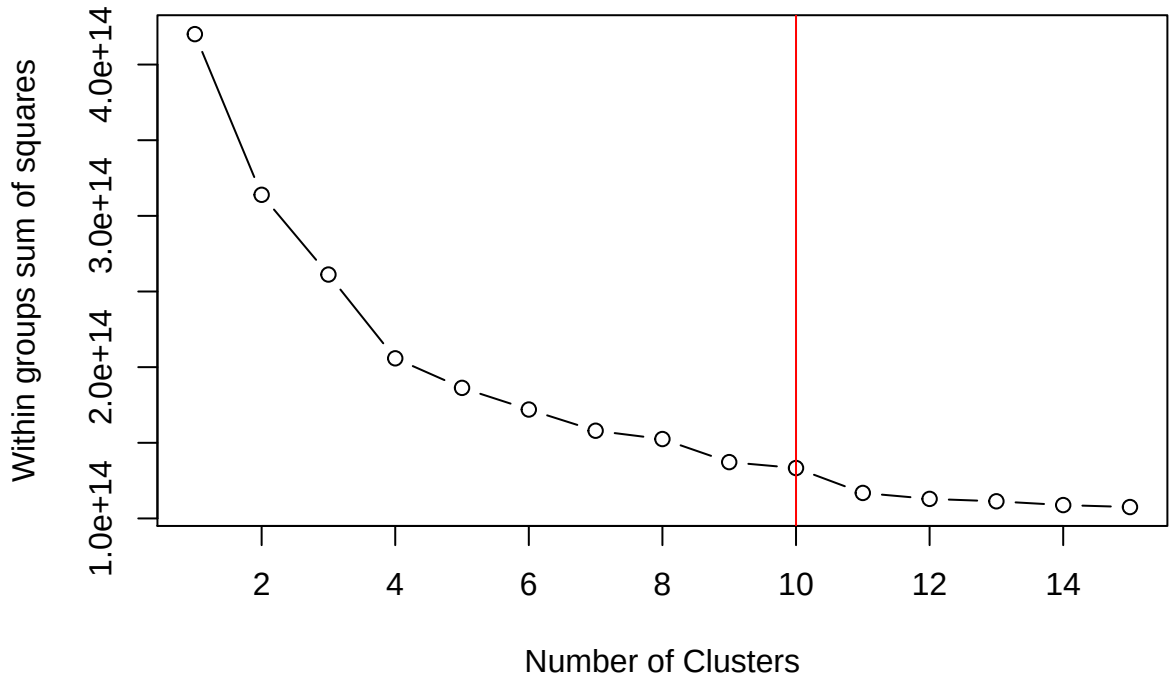
Elbow plot

Using elbow plot to check how many clusters would I need. It looks like a smaller number of cluster is not enough for these ~20k genes, because the within-cluster sum of square(wss) is very high.

```
wss = (nrow(df_fin)-1)*sum(apply(df_fin,2,var))


for (i in 2:15)
  wss[i] = sum(kmeans(df_fin,centers=i,iter.max=30)$withinss)

options(scipen=3)
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
abline(v=10, col="red")
```



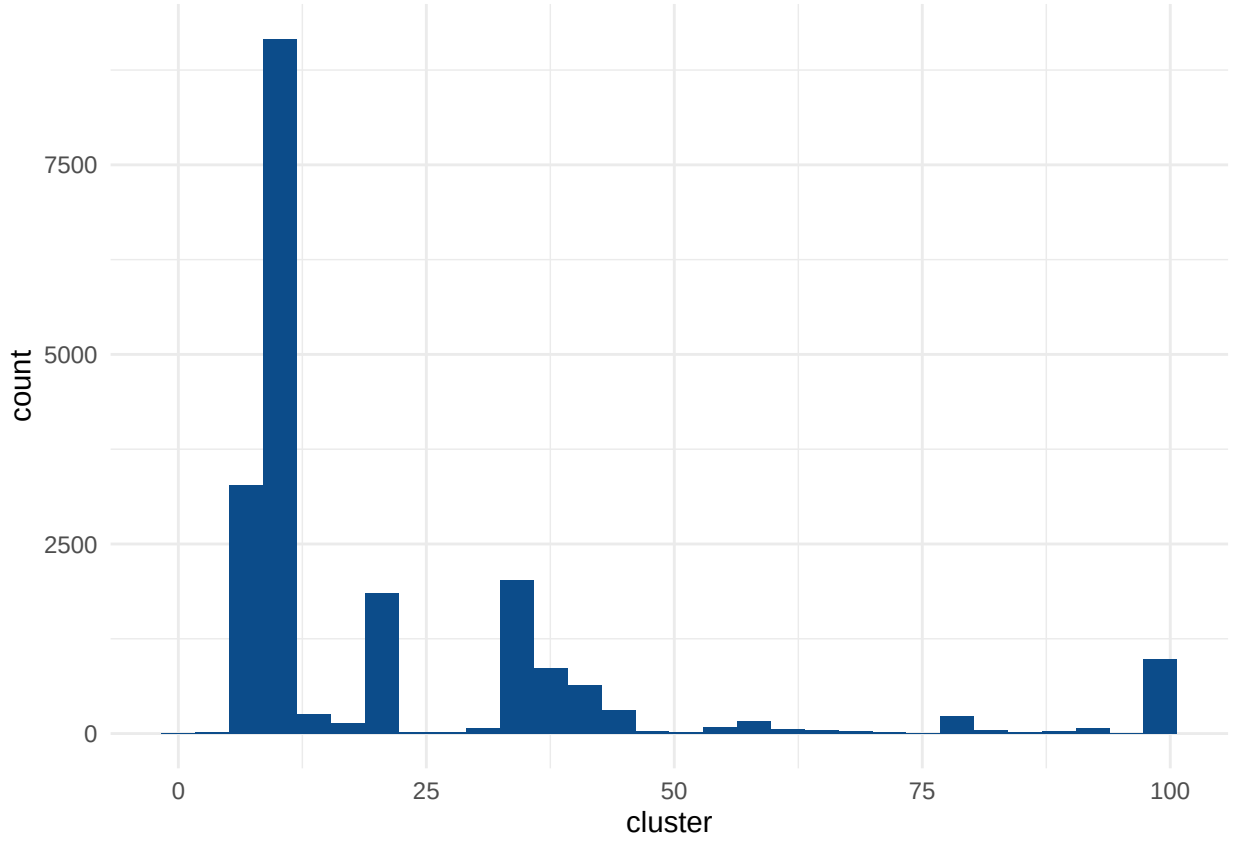
K-means clustering with different cluster size

K-means clustering on the data when clusters=100

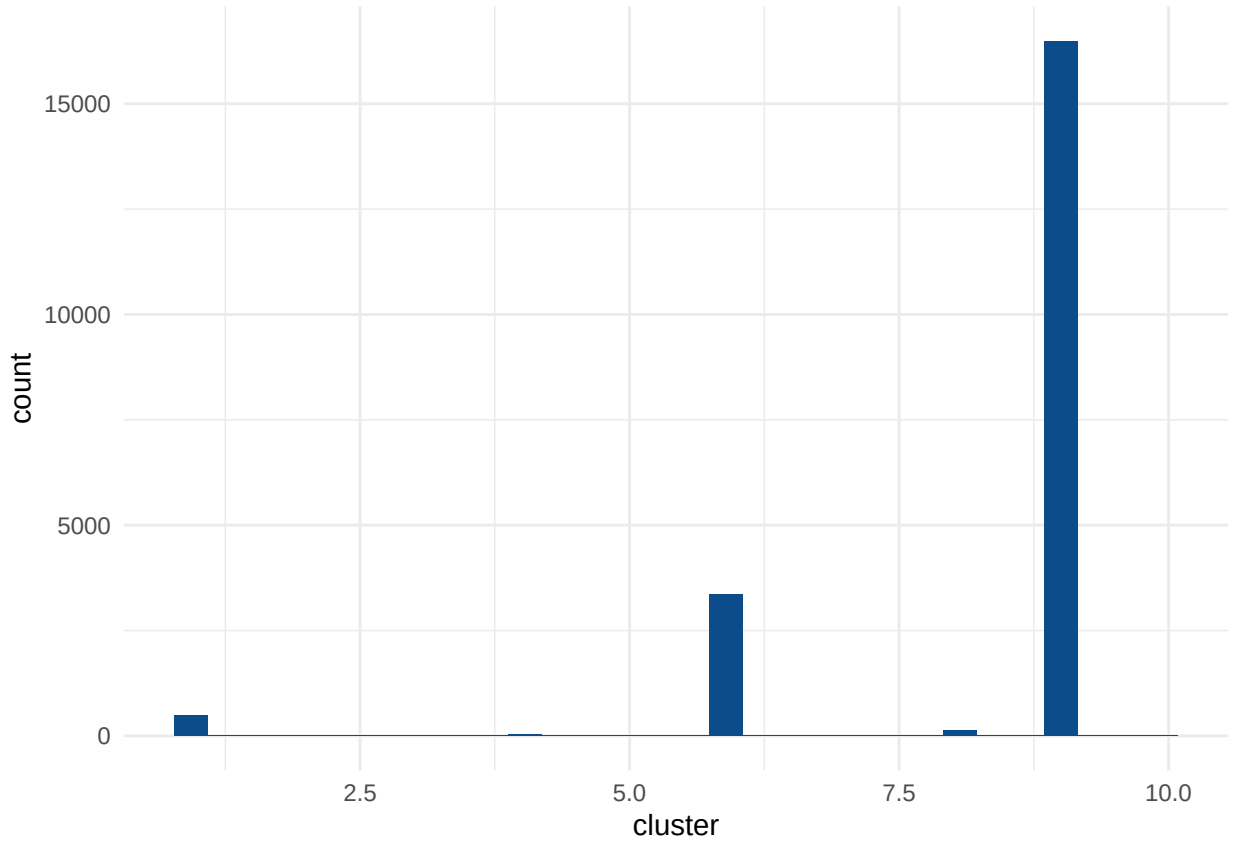
```
set.seed(100)
k_fit1 = kmeans(df_fin, centers = 100,iter.max = 30)
#print(k_fit1)
x=as.data.frame(k_fit1$cluster)

#50% of the data is clustered together.100 clusters are not sufficient to visualise
  #any sort of clustering in the data
ggplot(data = x) +
  aes(x = `k_fit1$cluster`) +
  geom_histogram(bins = 30, fill = '#0c4c8a') +
```

```
  xlab("cluster") +
  theme_minimal()
```



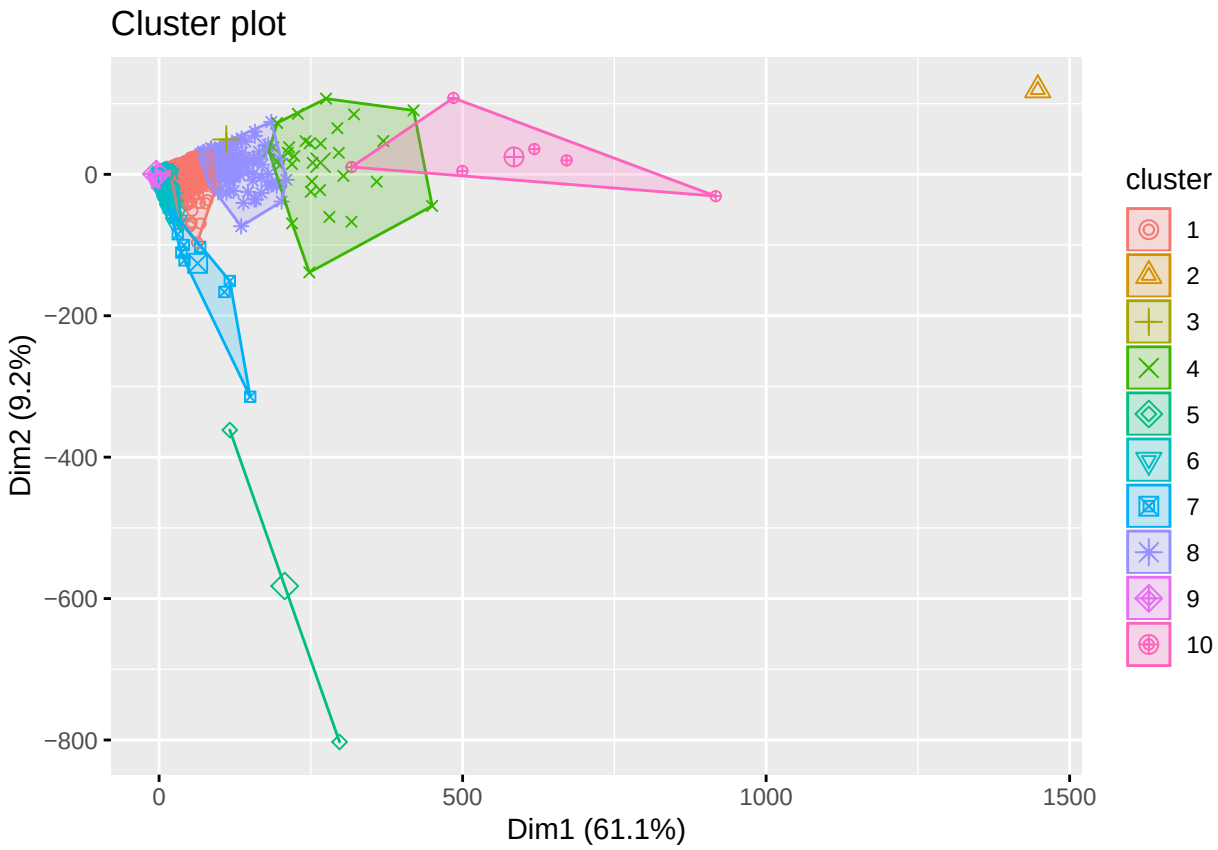

```
#K-means clustering after reducing the number of clusters to 10. Its easier to visualise
set.seed(100)
k_fit2 = kmeans(df_fin, centers = 10,iter.max = 30)
#print(k_fit)
y=as.data.frame(k_fit2$cluster)
ggplot(y) +
  aes(x = `k_fit2$cluster`) +
  geom_histogram(bins = 30L, fill = "#0c4c8a") +
  xlab("cluster") +
  theme_minimal()
```



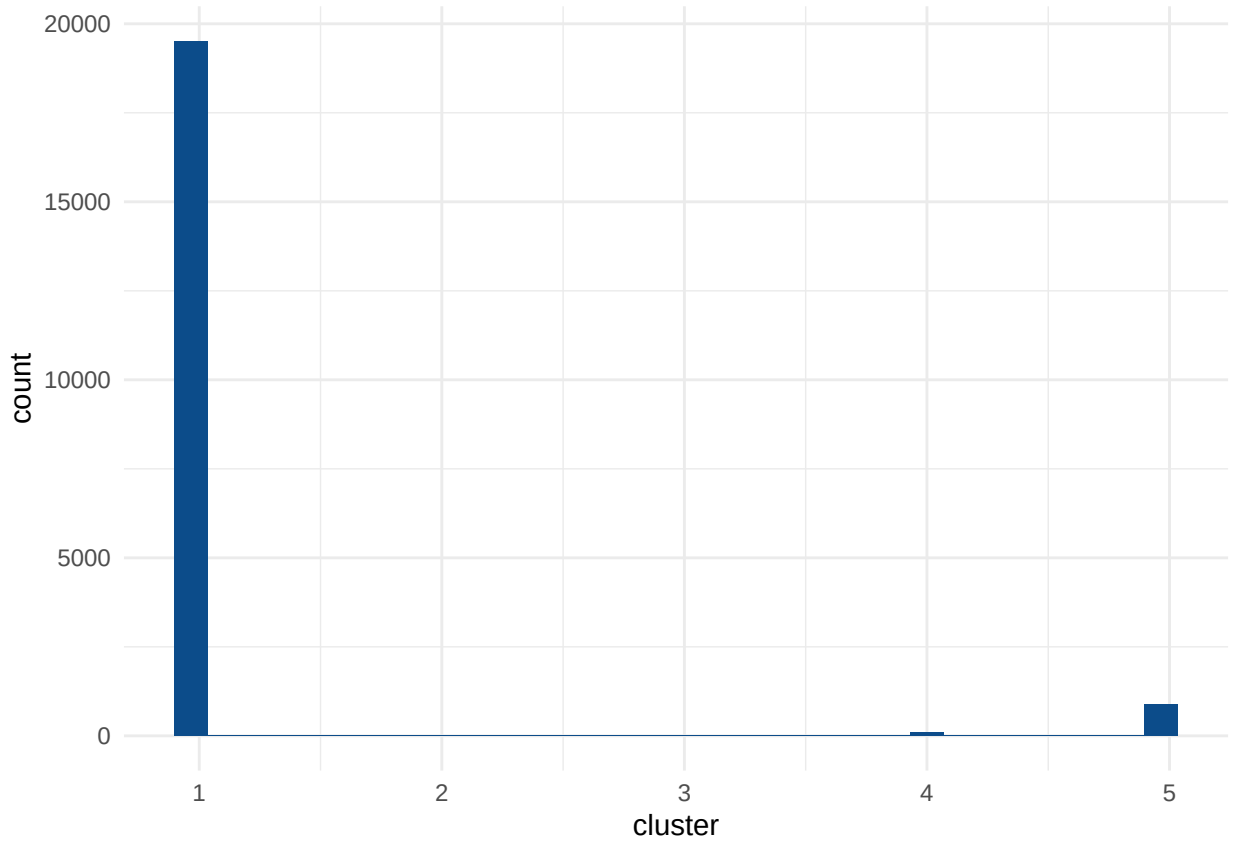
Visualizing the 10 clusters

```
fviz_cluster(k_fit2, df_matrix, frame = FALSE, geom = "point")
```

```
## Warning: argument frame is deprecated; please use ellipse instead.
```



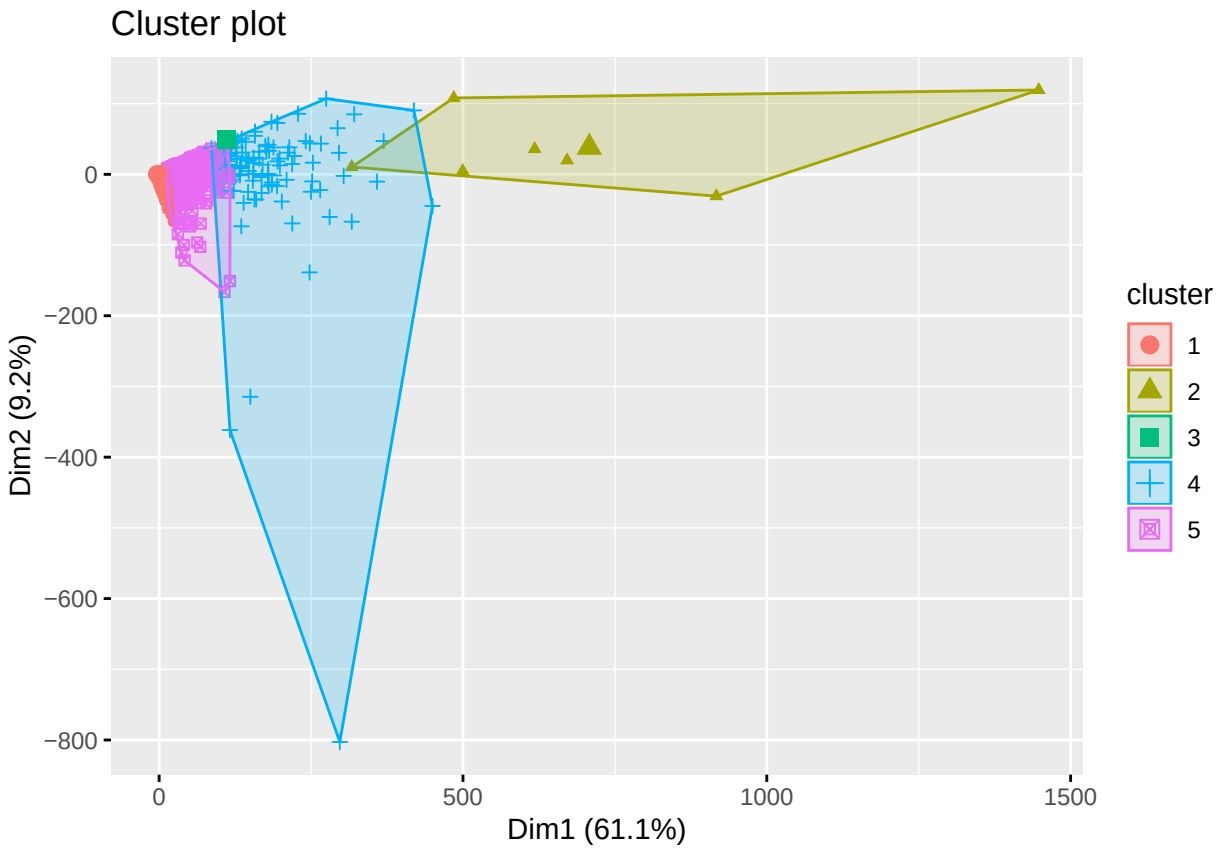

```
#K-means clustering after reducing the number of clusters to 5
set.seed(100)
k_fit = kmeans(df_fin, centers = 5,iter.max = 30)
#print(k_fit)
z=as.data.frame(k_fit$cluster)
ggplot(z) +
  aes(x = `k_fit$cluster`) +
  geom_histogram(bins = 30L, fill = "#0c4c8a") +
  xlab("cluster") +
  theme_minimal()
```



Visualizing 5 clusters

```
fviz_cluster(k_fit, df_matrix, frame = FALSE, geom = "point",ellipse = TRUE)
```

```
## Warning: argument frame is deprecated; please use ellipse instead.
```



Hierarichal clustering(not feasible for huge dataset)

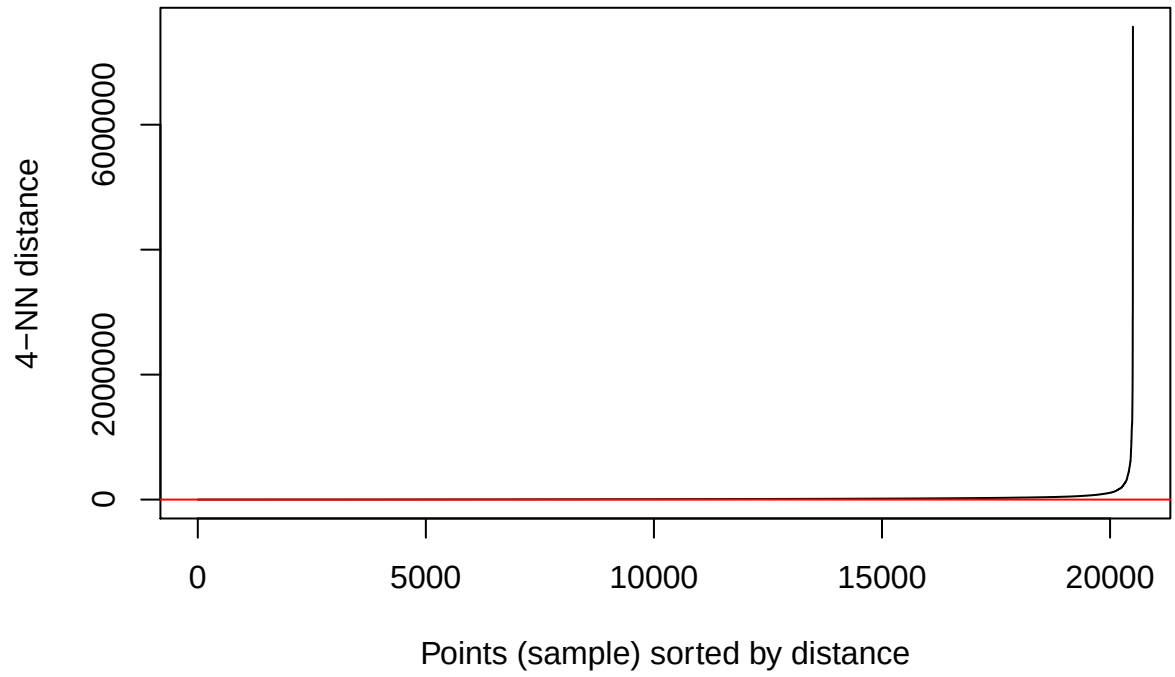
```
#does not seem very feasible to me, because it creates a distance matrix
#so computer will run out of space

#dist_mat = dist(df_fin, method = 'euclidean')
```

Density based clustering

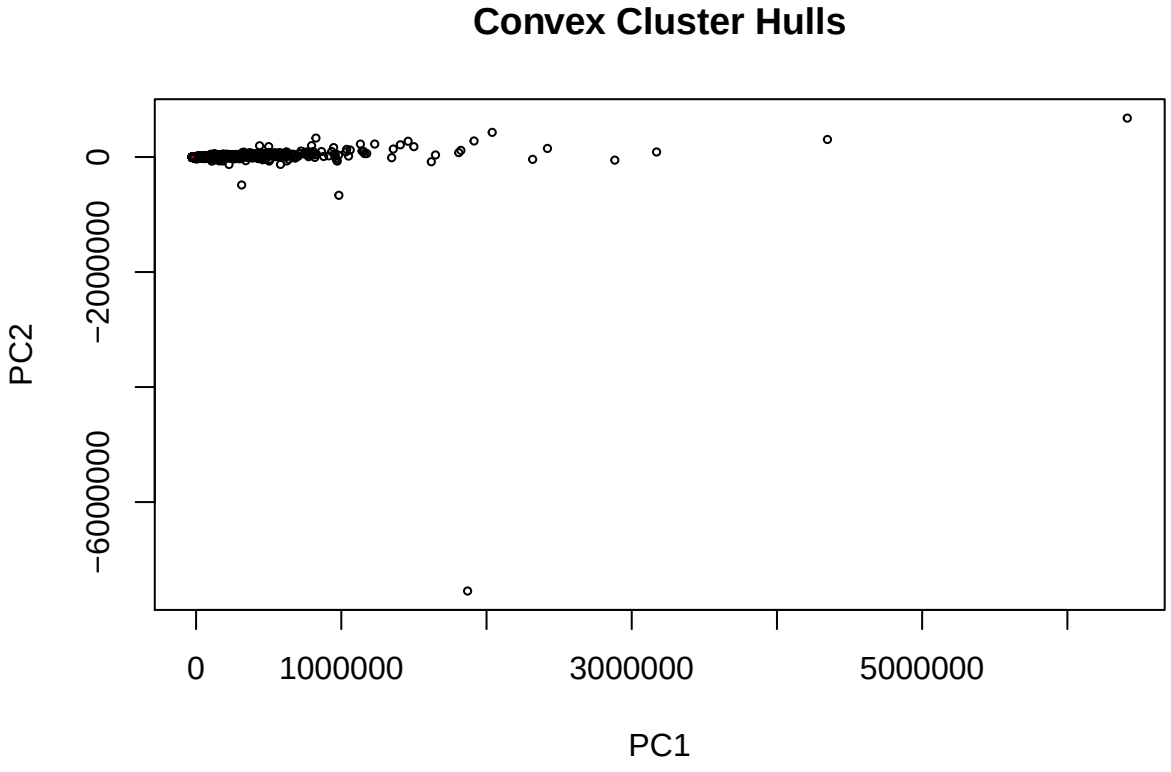
```
#https://en.proft.me/2017/02/3/density-based-clustering-r/
df_matrix = as.matrix(sapply(df_fin,as.numeric))

kNNdistplot(df_matrix, k=4)
abline(h=0.4, col="red")
```




```
db = dbscan(df_matrix, 0.4, 50)

hullplot(df_matrix, db$cluster)
```

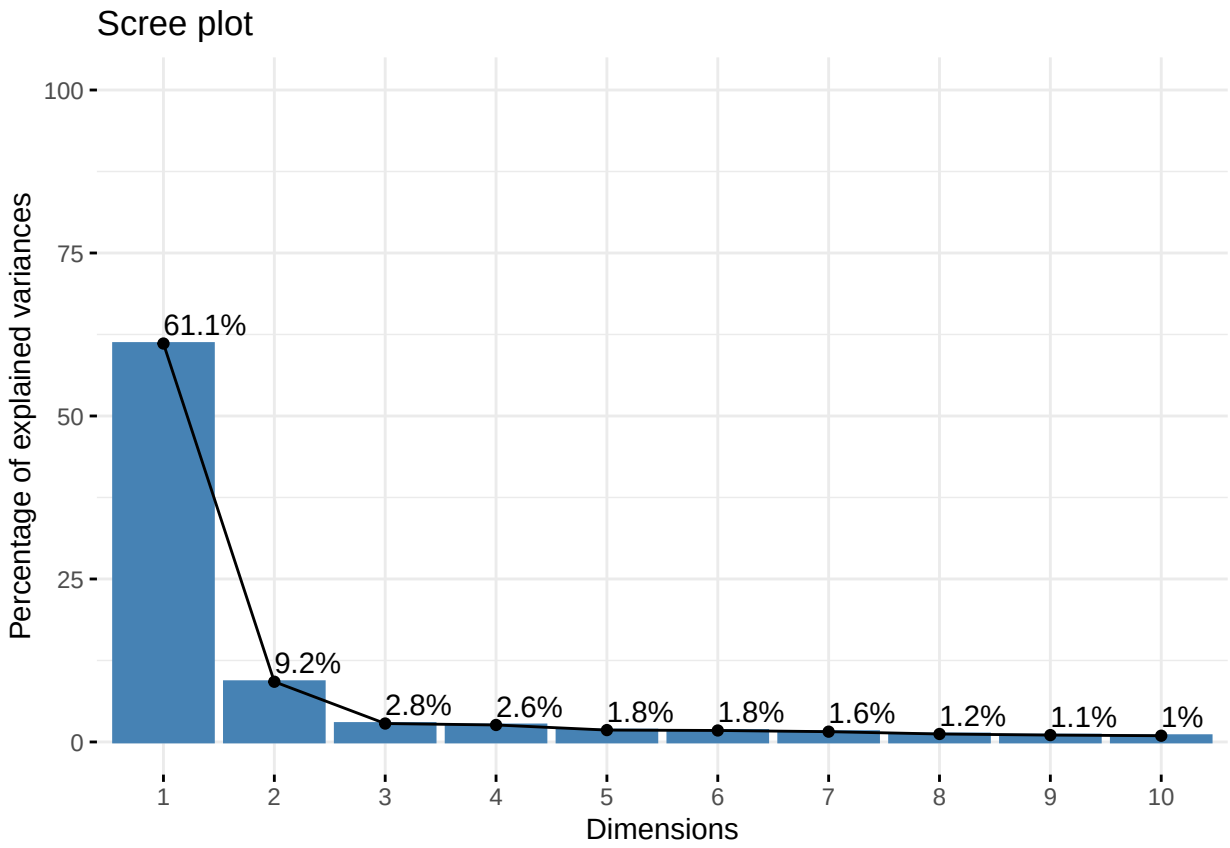


PCA

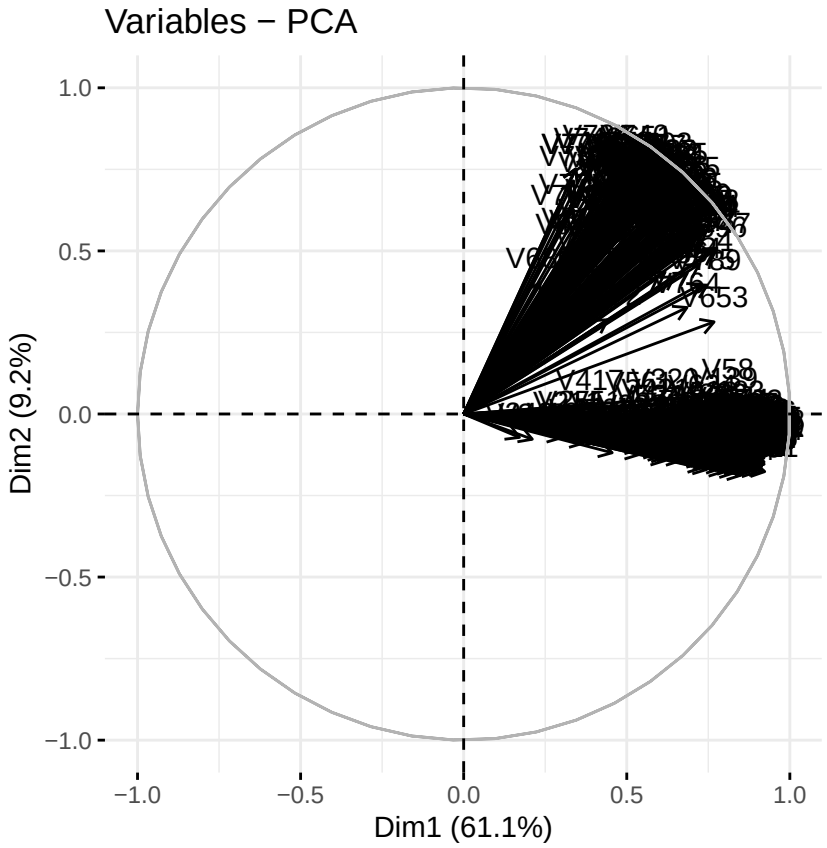
```
res.pca = PCA(df_matrix, graph = FALSE)
print(res.pca)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 20502 individuals, described by 819 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

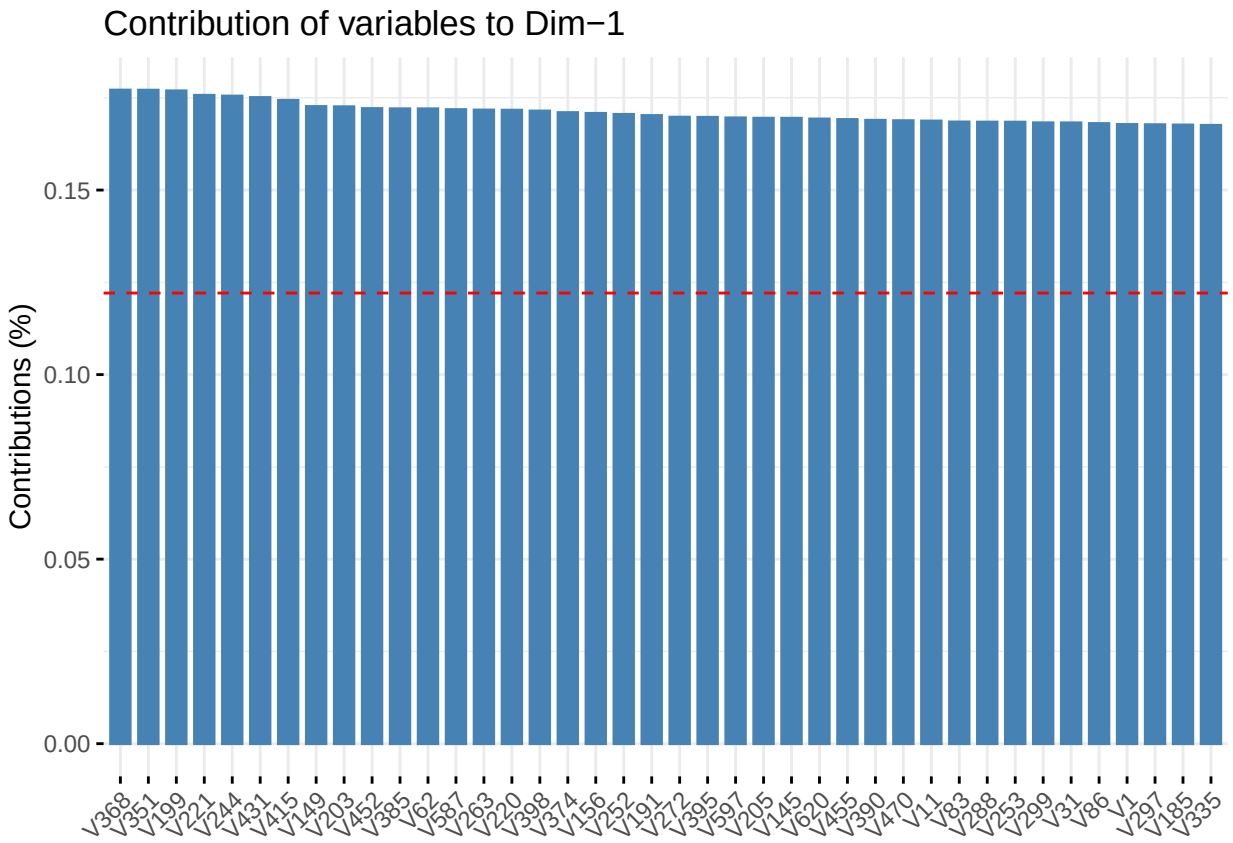
```
#Eigen values can be used to determine the number of components to keep after PCA
eig.val =  get_eigenvalue(res.pca)
#View(eig.val)
fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 100))
```



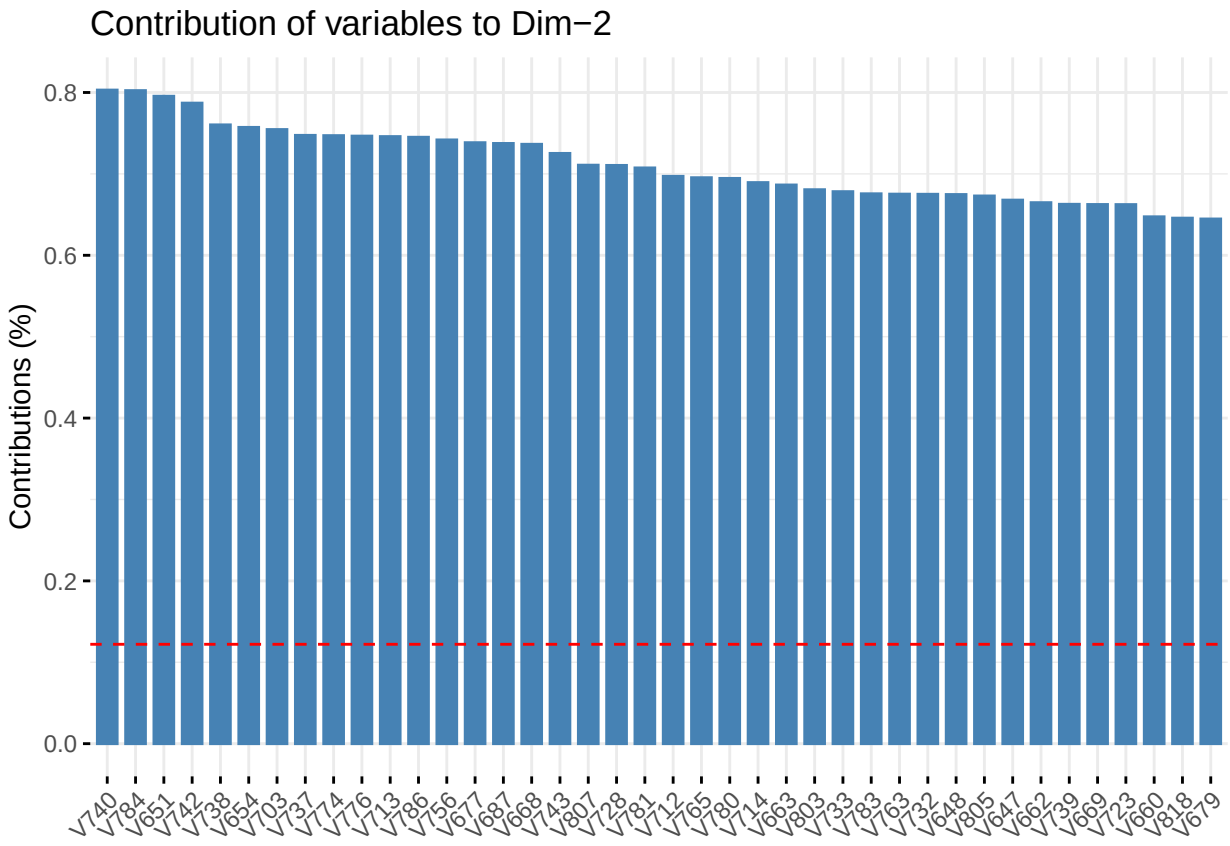

```
#Contribution of the variables
fviz_pca_var(res.pca, col.var = "black")
```




```
# Contributions of variables to PC1
fviz_contrib(res.pca, choice = "var", axes = 1, top = 40)
```




```
# Contributions of variables to PC2
fviz_contrib(res.pca, choice = "var", axes = 2, top = 40)
```



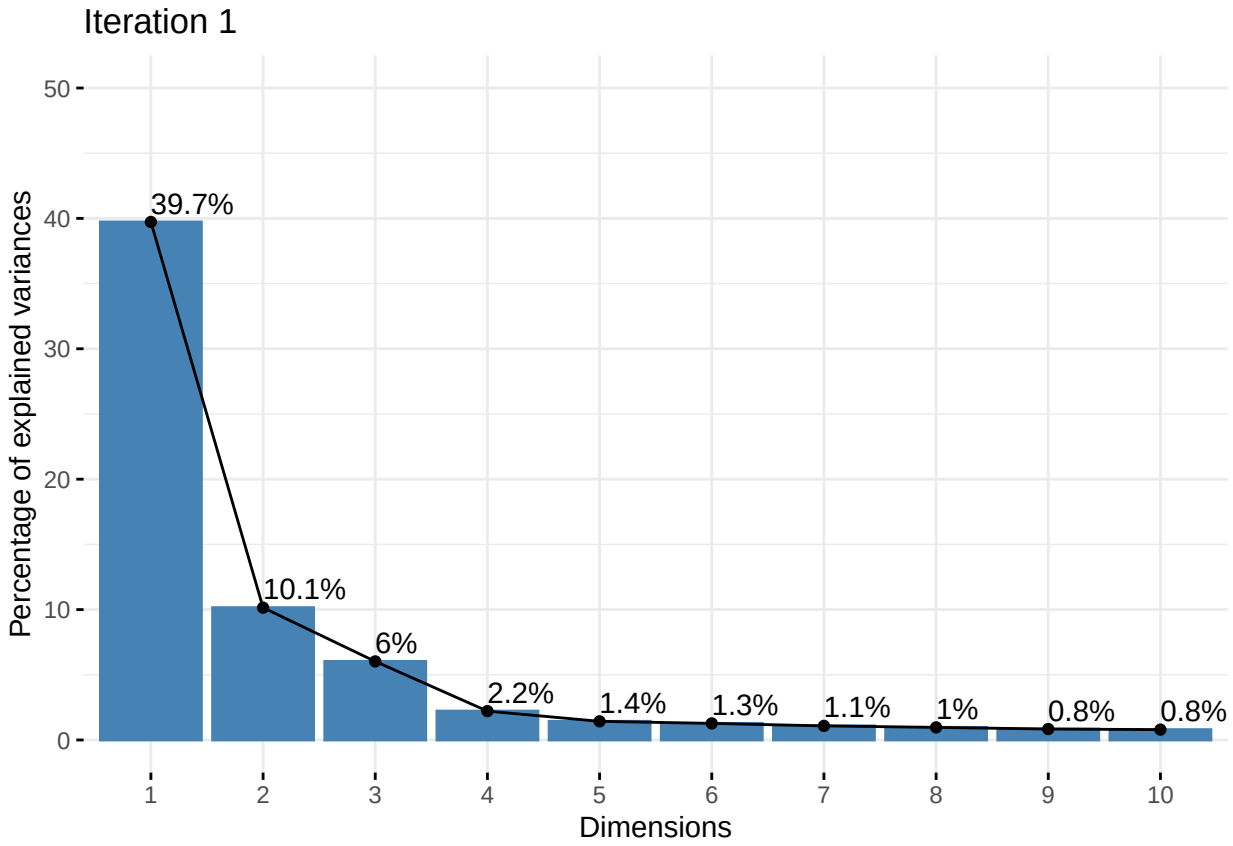
The real process starts now...

Iteration 1

```
#Adding the k-means cluster to the df
df_fin1 = cbind(df_fin,y$`k_fit2$cluster`)
colnames(df_fin1)[820] = "k_cluster"
table(k_fit2$cluster) #9th cluster has highest population
```

```
##
##     1     2     3     4     5     6     7     8     9    10
##   488     1     1    30     2  3352    10   123 16489     6
```

```
#PCA on the most dense cluster
df_subset1 = df_fin1[which(df_fin1$k_cluster==9),]
res.pca = PCA(as.matrix(sapply(df_subset1,as.numeric)), graph = FALSE)
p1 = fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 50), main = "Iteration 1")
p1
```



Defining some functions

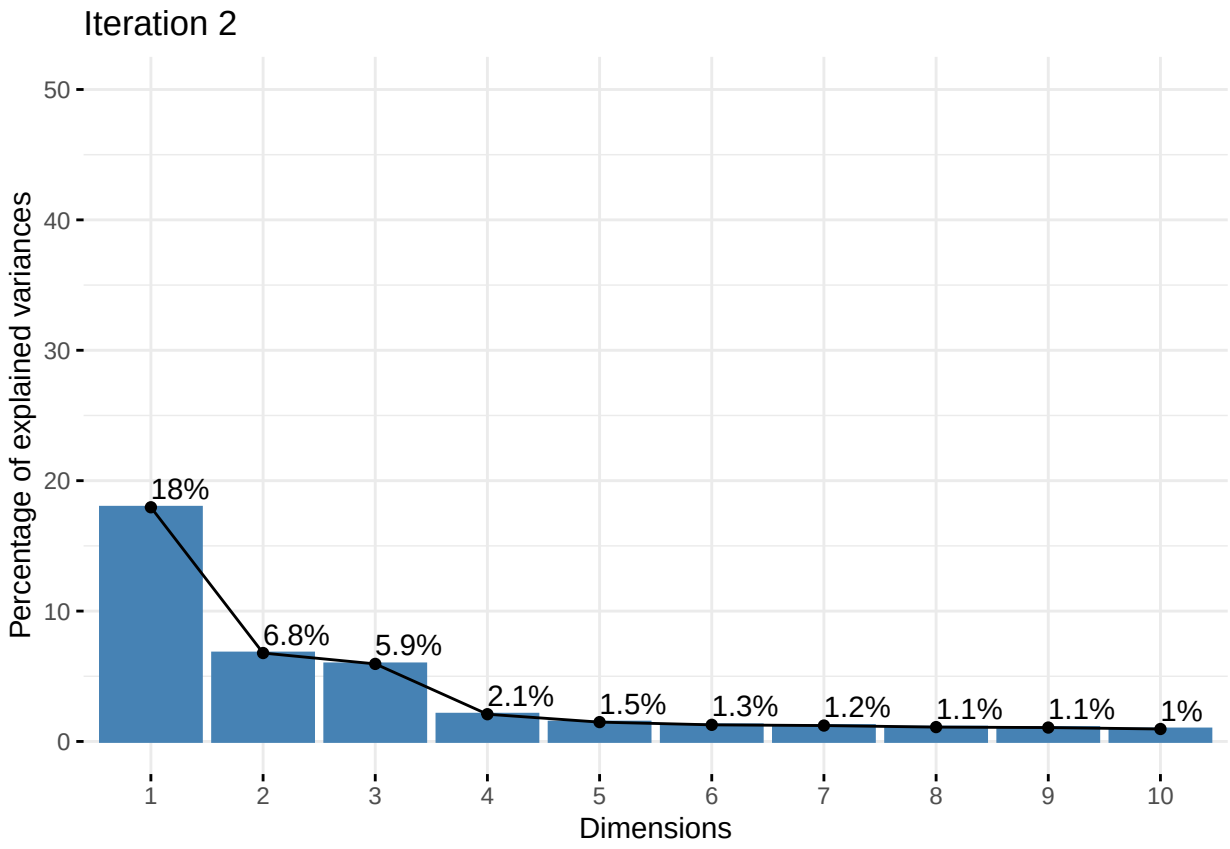
```
#The cluster_subset function performs k-means clustering(cluster size=10)
#and returns the cluster number that has most number of genes
cluster_subset = function(df_subset)
{
  set.seed(100)
  iter = kmeans(df_subset[,-820], centers = 10,iter.max = 30)#k-means clustering
  df_fin = cbind(df_subset[,-820],iter$cluster)
  colnames(df_fin)[820] = "k_cluster"
  tt = table(iter$cluster)
  max_cluster = as.numeric(names(tt[which.max(tt)])) #cluster with highest population
  df_subset = df_fin[which(df_fin$k_cluster==max_cluster),]#subset data
  return(df_subset)
}

#The pca_plot function performs PCA on the clustered dataset and returns the plot
# explaining the percentage of variance by the first 10 principal components
pca_plot = function(df_subset,iteration)
{
res.pca = PCA(as.matrix(sapply(df_subset,as.numeric)), graph = FALSE)#PCA
plt = fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 50), main = iteration) #plotting
return(plt)
}
```

Iterating the Clustering–>PCA–>Clustering–>PCA. . . .process 11 times

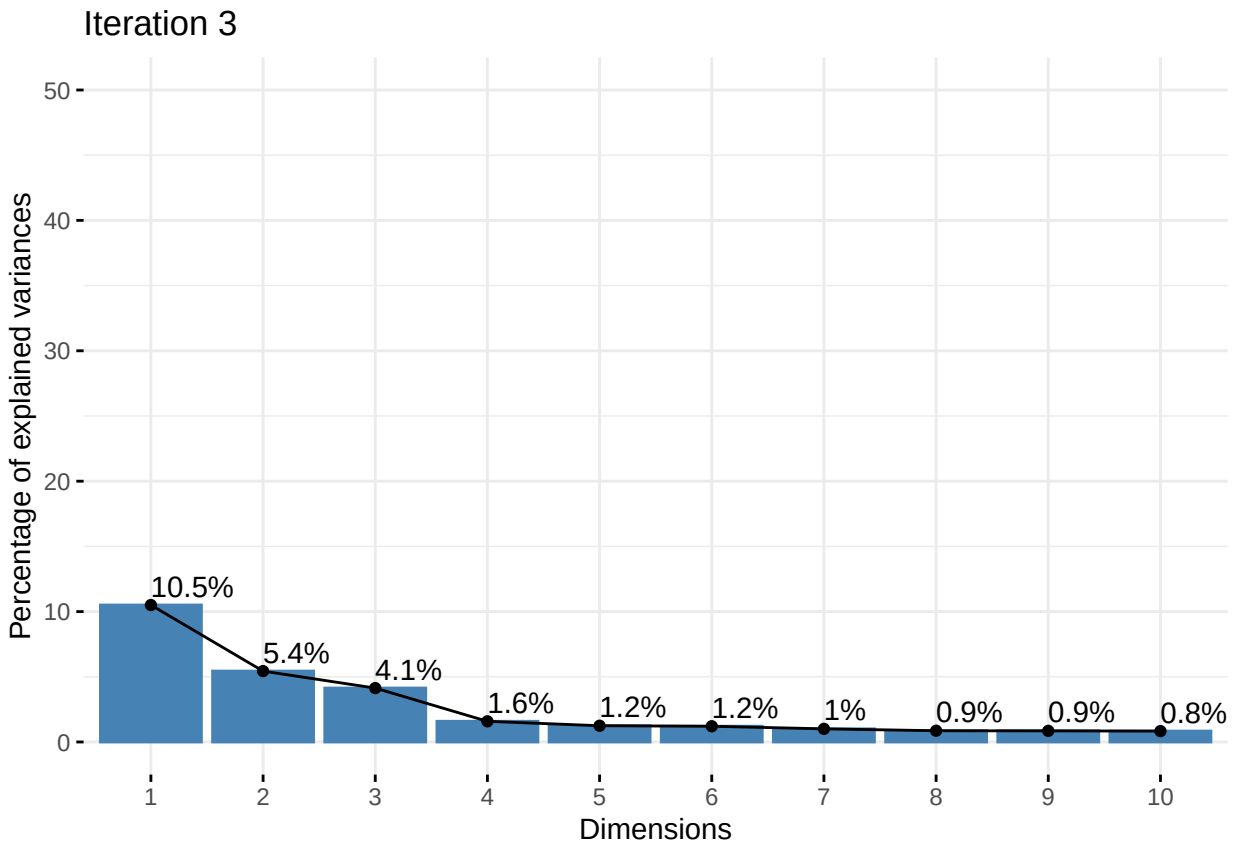
Iteration 2

```
subset2 = cluster_subset(df_subset1)
pca_plot(subset2, "Iteration 2")
```



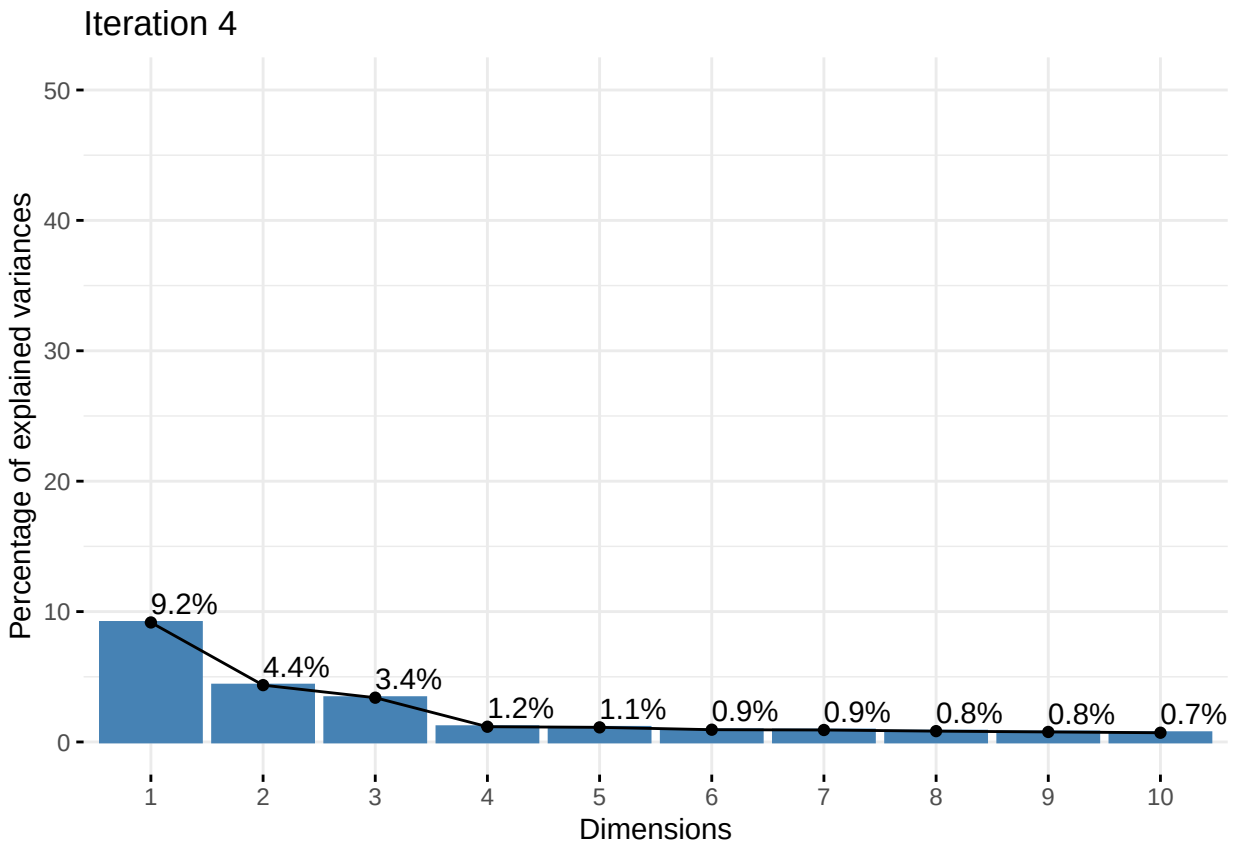
Iteration 3

```
subset3 = cluster_subset(subset2)
pca_plot(subset3, "Iteration 3")
```



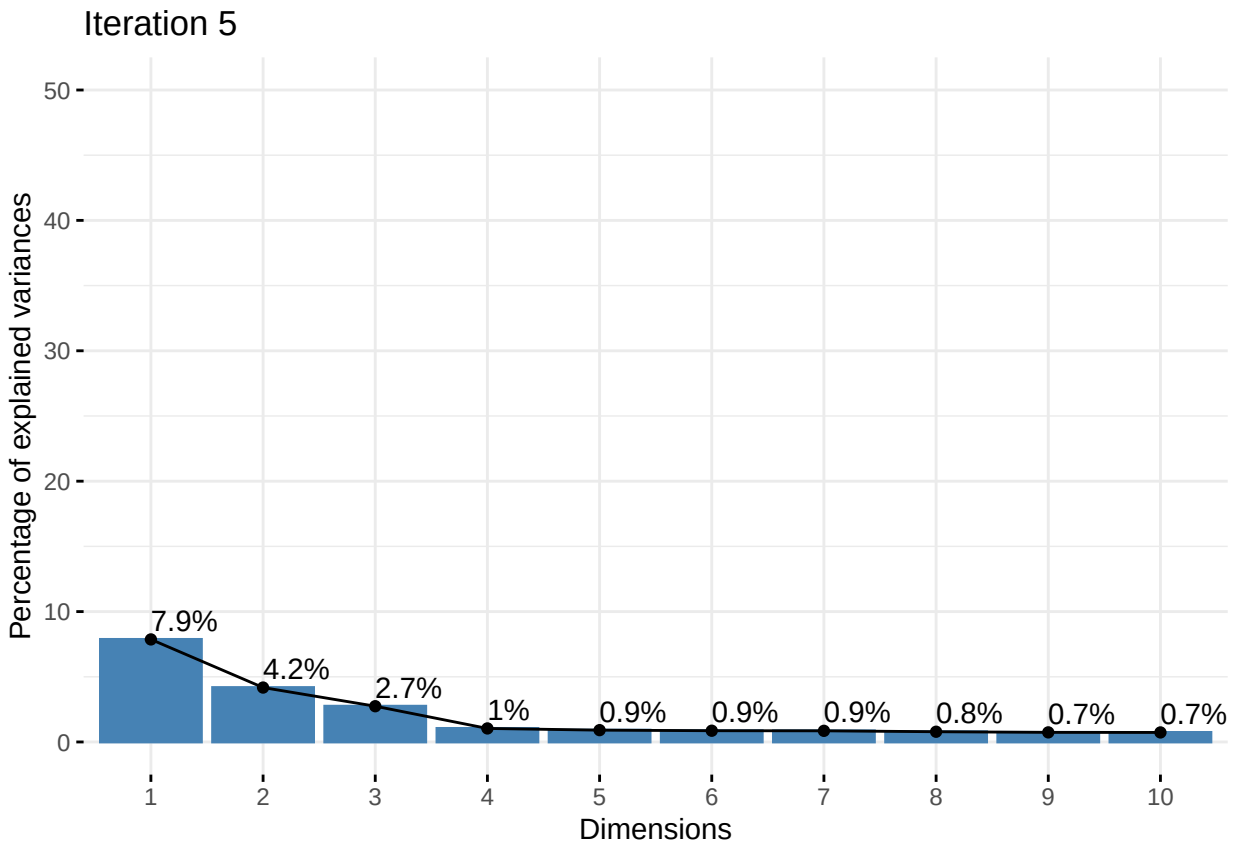
Iteration 4

```
subset4 = cluster_subset(subset3)
pca_plot(subset4, "Iteration 4")
```



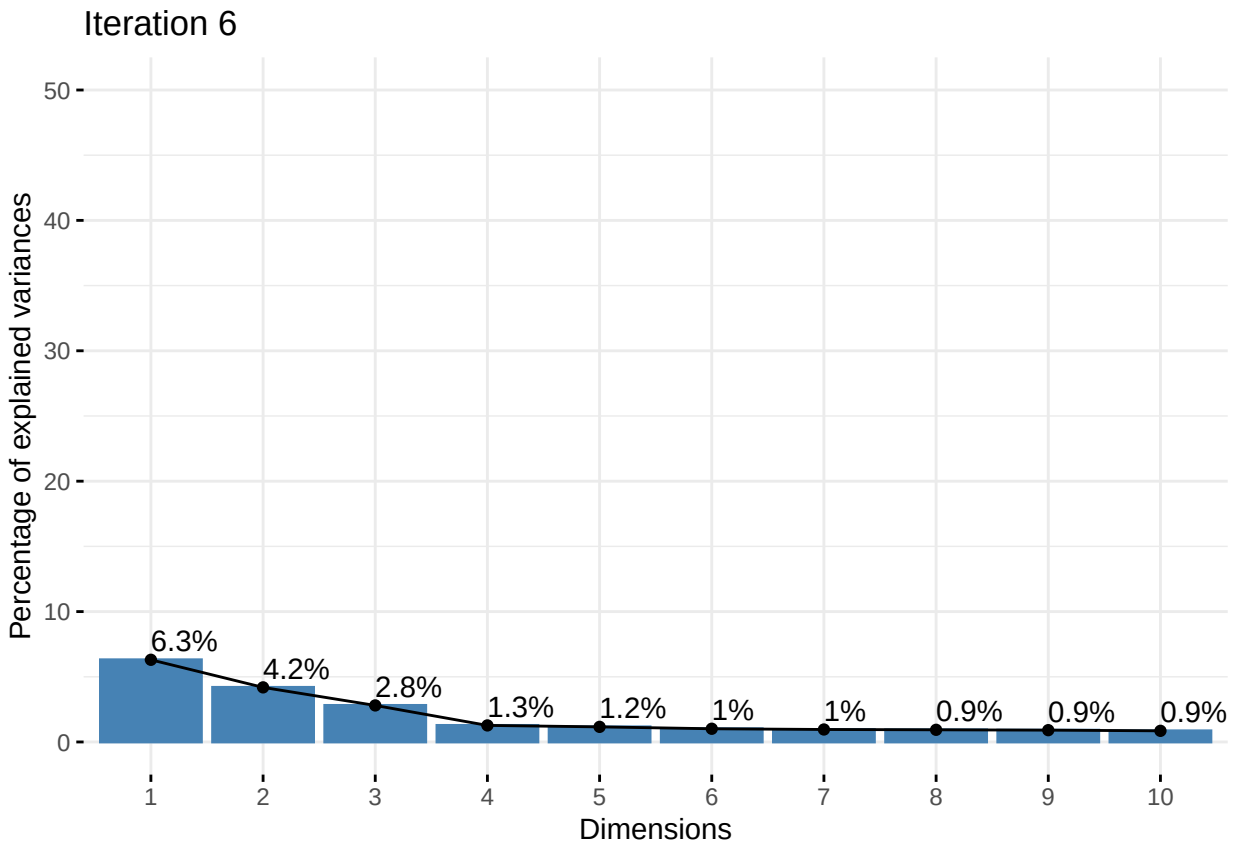
Iteration 5

```
subset5 = cluster_subset(subset4)
pca_plot(subset5, "Iteration 5")
```



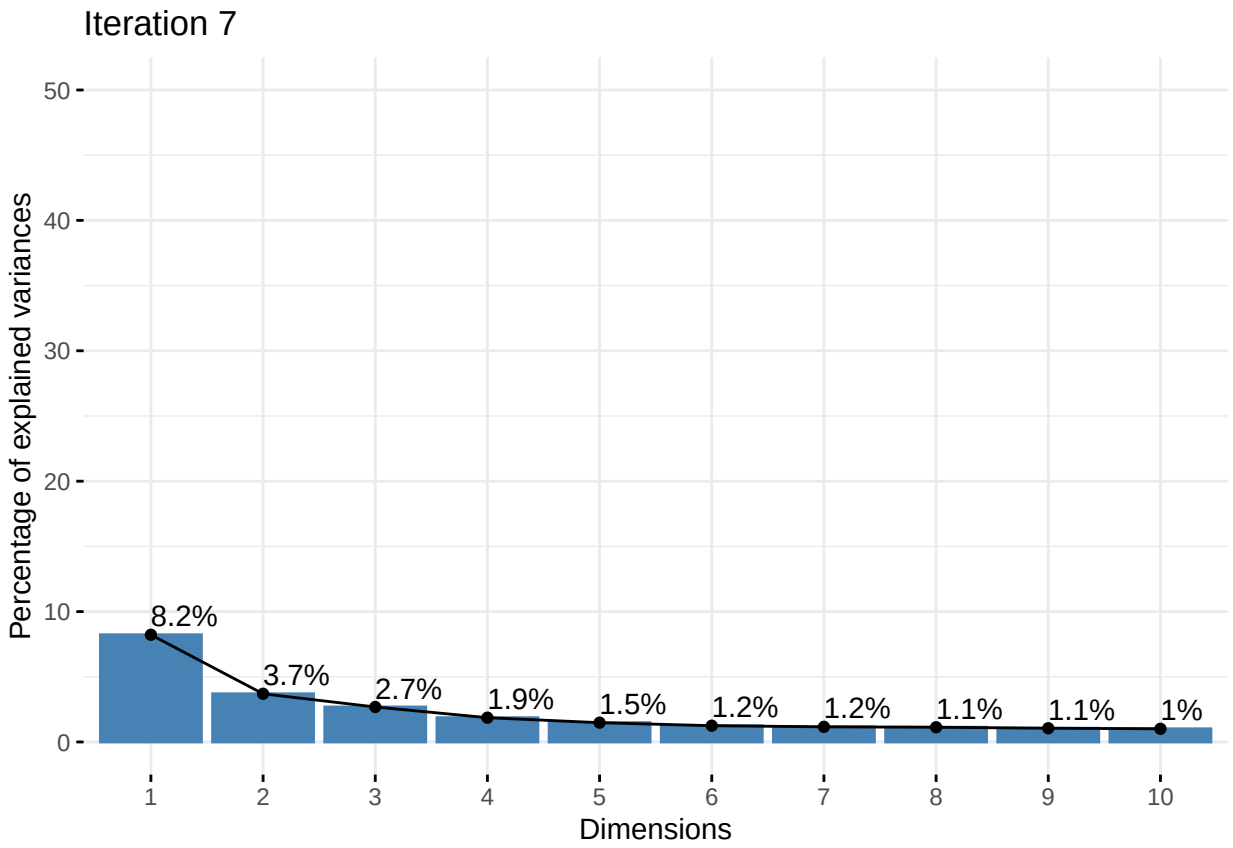
Iteration 6

```
subset6 = cluster_subset(subset5)
pca_plot(subset6, "Iteration 6")
```



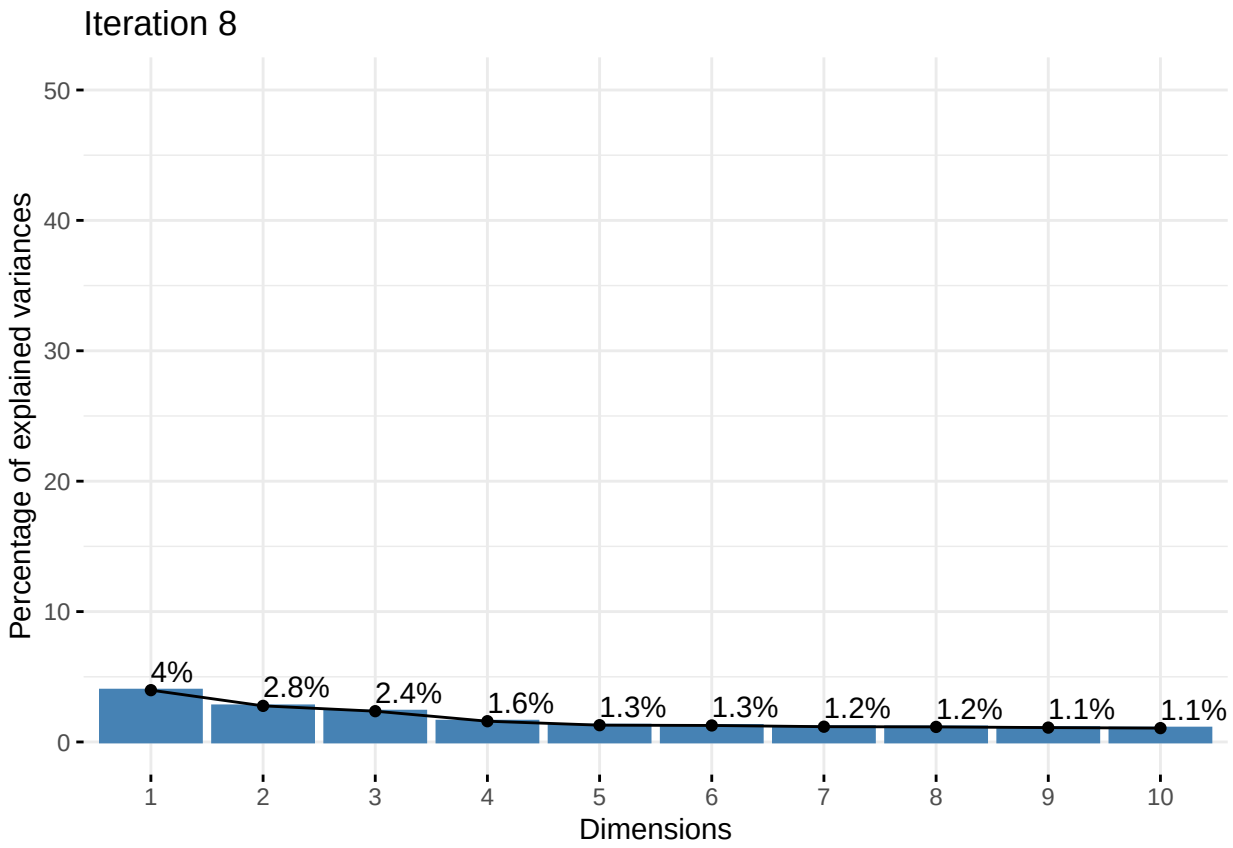
Iteration 7

```
subset7 = cluster_subset(subset6)
pca_plot(subset7, "Iteration 7")
```



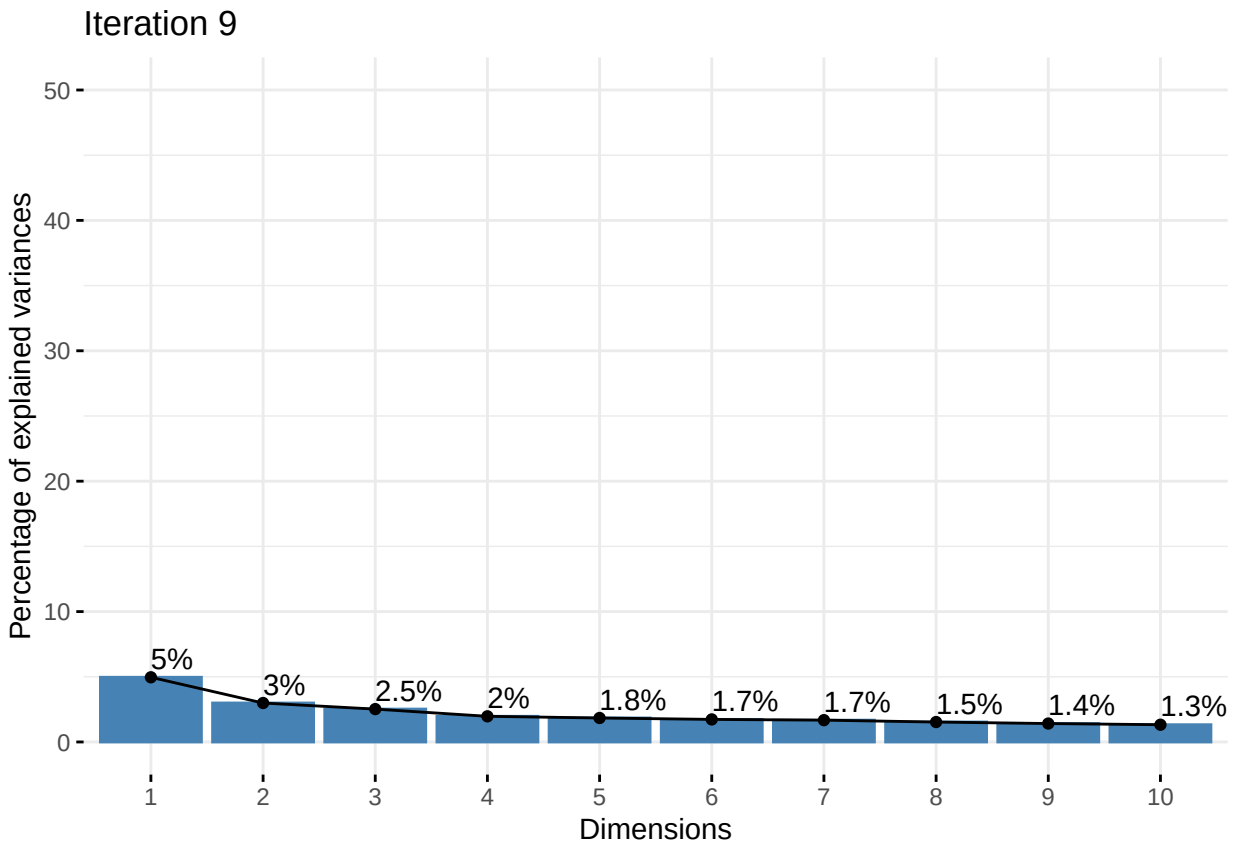
Iteration 8

```
subset8 = cluster_subset(subset7)
pca_plot(subset8, "Iteration 8")
```



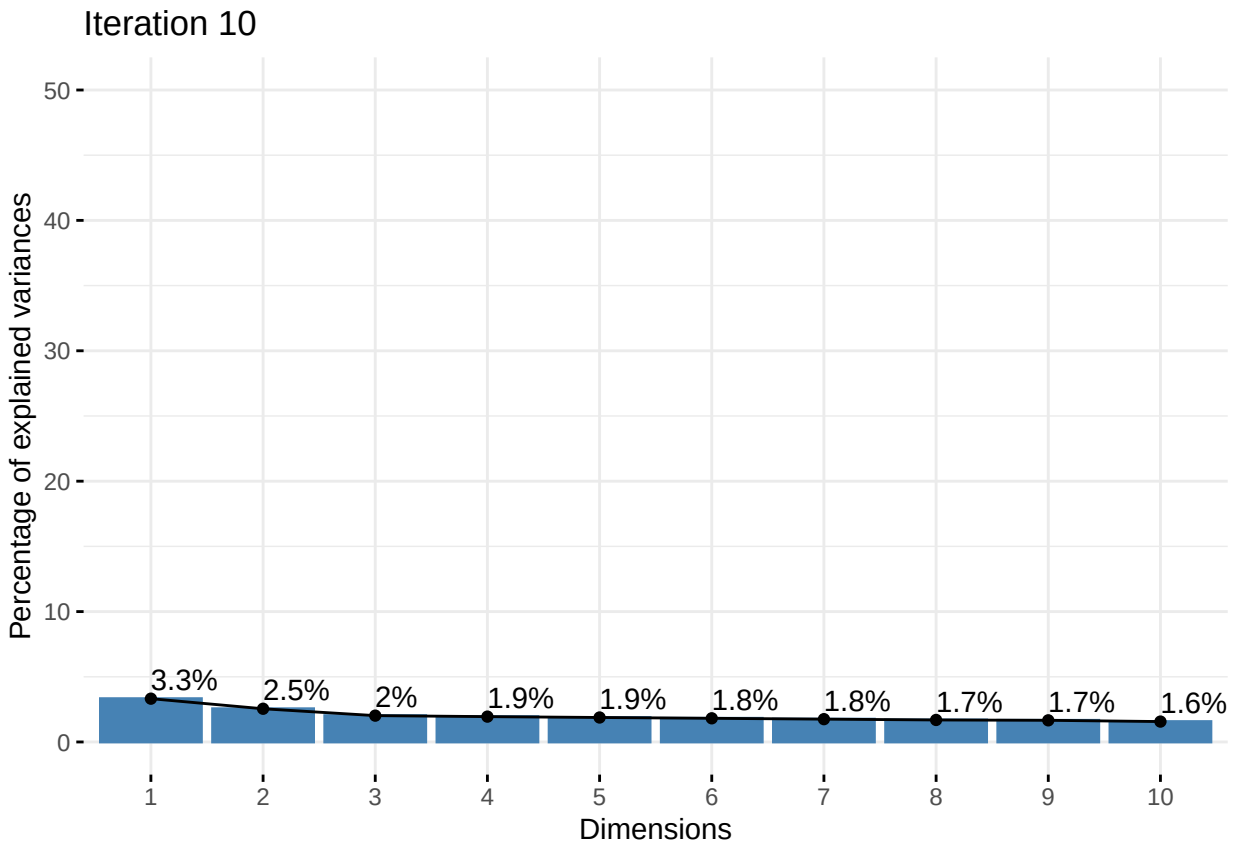
Iteration 9

```
subset9 = cluster_subset(subset8)
pca_plot(subset9, "Iteration 9")
```



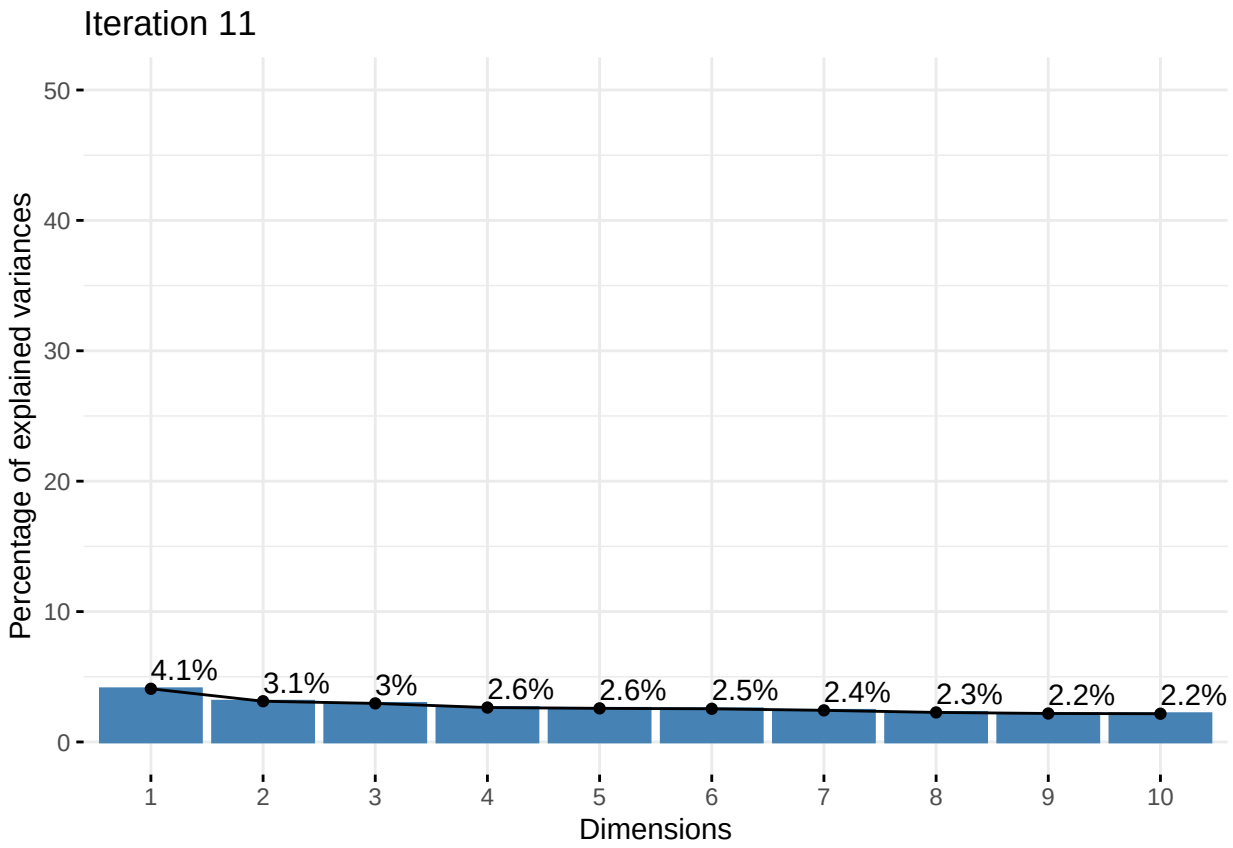
Iteration 10

```
subset10 = cluster_subset(subset9)
pca_plot(subset10, "Iteration 10")
```



Iteration 11

```
subset11 = cluster_subset(subset10)
pca_plot(subset11, "Iteration 11")
```


Iteration 11
Percentage of explained variances
50
40
30
20
10
0
4.1%
3.1%
3%
2.6%
2.6%
2.5%
2.4%
2.3%
2.2%
2.2%
1
2
3
4
5
6
7
8
9
10
Dimensions