AirbnbAnalysis.R

AnujPC

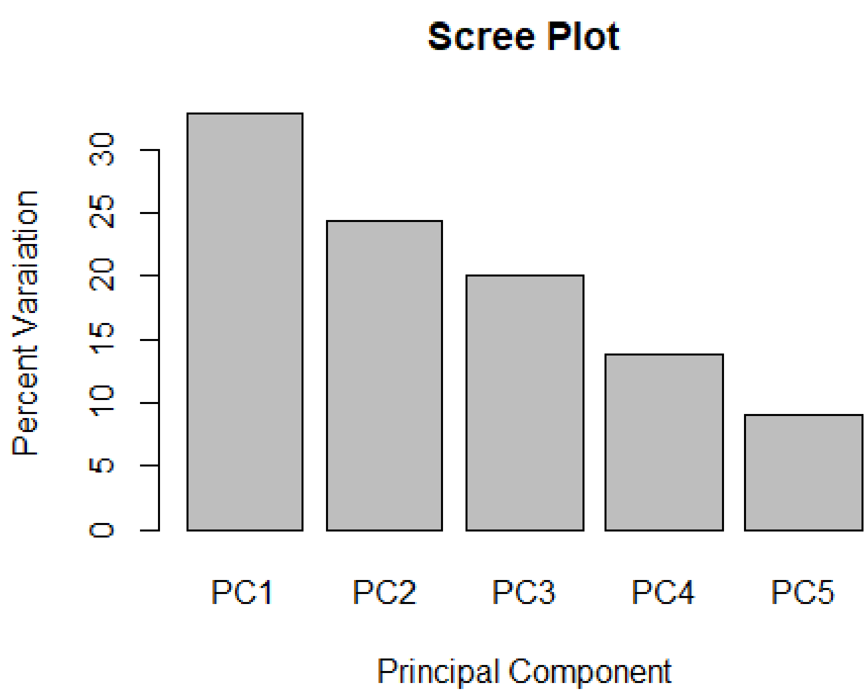
2019-10-17

```
library(data.table)
library(ggplot2)  # tidyverse data visualization package
library(stringr)


#Importing csv file from my local computer
airbnbOriginalDF =read.csv("D:/Priyam/FirstSemester/MVA project/airbnb-host-analysis-for-newyork/Airbnb Host Data For Newyork City.csv")

##Converting data frame to data table
setDT(airbnbOriginalDF)

#Removing values which are null and storing in new table.
airbnbNoNADT = airbnbOriginalDF[airbnbOriginalDF$reviews_per_month != 'NA']

#Converting datatype of last review date to DAte Format.
airbnbNoNADT[,last_review:=as.Date(last_review, '%m/%d/%Y')]

#As the neighbourhood_group column has 5 categorical values, we can factor it, and convert our string data type.
airbnbNoNADT[,neighbourhood_group:= factor(neighbourhood_group)]

#For room type, we get 3 unique categorical values. we can factor it, and convert our string datatype.
airbnbNoNADT[,room_type:= factor(room_type)]

#With earlier analysis/ summary and plot we found few ouliers, therefore that data  we have dropped below, conforming it is not impact our main dataset.
airbnbCleaned = airbnbNoNADT[price<2500 & number_of_reviews<400 & reviews_per_month<10]
##Manhattan area dataset
airbnbManhattan = airbnbCleaned[neighbourhood_group=='Manhattan']
nrow(airbnbManhattan)

## [1] 16584

            ####### PCA ########

library(dplyr)
```

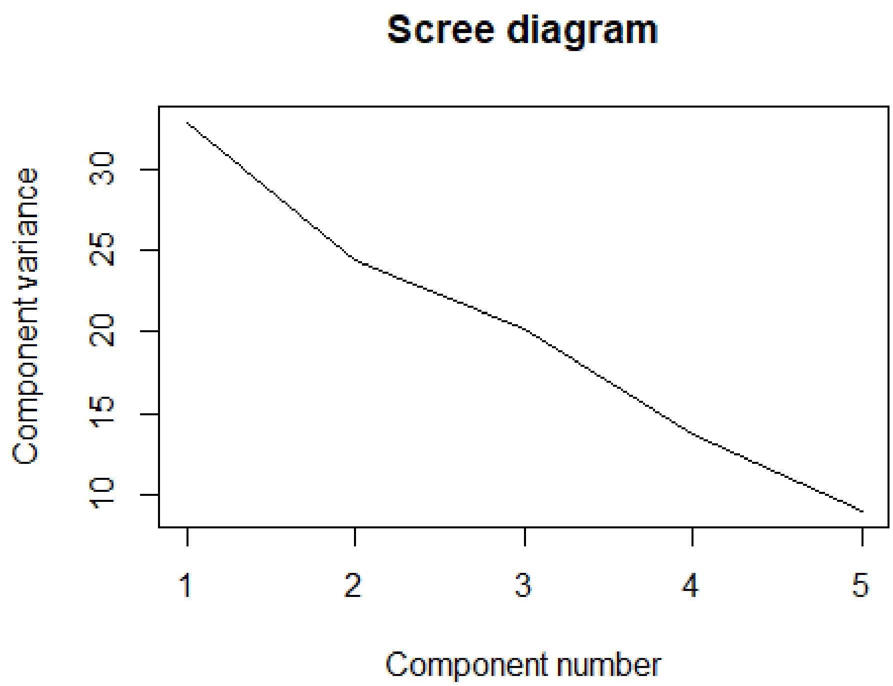
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(data.table)

##Taking the numeric columns that will contribute for variance in data
airbnbManhattanPCA = data.frame(
  airbnbManhattan$id,
  airbnbManhattan$host_id,
  airbnbManhattan$room_type,
  airbnbManhattan$price,
  airbnbManhattan$minimum_nights,
  airbnbManhattan$number_of_reviews,
  airbnbManhattan$reviews_per_month,
  airbnbManhattan$availability_365)

setDT(airbnbManhattanPCA)

##Setting column names for our new dataframe
names(airbnbManhattanPCA) <- c(
  'id',
  'host_id',
  'room_type',
  'price',
  'minimum_nights',
  'number_of_reviews',
  'reviews_per_month',
  'availability_365')

head(airbnbManhattanPCA, 5)

##      id host_id       room_type price minimum_nights number_of_reviews
## 1: 2595    2845 Entire home/apt   225              1                45
## 2: 5022    7192 Entire home/apt    80             10                 9
## 3: 5099    7322 Entire home/apt   200              3                74
## 4: 5203    7490    Private room    79              2               118
## 5: 5238    7549 Entire home/apt   150              1               160
##    reviews_per_month availability_365
## 1:              0.38              355
```

```
## 2:              0.10                0
## 3:              0.59              129
## 4:              0.99                0
## 5:              1.33              188

##Here we have used prcomp function to get Principal components of data
airbnbPC <- prcomp(airbnbManhattanPCA[,-1:-3], scale=TRUE)
airbnbPC

## Standard deviations (1, .., p=5):
## [1] 1.2809561 1.1025899 1.0029067 0.8293291 0.6706998
##
## Rotation (n x k) = (5 x 5):
##                           PC1        PC2          PC3        PC4
## price             -0.02776573 -0.5452081  0.699961309  0.4546543
## minimum_nights     0.14164147 -0.5148879 -0.696443140  0.4744301
## number_of_reviews -0.66002627  0.1113698 -0.145619680  0.2052838
## reviews_per_month -0.66848481  0.1030587 -0.004197668  0.2028235
## availability_365  -0.31090215 -0.6439054 -0.061631210 -0.6963668
##                             PC5
## price             -0.0729439867
## minimum_nights     0.0686378512
## number_of_reviews -0.6990104042
## reviews_per_month  0.7080621380
## availability_365  -0.0006955398

##prcomp() gives three values x, sdev, rotation
names(airbnbPC)

## [1] "sdev"     "rotation" "center"   "scale"    "x"

## x contains principal components for drawing a graph.
##since there are 5 samples(COLUMNS), there are 5 PC

#To get a sense how meaningful this is, let's see how much variation in the
original data PC1 or together with PC2 accounts for
#To do this we require the square of sdev, to see how much variance in the
original data each PC accounts for
##The goal is to draw a graph that shows how the samples are related(not
related)to each other

##Creating eigen values for airbnb (sqaure of sdev)  ----> representing by
pca_var
(pca_var <- airbnbPC$sdev^2)

## [1] 1.6408485 1.2157045 1.0058219 0.6877868 0.4498382

names(pca_var)

## NULL
```

```r
names(pca_var) <- paste("PC",1:5,sep="")
names(pca_var)

## [1] "PC1" "PC2" "PC3" "PC4" "PC5"

pca_var

##       PC1       PC2       PC3       PC4       PC5
## 1.6408485 1.2157045 1.0058219 0.6877868 0.4498382

##Taking sum of all eigen values
sum_var <- sum(pca_var)
sum_var

## [1] 5

##Calculating percentage of variance to better visualize each PC proportion
in data
pcavarpercent <- (pca_var/sum_var)*100
##Visulaization using Bar chart
barplot(pcavarpercent, main="Scree Plot", xlab="Principal Component", ylab =
"Percent Varaiation")
```



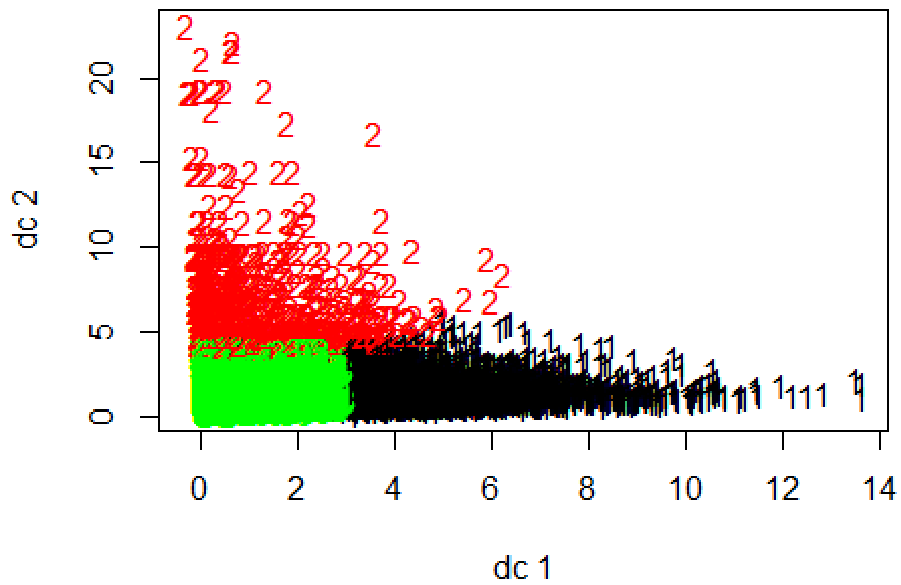

```r
##Visualization using scree plot
plot(pcavarpercent, xlab = "Component number", ylab = "Component variance",
type = "l", main = "Scree diagram")
```




```
##From the plot it can be deciphered that all PC components have good amount
of information with them, approx 80% of variance is presented with PC1,PC2,
PC3, and thus we cannot choose only two PC for dimensionality reduction
##since it will lead to information loss.

##Now we will apply Clustering technique on our data set.
##Since our data set is huge. It has around 16k plus rows, we will go with
Non Hierarichial Clustering.

               ########## K-means Clustering #########
library(cluster)
airbnbManhattanClust = data.frame(
  airbnbManhattan$price,
  airbnbManhattan$number_of_reviews,
  airbnbManhattan$reviews_per_month)

##Making property id as Rownames, so cluster will be formed iwth these
points.
rownames(airbnbManhattanClust) <- airbnbManhattan$id

##Scaling done to make the data on one scale.
scaleManhattan <- scale(airbnbManhattanClust[,1:ncol(airbnbManhattanClust)])

#Here we have selected first row to see how our scaled matrix is like
head(scaleManhattan,1)
```

```
##      airbnbManhattan.price airbnbManhattan.number_of_reviews
## 2595             0.3309951                         0.3986977
##      airbnbManhattan.reviews_per_month
## 2595                        -0.5762157

# We will find K-means by taking k=2, 3, 4, 5, 6...
# Centers (k's) are numbers thus, 10 random sets are chosen

#For 2 clusters, k-means = 2
kmeans2.Manhattan <- kmeans(scaleManhattan,2,nstart = 10)
# Computing the percentage of variation accounted for two clusters
perc_var_kmeans2 <- round(100*(1 -
kmeans2.Manhattan$betweenss/kmeans2.Manhattan$totss),1)
names(perc_var_kmeans2) <- "Perc. 2 clus"
perc_var_kmeans2

## Perc. 2 clus
##         63.5

#For 3 clusters, k-means = 3
kmeans3.Manhattan <- kmeans(scaleManhattan,3,nstart = 10)
# Computing the percentage of variation accounted for three  clusters
perc_var_kmeans3 <- round(100*(1 -
kmeans3.Manhattan$betweenss/kmeans3.Manhattan$totss),1)
names(perc_var_kmeans3) <- "Perc. 3 clus"
perc_var_kmeans3

## Perc. 3 clus
##         46.3

#For 4 clusters, k-means = 4
kmeans4.Manhattan <- kmeans(scaleManhattan,4,nstart = 10)
# Computing the percentage of variation accounted for four clusters
perc_var_kmeans4 <- round(100*(1 -
kmeans4.Manhattan$betweenss/kmeans4.Manhattan$totss),1)
names(perc_var_kmeans4) <- "Perc. 4 clus"
perc_var_kmeans4

## Perc. 4 clus
##         35.5

#From above, after computing percentage of variation for each k means, we
found that k means 3 could be good to preseent our data
# Saving above 3 k-means (1,2,3)  in a list

#Filtering properties which are in 1 cluster of k mean 3
clus1 <- matrix(names(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster ==
1]),
                ncol=1,
nrow=length(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster == 1]))
```

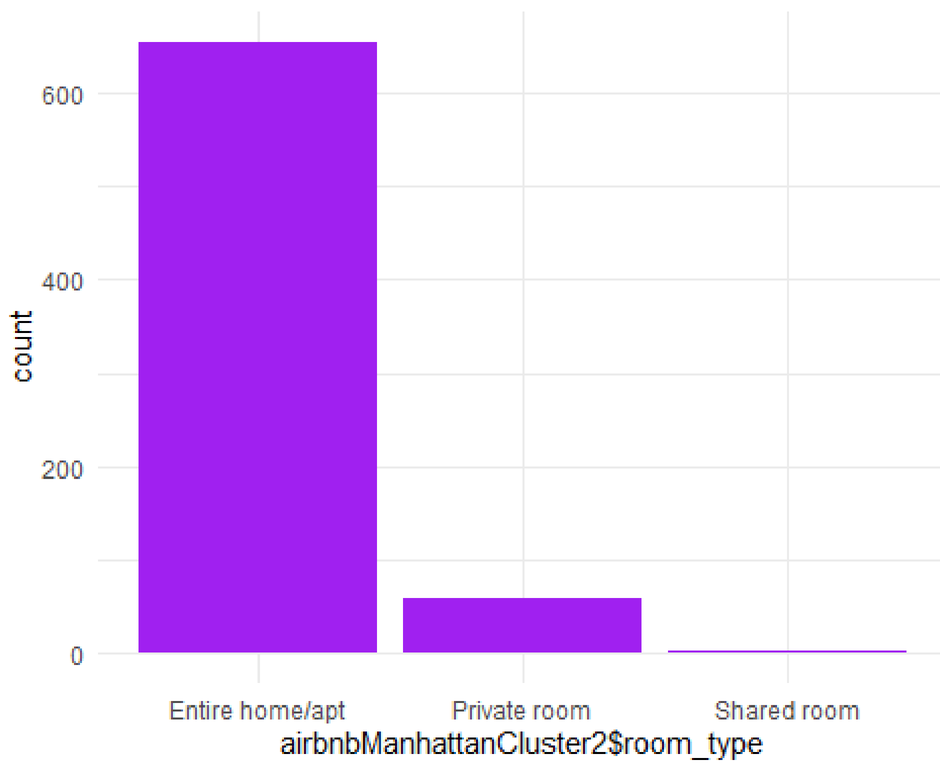
```r
colnames(clus1) <- "Cluster 1"

#Filtering properties which are in 2 cluster of k mean 3
clus2 <- matrix(names(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster ==
2]),
                ncol=1,
nrow=length(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster == 2]))
colnames(clus2) <- "Cluster 2"

#Filtering properties which are in 3 cluster of k mean 3
clus3 <- matrix(names(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster ==
3]),
                ncol=1,
nrow=length(kmeans3.Manhattan$cluster[kmeans3.Manhattan$cluster == 3]))
colnames(clus3) <- "Cluster 3"

head(clus1,5)

##      Cluster 1
## [1,] "5203"
## [2,] "5238"
## [3,] "5441"
## [4,] "6021"
## [5,] "7322"

head(clus2,5)

##      Cluster 2
## [1,] "23686"
## [2,] "60164"
## [3,] "61224"
## [4,] "70609"
## [5,] "174966"

head(clus3,5)

##      Cluster 3
## [1,] "2595"
## [2,] "5022"
## [3,] "5099"
## [4,] "5295"
## [5,] "6090"

##Now we will plot these clusters
library(fpc)
plotcluster(airbnbManhattanClust,kmeans3.Manhattan$cluster)
```




```
##We can make three subsets for three clusters by row filtering
airbnbManhattanCluster1 <- subset(airbnbManhattan, airbnbManhattan$id %in%
clus1)
airbnbManhattanCluster2 <- subset(airbnbManhattan, airbnbManhattan$id %in%
clus2)
airbnbManhattanCluster3 <- subset(airbnbManhattan, airbnbManhattan$id %in%
clus3)

##Tried checking if properties in particular clusters are located in some
specific area in Manhattan
length(unique(airbnbManhattanCluster1$neighbourhood))

## [1] 32

length(unique(airbnbManhattanCluster2$neighbourhood))

## [1] 28

length(unique(airbnbManhattanCluster3$neighbourhood))

## [1] 32

#3We did not get any idea, as all clusters have almost all locations.

##This is to check the mean of 3 clusters
kmeans3.Manhattan$centers
```

```
##   airbnbManhattan.price airbnbManhattan.number_of_reviews
## 1            -0.1646776                         1.4421421
## 2             3.4221035                        -0.2310378
## 3            -0.1516624                        -0.3342583
##   airbnbManhattan.reviews_per_month
## 1                        1.62693947
## 2                       -0.01131017
## 3                       -0.39102892

##We will see average price, average number of reviews , average reviews per month for houses in each cluster to get a better idea of most recommendable properties.
mean(airbnbManhattanCluster1$price)

## [1] 150.0461

mean(airbnbManhattanCluster1$number_of_reviews)

## [1] 92.08769

mean(airbnbManhattanCluster1$reviews_per_month)

## [1] 3.728909

mean(airbnbManhattanCluster2$price)

## [1] 692.4266

mean(airbnbManhattanCluster2$number_of_reviews)

## [1] 16.58182

mean(airbnbManhattanCluster2$reviews_per_month)

## [1] 1.238685

mean(airbnbManhattanCluster3$price)

## [1] 152.0142

mean(airbnbManhattanCluster3$number_of_reviews)

## [1] 11.92377

mean(airbnbManhattanCluster3$reviews_per_month)

## [1] 0.6614934

##From above means , we find that properties in cluter 2 have average price of 150 and avargae no of reviews as 16.
##However for clust 1, avg price is 150 and avg no.of reviews is 11.
## for 3, avg price is 692 and avg no of reviews is 16

##Thus the most recommended properties for people to stay in Manhattan lies
```

```r
in Cluster 2


setDT(airbnbManhattanCluster2)

##Here we are trying to see the top apartment type available in cluster 2.


nrow(airbnbManhattanCluster2[airbnbManhattanCluster2$room_type == 'Entire
home/apt'])

## [1] 655

nrow(airbnbManhattanCluster2[airbnbManhattanCluster2$room_type == 'Private
room'])

## [1] 58

nrow(airbnbManhattanCluster2[airbnbManhattanCluster2$room_type == 'Shared
room'])

## [1] 2

ggplot(airbnbManhattanCluster2, aes(x=airbnbManhattanCluster2$room_type))
+geom_bar(fill ='purple') +theme_minimal()
```




```r
## From this we see, that Entire home/apt and private room are the most
available ones.
```

```r
##Below we have shown the araes in Manhattan which have these properties in Cluster 2.
##There is no soecific location in Manhattan an dis spread out.
ggplot(airbnbManhattanCluster2,
aes(x=airbnbManhattanCluster2$longitude,y=airbnbManhattanCluster2$latitude))
+ geom_point(size=0.1, color = 'dark blue')
```