Clustering\_ananlysis\_on\_Abnb\_Istanbul

Pritesh

3/5/2020

knitr::opts\_chunk$set(echo = TRUE)

library(data.table)

## Warning: package 'data.table' was built under R version 3.6.2

library(fpp)

## Loading required package: forecast

## Warning: package 'forecast' was built under R version 3.6.2

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Loading required package: fma

## Warning: package 'fma' was built under R version 3.6.2

## Loading required package: expsmooth

## Loading required package: lmtest

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.6.2

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Loading required package: tseries

library(fpp2)

## Loading required package: ggplot2

##   
## Attaching package: 'fpp2'

## The following objects are masked from 'package:fpp':  
##   
## ausair, ausbeer, austa, austourists, debitcards, departures,  
## elecequip, euretail, guinearice, oil, sunspotarea, usmelec

library(cowplot)

## Warning: package 'cowplot' was built under R version 3.6.2

##   
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Note: As of version 1.0.0, cowplot does not change the

## default ggplot2 theme anymore. To recover the previous

## behavior, execute:  
## theme\_set(theme\_cowplot())

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.6.2

## -- Attaching packages ---------------------------------------------------- tidyverse 1.3.0 --

## v tibble 2.1.3 v dplyr 0.8.4  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.4.0  
## v purrr 0.3.3

## Warning: package 'tidyr' was built under R version 3.6.2

## Warning: package 'purrr' was built under R version 3.6.2

## Warning: package 'dplyr' was built under R version 3.6.2

## Warning: package 'forcats' was built under R version 3.6.2

## -- Conflicts ------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::between() masks data.table::between()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::first() masks data.table::first()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::last() masks data.table::last()  
## x purrr::transpose() masks data.table::transpose()

library(psych)

## Warning: package 'psych' was built under R version 3.6.2

##   
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

library(e1071)

## Warning: package 'e1071' was built under R version 3.6.2

library(dplyr)  
library(corrplot)

## Warning: package 'corrplot' was built under R version 3.6.2

## corrplot 0.84 loaded

library(GGally)

## Warning: package 'GGally' was built under R version 3.6.2

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

##   
## Attaching package: 'GGally'

## The following object is masked from 'package:dplyr':  
##   
## nasa

## The following object is masked from 'package:fma':  
##   
## pigs

library(reshape2)

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

## The following objects are masked from 'package:data.table':  
##   
## dcast, melt

AirbnbIstanbul <- read.csv("C:/Pritesh/Rutgers/Courses/Projects/MVA/Dataset/AirbnbIstanbul.csv", stringsAsFactors=FALSE)  
Istanbul <- copy(AirbnbIstanbul)  
class(Istanbul)

## [1] "data.frame"

setDT(Istanbul)  
  
str(Istanbul)

## Classes 'data.table' and 'data.frame': 16251 obs. of 16 variables:  
## $ id : int 4826 20815 25436 27271 28277 28308 28318 29241 30697 33368 ...  
## $ name : chr "The Place" "The Bosphorus from The Comfy Hill" "House for vacation rental furnutare" "LOVELY APT. IN PERFECT LOCATION" ...  
## $ host\_id : int 6603 78838 105823 117026 121607 121695 121721 125742 132137 135136 ...  
## $ host\_name : chr "Kaan" "GÃ¼lder" "Yesim" "Mutlu" ...  
## $ neighbourhood\_group : logi NA NA NA NA NA NA ...  
## $ neighbourhood : chr "Uskudar" "Besiktas" "Besiktas" "Beyoglu" ...  
## $ latitude : num 41.1 41.1 41.1 41 41 ...  
## $ longitude : num 29.1 29 29 29 29 ...  
## $ room\_type : chr "Entire home/apt" "Entire home/apt" "Entire home/apt" "Entire home/apt" ...  
## $ price : int 554 100 211 237 591 237 633 264 596 295 ...  
## $ minimum\_nights : int 1 30 21 5 3 1 3 3 1 2 ...  
## $ number\_of\_reviews : int 1 41 0 2 0 0 0 0 1 1 ...  
## $ last\_review : chr "2009-06-01" "2018-11-07" "" "2018-05-04" ...  
## $ reviews\_per\_month : num 0.01 0.38 NA 0.04 NA NA NA NA 0.01 0.02 ...  
## $ calculated\_host\_listings\_count: int 1 2 1 1 13 1 1 1 1 2 ...  
## $ availability\_365 : int 365 49 83 228 356 365 365 365 365 232 ...  
## - attr(\*, ".internal.selfref")=<externalptr>

Istanbul[,room\_type:=factor(room\_type)]  
Istanbul[,neighbourhood:=factor(neighbourhood)]  
Istanbul[,last\_review:=as.Date(last\_review,'%Y-%m-%d')] ## converting last\_review to date datatype  
  
# datatypes looks better now. hence will see again for NA values   
grep ('NA',Istanbul) # 2, 5, 13 and 14 column have NA values

## [1] 2 5 13 14

Istanbul[is.na(neighbourhood\_group),NROW(neighbourhood\_group)] # entire obs. is blank, will drop this var

## [1] 16251

Istanbul[is.na(last\_review),NROW(last\_review)] ## there are 8484 NA values

## [1] 8484

Istanbul[is.na(reviews\_per\_month),NROW(reviews\_per\_month)] ## there are 8484 NA values

## [1] 8484

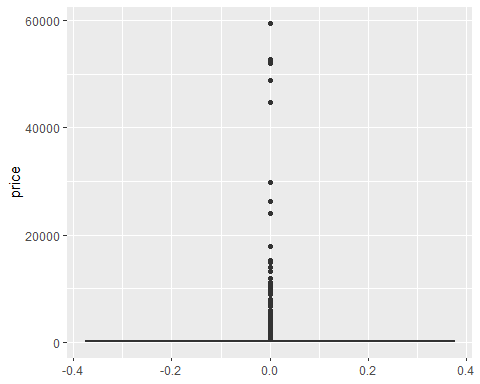
Istanbul$neighbourhood\_group <- NULL ## removing neighbourhood\_group column  
Istanbul[is.na(reviews\_per\_month),reviews\_per\_month:=0] ## nearly 50% of the dataset is filled with NA.  
# hence we can't simply remove these many rows. Hence imputing with 0 values.  
  
  
range(Istanbul$price) ## range of price

## [1] 0 59561

avgNeighbourhood=Istanbul[,avgneighprice:=mean(price),by=neighbourhood]  
summary(Istanbul$price)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0 105.0 190.0 354.7 327.0 59561.0

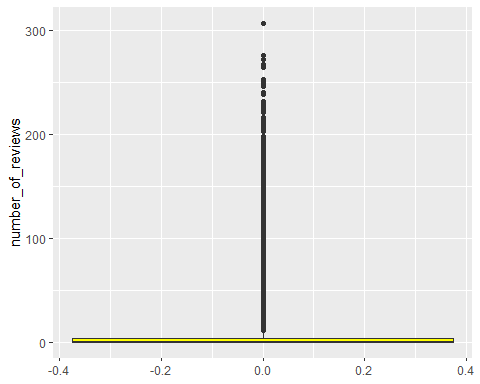
ggplot(Istanbul,aes(y=price)) + geom\_boxplot(fill='yellow') # the boxplot shows that most of the units have price less than 10000



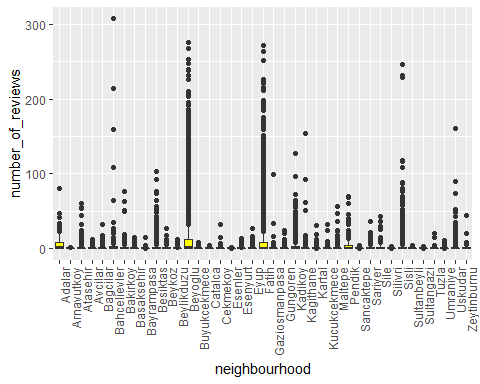
## no. of reviews and neighbourhood relation  
summary(Istanbul$number\_of\_reviews)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 0.000 7.187 4.000 307.000

ggplot(Istanbul,aes(y=number\_of\_reviews)) + geom\_boxplot(fill='yellow')



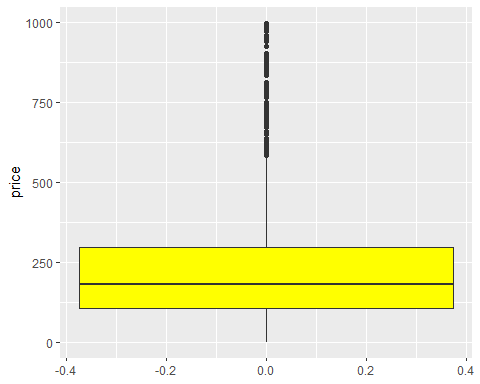
ggplot(Istanbul,aes(x=neighbourhood,y=number\_of\_reviews)) + geom\_boxplot(fill='yellow') + theme(axis.text.x = element\_text(angle = 90, hjust = 1))



nrow(Istanbul[price > 1000]) ## price > 1000, there are only 613 units out of ~16000 which have price > 1000

## [1] 613

# hence we'll remove those.   
Istanbul.clust <- Istanbul[price < 1000 & number\_of\_reviews > 0] ## price > 1000  
ggplot(Istanbul.clust,aes(y=price)) + geom\_boxplot(fill='yellow') # ggplot looks better now



########## K-means Clustering #########  
library(cluster)

## Warning: package 'cluster' was built under R version 3.6.2

Istanbul\_clus = data.frame(  
 Istanbul.clust$price,  
 Istanbul.clust$number\_of\_reviews,  
 Istanbul.clust$reviews\_per\_month)  
  
# Making property id as rownames, clusters will be formed with id as observations.  
rownames(Istanbul\_clus) <- Istanbul.clust$id  
##Scaling done to make the data on one scale.  
Istanbul.Scale <- scale(Istanbul\_clus[,1:3])  
#Here we have selected first row to see how our scaled matrix is like  
head(Istanbul.Scale,1)

## Istanbul.clust.price Istanbul.clust.number\_of\_reviews  
## 4826 1.9566 -0.4855794  
## Istanbul.clust.reviews\_per\_month  
## 4826 -0.8383381

# 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  
set.seed(123)  
kmeans2.Istanbul <- kmeans(Istanbul.Scale,2,nstart = 10)  
# Computing the percentage of variation accounted for two clusters  
perc\_var\_kmeans2 <- round(100\*(1 - kmeans2.Istanbul$betweenss/kmeans2.Istanbul$totss),1)  
names(perc\_var\_kmeans2) <- "Perc. 2 clus"  
perc\_var\_kmeans2

## Perc. 2 clus   
## 66.8

# Computing the percentage of variation accounted for. Three clusters  
kmeans3.Istanbul <- kmeans(Istanbul.Scale,3,nstart = 10)  
perc.var.3 <- round(100\*(1 - kmeans3.Istanbul$betweenss/kmeans3.Istanbul$totss),1)  
names(perc.var.3) <- "Perc. 3 clus"  
perc.var.3

## Perc. 3 clus   
## 47.3

# Computing the percentage of variation accounted for. Four clusters  
kmeans4.Istanbul <- kmeans(Istanbul.Scale,4,nstart = 10)  
perc.var.4 <- round(100\*(1 - kmeans4.Istanbul$betweenss/kmeans4.Istanbul$totss),1)  
names(perc.var.4) <- "Perc. 4 clus"  
perc.var.4

## Perc. 4 clus   
## 35.1

# Computing the percentage of variation accounted for. Five clusters  
kmeans5.Istanbul <- kmeans(Istanbul.Scale,5,nstart = 10)  
perc.var.5 <- round(100\*(1 - kmeans5.Istanbul$betweenss/kmeans5.Istanbul$totss),1)  
names(perc.var.5) <- "Perc. 5 clus"  
perc.var.5

## Perc. 5 clus   
## 29.9

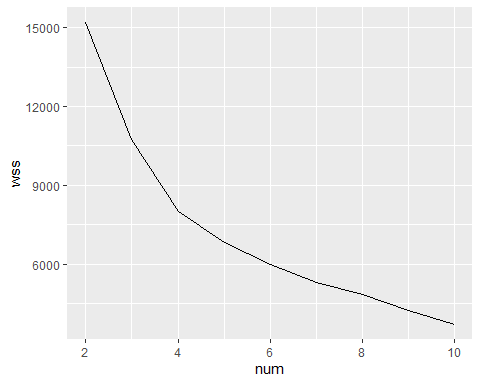
# Computing the percentage of variation accounted for. Six clusters  
kmeans6.Istanbul <- kmeans(Istanbul.Scale,6,nstart = 10)  
perc.var.6 <- round(100\*(1 - kmeans6.Istanbul$betweenss/kmeans6.Istanbul$totss),1)  
names(perc.var.6) <- "Perc. 6 clus"  
perc.var.6

## Perc. 6 clus   
## 25.2

#  
  
# Identify the Best K Clusters  
wss=c()########## empty vector to hold wss  
for(i in 2:10)#### from 2 to 10 cluster  
{  
 km = kmeans(Istanbul.Scale[,1:3],i)  
 wss[i-1]=km$tot.withinss  
}  
wss

## [1] 15197.254 10745.783 7987.996 6808.887 5980.367 5311.900 4846.853  
## [8] 4240.790 3709.000

elbowdt = data.table(num=2:10,wss)  
ggplot(elbowdt,aes(x=num,y=wss)) + geom\_line()



elbowdt

## num wss  
## 1: 2 15197.254  
## 2: 3 10745.783  
## 3: 4 7987.996  
## 4: 5 6808.887  
## 5: 6 5980.367  
## 6: 7 5311.900  
## 7: 8 4846.853  
## 8: 9 4240.790  
## 9: 10 3709.000

# For k = 6 the between sum of square/total sum of square ratio tends to change slowly   
# and remain less changing as compared to others. Therefore, k = 6 should be a good choice for the number of clusters.  
  
# Saving six k-means clusters in a list  
clus1 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 1]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 1]))  
  
colnames(clus1) <- "Cluster 1"  
  
clus2 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 2]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 2]))  
colnames(clus2) <- "Cluster 2"  
  
clus3 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 3]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 3]))  
colnames(clus3) <- "Cluster 3"  
  
clus4 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 4]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 4]))  
colnames(clus4) <- "Cluster 4"  
  
clus5 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 5]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 5]))  
colnames(clus5) <- "Cluster 5"  
  
clus6 <- matrix(names(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 6]),   
 ncol=1, nrow=length(kmeans6.Istanbul$cluster[kmeans6.Istanbul$cluster == 6]))  
colnames(clus6) <- "Cluster 6"  
  
  
#list(clus1,clus2,clus3,clus4,clus5,clus6)  
  
Istanbul\_clus\_Out <- cbind(Istanbul\_clus, clusterNumber = kmeans6.Istanbul$cluster)  
  
class(Istanbul\_clus\_Out)

## [1] "data.frame"

setDT(Istanbul\_clus\_Out)  
  
Istanbul\_cluster1 <- Istanbul\_clus\_Out[clusterNumber == 1]  
Istanbul\_cluster2 <- Istanbul\_clus\_Out[clusterNumber == 2]  
Istanbul\_cluster3 <- Istanbul\_clus\_Out[clusterNumber == 3]  
Istanbul\_cluster4 <- Istanbul\_clus\_Out[clusterNumber == 4]  
Istanbul\_cluster5 <- Istanbul\_clus\_Out[clusterNumber == 5]  
Istanbul\_cluster6 <- Istanbul\_clus\_Out[clusterNumber == 6]  
  
names(Istanbul\_cluster1) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
names(Istanbul\_cluster2) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
names(Istanbul\_cluster3) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
names(Istanbul\_cluster4) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
names(Istanbul\_cluster5) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
names(Istanbul\_cluster6) <- c("price","number\_of\_reviews","reviews\_per\_month","clusterNumber")  
  
head(Istanbul\_cluster1)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 142 13 3.64 1  
## 2: 185 17 3.81 1  
## 3: 190 54 5.47 1  
## 4: 322 21 2.83 1  
## 5: 448 52 3.70 1  
## 6: 369 36 3.32 1

mean(Istanbul\_cluster1$price)

## [1] 206.7967

mean(Istanbul\_cluster1$number\_of\_reviews)

## [1] 25.51636

mean(Istanbul\_cluster1$reviews\_per\_month)

## [1] 3.958248

head(Istanbul\_cluster2)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 237 2 0.04 2  
## 2: 295 1 0.02 2  
## 3: 237 8 0.15 2  
## 4: 359 37 0.59 2  
## 5: 353 46 0.45 2  
## 6: 248 6 0.92 2

mean(Istanbul\_cluster2$price)

## [1] 322.6325

mean(Istanbul\_cluster2$number\_of\_reviews)

## [1] 6.37672

mean(Istanbul\_cluster2$reviews\_per\_month)

## [1] 0.4308716

head(Istanbul\_cluster3)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 554 1 0.01 3  
## 2: 596 1 0.01 3  
## 3: 501 20 0.24 3  
## 4: 738 1 0.01 3  
## 5: 533 34 0.39 3  
## 6: 791 3 0.03 3

mean(Istanbul\_cluster3$price)

## [1] 643.1644

mean(Istanbul\_cluster3$number\_of\_reviews)

## [1] 10.37329

mean(Istanbul\_cluster3$reviews\_per\_month)

## [1] 0.6058733

head(Istanbul\_cluster4)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 232 74 0.79 4  
## 2: 322 81 0.99 4  
## 3: 158 83 0.88 4  
## 4: 90 54 0.59 4  
## 5: 264 74 0.84 4  
## 6: 53 56 0.62 4

mean(Istanbul\_cluster4$price)

## [1] 186.7068

mean(Istanbul\_cluster4$number\_of\_reviews)

## [1] 27.27365

mean(Istanbul\_cluster4$reviews\_per\_month)

## [1] 1.663276

head(Istanbul\_cluster5)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 100 41 0.38 5  
## 2: 158 10 0.09 5  
## 3: 105 11 0.21 5  
## 4: 179 16 0.19 5  
## 5: 132 33 0.36 5  
## 6: 105 6 0.07 5

mean(Istanbul\_cluster5$price)

## [1] 121.8919

mean(Istanbul\_cluster5$number\_of\_reviews)

## [1] 4.596459

mean(Istanbul\_cluster5$reviews\_per\_month)

## [1] 0.4090354

head(Istanbul\_cluster6)

## price number\_of\_reviews reviews\_per\_month clusterNumber  
## 1: 295 128 1.38 6  
## 2: 232 119 1.66 6  
## 3: 316 99 1.10 6  
## 4: 364 113 1.30 6  
## 5: 58 106 1.26 6  
## 6: 100 211 2.58 6

mean(Istanbul\_cluster6$price)

## [1] 260.4667

mean(Istanbul\_cluster6$number\_of\_reviews)

## [1] 137.0222

mean(Istanbul\_cluster6$reviews\_per\_month)

## [1] 2.736667

## From observing the mean price and no. of reviews for the all the six clusters,   
## cluster with mean price of 260 and average rating 137 is the best choice for customers  
  
##Now we will plot these clusters   
library(fpc)

## Warning: package 'fpc' was built under R version 3.6.3

plotcluster(Istanbul\_clus,kmeans6.Istanbul$cluster)

