K - means and Hierarchical clustering

Code

Manuel Romero data taken from: https://www.kaggle.com/datasets/stevezhenghp/airbnb-price-prediction (https://www.kaggle.com/datasets/stevezhenghp/airbnb-price-prediction)

train.csv using as refrence code from book and github:

https://github.com/kjmazidi/Machine_Learning_2nd_edition/blob/master/Part_4_Search_Similarity/13-2-cluster_kmean_wine.pdf (https://github.com/kjmazidi/Machine_Learning_2nd_edition/blob/master/Part_4_Search_Similarity/13-2-cluster_kmean_wine.pdf)

reading the data into dfO - df original

Hide

```
dfO <- read.csv("/Users/fernandoromero/Documents/train.csv", na.strings = "NA", header =TRUE, nrows = 15000)
head(dfO)
```

	id	log_price	property_type	room_type
	<int>	<dbl>	<chr>	<chr>
1	6901257	5.010635	Apartment	Entire home/apt
2	6304928	5.129899	Apartment	Entire home/apt
3	7919400	4.976734	Apartment	Entire home/apt
4	13418779	6.620073	House	Entire home/apt
5	3808709	4.744932	Apartment	Entire home/apt
6	12422935	4.442651	Apartment	Private room

6 rows | 1-5 of 29 columns

Hide

```
names(dfO)
```

```
 [1] "id"                    "log_price"             "property_type"
 [4] "room_type"             "amenities"             "accommodates"
 [7] "bathrooms"             "bed_type"              "cancellation_policy"
[10] "cleaning_fee"          "city"                  "description"
[13] "first_review"          "host_has_profile_pic"  "host_identity_verified"
[16] "host_response_rate"    "host_since"            "instant_bookable"
[19] "last_review"           "latitude"              "longitude"
[22] "name"                  "neighbourhood"         "number_of_reviews"
[25] "review_scores_rating"  "thumbnail_url"         "zipcode"
[28] "bedrooms"              "beds"
```

Hide

```
nrow(df0)
```

```
[1] 15000
```

cleaning the data checking if there are any NA's

Hide

```
sapply(df0, function(x) sum(is.na(x)==TRUE))
```

```
                  id             log_price         property_type             room_type
                   0                     0                     0                     0
           amenities          accommodates             bathrooms             bed_type
                   0                     0                    46                     0
  cancellation_policy          cleaning_fee                  city           description
                   0                     0                     0                     0
        first_review host_has_profile_pic host_identity_verified     host_response_rate
                   0                     0                     0                     0
         host_since      instant_bookable           last_review              latitude
                   0                     0                     0                     0
           longitude                  name         neighbourhood      number_of_reviews
                   0                     0                     0                     0
 review_scores_rating         thumbnail_url               zipcode              bedrooms
                3387                     0                     0                    13
                beds
                  23
```

this data has many NA's but we still have plenty of data after removing them so we will remove those rows making columns that have strings factors and then making them ints removing columns that i wont be using

Hide

```
df0 <- df0[complete.cases(df0),]
head(df0)
```

	id	log_price	property_type	room_type
	<int>	<dbl>	<chr>	<chr>
1	6901257	5.010635	Apartment	Entire home/apt
2	6304928	5.129899	Apartment	Entire home/apt
3	7919400	4.976734	Apartment	Entire home/apt
5	3808709	4.744932	Apartment	Entire home/apt
6	12422935	4.442651	Apartment	Private room
7	11825529	4.418841	Apartment	Entire home/apt

6 rows | 1-5 of 29 columns

Hide

```
df0$property_type <- as.numeric(as.factor(df0$property_type))
df0$room_type <- as.numeric(as.factor(df0$room_type))
df0$bed_type <- as.numeric(as.factor(df0$bed_type))
df0$cancellation_policy <- as.numeric(as.factor(df0$cancellation_policy))
df0$cleaning_fee <- as.numeric(as.factor(df0$cleaning_fee))
df0$cleaning_fee <- as.numeric(as.factor(df0$cleaning_fee))
df0$host_has_profile_pic <- as.numeric(as.factor(df0$host_has_profile_pic))
df0$host_identity_verified <- as.numeric(as.factor(df0$host_identity_verified))
df0$property_type <- as.numeric(as.factor(df0$property_type))

df0 <- df0[ , unlist(lapply(df0,is.numeric))]
df0 <- df0[,c(2,3,4,5,6,7,8,9,10,11,14,15,16,17)]
```

Now we will scale the data and keep the original data set for later use put the scaled data into df and then print out the head to see the data scaled

Hide

```
df <- scale(df0[])
head(df)
```

```
     log_price property_type  room_type accommodates  bathrooms  bed_type cancellation_policy
1  0.389431001    -0.6818272 -0.8243209   -0.1037054 -0.4003147 0.1527278           0.9137656
2  0.567531522    -0.6818272 -0.8243209    1.7576821 -0.4003147 0.1527278           0.9137656
3  0.338804548    -0.6818272 -0.8243209    0.8269884 -0.4003147 0.1527278          -0.3318673
5 -0.007353459    -0.6818272 -0.8243209   -0.5690523 -0.4003147 0.1527278          -0.3318673
6 -0.458760775    -0.6818272  1.0035005   -0.5690523 -0.4003147 0.1527278           0.9137656
7 -0.494318106    -0.6818272 -0.8243209   -0.1037054 -0.4003147 0.1527278          -0.3318673
  cleaning_fee host_has_profile_pic host_identity_verified number_of_reviews review_scores_rating
1    0.4953007           0.06167229              0.6050691        -0.6044981            0.7426474
2    0.4953007           0.06167229             -1.6064441        -0.5074101           -0.1286073
3    0.4953007           0.06167229              0.6050691        -0.4103221           -0.2530723
5    0.4953007           0.06167229              0.6050691        -0.5559541           -6.7252506
6    0.4953007           0.06167229              0.6050691        -0.5802261            0.7426474
7    0.4953007           0.06167229             -1.6064441        -0.2889622            0.3692525
    bedrooms       beds
1 -0.3034659 -0.5865974
2  2.0911410  0.9999724
3 -0.3034659  0.9999724
5 -1.5007694 -0.5865974
6 -0.3034659 -0.5865974
7 -0.3034659 -0.5865974
```

checking if any NA's which there shouldnt be any but NbCluster was giving issues but that may be some other error vs actual empty rows/ columns

Hide

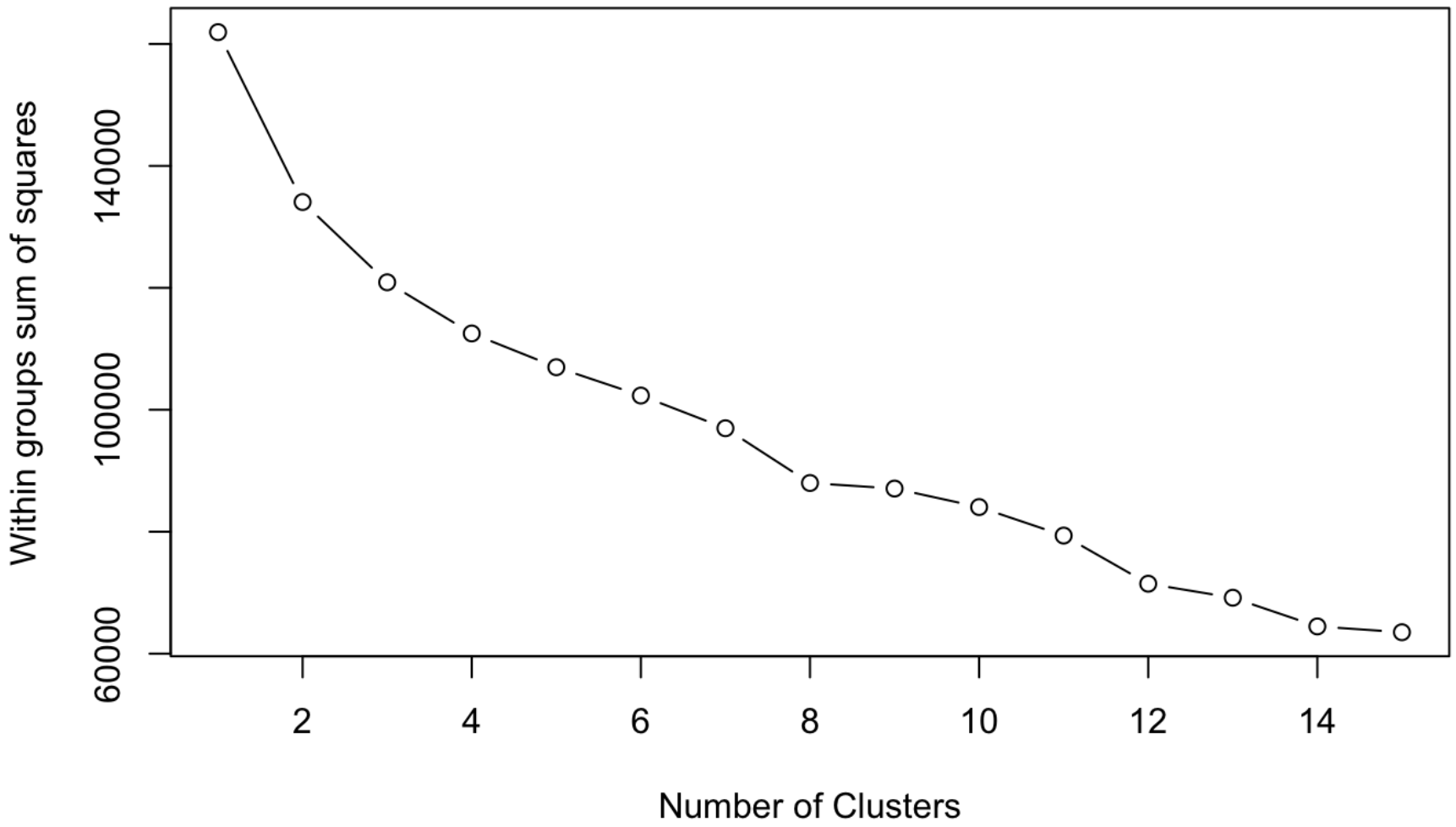
```
colSums(is.na(df))
```

```
            log_price          property_type              room_type           accommodates
                    0                      0                      0                      0
            bathrooms              bed_type    cancellation_policy           cleaning_fee
                    0                      0                      0                      0
  host_has_profile_pic host_identity_verified      number_of_reviews   review_scores_rating
                    0                      0                      0                      0
             bedrooms                   beds
                    0                      0
```

using function from github example to plot the within-groups sums of squares vs number of clusters

Hide

```
wsplot <- function(df, nc=15, seed=1234){
  wss <- (nrow(df)-1)*sum(apply(df,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(df,centers=i)$withinss)
  }
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")
}
wsplot(df)
```



Using this method we see that there is an elbow around 8 and 12 which suggest that 8 and 12 clusters might be a good choice.

Now we will use the NBClust to choose the number of clusters better NbClust not working atm switched data set and now I have a vector memory reached, set the memory significantly higher and now it is taking super long to run.

Hide

```
library(NbClust)
set.seed(1234)
nc <- NbClust(df, min.nc=2, max.nc=15, method="kmeans")
```

this would be to see the best k value but since Nbcluster isnt working we cant run this so we will use 8 and then 12 and see what gives use the best results at the end

Hide

```
table(nc$Best.n[1,])
```

Hide

```
barplot(table(nc$Best.n[1,]),
        xlab="Number of Clusters", ylab="Number of Criteria",
        main="Number of Clusters Chosen by 26 Criteria")
```

for now we will use 12 and then 8

Hide

```
set.seed(1234)
fit.km <- kmeans(df,12, nstart = 25, iter.max = 30)
fit.km$size
```

```
 [1]  565 1374 1713 1313 1793 1521  226  403  634   47 1763  216
```

Hide

```
fit.km$centers
```

	log_price	property_type	room_type	accommodates	bathrooms	bed_type	cancellation_policy
1	-0.07958856	0.03471693	-0.01555037	-0.1597118	-0.23617017	0.13073762	0.31850742
2	1.01861944	0.27185198	-0.80968766	1.0826921	0.63616228	0.14911076	0.40336436
3	0.30706897	-0.41024241	-0.81365058	-0.1281545	-0.36803897	0.12806748	0.91667426
4	-0.81591856	1.44700044	1.01602932	-0.5088018	-0.05054354	0.12433987	-0.06148998
5	-0.69127601	-0.67402369	1.07995701	-0.6072040	-0.23519483	0.10837953	-0.04633736
6	0.18096265	-0.38745830	-0.80869847	-0.1596939	-0.36924132	0.12495454	-0.78311231
7	-0.49947365	-0.19125557	0.44544881	-0.2251898	-0.19513404	0.14173270	-0.39800706
8	1.79393295	1.01977696	-0.68825477	3.0463350	3.07152263	0.14039583	0.51813035
9	0.86166747	0.11241556	-0.75512890	1.0317703	0.35921714	0.14488901	0.22218395
10	-0.13956827	-0.42943260	0.26459396	-0.2225174	-0.30544801	0.09985783	-0.14634747
11	-0.47959754	-0.11420820	0.51207260	-0.5035923	-0.23491414	0.11749114	-0.65122407
12	-0.69162448	-0.04603186	0.99503832	-0.5130383	-0.24756178	-6.65770407	-0.25113180

	cleaning_fee	host_has_profile_pic	host_identity_verified	number_of_reviews	review_scores_rating
1	-0.003069965	0.06167229	0.36238977	3.310724521	0.04300011
2	0.314153588	0.06167229	0.60506909	-0.076184778	0.09423386
3	0.448335538	0.06167229	0.15708598	-0.121608703	0.04715492
4	0.447431245	0.06167229	0.06103348	-0.089868989	0.11093795
5	0.495300663	0.06167229	0.05496598	-0.200984662	0.09824793
6	0.493647736	0.06167229	0.08454199	-0.216210028	0.19986233
7	-0.450268609	0.06167229	-0.53982933	-0.578400299	-4.60108518
8	0.320623613	0.06167229	0.12764565	-0.107434175	0.05299415
9	0.249442124	0.06167229	-1.60644410	-0.213581175	-0.01454718
10	-0.146597676	-15.11758375	-3.15920867	-0.196522033	-0.04916162
11	-2.018801165	0.06167229	-0.20652991	-0.243694838	0.10189641
12	-0.203060956	0.06167229	-0.05019408	-0.009497158	-0.04620693

	bedrooms	beds
1	-0.3373719	-0.14993972
2	1.1604859	0.94396900
3	-0.4942796	-0.23047595
4	-0.2970827	-0.39023978
5	-0.2887751	-0.48704968
6	-0.5104947	-0.26897053
7	-0.1763186	-0.09869211
8	2.9675790	3.13377096
9	1.0392514	0.89611962
10	-0.2015677	-0.09712377

```
11 -0.3442136 -0.41336164
12 -0.3311813 -0.45438328
```

k = 8

Hide

```
set.seed(1234)
fit.km2 <- kmeans(df,8, nstart = 25, iter.max = 30)
fit.km2$size
```

```
[1] 1898 1691  431  216 1839 2337 2564  592
```

Hide

```
fit.km2$centers
```

```
    log_price property_type     room_type accommodates  bathrooms   bed_type cancellation_policy
1 -0.46660698   -0.09728963  4.969493e-01   -0.4778463 -0.2363423  0.1199974         -0.63507189
2 -0.24218081   -0.13267019  5.662199e-02   -0.3142260 -0.2800792  0.1203993         -0.06889214
3  1.73686290    1.02073660 -6.589264e-01    2.9550596  3.0135645  0.1411970          0.50915168
4 -0.69162448   -0.04603186  9.950383e-01   -0.5130383 -0.2475618 -6.6577041         -0.25113180
5  1.01489938    0.22224109 -8.004668e-01    1.0995161  0.5956873  0.1473229          0.37866503
6 -0.74579384    0.17321381  1.109869e+00   -0.5700479 -0.1553419  0.1144497         -0.04564354
7  0.24034574   -0.33951972 -8.157663e-01   -0.1282069 -0.3585793  0.1294683          0.14228940
8 -0.07394504    0.06086648  5.124541e-05   -0.1445805 -0.2300995  0.1275431          0.34144780
  cleaning_fee host_has_profile_pic host_identity_verified number_of_reviews review_scores_rating
1 -2.018801165           0.06167229            -0.22453965      -0.260137582        -0.1139181160
2  0.477459609          -0.36022306            -1.64960195      -0.261374490        -0.0656756183
3  0.291139261           0.06167229             0.08682586      -0.115397721         0.0437954063
4 -0.203060956           0.06167229            -0.05019408      -0.009497158        -0.0462069253
5  0.299804979           0.06167229             0.04828246      -0.130250536        -0.0002168909
6  0.492073318           0.06167229             0.57194844      -0.144327274         0.0332983043
7  0.482553656           0.06167229             0.60506909      -0.140992008         0.0814879668
8  0.006919395           0.06167229             0.35851525       3.253108584         0.0540954539
    bedrooms       beds
1 -0.3337455 -0.3922468
2 -0.3594014 -0.3098156
3  2.8967512  3.1000607
4 -0.3311813 -0.4543833
5  1.1731432  0.9637375
6 -0.2937317 -0.4481035
7 -0.4748428 -0.2372922
8 -0.3196457 -0.1457347
```

Running aggregate to get the variable means for each cluster using the original unscaled data

Hide

```
aggregate(df0[], by=list(cluster=fit.km$cluster), mean)
```

cluster <int>	log_price <dbl>	property_type <dbl>	room_type <dbl>	accommodates <dbl>	bathrooms <dbl>	bed_type <dbl>	cancellation_policy <dbl>
1	4.696561	5.711504	1.442478	2.879646	1.092035	4.991150	2.522124

cluster <int>	log_price <dbl>	property_type <dbl>	room_type <dbl>	accommodates <dbl>	bathrooms <dbl>	bed_type <dbl>	cancellation_policy <dbl>
2	5.431966	7.270742	1.008006	5.549491	1.581150	4.998544	2.590247
3	4.955482	2.785756	1.005838	2.947461	1.018097	4.990076	3.002335
4	4.203484	14.997715	2.006855	2.129474	1.196116	4.988576	2.217060
5	4.286950	1.051311	2.041829	1.918015	1.092582	4.982153	2.229225
6	4.871036	2.935569	1.008547	2.879684	1.017423	4.988823	1.637738
7	4.415388	4.225664	1.694690	2.738938	1.115044	4.995575	1.946903
8	5.951147	12.188586	1.074442	9.769231	2.946650	4.995037	2.682382
9	5.326864	6.222397	1.037855	5.440063	1.425868	4.996845	2.444795
10	4.656396	2.659574	1.595745	2.744681	1.053191	4.978723	2.148936

1-10 of 12 rows | 1-8 of 15 columns Previous **1** 2 Next

Running aggregate to get the variable means for each cluster using the original unscaled data for k = 8

Hide

```
aggregate(df0[], by=list(cluster=fit.km2$cluster), mean)
```

cluster <int>	log_price <dbl>	property_type <dbl>	room_type <dbl>	accommodates <dbl>	bathrooms <dbl>	bed_type <dbl>	cancellation_policy <dbl>
1	4.437397	4.843519	1.722866	2.195996	1.091939	4.986828	1.756586
2	4.587682	4.610881	1.481963	2.547605	1.067416	4.986990	2.211118
3	5.912931	12.194896	1.090487	9.573086	2.914153	4.995360	2.675174
4	4.286716	5.180556	1.995370	2.120370	1.085648	2.259259	2.064815
5	5.429475	6.944535	1.013051	5.585644	1.558456	4.997825	2.570419
6	4.250442	6.622165	2.058194	1.997861	1.137356	4.984596	2.229782

cluster <int>	log_price <dbl>	property_type <dbl>	room_type <dbl>	accommodates <dbl>	bathrooms <dbl>	bed_type <dbl>	cancellation_policy <dbl>
7	4.910802	3.250780	1.004680	2.947348	1.023401	4.990640	2.380655
8	4.700340	5.883446	1.451014	2.912162	1.095439	4.989865	2.540541

8 rows | 1-8 of 15 columns

creating table for k =12 tried different values and the strongest correlation was with room_type

Hide

```
ct.km <- table(df0$room_type, fit.km$cluster)
ct.km
```

```
     1    2    3    4    5    6    7    8    9   10   11   12
1  316 1363 1703   58    1 1508   86  386  610   20  551   48
2  248   11   10 1188 1716   13  123    4   24   26 1135  121
3    1    0    0   67   76    0   17   13    0    1   77   47
```

creating table for k =12 tried different values and the strongest correlation was with room_type

Hide

```
ct.km2 <- table(df0$room_type, fit.km2$cluster)
ct.km2
```

```
     1    2    3    4    5    6    7    8
1  613  882  409   48 1815    5 2552  326
2 1198  803    5  121   24 2191   12  265
3   87    6   17   47    0  141    0    1
```

Hide

```
library(flexclust)
```

```
Loading required package: grid
Loading required package: lattice
Loading required package: modeltools
Loading required package: stats4
```

Hide

```
randIndex(ct.km)
```

```
      ARI
0.1756184
```

Hide

```
randIndex(ct.km2)
```

```
      ARI
0.2182345
```

the values are very low but still show a positive agreement with 12 clusters and 8 clusters Using 8 clusters results in a higher ARI Now we will use Hierarichal clustering using the same dataset

scaling data for hierarichal putting into var called data.scaled

Hide

```
data.scaled <- scale(df0)
head(data.scaled)
```

```
     log_price property_type  room_type accommodates  bathrooms  bed_type cancellation_policy
1  0.389431001    -0.6818272 -0.8243209   -0.1037054 -0.4003147 0.1527278           0.9137656
2  0.567531522    -0.6818272 -0.8243209    1.7576821 -0.4003147 0.1527278           0.9137656
3  0.338804548    -0.6818272 -0.8243209    0.8269884 -0.4003147 0.1527278          -0.3318673
5 -0.007353459    -0.6818272 -0.8243209   -0.5690523 -0.4003147 0.1527278          -0.3318673
6 -0.458760775    -0.6818272  1.0035005   -0.5690523 -0.4003147 0.1527278           0.9137656
7 -0.494318106    -0.6818272 -0.8243209   -0.1037054 -0.4003147 0.1527278          -0.3318673
  cleaning_fee host_has_profile_pic host_identity_verified number_of_reviews review_scores_rating
1    0.4953007           0.06167229              0.6050691       -0.6044981            0.7426474
2    0.4953007           0.06167229             -1.6064441       -0.5074101           -0.1286073
3    0.4953007           0.06167229              0.6050691       -0.4103221           -0.2530723
5    0.4953007           0.06167229              0.6050691       -0.5559541           -6.7252506
6    0.4953007           0.06167229              0.6050691       -0.5802261            0.7426474
7    0.4953007           0.06167229             -1.6064441       -0.2889622            0.3692525
    bedrooms       beds
1 -0.3034659 -0.5865974
2  2.0911410  0.9999724
3 -0.3034659  0.9999724
5 -1.5007694 -0.5865974
6 -0.3034659 -0.5865974
7 -0.3034659 -0.5865974
```

getting Eucleid distances between each of the

Hide

```
d <- dist(data.scaled)
fit.average <- hclust(d, method="average")
```

Hide

```
plot(fit.average, hang=-1, cex=.8,
     main="Hierarchical Clustering")
```

// hierarchical not working for my data set

model based clustering

The data set I have was not ideal for clustering I was able to get Kmeans to sort of work but it still provided a low agreement with the clusters. However we are able to deduce that the price is what separates most airbnbs apart. This makes sense since the more expensive the airbnb it follows in logic that the more rooms, bathroom, square foot etc, and Vice verso the cheaper the lower the number of rooms, bathrooms, ratings etc. I was unable to get the hierarchical to work since there are no labels in my data set, it is treating the rows as labels hence giving me errors and being extremely long processiong times. After runing model based clustering BIC suggest a VEV with one group and the EEV best values based on BIC are 9,8,6.