Teil 6

Cluserting

Rebecca Karwen

Contents

Vorbereitung

Zuerst laden wir den Datensatz und erstellen danach jeweils einen Datensatz für hierarchisches Clustering und für die K-Means Methode. Dabei werden jeweils die Felder, welche die Fachabteilung als relevant eingestuft hat, ausgewählt.

Zuerst skalieren wir die Daten, damit, obwohl in den Datensätzen unterschiedliche Messgrößen vorliegen, die Ergebnisse verglichen werden können.

```
scaled_house <- scale(house_data_km)
scaled_house_2 <- scale(house_data_h)
```

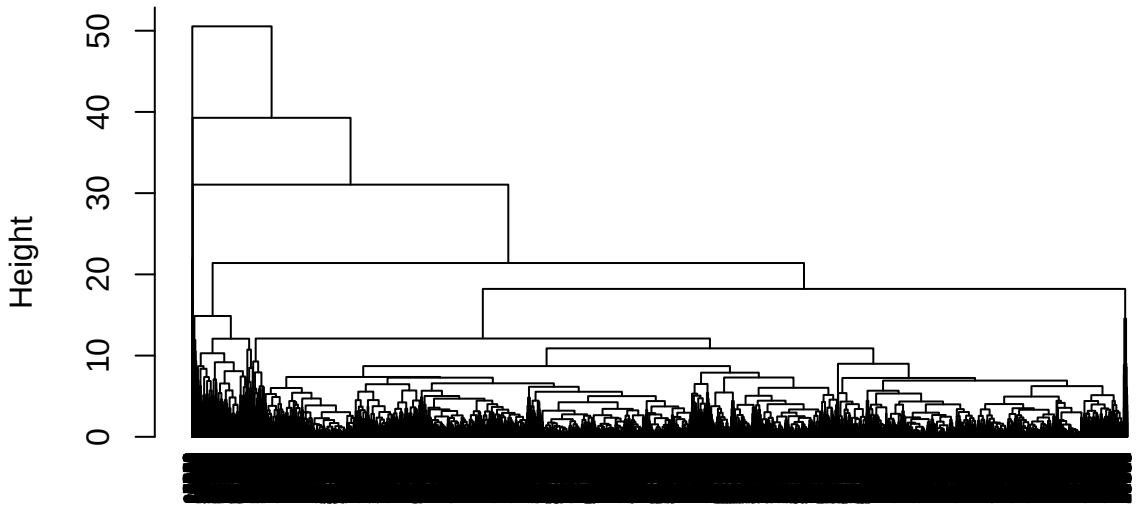
Hierarchisches Clustering

Als nächstes nutzen wir hierarchisches Clustering für den Datensatz. Dazu berechnen wir zunächst die Euklidische Distanz und danach führen wir das Clustering durch. Wir nutzen dafür die "complete" Methode, dies nimmt die maximale Distanz. Auch lassen wir uns noch das Dendrogramm darstellen.

```
my_dist <- dist(scaled_house_2, method = "euclidean")

my_hclust <- hclust(my_dist, method = "complete")

plot(my_hclust, hang = -1, cex = 0.6)
```

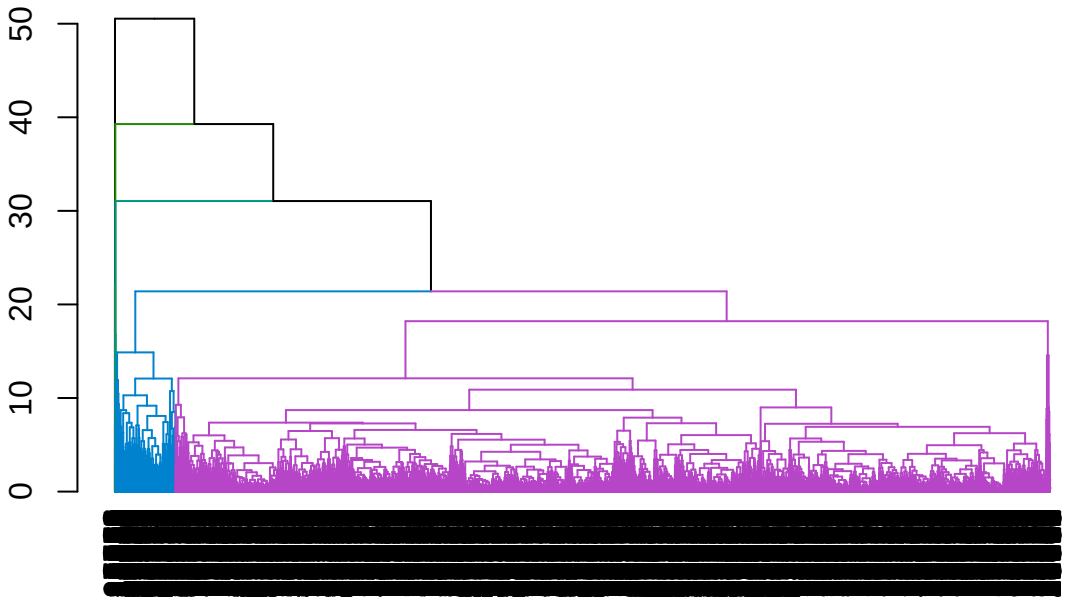


Danach nutzen wir cutree, um das Dendogramm ab Clustergröße von 6 zu schneiden, da, wie zuvor gesehen, dort die Zweige immer mehr auseinander gehen. Damit wären wir bei eine Clustergröße von 4, was mit den Ergebnissen des Elbowtestes am Anfang übereinstimmt. Zudem lassen wir es noch farbig darstellen, damit wir die Ergebnisse besser sehen können.

```
cluster_k2 <- cutree(my_hclust, k = 6)


dend_clust <- as.dendrogram(my_hclust)
dend_colored <- color_branches(dend_clust, k = 6)

# Plot the colored dendrogram
plot(dend_colored)
```



Danach fügen wir die Cluster in den Orginaldatensatz ein und bestimmen den Durchschnittswert aller Cluster.

```
k2_complete <- mutate(house_data_h, cluster = cluster_k2)
count(k2_complete, cluster)
```

```
## # A tibble: 6 x 2
##   cluster     n
##     <int> <int>
## 1       1 20209
## 2       2  1381
## 3       3    11
## 4       4     1
## 5       5    10
## 6       6     1
```

```
k2_complete %>%
  group_by(cluster) %>%
  summarise_all(list(mean)) %>%
  select(price, sqft_living, sqft_lot)
```

```
## # A tibble: 6 x 3
##      price sqft_living sqft_lot
##      <dbl>       <dbl>    <dbl>
## 1  483166.       1944.   14345.
## 2 1331674.       4004.   17925.
## 3 5284936.       9298.   53148.
## 4  700000        1300  1651359
## 5 1011100        3762.  961614.
## 6  640000        1620     6000
```

Wie sich zeigt, sind die Datensätze sehr schlecht in den Clustern verteilt, deswegen versuchen wir es nun als zweites mit der K-Means Methode.

K-Means

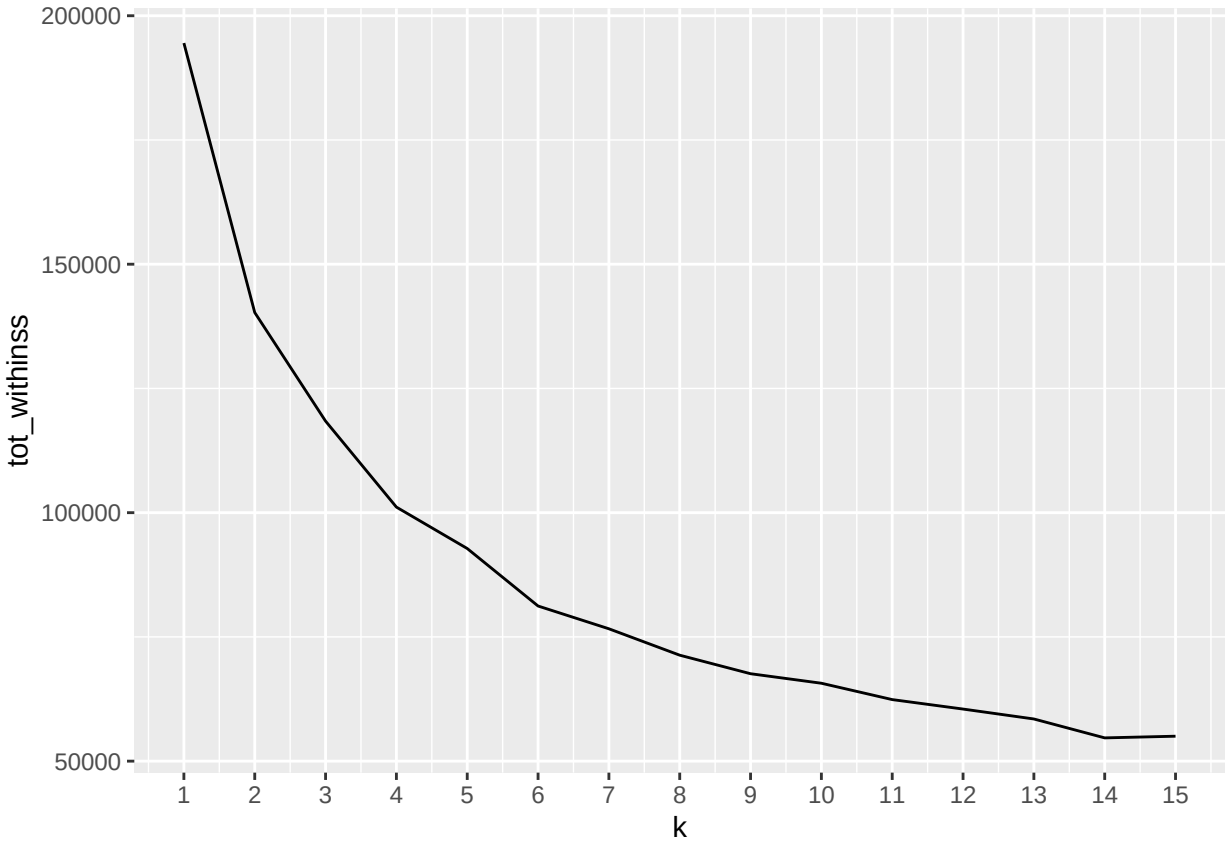
Zunächst schauen wir uns den Elbow an, damit wir sehen können, wie viele Cluster wir brauchen.

```
tot_withinss <- map_dbl(1:15,  function(k){
  model <- kmeans(x = scaled_house, centers = k)
  model$tot.withinss
})
```

```
## Warning: Quick-TRANSfer stage steps exceeded maximum (= 1080650)
```

```
# Generate a data frame containing both k and tot_withinss
elbow_df <- data.frame(
  k = 1:15,
  tot_withinss = tot_withinss
)

# Plot the elbow plot
ggplot(elbow_df, aes(x = k, y = tot_withinss)) +
  geom_line() +
  scale_x_continuous(breaks = 1:15)
```



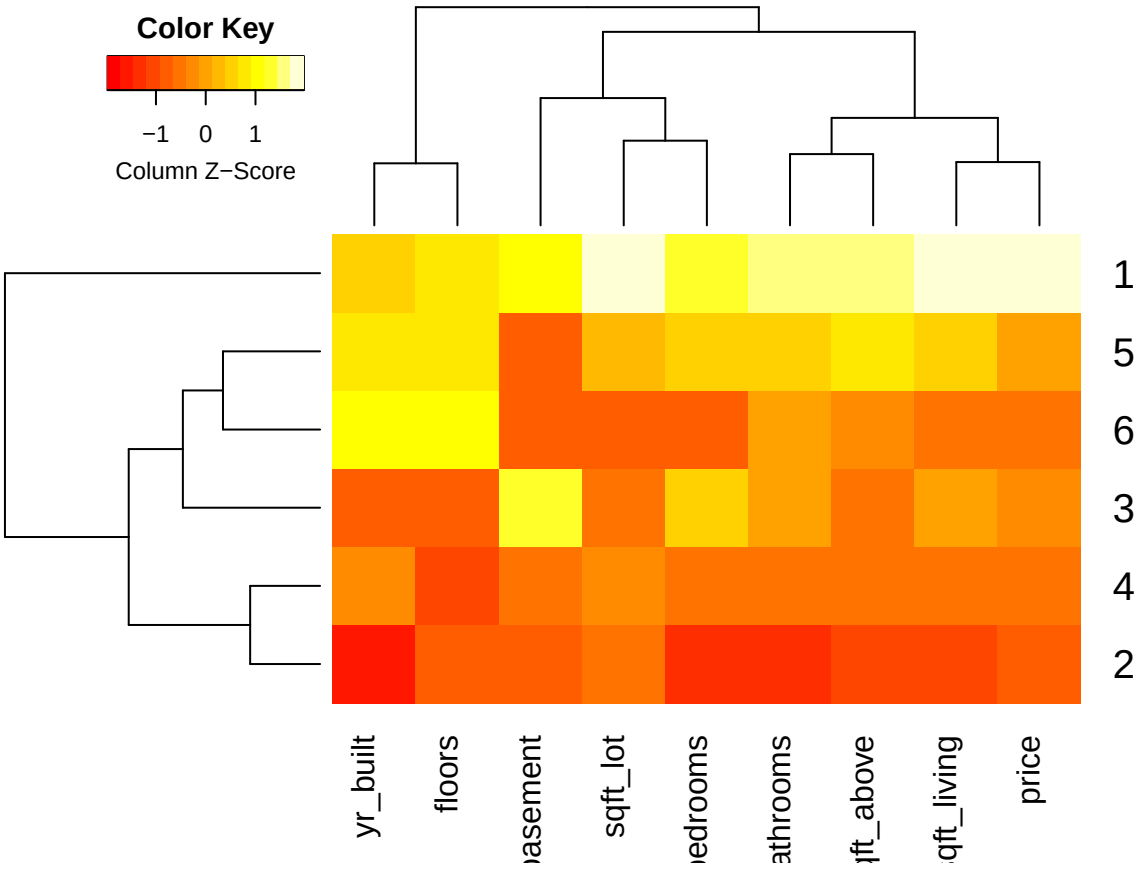
Der Eblow ist bei 6 zu sehen, deswegen nutzen wir dies als Größe für unser k.

Nun verwenden wir K-Means und lassen uns danach eine Heatmap der Ergebnisse anzeigen

```
my_km <- kmeans(scaled_house, centers = 6)

clust_houses <- my_km$cluster


heatmap.2(my_km$centers,scale = "column", trace = "none",  density.info = "none")
```



In der Heatmap lässt sich sehen, dass sich die Cluster generell nach der Größe ordnen. Meist sagt die Größe des Hauses ja auch etwas über die Anzahl an Stockwerken, Schlaf- und Badezimmern aus. Es lässt sich definitiv erkennen, dass Cluster 6 den höchsten Preis hat, Cluster 2 die geringste Fläche und Zimmer. Cluster 3 hat die kleinste Kellerfläche. Cluster 1 hat die ältesten Häuser - die von Cluster 4 und 6 sind dagegen neuer.

Nun werden die entstandenen Cluster dem Orginaldatensatz hinzugefügt und die Durchschnittswerte von Preis, Wohnfläche und Grundstücksfläche in Tabellenform angezeigt.

```
segment_houses <- mutate(house_data, cluster = clust_houses)

count(segment_houses, cluster)
```

```
## # A tibble: 6 x 2
##   cluster     n
##     <int> <int>
## 1       1   862
## 2       2  4448
## 3       3  3597
## 4       4  4791
## 5       5  3873
## 6       6  4042
```

```
segment_houses %>%
  group_by(cluster) %>%
  summarise_all(list(mean))  %>%
  select(price, sqft_living, sqft_lot, cluster)
```

```
## # A tibble: 6 x 4
##      price sqft_living sqft_lot cluster
##      <dbl>       <dbl>    <dbl>   <int>
## 1 1692276.       4553.   48844.       1
## 2  367334.       1183.    9757.       2
## 3  620836.       2498.   13026.       3
## 4  405932.       1681.   16123.       4
## 5  691567.       2977.   23148.       5
## 6  426491.       1782.    6742.       6
```

Es lässt sich sagen, dass die Gruppen unterschieden werden können. So ist z. B. 2 die Gruppe mit niedrigstem Preis aber auch kleinster Wohnfläche - trotzdem ist das Grundstück relativ groß. Bei Cluster 6 haben wir den höchsten Preis, aber auch die meiste Wohnfläche. Cluster 5 ist auch eher teurer, aber hat dafür die meiste gesamte Grundstücksfläche. Bei Cluster 4 ist das Verhältnis von eigentlicher Wohnfläche zu Grundstücksfläche am geringsten. Cluster 3 hat noch etwas mehr Wohnfläche und hält sich sonst generell im mittleren Bereich.

Bei der K-Means Methode erlangen wir besser verteilte Cluster. Somitwäre dieses Verfahren hier zu bevorzugen.