notebook

September 1, 2023

0.1 1. Targeting treatment for heart disease patients

There are many industries where understanding how things group together is beneficial. For example, retailers want to understand the similarities among their customers to direct advertisement campaigns, and botanists classify plants based on their shared similar characteristics. One way to group objects is to use clustering algorithms. We are going to explore the usefulness of unsupervised clustering algorithms to help doctors understand which treatments might work with their patients.

We are going to cluster anonymized data of patients who have been diagnosed with heart disease. Patients with similar characteristics might respond to the same treatments, and doctors could benefit from learning about the treatment outcomes of patients like those they are treating. The data we are analyzing comes from the V.A. Medical Center in Long Beach, CA. To download the data, visit here.

Before running any analysis, it is essential to get an idea of what the data look like. The clustering algorithms we will use require numeric data—we'll check that all the data are numeric. In this project, you will be brushing up on your base R skills. Let's go!

```
[20]: # Load the data
heart_disease <- read.csv("datasets/heart_disease_patients.csv")

# Print the first ten rows
# .... YOUR CODE FOR TASK 1 ....
head(heart_disease)
```

	id <int>	age <int>	sex <int>	cp <int>	trestbps <int>	chol <int>	fbs <int>	restecg <int>	thalach <int>	exang <int>
A data.frame: 6 x 12	1	63	1	1	145	233	1	2	150	0
	2	67	1	4	160	286	0	2	108	1
	3	67	1	4	120	229	0	2	129	1
	4	37	1	3	130	250	0	0	187	0
	5	41	0	2	130	204	0	2	172	0
	6	56	1	2	120	236	0	0	178	0

0.2 2. Quantifying patient differences

It is important to conduct some exploratory data analysis (EDA) to familiarize ourselves with the data before clustering. EDA will help us learn more about the variables and make an informed decision about whether we should scale the data. Because k-means and hierarchical clustering

measure similarity between points using a distance formula, it can place extra emphasis on certain variables that have a larger scale and thus larger differences between points.

Exploratory data analysis helps us to understand the characteristics of the patients in the data. We need to get an idea of the value ranges of the variables and their distributions. This will also be helpful when we evaluate the clusters of patients from the algorithms. Are there more patients of one gender? What might an outlier look like?

```r
# Evidence that the data should be scaled?
# .... YOUR CODE FOR TASK 2 ....
summary(heart_disease)
# Remove id
heart_disease <- heart_disease[ , !(names(heart_disease) %in% c("id"))]

# Scaling data and saving as a data frame
# .... YOUR CODE FOR TASK 2 ....
scaled <- scale(heart_disease)


# What do the data look like now?
# .... YOUR CODE FOR TASK 2 ....
summary(scaled)
```

```
      id               age             sex               cp
 Min.   :  1.0    Min.   :29.00    Min.   :0.0000    Min.   :1.000
 1st Qu.: 76.5    1st Qu.:48.00    1st Qu.:0.0000    1st Qu.:3.000
 Median :152.0    Median :56.00    Median :1.0000    Median :3.000
 Mean   :152.0    Mean   :54.44    Mean   :0.6799    Mean   :3.158
 3rd Qu.:227.5    3rd Qu.:61.00    3rd Qu.:1.0000    3rd Qu.:4.000
 Max.   :303.0    Max.   :77.00    Max.   :1.0000    Max.   :4.000
    trestbps          chol              fbs              restecg
 Min.   : 94.0    Min.   :126.0    Min.   :0.0000    Min.   :0.0000
 1st Qu.:120.0    1st Qu.:211.0    1st Qu.:0.0000    1st Qu.:0.0000
 Median :130.0    Median :241.0    Median :0.0000    Median :1.0000
 Mean   :131.7    Mean   :246.7    Mean   :0.1485    Mean   :0.9901
 3rd Qu.:140.0    3rd Qu.:275.0    3rd Qu.:0.0000    3rd Qu.:2.0000
 Max.   :200.0    Max.   :564.0    Max.   :1.0000    Max.   :2.0000
    thalach           exang            oldpeak           slope
 Min.   : 71.0    Min.   :0.0000    Min.   :0.00     Min.   :1.000
 1st Qu.:133.5    1st Qu.:0.0000    1st Qu.:0.00     1st Qu.:1.000
 Median :153.0    Median :0.0000    Median :0.80     Median :2.000
 Mean   :149.6    Mean   :0.3267    Mean   :1.04     Mean   :1.601
 3rd Qu.:166.0    3rd Qu.:1.0000    3rd Qu.:1.60     3rd Qu.:2.000
 Max.   :202.0    Max.   :1.0000    Max.   :6.20     Max.   :3.000

      age               sex                cp              trestbps
 Min.   :-2.8145    Min.   :-1.4549    Min.   :-2.2481    Min.   :-2.14149
 1st Qu.:-0.7124    1st Qu.:-1.4549    1st Qu.:-0.1650    1st Qu.:-0.66420
 Median : 0.1727    Median : 0.6851    Median :-0.1650    Median :-0.09601
```

```
Mean    : 0.0000    Mean    : 0.0000    Mean    : 0.0000    Mean     : 0.00000
3rd Qu.: 0.7259    3rd Qu.: 0.6851    3rd Qu.: 0.8765    3rd Qu.: 0.47218
Max.    : 2.4961    Max.    : 0.6851    Max.    : 0.8765    Max.     : 3.88132
      chol                fbs               restecg              thalach
Min.    :-2.3310    Min.    :-0.4169    Min.    :-0.995103    Min.    :-3.4364
1st Qu.:-0.6894    1st Qu.:-0.4169    1st Qu.:-0.995103    1st Qu.:-0.7041
Median :-0.1100    Median :-0.4169    Median : 0.009951    Median : 0.1483
Mean    : 0.0000    Mean    : 0.0000    Mean    : 0.000000    Mean    : 0.0000
3rd Qu.: 0.5467    3rd Qu.:-0.4169    3rd Qu.: 1.015005    3rd Qu.: 0.7166
Max.    : 6.1283    Max.    : 2.3905    Max.    : 1.015005    Max.    : 2.2904
      exang              oldpeak              slope
Min.    :-0.6955    Min.    :-0.8954    Min.    :-0.9747
1st Qu.:-0.6955    1st Qu.:-0.8954    1st Qu.:-0.9747
Median :-0.6955    Median :-0.2064    Median : 0.6480
Mean    : 0.0000    Mean    : 0.0000    Mean    : 0.0000
3rd Qu.: 1.4331    3rd Qu.: 0.4827    3rd Qu.: 0.6480
Max.    : 1.4331    Max.    : 4.4445    Max.    : 2.2708
```

0.3 3. Let's start grouping patients

Now that we have scaled the data, we can start the clustering process. For the k-means algorithm, it is necessary to select the number of clusters in advance.

It is also important to make sure that our results are reproducible when conducting a statistical analysis. This means that when someone runs our code on the same data, they will get the same results. Because we are doing an analysis that has a random aspect, it is necessary to set a seed to ensure reproducibility.

Reproducibility is especially important because doctors will potentially use our results to treat patients. It is vital that other analysts see where the groups come from and can verify the results.

```
[40]: # Set the seed so that results are reproducible
seed_val  <- 10
set.seed(seed_val)

# Select a number of clusters
# .... YOUR CODE FOR TASK 3 ....
k <-5

# Run the k-means algorithm
first_clust <- kmeans(scaled, centers=k, nstart=1)

# How many patients are in each cluster?
# .... YOUR CODE FOR TASK 3 ....
first_clust$size
```

1. 24 2. 53 3. 87 4. 72 5. 67

0.4 4. Another round of k-means

Because the k-means algorithm initially selects the cluster centers by randomly selecting points, different iterations of the algorithm can result in different clusters. If the algorithm is genuinely grouping similar observations (as opposed to clustering noise), then cluster assignments will be somewhat robust between various iterations of the algorithm.

With regards to the heart disease data, this would mean that the same patients would be grouped even when the algorithm is initialized at different random points. If patients are not in similar clusters with various algorithm runs, then the clustering method is not picking up on meaningful relationships between patients.

We're going to explore how the patients are grouped with another iteration of the k-means algorithm. We will then be able to compare the resulting groups of patients.

```r
[26]: # Set the seed
seed_val <- 38
set.seed(seed_val)

# Select a number of clusters and run the k-means algorithm
# .... YOUR CODE FOR TASK 4 ....
k <- 5
second_clust <- kmeans(scaled, centers=k, nstart=1)

# How many patients are in each cluster?
# .... YOUR CODE FOR TASK 4 ....
second_clust$size
```

1. 51 2. 48 3. 42 4. 67 5. 95

0.5 5. Comparing patient clusters

It is important that the clusters are stable. Even though the algorithm begins by randomly initializing the cluster centers, if the k-means algorithm is the right choice for the data, then different initializations of the algorithm will result in similar clusters.

The clusters from different iterations may not be the same, but the clusters should be roughly the same size and have similar distributions of variables. If there is a lot of change in clusters between different iterations of the algorithm, then k-means clustering is not the right choice for the data.

It is not possible to validate that the clusters obtained from the algorithm are accurate because there is no patient labeling. Thus, it is necessary to examine how the clusters change between different iterations of the algorithm. We're going to use some visualizations to get an idea of the cluster stabilities. That way we can see how certain patient characteristics may have been used to group patients together.

```r
[28]: # Add cluster assignments to the data
heart_disease['first_clust'] <- first_clust$cluster
heart_disease['second_clust'] <- second_clust$cluster

# Load ggplot2
```

```
# .... YOUR CODE FOR TASK 5 ....
library(ggplot2)

# Create and print the plot of age and chol for the first clustering algorithm
plot_one <- ggplot(heart_disease, aes(x=age, y=chol, color=as.
 ↪factor(first_clust))) +
  geom_point()
plot_one

# Create and print the plot of age and chol for the second clustering algorithm
# .... YOUR CODE FOR TASK 5 ....
plot_two <-ggplot(heart_disease, aes(x=age, y=chol, color=as.
 ↪factor(second_clust)))+ geom_point()
plot_two
```
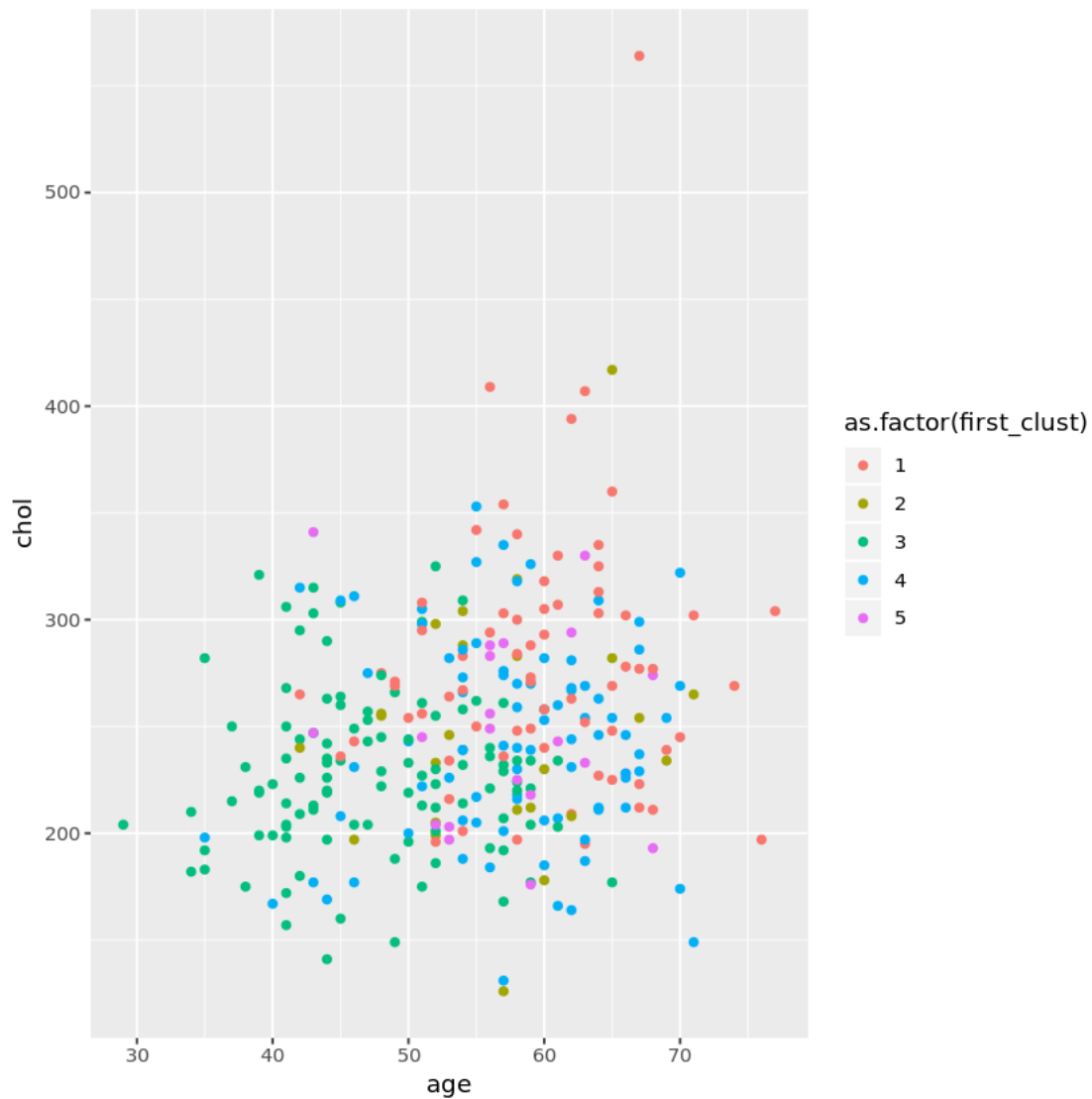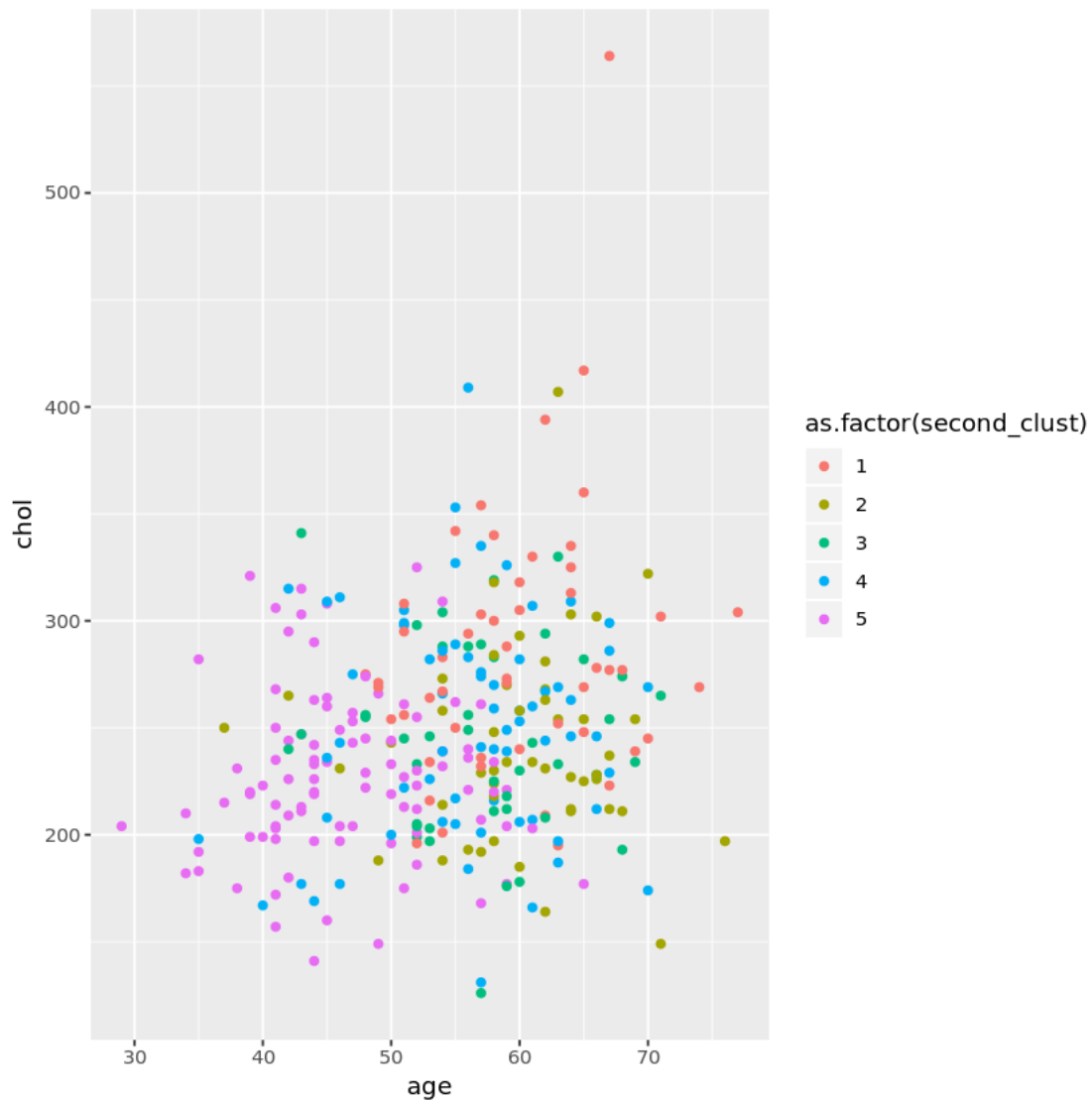
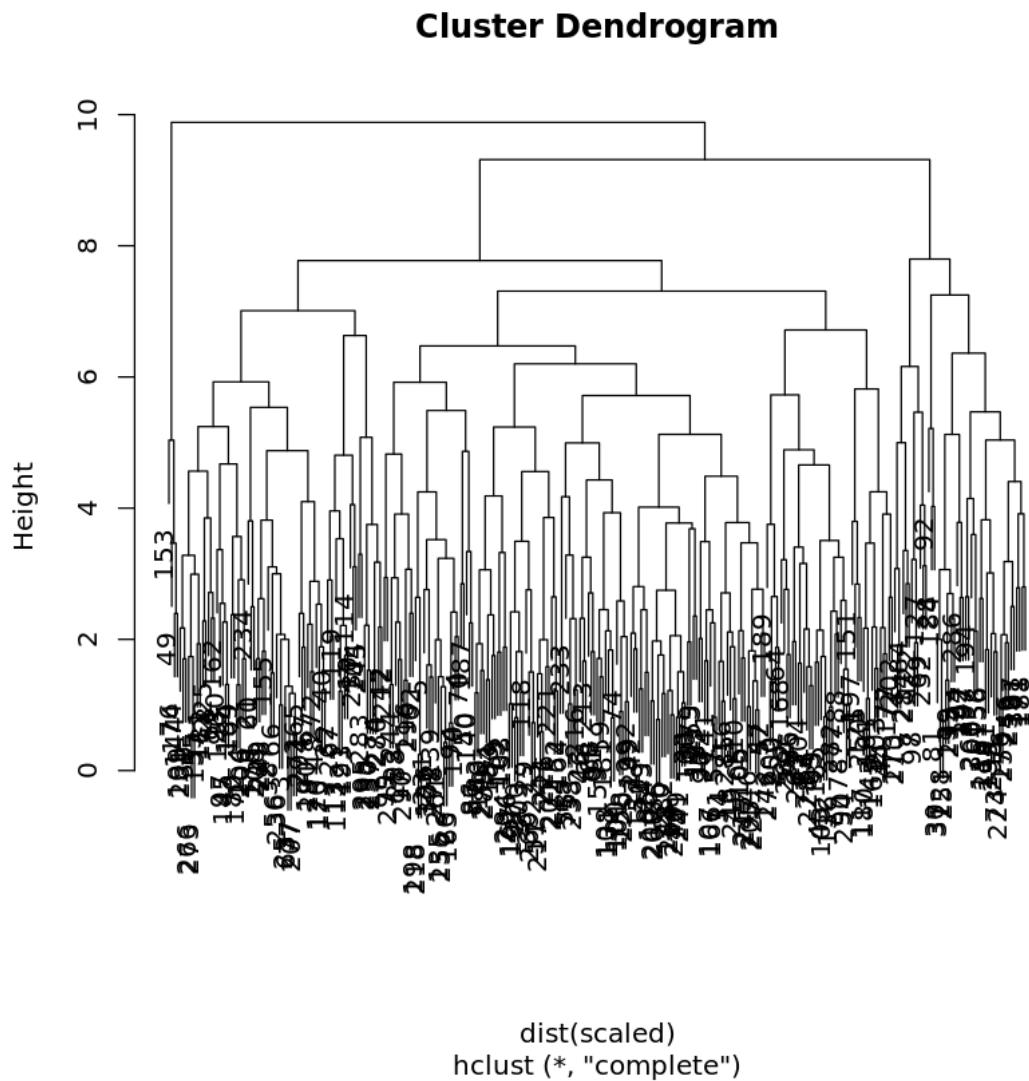## 0.6   6. Hierarchical clustering: another clustering approach

An alternative to k-means clustering is hierarchical clustering. This method works well when data have a nested structure. Heart disease patient data might follow this type of structure. For example, if men are more likely to exhibit specific characteristics, those characteristics might be nested inside the gender variable. Hierarchical clustering also does not require the number of clusters to be selected before running the algorithm.

Clusters can be selected by using the dendrogram. The dendrogram allows us to see how similar observations are to one another, and they are useful in helping us choose the number of clusters to group the data. It is now time for us to see how hierarchical clustering groups the data.

```
[30]: # Execute hierarchical clustering with complete linkage
      hier_clust_1 <- hclust(dist(scaled), method = "complete")

      # Print the dendrogram
      # .... YOUR CODE FOR TASK 6 ....
      plot(hier_clust_1)

      # Get cluster assignments based on number of selected clusters
      # .... YOUR CODE FOR TASK 6 ....
      hc_1_assign <- cutree(hier_clust_1,5)
```

Cluster Dendrogram

dist(scaled)
hclust (*, "complete")

0.7 7. Hierarchical clustering round two

In hierarchical clustering, there are multiple ways to measure the dissimilarity between clusters of observations. Complete linkage records the largest dissimilarity between any two points in the two clusters being compared. On the other hand, single linkage is the smallest dissimilarity between any two points in the clusters. Different linkages will result in different clusters being formed.

We want to explore different algorithms to group our heart disease patients. The best way to measure dissimilarity between patients could be to look at the smallest difference between patients and minimize that difference when grouping together clusters. It is always a good idea to explore different dissimilarity measures. Let's implement hierarchical clustering using a new linkage function.

```
[32]:  # Execute hierarchical clustering with single linkage
       # .... YOUR CODE FOR TASK 7 ....
       hier_clust2 <- hclust(dist(scaled), method='single')
       # Print the dendrogram
       # .... YOUR CODE FOR TASK 7 ....
       plot(hier_clust2)
       # Get cluster assignments based on number of selected clusters
       # .... YOUR CODE FOR TASK 7 ....
       hc_2_assgin <- cutree(hier_clust2,5)
```

Cluster Dendrogram

Height

dist(scaled)
hclust (*, "single")
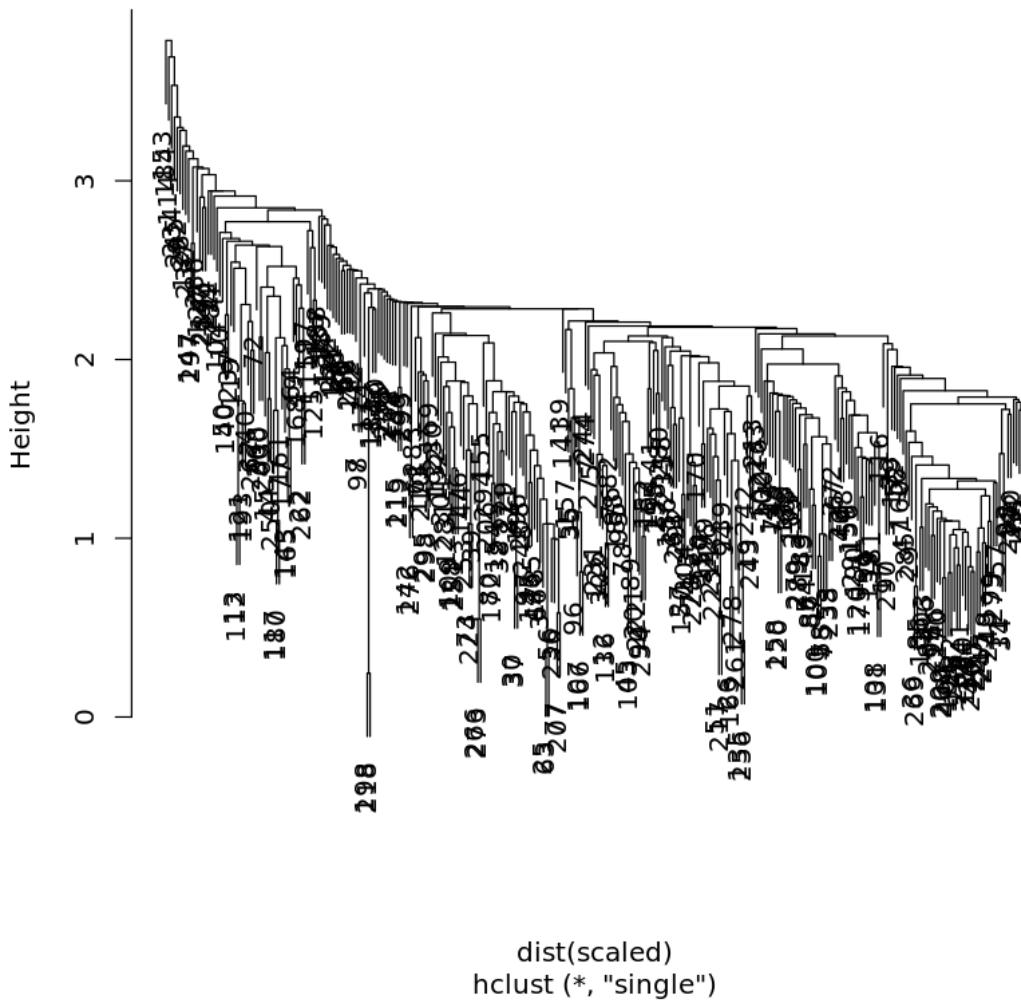
0.8   8. Comparing clustering results

The doctors are interested in grouping similar patients together to determine appropriate treatments. Therefore, they want clusters with more than a few patients to see different treatment options. While a patient can be in a cluster by themselves, this means that the treatment they received might not be recommended for someone else in the group.

Like the k-means algorithm, the way to evaluate hierarchical clusters is to investigate which patients are grouped together. Are there patterns evident in the cluster assignments, or do they seem to be groups of noise? We're going to examine the clusters resulting from the two hierarchical algorithms.

```
[34]:  # Add assignment of chosen hierarchical linkage
       heart_disease["hc_clust"] <- hc_1_assign
```

```
# Remove the sex, first_clust, and second_clust variables
hd_simple <- heart_disease[, !(names(heart_disease) %in% c("sex",␣
 ↪"first_clust", "second_clust"))]

# Get the mean and standard deviation summary statistics
clust_summary <- do.call(data.frame, aggregate(. ~hc_clust, data = hd_simple,␣
 ↪function(x) c(avg = mean(x), sd = sd(x))))
clust_summary
```

	hc_clust	age.avg	age.sd	cp.avg	cp.sd	trestbps.avg	trestbps.sd	chol.av
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
A data.frame: 5 x 21	1	51.41667	8.540979	2.783333	0.9470625	129.1389	15.93800	239.87
	2	58.11111	7.754246	3.763889	0.6165112	130.0417	13.90657	253.22
	3	61.00000	3.908034	3.916667	0.2886751	168.5000	17.45904	284.91
	4	59.00000	9.203580	3.571429	0.8501112	134.7714	18.64070	233.85
	5	64.75000	2.061553	3.250000	0.5000000	138.7500	18.42779	433.75

## 0.9 9. Visualizing the cluster contents

In addition to looking at the distributions of variables in each of the hierarchical clustering runs, we will make visualizations to evaluate the algorithms. Even though the data has more than two dimensions, we can get an idea of how the data clusters by looking at a scatter plot of two variables. We want to look for patterns that appear in the data and see what patients get clustered together.
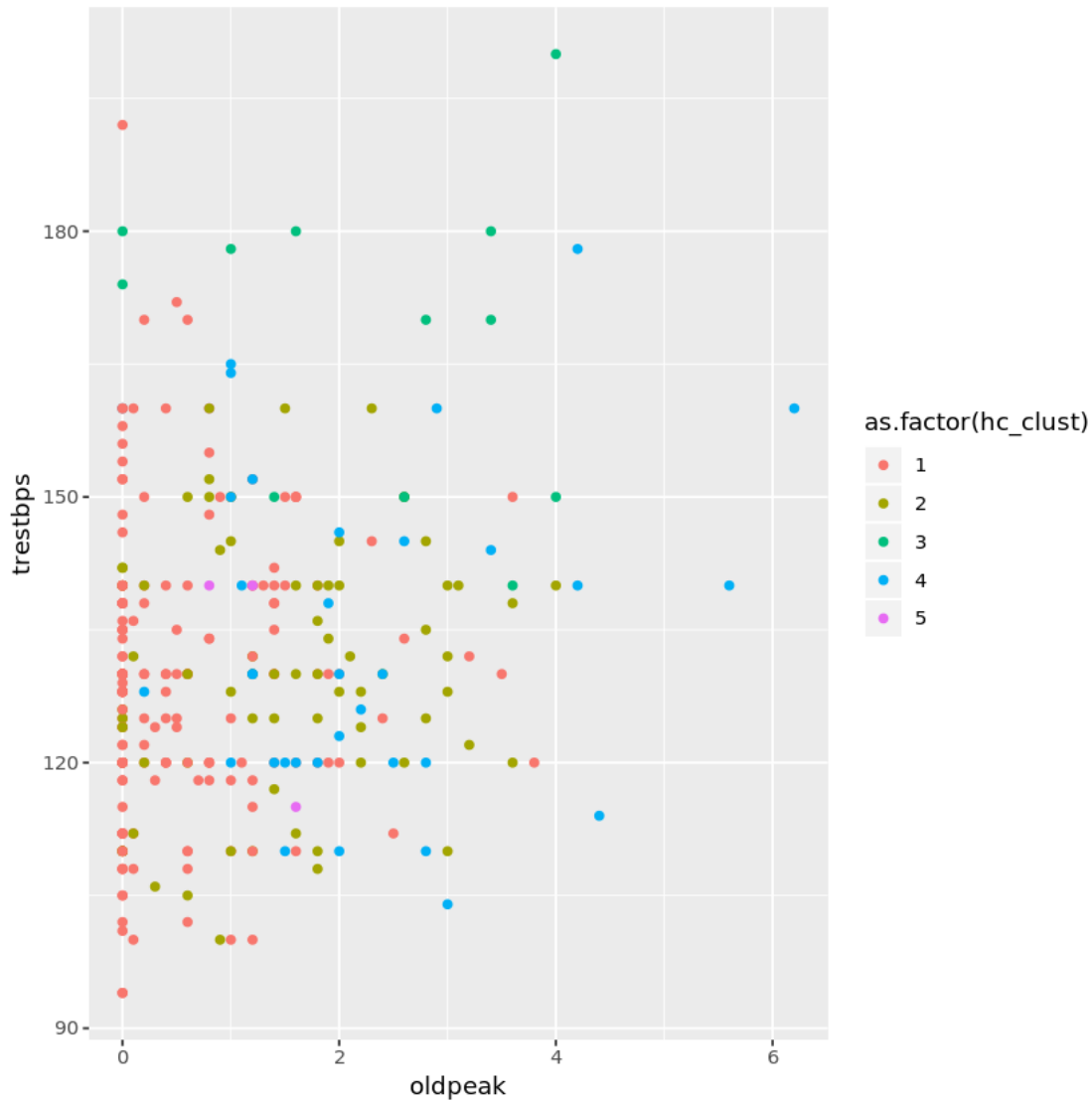
```
[36]: # Plot age and chol
plot_one <- ggplot(heart_disease, aes(x = age, y = chol,
                                      color = as.factor(hc_clust))) +
  geom_point()
plot_one

# Plot oldpeak and trestbps
plot_two <- ggplot(heart_disease, aes(x = oldpeak, y = trestbps,
                                      color = as.factor(hc_clust))) +
  geom_point()
plot_two
```
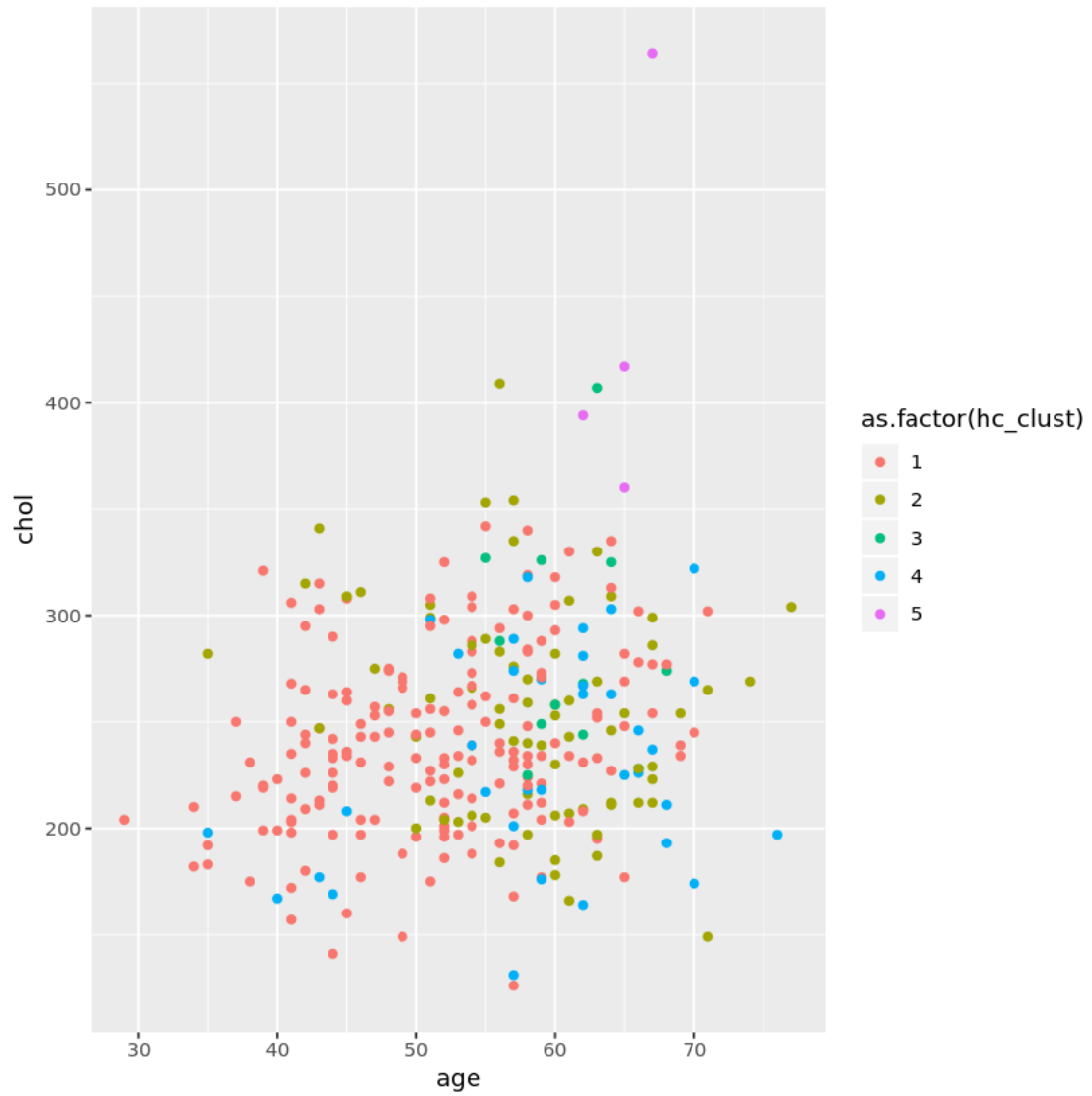
0.10 10. Conclusion

Now that we've tried multiple clustering algorithms, it is necessary to determine if we think any of them will work for clustering our patients. For the k-means algorithm, similar clusters must be produced for each iteration of the algorithm to make sure that the algorithm clusters the signal, not the noise.

For the sake of the doctors, we also want to have multiple patients in each group so they can compare treatments. We only did some preliminary work to explore the performance of the algorithms, and it is necessary to explore further before making a recommendation. Based on the above analysis, are there any algorithms that you would want to investigate further to group patients?

Remember that it is important the k-mean algorithm seems stable when running multiple iterations. This means that we would see similar groups of patients showing up in the plots from the different

iterations of the algorithm. For the hierarchical clustering, we need a method that puts a balanced number of patients in each group.

[38]:
```
# Add TRUE if the algorithm shows promise, add FALSE if it does not
explore_kmeans <- FALSE
explore_hierarch_complete <- TRUE
explore_hierarch_single <- FALSE
```