Exercise 4

Business Analytics and Data Science WS18/19

Asking for help

We now come to a point where issues like errors and warnings will become more frequent and where you might want to ask for help when things won't work. This is great, remember that you have your classmates, the tutorial class, and your teachers to help you out - preferably in this order. We enjoy hard questions. In order to ask for help efficiently, learn the following steps by heart:

1. Try to understand the error or warning message.
2. Check if the objects that where saved before the error have the expected format and values.
3. Check for typos or errors in your logic.
4. Check the function help.
5. Google the error message or your question. I highly recommend Stackoverflow.com .

At this point, maybe you need some outside help. No matter who you aks, they will need the following information from you:

1. What you were trying to do and how you were trying to do it.
2. The exact problem that occured and the exact error message.
3. What you have tried to solve the problem and why it didn't work.
4. A reproducable example from your code. Don't send the whole code, just the parts that are needed to create the error (see the FAQ on stackoverflow). You can copy/paste objects via **dput()**.

Prepare your data

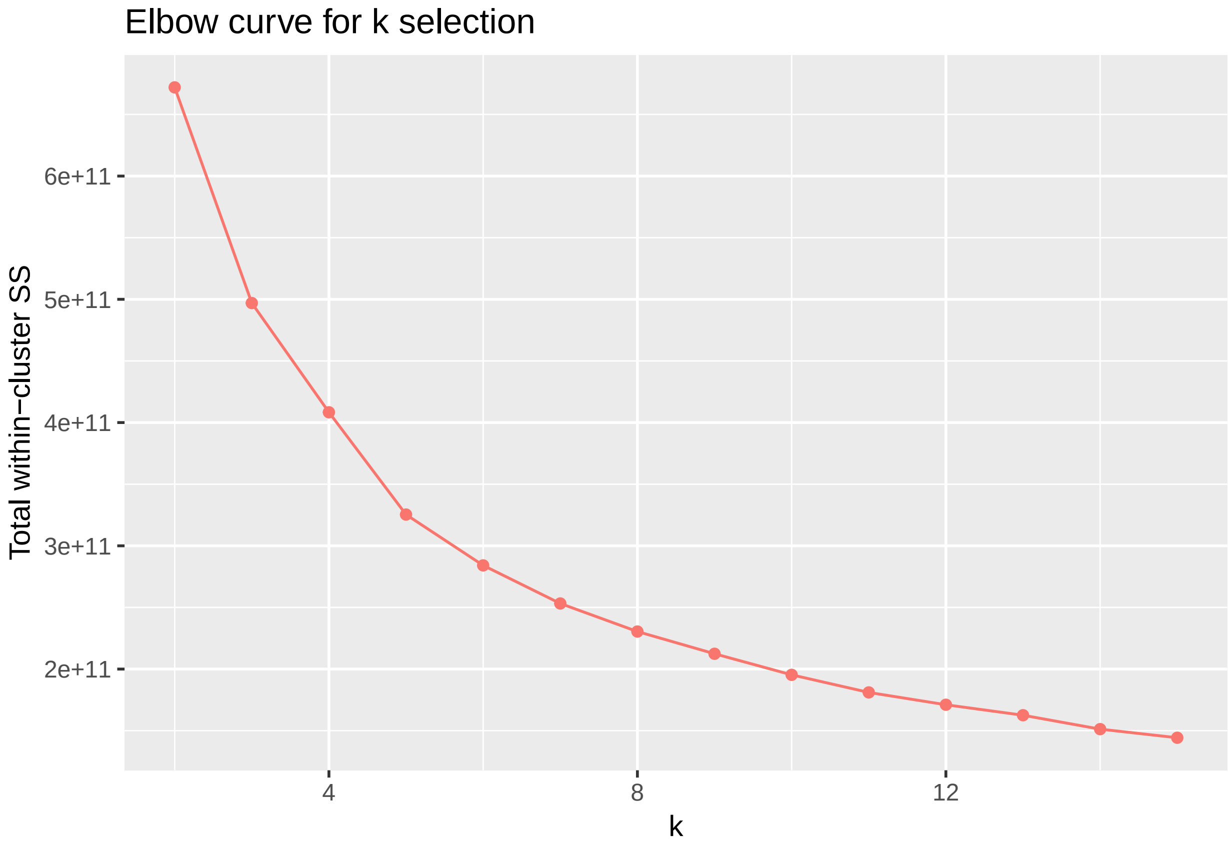
1. Use your custom function **get.loan.dataset()** to load and clean the data and save the resulting data frame to an object **loans**.
2. Clustering works by calculating the distance between the observations. Because distance calculations are more complicated (although not impossible) if they include data that is not numeric, for example your field of study, we will restrict ourselves to the numeric variables for this exercise.
3. Save the indices (=column number) of all numeric variables in the data to a vector *idx_numeric*. Don't do this by hand - it may work here, but it won't work when your data comrpises 200 variables.

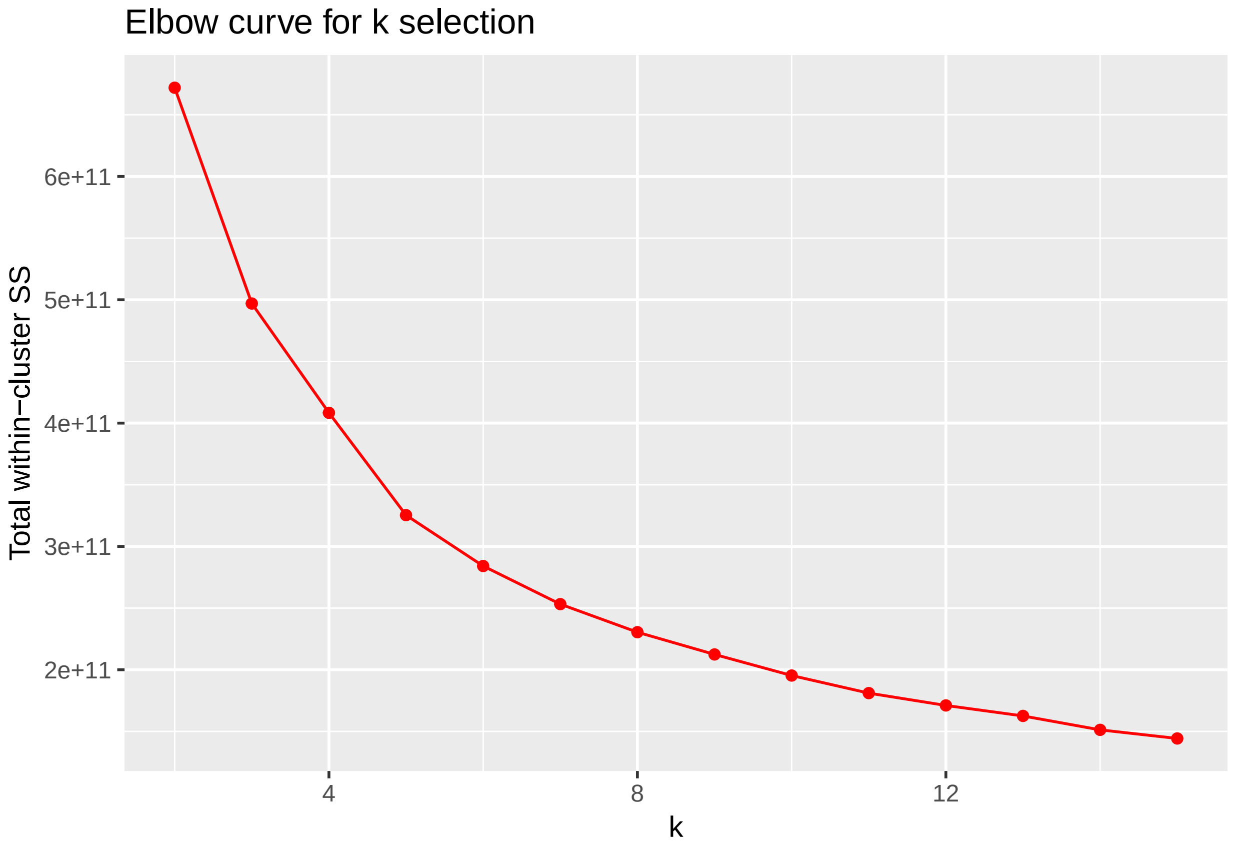
k-means Clustering

Clustering is a popular approach for unsupervised learning. A standard method used in many data mining applications is k-means clustering.

1. Cluster the data into 5 groups using the k-means algorithm and increase the maximum number of iterations to 50. Look at the *structure* of the result object and extract a vector **clusters** indicating the cluster identity for each observation. Note that the standard k-means algorithm only works for numeric variables, so you will have to select these.
2. The k-means algorithm requires that you specify the number of clusters beforehand. We can empirically test which number of clusters will give the best 'fit'. Let's say we want to test between 1 and 15 clusters using a loop. To loop over the number of clusters, k, create a vector **k.settings** with the values 1 to 15. Also create an empty vector **obj.values** with the length of **k.settings** to store the results. Then, loop over the numer of values in **k.settings** and, for each **i**, perform the following steps in the body of the for-loop:
    1. Calculate the k-means clusters for the number of clusters given in **k.settings[i]** and save the results in an object *cluster solution* **clu.sol**.
    2. This object is a list with, among others, an element **tot.withinss**. Extract the within-cluster sum-of-squares from **clu.sol** and save the result to the result vector at position i **obj.values[i]**.

3. Plot the results with the number of clusters on the x-axis and the within-cluster sum-of-squares on the y axis. What is the optimal number of clusters according to the elbow criterion?




Elbow curve for k selection
Total within−cluster SS
6e+11
5e+11
4e+11
3e+11
2e+11
4
8
12
k