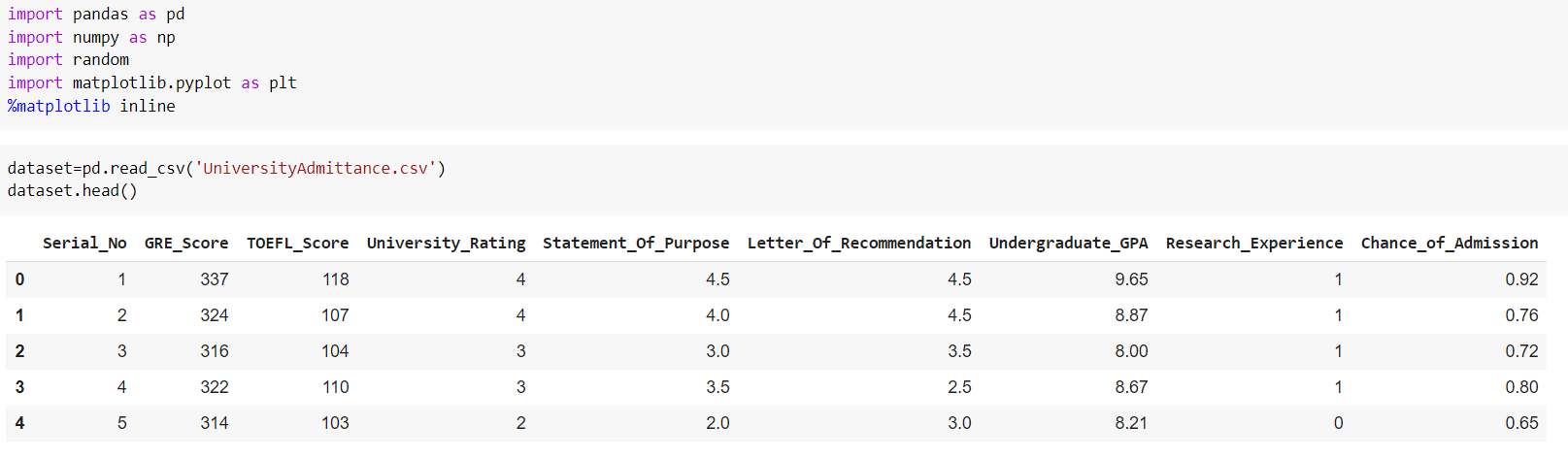
Implementation of the KMeans algorithm

The following example uses a dataset which contains information about university admittance. It is made up out of nine features/columns (Serial\_No, GRE\_Score, TOEFL\_Score, University\_Rating, Statement\_Of\_Purpose, Letter\_Of\_Recommendation, Undergraduate\_GPA, Research\_Experience, Chance\_of\_Admission) and four hundred instances/rows, each representing one individual student.   
 For this example I have chosen to work with the information about the Graduate Record Examination score (given in the GRE\_Score column) and to see how it affects ones chances of being admitted to the desired university. This is done by cross-comparing the above mentioned GRE\_Score column with the Chance\_of\_Admission column. But, in order to better understand and grasp the given results, as well as to further analyze given data, implementation of the KMeans algorithm is needed.

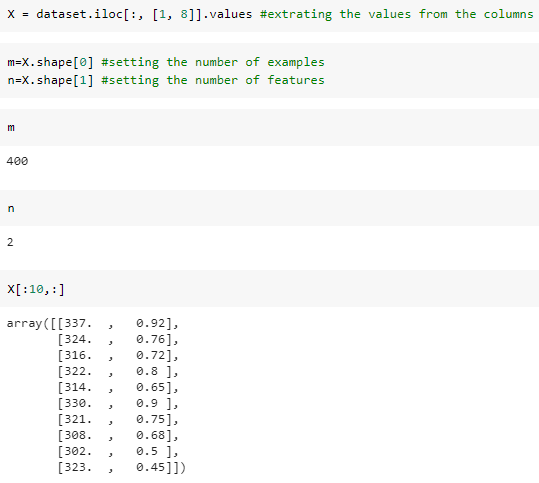
1. Setting up the dataset

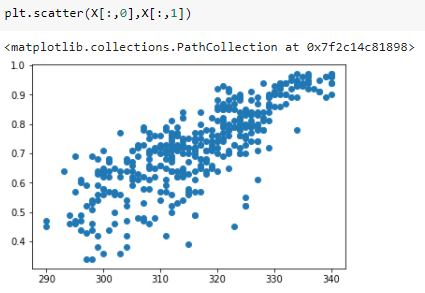
First and foremost, before doing anything else, we need to import all the required libraries and also define our dataset. This is done in the following manner:



1. Acquiring the data

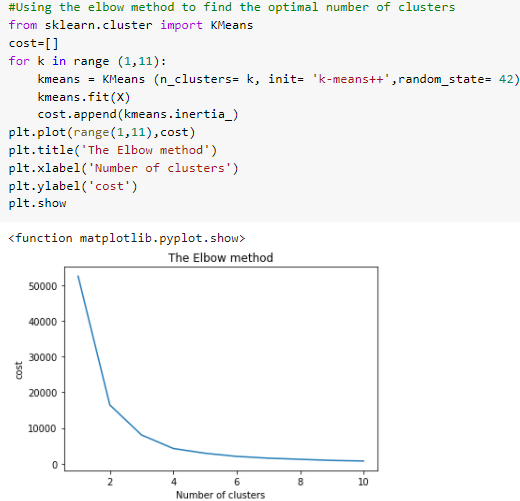
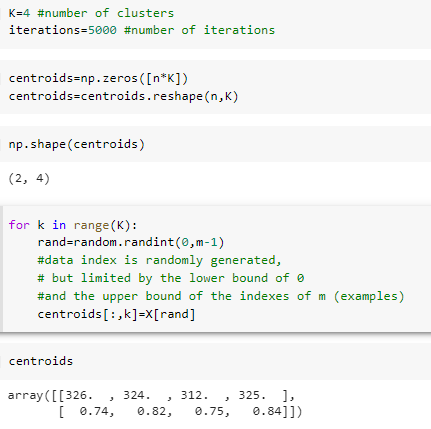
Next, in order to be able to run the KMeans algorithm we need to define the data that we want to use. As it was said earlier, the data will consist of all the values (rows) for the features GRE Score and Chance of Admission. Also, at this point we will define the number of instances, the number of features and visualize the data by plotting it.

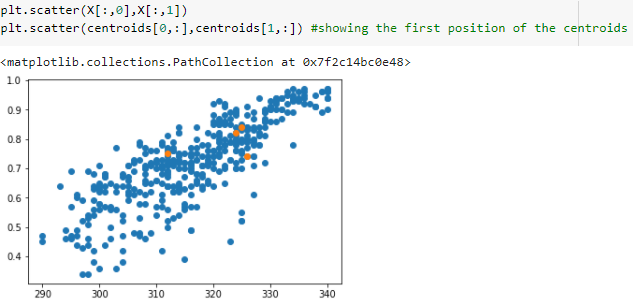


1. Setting up the clusters and centroids

Because this dataset is linear it is much harder to determine the number of clusters that we need to define just by looking at the plotted data. In order to deal with this we will use a so-called Elbow method. It is essentially an algorithm that does clustering of the dataset for a given range of values (in this case 1-11) and for every said value it also computes an average score for all clusters.

Once plotted the elbow method will help us decide on the number of clusters. This is done by observing the cost change. In this case the number of clusters will be four since that was the point at which the cost line had stopped sharply descending.

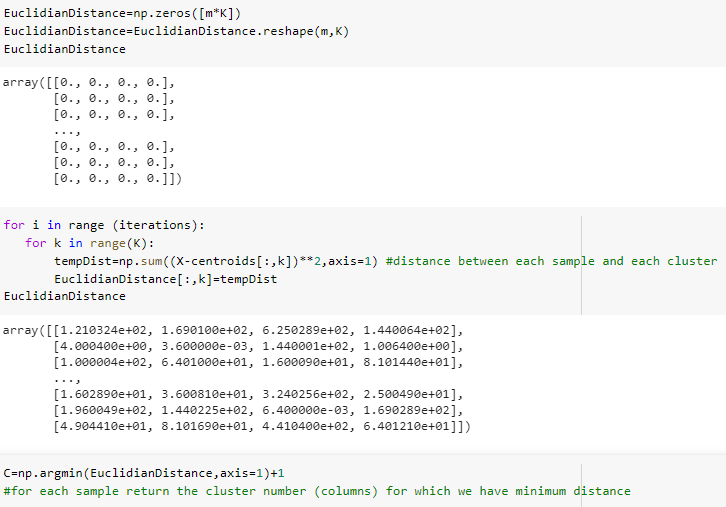


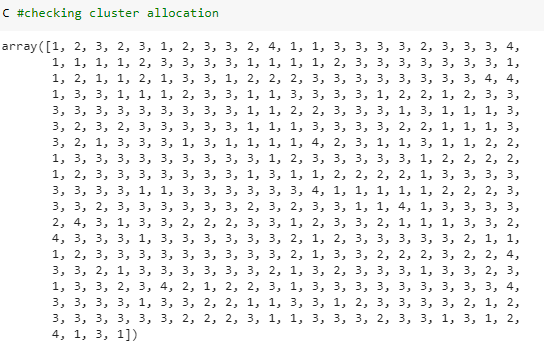
Again, by plotting the data we can now observe the defined centroids. However, the positions of these centroids are random and must be further tweaked in order to represent the actual cluster centers.

This is done by calculating the Euclidian Distance.

1. The Euclidian Distance

Euclidian distance is referring to the calculation of the distance between any two points. It can help us with clustering, since in order to move the centroids to a more accurate position we first need to find out which samples go with which cluster. This is done by calculating the distance between each sample and each centroid, and then choosing the minimal distance value to associate them to the cluster they belong to.





1. Grouping and plotting

To finalize the implementation of the KMeans algorithm we must now group the samples based on their representative centroids. These groups will then be used to reposition those centroids by calculating the mean values of the defined groupes.

Once all of this is completed, we plot the data to show both clusters and centroids, thus successfully implementing the whole KMeans algorithm from scratch.

