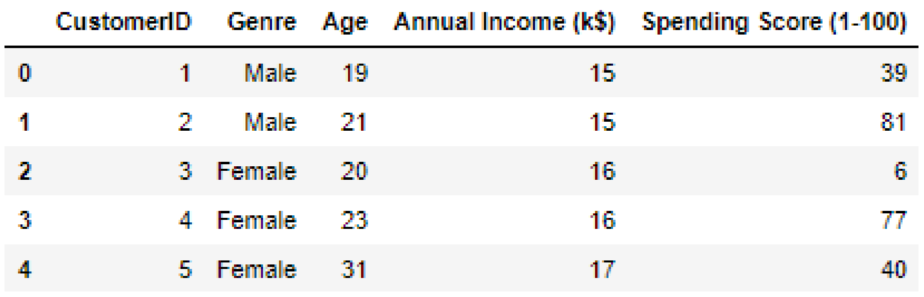
For this assignment, we will be examining the Mall_Customers dataset found on Kaggle to showcase our K-Means clustering algorithm. Before we do that, we must examine the data and formulate questions to motivate our research.

The set in question contains information about some customers who each own a membership card for a certain mall. Over the course of six months, you gather data on these customers including their age, income, gender and spending score. This is a metric, on a scale from 1-100, that measures customer activity (with higher values correlating to being a "better customer" both in frequency and total amount spent on purchases). A sample table is found below:



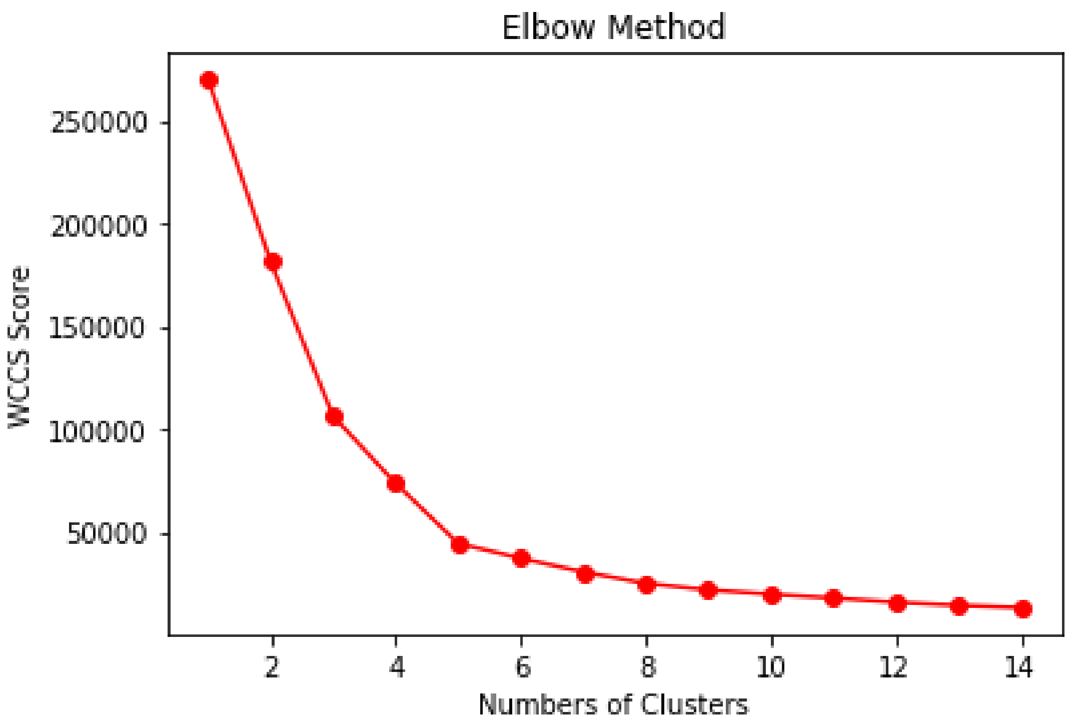
	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Based on this data, we wish to find clusters of customers to better understand shopping patterns in our mall, and we want to focus in particular on the income and spending variables. One question we might be interested in is how many discernable clusters could there be? We might also ask if there is a group that is spending well and we would like to reward their spending with increased benefits and discounts in our stores. Or perhaps, is there a group that is somewhat behind and we might be interested in marketing to them get them to become more fervent shoppers. And finally, we might postulate if there is a group that warrants churn consideration: could customers be on the verge of leaving their memberships due to dissatisfaction, and are there therefore measures that we can take to convince them to stay?

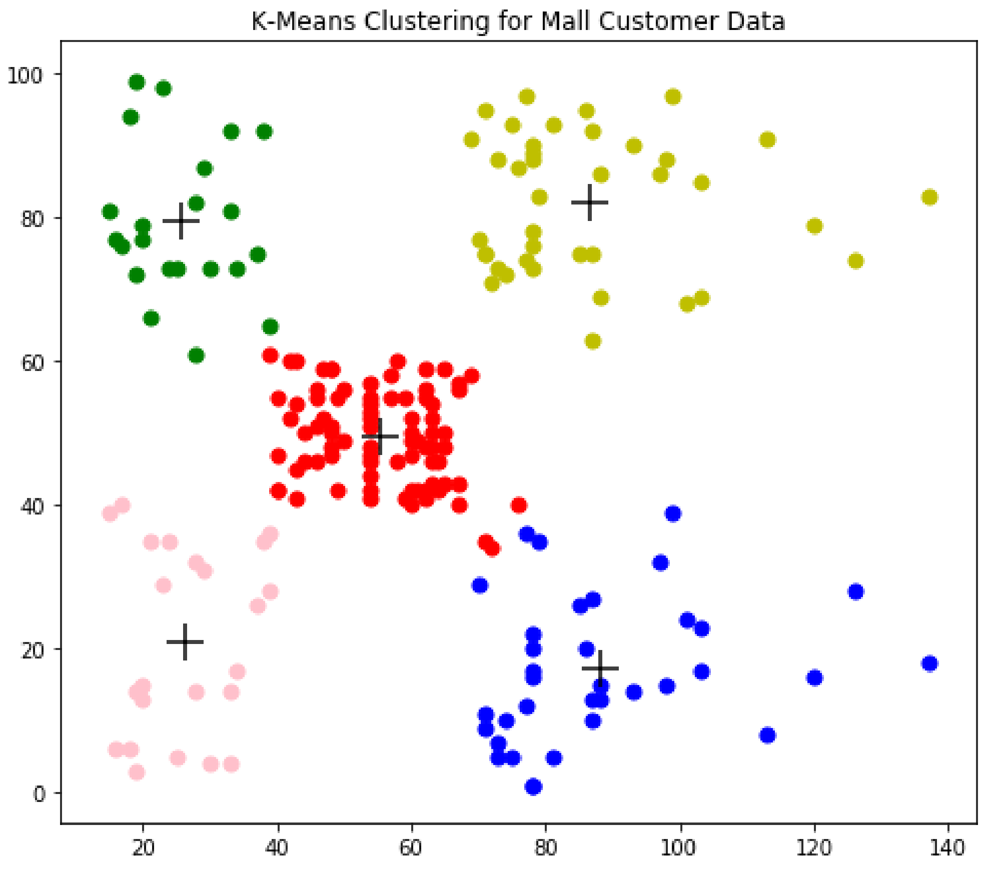
The KMeans algorithm is implemented in the corresponding Jupyter Notebook, but theoretical explanation is left to this report. The algorithm is designed to create "clusters" of the datapoints in the input matrix $\boldsymbol{X}$. Since this is a partitioning task, the criterion for the partitions is defined to be the distance between cluster centers. If N_k is the number of samples in cluster k, then the center is defined as: $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{i=1}^{N_k} \boldsymbol{x}^{(i)}$. This is also the mean of the cluster. The partitioning criterion, the distance measure, can be written simply as the norm of the data minus the mean: $d_{ik} = \left\|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\right\|$. If we compute the double sum of this value across all clusters K: $J = \sum_{k=1}^{K}\sum_{i=1}^{N_k}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\right\|$, we get the sum-of-squares error criterion, which helps to optimize (through taking the partial derivative of J with respect to $\boldsymbol{\mu}_k$) the distance and find the cluster centers. Defining these means is a recursive process that proceeds until a stopping criterion is met, and the distances are determined to be sufficient.

All that is left to do is to find the number of clusters K. While this must be user-defined prior to the implementation of the algorithm, it is not a trivial "guess and check" method. One popular way to compute this number mathematically is referred to as the Elbow method. This method computes the KMeans algorithm for every value in a specified range, say from 1 to 15, and then plots a value known in the documentation as WCSS. This stands for *within-cluster-sum-square*, and is a feature we would obviously like to have optimized. However, we also want to guard against overfitting our data.

The solution to this problem is to plot the WCSS for every value in the predefined range, and then look for the "elbow" of the graph. This is mean to represent the inflection point of the data, but due to its discrete definition it is more often said to look like an elbow. Below is said plot for this dataset:

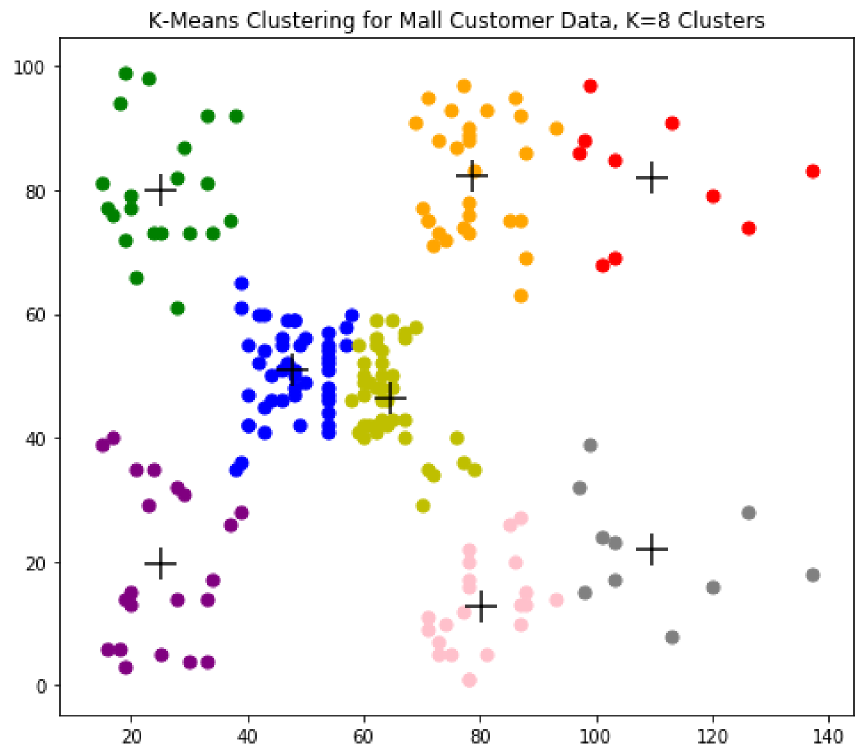


From here, we judge that the bend at $k = 5$ is the elbow of this graph, so we will set our algorithm to compute the KMeans for 5 clusters. Below is the plot of the results:



From a quick visual scan, it appears 5 clusters might have been as close to a perfect fit as possible for this data. What we notice is that there is a very distinct population in the center cluster and more sparsely populated ones on the corners. Recall that we did not require clusters to be of equal size in this algorithm.

This plot contains information on our two focus variables, income (in thousands of dollars, along x-axis) and spending score (y-axis). What we may notice first is that the data is far from linear: we cannot simply say that wealthier people are better shoppers at this mall. What we do notice is that this center cluster (in red) is far more populous than any other, meaning most customers have moderate income and are moderate shoppers. Interestingly, almost everyone who earns between \$40k-\$60k has a spending score of 40-60. This will be useful for predictions in the future. There appear to be two groups who are great shoppers (green and gold) and we might be interested in rewarding them with more benefits. The groups in pink and blue might represent customers who are dissatisfied, and we can try to approach them as well to try to convince them to stay.



Note that if we decided to go with more clusters, say $k = 8$, then three new clusters are formed, and two might be of particular interest, being the red and grey ones. Since these represent wealthier individuals, we might pay more attention to them when deciding to offer rewards or benefits.

References:

Gopal, M. (2020). *Applied Machine Learning* (1st ed.) [E-book]. McGraw-Hill Education.

https://www.gcumedia.com/digital-resources/mcgraw-hill/2019/applied-machine-learning_1e.php