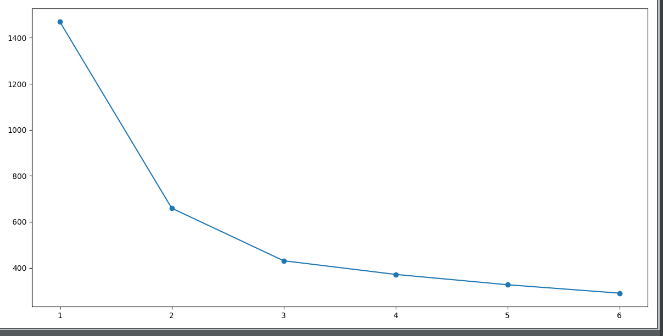
Report on Data Clustering – Kmeans and Gaussian Mixture Model.

Preparing data

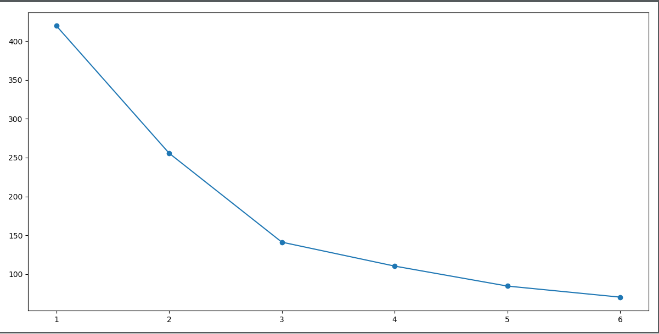
Before we could apply kmeans and gaussian mixture model (hereby referred to as GMM) to the seed dataset, we had to make the dataset fully processable. At first, we used regex to properly position data with respect to delimiters tabs and double tabs. Some columns were separated with more tabs in between than other columns. After the data was rightfully positioned, we used NumPy and pandas to convert the dataset to a NumPy array.

Balancing under- and overfitting – finding the right number of clusters

When starting to apply the algorithms, the last feature of the dataset was excluded. This column shows how many different clusters our algorithm should have to match actual classification. However, this was something we were to find out ourselves – if we were not given the classification in advance, maybe we would find another number of clusters classifying the seed dataset in a more effective way. We looked for a suitable technique for analyzing the dataset to find the ideal number of clusters. In both kmeans and gmm, specifying the number of components/clusters is key in avoiding under and overfitting. The greater the number of clusters, the smaller the number of wrongly estimated datapoints. However, by overfitting the dataset, the number of clusters would surpass the real number of clusters and become too specific. A number of clusters too small, underfitting, would result in a too general classification. The techniques we applied to our dataset were ‘elbow analysis’ and ‘silhouette score analysis’. When using the elbow analysis, we wanted to identify the cluster where the gain in reducing wrong classifications by adding more clusters are insignificant – we looked for the graph’s “elbow”. At first, we applied the elbow method on our dataset without any specific preparation, shown in figure 1. When analyzing the graph, we had difficulties finding the graph’s elbow point: it was unclear whether two or three components would be the best fit. After a period of research, we found that dimensionality reduction solved this problem. By extracting the least significant features, we end up with a space with fewer dimensions than originally when applying PCA. The information variance of the data we get is maximized. Looking at figure 2, we can identify a much more clear elbow point at the point of three clusters by the x-axis.

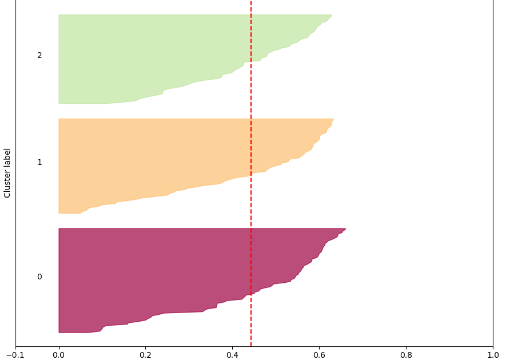


*Figure 1 – elbow analysis without PCA reduction. X-axis: number of clusters, Y-axis: Percentage of information variance*



*Figure 2 – elbow analysis with PCA reduction. X-axis: number of clusters, Y-axis: percentage of information variance.*

We also used the silhouette method to ensure that three clusters were the right number of clusters to pursue. The silhouette method presents graphical presentations of how well different variances of clusters perform in obtaining the datapoints. The higher score at the x-axis, the better the number of clusters matches a dataset, and the less the datapoints obtained in the specific cluster shares feature similarities with other clusters’ datapoints. Figure 3 shows the best score, which was by dividing the dataset into three clusters. We used PCA reduction also for the silhouette method.



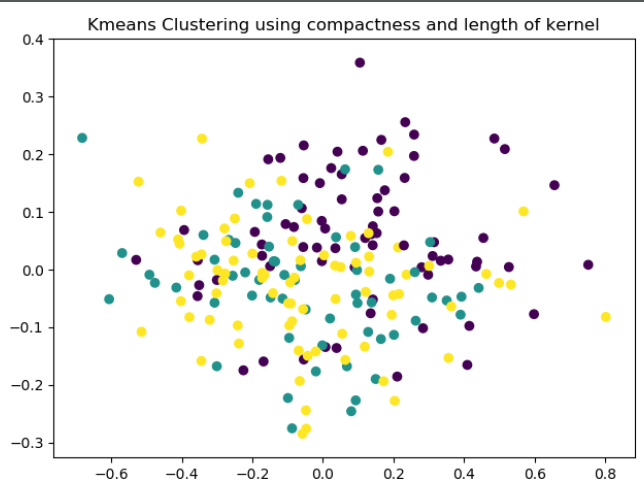
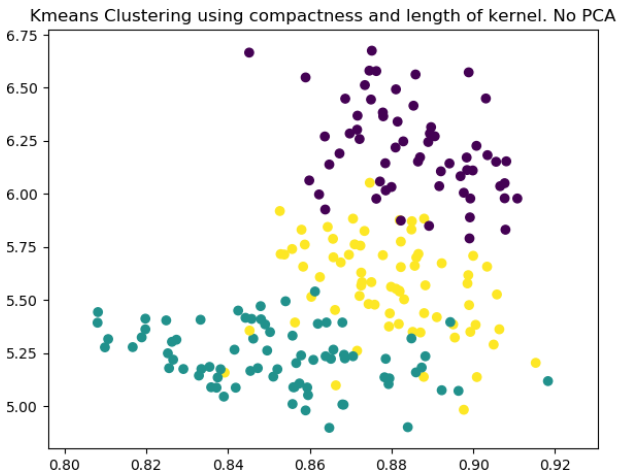
*Figure 3 – Silhouette method with PCA reduction. X-axis: cluster label. Y-axis: score.*

Applying the clustering algorithms

Furthermore, we fitted kmeans and the GMM to our dataset. We could not measure how the two methods performed before visualizing them through a 2D chart. For visualizing the dataset, we used the matlotlib library. Here, we had to decide which two features of seven that provided the best cluster variance. We were easily able to tell that the two first features of the dataset, ‘area’ and ‘perimeter’ respectively, were the most cluster-significant features. Despite being aware of this, we tried various other feature combinations, trying to find the visualization where datapoints overlap the least. For the parameters included in the kmeans and the gaussian mixture model sklearn classes, we did a lot of tweaking to find the values that gave valuable results. We also experimented by including and excluding PCA reduction through the process.

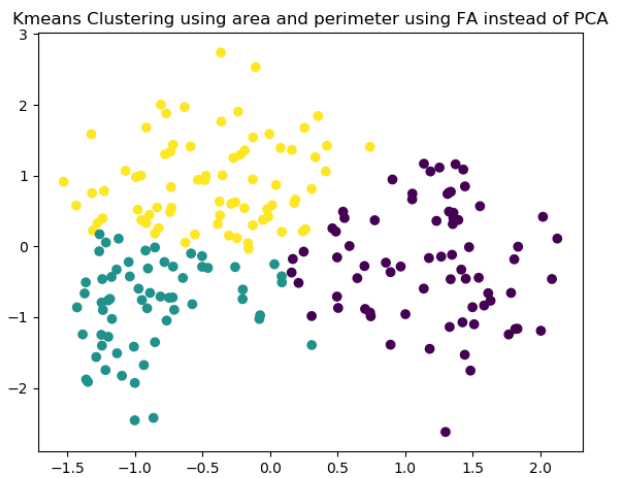
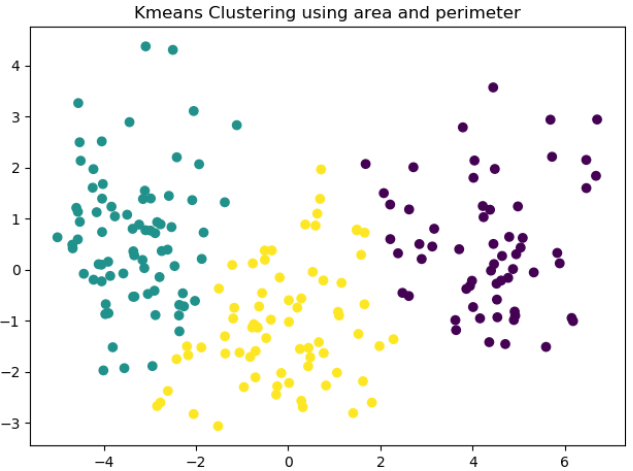
Kmeans

All other combinations than including the two first combinations for the 2D chart, gave poor results. Feature three and four (compactness and length of kernel) shown in figure 4 and 5, shows an approach to the dataset where the datapoints overlap and are very little clustered. An interesting finding when trying various feature combinations that did not cluster well, was observing that some combinations worked better without dimensionality reduction. We found that figure 4 shows a relatively satisfying clustering result.



*Figure 4 and 5 – kmeans with compactness and kernel length as features with and without PCA reduction.*

It was easy to figure out that feature one and two clustered our dataset the best, shown in figure 6. Here we used PCA reduction for dimensionality reduction, and reduced dimensionality to two components. Adjusting the component parameter for the PCA class had to be done with respect to which features we tried to combine, not to exceed indices. There were a lot of other parameters in the PCA class to experiment with, but none of them had any significant effect for us as they are made for bigger datasets, according to the sklearn docomentation. For the kmeans class parameters, we tried the various algorithm executions, but they all did the same clustering. In addition to applying PCA, we tried another reduction method: factor analysis, shown in figure 7.



*Figure 6 and 7: kmeans with area and perimeter as features, with PCA reduction in figure 6 and with FA reduction in figure 7.*

We found FA reduction also performed well, but that PCA reduction clustered the best.

Gaussian mixture model

Just as we did for the kmeans method, we experimented with various parameters, both for the GMM class and for PCA and FA reduction, when applying GMM. The GMM class has a parameter Here too, we experienced that feature combinations other than the first two features, resulted in fairly good clustering, but only without the use of