Electricity Consumption Segmentation and Prediction

Group 14, project 5 – Håkon Teppan, Kristoffer Rødne, Sondre Tennø

Project

In this project, we were given a dataset which provided daily electricity usage for 6445 houses and companies between 2009 and 2010. The purpose of the project was to group customers with similar electricity consumption behavior, analyze their behavior and predict loads consumption of each cluster.

Summary of Work

Clean the data

We used a Pivot Table to convert the dataset to a format where the dates are the index, the different MeterIDs are the columns, and the values are the total kW per day. By turning it to a Pivot Table there are generating some NaN values. This is because some of the customers did not partake the whole period. We removed meterIDs with more than 50 % missing days. MeterIDs with missing values that remains after this, will get its NaN values filled with the average value (.mean()).

Adding Features

We decided to use the following features: Total kW, Average daily kW, percentage of Monday to Sunday, percentage of weekdays and weekends, and percentage for the different seasons.

Standardizing the data

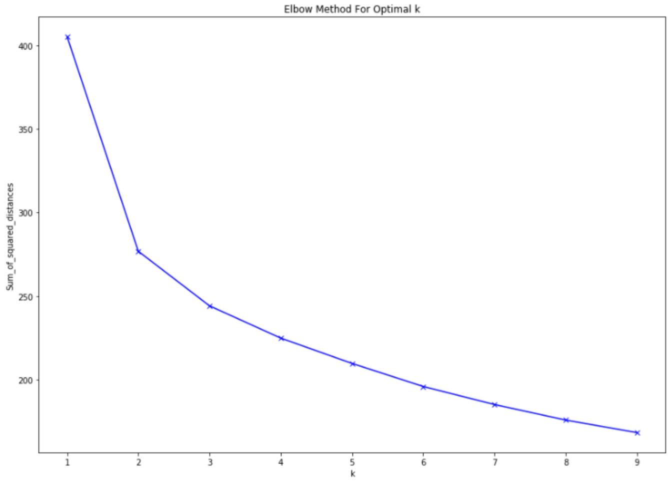
Whenever we perform clustering it is always a good idea to scale out our data to give equal importance to all the variables, in this way we can avoid misinterpretation of the clustering algorithm to make the clusters. As our data consists of continuous values, hence we prefer to use MinMaxScaler to serve the purpose. MinMaxScaler gives the dataset a mean of 0 and a standard deviation of 1.

Cluster Technique

Firstly, we plotted the dataset, using PCA for two dimensions. We could see It is hard to distinguish the different clusters just by looking at the plot. We decided therefore to test different clustering methods, which are Agglomerative Hierarchical and KMean clustering. These cluster methods are good for datasets with relatively short distances between the clusters.

To test our different clustering methods, and what number of clusters to use, we used different techniques to find the optimal ones. We used Silhouette Coefficient to test the performance of the different clustering methods. To get the optimal number of clusters we used again the results from the Silhouette Coefficient and BIC score (elbow-method).

The Elbow-method

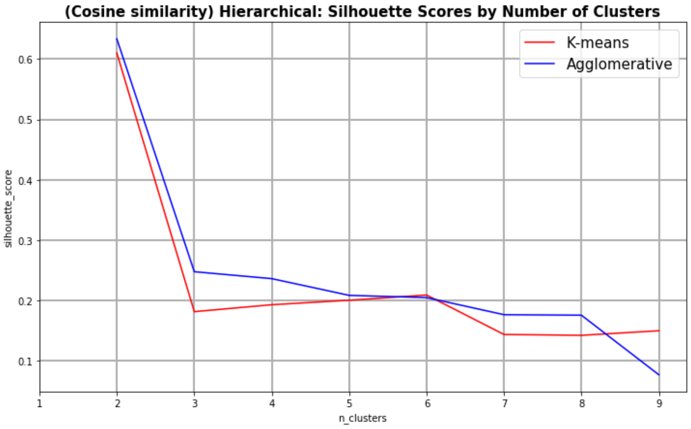


To figure out how many clusters we wanted to use in our project, we used two of several ways to determine this. First we used the elbow method, and then we used the silhouette method and compared them to get optimal number of clusters.

We used the built in K-means from the scikit library to get the "sum_of_n_squared" numbers to show us on the graph visually which clusters we should use. The x-axis are the number of clusters, and y-axis shows the sum of n squared instances.

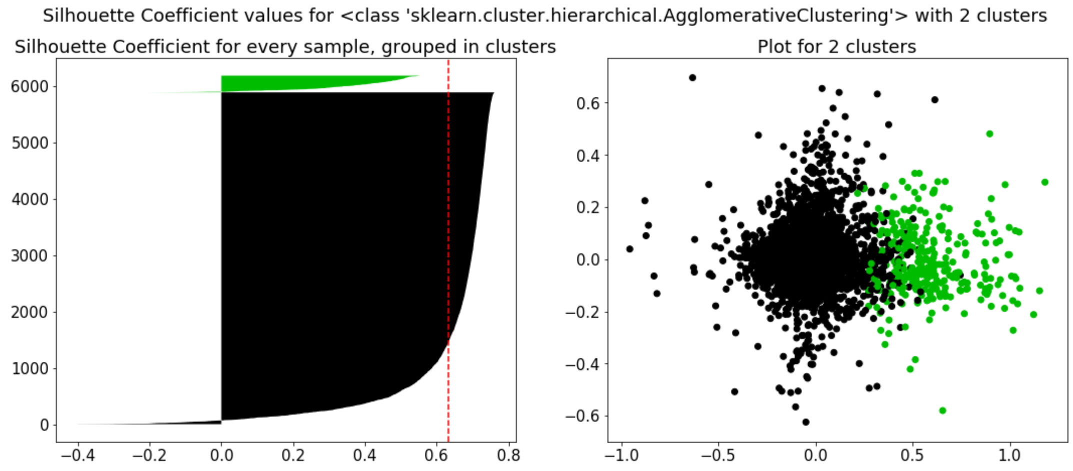
The right amount of clusters is suggested by picking the number, where the elbow "breaks", and it starts to flatten out. Several different answers could be the answer sometimes, if there are no significant breaking point. In our project, both 2 and 3 clusters could be the answer, since both could be explained as where the elbow breaks.

Silhouette-method



The silhouette_score gives the average value for all the samples. This gives a perspective into the density and separation of the formed clusters. This technique provides a graphical representation of how well each object has been classified.

The silhouette value is a measure of how similar an object is to its own cluster compared to other clusters. The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

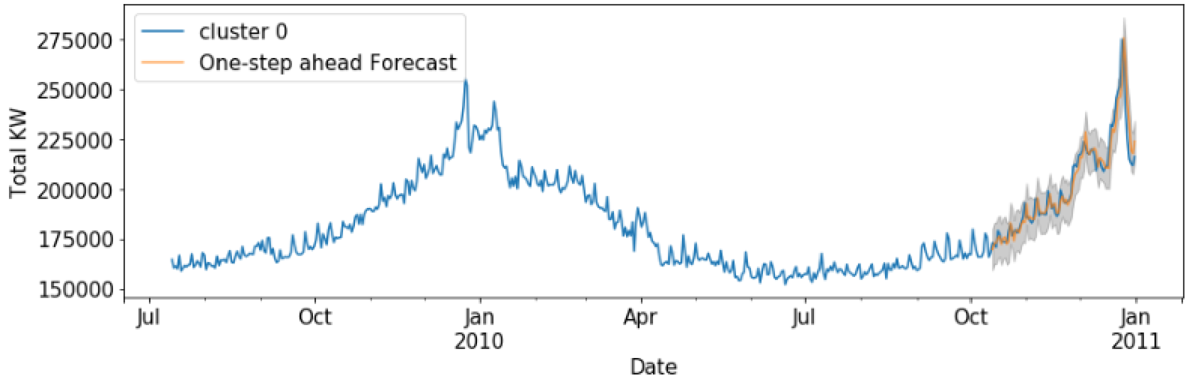


With the silhouette method it is easier to see the optimal amount for cluster number, than the elbow method.
In both methods, we can see that we get two clusters as a possible answer. Since Agglomerative as a higher score, we decided to use 2 clusters with Agglomerative clustering for this project.

Create a Prediction Model for the total daily consumption

In order to find which group of customers has more predictable consumption behavior, use a machine learning algorithm to predict the total daily consumption of each cluster in the last 80 days of the dataset.

We have used a machine learning algorithm called SARIMAX to do the prediction of the total daily consumption of each cluster. Before we were able to use this algorithm we needed to get our cluster ID's into a dataframe containing dates(which we didn't have because when we cleaned the data we removed the dates and instead used the Meter ID's as rows). When we had created the dataframes for each cluster, we simply used the dataframe from each cluster to use a prediction method from
SARIMAX.



The model we used is called statsmodels.tsa.statespace.sarimax, and from that fitted model we got our predictions by using the predict method within SARIMAX. Above is a sample of the results of cluster 0, and as you can see the prediction is quite close to the actual value of cluster 0. To get these results plotted we used another method within SARIMAX called get_prediction.

Evaluation of Prediction Model

Root Mean Squared Error:

RMSE(Root Mean Squared Error) is a popular formula to measure the error rate of a regression model. It represents the sample standard deviation of the differences between predicted values and observed values.

While still being more complex and biased towards higher deviation, RMSE is still the default metric of many models. RMSE are used on the train and test data for machine learning algorithms. When compared to other RMSE from the same data, the lower RMSE score is favored.

Mean absolute error:

MAE (Mean Absolute Error) is very similar to RMSE.

MAE is the average of the absolute difference between the predicted values and observed values. The MAE is a linear score which means that all the individual differences are weighted equally in the average. For example, the difference between 10 and 0 will be twice the difference between 5 and 0. However, the same is not true for RMSE.

MAE directly takes the average offsets whereas RMSE penalizes the higher difference more than MAE.

Final:

We ended up with using the Agglomerative algorithm, with 2 clusters.

Our MAE scores:
Cluster 0: 4029
Cluster 1: 2769
All clusters: 5349

Our RMSE scores:
Cluster 0: 6208
Cluster 1: 4015
All clusters: 7995

Optimal RMSE score are usually lowest for the best fit cluster. It indicates the absolute fit of the model to the data–how close the observed data points are to the model’s predicted values. So in this case, the RMSE score indicates that cluster 1 has a better fit than cluster 0. Both clusters have good RMSE score in general and are objectively good clusters. We have a lot more instances in cluster 0 than in cluster 1, which can skew the results.

The MAE score is also lower for cluster 1 than cluster 0, which indicates the absolute difference between prediction and actual observations. Since MAE is lower for cluster 1 than cluster 0, we have a lower margin for error in cluster 1. We have a lot more instances in cluster 0 than in cluster 1, which can skew the results.

Contribution of Each Member

For this project, we separated the work, so that each member had some primary parts they were responsible for. It was separated like this:

Håkon:
The silhouette score graphs and adding features, and being the lead of the code.

Kristoffer:
The prediction model and the data cleaning process.

Sondre:
The illustration of the data, creating the elbow graph and the evaluation process. In addition, being the lead of the report.

We are a small group, so it felt naturally to partake in more than the above suggestion. Most of the modules are completed as a result of all three members partaking and giving their participation.