Tennis Matches

Edoardo Federici 616667
Michele Morisco 505252
Carlo Tosoni 644824

Data Understanding

Data Quality

- **Duplicates**
  we removed all duplicated records

- **Missing Values**
  if there were too many blank attributes or if we had no way to extract the correct value by cross-referencing other attributes, we deleted the missing values

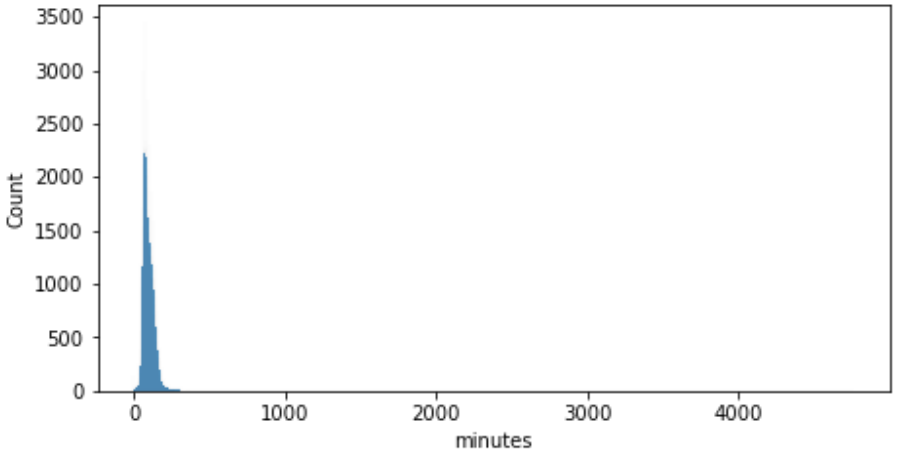
- **Ad-hoc replacement**
  when we could infer the correct (or a unique identifier) value for the analyzed attribute

Outliers

We studied the dataset's **outliers** comparing the distribution before and after their removal.

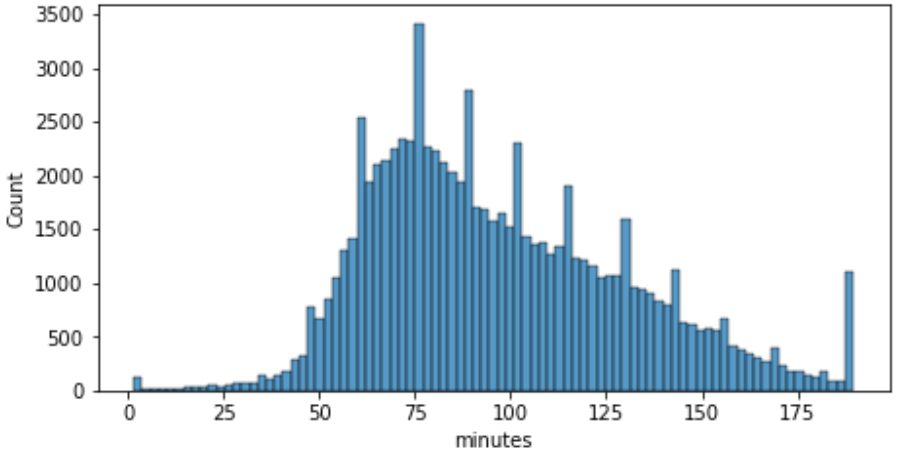
In our analysis we have noticed the outliers in some features, for example:

- **Minutes** of match
- **Height** of players - no benefit with outliers removal
- **Age** of players - no benefit with outliers removal



We focused on the **minutes** attribute.

We have noticed that the values are not approximately normally distributed.

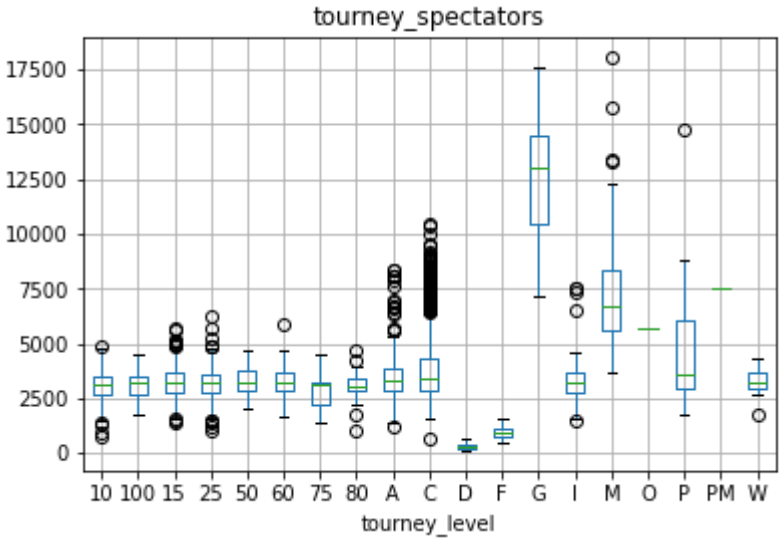


To detect the outliers, we have used **Interquartile Range (IQR) proximity rule**.

After detect upper and lower limit, we removed each value $\notin$ [lower limit, upper limit]

Data Distribution

We have deeply studied the data distribution in our dataframes, plotting many charts and commenting them.



An example is given on the left, here we plotted the relation between the number of spectators of the tourney and its level.
As we can see, Grand Slam (G) and Masters 1000s (M) are the categories with the highest number of spectators.

Data Preparation

Indicators

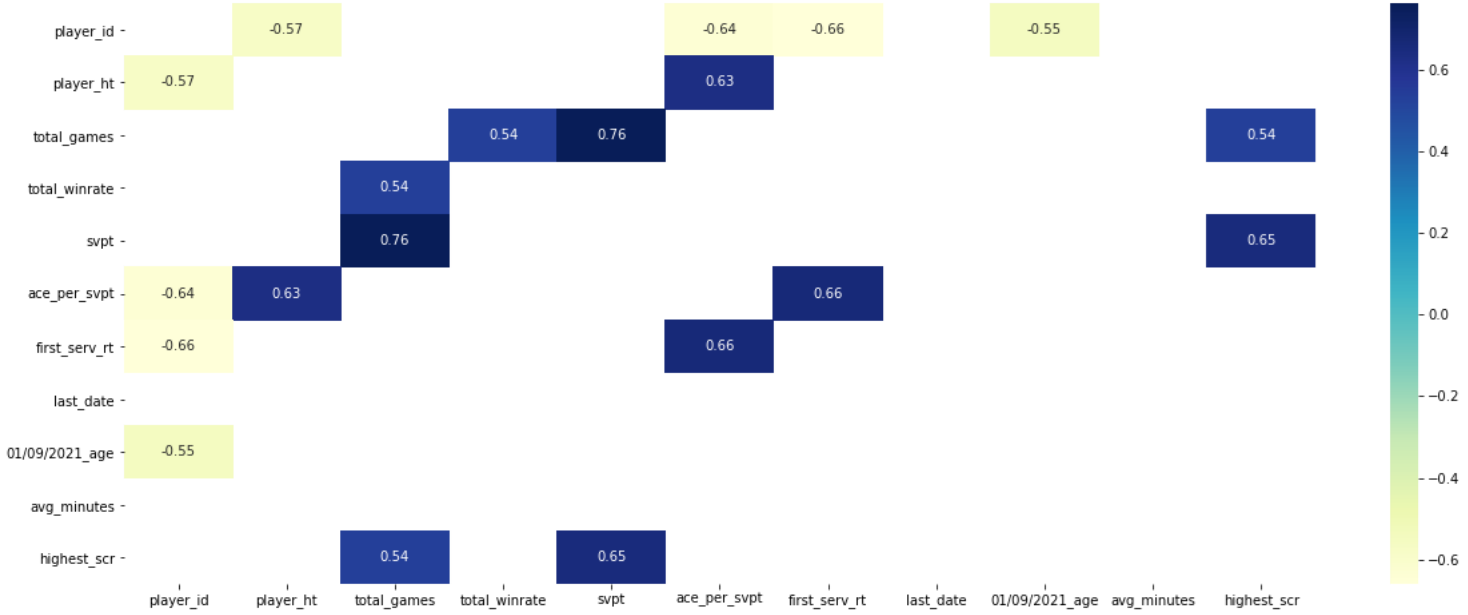
To get a better understanding of the **skill** level of each player, we created different artificial performance indicators. Among them we can find:

- Wins, losses and winrate for each year (**total_wins**, **total_losses**, **total_winrate**)
- The total number of aces performed per serve point (**ace_per_svpt**)
- The percentage of points scored with the first service (**first_serv_rt**)
- The average number of aces performed by a player in a match (**avg_ace**)
- The percentage of breakpoints saved (**perc_bs**)

To produce these indicators we both referred to our **intuition** and to a study: Winning matches in Grand Slam men's singles: An analysis of player performance-related variables from 1991 to 2008

Correlation

We found that many of the original attributes had a correlation value higher than 0.80, so we decided to **remove** them. The process left us with **9** total attributes to work with for the clustering analysis.



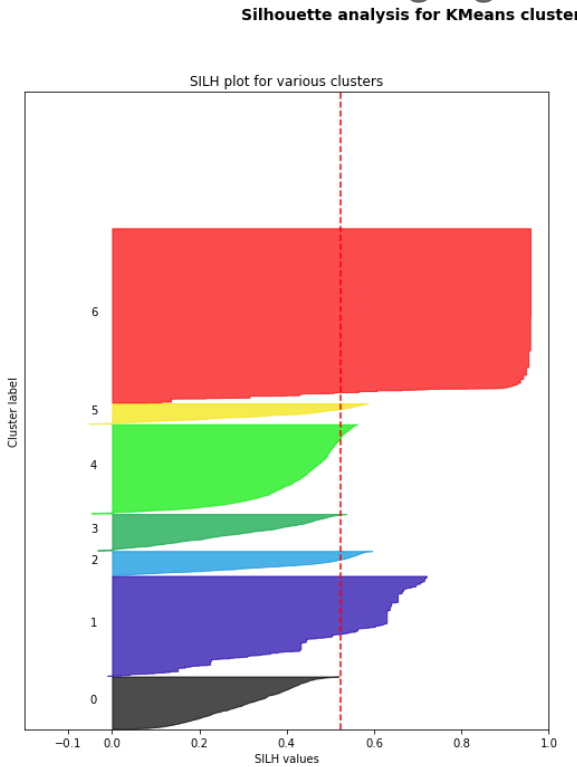
Clustering

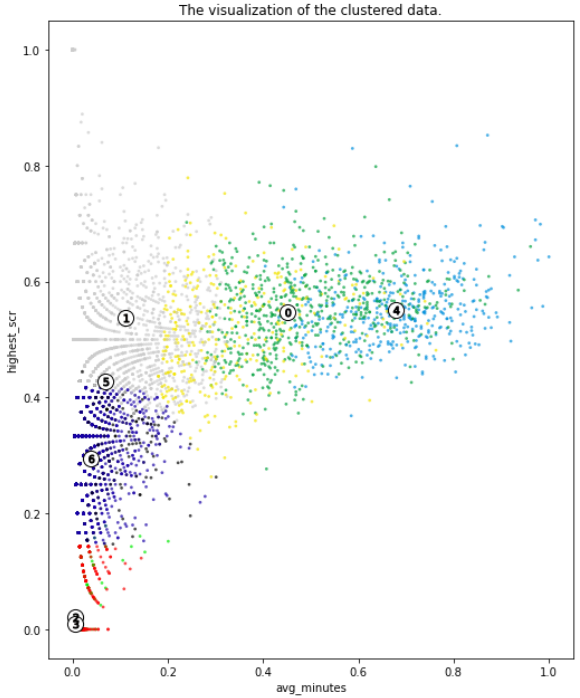
K-Means

We ran the **K-Means** algorithm with values of k ranging from 2 to 30, computing the **SSE** and the **Silhoutte** Score.

We then **refined** the search comparing the Silhoutte Score with the cluster size.

We obtained the best results with k=**7**.





DBSCAN

We computed **DBSCAN** using many values of **min_samples** (from 2 up to 9). For each value of **min_sample** we found the best **eps** through the **elbow method**.

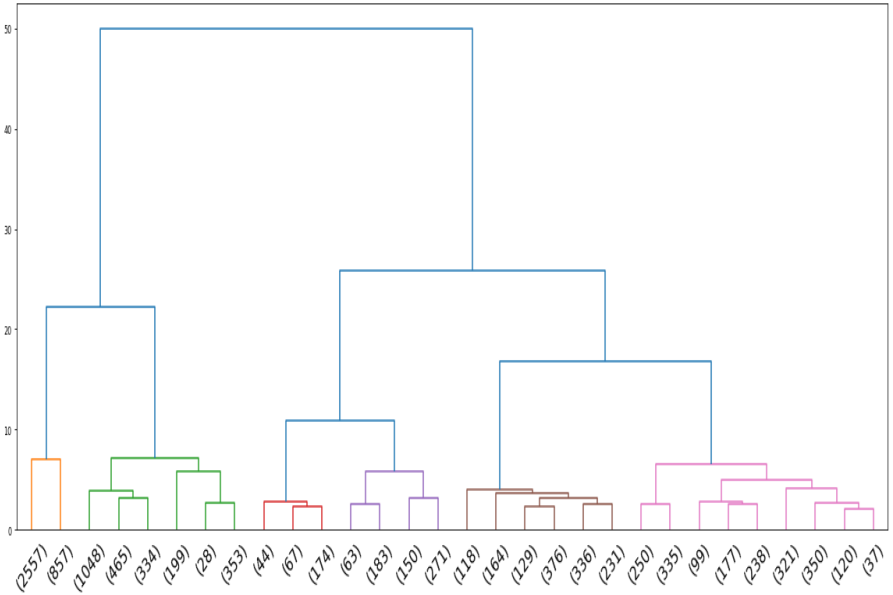
The clusters obtained by this algorithm were not amazing, in fact, for instance, there were many **noise points** (about one-tenth of the points).

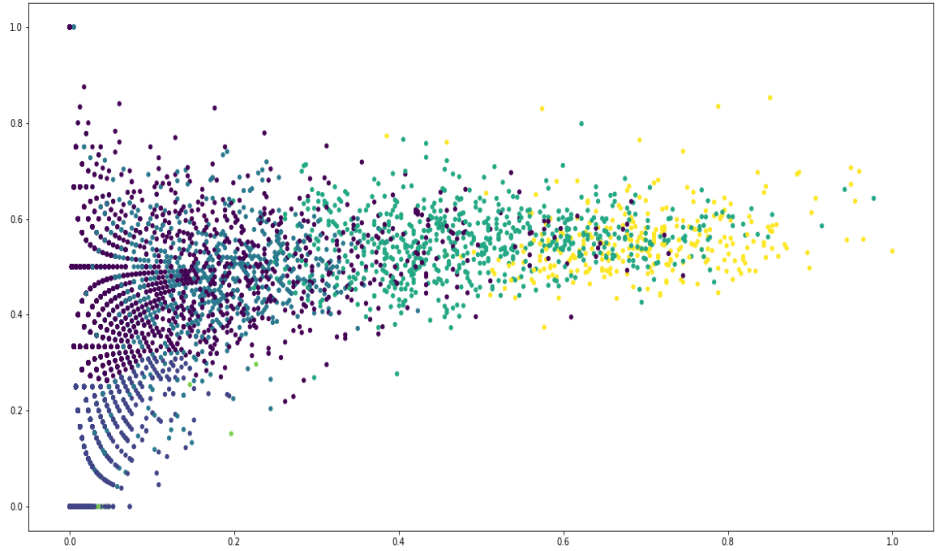
Moreover, the clusters obtained were not exceptional, since in every execution of the algorithm most of the points belonged to a huge cluster (more than half).

Hierarchical clustering

We have used some methods to compute hierarchical clustering.

Ward - best results

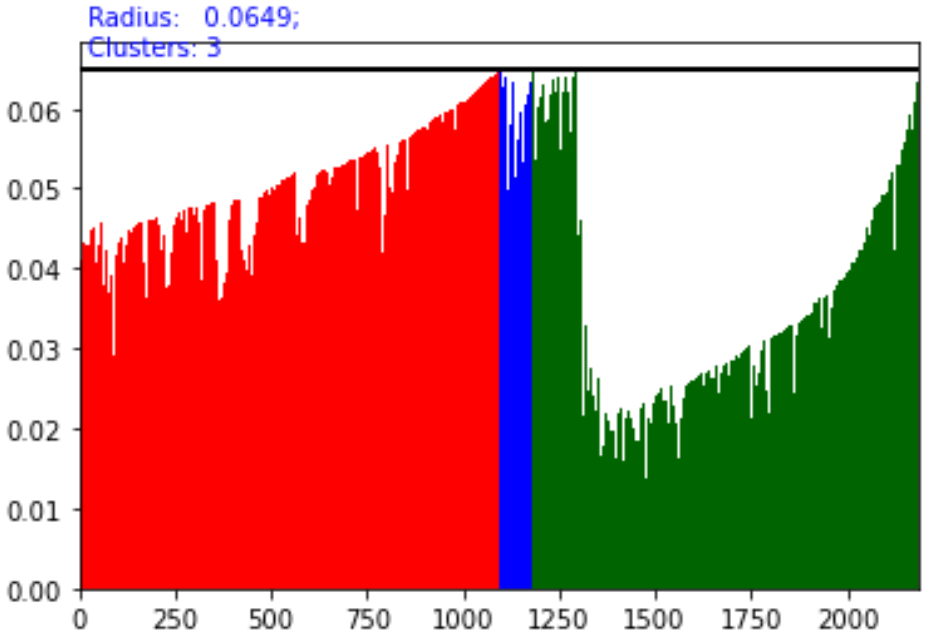




OPTICS

The main difference between **DBSCAN** and **OPTICS** is that in the latter the density used to identify the clusters is not equal for all the clusters.

However this algorithm was the one that performed the worst, probably because our dataframe cannot be divided according to different levels of density among its points.



Analysis

We obtained the best results with the Ward method of the Hierarchical Clustering. Here is a proposed interpretation of them:

- **Cluster 1** Players with low score, they played about 30 matches in average **(2427 points)**
- **Cluster 2** Players whose score is close to zero, they played about 5 matches **(3414 points)**
- **Cluster 3** Players with a medium score, they played about 30 matches **(1927 points)**
- **Cluster 4** Players with a good score that played many matches (about 200) **(667 points)**
- **Cluster 5** Players with a bad score that played only 1-2 matched **(1354 points)**
- **Cluster 6** Champions, amazing score and many matches played **(285 points)**

Predictive Analysis

Label Computation

We used 5 attributes to compute the labels for our dataframe: **first_serv_rt**, **fvrt_level**, **perc_bs**, **highest_scr**, **avg_scr**.

The label **1** indicates **high-ranked** players, while the label **0** is for **low-ranked** players.

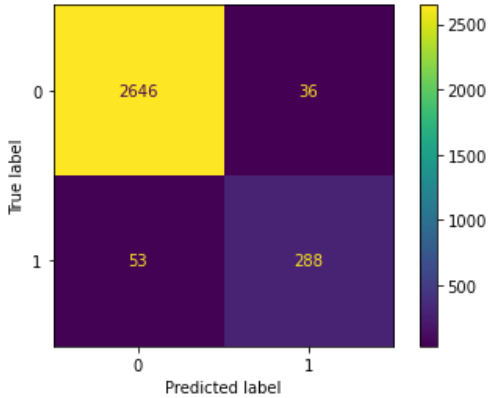
We tried to use the other **indicators** (computed in the **Data Preparation's Task**) to predict which label we should assign to each player.

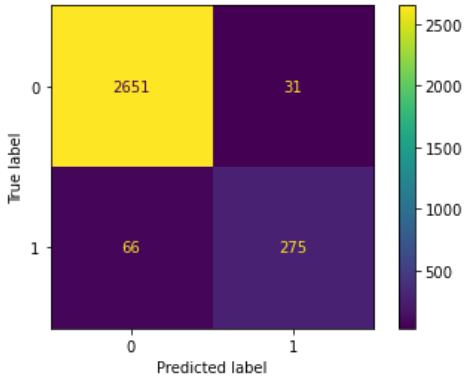
Firstly, we have selected only a subset of these attributes, pruning attributes that were strongly correlated.

Decision tree

We have tested two versions of the DT: one with the ’random’ strategy splitter and one with the ’best’ strategy splitter.

We have achieved the best result with *min_samples_split* equal to 4 and *min_samples_leaf* equal to 6.





'Best' Decision Tree			
Label	precision	recall	f1-score
low ranked	0.98	0.99	0.98
high ranked	0.89	0.84	0.87

'Random' Decision Tree			
Label	precision	recall	f1-score
low ranked	0.99	0.98	0.98
high ranked	0.84	0.90	0.87

Support Vector Machine

SVM			
Label	precision	recall	f1-score
low ranked	0.99	0.99	0.99
high ranked	0.92	0.99	0.95

To achieve the best result possible through the **Support Vector Machine**.

- Firstly, we searched the best values for the parameters **C** and **gamma** through the **Grid Search**.
- Secondly, we tried to downsample our dataframe to reduce the numbers of points.
- Finally, we used the **Principal Component Analysis** to further improve the results.

The results achieved using the **Grid Search** and the **PCA** were very good, while the **Downsampling** strategy performed poorly.

SVM with downsampling			
Label	precision	recall	f1-score
low ranked	0.95	0.99	0.97
high ranked	0.96	0.63	0.76

Neural Network

Perceptron			
Label	precision	recall	f1-score
low ranked	0.98	0.98	0.98
high ranked	0.91	0.87	0.89

We settled on a **single layer feedforward** neural network with an added **dropout** layer right after the input layer.

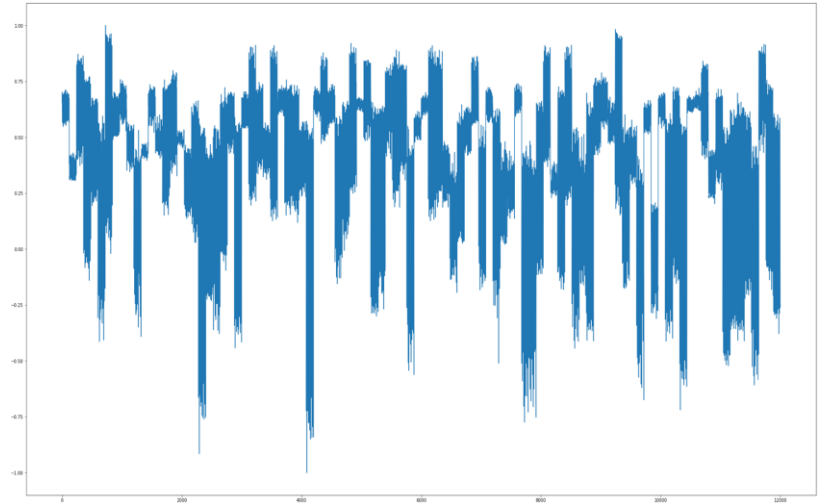
We tried other configurations (both in depth and width) but they either resulted in poor performance or in overfitting.

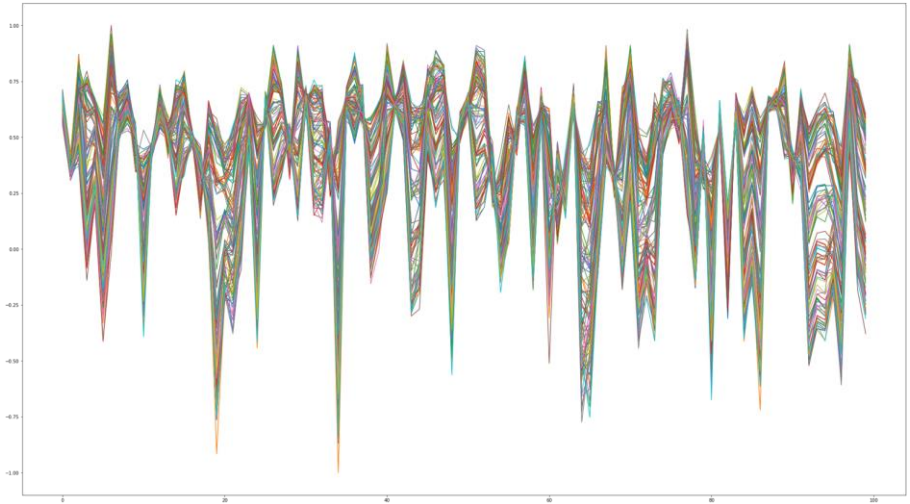
dropout	0.2
learning rate	0.01
neurons	8
batch size	12
activation	relu
momentum	0.001

Time series analysis

Removing noise

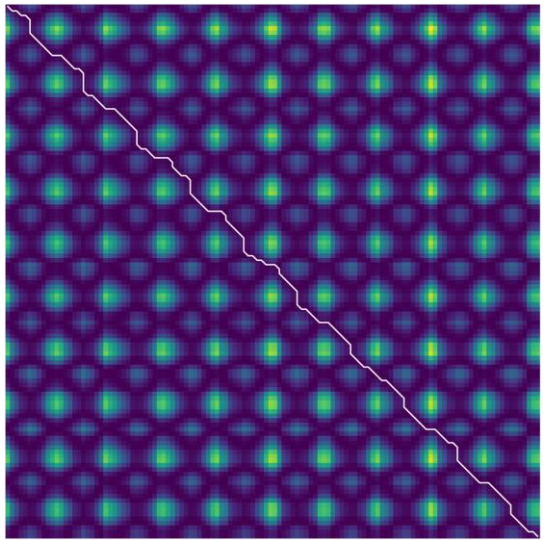
First, we have removed the noise by applying the **min-max** scaler with range [-1, 1]





Distance between TS

In the analysis phase, we have computed the distance among some of the time series to examine the similarity of some temperature trends.



In our analysis, we have found a good result between a group of time series

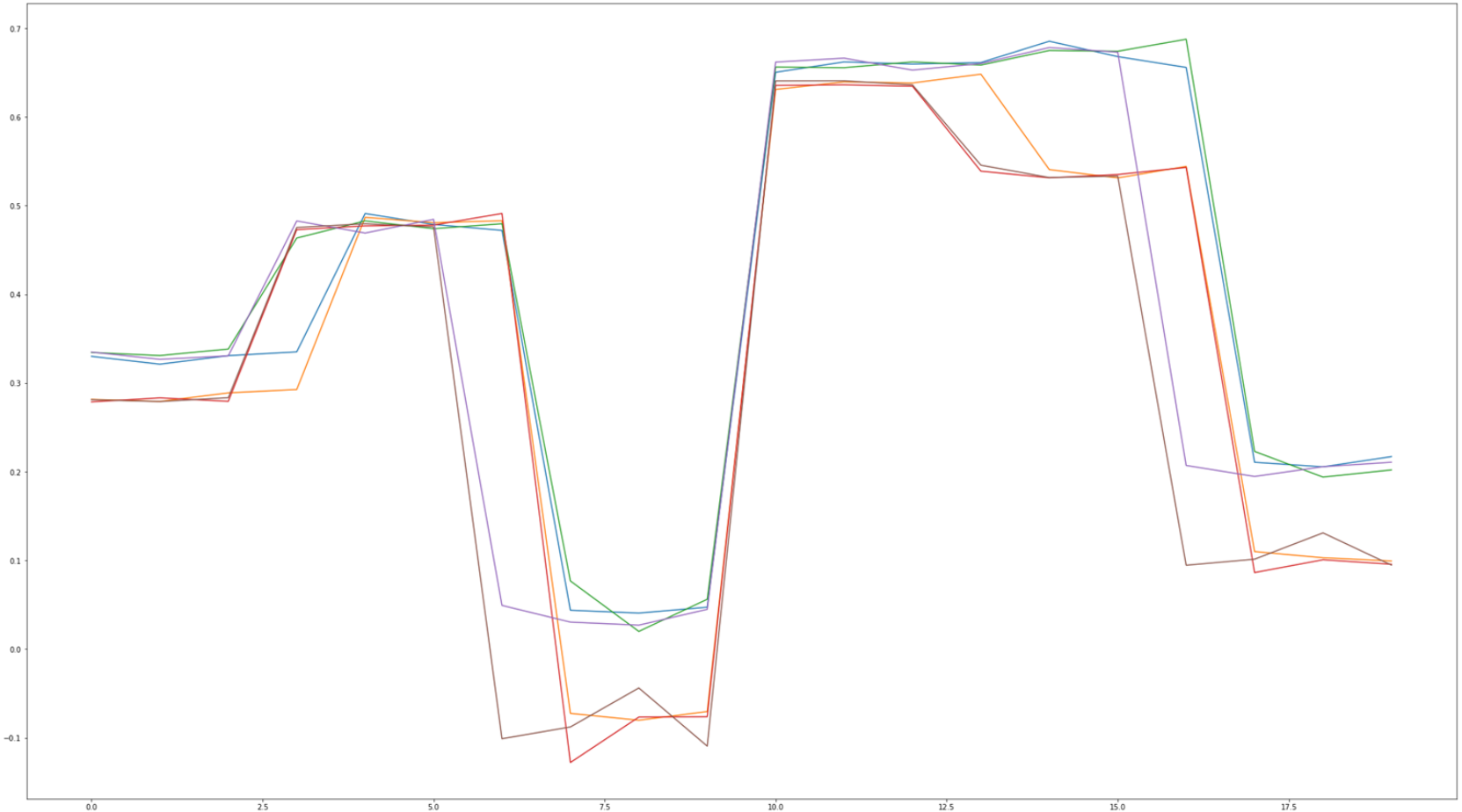
Using **itakura** constraint

The distance path amounts to 1.48.

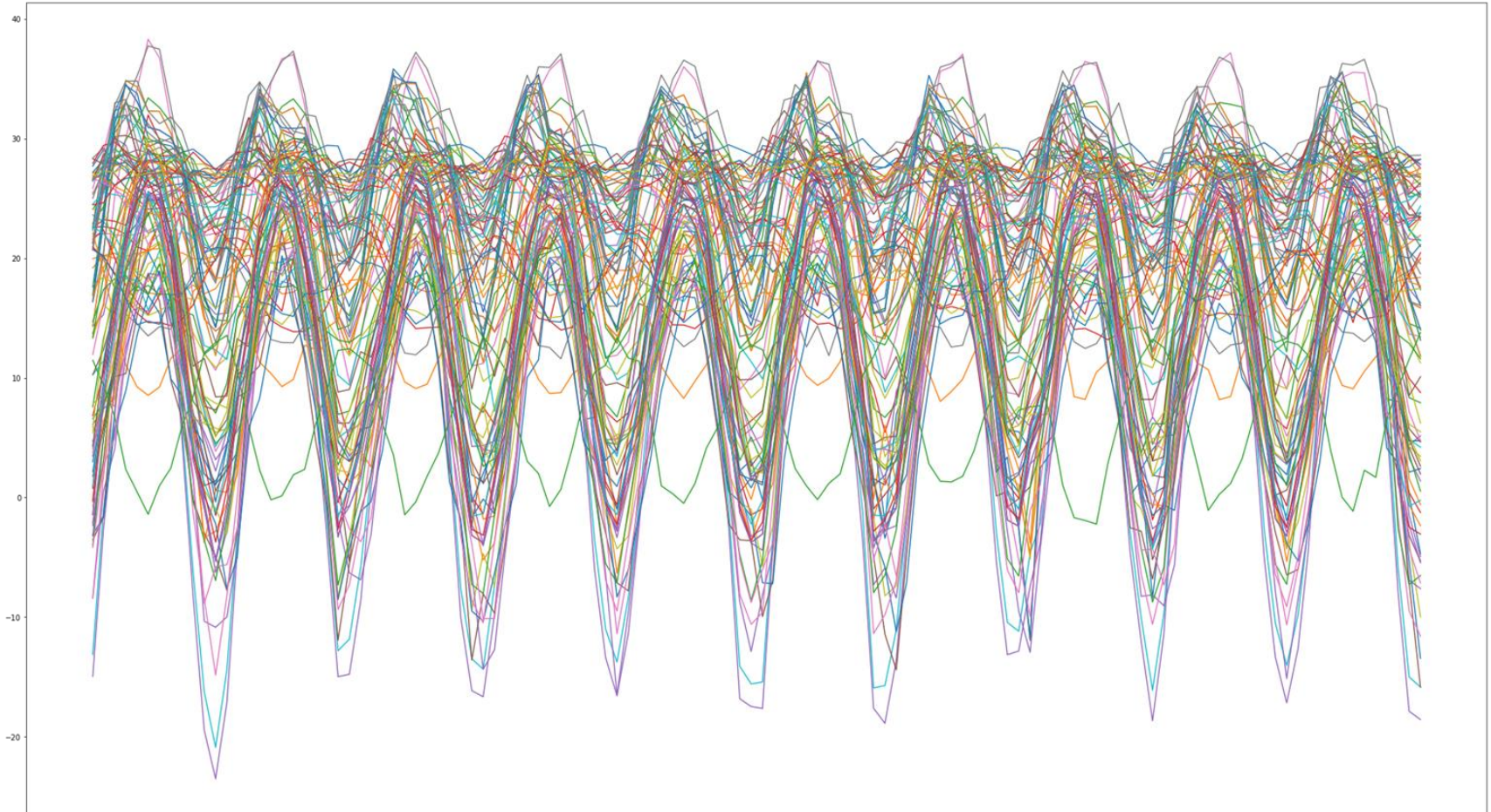
TS Clustering

- **Shape-based** clustering
  - k=7
    - euclidean distance: 1.01 inertia
    - dtw distance: 0.43 inertia
- **Feature-based** clustering
  - k=7
    - 183.7 inertia
- **Compression-based** (piece wise aggregation) clustering
  - k=6
    - euclidean distance: 245.2 inertia
    - *** dtw distance: 0.47 inertia ***

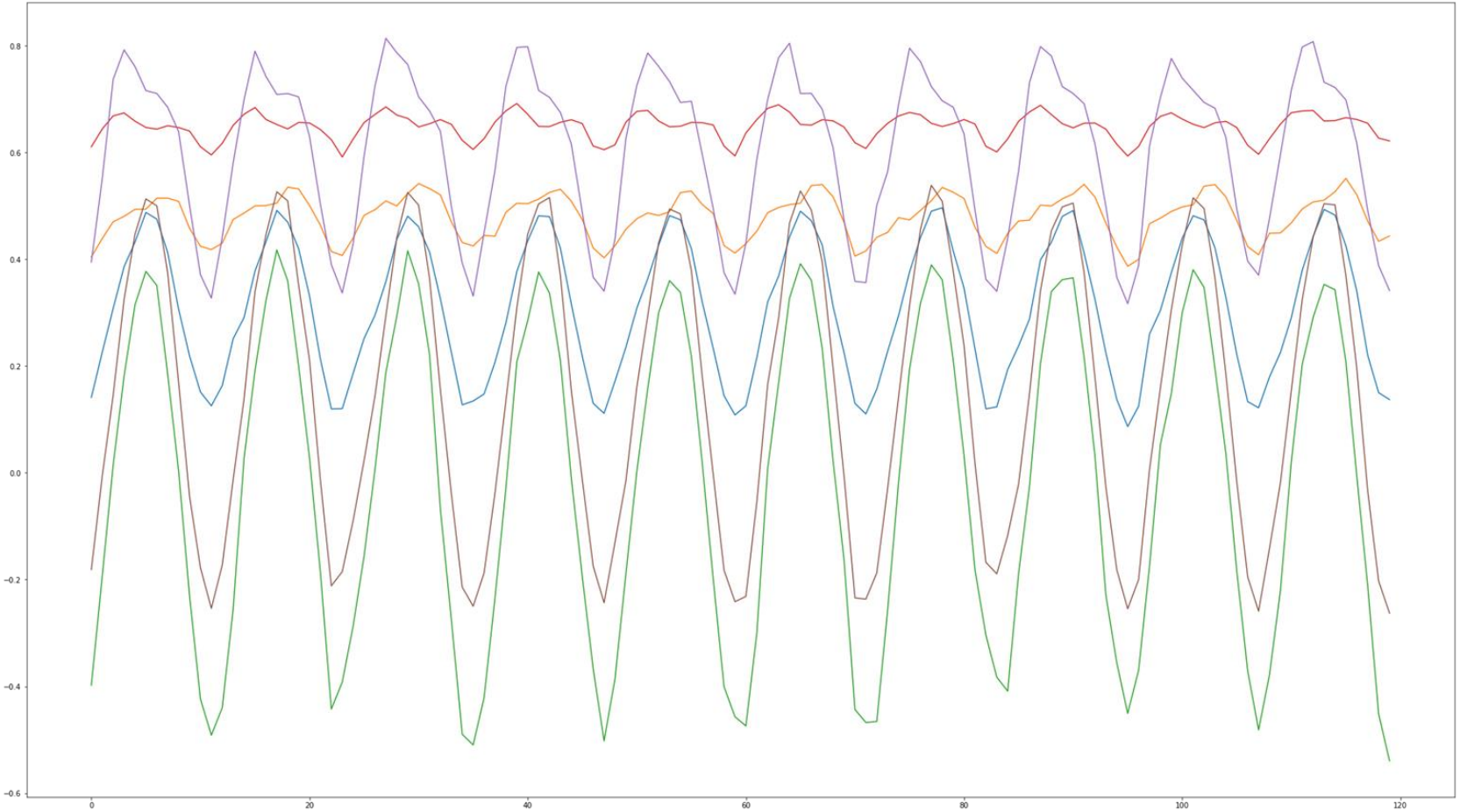
TS Clustering - PAA



TS Clustering - Original



TS Clustering - Final



Thank you =)