RZESZOW UNIVERSITY OF TECHNOLOGY
FACULTY OF MECHANICAL ENGINEERING AND AERONAUTICS
DEPARTMENT OF COMPUTER SCIENCE

DECISION SUPPORT SYSTEMS

LEARNING MATERIALS

PART 2

Time series in machine learning: time series clustering

Co-funded by the Erasmus+ Programme of the European Union

The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

COURSE INDUSTRIAL ENGINEERING
GROUP I MP-DU

Developed by: Łukasz Paśko

Table of contents

2021/2022

Introduction

Objectives of the classes

- Presentation of the concept of clustering time series based on their characteristics.
- Feature extraction from time series.
- Application of non-hierarchical methods for clustering.

Time series - a sequence of data (observations) that are **ordered** (the order of the data is important).

Clustering (grouping, patternless classification) - the task of finding clusters, also called clusters or groups, in a set of natural objects.

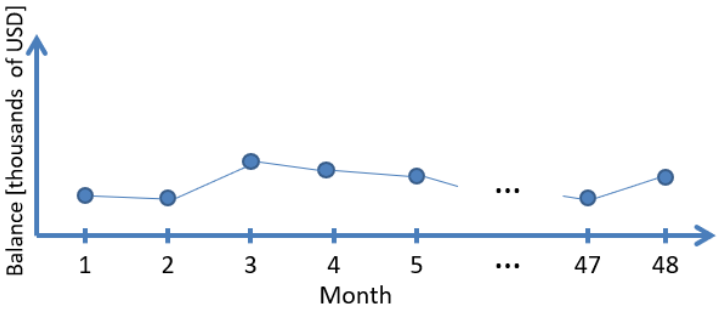
Time series clustering is an important task because:

- Facilitates the discovery of patterns present in the data - generating clusters for a certain set of data makes it easier to understand the structure of the data, allows you to more easily spot anomalies or other types of patterns present in the time series,
- Makes it easier to review large amounts of data - datasets containing time series can be huge (especially nowadays), making it difficult for an analyst to review such a large amount of data,
- Clustering is the most commonly used exploratory technique - it is part of the more complex tasks of:
    - Discovering rules (patterns) in time series,
    - Classification of time series,
    - Detection of anomalies in time series.

The example used in classes

Clustering of bank customers

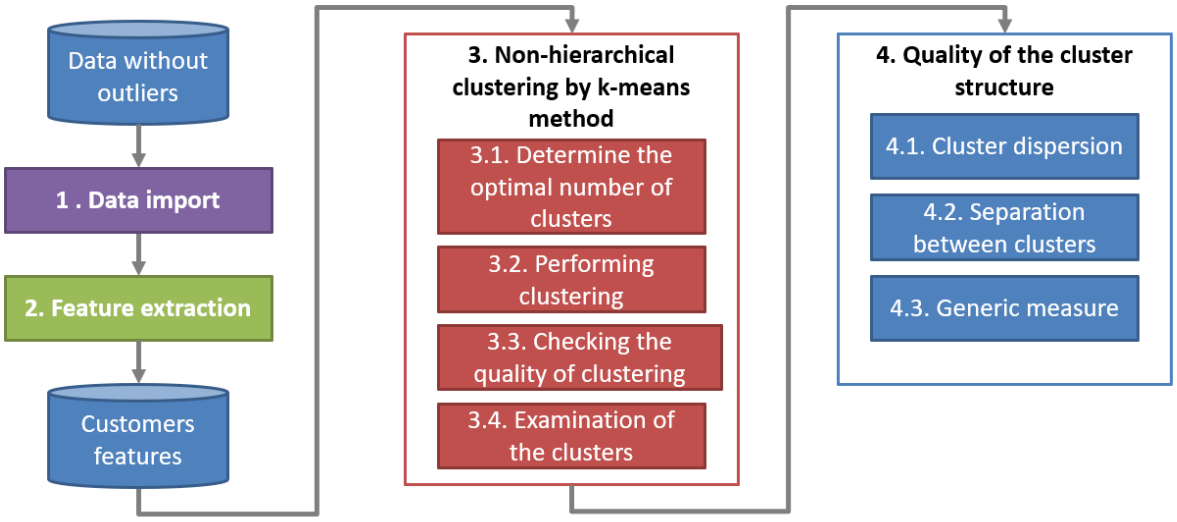
We have data on the values of bank account balances of 5869 customers of a certain bank. These are the values in thousands of USD recorded at the end of the month. The data are for 4 years (48 months).



The account balance data of a single customer is a time series (each customer is one time series).
The task is to find groups made up of similar customers (similar in terms of their account balance in consecutive months), so we are looking for groups of similar time series.

Customer clustering (dividing customers into groups) is important for various industries and institutions. It is used, among others, in banking, where customer clustering improves, for example marketing campaigns or risk management.

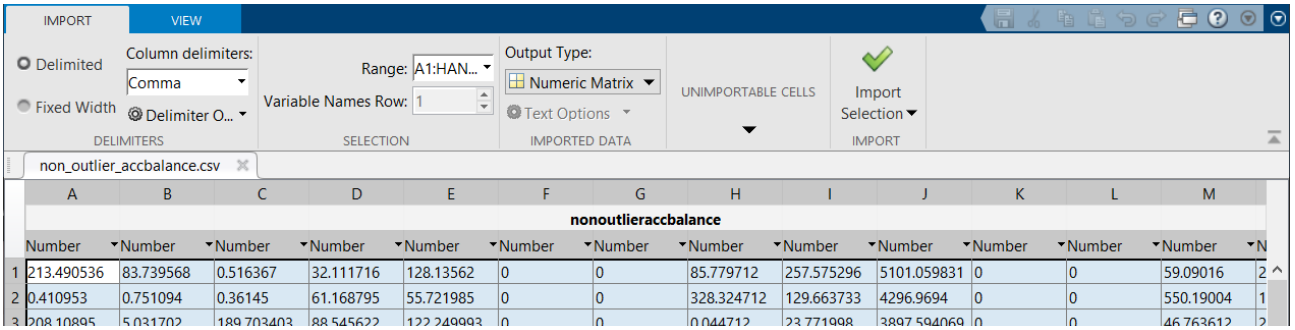
Course of the task



1 Data import

Load the file **non_outlier_accbalance** using the import wizard: Home ⍰ Import Data.

In the "Range" field enter A1:HAN48 (data range), in the "Output Type" field select "Numeric Matrix". Push the "Import Selection" button, select "Import Data".



Check the size of the loaded matrix.

```
[rows, cols] = size(nonoutlieraccbalance)
```

2 Feature extraction

For each customer (ignoring outliers), generate the following set of features based on account balance values:

1. Minimum balance during the period under study (48 months),
2. Maximum balance in the period under study,
3. Average balance in the period under study,
4. Standard deviation of the balance in the period under study,
5. Change in balance during the first year under study (difference in balance in the 12th and 1st month),
6. Change in balance during the last year studied (difference in balance in the 48th and 37th month),

7. The second maximum, i.e. the second highest value of the balance in the period under study,
8. The second minimum, i.e. the second lowest value of the balance in the period under study,
9. Median of January balances,
10. The sum of balances from all summer months in the period under study (June, July, August),
11. Number of months with zero account balance,
12. Coefficients of the first principal component,
13. Coefficients of the second principal component,
14. Median change in account balances (balance change is the absolute value from the difference between the next month's balance and the previous month's balance),
15. The variance of the balance in each of the twelve months of the year (first calculate the variance of the balances in each of the twelve months, and then calculate the variance from the calculated twelve-monthly variances),
16. Stability of the balance in each of the twelve months of the year (first calculate the average balance in each of the twelve months, and then calculate the variance from the calculated twelve-monthly averages).

```
mins = min(nonoutlieraccbalance)
maxs = max(nonoutlieraccbalance)
means = mean(nonoutlieraccbalance)
stds = std(nonoutlieraccbalance)
change_1st_y = zeros(1,cols);
change_last_y = zeros(1,cols);
second_mins = zeros(1,cols);
second_maxs = zeros(1,cols);
January = zeros(1,cols);
summer = zeros(1,cols);
num_of_zeros = sum(nonoutlieraccbalance==0);
[coef, score, latent] = pca(nonoutlieraccbalance');
pc1 = score(:,1)';
pc2 = score(:,2)';
median_change = zeros(1,cols);
variance = zeros(1,cols);
stability = zeros(1,cols);
for i=1:cols
    change_1st_y(i) = nonoutlieraccbalance(12,i)- nonoutlieraccbalance(1,i);
    change_last_y(i) = nonoutlieraccbalance(48,i)- nonoutlieraccbalance(37,i);
    sorted_values = sort(nonoutlieraccbalance(:,i),"descend");
    second_maxs(i) = sorted_values(2);
    second_mins(i) = sorted_values(end-1);
    January(i) = median(nonoutlieraccbalance([1 13 25 37],i));
    summer(i) = sum(nonoutlieraccbalance([6 7 8 18 19 20 30 31 32 42 43 44], i));
    median_change(i) = median(abs(diff(nonoutlieraccbalance(:, i))));
    vars = zeros(1,12);
    averages = zeros(1,12);
    for j=0:11
        vars(j+1) = var(nonoutlieraccbalance((1+4*j):(4+4*j), i));
```

```
        averages(j+1) = mean(nonoutlieraccbalance((1+4*j):(4+4*j), i));
    end
    variance(i) = var(vars);
    stability(i) = var(averages);
end
```

Place all the determined features in a single matrix, in which the features will be in columns.

```
features = [mins; maxs; means; stds; change_1st_y; change_last_y; second_maxs;...
    second_mins; January; summer; num_of_zeros; pc1; pc2; median_change;...
    variance; stability]'
```

Normalize each feature separately. Formula: $x^* = \frac{x - \min(x)}{\max(x) - \min(x)}$, where x^* is the data after normalization, and x is the data before normalization.

```
for i=1:16
    features(:,i)=(features(:,i)-min(features(:,i)))/ ...
        (max(features(:,i))-min(features(:,i)));
end
features
```

3 Non-hierarchical clustering by k-means method

3.1 Determine the optimal number of clusters

Determine the optimal number of clusters for k-means clustering. Consider the number of clusters from 2 to 8. Use the following indexes: Calinski-Harabasz, Davies-Bouldin, silhouette index.

```
eval_CH = evalclusters(features, 'kmeans', 'CalinskiHarabasz','KList', 1:8)
% the higher the index value, the better the cluster structure
eval_CH.CriterionValues
eval_DB = evalclusters(features, "kmeans", "DaviesBouldin", "KList", 1:8)
% the lower the index value, the better the cluster structure
eval_Sil = evalclusters(features, "kmeans", "silhouette", "KList", 1:8)
% the higher the index value, the better the cluster structure
```

3.2 Performing clustering

Perform clustering with the k-means method, using the square of the Euclidean distance. Determine three cluster structures: k=2, k=3, k=4. To avoid local minima, use a 30-fold replication of the k-means method.

```
% for k=2
[idx2,C2,sumdist2] = ...
    kmeans(features,2,'Distance','sqeuclidean','Display','final','Replicates',30);
% for k=3
[idx3,C3,sumdist3] = ...
    kmeans(features,3,'Distance','sqeuclidean','Display','final','Replicates',30);
% for k=4
[idx4,C4,sumdist4] = ...
    kmeans(features,4,'Distance','sqeuclidean','Display','final','Replicates',30);
```

3.3 Checking the quality of clustering

Check the quality of the three created cluster structures using the silhouette graph and the average silhouette measure.

```
[silh2,h] = silhouette(features,idx2,'sqeuclidean'); % for k=2
xlabel('Silhouette')
ylabel('Cluster')
mean(silh2)
[silh3,h] = silhouette(features,idx3,'sqeuclidean'); % for k=3
xlabel('Silhouette')
ylabel('Cluster')
mean(silh3)
[silh4,h] = silhouette(features,idx4,'sqeuclidean'); % for k=4
xlabel('Silhouette')
ylabel('Cluster')
mean(silh4)
```

3.4 Examination of the clusters

Assume that customers will be divided into two clusters. Check how many customers belong to each of the defined clusters.

```
clusters_NH = idx2;
quantities = [sum(clusters_NH==1) sum(clusters_NH==2)]
```

Calculate the average balance of customers in each cluster separately. When calculating the average, assume that the account balances from all 48 months will be added together.

```
mean_accbalance=[mean(sum(nonoutlieraccbalance(:,clusters_NH==1))), ...
    mean(sum(nonoutlieraccbalance(:,clusters_NH==2))) ]
```

Calculate the standard deviation of the total customer account balance in each cluster separately.

```
std_accbalance=[std(sum(nonoutlieraccbalance(:,clusters_NH==1))), ...
    std(sum(nonoutlieraccbalance(:,clusters_NH==2)))]
```

Draw a graph comparing the determined means and standard deviations in each cluster.

```
bar([mean_accbalance ; std_accbalance]')
legend('mean','std')
```

Present the time series totals for each cluster on a single graph. The calculated totals should include the account balances of all customers belonging to a given cluster.

```
plot(sum(nonoutlieraccbalance(:,clusters_NH==1)'))
hold on
plot(sum(nonoutlieraccbalance(:,clusters_NH==2)'))
hold off
legend('cluster1','cluster2')
```

For comparison, show a graph of the total balance of all customers.

```
plot(sum(nonoutlieraccbalance'))
```

Determine the centroid of each cluster. Assume that the centroid will be the so-called "averaged object", i.e. a vector whose values are equal to the average value of all objects (customers) in a given cluster. Note - this is not the average balance, but the average value of normalized data.

```
centroids(1,:)=mean(features(clusters_NH==1,:));
centroids(2,:)=mean(features(clusters_NH==2,:));
```

Depict all centroids on one common graph.

```
plot(centroids(1,:))
hold on
plot(centroids(2,:))
hold off
legend('c1','c2')
```

Create a separate chart for each defined cluster. On each chart, place the normalized balances of all customers belonging to that cluster and the centroid determined for that cluster. Mark the centroid with a bold red line.

```
plot(1:16,features(clusters_NH==1,:),'Color',[.8 .8 .8],'LineWidth',0.1)
hold on
plot(centroids(1,:),'LineWidth',5,'Color','r')
hold off
plot(1:16, features (clusters_NH==2,:),'Color',[.8 .8 .8],'LineWidth',0.1)
hold on
plot(centroids(2,:),'LineWidth',5,'Color','r')

hold off
```

4 Quality of the cluster structure

4.1 Cluster dispersion

Examine the dispersion of customers within each cluster. Use a measure of the average dispersion of a cluster k by considering the distances between the x and y vectors included in a given cluster k, as expressed by the formula:

$$\sigma_1(k) = \frac{1}{m} \sum_{\substack{\boldsymbol{x}\in k \\ \boldsymbol{y}\in k}} d^2(\boldsymbol{x},\boldsymbol{y})$$

where:

- $m = \frac{n_k(n_k-1)}{2}$
- n_k is the number of vectors in the cluster *k,*
- *d* is Euclidean distance.

```
sigma1=zeros(1,2);
for i=1:2
```

```
    sigma1(i)=...
        1/(quantities(i)*(quantities(i)-1)/2)*...
        sum(pdist(features(clusters_NH==i,:),"squaredeuclidean"));
end
sigma1
```

4.2 Separation between clusters

Examine the separation between each pair of clusters. Use the measure of distance between cluster centroids. For clusters k_i i k_j, the measure so defined is expressed by the formula:

$$s_2(k_i, k_j) = d^2(\boldsymbol{c}_{k_i}, \boldsymbol{c}_{k_j}).$$

```
s2=zeros(2,2);
for i=1:2
    for j=1:2
        s2(i,j)=sum((centroids(i,:)-centroids(j,:)).^2);
    end
end
s2
```

4.3 Generic measure

Determine a generic measure of the quality of the cluster structure. Use the measure of inter-cluster separation across the entire data space for measures s_2 and σ_1:

$$generic\_measure = \sum_{i,j=1\, j\neq i}^{K} \frac{s_2(k_i k_j)}{\sigma_1(k_i)}$$

```
generic_measure=0;
for i=1:2
    for j=1:2
        generic_measure= generic_measure+s2(i,j)/sigma1(i);
    end
end
generic_measure
```

Student assignments

1. Determine the optimal number of clusters for clustering using the hierarchical (linkage) method. Use the evalclusters function. Consider the number of clusters from 2 to 8. Use the following three indexes: Calinski-Harabasz, Davies-Bouldin, silhouette index.
2. Perform hierarchical clustering. Calculate the distances between each pair of clients using Euclidean distances. Create a dendrogram using Ward's method. Divide the clients into clusters at the location of the longest branches of the dendrogram. Display information about the resulting clusters, analogously to Section 3.4.
3. Examine the quality of the structure of the clusters defined by the hierarchical method. Use a measure of cluster dispersion, inter-cluster separation and a generic measure of quality.

4. Suggest additional characteristics that can be determined from the time series describing the account balance.