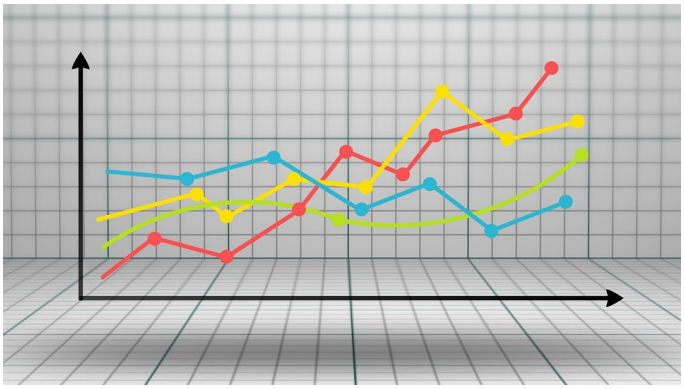

University of Zurich

Basic's Module

Multivariate Time Series

Isabel Margolis

supervised by
Sven Helmer

June 12, 2019

Introduction

The goal of this paper is to introduce two existing methods of clustering and comparing multivariate time series and to discuss the use of those methods on a specific movie database, in hope that it will become useful in the prospective Master Project.

Multivariate time series consist of multiple variables that vary over time. Each time step can be represented by a vector containing information for each variable. Because of their natural temporal order, multivariate time series can be useful if one wants to explain the interactions and co-movements among a group of variables over time [Singh, 2018].

Clustering Multivariate Time Series Data

Singhal and Seborg present a new methodology for clustering multivariate time series data [Singhal and Seborg, 2005]. This methodology uses a combination of similarity factors and a modified K-means clustering algorithm.

PCA Similarity Factor: The first similarity factor is based on PCA (Principal Component Analysis). The goal of PCA is to reduce the dimensionality of a dataset by extracting the principle components and projecting the dataset onto the hyperplane defined by the first k principal components. The first principal component is the axis that accounts for the largest amount of variance in the data. It is a linear combination of the features. In Figure 1 (A) it is the diagonal line labeled as "PC1" [Reif, 2018]. Figure 1 (C) shows the transformed dataset if one would project the original dataset to the 1-dimensional hyperplane.

The subsequent principle components are the axes that account for the largest amount of remaining variance and are orthogonal to all previous axes. In Figure 1 (A) it is the orthogonal line labeled as "PC2". Figure 1 (D) shows the transformed dataset on a 1-dimensional hyperplane defined by the second principle component. The variability is clearly smaller compared to the first principle component, because the length of the line in Figure 1 (C) is 150 and the length of the line in Figure 1 (D) is 40. Figure 1 (B) shows the transformed data to its two principal components. The center of the original data is projected to the center of the new coordinate system defined by the principle components [Gron, 2017].

The PCA similarity factor contains the difference between two multivariate datasets, X_1 and X_2, reduced to their k largest principal components. Each dataset contains columns for the number of variables and rows for the number of observations at a different moment in time. The similarity factor is calculated using the following formula

$$S^{\lambda}_{PCA} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{k} (\lambda_i^{(1)} \lambda_j^{(2)}) \cos^2 \theta_{ij}}{\sum_{i=1}^{k} \lambda_i^{(1)} \lambda_i^{(2)}}$$

where θ_{ij} is the angle between the ith and jth principal components of the first and second datasets, respectively. $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$ are the ith eigenvalues of each dataset and serve as weights based on their contributing variability in the data.

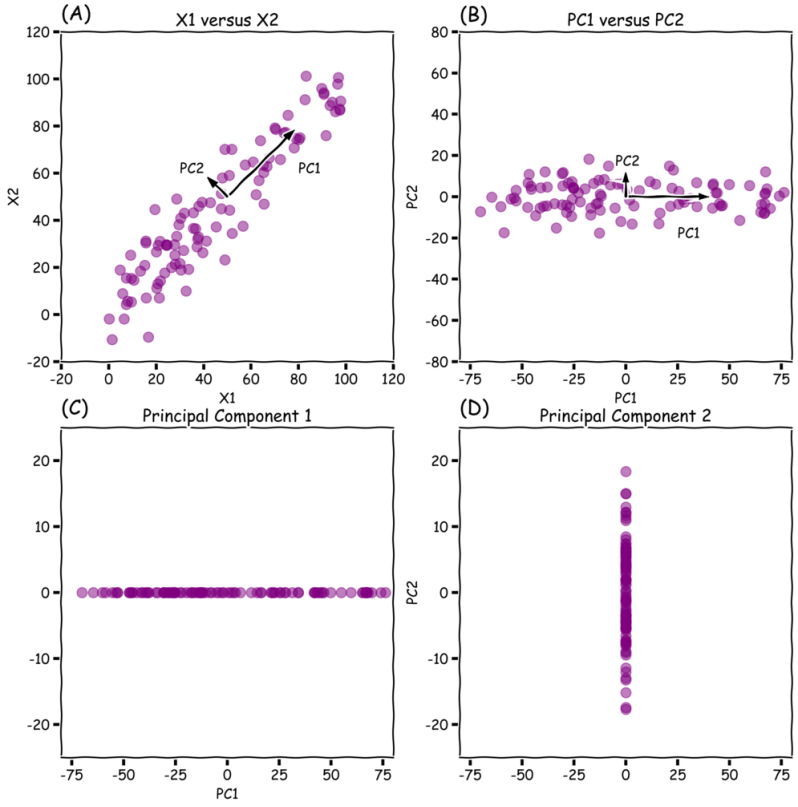

Figure 1: (A) Original data set showing X1 and X2, (B) Data transformed to its principal components 1 and 2, (C) Figure presenting only the projection of principal component 1, and (D) Figure presenting only the projection of principal component 2 (Source: [Reif, 2018])

Distance Similarity Factor: The second similarity factor is based on the Mahalanobis distance, which is defined as

$$\Phi = \sqrt{(\bar{x}_2 - \bar{x}_1)\Sigma_1^{*-1}(\bar{x}_2 - \bar{x}_1)^T}$$

where $\bar{x}_1$ and $\bar{x}_2$ are the centroids of the two datasets, X_1 and X_2. A centroid is a vector that contains the mean over all observations in the dataset for each variable. Σ_1^{*-1} is the pseudo-inverse of the covariance matrix of the reference dataset X_1. In contrast to the Euclidean distance, the Mahalanobis distance takes the correlation between the variables into account. In Figure 2 the data is clearly correlated and the green point fits better into the cluster represented by the red circle than the red point [McCormick, 2014]. According to the Euclidean distance, both points are equally far away from the center of the cluster, thus both points are equally likely to belong to that cluster. The Mahalanobis distance, on the other hand, results in a smaller distance from the green point to the center than the red point. Hence, this distance is more meaningful for correlated data, which can often be the case for time series.

A value between 0 and 1 is preferred, because otherwise the distance can become infinitely large. Hence, a monotonic mapping from Φ to S_{dist} is desired, where $0 \leq S_{dist} \leq 1$ and $\Phi_i \geq \Phi_j \implies S_{dist,i} \leq S_{dist,j}$ holds. Singhal and Seborg used the Gaussian

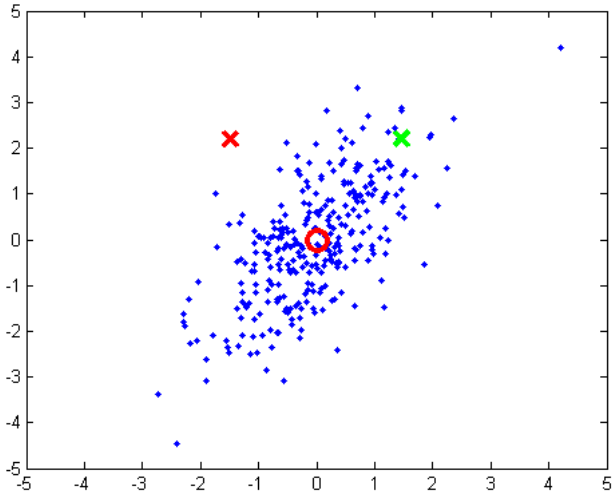

Figure 2: Data with correlation (Source: [McCormick, 2014])

probability function with mean 0 and variance 1 to map Φ to S_{dist}:

$$S_{dist} = 2 \times [1 - \frac{1}{2\pi} \int_{-\infty}^{\Phi} e^{-\frac{z^2}{2}} dz]$$

This similarity factor calculates the distance between the centers of each dataset and is especially significant if both datasets have very similar principal components but are located far apart from each other. The PCA similarity factor does not take the original position of the data into account, thus two datasets that have their center in completely different locations might have perfect similarity based on the PCA similarity factor.

Combination of Similarity Factors: To achieve a single measure for the similarity of two datasets, the weighted average of all the factors are taken:

$$SF = \alpha_1 S_{PCA}^{\lambda} + \alpha_2 S_{dist}$$

with $\alpha_1 + \alpha_2 = 1$. The value of the weights depend on the dataset.

K-Means Clustering Algorithm: K-means clustering is an unsupervised algorithm with the objective of partitioning a given dataset into a fixed number of clusters. Each cluster is represented by its centroid; the mean of all the data points in that cluster. The algorithm tries to keep the datasets in the same cluster as similar as possible by minimizing the squared Euclidean distance between the data points and the cluster's centroid. At the same time it tries to keep the clusters as different from each other as possible [Dabbura, 2018].

In the case of multivariate time series, Singhal and Seborg propose a modification of the K-means clustering algorithm, where the similarity factor introduced above replaces the Euclidean distance.
Given Q datasets, $\{X_1, ..., X_q, ..., X_Q\}$, and K clusters, each dataset is initialized to exactly one cluster. Each cluster is represented by an aggregate dataset χ_i $(i = 1, 2, ...K)$, where all datasets belonging to the corresponding cluster are concatenated.

Iterating through the datasets, the dissimilarity

$$d_{i,q} = 1 - SF_{i,q}$$

of each dataset to each cluster is calculated using the similarity factor SF introduced above. Based on that factor, each dataset is assigned to the cluster it is least dissimilar to.

After each dataset is assigned to a cluster, the average dissimilarity $J(K)$ of each dataset from its cluster is calculated:

$$J(K) = \frac{1}{Q} \sum_{i=1}^{K} \sum_{X_q \in \chi_i} d_{i,q}$$

If the average dissimilarity has changed, thus has not converged yet, the procedure is repeated until the algorithm converges.

Selection of the Number of Clusters: Singhal and Seborg propose a new method on finding the optimum number of clusters K. The K-means clustering algorithm introduced above is evaluated using different values for K and the average dissimilarity $J(K)$ is calculated, which decreases with increasing number of clusters. But after a certain amount, the algorithm starts to over-fit. The point where $J(K)$ changes significantly results in the optimal number K. This point can be found by calculating the percentage change in $J(K)$:

$$dJ(K) = \frac{|J(K+1) - J(K)|}{J(K)} \times 100\%$$

As soon as $dJ(K+1)$ is larger than $dJ(K)$ for a K, there is a significant change and the corresponding K is the chosen optimal number of clusters.

Discussion: Singhal and Seborg have shown that their method yields a higher percentage of correctly classified samples compared to earlier methods. In contrast to the Euclidean distance and other similarity measures, the similarity factor proposed here has the advantage that it is not necessary for all datasets to have the same number of observations. This is very important for time series data, because the number of observations often varies. We will see that in the movie dataset this is indeed the case.
However, one drawback is that the K-means clustering algorithm is not guaranteed to converge and depends on the initialization. It makes sense to repeat the algorithm multiple times with different random initial guesses, but this could become computationally expensive. Another possible drawback is that all datasets must have the same number of variables, because both the PCA similarity factor and the distance similarity factor would result in an error otherwise.

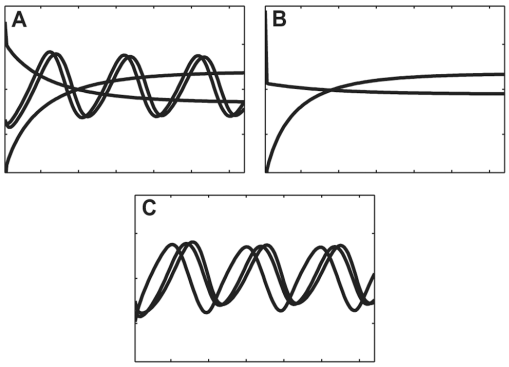

Figure 3: Three dynamic models with different dimensionality. (A) Model with 4 variables, (B) Model with 2 variables and (C) Model with 3 variables (Source: [Tapinos and Mendes, 2013])

Semi Metric Ensemble Time Series (SMETS)

SMETS is a new method to compare multivariate time series of arbitrary dimensions proposed by Tapinos and Mendes [Tapinos and Mendes, 2013]. It is a semi-metric, meaning that the triangle inequality does not necessarily hold. Two datasets have a different dimension if the number of variables or the number of observations vary. This method allows both datasets to have a different number of variables, but not necessarily a different number of observations; this depends on the distance metric that is used. Figure 3 shows an example of three models each with a different number of variables [Tapinos and Mendes, 2013]. This method makes it possible to decide whether model A is more similar to model B or to model C.

From now on, let there be two time series and let n denote the dimension of the smaller time series and m the dimension of the larger time series.

Partial Matching: The first step of SMETS consists of matching the univariate time series of two models with possibly different dimensionality. First, the distances from each univariate time series from the first model to each univariate time series of the second model are calculated. The two univariate time series, one from each model, with the smallest distance out of all, are chosen. Here, the Euclidean distance is used, which restricts the time series to have the same number of observations. The distance is recorded in a vector d and the matched univariate time series are removed from their models. Continuing with this procedure, the next two univariate time series with the smallest distance are chosen, the distance recorded, and the time series removed from the models. This is done until each univariate time series from the smaller model is matched with a corresponding univariate time series from the larger model.

The overall distance of both models is calculated using a p-norm of d where $p = n$, the dimension of the smaller time series:

$$\|d\|_n = \sqrt[n]{\sum_{i=1}^{n} (d_i)^n}$$

It is not clear why p=n was chosen, but probably because extreme values have a larger effect on the overall distance for p=n than for p=2, which would be equal to the Euclidean distance. Since the dimensionality of both time series are assumed to be different, there will exist some univariate time series from the larger model that were not matched. Those time series will contribute to the penalization.

Penalization: In the second step, two penalties are added. The first penalty, EP, depends on the importance of the univariate time series of the larger model that were not matched. Because of the assumption that the larger model has dimensionality m and the smaller model has dimensionality n, there are $m - n$ univariate time series from the larger model without a match. The penalty is defined as:

$$EP = \sum_{j=1}^{m-n} \frac{min(d_j)'(H_j)}{\sum_{i=1}^{m} H_i}$$

where d_j is the smallest distance between the j-th unmatched univariate time series in the larger model from any time series from the smaller model. H_j is the Shannon entropy defined as:

$$H_j = -\sum_{i=1}^{q} p(t_{j,i}) log_2 p(t_{j,i})$$

where $p(t_{j,i})$ is the frequency of the i-th data point of the j-th unmatched univariate time series t_j and q is the length of t_j. If the time series is constant, then it adds zero information and the first penalty is thus zero.

The second penalty, P, accounts for the difference in dimensionality of the two models:

$$P = \frac{m - n}{m + n}$$

where m and n are the dimensions of the two models.

SMETS: By adding the difference and both penalties, and adjusting the sum with a 2-norm in order to make the second penalty weaker, the following formula is arrived at:

$$SMETS = \sqrt{(\|d\|_n + EP)^2 + P^2}$$

Discussion: Tapinos and Mendes show in several experiments that SMETS results in more accurate distance values for time series with different dimensionality compared to the weighted average method. SMETS handles large differences in dimensionality well and detects similar behavior between two time series. However, this method seems to be restricted to cases where each variable can be compared to any other variable.

	trailer	frame	1	2	3	4	5	6	7	8	...	173	174	175	176	177	178	179	180	181	182
0	102	26	-202	56	12	54	-7	14	22	29	...	16	16	16	16	16	16	16	16	16	16
1	102	27	-208	63	20	55	0	14	22	28	...	16	16	16	16	16	16	16	16	16	16
2	102	28	-216	71	28	55	8	14	22	28	...	16	16	16	16	16	16	16	16	16	16
3	102	29	-214	71	30	57	8	14	22	26	...	16	16	16	16	16	16	16	16	16	16
4	102	30	-215	71	29	56	8	14	22	27	...	16	16	16	16	16	16	16	16	16	16
5	102	587	-194	64	36	66	1	14	20	15	...	16	16	16	16	16	16	16	16	16	16

Figure 4: Sample of trailer 102 from the movie database with the frame column serving as the time stamp. Each row represents the state of the trailer at a specific point in time (Source: [Yashar Deldjoo and Piazzolla, 2016])

Movie Database

The database contains movie trailers, and the goal of the Master Project is to extensively explore the data using PCA among other techniques and to create an artificial neural network that assigns the genre to the movie based on the trailer clip [Yashar Deldjoo and Piazzolla, 2016]. Thus, each genre can be regarded as a separate cluster and for a given trailer we must find the cluster which is most similar to the trailer. Each trailer is described by a number of key-frames, varying between 1 and 6000 per trailer, each consisting of 182 descriptor variables and some additional aggregate variables. Figure 4 shows a small sample of the data. Each row represents the state of the trailer at a specific time. Thus, each trailer can be represented as a time series with the "frame" column serving as the time stamp.

The crossparsing algorithm proposed by Giulia Burgio in her Bachelor's Thesis compares two sequences based on how often we must split the first sequence into subsequences to find a match in the second sequence [Burgio, 2018]. The sequences consist of boolean vectors, one vector per feature, with the length of the vector equal to the number of frames in the trailer. And the boolean entries represent whether or not the integer entry for the specific frame and feature is greater or smaller then the median of that feature over all frames and trailers. It is necessary to convert the integer values to boolean values for the algorithm to work properly.

Those new similarity measures proposed above are not constrained to converting the integer values to boolean values. This might be an advantage compared to the crossparsing algorithm.

The SMETS algorithm does not differentiate between the variables in its comparison. Each variable of the first time series is matched to the most similar variable in the second time series. This method might not work well, because each of the 182 features represents a specific attribute of a frame in the trailer and SMETS would allow for one feature to be matched with another, which in reality might not make sense.

The k-means clustering algorithm seems more promising. Especially the use of PCA might work well. The principle components are linear combinations of the features that go through the most variability throughout the frames. If the angle between the principle components of two trailers is small, this means that in both trailers the same set of

features changes a lot and this could indicate a similarity in the genre.
An advantage of this proposed similarity factor is that not all datasets must have the same number of observations. Because not all trailers are of the same length, the number of frames per trailer can vary, thus the number of observations varies.
The number of clusters would be the number of given genres. We initialize each trailer by assigning it to a cluster. There are several details worth noting. First, it is not clear from the proposed algorithm how to handle special cases when a cluster remains empty after the initialization or becomes empty during the process. In this case, the aggregated dataset is empty and no other dataset can be assigned to that cluster, so that genre will remain empty, which might not be ideal. This case occurred with the sample data of 20 trailers. The probability of this happening with considerably more data, however, is very small.
Second, the data must be scaled to unit variance and zero mean before applying the similarity functions. Both the PCA similarity factor and the distance similarity factor require standardized data to work properly.
Third, since the algorithm depends a lot on the initialization, it is necessary to run the algorithm multiple times to get a stable result. It is possible that the algorithm does not converge or that, depending on the initialization, we get different end results.
Lastly, the temporal order of the data is not considered in this method. The frames in a trailer could be shuffled randomly and we would still get the same result after running the k-means clustering algorithm.

In conclusion, the SMETS algorithm is most likely not a good approach for the specific case of clustering trailers based on their genre, because we do not want to compare different features with each other. The k-means clustering algorithm, however, might be applicable, possibly with some modifications. This approach is a possibility to be further pursued in the Master Project. Analyzing the performance of this algorithm can help understand which aspects in a trailer account for the genre, which could be useful in building a classification model that assigns the genre to a new trailer.

References

[Burgio, 2018] Burgio, G. (2018). Video similarity analysis based on mpeg-7 visual features. Free University of Bolzano.

[Dabbura, 2018] Dabbura, I. (2018). K-means clustering: Algorithm, applications, evaluation methods, and drawbacks. https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

[Gron, 2017] Gron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 1st edition.

[McCormick, 2014] McCormick, C. (2014). Mahalanobis distance. http://mccormickml.com/2014/07/22/mahalanobis-distance/.

[Reif, 2018] Reif, R. (2018). Limitations of applying dimensionality reduction using pca. https://www.robertoreif.com/blog/2018/1/9/pca.

[Singh, 2018] Singh, A. (2018). A multivariate time series guide to forecasting and modeling. https://www.analyticsvidhya.com/blog/2018/09/multivariate-time-series-guide-forecasting-modeling-python-codes/.

[Singhal and Seborg, 2005] Singhal, A. and Seborg, D. E. (2005). Clustering multivariate timeseries data.

[Tapinos and Mendes, 2013] Tapinos, A. and Mendes, P. (2013). A method for comparing multivariate time series with different dimensions. In *PloS one*.

[Yashar Deldjoo and Piazzolla, 2016] Yashar Deldjoo, Mehdi Elahi, P. C. F. G. and Piazzolla, P. (2016). Recommending movies based on mise-en-scene design. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*, pages 1540–1547, New York, NY, USA. ACM. https://doi.org/10.1145/2851581.2892551.