Journal of Machine Learning Research 23 (2022) 1-7 Submitted 1/21; Revised 5/22; Published 9/22

Feature selection for time series

Author One ONE@STAT.WASHINGTON.EDU
Department of Statistics
University of Washington
Seattle, WA 98195-4322, USA

Author Two TWO@CS.BERKELEY.EDU
Division of Computer Science
University of California
Berkeley, CA 94720-1776, USA

Editor: My editor

Abstract

Feature selection is a crucial task when working with time series data as it helps to identify the most relevant features that can improve the performance of machine learning models. In recent years, with the rapid advance in technology, the amount of time series data has exploded, making feature selection even more important. However, feature selection for time series data is different from that of non-time dependent data due to the temporal nature of such data.

In this paper, we propose two novel methods for feature selection in time series data that utilize local features to identify the most dominant series. Specifically, we construct a network that represents the similarity between local features over time. We then apply the PageRank algorithm to this network to extract the most important features. Our experiments demonstrate that the proposed models outperform existing feature selection methods for time series data, highlighting their effectiveness in improving the performance of time series models.

Keywords: time series,temporal feature selection

1 Introduction

Multidimensional time series data is characterized by the presence of multiple variables or dimensions that change over time. These types of data are commonly encountered in a range of fields, from finance and economics to healthcare and engineering. When analyzing multidimensional time series data, it is often useful to identify the dominant series or variables that are driving the overall behavior of the system. By doing so, we can gain insights into the underlying dynamics of the system and potentially use this information for feature selection or other downstream analysis. The goal of feature selection is to identify the most relevant features for a particular machine learning task, while removing irrelevant or redundant ones. This can help in reducing the dimensionality of the data, improve model accuracy and generalization, and reduce overfitting. For time series data, feature selection can be different from traditional feature selection methods because the data is structured in a time-dependent manner. The main difference is that features are often selected based on their ability to predict future values of the time series, rather than just

their correlation with the target variable.Furthermore, unlike traditional feature selection where all data points are independent, in time series data, each observation is dependent on past observations. Therefore, the choice of relevant features becomes more critical as the impact of each feature on the prediction changes over time. Hence, feature selection for time series data requires specialized techniques such as lag features, rolling statistics, and time-based transformations. In this study, we propose two novel feature selection methods for time series analysis. In the following sections, we provide a detailed explanation of both methods and compare their performance with other baseline methods in section 3.

2 Existing approaches

2.1 PCA

Another noteworthy method is Principal Component Analysis (PCA), which involves eigenanalysis of the covariance matrix of the data. PCA seeks to reduce dimensionality by identifying a small number of principal components, which are linear combinations of the original variables. While PCA is commonly used for dimensionality reduction in static and independent multivariate data, it is not well-suited for dynamic multivariate time series data due to its static nature. Consequently, PCA cannot capture the dynamic relationships between variables in a time series. Despite this limitation, PCA remains a powerful tool, and for comparative purposes, the results are analyzed alongside those obtained from our method.

2.2 Autoregressive Temporal Regularizer in TRMF

The objective function discussed in **?** is given by

$$\min_{F,X,\Theta} \sum_{(i,t)\in\Omega} \left(Y_{it} - \mathbf{f}_i^\top \mathbf{x}_t\right)^2 + \lambda_f R_f(F) + \lambda_x \mathcal{T}_{\mathcal{M}}(\mathbf{X} \mid \Theta) + \lambda_\theta R_\theta(\Theta), \tag{1}$$

where Ω denotes the set of observed entries, and $R_f(F)$ and $R_x(X)$ are regularizers for F and X(Time- dependent variables), respectively. An autoregressive (AR) model is integrated into this framework to account for temporal dependencies. This objective function can be solved using an alternating minimization procedure over F, X, and Θ.

In this context, $\mathbf{X}$ represents a reduced-dimensional version of $\mathbf{Y}$ that retains the time component, thus facilitating the incorporation of temporal dependencies into the Temporally Regularized Matrix Factorization (TRMF) framework.

We compare this reduced matrix, which captures temporal features, with our feature selection model. This comparison highlights how effectively each model captures and utilizes temporal patterns in the data.

3 Methodology

In time series analysis, we often encounter data where patterns can change over time. In order to identify dominant patterns that persist over longer periods, it is necessary to observe how similar these patterns are across different time periods. One way to achieve this is through the comparison of the patterns of the same short period of time in all

series. For example, let's say we have three time series data sets A, B, and C, with each containing observations at different time points. To determine which of these series has the most dominant pattern, we can take a sliding window of a fixed length (e.g., one week), and compute the cosine similarity between the patterns of A, B, and C during that window. This allows us to compare the similarity of the patterns of each series at different time periods. By using cosine similarity, we can analyze time series data to identify dominant patterns that persist over longer periods of time. The size of the moving window is an important hyperparameter that must be chosen carefully based on the characteristics of the time series data. The choice of window size depends on various factors, such as the frequency of the data, the length of the time series, the complexity of the patterns, and the objectives of the analysis. A larger window size can provide a broader view of the data, allowing us to capture longer-term patterns, but it may not be sensitive enough to detect changes in short-term patterns. Conversely, a smaller window size can detect short-term changes more accurately, but it may not capture longer-term patterns with enough granularity. Choosing an appropriate window size requires balancing these considerations while taking into account the specific goals of the analysis. In some cases, it may be necessary to experiment with different window sizes to find the one that best captures the patterns of interest in the data.

3.1 second-order similarities

After dividing the main time series $X \in \mathbb{R}^{n \times w}$ into smaller time series, we obtain a set containing

$$\tilde{X}_i = [x_{(i-1)w+1}, \ldots, x_{(i-1)w+w}] \in \mathbb{R}^{n \times w}, \quad i = 1, \ldots, k = \lfloor \frac{T}{w} \rfloor$$

where w is the size of the window.

We can compare the similarity of the patterns between each pair of time series using cosine similarity. This results in a matrix of pairwise similarities for each time period. If we stack these matrices in order of their time periods, we obtain a 3D tensor $S \in \mathbb{R}^{n \times n \times K}$ where n is the number of features in the time series and K is the number of smaller time series, so:

$$S_{:,:,i} = |\tilde{X}_i \tilde{X}_i^\top|, \quad i = 1, \ldots, k$$

Based on the observation that each mode 2 fiber (i.e., $S_{i,j,:}$) in the similarity tensor reflects changes in the similarity of the same pair over time, the next step could be to analyze these fibers to gain insights into how the similarity between pairs evolves over time.

To gain insights into how the relationships between pairs of objects evolve over time, we analyzed the similarity changes of fibers in our similarity tensor. Specifically, we used cosine similarity to calculate the similarity between each pair of fibers. We examined the evolution of relationships between pairs of objects over time by analyzing the changes in the similarity of fibers within our similarity tensor.

To keep track of which similarity value corresponds to which pair in the final similarity matrix, the tensor was unfolded along its third mode (fibers). Specifically, the fibers were arranged under each other in a specific order, with $S_{1,1,:}, S_{2,1,:}, \ldots, S_{n,1,:}$ for the first set of fibers, $S_{1,2,:}, S_{2,2,:}, \ldots, S_{n,2,:}$ for the second set of fibers, and so on until the last set of

fibers was listed as $S_{1,n,:}, S_{2,n,:}, \ldots, S_{n,n,:}$. The resulting matrix $H \in \mathbb{R}^{n^2 \times n^2}$ is the cosine similarity matrix of unfold(S), so:

$$H_{n(i-1)+j,n(p-1)+q} = \frac{S_{i,j,:} S_{p,q,:}^{\top}}{\|S_{i,j,:}\| \cdot \|S_{p,q,:}\|}$$

This matrix can be considered as an adjacency matrix of a graph, where each node corresponds to a fiber and edges represent the similarity between pairs of fibers. By treating the similarity matrix as a graph, we can use various graph theory-based techniques to analyze it further. After constructing the similarity graph, one approach we can use to further analyze and understand the data is static rank. To compute the static rank of each node, one common method is to use the PageRank algorithm. PageRank computes a score for each node based on the number and quality of incoming edges, where higher scores indicate greater importance. By identifying which features were involved in the static rank and relate them back to the original data, we can use this formula to calculate the total importance of each node:

$$\text{score}(i) = \sum_{j=1}^{n} r_{n(i-1)+j}$$

where r is the PageRank vector of matrix H. After computing the scores for each node in our similarity graph, we selected the features with the highest scores to use in our modeling. We call our method *Feature Selection Based on Order 2 Similarity* to emphasize its use of order 2 similarity measures to identify important features in high-dimensional datasets. Figure 1 provides a visual representation of the actions taken in each step of the process.

3.2 first-order similarities

as mentioned earlier, When analyzing time series data, it is common to use sliding windows to capture subsets of the data over a given period. The length of the sliding window can vary depending on the analysis being performed and the characteristics of the data being analyzed. After we have created the similarity tensor ,S Rn×n×k with the frontal slices arranged in time order, the next step is to project each frontal slice onto an orthogonal space using SVD (singular value decomposition).suppose:

$$S_{:,:,i} = U_i \Sigma_i V_i \quad \forall i \in \{1, \ldots, k\}$$

Then $\mathcal{A}$, the denoised version of S is:

$$\mathcal{A}_{:,:,i} = U_i S_{:,:,i} \quad \forall i \in \{1, \ldots, k\}$$

It's important to note that the main goal of this projection step is not to reduce the dimensionality of the data, but rather to reduce the amount of noise in the data. By removing noise, we can more accurately identify and analyze the underlying patterns and features in the time-series data. By treating this tensor as an adjacency tensor of a dynamic graph, we can analyze the behavior of the features over time.moreover, It is worth mentioning that interpreting tensors as graphs allows us to leverage the mathematical properties of

graph theory to perform analysis and modeling tasks on our data. For example The Temporal PageRank algorithm is a powerful tool in graph theory that extends the traditional PageRank algorithm to dynamic networks. Using the temporal PageRank values to select important nodes as a feature selection technique has several advantages. First, it allows us to capture the time-varying behavior of the network and identify nodes that are consistently important over time. Second, it takes into account the importance of both direct and indirect connections between nodes in determining their overall influence within the network. Specifically, Polina Rozenshtein and Aristides Gionis use the concept of "temporal walk" in their paper on Temporal PageRank. A temporal walk is a path in a temporal network that follows the edges of the network in chronological order, with each step only moving to nodes that are reachable from the current node at the current time or later. This allows them to take into account the time dimension of the network when computing the transition probabilities between nodes. By analyzing such temporal walks, they develop a method for assigning time-dependent weights to the edges in the network, which is the basis for the Temporal PageRank algorithm.

In Figure2, the process of what we do in each step is illustrated. Finally, once we have selected important nodes based on their temporal PageRank values, we can use these nodes as features for further analysis or modeling tasks.

4 Experiments

To compare the performance of different feature selection algorithms, we will divide the time series data into successive subsets, with each subset containing 2000 records. Feature selection techniques will be applied to each subset, and then XGBoost models will be trained and evaluated on these processed subsets. Performance will be assessed in terms of root mean squared error (RMSE) and running time.

For a baseline comparison, we will also evaluate the performance of our XGBoost models against those derived from principal component analysis (PCA), temporally regularized matrix factorization (TRMF), and the feature selection methods we introduced.

In the first experiment, we select an appropriate number of features for each subset using the energy retained in Principal Component Analysis (PCA). For each subset, the number of components is determined based on the desired energy retention. Specifically, for the dataset X, if λ_i is the eigenvalue corresponding to the component v_i such that $X^\top X v_i = \lambda_i v_i$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, then the energy e_j is given by:

$$e_j = \frac{\sum_{i=1}^{j} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

This ratio tells us how much of the total variance in the data is retained in the reduced k-dimensional representation.

The number of selected components k is determined by:

$$k = \arg\min\{j | e_j \geq e\}$$

For example,in Appliances Energy Prediction for an energy retention of 89%, the first segment requires only 1 component, while the 20th segment requires 4 components. After

selecting the appropriate number of components using PCA, we then applied all feature selection and extraction methods on each segment to obtain that number of features and determine which method resulted in the least RMSE when using the XGBoost model.

For first-order and second-order similarity, the window length hyperparameter was selected from the set $w = \{2, 3, 5, 7, 10, 30\}$. The experiment was repeated for different window lengths, and the best result was reported. Additionally, the time reported for each segment is the average execution time of the method across all window lengths in the set w.

Appendix A. More Details about Experiments

A.1 Datasets:

- **Appliances Energy Prediction:** The data was recorded at 10-minute intervals over approximately 4.5 months, collected from a low-energy building to model appliance energy usage. This resulted in a multivariate time-series dataset with 19,735 instances and 28 features.[1]

- **gas sensor array temperature modulation:** This dataset contains 4,095,000 instances recorded over a 3-week period. It includes 20 features collected from a chemical detection platform composed of 14 temperature-modulated metal oxide (MOX) gas sensors. The data captures time-series measurements of CO concentration, humidity, temperature, flow rate, heater voltage, and sensor resistance values.[2]

- **Gas sensor array under dynamic gas mixtures:** This dataset consists of 4,178,504 instances recorded over approximately 12 hours from 16 chemical sensors exposed to two dynamic gas mixtures. It includes 19 features, capturing time-series measurements of gas concentrations and sensor responses. The data was collected to study the behavior of chemical sensors under varying concentrations of gases like Ethylene, Methane, and CO in air.[3]

Appendix B.

References

C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, IT-14(3):462–467, 1968.

1. https://github.com/LuisM78/Appliances-energy-prediction-data
2. https://archive.ics.uci.edu/dataset/487/gas+sensor+array+temperature+modulation
3. https://archive.ics.uci.edu/dataset/322/gas+sensor+array+under+dynamic+gas+mixtures