Pattern Analysis and Applications (2024) 27:39
https://doi.org/10.1007/s10044-024-01262-3

ORIGINAL ARTICLE

Focalize K-NN: an imputation algorithm for time series datasets

Ana Almeida[1,2] · Susana Brás[1,3] · Susana Sargento[1,2] · Filipe Cabral Pinto[4]

Received: 30 October 2023 / Accepted: 27 February 2024

Abstract
The effective use of time series data is crucial in business decision-making. Temporal data reveals temporal trends and patterns, enabling decision-makers to make informed decisions and prevent potential problems. However, missing values in time series data can interfere with the analysis and lead to inaccurate conclusions. Thus, our work proposes a Focalize K-NN method that leverages time series properties to perform missing data imputation. This approach shows the benefits of taking advantage of correlated features and temporal lags to improve the performance of the traditional K-NN imputer. A similar approach could be employed in other methods. We tested this approach with two datasets, various parameter and feature combinations, and observed that it is beneficial in scenarios with disjoint missing patterns. Our findings demonstrate the effectiveness of Focalize K-NN for imputing missing values in time series data. The more noticeable benefits of our methods occur when there is a high percentage of missing data. However, as the amount of missing data increases, so does the error.

Keywords Missing data imputation · Machine learning · K-nearest neighbors · Time series

1 Introduction

Missing data in time series can obscure patterns and trends, potentially affecting the analysis of the data and the conclusions related to trends and predictions. Inaccurate studies can have high costs and negative impacts on businesses and people's lives, leading to a lack of confidence in data and methods, and hindering decision-makers from utilizing valuable data insights to perform a coherent analysis.

Imputation algorithms emerged to solve the problem of missing data. An adequate imputation algorithm improves the data usability, allowing better data analysis, visualization, and exploration, helping to avoid bias in the analysis and models. Furthermore, in the context of missing data in time series, adequate imputation methods can improve the performance of time series-related algorithms, such as forecasting tasks, and maintain data continuity.

In smart cities, missing data is mainly caused by faults and failures in the infrastructure, with distinct causes such as humans, natural phenomena, and software or hardware problems. In this context, a fault can be described as a mistake, something wrong, an omission, a crash, or incorrect timing. A failure is described as a behavior that misrepresents or contradicts the system claimed specification [3]. This can result in blocks of missing data lasting from minutes to months, or even worse.

These faults and failures can significantly affect the application of analytical, statistical, and machine learning methods. They can impact the time series decomposition, detecting relevant lags, clustering time series, and forecasting tasks. Furthermore, they can lead to service degradation, negatively affecting the provided services of smart cities [16].

This article aims to reduce the impact of faults and failures on the data analysis in smart cities and other contexts

✉ Ana Almeida
anaa@ua.pt

Susana Brás
susana.bras@ua.pt

Susana Sargento
susana@ua.pt

Filipe Cabral Pinto
filipe-c-pinto@alticelabs.com

1 Departamento de Eletrónica, Telecomunicações e Informática, Universidade de Aveiro, 3810-193 Aveiro, Portugal

2 Instituto de Telecomunicações de Aveiro, 3810-193 Aveiro, Portugal

3 IEETA, DETI, LASI, Universidade de Aveiro, 3810-193 Aveiro, Portugal

4 Altice Labs, Aveiro, Portugal

by proposing a method to perform missing data imputation on time series. This work focuses on specific patterns in missing data, such as consecutive missing data, which is often overlooked in the literature. The main contributions are the following:

- A study focused on real-world missing data patterns observed in time series.
- A proposal of two mechanisms to generate two particular types of synthetic missing data.
- A study of the impact of different feature engineering approaches on the time series missing data imputation.
- A proposal of a method to perform imputation in time series named Focalize K-Nearest Neighbors (FKNN).
- An evaluation study of the proposed method using two real-world datasets.

Our FKNN method leverages the properties of time series, as correlated temporal features and temporal lags to the traditional KNN imputation. The study findings show that FKNN can handle the missing data and perform imputations in overlapping and disjoint missing patterns. We tested with two datasets with different characteristics and many combinations of missing data. While it only presents small benefits in the overlapping missing patterns, the mechanism can capture and perform a more accurate missing data imputation in disjoint patterns. Nevertheless, and as expected, as the percentage of missing data increases, we can observe a degradation of the method's performance. That is expected, since increasing the percentage of missing data also increases its uncertainty.

The paper is organized as follows. The related work on missing data imputation is presented in Sect. 2, while the proposed approach is described in Sect. 3. Then, the experiments and results are depicted in Sect. 4, and conclusions and directions for future work are presented in Sect. 5.

2 Related work

When an observation lacks a stored data value for a variable, it is considered as missing. A straightforward approach commonly used to deal with missing data is replacing the missing values with a specific value. For instance, we can use the value 0, the mean, the median, the mode, among others. When dealing with missing data in time series, forward and backward fill, the moving average, the Kalman filter, and interpolation methods are more effective than replacing missing values with a specific value. These methods are suitable for univariate time series imputation. However, these methods can severely fail when we have long sequences of missing data. Machine and deep learning approaches have gained popularity in recent years, since they are suitable for multivariate time series and can capture more patterns.

K-Nearest Neighbors (K-NN) is a popular predictive machine learning model that can be used for data imputation, using similar points in a dataset to guess missing data. While K-NN has been widely used, it does have some limitations. One issue is the curse of high dimensionality [9], which can make the model less accurate. Additionally, K-NN stores the complete dataset in memory [19], which can be a problem for larger datasets.

To address the limitations of KNN, researchers have proposed various improvements over the years. For example, Wettschereck et al. [19] developed a locally adaptive nearest-neighbor method for classification. This approach adapts the value of k locally within subsets of data with specific characteristics, rather than using a global value of k computed during cross-validation. The researchers tested four different variations of the method and found it very effective for particular datasets.

Another application of K-NN is imputing missing values in time series data. Oehmcke et al. [15] proposed a weighted version of K-NN ensembles with penalized Dynamic Time Warping (DTW) as a distance measure to perform multivariate time series imputation. They created windows for DTW using linear interpolation, and the correlation coefficient for global weights. The model gives more weight to consecutive missing data, expects highly correlated time series, and is more effective at higher missing ratios. Sun et al. [17] focused on traffic data and used a temporal gap-sensitive windowed K-NN method. This approach has proven to be robust even with high missing ratios. However, one potential issue is that there may be windows with only missing data, which can be problematic. Additionally, the dataset can be significantly larger than the original dataset. This can be computationally expensive, particularly for larger datasets.

Overall, K-NN is a valuable tool for data imputation, allowing researchers to explore ways to improve its accuracy and efficiency continually. As no single method is suitable for all datasets and missing data patterns, different approaches should be considered [19].

Khayati et al. [10] conducted a study to evaluate different imputation methods for time series data. Their research compared twelve methods based on algorithms such as Principal Component Analysis (PCA), Singular Value Decomposition (SVD), Expectation-Maximization (EM), and Autoregressive (AR) models. They tested these methods on several datasets with varying properties and missing data patterns. The authors found that no single method was the best for all cases. They suggested that the imputation method should depend on the specific dataset and missing data patterns.

The works in [6, 7, 12] propose the use of Artificial Neural Network (ANNs) to impute missing values in time series. Since these works consider multivariate time series, they use ANN designed to deal with time series data, such as Recurrent Neural Network (RNNs). However, since RNNs can face the vanishing gradients problem, these works do not use the traditional RNNs, but improved versions of this type of network.

Cao et al. [6] utilized a bidirectional Long Short-Term Memory (LSTM) graph network to handle multiple correlated missing values and provide generalized results. Che et al. [7] proposed a model based on Gated Recurrent Unit (GRU) with trainable decays, which is designed to capture previous properties. The main objective of this model is to make accurate predictions, even when the dataset may have missing data. However, the model's performance depends on the correlation between the missing data patterns and the prediction phase. If the two tasks are not correlated, the model may perform poorly. Li et al. [12] proposed a method that combines LSTM, Support Vector Regression (SVR), and collaborative filtering for multi-view learning. They focus on block missing patterns, considering spatial-temporal dependencies and different missing patterns despite a high missing ratio.

Other approaches include Generative Adversarial Network (GANs), AutoEncoder (AEs), and attention mechanisms. Luo et al. [13] proposed a model based on GANs. The model performs imputations using the discriminative and squared error loss and the closest generated complete time series. Kuppannagari et al. [11] proposed a method based on Graph Neural Network (GNN) and Denoising Autoencoder (DAE). Their approach can deal with spatio-temporal data and adapt to different combinations of missing blocks. Jing Bi et al. [4] presented a method combining GANs with an AE and an AR model. The GANs were utilized to understand the probability distribution of multivariate time series, the AE to extract features, and the AR model to capture time patterns. They compared the proposed model with K-NN and other algorithms. The proposed model achieved the best results; however, K-NN was the second-best model, presenting a similar performance (Mean Squared Error (MSE), Mean Absolute Error (MAE), and Area Under the ROC Curve (AUC)) to the proposed model.

Christopher Bülte et al. [5] proposed a two-stage multivariate time series imputation algorithm. The first stage involves a training phase, followed by an inference phase. Their algorithm uses LSTMs and an Attention mechanism. During the training phase, the LSTM model estimates the distribution of the missing data, using known values to predict the next ones. The output of this stage is a probability distribution for each missing value. The estimated distribution is then used to train the attention-based imputation model. The trained attention mechanism is used for the missing data during the inference phase. Only the attention mechanism is required to predict missing values. The authors evaluated and compared their algorithm using the MSE and the MAE, and observed that it achieved good results. One of the methods used in the comparison was the K-NN. The K-NN imputer was several times the best second overall model when considering all the features.

In our previous research [1], we introduced a technique called Focalize K-NN that can impute missing data in time series by taking advantage of their properties. Our research showed that Focalize K-NN performs particularly well when there are disjoint patterns present. However, the evaluation of the approach was very limited. In this article, we aim to strengthen our conclusions by improving the approach, the methodology, and by testing our method with different time series datasets that have different properties and characteristics.

Training the model with the appropriate dataset is crucial to achieve desirable results when dealing with deep learning algorithms. Deep learning algorithms usually require a training phase with substantial data. Contrarily, machine learning algorithms do not require such an extensive dataset; they are best suited to smaller datasets. Given the nature of the K-NN model, there is no need to train the model. This algorithm looks to similar neighbors to guess the missing values. We can utilize a portion of the model to test and determine the optimal parameters before validating the model. However, methods such as K-NN can be problematic with high-dimensional datasets. In this work, a modification of the traditional K-NN method is proposed and evaluated for data imputation in time series.

This paper considers K-NN instead of deep learning algorithms, since they have fundamental differences in their approaches and capabilities and distinct contexts in which they are best applied. Deep learning models are more complex and have a significantly larger number of hyperparameters than machine learning models. This complexity allows deep learning models to deal with high-dimensional data, capture more intricate patterns, and learn the best representations from data. Conversely, K-NN algorithms are optimal for smaller datasets. Furthermore, it can be difficult to train deep learning models when the missing data is spread through all the dataset. Moreover, they require a large set of data which, with missing data, is usually not available. The choice between deep learning and traditional machine learning methods should be driven by the specific requirements and characteristics of the problem being solved.

3 Methods

3.1 Handling time series data

A multivariate time series is a collection of time-dependent variables. Usually, when we mention multivariate time series, we have multiple time series aligned in time, meaning that they start and end in the same timestamp and have the same periodicity. Each time series can be seen as a feature of the multivariate time series. Therefore, we can represent the multivariate time series as a matrix $\mathcal{M}$ of dimensions m per n, where m is the number of time steps of the multivariate time series and the number of rows of the matrix, and n is the number of features and the number of columns of the matrix. Thus, $\mathcal{M}_{ij}$ refers to the value of feature j at time i. Furthermore, we expect to observe patterns between the different features and within each feature.

We consider two datasets, one from ENTSO-e[1] and another one from OpenWeather.[2] These datasets have different characteristics. From ENTSO-e, we obtained electricity-related data from 24 countries during 240 weeks (almost five years) since the beginning of 2019. We received the actual total load per bidding zone per market time unit. We applied scaling techniques to ensure uniformity among data from different countries. OpenWeather provided a weather dataset containing one year of hourly measurements in 2022 from twenty cities. The dataset includes variables such as temperature, pressure, humidity, wind speed, wind direction, wind gusts, and cloudiness. Therefore, for the dataset from ENTSO-e, we had 24 vectors. For the dataset from OpenWeather, we had 140 vectors (20 cities and 7 sensors per city). Once more, we applied scaling techniques to ensure uniformity among data from the distinct sensor types.[3]

Regarding the scaling technique used, we decided to apply a min-max scaler to normalize data between 0 and 1, since K-NN is an algorithm based on distances, such as the Euclidean distance. Thus, we can ensure that all features are on the same scale and validate that this is not influencing the model output.

Feature engineering becomes crucial when dealing with time series data in data science. One common method to tackle cyclic and seasonal patterns is selecting the most relevant lags [2]. Some common phenomena that we can observe when working with electricity and weather data are diurnal variations (day versus night) and seasonal variations (summer versus winter). However, we can also observe other phenomena, even though they might occur less frequently, such as droughts and heat waves. Therefore, the past is related to the present and the future. We decided to use lags of the previous 24 h (1 day), 48 h (2 days), and 168 h (7 days). Since we are studying missing values imputation, it is feasible to use data such as 24, 48, and 168 h afterward. This methodology allows to better capture the missing values pattern.

3.2 Generation of synthetic missing data

Each fault or failure within the system can result in sequences of missing data of varying durations. Depending on the cause of the fault or failure, we may experience missing data in a single time series (such as a damaged sensor), multiple time series (such as when communication with a data-collecting unit is lost), or all time series (such as during a network outage or blackout). This work uses the term "synthetic missing data" to refer to the dataset created simulating missing data.

The impact of the faults or failures in the system can be different, even under similar conditions, according to the dataset's characteristics, including the relationship between the different features. For instance, in the dataset from ENTSO-e, if we lose load information from countries with similar patterns, achieving good imputation performance can be more challenging. In the case of the dataset from OpenWeather, we have more redundancy since we have more relationships between the features. For example, suppose we lose temperature data from cities with similar patterns. In that case, if we have that city's humidity and wind information, the imputation may be straightforward to implement. Nevertheless, we can also have missing data in the most correlated cities and the sensors from the city.

Since we expect a particular pattern type regarding our missing data, we created mechanisms to generate similar patterns. Figure 1 contains the two patterns designed to evaluate the experience. We have the overlap pattern on the left and the disjoint pattern on the right.

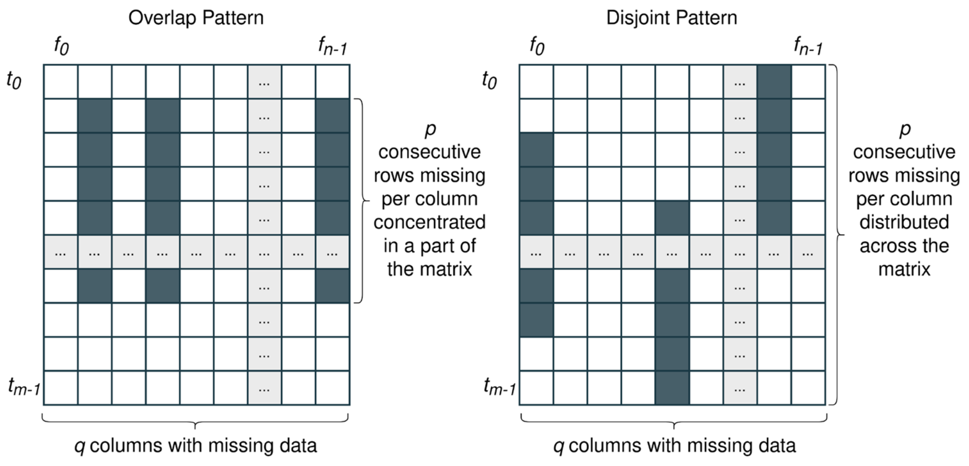
In the overlap pattern, q sensors fail simultaneously during a period p. In a more severe case, we will have a blackout if all sensors fail simultaneously. In the disjoint pattern, the sensors fail at different times. In the case of the dataset from ENTSO-e, we can simultaneously remove data from countries with similar patterns. In the dataset from OpenWeather, we can lose data from similar cities and data from the different sensors from the same station.

Figure 2 shows the impact of missing data according to the time interval, number of sensors, and station in each dataset. Note that when we have, for instance, one sensor affected in 10 different stations, we force that sensor to be the same in all stations (e.g. temperature sensor). The same

[1] https://www.entsoe.eu/.

[2] https://openweathermap.org/.

[3] We do not have permission to share the datasets; nevertheless, both ENTSO-e and OpenWeather provide APIs to access the data and API keys upon request.



Overlap Pattern
f_0
f_{n-1}
t_0
t_{m-1}
p consecutive rows missing per column concentrated in a part of the matrix
q columns with missing data
Disjoint Pattern
p consecutive rows missing per column distributed across the matrix
q columns with missing data

Fig. 1 Missing block patterns: overlap and disjoint

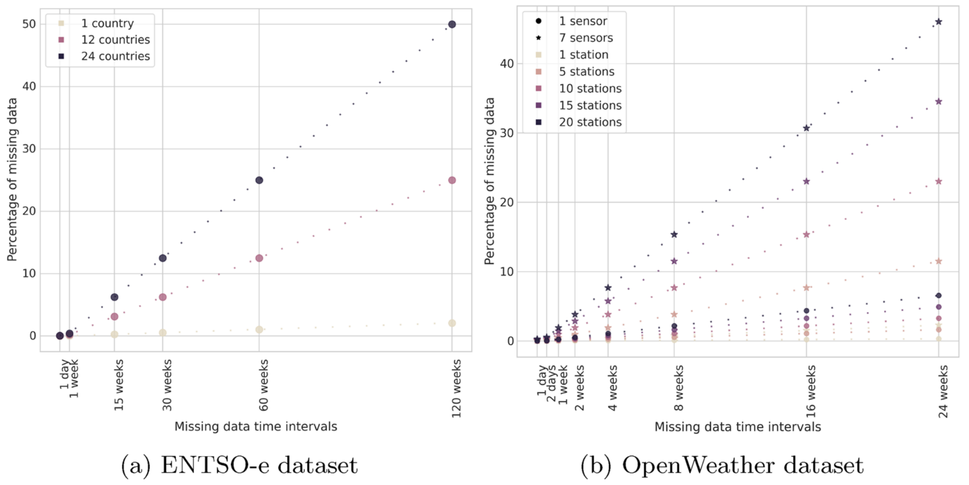


Fig. 2 Percentage of missing data

happens if we have, for instance, seven sensors affected in one station. Dealing with missing data in highly correlated sensors poses additional challenges. We design 36 (6 time intervals × 3 countries combinations × 2 missing patterns) different configurations to test the impact of missing data in the ENTSO-e dataset, and 160 configurations (8 time intervals × 2 sensor combinations × 5 station combinations × 2 missing patterns) for the OpenWeather dataset. With this, we try to approximate our study to real-world imputation problems in multivariate time series.

3.3 Algorithms implementation and evaluation

Various methods can be used to impute missing values in data, such as replacing the missing value or interpolating to estimate a probable value. Under the scope of this work, 21 baseline approaches are selected from both methods for analysis. The selected methods are described in Table 1. We chose some of the most used techniques to deal with missing data, such as replacing the missing data with a value (zero, the mean, and the median value). We also tried interpolation techniques based on polynomials, forward fill, and backward fill, among others. From these 21 methods, we selected the best three and compared them with K-NN imputation and our proposed method.

From the presented methods in Table 1, some of them may still leave missing values after imputation. Considering the forward fill, the method will use the previous value to impute the missing values. However, suppose we have missing values at the beginning of our time series. In that case, the method has no previous value to use in the imputation, remaining the missing values in the time series. Something similar can happen with the backward fill at the end of the series. This issue is solved using a forward/ backward strategy until all missing data has been imputed.

During our experiments, we compare the performance of two types of K-NN imputation—uniform K-NN, where uniform weights are assigned to all the nearest neighbors, and weighted K-NN, where different weights are assigned. The weight is calculated as a function inverse of the distance. After evaluating the performance of both types of methods, we decided to present the results for the best-performing K-NN imputation technique. Furthermore, we use different values for the number of neighbors in the analysis: 2, 3, 5, 7, 9, 11, 13, 15, and 19.

Table 1 Baseline methods [14, 18]

Type of method	Details	Description
Replace missing data with a value	Zero value	Uses the zero to replace the missing data
	Mean	Uses the mean value to replace missing data
	Median	Uses the median value to replace missing data
	Nearest	Interpolates half-integers round down
Interpolation	Forward Fill	Interpolates performing forward fill of values
	Backward Fill	Interpolates performing backward fill of values
	Linear	Interpolates ignoring the time series index, assuming data is equally spaced
	Time	Interpolates using temporal resolutions
	Spline	One dimensional smoothing cubic spline
	Barycentric	Polynomial interpolation for a set of points that can adapt the number of values being interpolated
	pchip	Interpolates using a one dimension monotonic cubic interpolation
	cubicspline	Interpolates using a piecewise cubic polynomial, that is twice continuously differentiable
	Zero	Spline interpolation of order 0
	Slinear	Spline interpolation of order 1
	Quadratic	Spline interpolation of order 2
	Cubic	Spline interpolation of order 3
	Piecewise Polynomial	Interpolates using piecewise polynomials
	Akima	Interpolates using piecewise cubic polynomials, using a continuously differentiable sub-spline
	Polynomial, Order 3	Polinomial interpolation of order 3
	Polynomial, Order 5	Polinomial interpolation of order 5
	Polynomial, Order 7	Polinomial interpolation of order 7

Based on the K-NN algorithm, we developed Algorithm 1 for the Focalized KNN. Our algorithm receives one or several time series in a matrix $\mathcal{M}$ with *m* rows (time steps) and *n* columns (time series). *f* starts as an empty dictionary and will receive as keys the columns with missing data, and each column with missing data will store a list with the most correlated features and the most relevant time lags. If the matrix contains missing data, we will fill in the missing data one column at a time. Note that, this implementation of K-NN does not change the values that were not missing; it only tries to guess the missing ones. Therefore, by replacing the *i*-th column of the original matrix with the column with the imputed values, we are not changing the real values.

Algorithm 1 FKNN for Time Series Missing Data Imputation

```
1:  procedure TimeSeriesImputation($\mathcal{M}$)
2:      $f \leftarrow$ empty dictionary
3:      for $i \leftarrow 1$ to $n$ do
4:          if $\mathcal{M}[:, i]$ has missing values then
5:              Add most correlated features of $\mathcal{M}[:, i]$ to $f[i]$
6:              Add temporal lags of $\mathcal{M}[:, i]$ to $f[i]$
7:          end if
8:      end for
9:      while $\mathcal{M}$ has missing data do
10:         $i \leftarrow$ index of column with minimum missing values (number of missing
11:             values bigger than 0)
12:         $c \leftarrow$ column with index $i$
13:         $\mathcal{X} \leftarrow$ matrix with column $c$, relevant features, and lags
14:         $\mathcal{X}_{imputed} \leftarrow$ apply KNN to matrix $\mathcal{X}$
15:         $\mathcal{M}[:, i] \leftarrow \mathcal{X}_{imputed}[:, i]$
16:     end while
17:     return $\mathcal{M}$
18: end procedure
```

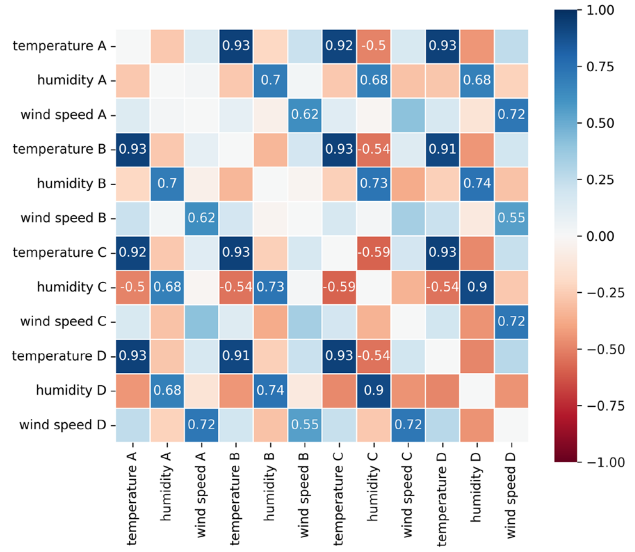
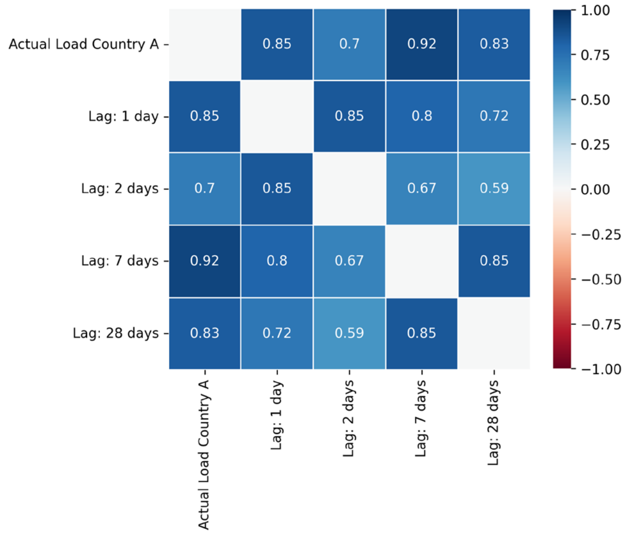


Fig. 3 Correlation matrix using data from 4 cities and 3 sensors from OpenWeather dataset



	Actual Load Country A	Lag: 1 day	Lag: 2 days	Lag: 7 days	Lag: 28 days
Actual Load Country A		0.85	0.7	0.92	0.83
Lag: 1 day	0.85		0.85	0.8	0.72
Lag: 2 days	0.7	0.85		0.67	0.59
Lag: 7 days	0.92	0.8	0.67		0.85
Lag: 28 days	0.83	0.72	0.59	0.85	

Fig. 4 Correlation matrix using data from 1 country and 4 temporal lags from ENTSO-e dataset

Since K-NN relies on distances, if our dataset includes numerous or irrelevant features, the performance of the K-NN will suffer. Given the nature of our problem, we are dealing with time series data where some features are highly correlated with others. Therefore, we can select features given the properties of the time series. We perform imputation on the matrix by applying K-NN imputation focalized on each column with missing data. We apply this method to each column separately. First, we identify the column with the fewest missing values. Then, we chose the best features and relevant lags for that column to use as inputs for the K-NN imputation. We repeat this process for each column until all missing values were filled in.

Feature selection methods are evaluated to identify the most important features in the data domain. However, one requirement of the selected method is to identify the important features even in the presence of a high recurrence of missing data. We select the *Spearman*'s correlation coefficient since it is a non-parametric, monotonic method that allows to measure the relationship between two features. We consider selecting a constant value for the number of features, such as choosing the top ten most correlated features, or using a threshold to select features with a correlation value greater than the threshold. Figure 3 showcases an example with four cities, three sensors, and a threshold of 0.5. For this case, we consider features with values greater than 0.5 or lower than − 0.5 to be highly correlated. We exclude the diagonal of the matrix since it refers to self-comparison. For a performance improvement and a better feature selection, lag selection (Sect. 3.1) is also applied.

We perform a similar analysis for the temporal lags. Figure 4 depicts the relationship between the temporal lags and one of the load data from one of the countries. As we can observe, there is a high correlation between each lag and the original time series, especially considering the seven-days lag.

In this work, the workflow in Fig. 5 is followed. Initially, we use a dataset with a multivariate time series with no missing data. Then, we generate synthetic missing data to obtain a new multivariate time series containing missing data. After that, we apply imputation methods to the time series with missing data, and obtain a reconstructed time series. Finally, we evaluate the methods by comparing the original time series with the reconstructed one. The N× indicates that the process can be repeated multiple times.

For the evaluation purpose, we select two metrics: MSE and R^2-Score [2, 8]. Considering that the problem being evaluated in this work matches the requisites of a regression problem, R-squared and MSE are the suitable measures to evaluate the accordance of the model with the data. These metrics assess the error of the values predicted and the variability of the data explained by the model. We therefore use R^2-Score to choose the best model. MSE can have values from 0 to plus infinity, and it penalizes big errors when compared to similar metrics such as MAE. A good model has a low MSE. R^2-Score provides valuable information about how well the model can fit the data. Its values range from less than infinite to 1, and a good model has an R^2-Score close to 1.

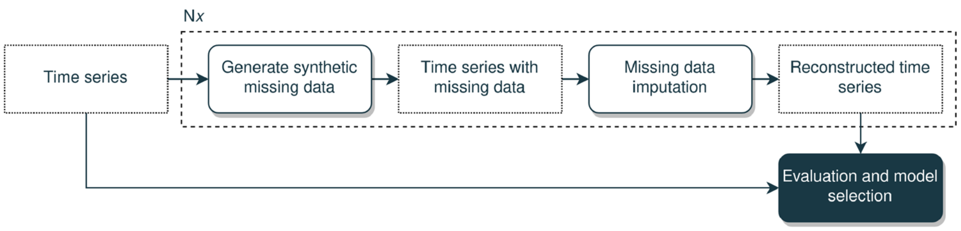

Nx
Time series
Generate synthetic missing data
Time series with missing data
Missing data imputation
Reconstructed time series
Evaluation and model selection

Fig. 5 Workflow to evaluate and select the best model

4 Results and discussion

This section presents the results obtained by comparing the statistical baseline methods, K-NN, and the proposed FKNN. We evaluate the methods in two multivariate time series datasets with different characteristics, as described in Sect. 3. We conduct tests with different parameters to improve the suitability of the models. We test the uniform and the weighted version of the K-NN imputation and the FKNN imputation. We explore different scenarios considering the number of neighbors, such as 2, 3, 5, 7, 9, 11, 13, 15, and 19.

Furthermore, we also try different approaches for the lag combinations to evaluate how they affect FKNN's performance. For the ENTSO-e dataset, we test lags of one hour, one day, two days, seven days, and 28 days, as well as no lags. For the OpenWeather dataset, we only test lags of one hour, one day, two days, and seven days, as well as no lags. We adopt the lag selection to the properties of each dataset. For instance, while for the ENTSO-e dataset, we have almost five years of data, for the OpenWeather, we only have one year of data. We also experiment with different thresholds for selecting the most relevant features, including values of 0.5, 0.75, and 0.9. Additionally, we test the selection of the ten most correlated features based on these thresholds and the best 10, 20, 30, or 40 features.

4.1 Analyzing FKNN performance across different values of *k*

Choosing the best k (number of neighbors) in K-NN is a challenging task. One common strategy is to compute the square root of the number of samples. However, for our datasets, this would result in a very high value of k: 200 for the ENTSO-e dataset and 94 for the OpenWeather dataset. Larger values of k can lead to high bias and lower variance and require more computation capabilities. On the other side, lower values present high variance and low bias. To determine the optimal value of k, we test the model's performance with different k values. We chose odd numbers for k, starting with two instead of one, because we expect more than one neighbor. The results can be visualized in Figs. 6 and 7. As we can observe, the curves start to flatten around nine neighbors; therefore, we decide to use k with the value of 9 for both datasets and patterns.

4.2 Assessing the influence of missing data on model performance

This subsection assesses the impact of different missing data patterns on two distinct datasets. We start by analyzing the overlap missing pattern. From our experiments, the three best statistical baseline methods are, in most cases, replacing the missing values with the mean and median values, and performing linear interpolation. Figures 8a and 9a contain the R^2-Score of the best statistical models, the best K-NN model, and the best FKNN for the ENTSO-e and OpenWeather datasets, respectively.

Strategies like replacing missing data with the mean value are better for statistical analysis when long blocks of data are missing instead of using interpolation techniques. Note that all statistical methods only consider the univariate time series for replacing the missing values. For both datasets, using a k equal to 9 and uniform weights presents a good approach for both K-NN and FKNN. The best FKNN model uses nine neighbors, uniform weights, and a threshold of

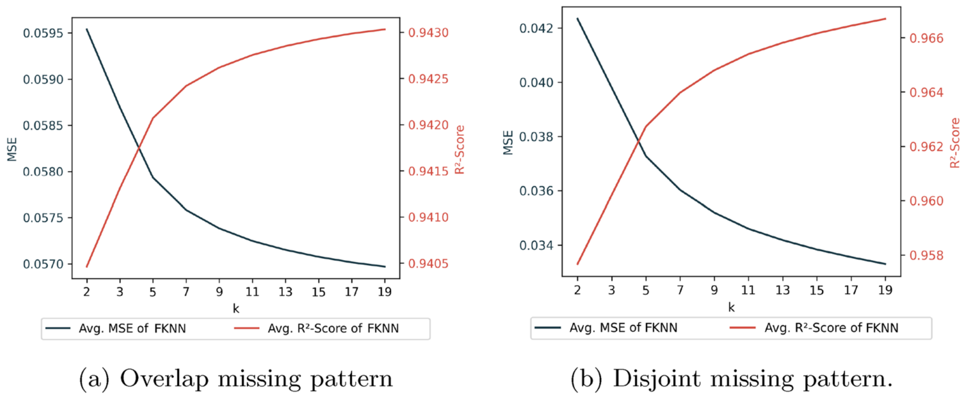
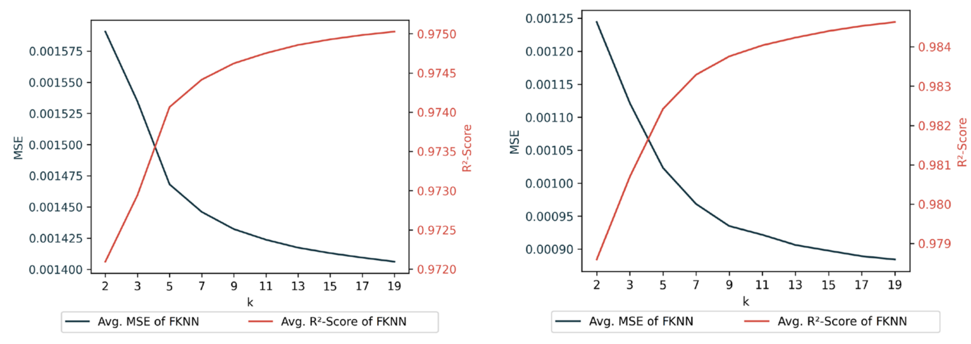
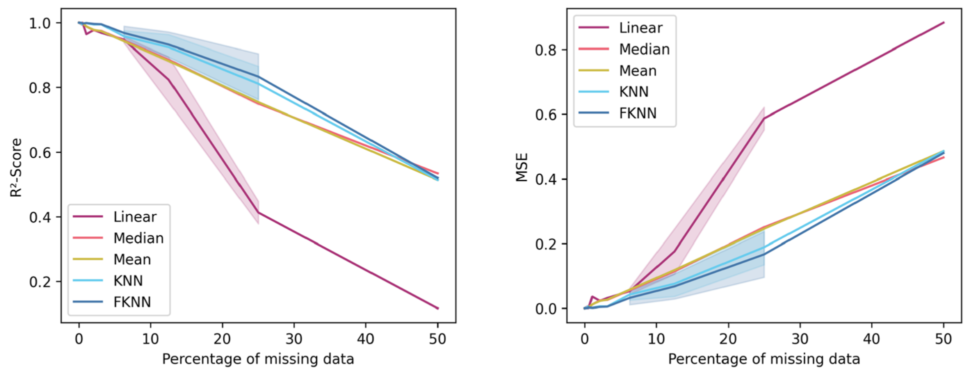


Fig. 6 Dataset ENTSO-e: performance of FKNN versus k



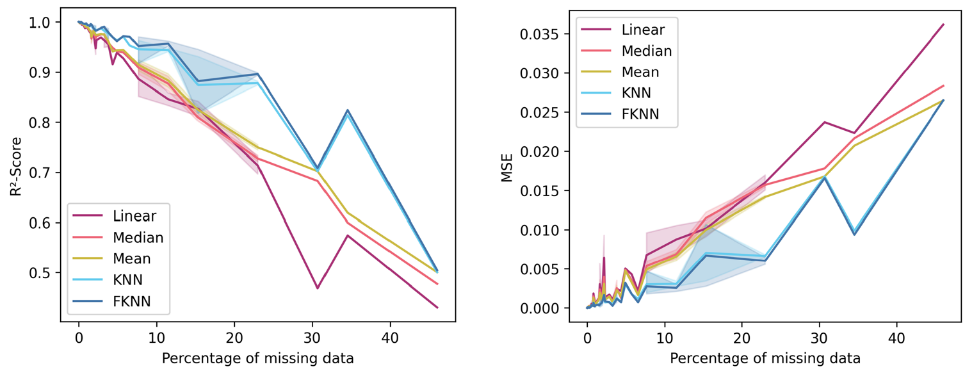
(a) Overlap missing pattern (b) Disjoint missing pattern.

Fig. 7 Dataset OpenWeather: performance of FKNN versus *k*



(a) R^2-Score (b) MSE

Fig. 8 Imputation methods with dataset ENTSO-e: overlap missing pattern



(a) R^2-Score (b) MSE

Fig. 9 Imputation methods with dataset OpenWeather: overlap missing pattern

0.5. Regarding the best lags, it depends on the dataset. For the dataset from ENTSO-e, we use the previous day and week. For the dataset from OpenWeather, we use the last day, two days, and one week. Figures 8b and 9b compare the obtained MSE values. After analyzing Figs. 8 and 9, we can conclude that the improvement of our method compared to the standard version of K-NN is low. Furthermore, when we start having percentages of missing data of 50% or higher, the best strategy might be to replace the missing data with the mean or median value.

In Figs. 9a and 9b, we can observe a zig-zag pattern. Each presented experiment is independent, having the missing data affecting different columns and rows. The relationship between the percentage of missing values and MSE or R^{2}²-Score does not present a linear behavior. This can be explained by its dependence on the patterns printed

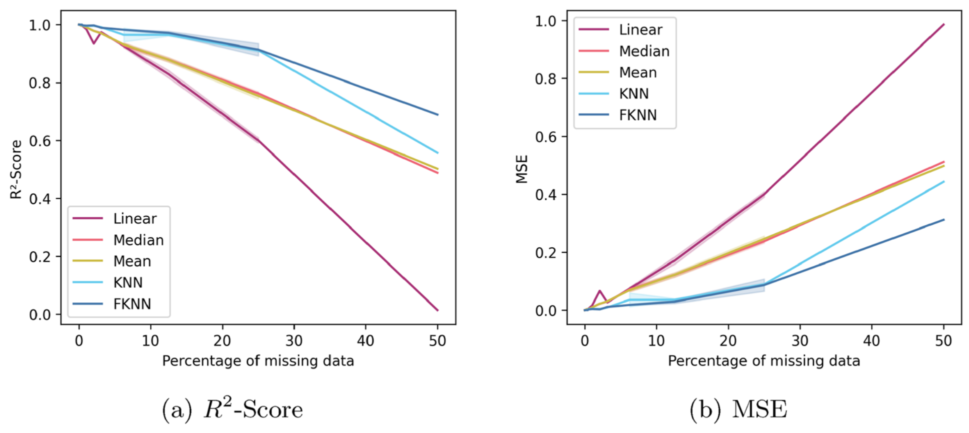


Fig. 10 Imputation methods with dataset ENTSO-e: disjoint missing pattern

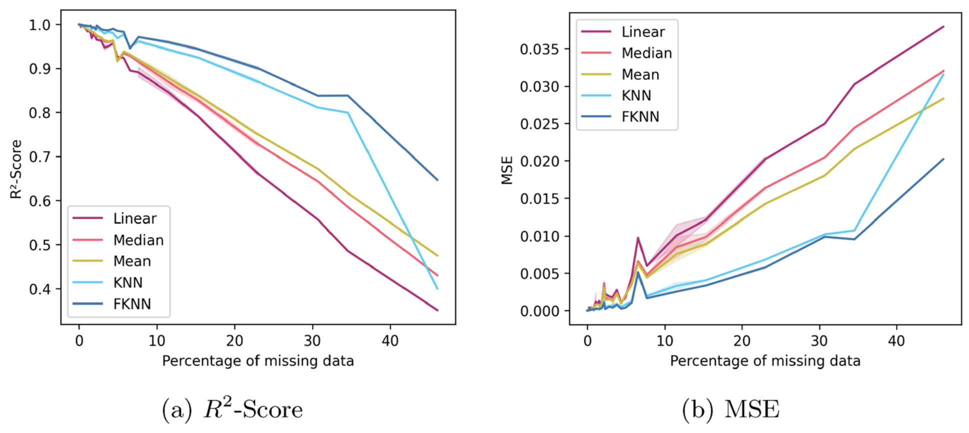


Fig. 11 Imputation methods with dataset OpenWeather: disjoint missing pattern

on the data and the ability of the model to predict the missing values given those patterns.

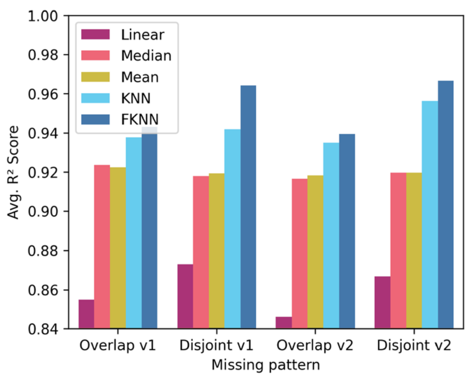
Regarding the disjoint pattern, once more, the best models are the linear interpolation, replacement of missing data with the median and the mean, and the K-NN with nine neighbors and uniform weights. The best version of FKNN also uses nine neighbors and uniform weights. Besides, for the best version, we use a threshold of 0.5. For the lags, we use the last day and last week for the ENTSO-e dataset, and the previous hour and the previous day for the OpenWeather dataset. The proposed FKNN algorithm performs well for the disjoint pattern, as shown in Figs. 10 and 11, especially when there is a more significant percentage of missing data. In this case, we can observe considerable gains when applying the proposed method, FKNN.

The zig-zag pattern observed in Fig. 11 for the imputation in the OpenWether dataset in the disjoint missing pattern is less significative when compared with the overlap missing pattern in Fig. 9. The nature of the generation of synthetic missing data impacts the performance of the algorithms profoundly.

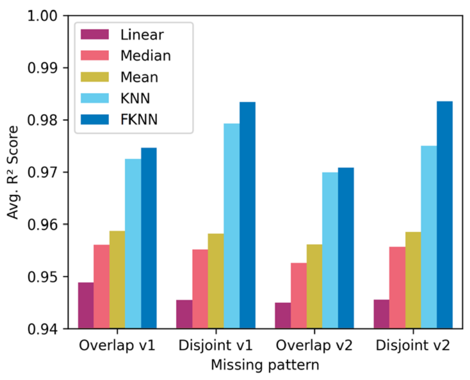
4.3 Comparing FKNN performance

Based on our experiments, we have observed that the uniform version of K-NN tends to give better results than the weighted version of K-NN. Moreover, we found that, using the next temporal periods as features does not result in improved performance as compared to using only past lags. We also experiment with using only the best positive correlation versus the best absolute correlations, and we notice that when we consider the best absolute correlations, the performance of our model slightly improve.

Furthermore, we compare the average of the R^2-Score for all missing data values. We repeate the experiment by generating more missing data patterns for both datasets. The patterns we use for testing are overlap v1 and disjoint v1, while for evaluation, we use overlap v2 and disjoint v2. Note



(a) Dataset ENTSO-e



(b) Dataset OpenWeather

Fig. 12 Avg. R^2-Score per method

that we have more values with a low percentage of missing data, less than 10%. Nevertheless, Fig. 12 shows that our model achieves the best performance overall.

On a final note, we expect to achieve the best results in a dataset with stronger temporal patterns, such as seasonality and cyclic patterns. However, these time-series datasets are not straightforward to find freely available.

5 Conclusion and future work

Missing data in time series can severely impact the analysis of data and its applications. A common issue is the block missing data pattern, which causes a loss of sequential information. Within block missing data patterns, we can observe two patterns: an overlapping pattern and a disjoint missing pattern.

This article proposed a method based on the traditional K-NN method to impute missing data in multivariate time series. The proposed method, called FKNN, is more effective for disjoint missing patterns than overlapping ones. FKNN enables the use of accurate data even in the presence of data loss. In these experiments, the model performance seems to increase with the number of neighbors in the KNN model. However, this increase in the number of neighbors also impacts the bias since the model presents more difficulty in generalizing due to the lack of variance in data. So, to better describe the data without decreasing the model performance, the decision of the model performance should consider the balance between bias and variance.

In the future, we plan to develop additional techniques for imputing missing data in time series, including those based on deep learning. We also aim to explore more patterns of missing data.

Author contributions AA: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Visualization, Writing—original draft. AA and SB: Validation. SS: Funding acquisition, Project administration. SS, SB, and FCP: Supervision, Writing—review and editing

Funding Open access funding provided by FCT|FCCN (b-on). This study was funded by the PRR - Plano de Recuperação e Resiliência and by the NextGenerationEU funds at Universidade de Aveiro, through the scope of the Agenda for Business Innovation "ATT - Agenda Mobilizadora Acelerar e Transformar o Turismo" (Project no. 47 with the application C645192610-00000060). Ana Almeida acknowledges the Doctoral Grant from Fundação para a Ciência e Tecnologia (2021.06222.BD). Susana Brás is funded by national funds, European Regional Development Fund, FSE, through COMPETE2020 and FCT, in the scope of the framework contract foreseen in the numbers 4, 5 and 6 of the article 23, of the Decree-Law 57/2016, of August 29, changed by Law 57/2017, of July 19. We thank OpenWeather and ENTSO-e for providing the datasets.

Availability of data and materials Not applicable.

References

1. Almeida A, Brás S, Sargento S, Pinto FC (2023) Time series imputation in faulty systems. In: Pertusa A, Gallego AJ, Sánchez JA, Domingues I (eds) Pattern recognition and image analysis. Springer, Cham, pp 28–39. https://doi.org/10.1007/978-3-031-36616-1_3
2. Almeida A, Brás S, Oliveira I, Sargento S (2022) Vehicular traffic flow prediction using deployed traffic counters in a city. Futur

Gener Comput Syst 128:429–442. https://doi.org/10.1016/j.future.2021.10.022
3. Bass L, Clements P, Kazman R, Safari OMC (2021) Software architecture in practice, 4th edn. SEI series in software engineering. Addison-Wesley. https://books.google.pt/books?id=BWpuzgEACAAJ
4. Bi J, Wang Z, Yuan H, Ni K, Qiao J (2022) Multi-indicator water time series imputation with autoregressive generative adversarial networks. In: 2022 IEEE international conference on systems, man, and cybernetics (SMC), pp 2003–2008. https://doi.org/10.1109/SMC53654.2022.9945604
5. Bülte C, Kleinebrahm M, Ümitcan Yilmaz H, Gómez-Romero J (2023) Multivariate time series imputation for energy data using neural networks. Energy AI 13:100239. https://doi.org/10.1016/j.egyai.2023.100239
6. Cao W, Wang D, Li J, Zhou H, Li L, Li Y (2018) Brits: bidirectional recurrent imputation for time series. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R (eds) Advances in neural information processing systems, vol 31. Curran Associates Inc, New York
7. Che Z, Purushotham S, Cho K, Sontag D, Liu Y (2018) Recurrent neural networks for multivariate time series with missing values. Sci Rep. https://doi.org/10.1038/s41598-018-24271-9
8. Davide Chicco Matthijs J, Warrens GJ (2021) The coefficient of determination r-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Comput Sci 7:7. https://doi.org/10.7717/peerj-cs.623
9. Grus J (2019) Data science from scratch: first principles with Python. O'Reilly Media Inc
10. Khayati M, Lerner A, Tymchenko Z, Cudre-Mauroux P (2020) Mind the gap: an experimental evaluation of imputation of missing values techniques in time series. Proc VLDB Endow 13:768–782. https://doi.org/10.14778/3377369.3377383
11. Kuppannagari SR, Fu Y, Chueng CM, Prasanna VK (2021) Spatio-temporal missing data imputation for smart power grids. In: Proceedings of the twelfth ACM international conference on future energy systems, pp 458–465. e-Energy '21, Association for Computing Machinery, New York. https://doi.org/10.1145/3447555.3466586
12. Li L, Zhang J, Wang Y, Ran B (2019) Missing value imputation for traffic-related time series data based on a multi-view learning method. IEEE Trans Intell Transp Syst 20(8):2933–2943. https://doi.org/10.1109/TITS.2018.2869768
13. Luo Y, Zhang Y, Cai X, Yuan X (2019) E^{2}gan: end-to-end generative adversarial network for multivariate time series imputation. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19. International joint conferences on artificial intelligence organization, pp 3094–3100. https://doi.org/10.24963/ijcai.2019/429
14. McKinney W (2010) Data structures for statistical computing in python. In: van der Walt S, Millman J (eds) Proceedings of the 9th python in science conference, pp 56–61. https://doi.org/10.25080/Majora-92bf1922-00a
15. Oehmcke S, Zielinski O, Kramer O (07 2016) KNN ensembles with penalized DTW for multivariate time series imputation, pp 2774–2781. https://doi.org/10.1109/IJCNN.2016.7727549
16. Shamsi JA (2020) Resilience in smart city applications: faults, failures, and solutions. IT Prof 22(6):74–81. https://doi.org/10.1109/MITP.2020.3016728
17. Sun B, Ma L, Cheng W, Wen W, Goswami P, Bai G (2017) An improved k-nearest neighbours method for traffic time series imputation. In: 2017 Chinese automation congress (CAC), pp 7346–7351. https://doi.org/10.1109/CAC.2017.8244105
18. ...Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat İ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P (2020) SciPy 1.0 contributors: SciPy 1.0: fundamental algorithms for scientific computing in python. Nat Methods 17:261–272. https://doi.org/10.1038/s41592-019-0686-2
19. Wettschereck D, Dietterich T (1993) Locally adaptive nearest neighbor algorithms. In: Cowan J, Tesauro G, Alspector J (eds) Advances in neural information processing systems, vol 6. Morgan-Kaufmann, New York