**Improving the Forecast Accuracy of Protected Data Using Time Series Features**

**Abstract**

Existing data privacy methods degrade forecast accuracy to unusable levels. To overcome this problem, we incorporate time series features that are predictive of forecast accuracy into a matrix-based privacy method called k-nearest time series + (k-nTS+) swapping. We apply our method to a forecasting competition data set where the identities of the time series are hidden but an adversary attempts to identify the protected series. Compared to other privacy methods, we find that k-nTS+ swapping degrades forecast accuracy by only 13.9% while maintaining similar privacy levels. We also find that k-nTS+ protected time series are more representative of the original data and time series features, leading to increased trust between data owners and forecasters.

Keywords: data privacy; time series features; forecast accuracy; identification disclosure risk

1. **Introduction**

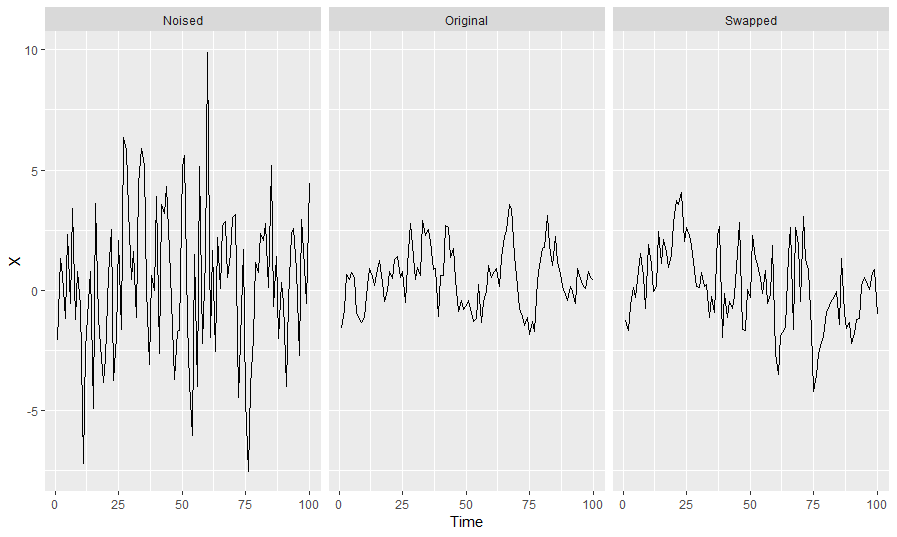
Personally identifiable time series data are abundant and require protection (Boone et al., 2019). Recently, the General Data Protection Regulation (GDPR)[[1]](#footnote-1) forced data owners to either anonymize their personal data or place strict limitations on data transfers and processing[[2]](#footnote-2). However, past research found that anonymizing time series data significantly reduced forecast accuracy (Gonçalves et al. 2021). As a result, forecasters (Gonçalves et al., 2021b) used complex data processing solutions to share accurate forecasts without sharing the adjoining time series data. In this paper, we propose a privacy solution for data owners to anonymize time series data directly with good forecast accuracy.

Time series data are either stored in a single data set (centralized) or spread across multiple data owners and/or data sets (decentralized). In the decentralized scenario, multi-party computation and federated learning enable privacy-preserving collaborative forecasting to ensure accurate forecasts while protecting sensitive data (Gonçalves et al., 2021a; Goncalves, Bessa et al., 2021b; Sommer et al., 2021). For example, in a decentralized scenario, data owners can sell time series data to a market operator who then sells forecasts of the time series data to multiple buyers. This approach has the advantage of creating a market of economic incentives for data sharing while limiting data transfer and protecting privacy. However, there are still privacy risks including potential data breaches with the transfer and storage of the time series data to the market operator.

Our paper focuses on the centralized scenario where a single data owner uses privacy methods to protect the original time series data set. These privacy methods anonymize the time series data by directly altering the values within the data set to increase the privacy level of the protected data set. This approach assumes the data gets out eventually through an internal employee (or data breach) and protects against this worst-case scenario. The primary goal of the privacy methods is to limit the ability of an adversary to identify data subjects (identification disclosure, in our case, discovering the true identity of a time series) and learn sensitive information about them (attribute disclosure, learning sensitive values after an identification is made). The secondary goal of the privacy methods and the primary concern for forecasters is that these privacy methods do not significantly reduce forecast accuracy.

For illustration of the centralized scenario, consider the example shown in Figure 1. The time series in the middle plot is a simulated AR(1) process with autoregressive parameter . The series on the left is the same simulated series with random noise added to each time period proportional to the standard deviation of the simulated series. The series on the right is generated by randomly swapping the values of the middle series with values from two other simulated AR(1) processes, both with .

**Figure 1:** Comparison of protected AR(1) processes to the original AR(1) process.



Estimating an ARIMA(1, 0, 0) model on the simulated series in the middle yields an estimate of with a standard error of 0.07, while the noised series on the left yields an estimate of with a standard error of 0.10. The swapped series on the right yields an estimate of and a standard error of 0.08. Figure 1 shows the series on the right is more representative than the noised version, but in both cases, the time series features (*e.g.*, AR (1) parameter, variance, spike, maximum variance shift, spectral entropy, etc.) from the middle series change. However, it is not immediately clear how these changes affect forecast accuracy.

The proposed method in this paper improves the series on the right by swapping time series values with each other only if their underlying features are likely to maintain forecast accuracy. The k-nTS+ swapping method is illustrated in Figure 2 and begins by generating protected data using baseline privacy methods (*e.g.*, additive noise, differential privacy) commonly used in practice. Then, the method generates forecasts for the original and protected series and compares their accuracies. To improve forecast accuracy (blue arrows in Figure 2) of the protected data, k-nTS+ swapping uses a machine learning-based feedback loop with RReliefF (Robnik-Sikonja & Kononenko, 2003) and Recursive Feature Elimination (RFE) (Gregorutti et al., 2017) on the accuracy results to rank (RReliefF) and select (RFE) the time series features most predictive of forecast accuracy. For privacy protection, it computes a feature-based distance matrix to randomly choose a time series to swap values with. As a result, the k-nTS+ swapping method produces protected time series that are more likely to preserve useful time series features for forecasting.

**Figure 2: k-nTS+ swapping method (Blue arrows indicate the feedback loop which informs the swapping).**

Graphical user interface, application

Description automatically generated

Our contributions to the literature are two-fold. First, we measure the changes in forecast accuracy from multiple forecasting models and privacy methods. The literature demonstrated that differential privacy degrades forecast accuracy for VAR models and recurrent neural networks (RNNs) (Gonçalves et al., 2021a), but little work explains why different forecasting models produce different accuracies on protected data. We newly analyze how time series features change vis-à-vis accuracy and show which forecasting models are more robust to changes in time series features after privacy protection. We also examine whether the magnitude, direction, or volatility of these privacy adjusted forecasts improve forecast accuracy.

Second, we propose a matrix-based method that limits the degradation in forecast accuracy due to privacy protection. To the best of our knowledge, previous research does not improve the forecast accuracy of protected data in a centralized scenario where the entire data set is shared. In the privacy literature, the usefulness of protected data is often overlooked (Blanco-Justicia et al., 2022), but is of utmost concern to forecasters. Recent research (Schneider et. al., 2018) maintained marketing metrics within 10-15% of the original by including a marketing loss function in their privacy method. Similarly, we use time series features predictive of forecast accuracy in our k-nTS+ swapping method.

Results show that our method provides significantly better forecast accuracy (+13.9% worse than the original forecasts) at similar levels of privacy to baseline privacy methods. Furthermore, using the *performance gap* from Petropoulos & Siemsen (2022), we show that k-nTS+ protected time series are more representative of the original series, leading to improved trust between data owners and forecasters.

In Section 2, we review the relevant literature. Section 3 describes the *k*-nTS swapping method and proposes the k-nTS+ swapping method with a feedback loop of the time series features. Section 4 presents the empirical application and Section 5 concludes.

1. **Literature Review**
   1. *Privacy Methods*

In a decentralized scenario, Goncalves et al. (2021c) modeled a data market where data owners are compensated for sharing their time series data and purchase only forecasts based on the data from other parties. However, the original time series were still shared with a central party which discourages data owners from sharing time series due to privacy concerns such as a data breach. Other privacy-preserving solutions for collaborative forecasting include secure multi-party computation, decomposition-based methods, and data transformation techniques (see Gonçalves et al. 2021a).

In a centralized scenario, the data owner uses privacy methods to generate protected data sets for forecasting. Gonçalves et al. (2021a) showed that differential privacy reduces the forecast accuracy of VAR models under very high values of the privacy parameter (weak privacy protection). Others have also studied the application of differential privacy to time series (Imtiaz et al., 2020; Liyue Fan & Li Xiong, 2014). Luo et al. (2018) simulated data integrity attacks and found that multiplicative noise reduces forecast accuracy by over 21% when only half the data points are altered. Their results likely understate the reduction in forecast accuracy from privacy methods because only half the data points were altered.

Other privacy methods include generalization where the structure of the original data set is changed. Data records can be aggregated or combined to make every record (or time series) identical to at least other records (or time series). For example, daily time series data can be aggregated to weekly time series data (frequency aggregation), or each time series can be averaged with its most similar time series ( anonymity). Using anonymity (weak privacy), Nin & Torra (2009) evaluated the change in forecast accuracy for simple exponential smoothing, double exponential smoothing, linear regression, multiple linear regression, and polynomial regression. They found an overall reduction in forecast accuracy but did not provide the accuracy of each model individually. Also, top- and bottom-coding can be used to replace the tails of distributions with a threshold value, such as $150,000 for income or 10 kilowatts-hours for household energy usage). Top- and bottom-coding limit attribute disclosure risk (*i.e.*, preventing knowledge of specific values within a time series), but may not be effective at limiting identification disclosure risk (*i.e.*, preventing the identification of an entire time series). Top- and bottom-coding could have an effect similar to adjusting for outliers which improves forecast accuracy when the outliers are close to the forecast origin (Chen & Liu, 1993).

* 1. *Adjusted Forecasts*

Privacy methods adjust forecasts by altering the original time series data. Similar to judgmental adjustments, this presents the forecaster with multiple forecasts to choose from. We reference the long history on judgmental forecasting (Petropoulos et al., 2022, see sections 2.11.2 and 3.7.3) investigating how the direction and magnitude of adjustments, and the volatility of forecasts affect forecast accuracy.

There are two critical differences between privacy adjustments and judgmental adjustments.

First, judgmental adjustments alter a forecast after it is output from a forecasting model. The underlying time series and their features are not changed. For the direction of the adjustment, Davydenko & Fildes (2013) found that both positive and negative adjustments can improve accuracy, but positive adjustments tend to give only a marginal improvement. Khosrowabadi et al. (2022) similarly found that beneficial positive adjustments tended to be small, and beneficial negative adjustments tended to be large. Fildes et al. (2019) showed that negative adjustments reduce forecast bias, whereas positive adjustments maintain bias or exacerbate it. The magnitude of judgmental adjustments is also positively associated with the size of accuracy improvements when adjustments are based on reliable information. For the volatility of forecasts, accuracy improvements are more significant for time series (or forecasts?) with low volatility that are easier to forecast (Fildes et al., 2009).

Second, the motivation for judgmental adjustments is different. Motivations include gaining control of the forecasting process, incorporating practitioner expectations, and compensating for judgmental biases (Petropoulos et al., 2022, sec. 3.7.3). The goal is to incorporate the intuition and experience of the adjuster, knowledge of special events, or insider or confidential information to improve forecast accuracy (Fildes et al., 2019). Despite varying motivations, judgmental adjustments have been shown to improve forecast accuracy by 5-10% on average (Davydenko & Fildes, 2013; Khosrowabadi et al., 2022). For privacy adjustments, the goal is to improve privacy by blurring the data. The assumption is that forecast accuracy will not improve – instead, utility (forecast accuracy) will tradeoff with privacy (cite Duncan R-U curve XXX).

* 1. *Time Series Features for Forecast Accuracy*

Thousands of features have been used for time series classification (Fulcher & Jones, 2014) and a subset of those are useful for forecast accuracy. Bandara et al. (2018) clustered similar time series based on eighteen interpretable features, including the mean, variance, and strength of seasonality to improve the accuracy of RNNs between 2 and 11%. The initial results from the M4 competition suggested that the randomness and linearity of time series were the most important determinants of forecast accuracy and that seasonal time series (typically less noisy) are easier to forecast (Makridakis et al., 2018). In a follow-up study, Spiliotis et al. (2020) used multiple linear regression to confirm the importance of randomness, linearity, and seasonal strength in predicting mean absolute scaled error (MASE) values of the ETS, ARIMA, Theta, and Naïve 2 (random walk applied to seasonally adjusted data) models from the M4 competition. They found that increasing the frequency, kurtosis, linearity, and seasonal strength of time series improved forecast accuracy, but increasing skewness, self-similarity, and randomness degraded forecast accuracy.

Time series features are also used for model selection and forecast combination. Qi et al. (2022) found that forecasts using the strength of trend and seasonality for exponential smoothing model selection had lower errors across multiple forecast accuracy metrics than information-based selection methods for the majority of forecast horizons. Talagala et al. (2022) applied a meta-learning algorithm based on Bayesian multivariate surface regression to 37 features, including spectral entropy and the Hurst exponent, to predict the model combination that would yield the minimum forecast error for the M4 competition data. This approach achieved forecast accuracy on par with the top M4 competition methods with less computational cost. Li et al. (2022) used features such as the first ACF value to propose an interpretable Bayesian forecast combination framework with time-varying weights. In experiments using the M3 competition data, this method reduced the MASE by approximately 1.1% relative to the next-best forecast combination method. Petropoulos & Siemsen (2022) created a representativeness metric that selects models with trend and seasonality components when the respective signals of these components are strong. For most data frequencies, their approach lowered MASE on the M, M3, and M4 competition data and selected the best forecasting model approximately 3% more often than the other selection methods.

1. **The k-nearest Time Series (nTS) Swapping Method**

Let be a set of time series data (*n*-vectors). Using baseline privacy methods such as differential privacy, a data owner releases protected data point for each time series based on the original values at time and before. Baseline methods choose protected values based on predefined rules, not changes in forecast accuracy. The goal of the data owner should be to change to with minimal reductions in forecast accuracy while increasing privacy to an acceptable level.

We solve the data protection problem for the data owner using a matrix-based k-nTS (k-nearest time series) swapping method, where the data owner releases a set of protected time series where is based on , the original values of all series through time . To create a protected series , the *k*-nTS method finds the k most similar time series to where similarity is based on the time series features. For each period *t*, it randomly chooses one of the k similar series to and replaces with the original value from period *t* from the randomly chosen series.

Depending on the quantity of available data, *k*-nTS can use rolling windows of data that adjust for dynamic changes in time series features. For example, if we choose a rolling window of size *n*, then where . Protection in subsequent periods from to rolls forward from to , respectively.

We label the time series features for the current window as which we refer to as the feature vector for time series *j* in time period  
*t* based on the *n* values in . Li et al. (2022) also compute time series features over rolling, fixed-length windows. For simplicity, we omit the *t* subscript for the feature vectors and write .

For each time series , the data owner computes the feature vector . This vector can contain any single-valued feature calculated from the values in , such as the strength of the trend and seasonality, the spectral entropy, or the mean value of the current window. Let be the set of *m*-vectors containing the features from each of the *J* time series windows. For each , the data owner computes a set of squared distances of the elements of . We define as the distance between and , i.e., the feature vectors corresponding to two distinct time series from . Without loss of generality, we use the Euclidean norm, or ℓ2-norm, as a distance metric[[3]](#footnote-3). Since our case is multivariate and partially ordered, we can get a totally ordered set based on the Euclidean distance.

We define as the *k*th nearest neighbor of , with the corresponding feature vector . Then, for a time series , we have such that for any integers where . Note that and the superscript means the *i*th order statistic of the related Euclidean distances of all from . Thus, for a given time series vector , its *k*-nearest time series can be represented as the set based on an ordered set .

For more efficient computation, we introduce a symmetric distance matrix containing the squared distances between time series feature vectors. The squared distance between and is given by , that is the (*i, j*)th entry of (also note that ). Suppose we have a original data matrix , where (i.e., We calculate the desired features based on each and construct a feature matrix (where ) as follows:

A screenshot of a computer

Description automatically generated with medium confidence

**Algorithm 1: The k-nTS Swapping Method**

**Require [Initialization]:**

: the matrix of original time series.

: the matrix of time series features.

: the feature distance matrix.

**for do**

1. Find the set for by sorting the th column of from smallest to largest and finding the th smallest component.
2. Replace the last component of with the last component of for a randomly chosen

**end for**

We calculate the matrix using the fact that , which can be written as the following:

,

where **1** denotes a column vector of ones[[4]](#footnote-4). It is easy to see that the column vector Let denote the *j*th column of . Then we can write the distance matrix where In the general case where , for each time series we sort and take the *k* smallest components so that we have

.

That is, the data owner selects a value of *k* from 1 to a maximum of and selects the *k*-nearest time series to based on the *m* features. When , swapping is performed by rearranging the components in the last row of the matrix . Let the *i*th most similar time series to be where *n* is the length of the rolling window of past data. Swapping the last component of with the last component of one of its *k*-nearest time series , is: with probability 1/k for i=1,…k, which is equivalent to the last component of being replaced by the last component of with probability for

By Algorithm 1, we can obtain : a matrix of protected time series data through time *t* for all *J* time series for a rolling window size *n*. The *k*-nTS privacy method can be written as the following protected data matrix,

A screenshot of a computer

Description automatically generated

and the protection can be rolled forward each successive time period.

* 1. **The *k*-nearest Time Series + (nTS+) Swapping Method**

The k-nTS+ swapping method adds a feature selection process to k-nTS which selects features that are good predictors of forecast errors.. The goal is to obtain a small set of features that predict forecast accuracy well. k-nTS+ swapping can be used collaboratively between a data owner and the forecaster. The forecaster specifies their preferred forecasting model , and the data owner applies the model to the original and protected data up through time period *T – 1*, assesses which features are most predictive of accuracy for the specified model, and releases protected data to the forecaster using *k*-nTS+ based on these features up through time period . The data owner can repeat this process over successive time periods for multiple data releases at times and beyond.

For feature selection using RRelieff, Urbanowicz et. al. (2018) explains that there is no straightforward method for choosing the number of features to keep even if RReliefF is applied recursively. Including all of the features with large weights would significantly increase the dimensionality and reduce the efficiency of a swapping process. To address this problem for our k-nTS+ swapping method, a random forest-based recursive feature elimination (RFE) algorithm can be applied to the features selected by RReliefF. Prior work has shown that random forest-based RFE is efficient when applied to sets of highly correlated features (Gregorutti et al. 2017).

Algorithm 2 specifies the k-nTS Swapping Method below. It starts with the data owner generating forecasts for period for the original data and protected data using baseline privacy methods. The data owner measures the forecast errors and time series features for each original and protected time series. First, they use RReliefF (Robnik-Sikonja & Kononenko, 2003) to weight whether differences in a given feature are predictive of differences in forecast errors between nearest-neighbor time series. Next, similar to Li et al. (2022), the RReliefF algorithm (Robnik-Sikonja & Kononenko, 2003) is used to select the features which have a higher probability of varying across nearest-neighbor time series with different forecast errors than nearest neighbor time series with similar forecast errors.

The second stage of feature selection uses random forest-based RFE to recursively remove the feature that is least important for predicting forecast errors from the time series feature matrix. The RFE algorithm is repeated times. In each iteration of the RFE algorithm, a random forest is used to predict forecast errors using the current subset of time series features. The out-of-bag (OOB) mean-squared error (MSE) and permutation-based feature importance values are saved, the least important feature is removed, and the model is retrained for the next iteration. These steps repeat until one feature remains. The value is calculated using the averages of the OOB MSE across the repetitions for each . We include the features with the highest average ranks in the *k*-nTS+ swapping method.

**Algorithm 2: The k-nTS+ Swapping Method**

**Require [Initialization]**:

: the matrix of original time series.

: the baseline privacy methods.

: the desired forecasting model.

: the initial set of time series features’ names.

: the number of nearest neighbor time series to consider for swapping.

: the number of nearest neighbor time series to consider for RReliefF.

the number of recursive feature elimination iterations

: recursive feature elimination prediction error threshold

*Create Baseline Protected Datasets*

1. Store the data values from all series from time period in as a test set:
2. **for** each baseline privacy method :
   1. Create protected data set

**end for**

*Generate Baseline Forecasts and Extract Time Series Features*

1. Generate forecasts for time based on the original data
2. Compute forecast errors for time
3. Extract cross-sectional time series features matrix from
4. **for** each protected data set
   1. Generate forecasts for time based on the protected data
   2. Compute forecast errors
   3. Extract cross-sectional protected time series feature matrix

*First Stage Feature Selection: RReliefF*

1. Create cross-sectional feature matrix and forecast error vector by concatenating the feature matices and error vectors from the baseline protected and original data sets
2. Treat forecast errors as the target and time series features as the predictors. Generate weight for each feature RReliefF algorithm (Robnik-Sikonja & Kononenko, 2003) with nearest neighbor parameter .
3. Select features which contains the names of the features with RReliefF weights greater than zero.
4. Create cross-sectional feature matrix from such that contains the features for each original and baseline protected time series.

*Second Stage Feature Selection: Random Forest Based RFE*

1. for :
   1. Train a random forest to predict using .
   2. Calculate , the mean-squared error of the random forest out-of-bag forecast error predictions for iteration and number of features .
   3. Calculate the importance of each feature as the change in mean-squared error of the out-of-bag forecast error predictions after randomly permuting the feature in
   4. for subset size :
      1. Drop the feature with the lowest importance from such that features remain.
      2. Assign rank to for iteration .
      3. **if** 
         1. Repeat steps b. and c. from above.

**end for**

**end for**

1. **for** 
   1. Compute , the average mean-squared error of the out-of-bag forecast error predictions for features

**end for**

1. **for** 
   1. Compute , the average rank of each feature

**end for**

1. Identify the number of features with the minimum out-of-bag mean-squared error
2. **for** 
   1. Calculate , the percentage increase in the average out-of-bag mean squared error from using features in the random forest model

**end for**

1. Set to the smallest value of such that
2. Select the features with the best (lowest) average ranks

*k-nTS Swapping*

1. Use the selected features to perform swapping using **Algorithm 1:** The k-nTS Swapping Method.
2. **Empirical Application**
   1. *Data*

The organizers of the early M competitions did not disclose the true identity of the time series used in their competitions (Makridakis & Hibon, 2000). For our application, this provides a natural connection to privacy because we can compute the identification disclosure risk of each protected time series. We define identification disclosure risk as the probability of matching a protected time series to its original time series in the original data set. Good privacy implies the identification disclosure risk is low or similar to random guessing. To be conservative, we assume that an adversary (possibly a forecaster) has external data of at least one original time series and attempts to match each protected time series to its original time series. The data owner seeks to alter the time series with privacy methods to reduce the identification disclosure risk while maintaining as much forecast accuracy as possible.

Recent work by Spiliotis et al. (2020) shows that the M3 competition data contain time series features representative of the real world data which also makes it suitable for our feature-based k-nTS+ swapping method. As a result, we use the monthly micro dataset from the M3 competition, which includes 474 strictly positive time series with values ranging from 120 to 18,100. Of the 474 series, 18 consist of 67 time periods, 259 consist of 68 time periods, and 197 consist of 125 time periods.

All data privacy methods are run by the data owner who protects every single time series value from time period 1 to T. The protected time series are given to the forecaster to produce one-step ahead forecasts for time T + 1. The data owner then measures the accuracy of those forecasts (and the forecasts using the original data) against the actual values from T + 1.

We assume that the forecaster may also be an adversary attempting to identify an original time series by using the protected time series. For calculating identification disclosure risk (see subsection 4.3.4 for further details), we take the most conservative approach and assume that the adversary possesses time series of equal lengths to each protected series. Thus, we separate the privacy analysis and protection into three groups of 18, 259, and 197 time series. If an adversary were to attempt to match an original time series to a protected time series, random guessing becomes for the 18 series, for the 259 series, and for the 197 series. Random guessing for identification disclosure risk becomes approximately 0.6% instead of when all time series lengths are equal.

The rest of the empirical application is outlined as follows. Subsection 4.2 defines the time series features used for the k-nTS+ swapping method, subsection 4.3 describes the privacy methods and defines the identification disclosure results, subsection 4.4 presents the privacy and forecast accuracy results, and subsection 4.5 analyses how the time series features change after privacy protection and whether the magnitude, direction, or volatility of privacy adjustments were helpful in maintaining forecast accuracy.

* 1. *Time Series Features for Forecast Accuracy*

|  |  |  |
| --- | --- | --- |
| **Feature** | **Description** | **Value Range** |
| *Spectral Entropy* |  | [0, 1] |
| *Hurst* |  | [0, 1] |
| *Skewness* |  |  |
| *Kurtosis* |  |  |
| *Error ACF* |  | [-1, 1] |
| *Trend* |  | [0, 1] |
| *Seasonality* |  | [0, 1] |
| *Mean* |  |  |
| *Variance* |  |  |
| *Spike* |  |  |
| *Max Variance Shift* |  |  |
| *Max Level Shift* |  |  |

Based on the literature review in Section 2.3, we define the time series features that had a relationship with forecast accuracy and add additional features including skewness, kurtosis, linearity, and strength of trend and seasonality. We omit stability and non-linearity since these features had little to no effect on accuracy. We also omit frequency because none of the privacy methods we consider decrease the frequency of the data. However, we note that higher frequencies are associated with improved forecast accuracy (Spiliotis et al., 2020).

* + 1. *Spectral Entropy*

Suppose is a univariate stationary time series with a finite mean and constant variance. The spectral density of is estimated as the scaled Fourier transform of the autocovariance function of . The spectral density can be thought of as the probability density function of a random variable on the unit circle (Goerg, 2013), where for a non-zero integer , when , the spectral density will have a peak at the corresponding frequency . The forecastability, or spectral entropy, of is measured using the Shannon entropy of , given by

,

where the maximum entropy occurs when . In practice, estimates of , where high values represent a low signal-to-noise ratio, indicating that is difficult to forecast (Kang et al., 2017).

* + 1. *Hurst*

Next, we consider a self-similarity feature quantified using the Hurst parameter (Wang et al., 2006), which measures the long-range dependence of a time series. Spiliotis et al. (2020) found this feature had the largest effect on forecast accuracy. We use the definition of self-similarity of a time series described by (Willinger et al., 1998). Suppose that is the increment process of , *i.e.*, . An aggregate sequence, denoted , is created by averaging over non-overlapping blocks of size , where

and indexes the block. If is a self-similar time series, then

for all integers . We use the definition of second-order self-similarity, where s exactly second-order self-similar if has the same variance and autocorrelation as for all values of , or is asymptotically second-order self-similar if this holds as (Rose, 1996). The parameter is the Hurst exponent, which is estimated using the differencing term from a fractional ARIMA model, i.e., FARIMA(0, , 0) (Wang et al., 2006; Hyndman et al., 2022), where

5.

Estimates of range from 0 to 1, where corresponds to a random walk (Sobolev, 2017), corresponds to anti-persistent or mean-reverting series, and corresponds to persistent time series that are more likely to maintain their current trend.

* + 1. *Skewness*

*Skewness* measures the lack of symmetry in the distribution of the values of (Wang et al., 2006), where positive (or negative) values are associated with a right- (or left-) skewed data distribution,

* + 1. *Kurtosis*

We measure *Kurtosis* relative to the standard normal distribution (Wang et al., 2006). Positive *Kurtosis* corresponds to distributions that tend to have a distinct peak near the mean with heavy tails, whereas negative *Kurtosis* corresponds to distributions that are relatively flat near the mean,

where 3 is the *Kurtosis* of the standard normal distribution.

* + 1. *Error Autocorrelation Function (E\_acf)*

Next, we perform STL decomposition (Cleveland et al. 1990) to obtain the trend, seasonal, and remainder components of . We use the approach of Hyndman et al. (2019) to obtain

,

where , , and are the trend, seasonal, and remainder components, respectively.

We extract the first-order autocorrelation coefficient of the detrended and deseasonalized series, referred to as 'linearity' by Spiliotis et al. (2018):

*Error ACF* .

*Error ACF* is a measure of the predictability of a time series after the trend and seasonality have been accounted for (Kang et al. 2017).

* + 1. *Trend and Seasonality*

We also compute the strength of trend (*Trend*) and strength of the seasonal component () as follows,

,

and

.

In practice, the values of and range from 0 to 1 (Hyndman 2022).

* + 1. *Mean and Variance*

The next two features are the *Mean* and *Variance*, also used by Bandara et al. (2018) to cluster similar time series for forecasting, which are written as follows,

.

* + 1. *Spike, Maximum Variance Shift, Maximum Level Shift, and Other Features*

Finally, we include many features from the *tsfeatures* package in R including *Spike* (variance of the leave-one-out variances of the remainder component of the decomposed series), *Max Variance Shift* (largest variance shift between two consecutive sliding windows), and *Max Level Shift* (largest mean shift between two consecutive sliding windows). We refer the reader to (Hyndman et al., 2022) for a detailed explanation of these features.

* 1. *Privacy Protection*
     1. *Differential Privacy*

A mechanism satisfies -differential privacy by guaranteeing that, for every output of and every pair of series and ,

.

A differentially private time series can be created using a randomized mechanism

that adds a vector of random noise values, each of which is drawn from a Laplace distribution with scale parameter , to an original time series . The sensitivity is determined as the maximum absolute difference between two time series and which differ in at most one observation, where . We follow Gonçalves et al. (2021a) for our implementation and set which ranges from least private to most private.

* + 1. *Additive Noise*

Additive noise adds a normally distributed random number with mean zero and standard deviation to each value in an original time series . Protected values can be written , where and . The protection parameter denotes the number of standard deviations of that define the standard deviation of the sampling distribution of . We set which ranges from least private to most private.

* + 1. *k-nTS and k-nTS+*

The k-nearest series for k-nTS and k-nTS+ swapping are determined using features computed by the data owner from the original data set. The data owner swaps original values to create the protected values from time 1 to T for each time series. We use the features described in Section 4.2 as well as others available in R package *tsfeatures*.

To perform feature selection for *k*-nTS+, the data owner first creates protected versions of the original data using additive noise and differential privacy from time period 1 to . Then, using this protected data from time period 1 to , the data owner generates forecasts for each of the protected data sets at time and computes the absolute error of each forecast for each series. The k-nTS+ swapping method is applied to each forecasting model separately in order to detect the variation in forecast accuracy due to changes in time series features and not the forecasting.

We select so the features are within 5% of the minimum average prediction error. For the k-nTS+ protected data, we use the six features with the highest average rank across the RFE iterations for all forecasting models with . Next, the data owner uses these six features selected by this feedback to swap all series for time periods 1 to T. The data owner shares this protected data with forecasters who forecast time period .

* + 1. *Identification disclosure risk*

As previously mentioned, the forecasters of the M3 competition did not have the identities of the original time series. For our privacy metric, we assess the ability of each privacy method to protect against *identification disclosure*, which occurs when an adversary correctly predicts the identity of a protected time series. Each protected dataset consists of the protected series along with a pseudo identifier, *i.e.*, . The pseudo identifier in our application is the `Series` column from the original M3 data, which contains a PID for each time series, *e.g.*, `N1402`. Identification disclosure occurs if an adversary (or forecaster) correctly predicted the identity of one or more of the time series in the M3 data set based on the protected time series and some outside information the adversary possesses. For example, identification disclosure occurs when an adversary correctly states, "Series N1402 comes from the monthly sales of the Roseville, Minnesota Target store.”

We perform simulations of a privacy attack in which an adversary uses original time series values to identify the protected time series. In each simulation, we sample ten values from each original time series and treat these as external information available to the adversary. The adversary predicts the identity of each protected series based on which original values are closest to the protected values from the same time periods.

The metric we use is identification disclosure risk for time series (Nin & Torra, 2006, 2009), , the average proportion of the time series which are correctly identified across the simulated privacy attacks,

where is the adversary’s prediction of the identity of the th protected time series, and identification disclosure occurs when the predicted identity is equal to the true identity . We refer the reader to the Appendix for added mathematical details.

* 1. *Results*

For all privacy methods, we generate one-step ahead forecasts for time T+1 using off-the-shelf models in R and Python. Similar to the M3 Competition, all reported forecast accuracy and standard deviation results are derived from comparing the forecasts for T + 1 to the actual data from T + 1. Reported privacy results are derived from calculating the identification disclosure risk using the protected data from time period 1 to T. Also, the global forecasting models and the VAR model use the three subsets of 18, 259, and 197 time series separately.

We perform minimal data pre-processing and use the standard settings in the off-the-shelf packages. Implementation details can be found in the appendix.

**Table 2: Univariate and Multivariate Forecast Models**

|  |  |  |
| --- | --- | --- |
|  | **Model Name** | **Variant** |
| Local Models | SES | - |
| DES | Additive trend |
| TES | Additive trend/seasonality |
| Auto-ARIMA | Seasonal |
|  | VAR | - |
| Global Models | LGBM | - |
| RNN | LSTM |

Forecasters also prefer protected data that are representative of the original time series. Representativeness improves trust between data owners and forecasters and makes it more likely for forecasters to use protected data. We apply the *performance gap* of Petropoulos & Siemsen (2022) to measure the distance between the protected and original time series values,

Performance gap = ,

which is calculated after applying a Box-Cox transformation and scaling the original and protected series. Note that this differs from Petropoulos & Siemsen (2022) where the performance gap is calculated using the fitted values of forecasting models relative to the training data (which we include in the first row of Table 3).

Table 3displays the average MAE of one-step ahead point forecasts across all models and series, the identification disclosure metric , and the average performance gap across all series. The percentages in parentheses are the increase in average MAE relative to the average MAE from the original data. The results show an inverse relationship between forecast accuracy and the strength of privacy protection. While strong differential privacy provides the lowest identification disclosure risk at 1.85% (random guessing is 0.6%), it nearly quintuples (+383%) the average forecast error relative to the original data resulting in unusable forecasts. Under weak differential privacy with , over 49% of series are identified correctly on average, which is poor identification disclosure risk.

Protection against identification disclosure is better under additive noise with where 22.5% of series are correctly identified on average. However, this comes at a cost to forecast accuracy, which degrades by nearly 45%.[[5]](#footnote-5)

Standard *k*-nTS with *k* = 3 offers a good identification disclosure risk of 2.1%, but forecast accuracy degrades by 39.6%. Our proposed method of *k-*nTS+ swapping with provides similar levels of protection against reidentification (3.3%) with a reduction in forecast accuracy of only 13.9%. Part of this improvement in forecast accuracy at a minimal tradeoff to identification disclosure risk is due to the incorporation of the accuracy feedback loop for time series features. Thus, we recommend data owners to use our k-nTS+ swapping method (k=3) with the relevant time series features to balance the tradeoff between privacy and forecast accuracy. The results in Table 3 also show that k-nts+ and *k*-nTS produce protected time series with the smallest performance gaps by a large margin. However, we note that these performance gap values (62.1 for k-nTS+ with k=3) are significantly larger than the average performance gap (42.7) of the fitted values from all forecasting models.

**Table 3: Identification disclosure risk, forecast accuracy, and representativeness for original and protected data sets.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Privacy Method** | **Parameter Value** | Privacy (Identification Disclosure Risk) | Accuracy (MAE) | Representativeness (Performance Gap) |
| **Original Data** | **-** | 100.0% | 685.71 (0.0%) | 42.7 |
| ***k*-nTS+** | **15** | 2.7% | 839.8  (+22.5%) | 73.0 |
| **7** | 3.5% | 822.3  (+19.9%) | 66.5 |
| **3** | 3.3% | 781.0  (+13.9%) | 62.1 |
| ***k*-nTS** | **15** | 1.6% | 1066.2  (+55.5%) | 106.3 |
| **7** | 2.1% | 987.0  (+43.9%) | 100.4 |
| **3** | 2.1% | 956.9  (+39.6%) | 92.9 |
| **Differential Privacy** | **1.0** | 1.9% | 3310.3  (+382.8%) | 1,826,437.0 |
| **4.6** | 13.6% | 1401.0  (+104.3%) | 311,037.7 |
| **10** | 49.0% | 899.4  (+31.2%) | 78,456.4 |
| **Additive Noise** | **2.0** | 5.8% | 1821.4  (+165.6%) | 503,658.2 |
| **1.5** | 10.4% | 1343.3  (+95.9%) | 326,834.5 |
| **1.0** | 22.5% | 994.0  (+45.0%) | 166,171.2 |

Past research on the M3 competition also found that complex forecasting models forecast more accurately than simple models using the monthly micro data (Koning et al., 2005). Table 4 displaysthe ranks of the MAE and forecast error variance across all forecasting models using the original data and k-nTS+ swapping with ­. The results show that k-nTS+ preserves the ranking of the best and worst models on MAE for the monthly micro data in the M3 Competition. Univariate models (SES and DES) moved up in the ranking and more complex models (Auto-ARIMA and RNN) moved down.

**Table 4: Ranks of MAE and standard deviation of forecast error for the original data and the k-nTS+ swapping (*k*=3) data.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | MAE Ranks | | Standard Deviation of Forecast Error Ranks | |
| Model | Original | Protected | Original | Protected |
| TES | 1 (637.90) | 1 (731.30) | 2 (859.30) | 4 (920.57) |
| Auto-ARIMA | 2 (646.07) | 4 (764.83) | 1 (834.78) | 1 (897.67) |
| RNN | 3 (665.38) | 5 (783.15) | 5 (883.86) | 5 (966.35) |
| DES | 4 (680.54) | 2 (743.68) | 3 (866.35) | 2 (901.22) |
| SES | 5 (686.71) | 3 (752.08) | 4 (867.13) | 3 (914.20) |
| LGBM | 6 (709.48) | 6 (809.00) | 7 (919.67) | 6 (982.35) |
| VAR | 7 (773.90) | 7 (883.07) | 6 (892.62) | 7 (998.08) |

* 1. *Analysis of Time Series Features*

In this subsection, we investigate how the time series features affected the tradeoff between privacy and forecast accuracy for k-nTS+.

* + 1. *Importance of Time Series Features*

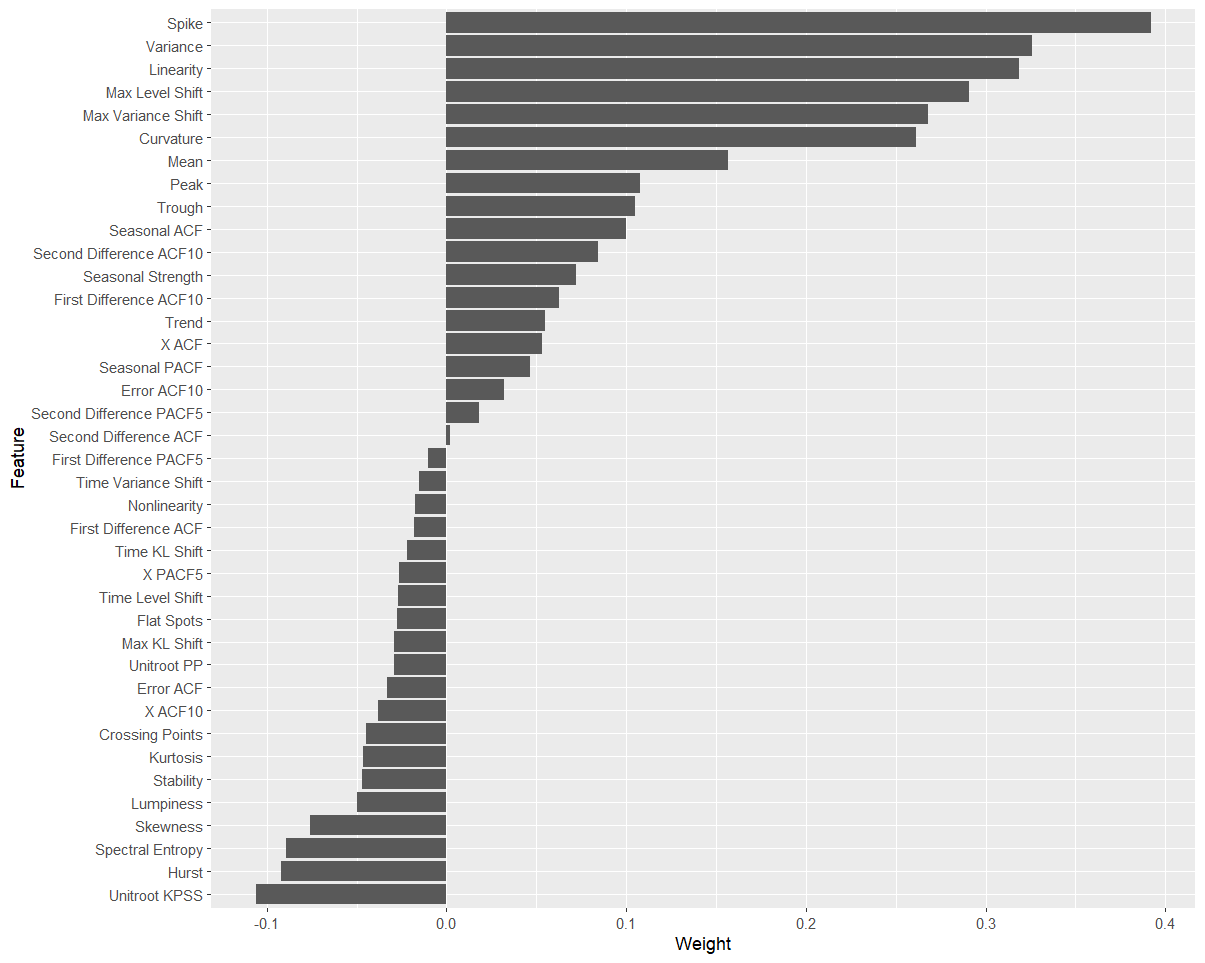
The RReliefF weights approximate the difference between the probability that feature discriminates between series with different forecast errors, and the probability that feature discriminates between series with the same forecast error. If we consider feature for k-nTS+ swapping since maintaining the value of after data protection is likely to limit changes in forecast accuracy.

For the interpretation of feature weights, let denote one of the nearest neighbor feature vectors to , where is the number of nearest neighbors considered by RReliefF, and let and denote the forecast errors for the corresponding time series. Let and denote the events that series and have different forecast errors and different values for feature , respectively, conditional on being nearest neighbors. The RReliefF weight for feature approximates the following difference in probabilities:

.

Figure 4 shows the RReliefF weights for each of the 39 features averaged across all forecasting models. RReliefF was used to predict the absolute forecast errors for each model and series across the original and protected data sets. Surprisingly, *Hurst* and *Spectral Entropy* had negative weights which implied they were not useful to improve forecast accuracy for swapping in the protected data. On the other hand, *Spike*, *Variance*, *Error ACF*, *Max Level Shift*, and *Max Variance Shift* had large positive weights and were very important to improve forecast accuracy.

**Figure 4: RReliefF weights averaged across the results for each forecasting model.**

****

* + 1. *Selection of Time Series Features*

Figure 5 presents the features included for each forecasting model after k-nTS+ eliminated features with negative weights that were poor predictors of forecast error. Over iterations, most of the reduction in OOB MSE occurred using five or fewer features for all forecasting models.

**Fig. 5: Average OOB MSE across feature subset sizes when predicting the MAE of each forecasting model.**

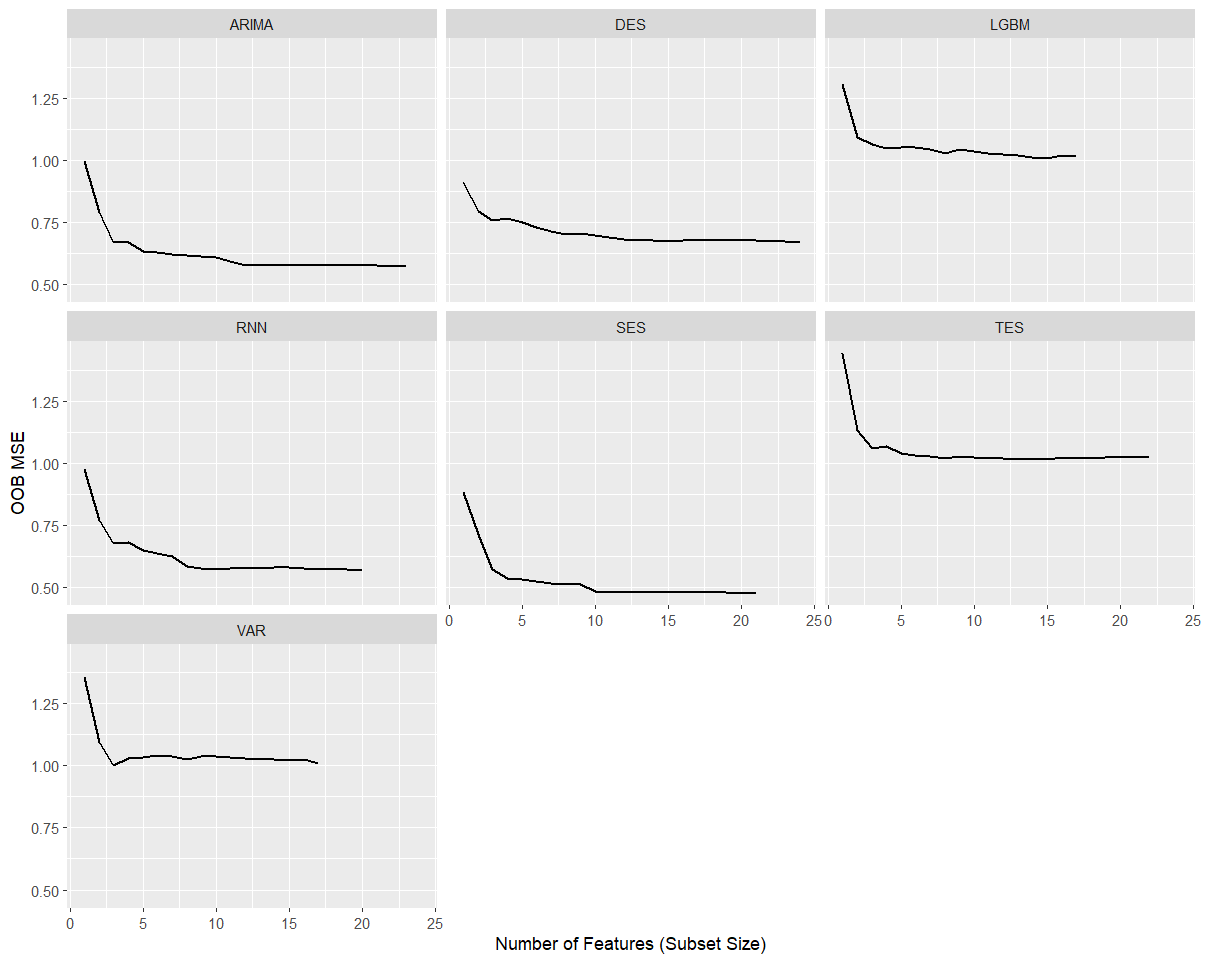


Figure 6 summarizes the results of RFE in Figure 5 and displays the permutation-based importance values for each forecasting model's six most highly ranked features. Some features, such as *spike,* *max variance shift, max level shift, mean,* and *variance,* are highly ranked across most or all forecasting models. Other features appear to be highly important only for specific forecasting models. Examples include *trend*, which is required for DES and TES, *seasonal strength*, which is required for TES, and *X ACF,* which is important for Auto-ARIMA and RNN.

**Figure 6: Permutation-based importance for the top six features for each forecasting model.**

Chart

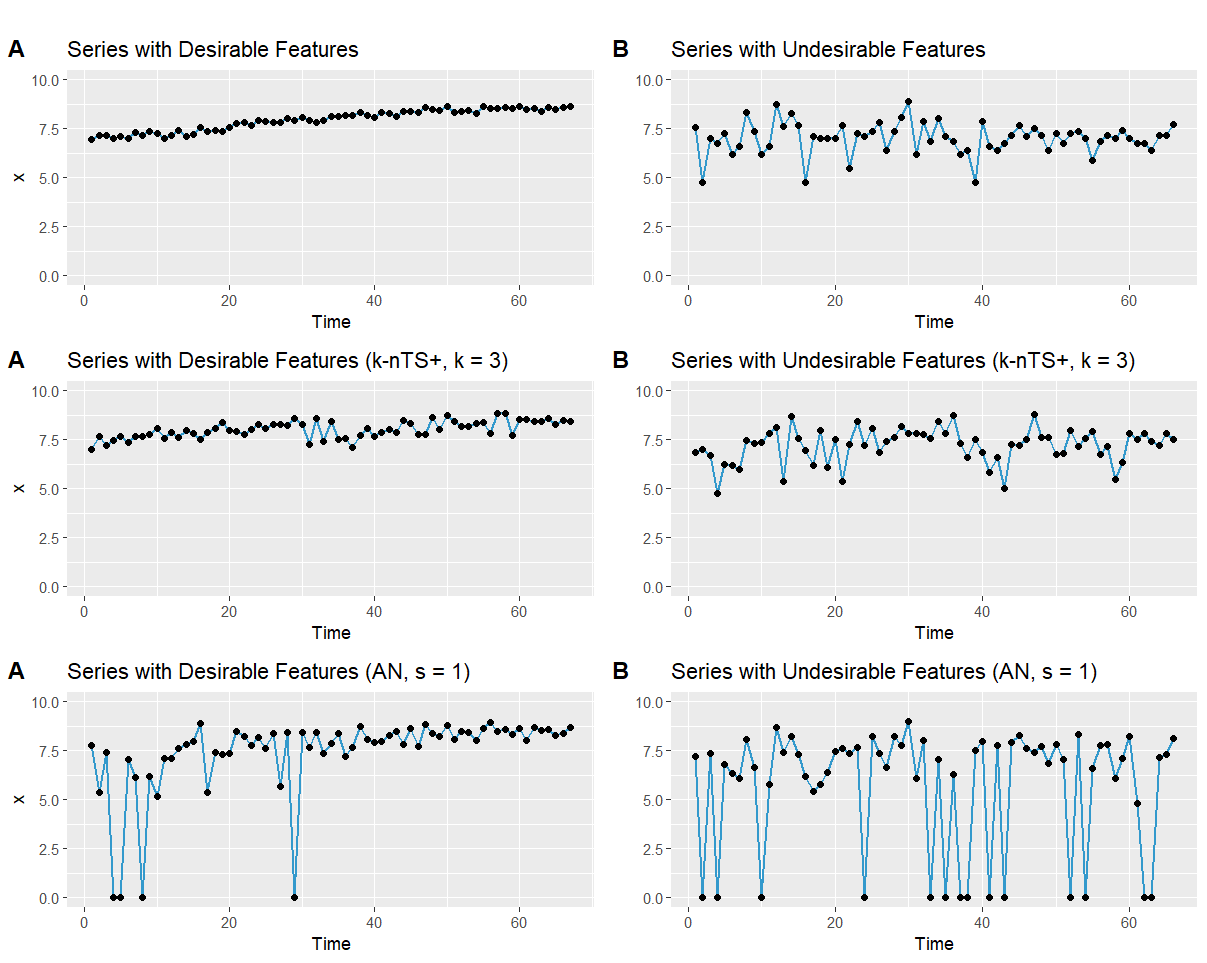
Description automatically generated

* + 1. *Illustration of Changes in Time Series Features After Protection*

Figure 7 displays two monthly time series from the M3 monthly micro data with desirable and undesirable features. After applying the privacy methods to the original time series, Figure 7 illustrates the results using k-nTS+ with and additive noise with . We can see that for k-nTS+ with k=3, there is little visual change for the undesirable series. For additive noise, there are drastic changes to both series.

Table 5 displays the values of the time series features before and after protection. Table 5 shows that the low spectral entropy and high Hurst coefficient values of the desirable time series indicate good forecastability. Table 5 shows that the undesirable series is essentially a random walk as indicated by the 0.50 value of the Hurst coefficient. Furthermore, the undesirable series has a spectral entropy of 1 indicating a low signal-to-noise ratio. When comparing the two series, the variance of the desirable series is due to a forecastable trend, whereas the variance of the undesirable series is due to randomness. The desirable series also has low *Kurtosis* with a light tailed distribution compared to the undesirable series.

**Figure 7: Comparison of original, AN (s = 1), and *k*-nTS+ (*k* = 3) protected series with desirable and undesirable features.**

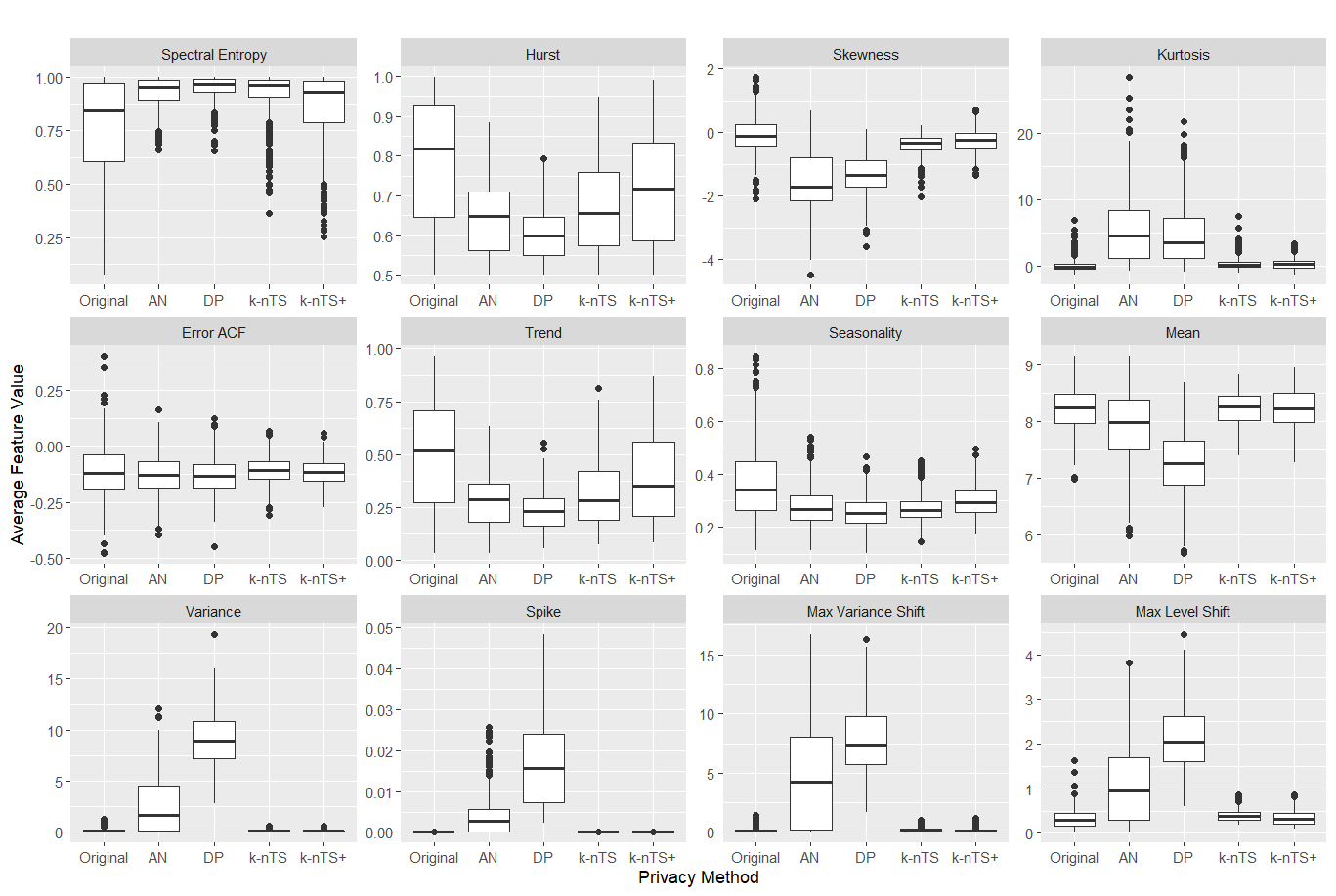


**Table 5: Time series feature values from undesirable and desirable time series.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feature** | **Desirable Time Series (left Fig. 7)** | | | **Undesirable Time Series (right Fig. 7)** | | |
|  | **Original** | **k-nTS+ (k=3)** | **AN (s=1)** | **Original** | **k-nTS+ (k=3)** | **AN (s=1)** |
| *Spectral Entropy* | 0.07 | 0.89 | 0.92 | 1.00 | 0.98 | 1.00 |
| *Hurst* | 0.99 | 0.81 | 0.76 | 0.50 | 0.66 | 0.50 |
| *Skewness* | -0.41 | -0.18 | -2.74 | -0.57 | -0.71 | -1.17 |
| *Kurtosis* | -1.24 | -0.74 | 6.99 | 1.16 | 0.25 | -0.37 |
| *Error ACF* | -0.09 | -0.22 | -0.20 | -0.19 | -0.06 | -0.21 |
| *Trend* | 0.97 | 0.58 | 0.49 | 0.12 | 0.22 | 0.11 |
| *Seasonality* | 0.16 | 0.25 | 0.39 | 0.23 | 0.24 | 0.13 |
| *Mean* | 7.96 | 8.02 | 7.41 | 7.01 | 7.21 | 5.73 |
| *Variance* | 0.29 | 0.19 | 4.27 | 0.65 | 0.76 | 9.57 |
| *Spike* | 0.31e-07 | 0.22e-7 | 0.37e-2 | 0.14e-3 | 0.12e-3 | 0.27e-1 |
| *Max Variance Shift* | 0.05 | 0.24 | 9.37 | 1.10 | 1.12 | 11.44 |
| *Max Level Shift* | 0.57 | 0.51 | 2.77 | 0.70 | 0.84 | 3.29 |

Figure 8 displays boxplots of the time series feature values before and after protection across all time series in our application. Random noise protection (AN and DP) methods increase the randomness and significantly change distributional characteristics of all features except E\_acf, leading to poor forecast accuracy. Random noise also produces a negative bias in the means of the protected series and significantly increases the variance. On the other hand, the *k*-nTS swapping method increases the spectral entropy but better preserves most feature distributions. The feature distributions of *k*-nTS+ swapping are much closer to the original distributions for those features important for forecast accuracy (list here XXX), which led to improved forecast accuracy results.

**Fig 8: Distributions of time series features for each privacy method.**



We note that while the base *k*-nTS method performed swapping based on the values of *Spectral Entropy, Hurst*, and *Seasonality*, it does not perform as well as *k*-nTS+ at preserving the distributions of these features, leading to poorer forecast accuracy. Importantly, all three of these features are based on autocorrelation, and the k-nts swapping ruins the autocorrelation. K-nts+…

*k*-nTS+ did not explicitly swap based on the values of these features which demonstrates the value addition of the k-nTS+ feedback loop. Although *Spectral Entropy* and *Hurst* were correlated with forecast accuracy across series, they were eliminated in the first stage of the *k*-nTS+ feature selection process using RReliefF. This is because k-nTS+ discriminated between nearest time series with different forecast errors. As a result, k-nTS+ achieved better feature preservation by swapping using the time series features which are correlated with overall measures of forecastability. We found that by using the k-nTS+ feature-based method, we could reduce the degradation in forecast accuracy from +39.6% to +13.9%, at a minimal tradeoff to privacy.

When applied to the monthly micro M3 data, our results show that the accuracy of the forecasting models we use are sensitive to changes in the shape (*e.g.*, spike) and location (*e.g.*, mean) of time series. To explain, if we have two similar time series (nearest time series), there is a higher likelihood that they differ on these features when their forecast errors are different than when their errors are the same. On the other hand, there is a higher probability that these two nearest neighbor series have different spectral entropies when their forecast errors are the same than when they are different. When privacy protection changes the spectral entropy and Hurst coefficient of a time series, the effect on accuracy is likely small because the important location and shape features are maintained. A simple example would be a time series with small variance and constant mean. When we randomly shuffle the time series values, we have the same variance and mean, but with little to no autocorrelation (high spectral entropy). When a forecasting model just forecasts the long run mean of this series, it achieves similar accuracy to the original series.

* + 1. *Privacy Adjusted Forecasts*

Similar to Fildes et. al. (2009), we compare the percentage of forecast adjustments that improved accuracy across adjustment direction, magnitude, and the coefficient of variation. We use the adjusted forecasts using the k-nTS+ (k = 3) protected data set which was the top performing data privacy method.

To compute adjustment magnitude, we normalize the absolute difference between the adjusted and original forecasts using the mean of the original series,

where the and superscripts denote a forecast based on the original and protected data, respectively. The magnitudes are binned into “high” (*≥* quantile), “low” (*≤* quantile) and “medium” (*≥ 0.25* quantile and quantile) (Khosrowabadi et al. 2022).

The average relative absolute error (AvgRelAE) of the adjusted forecasts is computed as

,

where and are the absolute forecast error for the protected and original versions of series An less than one indicates an average improvement in accuracy and an greater than one indicates an average reduction in accuracy.[[6]](#footnote-6) We implement a five percent trim based on the values of (Fildes et al., 2013).

Using the k-nTS+ (k=3) protected data, we find that less than half (43%) of the adjusted forecasts improved forecast accuracy, which is less than the reported 49.9% of judgmentally adjusted forecasts that improved accuracy in Khosrowabadi et al. (2022). Table 6 breaks down the results by adjustment magnitude and direction and displays the AvgRelAE and percentage of adjusted forecasts that improved accuracy. The results show that most privacy adjusted forecasts degraded accuracy since the AvgRelAE is greater than one in five out of six cases. Also, our results are contrary to the findings in the judgmental literature which shows that large adjustments and negative adjustments improve forecast accuracy. We find that small privacy adjustments improved (47.9% of cases) forecast accuracy more frequently than large privacy adjustments (35.6% of cases). Furthermore, positive adjustments improved (44.6% of cases) forecast accuracy more than negative adjustments (40.2% of cases). However, none of these cases improved forecast accuracy overall.

**Table 6: (and percentage of adjustments that improved accuracy) by magnitude and direction.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Direction | |  |
|  |  | Positive | Negative | **Total** |
| Magnitude | Large | 1.35 (40.5%) | 1.47 (30.4%) | **1.41 (35.6%)** |
| Medium | 1.12 (44.4%) | 1.17 (41.9%) | **1.14 (43.2%)** |
| Small | 0.99 (49.1%) | 1.06 (46.8%) | **1.03 (47.9%)** |
|  | **Total** | **1.14 (44.6%)** | **1.21 (40.2%)** | **1.17 (42.5%)** |

One issue with our data is that 73% of the series have negative slopes, which could cause positive adjustments to have a dampening effect on forecasts, and negative adjustments to overestimate the impact of the trend (Hyndman & Athanasopoulos, 2021). Table 7 displays the *AvgRelAE* and the percentage of adjustments for time series with positive slopes vs. negative slopes. To measure the slope, we calculate slope coefficient of a simple linear regression that regresses the time series values on a continuous time variable. Table 7 shows that time series with negative slopes and negative adjustments (32% of all time series were in this category) tended to degrade forecast accuracy the most.

**Table 7: (and the percentage of adjustments that improved accuracy) by slope and direction.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Direction | |  |
|  |  | Positive | Negative | **Total** |
| Slope | Positive | 1.13 (42.9%) | 1.14 (41.6%) | **1.14 (42.2%)** |
| Negative | 1.14 (45.0%) | 1.24 (39.6%) | **1.18 (42.6%)** |
|  | **Total** | **1.14 (44.6%)** | **1.21 (40.2%)** | **1.17 (42.5%)** |

Table 8 measures the (and percentage of adjustments that improved accuracy) categorized by the coefficient of variation of the original series and whether k-nTS+ (k=3) swapping increased, decreased, or maintained (within five percent) the coefficient of variation. We apply the same binning process as Fildes et. al. (2009) to the coefficient of variation but measure the coefficient of variation using the original time series values since we only analyzed one forecast horizon. We find that none of the coefficient of variation categories improve forecast accuracy compared to the original data. However, forecast accuracy degraded the most when k-nTS+ (k=3) was applied to time series with small coefficients of variation, which is the opposite finding of when judgmental adjustments were applied to low volatility data in Files et. al. (2009).

**Table 8: AvgRelAE (and percentage of adjustments that improved accuracy) by coefficient of variation of the original series and the change in coefficient of variation in the protected series.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Change in Coefficient of Variation | | |  |
|  |  | Decreased | Maintained (+/- 5%) | Increased | Total |
| Original Coefficient of Variation | Large | 1.06 (49.1%) | 1.14 (40.5%) | 1.30 (44.7%) | **1.09 (46.2%)** |
| Medium | 1.18 (42.5%) | 1.19 (40.3%) | 1.12 (49.1%) | **1.16 (44.1%)** |
| Small | 1.15 (38.2%) | 1.36 (31.6%) | 1.27 (36.1%) | **1.27 (35.7%)** |
|  | **Total** | **1.13 (45.0%)** | **1.19 (39.4%)** | **1.21 (41.7%)** | **1.17 (42.5%)** |
|  |  | Proportion of Time Series | | |  |
|  |  | Decreased | Maintained (+/- 5%) | Increased | Total |
| Original Coefficient of Variation | Large | 15.9% | 7.8% | 1.3% | **25.0%** |
| Medium | 22.7% | 11.7% | 15.6% | **50.0%** |
| Small | 1.1% | 2.5% | 21.4% | **25.0%** |
|  | **Total** | **39.7%** | **22.0%** | **38.3%** | **100.0%** |

Overall, our empirical results show that privacy adjustments affect forecast accuracy oppositely than judgmental adjustments. Specifically, we found that privacy adjustments had better forecast accuracy when the adjustments were small or positive, or when the coefficient of variation of the original series was large. However, on average, forecast accuracy worsened for nearly every combination of magnitude, direction, and coefficient of variation. This is not surprising since one motivation of judgmental adjustments is to improve forecast accuracy (Fildes et al., 2019) and judgmental adjustments have been shown to improve forecast accuracy by 5-10% on average (Davydenko & Fildes, 2013; Khosrowabadi et al., 2022). For our application, privacy adjustments blur the data for privacy reasons and therefore reduce forecast accuracy. The secondary goal of our proposed privacy method is to maintain forecast accuracy, which the top performing method (k-nTS+ (k=3) swapping) did with only a +13.9% average degradation. Baseline privacy methods like differential privacy and additive noise infuse a huge amount of noise leading to much larger increases in coefficients of variation and severe degradations in forecast accuracy.

1. **Conclusions**

This paper examined the impact of data privacy on forecast accuracy in a centralized scenario where a data owner shares a protected data set with forecasters. Our proposed *k*-nTS+ privacy method used time series features to swap the values between time series to mitigate the loss in forecast accuracy due to privacy protection. We demonstrated the effectiveness of our protection approach using data from a well-known forecasting competition where the identities of the time series needed to be kept confidential. The proposed method limited the average reduction in forecast accuracy to +13.9% of the original forecast accuracy. Nearly all other privacy methods we studied degraded forecast accuracy to unusable levels (over 100%) at similar levels of privacy.

To the best of our knowledge, this paper is the first to create a protected time series data set tailored to maintain forecast accuracy. Our method enabled data owners to share time series data sets instead of only parameter estimates or forecasts. The protected data also preserved important features for forecasting such as spike, max variance shift, max level shift, mean, variance, strength of trend, and strength of seasonality. Furthermore, we showed that our k-nTS+ protected data was more representative of the original time series, potentially leading to increased trust and adoptability between organizations.

A substantial portion of the privacy literature is focused on theoretical privacy guarantees such as differential privacy. Our findings agree with past research (Goncalves et al. 2021a) and show that differential privacy (and additive noise) generates unusable forecasts at reasonable levels of privacy. This undesirable privacy-utility tradeoff has also been demonstrated in contexts other than forecasting. A recent paper by Blanco-Justicia et al. (2022) found that much of the work on differential privacy and deep learning utilized relaxed versions of differential privacy with large values of that theoretically do not provide meaningful levels of privacy protection. Their experiments found that model regularization (e.g., L2-regularization) provided comparable privacy protection with better accuracy and lower model learning cost than differential privacy. In our application, we also found that our k-nTS+ swapping method had better forecast accuracy at comparable levels of identification disclosure risk with differential privacy.

Although we showed that *k*-nTS+ swapping balanced the tradeoff between forecast accuracy and privacy well, future work could also examine the utility of use cases beyond forecasting. Since many of the time series features were preserved and the entire protected data set was shared, other time series use cases requiring privacy may maintain utility. A major limitation of our study was that we did not consider privacy metrics other than identification disclosure risk, such as attribute disclosure risk. Finally, we did not address whether combinations of forecasts using protected data improve forecast accuracy.

**References**

Bandara, K., Bergmeir, C., & Smyl, S. (2018). Forecasting Across Time Series Databases using Recurrent Neural Networks on Groups of Similar Series: A Clustering Approach. ArXiv:1710.03222 [Cs, Econ, Stat]. http://arxiv.org/abs/1710.03222

Blanco-Justicia, A., Sanchez, D., Domingo-Ferrer, J., & Muralidhar, K. (2022). A Critical Review on the Use (and Misuse) of Differential Privacy in Machine Learning. arXiv preprint arXiv:2206.04621.

Boone, T., Ganeshan, R., Jain, A., & Sanders, N. R. (2019). Forecasting sales in the supply chain: Consumer analytics in the big data era. *International Journal of Forecasting*, *35*(1), 170–180. https://doi.org/10.1016/j.ijforecast.2018.09.003

Chen, C., & Liu, L.-M. (1993). Forecasting time series with outliers. *Journal of Forecasting*, *12*(1), 13–35. https://doi.org/10.1002/for.3980120103

Crimi, N., & Eddy, W. (2014). Top-Coding and Public Use Microdata Samples from the U.S. Census Bureau. *Journal of Privacy and Confidentiality*, *6*(2). https://doi.org/10.29012/jpc.v6i2.639

Davydenko, A., & Fildes, R. (2013). Measuring forecasting accuracy: The case of judgmental adjustments to SKU-level demand forecasts. *International Journal of Forecasting*, *29*(3), 510–522. https://doi.org/10.1016/j.ijforecast.2012.09.002

de Montjoye, Y.-A., Hidalgo, C. A., Verleysen, M., & Blondel, V. D. (2013). Unique in the Crowd: The privacy bounds of human mobility. *Scientific Reports*, *3*(1), 1376. https://doi.org/10.1038/srep01376

Fildes, R., Goodwin, P., Lawrence, M., & Nikolopoulos, K. (2009). Effective forecasting and judgmental adjustments: An empirical evaluation and strategies for improvement in supply-chain planning. *International Journal of Forecasting*, *25*(1), 3–23. https://doi.org/10.1016/j.ijforecast.2008.11.010

Fildes, R., Goodwin, P., & Önkal, D. (2019). Use and misuse of information in supply chain forecasting of promotion effects. *International Journal of Forecasting*, *35*(1), 144–156. https://doi.org/10.1016/j.ijforecast.2017.12.006

Fulcher, B. D., & Jones, N. S. (2014). Highly Comparative Feature-Based Time-Series Classification. *IEEE Transactions on Knowledge and Data Engineering*, *26*(12), 3026–3037. https://doi.org/10.1109/TKDE.2014.2316504

Goerg, G. (2013). Forecastable component analysis. In *International conference on machine learning* (pp. 64-72). PMLR.

Goldfarb, A., & Tucker, C. (2013). Why managing consumer privacy can be an opportunity. *MIT Sloan Management Review*, 54(3), 10.

Goldfarb, A., & Tucker, C. E. (2011). Privacy Regulation and Online Advertising. *Management Science*, *57*(1), 57–71. https://doi.org/10.1287/mnsc.1100.1246

Gonçalves, C., Bessa, R. J., & Pinson, P. (2021a). A critical overview of privacy-preserving approaches for collaborative forecasting. *International Journal of Forecasting*, *37*(1), 322–342. https://doi.org/10.1016/j.ijforecast.2020.06.003

Gonçalves, C., Bessa, R. J., & Pinson, P. (2021b). Privacy-Preserving Distributed Learning for Renewable Energy Forecasting. *IEEE Transactions on Sustainable Energy*, *12*(3), 1777–1787. https://doi.org/10.1109/TSTE.2021.3065117

Gonçalves, C., Pinson, P., & Bessa, R. J. (2021c). Towards Data Markets in Renewable Energy Forecasting. IEEE Transactions on Sustainable Energy, 12(1), 533–542. https://doi.org/10.1109/TSTE.2020.3009615

Gregorutti, B., Michel, B., & Saint-Pierre, P. (2017). Correlation and variable importance in random forests. *Statistics and Computing*, 27(3), 659-678.

Herzen, J. et al. (2022). Darts: User-friendly modern machine learning for time series. Journal of Machine Learning Research, vol. 23, num. 124, pg. 1-6. URL: http://jmlr.org/papers/v23/21-1177.html.

Hewamalage, H., Bergmeir, C., & Bandara, K. (2021). Recurrent Neural Networks for Time Series Forecasting: Current status and future directions. International Journal of Forecasting, 37(1), 388–427. https://doi.org/10.1016/j.ijforecast.2020.06.008

Hewamalage, H., Bergmeir, C., & Bandara, K. (2022). Global models for time series forecasting: A Simulation study. *Pattern Recognition*, *124*, 108441. https://doi.org/10.1016/j.patcog.2021.108441

Hyndman, R.J., & Athanasopoulos, G. (2021) *Forecasting: principles and practice*, 3rd edition, OTexts: Melbourne, Australia. OTexts.com/fpp3. Accessed on Feb. 23rd, 2023.

Imtiaz, S., Horchidan, S.-F., Abbas, Z., Arsalan, M., Chaudhry, H. N., & Vlassov, V. (2020). Privacy Preserving Time-Series Forecasting of User Health Data Streams. *2020 IEEE International Conference on Big Data (Big Data)*, 3428–3437. https://doi.org/10.1109/BigData50022.2020.9378186

Kang, Y., Hyndman, R. J., & Smith-Miles, K. (2017). Visualising forecasting algorithm performance using time series instance spaces. *International Journal of Forecasting*, *33*(2), 345–358. https://doi.org/10.1016/j.ijforecast.2016.09.004

Khosrowabadi, N., Hoberg, K., & Imdahl, C. (2022). Evaluating human behaviour in response to AI recommendations for judgemental forecasting. *European Journal of Operational Research*, *303*(3), 1151–1167. https://doi.org/10.1016/j.ejor.2022.03.017

Koning, A. J., Franses, P. H., Hibon, M., & Stekler, H. O. (2005). The M3 competition: Statistical tests of the results. International Journal of Forecasting, 21(3), 397–409. https://doi.org/10.1016/j.ijforecast.2004.10.003

Li, N., Li, T., & Venkatasubramanian, S. (2006, April). t-closeness: Privacy beyond k-anonymity and l-diversity. In 2007 IEEE 23rd international conference on data engineering (pp. 106-115). IEEE.

Li, L., Kang, Y., & Li, F. (2022). Bayesian forecast combination using time-varying features. International Journal of Forecasting, S0169207022000930. https://doi.org/10.1016/j.ijforecast.2022.06.002

Liyue Fan & Li Xiong. (2014). An Adaptive Approach to Real-Time Aggregate Monitoring With Differential Privacy. *IEEE Transactions on Knowledge and Data Engineering*, *26*(9), 2094–2106. https://doi.org/10.1109/TKDE.2013.96

Loning, M., et al. (2022). sktime/sktime: v0.13.4. doi: 10.5281/zenodo.7117735.

Luo, J., Hong, T., & Fang, S.-C. (2018). Benchmarking robustness of load forecasting models under data integrity attacks. *International Journal of Forecasting*, *34*(1), 89–104. https://doi.org/10.1016/j.ijforecast.2017.08.004

Makridakis, S., & Hibon, M. (2000). The M3-Competition: Results, conclusions and implications. *International Journal of Forecasting*, *16*(4), 451–476. https://doi.org/10.1016/S0169-2070(00)00057-1

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802–808. https://doi.org/10.1016/j.ijforecast.2018.06.001

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, S0169207021001874. https://doi.org/10.1016/j.ijforecast.2021.11.013

Martin, K. D., Borah, A., & Palmatier, R. W. (2017). Data Privacy: Effects on Customer and Firm Performance. *Journal of Marketing*, *81*(1), 36–58. https://doi.org/10.1509/jm.15.0497

Nin, J., & Torra, V. (2009). Towards the evaluation of time series protection methods. *Information Sciences*, *179*(11), 1663–1677. https://doi.org/10.1016/j.ins.2009.01.024

Nogueira, Fernando (2014). Bayesian Optimization: Open source constrained global optimization tool for Python. URL: https://github.com/fmfn/BayesianOptimization.

Petropoulos, F., Apiletti, D., Assimakopoulos, V., Babai, M. Z., Barrow, D. K., Taieb, S. B., Bergmeir, C., Bessa, R. J., Bijak, J., Boylan, J. E., Browell, J., Carnevale, C., Castle, J. L., Cirillo, P., Clements, M. P., Cordeiro, C., Oliveira, F. L. C., Baets, S. D., Dokumentov, A., … Ziel, F. (2022). *Forecasting: Theory and practice*. 167.

Petropoulos, F., & Siemsen, E. (2022). Forecast Selection and Representativeness. *Management Science*, mnsc.2022.4485. https://doi.org/10.1287/mnsc.2022.4485

Qi, L., Li, X., Wang, Q., & Jia, S. (2022). fETSmcs: Feature-based ETS model component selection. *International Journal of Forecasting*, S0169207022000954. https://doi.org/10.1016/j.ijforecast.2022.06.004

Rose, O. (1996). *Estimation of the Hurst parameter of long-range dependent time series* (Vol. 137).

Report.

Schneider, M. J., Jagpal, S., Gupta, S., Li, S., & Yu, Y. (2018). A flexible method for protecting marketing data: An application to point-of-sale data. Marketing Science, 37(1), 153-171.

Seabold, S. & Perktold, J. (2010). statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*.

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, *36*(1), 75–85. https://doi.org/10.1016/j.ijforecast.2019.03.017

Sobolev, D. (2017). The effect of price volatility on judgmental forecasts: The correlated response model. *International Journal of Forecasting*, *33*(3), 605–617. https://doi.org/10.1016/j.ijforecast.2017.01.009

Sommer, B., Pinson, P., Messner, J. W., & Obst, D. (2021). Online distributed learning in wind power forecasting. *International Journal of Forecasting*, *37*(1), 205–223. https://doi.org/10.1016/j.ijforecast.2020.04.004

Spiliotis, E., Kouloumos, A., Assimakopoulos, V., & Makridakis, S. (2020). Are forecasting competitions data representative of the reality? *International Journal of Forecasting*, *36*(1), 37–53. https://doi.org/10.1016/j.ijforecast.2018.12.007

Sweeney, L. (2002). k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *10*(05), 557–570. https://doi.org/10.1142/S0218488502001648

Talagala, T. S., Li, F., & Kang, Y. (2022). FFORMPP: Feature-based forecast model performance prediction. *International Journal of Forecasting*, *38*(3), 920–943. https://doi.org/10.1016/j.ijforecast.2021.07.002

Véliz, C., & Grunewald, P. (2018). Protecting data privacy is key to a smart energy future. *Nature Energy*, *3*(9), 702–704. https://doi.org/10.1038/s41560-018-0203-3

Wang, X., Smith, K., & Hyndman, R. (2006). Characteristic-Based Clustering for Time Series Data. *Data Mining and Knowledge Discovery*, *13*(3), 335–364. https://doi.org/10.1007/s10618-005-0039-x

Willinger, W., Paxson, V., & Taqqu, M. S. (1998). Self-similarity and heavy tails: Structural modeling of network Traffic. R. Adler, R. Feldman, and MS Taqqu, editors, *A Practical Guide to Heavy Tails: Statistical Techniques for Analyzing Heavy Tailed Distributions*, Birkhauser Verlag, Boston.

1. **Appendix**

**Implementation Details**

*Preprocessing*

Time series were pre-processed by taking the log of the original series. For the VAR model, we took the first difference of the log series. For the global models (RNN and LGBM), we divided each log series by its respective mean.

*Model Implementations*

All local models (SES, DES, TES, and AutoARIMA) were implemented using the *sktime* forecasting module (Loning et al., 2022). We used additive trend and seasonality components for DES and TES. We set AutoARIMA to apply a seasonal model (SARIMA) with a maximum of twenty-five iterations.

The VAR model from the *statsmodels* (Seabold & Perktold, 2010) module was applied to subsets of approximately five time series at a time, ensuring that each subset consisted only of time series with the same length.

The LGBM and RNN models were implemented using the *darts* module (Herzen et al., 2022). For both models, we reserved a validation time period immediately prior to the desired forecast horizon. We used Bayesian optimization (Nogueira, 2014) to optimize the hyperparameters of each model to minimize the absolute forecast error (L1 loss) in the validation time period. We retrained each model using the optimized hyperparameters and the full training data (including the validation period) prior to generating forecasts for the desired forecast horizon.

The Bayesian optimizer was initialized for the LGBM models using ten starting points and run for 50 iterations. For the RNN the optimizer was initialized using five starting points and run for 25 iterations.

The hyperparameters for the LGBM and RNN models and their corresponding ranges are shown in Tables 9 and 10. We limited the RNN to the last ten input-output window samples from each time series for computational efficiency and trained ten RNN models taking the median of the forecasts as the final forecast (Hewamalage et al., 2022).

**Table 9: LGBM hyperparameter ranges.**

|  |  |
| --- | --- |
| **Hyperparameter** | **Range** |
| Input Window Length | 25 |
| Learning Rate | [0.01, 0.1] |
| Number of Boost Rounds | [50, 1000] |
| Number of Leaves | [10, 100] |
| Bagging Frequency | [1, 5] |
| Bagging Fraction | [0.01, 1] |
| L2 Regularization Parameter | [0, 0.5] |
| Minimum Observations in Leaf | [10, 100] |

**Table 10: RNN hyperparameter ranges.**

|  |  |
| --- | --- |
| **Hyperparameter** | **Range** |
| Input Window Length | 25 |
| Training Length | 30 |
| Learning Rate | [0.001, 0.1] |
| Weight Decay | [0.0001, 0.0008] |
| Number of Layers | [1, 2] |
| Hidden Dimension | [20, 50] |
| Batch Size | [200, 700] |
| Number of Epochs | [3, 30] |
| Dropout Rate | [0.1, 0.5] |

**Mathematical Details of Identification and Attribute Disclosure**

To perform identification disclosure, we assume a third party possesses some original data on a unit of interest in the protected dataset. Denote this original data which contains a direct identifier (e.g., the identity of retailer ) and original data which contains a sequence of values that are components of the original time series .

We let denote the random variable (from the perspective of the third party) that indicates the corresponding for , i.e., when the values in are components of the original version of the protected series . Since the true value is unknown, the third party predicts the value of to be the series with the highest match probability, conditional on the known values, as follows,

, (1)

where identification disclosure occurs when The probability is calculated as follows. Let denote the protected values of each time series that occur in the same time periods as . The third party computes the similarity between and the protected values using the Euclidean distance,

.

Using these similarities, the third party builds a probability mass function for over all protected series in as

,

and predicts as in (1).

To estimate the risk of identification disclosure, we perform simulations in which we sample sequential values from each original time series , and we measure the average proportion of series which are identified. The sampled values are denoted . Each of the vectors corresponds to one of the original time series, and we compute conditional on the sampled from series . We repeat this simulation times to obtain , and computethe average proportion of correctly identified time series across all external data samples and original time series,

where [.] are Iverson brackets.

These simulations assume that the third party in possession of predicts the match for each vector independently of the predicted matches for other vectors. The risk estimate from a given simulation is equivalent to the identification risk when independent third parties are each in possession of one of the vectors and each attempts identification risk as described above. Overall, multiple vectors may be matched to the same protected time series.

1. For legislation examples in the United States, see [this](https://iapp.org/resources/article/us-state-privacy-legislation-tracker/) map. [↑](#footnote-ref-1)
2. See articles 6, 45, and 46 of the GDPR. [↑](#footnote-ref-2)
3. All norms on are equivalent to the Euclidean norm. [↑](#footnote-ref-3)
4. Note that we could also define a distance matrix based on the actual time series values , where would become a function of rather than . [↑](#footnote-ref-4)
5. The averages for additive noise and differential privacy excludes the VAR model error for AN (s = 1) and DP () since the errors were over 1000% larger than the error of any other model. Due to the large noise infused from these privacy methods, the VAR could not fit small enough coefficients to smooth out the noise, resulting in extremely poor forecast accuracy. For example, the magnitude of the first lag coefficient for an AN (s = 1) protected time series increased from -0.372 in the original data to -0.679 in the protected data. This coefficient was multiplied by an extreme outlier at time causing the forecast at time to explode and skew the overall average forecast error. This problem did not occur for the other forecasting models, which did a better job smoothing out the random noise. [↑](#footnote-ref-5)
6. AvgRelAE can be generalized to accommodate multiple forecasts for each series. See Fildes et al. (2013) for the AvgRelMAE. [↑](#footnote-ref-6)