**Preserving Time Series Features for Forecasting with Protected Data**

**Abstract**

While data protection can enable access to otherwise restricted time series data, privacy methods alter time series values and can significantly change forecasts. Existing decentralized privacy solutions are complex to implement and only enable model parameters or forecasts to be shared with forecasters while existing centralized privacy methods are poorly suited for preserving forecast accuracy at meaningful levels of privacy.

We propose a novel matrix-based privacy method called *k*-nTS+, which maintains forecast accuracy by using machine learning techniques on time series features to choose values to swap between time series. We apply our method to multiple forecasting models and find that it balances the tradeoff between privacy and forecast accuracy well. Compared to unprotected data, *k*-nTS+ reduces forecast accuracy by only 14% at similar protection levels to other forms of protected data like differential privacy. We also show that the representativeness of k-nTS+ protected time series is improved which improves trust and makes it more likely for oshare protected time series data

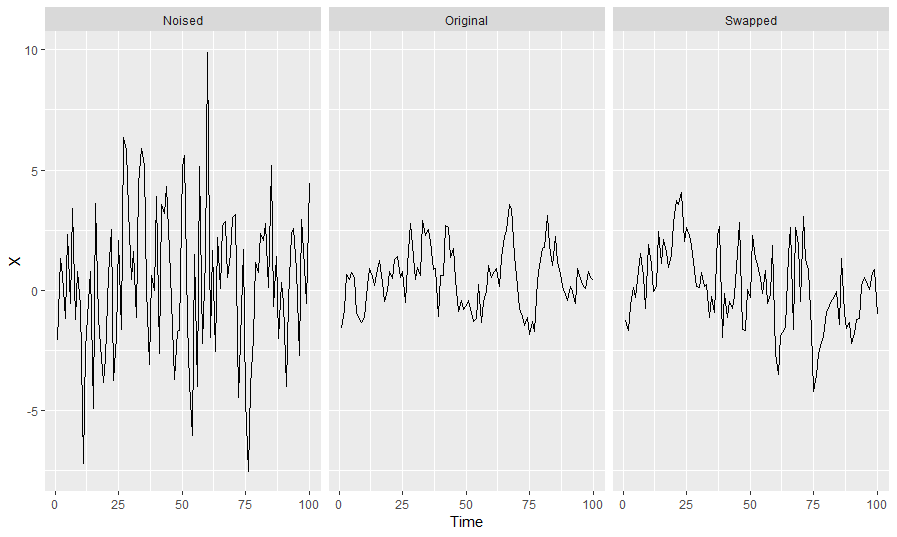
1. **Introduction**

Personally identifiable time series data is now ubiquitous and requires protection (Boone et al., 2019). Recently, the General Data Protection Regulation (GDPR)[[1]](#footnote-1) and other privacy laws require organizations to anonymize their personal data or place strict limitations on data transfers and processing[[2]](#footnote-2). However, Gonçalves et al. (2021) showed that anonymizing time series data with differential privacy produced unusable forecasts and required different solutions such as only sharing forecasts via federated learning. In this paper, we propose a matrix-based data protection solution for organizations to share the underlying time series data.

Various data protection methods are available depending on whether time series are stored in a single data set (centralized) or spread across multiple data owners/data sets (decentralized). In the decentralized scenario, multi-party computation or 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). We focus on the centralized scenario in which a single data owner uses privacy methods to protect a time series data set. These privacy methods directly alter the confidential data to produce protected time series. The goal is to limit the ability of a bad actor to identify data subjects (in our case, time series) and learn sensitive information about them. For example, Census applies random noise to time series to protect Quarterly Workforce Indicator data (Abowd et al., 2012). The concern for forecasters is that these noised time series degrade forecast accuracy to unusable levels.

Consider the example shown in Figure 1. The time series shown in the middle plot is a simulated AR(1) process with autoregressive parameter . The series on the left is the original series with random noise added to each period that is proportional to the standard deviation of the original series. Estimating an ARIMA(1, 0, 0) model on the original series yields an estimate of with a standard error of 0.07, while the noised series yields an estimate of with a standard error of 0.10. The series on the right was created by swapping the original series values with values from two other simulated AR(1) processes, both with . The swapped series better preserves the estimated autocorrelation of the original series, with an estimate of and a standard error of 0.08. Visually, the swapped version of the original series is more plausible than the noised version, but in both cases the time series features (e.g., AR (1) parameter, variance) are degraded. This paper improves data protection methods by focusing on the time series features predictive of forecast accuracy.

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



This paper examines how data protection changes time series features and how this affects forecasting model performance. As shown in Figure 2, we first create protected time series using As

Figure 2 provides an illustration of our k-nTS+ data protection framework. We begin our data protection framework as usual (black arrows) by generating protected data using baseline privacy methods such as additive noise or differential privacy. Then, we generate forecasts for both the original and protected series and compare their accuracies. To improve forecast accuracy with data protection (blue arrows) for our method, we use a machine learning-based feedback loop (CITE HERE XXX) on the accuracy results to rank the time series features that are most predictive of forecast accuracy. We then compute a distance-based matrix of these features to choose the time series values to swap in *k*-nTS+. The goal of data protection framework is to produce protected time series that maintain the useful features for forecasting from their original time series.

**Figure 2: k-nTS+ data protection framework (Blue arrows indicate the feedback loop which informs our proposed *k*-nTS protection method).**

Graphical user interface, application

Description automatically generated

Our contributions are two-fold. First, we analyze privacy adjusted forecasts from multiple forecasting models and privacy methods. This comparison is needed because different forms of data protection produce different data points which will ultimately have different forecasts than what would be produced based on the original data. We extract time series features which are predictive of forecast accuracy and show how these features change under data protection. We use these changes to explain the performance of various forecasting models in our empirical application. In such a situation, our work would help answer how forecast accuracy would be affected if the data owners applied data protection methods prior to sharing their data.

Overall, while recent attention has been paid to privacy preserving collaborative forecasting, our interest is in forecasting using a single protected dataset. There has been no work which compares multiple forecasting models' accuracies when forecasting for a single protected dataset, or a comparison of models' accuracies under various privacy methods. The works which have shown that data protection degrades forecast accuracy have also not given detailed explanations as to why model performance is worse on protected data.

Second, we propose a novel privacy method that preserves the usefulness to forecasters. Extant privacy methods apply protection do not consider how privacy methods degrade forecast accuracy or time series features useful for forecasting. We create a matrix-based k-nTS+ privacy method that uses a feedback loop based on the relationship between time series features and forecast accuracy. k-nTS+ swaps the values of time series with each other when they have similar features predictive of good forecast accuracy. Results show that our method provides significantly better accuracy at similar levels of privacy protection compared to competitor privacy methods. Further, using the *performance gap* from the REP metric (Petropoulos & Siemsen, 2022), we show that the protected time series produced under *k*-nTS+ are much more representative of the original series compared to the protected series from other methods, leading to improved trust in data sharing.

In Section 2, we review the relevant literature. Section 3 describes the swapping mechanism of the baseline *k*-nTS method, and Section 4 establishes the time series features used for swapping in k-nTS+. In Section 5 we provide the framework for the *k*-nTS+ method and Section 6 presents the results We discuss our conclusions in Section 7, including recommendations for future research.

1. **Literature Review**

Goncalves, Pinson, et al., (2021c) modeled a data market where data owners are compensated for sharing their data, and purchase forecasts based on the data from other parties. This gives data owners a monetary incentive to share their data. However, they may still be discouraged from sharing due to privacy concerns with a central party. Other privacy-preserving solutions for collaborative forecasting include secure multi-party computation, decomposition-based methods, and data transformation techniques, all of which are succinctly described by (Gonçalves et al., 2021a). a

The centralized scenario uses privacy methods which generate protected data sets for sharing. Gonçalves et al., (2021a) show that additive noise using 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). (found thatAcan be useful asHowever, Luo et al., (2018) simulated data integrity attacks and found that multiplicative noise reduces forecast accuracy... (by XXX %).

Another type of privacy method is generalization, where data records are aggregated or combined to make every record (or time series) identical to at least other records on a pre-determined set of attributes (or time periods). Nin & Torra (2009) evaluate the change in forecast accuracy for simple exponential smoothing, double exponential smoothing, linear regression, multiple linear regression, and polynomial regression applied to *k*-anonymized data. The authors find an overall reduction in forecast accuracy even for but do not provide the accuracy of each model individually. Top- and bottom-coding are used as another privacy method to replace the tails of distributions with a threshold value such as high income levels. Top- and bottom-coding are likely to 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. *Privacy Adjusted Forecasts*

As shown in Figure 2, privacy methods adjust forecasts by altering the underlying time series input to a forecasting model. Similar to judgmental adjustments, this presents the forecaster with multiple possible forecasts to select from. To inform this selection, we reference the long history on judgmental forecasting (Cite XXX). However, there are two key differences between privacy adjustments and judgmental adjustments.

First, judgmental adjustments alter forecasts after they are output from a forecasting model, and the underlying time series features are not changed. Forecasters discuss what characteristics of judgmental adjustments improveDavydenko & Fildes (2013) found that bFildes et. al. (2019) showed that f The accuracy improvements are greater for low volatility time series which are easier to forecast (Fildes et al., 2009). For privacy adjustments, the adjustment is applied directly to the data rather than the forecasts, which alters the underlying time series features. As a result, forecasters must carefully consider how the forecasting model interacts with time series features to preserve forecast accuracy.

Second, the motivations for judgmental adjustments are different than privacy protection. Motivations include gaining control of the forecasting process, incorporating practitioner expectations, and compensating for judgmental biases ((Petropoulos et al., 2022) section 3.7.3). Often, the goal is to incorporate the intuition and experience of the adjuster, knowledge of special events, or insider or confidential information to add information with high diagnosticity to improve forecast accuracy (Fildes et. al., 2019). Despite varying motivations for judgmentally adjusting forecasts, these adjustments have been found to improve the accuracy of monthly demand forecasts from statistical models by an average of 10% (Davydenko & Fildes, 2013). For privacy protection, the goal is not to improve forecast accuracy – instead, the goal is to improve privacy while maintaining forecast accuracy.

* 1. *Time Series Features and Forecasting*

To inform the selection of privacy methods, we review how time series features interact with forecasting models to maintain accuracy. Thousands of features have been used for time series classification (Fulcher & Jones, 2014) and some of those are useful for forecasting models. Bandara et al. (2018) used a small set eighteen interpretable features such as the mean, variance, and strength of seasonality, for clustering and forecasting similar time series and found that the accuracy of recurrent neural network models improved. 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 (which are 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 the 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 combination of forecasts. Qi et. al. (2022) found that forecasts using the strength of trend and seasonality for exponential smoothing model selection had lower MASE, sMAPE, and MSIS 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 what would yield the minimum forecast error for the M4 competition data. This approach had 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 average MASE by approximately 1.1% relative to the next best forecast combination method.which ones??). Petropoulos & Siemsen, (2022) created a representativeness metric to select models with trend and seasonality components when the respective signals of these components are strong. For most data frequencies, their approach produces lower average MASE on the M, M3, and M4 competition data, and selects the best forecasting model approximately 3% more often than the other methods.XXX%.

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

Let be a given set of time series data (*n*-vectors). Using standard privacy methods such as differential privacy, a data provider releases protected data for each time series based on the confidential values up until time . The main issue with these methods is that they choose protected values based on predefined rules and not changes in forecast accuracy. The goal of the data provider 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 provider using a matrix based k-nTS (k-nearest time series) swapping method, where the data provider releases a set of protected time series where is based on , the confidential values of all series up through time . The *k*-nTS method matches time series that have similar features within a rolling window of their past values. Then, it uses randomization to replace each with a confidential value from another time series with similar features to balance the trade-off between forecast accuracy and privacy.

Depending on the quantity of available data, *k*-nTS can use rolling windows of data which adjust for dynamic changes in time series features. For example, if we choose a rolling window of size *n*, then where . Protection in subsequent time 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 . For simplicity, we omit the *t* subscript for the feature vectors, and write . See Li et al. (2022) for another example of the  
computation of time series features based on a rolling, fixed-length window.

For each time series , the data provider computes the feature vector . This vector can contain any single-valued feature calculated based on the values in , such as the strength of the trend and seasonality, the spectral entropy, or the mean of the values in the current window. This produces a set of *m*-vectors containing the features of each of the *J* time series windows. For each of the feature vectors , the data provider computes a set of squared distances of the elements of the set . We define as the distance between and , i.e., the feature vectors corresponding to two distinct time series from a given set . Without loss of generality, we use the Euclidean norm, or ℓ2-norm as a distance metric[[3]](#footnote-8). Since our case is multivariate and partially ordered, we can get a totally ordered set based on the Euclidean distance.

Let us define as the *k*th nearest neighbor of , with 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*-nTS (*k*-nearest time series) can be represented as the set based on or an ordered set .

For a more efficient computation of such ordering, we introduce a symmetric distance matrix containing the squared distances between time series feature vectors. The squared distance between and is given by and is the (*i, j*)th entry of (also note that . Suppose that we are given a data matrix , where (i.e., We can write , i.e., a confidential data matrix, as the following:

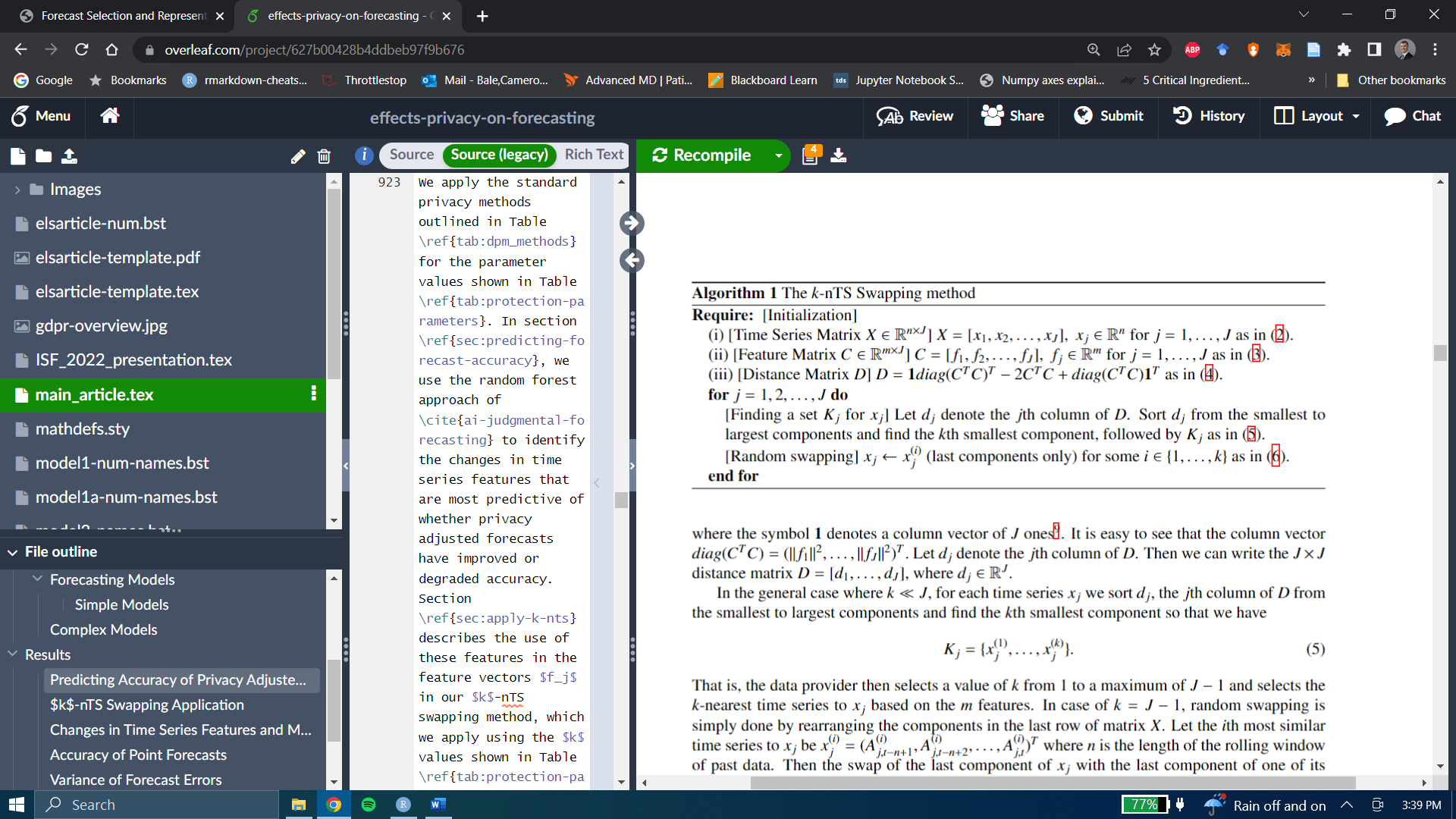
Graphical user interface, text, application

Description automatically generated

where , , and 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



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

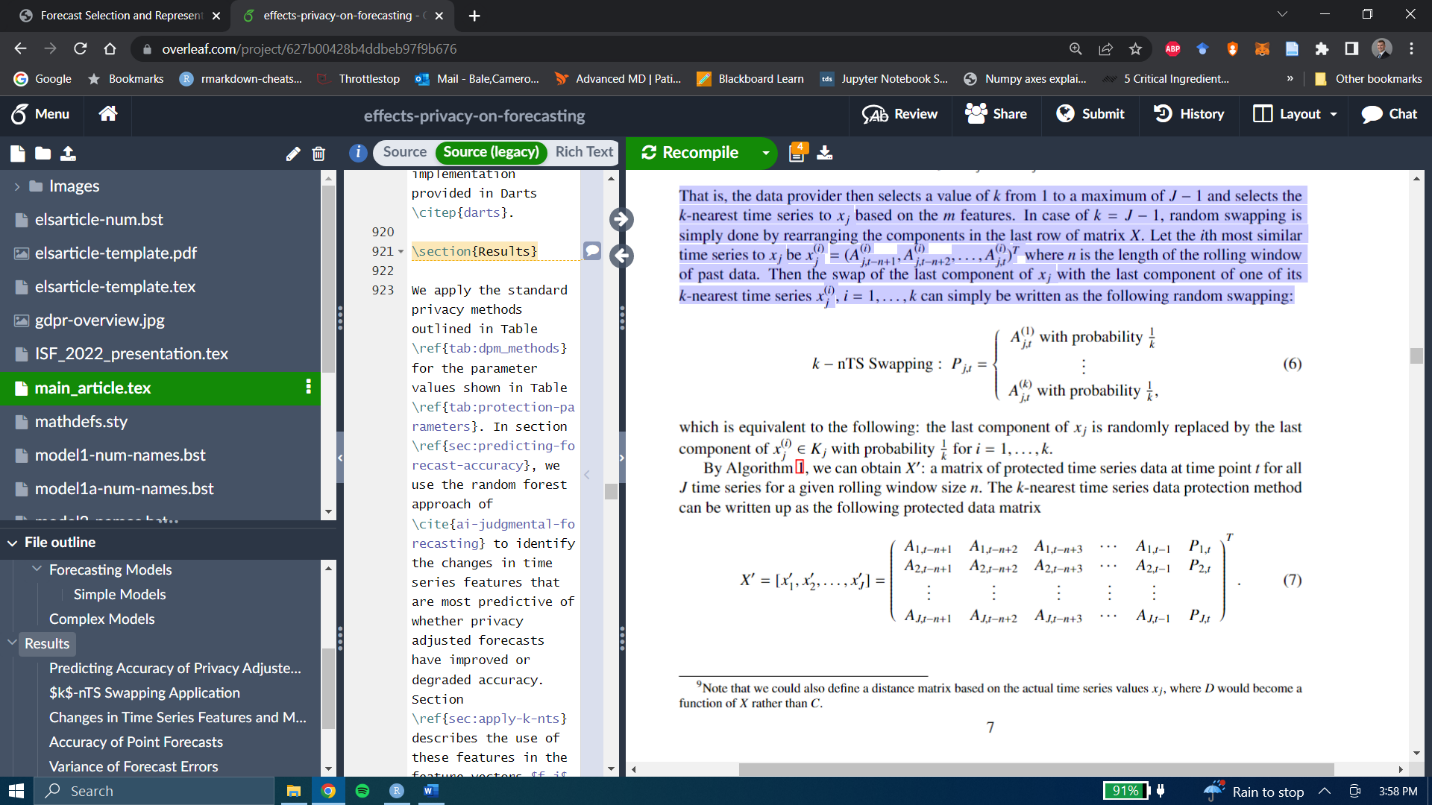
,

where the symbol **1** denotes a column vector of ones[[4]](#footnote-9). 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 , the *j*th column of from the smallest to largest components and find the *k*th smallest component so that we have

.

That is, the data provider then selects a value of *k* from 1 to a maximum of and selects the *k*-nearest time series to based on the *m* features. In case of , random swapping is simply done by rearranging the components in the last row of matrix . Let the *i*th most similar time series to be where *n* is the length of the rolling window of past data. Then the swap of the last component of with the last component of one of its *k*-nearest time series , can simply be written as the following random swapping:



which is equivalent to the following: the last component of is randomly replaced by the last component of with probability for

By Algorithm 1, we can obtain : a matrix of protected time series data at time point *t* for all *J* time series for a given rolling window size *n*. The *k*-nearest time series data protection method can be written up as the following protected data matrix,

A screenshot of a computer

Description automatically generated

As an example, consider a given time series where its last component is replaced by the  
last component of which was randomly selected among the 10-nearest time series (based on features) to . Using our notation, we can write

We can represent each time series as a vector, and then put them in a graph , which consists of a set *V* of vertices (or nodes) and a set *E* of undirected edges. In our case, we can use weighted edges to represent the Euclidean distance between associated time series feature vectors: . If we put all the nodes on the graph and assign weight on every edge (every pair of nodes), e.g., for all then we will have a complete graph. The *k*-nearest time series swapping method can be considered as a random edge selection problem of the graph. Figure 3 depicts the case of for the *k*-nearest time series of , where the last component of is swapped with that of .

A screenshot of a computer

Description automatically generated with medium confidence

**Fig 3**: Random edge selection for k-nTS Swapping.

1. **Time Series Features for Swapping Protection**

In this section, we describe the time series features which have been demonstrated to have a relationship with forecast accuracy. We let denote 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 is attained 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).

Next, we consider the self-similarity feature quantified using the Hurst parameter (Wang et al., 2006), which measures the long-range dependence of a time series. This feature had the largest magnitude effect on forecast accuracy in the study of the M4 data performed by (Spiliotis et al. 2020). 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 aggregated 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 focus on the definition of second-order self-similarity, where is 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 fall in the interval (0, 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. (Rose, 1996) notes that a self-similar process has a spectral density that follows a power law near , where as with . When , the spectral density increases rapidly as and will tend to have low spectral entropy, whereas when , the spectral density increases slowly as and will tend to have high spectral entropy. For a random walk with , i.e., the spectral density is finite at the origin (Rose, 1996).

We consider the remaining features from (Spiliotis et al., 2020) which had the largest effects on forecast accuracy. Since none of the privacy methods we consider will change the time series’ frequency, we omit this feature from consideration, noting that higher frequencies are associated with improved forecast accuracy. We include skewness and kurtosis which measure the shape of the distribution of time series’ values.

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

We use a measure of 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.

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) which is designed to handle multiple seasonalities 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).

*E\_acf*

This feature gives a measure of the forecastability of a time series after the trend and seasonality have been accounted for.

Continuing with the decomposed series, we compute the strength of trend and strength of the seasonal component as follows,

,

.

In practice, the values of and are bounded to (Hyndman 2022).

Our final two features are included to maintain data utility throughout the *k*-nTS swapping process. The idea is to swap values between series that not only have similar characteristics, but whose values have similar magnitudes. Toward this end, we include the mean and the variance of ,

.

Figure 3 compares two monthly time series on the nine time series features discussed in this section, the values of which are shown in Table 1. The good forecastability of the series on the left is indicated by the low spectral entropy and high Hurst coefficient values. The series on the right, however, is essentially a random walk as indicated by the value of the Hurst coefficient, and a spectral entropy of one indicates a very low signal to noise ratio. Another notable difference is in the strength of the trend of each series– most of the variance of the series on the left is due to a strong trend, which is forecastable, whereas the variance in the series on the right appears to be due to the randomness of the series. The series on the left has low *Kurtosis*, i.e., light tails relative to the standard normal distribution, whereas the opposite is true for the series on the right.

**Fig 3: Comparison of a time series with desirable features (easy to forecast) and a time series with undesirable features (difficult to forecast).**

Chart, scatter chart

Description automatically generated

**Table 1: feature value comparison between a series with desirable features (easy to forecast) and a series with undesirable features (difficult to forecast)**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Desirable Features (left Fig. 1)** | **Undesirable Features (right Fig. 1)** |
| *SpecEntropy* | 0.07 | 1.00 |
| *Hurst* | 1.00 | 0.50 |
| *Skewness* | -0.42 | -0.57 |
| *Kurtosis* | -1.24 | 1.16 |
| *E\_acf* | -0.09 | -0.19 |
| *Trend* | 0.97 | 0.12 |
| *Seasonality* | 0.16 | 0.23 |
| *SeriesMean* | 7.96 | 7.01 |
| *SeriesVariance* | 0.29 | 0.65 |

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

The *k*-nTS+ privacy method builds on *k*-nTS by including a feature selection process based on the time series features and forecast accuracy under baseline privacy methods. The idea is to perform swapping which limits the changes in features which are most predictive of forecast accuracy.

The feature selection process starts with the data controller generating forecasts for period for both the original data and the data protected using baseline privacy methods such as differential privacy and additive noise. The data controller measures the forecast errors and time series feature values for each original and protected time series. Our reasoning is that a feature should be included in the *k*-nTS swapping process (and preserved in the protected data) if it is predictive of the forecast error across the original and baseline protected data sets.

An initial filtering of the features is performed using the RReliefF algorithm (Robnik-Sikonja & Kononenko 2003) which assigns a weight to each feature that indicates the ability of that feature to discriminate between nearest neighbor time series with different forecast errors compared to nearest neighbor time series with similar forecast errors. This is an intuitive choice for a feature importance measure since our privacy method swaps values between nearest neighbor time series. A similar filtering was performed by Li et al. (2022) for their forecast combination methodology.

One downside of Relief-based algorithms is that they do not remove redundant features. These algorithms can be applied recursively, removing the feature(s) with the lowest weights in each iteration, but there is not a straight-forward method for choosing how many features to ultimately include (Urbanowicz et al. 2018). Including all important features from RReliefF in *k*-nTS would significantly increase the dimensionality and reduce the efficiency of the swapping process. To address this problem, we apply a recursive feature elimination (RFE) algorithm based on a random forest which predicts forecast accuracy using subsets of the most important features from the RReliefF algorithm. Prior work has shown that random forest based RFE is efficient when applied to sets of highly correlated features, i.e., it selects a small set of features with good prediction performance (Gregorutti et al. 2017).

In each iteration of the RFE algorithm, we store the out-of-bag (oob) mean-squared error (MSE) and permutation-based feature importance values from a random forest used to predict the accuracy of the forecasts for each original and protected time series using the time series features. The least important feature is removed, and the model is then retrained for the next iteration. These steps continue recursively until only one feature remains. The full RFE algorithm is repeated times.

We compute the average MSE of the oob predictions for each feature subset size across the repetitions of RFE. Similarly, for each repetition, we rank the features based on the inverse of the order in which they were eliminated and average these ranks. Let denote the subset size with the minimum average MSE across repetitions and let denote that minimum average MSE. Our final subset size is chosen as follows,

,

Where is the maximum percentage difference between the minimum MSE and the MSE of our chosen subset size. The chosen subset size is the smallest number of features which offers an average predictive accuracy with similar performance to the subset size with the best average accuracy. We include the features with the highest average ranks in the *k*-nTS+ swapping method.

**Algorithm: RFE for *k*-nTS+ Using Random Forest**

**for** i:

* Train random forest
* Calculate MSE of oob predictions
* Calculate permutation-based feature importance

**for** subset size :

* + Keep the most important features
  + Retrain random forest using only the most important features
  + Calculate MSE of oob predictions
  + Calculate permutation-based feature importance

**end**

**end**

Calculate the average MSE for each subset size across all iterations.

Calculate the rank of each feature as the average of the elimination orders from each iteration.

Calculate the desired number of features as the minimum number of features with an average prediction error within of the minimum average prediction error.

----------------------------------------------------------------

The *k*-nTS+ algorithm can be used collaboratively between the data controller and the forecaster. If, for example, the forecaster specifies their preferred forecasting model, the data controller can apply the model to the original and protected data up through time period *T – 1*, assess which changes in features are most predictive of changes in accuracy for the specified model, and release data to the forecaster using *k*-nTS+ based on these features up through time period .

1. **Empirical Application**
   1. *Data*

Recent work by (Spiliotis et al., 2020) showed that the M3 competition data are representative of the real world on the basis of time series features. Further, since the M3 competition data is publicly available, it will enable researchers to easily compare future privacy methods applied to the same data. It is also known that complex forecasting models forecast more accurately than simple models using the M3 competition monthly micro data (Koning et al., 2005), and models that explicitly capture trend and seasonality performed the best in the overall M3 competition (Makridakis & Hibon, 2000). We test whether these results hold when forecasting using protected versions of the data. For our analyses, 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.

* 1. *Forecasting Models*

The forecasting models under study are separated into “simple” models which are trained to forecast one series at a time, and “complex” models which are trained to generate forecasts for multiple series. We perform minimal data pre-processing and allow the models to capture the important components of the series. Our goal is to assess the effects of privacy protection on the accuracy of popular forecasting models which are readily available to implement in R and/or Python and have served as benchmarks or winners in recent forecasting competitions.

**Table 2: forecasting models under study. Includes relevant information for the variant of model and whether it is a local or global forecasting model.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Model Name** | **Variant** | **Global (Yes/No)** |
| Simple Models | SES | - | N |
|  | DES | Additive trend | N |
|  | TES | Additive trend/seasonality | N |
|  | Auto-ARIMA | seasonal | N |
| Complex Models | VAR | - | N |
|  | LGBM | - | Y |
|  | RNN | LSTM | Y |

* 1. *Privacy Methods*

We apply each of the privacy methods shown in Table 3 below to the original M3 monthly micro data for each of the displayed parameter values.

**Table 3: privacy methods, and their parameter values, which we apply to the m3 monthly micro data. Values are arranged in order of strength of privacy protection.**

|  |  |  |
| --- | --- | --- |
| ***Privacy Method*** | ***Parameter*** | ***Values*** |
| Additive Noise |  | 0.25, 0.50, 1.0, 1.5, 2.0 |
| Differential Privacy |  | 20.0, 10.0, 4.6, 1.0, 0.1 |
| *k*-nTS |  | 3, 5, 7, 10, 15 |
| *k*-nTS+ |  | 3, 5, 7, 10, 15 |

* + 1. *Differential Privacy*

Given an original time series , a differentially private time series can be created using a randomized mechanism which adds Laplace random noise with scale parameter . The sensitivity is determined as the maximum absolute difference between two time series and , which differ in at most one observation, where . The mechanism satisfies -differential privacy by guaranteeing that, for every output of and every pair of series and ,

.

* + 1. *Additive Noise*

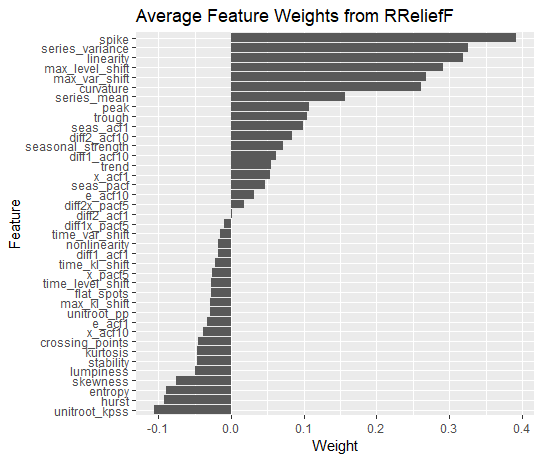
Additive noise protection is achieved by adding a normal random number with mean zero and standard deviation to each value in a time series . Protected values can be written , where and . The protection parameter denotes the number of standard deviations of that define the standard deviation of the sampling distribution of .

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

For *k*-nTS, the distance between time series is calculated using the nine features described in Section 4.3. To perform feature selection for *k*-nTS+, we create protected versions of our selected data using additive noise and differential privacy for all of the parameter values shown in Table 3 (i.e., 10 protected data sets and 1 original data set). We generate forecasts for each of the 11 data sets for time period using each of the forecasting models shown in Table 3 and compute the absolute error of each forecast for each series. So that the variation in forecast accuracy is due to changes in time series features and not the forecasting model used, we apply the *k*-nTS+ selection method for each forecasting model separately and select the features that are most important across all models.

We compute 39 time series features using the tsfeatures package in R, including the nine features described in Section 3, and use custom functions for the series mean, variance, skewness, and kurtosis. Using RReliefF, the features are used to predict the absolute forecast errors for each model for each series across the original and protected data sets. Figure 4 shows the average RreliefF weights for each of the 39 features across forecasting models.

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

****

Many features are poor predictors of forecast error and are assigned negative weights. We select the features for each forecasting model with positive weights for inclusion in the RFE algorithm.

We apply the RFE algorithm once for each forecasting model for Iterations. The average oob MSE across feature subset sizes and models is shown in Figure 5. For all of the forecasting models, most of the reduction in oob MSE is accomplished using five or fewer features.

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

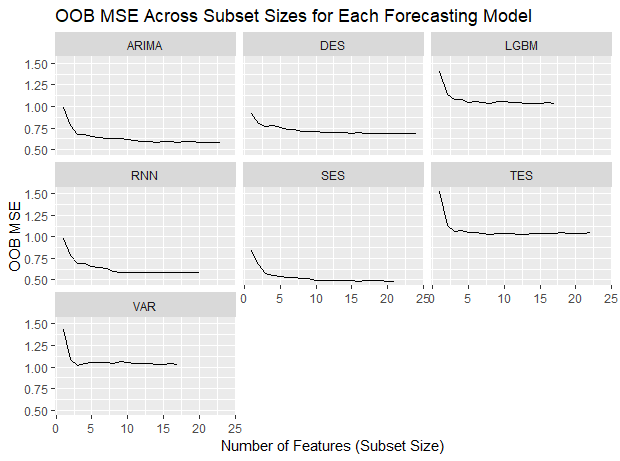
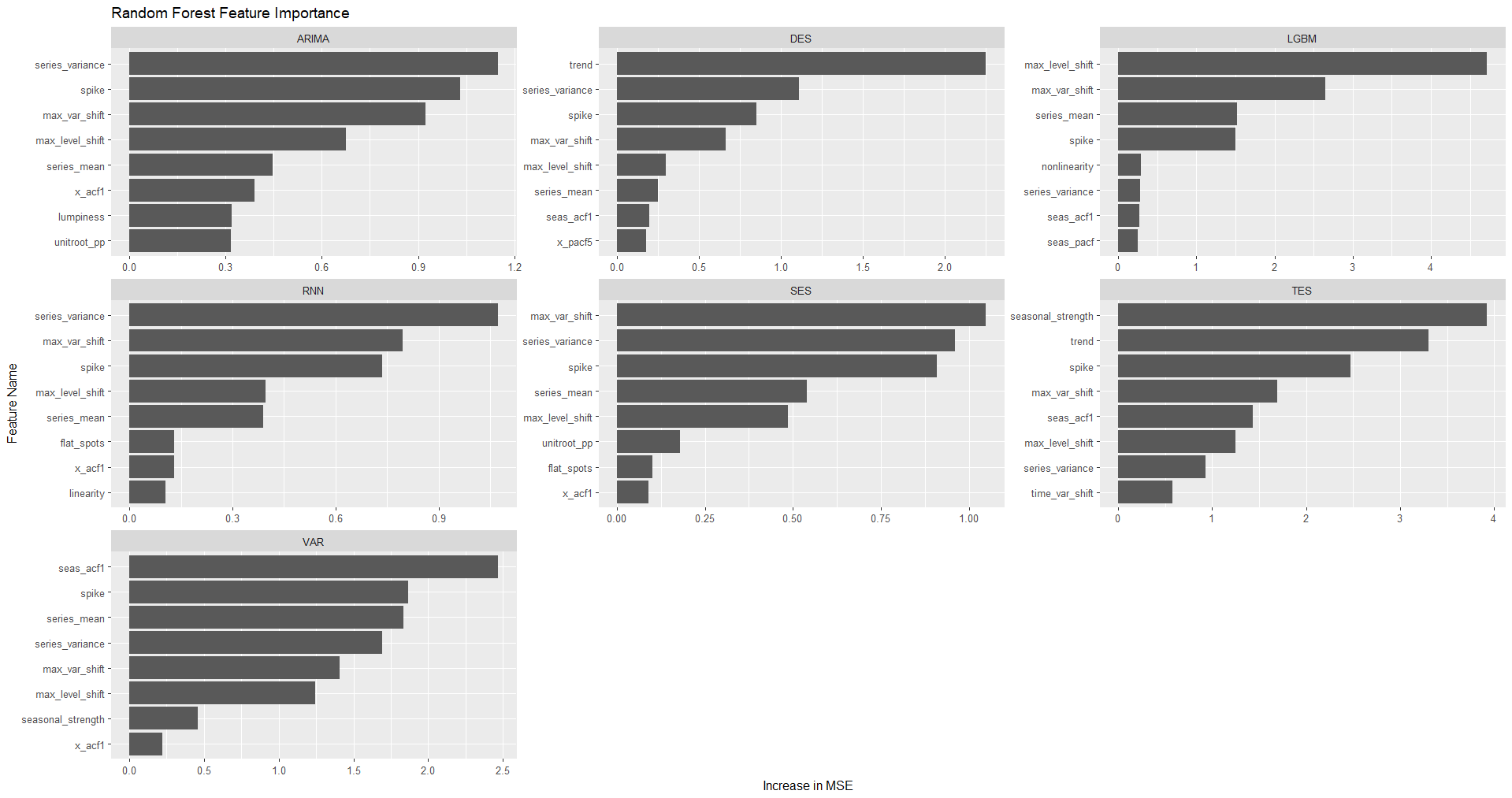


Figure 6 shows the permutation-based importance values for the eight most highly ranked features for each forecasting model. Some features, such as *spike,* *max\_var\_shift, max\_level\_shift, series\_mean* and *series\_variance* are highly ranked across most or all forecasting models. Other features appear to be highly important only for specific models. Examples include *trend*, which is highly important for DES and TES, *seasonal\_strength*, which is highly important for TES, and *unitroot\_pp,* which is important for Auto-ARIMA and SES.

**Fig 6: Permutation importance for the eight most important features (based on RFE elimination rank) for each model.**



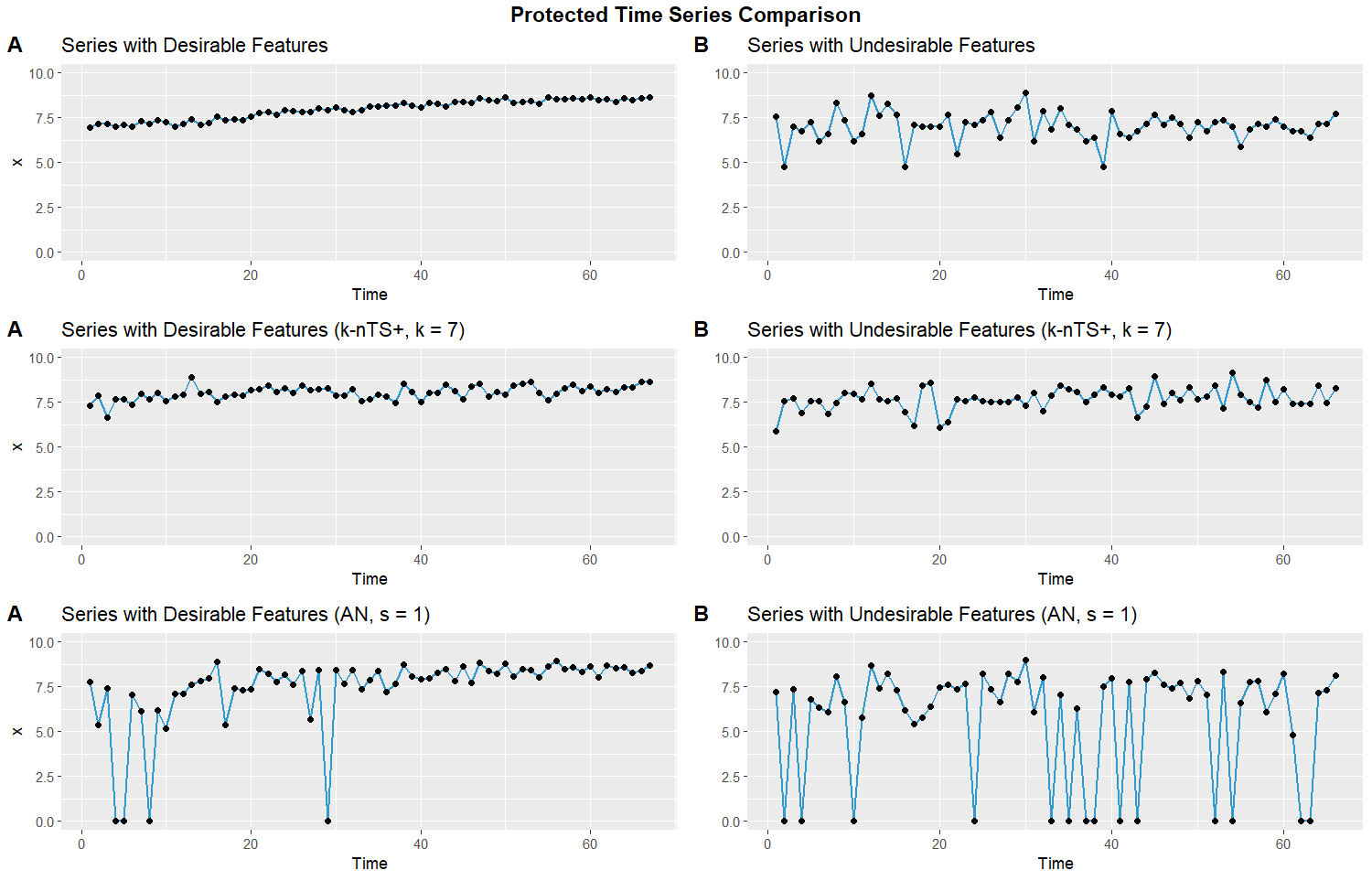
For our empirical application, we select . The average across forecasting models is six, so we select the six features with the highest average rank across the RFE iterations for all forecasting models. These features are shown in Table 4.

**Table 4: Names and descriptions of features selected for *k*-nTS+.**

|  |  |
| --- | --- |
| **Feature Name** | **Description** |
| trend | Strength of trend. |
| spike | Variance of the leave-one-out variances of the remainder component of the decomposed series. |
| max\_var\_shift | The largest variance shift between two consecutive sliding windows. |
| series\_variance | Variance of the series. |
| max\_level\_shift | The largest mean shift between two consecutive sliding windows. |
| series\_mean | Mean of the series. |

Figure 7 compares some protected versions of the series with desirable and undesirable features from Section 4.

**Figure 7: Comparison of original, AN (s = 1), and *k*-nTS+ (*k* = 7) versions of the series with desirable and undesirable features from Section 4.**



* 1. *Forecast Accuracy and Privacy Assessment*

The participants of the M3 competition were not supposed to identify the original time series. Therefore, we assess the ability of each privacy method to protect against *identification disclosure*, which occurs when a third party correctly predicts the identity of a protected time series. Each protected dataset consists of the protected series along with a pseudo identifier, i.e., . This pseudo-identifier has no relation with the true identity of a time series. 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 would occur if a competition participant (or any other third party) were to correctly predict the identity of one or more of the time series in the M3 data set based on the protected time series and some original time series values which the third party possesses. For example, identification disclosure would occur if a third party were to correctly make the statement, “Series N1402 comes from [retailer] store number 5249.” For simplicity, we assume the third party does not know which privacy method was applied to the data and knows that the time series of interest is contained in the protected data set.[[5]](#footnote-10)

For brevity, we have included the math behind our definition of identification disclosure in the appendix. The metric we use to measure the risk of identification disclosure, , gives the average proportion of the time series which are correctly identified across simulated privacy attacks:

where is the third party’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 .

While we seek to protect the time series identities, we also want the protected versions of the time series to be realistic. Forecasters prefer forecasts that are representative, i.e., typical of the data used to produce the forecast (Petropoulos & Siemsen, 2022). Likewise, organizations (and forecasters) will prefer using protected data that is representative of the original data, which our method provides through the preservation of time series features. We use the *performance gap* portion of the REP metric 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 transform and scaling both the original and protected series.

**Table 5** contains the average MAE of one-step ahead point forecasts across all models, the identification disclosure metric , and the average performance gap across all series for several privacy parameters for each privacy method. The percentages shown are the percentage increase in average MAE relative to the average MAE from the original data. The average across models for additive noise and differential privacy excludes the VAR model error for AN (s = 1) and DP () as the errors in these cases were over 1000% larger than the error of any other model. This was due to extremely large noise being added to the very end of some time series, causing the VAR forecasts to explode. This problem did not occur for the other forecasting models which did a better job of smoothing out the random noise. For the simulated privacy attacks, we extract random samples of external data containing ten values from each time series.

The results show a negative relationship between forecast accuracy and the strength of privacy protection. While strong differential privacy provides the lowest risk of identification, it more than triples the average forecast error relative to the original data. Essentially unusable forecasts are produced under differential privacy and additive noise unless privacy is quite weak (, or ). For example, under differential privacy with , nearly 50% of series are identified correctly on average, while MAE has increased by just over 31%. Protection against identification disclosure is better under additive noise with about 22% of series are correctly identified on average. But, this comes at further cost to forecast accuracy, which is reduced by nearly 45%. Standard *k*-nTS with *k* = 3 offers a better trade-off – protection against identification disclosure is quite good, since only 2% of series are correctly identified on average, which is similar to the protection from additive noise and differential privacy with and , but with significantly better accuracy. However, the reduction in accuracy relative to the original data is still ~40%. However, *k-*nTS+ with offers similar levels of protection against reidentification (only 3.26% of series correctly identified on average) with a reduction in forecast accuracy of only 14%.

The performance gap values in (Petropoulos & Siemsen 2022) are calculated using the fitted values of forecasting models relative to the training data. We note that these performance gap values are significantly smaller than any shown in Table 4, i.e., the protected time series are nowhere near as similar to the original series as the fitted values from a forecasting model. However, the results show that k-nts+ and *k*-nTS produce protected time series with the smallest performance gaps by a large margin. This is important because organizations need to trust the protection measures before implementing them – since k-nts+ produces plausible time series, this protection is more likely to be used. Further, *k*-nTS+ () produces forecasts which are usable, i.e., accuracy is only reduced by approximately 14%.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Privacy Method** | ***k*-nTS+** | **15** | 2.73% | 839.78  (+22.47%) | 90 |
| **7** | 3.50% | 822.26  (+19.91%) | 82 |
| **3** | 3.26% | 781.02  (+13.90%) | 77 |
| ***k*-nTS** | **15** | 1.57% | 1066.16  (+55.48%) | 127 |
| **7** | 2.12% | 987.04  (+43.94%) | 120 |
| **3** | 2.05% | 956.89  (+39.55%) | 112 |
| **Differential Privacy** | **1.0** | 1.85% | 3310.34  (+382.76%) | 1,930,653 |
| **4.6** | 13.55% | 1400.95  (+104.31%) | 305,396 |
| **10** | 49.03% | 899.38  (+31.16%) | 73,803 |
| **Additive Noise** | **2.0** | 5.84% | 1821.38  (+165.62%) | 489,840 |
| **1.5** | 10.35% | 1343.29  (+95.90%) | 304,482 |
| **1.0** | 22.51% | 993.95  (+44.95%) | 142,095 |
|  |  | **Type** | Privacy (Average Proportion of Identified Series) | Accuracy (Average MAE) | Representativeness (Average Performance Gap) |

In **Table 5,** the models are rank-ordered based on their MAE on the original data. TES and ARIMA, which explicitly model the seasonality of the series, had the best accuracy which is consistent with the findings of the original M3 competition (Makridakis & Hibon, 2000). The results from the protected data set preserved the ranking of the best model and worst models on MAE, while simple models (SES and DES) moved up in the ranking and more complex models (Auto-ARIMA and RNN) moved down.

**Table 5:** the rank of each model in terms of MAE and forecast error variance on the original data vs. the knts+ (*k*=3) data. The rightmost column contains the average of the protected MAE and error variance ranks.

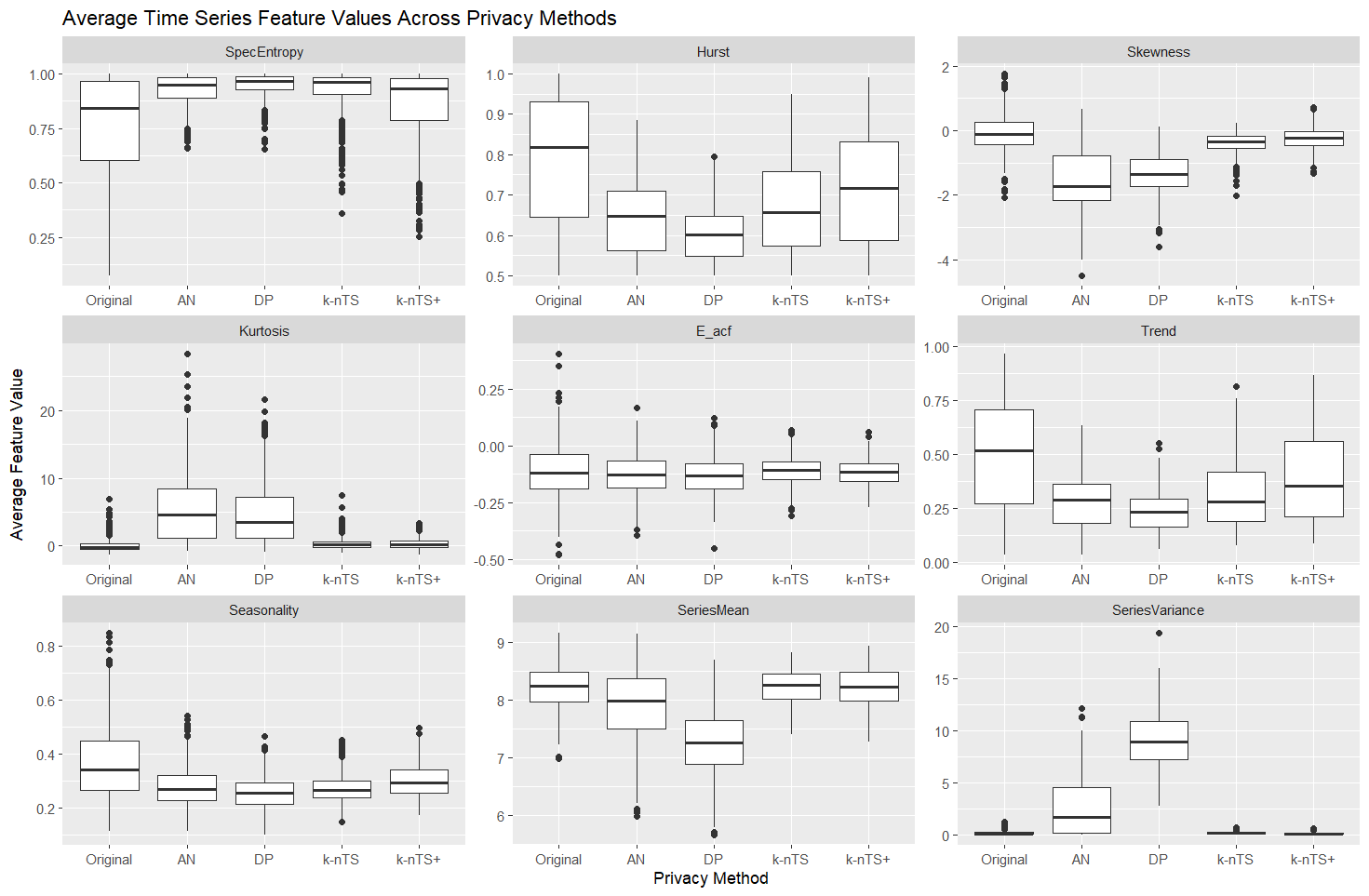
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | MAE Ranks | | Forecast Error Variance Ranks | | Average Rank |
| Model | Original | Protected | Original | Protected | Protected |
| TES | 1 | 1 | 2 | 4 | 2.5 |
| ARIMA | 2 | 4 | 1 | 1 | 2.5 |
| RNN | 3 | 5 | 5 | 5 | 5 |
| DES | 4 | 2 | 3 | 2 | 2 |
| SES | 5 | 3 | 4 | 3 | 3 |
| LGBM | 6 | 6 | 7 | 6 | 6 |
| VAR | 7 | 7 | 6 | 7 | 7 |

We also display the rank of each model based on forecast error variance in Table 5. TES and ARIMA had the lowest error variance on the original data, and ARIMA maintained this ranking on the protected data. TES, however, fell behind simpler methods (DES and SES). The error variance rankings of RNN, LGBM, and VAR remained the lowest. Overall, simpler models such as DES tended to perform the best on the protected version of this data.

* 1. *Changes in Time Series Features*

In **Figure 8**, we calculate the average feature value for each series across the protected datasets for each privacy method. We plot these distributions next to the distribution of each feature from the original data.

**Fig 8: distributions of average time series features across protected data sets for each privacy method.**



Random noise protection (AN and DP) methods not only increase the randomness of time series, they significantly change distributional characteristics, leading to poor forecast accuracy. For example, our forecasts were generated using the log-transformed series, and we see that random noise protection produces a negative bias in the means of the protected series and significantly increases the variance.

On the other hand, the *k*-nTS and *k*-nTS+ methods increase the randomness of time series but better preserve their feature distributions. The feature distributions of *k*-nTS+ are closer to the original distributions than for any other privacy method, which is consistent with the previous forecast accuracy results. We note that while the base *k*-nTS method performed swapping based on the values of *SpecEntropy, Hurst*, and *Seasonality*, it does not perform as well as *k*-nTS+ at preserving the distributions of these features, even though *k*-nTS+ did not explicitly swap based on the values of these features. While they correlate with forecast accuracy, *SpecEntropy* and *Hurst* were actually eliminated in the first stage of the *k*-nTS+ feature selection process using RReliefF. Instead, much better feature preservation and forecast accuracy is achieved by swapping using the ‘building blocks’ of time series features which are correlated with overall measures of forecastability.

1. **Conclusions**

This paper has examined forecasting using protected data. The privacy-utility tradeoff is manifest in the reduction in forecast accuracy that occurs when improving the privacy of time series. A substantial portion of the privacy literature is focused on theoretical privacy guarantees, i.e., differential privacy. Our findings agree with past research (Goncalves et al. 2021a) and show that differential privacy (and additive noise) lead forecasting models to generate unusable forecasts at meaningful levels of privacy. This undesirable privacy-utility tradeoff under differential privacy has been demonstrated in other contexts as well. 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 values of that theoretically do not provide meaningful levels of privacy protection. Their experiments found that model regularization (e.g., L2-regulatization) provided comparable privacy protection with better accuracy and lower model learning cost than differential privacy.

Rather than adding random noise to time series, our proposed *k*-nTS+ privacy method uses time series features as a basis for swapping the values between time series. 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 reduction in average forecast accuracy to 14% of the original accuracy and preserved the ranking of the best and worst forecasting models, which was the focus of the competition. Research in other domains found that a 10-15% loss in usability is often a best-case scenario under privacy protection (Schneider et al. 2018). Further, the proposed privacy method provides comparable levels of privacy to differential privacy at meaningful levels of while enabling models to produce usable forecasts.

Our goal is to increase the ease with which organizations can protect their data. The proposed *k*-nTS+ method reduces the frictions of implementing privacy protection since organizations need only select an appropriate value for *k* and apply the method to their data. Decentralization based methods, such as those in (Gonçalves et al., 2021a; Goncalves, Bessa, et al., 2021b; Sommer et al., 2021) are effective but require a more complicated decentralized framework. Further, our method enables entire data sets containing protected time series to be shared, rather than just parameter estimates or forecasts.

While we showed that *k*-nTS+ preserves data utility for forecasting, future work should examine the utility of protected time series for other use cases, such as classification. Future work should also assess forecasting with protected data using multiple forecast horizons. A theoretical examination of forecasting model performance on protected data would help us further understand which models will perform well under different conditions. This would help improve model selection procedures when forecasting with protected data since we are seek a model that forecasts well for unseen data, not necessarily the protected data. Future work should also assess whether forecast combinations, which tend to improve accuracy on unprotected data (Makridakis et al. 2018), are also beneficial when forecasting with protected data.

**References**

Abowd, J. M., Gittings, K., McKinney, K. L., Stephens, B. E., Vilhuber, L., & Woodcock, S. (2012). *Dynamically consistent noise infusion and partially synthetic data as confidentiality protection measures for related time-series*. 41.

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

Goncalves, 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

Goncalves, 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.

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

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

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

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.

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**

**Mathematical Details of Identification and Attribute Disclosure**

To perform identification disclosure, we assume a third party possesses some original data pertaining to a unit of interest in the protected dataset. For the above example, this would be some sequence of original daily sales quantities for a known retailer. Denote this original data which contains a direct identifier (e.g., the identity of retailer ) and original data which contains a sequence of values which 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 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 sequential values are sampled 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 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-8)
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-9)
5. We note that there are other privacy leaks such as attribute disclosure (Li et al. 2007) or membership inference (Shokri et al. 2017). Identification disclosure is the most applicable for our data, and we consider this a steppingstone to additional privacy leaks – e.g., identifying a time series within a protected data set enables a third party to learn unknown information with greater certainty. [↑](#footnote-ref-10)