The Effects of Privacy Protection on Forecast Accuracy

Cameron D. Bale, Matthew J. Schneider, Jinwook Lee

Drexel University Lebow College of Business

3220 Market St, Philadelphia, PA 19104

Abstract

Forecasts generated using protected time series change significantly from those using the original time series. While prior experiments have demonstrated severe degradations in forecast accuracy from a VAR model applied to differentially private time series, little is known about how privacy protection affects other forecasting models. We measure the effects of several data protection methods (top and bottom coding, additive noise, differential privacy, and cluster-based swapping) on both simple and complex forecasting models. We find that data protection degrades forecast accuracy the majority of the time regardless of forecast horizon. Surprisingly, when the time series are protected with differential privacy or additive noise, we find that exponential smoothing models have better accuracy than LGBM models for all forecast horizons. We investigate the reasons behind these results and offer guidance for practitioners in selecting a forecast model for privacy protected time series data.

Keywords: Etc., Etc., Etc.

1. Introduction

Forecasting is popular in a variety of fields, such as consumer analytics, renewable energy and power industries, and census tracking, all of which may benefit from the use of commercially or personally sensitive data. Examples include using social media information (Boone et al., 2019) and collaboratively shared power generation data (Gonçalves et al., 2021) to improve forecast accuracy. The privacy concerns associated with sensitive data have been demonstrated across many domains. Data ranging from consumer locations (De Montjoye et al., 2013) to smart meter usage (Véliz & Grunewald, 2018) can be used to identify individuals and/or infer sensitive information about them. The increased attention on data privacy in recent years has spurred the introduction of a large number of privacy laws such as the General Data Protection Regulation (GDPR)[1] which require organizations to protect their sensitive data to avoid fines.

In general, organizations have two broad categories of techniques to use to protect their data. The first category, which we call *security measures*, contains means of restricting access to sensitive data. The second category, which is central to the focus of this paper, consists of *privacy methods* which alter the sensitive data to produce protected data which limits the ability of a bad actor to identify data subjects or learn sensitive information about them. This second category may be particularly attractive to organizations since when applied correctly, the data produced

[1] For examples in the United States, see this map.

by these privacy methods can be exempt from privacy laws[2]. The concern for forecasters is that protection methods can drastically alter the data used to obtain forecasts, leading to changes in forecasted values which affects forecast accuracy. To limit any negative impacts on forecast accuracy, forecasters must adapt to using protected data.

Unfortunately, there has been relatively little work on data privacy and forecasting. While it has been demonstrated that differential privacy degrades forecast accuracy for VAR models and recurrent neural networks (RNNs) (Gonçalves et al., 2021; Imtiaz et al., 2020), there is no work which compares how multiple forecasting models perform on data protected using various protection methods, which is one of the focuses of this paper. 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. In terms of forecasting with protected data, our goals are two-fold. First, since forecasters may not have knowledge of, or be able to discern how, data was protected, we provide results to help forecasters choose a forecasting model which is likely to have the best performance regardless of the applied protection method. Second, for cases where forecasters do have knowledge of the protection method, they can select a model which has been demonstrated to perform well under that particular protection method.

In addition to reporting the accuracy of multiple forecasting models applied to various types of protected data, we provide analyze the reasons behind the changes in forecast accuracy for each model. Specifically, we examine how important time series characteristics such as spectral entropy give insight into why model performance changes. We also study the behavior of each forecasting model under each protection method to understand why certain models perform better than others on protected data. Lastly, we view the forecasts based on protected data as *privacy adjusted* forecasts, which are closely related to judgmentally adjusted forecasts, and assess the parallels between these two types of forecast adjustments based on time series characteristics and the type of adjustment.

1.1. Privacy Adjusted Forecasts

Forecasts made using protected data are closely related to judgmental forecasting (Petropoulos et al. (2022), sections 2.11.2 and 3.7.3). Judgmental adjustments to forecasts can improve accuracy by accounting for information that was not incorporated into a forecasting model (Fildes et al., 2009). While the reasons for adjusting forecasts may not always be to improve forecast accuracy, judgmental adjustments have been found to improve the accuracy of monthly demand forecasts from statistical models by an average of 10% (Davydenko & Fildes, 2013). When forecasting with protected data, however, the adjustments to forecasts occur indirectly as a result of changes to the data, which are made without regard to the effects on forecast accuracy.

Similar to Fildes et al. (2009), we classify the forecasts based on the protected data as either improving or degrading forecast accuracy. We seek insights that explain the improvement and/or degradation of forecast accuracy from data protection. To do this, we analyze the forecasts based on protected data from two perspectives: (1) How data protection changes time series characteristics which translate into changes in forecast accuracy, and (2) How changes to forecasts are related to changes in forecast accuracy.

[2]See the description of *anonymous information* given in Recital 26 of the GDPR

1.1.1. Changes to Time Series Characteristics

Information with high diagnosticity is useful for forecasting, whereas adding information with low diagnosticity can degrade forecast accuracy (Fildes et al., 2019). When accuracy does improve from judgmental adjustments, the improvements are greater for low volatility time series which are easier to forecast (Fildes et al., 2009). Some forms of data protection involve adding random noise to time series, which is information with low diagnosticity, and is likely to increase the volatility of time series. Other forms of data protection involve truncating the highest and lowest values in the data, which could have an effect similar to adjusting for outliers, which can improve forecast accuracy when the outliers are close to the forecast origin (Chen & Liu, 1993).

There are other time series characteristics we use to give insights into why forecasts change under data protection. Using the approach outlined by Kang et al. (2017), we extract measures of forecastability such as spectral entropy and first order autocorrelation parameters which show fundamental changes to the data from data protection, and help explain why certain models perform better than others on protected data.

1.1.2. Changes to Forecasts

Changes to forecasts are also related to changes in forecast accuracy. When forecasts are judgmentally adjusted, both positive and negative adjustments can improve accuracy, but positive adjustments tend to give only a marginal improvement (Davydenko & Fildes, 2013). Forecast bias can also be reduced by negative adjustments, whereas positive adjustments maintain bias or exacerbate it (Fildes et al., 2009). While the direction of adjustment is purposefully chosen in judgmental forecasting, the direction of privacy adjustments will occur indirectly via adjustments to the data. We will measure the proportions of forecasts that are adjusted positively and negatively under data protection, the respective accuracy for the forecasts in each adjustment group, and whether the changes affected forecast bias.

The size or magnitude of judgmental forecast adjustments is positively associated with the size of accuracy improvements, which can occur when larger adjustments are made by adjusters who are confident in reliable information (Fildes et al., 2009). Under data protection, the adjustment size will again be determined by the forecasting models' responses to data protection. These responses are likely related to the strength of data protection, where stronger data protection results in larger changes to the data. We measure the strength of protection by the magnitude of the change to the data, how this relates to the size of forecast adjustment, and whether these adjustments are related to the change in forecast accuracy from data protection.

1.2. Extant Studies on Data Privacy and Forecasting

Some forecasters have studied data privacy and forecasting in the context of collaborative forecasting. Goncalves et al. (2020) explored a data market where data owners are compensated for sharing their data, and purchase forecasts based on the data from other parties. While data owners have a monetary incentive to share their data, they may be discouraged from doing so due to privacy concerns over sharing data with a central party. 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 in the market. In the absence of a data market, 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. (2021).

Secure multi-party computation was found to be computationally expensive without fully preserving privacy. Decomposition based methods are more attractive, but may reveal sensitive

data to the participating parties. Solutions proposed by Sommer et al. (2021); Goncalves et al. (2021) help prevent data from being revealed. Importantly, Goncalves et al. (2021) accomplish privacy-preserving collaborative forecasting without reductions in forecast accuracy. However, the decomposition-based methods proposed in the literature are not easily extendable to complicated non-linear models such as recurrent neural networks.

The final category of privacy solutions is data transformation techniques which utilize privacy methods to generate protected data. Gonçalves et al. (2021) implement differential privacy based on the work of Dwork & Smith (2010), who show that masking data using properly calibrated noise can achieve differential privacy. Given a confidential time series $\mathbf{A}$, a differentially private time series can be created using a randomized mechanism $\mathcal{M}(\mathbf{A}) = \mathbf{A} + \mathbf{N}$, which adds Laplace random noise $\mathbf{N}$ with scale parameter $\Delta f_1/\epsilon$. The sensitivity Δf_1 is determined as the maximum absolute difference between two time series $\mathbf{A}$ and $\mathbf{A}'$, which differ in at most one observation, where $\Delta f_1 = \max\{\|\mathbf{A} - \mathbf{A}'\|_1\}$. The mechanism $\mathcal{M}$ satisfies ϵ-differential privacy by guaranteeing that, for every output $\mathbf{t}$ of $\mathcal{M}$ and every pair of series $\mathbf{A}$ and $\mathbf{A}'$,

$$\Pr(\mathcal{M}(\mathbf{A}) = \mathbf{t}) \leq \exp(\epsilon)\Pr(\mathcal{M}(\mathbf{A}') = \mathbf{t}). \tag{1}$$

Using this definition of ϵ-differential privacy, even under very high values of the privacy parameter ϵ (weak privacy protection), there were reductions in forecast accuracy for VAR models. Others have also studied the application of differential privacy to time series (Imtiaz et al., 2020; Fan & Xiong, 2012). An interesting result from Imtiaz et al. (2020) is that differentially private data did not always produce worse forecast accuracy when forecasting individuals' health data using a recurrent neural network. Differential privacy adds random noise to time series, which mirrors a technique used to prevent overfitting when forecasting with neural networks (Hewamalage et al., 2022, 2021). We explore whether data protection with random noise can achieve this same regularization, and improve accuracy for recurrent neural networks.

Other privacy methods, known as *additive* and *multiplicative noise*, infuse random noise in the data but without the theoretical privacy guarantees of differential privacy. For example, Abowd et al. (2012) discuss the Census' use of multiplicative random noise to perturb the individual or business level data that goes into calculating Quarterly Workforce Indicator data – but they do not assess the accuracy of forecasting with the perturbed data. More recently, Luo et al. (2018) use multiplicative random noise to simulate a data integrity attack in the context of load forecasting. These authors measure the accuracy of multiple linear regression, artificial neural network, support vector regression, and fuzzy interaction regression models on the original and attacked data. While multiple linear regression performed the best on the original data, the support vector regression was most robust and performed the best under the data integrity attacks.

Another privacy method is *generalization*, where data records are generalized to create equivalence classes of identical records. This privacy method is particularly popular for tabular data. The principle of k-anonymity (Sweeney, 2002) is used to describe when every record (or time series) is identical to at least $k - 1$ 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, and significant reductions in forecast accuracy even for k=2.

There are also privacy methods which have not been studied in the forecasting literature. Top- and bottom-coding are used to replace the top (bottom) p percent of observations with the

$1 - p$ (p) quantile. These techniques are useful for protecting data with sensitive values in the tails of distributions, such as income levels or smart meter data. Crimi & Eddy (2014) study the effect of top coding the Census' Public Use Microdata Samples on analyses of interest. They find that the sample correlation between two variables is shrunk towards zero when one or both of the variables are top coded.

1.3. The Gaps in Data Privacy and Forecasting

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. Finally, there exist no privacy methods which are specifically designed with forecasters in mind.

The contributions of this paper...

2. 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 (Makridakis & Hibon, 2000). Of the 474 series, 18 consist of 67 time periods, 259 consist of 68 time periods, and 197 consist of 125 time periods. Complicated forecasting models are known to forecast more accurately than simple models using the unprotected version of this data (Koning et al., 2005), and models that explicitly model trend and seasonality performed the best in the overall M3 competition (Makridakis & Hibon, 2000). We are interested in whether these results still occur when using protected versions of the data.

3. Protection Methods

At time t, the data provider uses a data protection method to release protected data $P_{j,t}$ for each time series j based on the confidential values of all time series up until time t. Standard DPMs previously mentioned in the introduction are formulated in Table 1. As Table 1 shows, the main issue with these methods is that they choose protected values based on predefined rules and not absolute change to forecasts. However, the goal of the data provider should be to change $A_{j,t}$ to $P_{j,t}$ with the minimal absolute change to forecasts while also increasing privacy to an acceptable threshold.

In this section, we design the k-nTS (k-nearest time series) Swapping method to alter the confidential data using randomization to balance the trade-off between absolute change to forecasts and privacy. Depending on the quantity of available data, both k-nTS Swapping and standard DPMs can use rolling windows of data. Rolling windows adjust for dynamic changes in the relationships between time series. For example, if we choose a rolling window size, say n, then $x_j = (A_{j,t-n+1}, A_{j,t-n+2}, \ldots, A_{j,t-1}, A_{j,t})^T$ where $x_j \in \mathbb{R}^n$. Also, define $F(x_j)$ as the empirical cumulative distribution function of x_j. Protection in subsequent time periods from $t+1$ to T rolls x_j forward from $x_j = (A_{j,t-n+2}, A_{j,t-n+3}, \ldots, A_{j,t}, A_{j,t+1})^T$ to $(A_{j,T-n+1}, A_{j,T-n+2}, \ldots, A_{j,T-1}, A_{j,T})^T$, respectively. Compared to standard DPMs, k-nTS Swapping matches similarly patterned time series together based on the distance of a rolling window of their past values. Then, it uses randomization to replace $A_{j,t}$ with a confidential value from a similar time series.

Table 1: Standard Data Protection Methods at Time t

Data Protection Method	**Description**	**Formulation**
None	Release confidential observation	$P_{j,t} = A_{j,t}$
Bottom-Coding	Bottom p percent of observations are replaced with the p quantile	$P_{j,t} = \begin{cases} B \text{ if } A_{j,t} \leq B \\ A_{j,t} \text{ if } A_{j,t} > B \end{cases}$ where $B = \inf\{x_j \in \mathbb{R} \mid F\left(x_j\right) \geq p\}$
Top-Coding	Top p percent of observations are replaced with the $1-p$ quantile	$P_{j,t} = \begin{cases} T \text{ if } A_{j,t} \geq T \\ A_{j,t} \text{ if } A_{j,t} < T \end{cases}$ where $T = \sup\{x_j \in \mathbb{R} \mid F\left(x_j\right) \leq p\}$
Additive Noise	Add a normal random number with mean zero and standard deviation σ	$P_{j,t} = A_{j,t} + r$, where $r \sim N(0, \sigma^2)$ and $\sigma^2 = E[(x_j - E[x_j])^2]$
Differential Privacy	Add a laplace random number with mean zero and scale parameter $\Delta f_1/\epsilon$	$P_{j,t} = A_{j,t} + r$, where $r \sim Lap(0, \Delta f_1/\epsilon)$, $\Delta f_1 = \max\{\|\|f(\mathbf{A}_j) - f(\mathbf{A}'_j)\|\|_1\}$, and ϵ is given

3.1. The k-nearest Time Series (nTS) Swapping method

Let $\mathbb{X} = \{x_1, \ldots, x_J\}$ be a given set of time series data (n-vectors). For each time series $x_j \in \mathbb{R}^n$, the data provider computes a set of squared distances of the elements of the set $\mathbb{X}$. Let us define $\mathbf{dist}(x_j, x_i) = d_{j,i}$ as the distance between x_j and x_i, i.e., two distinct time series data from a given set $\mathbb{X}$. Without loss of generality, we use the Euclidean norm, or ℓ^2-norm for this paper (all norms on $\mathbb{R}^n$ are equivalent to the Euclidean norm) as a distance metric. Since our case is multivariate and partially ordered, we can get a totally ordered set based on the Euclidean distance.

Let us define $x_j^{(k)}$ as the kth nearest neighbor of x_j. Then, for a time series x_j, we have $\{d_{j,(1)},\ d_{j,(2)}, \ldots, d_{j,(J-1)}\}$ such that $d_{j,(k)} \leq d_{j,(l)}$ for any integers $k < l$ where $d_{j,(k)} = \|x_j - x_j^{(k)}\|$. Note that $x_j^{(i)} \in \mathbb{X}\backslash\{x_j\}$ and the superscript (i) means the i^{th} order statistic of the related Euclidean distances from x_j. Thus, for a given time series vector x_j, its k-nTS (k-nearest time series) can be represented as the set $K_j = \{x_j^{(1)}, \ldots, x_j^{(k)}\}$ based on $\|x_j - x_j^{(1)}\| \leq \cdots \leq \|x_j - x_j^{(k)}\|$ or an ordered set $\{d_{j,(1)},\ d_{j,(2)}, \ldots, d_{j,(k)}\}$.

For more efficient computation for such ordering, let us introduce a distance matrix D using the squared distances. The squared distance between x_i and x_j is given by $d_{i,j} = \|x_i - x_j\|^2$, and $d_{i,j}$ is the (i, j)th entry of D. Hence D is symmetric. (Also note that $\text{rank}(D) \leq n + 2$.) For our applications, the vector $x_j \in \mathbb{R}^n$, $j = 1, \ldots, J$ where $n \ll J$, typically. Suppose that we are given a data matrix $X = [x_1, x_2, \ldots, x_J]$, $x_j \in \mathbb{R}^n$ (i.e., $X \in \mathbb{R}^{n\times J}$). We can write such data matrix X, i.e., a confidential data matrix, as the following.

$$X = [x_1, x_2, \ldots, x_J] = \begin{pmatrix} A_{1,t-n+1} & A_{1,t-n+2} & A_{1,t-n+3} & \cdots & A_{1,t-1} & A_{1,t} \\ A_{2,t-n+1} & A_{2,t-n+2} & A_{2,t-n+3} & \cdots & A_{2,t-1} & A_{2,t} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ A_{J,t-n+1} & A_{J,t-n+2} & A_{J,t-n+3} & \cdots & A_{J,t-1} & A_{J,t} \end{pmatrix}^T, \tag{2}$$

where $x_j = (A_{j,t-n+1}, A_{j,t-n+2},\ \ldots, A_{j,t-1}, A_{j,t})^T$ where $x_j \in \mathbb{R}^n$ and also $x_j \in \mathbb{X}$.

Note that we can find the distance matrix D using the fact that $\|x_i - x_j\|^2 = (x_i - x_j)^T(x_i - x_j) = x_i^T x_i - x_i^T x_j - x_j^T x_i + x_j^T x_j$, which can be written up as the following:

$$D = \mathbf{1}diag(X^T X)^T - 2X^T X + diag(X^T X)\mathbf{1}^T, \tag{3}$$

where the symbol $\mathbf{1}$ denotes a column vector of J ones. It is easy to see that the column vector $diag(X^T X) = (\|x_1\|^2, \ldots, \|x_J\|^2)^T$. Let d_j denote the jth column of D. Then we can write the $J \times J$ distance matrix $D = [d_1, \ldots, d_J]$, where $d_j \in \mathbb{R}^J$.

In the general case $k \ll J$, for each time series x_j we sort d_j, the jth column of D from the smallest to largest components and find the kth smallest component so that we have

$$K_j = \{x_j^{(1)}, \ldots, x_j^{(k)}\}. \tag{4}$$

That is, the data provider then selects a value of k from 1 to a maximum of $J - 1$ and selects the k-nearest time series. In case of $k = J$, random swapping is simply done by rearranging the components in the last row of matrix X. Let the ith nearest time series from x_j be $x_j^{(i)} = (A_{j,t-n+1}^{(i)}, A_{j,t-n+2}^{(i)}, \ldots, A_{j,t}^{(i)})^T$ where n is the length of the rolling window of past data. Then the swap of the last component of x_j with the last component of one of its k-nearest time series $x_j^{(i)}$,

Algorithm 1 The k-nTS Swapping method

Require: [Initialization]

(i) [Time Series Matrix $X \in \mathbb{R}^{n\times J}$] $X = [x_1, x_2, \ldots, x_J]$, $x_j \in \mathbb{R}^n$ for $j = 1, \ldots, J$ as in (2).

(ii) [Distance Matrix D] $D = \mathbf{1}diag(X^TX)^T - 2X^TX + diag(X^TX)\mathbf{1}^T$ as in (3).

for $j = 1, 2, \ldots, J$ **do**

[Finding a set K_j for x_j] Let d_j denote the jth column of D. Sort d_j from the smallest to largest components and find the kth smallest component, followed by K_j as in (4).

[Random swapping] $x_j \leftarrow x_j^{(i)}$ (last components only) for some $i \in \{1, \ldots, k\}$ as in (5).

end for

$i = 1, \ldots, k$ can simply be written as the following random swapping:

$$k - \text{nTS Swapping}: \; P_{j,t} = \begin{cases} A_{j,t}^{(1)} \text{ with probability } \frac{1}{k} \\ \qquad\vdots \\ A_{j,t}^{(k)} \text{ with probability } \frac{1}{k}, \end{cases} \tag{5}$$

which is equivalent to the following: the last component of x_j is randomly replaced by the last component of $x_j^{(i)} \in K_j$ with probability $\frac{1}{k}$ for $i = 1, \ldots, k$.

By Algorithm 1, we can obtain X': 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

$$X' = [x_1', x_2', \ldots, x_J'] = \begin{pmatrix} A_{1,t-n+1} & A_{1,t-n+2} & A_{1,t-n+3} & \cdots & A_{1,t-1} & P_{1,t} \\ A_{2,t-n+1} & A_{2,t-n+2} & A_{2,t-n+3} & \cdots & A_{2,t-1} & P_{2,t} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ A_{J,t-n+1} & A_{J,t-n+2} & A_{J,t-n+3} & \cdots & A_{J,t-1} & P_{J,t} \end{pmatrix}^T. \tag{6}$$

Remark 1 (Time complexity)**.** *Let $\mathbb{X} = \{x_1, \ldots, x_J\}$ be a given set of time series data (n-vectors). The time complexity to find k-nearest neighbors of each one of x_j's can be calculated by the following. First, the cost for the computation of distances $\|x_i - x_j\|$ for $j = 1, \ldots J$, $j \neq i$ is $3(J-1)n$ ($2(J-1)n$ if we use squares of distances), in either case it would be $O(Jn)$. In order to find the k-nearest neighbors we need to sort the calculated distances and it would be $O(J \log J)$. Since we typically have $J \gg n$, the cost of computation is $O(J \log J)$ for each x_j. If we use k-d tree algorithm (e.g., see* **?** *and references therein), the whole process can be done in $O(J \log J)$.*

In Figure 1, for a given time series x_3, its last component is replaced by the last component of x_9, that is randomly selected among x_3's 10-nearest time series data. Using our notation, we can write

$$K_3 = \{x_1, x_2, x_4, x_6, x_7, x_9, x_{14}, x_{16}, x_{21}, x_{27}\}.$$

We can represent each time series x_j, $j = 1, \ldots, J$ as a vector, and then put them in the graph $G = (V, E)$, 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 vectors: $w_{i,j} = \mathbf{dist}(x_i, x_j)$. If we put all the nodes on the graph and assign weight on every edge (every pair of nodes), e.g., $w_{i,j} = \mathbf{dist}(x_i, x_j)$ for all $i \neq j$, then we will have a complete graph. As described in Figure 1, the k-nearest time series swapping method can be considered as a random edge selection problem of the graph. Figure 1 depicts the case of $k = 10$ for the k-nearest time series of x_3, where the last component of x_3 is swapped with that of x_9.

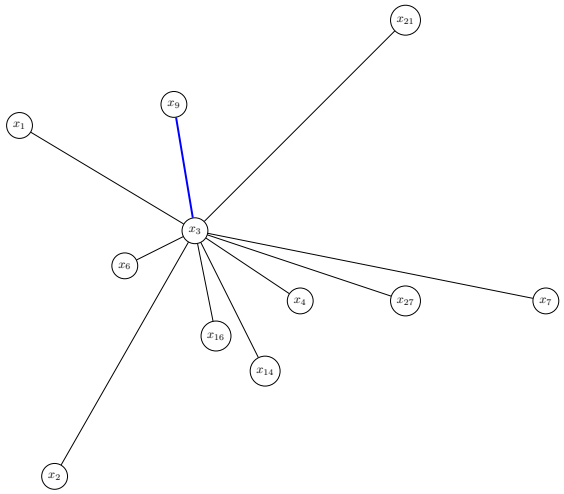

Figure 1: Random edge selection for k-nTS Swapping

3.2. Optimal time series selection for the k-nTS swapping using fixed-radius δ

As described in Figure 1 in Section 3.1, the k-nTS swapping can be considered a random edge selection problem for given k neighbors of each time series $j = 1, \ldots, J$. Since the selection of k plays an important role in both forecasting loss and data privacy perspectives, we need to be able to find an optimal k (i.e., optimal set K) before random swapping. For each time series $j \in J$, we can find the optimal k by the following :

$$\begin{aligned} &\max k \\ &\text{subject to} \\ &\|x_j - x_{(k)}\| \leq \delta \\ &\|x_j - x_{(k)}\|_\infty \leq M, \end{aligned} \tag{7}$$

where $\delta > 0$ and $M > 0$ are given constants determined by a data provider. The first constraint is to find k neighboring time series of $x_j \in \mathbb{R}^n$ within a given radius δ from x_j, and the second constraint requires to find time series in such a way that maximum of absolute values of components of a vector $x_j - x_{(k)}$ is bounded above by M. Let $\mathbf{x}^{(j_k)} = x_j - x_{(k)}$. Then the second constraint can be written up as $\|\mathbf{x}^{(j_k)}\|_\infty = \max_i |\mathbf{x}_i^{(j_k)}| \leq M$, where $\mathbf{x}_i^{(j_k)}$ denotes the ith component of a vector $x_j - x_{(k)}$. This means that, using (7), data providers can put an upper bound M for absolute change to forecasts because the second constraint is equivalent to $|A_t - P_t| \leq M$ for all $t = 1, \ldots, n$.

This problem formulation always provides us with the best k with a given δ. This problem formulation gives us different k for each time series $j = 1, \ldots, J$, which is ideal, since k-nearest neighbors of x_j may include points too far from most of neighbors of x_j. This is because the set K_j for x_j (as defined in (4)) is based on the ordering, no matter how far from the given data point. Swapping with such irrelevant time series for data protection may result in an unacceptable

Algorithm 2 The fixed-radius δ-nTS Swapping method

Require: [Initialization] Same as in Algorithm 1.
for $j = 1, 2, \ldots, J$ **do**
[Component sorting] Sort d_j from the smallest to largest, and let $D_j = \{d_j^{(1)}, \ldots, d_j^{(J-1)}\}$.
[Finding δ-near time series] Find $d_j^{(k)} = \sup\{d \in d_j \mid d < \delta^2\}$ and let $B_j = \{x_j^{(1)}, \ldots, x_j^{(k)}\}$.
[Random swapping] (last components) $x_j \leftarrow x_j^{(i)} \in B_j = \{x_j^{(1)}, \ldots, x_j^{(k)}\}$ with probability $1/k$.
end for

absolute change to forecasts level. For this reason, it would be useful to set an upper bound on the distance from x_j only to include neighbors with acceptable absolute change to forecasts levels.

Using the best k from (7), we do a random edge selection from a fixed-radius δ-near time series (as in Algorithm 2). Let us propose to topologize n-dimensional Euclidean space using open ball as the following:

$$\text{Fixed-radius } \delta\text{-near time series } x_i \in B_\delta(x_j) = \{y \in \mathbb{R}^n \mid \mathbf{dist}(x_j, y) < \delta\}, \tag{8}$$

where $x_i, x_j \in \mathbb{R}^n$ but $1 \leq i \neq j \leq J$. There is no problem also using subspace topology in case of weighted time series data only on certain time points. Figure 2 depicts the time series data within $B_\delta(x_3) = \{y \in \mathbb{R}^n \mid \mathbf{dist}(x_3, y) < \delta\} = \{x_4, x_6, x_9, x_{14}, x_{16}\}$, with which x_3 can be randomly swapped (summarized in Algorithm 2).

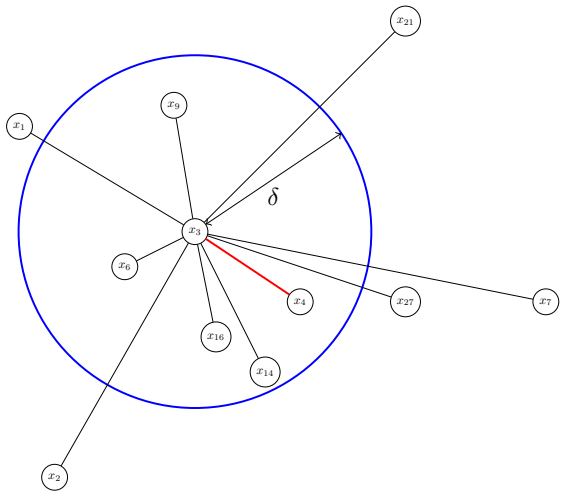

Figure 2: Fixed-radius δ-near time series

In Figure 2, the set $\{x_4, x_6, x_9, x_{14}, x_{16}\}$ consists of the time series within $B_\delta(x_3) = \{y \in \mathbb{R}^n \mid d(x_3, y) < \delta\}$. Note that we use the same example of Figure 1 that is the case of $k = 10$ for the k-nearest neighbors of x_3.

Remark 2 (A centroid as an artificial time series)**.** *It is also reasonable to swap with artificial*

time series data instead of a confidential data. Let K_j denote the set of k-nearest neighbors of x_j, i.e., $|K_j| = k$. We can use a centroid of the set of vectors, say c_j for swapping with x_j at time i. That is, we have the protected data matrix $X' = [x'_1, \ldots, x'_J]$, where $x'_j = (x_{j,1}, \ldots, x_{j,n-1}, c_{j,n})^T$ for $j = 1, \ldots, J$ for the confidential data matrix $X = [x_1, \ldots, x_J]$.

For description (see Figure 3), we consider, among others, three different representatives regarding K_j, i.e., the set of k-nearest neighbors of x_j: (i) centroid of the vertices of conv(K_j), the convex hull of K_j; (ii) centroid of the vectors in the K_j; (iii) centroid of the vectors in the $B_\delta(x_j)$, the set of δ-near time series of x_j. Note that there are a few other representatives for swapping, e.g., Chebyshev center, weighted mean center (based on the distances from the given point), etc. Swapping of the given point with an artificial data point such as a centroid removes the possibility of poor selection of a node which locates farther than most neighbors. This keeps absolute change to forecasts at an improved bound while protecting the given time series with randomization.

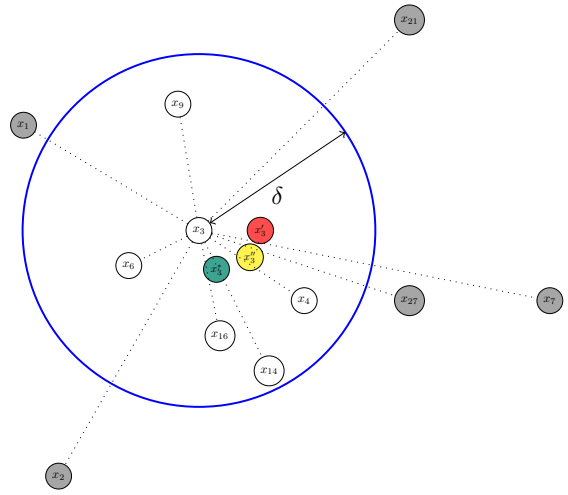

Figure 3: Alternative representatives for the group of neighbors for swapping

In Figure 3, $k = 10$ for the k-nearest neighbors of x_3, $K_3 = \{x_1, x_2, x_4, x_6, x_7, x_9, x_{14}, x_{16}, x_{21}, x_{27}\}$ (i) a red node: x'_3 = centroid of conv(K_3) (ii) a yellow node: x''_3 = centroid of $K_3 \cup \{x_3\}$ (iii) a green node: x^*_3 = centroid of $B_\delta(x_3) \cup \{x_3\}$.

4. Forecasting Models

In this section, we introduce the forecasting models which we apply to the original and protected data. The models 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. For all models, 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. Please see the appendix for full implementation details, including hyperparameter optimization for the complex models.

4.1. Simple Models

We consider four simple models. The first three are common variants of exponential smoothing: (1) single exponential smoothing; (2) double exponential smoothing with an additive trend; and (3) triple exponential smoothing with additive trend and seasonality components. Our final simple model is auto-arima which automatically selects the best seasonal ARIMA model parameterization. All of the simple models were implemented in Python using the sktime forecasting module (Löning et al., 2022).

4.2. Complex Models

We also consider three complex models. The first is a variant of the VAR model known as VARX-L (Nicholson et al., 2017), which we fit using the BigVAR package in R. Fitting large VAR models is already computationally expensive since the number of estimated coefficients is $K + pK^2$, where K is the number of series and p is the number of lags (Hyndman & Athanasopoulos, 2018). The problem is compounded in our case since we have fewer time periods than series. We use the VARX-L model which implements the Lasso penalty to induce sparsity in the coefficients and enable model estimation, and fit one VAR model to each subset of series with the same length, resulting in three VARX-L models being estimated per data set.

The second complex model we consider is a light gradient boosted machine (LGBM) (Ke et al., 2017), which was the top performing model in the M5 competition (Makridakis et al., 2022). This is a global forecasting model trained on all time series in our data set. We use the LGBM implementation provided in the Darts python module (Herzen et al., 2022).

Our final complex model is an RNN with long short-term memory cells (LSTM). This model was a component in the winning solution to the M4 competition (Smyl, 2020; Makridakis et al., 2018). Similar to the LGBM, the RNN is a global forecasting model, and we used the implementation provided in Darts (Herzen et al., 2022).

5. Analysis Framework

We first measure the point forecast accuracy for each forecasting model applied to the original and protected datasets, and examine the distributions of the forecast errors.

Next, we seek to determine why accuracy changed for each model using time series characteristics and examining changes in the models when applied to protected data relative to their unprotected counterparts.

Finally, we relate our findings to the literature on judgmental forecasting by examining the cases where privacy adjustments improved vs. degraded forecast accuracy.

6. Results

6.1. Point Forecast Accuracy and Error Distributions

We first examine the point forecast accuracy and error distributions for forecasts based on top-coded data. In Table 2, we see the MAE for each model on the original and top-coded datasets.

Model	Original	Top (10%)	Top (20%)	Top (40%)
SES	686.71	678.27	694.71	748.84
DES	680.54	671.48	682.41	736.33
TES	**637.90***	**627.09***	**642.72***	**716.24****
Auto-ARIMA	**647.34****	**631.61****	**653.35****	**714.68***
VAR	729.57	711.89	707.07	734.03
LGBM	700.89	694.93	705.39	772.65
RNN	671.28	671.05	694.50	749.51

Table 2: Original MAE vs. MAE under Top-Coding.* = best accuracy ** = second best accuracy

Unsurprisingly, TES (which explicitely models trend and seasonality of each series) outperforms the rest of the models on the original data, and is closely followed by auto-ARIMA. These models outperform the more complex LGBM and RNN models, although this is to be expected given that we did not apply advanced pre-processing techniques which can substantially improve the accuracy of machine learning models.[3]

In terms of the effect of top-coding on forecast accuracy, we see some remarkable results. When forecasting with 10% top-coded data, the MAE for each model improves (although the improvement for RNN is marginal). Typically, data users face a trade-off between the privacy and utility of data, where increasing privacy reduces data utility. In this case, some privacy is obtained for free while actually improving forecast accuracy. Stronger levels of top-coding tend to reduce forecast accuracy. Overall, TES and auto-ARIMA performed the best on this data under all levels of top-coding.

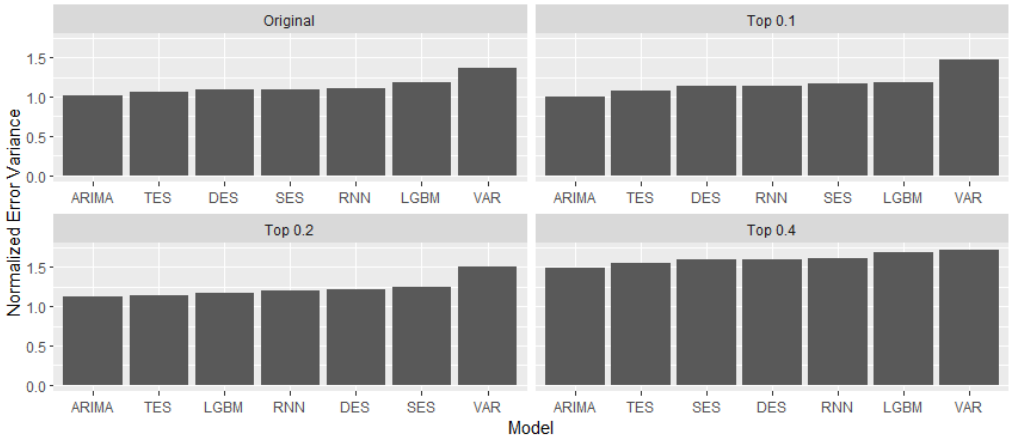

Figure 4: Model rankings for each top-coded dataset based on normalized forecast error variance.

[3]See the appendix for results using LGBM and RNN with advanced pre-processing techniques.

To compare forecast error distributions, we consider the variance of the forecast errors as a measure of forecast quality. Models with lower forecast error variance will be more desireable since forecasters can have higher confidence that they have not generated forecasts for some series which are significantly worse than others. For the forecasts for each model for each level of top coding and the original data, we calculate the variance of the forecast errors (MAE values) and normalize the variances by the lowest variance on the original data, which was for the auto-ARIMA forecasts. Figure 4 shows these normalized variances, where the models in each quadrant are arranged in ascending order. Auto-ARIMA and TES continue to offer the best performance, as these models have the lowest error variance and lowest mean absolute errors. Notably, the VAR model performs partcularly poorly, producing some of the highest mean forecast errors, and the highest forecast error variance. Another important finding is that the increase in error variance for all models is at least 50% compared to the forecast error variance for auto-ARIMA on the original data. So, while the increase in mean forecast error appears palatable, there will be much more variance in the quality of the forecasts from strongly protected data.

6.2. Changes in Time Series Characteristics

To help determine why model performance changes under top-coding, we examine the distributions of four time series characteristics which are closely related to the ability to forecast accurately: the spectral entropy, which measures the noise-to-signal ratio in the data (higher values indicate that series are more difficult to forecast), the strength of trend and seasonality, and the first order auto-correlation parameter. All features were extracted using the tsfeatures package in R (Hyndman et al., 2022).

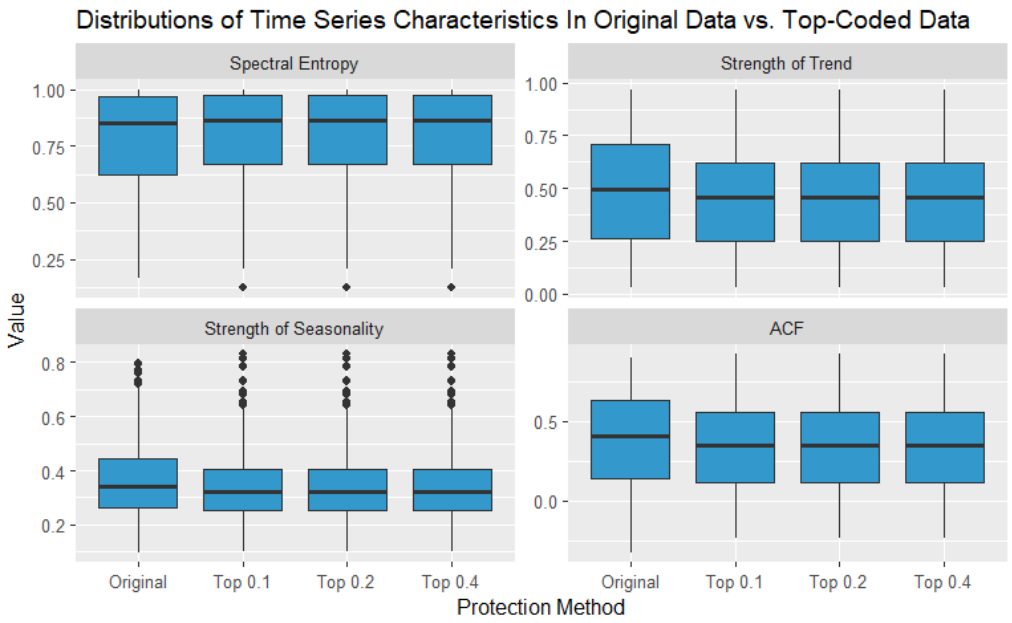

Figure 5: Distributions of time series characteristics in the original and top-coded data sets.

Overall, as the strength of data protection increases, we do not observe large changes in these characteristics, which is consistent with the relatively small reductions in forecast accuracy from top coding. However, we do see a small upward shift in the spectral entropy distributions and small downward shifts in the strength of trend and seasonality and the auto-correlation parameter, all of which would indicate a reduced ability to forecast the time series accurately. Yet, the mean accuracy for all models improved slightly under 10% top coding.

7. Assessing the Value of Privacy Adjustments

We examine the value of privacy adjustments through the changes in time series characteristics and the type of adjustment that was made. Similar to Khosrowabadi et al. (2022), we use a random forest assess which characteristics and adjustment types are predictive of whether an adjustment improved accuracy. For predictors, we include the previously mentioned time series characteristics as well as the direction of adjustment (positive, negative), the magnitude of the adjustment (measured by the absolute change in forecast normalized by the series mean) and indicators for whether the adjustment was in the first or fourth quartile of magnitudes.

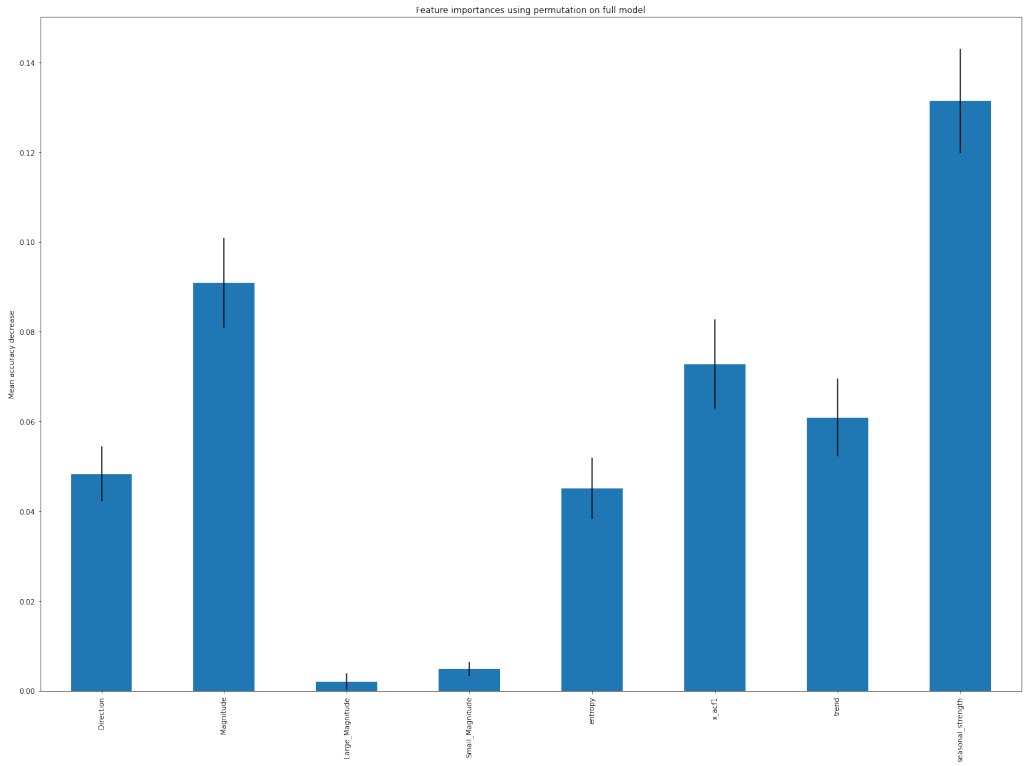

Figure 6: Feature importance from a random forest used to predict privacy adjustment value.

Figure 6 shows the feature importances for predicting whether a privacy adjustment improved accuracy for the TES model applied to data protected using 10% top coding. Seasonal strength

is the strongest predictor, followed by the magnitude of the adjustment. To interpret the interactions between these variables and how they lead to changes in forecast accuracy, we use a decision tree to predict whether a privacy adjustment improved accuracy using the top two most important characteristics and adjustment attributes, namely seasonal strength and the first order autocorrelation parameter, and adjustment direction and magnitude. The structure for this decision tree is depicted in Figure 7, where $\mathbf{X}$ = [Seasonal Strength, ACF, Magnitude, Direction]. We limit the depth of the tree to three, and provide interpretation for the two purest leaves that have more than one series.

We index the leaves from left to right as $i = 0, 1, \ldots, 7$, where we are interested in comparing the series in leaf $i = 2$ to the series in leaf $i = 7$. Most series in leaf $i = 2$ have improved accuracy, whereas most series in $i = 7$ had reduced accuracy.

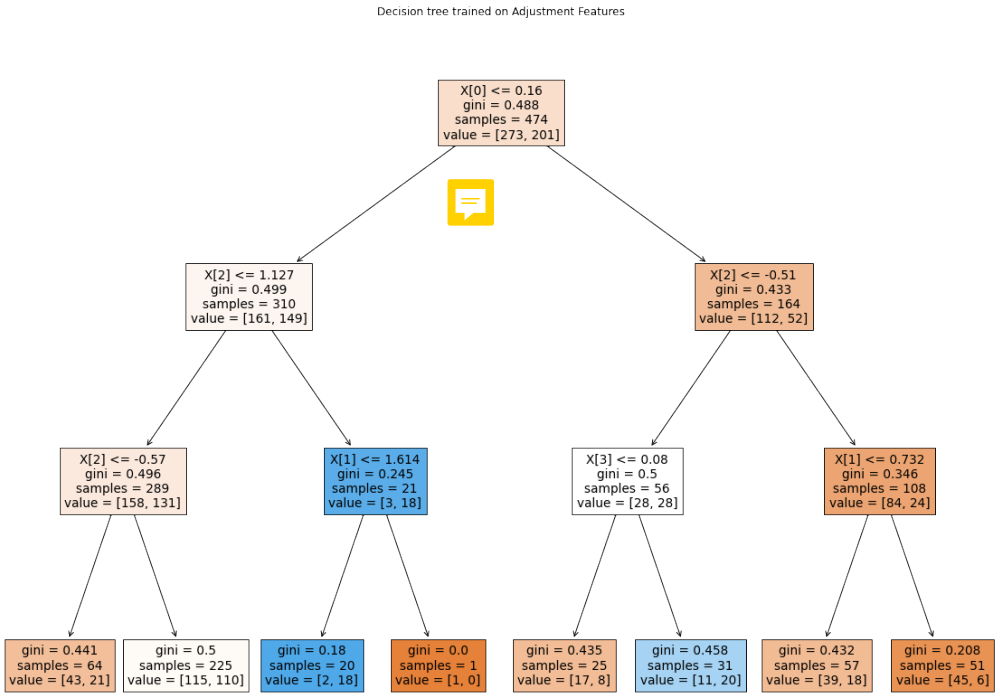

Figure 7: Decision tree structure for predicting adjustment value.

Table 3 compares the mean feature values for the series in the two leaves of interest. Note that Direction, Large Magnitude, and Small Magnitude are all binary features, so the values in the table for those features represent percentages. For both leaves, most of the forecast are adjusted downward, which is intuitive since top coding truncates the highest values in the data. There is a noticeable difference in the magnitude of adjustments between the two leaves - for $i = 2$, the adjustments are much larger on average and all fall into the 4th quartile, whereas only 37% of

adjustments are in the 4th quartile for $i = 7$. In terms of time series characteristics, the average entropy for $i = 2$ is much higher than for $i = 7$, and the ACF and strength of trend and seasonality are much lower for $i = 2$ than for $i = 7$. 10% top coding tends to improve accuracy for TES models by making large, downward adjustments for time series that are difficult to forecast. On the other hand, TES forecasts for series that are much easier to forecast tend to be worse when 10% top coding is applied.

Feature	Leaf $i = 2$	Leaf $i = 7$
Direction	0.35	0.16
Magnitude	0.42	0.19
Large Magnitude	1.00	0.37
Small Magnitude	0.00	0.00
Entropy	0.84	0.52
ACF	0.27	0.70
Strength of Trend	0.41	0.74
Seasonal Strength	0.29	0.59

Table 3: Feature comparison for series with improved accuracy (Leaf $i = 2$) vs. series with degraded accuracy (Leaf $i = 7$).

8. Conclusions and Future Research

9. Conclusion

References

Abowd, J. M., Gittings, R. K., McKinney, K. L., Stephens, B., Vilhuber, L., & Woodcock, S. D. (2012). Dynamically consistent noise infusion and partially synthetic data as confidentiality protection measures for related time series. *US Census Bureau Center for Economic Studies Paper No. CES-WP-12-13*, .

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*, 170–180.

Chen, C., & Liu, L.-M. (1993). Forecasting time series with outliers. *Journal of forecasting*, *12*, 13–35.

Crimi, N., & Eddy, W. (2014). Top-coding and public use microdata samples from the us census bureau. *Journal of Privacy and Confidentiality*, *6*.

Davydenko, A., & Fildes, R. (2013). Measuring forecasting accuracy: The case of judgmental adjustments to sku-level demand forecasts. *International Journal of Forecasting*, *29*, 510–522.

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*, 1–5.

Dwork, C., & Smith, A. (2010). Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, *1*.

Fan, L., & Xiong, L. (2012). Real-time aggregate monitoring with differential privacy. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (pp. 2169–2173).

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*, 3–23.

Fildes, R., Goodwin, P., & Önkal, D. (2019). Use and misuse of information in supply chain forecasting of promotion effects. *International Journal of Forecasting*, *35*, 144–156.

Gonçalves, C., Bessa, R. J., & Pinson, P. (2021). A critical overview of privacy-preserving approaches for collaborative forecasting. *International journal of Forecasting*, *37*, 322–342.

Goncalves, C., Bessa, R. J., & Pinson, P. (2021). Privacy-preserving distributed learning for renewable energy forecasting. *IEEE Transactions on Sustainable Energy*, *12*, 1777–1787.

Goncalves, C., Pinson, P., & Bessa, R. J. (2020). Towards data markets in renewable energy forecasting. *IEEE Transactions on Sustainable Energy*, *12*, 533–542.

Herzen, J., Lässig, F., Piazzetta, S. G., Neuer, T., Tafti, L., Raille, G., Pottelbergh, T. V., Pasieka, M., Skrodzki, A., Huguenin, N., Dumonal, M., Kościsz, J., Bader, D., Gusset, F., Benheddi, M., Williamson, C., Kosinski, M., Petrik, M., & Grosch, G. (2022). Darts: User-friendly modern machine learning for time series. *Journal of Machine Learning Research*, *23*, 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*, 388–427.

Hewamalage, H., Bergmeir, C., & Bandara, K. (2022). Global models for time series forecasting: A simulation study. *Pattern Recognition*, *124*, 108441.

Hyndman, R., Kang, Y., Montero-Manso, P., Talagala, T., Wang, E., Yang, Y., & O'Hara-Wild, M. (2022). *tsfeatures: Time Series Feature Extraction*. URL: `https://pkg.robjhyndman.com/tsfeatures/` r package version 1.0.2.9000.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

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. In *2020 IEEE International Conference on Big Data (Big Data)* (pp. 3428–3437). IEEE.

Kang, Y., Hyndman, R. J., & Smith-Miles, K. (2017). Visualising forecasting algorithm performance using time series instance spaces. *International Journal of Forecasting*, *33*, 345–358.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

Khosrowabadi, N., Hoberg, K., & Imdahl, C. (2022). Evaluating human behaviour in response to ai recommendations for judgemental forecasting. *European Journal of Operational Research*, .

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*, 397–409.

Luo, J., Hong, T., & Fang, S.-C. (2018). Benchmarking robustness of load forecasting models under data integrity attacks. *International Journal of Forecasting*, *34*, 89–104.

Löning, M., Király, F., Bagnall, T., Middlehurst, M., Ganesh, S., Oastler, G., Lines, J., Walter, M., ViktorKaz, Mentel, L., chrisholder, RNKuhns, Tsaprounis, L., Owoseni, T., Rockenschaub, P., danbartl, jesellier, eenticott shell, Gilbert, C., Bulatova, G., Lovkush, Parker, M., Take, K., Schäfer, P., Khrapov, S., Meyer, S. M., AidenRushbrooke, oleskiewicz, Xu, Y.-X., & Ansari, A. (2022). alan-turing-institute/sktime: v0.13.2. URL: `https://doi.org/10.5281/zenodo.7017832`. doi:`10.5281/zenodo.7017832`.

Makridakis, S., & Hibon, M. (2000). The m3-competition: results, conclusions and implications. *International journal of forecasting*, *16*, 451–476.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*, 802–808.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, .

Nicholson, W. B., Matteson, D. S., & Bien, J. (2017). Varx-l: Structured regularization for large vector autoregressions with exogenous variables. *International Journal of Forecasting*, *33*, 627–651.

Nin, J., & Torra, V. (2009). Towards the evaluation of time series protection methods. *Information Sciences*, *179*, 1663–1677.

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. et al. (2022). Forecasting: theory and practice. *International Journal of Forecasting*, .

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, *36*, 75–85.

Sommer, B., Pinson, P., Messner, J. W., & Obst, D. (2021). Online distributed learning in wind power forecasting. *International Journal of Forecasting*, *37*, 205–223.

Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, *10*, 557–570.

Véliz, C., & Grunewald, P. (2018). Protecting data privacy is key to a smart energy future. *Nature Energy*, *3*, 702–704.