Short-term forecasting of monthly reported crime in Chicago with SARIMA, Random Forest, XGBoost, and Prophet.

Economic Forecasting course
Department of Economics and Business Economics
Aarhus University

Name: Filip Mzyk, AU777882

Supervisor: Eric Hillebrand

Date: 15/05/2025

Subject Area: Crime forecasting

Specification: This assignment can be made public

Abstract

This study aims to compare different models and their abilities to forecast monthly reported crime in Chicago. The models compared are SARIMA, Random Forest, XGBoost, and Prophet. It is investigated whether a traditional univariate time series approach can be outperformed by machine learning methods with additional predictors or an automated Prophet method. Comparing error metrics and employing a Model Confidence Set test, it is concluded that the SARIMA models are not outperformed when forecasting one step ahead.

1 Introduction

Advancements in data collection and new forecasting methods, such as neural networks, have led to new opportunities in the domain of crime forecasting (Fitzpatrick et al. 2019). In this paper, models of different architectures are compared. Two specifications of traditional time series SARIMA models, two tree-based machine learning models, in default and tuned settings, Random Forest and XGBoost, and an accessible and automatic additive forecasting model, Prophet by Meta.

This study aims to test whether more sophisticated models can outperform the univariate SARIMA approach in forecasting one-step-ahead reported crime rates in Chicago. Additionally, it verifies whether hyperparameter tuning leads to improved predictive ability of the tree-based models.

In this paper, the data is not split into crime sub-categories but aggregated. The predictors used in the machine learning models are based on economic variables and not spatial values as it is usually done in the crime forecasting literature (Kounadi et al. 2020). This provides an additional insight into the driving forces of crime in the forecasting context from an economic standpoint, as the chosen variables are based on existing economics of crime literature.

Computational constraints had to be taken into consideration when creating the more demanding models. For example, limiting the combinations of the tuning hyperparameters. Nevertheless, the comparisons were constructed to be as fair as possible, supported by the agreed-upon practices in the literature.

The next section consists of a brief literature review of economics and crime, and crime forecasting. Next, the data used in the analysis is presented, followed by a section presenting the models used. A section presenting the results of the forecasting procedure for each model follows. Finally, a discussion of the results, suggestions on future research, and a conclusion are made. Additional graphs and tables are provided in the appendix.

2 Literature Review

In this section a brief literature review is presented. The first part of the review focuses on literature connecting economics and economic indicators to criminal activities. The second part presents some literature on crime forecasting, focusing more on the methods which are utilized.

2.1 Economic indicators and Crime

Economics of Crime is a branch of economics that tries to analyze criminal behavior, assuming that the actors are behaving rationally (Buonanno 2003). Apart from microe-

conomic models, macroeconomic factors and their effect on crime are also inspected. Yet, there seems to be little to no consensus in terms of which of those macroeconomic factors do ultimately affect crime, and how (Winter 2008).

One approach is using unemployment as an indicator of crime, believing that actors choose criminal activities in periods of high unemployment (Winter 2008). However, the magnitude or the significance to which unemployment dictates crime levels is often ambiguous and depends on a particular methodology and data used (Yearwood & Koinis 2011). As a counterintuitive example, it can be possible that being employed can actually lead to criminal activities such as embezzlement (Winter 2008).

An alternative economic variable that seems to be related to crime, and acquisitive crime specifically, is inflation (Rosenfeld et al. 2019). Other economic variables, such as average wage or per-capita personal income are also considered a good alternative to unemployment (Yearwood & Koinis 2011). Establishing a concrete link between such indicators and illicit activities seems almost impossible, due to the sheer amount of variables and causalities present.

2.2 Crime Forecasting

One of the earlier reviews on the literature of crime forecasting was done by Weller et al. (1979) and focuses on traditional time series analysis (like ARIMA or Box-Jenkins). Crime forecasting literature is characterized as insufficient, with research focusing on the determinants of crime rather than prediction. The authors suggest that more sophisticated models should be considered. However, more recent studies conclude that traditional time series methods like ARIMA do perform well in short-term forecasting of crime (Chen et al. 2008).

Due to the rise in computational capabilities, crime forecasting has resurfaced as a research topic (Mandalapu et al. 2023). Applications of machine learning algorithms like XGBoost have been particularly of interest and are shown to perform well in forecasting specific crime categories (Djon et al. 2023). A substantial part of the literature focuses on the spatio-temporal side of crime forecasting, where machine learning methods are employed to handle large amounts of data (Kounadi et al. 2020). Many researchers focus on spatial data with the aim of developing applicable smart policing algorithms that send out alerts to patrolling officers (Elluri et al. 2019). Crime forecasting literature seems to focus on developing methods that can translate to directly applicable methods of preventative policing.

3 Data

The following section presents the data used in the forecasting models. The data consists of reported incidents of crime in Chicago, monthly unemployment rate and consumer price index (CPI) for the Chicago-Naperville-Elgin area, as well as the monthly mean average temperature in Chicago.

3.1 Chicago Crime

The Chicago Police Department, through the Chicago Data Portal, provides a detailed dataset of daily reported incidents of crime from 2001 to present, not including the most

recent seven days. The data includes details such as the type of crime and geographical location. In this paper, only the amount of reported crime is considered.

For the analysis, a time series starting in January of 2001 and ending in January of 2020 is chosen. The raw daily data is cleaned from missing values that were detected across the span of the years. Most of the time, values like the geographical location or description of the crime were the cause of detection. Any reported crime with incomplete data was characterized as missing and removed from the dataset by the method of listwise deletion. The data was then aggregated on a monthly level based on the Date column. Finally, a series of 229 data points was acquired. The visual inspection of the data reveals a highly seasonal pattern and a decreasing trend, which flattens towards the end of the series. A decomposition of the series 6 reveals some spikes in the residuals, indicating outliers. In this study, outliers are considered genuine observations. Thus, no further investigation or attempt to normalize those datapoints is performed.

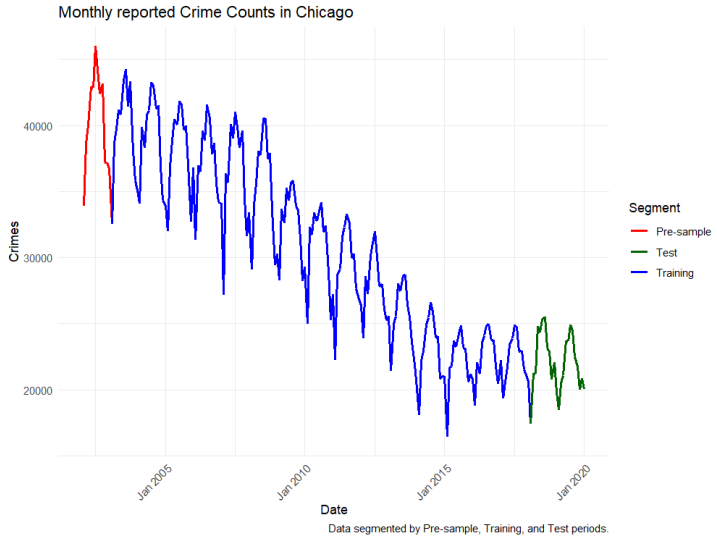


Figure 1: Segmentation of the data

Finally, the data is sectioned into three parts. The first 13 datapoints are characterized as the "Pre-sample" period. This is done to accommodate the expected differentiation of the series needed for the linear models and the lagged values for the tree based models correspondingly. The second 192 datapoints are allocated for the "training" period for each of the models, this is considered as the known datapoints for analysis. Finally, the last 24 datapoints are allocated for the pseudo out of sample testing period, on which the capabilities of each model's one-step-ahead forecasts will be evaluated. The testing period was chosen as a balance between making enough training data accessible to the machine learning models and an adequate testing period where at least two seasonal cycles are present.

3.2 Unemployment

The monthly unemployment rates for the period of 2001 to 2020 for Chicago-Naperville-Elgin were retrieved from FRED. Unemployment data will be used in the tree-based models. The choice for unemployment as one of the predictors stems form the somewhat simplistic hypothesis that unemployment dictates crime activity (Winter 2008). The

predictor will enter only in lagged terms, to avoid lookahead bias. Unfortunately, Chicago specific unemployment rates were not possible to retrieve. Visualization of the series is available in the Appendix 8b.

3.3 Consumer Price Index

The monthly Consumer Price Index (CPI) for the period of 2001 to 2020 for the Chicago-Naperville-Elgin area, for all urban consumers and all items, was retrieved from FRED. CPI, which can be characterized as the inflation that the consumers experience, is considered a better alternative to unemployment in terms of the relationship to crime (Yearwood & Koinis 2011). Similarly to unemployment rates, the CPI will be used in the tree-based models and will enter the models in lagged form. Chicago level CPI was not possible to retrieve. The visualization of the series is available in the Appendix 8a.

3.4 Average Temperature

The monthly mean average temperature for the period of 2001 to 2020 was retrieved from SC-ACIS (Applied Climate Information System) website. The Chicago Midway Airport's meteorological station was chosen as the source of measurements. The average temperature is deemed as a possible metric of seasonality for the tree-based models as well as a variable hypothesized to be related to the amount of crime committed, due to the increased amount of time spent outside the household (Field 1992). The series, as expected, is highly seasonal. Similarly as the other predictors, a lagged value will enter the models. The visualization is available in the Appendix 7.

3.5 Look ahead bias, Data Leakage

Look ahead bias or data leakage refers to the process of training a model on data that should not have been available at the time preceeding the forecasting period, ultimately characterizing the forecasts as not true since the model was already exposed to the said datapoints. This is addressed by creating the aforementioned split of the data. The identification, training, and the choice of hyperparmeters, in the case of Random Forest and XGBoost, is done using only the data prior to the testing split. Additionally, Random Forest and XGBoost models are provided only with the lagged values of the features (for example lagged values of CPI).

4 Methodology

In this section the chosen models are presented. The specific choice of training and testing procedures for each model is also discussed.

4.1 SARIMA

The ARIMA (Autoregressive Integrated Moving Average) model is chosen as a traditional approach to time series forecasting. Additionally, ARIMA models were utilized for the purpose of crime forecasting in the past (Chen et al. 2008). Specifically, a SARIMA (Seasonal ARIMA) is suspected to be the best fit due to the explicit seasonal pattern in the data. The SARIMA models are specified by choosing both a non-seasonal and seasonal

AR, I, and MA components, (p, d, q) and (P, D, Q) correspondingly. Additionally, an s component is chosen, based on the frequency of the data. In the case of monthly data $s = 12$.

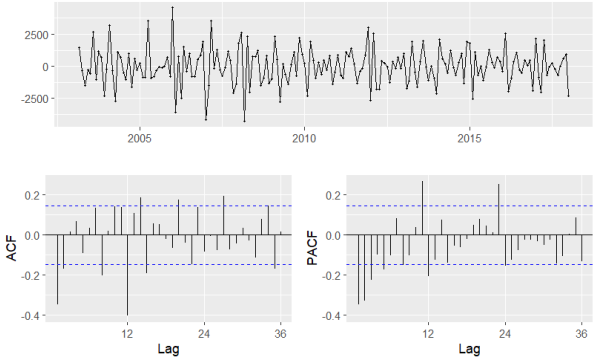
For the SARIMA models to be specified, the data is assumed to be stationary. Apart from the visual inspection performed in the Data section of the study, two formal tests are performed on the training dataset, not including the pre-sample period yet, an Augmented Dickey Fuller (ADF) test for a unit root in the series and a Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test for level stationarity of the series. The ADF test rejects indicating stationarity, but the KPSS test also rejects, indicating a unit root present in the series. An additional check using R-specific functions (*ndiffs*, *nsdiffs*) that determine the amount of non-seasonal and seasonal differencing required for the data to become stationary is performed, both indicating a need for seasonal and first differencing. Finally, seasonal differencing of 12 and first differencing is applied. ADF and KPSS tests are applied again. After the differentiations, the data is concluded to be stationary.

Table 1: Unit-root and stationarity tests for the crime series

Dickey–Fuller test				
Series	t-stat	p-value	lags	Decision
monthlyCrimes	-7.3033	0.01	5	Stationarity
$\Delta_{12}\Delta$ monthlyCrimes	-8.3654	0.01	5	Stationarity
KPSS test (level)				
Series	KPSS	p-value	lags	Decision
monthlyCrimes	3.5762	0.01	4	Unit Root likely
$\Delta_{12}\Delta$ monthlyCrimes	0.0204	0.1	4	Stationarity

The identification of the order of the model can be conducted on the now stationary data. It is important to note that the identification of the models is performed on the training set, without the initial 13 month period.

Four SARIMA models are specified. Three are decided on using the manual approach of investigating the ACF and PACF plots and their patterns (2). Both seasonal and non-seasonal patterns are present, as seen in the spikes of the plots. The fourth model is chosen by the *auto.arima* function in R which tests for various orders and decides on the best one based on information criteria.



The differenced data and the ACF and PACF plots

Chosen SARIMA models in lag operator form

$$\underbrace{(1-\Phi_1 L^{12})}_{\text{seasonal AR(1)}}\ \underbrace{(1-L)\,(1-L^{12})}_{\substack{\text{first non-seasonal}\\ \text{and seasonal differences}}}\ y_t \;=\; \underbrace{(1+\theta_1 L+\theta_2 L^2)}_{\text{non-seasonal MA(2)}}\ \underbrace{(1+\Theta_1 L^{12})}_{\text{seasonal MA(1)}}\ \varepsilon_t, \tag{1}$$

$$\underbrace{(1-\phi_1 L-\phi_2 L^2)}_{\text{non-seasonal AR(2)}}\ \underbrace{(1-L^{12})}_{\text{seasonal difference}}\ y_t \;=\; \underbrace{(1+\theta_1 L)}_{\text{non-seasonal MA(1)}}\ \underbrace{(1+\Theta_1 L^{12})}_{\text{seasonal MA(1)}}\ \varepsilon_t, \tag{2}$$

The models were then fitted on the training data and compared using information criteria, such as AIC. The residuals of the models are inspected. The results are available in the appendix (8). The manual model (1), is kept as it exhibits the lowest information criteria. Additionally, although the model that was chosen automatically (2) exhibits the highest information criteria, it is kept for further analysis on parsimony principle, since only seasonal differencing is done. Moreover, the other two manually specified models do not exhibit white noise residuals.

The two models are fitted on the total of the pre-sample and the training dataset. The Ljung-Box test is applied on the residuals of both models again (2). The null hypothesis of no autocorrelation of residuals fails to reject in both cases. An additional Ljung-Box test on the squared residuals is also performed to check for heteroskedasticity. In both cases the test fails to reject the null hypothesis of autocorrelation in the squares (2).

Table 2: Ljung–Box

(a) Test on residuals

Model	χ^2	p-value	lags	d.f.
SARIMA_m	7.368	0.288	10	6
SARIMA_a	7.112	0.310	10	6

(b) Test on squared residuals

Model	χ^2	p-value	lags	d.f.
SARIMA_m	9.122	0.166	10	6
SARIMA_a	8.321	0.215	10	6

The forecasting procedure is done on an expanding window basis. It can be described in the following steps: (i) The model is fitted on all data prior to the testing split, time T, and forecasts one step ahead. (ii) At $T+1$ the now known value is appended to the training dataset and the model is refitted. (iii) The model forecasts one step ahead again, and the process repeats. Refitting of the model is commonly used in the case of linear models as it is easy to employ and not computationally expensive (Hewamalage et al. 2022). The analysis of the results will follow in the Results section.

4.2 Random Forest

Random Forest is an ensemble learning method introduced in 2001 (Breiman 2001). The method constructs a multitude of decision trees using bootstrap samples and random

feature selection to enhance predictive accuracy. It is an improvement to the lackluster predictive ability of decision trees and the correlated predictions of bagged trees (James et al. 2023). In the regression case, once the ensemble of trees is grown, their predictions are aggregated and averaged 8. As a result, the forecasts are more stable and accurate compared to individual trees.

Random forest is also the first model in this study to use additional predictors, namely the lagged values of Unemployment, CPI, the Average Temperature, and an indicator of the month.

An advantage of the Random Forest is that there are no assumptions made about the stationarity of the data, as each observation will be treated independently. However, for the same reason, and because of the bootstrap sampling architecture, the data needs to be prepared carefully to respect its temporal characteristics. This means transforming the data into a supervised learning structure. In other words, for each observation of the predictors there needs to be a corresponding response measurement (James et al. 2023). A data frame is created and an example of its contents is presented in the following table 3. It is important to note that due to the lags the first 12 observations are lost, similarly to the case of seasonal differencing in the SARIMA models.

Table 3: Target and predictor variables used in the tree based models

N	N_lag_1–12	Un_lag_1–3	CPI_lag_1–3	AvgTemp_lag	Month
Target variable	Crime lags	Unemployment lags	CPI lags	Average-temperature lag	Month indicator

The chosen predictors are the 12 lags of reported crime, due to the autoregressive and seasonal nature of crime, 3 lags of unemployment as to account for the time someone would need to decide on performing an illegitimate activity, and 3 lags of CPI where similar logic to the lags of unemployment is applied. Finally, a lagged value of average temperature and a month indicator are added to account for the seasonal nature of crime and the connection of high temperature with illicit activities (Field 1992).

Initially, a random forest with default hyperparameters is calculated (RF_d). As seen in the table 4, 500 trees are created, at each split a randomly selected predictor is chosen from a set of 4, and the minimal node size is 5.

Next, a hyperparamter tuning process is performed with the method of cross-validation on the training set. To respect the temporal structure of the data, an expanding window is chosen with an initial training period of 60 months, performing a one step forecast each time to mimic the environment in which the Random Forest will have to perform the forecasting. Due to computational restrictions, a random sample of 15 hyperparameter combinations is chosen for this process. Those combinations are picked at random from a provided range for each hyperparameter. After the cross-validation process, the best hyperparameters are chosen and the tuned Random Forest (RF_t) is fitted onto the whole data up to the testing split.

Table 4: Default vs. tuned hyper-parameters for the two Random Forest models

	Trees	**mtry**	**min_n**
RF_d (default)	500	4	5
RF_t (tuned)	310	13	3

Both the initial forest and the one with tuned hyperparameters perform a one step ahead forecasting on the training set. This alligns with the usual procedure of training a machine learning algorithm only once and then updating it with new data without retraining, unless a structural change in the data occurs (Hewamalage et al. 2022).

4.3 XGBoost

XGBoost (Extreme Gradient Boosting) is an optimized boosted tree model that uses gradient descent to minimize loss. Boosting 8, in contrast to the Random Forest, develops its trees sequentially, using information from all the previously grown trees (James et al. 2023). This results in a slow learning procedure, which reduces overfitting. This is especially important in a forecasting setting. In the spatio-temporal data environment, XGBoost has been documented to perform well when forecasting theft (Djon et al. 2023).

The same features as in the Random Forest were selected for the XGBoost model 4. However, because XGBoost cannot handle categorical variables, the month indicator is transformed into a dummy variable, creating 12 columns. Similarly to the Random Forest approach, two XGBoost models were created. The first one without hyperparameter tuning (XGB_d) and a second one with hyperparameter tuning based on an expanding window cross-validation on the training dataset (XGB_t), with an initial window of 60 observations expanding each time by one step. Due to computational constraints, a random sample of 15 hyperparameter combinations were chosen for the tuning process.

Table 5: Default vs. tuned hyper-parameters for the two XGBoost models

	Trees	**Tree depth**	**Learning rate**	**`mtry`**	**`min_n`**
XGB_d (default)	15	6	0.3	31	1
XGB_t (tuned)	618	3	0.0095	15	8

The tuning process favored a hyperparamter combination with a very low learning rate. It also increased the amount of trees but decreased their depth. The amount of randomly selected predictors at each split ($mtry$) is also decreased. The minimal node size (min_n) is increased to 8. The above hyperparameters, and especially the decreased learning rate, seem to improve the model's ability to generalize and avoid overfitting.

The same approach to forecasting was applied as with the Random Forest models. The models perform a *de facto* one step ahead forecast, based on the true lagged values, thanks to the structure of the data. The model is not retrained at each forecast origin due to computational burden (Hewamalage et al. 2022).

4.4 Prophet

Prophet is an additive forecasting model introduced by Meta in 2017 (Taylor & Letham 2017). Although it was constructed as a business solution, it has been used in non-business related cases such as BITCOIN forecasting (Yenidoğan et al. 2018). It is characterized as providing fully automated forecasts, with some possibilities of tuning, like the choice between an additive or multiplicative seasonality. No data preparation is needed as Prophet is able to handle trends, seasonality, and other characteristics of real life data automatically. The prophet equation (3) consists of: the trend component $g(t)$, the

seasonal component $s(t)$, the holiday component $h(t)$, and the usual error term ε_t.

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \tag{3}$$

In this study, Prophet is considered as a potential tool used by a person without domain knowledge and therefore is used out-of-the-box with its default values. A crucial benefit of the Prophet model is its ability to produce decomposition graphs as well as automatically detect "changepoints" in the data. When applied to the crime series, it correctly identifies yearly seasonality as well as the trend and its change, visualizations are available in the appendix 9.

As Prophet works on decomposition basis, and a new observation cannot just be appended to the dataset to perform a one step ahead forecast, an expanding window forecast approach, as in the $SARIMA$ models, is chosen. This means that for each new one step ahead forecast the Prophet model is refitted.

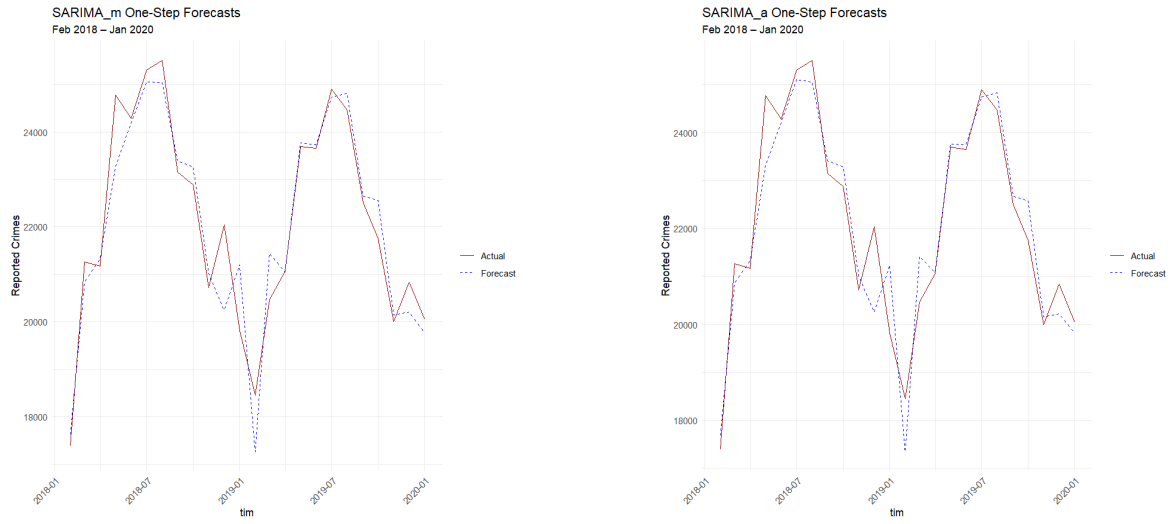
5 Results

In this section the results of the one-step-ahead forecasts of the models are presented. Firstly, the visual representations of the actual values and the forecasts are shown. Secondly, the accuracy of models is compared using different loss functions, a model confidence set test is also performed. Finally, feature importance graphs of the tree-based models are briefly discussed in comparison to the existing literature.

5.1 Visual Inspection of the Forecasts

The plots of the models' forecasts and the realized values are presented. Note that only the testing period of the data is shown to preserve clarity.

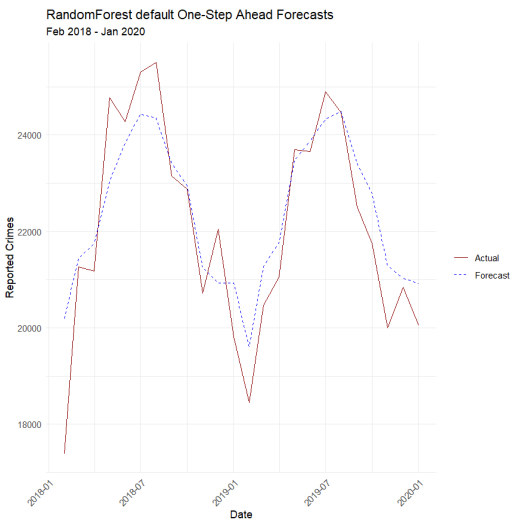
An initial visual inspection indicates that both $SARIMA_m$ and $SARIMA_a$ perform best. The worst-performing forecast seems to be that of the Prophet model.



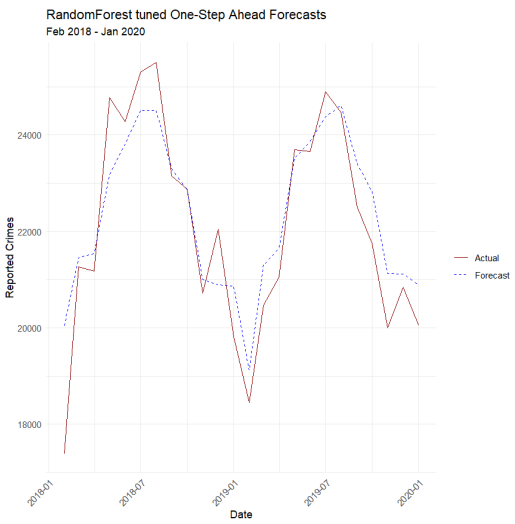
(a) SARIMA_m

(b) SARIMA_a

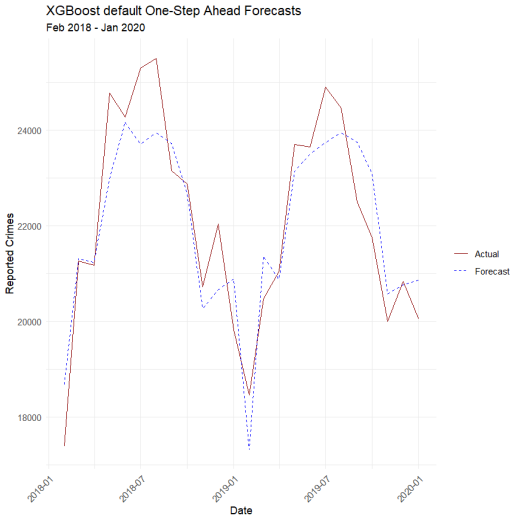
Figure 3: One-month-ahead expanding-window forecasts — **SARIMA models**



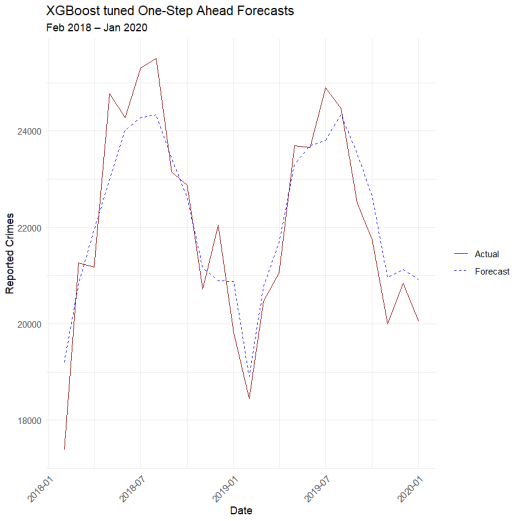
(a) RF_d



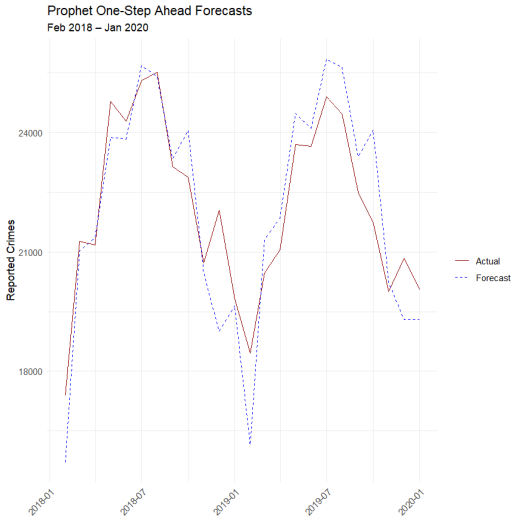
(b) RF_t



(c) XGB_d



(d) XGB_t



(e) Prophet

Figure 4: One-month-ahead expanding-window forecasts — **other models**

5.2 Model Accuracy

In order to asses model accuracy three loss functions were chosen. First one being the typical Root Mean Square Error (RMSE) 4, which penalizes larger deviations of a forecast from the true realized value. RMSE was chosen as it is one of the standard methods to evaluate a forecast and should be relevant to crime forecasting, larger deviations impose larger costs on society. The second metric is the Mean Absolute Error (MAE) 5, which is a symmetric loss function. It was chosen as a general metric of accuracy. Lastly, an asymmetric version of MAE, $lin - lin$ 6, is used to simulate a scenario where under-forecasting should be penalized more than over-forecasting, assuming that it is better to mobilize more police units rather than to under-mobilize.

The $SARIMA$ models perform best in all metrics, especially the $SARIMA_a$. In terms of the metrics, the tuning of the hyperparatmers improves on each untuned model. An interesting observation can be made based on the $lin - lin$ loss, both XGBoost models tend to underforecast in comparison to the Random Forest ones. However, based on RMSE, the tuned XGBoost model performs better in terms of the magnitude of the errors compared to the tuned Random Forest.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{4}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5}$$

$$\text{lin–lin} = \frac{1}{n}\sum_{i=1}^{n}\begin{cases} 0.7\,|y_i - \hat{y}_i|, & \hat{y}_i < y_i \quad \text{(under-forecast)} \\ 0.3\,|y_i - \hat{y}_i|, & \hat{y}_i \geq y_i \quad \text{(over-forecast)} \end{cases} \tag{6}$$

Table 6: Forecast-error metrics

	RMSE	**MAE**	**lin–lin**
SARIMA_m	705.603	503.230	265.067
SARIMA_a	696.887	499.540	258.370
RF_d	989.909	784.362	337.416
RF_t	912.349	711.850	308.94
XGB_d	955.258	782.073	415.411
XGB_t	871.834	730.993	347.930
Prophet	1253.244	1084.48	478.726

A Model Confidence Set test was performed to additionally assess the performance of the forecasts. This method was chosen for its ability to compare multiple forecasts at once. The testing begins with the initial set of models $M_0 = \{M_1, \ldots, M_m\}$ and their loss differentials $d_{ij,t} = L_{i,t} - L_{j,t}$, which are calculated according to a specific loss function. The null hypothesis states that all models perform equally well, $H_0 : \mathbb{E}[d_{ij,t}] = 0 \;\; \forall\, i, j \in M$. The test statistic is computed and the worse performing model is excluded from the initial set. The process continues until the null hypothesis cannot be rejected. An additional advantage of this method is that it does not require a benchmark model to

which all other forecasts are compared (Elliott & Timmermann 2016). For the test, the square losses of the models are chosen, penalizing the magnitude of the forecast error. The test is performed at the 0.05 confidence level, under the $Tmax$ statistic, which is sensitive to the worse-performing models. As seen from the results (7), none of the models

Table 7: Model Confidence Set test under square loss ($\alpha = 0.05$). Overall MCS p-value $= 0.2306$

	Rank_M	v_M	**MCS_M**	**Loss**
SARIMA_m	2	−2.126	1.0000	497 876.1
SARIMA_a	1	−2.237	1.0000	485 652.8
RF_d	6	0.578	0.9886	979 920.4
RF_t	4	−0.026	1.0000	832 381.2
XGB_d	5	0.409	0.9984	912 518.6
XGB_t	3	−0.545	1.0000	760 095.7
Prophet	7	1.676	0.2306	1 399 824.0

are removed from the set. However, the test outputs a ranking which indicates that the best performing models in the set are the two SARIMAs, followed by the tuned XGBoost. Prophet is the worst performing model according to the test.

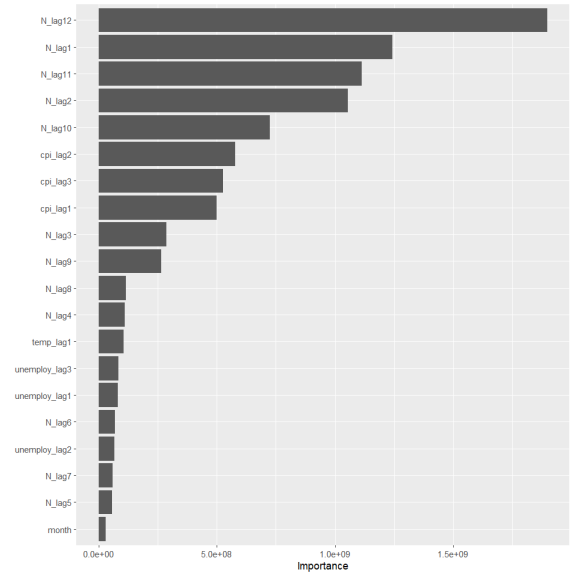
5.3 Feature Importance

A lot of machine learning methods, which Random Forest and XGBoost are a part of, are criticized for their lack of interpretability. A single decision tree is much easier to explain and even visualize, in comparison to hundreds or thousands of trees built simultaneously or iteratively.

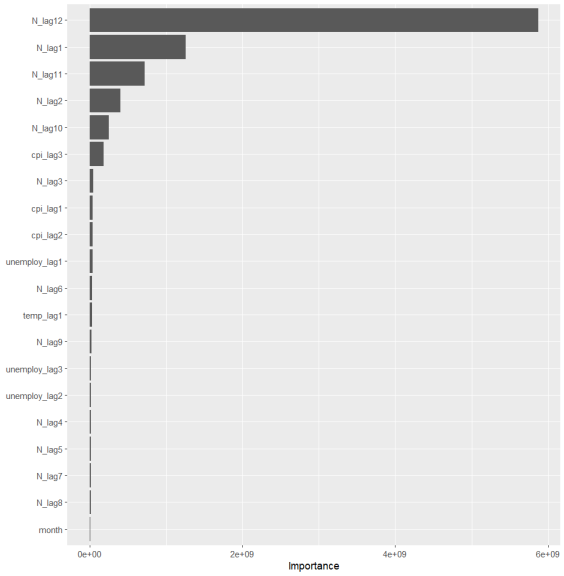
It is possible, however, to create a Feature Importance graph for each of the machine learning models employed in the study. Feature Importance graphs reveal the importance of a variable during the splitting phase, which aims at reducing the square error. In other words, how important a variable is in dictating the predictive ability of the model. An additional benefit of Feature Importance plots is the possibility to verify the hypothesis stated earlier in the literature review about the effect of unemployment, CPI, or temperature on crime levels.

Inspecting the graphs, a strong indication of the autoregressive and seasonal nature of crime is present. All of the models base their decisions on the lagged values of crime, with the lag of 12 being the most important. The default Random Forest model seems to extract information from a wider range of predictors, compared to the tuned version. The opposite can be noticed for the XGBoost models. The default version of XGBoost relies almost solely on the lag 12 of the reported crime when splitting the trees. After hyperparameter-tuning, the model uses a wider range of lags as well as the CPI.

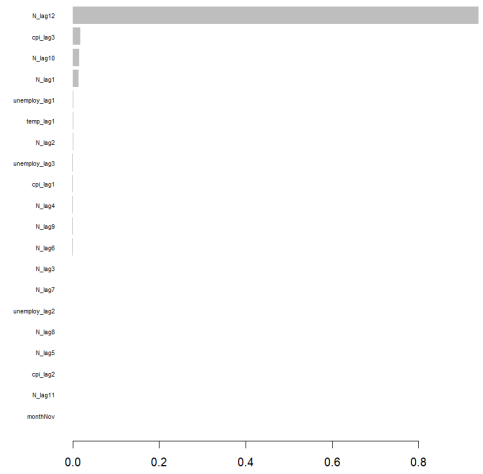
Another observation to be made is that CPI is used substantially more in comparison to unemployment. In the untuned version of XGBoost, and the tuned versions of Random Forest and XGBoost, the third lag of CPI is utilized most.



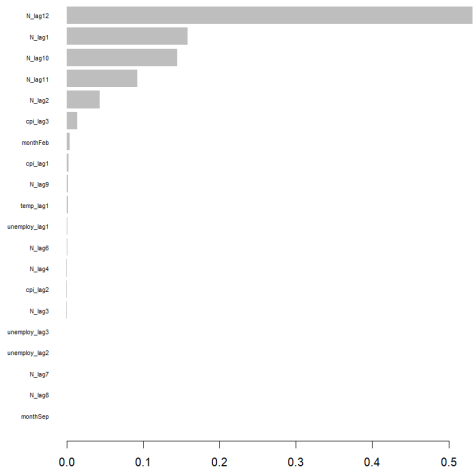
(a) Untuned Random Forest



(b) Tuned Random Forest



(c) Untuned XGBoost



(d) Tuned XGBoost

Figure 5: Feature-importance rankings for each model configuration

6 Discussion

As suggested by the MCS test 7 and the comparison of accuracy measures 6, the best performing models are the $SARIMA_a$ and $SARIMA_m$. Given the nature and amount of data considered in this study, it can be argued that the autoregressive and seasonal patterns of crime are enough to perform an accurate monthly forecast. The $SARIMA$ models, despite being the simplest ones in this study, seem to be the best choice for one step ahead monthly reported crime forecasting.

The next best models in terms of performance are the tuned machine learning ones, XGB_t and RF_t. It is confirmed that, despite being computationally intensive, cross-validation hyperparameter tuning improves the forecasting abilities of those models and

reduces the errors. Interestingly, as it can be seen from the asymmetric errors of lin-lin, XGB_t seems to underforecast more than RF_t. However, when comparing the penalizing RMSE, XGB_t performs better.

From the Feature Importance graphs, apart from the clear favorization of crime lags by the models, it is interesting to see CPI being one of the most important variables when it comes to decreasing square losses at each split. Comparatively, unemployment as a feature is not important. This may become an additional argument for the CPI being a better variable than unemployment for determining crime in the Economics of Crime's domain discussion. It should be noted that no additional feature selection methods were employed, which could potentially improve the predictive accuracy of the machine learning models further.

The worst performing model was *Prophet*, despite being refitted at each step, similarly to the $SARIMA$ models. However, since the *Prophet* model was developed with high-frequency business data in mind (weekly and even daily data, accounting for holidays, etc.), it was not expected to be the best performer. It may be the case that higher granularity of the data and increasing the forecasting horizon may result in better performance.

7 Future Research

The future research could be directed towards leveraging the available high frequency crime data and additionally employing a wider range of socio-economic features as predictors. This could lead to potential improvements in the tree-based models of this study as well as further connecting the domains of crime forecasting and economics, possibly adding value to the discussions in the field of Economics of Crime.

With sufficient computational power, more sophisticated machine learning models could be tested in their ability to forecast weekly or daily reported crime rates. Additional socio-economic variables, apart from CPI or unemployment, could be used as potential predictors. The difficulty in this setting would be finding or transforming said variables to be applicable to the higher-frequency context. Many economic indicators are calculated on a monthly, if not yearly, level.

As an example of tackling such a problem, if housing prices were a potential variable to be tested, instead of using a relevant index, high-frequency price data could be collected by methods of web-scraping. This would allow for both leveraging the already available detailed reported crimes data as well as verifying the predictive ability of a socio-economic feature. Finally, in terms of predictive ability, the sophisticated models should be compared with traditional time series techniques, like the $SARIMA$ models in this study.

8 Conclusion

In this study, seven forecasting models were tested in their ability to perform one-step-ahead predictions of monthly reported crime in Chicago. Two specifications of a traditional time series $SARIMA$ model, both untuned and tuned machine learning models, Random Forest and XGBoost, and an automated forecasting package, Prophet. The study concluded that the two $SARIMA$ models were the most accurate in terms of predictive ability, yielding the lowest RMSE, MAE, and an asymmetric $lin - lin$, which

penalized under-forecasting, as well as being ranked first in an MCS test.

Additionally, by plotting the Feature Importance for the tree-based models, it was concluded that CPI plays a more important role than unemployment in terms of its connection to crime forecasting, contributing additional arguments to the discussion of the relationship between unemployment and criminal behavior. Finally, a suggestion on future research was made, focusing on leveraging the availability of high-frequency data and testing additional socio-economic predictors.

References

Breiman, L. (2001), 'Random forests', **45**(1), 5–32.
URL: *http://link.springer.com/10.1023/A:1010933404324*

Buonanno, P. (2003), 'The socioeconomic determinants of crime. a review of the literature'. Accepted: 2011-06-09T13:22:06Z Publisher: IT.
URL: *https://boa.unimib.it/handle/10281/22981*

Chen, P., Yuan, H. & Shu, X. (2008), Forecasting crime using the ARIMA model, Vol. 5, pp. 627–630.

Djon, D., Jhawar, J., Drumm, K. & Tran, V. (2023), 'A comparative analysis of multiple methods for predicting a specific type of crime in the city of chicago'. Version Number: 1.
URL: *https://arxiv.org/abs/2304.13464*

Elliott, G. & Timmermann, A. (2016), *Economic forecasting*, Princeton University Press. OCLC: ocn928115332.

Elluri, L., Mandalapu, V. & Roy, N. (2019), Developing machine learning based predictive models for smart policing, *in* '2019 IEEE International Conference on Smart Computing (SMARTCOMP)', pp. 198–204.
URL: *https://ieeexplore.ieee.org/abstract/document/8784006*

Field, S. (1992), 'THE EFFECT OF TEMPERATURE ON CRIME', **32**(3), 340–351.
URL: *https://doi.org/10.1093/oxfordjournals.bjc.a048222*

Fitzpatrick, D. J., Gorr, W. L. & Neill, D. B. (2019), 'Keeping score: Predictive analytics in policing', *Annual Review of Criminology* **2**(1), 473–491.

Hewamalage, H., Ackermann, K. & Bergmeir, C. (2022), 'Forecast evaluation for data scientists: Common pitfalls and best practices'.
URL: *http://arxiv.org/abs/2203.10716*

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2023), *An introduction to statistical learning with applications in R*, second edition edn.
URL: *https://www.tandfonline.com/doi/full/10.1080/24754269.2021.1980261*

Kounadi, O., Ristea, A., Araujo, A. & Leitner, M. (2020), 'A systematic review on spatial crime forecasting', **9**(1), 7.
URL: *https://doi.org/10.1186/s40163-020-00116-7*

Mandalapu, V., Elluri, L., Vyas, P. & Roy, N. (2023), 'Crime prediction using machine learning and deep learning: A systematic review and future directions', *Ieee Access* **11**, 60153–60170.

Rosenfeld, R., Vogel, M. & McCuddy, T. (2019), 'Crime and inflation in u. s. cities', **35**(1), 195–210.
URL: *https://doi.org/10.1007/s10940-018-9377-x*

Taylor, S. J. & Letham, B. (2017), 'Forecasting at scale'.
URL: *https://peerj.com/preprints/3190v2*

Weller, D., Institution, H., for Econometric Studies of the Justice System, C., University, S. & of America, U. S. (1979), *Forecasting Crime Rates: A Review of the Available Methodology*, Center for Econometric Studies of the Justice System, Hoover Institution ....

Winter, H. (2008), *The Economics of Crime: An Introduction to Rational Crime Analysis*, Routledge.

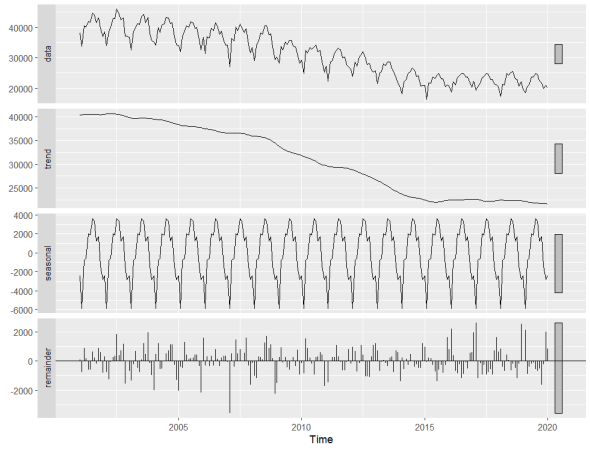
Yearwood, D. L. & Koinis, G. (2011), 'Revisiting property crime and economic conditions: An exploratory study to identify predictive indicators beyond unemployment rates', **48**(1), 145–158.
URL: *https://www.sciencedirect.com/science/article/pii/S0362331910000893*

Yenidoğan, I., Çayir, A., Kozan, O., Dağ, T. & Arslan, (2018), Bitcoin forecasting using arima and prophet, *in* '2018 3rd International Conference on Computer Science and Engineering (UBMK)', pp. 621–624.

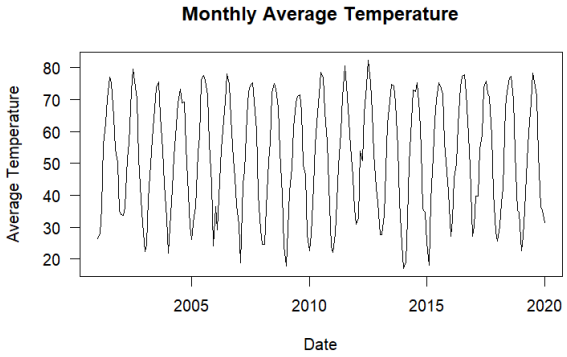
Data sources

- Chicago Data Portal. Crimes—2001 to Present. https://data.cityofchicago.org.
- U.S. Bureau of Labor Statistics, Unemployment Rate in Chicago-Naperville-Elgin, IL-IN-WI (MSA) [CHIC917URN], retrieved from FRED, Federal Reserve Bank of St. Louis. https://fred.stlouisfed.org/series/CHIC917URN.
- U.S. Bureau of Labor Statistics, Consumer Price Index for All Urban Consumers: All Items in Chicago-Naperville-Elgin, IL-IN-WI (CBSA) [CUURA207SA0], retrieved from FRED, Federal Reserve Bank of St. Louis. https://fred.stlouisfed.org/series/CUURA207SA0.
- Temperature data retrieved from SC ACIS. https://scacis.rcc-acis.org/.

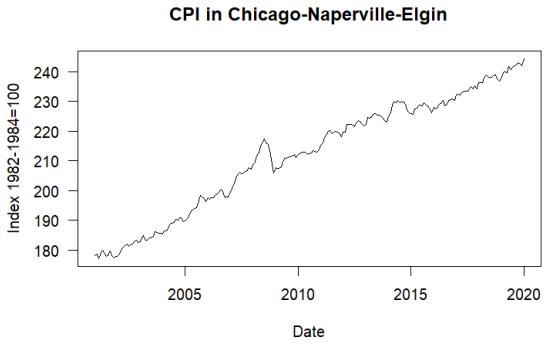
Appendix



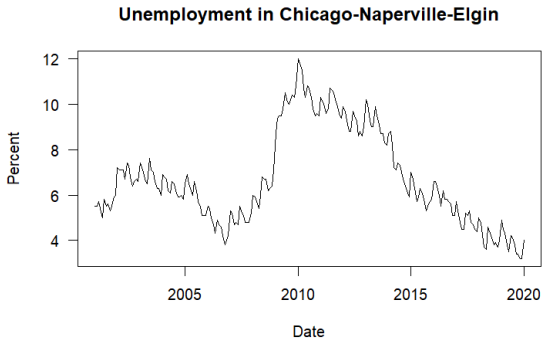
Decomposition of the monthly crime data



Monthly average temperature



(a) CPI for Chicago-Naperville-Elgin

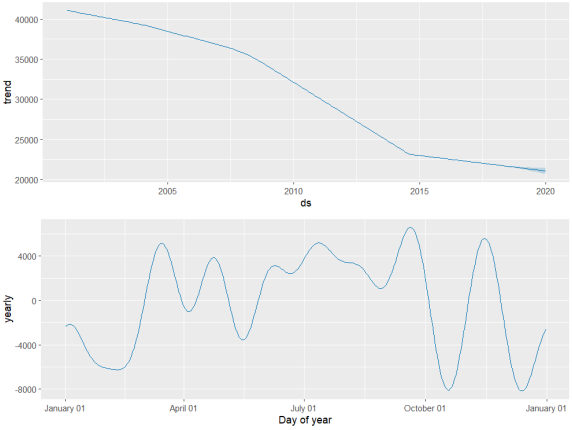


(b) Monthly unemployment for Chicago-Naperville-Elgin

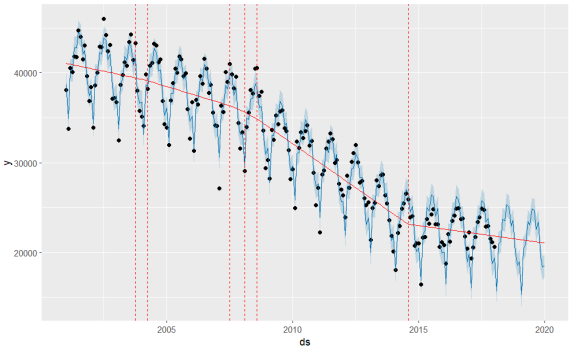
Economic indicators

Table 8: Information criteria for competing SARIMA specifications

	AIC	**AICc**	**BIC**
SARIMA(0,1,2)(1,1,1)$_{[12]}$	3007.390	3007.712	3023.326
SARIMA(3,1,0)(1,1,1)$_{[12]}$	3017.413	3017.867	3036.538
SARIMA(2,1,0)(1,1,1)$_{[12]}$	3022.868	3023.191	3038.805
SARIMA(2,0,1)(0,1,1)$_{[12]}$	3022.769	3023.223	3041.926



(a) Decomposition produced by Prophet



(b) Changepoints detected by Prophet

Prophet generated plots

Random Forest. Given T regression trees $\{f_t(\mathbf{x})\}_{t=1}^{T}$ trained on bootstrap samples, respecting temporal nature, and split on random feature subsets, the ensemble prediction is the simple bagging average

$$\hat{y}_{\mathrm{RF}}(\mathbf{x}) \;=\; \frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{x}).$$

Each f_t is grown to full depth (or a prescribed maximum); randomness in both the bootstrap sample and the candidate-split feature set decorrelates the trees and reduces variance.

Boosting for Regression Trees algorithm [1]

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
   (a) Fit a tree f^b with d splits ($d+1$ terminal nodes) to the training data (X, r).
   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda\, f^b(x).$$
   (c) Update the residuals,
   $$r_i \leftarrow r_i - \lambda\, f^b(x_i).$$

[1] Adapted from James et al. (2023)

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \, f^b(x).$$