Automated Machine Learning Framework for Demand Forecasting in Wholesale Beverage Alcohol Distribution

Jenna Ford, Christian Nava, Jonathan Tan, Bivin Sadler

Master of Science in Data Science, Southern Methodist University,  
Dallas, TX 75275 USA

{jennaf, cjnava, jhtan, bsadler}@smu.edu

**Abstract.** This paper covers the development, testing, and implementation of an automatic framework for analyzing and forecasting demand for an alcoholic beverage distributor’s products at varying levels of granularity. Rather than look at macroscale geographic demand for a product from a distribution center, this framework will look at the localized customer level demand for that product before aggregating total demand. The approach will better capture individual behavior variations in each customer and allow for a more accurate estimation of the total monthly demand for that product. To best account for each product’s influencing factors, each product is analyzed separately per customer with both traditional time series and machine learning models to identify the best performing forecast results. This research sets up an AutoML framework to individually identify the best forecasting model for different product/customer combinations in the data.

1 Introduction

Demand forecasting is a vital part of retail business establishments. For a wholesale beverage alcohol distribution company, having too much supply on-hand can lead to excess storage costs, while not having enough supply on-hand leaves revenue on the table. In the United States, getting alcoholic beverages from producer to consumer is a three-tiered distribution process where a producer sells directly to a wholesale distributor who, in turn, sells to a direct retailer that then sells to a consumer. Predicting retail demand, or forecasting demand, for a wholesale distributor can be a valuable tool, which would allow the distributor to more accurately stock its inventory throughout the year. Demand forecasting is particularly valuable when dealing with a perishable product like beer, which may be discarded by, or returned for disposal to, the distributor, increasing the distributor’s costs.

The demand profile for each product varies according to region, customer, and time of year. This may require a different model for each customer and product combination, which can result in a costly investment of resources. Additionally, not every product has enough historical data to be modeled and going through each store and product combination can be a laborious task for which a retail organization may not have adequate staff. Employing an automated machine learning (AutoML) solution can allow retail organizations with smaller teams to extract meaningful insights while lowering technical barriers. This study focuses on creating an AutoML approach to demand forecasting where all product/customer combinations are forecasted and a winning model is selected automatically, taking into account stationarity and white noise. Traditional time series models as well as deep learning and ensemble models are incorporated into the framework.

Previous work on beverage alcohol distribution demand forecasting for this company focused on single product and customer combinations [1,2]. Traditional time series techniques and deep learning methods were employed in both studies. The results varied for the different product/customer combinations as to which model performed best.

There are two primary concerns to address for the AutoML framework for time series analysis: stationarity and whether the original dataset is white noise. Whether a time series is stationary will determine which models are appropriate and if any transformations need to take place. Stationarity is typically determined based on visual inspection of the time series. However, with an AutoML framework, we will need to identify another way to evaluate stationarity. Additionally, if a time series is white noise, there is no need to model the series and a simple equal means forecast is sufficient.

The benefit of an AutoML approach is the speed at which forecasts are delivered. However, prediction accuracy is expected to suffer slightly without human intervention. This kind of AutoML framework could be useful to make quick and impactful changes to the supply chain while a more in-depth analysis of each individual time series is undertaken.

The remainder of this paper is organized as follows: In Section 2, related literature is reviewed. Section 3 provides information about the dataset and time series that will be evaluated. Section 4 goes through a traditional exploratory data analysis (EDA) for one of the time series in the dataset. In Section 5, the AutoML framework is reviewed along with in-depth descriptions of how determinations are made for whether a time series is white noise and whether a time series is stationary. Section 6 provides an overview of the models used in the AutoML framework. In Section 7, model evaluation techniques are reviewed for identifying the winning model. Section 8 provides the results from the AutoML framework and Section 9 highlights conclusions of this research. Lastly, in Section 10, topics for further research are explored.

2 Related Work

Previous work on demand forecasting has been done for the same company and location as this study. Aurora et al. (2020) performed demand forecasting on aggregated case sales for two products: Taaka Vodka 80 1L and Jack Daniel’s Whiskey [1]. Aurora et al. used S-ARIMA, Vector Auto-Regression (VAR), Long Short-Term Memory Networks (LSTM), and ensemble models to forecast monthly case sales. A weighted ensemble model combining forecasts for S-ARIMA, VAR and LSTM was used for Taaka Vodka and an average ensemble model was used for Jack Daniel’s Whiskey. A rolling-window ASE was used to determine the best model for each product. For both products, the LSTM model achieved the lowest rolling-window ASE. Due to concerns of overfitting with the LSTM models, Aurora et al. identified the ensemble models as the models with the best fit. The root mean squared error (RMSE) for forecasting monthly case sales with the ensemble model was reduced 50% for Taaka Vodka and 33.5% for Jack Daniel’s Whiskey, compared to the naïve forecasts of the same period from the previous year.

Jiang et al. (2020) also worked on demand forecasting for the same company, but a different location [2]. Jiang et al. focused on vodka products and noted that there are different seasonal patterns found for different products within the vodka category. Two vodka products for three different customers were ultimately selected for forecasting. One of the products displayed a strong seasonal trend and the other did not. The following models were run for each of the six time series: naïve using the monthly value from the previous year as the forecast, naïve using an average of the monthly value from the previous two years as the forecast, ARMA, ARIMA with d=1, ARUMA with s=5, signal-plus-noise, Multiple Linear Regression (MLR), biLSTM, CNN LSTM and a multivariate LSTM. The results indicated that in five of the six time series being forecasted, Jiang et al. were able to improve forecast accuracy compared to the naïve models. The conclusion was that there is no single model that performed best in all instances. This conclusion, in addition to the findings in Aurora et al., lead to the prospect of an AutoML approach to identify different models for different time series to achieve higher forecasting accuracy.

AutoML is a quickly growing field in Data Science with a goal of reducing human interaction in the process of model development [3]. AutoML algorithms typically create a static AutoML template by performing data preprocessing and feature selection followed by the primary task such as classification or regression [4]. This static template presents issues with parameter optimization and scalability [4]. A variety of AutoML tools are increasingly available in both for-purchase and open source environments. In reviewing open-source options, Budjač et al. notes that these tools are limited by the tasks they can be applied to and are not one-size-fits-all solutions [3].

Literature on AutoML for traditional Time Series applications is sparse. This is not surprising given the need to visually inspect the data to determine if the conditions for stationary are met before proceeding with modeling. Newer, deep-learning models do not typically suffer from the same constraints. Take, for example, artificial neural network modeling schemes based on the generalized regression neural network (GRNN) [5]. A GRNN is fast-learning and has a single design parameter. The model was awarded best prediction in the NN3 time-series competition among 60 models submitted. Additional work that involves time-series models and AutoML includes the use of multiple kernel learning (MKL) to automatically select the optimal size of sliding windows and find the pattern of time series [6]. Allen and Balaji’s work benchmarked the auto-sklearn and TPOT frameworks against H20’s AutoML using datasets from OpenML and found auto-sklearn outperformed for classification datasets and TPOT outperformed for regression datasets [7]. However, Allen did not include mention of time series data. In their review of AutoML frameworks from a computer science and biomedical perspective. Waring, Lindvall, and Umeton (2020) present an effort to better utilize “off-the-shelf” machine learning models [8]. They focus on open-source AutoML tools and find efficiency limitations of AutoML on large-scale datasets.

The authors are unaware of research on automated machine learning applied specifically to supply chain logistics or for retail demand forecasting. With respect to demand forecasting, Ahmed et al. (2010) compare several machine learning methods on business-type time series, more specifically, a subset of the monthly M3 time series competition data [9]. They found that multilayer perceptron, Gaussian processes, and Bayesian neural networks performed best among the models compared, and they noted that preprocessing can have a large impact on performance. Other studies have shown that holidays or special days can pose a challenge when forecasting retail demand. Huber and Stukenschmidt (2020) address the problem of forecasting daily demand in the presence of special days for a bakery chain by using artificial neural networks and gradient boosted decision trees [10]. They found that classification-based approaches outperformed regression-based approaches.

The objective of the AutoML application of this research is to use the resulting forecasts to make decisions on purchasing inventory. This implies that the accuracy of the forecasts is more important than their interpretability. As such, “black-box” models that are not easily interpretable are can be explored. Elsayed, Maida, and Bayoumi (2019) compare a long short-term memory (LSTM) model to a gated recurrent unit (GRU) model to create a hybrid, fully convolutional GRU (FCU-GRU) model [11]. They found that the FCU-GRU model outperformed the LSTM model for a univariate time series.

Improving the performance of AutoML models can be achieved by combining model or using pretrained models. Combining algorithms has also shown to improve performance [12, 13]. Noh et al. (2020) used a hybrid model using a genetic algorithm and a gated recurrent unit (GA-GRU) where the GA model was used to find the optimal hyperparamters of the GRU model [12]. They found the GA-GRU model outperformed ARIMA, LSTM, and RNN models. Helmini, Jayasinghe, and Perera (2019) use an LSTM with “peephole connections” on the Rossmann data set for sales forecasting and found that the peephole connection LSTM outperformed extreme gradient boosting (XGB) and random forest models [14]. Additionally, LSTM models will be considered given they tend to outperform traditional ARIMA models in certain use cases. Weytjens, Lohmann, Kleinsteuber (2019) use an LSTM model to forecast cash flows and compared the LSTM model’s performance to ARIMA, multiple-layer perceptron (MLP), Facebook’s Prophet forecasting tool [15]. They found that the LSTM model outperformed ARIMA, MLP, and Prophet for periods between 1 and 30 days.

Pretrained models are those that have been trained on other datasets that are similar to the data of interest. Using metadata or pretrained models can lead to increased speed in AutoML, which can benefit use cases where data is with a similar distribution is generated on a frequent basis [16]. This is particularly interesting for an alcohol beverage distributor that generates weekly sales data.

3 Dataset

The dataset was provided by a large beverage alcohol distribution company in the United States. The data is monthly case sales from 2013-2019 for one metropolitan region in the United States. Each row of data represents one month of case sales for one product and one customer. Each customer represents a unique store location. There are 4,017 different products, 34 different customers and a combined total of 37,391 different combinations of product and customers to forecast.

Thirty-one columns of data are provided, the most important of which are the names and IDs for customers and products, as well as the number of cases sold and the total purchase price per transaction. Several other characteristics are also included about each product such as alcohol content, product categorization, and container volume.

Many of the product/customer combinations found in the dataset have a sparse number of records indicating that the product is not purchased on a consistent monthly basis by the customer. Missing observations are filled in with case sales of 0 and a total purchase price of $0. Due to the nature of time series analysis and the need to have historical data in order to forecast, product/customer combinations with data for fewer than half of the months from 2013-2019 are removed and not forecasted.

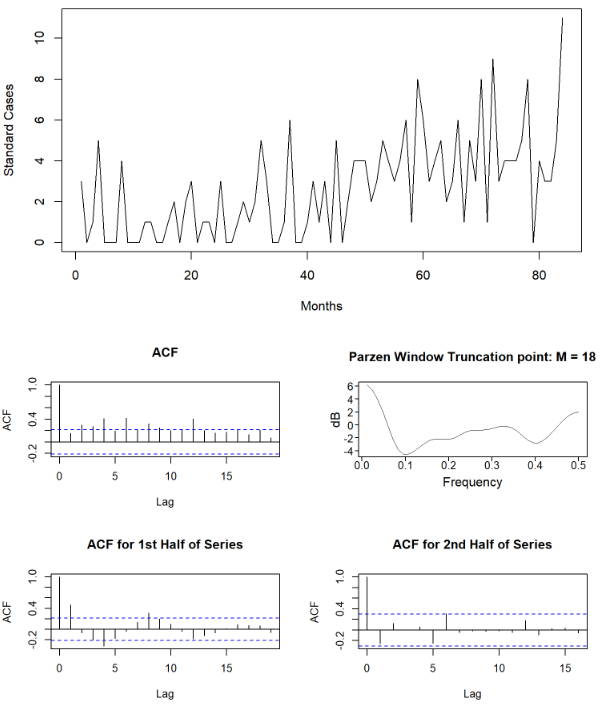
4 Exploratory Data Analysis

Time series exploratory data analysis (EDA) is conducted for a sample of ten product/customer combinations found in the dataset.

Analysis of the time series begins with a determination on stationarity. A stationary time series will have a constant mean, constant variance and autocorrelations that depend only on the time between observations. Fig. 1 shows a series of plots to assist in visually determining stationarity for one product/customer combination. The plot on the top row plots the time series. There appears to be wandering behavior and possibly a difference in the mean in the latter half of the series. There is not sufficient evidence to suggest that the variance in not constant. Autocorrelation (ACF) plots are used to help determine if the autocorrelation structure differs for different segments of time. The bottom graphs show two ACF plots: one for the first half of the time series and one for the second half of the time series. Since the ACF plots look substantially different from one another, it is reasonable to conclude that the autocorrelations do not only depend on how far apart the observations are, but also on when the observations occur. This time series is not stationary since it does not meet the conditions of constant mean and autocorrelation.

A determination of whether the time series is white noise needs to be made as well. Modeling time series that are white noise is not effective or an efficient use of a data scientist’s time. The middle left graph in Fig. 1 is a plot of the autocorrelations (ACF). At a 95% confidence level, approximately one lag out of 20 would be expected to cross outside of the blue stripped bands if this series was white noise. With six lags extending beyond the 95% confidence level, this series does not appear to be white noise. Estimating parameters for an ARMA model also offers insight into whether the time series is white noise. If the model with the lowest selection criteria is an ARMA(0,0) it would be reasonable to suspect the time series is white noise. For the time series in Fig. 1, the model with the lowest BIC was an ARMA(1,2) and an ARMA(0,0) did not show up in the top five models. This is another piece of evidence that this time series is not white noise. Finally, Ljung-Box tests with K=24 and K=48 were run as another test for white noise. At as significance level of 0.05, the chi-square value for K=24 was 124.3111 and the chi-square value for K=48 was 194.4468; we fail to reject the null hypothesis that this dataset is white noise. The Ljung-Box test is not as conclusive as other tests for white noise and would indicate that this series is white noise.

Traditional time series modeling requires the time series to be stationary. For the time series in Fig. 1, transformation is necessary since the time series is not stationary. The Parzen Window graph in Fig. 1 can help identify possible transformations. A seasonal yearly pattern where s=12 is not evident in the Parzen Window since there is no peak at 0.0833. There are slight peaks at 0.1667 and 0.25, indicative of seasonal pattern of s=6 and s=4, respectively. Possible transformations to try would be differencing to account for the seasonal behavior.



**Fig. 1.** EDA performed for one of the product/customer groups.

The EDA performed here was done by visually inspecting a variety of plots and running a few different statistical tests. In an Auto-ML framework, visual inspection is removed. However, the issues of stationarity and white noise need to be addressed in another manner.

5 AutoML Framework

The primary goal for the AutoML framework is to determine which model most accurately forecasts the number of cases sold, without the need for human intervention. In time series analysis, the first things that a data scientist looks at are whether the time series is white noise and whether the time series is stationary.

For the AutoML framework developed for this paper, all time series are modeled, regardless of the determination of white noise and stationarity. Ljung-Box tests and an evaluation of ARMA models are done to indicate if the time series is white noise. If a time series is truly white noise, the equal means model is expected to outperform other models. For each product/customer combination forecasted, the winning model will be displayed, along with a note indicating if the time series was white noise.

A similar approach is taken to make a determination of stationarity. (We still need to decide on which tests and how to implement. More to come here).

Sections 5.1 and 5.2 explain how the determinations of white noise and stationarity are made and why they are important in the AutoML framework. Section 6 will provide details about the forecasting models used inside the AutoML framework.

5.1 White-Noise

Visualization of the time series is typically the first thing a data scientist does when attempting to model the data. One reason for this is determining whether the time series is white noise. If a time series is white noise, there may be no benefit gained from modeling. Forecast residuals are also typically reviewed to determine if the residuals are white noise. If the residuals are not white noise, this suggests that further modeling may be necessary to better explain the behavior in the data. Whether reviewing a dataset or forecast residuals, a visual inspection and a Ljung-Box test can be employed to assist with making a determination on white noise.

A visual inspection involves looking at the plot of the time series and an ACF plot. The lack of an identifiable pattern in the plot of the time series and lags that are not statistically significant in an ACF plot indicate this dataset may be white noise.

An alternative approach to checking for white noise is performing a Ljung-Box test. The Ljung-Box test approaches the autocorrelations as a group to determine if the residuals are white noise. It tests the null hypothesis, *H*0, that all autocorrelations, *ρ*, are zero (i.e., the residuals are white noise). If at least one autocorrelation is not zero, then white noise is not present.

|  |  |  |
| --- | --- | --- |
|  |  | (**1**) |
| . |  |

The Ljung-Box test is traditionally run with at least 2 different values of K. This is due to the nature of hypothesis testing where the significance level indicates how often incorrect results are expected. Both tests indicating the dataset is white noise gives more weight to the results. Different results from the 2 tests can occur. This indicates the dataset may be white noise.

For the AutoML framework used for this paper, no visual inspection can be done. Instead, Ljung-Box tests with K=10 and K=24 are performed. Identical results from these 2 tests on whether to reject or fail to reject the null hypothesis indicate that the dataset is white noise or not white noise, respectively. Differing results from the 2 tests indicate the Ljung-Box test is inconclusive.

One further method is employed to help determine if the dataset is white noise. The top 5 ARMA models are automatically generated using the aic5.wge function of the R package tswge. The Bayesian Information Criterion (BIC) is used to evaluate different models and the 5 models with the lowest BIC are determined. If any of these 5 models is an ARMA(0,0) this is an indication that the dataset may be white noise. This evaluation method is not as conclusive as the Ljung-Box test, but is another piece of information gathered about the determination of white noise.

If a dataset is white noise, it may not make sense for a data scientist to put a lot of time and effort into identifying an optimal model for the data. However, with an AutoML approach, the data scientist does not need to put a lot of time and effort into identifying an optimal model for the data. As long as processing time is not an issue and any associated costs are not a concern, there may be no harm in attempting to find a model for a dataset that is white noise. That is the approach taken here. All datasets, whether white noise or not, are run through all various models. Indications will be given to the user of the framework as to the determination of white noise. If the winning model happens to be something other than the equal means model, the user can determine if the equal means model is more appropriate.

5.2 Stationarity

Visualization of the time series is also important for making a determination on stationarity. The data scientist reviews the time series to see if the 3 conditions for stationarity are met: constant mean, constant variance, and autocorrelations that depend only on how far apart the observations are, not when they occur in time.

The first condition of stationarity is constant mean. If a time series has a constant mean, the mean does not depend on time. 2 possible reasons a time series may not have a constant mean are: a linear trend is present where the mean increases over time and a seasonal or cyclic pattern is present. For example, monthly temperature data would show higher temperatures in summer months and lower temperatures in winter months. This pattern is predictable and expected.

The second condition of stationarity is constant variance. If a time series has constant variance, the variance does not depend on time. This condition is a more difficult to evaluate. If the first condition of stationarity is not met, it is increasingly difficult to make a determination about constant variance. If multiple realizations of a time series can be imagined, the variances for each time point should not change throughout the series, if the variance is constant. For example, consider the time series in Fig. 2 that represents the daily difference between the high and low temperature in Austin, Texas from 2014-2016. The variation seen in winter months is much larger than the variation seen in summer months. This dataset does not have constant variance.

The third condition of stationarity is that autocorrelations depend only on how far apart the observations are, not where in time the observations are. This can be viewed by creating separate ACF plots for different ranges of the times series. The bottom row of ACF plots in Fig. 1 shows an ACF plot using the first half of a time series and another ACF plot using the second half of the same time series. The patterns in these 2 plots should match if the 3rd condition of stationarity is met. For the time series in Fig. 1, the patterns in the ACF plots do not match and this time series does not meet the 3rd condition for stationarity.

**Fig. 2.** Daily difference between high and low temperatures in Austin, TX. Data source: [https://www.kaggle.com/grubenm/austin-weather/data#](https://www.kaggle.com/grubenm/austin-weather/data)

Many forecasting models assume that the time series is stationary. If a non-stationary time series was modeled using a model that assumes stationarity, the forecast residuals would be larger than if the time series was first transformed and then modeled. Typically, the determination of stationarity is made by visualizing inspecting the data. This poses a problem for an AutoML framework, where no human interaction exists. Traditionally, if a time series is determined to be non-stationary, the data scientist will transform the data. Typical transformation options are differencing or averaging. Transforming the time series with a first difference would remove a linear trend in the data. Other differencing techniques are used to remove seasonal or cyclic behaviors in the dataset. Stationarity is reassessed on the transformed data. This process repeats until a stationary dataset is identified.

For the AutoML framework developed for this paper, several separate transformations are made on the data and the resulting dataset is modeled. Since the forecast residuals are expected to be higher if a non-stationary dataset is modeled as stationary and several different transformations are used, there is little concern that the winning model would be an inappropriate model with respect to stationarity.

6 Forecasting Models

Traditional time series forecasting techniques approach modeling and forecasting from a statistical perspective and has been applied in various fields since the industrial revolution. Classical equations and algorithms that represent past, present, and future values as stochastic variables are the most common and basic form of time series analysis such as ARMA/ARIMA models. These algorithms forecast future values of a time series by calculating the statistical likelihood of future values.

More complex and recent approaches such as decision trees, multilayer perceptron, and long-term short memory networks have been only recently made practical advances in computing capacity and speed. Each of these algorithms forecasts the future variables of a time series in different ways, and often don’t have the same conditions of use as traditional time series models. What follows in the remainder of Section 6 are descriptions of the models used in the AutoML framework used for this research.

6.1 Equal Means

The equal means model takes the mean of the time series and uses that value as the forecast. Residuals are calculated as the difference between the forecasted value and the mean. This model is most appropriate for a time series that is white noise. In a white noise time series, previous observations do not help with forecasting future observations and a mean is the best forecast available.

6.2 ARMA

ARMA (Autoregressive Moving Average) is a traditional model for univariate time series analysis. The AR part of the model uses regression to represent each value of the time series relative to previous values by expressing the current value as a function of past values.

|  |  |  |
| --- | --- | --- |
| . |  | (**2**) |

The MA part of the model uses a moving average with q number of coefficients. It quantifies the moving average of error terms for each point in the series, where error is the difference between expected and observed values. is the number of error terms that are averaged by the model, if , then previous three terms are averaged for each point.

|  |  |  |
| --- | --- | --- |
| . |  | (**3**) |

6.3 ARIMA

An ARIMA model is an Autoregressive Integrated Moving Average model that contains the same AR and MA components with an additional Integrated component. This represents the order of difference applied to the time series.

|  |  |  |
| --- | --- | --- |
| . |  | (**4**) |

The first order difference of a time series is the difference between a single point in the series and its neighbor. This can be useful for making nonstationary time series appear more stationary by stabilizing the variance and allowing stationary dependent techniques to be applied.

Seasonality of is another component of time series analysis that must be taken into account when modeling. Seasonality is the presence of an identifiable pattern within the time series, such as cyclical consistent increases or decreases in values. This trend can be identified by several methods, such as autocorrelation plots, spectral density estimation, or simply visual inspection of a realization.  A seasonal ARIMA model with the term S = n can account for cyclical changes that repeat every n terms in the time series. Examples of pattern identification would be S = 7 in a daily time series for a weekly pattern, S = 26 in a weekly data for biannual, and S = 12 in hourly data for a 12-hour pattern.

6.4 MLR

Multiple Linear Regression is a model that uses the linear relationship between points in the time series. A simple linear regression model assumes that the relationship between two variables is linear and uses one to forecast the other. Multiple linear regression operates on the same assumptions, just with more than one variable affecting the variable to be forecasted.

6.5 VAR

Placeholder.

6.6 LSTM

Placeholder.

7 Model Evaluation Method

There are various ways to identify the winning model. The AutoML framework for this paper uses a rolling-window ASE to identify the model the has the most accurate forecasts over time. Section 7.1 details the process and calculations for computing a rolling-window ASE.

Even if the time series appears to not be white noise and the winning model is not the equal means model, it is useful to check if there is a statistically significant difference between the equal means model and the winning model. An analysis of variance (ANOVA) test can be run to determine if there is a statistically significant difference between the model with the lowest rolling-window ASE and the equal means model. Section 7.2 provides the methodology to determine if there is a statistically significant difference between models.

7.1 Rolling-Window ASE

A rolling-window ASE will be used to measure the goodness of fit for model performance. The ASE measure takes the sum of the square of the difference between the predicted value (forecast), , and the actual value, . It then averages the error over the number of observations. A lower ASE value indicates a more accurate model.

|  |  |  |
| --- | --- | --- |
| . |  | (**5**) |

It should be noted that the ASE is a snapshot in time and can vary for the same data set depending on the size of the training data. It uses values, where is the length of the time series, to train the model and then uses the last values to validate forecasted values.

A more useful approach is to shorten the training period and fit the model on a smaller training set (a shorter "window" of time) and then validate the data on the subsequent values. In a process similar to cross-validation, the training set, or window, then “rolls” or “slides” to the subsequent period and evaluated again and again. Fig. 3 shows which observations are included in the training set and which observations are forecasted for different windows. 12 observations of data will be used to forecast the next 6 observations. Window 1 uses the first 12 observations to train the model. Observations 13-18 are forecasted with the model and an ASE is obtained. Window 2 slides over one observation and uses observations 2-13 to train the model. Observations 14-19 are forecasted with the model and an ASE is obtained. Once the windowing process has completed to the end of the dataset, the ASEs for the different windows are averaged together to get a rolling-window ASE.

The rolling-window ASE method can prove to be a more stable representation of the overall model ASE. For example, if there was some particularly odd behavior in the recent past of a time series, a single ASE could be misleading. The winning model is determined by the model with the lowest rolling-window ASE.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Window | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| Window 1 | Training Set | | | | | | | | | | | | Forecast | | | | | |  |  |  |  |
| Window 2 |  | Training Set | | | | | | | | | | | | Forecast | | | | | |  |  |  |
| Window 3 |  | | Training Set | | | | | | | | | | | | Forecast | | | | | |  |  |
| Window 4 |  | | | Training Set | | | | | | | | | | | | Forecast | | | | | |  |
| Window 5 |  |  |  |  | Training Set | | | | | | | | | | | | Forecast | | | | | |

**Fig. 3.** Rolling window training and test splits.

7.2 ANOVA

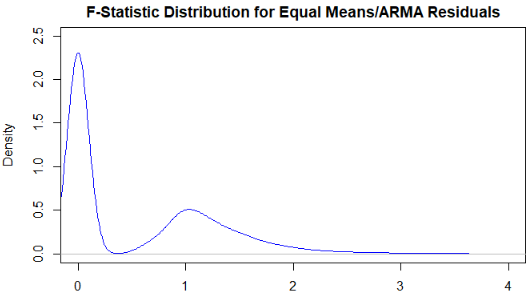
An ANOVA is performed to determine if the model with the lowest rolling-window ASE is statistically better than the equal means model. This calculation will be performed when the winning model is not the equal means model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Degrees of Freedom | Sum of Squared Residuals | MS | F-statistic |
| Extra Sum  of Squares |  |  |  |  |
| ARMA  Model |  |  |  |  |
| Equal  Means  Model |  |  |  |  |

**Fig. 4.** ANOVA table to determine if there is a statistically significant difference between the equal means model and the ARMA model for a time series.

Fig. 4 shows the ANOVA table used to calculate an F-statistic. In order to calculate the F statistic, the degrees of freedom and sum of squared residuals must be calculated for each model. The equal means model has degrees of freedom. Fig. 4 compares the equal means model to the ARMA model. The ARMA model has degrees of freedom, where is the number of observations in the dataset, is the number of autoregressive components in the model, and is the number of white noise components in the model. 1 is added to the number of parameters if the mean is not specified for the model. If the p-value associated with the F-statistic calculated from the ANOVA table with degrees of freedom is less than 0.05, then the null hypothesis that the ARMA and equal means models do not differ in forecasting precision can be rejected.

A simulation was performed with the ANOVA table in Fig. 4 to determine the distribution of the F-statistic comparing an equal means model and an ARMA model from a white noise dataset. 10,000 white noise time series were generated. An equal means model and an ARMA model were generated for each of the 10,000 time series and an F-statistic was calculated. The resulting density plot in Fig. 5 shows the distribution of the F-statistic for this simulation.



**Fig. 5.** F-statistic distribution for a simulation of 10,000 white noise time series and associated equal means and ARMA models.

8 Results and Analysis

A simple random sample of 10 product/customer combinations was run through the AutoML framework. Detailed results for 3 product/customer combinations are reviewed in the remainder of this section. Results for forecast horizons of 3 months and 6 months are provided in Tables 1-3.

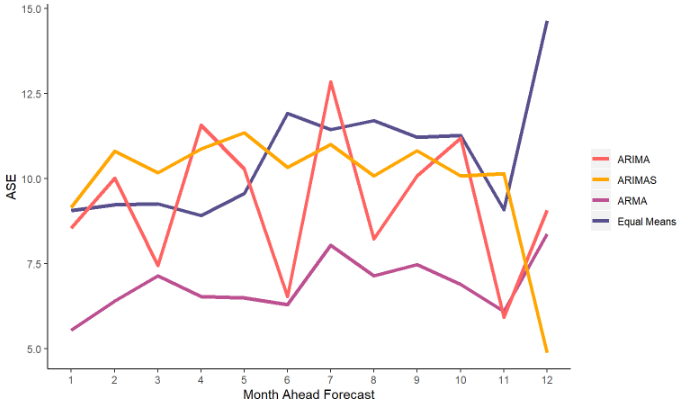
8.1 Product/Customer Combination 1

Table 1 shows model results for product/customer combination 1. All methods for making a determination on white noise indicate that this time series is not white noise. The winning model for both the 3 month and 12 month forecast horizon is the ARMA model.

**Table 1.** Model results for product/customer combination 1.

|  |  |  |  |
| --- | --- | --- | --- |
| White Noise Results | Model | Rolling-Window ASE | |
| 3 Month Forecast | 12 Month Forecast |
| Not white noise | Equal Means | 7.49 | 9.07 |
| ARMA | 6.44 | 6.87 |
| ARIMA, d=1 | 9.90 | 9.31 |
| ARIMA, s=12 | 8.66 | 9.98 |

In addition to determining which model is the winning model, it may also be of interest to see how the different models perform at different forecast horizons. Fig. 6 shows the average ASE at each month-ahead forecast by the different models in the AutoML framework. Fig. 6 shows that the forecast accuracies for the ARIMA with d=1 and the ARIMAS with s=12 have wide swings in prediction accuracy from month to month.



**Fig. 6.** ASE results by month-ahead and model for product/customer combination 1.

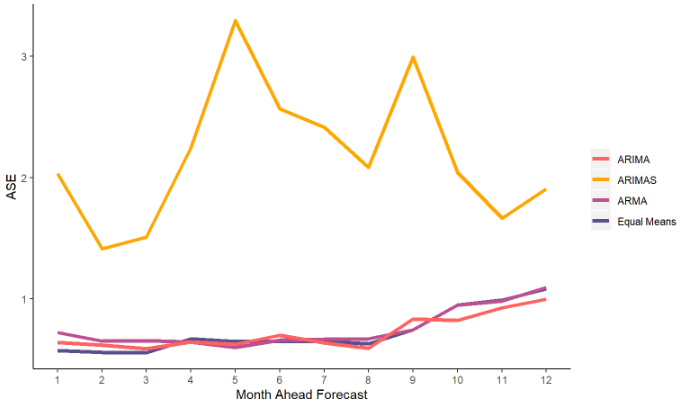
8.2 Product/Customer Combination 2

Table 2 shows model results for product/customer combination 2. All methods for making a determination on white noise indicate that this time series is white noise. The winning model for both the 3 month and 12 month forecast horizon is the equal means model, as is expected for a white noise time series.

**Table 2.** Model results for product/customer combination 2.

|  |  |  |  |
| --- | --- | --- | --- |
| White Noise Results | Model | Rolling-Window ASE | |
| 3 Month Forecast | 12 Month Forecast |
| White noise | Equal Means | 0.74 | 0.57 |
| ARMA | 0.88 | 0.75 |
| ARIMA, d=1 | 0.86 | 0.72 |
| ARIMA, s=12 | 1.47 | 2.18 |

The average ASE at each month-ahead forecast by the different models in the AutoML framework in plotted in Fig. 7. Fig. 7 shows that the forecast accuracies for the ARIMAS with s=12 are worse than the other 3 models. The equal means, ARMA, and ARIMA with d=1 have similar ASEs overall in Table 2 and similar patterns by month-ahead forecast in Fig. 7.



**Fig. 7.** ASE results by month-ahead and model for product/customer combination 2.

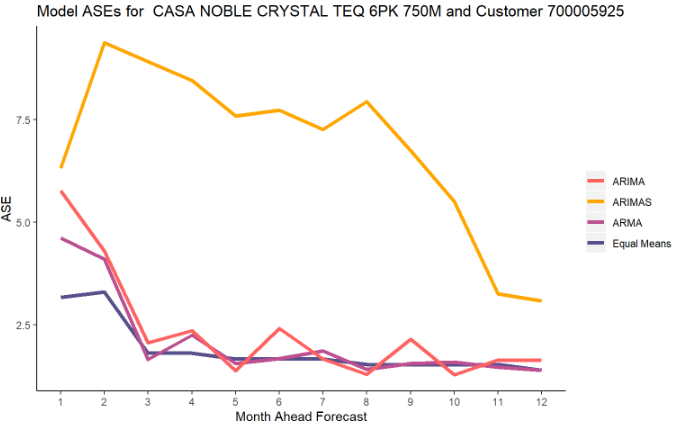
8.3 Product/Customer Combination 4

Table 3 shows model results for product/customer combination 4. The different methods for determining if the time series is white noise have different results for product/customer combination 4. The Ljung-Box tests both indicate the time series is not white noise, however, one of the top 5 ARMA models with the lowest BIC is an ARMA(0,0), indicating the time series is white noise. The winning model for both the 3 month and 12 month forecast horizon is the ARMA model.

**Table 3.** Model results for product/customer combination 4.

|  |  |  |  |
| --- | --- | --- | --- |
| White Noise Results | Model | Rolling-Window ASE | |
| 3 Month Forecast | 12 Month Forecast |
| Inconclusive | Equal Means | 2.40 | 3.17 |
| ARMA | 2.23 | 2.10 |
| ARIMA, d=1 | 2.47 | 2.33 |
| ARIMA, s=12 | 6.11 | 6.84 |

The average ASE at each month-ahead forecast by the different models in the AutoML framework in plotted in Fig. 8. Fig. 8 shows that the forecast accuracies for the ARIMAS with s=12 are worse than the other 3 models. The equal means model has a more constant ASE across the different month-ahead forecasts, but the ARMA was the winning model. There is more evidence to suggest the time series for product/customer combination 4 is not white noise. As such, the ARMA model is an appropriate model for the data.



**Fig. 8.** ASE results by month-ahead and model for product/customer combination 4.

9 Conclusion

Placeholder.

10 Future Work

Placeholder.

References

1. Arora, T., Chandna, R., Conant, S., Sadler, B., & Slater, R. (2020). Demand forecasting in wholesale alcohol distribution: An ensemble approach. SMU Scholar.
2. Jiang, L., Rollins, K. M., Ludlow, M., & Sadler, B. (2020). Demand forecasting for alcoholic beverage distribution. SMU Scholar.
3. Budjač Roman, Marcel, N., Peter, S., Zahradníková Barbora, & Janáčová Dagmar. (2019). Automated machine learning overview. Research Papers.Faculty of Materials Science and Technology.Slovak University of Technology in Trnava, 27(45), 107-112. doi:10.2478/rput-2019-0033.
4. Mohr, F., Wever, M., & Hüllermeier, E. (2018). ML-plan: Automated machine learning via hierarchical planning. Machine Learning, 107(8-10), 1495-1515. doi:10.1007/s10994-018-5735-z.
5. Yan, W. (2012). Toward automatic time-series forecasting using neural networks. IEEE Transactions on Neural Networks and Learning Systems, 23(7), 1028-1039. doi:10.1109/TNNLS.2012.2198074
6. Widodo, A., Budi, I., & Widjaja, B. (2016). Automatic lag selection in time series forecasting using multiple kernel learning. International Journal of Machine Learning and Cybernetics, 7(1), 95–110. https://doi.org/10.1007/s13042-015-0409-7.
7. Allen, A., Balaji, A. (2018). Benchmarking Automatic Machine Learning Frameworks. arXiv.org. <http://search.proquest.com/docview/2092781703/>
8. Waring, J., Lindvall, C., & Umeton, R. (2020). Automated machine learning: Review of the state-of-the-art and opportunities for healthcare.*Artificial Intelligence in Medicine, 104.* doi:10.1016/j.artmed.2020.101822.
9. Ahmed, N. K., Atiya, A. F., Gayar, N. E., & El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. Econometric Reviews: The Link between Statistical Learning Theory and Econometrics: Applications in Econometrics, Finance, and Marketing, 29(5-6), 594-621. doi:10.1080/07474938.2010.481556.
10. Huber, J., & Stuckenschmidt, H. (2020). Daily retail demand forecasting using machine learning with emphasis on calendric special days. International Journal of Forecasting, 36(2). doi:10.1016/j.ijforecast.2020.02.005.
11. Elsayed, N., Maida, A. S., & Bayoumi, M. (2019). Gated recurrent neural networks empirical utilization for time series classification IEEE. doi:10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00202.
12. Jiseong Noh, Hyun-Ji Park, Jong Soo Kim, & Seung-June Hwang. (2020). Gated Recurrent Unit with Genetic Algorithm for Product Demand Forecasting in Supply Chain Management. Mathematics (Basel), 8(565). https://doi.org/10.3390/math8040565
13. Weng, T., Liu, W., & Xiao, J. (2019). Supply chain sales forecasting based on lightGBM and LSTM combination model. Industrial Management & Data Systems, 120(2), 265–279. <https://doi.org/10.1108/IMDS-03-2019-0170>
14. Helmini, S., Jayasinghe, M., & Perera, S. (2019). Sales forecasting using multivariate long short term memory network models. PeerJ PrePrints. https://doi.org/10.7287/peerj.preprints.27712v1
15. Weytjens, H., Lohmann, E., & Kleinsteuber, M. (2019). Cash flow prediction: MLP and LSTM compared to ARIMA and Prophet. Electronic Commerce Research, 1–21. <https://doi.org/10.1007/s10660-019-09362-7>
16. Tuggener, L., Amirian, M., Rombach, K., Lorwald, S., Varlet, A., Westermann, C., & Stadelmann, T. (2019). Automated Machine Learning in Practice: State of the Art and Recent Results. 2019 6th Swiss Conference on Data Science (SDS), 31–36. <https://doi.org/10.1109/SDS.2019.00-11>

(not yet in lit review)

1. Abbasimehr, H., Shabani, M., & Yousefi, M. (2020). An optimized model using LSTM network for demand forecasting. Computers & Industrial Engineering, 143 doi: 10.1016/j.cie.2020.106435.
2. Abolghasemi, M., Beh, E., Tarr, G., & Gerlach, R. (2020). Demand forecasting in supply chain: The impact of demand volatility in the presence of promotion. Computers & Industrial Engineering, 142. doi:10.1016/j.cie.2020.106380.
3. Ali, Ö G., Sayın, S., van Woensel, T., & Fransoo, J. (2009). SKU demand forecasting in the presence of promotions. Expert Systems with Applications, 36(10), 12340-12348. doi:10.1016/j.eswa.2009.04.052.
4. Bottani, E., Centobelli, P., Gallo, M., Kaviani, M. A., Jain, V., & Murino, T. (2019). Modelling wholesale distribution operations: An artificial intelligence framework. Industrial Management & Data Systems, 119(4), 698-718. doi:10.1108/IMDS-04-2018-0164.
5. Freeman, D. G. (2012). Beer in good times and bad: A U.S. state-level analysis of economic conditions and alcohol consumption. Journal of Wine Economics, 6(2), 20. Retrieved from <https://www.cambridge.org/core/journals/journal-of-wine-economics/article/beer-in-good-times-and-bad-a-us-statelevel-analysis-of-economic-conditions-and-alcohol-consumption/44A01D9530005F6CE7C16B101CAD4D17>.
6. Hu, Y., & Huang, S. (2017). Challenges of automated machine learning on causal impact analytics for policy evaluation IEEE. doi:10.1109/TEL-NET.2017.8343571.
7. Kim, M., Choi, W., Jeon, Y., & Liu, L. (2019). A hybrid neural network model for power demand forecasting. Energies, 12(5) doi:10.3390/en12050931.
8. Manthey, J., Shield, K. D., Rylett, M., Hasan, O. S. M., Probst, C., & Rehm, J. (2019). Global alcohol exposure between 1990 and 2017 and forecasts until 2030: A modelling study. The Lancet, 393(10190), 2493-2502. doi:10.1016/S0140-6736(18)32744-2.
9. Zhang, Y., Wang, Y., & Luo, G. (2020). A new optimization algorithm for non-stationary time series prediction based on recurrent neural networks. Future Generation Computer Systems, 102, 738-745. doi:10.1016/j.future.2019.09.018.
10. Olson, R.S. & Moore, J.H.. (2016). TPOT: A Tree-based Pipeline Optimization Tool for Automating Machine Learning. Proceedings of the Workshop on Automatic Machine Learning, in PMLR 64:66-74