|  |
| --- |
| Elmhurst College |
| State of Illinois Revenue Data – Time Series Forecasting |
| MSD 570 Fall A 2019 |

|  |
| --- |
| Megan Cusey  10-13-2019 |

# Overview

At the Illinois Office of the Comptroller, one of the daily tasks is to determine how much revenue in order to advise how much money can be issued on warrants that day. Currently this is done by using a judgmental forecast approach by comparing historical numbers and current conditions. This project seeks to understand the data and the different components of the univariate dataset and apply a series of time series methods to determine the best model. This is a high level approach that will, in a subsequent effort, assist us in automating the judgmental forecast process and allow more a hybrid approach where a model can be applied and, if necessary, a series of consistent steps by the business analyst to determine the best daily figure that leverages all information available.

# Data Exploration

The data used in this project consists of 62 data points that describes the monthly revenue reported by agencies between July 2014 to August 2019. In order to prepare for data exploration, I executed R code that allows me to load the data into a variable from my working directory, convert it into a time series object and produce plots of the original data. See **Figure 1**:

A screenshot of a social media post

Description automatically generated

Figure 1 - Original Plot of Revenue Data

Plotting the original data tells us a few bits of significant information:

* The data displays some characteristics of seasonality
* It appears that there is a slight upward trend between 2017 – 2019
* There is a large spike towards the end of 2017. The reason is unknown. A data transformation may be performed to limit the effect of the outlier and reduce variance.

The following plot (**Figure 2**) tell us more about the seasonality of the data:

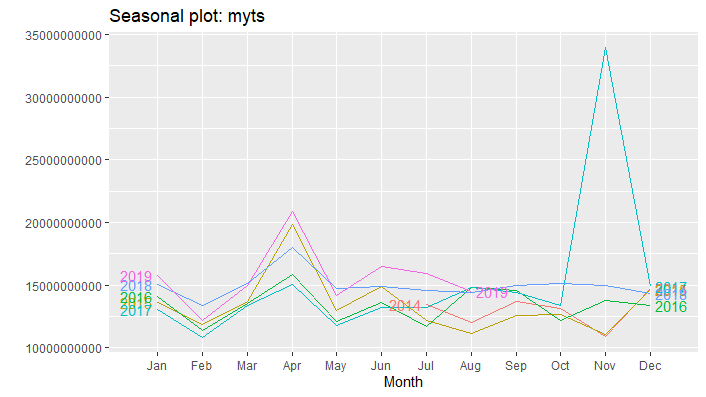


Figure 2 – Seasonality Plot Of Original Data

From the seasonality plot, you can see that each year there is a decrease in revenue between January and February. In addition, there is a spike in revenue in April.

While preparing to continue this project and produce models, I had to consider a few issues regarding the large outlier and how to split my test and train data, if I did so at all.

* One option is to utilize all the data points and evaluate each model based off the statistics produced on the training data alone. The disadvantage of this option is that I don’t get to see how the model performs on new data.
* Splitting the data into test and train limits the data points that can be used to train the data (specifically the most recent data points that may be the most valuable). If I was going to have a test set, it would need to be the length of the seasonality, which is a whole year. This means the latest 12 data points are separated into the test group and 50 data points are used to train the data.
* The way the train and test data splits up, the outlier (November 2017) is the last data point for that month. Some models will apply a good amount of weight to this number for an event that isn’t clear what happened and nothing as unexpected has happened in the rest of the data set.

As a result of these considerations, I decided to create test/train subsets. I believe it is important to see how the model performs against an unseen dataset. I also decided to smooth the November 2017 value by averaging its value with the last observation of the same season (November 2016). As a result, the outlier signal isn’t completely gone, but diluted just a bit.

## Time Series Decomposition

As discussed briefly thus far, time series data typically consists of three elements: trend-cycle, seasonality, and the remainder (everything else). Time Series decomposition seeks to extract each component from a time series dataset.

Time Series Decomposition can apply an additive or multiplicative approach. Additive decomposition is typically applied to datasets that have consistent seasonality whereas multiplicative is designed for a more volatile seasonality component.

As you can see in **Figure 3**, decomposing the dataset results in three graphs that represent the trend-cycle, seasonality, and remainder of the original data. If you combine these three graphs, you would get the original data set. Time Series Decomposition Methods can also create a seasonally adjusted dataset of the original dataset. To seasonally adjust the data is to remove the seasonal component and retain the trend-cycle and remainder components instead. This method is interesting if you want to remove the seasonal impact of the data and explore other predictor variables that cause turning points in the series.

A picture containing text, photo

Description automatically generated

Figure 3 – Time Series Decomposition Using STL Method

The gray bar on the right represents the scale of the data related in each model. Therefor, the range of values for the trend and seasonal components are much less than the original data set (top graph) and the remainder component. The trend component confirms that there is some upward and downward trend going on throughout the data set. An upward trend has been occurring since about 2017. In addition, there is also some seasonality in the data set as indicates by the seasonal graph. However, keep in mind the scale of the original data. These components seem to only make up a small portion of the dataset. Trend appears to play a bigger role than seasonality.

## Model Evaluation

The models produced by the series of methods discussed in this project will be evaluated against each other using a series of values.

RMSE – Root Mean Squared Error; The standard deviation of residuals. RMSE is a metric that measures how far the fitted values are from the observed values.

MAPE – Mean Absolute Percentage Error; Unit free measurement that represents the percentage of errors.

The goal of RMSE and MAPE is to minimize these values. The two metrics can be used to evaluate models across different families of model types. Keep in mind that MAPE puts a heavier penalty on negative errors than positive errors.

ETS and ARIMA models will produce the following information criterion to assist with model evaluation:

AIC – Akaike’s Information Criterion; Assesses each model by considering the fit of the model to observed values while also penalizing model for number of parameters needed to be estimate (i.e. model complexity).

## R Script

|  |
| --- |
| ## LOAD REQUIRED LIBRARIES  library(fpp2)  library(forecast)  library(urca)  ## remove scientific notation  options(scipen = 999)  ## Sometimes my working directory gets reset to my OneDrive so  ## need to set wd when getting started.  wd <- "C:/Users/cusey/source/repos/TimeSeriesProjectts/TimeSeriesProjects/TimeSeriesProjects"  setwd(wd)  ## Import CSV Data.  data <- read.csv("Revenue Data.csv", header = TRUE)  ## Convert to Time Series Object  myts <- ts(data = data[, 'Revenue'], frequency = 12, start = c(2014, 7), end = c(2019, 8))  #########################################################  ## DATA EXPLORATION:  ## Plot Original Data  ## FIGURE 1  autoplot(myts) + xlab("Month/Year") + ylab("Monthly Revenue")+ggtitle("Time Plot of Revenue")  ## Maybe apply transformation to remove variance in outlier before  ## 2018?  BoxCox(myts,BoxCox.lambda(myts)) %>% autoplot()  myts.boxcox <- BoxCox(myts,BoxCox.lambda(myts))  log(myts) %>% autoplot()  ## Plot Original Data  autoplot(myts) + xlab("Month/Year") + ylab("Monthly Revenue")+ggtitle("Time Plot of Revenue")  ## Seasonal Plots  ## FIGURE 2  ggseasonplot(myts, year.labels = TRUE, year.labels.left = TRUE)  ggseasonplot(myts, polar = TRUE, year.labels = TRUE, year.labels.left = TRUE)  ggsubseriesplot(myts)  gglagplot(myts)  ## Time Series Decomposition Figure 3  myts %>% stl(t.window=13, s.window="periodic", robust=TRUE) %>% autoplot  ## 11/1/2017 is the outlier, taking the mean of all November 2017 and November 2016  ## I don't like that it's one of the last observations seen in the training data  ## and is used to forecast the test data. I think it's more important that we smooth  ## the value and get an entire year for the test data. The alternative is to only  ## evaluate the models based on training data which I don't want to do either.  replace.outlier <- mean(c(13760762315,33983939406))  myts[41] <- replace.outlier  ## Plot Original Data W/ Smoothed Outlier  autoplot(myts) + xlab("Month/Year") + ylab("Monthly Revenue")+ggtitle("Time Plot of Revenue")  ######################################################  ## Split into Test/Train Data Sets  myts.train <- window(myts, start=c(2014,7), end=c(2018,08))  myts.test <- window(myts, start=c(2018,09), end=c(2019,8))  myts.h <- 12  ####################################################### |

# Simple Methods

## Simple Method Models and Forecasts

Before diving into more complex models, producing models using simpler methods helps to provide a good benchmark and learn more about the data. Each model is built with a subset of the data called the training set. The models are tested for accuracy using the rest of the data called the test data set.

**Figure 4** displays the forecast results of each simple method evaluated for the data set. **Table 1** lists each model, the description, RMSE and MAPE values and if there is a trend or seasonality component in the model.

As a result of the simple model method evaluations, the Drift Method is noted to be the best method before diving deeper into more complex algorithms. Also be advised that this method applies a trend component and not a seasonality component. The Drift Method RMSE and MAPE will be used as benchmarks going forward to compare against more complicated methods.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **Description** | **RMSE** | **MAPE** | **TREND** | **SEASONALITY** |
| **Drift Method** | **Averages the change in the historical data to calculate a slope and uses the resulting line to forecast future values.** | **2114826753** | **8.127975** | **Yes** | **No** |
| Naïve Method | Forecasts all future values to the last observation. | 2178374718 | 4.924187 | No | No |
| Average Method | Forecasts all future values to the average of the historical data | 2465492089 | 10.80514 | No | No |
| Seasonal Naïve | Forecasts all future values to the last observation of the same season. | 2848729922 | 10.822929 | No | Yes |

Table 1 – Results of Simple Method Models

A screenshot of a cell phone

Description automatically generated

Figure 4 – Forecast Results of the Simple Methods

## Check Residuals and Assumptions

Since it has been determined that the Drift Method yields the lowest RMSE, the next step is to review the residuals of the model.

Residuals of a time series model represents the information left over after fitting the model. In other words, residuals are the difference of the forecasted (or fitted) values that the model provided and the observed values of the training set. The following properties will be investigated:

1. The residuals of the model are not correlated. If correlation is observed in a model, it means that there is information in the left-over data that isn’t being utilized in the model. Uncorrelated residuals indicate that the remaining information is white noise.
2. The residuals have a mean of zero. Residuals that do not have a mean of zero is an indicator that the forecasts are biased.
3. The residuals have a constant variance.
4. The residuals are normally distributed.

If the first two properties are not met, then the model needs to be improved. The last two properties are useful in order to make prediction intervals more accurate.

**Figure 5** displays several graphs of the residuals for Drift Method.

A screenshot of a map

Description automatically generated

Figure 5 – Drift Method Residuals

The top graph is a plot of the residuals. The data points hover around 0 suggesting that the mean is about 0. The bottom left graph tests for the autocorrelation of the residuals. In lag 1 and 13, it slightly extends past the point of significance but it not too bad or too concerning for this model. The bottom right graph is a histogram of the distribution of the residuals. The distribution is certainly not normal. I am also a bit concerned with the variance of the data in the residual plot (top graph). With this information, I believe the residuals do not indicate a bias in the forecasts or unaccounted information, but I would be leery of the prediction intervals calculated as a result of this model.

## R Script

|  |
| --- |
| #######################################################  ## Simple Approaches  ## Average Method  fit.average.method <- myts.train %>% meanf(h=myts.h)  accuracy(fit.average.method, myts.test)  ## Naive Method  fit.naive.method <- naive(myts.train, h = myts.h)  accuracy(fit.naive.method, myts.test)  ## Seasonal Naive Method  fit.seasonal.naive.method <- snaive(myts.train, h = myts.h)  accuracy(fit.seasonal.naive.method, myts.test)  ## Drift Method  fit.drift.method <- rwf(myts.train, h = myts.h, drift = TRUE)  accuracy(fit.drift.method, myts.test)  ## FIGURE 4  autoplot(myts) +  autolayer(fit.average.method, series = "Average Method", PI = FALSE) +  autolayer(fit.naive.method, series = "Naive Method", PI = FALSE) +  autolayer(fit.seasonal.naive.method, series = "Seasonal Naive Method", PI = FALSE) +  autolayer(fit.drift.method, series = "Drift Method", PI = FALSE) +  xlab("Month/Year") + ylab("Revenue") + ggtitle("Simple Methods Forecast Results")  ## SEASONAL NAIVE METHOD WITH BOXCOX TRANSFORMATION  ## PRODUCES THE LOWEST RMSE.  ## CHECK RESIDUALS – FIGURE 5  checkresiduals(residuals(fit.drift.method)) |

R Script 2 – Code Used to Execute Simple Method Forecasts

# Linear Regression

## Linear Regression Model and Forecasts

The concepts of linear regression can be applied to a time series data set to identify the linear relationship between y (the value we want to forecast) and x (the predictor variable). As a result, linear regression builds a line of best fit which reduced errors between the fitted values and the training data set.

**Table 2** supplies the results of the linear regression model while **Figure 6** provides a graph of the forecasts. Note that the Linear Regression proves to supply better forecast values and a lower RMSE/MAPE than our benchmark, the Drift Method. The Linear Regression Method considers both the trend and seasonality of the data unlike the Drift Method.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **Description** | **RMSE** | **MAPE** | **TREND** | **SEASONALITY** |
| **Linear Regression** | **Uses the linear relationship between the forecast variable (y) and the predictor variable (x) to obtain the forecast value.** | **1084874242** | **5.814253** | **Yes** | **Yes** |

Table 2 – Results of Linear Regression Model

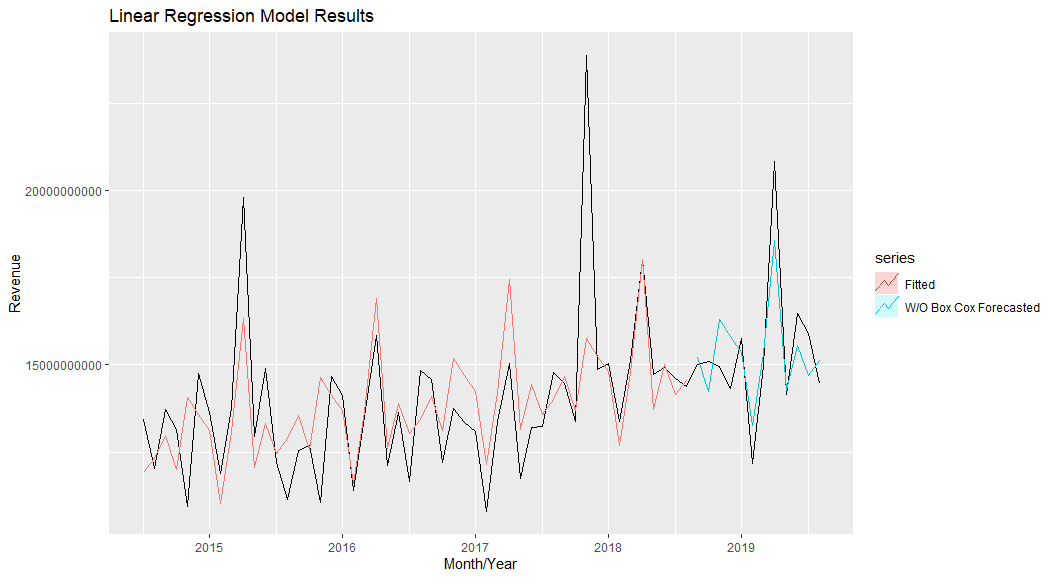


Figure 6 – Linear Regression Method Residuals

## Check Residuals and Assumptions

Like the methods used in the Simple Methods section, the residuals of a linear regression model should have an average of zero and no correlation. See **Figure 7** below. While the variance in the residuals aren’t systematic and the histogram seems skewed, the Breusch-Godfrey test doesn’t show significant autocorrelations and the residuals seems to hover adequately around 0. With this said, it appears the linear regression model utilized the most information from the data to calculate its model and the leftover residuals are white noise. The prediction intervals, however, will be more difficult the calculate since the variance of the residuals is sporadic and the distribution isn’t quite normal.

A close up of a map

Description automatically generated

Figure 7 – Linear Regression Method Residuals

**Figure 8** is a scatter plot that graphs the residuals against the fitted values of the linear regression model. If a systemic pattern displays in the scattered plot, then it suggests that a nonlinear relationship exists between the data. In that case, the model may need to be modified. Otherwise, if the scatter plot points are random then that is what is expected of a linear regression model. In **Figure 6**, the data points appear to be random indicating a nonlinear relationship does not exist.

A screenshot of a cell phone

Description automatically generated

Figure 8 – Linear Regression Method Residuals VS Fitted Values Scatter Plot

## R Script

|  |
| --- |
| #############################################################  ## Linear Regression  fit.linear.regression <- tslm(myts.train~ trend+season)  fit.linear.regression.forecast <- fit.linear.regression %>% forecast(h=myts.h)  accuracy(fit.linear.regression.forecast, myts.test)  ## FIGURE 5  autoplot(myts) +  autolayer(fitted(fit.linear.regression), series = "Fitted") +  autolayer(fit.linear.regression.forecast, series = "W/O Box Cox Forecasted", PI = FALSE) +  xlab("Month/Year") + ylab("Revenue") + ggtitle("Linear Regression Model Results")  ## check residuals  autoplot(residuals(fit.linear.regression))  ## FIGURE 7  qplot(fitted(fit.linear.regression), residuals(fit.linear.regression))  ## FIGURE 6  checkresiduals(fit.linear.regression)  ## a small p value indicates there is significant autocorrelation  ## remaining in the residuals  ## Breusch-Godfrey test for serial correlation of order up to 16  ##data: Residuals from Linear regression model  ##LM test = 16.011, df = 16, p-value = 0.4522  ############################################################# |

R Script 3 – Code Used to Produce Linear Regression Model Residual Evaluation

# Exponential Smoothing

## Exponential Smoothing Models and Forecasts

The main concept being Exponential Smoothing is to optimize weighted averages for different components of the data set. For example, in Simple Exponential Smoothing, the method improves upon concepts introduced by the Averaging Method and the Naïve Method. As you may recall, the Averaging Method averages all the observations in the training set and sets all future values to result. Similarly, the Naïve Method sets all future values to the value of the last observation. Simple Exponential Smoothing optimizes a smoothing parameter, or the weights of the first n data points, and applies it to the overall data. The more complex models in the Exponential Smoothing family introduces the same concept except applies smoothing parameters to trend and seasonality also. The concept of characterizing components of the data as multiplicative or additive also plays a role in Exponential Smoothing. In fact the Error, Trend, and Seasonality (ETS) method characterizes each component (if it applies) as additive or multiplicative. For example, an (A,A,M) ETS model would calculate the error component as additive, the trend component as additive, and the seasonality component as multiplicative.

**Figure 9** and **Figure 10** display the forecast results. **Table 3** displays the metrics produced by the various Exponential Smoothing Method Models. The model from that minimized RMSE the most from this family of methods is the Holt Winter’s Seasonal Multiplicative Method.A screenshot of a cell phone

Description automatically generated

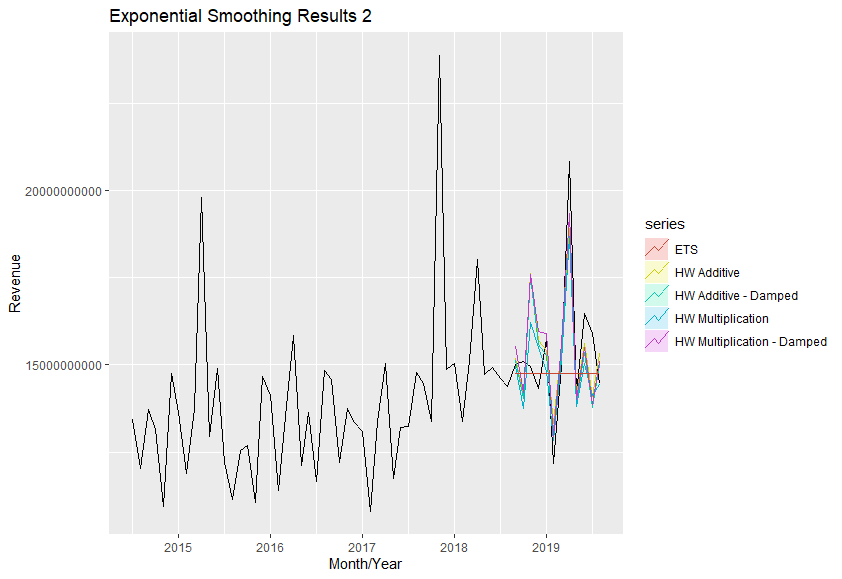
Figure 9 – Exponential Smoothing Forecast Results, Plot 1

Figure 10 – Exponential Smoothing Forecast Results Continued, Plot 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **Description** | **RMSE** | **MAPE** | **TREND** | **SEASONALITY** |
| **Holt-Winters' Seasonal Method Multiplicative** | **Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Multiplicative - The variation of the seasonal component is proportional to the level of the series.** | **1163163677** | **5.916772** | **Yes** | **Yes** |
| Holt-Winters' Seasonal Method  Multiplicative - Damped | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Multiplicative - The variation of the seasonal component is proportional to the level of the series.  Damped - Parameter that seeks to reduce over forecasting values. | 1272878987 | 6.771351 | Yes | Yes |
| Holt-Winters' Seasonal Method Additive | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Additive - Describes the seasonal component. Additive is most appropriate when the seasonal variances is typically consistent throughout the time series. | 1295679000 | 6.847373 | Yes | Yes |
| Holt-Winters' Seasonal Method Additive - Damped | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Additive - Describes the seasonal component. Additive is most appropriate when the seasonal variances is typically consistent throughout the time series.  Damped - Parameter that seeks to reduce over forecasting values. | 1332604597 | 6.929021 | Yes | Yes |
| Holt's Linear Trend w Damped | Extends Simple Exponential Smoothing by applying a trend component. This method often over forecasts values. The damped parameter seeks to reduce over forecasting value. | 2031747746 | 7.522392 | Yes | No |
| Holt's Linear Trend | Extends Simple Exponential Smoothing by applying a trend component. | 2039305482 | -5.822928 | Yes | No |
| Simple Exponential Smoothing (SES) | Assigns a smoothing parameter that optimizes the fitted data to the observations and applies a weight the x number of data values from the last observed values. Compare to Naïve Method were the last observation has a weight of 1 (all of the weight) Compare to the Average Method were all observations have an equal weight. | 2045734084 | 7.607342 | No | No |
| ETS (Error, Trend, Seasonality) State Space Model - pick ETS(A,N,N) | ETS models estimates the Error, Trend, and Seasonality parameters by maximizing likelihood. The parameters include Error (Additive or Multiplication), Trend (None, Additive, Multiplicative), Seasonality (None, Additive, Multiplicative). | 2194712527 | 9.360251 | Yes | Yes |

Table 3 – Results of Exponential Smoothing Method Models

## Check Residuals and Assumptions

The residuals from the Holt-Winters’ Seasonal Multiplicative Method looks similar to the Drift Methods and Linear Regression in the characteristics that the residual variances aren’t stable, and the distribution isn’t normal. Again, these two facts make prediction intervals a bit more unreliable and harder to calculate. However, it doesn’t have significant autocorrelation and the residuals appear to have a mean of about zero which indicate that the forecasts are not bias and there isn’t any information left out of the model.

A close up of a map

Description automatically generated

Figure 11 – Residuals From the Holt-Winters Seasonal Multiplicative Method

## R Script

|  |
| --- |
| #############################################################  ## Exponential Smoothing  ## Simple Exponential Smoothing  fit.ses <- ses(myts.train,h=myts.h)  accuracy(fit.ses,myts.test)  fit.ses[["model"]]  ## Holt Linear Trend  fit.holt.linear.trend <- holt(myts.train,h=myts.h)  accuracy(fit.holt.linear.trend,myts.test)  fit.holt.linear.trend[["model"]]  ## Holt Linear Trend Damped  fit.holt.linear.trend.damped <- holt(myts.train,h=myts.h, damped=TRUE)  accuracy(fit.holt.linear.trend.damped,myts.test)  fit.holt.linear.trend.damped[["model"]]  ## Holt Winters Seasonal Method  ## Additive  fit.hw.additive <- hw(myts.train, h=myts.h, seasonal="additive")  accuracy(fit.hw.additive,myts.test)  fit.hw.additive[["model"]]  fit.hw.additive.damped <- hw(myts.train, h=myts.h, seasonal="additive", damped=TRUE)  accuracy(fit.hw.additive.damped,myts.test)  fit.hw.additive.damped[["model"]]  ## Multiplicative  fit.hw.multiplicative <- hw(myts.train, h=myts.h, seasonal="multiplicative")  accuracy(fit.hw.multiplicative,myts.test)  fit.hw.multiplicative[["model"]]  fit.hw.multiplicative.damped <- hw(myts.train, h=myts.h, seasonal="multiplicative", damped=TRUE)  accuracy(fit.hw.multiplicative.damped,myts.test)  fit.hw.multiplicative.damped[["model"]]  ## ETS – State Space Model  fit.ets <- ets(myts.train)  summary(fit.ets,myts.test)  forecast.ets <- fit.ets %>% forecast(h=myts.h)  ## Figure XX  autoplot(myts) +  autolayer(fit.ses, series = "Simple E.S.", PI = FALSE) +  autolayer(fit.holt.linear.trend, series = "Holt Linear W/O Damp", PI = FALSE) +  autolayer(fit.holt.linear.trend.damped, series = "Holt Linear W/ Damp", PI = FALSE) +  xlab("Month/Year") + ylab("Revenue") + ggtitle("Exponential Smoothing Results 1")  ## Figure XX  autoplot(myts) +  autolayer(fit.hw.additive, series = "HW Additive", PI = FALSE) +  autolayer(fit.hw.additive.damped, series = "HW Additive - Damped", PI = FALSE) +  autolayer(fit.hw.multiplicative, series = "HW Multiplication", PI = FALSE) +  autolayer(fit.hw.multiplicative.damped, series = "HW Multiplication - Damped", PI = FALSE) +  autolayer(forecast.ets, series = "ETS", PI = FALSE) +  xlab("Month/Year") + ylab("Revenue") + ggtitle("Exponential Smoothing Results 2")  ## Figure XX  checkresiduals(fit.hw.multiplicative)  ######################################################## |

R Script 5 – Code Used to Produce Exponential Smoothing Model

# ARIMA

## ARIMA Models and Forecasts

Autoregression, Moving Average, and ARIMA models require data to be stationary. Most data are not stationary; therefore, a series of methods can be applied to obtain stationary data. The first differencing method can be applied which takes each observation subtracted by its previous observation. Seasonal differencing maintains the seasonal relationship in a data set by subtracting each observation by the previous observation of the same seasonality. Each time a differencing step takes place, the model loses a degree of freedom. This is important to know for evaluating models. Information criterion such as AIC cannot be used to compare the performance of models with different degrees of freedom. Some data may need to be difference more than once and sometimes one of each differencing step may apply. In my R Script, you’ll see I ran several tests to see which differencing steps may be necessary. In addition to differencing, stationary may be obtained by performing a data transformation or seasonally adjusting the data.

ARIMA (AutoRegressive Integrated Moving Average) implements both Autoregressive and Moving Average approaches into one model. Autoregressive Models uses the target variable’s previous observations to create a line of regression to create it’s forecast values. Moving Average is similar, but it uses the previous forecast errors to create it’s forecast values.

**Figure 11** displays the ARIMA Method Model results. **Table 4** displays the ARIMA Method Model metrics used to evaluate the models. I used auto.arima() which is supposed to take a step-wise approach to finding the most optimal ARIMA model. The result was a random walk forecast which is typical of financial data anyway. I also tried just a plain Moving Average model and AutoRegressive Model. I was surprised to see that the Moving Average model minimized the RMSE more than what the auto.arima() function came up with.

A screenshot of a cell phone

Description automatically generated

Figure 11 – ARIMA Model Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model Name** | **Description** | **RMSE** | **MAPE** | **TREND** | **SEASONALITY** |
| Moving Average | The moving average model uses its previous forecast errors to produce a regression like model for future forecast values. | 2193409890 | 9.360846 | Yes | Yes |
| Autoregressive | The autoregressive model uses previous values of the dependent variable (the variable we want to forecast) and creates a linear line of regression to produce forecast values. | 2592679768 | 13.2813 | Yes | Yes |
| ARIMA | ARIMA stands for AutoRegressive Integrated Moving Average model. It applies both concepts of the MA and AR models with stationary data. | 2999944065 | 15.02368 | Yes | Yes |

Table 4 – Results of ARIMA Method Models

## Check Residuals and Assumptions

While the Moving Average Model residuals are not autocorrelated, the rest of the graphs do not indicate that the model will produces as well of results as some of our other methods have throughout this project. The residual mean does not appear to be zero which indicates that there is some forecast bias. Forecast bias is easy to adjust by adding the mean of the residuals to each forecast result. However, the residual variance also seems unstable and the distribution of the residuals do not appear to be normal. This is just an instance where the most complex model doesn’t equate to being the best model to use.

A screenshot of a social media post

Description automatically generated

Figure 12 – Moving Average Residuals

## R Script

|  |
| --- |
| ########################################################  ## ARIMA  ## DIFFERENCING  myts %>% diff() %>% autoplot()  myts %>% diff(lag=12) %>% autoplot()  ## small p-values (less than .05) indicate  ## more differencing is required  myts %>% diff() %>% ur.kpss() %>% summary()  myts %>% diff(lag=12) %>% ur.kpss() %>% summary()  ## first differencing appears to be better  myts %>% ndiffs()  myts %>% nsdiffs()  ## no seaonal differencing is required.  ## ACF & PACF  myts %>% diff() %>% ggAcf()  myts %>% diff() %>% ggPacf()  fit.ARIMA <- myts.train %>% auto.arima()  fit.ARIMA.forecast <- fit.ARIMA %>% forecast(h=myts.h)  accuracy(fit.ARIMA.forecast, myts.test)  summary(fit.ARIMA,myts.test)  fit.AR <- myts.train %>% Arima(order=c(1,1,0))  fit.AR.forecast <- fit.AR %>% forecast(h=myts.h)  summary(fit.AR,myts.test)  fit.MA <- myts.train %>% Arima(order=c(0,1,1))  fit.MA.forecast <- fit.MA %>% forecast(h=myts.h)  summary(fit.MA,myts.test)  ## FIGURE XX  autoplot(myts) +  autolayer(fit.MA.forecast, series = "Moving Average", PI = FALSE) +  autolayer(fit.AR.forecast, series = "Autoregression", PI = FALSE) +  autolayer(fit.ARIMA.forecast, series = "ARIMA", PI = FALSE) +  xlab("Month/Year") + ylab("Revenue") + ggtitle("ARIMA Results")  ## Figure XX  checkresiduals(fit.MA.forecast) |

R Script 6 – Code Used to Produce Arima Model Methods

# Conclusion

To conclude, the Linear Regression model appears to be the best model for this dataset as it reduces both RMSE and MAPE the best. The Linear Regression model does a good job implementing the seasonal and trend components, but over forecasting the components either. In addition, it appears to handle the outlier quite nicely, not giving it too much weight to throw off the entire forecast.

For the next steps in this project, I would like to see how the average daily data does with the same methods applied. I believe that some of the remainder component could have a hint of additional seasonality we are missing by aggregating per month. I also think that there is a few key predictor variables that we could be missing such as legislative changes that have occurred over the years and different leadership in the Comptroller’s Office. In addition, I’d like to get some perspective as to why there was a big hike in November of 2017, why February appears to have a consistent decrease in revenue, and why April has a consistent peak in revenue. These are factors I would like to continue to investigate.

Additional steps could be performed that might also improve this model such as producing forecasts on decomposed data or seeing if data transformations (instead of smoothing the outlier) improves the quality of the forecast. I did play around with performing a data transformation (BoxCox) at the start of this project and found that the smoothed outlier approach typically performed better than the transformed data set, but other data transformations could be applied.

# Results Table

The below table shows the results of all the models, sorted lowest RMSE to highest.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model Name** | **Model Method** | **Description** | **RMSE** | **MAPE** | **AIC** | **AICc** | **TREND** | **SEASONALITY** |
| Linear Regression | Regression | Uses the linear relationship between the forecast variable (y) and the predictor variable (x) to obtain the forecast value. | 1084874242 | 5.814253 |  |  | Yes | Yes |
| Holt-Winters' Seasonal Method Multiplicative | Exponential Smoothing | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Multiplicative - The variation of the seasonal component is proportional to the level of the series. | 1163163677 | 5.916772 | 2350.6 | 2370 | Yes | Yes |
| Holt-Winters' Seasonal Method  Multiplicative - Damped | Exponential Smoothing | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Multiplicative - The variation of the seasonal component is proportional to the level of the series.  Damped - Parameter that seeks to reduce over forecasting values. | 1272878987 | 6.771351 | 2350.5 | 2373 | Yes | Yes |
| Holt-Winters' Seasonal Method Additive | Exponential Smoothing | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Additive - Describes the seasonal component. Additive is most appropriate when the seasonal variances is typically consistent throughout the time series. | 1295679000 | 6.847373 | 2359.8 | 2379 | Yes | Yes |
| Holt-Winters' Seasonal Method Additive - Damped | Exponential Smoothing | Extends Simple Exponential Smoothing by applying both a trend and seasonal component.  Additive - Describes the seasonal component. Additive is most appropriate when the seasonal variances is typically consistent throughout the time series.  Damped - Parameter that seeks to reduce over forecasting values. | 1332604597 | 6.929021 | 2362.5 | 2385 | Yes | Yes |
| Holt's Linear Trend w Damped | Exponential Smoothing | Extends Simple Exponential Smoothing by applying a trend component. This method often over forecasts values. The damped parameter seeks to reduce over forecasating value. | 2031747746 | 7.522392 | 2358.6 | 2361 | Yes | No |
| Holt's Linear Trend | Exponential Smoothing | Extends Simple Exponential Smoothing by applying a trend component. | 2039305482 | -5.822928 | 2363.3 | 2365 | Yes | No |
| Simple Exponential Smoothing (SES) | Exponential Smoothing | Assigns a smoothing parameter that optimizes the fitted data to the observations and applies a weight the x number of data values from the last observed values. Compare to Naïve Method were the last observation has a weight of 1 (all of the weight) Compare to the Average Method were all observations have an equal weight. | 2045734084 | 7.607342 | 2352.5 | 2353 | No | No |
| Drift Method | Simple | Averages the change in the historical data to calculate a slope and uses the resulting line to forecast future values. | 2114826753 | 8.127975 |  |  | Yes | No |
| Naïve Method | Simple | Forecasts all future values to the last observation. | 2178374718 | 4.924187 |  |  | No | No |
| Moving Average | ARIMA | The moving average model uses its previous forecast errors to produce a regression like model for future forecast values. | 2193409890 | 9.360846 | 2253.4 | 2254 | Yes | Yes |
| ETS (Error, Trend, Seasonality) State Space Model - pick ETS(A,N,N) | Exponential Smoothing | ETS models estimates the Error, Trend, and Seasonality parameters by maximizing likelihood. The parameters include Error (Additive or Multiplication), Trend (None, Additive, Multiplicative), Seasonality (None, Additive, Multiplicative). | 2194712527 | 9.360251 | 2352.5 | 2353 | Yes | Yes |
| Average Method | Simple | Forecasts all future values to the average of the historical data | 2465492089 | 10.80514 |  |  | No | No |
| Autoregressive | ARIMA | The autoregressive model uses previous values of the dependent variable (the variable we want to forecast) and creates a linear line of regression to produce forecast values. | 2592679768 | 13.2813 | 2268.6 | 2269 | Yes | Yes |
| Seasonal Naïve | Simple | Forecasts all future values to the last observation of the same season. | 2848729922 | 10.822929 |  |  | No | Yes |
| ARIMA | ARIMA | ARIMA stands for AutoRegressive Integrated Moving Average model. It applies both concepts of the MA and AR models with stationary data. | 2999944065 | 15.02368 | 2280.6 | 2281 | Yes | Yes |