**Part I:  Research Question**

1. Describe the purpose of this data analysis by doing the following:
2. Can we reliably forecast changes in daily revenue with an ARIMA model on the supplied timeseries data?
3. This analysis has a series of connected goals. First, we want to identify any seasonal components or trends. Then, we want to identify the best ARIMA model for the timeseries. Finally, we will use this model to forecast the next month’s daily revenue.

**Part II:  Method Justification**

1. Time series analysis has two major assumptions. The first assumption is that time series data is stationary. As the name implies, stationarity is a characteristic of data that has a stable, unchanging mean, variance, and autocorrelation. Stationarity may not be present in the raw time series data, but is achievable through simple transformations, such as taking the difference, or change per time step, of the time series or performing a log transformation on the values of the time series.

Autocorrelation is our second assumption; it is a characteristic of data where any value is dependent upon one or more previous values in the dataset. We require data to be autocorrelated in time series analysis because the time series models derive their predictions based on the relationship between past and future values.

There are two additional assumptions related to the nature of the data rather than it’s contents. These assumptions are that there are no anomalies, as anomalies interfere with the autocorrelation assumption described previously, and that the model parameters and error term are constant.

**Part III:  Data Preparation**

1. Summarize the data cleaning process by doing the following:
2. Input:

data <- read.csv('medical\_time\_series.csv')

ts\_data <- ts(data[, 2])

ts.plot(ts\_data)

Output:

Chart, line chart

Description automatically generated

1. The data dictionary tells us that the data consists of two years of daily data, showing us that the time step of the time series is one day. The data dictionary does not provide us with a date for the start of the data and the days variable is not in date format, so we cannot discern the date at which data collection began or if any of the days are missing. However, the days variable has no gaps so the analysis may continue.

Input:

data$Day

Output:

A picture containing text

Description automatically generated

1. We use an augmented Dickey-Fuller test to evaluate the stationarity of the timeseries.

Input:

adf.test(ts\_data)

Output:

Text

Description automatically generated

We see that the p-value is greater than 0.05, so we fail to reject the null hypothesis. We see that the alternative hypothesis is that the timeseries is stationary. Combining this information, we can conclude that the timeseries is not stationary.

1. We choose the test data to be approximately the last five months of the time series as there appears to be some level of monthly, seasonal change and five months is approximately 20% of two years, which is the length of the time series data.

Input:

ts\_data.train <- ts\_data[1:580]

ts\_data.test <- data[580:731, 2]

ts\_data.test <- ts(ts\_data.test, start=580)

We then take the first difference of the training data and evaluate it for stationarity as we have seen that the undifferenced data is not stationary.

Input:

diff\_ts <- diff(ts\_data)

ts.plot(diff\_ts)

adf.test(diff\_ts)

Output:

Chart

Description automatically generated

Text

Description automatically generated

The resulting plot does not indicate any trend or seasonality and the Augmented Dickey-Fuller confirms that the differenced data is stationary.

1. The differenced, train, and test data are provided as the attachments diff.csv, train.csv, and test.csv respectively.

**Part IV:  Model Identification and Analysis**

1. Analyze the time series dataset by doing the following:
2. Report the annotated findings with visualizations of your data analysis, including the following elements:
   1. In order to determine the seasonal component, if there is one, of our time series, we must create the time series with a monthly frequency. We estimate this frequency at 30 days as the provided data does not include the dates, but only the time step. Taking the decomposition of this monthly time series divides the time series into the trend, seasonal, and residual components. We then plot the seasonal component to check for a seasonal trend.

Input:

ts\_data\_months <- ts(data[, 2], start=1, frequency=30)

decomposed\_ts <- decompose(ts\_data\_months)

plot(decomposed\_ts$seasonal)

Output:

A picture containing chart

Description automatically generated

The graph shows that each month, the hospital’s revenue alternates between increasing and decreasing with peaks in the middle of the month and troughs at the end/beginning of the month.

* 1. We use the previously computed decomposition to visualize the trend of the data.

Input:

plot(decomposed\_ts$trend)

Output:

Chart, line chart

Description automatically generated

We see from the trend that the hospital started with a negative revenue, rising sharply over the first ten months. After this, the data appears to oscillate as it converges in the last months of the data.

* 1. We use the acf and pacf functions to plot the autocorrelation function and partial autocorrelation function of the time series, respectively.

Input:

acf(ts\_data, lag.max=365)

pacf(ts\_data, lag.max=10)

Output:

Chart, histogram

Description automatically generatedChart

Description automatically generated with low confidence

The acf function shows us that we have a strong, positive correlation up to approximately 150 lags. However, the pacf functions shows us that, when accounting for all previous lags, only one lag provides meaningful information. These plots tell us that our data is most likely to be best modeled by an AR-1 process, as evidenced by the pacf having only one significant lag (Salvi, 2021). Performing these tests on the differenced, stationary data will give us a deeper look at these processes.

Input:

acf(diff\_ts, lag.max=10)

pacf(diff\_ts, lag.max = 10)

Output:

Chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

The acf and pacf of the differenced data confirms the presence of an AR-1 process, as evidenced by the pacf having only one significant lag, as well as an MA-3 process, as evidenced by the acf having three significant lags (Salvi, 2021).

* 1. Input:

spectrum(diff\_ts)

Output:

A picture containing graphical user interface

Description automatically generated

Spectral Density of diff\_ts

The spectral density graph of the differenced time series data shown above does not appear to be representative of any trend or seasonality, as we would expect from differenced, normalized data. The periodogram does decrease slightly as the frequency increases, though it never grows much beyond 0.5. The low value of the spectral density, coupled with the apparently random distribution of the spectral density, implies that there is no periodic trend in the differenced data.

* 1. The decomposed time series includes the trend, seasonal component, and residuals of the initial time series. The trend and seasonal component have been discussed in previous parts of the analysis, but the residual is a new component within this analysis. The residuals are simply what remains of the original time series once the seasonality and trend have been removed. The residuals of the time series are included in the decomposition as the random variable of the decomposition.

Input:

plot(decomposed\_ts)

Output:

Histogram

Description automatically generated with low confidence

* 1. We can confirm that the residuals are without a trend by performing an Augmented Dickey-Fuller test on the residuals of the decomposed time series.

Input:

res <- ts(na.omit(decomposed\_ts$random))

adf.test(res)

Output:

Text

Description automatically generated

We see that the Augmented Dickey-Fuller test confirms that the residual data is stationary and contains no trend. This is supported by analyzing the graph of the residuals, called random in the decomposition, in the part e above. This visualization appears to have no trend, positive or negative, and appears to oscillate around the mean of zero.

1. Using the results of the acf and pacf functions performed earlier, we will determine a range of likely values for p, representing the degree of the AR process of the time series, and q, representing the degree of the MA process of the time series. We will examine the AIC and BIC scores of every combination of these terms, both differenced and not, on the differenced time series. We will evaluate the AIC and BIC scores to determine the most effective model parameters.

Input:

a <- c()

b <- c()

i <- 1

for (p in 0:2) {

for (d in 0:1) {

for (q in 0:4) {

model <- arima(diff\_ts, order=c(p,d,q))

a[i] <- AIC(model)

b[i] <- BIC(model)

print(paste('model', as.character(i), ' -- AIC: ', round(a[i], 2),

' BIC: ', round(b[i], 2), ' Params: ', p, d, q, sep=''))

i <- i+1

}

}

}

min(a)

min(b)

Output:

Text

Description automatically generated

We see that the lowest AIC score belongs to model3, with the parameters (0, 0, 2), which is an MA-2 model with no AR component. The lowest BIC score belongs to model11, with the parameters (1, 0, 0), which is an AR-1 model with no MA component. This information supports the findings of our previous analyses that there is an AR-1 component and an MA-2 or MA-3 component. When these models are combined, in model13 with parameters (1, 0, 2), we see that the AIC score is between the AIC scores of model3 and model11. As this accounts for the observed AR and MA components, sacrificing a small amount of AIC and BIC score is acceptable to ensure we capture all aspects of the model. We arrive at the conclusion to use the ARIMA model with parameters (1, 0, 2) as our final model.

1. We will use the selected ARIMA model from the previous part to perform a forecast 150 days into the future, the length of the test set.

Input:

fin\_model <- arima(ts\_data.train, order = c(1,0,2))

pred <- predict(fin\_model, n.ahead=150)

upper\_bound <- pred$pred+2\*pred$se

lower\_bound <- pred$pred-2\*pred$se

predictions <- pred$pred

plot.ts(ts\_data, main='Forecast of Revenue', ylab='Revenue (Millions of Dollars)', xlab='Time (Days)')

lines(predictions, col='blue')

lines(upper\_bound, col='red')

lines(lower\_bound, col='red')

lines(ts\_data.test, col='purple')

legend('topleft',

legend=c('Training Data', 'Predictions', 'Error Boundaries', 'Test Data'),

col=c('black', 'blue', 'red', 'purple'),

pch=15

)

Output:

Chart, line chart

Description automatically generated

In black, we see the training set of the time series. The test set is included in purple. The forecast’s predictions are shown in blue with the margin of error, two standard errors from the prediction, shown in red. We see that the test set remains within the boundaries of the error lines, meaning the real data was within two standard errors of the forecasted values. Additionally, the forecast correctly predicts the slight, negative trend of the data over the five month forecast.

We calculate the mean-squared error of the predictions compared with the test data to see how well it fits the test data.

Input:

mean((ts\_data.test - pred$pred)^2)

Output:



The mean-squared error of the predictions compared to the trend is also calculated to show how well the forecast fits the trend of the test data.

Input:

ts.test.monthly <- ts(data[580:731, 2], start=580, frequency=30)

decomposed.test <- decompose(ts.test.monthly)

pred.monthly <- ts(pred$pred, start=580, frequency=30)

mean((na.omit(decomposed.test$trend) - na.omit(pred.monthly))^2)

Output:



We see that the mse is lower when compared to the trend of the test data, confirming that the model predicts the trend of the time series better than the day-to-day changes in revenue.

1. Outputs for every calculation are provided in the previous sections, immediately preceded by the code that generated it.
2. Provide the code used to support the implementation of the time series model.

data <- read.csv('medical\_time\_series.csv')

ts\_data <- ts(data[, 2])

ts.plot(ts\_data)

data$Day

library(tseries)

adf.test(ts\_data)

#nonstationary b/c p-val > 0.05

ts\_data.train <- ts\_data[1:580]

ts\_data.test <- data[580:731, 2]

ts\_data.test <- ts(ts\_data.test, start=580)

diff\_ts <- diff(ts\_data)

ts.plot(diff\_ts)

adf.test(diff\_ts)

ts\_data\_months <- ts(data[, 2], start=1, frequency=30)

decomposed\_ts <- decompose(ts\_data\_months)

plot(decomposed\_ts$seasonal)

plot(decomposed\_ts$trend)

acf(ts\_data, lag.max=365)

pacf(ts\_data, lag.max=10)

acf(diff\_ts, lag.max=10)

pacf(diff\_ts, lag.max = 10)

spectrum(ts\_data)

spectrum(diff\_ts)

plot(decomposed\_ts)

res <- ts(na.omit(decomposed\_ts$random))

adf.test(res)

library(forecast)

a <- c()

b <- c()

i <- 1

for (p in 0:2) {

for (d in 0:1) {

for (q in 0:4) {

model <- arima(diff\_ts, order=c(p,d,q))

a[i] <- AIC(model)

b[i] <- BIC(model)

print(paste('model', as.character(i), ' -- AIC: ', round(a[i], 2),

' BIC: ', round(b[i], 2), ' Params: ', p, d, q, sep=''))

i <- i+1

}

}

}

min(a)

min(b)

fin\_model <- arima(ts\_data.train, order = c(1,0,2))

pred <- predict(fin\_model, n.ahead=150)

fin\_model <- arima(ts\_data.train, order = c(1,0,2))

pred <- predict(fin\_model, n.ahead=150)

upper\_bound <- pred$pred+2\*pred$se

lower\_bound <- pred$pred-2\*pred$se

predictions <- pred$pred

plot.ts(ts\_data, main='Forecast of Revenue', ylab='Revenue (Millions of Dollars)', xlab='Time (Days)')

lines(predictions, col='blue')

lines(upper\_bound, col='red')

lines(lower\_bound, col='red')

lines(ts\_data.test, col='purple')

legend('topleft',

legend=c('Training Data', 'Predictions', 'Error Boundaries', 'Test Data'),

col=c('black', 'blue', 'red', 'purple'),

pch=15

)

mean((ts\_data.test - pred$pred)^2)

ts.test.monthly <- ts(data[580:731, 2], start=580, frequency=30)

decomposed.test <- decompose(ts.test.monthly)

pred.monthly <- ts(pred$pred, start=580, frequency=30)

mean((na.omit(decomposed.test$trend) - na.omit(pred.monthly))^2)

**Part V:  Data Summary and Implications**

1. Summarize your findings and assumptions, including the following points:
2. Discuss the results of your data analysis, including the following:
   1. The selected ARIMA model, with parameters (1, 0, 2), was discovered by evaluating a variety of ARIMA models with parameters similar to the components observed in the analysis of the time series’ acf and pacf graphs. The acf and pacf told us that the model showed both AR and MA components that we estimated to be best modeled by an AR-1 and MA-3 component. In order to discern the best possible model, we set the range of parameters to evaluate every combination of p from zero to two and q from zero to four, both differenced and not differenced, where p represents the degree of AR component and q represents the degree of the MA component. The AIC and BIC scores of all of these models are seen in part D.2, with lower scores in both being preferable. The pure AR-1 model scores the best on AIC and a pure MA-2 model scores the best on BIC. Based on our previous observation that the timeseries exhibits characteristics of both an AR and an MA process, we decide to combine these and see that this combination leads to a minimal sacrifice in AIC and BIC so not much accuracy is sacrificed.
   2. The predictions of this ARIMA model are daily predictions, mimicking the time step of the time series. As the forecast graph shows, the difference in the daily prediction is minimal and does not capture the daily change in the way the time series does, but rather captures the trend of the data, with increasing margin of error as time passes.
   3. We chose the forecast length to be 150 days, approximately five months, to match the test data. Additionally, this length of forecast allows hospital staff to see the forecasted trend over the next quarter and adjust decision making to account for anticipated drops or rises in average revenue.
   4. The model is evaluated by computing the mean-squared error of the predictions compared to the test data and the value of the mean-squared error of this model is 7.627175 as shown in the screenshot in part D.3. This mean-squared error implies that the prediction has a fair amount of difference from the daily revenue reported in the test data. When compared with the trend of the test data, the mean-squared error drops to 5.911304, as shown in the screenshot in part D.3, confirming that the forecast is more effective at predicting the trend of the revenue than the daily revenue values.
3. The annotated visualization is included in part D.3 and the output will be provided again here.

Chart, line chart

Description automatically generated

1. Based on the forecast, it appears that revenue will be trending slightly lower, and this is supported by the test data, indicating that the hospital executives will need to account for this when budgeting. If previous revenue levels are desired, the hospital will need to increase pricing of services. However, the peaks of the revenue in the test set are only $5 million below the peaks at the highest levels of revenue in the training data and may be acceptable levels as long as executives can stave off any further drops through marketing and ideal pricing.

**Part VI:  Reporting**

1. The R Markdown report is included as the attachment report.html.
2. All code was hand-written and no online sources were used to generate code.
3. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

Salvi, J. (2019, March 27). Significance of ACF and PACF plots in time series analysis. Medium. Retrieved November 8, 2021, from https://towardsdatascience.com/significance-of-acf-and-pacf-plots-in-time-series-analysis-2fa11a5d10a8.