D213: Advanced Data Analytics

Performance Assessment

Task 1

Logan Rosemeyer

Western Governor's University

D213: Advanced Data Analytics

Dr. Festus Elleh

March 27, 2023

Contents

A1: Research Question

My research question is: Will the company's revenue increase or decrease in the future?

A2: Objectives and Goals

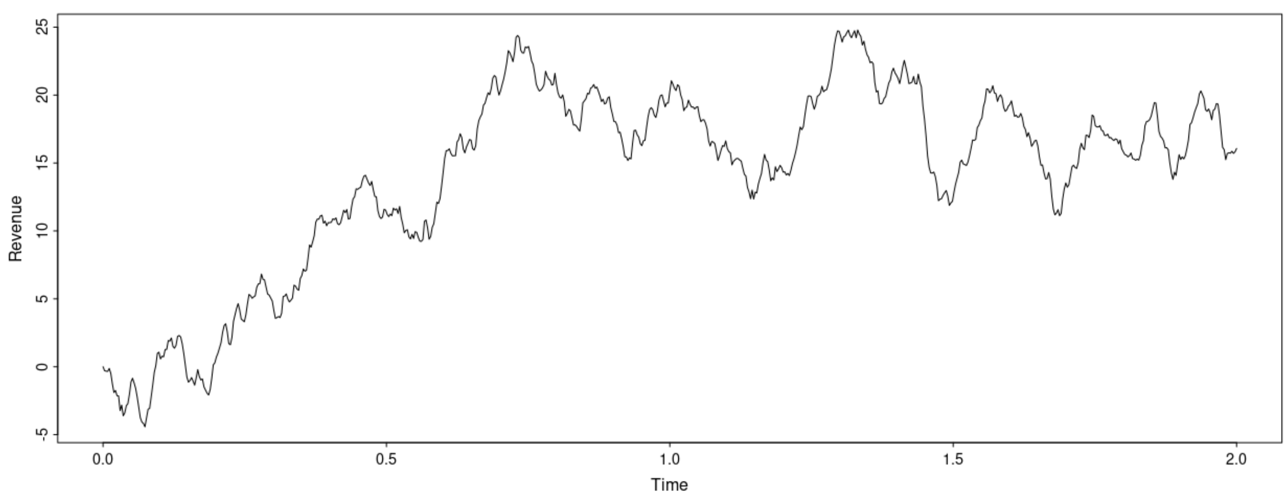
The goal of the analysis is to forecast the data to predict whether revenue will increase or not.

B: Summary of Assumptions

One assumption of a time series model is that the data is stationary. A stationary dataset has the property that the mean, variance, and autocorrelation structure do not change over time. (NIST)

C1: Line Graph Visualization

Here is a line graph of my data as a time series.



C2: Time Step Formatting

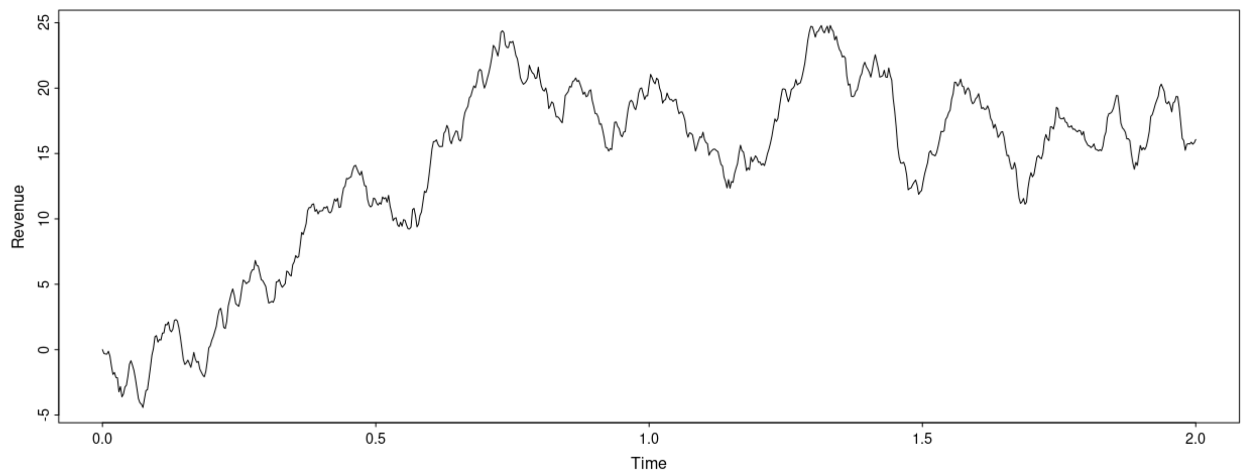
Below is the code I used to format the data to a time series.

```
med <- ts(med, start=0, frequency = 365)
```

I started at 0 and used a frequency of 365 because the data was taken daily. This way a time of 1 would represent 1 year. There were no gaps in the sequence, and the data was recorded daily for two years.

C3: Stationarity

The data on its own was not stationary as you can see from the graph below.

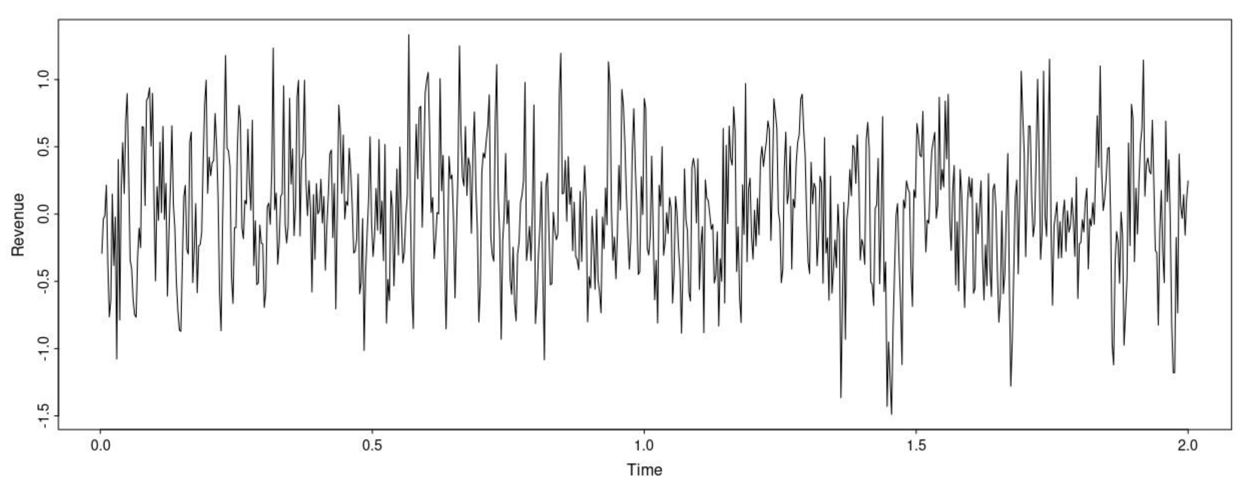


I also ran the Augmented Dickey-Fuller Test so make sure the data was not stationary. With a p-value of 0.542, the data is definitely not stationary.

```
> adfTest(med)

Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 1
  STATISTIC:
    Dickey-Fuller: -0.2333
  P VALUE:
    0.542
```

I used the diff function in R to make the data stationary. The code used and the stationary line graph are shown below.

```
d_med<-diff(med)
```



After that, I ran the Augmented Dickey-Fuller test to make sure the data was stationary. With a p-value of 0.01.

```
> adfTest(d_med)

 Title:
  Augmented Dickey-Fuller Test

 Test Results:
   PARAMETER:
     Lag Order: 1
   STATISTIC:
     Dickey-Fuller: -14.4301
   P VALUE:
     0.01
```

C4: Steps to Prepare the Data

The first thing I did was to check for duplicates and nulls. Then, I made the data into a time series. Finally, I split the data into train and test data. I used an 80/20 split.

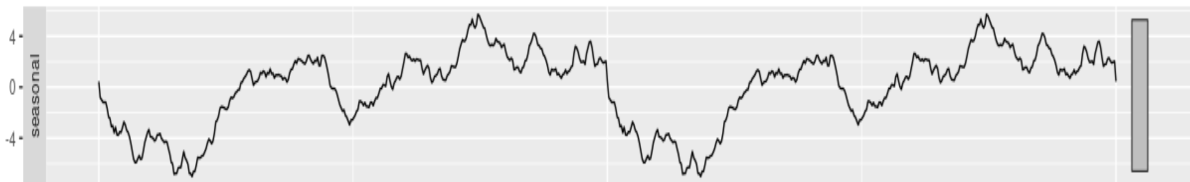
```
# Clean data

sum(duplicated(med))

sum(is.na(med))

print(med)

dim(med)


 # Create main time series

med <- ts(med, start=0, frequency = 365)

plot(med)


 # Create train and test data

med_train <- head(med, 585)

plot(med_train)

print(med_train)


med_test <- tail(med,146)

plot(med_test)

print(med_test)
```

C5: Prepared Dataset

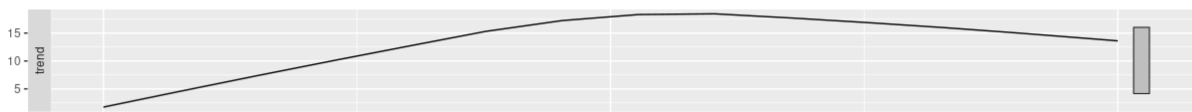
A copy of the cleaned train and test data is included in the submission. They are named "med_train" and "med_test".

D1: Report Findings and Visualizations

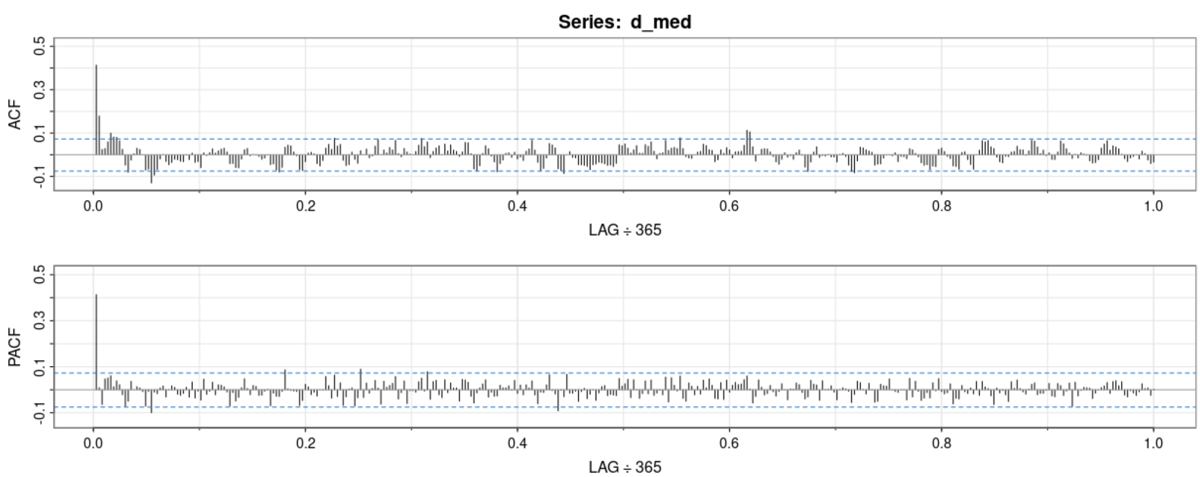
There is a seasonal component, which you can see from the following decomposed plot. The seasonal plot typically decreases at the start of the period, and then will increase to above where it started. Where time is equal to 1, the data decreases again, and again increases to above where it starts.



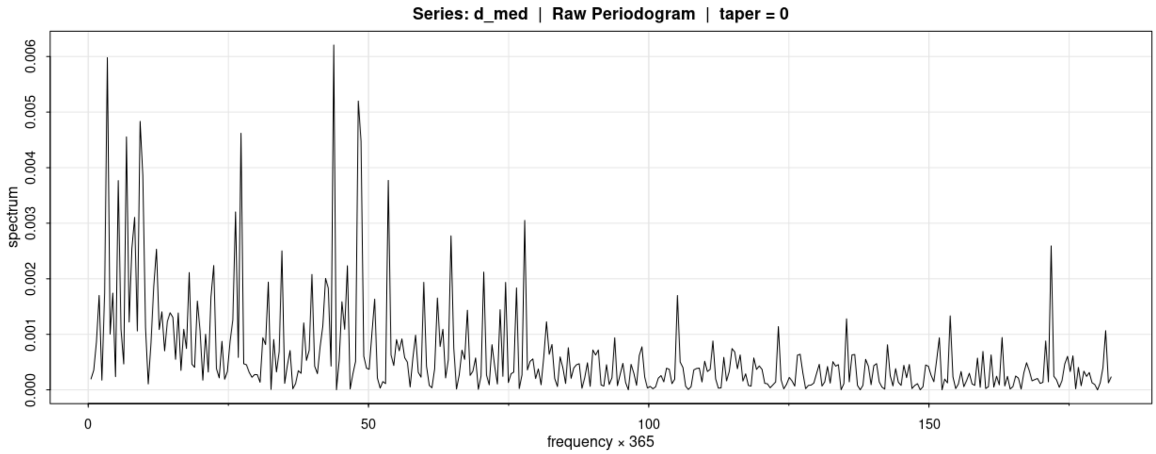
There was an observed increasing trend in the data.



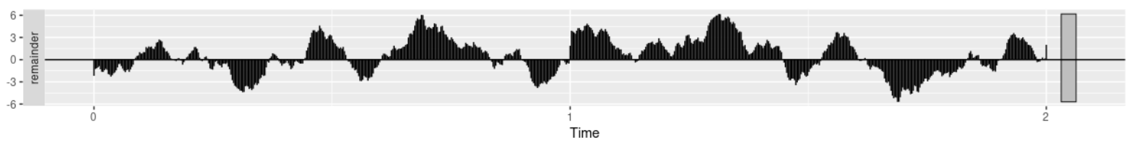
I also have plots of my autocorrelation function and partial autocorrelation function.



Next is the spectral density plot.



Finally, I have the plot of my residuals from my decomposed time series. This centers around 0.

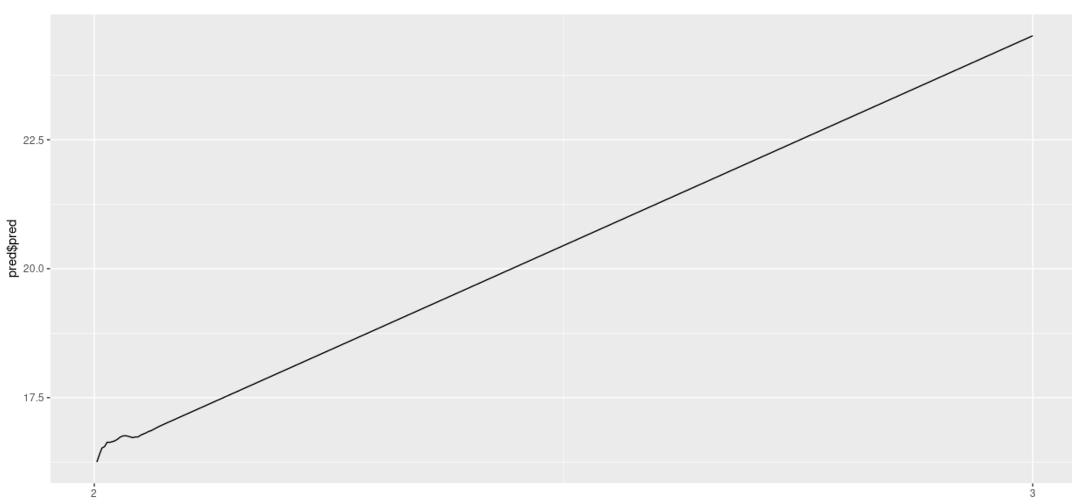


D2: ARIMA Model

I began finding the best arima model by running the auto.arima function in R. It gave me the best model of (1,1,0). Because there was definitely seasonality in the data, I decided to guess and check the seasonality component until I got an AIC lower than my auto.arima output. I ended up with $(1,1,0)(0,0,2)_{12}$. This improved my AIC from 1.207342 to 1.205866.

D3: Forcasting Using ARIMA Model

Below is a plot of my predicted data, from the start of year 2 to the start of year 3. Below that is the actual values of my prediction.



```
> pred <- sarima.for(med,n.ahead = 365,1,1,0,0,0,2,12, plot.all = T)
> pred
$pred
Time Series:
Start = c(2, 2)
End = c(3, 1)
Frequency = 365
  [1] 16.25019 16.39668 16.52359 16.55066 16.63557 16.63316 16.64640 16.66259
  [9] 16.69213 16.73147 16.75599 16.76201 16.75342 16.73773 16.72335 16.73725
 [17] 16.73688 16.77106 16.79373 16.81290 16.84000 16.85699 16.88254 16.91078
 [25] 16.93553 16.95884 16.98155 17.00402 17.02638 17.04870 17.07101 17.09330
 [33] 17.11560 17.13789 17.16018 17.18247 17.20476 17.22705 17.24934 17.27163
 [41] 17.29392 17.31621 17.33850 17.36079 17.38308 17.40537 17.42766 17.44995
 [49] 17.47224 17.49454 17.51683 17.53912 17.56141 17.58370 17.60599 17.62828
 [57] 17.65057 17.67286 17.69515 17.71744 17.73973 17.76202 17.78431 17.80660
 [65] 17.82889 17.85118 17.87347 17.89576 17.91805 17.94035 17.96264 17.98493
 [73] 18.00722 18.02951 18.05180 18.07409 18.09638 18.11867 18.14096 18.16325
 [81] 18.18554 18.20783 18.23012 18.25241 18.27470 18.29699 18.31928 18.34157
 [89] 18.36386 18.38616 18.40845 18.43074 18.45303 18.47532 18.49761 18.51990
 [97] 18.54219 18.56448 18.58677 18.60906 18.63135 18.65364 18.67593 18.69822
[105] 18.72051 18.74280 18.76509 18.78738 18.80967 18.83197 18.85426 18.87655
[113] 18.89884 18.92113 18.94342 18.96571 18.98800 19.01029 19.03258 19.05487
[121] 19.07716 19.09945 19.12174 19.14403 19.16632 19.18861 19.21090 19.23319
[129] 19.25548 19.27778 19.30007 19.32236 19.34465 19.36694 19.38923 19.41152
[137] 19.43381 19.45610 19.47839 19.50068 19.52297 19.54526 19.56755 19.58984
[145] 19.61213 19.63442 19.65671 19.67900 19.70130 19.72359 19.74588 19.76817
[153] 19.79046 19.81275 19.83504 19.85733 19.87962 19.90191 19.92420 19.94649
[161] 19.96878 19.99107 20.01336 20.03565 20.05794 20.08023 20.10252 20.12481
[169] 20.14711 20.16940 20.19169 20.21398 20.23627 20.25856 20.28085 20.30314
[177] 20.32543 20.34772 20.37001 20.39230 20.41459 20.43688 20.45917 20.48146
[185] 20.50375 20.52604 20.54833 20.57062 20.59292 20.61521 20.63750 20.65979
[193] 20.68208 20.70437 20.72666 20.74895 20.77124 20.79353 20.81582 20.83811
[201] 20.86040 20.88269 20.90498 20.92727 20.94956 20.97185 20.99414 21.01643
[209] 21.03873 21.06102 21.08331 21.10560 21.12789 21.15018 21.17247 21.19476
[217] 21.21705 21.23934 21.26163 21.28392 21.30621 21.32850 21.35079 21.37308
[225] 21.39537 21.41766 21.43995 21.46224 21.48454 21.50683 21.52912 21.55141
[233] 21.57370 21.59599 21.61828 21.64057 21.66286 21.68515 21.70744 21.72973
[241] 21.75202 21.77431 21.79660 21.81889 21.84118 21.86347 21.88576 21.90805
[249] 21.93035 21.95264 21.97493 21.99722 22.01951 22.04180 22.06409 22.08638
[257] 22.10867 22.13096 22.15325 22.17554 22.19783 22.22012 22.24241 22.26470
[265] 22.28699 22.30928 22.33157 22.35386 22.37616 22.39845 22.42074 22.44303
[273] 22.46532 22.48761 22.50990 22.53219 22.55448 22.57677 22.59906 22.62135
[281] 22.64364 22.66593 22.68822 22.71051 22.73280 22.75509 22.77738 22.79967
[289] 22.82197 22.84426 22.86655 22.88884 22.91113 22.93342 22.95571 22.97800
[297] 23.00029 23.02258 23.04487 23.06716 23.08945 23.11174 23.13403 23.15632
[305] 23.17861 23.20090 23.22319 23.24549 23.26778 23.29007 23.31236 23.33465
[313] 23.35694 23.37923 23.40152 23.42381 23.44610 23.46839 23.49068 23.51297
[321] 23.53526 23.55755 23.57984 23.60213 23.62442 23.64671 23.66900 23.69130
[329] 23.71359 23.73588 23.75817 23.78046 23.80275 23.82504 23.84733 23.86962
[337] 23.89191 23.91420 23.93649 23.95878 23.98107 24.00336 24.02565 24.04794
[345] 24.07023 24.09252 24.11481 24.13711 24.15940 24.18169 24.20398 24.22627
[353] 24.24856 24.27085 24.29314 24.31543 24.33772 24.36001 24.38230 24.40459
[361] 24.42688 24.44917 24.47146 24.49375 24.51604
```

D4: Output and Calculations

Here is my use of the auto.arima function.

```
> arima_med <- auto.arima(med)
> arima_med
Series: med
ARIMA(1,1,0)

Coefficients:
         ar1
      0.4142
s.e.  0.0336

sigma^2 = 0.1946:  log likelihood = -437.99
AIC=879.98   AICc=880   BIC=889.17
```

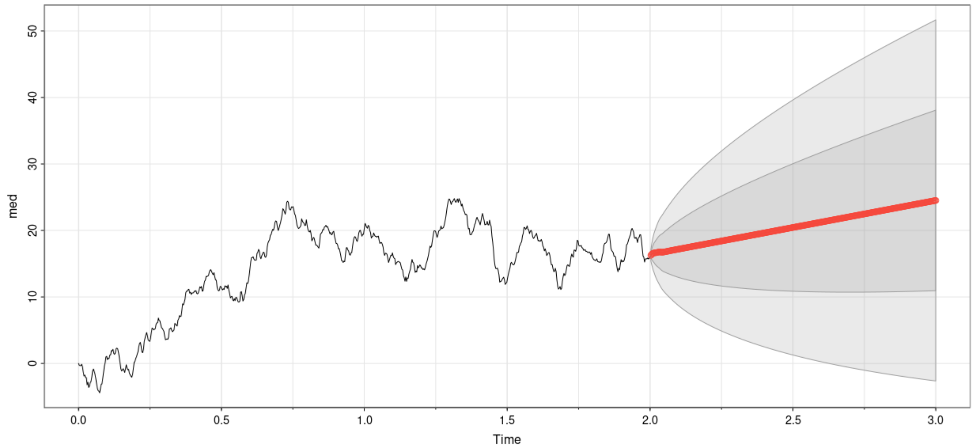
Then I ran the sarima function with the given values.

```
> sarima(med,1,1,0) #1.207342
initial  value -0.725411
iter   2 value -0.819109
iter   3 value -0.819109
iter   4 value -0.819110
iter   5 value -0.819110
iter   6 value -0.819110
iter   6 value -0.819110
final  value -0.819110
converged
initial  value -0.819376
iter   2 value -0.819376
iter   3 value -0.819377
iter   4 value -0.819377
iter   5 value -0.819377
iter   6 value -0.819377
iter   6 value -0.819377
final  value -0.819377
converged
$fit

Call:
arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period =
S),
    xreg = constant, transform.pars = trans, fixed = fixed, optim.control = list(t
race = trc,
        REPORT = 1, reltol = tol))

Coefficients:
         ar1  constant
      0.4130    0.0219
s.e.  0.0337    0.0278

sigma^2 estimated as 0.1942:  log likelihood = -437.68,  aic = 881.36

$degrees_of_freedom
[1] 728

$ttable
         Estimate     SE t.value p.value
ar1        0.4130 0.0337 12.2657  0.0000
constant   0.0219 0.0278  0.7895  0.4301

$AIC
[1] 1.207342

$AICc
[1] 1.207364

$BIC
[1] 1.226217
```

After that, I guessed and checked to improve my model with a seasonal component.

```
> sarima(med,1,1,0,0,0,2,12)
initial  value -0.725411
iter   2 value -0.822293
iter   3 value -0.822653
iter   4 value -0.822655
iter   5 value -0.822655
iter   6 value -0.822656
iter   7 value -0.822656
iter   8 value -0.822656
iter   8 value -0.822656
iter   8 value -0.822656
final  value -0.822656
converged
initial  value -0.822852
iter   2 value -0.822854
iter   3 value -0.822854
iter   4 value -0.822855
iter   5 value -0.822855
iter   5 value -0.822855
iter   5 value -0.822855
final  value -0.822855
converged
$fit

Call:
arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D, Q), period =
S),
    xreg = constant, transform.pars = trans, fixed = fixed, optim.control = list(t
race = trc,
        REPORT = 1, reltol = tol))

Coefficients:
         ar1     sma1    sma2  constant
      0.4142  -0.0799  0.0289    0.0223
s.e.  0.0337   0.0376  0.0380    0.0263

sigma^2 estimated as 0.1928:  log likelihood = -435.14,  aic = 880.28

$degrees_of_freedom
[1] 726

$ttable
         Estimate     SE t.value p.value
ar1        0.4142 0.0337 12.2805  0.0000
sma1      -0.0799 0.0376 -2.1266  0.0338
sma2       0.0289 0.0380  0.7594  0.4479
constant   0.0223 0.0263  0.8470  0.3973

$AIC
[1] 1.205866

$AICc
[1] 1.205942

$BIC
[1] 1.237325
```

Finally, I plotted my forecast.



D5: Code

Below is my code for the ARIMA model.

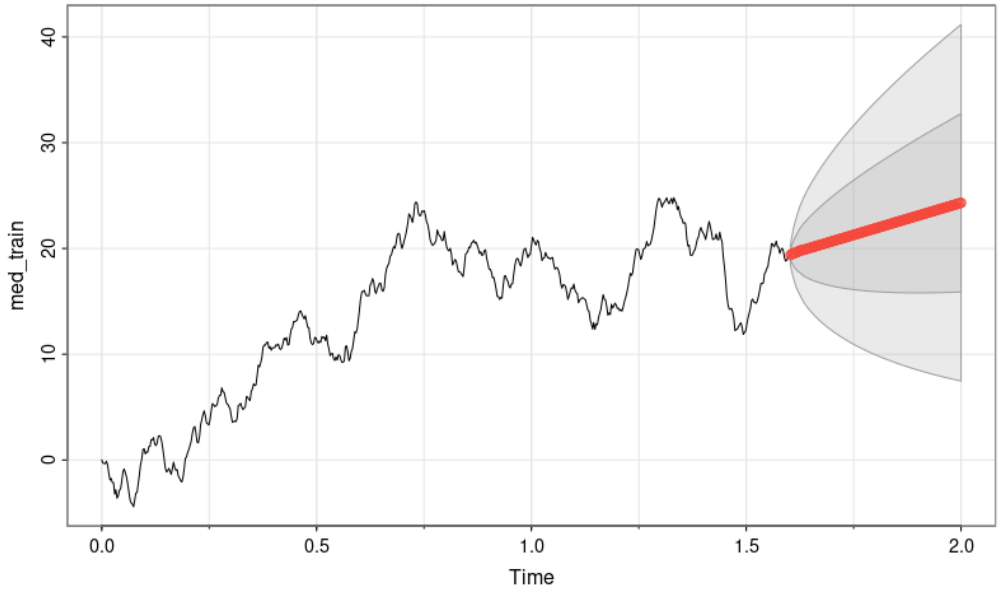
```
 # Run auto arima
arima_med <- auto.arima(med)
arima_med
arima_med_train <- auto.arima(med_train)
arima_med_train

sarima(med,1,1,0) #1.207342

sarima(med,1,1,0,0,0,2,12)

sarima(med_train,1,1,0)
sarima(med_train,1,1,0,0,0,2,12)

pred <- sarima.for(med,n.ahead = 365,1,1,0,0,0,2,12, plot.all = T)
pred

med_pred <- sarima.for(med_train, n.ahead=146,1,1,0,0,0,2,12, plot.all = T) + autolayer(med_test)
```

My entire code is also included in my submission.

E1: Results

I ended up selecting an ARIMA model with the following values: p=1, d=1, q=0, P=0, D=0, Q=2, S=12. There is a wide prediction interval. After only a half year, the revenue ranges from just above 0 to nearly 40. I chose a forecast length of 146 days because that is the same length of my train data.

E2: Annotated Visualization



E3: Recommendations

Based on the results, I would recommend the company look at why the beginning of each year has a little dip. Overall, our revenue is increasing, but minimizing the dip at the beginning the year would help us to maximize our revenue.

F: Reporting

A report has been included in the submission.

G: Sources for Third-Party Code

Auto.arima: Fit best ARIMA model to univariate time series. RDocumentation. Retrieved March 28, 2023. https://www.rdocumentation.org/packages/forecast/versions/8.21/topics/auto.arima

sarima.for: ARIMA Forecasting. RDocumentation. Retrieved March 28, 2023. https://www.rdocumentation.org/packages/astsa/versions/2.0/topics/sarima.for

H: Sources

Stationarity. National Institute of Standards and Technology. Retrieved March 27, 2023. https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc442.htm#:~:text=A%20common%20assumption%20in%20many,do%20not%20change%20over%20time.