Item Demand

Naomi Zubeldia, Sam Spackman

2023-03-03

0

```
#libraries
library(tidyverse)
library(nlme)
library(tidyverse)
library(GGally)
library(car)
library(MASS)
library(lmtest)
library(multcomp)
library(forecast)
#reading in data and creating yearmon variable
item<-read.csv('C:/Users/naomi/OneDrive/Desktop/STAT469/ItemDemandd.csv',header=T)
item$YrMon <- item$year + (item$month - 0.5)/12
```

1.

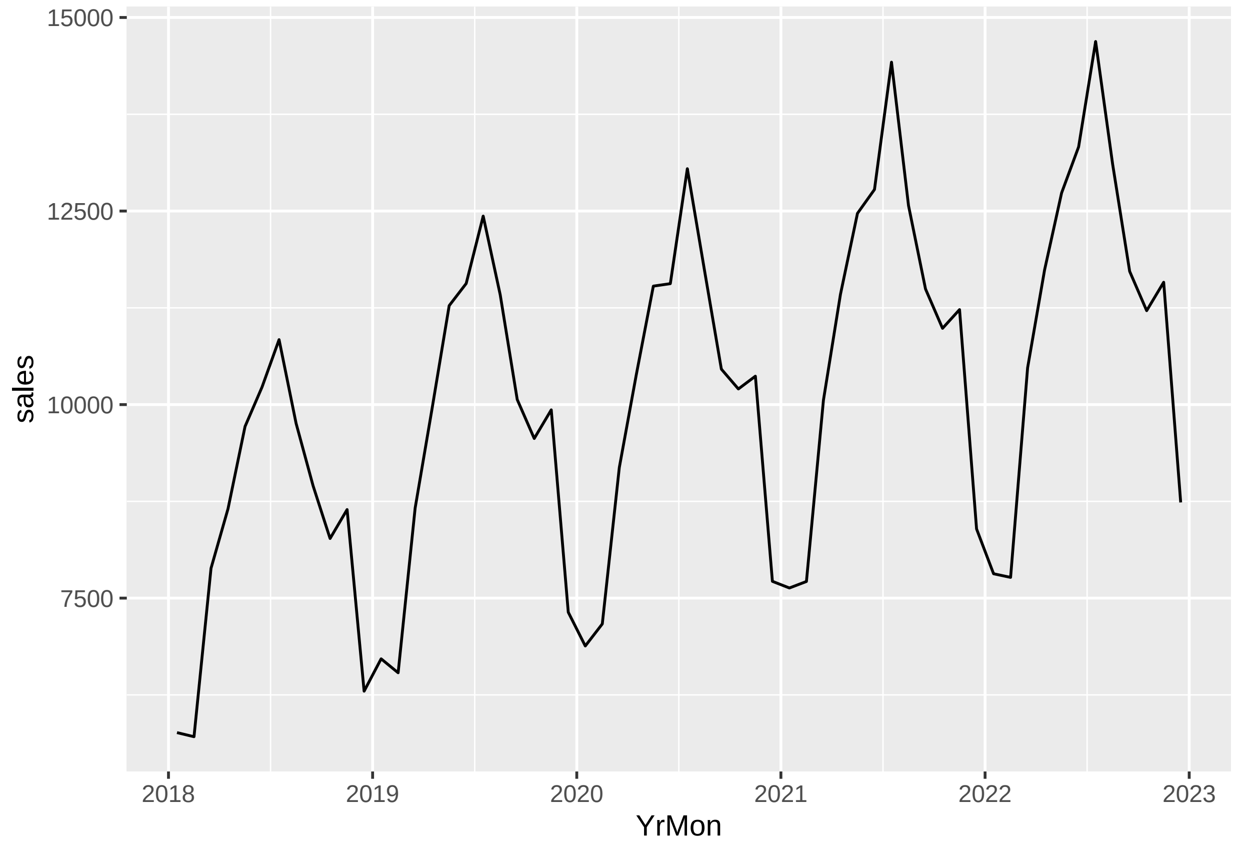
To observe the data and understand the relationship of the sales variable with other variables, we created the summary table below. It looks like the data ranges from the year 2018-2023 over 12 months each year, which shows how fairly recent the data is. The sales variable has a wide range of 5710 to 14690. It appears that the sales of the product over the 5 years had a big increase at some point, which it will be interesting to see when that was.

```
#summary table of the data
summary(item)
```

```
##       year          month           sales           YrMon     
##  Min.   :2018   Min.   : 1.00   Min.   : 5710   Min.   :2018  
##  1st Qu.:2019   1st Qu.: 3.75   1st Qu.: 8363   1st Qu.:2019  
##  Median :2020   Median : 6.50   Median :10214   Median :2020  
##  Mean   :2020   Mean   : 6.50   Mean   :10041   Mean   :2020  
##  3rd Qu.:2021   3rd Qu.: 9.25   3rd Qu.:11562   3rd Qu.:2022  
##  Max.   :2022   Max.   :12.00   Max.   :14690   Max.   :2023  
```

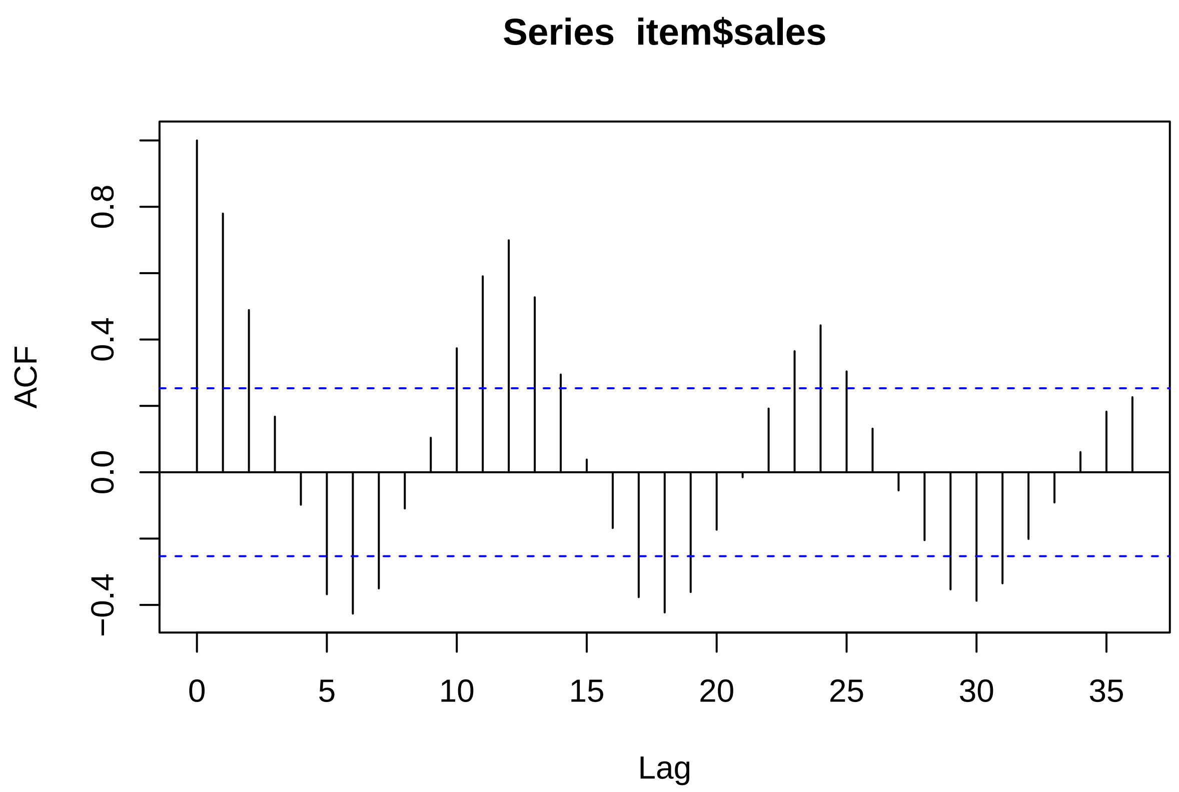
We then created a time series plot, which is attached below, to see the sales over the five year period. It looks like the beginning and end of each year have a decrease in sales with the middle of the year around June being the highest point. Even with the yearly cycles, the sales appear to increase from year to year.

```r
#plotting a time series plot with the smooth line between dots
ggplot(data=item, mapping=aes(x=YrMon,y=sales)) + geom_line()
```



To see how the time series is correlated within itself, we created the ACF plot below. We can see that there are many lags outside of the significance threshold showing their strong correlation. It looks like there is some month to month correlation and then seasonal correlation between year to year as well.

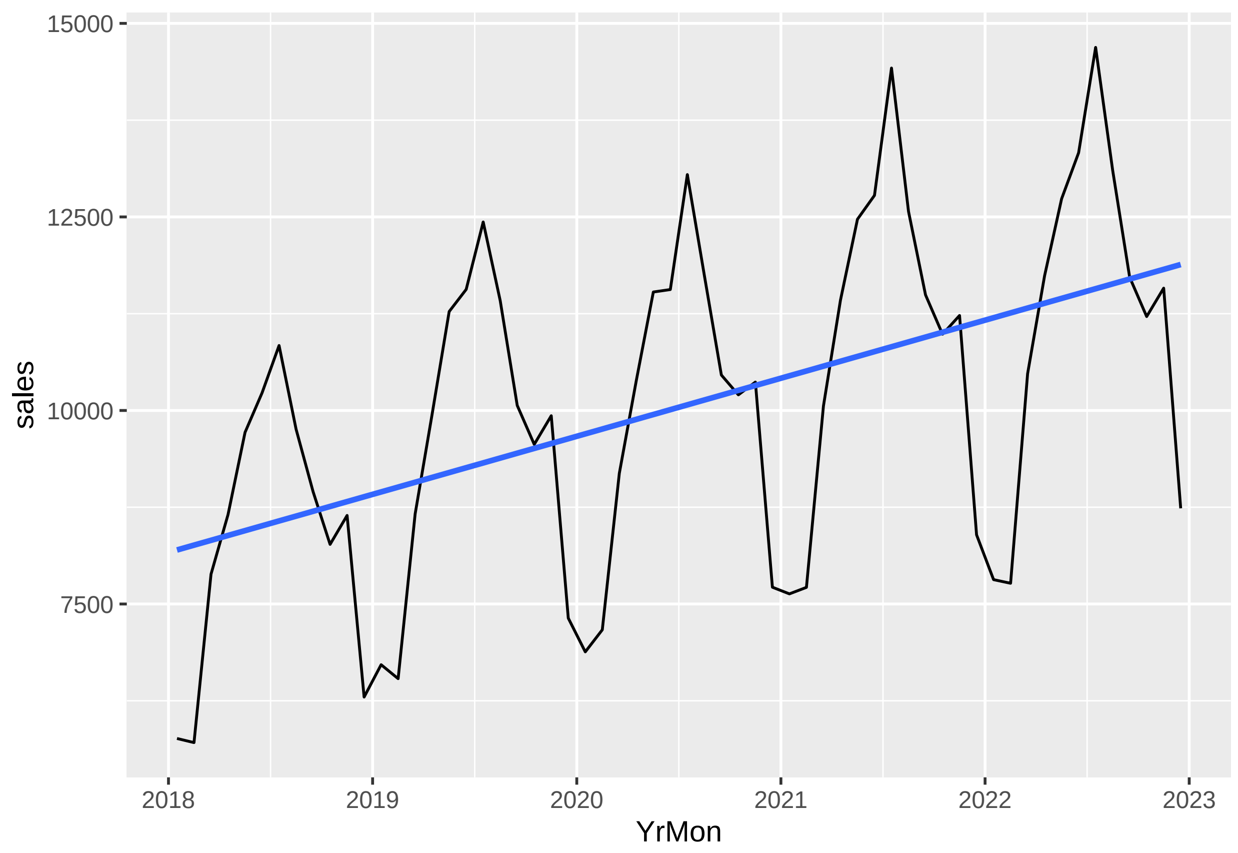
```r
#Running an ACF plot on just the data for 3 seasons
my.ACF <- acf(item$sales, lag.max=36)

plot(my.ACF)
```



2.

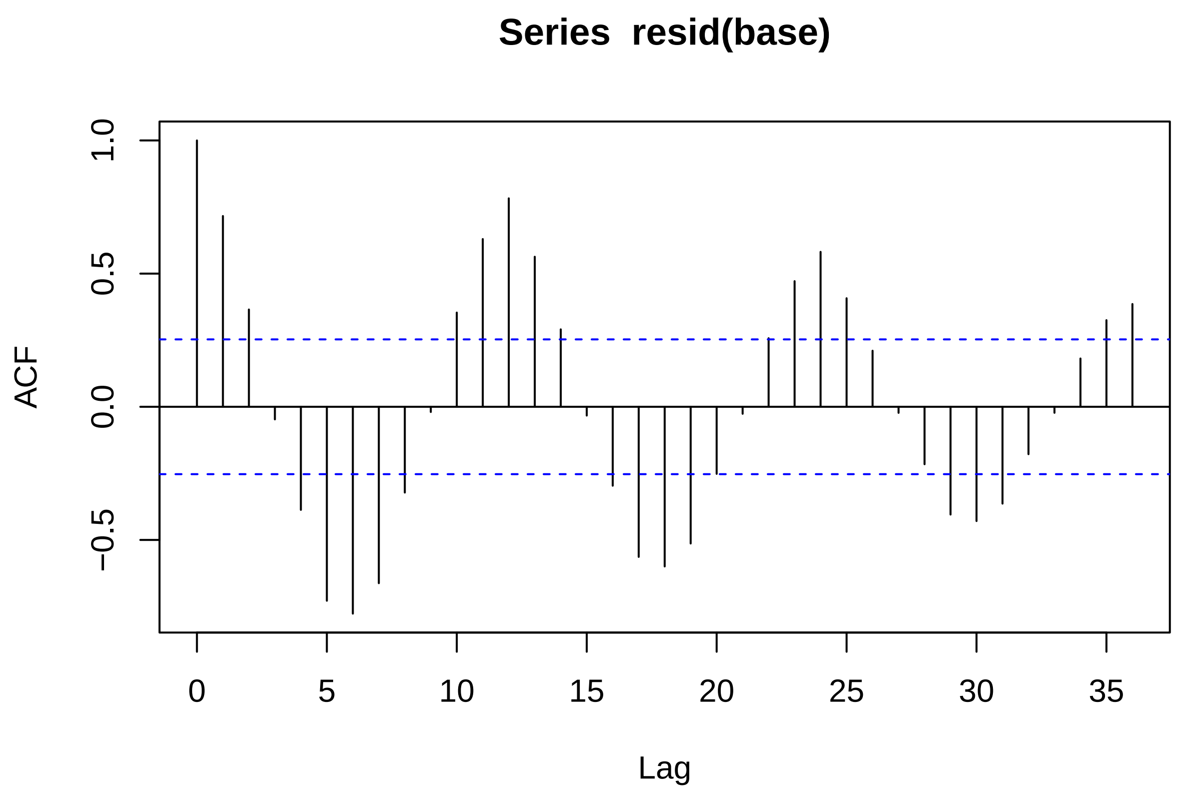
We fit the data to a basic linear model first to see how well it fits the data. From the plot below, the regression line fit is seen to under fit the data, but does show the increase of the sales.

```
#Fitting a basic lm plot
base<- lm(sales~YrMon,data=item)

#Plotting the time series with the regression line on it
ggplot(data=item, mapping=aes(x=YrMon,y=sales)) + geom_line()+geom_smooth(method=lm, formula=y~x, se=FA
```



To see if this model helps overcome the correlation we observed above, we created an ACF plot of the residuals of the model, which is below. It appears there is the same temporal correlation as before within the residuals of this model. If we want to understand the sales and try to predict how they will go in the future, this temporal correlation needs to be accounted for in the model. The sales are affected by those from the month before and the season as well. Understanding this correlation, will help us get correct predictions.

```
#Doing an ACF on the residuals of the normal lm
my.ACF <- acf(resid(base), lag.max=36)

plot(my.ACF)
```



3.

Now we see that the normal linear model is a bad fit for our data, we will try a Seasonal auto-regressive, integrated, moving average model (SARIMA) that allows for temporally correlated data through its different parameters of time point, seasonal, and changes of both, which are known as p,q,d,P,Q,D. We determined the best numbers for these parameters to be (0,1,1)(0,1,0). We found these parameters by using the auto.arima function to perform the model comparison metric of ‘aic’ without any stepwise selection.

```
#creating a time series object
my.ts <- ts(data=item$sales, start=c(2018,1), frequency=12)
```

```
#Finding the correct parameters using auto.arima
auto.arima(my.ts, max.p=2, max.q=2, max.P=1, max.Q=1, max.d=1, max.D=1, ic="aic", stepwise=FALSE)
```

```
## Series: my.ts
## ARIMA(0,1,1)(0,1,0)[12]
##
## Coefficients:
##           ma1
##       -0.3170
## s.e.   0.1252
##
## sigma^2 = 80431:  log likelihood = -331.67
## AIC=667.34   AICc=667.61   BIC=671.04
```

4.

Below is the formula for the SARIMA model we are using:

$y = \mathbf{X}\beta + \boldsymbol{\epsilon}\ \boldsymbol{\epsilon} \sim SARIMA(p, d, q, P, D, Q)$

y is the expected sales for a given month in a year

$\mathbf{X}$ is the design matrix, or matrix of our values for our explanatory variables. It starts with a column of ones (to be multiplied against the intercept coefficient) and then contains a column for the time.

β is the vector containing both of the coefficients, or the effect of time on expected sales. We will multiply this vector against $\mathbf{X}$ to get y.

$\boldsymbol{\epsilon}$ is a vector containing the expected variances of our fitted values. This vector follows a SARIMA model.

p is the autoregressive component of our SARIMA model in terms of months. Essentially, how many months back our SARIMA model will look for information. However, our SARIMA model does not use this term.

d is the differencing order our SARIMA model will use. Our SARIMA model does use this parameter, so our SARIMA model will look at the differences between months rather than the actual values associated with the months themselves.

q is the moving-average component of our SARIMA model in terms of months. Essentially, how many months back our SARIMA model will look for the unique information from that month. Our SARIMA model will only be looking at the last month for its unique information.

P is the autoregressive component of our SARIMA model in terms of seasons. Essentially, how many seasons back our SARIMA model will look for information. Our SARIMA model does not use this variable.

D is the differencing order our SARIMA model will use. Our SARIMA model does use this variable, so our SARIMA model will look at the differences between seasons rather than the actual values associated with the seasons themselves.

Q is the moving-average component of our SARIMA model in terms of seasons. Essentially, how many seasons back our SARIMA model will look for the unique information from that season. Our SARIMA model does not use this variable.

Our model will help us forecast the number of items sold in the future by using past sales information to predict future sales.

5.

```r
#Fitting the parameters found above to a time series model
my.sarima.model <- Arima(my.ts, order=c(0,1,1), seasonal=c(0,1,0))

summary(my.sarima.model)
```

```
## Series: my.ts
## ARIMA(0,1,1)(0,1,0)[12]
##
## Coefficients:
##           ma1
##       -0.3170
## s.e.   0.1252
##
## sigma^2 = 80431:  log likelihood = -331.67
## AIC=667.34   AICc=667.61   BIC=671.04
```

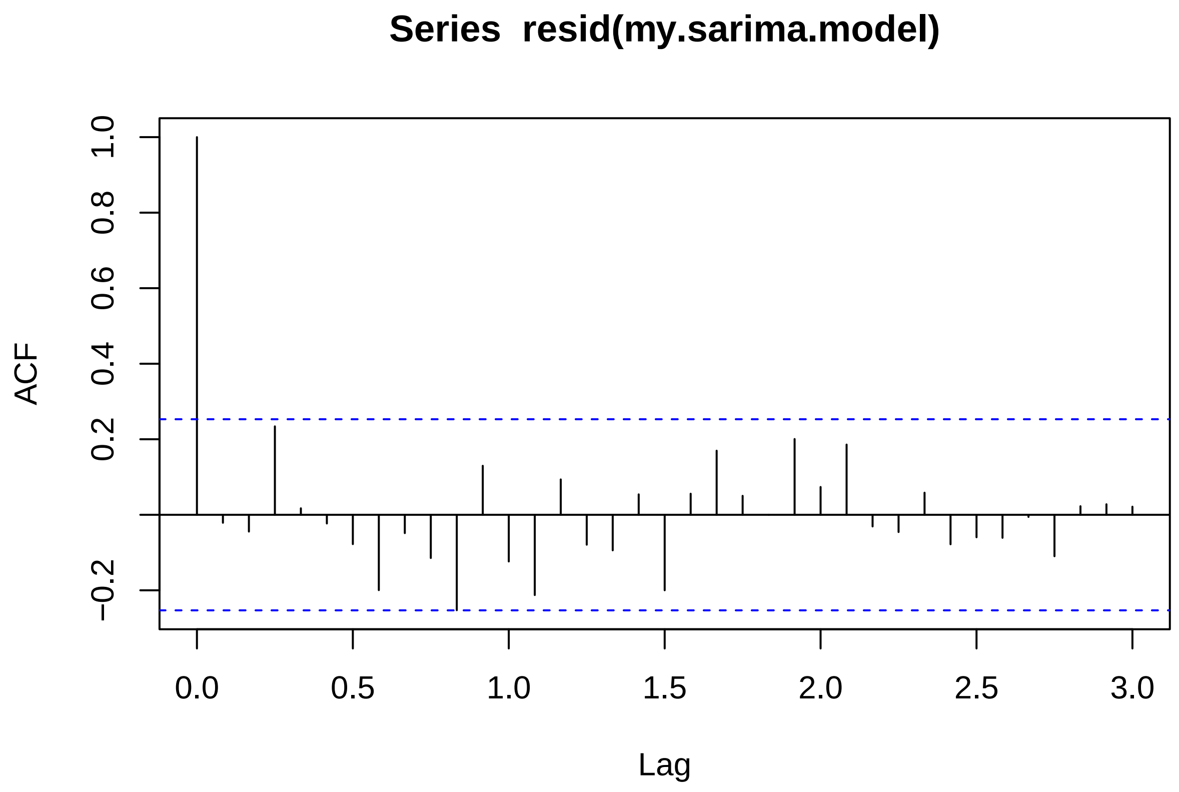
```
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -15.02566 248.3224 168.4477 -0.3114363 1.720868 0.2364801
##                      ACF1
## Training set -0.02114527
```

After creating a SARIMA model, we need to check the LINE assumptions to make sure our model is valid.

For testing linearity, we observed the time series plot of the data, which was displayed earlier. The time series displays a general linear relationship between the two variables.

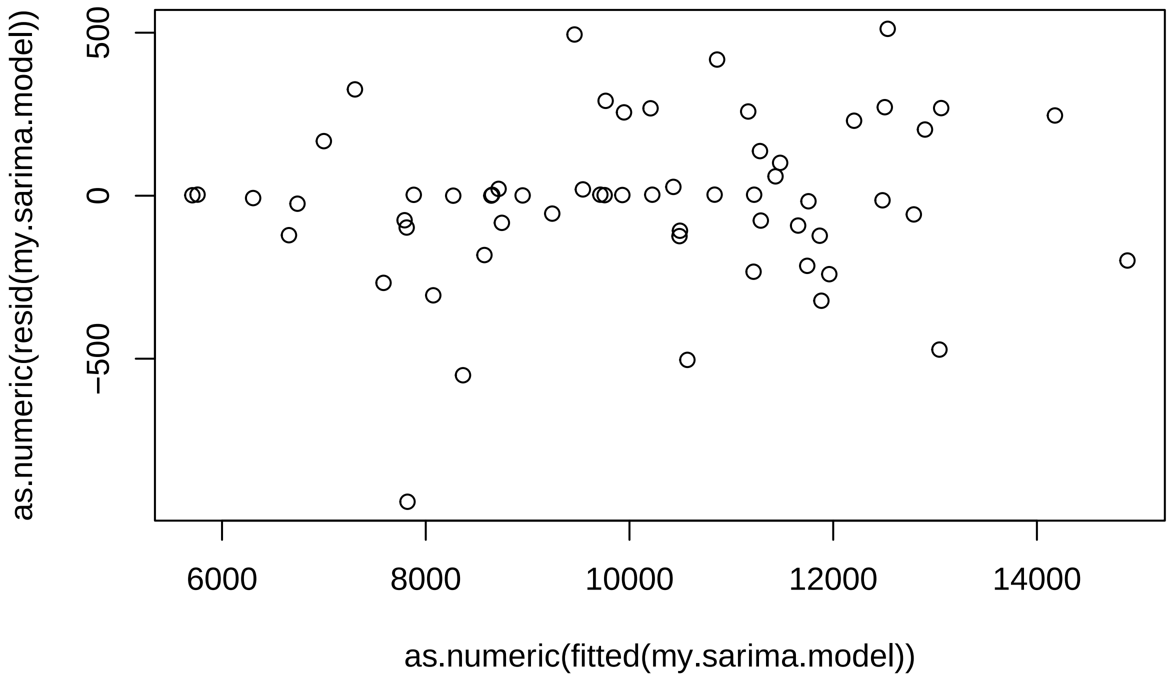
To check independence of the model, we created an ACF plot of the residuals below. We observe that none of the lags are outside of the significance threshold so the model meets the assumption of independence.

```r
#Plotting the ACF of the residuals of the time series plot
my.ACF <- acf(resid(my.sarima.model), lag.max=36)

plot(my.ACF)
```



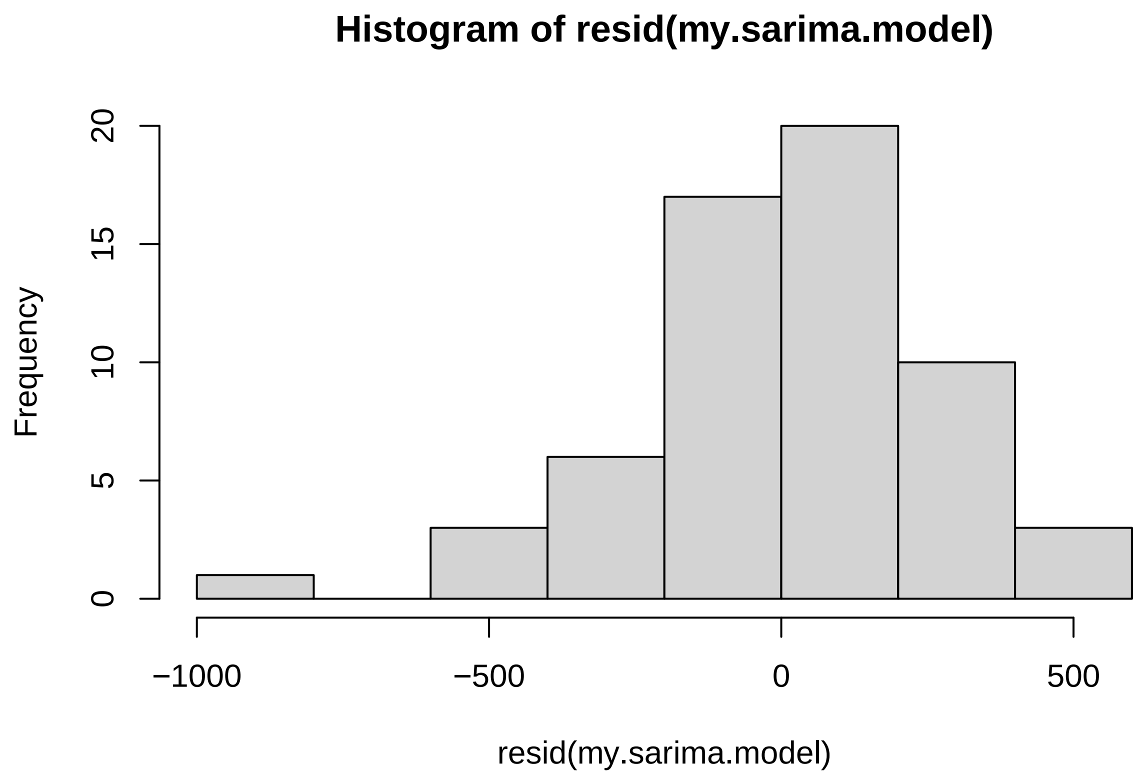
To check if the variance is equal, we plotted the fitted values of the model versus the residuals in the plot below. The spread of the points looks pretty equal, so the model meets this assumption.

```r
#Checking equal variance by plotting fitted vs residuals
plot(as.numeric(fitted(my.sarima.model)),as.numeric(resid(my.sarima.model)))
```



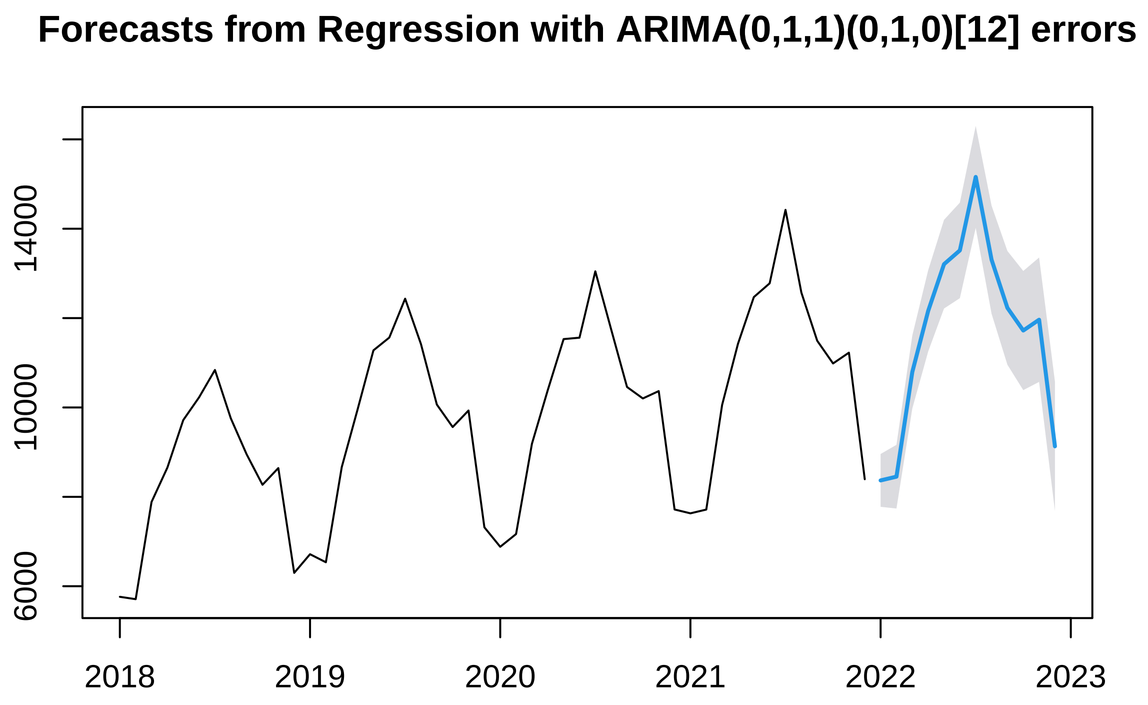
To check normality of the model, we created the histogram of the residual below. The histogram displays a pretty normal distribution of the data meeting the normality assumption for the model.

```
#Determining normality from the residuals of time series
hist(resid(my.sarima.model))
```



6.

```
train_df <- item[1:48,]
test_df <- tail(item,12)

train.ts <- ts(data = train_df$sales, start = c(2018,1), frequency = 12)

X<- model.matrix(base)
test.X <- tail(X,12)[,2]
train.X <- X[1:48,2]

# Fitting the model

model <- Arima(train.ts, order=c(0,1,1), seasonal=c(0,1,0), xreg = train.X)

preds <- forecast(model, h=12, xreg=test.X, level = .95)

  ## Calculate Coverage
cvg <- ((test_df[['sales']] > preds[['lower']]) & (test_df[['sales']] < preds[['upper']])) %>% mean()

rpmse <- (test_df[['sales']]-preds[['mean']])^2 %>% mean() %>% sqrt()

plot(preds)
```



```r
knitr::kable(data.frame(RPMSE = rpmse, 'Standard Devation' = sd(item$sales)))
```

RPMSE	Standard.Devation
446.2982	2179.174

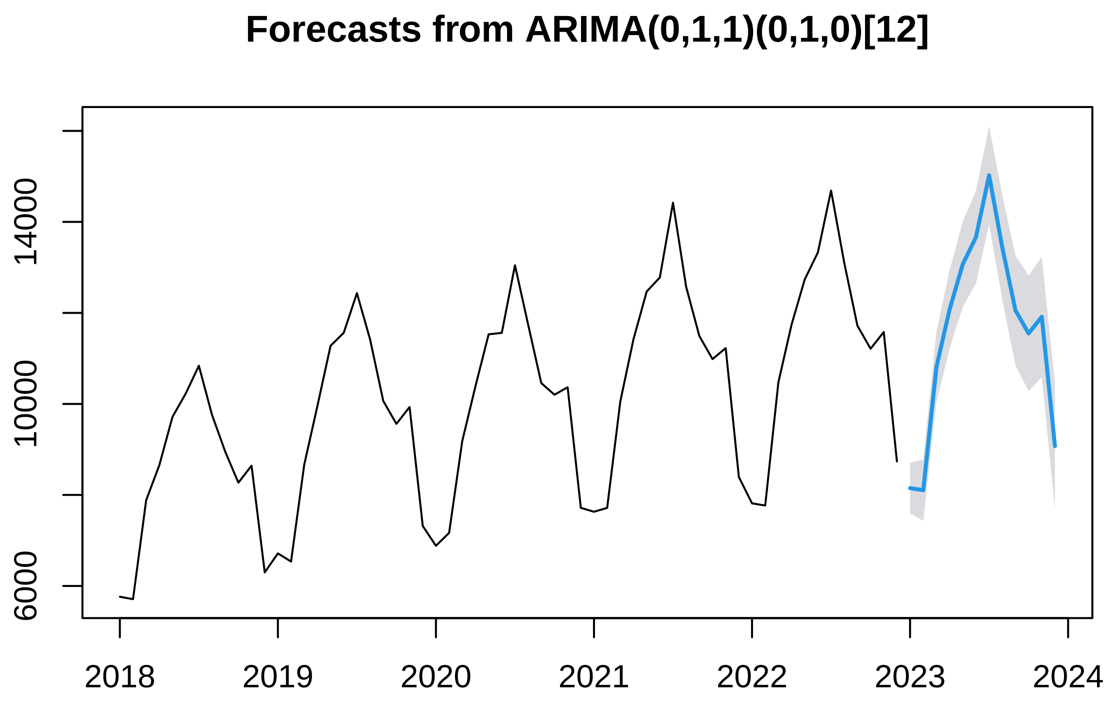
Our cross-validation of our model returned an RPMSE of 446.3. Compared to the original standard deviation of sales (2179.17), our model is very effective at predicting sales.

7.

```r
preds <- forecast(my.sarima.model, h=12, xreg=tail(item$YrMon, 12)+1, level = .95)
```

```
## Warning in forecast.forecast_ARIMA(my.sarima.model, h = 12, xreg =
## tail(item$YrMon, : xreg not required by this model, ignoring the provided
## regressors
```

```r
plot(preds)
```



It looks like sales in the year of 2023 will follow a very similar pattern to sales in 2022 and other years previous, but that sales will grow overall. Executives can use our model and the associated confident intervals to judge how much the company can expect to sell, assuming conditions stay similar to how they have been in the past. The company should expect to produce at least 16000 units to be ready for the highest volume of possible orders.