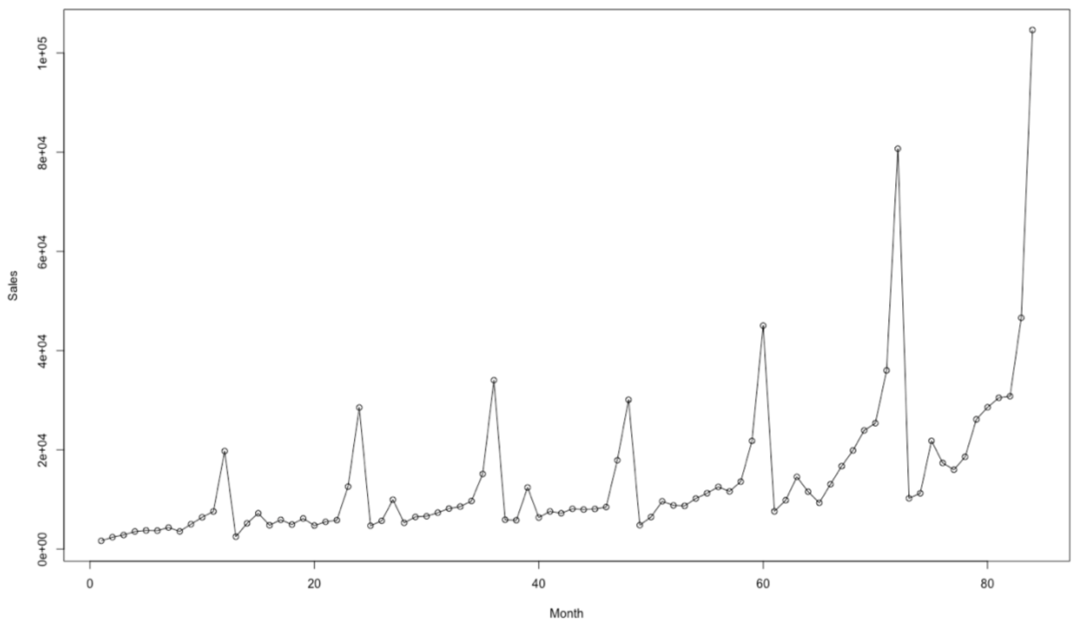
BSAN 450 Final

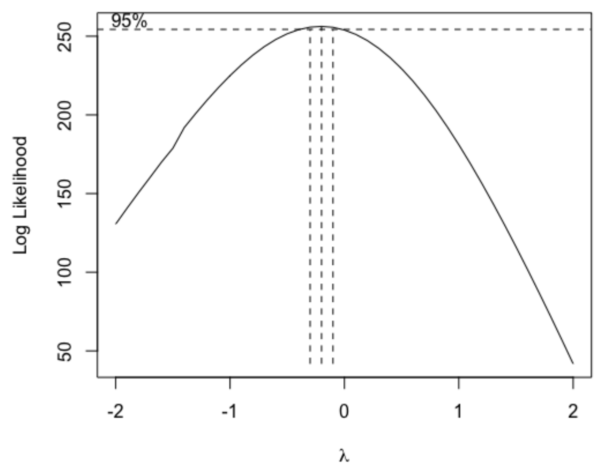
Grant Healy

1. Souvenir Shop Time Series

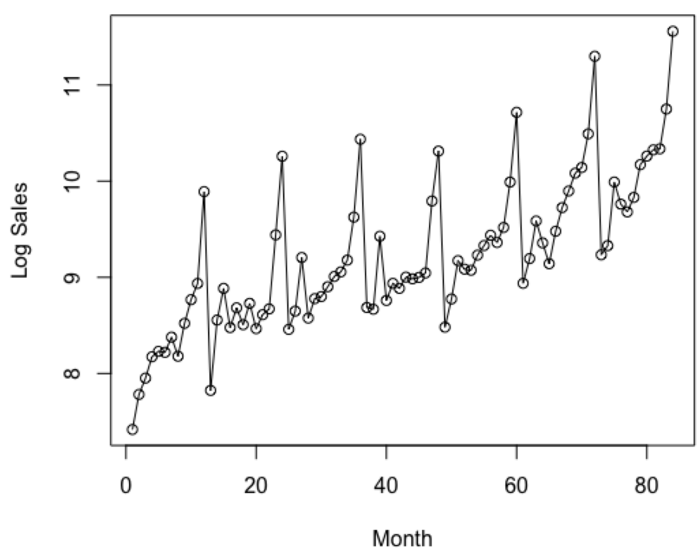



To begin my analysis, I plotted the initial sales data over time. From this plot I can observe that this data has a changing average, a seasonal pattern, and a changing variability. My initial reaction is that this data will need a non-linear transformation, a 1st differencing and 12th differencing to become stationary.

```r
BoxCox.ar(y=Sales, method = "yule-walker")

souvenir$logSales <- log(Sales)

plot(souvenir$logSales, ylab='Log Sales', xlab='Month', type='o')
```

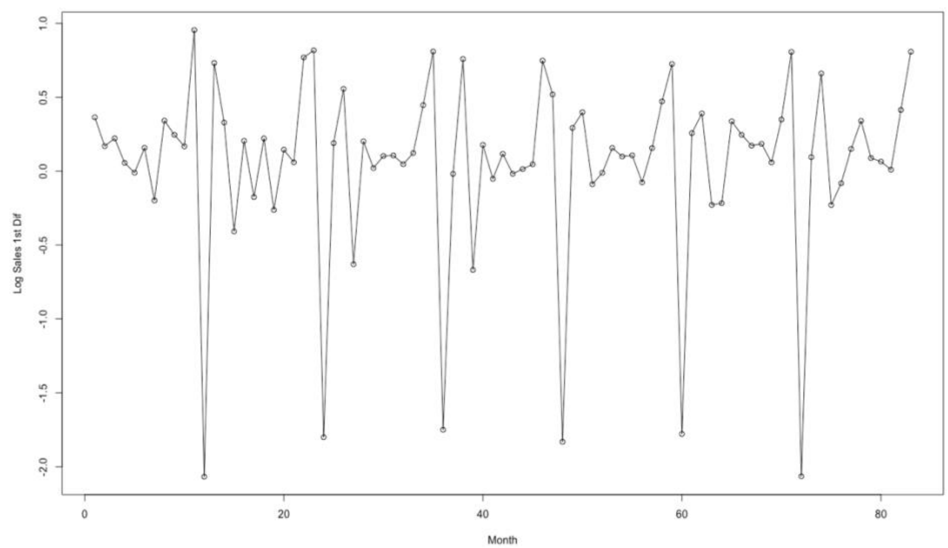



Determining which non-linear transformation to use, I made a Box-Cox plot to determine the value of lambda. Seeing as the value of lambda was near 0, the log of sales would be the appropriate non-linear transformation.

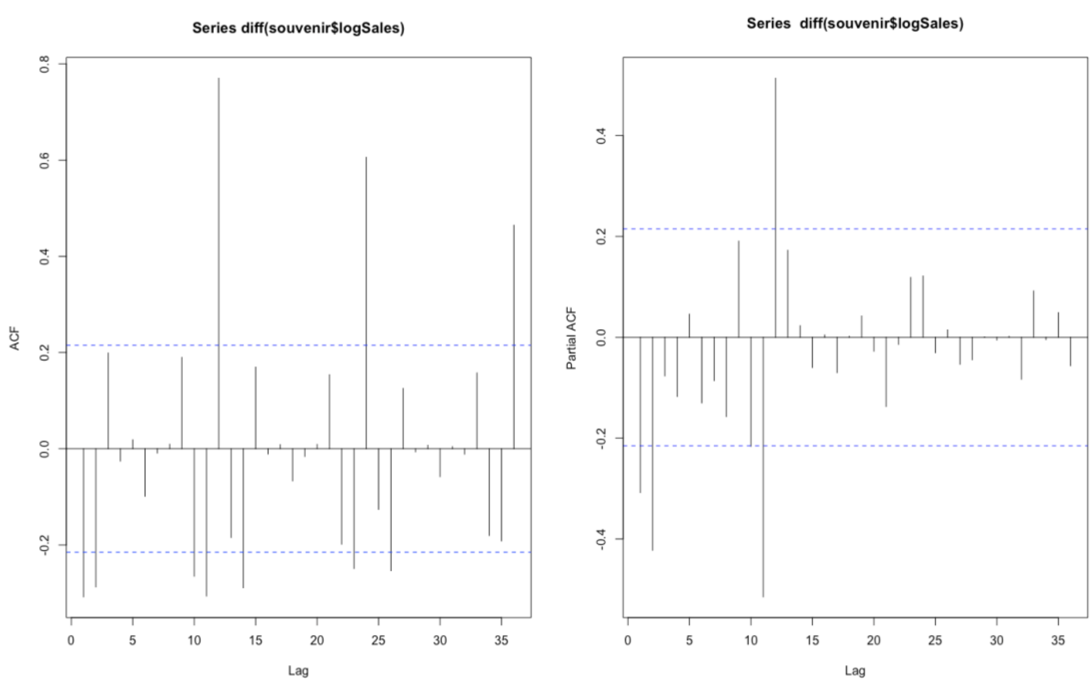



Plotting log(sales) over time, it can be observed that variability over time is more stable now.

```r
par(mfrow=c(1,1))
plot(diff(souvenir$logSales), ylab='Log Sales 1st Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(souvenir$logSales), lag.max=36)
pacf(diff(souvenir$logSales), lag.max=36)
```

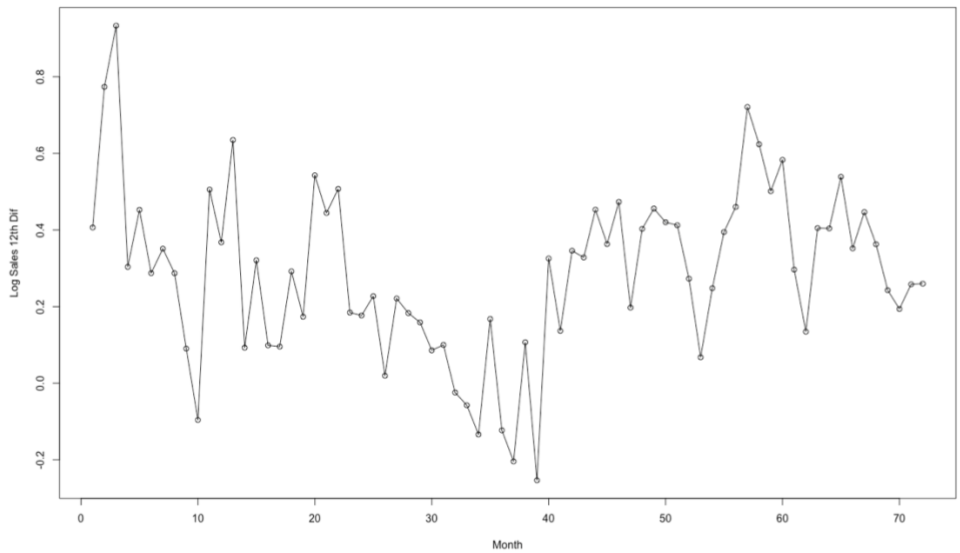



Looking at the plot of the 1st difference log(sales) data, it appears the average of the data over time is stable.

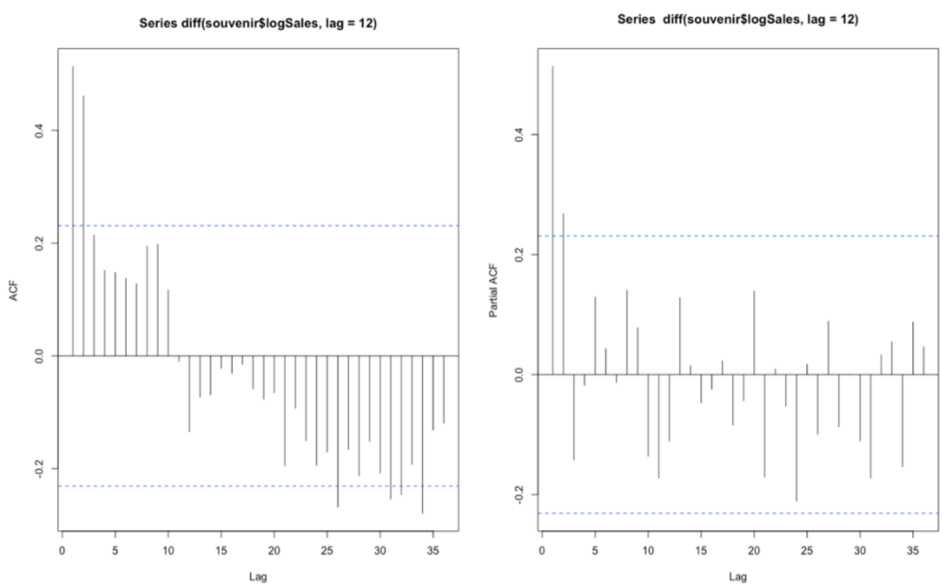



On the ACF plot, significant autocorrelations can be seen at the 12th, 24th and 36th observations. On the PACF plot, a significant autocorrelation can be seen at the 12th observation. This indicates that this data is seasonal and needs to be 12th differenced.

```
par(mfrow=c(1,1))
plot(diff(souvenir$logSales,lag=12), ylab='Log Sales 12th Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(souvenir$logSales,lag=12), lag.max=36)
pacf(diff(souvenir$logSales,lag=12), lag.max=36)
```

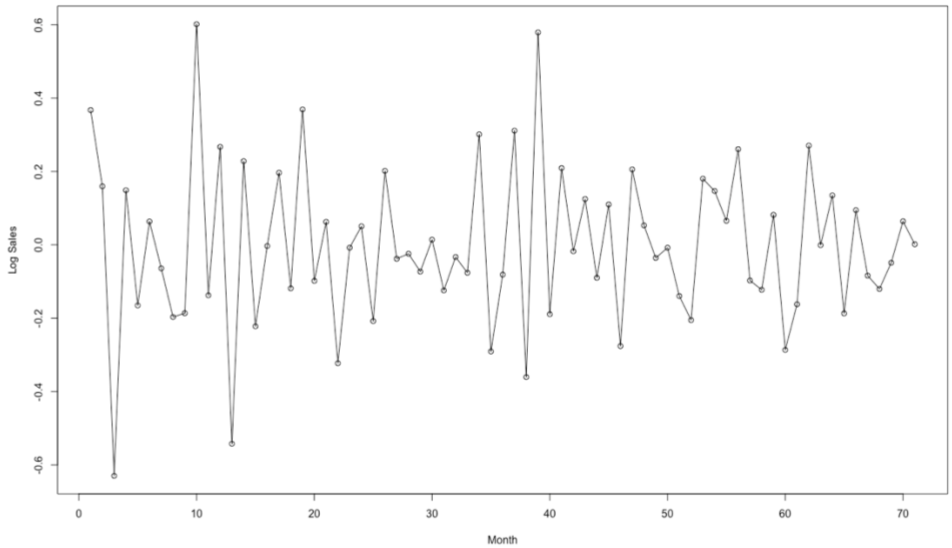



The plot of the 12th differenced log(sales) time series does not appear to be stationary as the average appears to be moving throughout the data.

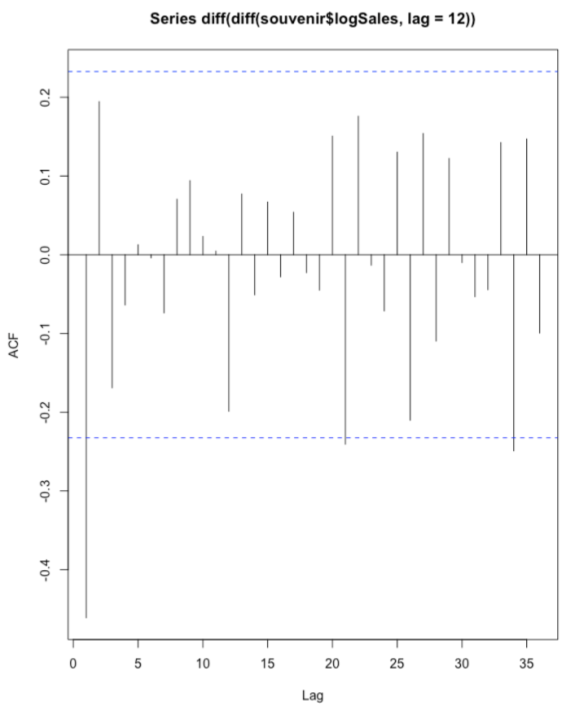


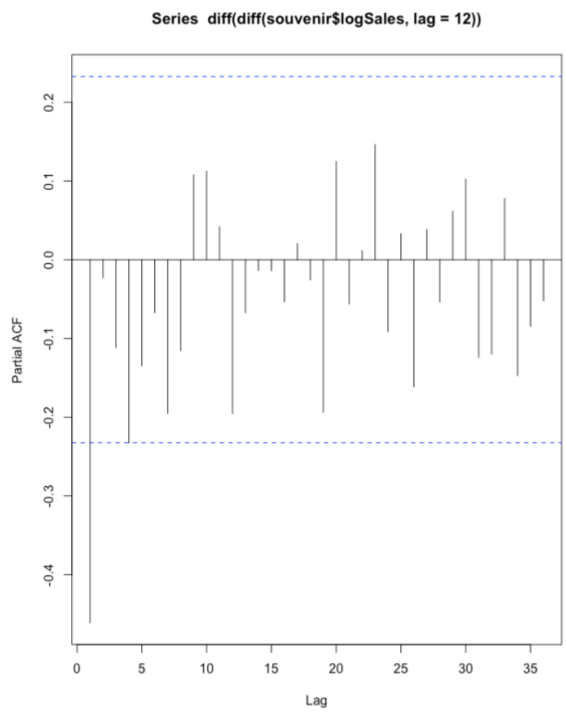

The ACF plot of the 12th difference (somewhat) shows autocorrelations die out at a linear rate and seem to alternate to showing significance again around the 25th – 30th observations. This is another indication that a first difference may be necessary.

```
par(mfrow=c(1,1))
plot(diff(diff(souvenir$logSales),lag=12), ylab='Log Sales', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(diff(souvenir$logSales,lag=12)), lag.max=36)
pacf(diff(diff(souvenir$logSales,lag=12)), lag.max=36)
```




The 1st and 12th differenced log(sales) plot shows that it is now a stable time series. The average appears to be constant and variability (although spiking at some points) appears to remain stable throughout the plot.




The ACF of the 1st and 12th differenced data shows that an ARIMA(0,1,1)*SARIMA(0,1,0) model could be appropriate as there is a significant autocorrelation at the first observation before immediately dying out with no significant autocorrelation at the twelfth observation.

The PACF plot of the data, shows that an ARIMA(1,1,0)*SARIMA(0,1,0) model could be appropriate as there is a significant autocorrelation at the 1st observation before immediately dying out and no significant autocorrelations after.

Based on the ACF plot, I did consider an ARIMA(0,1,1)*SARIMA(0,1,2) model, but determined (not only would that be overly complex) but it is unnecessary as there is no significant autocorrelation at the 12th observation.

```
souvenir.fit1 = arima(souvenir$logSales, order=c(1,1,0), seasonal=list(order=c(0,1,0), period=12))
souvenir.fit1
coeftest(souvenir.fit1)
Box.test(residuals(souvenir.fit1),lag=10, type="Ljung", fitdf=1)
```

```
Call:
arima(x = souvenir$logSales, order = c(1, 1, 0), seasonal = list(order = c(0,
    1, 0), period = 12))

Coefficients:
          ar1
      -0.4726
s.e.   0.1057

sigma^2 estimated as 0.03766:  log likelihood = 15.54,  aic = -29.08
```

```
z test of coefficients:

     Estimate Std. Error z value  Pr(>|z|)
ar1 -0.47264    0.10571  -4.471 7.786e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
        Box-Ljung test

data:  residuals(souvenir.fit1)
X-squared = 13.428, df = 9, p-value = 0.1442
```

The ARIMA(1,1,0)*SARIMA(0,1,0) model fit has a sigma^2 is .0376, log likelihood of 15.54, and the aic is -29.08. Through the coefficients significance test it can be determined that the AR(1) coefficient is in fact significant. This model passes the Box-Ljung test.

```r
souvenir.fit2 = arima(souvenir$logSales, order=c(0,1,1), seasonal=list(order=c(0,1,0), period=12))
souvenir.fit2
coeftest(souvenir.fit2)
Box.test(residuals(souvenir.fit2),lag=10, type="Ljung", fitdf=1)
```

```
Call:
arima(x = souvenir$logSales, order = c(0, 1, 1), seasonal = list(order = c(0,
    1, 0), period = 12))

Coefficients:
          ma1
      -0.5653
s.e.   0.1227

sigma^2 estimated as 0.03722:  log likelihood = 15.89,  aic = -29.79
```

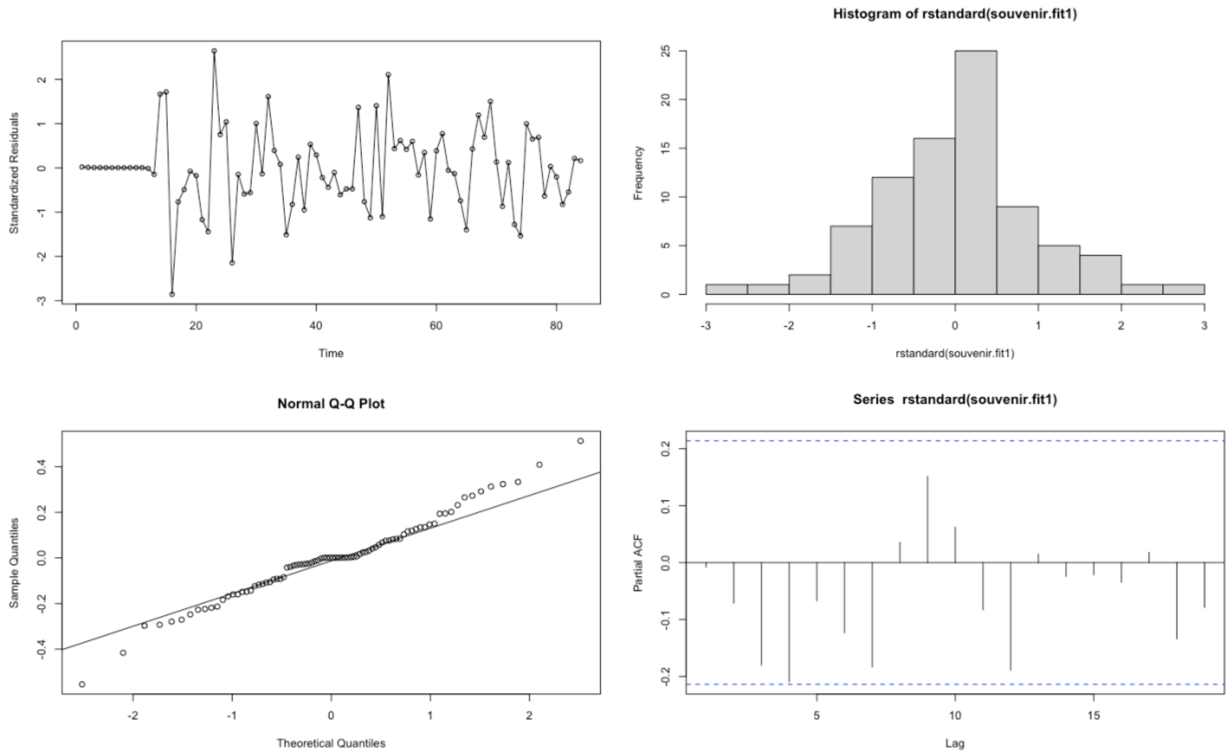
```
z test of coefficients:

    Estimate Std. Error z value  Pr(>|z|)
ma1 -0.56530    0.12275 -4.6055 4.115e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
                Box-Ljung test

        data:  residuals(souvenir.fit2)
        X-squared = 12.784, df = 9, p-value = 0.1726
```

The ARIMA(0,1,1)*SARIMA(0,1,0) model has a sigma^2 of .0372, log likelihood of 15.89, and the aic is -29.79. Through the coefficients significance test it can be determined that the MA(1) coefficient is significant. This model passes the Box-Ljung test.

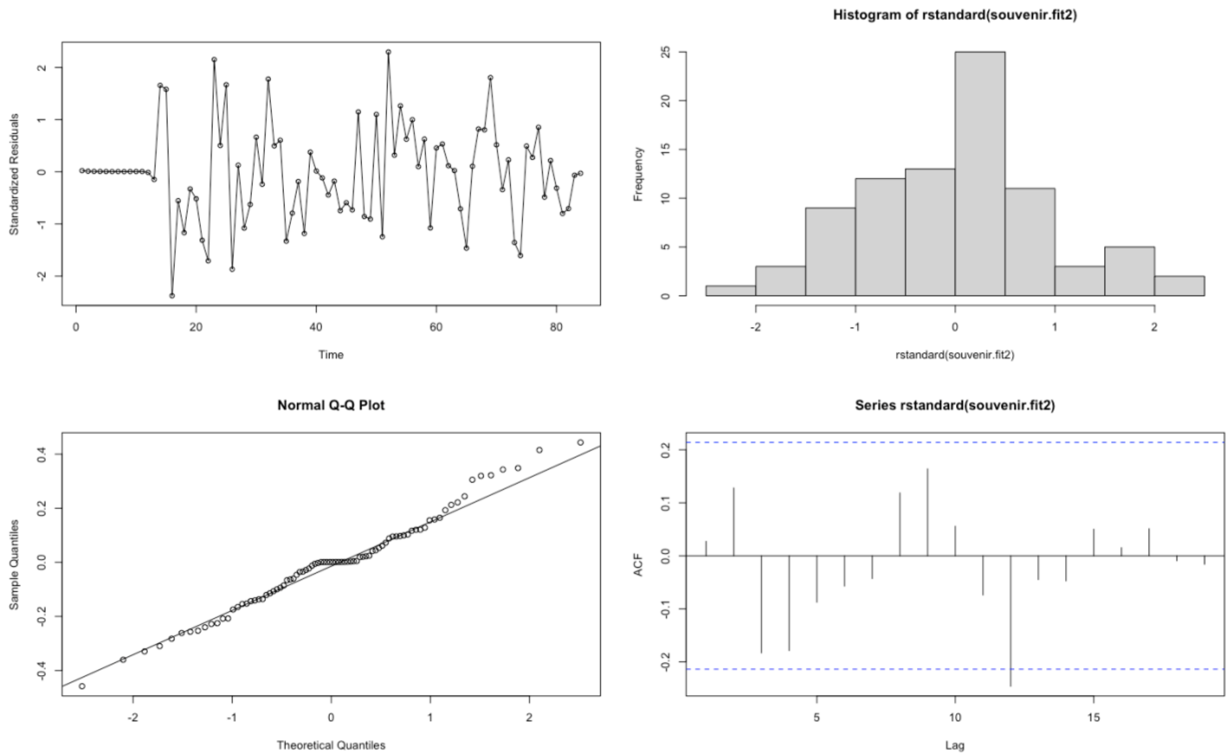
Based on these results alone, it would appear that the MA(1) model has a slight edge over the AR(1) model but diagnostic plots still need to be checked.

```
par(mfrow = c(2,2))
plot(rstandard(souvenir.fit1), ylab='Standardized Residuals', type='o')
hist(rstandard(souvenir.fit1))
qqnorm(residuals(souvenir.fit1))
qqline(residuals(souvenir.fit1))
pacf(rstandard(souvenir.fit1))
```




The residuals diagnostics of the AR(1) model, show no cause for concern. It should be noted that there are some observations that could be considered close to being outliers in the histogram, in additional to some undesirable variation on the QQ-Plot.

```r
par(mfrow = c(2,2))
plot(rstandard(souvenir.fit2), ylab='Standardized Residuals', type='o')
hist(rstandard(souvenir.fit2))
qqnorm(residuals(souvenir.fit2))
qqline(residuals(souvenir.fit2))
acf(rstandard(souvenir.fit2))
```



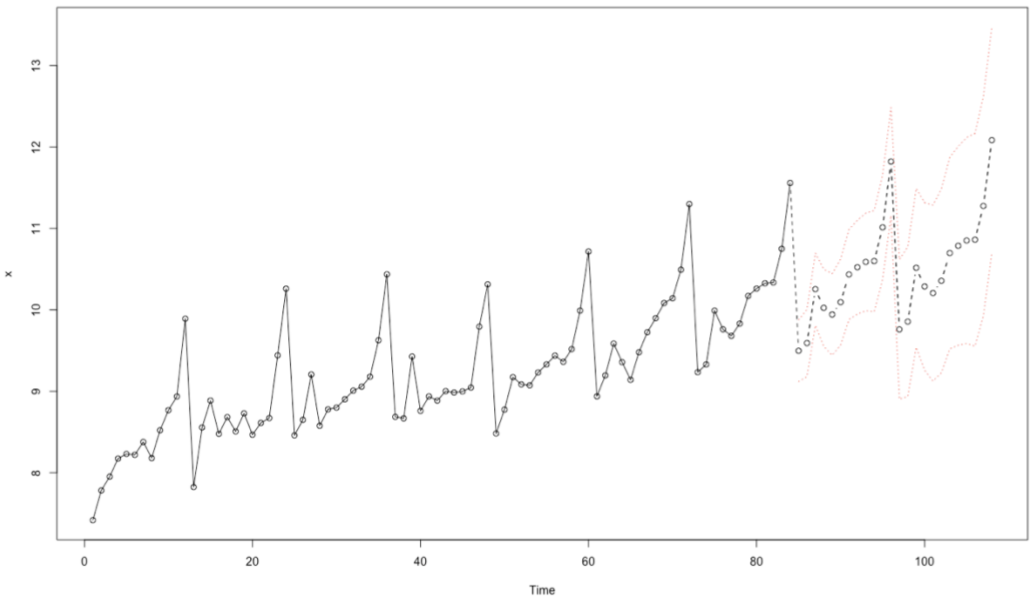

The residual diagnostics of the MA(1) model also show no cause for concern. This model seems to have a tighter distribution of the histogram as well as some reduced variance on the QQ-Plot. The only thing worth note is a slightly significant PACF autocorrelation at the 12th observation. Both models' diagnostics indicate no problems.

Based on the observation that the MA(1) model has a smaller singma^2, larger log likelihood, and lower AIC, I believe it is the appropriate model for this dataset.

```
> detectAO(souvenir.fit2)
[1] "No AO detected"
> detectIO(souvenir.fit2)
[1] "No IO detected"
```

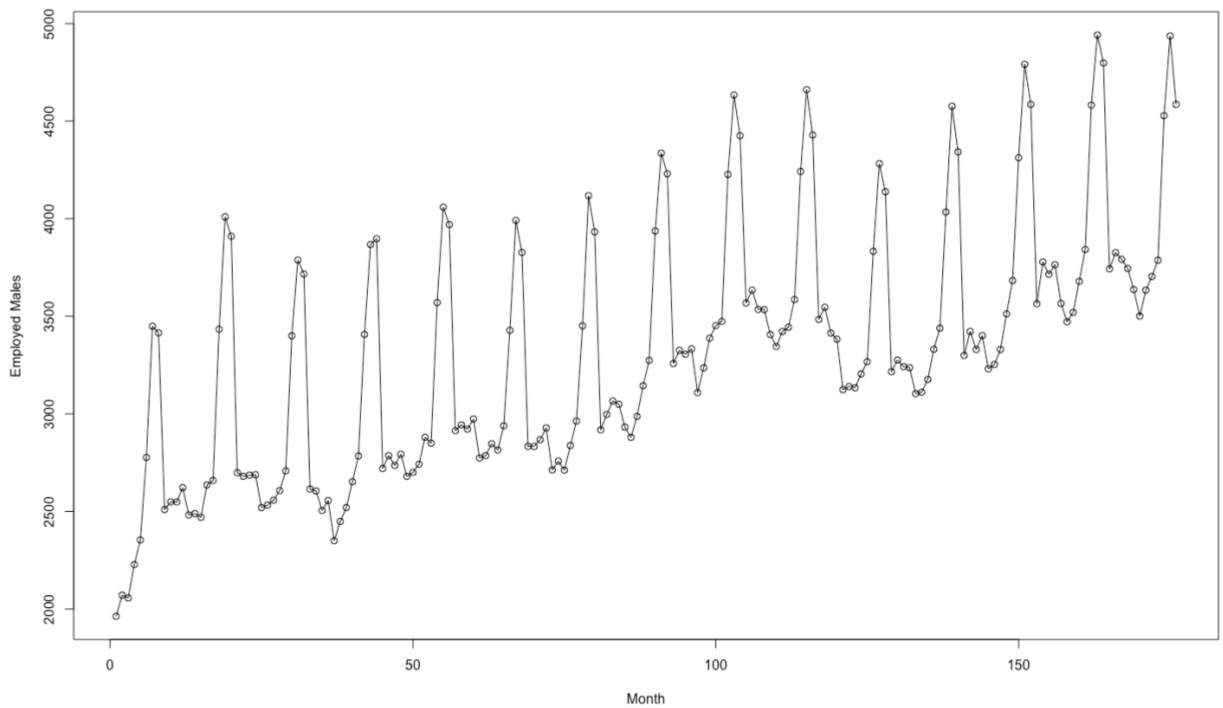
No outliers could be detected in the selected MA(1) model

```
predict(souvenir.fit2, n.ahead=24)

par(mfrow=c(1,1))
plot(souvenir.fit2, n.ahead=24, type='b', col='red')
```



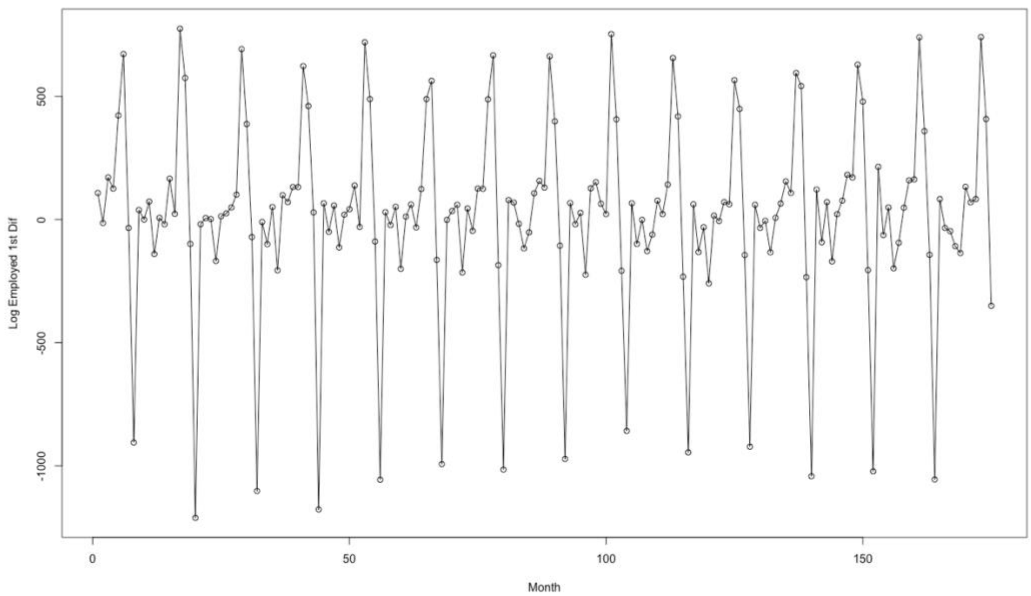

The MA(1) model predictions appears to be in line with the previous observations seasonal pattern and increasing moving average.

2. Employed Males Time Series

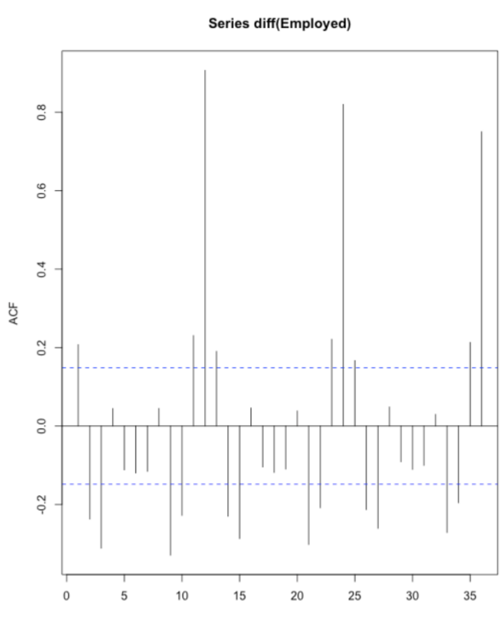


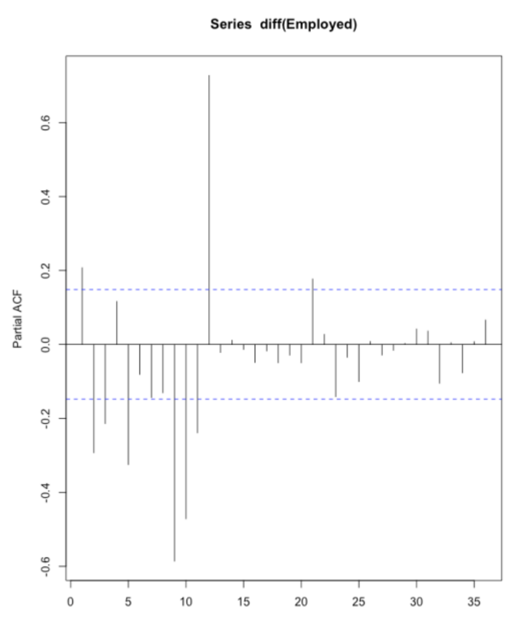

To start my analysis, I plotted the number of employed males over time. Looking at this plot I believe that variability is stable, there is a changing average over time, and a seasonal pattern. There appears to be no need for a non-linear transformation, but this series will likely need a $1^{st}$ and $12^{th}$ differencing.

```
par(mfrow=c(1,1))
plot(diff(Employed), ylab='Log Employed 1st Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(Employed), lag.max=36)
pacf(diff(Employed), lag.max=36)
```



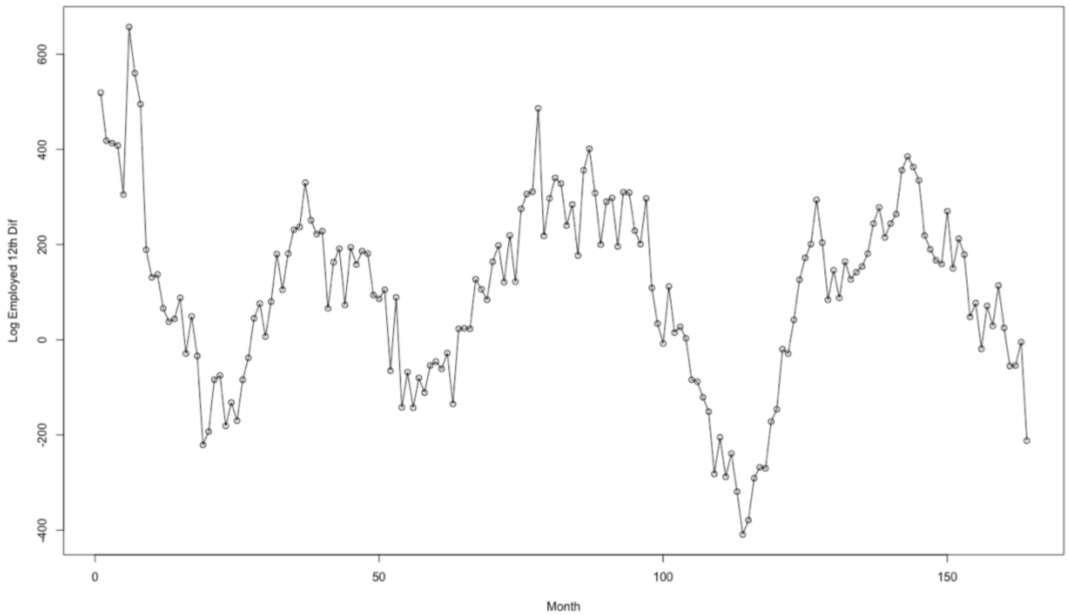

Looking at the plot of the 1st differenced time series, it appears that the average of the time series has been stabilized over time.



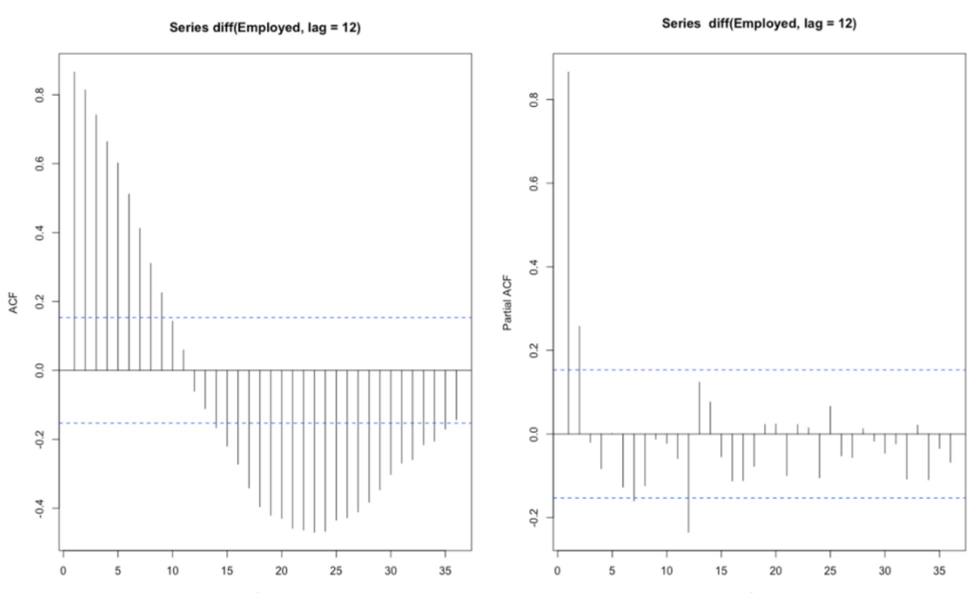




The ACF and PACF plots of the 1st differenced time series indicate that there is a need for a 12th differencing. This is seen through its significant autocorrelations at the 12th observations. This pattern continues throughout the ACF.

```
par(mfrow=c(1,1))
plot(diff(Employed,lag=12), ylab='Log Employed 12th Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(Employed,lag=12), lag.max=36)
pacf(diff(Employed,lag=12), lag.max=36)
```

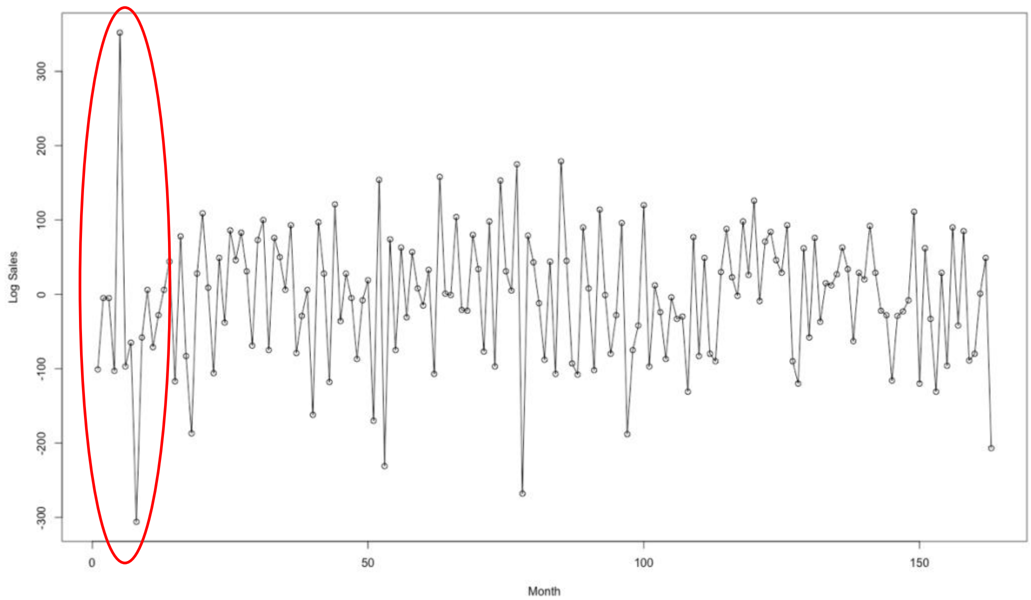



Looking at the plot of the 12th differenced time series, the seasonal pattern seems to be removed but the average appears to change over time.

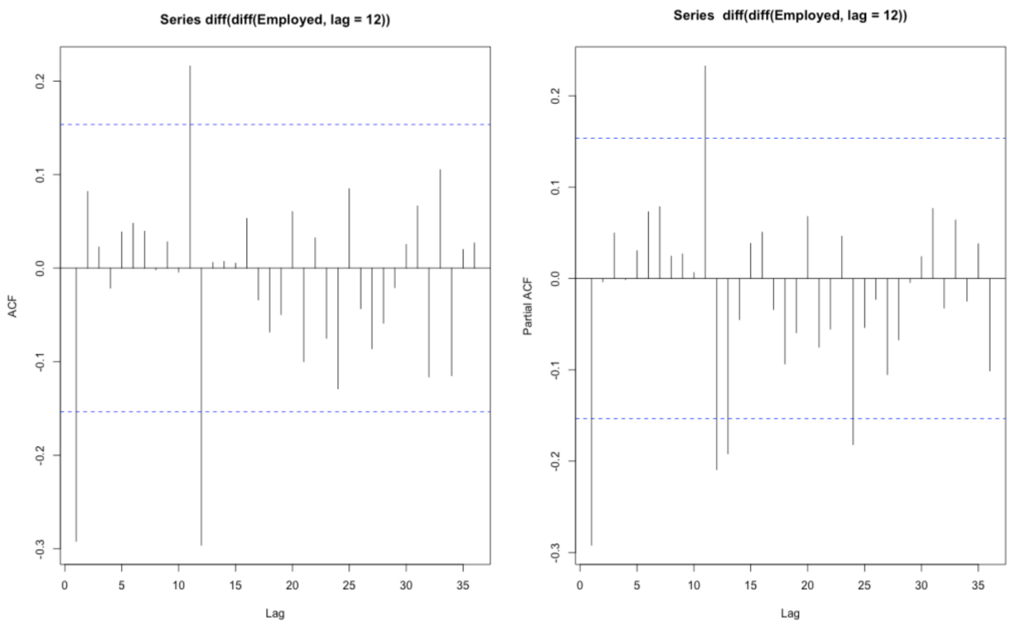



The ACF of the 12th differenced time series also seems to indicate that a 1st difference may be necessary through its linear decay of autocorrelations.

```
par(mfrow=c(1,1))
plot(diff(diff(Employed),lag=12), ylab='Log Sales', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(diff(Employed,lag=12)), lag.max=36)
pacf(diff(diff(Employed,lag=12)), lag.max=36)
```




The 1st and 12th differenced plot show the time series is now stationary. The circled point above should be considered when determining outliers.




The ACF plot of the data shows that an ARIMA(0,1,1)*SARIMA(0,1,1) model could be appropriate for this data. This is seen in its significant autocorrelations at the 1st and 12th observations.

The PACF plot of the data shows that an ARIMA(1,1,0)*SARIMA(1,1,0) model could be appropriate for this data. This is seen in its significant autocorrelations as the 1st and 12th observations. Consideration should also be given to an ARIMA(1,1,0)*SARIMA(2,1,0) as there is an additional significant autocorrelation at the 24th observation.

```
employed.fit1 = arima(Employed, order=c(0,1,1), seasonal=list(order=c(0,1,1), period=12))
employed.fit1
coeftest(employed.fit1)
Box.test(residuals(employed.fit1),lag=10, type="Ljung", fitdf=2)
```

```
Call:
arima(x = Employed, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1),
    period = 12))

Coefficients:
          ma1     sma1
      -0.2643  -0.7205
s.e.   0.0713   0.0658

sigma^2 estimated as 5554:  log likelihood = -938.43,  aic = 1880.86
```

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ma1  -0.264271   0.071259  -3.7086 0.0002084 ***
sma1 -0.720481   0.065848 -10.9416 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
                  Box-Ljung test

            data:  residuals(employed.fit1)
            X-squared = 4.821, df = 8, p-value = 0.7765
```

The ARIMA(0,1,1)*SARIMA(0,1,1) model has a sigma^2 of 5554, a log likelihood of -938.43, and an aic of 1880.86. Both coefficients of this model are significant, and this model passes the Box-Ljung test.

```
employed.fit2 = arima(Employed, order=c(1,1,0), seasonal=list(order=c(1,1,0), period=12))
employed.fit2
coeftest(employed.fit2)
Box.test(residuals(employed.fit2),lag=10, type="Ljung", fitdf=2)
```

```
Call:
arima(x = Employed, order = c(1, 1, 0), seasonal = list(order = c(1, 1, 0),
    period = 12))

Coefficients:
          ar1      sar1
      -0.2869   -0.3609
s.e.   0.0763    0.0824

sigma^2 estimated as 6728:  log likelihood = -950.52,  aic = 1905.03
```

```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1  -0.286894   0.076298 -3.7602 0.0001698 ***
sar1 -0.360945   0.082381 -4.3814 1.179e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
                    Box-Ljung test

data:  residuals(employed.fit2)
X-squared = 5.983, df = 8, p-value = 0.6491
```

The ARIMA(1,1,0)*SARIMA(1,1,0) model has a sigma^2 of 6728, a log likelihood of -950.52, and an aic of 1905.03. Both coefficients of this model are significant, and this model passes the Box-Ljung test.

```r
employed.fit3 = arima(Employed, order=c(1,1,0), seasonal=list(order=c(2,1,0), period=12))
employed.fit3
coeftest(employed.fit3)
Box.test(residuals(employed.fit3),lag=10, type="Ljung", fitdf=3)
```

```
Call:
arima(x = Employed, order = c(1, 1, 0), seasonal = list(order = c(2, 1, 0),
    period = 12))

Coefficients:
          ar1     sar1     sar2
      -0.3008  -0.4769  -0.3532
s.e.   0.0759   0.0845   0.0836

sigma^2 estimated as 5979:  log likelihood = -942.46,  aic = 1890.91
```

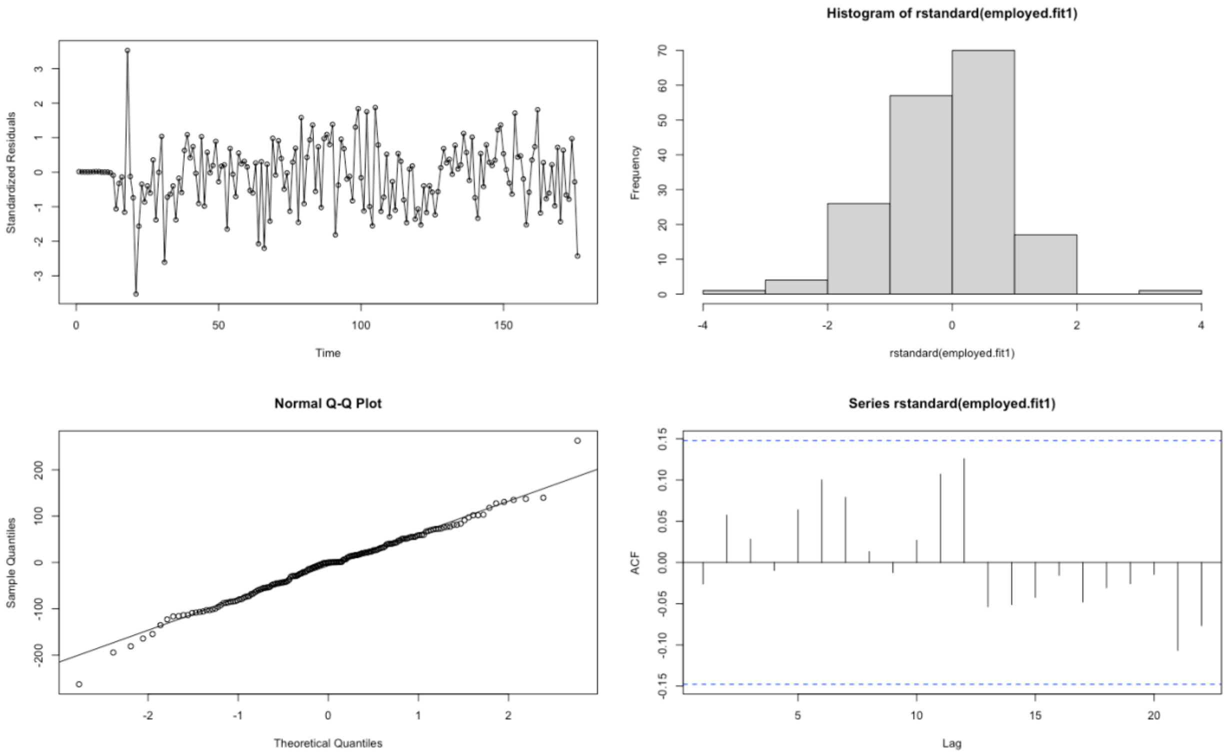
```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1  -0.300760   0.075929 -3.9611 7.462e-05 ***
sar1 -0.476914   0.084456 -5.6469 1.634e-08 ***
sar2 -0.353217   0.083577 -4.2263 2.376e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
        Box-Ljung test

data:  residuals(employed.fit3)
X-squared = 3.6088, df = 7, p-value = 0.8236
```

The ARIMA(1,1,0)*SARIMA(2,1,0) model has a sigma^2 of 5979, a log likelihood of -942.46, and an aic of 1890.91. All three coefficients of this model are significant, and this model passes the Box-Ljung test.

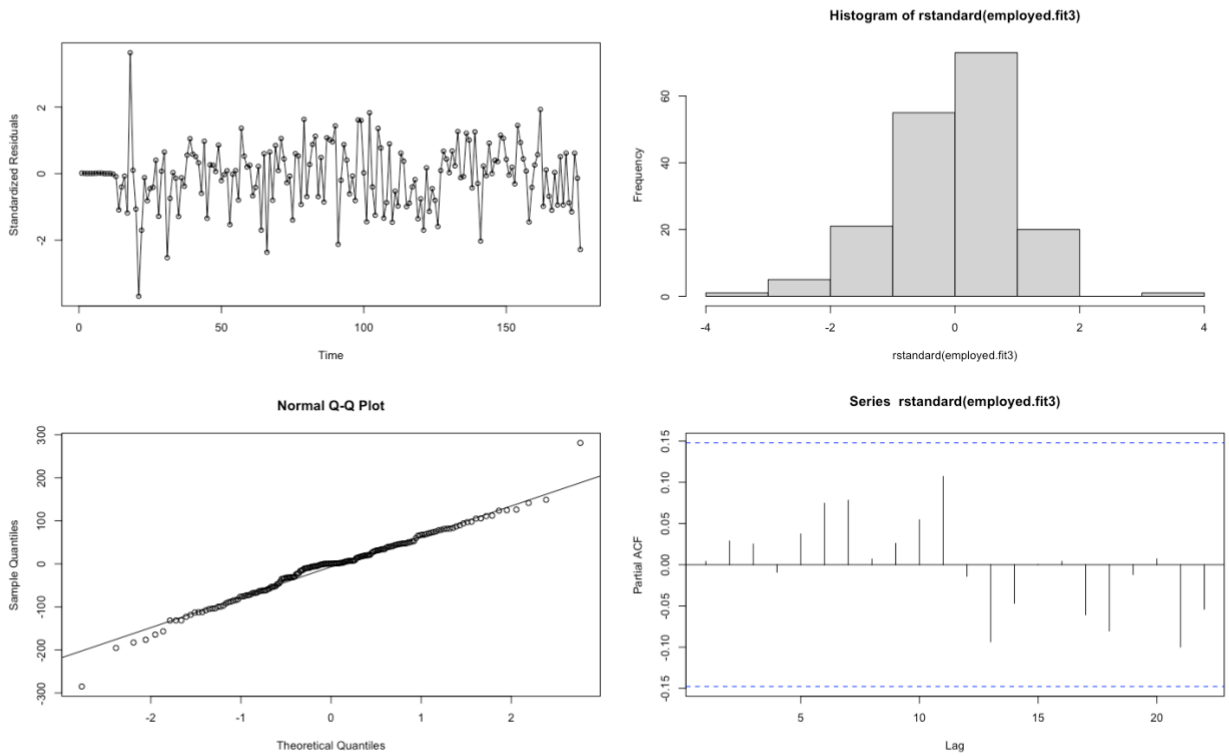
Based on the results of these models, I have elected to proceed with the MA(1)*SMA(1) model and the AR(1)*SAR(2) model. I selected these models because of their performance and matches with the ACF / PACF plots of the time series.

```r
par(mfrow = c(2,2))
plot(rstandard(employed.fit1), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fit1))
qqnorm(residuals(employed.fit1))
qqline(residuals(employed.fit1))
acf(rstandard(employed.fit1))
```




The residuals diagnostics of the MA(1)*SMA(1) model shows no significant cause for concern. It should be noted that the histogram seems to indicate outliers may be present for this model.

```r
par(mfrow = c(2,2))
plot(rstandard(employed.fit3), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fit3))
qqnorm(residuals(employed.fit3))
qqline(residuals(employed.fit3))
pacf(rstandard(employed.fit3))
```




The residuals diagnostics of the AR(1)*SAR(2) model also shows no significant cause for concern. The histogram seems to indicate outliers may also be present for this model.

```
> detectAO(employed.fit1)
[1] "No AO detected"
> detectIO(employed.fit1)
              [,1]       [,2]
ind      18.000000  21.000000
lambda1   3.824261  -3.825527
```

```
> detectAO(employed.fit3)
              [,1]
ind      21.000000
lambda2  -4.259211
> detectIO(employed.fit3)
              [,1]       [,2]
ind      18.000000  21.000000
lambda1   4.044869  -4.101063
```

Checking for outliers, two outliers were found for both models. These outliers were found at observations 18 and 21 for both models. I will proceed by removing these innovative outliers.

```r
employed.fitIO1 = arima(Employed, order=c(0,1,1), io=c(18,21), seasonal=list(order=c(0,1,1), period=12))
employed.fitIO1
coeftest(employed.fitIO1)
Box.test(residuals(employed.fitIO1),lag=10, type="Ljung", fitdf=4)
```

```
Call:
arima(x = Employed, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1),
    period = 12), io = c(18, 21))

Coefficients:
          ma1     sma1     IO.18      IO.21
      -0.2466  -0.6100  300.3995  -301.0934
s.e.   0.0631   0.0782   81.8300    81.0647

sigma^2 estimated as 4895:  log likelihood = -926.54,  aic = 1861.08
```

```
z test of coefficients:

         Estimate  Std. Error z value  Pr(>|z|)    
ma1     -0.246629    0.063087 -3.9093 9.255e-05 ***
sma1    -0.610039    0.078222 -7.7988 6.250e-15 ***
IO.18  300.399537   81.830023  3.6710 0.0002416 ***
IO.21 -301.093422   81.064685 -3.7142 0.0002038 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
                Box-Ljung test

data:  residuals(employed.fitIO1)
X-squared = 10.17, df = 6, p-value = 0.1177
```

The MA(1)*SMA(1) Outlier Removed model has a sigma^2 of 4895, a log likelihood of -926.54, and an aic of 1861.08. All coefficients of this model are significant, and this model passes the Box-Ljung test.

```r
employed.fitIO3 = arima(Employed, order=c(1,1,0), io=c(18,21), seasonal=list(order=c(2,1,0), period=12))
employed.fitIO3
coeftest(employed.fitIO3)
Box.test(residuals(employed.fitIO3),lag=10, type="Ljung", fitdf=5)
```

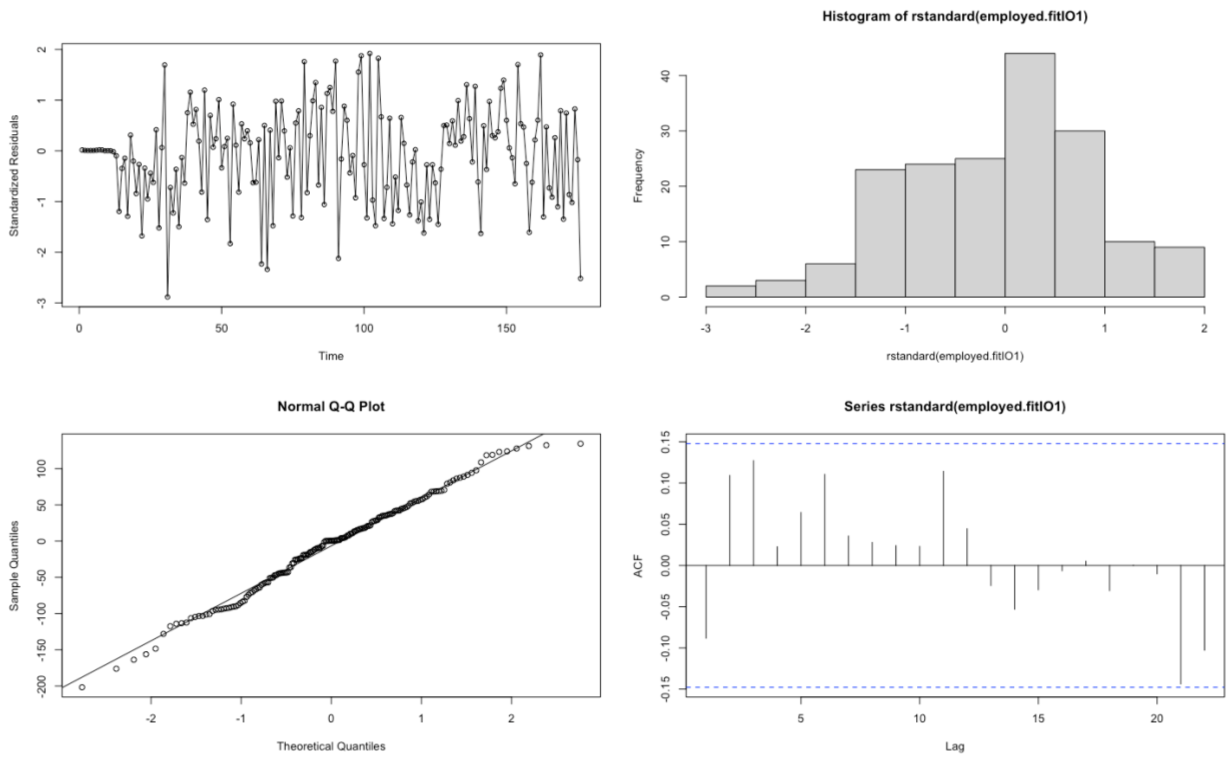
```
Call:
arima(x = Employed, order = c(1, 1, 0), seasonal = list(order = c(2, 1, 0),
    period = 12), io = c(18, 21))

Coefficients:
          ar1      sar1      sar2     IO.18      IO.21
      -0.2910   -0.3930   -0.2893  315.6377  -323.8575
s.e.   0.0695    0.0711    0.0710   77.9123    77.6286

sigma^2 estimated as 5029:  log likelihood = -927.59,  aic = 1865.18
```

```
z test of coefficients:

        Estimate  Std. Error z value  Pr(>|z|)
ar1    -0.291010    0.069485 -4.1881 2.813e-05 ***
sar1   -0.392964    0.071101 -5.5268 3.261e-08 ***
sar2   -0.289309    0.070986 -4.0756 4.590e-05 ***
IO.18  315.637674   77.912259  4.0512 5.096e-05 ***
IO.21 -323.857526   77.628609 -4.1719 3.021e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
	Box-Ljung test

data:  residuals(employed.fitIO3)
X-squared = 7.7728, df = 5, p-value = 0.1692
```

The AR(1)*SAR(2) Outlier Removed model has a sigma^2 of 5029, a log likelihood of -927.59, and an aic of 1865.18. All coefficients of this model are significant, and this model passes the Box-Ljung test.

```r
par(mfrow = c(2,2))
plot(rstandard(employed.fitIO1), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fitIO1))
qqnorm(residuals(employed.fitIO1))
qqline(residuals(employed.fitIO1))
acf(rstandard(employed.fitIO1))
```

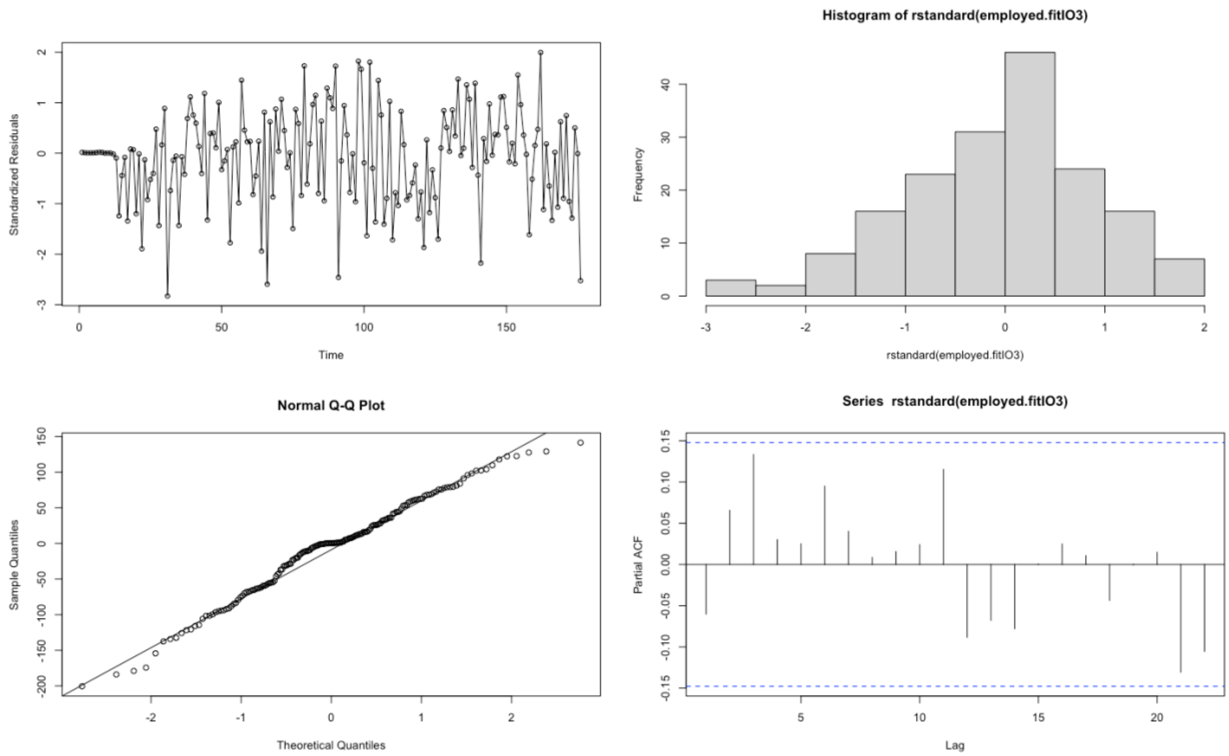



The MA(1)*SMA(1) Outlier Removed model's residual plots show no cause for concern and does not indicate the presence of any more outliers.

```
> detectAO(employed.fitIO1)
[1] "No AO detected"
> detectIO(employed.fitIO1)
[1] "No IO detected"
```

No outliers remain present for this model.

```
par(mfrow = c(2,2))
plot(rstandard(employed.fitIO3), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fitIO3))
qqnorm(residuals(employed.fitIO3))
qqline(residuals(employed.fitIO3))
pacf(rstandard(employed.fitIO3))
```




The AR(1)*SAR(2) Outlier Removed model's residual plots show no cause for concern and does not indicate the presence of any more outliers. There is a little unwanted variation around the middle of the QQ-Plot.

```
> detectAO(employed.fitIO3)
[1] "No AO detected"
> detectIO(employed.fitIO3)
[1] "No IO detected"
```

No outliers remain present for this model.

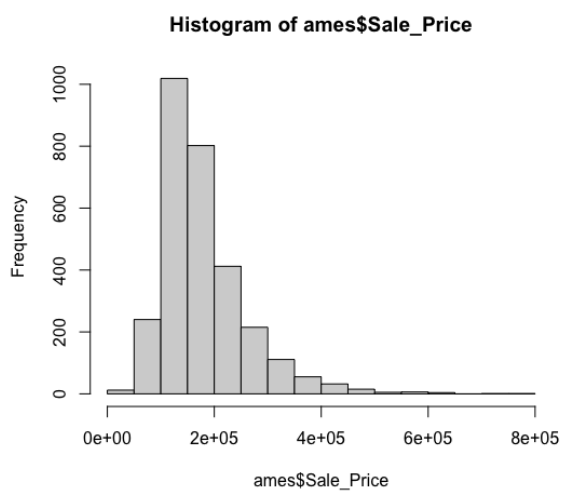
Based on the observation that the MA(1)*SMA(1) model is less complex, has a smaller singma^2, larger log likelihood, and lower AIC, I believe it is the appropriate model for this dataset.

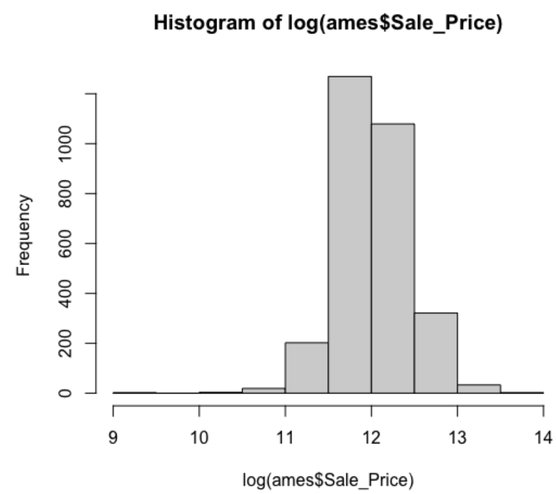
I did not plot the predictions for this model because I an error kept occurring (due to the presence of the Innovative Outlier removal process) that prevented me from doing so.

3. Ames Housing Trees
    a. Splitting Data

```
ames <- na.omit(ames)
ames <- mutate_if(ames, is.character, as.factor)
```

Before splitting the data into training and testing set, I cleaned the data by omitting all observations that contained NA values and converted all character variables into factors out of necessity for many of the decision tree functions to run.




Additionally, I found that it was necessary to take the log of Sale_Price for more normally distributed data and better decision tree performance.

```
set.seed(1)
idx = sample(1:nrow(ames), ceiling(nrow(ames)/2))
ames.train = ames[idx,]
ames.test = ames[-idx,]
```

| | |
|---|---|
| ames.test | 1465 obs. of 81 variables |
| ames.train | 1465 obs. of 81 variables |

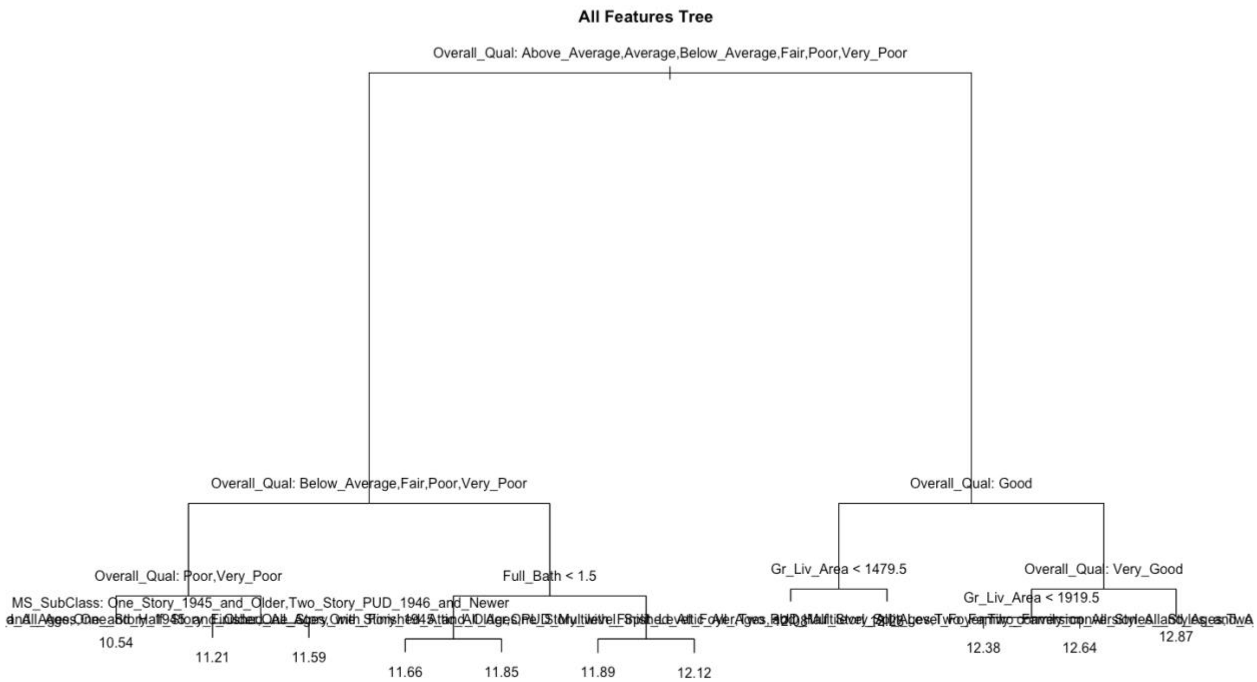
Finally, I was ready to split the data into training and testing sets. Each set consisted of 1465 observations of 81 variables.

b. Regression Tree with All Variables

```
tree.ames = tree(log(Sale_Price) ~ ., data = ames.train)
summary(tree.ames)
```

```
Regression tree:
tree(formula = log(Sale_Price) ~ ., data = ames.train)
Variables actually used in tree construction:
[1] "Overall_Qual" "MS_SubClass"  "Full_Bath"    "Gr_Liv_Area"
Number of terminal nodes:  12
Residual mean deviance:  0.03773 = 54.82 / 1453
Distribution of residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-1.0840000 -0.1090000  0.0006095  0.0000000  0.1108000  0.8167000
```

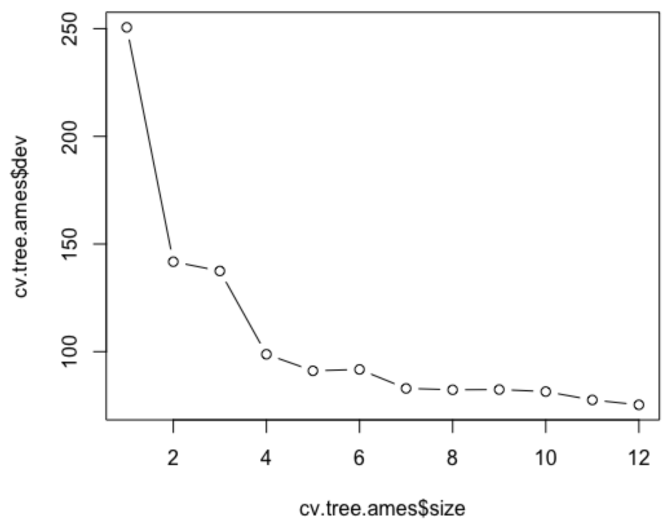
Using the above code, I created a regression tree considering all independent variables. The created tree had 12 terminal nodes decided by the classifications of Overall_Qual, MS_SubClass, Full_Bath, and Gr_Liv_Area. Although 80 factors were available for consideration, these four independent variables were determined by R as the deciding factors for the regression decision tree.




The plot of the unpruned regression tree is extremely complex and has 12 end nodes that each give a predicted value of log(Sale_Price). The initial predictor is Overall_Qual with observations that fall into the classification of Above_Average, Average, Below_Average, Fair, Poor, and Very_Poor following down the left side of the tree and any other classification following down the right. This process continues down both sides of the tree until each observation is placed into its predicted value of log(Sale_Price). It is my opinion that this tree is too complex and not easily interpretable with 12 end notes that are nearly impossible to read due to the structure of the plot.

c. Cross Validation Pruning and MSE Comparison

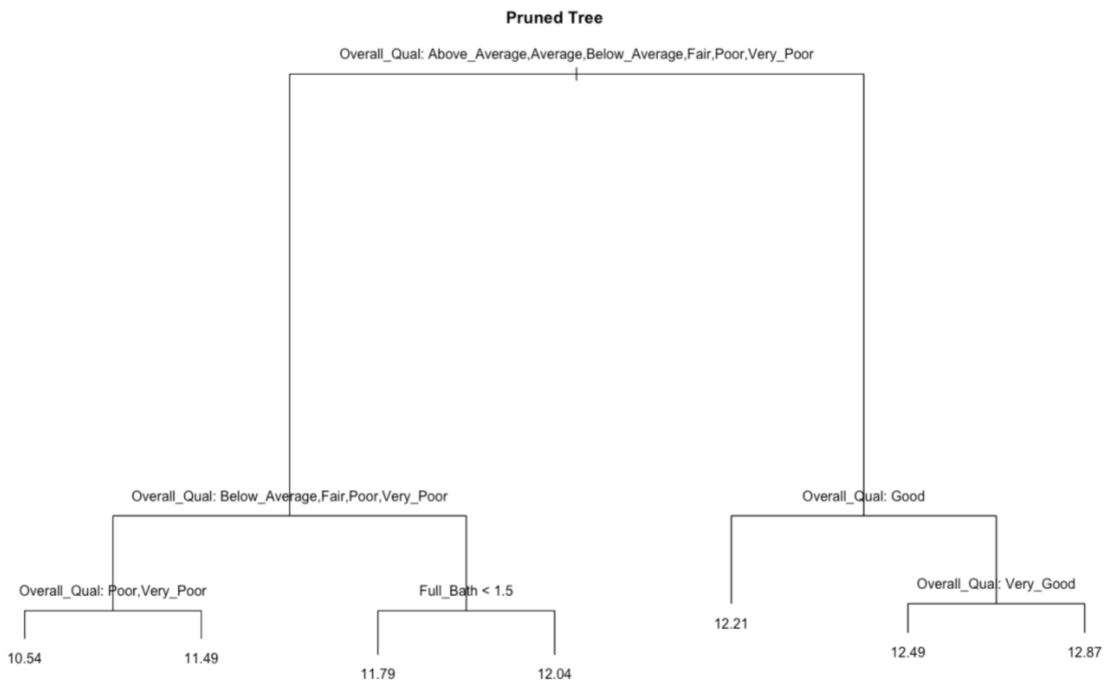
```
set.seed(1)
cv.tree.ames=cv.tree(tree.ames)
cv.tree.ames

plot(cv.tree.ames$size, cv.tree.ames$dev, type="b")
```




To prune the tree using cross validation I plotted the MSE with varying numbers of end nodes. From this plot I determined that a tree featuring seven nodes would be best for minimizing MSE while maximizing the interpretability of the tree.

```
prune.cv.mytree = prune.tree(tree.ames, best = 7)
summary(prune.cv.mytree)
```

```
Regression tree:
snip.tree(tree = tree.ames, nodes = c(14L, 6L, 10L, 11L, 9L))
Variables actually used in tree construction:
[1] "Overall_Qual" "Full_Bath"
Number of terminal nodes:  7
Residual mean deviance:  0.04895 = 71.37 / 1458
Distribution of residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-1.08400 -0.13060  0.01367  0.00000  0.12650  0.89930
```




Creating the seven node pruned tree, I observed that this tree only used two variables to predict log(Sale_Price). These variables were Overall_Qual and Full_Bath (number of full bathrooms). Starting off, this tree begins its decision making with Overall_Qual. Observations that have Overall_Qual of Above_Average, Average, Below_Average, Fair, Poor, and Very_Poor proceed down the left side of the tree while any other classification proceeds down the right. From there this process continues down both sides by further splitting observations by Overal_Qual, and Full_Bath (in one instance) to create a predicted Log(Sale_Price) for the observations. Overall, this tree is much simpler and far more interpretable than the unpruned tree.

```
yhat.pruned <- predict(prune.cv.mytree, ames.test)

mse.pruned <- mean((exp(yhat.pruned) - (ames.test$Sale_Price))^2)
mse.pruned

yhat.unpruned <- predict(tree.ames, ames.test)

mse.unpruned <- mean((exp(yhat.unpruned) - (ames.test$Sale_Price))^2)
mse.unpruned
```

```
> mse.unpruned
[1] 1544940571
```

```
> mse.pruned
[1] 1846488155
```

When making predictions, I wanted the results to be more interpretable. To do this, I called the exp() function on the predicted values when calculating the MSE. The MSE value for the unpruned tree was 1,544,940,571 and 1,846,488,155 for the pruned tree. Both values do not seem to be very good. Despite the unpruned tree having a lower MSE, I would recommend the pruned tree due to its significantly increased interpretability. The unpruned tree could be ideal if you did not have to explain how a prediction was made, but that seems circumstantial so I cannot recommend it.

d. Bagged Regression Tree

```
set.seed(1)
bag.ames = randomForest(log(Sale_Price) ~ ., data = ames.train, mtry=80, importance =TRUE)
bag.ames

yhat.bag = predict(bag.ames , newdata = ames.test)

mse.bag <- mean((exp(yhat.bag) - (ames.test$Sale_Price))^2)
mse.bag
```

```
Call:
 randomForest(formula = log(Sale_Price) ~ ., data = ames.train,      mtry = 80, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 80

          Mean of squared residuals: 0.02182386
                    % Var explained: 87.21
```

```
> mse.bag
[1] 613284495
```

I used the above code to create the bagged regression tree and calculate the MSE. For this tree I specified an mtry parameter of 80 as it coincides with the number of independent variables in this data. During this process all 80 variables were tested at each split to determine the best tree. I then created predictions for the test set and found the MSE of these predictions to be 613,284,495 which is a significant upgrade from the pruned and unpruned regression trees.

e. Random Forrest Regression Tree

```
set.seed(1)
rf.ames = randomForest(log(Sale_Price) ~ ., data = ames.train, mtry=27, importance =TRUE)
rf.ames

yhat.rf = predict(rf.ames, newdata = ames.test)
mse.rf <- mean((exp(yhat.rf) - (ames.test$Sale_Price))^2)
mse.rf
```

```
Call:
 randomForest(formula = log(Sale_Price) ~ ., data = ames.train,      mtry = 27, importance = TRUE)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 27

          Mean of squared residuals: 0.02070994
                    % Var explained: 87.86
```

```
> mse.rf
[1] 589629951
```

I used the above code to create the random forest regression tree and calculate the MSE. I specified a mtry parameter of 27 as it was the standard recommendation this method of tree (mtry = P/3). Using this random forest regression tree, I then created predictions for the test set and found the MSE of these predictions to be 589,629,951 a further upgrade from the bagged regression tree.

f. Boosted Regression Tree

```
set.seed(1)
boost.ames1 = gbm(log(Sale_Price) ~ ., data=ames.train, distribution="gaussian",
                  n.trees=500, interaction.depth=4)
boost.ames1

yhat.boost1 = predict(boost.ames1 , newdata = ames.test, n.trees=500)

mse.boost1 <- mean((exp(yhat.boost1)-(ames.test$Sale_Price))^2)
mse.boost1
```

```
gbm(formula = log(Sale_Price) ~ ., distribution = "gaussian",
    data = ames.train, n.trees = 500, interaction.depth = 4)
A gradient boosted model with gaussian loss function.
500 iterations were performed.
There were 80 predictors of which 70 had non-zero influence.
```

```
> mse.boost1
[1] 415987831
```

I used the above code to create a boosted regression tree and calculate the MSE. When making this tree I specified the distribution method as gaussian as this was not an instance of binary classification. Looking at the output summary of this tree, R determined that 70 of the 80 variables had some sort of influence on log(Sale_Price). I then created predictions for the test set and found the MSE of those predictions to be 415,987,831.

When trying to tweak the n.tree and shrinkage parameters I found that they it did not help the model. When increasing the n.tree parameter it also increased the MSE. This is likely due to a larger number of created trees which led to overfitting on the training data hurting its ability to make prediction on the testing data.

g. All Trees MSE Comparison

```
# Unpruned          # Pruned           # Bagged         # Random Forrest
mse.unpruned        mse.pruned         mse.bag          mse.rf
# 1,544,940,571     # 1,846,488,155    # 613,284,495    # 589,629,951
```

```
# Boosted
mse.boost1
# 415,987,831
```

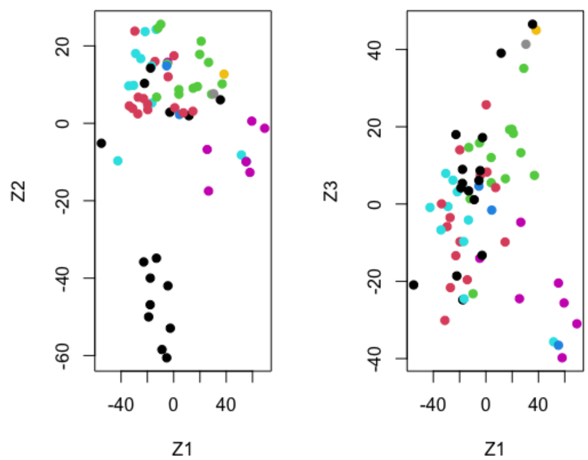
Comparing the MSE of each of the trees it can be seen that the boosted regression tree performed the best with an Mean Square Error of 415,987,831.

4. NCI Labs Principal Components and K-Means Clustering
    a. PCA

```
pr.out = prcomp(nci.data, scale=TRUE)
```

The above code was used to preform PCA on the NCI data.

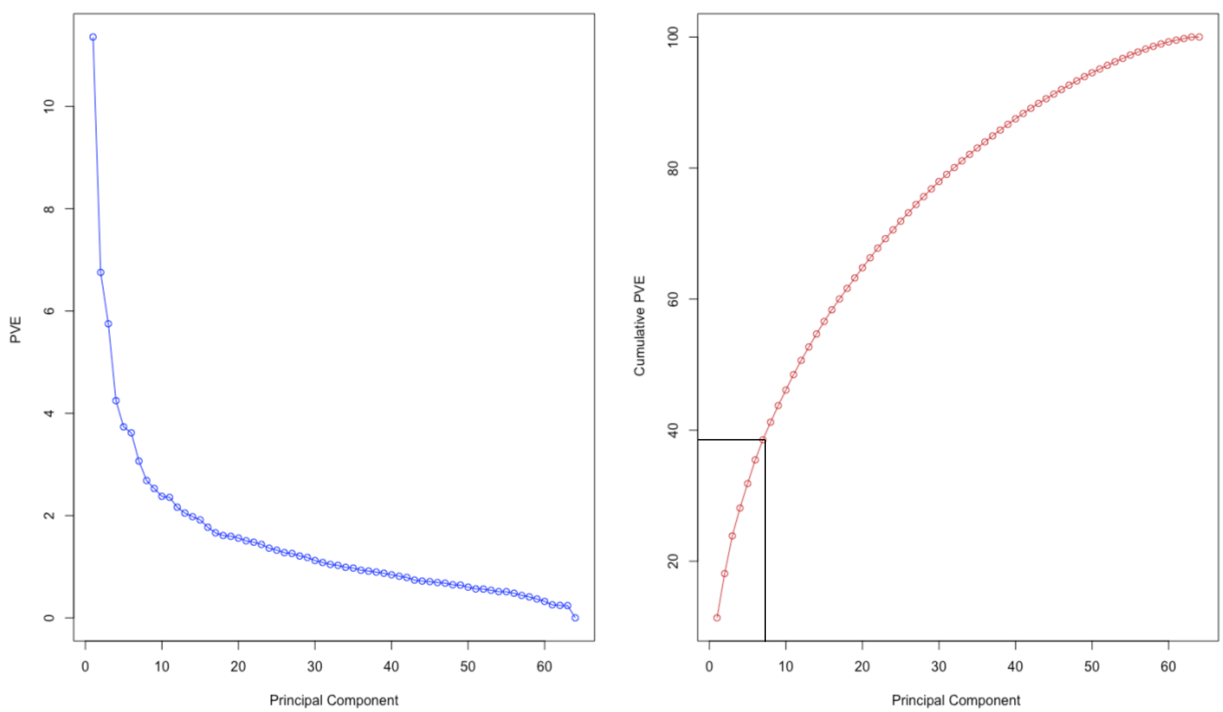
    b. First Two Principal Component Plots




Using the code initially found in the Assignment 10 Instructions PDF, I have plotted the first two principal components. It can in fact be observed that observations belonging to a single cancer type are somewhat group together in this plot.

c. PVE (Scree) and Cumulative PVE Plots

```
pve =100* pr.out$sdev ^2/ sum(pr.out$sdev ^2)


par(mfrow =c(1,2))

plot(pve , type ="o", ylab="PVE ", xlab=" Principal Component ",
     col =" blue")
plot(cumsum (pve ), type="o", ylab =" Cumulative PVE", xlab="
Principal Component ", col =" brown3 ")
```




Using the above code, I made both a PVE (Scree) and Cumulative PVE plot for the NCI data. Observing the Cumulative PVE plot, we see that just under 40% of variance is based on the first seven principal components. Based on the PVE (Scree) Plot we would likely want to choose anywhere from eight to fifteen principal components (I would lean towards eight). At eight principal components we see diminishing returns as the velocity of the plot begin to flatten and by the fifteenth principal component the rate of change in the plot appears to be linear. Depending on the desired complexity your model you would likely choose within this range.

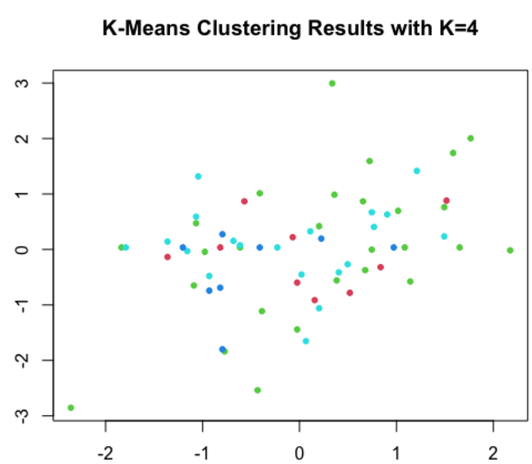
d. K-Means Clustering

```
scaled_nci_data <- scale(nci.data)

mean(scaled_nci_data)
sd(scaled_nci_data)
```

```
> mean(scaled_nci_data)
[1] 4.108049e-19
> sd(scaled_nci_data)
[1] 0.9921579
```

Scaling the NCI data using the scale() function, we can see that there is a mean of approximately 0 (4.108e-19) and standard deviation of approximately 1 (.9922).

```
set.seed(1)
NCI_data_K4 <- kmeans(scaled_nci_data, 4, nstart = 20)

par(mfrow = c(1,1))
plot(scaled_nci_data, col =(NCI_data_K4$cluster +1 ) , main="K-Means Clustering Results with K=4",
     xlab ="", ylab="", pch =20, cex =1)
```




Using the above code I preformed K-Means Clustering featuring four clusters. Plotting these clusters, it can be observed that each cluster is plotted using a different color, it is important to note that since I are clustering in over 6000 dimensions, a 2-D plot is not an inherently helpful for visualizing this K-Means Clustering.

```
> NCI_data_K4$size
[1]  9 27  8 20
>
>
> 100*NCI_data_K4$betweenss/NCI_data_K4$totss
[1] 19.92218
```

This K-Means Clustering produced clusters of sizes 9, 27, 8, and 20. Additionally, the between clusters sum of squares as a % of total sum of squares for this K-Means Clustering was 19.922%.

CODE

```
# BSAN 450 Final
# Grant Healy

library(TSA)
library(xts)
library(lmtest)
library(ISLR)
library(dplyr)
library(randomForest)
library(gbm)
library(tree)


# Question 1 ----

# Read in Data

souvenir <- read.csv("souvenir.csv")
attach(souvenir)


# Initial Plot

par(mfrow=c(1,1))
plot(Sales, ylab='Sales', xlab='Month', type='o')


# Variability Difference Fix

BoxCox.ar(y=Sales, method = "yule-walker")

souvenir$logSales <- log(Sales)

plot(souvenir$logSales, ylab='Log Sales', xlab='Month', type='o')
```

# Determining Differences

```
par(mfrow=c(1,1))
plot(diff(souvenir$logSales), ylab='Log Sales 1st Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(souvenir$logSales), lag.max=36)
pacf(diff(souvenir$logSales), lag.max=36)


par(mfrow=c(1,1))
plot(diff(souvenir$logSales,lag=12), ylab='Log Sales 12th Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(souvenir$logSales,lag=12), lag.max=36)
pacf(diff(souvenir$logSales,lag=12), lag.max=36)


par(mfrow=c(1,1))
plot(diff(diff(souvenir$logSales),lag=12), ylab='Log Sales 1st and 12th', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(diff(souvenir$logSales,lag=12)), lag.max=36)
pacf(diff(diff(souvenir$logSales,lag=12)), lag.max=36)
```

# Model Fitting

```
souvenir.fit1 = arima(souvenir$logSales, order=c(1,1,0), seasonal=list(order=c(0,1,0),
period=12))
souvenir.fit1
coeftest(souvenir.fit1)
Box.test(residuals(souvenir.fit1),lag=10, type="Ljung", fitdf=1)


souvenir.fit2 = arima(souvenir$logSales, order=c(0,1,1), seasonal=list(order=c(0,1,0),
period=12))
souvenir.fit2
coeftest(souvenir.fit2)
Box.test(residuals(souvenir.fit2),lag=10, type="Ljung", fitdf=1)
```

```
souvenir.fit3 = arima(souvenir$logSales, order=c(0,1,1), seasonal=list(order=c(0,1,2),
period=12))
souvenir.fit3
coeftest(souvenir.fit3)
Box.test(residuals(souvenir.fit3),lag=10, type="Ljung", fitdf=3)

# Residual Diagnostics

par(mfrow = c(2,2))
plot(rstandard(souvenir.fit1), ylab='Standardized Residuals', type='o')
hist(rstandard(souvenir.fit1))
qqnorm(residuals(souvenir.fit1))
qqline(residuals(souvenir.fit1))
pacf(rstandard(souvenir.fit1))


par(mfrow = c(2,2))
plot(rstandard(souvenir.fit2), ylab='Standardized Residuals', type='o')
hist(rstandard(souvenir.fit2))
qqnorm(residuals(souvenir.fit2))
qqline(residuals(souvenir.fit2))
acf(rstandard(souvenir.fit2))


par(mfrow = c(2,2))
plot(rstandard(souvenir.fit3), ylab='Standardized Residuals', type='o')
hist(rstandard(souvenir.fit3))
qqnorm(residuals(souvenir.fit3))
qqline(residuals(souvenir.fit3))
acf(rstandard(souvenir.fit3))


# Outliers

detectAO(souvenir.fit2)
detectIO(souvenir.fit2)


# Prediction Check

predict(souvenir.fit2, n.ahead=24)

par(mfrow=c(1,1))
plot(souvenir.fit2, n.ahead=24, type='b', col='red')
```

```
# Question 2 ----

# Read in Data

emales <- read.csv("emales.CSV")
attach(emales)


# Initial Plot

par(mfrow=c(1,1))
plot(Employed, ylab='Employed Males', xlab='Month', type='o')


# Determining Differences

par(mfrow=c(1,1))
plot(diff(Employed), ylab='Log Employed 1st Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(Employed), lag.max=36)
pacf(diff(Employed), lag.max=36)


par(mfrow=c(1,1))
plot(diff(Employed,lag=12), ylab='Log Employed 12th Dif', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(Employed,lag=12), lag.max=36)
pacf(diff(Employed,lag=12), lag.max=36)


par(mfrow=c(1,1))
plot(diff(diff(Employed),lag=12), ylab='Log Sales', xlab='Month', type='o')

par(mfrow=c(1,2))
acf(diff(diff(Employed,lag=12)), lag.max=36)
pacf(diff(diff(Employed,lag=12)), lag.max=36)
```

```
# Model Fitting

employed.fit1 = arima(Employed, order=c(0,1,1), seasonal=list(order=c(0,1,1), period=12))
employed.fit1
coeftest(employed.fit1)
Box.test(residuals(employed.fit1),lag=10, type="Ljung", fitdf=2)


employed.fit2 = arima(Employed, order=c(1,1,0), seasonal=list(order=c(1,1,0), period=12))
employed.fit2
coeftest(employed.fit2)
Box.test(residuals(employed.fit2),lag=10, type="Ljung", fitdf=2)


employed.fit3 = arima(Employed, order=c(1,1,0), seasonal=list(order=c(2,1,0), period=12))
employed.fit3
coeftest(employed.fit3)
Box.test(residuals(employed.fit3),lag=10, type="Ljung", fitdf=3)


# Residual Diagnostics

par(mfrow = c(2,2))
plot(rstandard(employed.fit1), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fit1))
qqnorm(residuals(employed.fit1))
qqline(residuals(employed.fit1))
acf(rstandard(employed.fit1))


par(mfrow = c(2,2))
plot(rstandard(employed.fit2), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fit2))
qqnorm(residuals(employed.fit2))
qqline(residuals(employed.fit2))
pacf(rstandard(employed.fit2))


par(mfrow = c(2,2))
plot(rstandard(employed.fit3), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fit3))
qqnorm(residuals(employed.fit3))
qqline(residuals(employed.fit3))
pacf(rstandard(employed.fit3))
```

```
# Outliers

detectAO(employed.fit1)
detectIO(employed.fit1)

detectAO(employed.fit3)
detectIO(employed.fit3)

# Resolve Outliers

employed.fitIO1 = arima(Employed, order=c(0,1,1), io=c(18,21), seasonal=list(order=c(0,1,1),
period=12))
employed.fitIO1
coeftest(employed.fitIO1)
Box.test(residuals(employed.fitIO1),lag=10, type="Ljung", fitdf=4)

par(mfrow = c(2,2))
plot(rstandard(employed.fitIO1), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fitIO1))
qqnorm(residuals(employed.fitIO1))
qqline(residuals(employed.fitIO1))
acf(rstandard(employed.fitIO1))

detectAO(employed.fitIO1)
detectIO(employed.fitIO1)


employed.fitIO3 = arima(Employed, order=c(1,1,0), io=c(18,21), seasonal=list(order=c(2,1,0),
period=12))
employed.fitIO3
coeftest(employed.fitIO3)
Box.test(residuals(employed.fitIO3),lag=10, type="Ljung", fitdf=5)

par(mfrow = c(2,2))
plot(rstandard(employed.fitIO3), ylab='Standardized Residuals', type='o')
hist(rstandard(employed.fitIO3))
qqnorm(residuals(employed.fitIO3))
qqline(residuals(employed.fitIO3))
pacf(rstandard(employed.fitIO3))

detectAO(employed.fitIO3)
detectIO(employed.fitIO3)
```

```
# Question 3 ----

# Read in Data

ames <- read.csv("ames-1.csv")
attach(ames)


# Cleaning Data

ames <- na.omit(ames)
ames <- mutate_if(ames, is.character, as.factor)


# Hist

par(mfrow=c(1,1))
hist(ames$Sale_Price)
hist(log(ames$Sale_Price))


# A. Split the Data

set.seed(1)
idx = sample(1:nrow(ames), ceiling(nrow(ames)/2))
ames.train = ames[idx,]
ames.test = ames[-idx,]


# B. Initial Tree with All Variables

tree.ames = tree(log(Sale_Price) ~ ., data = ames.train)
summary(tree.ames)

plot(tree.ames)
text(tree.ames, pretty = 0)
title("All Features Tree")
```

# C. Tree Pruning

```
set.seed(1)
cv.tree.ames=cv.tree(tree.ames)
cv.tree.ames

plot(cv.tree.ames$size, cv.tree.ames$dev, type="b")

cv.tree.ames$size[which.min(cv.tree.ames$dev)]

prune.cv.mytree = prune.tree(tree.ames, best = 7)
summary(prune.cv.mytree)

plot(prune.cv.mytree)
text(prune.cv.mytree, pretty = 0)
title("Pruned Tree")
```

# C2. MSE Comparison

```
yhat.pruned <- predict(prune.cv.mytree, ames.test)

mse.pruned <- mean((exp(yhat.pruned) - (ames.test$Sale_Price))^2)
mse.pruned

yhat.unpruned <- predict(tree.ames, ames.test)

mse.unpruned <- mean((exp(yhat.unpruned) - (ames.test$Sale_Price))^2)
mse.unpruned
```

# D. Bagged Tree

```
set.seed(1)
bag.ames = randomForest(log(Sale_Price) ~ ., data = ames.train, mtry=80, importance =TRUE)
bag.ames

yhat.bag = predict(bag.ames , newdata = ames.test)

mse.bag <- mean((exp(yhat.bag) - (ames.test$Sale_Price))^2)
mse.bag
```

# E. Random Forrest

```
set.seed(1)
rf.ames = randomForest(log(Sale_Price) ~ ., data = ames.train, mtry=27, importance =TRUE)
rf.ames

yhat.rf = predict(rf.ames, newdata = ames.test)
mse.rf <- mean((exp(yhat.rf) - (ames.test$Sale_Price))^2)
mse.rf
```

# F. Boosted Tree

```
set.seed(1)
boost.ames1 = gbm(log(Sale_Price) ~ ., data=ames.train, distribution="gaussian",
            n.trees=500, interaction.depth=4)
boost.ames1

yhat.boost1 = predict(boost.ames1 , newdata = ames.test, n.trees=500)

mse.boost1 <- mean((exp(yhat.boost1)-(ames.test$Sale_Price))^2)
mse.boost1


set.seed(1)
boost.ames2 = gbm(log(Sale_Price) ~ ., data=ames.train, distribution="gaussian",
            n.trees=1000, interaction.depth=6)
boost.ames2

yhat.boost2 = predict(boost.ames2 , newdata = ames.test, n.trees=1000)

mse.boost2 <- mean((exp(yhat.boost2)-(ames.test$Sale_Price))^2)
mse.boost2
```

# G. Tree Comparison

```
# Unpruned
mse.unpruned
# 1,544,940,571

# Pruned
mse.pruned
# 1,846,488,155

# Bagged
mse.bag
# 613,284,495

# Random Forrest
mse.rf
# 589,629,951

# Boosted
mse.boost1
# 415,987,831

mse.boost2
# 442,262,217
```

# Question 4 ----

# Read in Data

```
nci.labs=NCI60$labs
nci.data=NCI60$data
```

# A. Precomp with PCA Function / Scale = True

```
pr.out = prcomp(nci.data, scale=TRUE)
```

# B. First Two Principal Component Plots

```
par(mfrow =c(1,2))

plot(pr.out$x [,1:2], col=as.factor(nci.labs), pch =19,
    xlab ="Z1",ylab="Z2")
plot(pr.out$x[,c(1,3) ], col=as.factor(nci.labs), pch =19,
    xlab ="Z1",ylab="Z3")
```

# C. PVE and Cumulative PVE Plots

```
pve =100* pr.out$sdev ^2/ sum(pr.out$sdev ^2)


par(mfrow =c(1,2))

plot(pve , type ="o", ylab="PVE ", xlab=" Principal Component ",
    col =" blue")
plot(cumsum (pve ), type="o", ylab =" Cumulative PVE", xlab="
Principal Component ", col =" brown3 ")
```

# D. Scaling and K-Means Clustering

```
scaled_nci_data <- scale(nci.data)

mean(scaled_nci_data)
sd(scaled_nci_data)

set.seed(1)
NCI_data_K4 <- kmeans(scaled_nci_data, 4, nstart = 20)

par(mfrow = c(1,1))
plot(scaled_nci_data, col =(NCI_data_K4$cluster +1 ) , main="K-Means Clustering Results with
K=4",
    xlab ="", ylab="", pch =20, cex =1)


NCI_data_K4$size

100*NCI_data_K4$betweenss/NCI_data_K4$totss
```