Patrick O’Brien

Stat 3504

Final Project

Analysis of Average Superbowl Viewership Data

**Abstract**

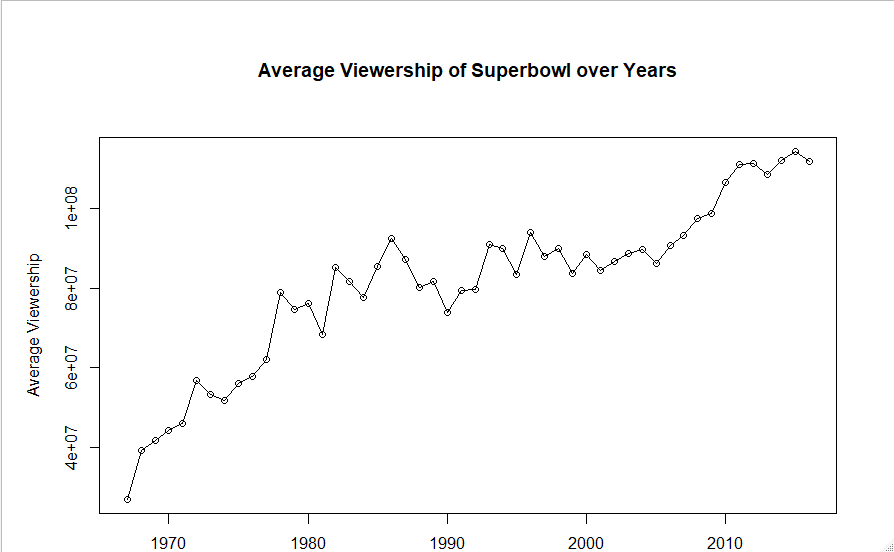
This report explores the history of the average viewership of the Superbowl, which originally began in 1967, with complete data to 2019. After many tests, we found that we can model this viewership with a properly specified and viable ARIMA(1,2,1) model. We then compared the model based on the trained data from 1967-2016, with actual observations from 2017-2019 to see if our model did the trick.

**Introduction**

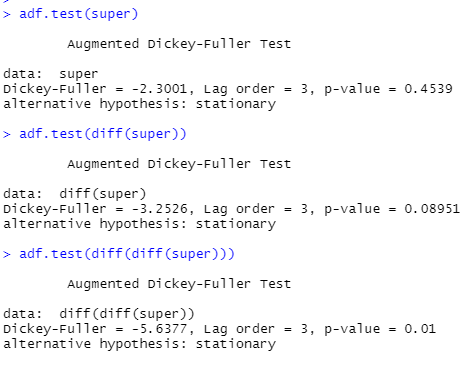
The Superbowl is now a classic American tradition that a large portion of the country takes part in viewing. It has not always been that way, however. Viewership has steadily increased since its inception in 1967, and I was curious if there was a way to model this average viewership, to predict just how many people will tune in to watch in the future. Corporate companies, political candidates and more use the Superbowl as a way to advertise their product or message, and finding if this increase in viewership will continue and is consistent is important to see if it is worth the money it costs to advertise during the event. This paper will apply several tools to identify, check, and find a proper model that we can then use to forecast future average viewership of the Superbowl.

**Identifying Models**

First, I plotted my data to get a general sense of what I was working with.



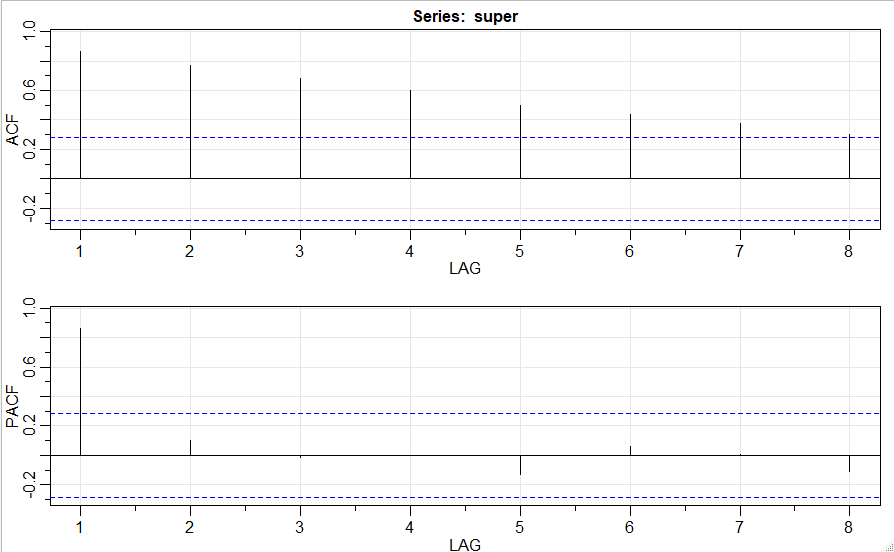
After this, I decided to run a Dicky-Fuller root test, to see if our series is stationary, and if not, at which order of difference it is likely to be stationary.



From these tests, we can see that we will likely be the most successful with models with first and second order differences. Keeping this in mind, we can create the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots for the time series, and its first and second order differences and interpret them.

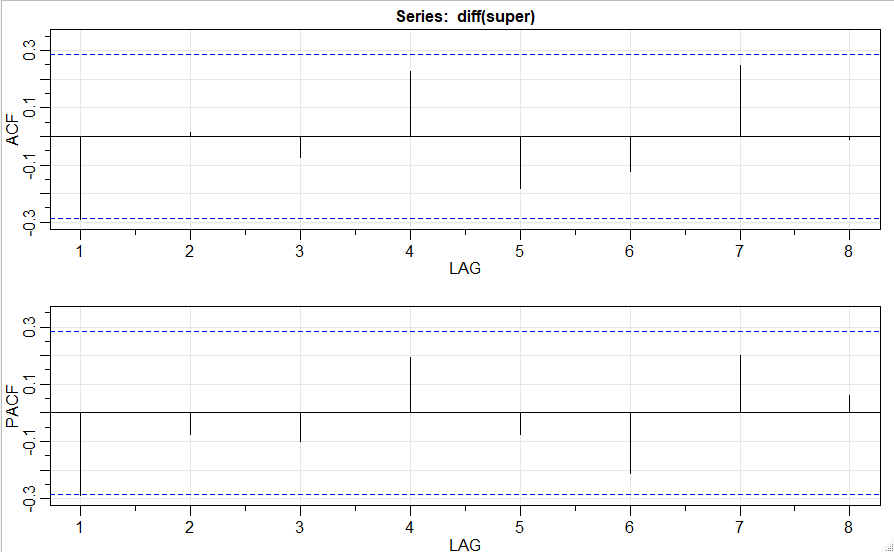
As I find options of models, I will label them in parentheses as such: (#), to keep track.

For the normal dataset:



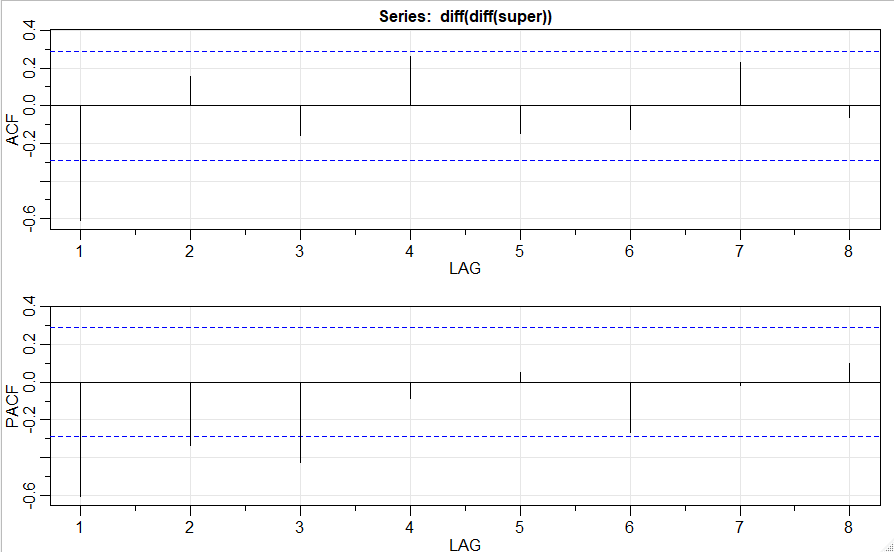
The ACF decays to 0 and the PACF and cuts off after lag 1, so this suggests that the AR(1) model (#1) could be a good fit.

For the first order difference:



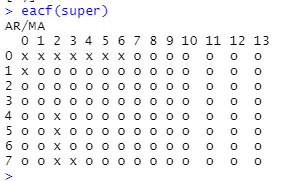
All lags are within the bounds of the ACF and PACF, so we can not gather much information from this as to which models may be a good fit for our data.

Lastly, for the second order difference:

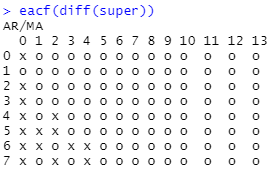


For this ACF plot, we see that it cuts off after lag 1, and the PACF decays to 0. This suggests that the ARIMA(0,2,1) model (#2) could be a good fit for our data.

In addition to the ACF and PACF plots, we can also use the Extended Autocorrelation Function (EACF) plot to find possible models. The EACF plot for the normal data is:

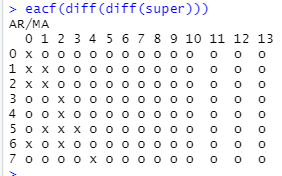
  
From this, we can see that the ARMA(1,1) (#3) and ARMA (1,2) (#4) models are models we can possibly fit to our data.

For the first order difference:



From this, we can see that ARIMA(0,1,1) (#5) and ARIMA(1,1,1) (#6) are possible options.

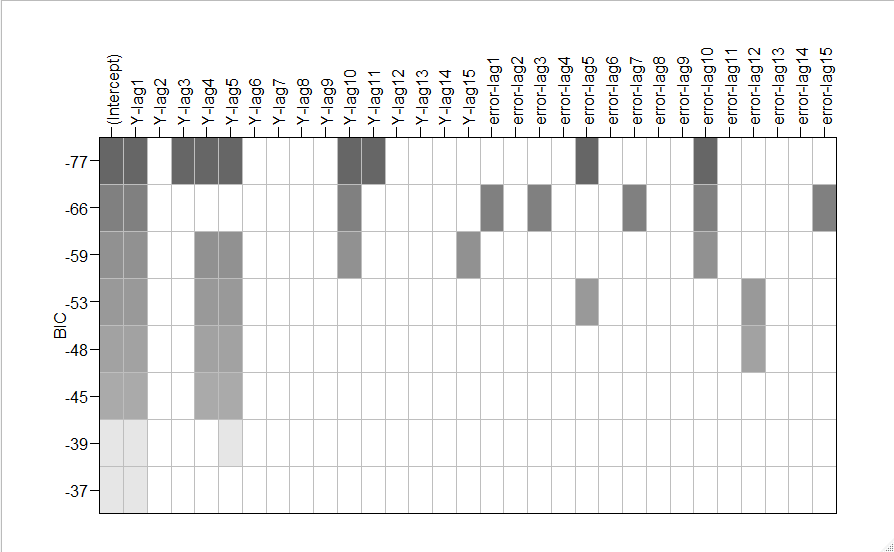
For the second order difference:



From this, the best option would definitely be an ARIMA(0,2,1) model, which we have already interpreted from the ACF and PACF plots, so this may be a strong fit!

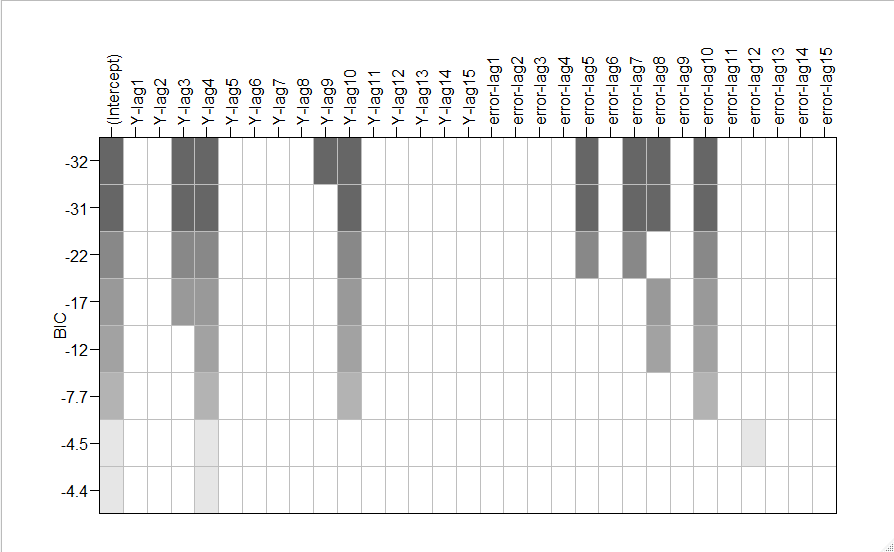
We can use the armasubsets function in R to find possible specifications for a strong model as well, based on the best ARMA subsets.

For the normal data:



From this, we can see that the first and third lags are definitely strong, and the 5th MA term is significant too. From this, we can add ARMA(1,5) (#7) and AR(3) (#8) to our list.

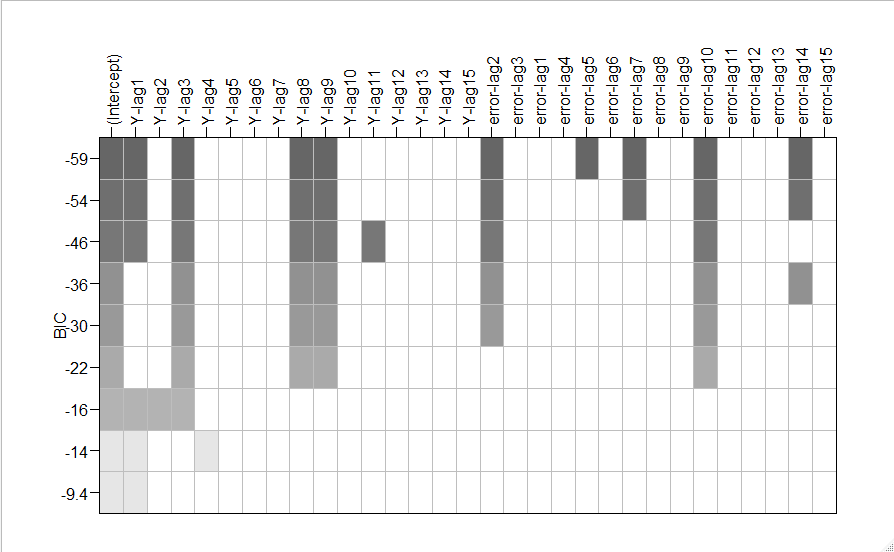
For first order differencing, this is the ARMA subset:



From this, though they will likely be overfit as the orders are quite high, we will add ARIMA(3,1,0) (#9)

and ARIMA(3,1,5) (#10) to our list as well.

Lastly, for the second order difference:



And from this, we can see that ARIMA(1,2,0) (#11), ARIMA(1,2,2) (#12), and ARIMA(3,2,2) (#13) as possible options.

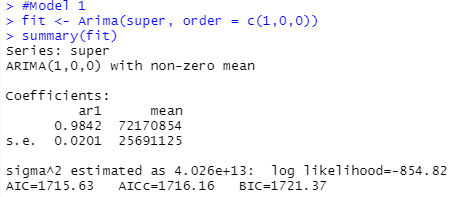
I would like to add ARIMA(1,2,1) (#14) as an option, as when checking the parameters of ARIMA(1,2,2), the second MA term was the only one that was overfit. Also, a very popular function in R, auto.arima, specified an ARIMA(0,1,1) (#15) with drift as the best option, so we will add that as well.

**Check Overfitting**

The first and easiest way to narrow down our list of options is to make sure that the models are not overfit. A term in a model, and therefore the model itself, is overfit if 2 times the standard error of the parameter, subtracted by the absolute value of the parameter itself, is less than 0. We will go through this process with our models from the previous section to determine which models are not overfit, and therefore still valid options.

Model 1:

AR(1)



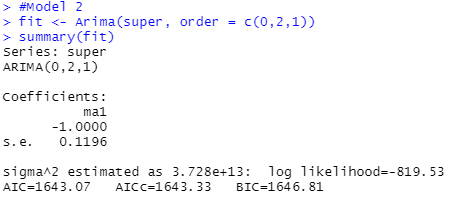
ϕ1 = 0.9842

0.9842 – 2(0.0201) > 0, so it is significantly different than 0.

Model 1 is still an option, as it is not overfit.

Model 2:

ARIMA(0,2,1)



θ1 = -1

|-1| - 2(0.1196) > 0, so it is significantly different than 0.

Model 2 is still an option, as it is not overfit.

Model 3:

ARMA(1,1)



R cannot compute this as a possible fit, so we will not consider it moving forward.

Model 4:

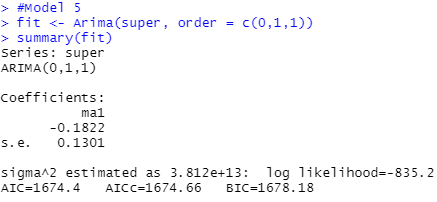
ARMA(2,1)



R cannot compute this as a possible fit, so we will not consider it moving forward.

Model 5:

IMA(1,1)



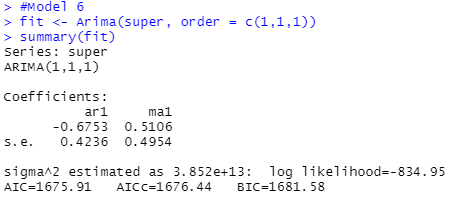
θ1 = -0.1822

|-0.1822| - 2(0.1301) < 0, so it is not significantly different from 0.

Model 5 is overfit, and we will not consider it moving forward.

Model 6:

ARIMA(1,1,1)



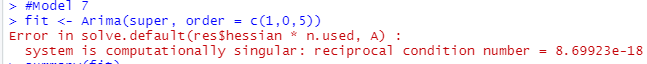
ϕ1 = -0.6753

|-0.6753| - 2(0.4236) < 0, so it is not significantly different from 0.

Model 6 is overfit, and we will not consider it moving forward.

Model 7:

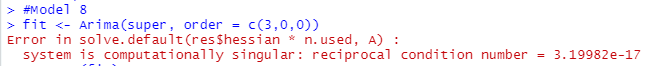
ARMA(1,5)



R cannot compute this as a possible fit, so we will not consider it moving forward.

Model 8:

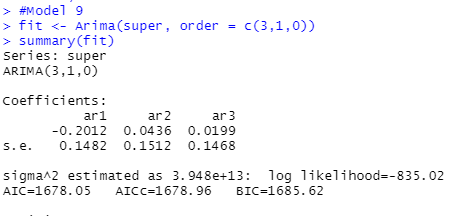
AR(3)



R cannot compute this as a possible fit, so we will not consider it moving forward.

Model 9:

ARI(3,1)



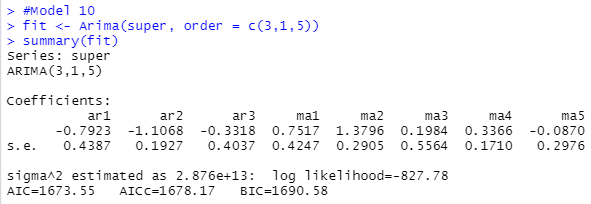
ϕ1 = -0.2012

|-0.2012| - 2(0.1482) < 0, so it is not significantly different from 0.

Model 9 is overfit, and we will not consider it moving forward.

Model 10:

ARIMA(3,1,5)



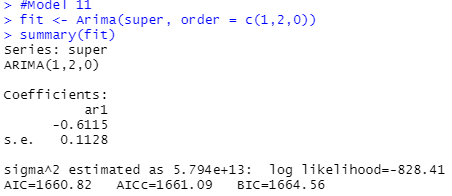
ϕ1 = -0.7923

|-0.7923| - 2(0.4387) < 0, so it is not significantly different from 0.

Model 10 is overfit, and we will not consider it moving forward.

Model 11:

ARI(1,2)



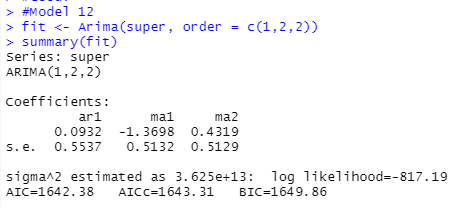
ϕ1 = -0.6115

|-0.6115| – 2(0.1128) > 0, so it is significantly different than 0.

Model 11 is still an option, as it is not overfit.

Model 12:

ARIMA(1,2,2)



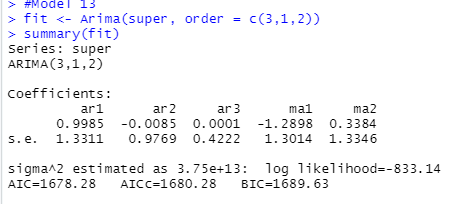
ϕ1 = 0.0932

0.0932 - 2(0.5537) < 0, so it is not significantly different from 0.

Model 12 is overfit, and we will not consider it moving forward.

Model 13:

ARIMA(3,1,2)



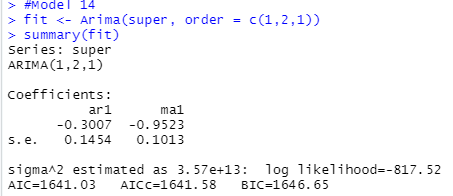
ϕ1 = 0.9985

0.9985 – 2(1.3311) < 0, so it is not significantly different from 0.

Model 13 is overfit, and we will not consider it moving forward.

Model 14:

ARIMA(1,2,1)



ϕ1 = -0.3007

|-0.3007| – 2(0.1454) > 0, so it is significantly different than 0.

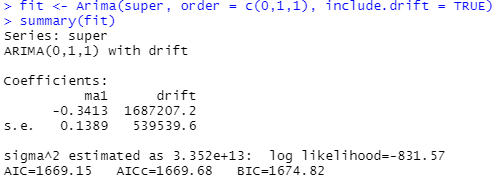
θ1 = -0.6115

|-0.9523| – 2(0.1013) > 0, so it is significantly different than 0.

Model 14 is still an option, as it is not overfit.

Model 15:

ARMA(0,1,1) with Drift



θ1 = -0.3413

|-0.3413| – 2(0.1389) > 0, so it is significantly different than 0.

drift = 1687207.2

1687207.2 – 2(539539.6) > 0, so it is significantly different than 0.

Model 15 is still an option, as it is not overfit.

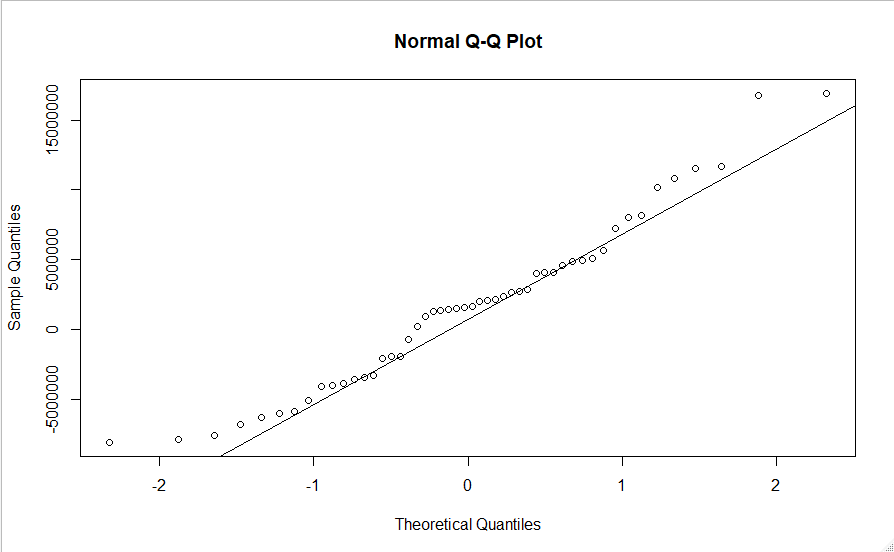
After all of this, we narrowed down our 15 options for models to 5. Models 1, 2, 11, 14, 15 are not overfit, and are still all valid options.

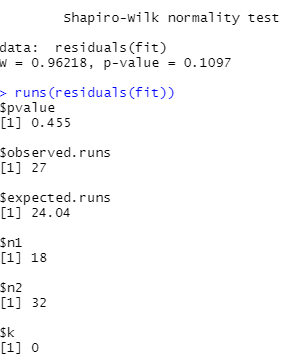
**Model Diagnostics**

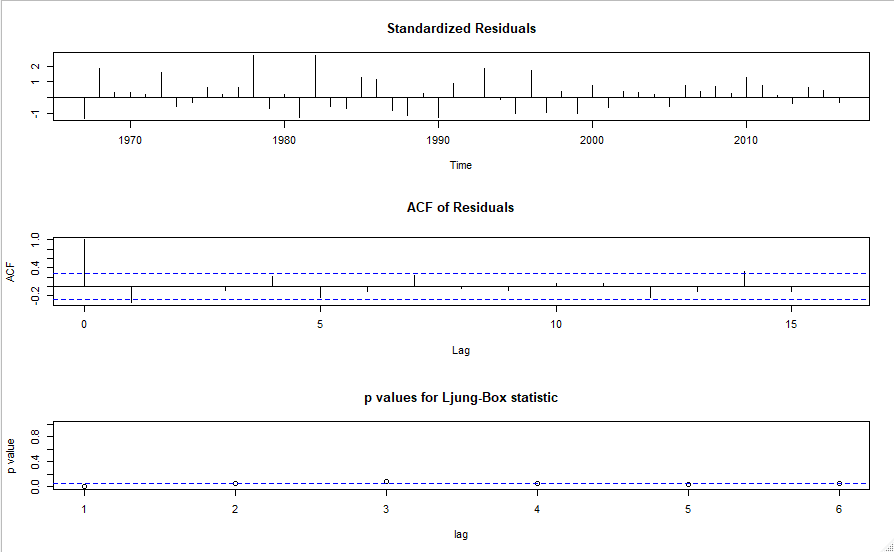
The next way we can narrow down our options is to make sure that the residuals are normal by looking at the qq plot and running a Shapiro-Wilk test, and to make sure that the residuals are independent through the runs test and the Ljung-Box test. We will also check the ACF of the residuals We will run each of these tests for our remaining 5 models, and move forward with those that pass.

Model 1:

AR(1)



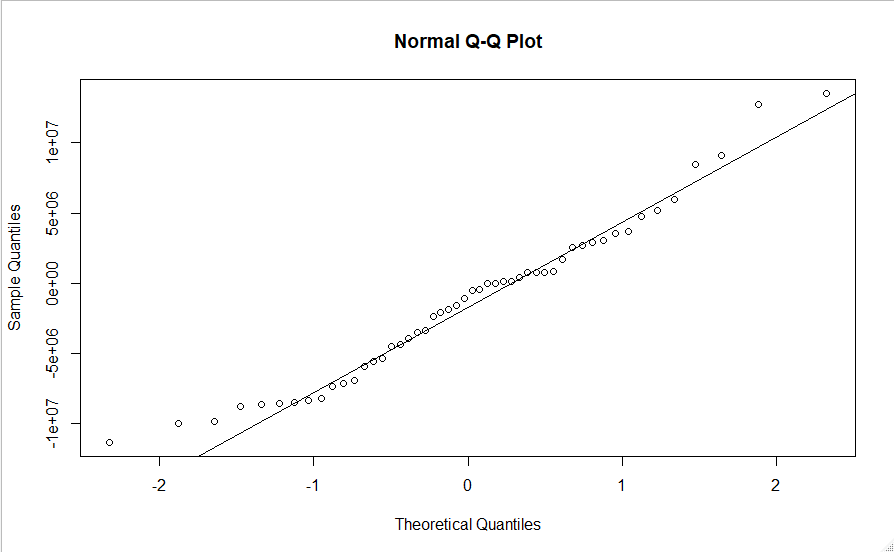


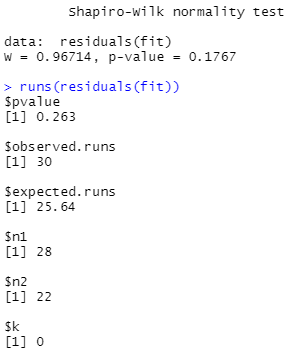


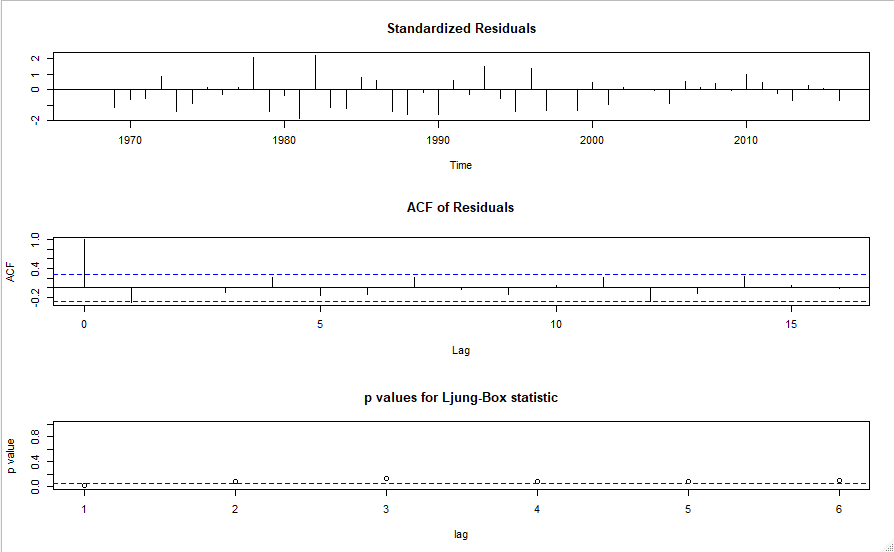
In the QQ-plot, we can see that most points follow the line, but at either end they tend to distance slightly. The p-value of the Shapiro-Wilk test is greater that 0.05, but only barely at 0.1097. We can say that the residuals are normal, but only barely. The residuals pass the runs test to say they are independent, but fail the Ljung-Box test, as several lags fall below the blue line, which is the test’s test statistic. The ACF of the residuals looks ok, but at lag 1 and lag 12, they are both slightly out of control. As the AR(1) model fails the Ljung-Box test, and does not really follow the QQ line, we can eliminate this as an option for our final model.

Model 2:

IMA(2,1)



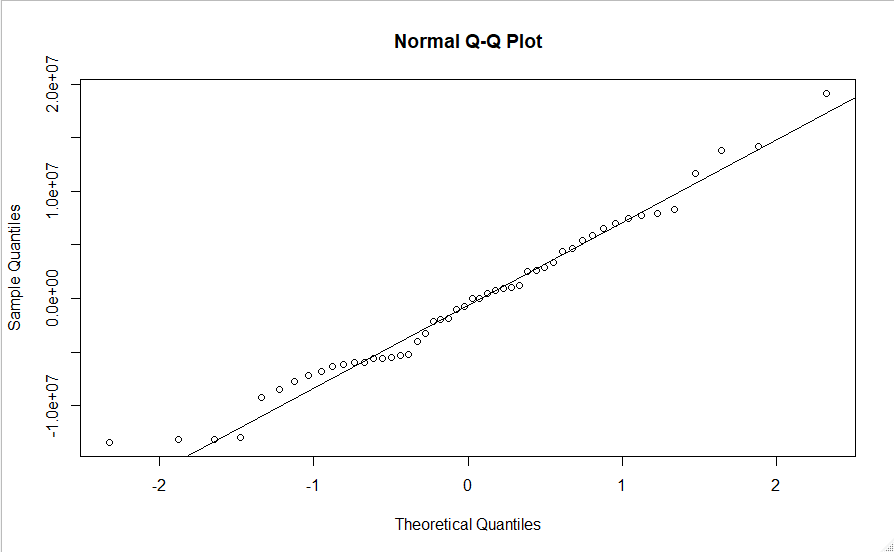


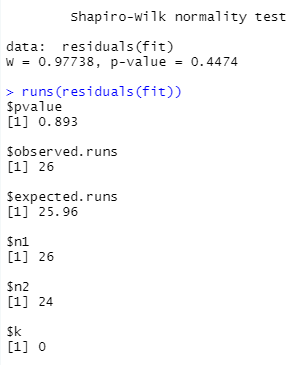


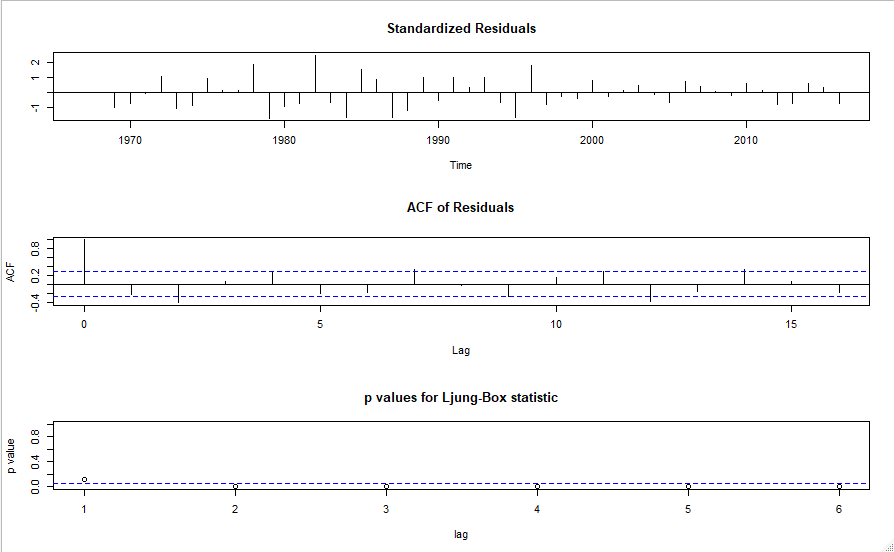
Again, we can see that the ends of the QQ-plot tend to trail off from the QQ-line they should be following. It is better, but not exactly where it should be. It again passes the Shapiro-Wilk test, better than it did for the AR(1) model. It passes the runs test as well, but not as well as the AR(1) model. It still fails the Ljung-Box test as well, as the first lag is below the level of significance. We can say that the residuals aren’t normal or independent, so we are also eliminating IMA(2,1) as well.

Model 11:

ARI(1,2)



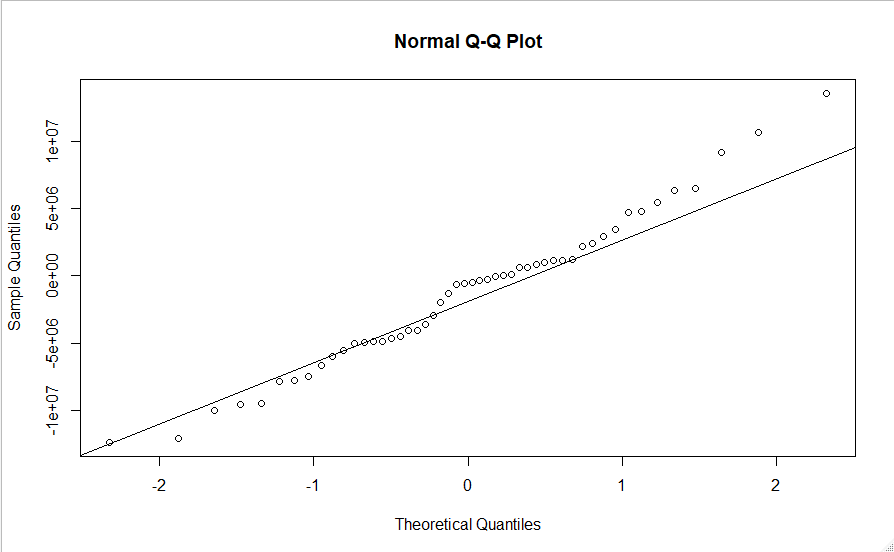


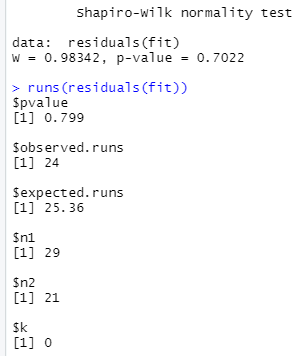


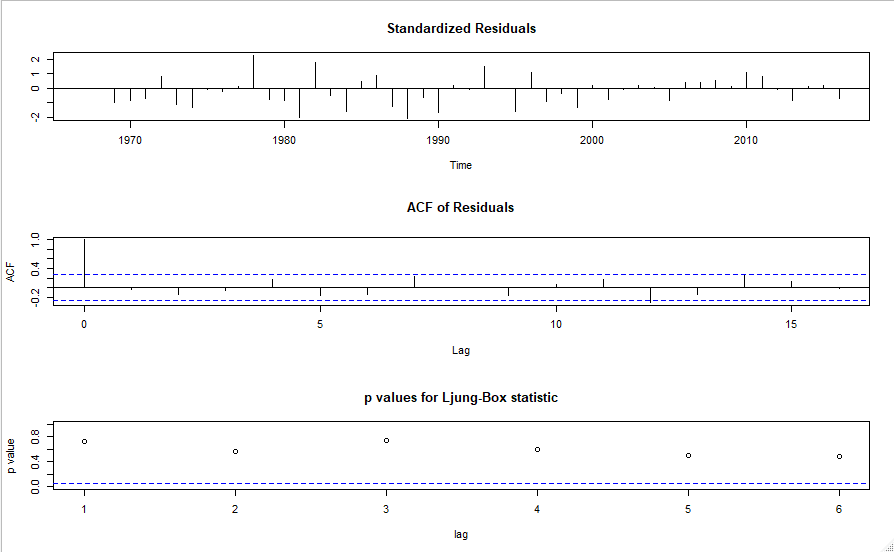
ARI(1,2) passes the QQ-plot, the Shapiro-Wilk test and the runs test the best of all of our models tried so far, but fails the Ljung-Box test. The ACF plot has lags 2 and 12 outside of the bounds, which fails the randomness of the correlations we seek. While the residuals may be normal, they do not seem to be random and independent, so we are eliminating ARI(1,2) as well.

Model 14:

ARIMA(1,2,1)



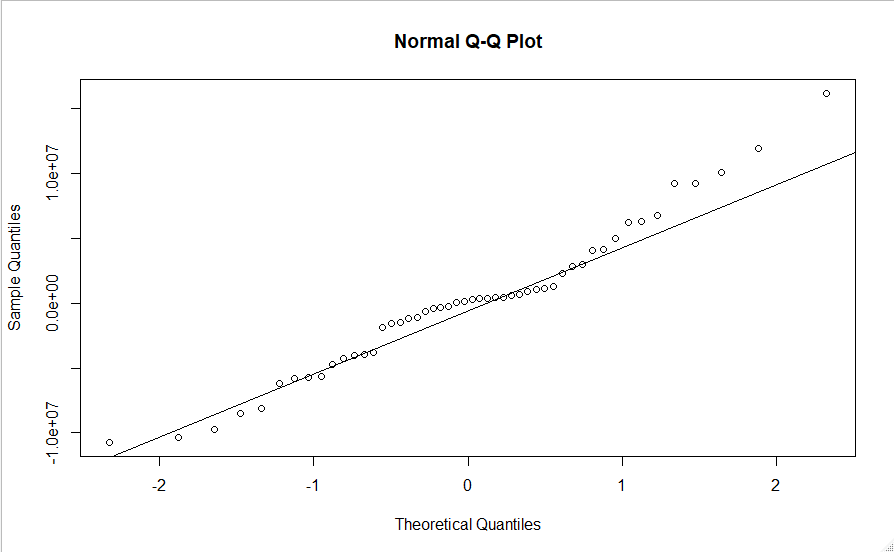


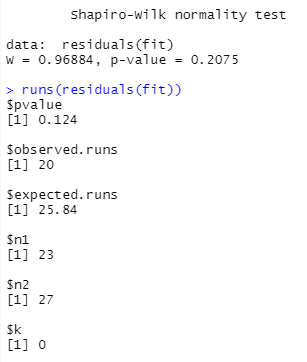


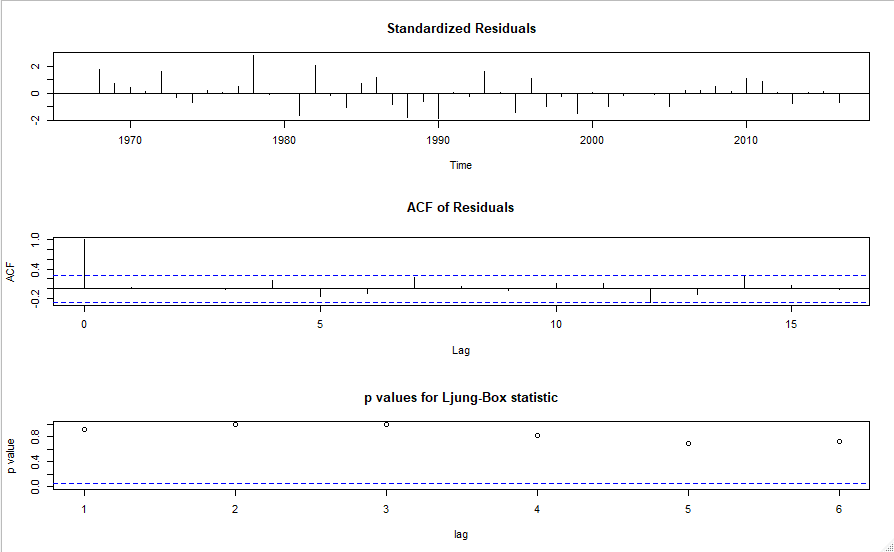
The QQ-plot trails off at the end very slightly, but passes all of the other three tests very well. From these tests, we can conclude that the residuals are independent and normal, so we will include ARIMA(1,2,1) as an option to consider moving forward.

Model 15:

IMA(1,1) with drift





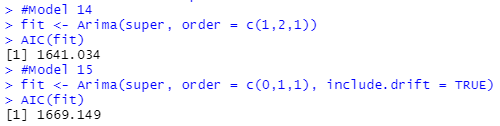


Again, this model passes all the test, though the QQ-plot is a little messy. The p-values for the Shapiro-Wilk and Runs test are not as high as well, but still higher than the significance level of 0.05. We can say that the residuals of this model are independent of each other and normal, so this model, IMA(1,1) with drift, is also a possible option.

From these 5 possible models, only 2 pass all of the tests we need, so ARIMA(1,2,1) and IMA(1,1) with drift are our final choices for a model.

**Model Selection**

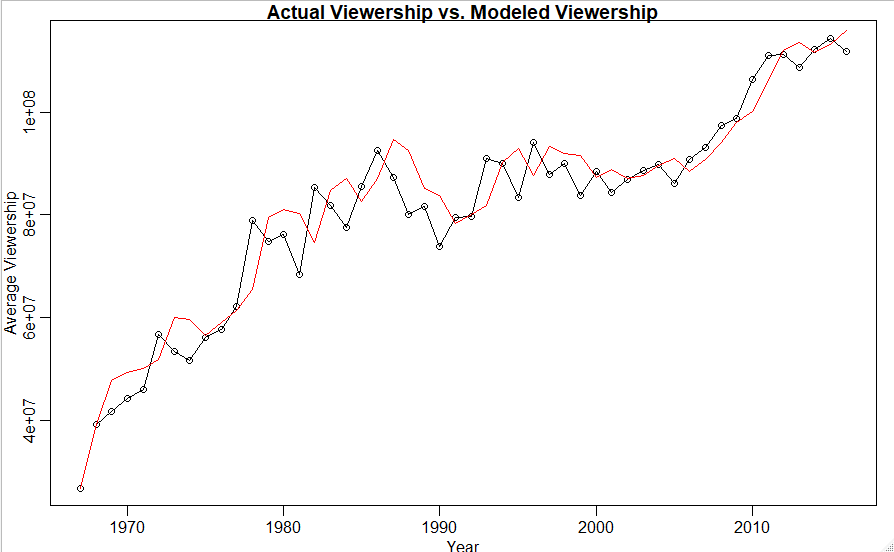
To select a final model from each of these final two, we will find the model with the lowest AIC, as the AIC is a statistic to determine the strength of a model to its original distribution.



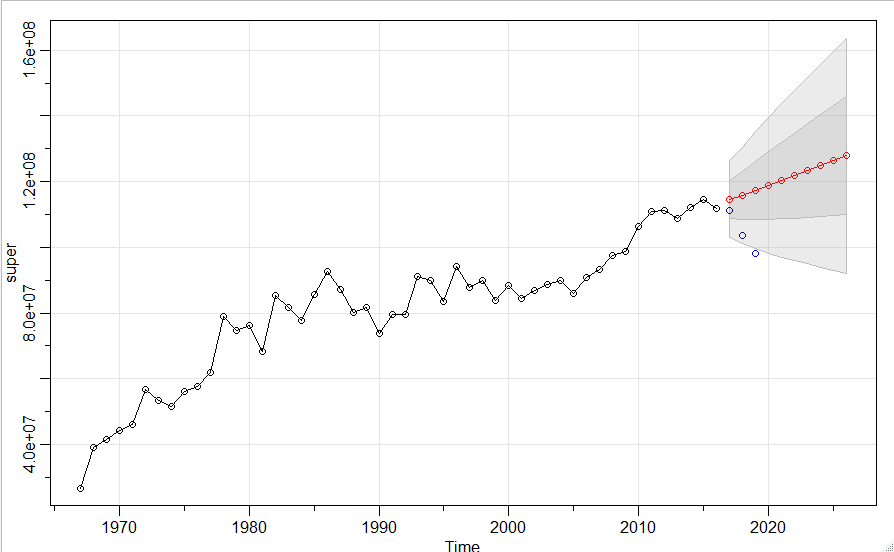
ARIMA(1,2,1) has the lowest AIC, so this is our final model we will use to forecast the future average viewership of the Superbowl.

**Forecasting**

Now that we have our model, we can use it to predict future Superbowl viewership. The three most recent years of the Superbowl, 2017-2019, were not included from our data, so we can compare our model with the actual values that were seen.



We can see that ARIMA(1,2,1) follows our actual values pretty well, which makes sense as it is trained to the data! Now, let’s look at the forecasts of our model compared with the actual results for the years that were not apart of our trained data.



**Conclusion**

From this process, we have found that ARIMA(1,2,1) can model the average viewership of the Superbowl. When we compare the actual values of 2017, 2018, and 2019 to the forecasted values, we can see that the viewership has declined much more than the model would have expected. 2017’s viewership is within one error bound of the forecasts, and 2018’s is within two, but 2019 falls out of the error bound after that.

Even though we can make a strong model to predict average viewership, it can not always account for everything. For some reason, Superbowl viewership has declined rapidly in the last three years. Additionally, another issue may be the fact that there has only been 53 Superbowl’s so far. Had we trained a model on viewership for 500 Superbowl’s, if there were that many, our model would be able to gather much more information, and we would likely have the forecasts be stronger. This is something to consider for the future.

Overall, viewership for the Superbowl has increased since its inception, and will likely continue to grow as time progresses.

**References:**

<https://www.rdocumentation.org/packages/astsa/versions/1.9/topics/sarima.for>

<https://en.wikipedia.org/wiki/Super_Bowl_television_ratings>

**Appendix:**

Raw R code:

library(readr)

library(readr)

library(readr)

library(fpp)

library(readr)

library(TSA)

library(imputeTS)

library(EnvStats)

library(astsa)

library(readxl)

Superbowl <- read\_excel("C:/Users/patty/Desktop/School/Senior/Semester 1/STAT 3504/Superbowl.xlsx",

range = "A3:C55", col\_names = FALSE)

Superbowl

this <- Superbowl[1:50,]

that<- Superbowl[51:53,]

super <- ts(this[,3], start = 1967)

forecasts <- ts(that[,3], start = 2017)

all <- ts(Superbowl[,3], start = 1967)

super

plot(super, type = 'o', ylab = 'Average Viewership', xlab = 'Year', main = 'Average Viewership of Superbowl over Years')

adf.test(super)

adf.test(diff(super))

adf.test(diff(diff(super)))

#Dicky-Fuller unit root test tells us that 2 order differencing will be the best, as it results

#in a stationary process for us.

acf2(super)

#ACF Decays to 0 and PACF spikes at 1 then goes to 0, suggests AR(1)

acf2(diff(super))

#Does not give much information.

acf2(diff(diff(super)))

#Possibly can suggest Arima(0,2,1)

eacf(super)

#Supports ARMA(1,1) or ARMA(2,1) process!

eacf(diff(super))

#Supports ARIMA(0,1,1), (0,1,2) or (0,1,3)

eacf(diff(diff(super)))

#Suggests Arima(0,2,1)

res = armasubsets(y=super, nar = 15, nma = 15)

plot(res)

#Suggests ARMA(1,5) or AR(3) or ARMA(3,5) or AR(1)

res = armasubsets(y=diff(super), nar = 15, nma = 15)

plot(res)

#Suggests ARIMA(3,1,0), ARIMA(3,1,5), ARIMA(4,1,0)

res = armasubsets(y=diff(diff(super)), nar = 15, nma = 15)

plot(res)

#Suggests Arima(1,2,0) or (1,2,2) or (3,2,2)

summary(auto.arima(super))

#Model 1

fit <- Arima(super, order = c(1,0,0))

summary(fit)

#Not overfit, Check!

#Model 2

fit <- Arima(super, order = c(0,2,1))

summary(fit)

#All good, check!

#Model 3

fit <- Arima(super, order = c(1,0,1))

summary(fit)

#Error, doesn't work

#Model 4

fit <- Arima(super, order = c(2,0,1))

summary(fit)

#Error, doesn't work

#Model 5

fit <- Arima(super, order = c(0,1,1))

summary(fit)

#overfit

#Model 6

fit <- Arima(super, order = c(1,1,1))

summary(fit)

#overfit

#Model 7

fit <- Arima(super, order = c(1,0,5))

summary(fit)

#Error, doesn't work

#Model 8

fit <- Arima(super, order = c(3,0,0))

summary(fit)

#Error, doesn't work

#Model 9

fit <- Arima(super, order = c(3,1,0))

summary(fit)

#overfit

#Model 10

fit <- Arima(super, order = c(3,1,5))

summary(fit)

#overfit

#Model 11

fit <- Arima(super, order = c(1,2,0))

summary(fit)

#Good!

#Model 12

fit <- Arima(super, order = c(1,2,2))

summary(fit)

#Overfit

#Model 13

fit <- Arima(super, order = c(3,1,2))

summary(fit)

#Overfit

#Model 14

fit <- Arima(super, order = c(1,2,1))

summary(fit)

#Good!

#Wanted to Try this because ARIMA(1,2,2) was only overfit on second MA term.

#Model 15

summary(auto.arima(super))

fit <- Arima(super, order = c(0,1,1), include.drift = TRUE)

summary(fit)

#1, 2, 11, 14, 15

#Model 1

fit <- Arima(super, order = c(1,0,0))

summary(fit)

qqnorm(residuals(fit))

qqline(residuals(fit))

shapiro.test(residuals(fit))

runs(residuals(fit))

quartz() # open a new graph window

tsdiag(fit,gof=6,omit.initial=F)

acf(residuals(fit))

#Passes all but the box test. Nope.

#Model 2

fit <- Arima(super, order = c(0,2,1))

summary(fit)

qqnorm(residuals(fit))

qqline(residuals(fit))

shapiro.test(residuals(fit))

runs(residuals(fit))

quartz() # open a new graph window

tsdiag(fit,gof=6,omit.initial=F)

acf(residuals(fit))

#Passes all but the box test. Nope.

#Model 11

fit <- Arima(super, order = c(1,2,0))

summary(fit)

qqnorm(residuals(fit))

qqline(residuals(fit))

shapiro.test(residuals(fit))

runs(residuals(fit))

quartz() # open a new graph window

tsdiag(fit,gof=6,omit.initial=F)

acf(residuals(fit))

#Passes all but the box test. Nope.

#Model 14

fit <- Arima(super, order = c(1,2,1))

summary(fit)

qqnorm(residuals(fit))

qqline(residuals(fit))

shapiro.test(residuals(fit))

runs(residuals(fit))

quartz() # open a new graph window

tsdiag(fit,gof=6,omit.initial=F)

acf(residuals(fit))

#Passes all tests!! Check!!

#Model 15

fit <- Arima(super, order = c(0,1,1), include.drift = TRUE)

summary(fit)

qqnorm(residuals(fit))

qqline(residuals(fit))

shapiro.test(residuals(fit))

runs(residuals(fit))

quartz() # open a new graph window

tsdiag(fit,gof=6,omit.initial=F)

acf(residuals(fit))

#Passes all tests!! Check!!

#Which is better? Compare AIC's

#Model 14

fit <- Arima(super, order = c(1,2,1))

AIC(fit)

#Model 15

fit <- Arima(super, order = c(0,1,1), include.drift = TRUE)

AIC(fit)

#Model 15 has the lowest AIC, so is therefore the strongest model!

fit <- Arima(super, order = c(1,2,1))

plot(super, type = 'o', xlab = 'Year', ylab = 'Average Viewership', main = 'Actual Viewership vs. Modeled Viewership')

lines(as.vector(time(super)), fitted(fit), col ='red')

forecast <- sarima.for(super, 10, 1, 2, 1)

points(forecasts, col = 'blue')