Project 1

ARIMA Model for the VXX

## Abstract

The markets of today are seemingly far more complex than they were even twenty years ago. Not only are there the usual factors of supply and demand but there is also now high frequency trading, front running orders, and market churn from liquidity providers. In this project we will analyze tick data for the VXX, identify the most accurate ARIMA model, and attempt to predict the coming price range for the VXX in the near future. We will find that the best model for daily data is an ARIMA(2, 1, 2) with drift and that the price movement in our forecast period may indicate a structural change in the VXX.

## Introduction

The VXX is an Exchange Traded Product (ETP) that attempts to mimic the CBOE’s VIX index. In essence the VXX is a tradeable product that moves, as similarly as possible, like the CBOE’s measure of volatility. The VXX was introduced in 2009 by Barclays PLC, a British bank, and is not backed by any real assets only by the banks credit.

In 2017, the VXX traded on average around $2 billion a day. Historically, the VXX has a downward bias due to the way the ETP was setup to mimic the VIX movement. On 2/5/18 and 2/6/18 the VXX underwent a very sharp upward move, which is very unlike its historical price movement. In this project we want to see how well an ARIMA model would have been able to predict such a large movement.

Our data consists of every trade that occurred for the VXX from the dates 2/24/17 to 2/13/18, almost an entire trading year. The data set contains the Date/Time, Last, Ask, Bid, and Volume of each trade. Due to our data set containing 36,131,279 rows our processing time was immense. Depending on the task it would take anywhere from 30 minutes to several hours.

We will investigate the ability to build and utilize an ARIMA model on our data starting with the raw tick data and using the tick data to build daily bars consisting of the day’s Open, High, Low, and Close. Figure 1 below is a chart of our entire tick data set:

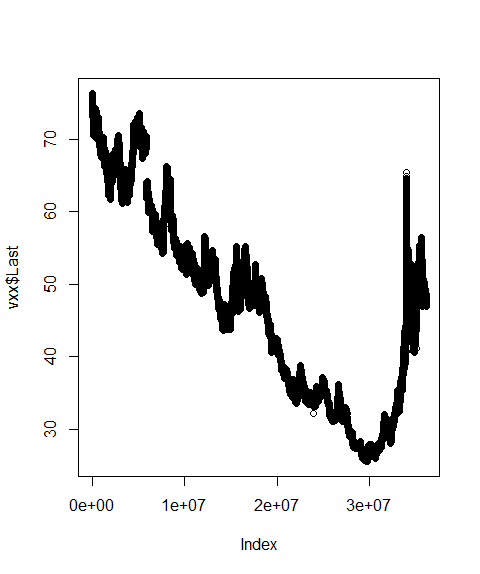


Figure 1. VXX Tick Chart.

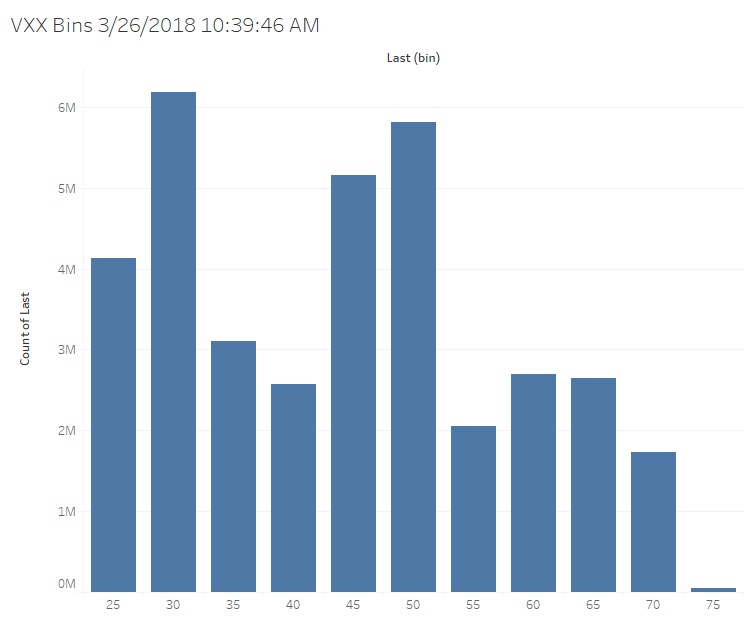


Figure 2. Tableau - VXX Price Distribution.

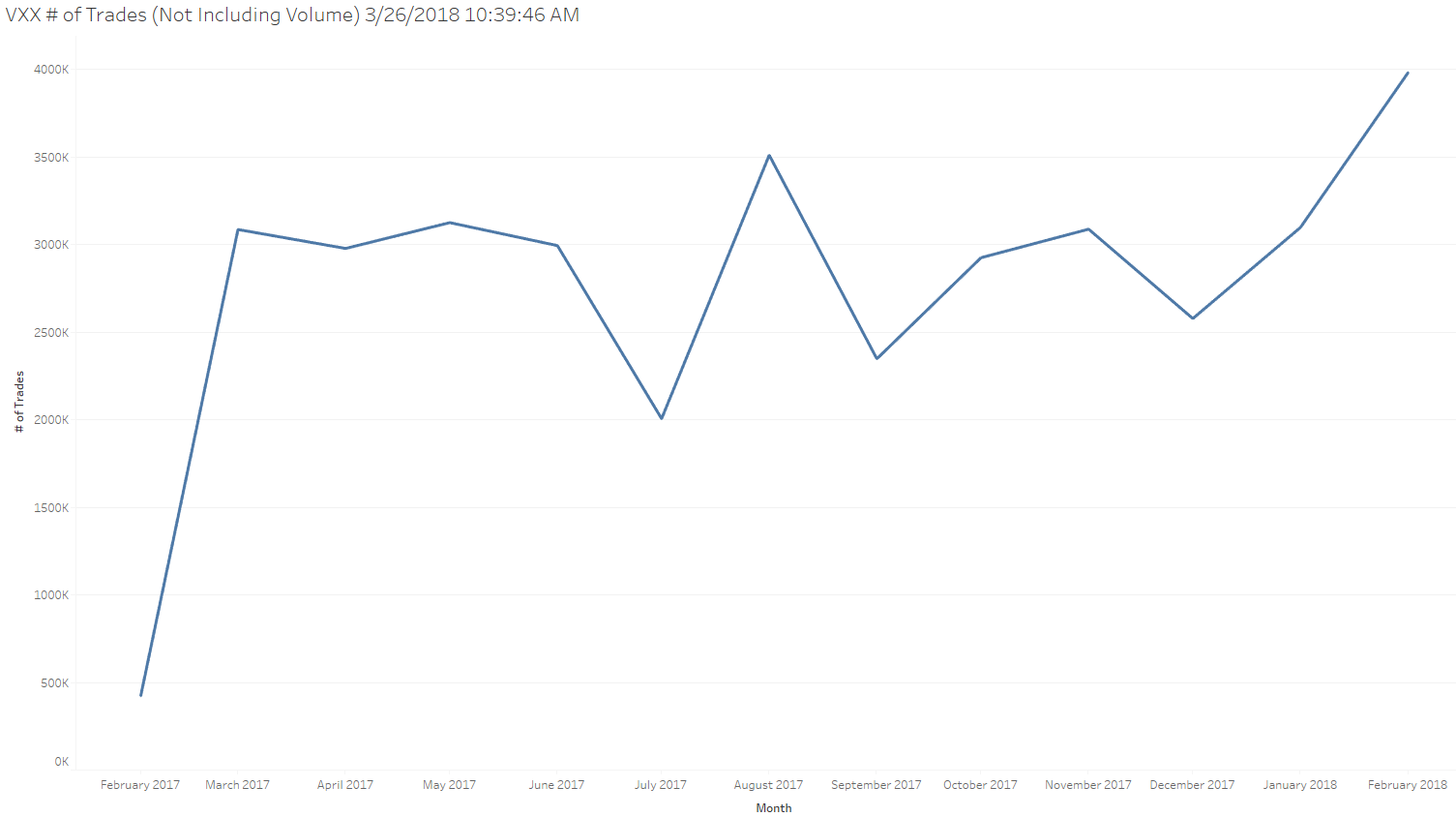


Figure 3. Tableau - Number of Trades.

## Methods

Upon importing our data set to R, we discovered that the tick data was not reverse-adjusted. The VXX has undergone several reverse-splits in order to keep the price of it high and tradeable. Due to the downward bias of the VXX, if no reverse-splits occurred then it would go to zero. In order to resolve the data issue we simply isolated and adjusted the necessary rows like so:

# Find where the first reverse split corrected value is

match("2017-08-23",vxx$Day)

vxx[16991649,]

# Check the previous row to ensure it is where we want to end our adjustment

vxx[16991648,]

# Visualizing the price jump caused by the reverse split

plot(vxx$Last[16991637:16991657])

# We need to correct for the 4:1 split that occurred on 8/23/17

# To do this we will multiply the price by 4 for the rows 1:16991287

vxx$Last[c(1:16991648)] <- vxx$Last[c(1:16991648)]\*4

## 

## After we successfully imported and munged our tick data we went about to create the daily bars:

# Now lets build some bar data Open,High,Low,Close for our vxxDaily dataset

last <- xts(vxx$Last, vxx$Date)

last2 <- to.daily(last)

last3 <- data.frame(date=index(last2), coredata(last2))

names(last3) <- c("Day","Open","High","Low","Close")

last3$Day <- format(last3$Day, "%Y-%m-%d")

# Merge our OHLC data with vxxDaily based on the Day columns

vxxDaily <- last3

# Convert our date from a character to a POSIX format

vxxDaily$Day <- as.POSIXct(vxxDaily$Day)

## 

Figure 4 is a plot of the closing price from the daily bars we generated:

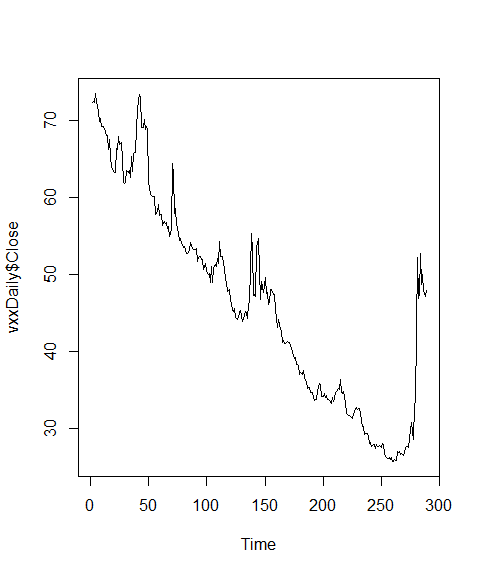


Figure 4. Closing Price of Daily Bars.

After creating our daily data, we created a QQ plot. We discovered that the price data needed to be log adjusted due to its non-normal distribution. Figure 5 below shows the QQ plot for the unadjusted daily data on the left and the QQ plot for the log adjusted data on the right. We can see that the data is now normally distributed from the initial large tails at each end.

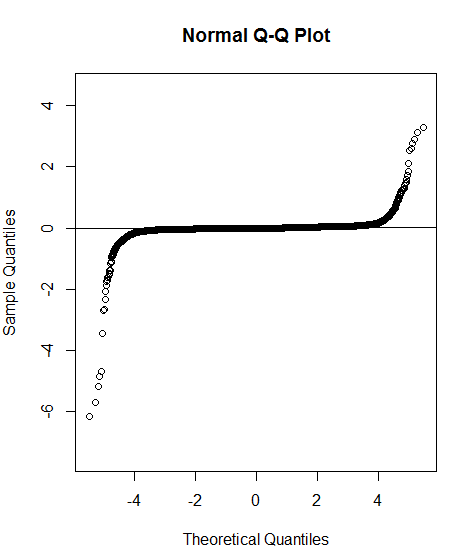
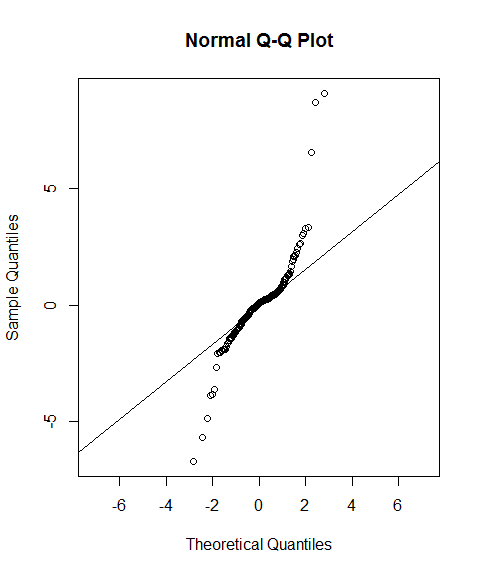


Figure 5. Closing Price of Daily Bars for Training Data.

In order to assess the models we create for the data correctly we set the cut-off date for the training set to be 2017-10-28. Figure 6 shows us how the training data does not include the large spike upwards at the end of our data.

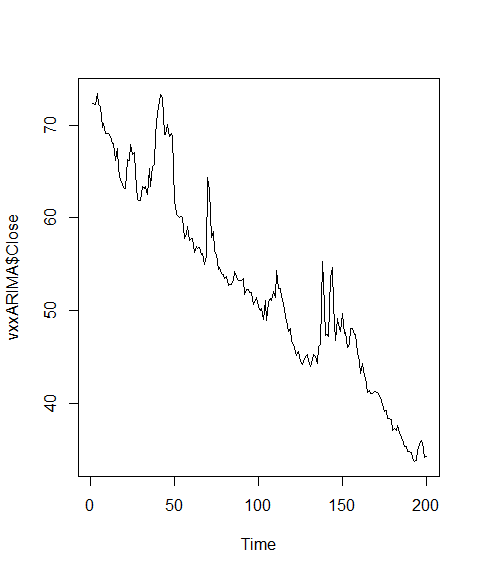


Figure 6. Closing Price of Daily Bars for Training Data.

Now that we have split out our training and testing data, we move onto creating and analyzing the best model. Using the methods that Professor Maybin [2] provided and instructed us on, we were able to identify the best model using the auto.arima function in the R package “forecast”. Here is the code we utilized for creating the ARIMA models for our dataset:

arima1 <- auto.arima(data.train.vxxClose, trace=TRUE, seasonal = TRUE, stepwise = FALSE, approximation = FALSE)

BestModel1 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="aic")

BestModel2 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="bic")

BestModel3 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="aicc")

BestModel4 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="aic")

BestModel5 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="bic")

BestModel6 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="aicc")

BestModel7 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="aic")

BestModel8 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="bic")

BestModel9 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="aicc")

Once we created the models we compared them using the Mean Absolute Percentage Error (MAPE) measure of accuracy. We then selected the top three models based on their MAPE, the lower the better. From here, the top three models were plotted with their forecasts and the actual values to see which ones fit our data the best.

## Results

For our data set we identified the best ARIMA model to be an ARIMA(2,1,2) with drift. Figure 7 is a plot of the ARIMA(2, 1, 2) with drift model, the 85% and 95% predictions and the actual values (dotted line).

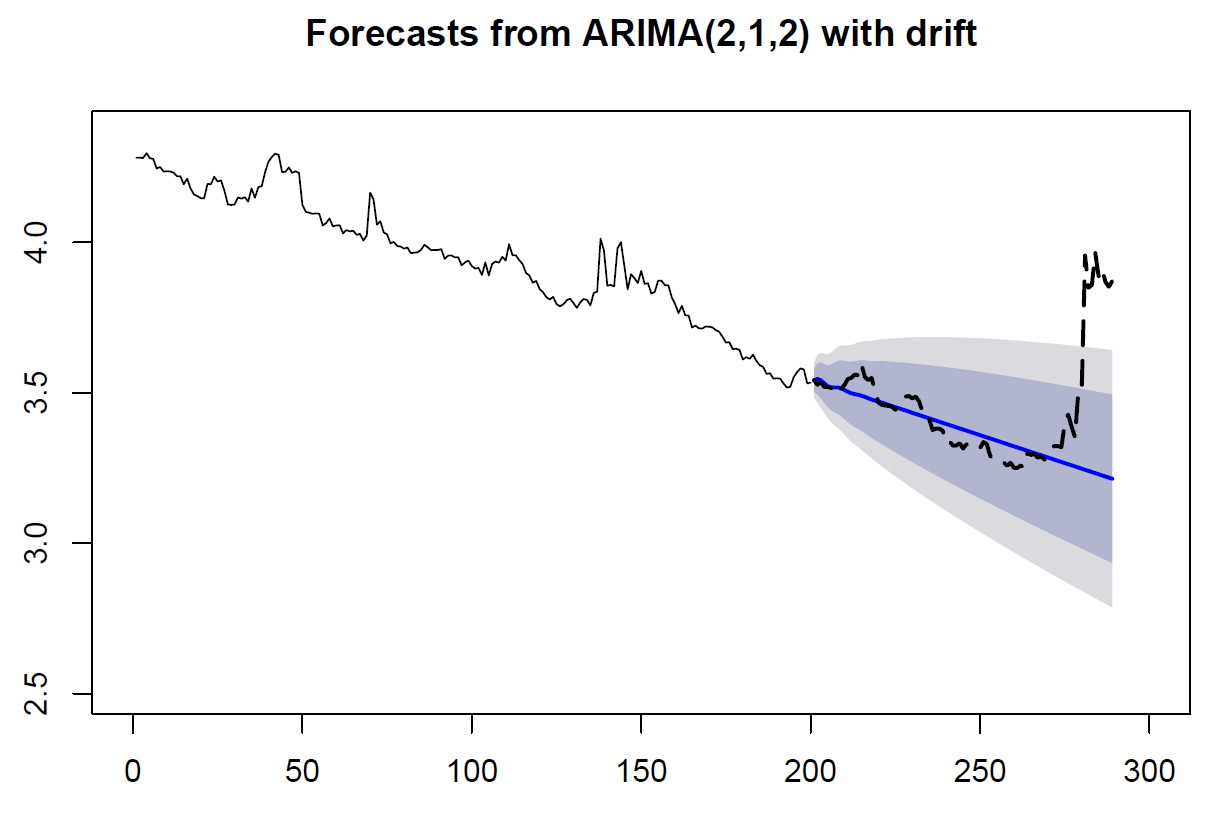


Figure 7. Daily ARIMA(2, 1, 2) with drift model including predictions and actual values.

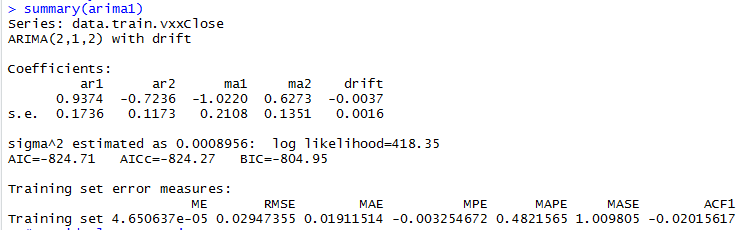


Figure 8. Daily ARIMA(2, 1, 2) with drift model including predictions and actual values.

The above Figure 8 shows the accuracy of the ARIMA(2, 1, 2) with drift model and we can see that the MAPE = 0.4821565. We also check the coefficients of the AR and MA terms by summing them up which yields us 0.2138 and -0.3947 respectively. The coefficients of the AR and MA terms do not add up to 1 which means that our model is valid from the perspective of being better than a random model that happened to guess correctly at the right time. Now we run the diagnostics for the ARIMA(2, 1, 2) with drift model, including the Standardized Residuals, ACF of Residuals, and p-value for Ljung-Box statistic.

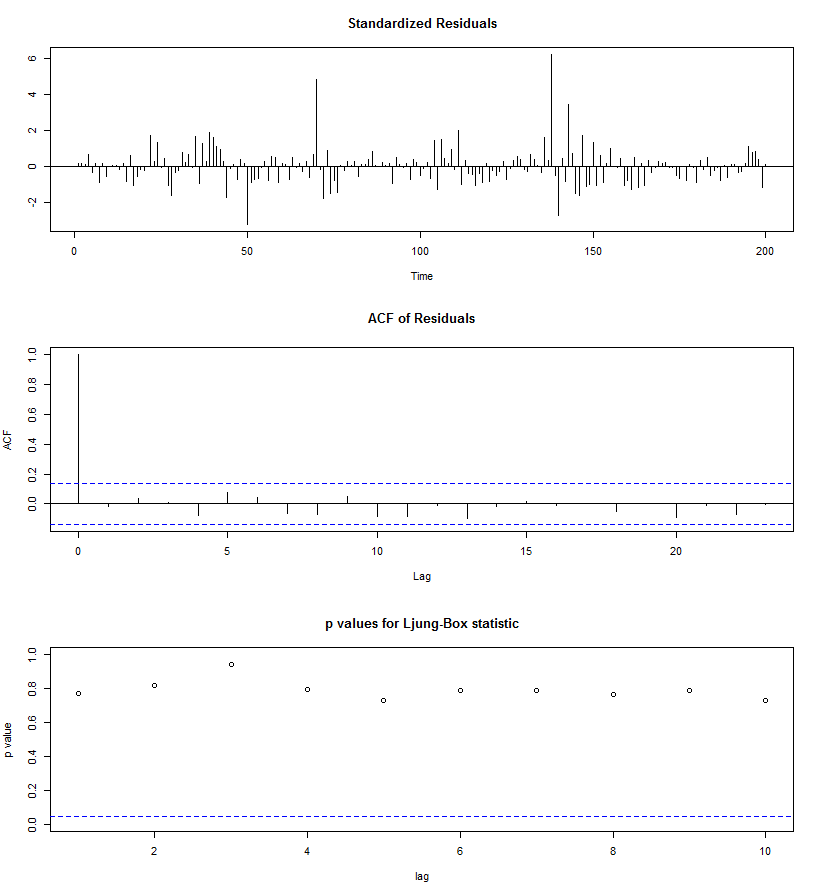


Figure 9. Standardized Residuals, ACF of Residuals, and Ljung-Box statistic.

## From Figure 9, we can see that the Standardized show that we have several outliers, around 6 of them. These outliers are not an issue and come from the major spikes in price movement. The residuals show no sign of non-linearity or heteroscedasticity. The ACF of residuals shows us a value of 1 for the lag 0 which means that our model needs an AR for the first term, which it does. The p-values for the Ljung-Box statistic are all > 0.05 which means that for the first 25 lags we reject the alternative hypothesis that the residuals are dependent.

## To compare what the selected best model looks like we compare it with the next best models based on their MAPE values. The second best model is the ARIMA(0, 1, 0) model which can be seen in Figure 10 and the third best model is the ARIMA (0, 0, 5) with non-zero mean in Figure 11. Both models visually look worse than our selected best model. The ARIMA(0, 1, 0) is accurate in capturing a horizontal mean and the future price movement. However, the mean is not downward sloping and does not capture the previous bias. The ARIMA(0, 0, 5) with non-zero mean is only able to predict the very end of our testing data and does not capture the downward bias. Both models are a great example of why just using a measure of accuracy does not ensure that the model is a good fit for the data.

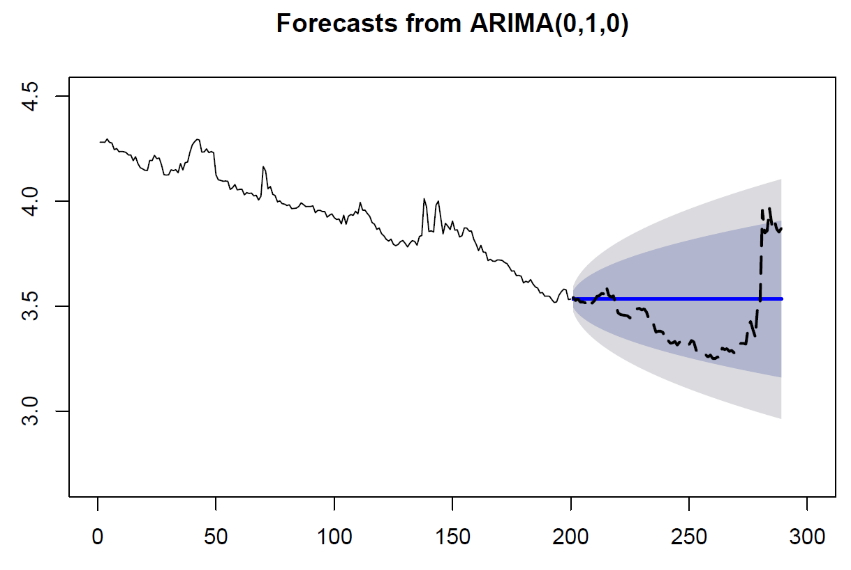


Figure 10. Daily ARIMA(0, 1, 0) model.

## 

Figure 11. Daily ARIMA(0, 0, 5) with non-zero mean model.

## Conclusion

In conclusion, we find that the best model for our daily data set that we generated from 36 million rows of tick data was an ARIMA(2, 1, 2) with drift model. The two next best models were visually lacking fit as the ARIMA(0, 1, 0) model did not capture the VXX downward bias while the ARIMA(0, 0, 5) with non-zero mean model failed to predict most future values. While the ARIMA(2, 1, 2) with drift model captured the future values within its forecast the latest price movement of the VXX may actually be a structural change in how the VXX price moves.

## Appendix

### 

### Reference

1. Daróczi Gergely (2013). Introduction to R for Quantitative Finance. Packt Publishing.
2. Professor Chad Maybin (2018). Southern Methodist University.

### 

### Complete Code Base

##########################################################

## Clean Up, only use on first time code run

##########################################################

#rm(list = ls())

#gc()

#par(mfrow=c(1,1))

##########################################################

## Read in VXX Tick Data (2/24/17 until 2/13/18)

##########################################################

setwd("C:/Users/jbeer/Documents/R/VXX")

library(data.table)

require(data.table)

library(zoo)

library(dplyr)

library(tidyr)

library(xts)

library(forecast)

library(tseries)

library(TSPred)

library(changepoint)

# Read in our tick data

vxx <- fread("VXX.Last.txt", sep=";")

# Check the data

head(vxx)

# Remove unwanted columns

vxx <- vxx[,-c(3:5)]

# Re-check we removed correctly

head(vxx)

# Label the columns

names(vxx) <- c("Date", "Last")

# For extremely fast writing out use fwrite(vxx, "vxx.csv")

##########################################################

## Munge VXX Tick Data

##########################################################

# Need to massage the Date column into a Date format that will hold the milliseconds

vxx2 <- vxx

vxx2 <- vxx2 %>% separate(Date, c("Date", "2", "3"), " ")

vxx2$`2` <- paste(vxx2$`2`, vxx2$`3`, sep=".")

vxx2$`3` <- NULL

vxx2$Date <- paste(vxx2$Date, vxx2$`2`, sep=" ")

vxx2$`2` <- NULL

vxx2$Date <- as.POSIXct(vxx2$Date, tz = "UTC","%Y%m%d %H%M%OS")

# Write it back to our original DF

vxx <- vxx2

# Clean up

rm(vxx2)

gc()

##########################################################

## Perform Check, Create Day Column, Fix Reverse Split

##########################################################

# Ensure there are no Null values

colSums(is.na(vxx))

# Final check of DF

str(vxx)

summary(vxx)

# Create a Day column for grouping by later

vxx$Day <- format(vxx$Date, "%Y-%m-%d")

# Find where the first reverse split corrected value is

match("2017-08-23",vxx$Day)

vxx[16991649,]

# Check the previous row to ensure it is where we want to end our adjustment

vxx[16991648,]

# Visualizing the price jump caused by the reverse split

plot(vxx$Last[16991637:16991657])

# We need to correct for the 4:1 split that occurred on 8/23/17

# To do this we will multiply the price by 4 for the rows 1:16991287

vxx$Last[c(1:16991648)] <- vxx$Last[c(1:16991648)]\*4

# Check to ensure it changed correctly

plot(vxx$Last[16991637:16991657])

##########################################################

## Create Daily Bars with Open, High, Low, Close Values

##########################################################

# Create a new data frame that will store our 260 daily bars.

# Create a Day column in new DF from unique vxx$Day values.

vxxDaily <- data.frame(matrix(ncol = 1, nrow = nrow(vxx)))

names(vxxDaily) <- c("Day")

# Now lets build some bar data Open,High,Low,Close for our vxxDaily dataset

last <- xts(vxx$Last, vxx$Date)

last2 <- to.daily(last)

last3 <- data.frame(date=index(last2), coredata(last2))

names(last3) <- c("Day","Open","High","Low","Close")

last3$Day <- format(last3$Day, "%Y-%m-%d")

# Merge our OHLC data with vxxDaily based on the Day columns

vxxDaily <- last3

# Convert our date from a character to a POSIX format

vxxDaily$Day <- as.POSIXct(vxxDaily$Day)

# Ensure there are no Null values

colSums(is.na(vxxDaily))

# Final check of DF

str(vxxDaily)

# Remove unwanted data

vxx$Day <- NULL

rm("last3","last2","last")

gc()

# Let's plot our data, we will see an issue with price near the midpoint of our data set. It jumps from around 14 to 50.

plot(vxxDaily$Close)

plot.ts(vxxDaily$Close)

##########################################################

# Splitting our Training and Testing data: vxxClose

##########################################################

# Isolate just the closing price

vxxClose <- vxxDaily[,5]

# How many days of data do we have?

length(vxxClose)

# Check with a plot to ensure we have what we want.

plot.ts(vxxClose)

# Log Transform our data since the QQ plots of the ARIMA model residuals are non-normal (heavy tailed)

vxxClose <- log(vxxClose)

# Create our time series object

ts.vxxClose <- ts(vxxClose, frequency=1, start=1)

#Split data into training and testing data set according to using the first 200 obs as our training and the last 89 for prediction

data.train.vxxClose <- window(ts.vxxClose, start=1, end=200)

data.test.vxxClose <- window(ts.vxxClose, start=201, end=289)

# Make sure we have 289 observations still

length(data.test.vxxClose)+length(data.train.vxxClose)

gc()

##########################################################

# ARIMA Model Creation: vxxClose

##########################################################

# Per the documentation of the "forecast" package, if we are analyzing only one time series and can afford the

# computations, we should set stepwise = FALSE and approximation = FALSE, which we will do here.

# Best model is an ARIMA(2,1,2) with drift

arima1 <- auto.arima(data.train.vxxClose, trace=TRUE, seasonal = TRUE, stepwise = FALSE, approximation = FALSE)

BestModel1 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="aic")

BestModel2 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="bic")

BestModel3 <- auto.arima(data.train.vxxClose, trace=TRUE, test="kpss", ic="aicc")

BestModel4 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="aic")

BestModel5 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="bic")

BestModel6 <- auto.arima(data.train.vxxClose, trace=TRUE, test="adf", ic="aicc")

BestModel7 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="aic")

BestModel8 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="bic")

BestModel9 <- auto.arima(data.train.vxxClose, trace=TRUE, test="pp", ic="aicc")

gc()

modList <- list(BestModel1, BestModel2, BestModel3, BestModel4, BestModel5, BestModel6, BestModel7, BestModel8, BestModel9)

##########################################################

# Forecasting & Prediction: vxxClose

##########################################################

# We used the first 200 days to train our ARIMA models, we have 89 days left we can use to forecast and compare with.

# Use the last 89 days to test on

# ARIMA1

# Forecasting

arima1.forecast <- forecast(arima1, h=89)

# Prediction

plotarimapred(data.test.vxxClose, arima1, xlim=c(0,300), range.percent = 0.1)

accuracy(arima1.forecast, data.test.vxxClose)

# Export a forecast for each of these best models.

for (Amodel in modList){

i=0

arima1.forecast <- forecast(Amodel, h=89) #forecast 24 periods ahead

arima1.forecast

plot(arima1.forecast, xlab="Date", ylab="VXX Price")

library(TSPred)

plotarimapred(data.test.vxxClose, Amodel, xlim=c(0,300), range.percent = 0.1)

accuracy(arima1.forecast, data.test.vxxClose)

#plot as confirmation

pdf( paste0("Plot - ", Amodel,format(Sys.time(), "%a %b %d %H %M %S %Y"), " .pdf"),width=7,height=5)

plotarimapred(data.test.vxxClose, Amodel, xlim=c(0,300), range.percent = 0.1)

accuracy(arima1.forecast, data.test.vxxClose)

dev.off()

}

gc()

##########################################################

# Model Diagnotics: vxxClose

##########################################################

# Our model looks good since the standardized residuals don't show volatility clusters, no significant autocorrelations between the residuals

# according to the ACF plot, and the Ljung-Box test for autocorrelation shows high p-values, so the null hypothesis of independent residuals cannot be rejected.

# To assess how well the model represents the data in the sample, we can plot the raw monthly returns (the thin black solid line) versus the fitted values

# (the thick red dotted line).

# ARIMA1 plot of time series and fitted model values

plot(arima1$x, lty = 1, main = "VXX raw data vs. fitted values", ylab = "VXX Price", xlab = "Date")

lines(fitted(arima1), lty = 2,lwd = 2, col = "red")

# Confidence Interval

confint(arima1)

# Summary

summary(arima1)

# Residuals, ACF, Ljung-Box

tsdiag(arima1)

# if p-value >0.05 then the residuals have a normal distibution (not reject null Hypothesis of normality)

# We see that the p-value < 2.2e-16 therefore we reject the null hypo that we have normally distributed residuals.

jarque.bera.test(arima1$residuals)

# Plot our residuals so we can see the distribution.

qqnorm(arima1$residuals)

qqline(arima1$residuals)

# Plot the distribution of our residuals

plot(density(arima1$residuals))

# Calculate Accuracy Measures

accuracy(arima1)

# Residual Diagnostics

plot.ts(arima1$residuals)

# From the Box-Ljung test we see that the p-value = 0.3046 therefore fail to reject null that the data are independent (i.e. no autocorrelation exists)

Box.test(arima1$residuals, lag=20, type="Ljung-Box")

# We see that the lag 6 barely crosses the boundary but no other lags do therefore no autocorrelation.

acf(arima1$residuals, lag.max = 24, main="ACF of the Model")

# Test for GARCH effect, p-value = 0.1163 therefore no ARCH effect present and GARCH model shouldn't be considered.

Box.test(arima1$residuals^2, lag = 20, type = "Ljung-Box")