Stock Prediction and Time Series Model Comparisons

Joe Cook, Kaitlin Kirasich, Damon Resnick

April 21, 2018

1 Introduction

In this paper, we analyzed a portfolio of diverse securities. The team fit, back-tested, and forecasted the returns of the portfolio with ARIMA and GARCH models. Additionally, we used an efficient frontier technique to optimize our portfolio to get the best expected returns for our level of risk. We compared the root squared mean error (RMSE) of the forecasts from both models of our portfolio over the last 5 days of returns to analyze which model performed better.

2 Data Selection & Models

First, we selected the securities for our portfolio. For our data source, we used the R library "quantmod" to pull daily returns with a start date of April 19, 2017 and an end date of April 19, 2018 [1]. We then ran a log transformation on the returns and chose 3 companies from 6 different industries to find an optimal combination of one company per industry to minimize risk and maximize reward. The final stocks we chose for our initial portfolio were: Nike, Disney, Coca Cola, Johnson & Johnson, Amazon, and Goldman Sachs. We also included SPY to balance out our portfolio because of its diverse nature.

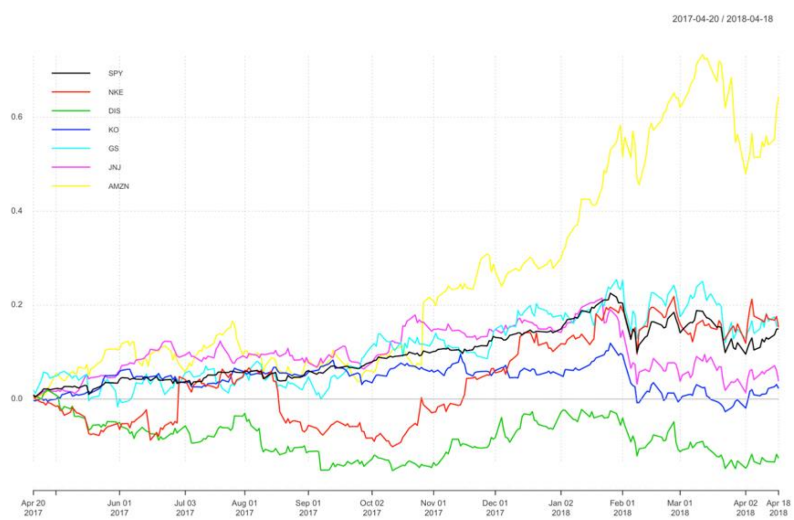


Figure 1. The daily returns for the first 7 chosen securities over the past year.

Next, we fit, back-tested, and forecasted the portfolio with an ARIMA model [2, 3]. We found that for most stocks, such as AMZN, KO, SPY, NKE, DIS, this visually did not look like it had much of a time series component. The R function *auto.arima()* chose ARIMA(0,0,0) with zero mean model for each of the securities. For the purposes of this project, we went with either an ARIMA(1,0,1) or ARIMA(2,0,2) based on an optimal forecasting (lowest RMSE).

As seen in Figure 3, our most efficient securities were AMZN, SPY, and KO. AMZN was the most efficient security lying directly on the efficiency frontier while SPY and KO tied in second place lying closely below the curve.

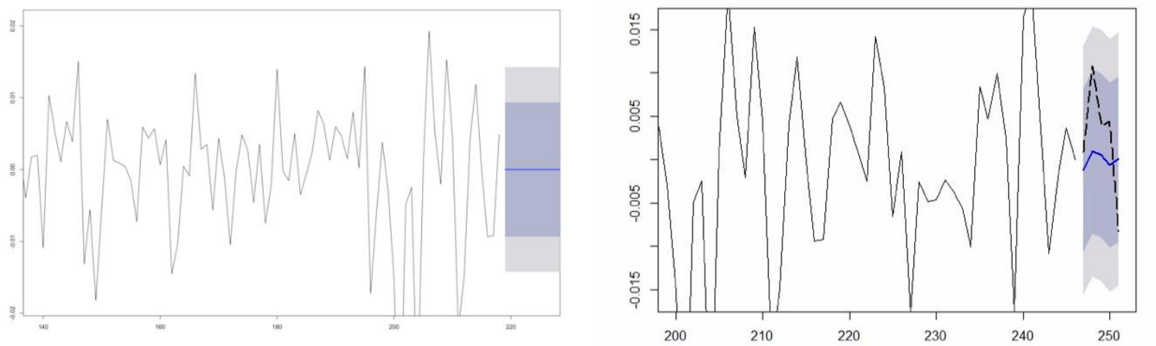


Figure 2. The daily returns of Coca-Cola model with two different ARIMA models. The right graph shows the ARIMA (2,0,2) predictions for the next 5 days while the left graph shows the ARIMA (0,0,0) with zero mean prediction.

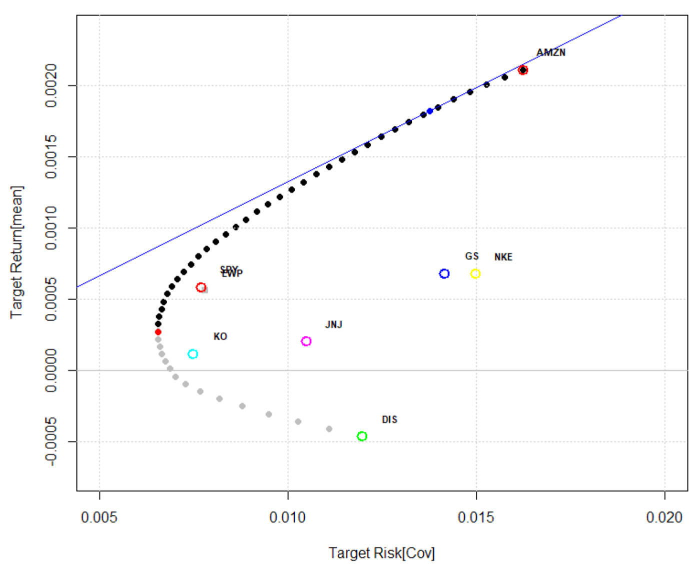
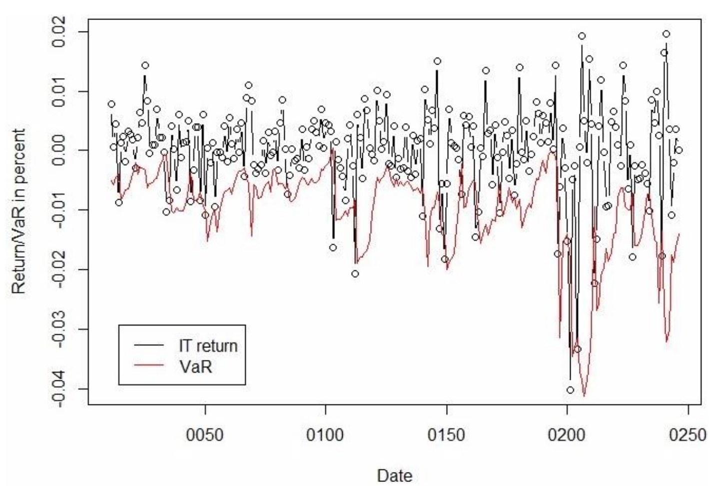


Figure 3. The Tailored Efficient Frontier curve for the first 7 securities selected. This graph was used to help determine the best stocks to include in our portfolio.

Next, we fit, back-tested, and forecasted a GARCH model [4]. The GARCH model produced the correct exceedance of volatility. With a p-value of 0.074 on the unconditional coverage test and a p-value of 0.083 on the conditional coverage test. To further test the model we did additional back testing to confirm the results.



In the back testing of the GARCH model we see that our model is conservative with the variation in returns it perceives with what will happen with the stock. The model does a successful job of simulating the returns in the months selected for back-testing even if it is pessimistic. The back-testing does show us that the model can predict the past returns with some certainty.

Figure 4. Value-at-Risk Back-testing graph using a GARCH model. The VaR and returns are plots back a year. *There is considerable overlap.*

3 Results

Finally, we optimized our portfolio. From Figure 3 above, we can see that JNJ, GS, and NKE are not very efficient. For those reasons, we first removed the three securities from our portfolio and compared the efficiency frontiers. This made it slightly worse. Instead, we swapped the low efficiency securities out for other companies in their industries to see if that would improve our efficiency frontier. We replaced Johnson & Johnson with Estee Lauder, Goldman Sachs with Berkshire-B, and Nike with Under Armour. The results are shown in blue below and drastically increase the returns while minimizing risk.

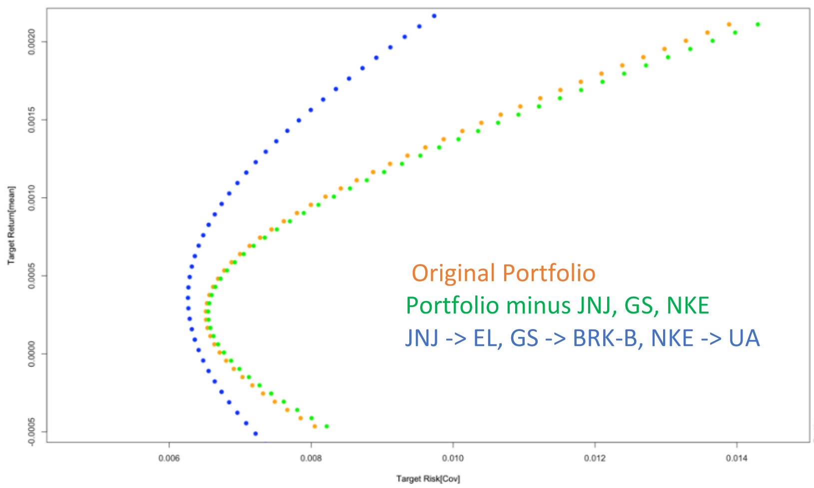


Figure 5. The Efficient Frontier curve for the 3 portfolios we looked at. The orange is the original portfolio. The green is the original with JNJ, GS, and NKE removed. And the blue is the new portfolio with EL replacing JNJ, BRK-B replacing GS, and UA replacing NKE.

We also compared the RMSE's for the two different models on the chosen stocks. Not surprisingly the GARCH model outperformed the ARIMA in all but 2 cases. Table 1 shows the RMSE values for each stock using the two models.

RMSE	SPY	UA	DIS	BRKB	EL	KO	AMZN
ARIMA	0.0073	0.0320	0.0117	0.0104	0.0118	0.0073	0.0160
GARCH	0.0065	0.0224	0.0098	0.0074	0.0127	0.0066	0.0211

Table 1: RMSE for each stock in portfolio for each model and the portfolio average

4 Conclusion

After analyzing our portfolio using ARIMA and GARCH models, we compared the RMSE of the forecasts from both models over the last 5 days of returns to determine which model performed better. We found that in 5 of the 7 cases listed in Table 1, the GARCH model outperformed the ARIMA model. When optimizing our portfolio, we were interested to see that by removing low performing stocks (stocks far below the efficiency frontier curve), this did decreased the efficiency of our portfolio. However, by replacing low performing stocks with other securities, this increased our efficiency frontier. This shows that decreasing the number of securities in your portfolio does have an impact on your risk and returns. If we want to lower our risk and increase our returns, removing low performing securities may not be the best solution. Instead, we should aim to replace low performing or add higher performing securities.

References

[1] J. Ryan and J. Ulrich et al. (2017). https://cran.r-project.org/web/packages/quantmod/quantmod.pdf

[2] R. Hyndman, G. Athanasopoulos et. al., Online Texts (2018) https://www.otexts.org/fpp/8

[3] R. Hyndman and Y. Khandakar, (2008) "Automatic time series forecasting: The forecast package for R", Journal of Statistical Software, 26(3). https://www.rdocumentation.org/packages/forecast/versions/8.1/topics/auto.arima

[4] Engle, Robert F. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation." Econometrica 50, no. 4 (1982): 987-1007. doi:10.2307/1912773.
https://www.jstor.org/stable/1912773?seq=1#page_scan_tab_contents

Appendix A: Code (Separate R script file attached as well)

```
rm(list = ls())
gc()
par(mfrow=c(1,1))

library(ggplot2)
library(quantmod)
library(tseries)
library(forecast)
library(TSPred)
library(fAssets)
library(fPortfolio)
#library(fGarch)
library(timeSeries)
library(zoo)
library(PerformanceAnalytics)
library(dplyr)
library(rugarch)

call_stock <- function(x){
  prices = get.hist.quote(x, start = "2017-04-19", end = "2018-04-19", quote = "Close", compression =
"d")
  #returns = CalculateReturns(prices, method="simple")
  df <- as.data.frame(prices)
  #df$date <- row.names(df)
  df$date <- time(prices)
  df <- df[,c(2,1)]
}

SPY <- call_stock("SPY")
NKE <- call_stock("NKE")
DIS <- call_stock("DIS")
KO <- call_stock("KO")
GS <- call_stock("GS")
JNJ <- call_stock("JNJ")
AMZN <- call_stock("AMZN")

SPY <- call_stock("SPY")
UA <- call_stock("UA")
DIS <- call_stock("DIS")
KO <- call_stock("KO")
BRKB <- call_stock("BRK-B")
EL <- call_stock("EL")
AMZN <- call_stock("AMZN")

colnames(SPY) <- c("Date", "SPY")
colnames(NKE) <- c("Date", "NKE")
colnames(DIS) <- c("Date", "DIS")
colnames(KO) <- c("Date", "KO")
colnames(GS) <- c("Date", "GS")
colnames(JNJ) <- c("Date", "JNJ")
colnames(AMZN) <- c("Date", "AMZN")

# Replace NKE -> UA, GS -> BRK-B, JNJ -> EL
UA <- call_stock("UA")
BRKB <- call_stock("BRK-B")
EL <- call_stock("EL")
colnames(UA) <- c("Date", "UA")
colnames(BRKB) <- c("Date", "BRK-B")
colnames(EL) <- c("Date", "EL")

##################################### END PULL DATA #####################################

##################################### START DATA EXPLORATION #####################################

IT <- Reduce(dplyr::inner_join,list(SPY, UA, DIS, KO, BRKB, EL, AMZN))
IT_test <- do.call(rbind,  lapply(split(IT,"Date"), function(w) last(w)))

IT <- timeSeries(IT[, 2:8], IT[, 1])
#log(lag(IT) / IT)
IT_return <- returns(IT)
```

```
IT_return
chart.CumReturns(IT_return, legend.loc = 'topleft', main = '')

Spec = portfolioSpec()
setSolver(Spec) = "solveRshortExact"
Frontier <- portfolioFrontier(as.timeSeries(IT_return), Spec, constraints = "Short") #constraints could
also ="Short"
frontierPlot(Frontier, col = rep('orange', 2), pch = 19)

#Split data into training and testing data set
data.train <- window(IT, start="2017-04-19", end="2018-03-01")
dim(as.matrix(data.train)) # 219
data.test <- window(IT, start="2018-03-02", end="2018-04-19")
dim(as.matrix(data.test)) # 33

# Split returns to test train
IT_return.ts <- ts(IT_return)
#IT_return.ts.test <- IT_return.ts[247:251,]
data_return.train <- window(IT_return.ts, start=1, end=246)
dim(as.matrix(data.train)) # 219
data_return.test <- window(IT_return.ts, start=247, end=251)
dim(as.matrix(data.test)) # 33

#################################### END DATA EXPLORATION ####################################

#################################### START ARIMA ####################################
plot(IT_return)
plot(IT_return[,'KO'])
plot(data_return.train[,'KO'])
plot(data_return.test[,'KO'])

# Get best variations
#bestmodel_KO <- auto.arima(data_return.train[,'KO'], trace=TRUE, test="pp", ic="bic")
bestmodel_KO <- arima(data_return.train[,'KO'], c(2,0,2))
summary(bestmodel_KO)
confint(bestmodel_KO)
tsdiag(bestmodel_KO)

# Forecast data
bestmodel_KO.forecast <- forecast(bestmodel_KO, h=100)
bestmodel_KO.forecast
plot(bestmodel_KO.forecast[["residuals"]], xlab="Day", ylab="Residuals")
plot(bestmodel_KO.forecast, xlab="Day", ylab="Returns")


# Plot close up
plotarimapred(data_return.test[,'KO'], bestmodel_KO, xlim=c(100, 200), range.percent = 0.05)
accuracy(bestmodel_KO.forecast, data_return.test[,"KO"])
plot(bestmodel_KO)
accuracy(bestmodel_KO.forecast)
# TODO: Rest of ARIMAS

#################################### END ARIMA ####################################

#################################### START ARCH ####################################

# Set to GARCH(1,1)
IT_garch11_spec <- ugarchspec(variance.model = list(
  garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(0, 0)))

IT_garch11_fit <- ugarchfit(spec = IT_garch11_spec, data = data_return.train[,'KO'])
IT_garch11_fit

# Forecast using GARCH
IT_garch11_fcst <- ugarchforecast(IT_garch11_fit, n.ahead = 5)
garch.ts <-ts(IT_garch11_fcst@forecast$seriesFor, start=247, end=251)

accuracy(garch.ts, data_return.test[,'KO'])


# Backtesting
IT_garch11_roll <- ugarchroll(IT_garch11_spec,  data_return.train[,'KO'],
  n.start = 10, refit.every = 1, refit.window = "moving",
```

```
  solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = 0.05,
  keep.coef = TRUE, solver.control=list(tol=1e-6, trace=1), fit.control=list(scale=1))
#warnings()
# Try to resume - not working
IT_garch11_roll = resume(IT_garch11_roll, solver="gosolnp")

report(IT_garch11_roll, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)
# If the return is more negative than the VaR, we have a VaR exceedance. In our case, a VaR exceedance
should only occur in 5% of the cases (since we speci ed a 95% con dence level).
IT_VaR <- zoo(IT_garch11_roll@forecast$VaR[, 1])
index(IT_VaR) <- as.yearmon(rownames(IT_garch11_roll@forecast$VaR))
IT_actual <- zoo(IT_garch11_roll@forecast$VaR[, 2])
index(IT_actual) <-
  as.yearmon(rownames(IT_garch11_roll@forecast$VaR))

plot(IT_actual, type = "b", main = "95% VaR Backtesting",
   xlab = "Date", ylab = "Return/VaR in percent")
lines(IT_VaR, col = "red")
legend("bottomleft", inset=.05, c("IT return","VaR"), col = c("black","red"), lty = c(1,1))

##################################### END ARCH #####################################

##################################### START OPTIMIZATIONS #####################################

## To solve for a return...
#??portfolioSpec
Spec <- portfolioSpec()
setSolver(Spec) <- "solveRshortExact" #Set the method for solving...See documentation for calculation
options...
#solveRshortExact allows for unlimited short selling..."solveRquadprog" can be used for not short selling
setTargetReturn(Spec) <- mean(colMeans(IT_return)) # to set target at average returns of all columns
#setTargetReturn(Spec) <- 0.08/52 # for weekly data

efficientPortfolio(IT_return, Spec, 'Short') #Could set for LongOnly...Need to make sure spec is not
solveRshortExact or will override...

minvariancePortfolio(IT_return, Spec, 'Short')
minriskPortfolio(IT_return, Spec)
maxreturnPortfolio(IT_return, Spec)

tangencyPortfolio(IT_return, Spec, 'Short')

#highest return/risk ratio on the efficient frontier
#For the MarKOwitz portfolio this is the same as the Sharpae ratio. To find this point on the
#frontier the return/risk ratio calculated from the target return and target risk returned
#by the function efficientPortfolio. Note, the default value of the risk free rate is zero.

#now let's have some fun plotting this...
#?frontierPlot #your available plots...

#Lets get the frontier in a little different way...
frontier=portfolioFrontier(as.timeSeries(IT_return))
frontierPlot(frontier)
grid()



tailoredFrontierPlot(frontier,
                     return = c("mean", "mu"), risk = c("Cov", "Sigma", "CVaR", "VaR"),
                     mText = NULL, col = NULL, ylim = NULL,
                     twoAssets = FALSE, sharpeRatio = FALSE, title = TRUE,
                     xlim = c(0.005,0.02))


frontier-plot(frontier,
              return = c("mean", "mu"), risk = c("Cov", "Sigma", "CVaR", "VaR"),
              mText = NULL, col = NULL, xlim = NULL, ylim = NULL,
              twoAssets = FALSE, sharpeRatio = TRUE, title = TRUE)

weightsPlot(frontier) #Black line is minimum variance portfolio

#Tangency Portfolio Graphs
tgPort=tangencyPortfolio(IT_return)
weightsPie(tgPort) #weights of securities in tangency portfolio
```

```
weightedReturnsPie(tgPort) #pie chart of weighted returns of the tangency portfolio

# Remove NKE, GS, JNJ
IT2 <- Reduce(dplyr::inner_join,list(SPY, DIS, KO, AMZN))
IT2_test <- do.call(rbind,  lapply(split(IT2,"Date"), function(w) last(w)))
IT2 <- timeSeries(IT2[, 2:5], IT2[, 1])
log(lag(IT2) / IT2)
IT2_return <- returns(IT2)
IT2_return

# Replace NKE -> UA -> NFLX, GS -> BRK-B, JNJ -> EL -> FB
NFLX <- call_stock("NFLX")
BRKB <- call_stock("BRK-B")
FB <- call_stock("FB")

colnames(NFLX) <- c("Date", "NFLX")
colnames(BRKB) <- c("Date", "BRK-B")
colnames(FB) <- c("Date", "FB")

IT3 <- Reduce(dplyr::inner_join,list(SPY, DIS, KO, AMZN, UA, BRKB, EL))
IT3_test <- do.call(rbind,  lapply(split(IT3,"Date"), function(w) last(w)))
IT3 <- timeSeries(IT3[, 2:8], IT3[, 1])
log(lag(IT3) / IT3)
IT3_return <- returns(IT3)
IT3_return

Spec = portfolioSpec()
setSolver(Spec) = "solveRshortExact"
Frontier <- portfolioFrontier(as.timeSeries(IT_return), Spec, constraints = "Short") #constraints could
also ="Short"
Frontier2 <- portfolioFrontier(as.timeSeries(IT2_return), Spec, constraints = "Short")
Frontier3 <- portfolioFrontier(as.timeSeries(IT3_return), Spec, constraints = "Short")
frontierPlot(Frontier, col = rep('orange', 2), pch = 19)
frontierPlot(Frontier2, col = rep('green',2), pch = 19, add = TRUE)
frontierPlot(Frontier3, col = rep('blue',2), pch = 19, add = TRUE)

##################################### END OPTIMIZATIONS #####################################

# Set to GARCH(1,1)
IT_garch11_spec <- ugarchspec(variance.model = list(
  garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(0, 0)))

custom_garch <- function(spec, df, column){

IT_garch11_fit <- ugarchfit(spec = spec, data = df[,column])

# Forecast using GARCH
IT_garch11_fcst <- ugarchforecast(IT_garch11_fit, n.ahead = 12)



# Backtesting
IT_garch11_roll <- ugarchroll(spec,  df[,column],
                              n.start = 10, refit.every = 1, refit.window = "moving",
                              solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = 0.05,
                              keep.coef     =     TRUE,     solver.control=list(tol=1e-6,     trace=1),
fit.control=list(scale=1))
#warnings()
# Try to resume - not working
IT_garch11_roll = resume(IT_garch11_roll, solver="gosolnp")

report <-report(IT_garch11_roll, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)

# If the return is more negative than the VaR, we have a VaR exceedance. In our case, a VaR exceedance
should only occur in 5% of the cases (since we speci ed a 95% con dence level).
IT_VaR <- zoo(IT_garch11_roll@forecast$VaR[, 1])
index(IT_VaR) <- as.yearmon(rownames(IT_garch11_roll@forecast$VaR))
IT_actual <- zoo(IT_garch11_roll@forecast$VaR[, 2])
index(IT_actual) <-
  as.yearmon(rownames(IT_garch11_roll@forecast$VaR))

#plot(IT_actual, type = "b", main = "95% VaR Backtesting",
```

```
#     xlab = "Date", ylab = "Return/VaR in percent")
#lines(IT_VaR, col = "red")
#legend("topright", inset=.05, c("IT return","VaR"), col = c("black","red"), lty = c(1,1))
list(report =report, IT_VAR = IT_VaR, IT_actual =IT_actual)
}

#KO <- call_stock("KO")
#KO <- call_stock("KO")
#KO <- call_stock("KO")
#CO <- call_stock("CO")
#GS <- call_stock("GS")
#JNJ <- call_stock("JNJ")
#KO <- call_stock("KO")
#joe_garch <- custom_garch(IT_garch11_spec,data_return.train,'KO')
stocks <- c("SPY","NKE","DIS","GS","JNJ","KO","AMZN")
stock_garch <- list()

for(i in 1:length(stocks)){
  stock_garch[[i]] <- custom_garch(IT_garch11_spec,data_return.train,stocks[i])
}


SPY <- call_stock("SPY")
UA <- call_stock("UA")
DIS <- call_stock("DIS")
KO <- call_stock("KO")
BRKB <- call_stock("BRK-B")
EL <- call_stock("EL")
AMZN <- call_stock("AMZN")


stocks <- c("SPY","UA","DIS","BRK-B","EL","KO","AMZN")


garch_accuracy <- function(x){
IT_garch11_spec <- ugarchspec(variance.model = list(
  garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(0, 0)))

IT_garch11_fit <- ugarchfit(spec = IT_garch11_spec, data = data_return.train[,x])
IT_garch11_fit

# Forecast using GARCH
IT_garch11_fcst <- ugarchforecast(IT_garch11_fit, n.ahead = 5)
garch.ts <-ts(IT_garch11_fcst@forecast$seriesFor, start=247, end=251)

print(accuracy(garch.ts, data_return.test[,x]))
}


for (i in stocks){
  garch_accuracy(i)
  print(i)
}

arima_accuracy <- function(x){
  bestmodel_KO <- arima(data_return.train[,x], c(2,0,2))

  # Forecast data
  bestmodel_KO.forecast <- forecast(bestmodel_KO, h=100)


  # Plot close up
  accuracy(bestmodel_KO.forecast)
}



for (i in stocks){
  print(arima_accuracy(i))
  print(i)
}
```