Stock Portfolio Predictions

Joe Cook, Kaitlin Kirasich, Damon Resnick

March 31, 2018

1 Introduction

In this paper we analyzed and predicted a portfolio of three of each of the three team members' favorite stocks (NFLX, NVDA, FB, SQ, SHOP, RHT, Z, TXN, MTCH). For our data source, we used the R library "*quantmod*". *Quantmod* is a quantitative financial modelling framework used to specify, build, trade, and analyze stock prices and other financial trading strategies [1]. For the purposes of our project, we only used this library to fetch the daily open, close, high, and low prices for the last two years.

We used these daily values from the past two years (start date: January 1, 2016) to predict the closing price of our portfolio for both the last two weeks of March, most recent, and the first two weeks of 2018, before a large structural change in the markets occurred. To predict these prices, we created simple Regression models and AutoRegressive Integrated Moving Average (ARIMA) models.

To optimize the models, we split the data into train and test sets to find the best model to use to get the best accuracy in forecasting the closing price over a span of 10 days. The RMSE was calculated by predicting the closing prices of the all the stock prices in the portfolio and their Root Mean Squared Error (RMSE) to the actual closing prices during those 10 days.

2 Models

For our time series analysis, we chose to use a simple linear regression and an ARIMA model [2]. Linear regression can be very good at fitting the data inside of the domain of the data range but will be very bad at predicting outside of that range. Since the price of a stock is heavily correlated to the previous price of the stock this violates the assumption of independence between dependent variables. For our model we only used the dates and closing prices. This resulted in a model that was good at predicting the price inside our training set but was a poor predictor of our test set as can be seen in Figure 1.

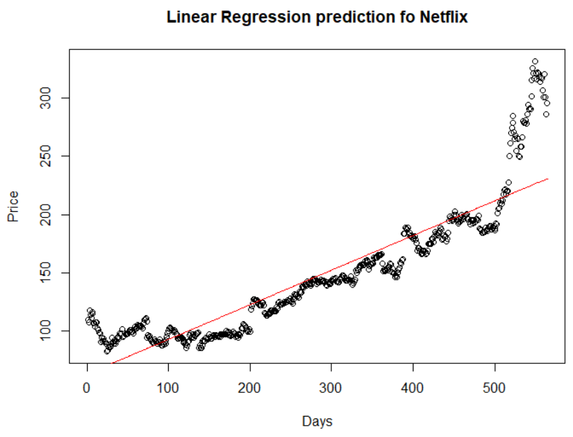

Figure 1: Linear Regression line was fit over all the entire range of data for Netflix except the last 10 days. The fit looks good over that range but does not predict the large increase in the price over the last 10 days very well.

ARIMA is a forecasting technique that is good at projecting future values of a series based on its own inertia. The major application of ARIMA is for short term forecasting and usually requires at least 40 data points. ARIMA works best when the data exhibits a stable or consistent pattern over time with a minimum number of outliers. ARIMA models aim to describe the autocorrelations in the data and are applied in cases where an initial differencing step can be applied one or more times to eliminate non-stationarity in data [3]. Using the R "*forecast*" package, we are able to run the *auto.arima*() function to determine which ARIMA model meets the chosen criteria best [4]. This technique allows us to easily customize the relevant criteria to get the best model.

For our models, we used the unit root/stationarity *test = KPSS* as a criterion for stationarity since we got the same results with the *ADF* and *PP*. For the fit criteria, we went with *AIC* because *BIC* seemed to be a little too conservative, while *AICC* gave the same results as *AIC*. We also set *stepwise* equal to *FALSE* so that it would search over all models. This slowed down the run time but ensured that the best model was not being skipped over. In Figure 2, we confirmed that the data for each model were good candidates for an ARIMA model. The standardized residuals plots do not show any clear patterns but do have peaks and dips occasionally. We also see from our ACF of residuals a strong first lag but no other notable concerns.

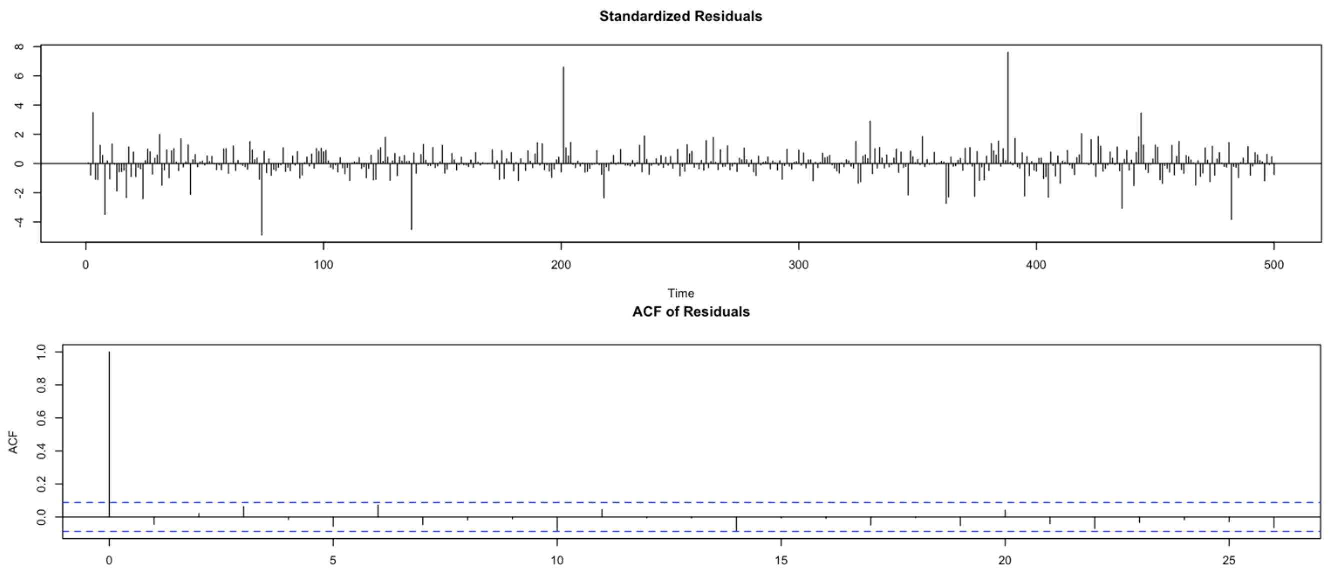

Figure 2: Standardized residuals and ACF of residuals for Netflix. The plots for the other stocks were similar.

What we found most interesting were the differences in the ARIMA models chosen when training over different time periods. The stock market has been extremely volatile in the past few months causing the *auto.arima*() function to give *AR* terms of values 3 to 5. When we pulled back and only trained the data before the start of the year, we got better forecasting. However, the models still did not prove to be very accurate because stocks like Netflix have skyrocketed compared to their previous performances while stocks like Facebook have tanked.

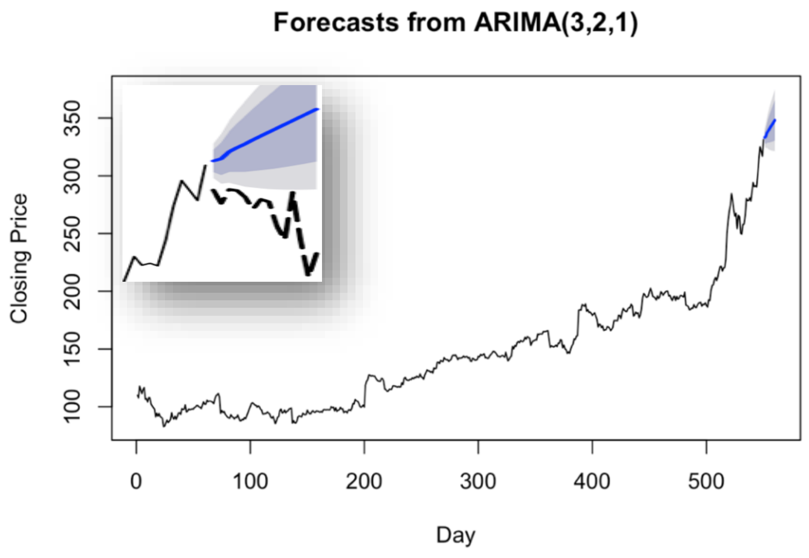

Figure 3: Historical performance of the closing price of Netflix over the last few years. Predictions based on this data do poorly to account for the recent news of the data breaches that have plunged the Netflix stock price as can be seen in the inlay. The predicted blue curve is way off the actual closing price over the 10-day forecast.

For example, Figure 3 shows Netflix's performance over the last few years. When training on data from January 4, 2016 up to day 550 (March 9, 2018), we would assume that the prices would continue to climb. However, the prices dropped drastically in one week, causing our predictions to be significantly off. As each day progressed, our model degraded in accuracy. The predictions went up while the actual prices went down. The Figure 3 inlay shows a close-up of the predictions overlaid with the actual data. You can clearly see that predictions do well to continue the relative trend in the training data while completely missing the large structural change caused by recent news of an embarrassing data breach.

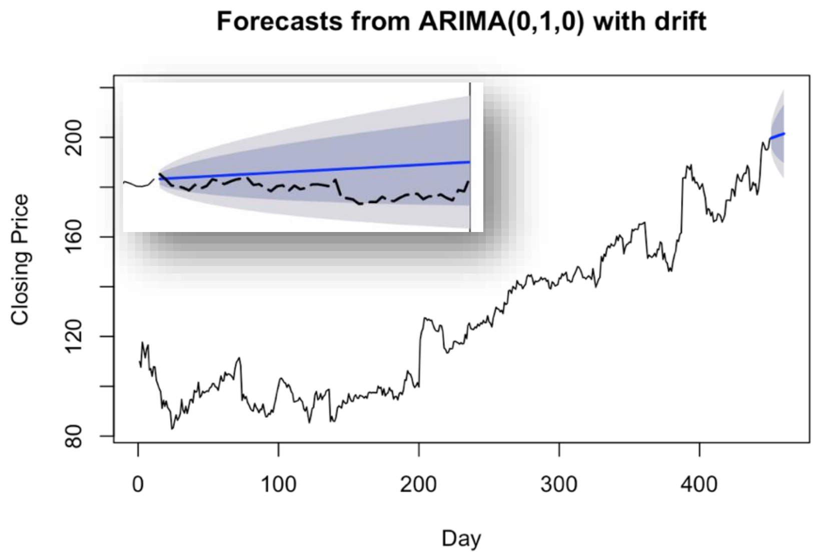

Figure 4: Historical performance of the closing price of Netflix up to the start of 2018. The best fit ARIMA model for this data is much simpler leading to more accurate forecasting before the structural change.

The data in Figure 4 was trained on data up to day 450 (October 13, 2017). This give a best fit ARIMA model of (0,1,0). The ARIMA model does a much better job at forecasting. The difference in RMSE for all the models is shown in Table 1. You can see that by and large the ARIMA model proved better at forecasting the closing prices than the simple linear regression model.

RMSE	FB	NFLX	NVDA	SQ	SHOP	RHT	Z	TXN	MTCH	Average
Regression	22.66733	80.45316	15.51545	14.98489	17.5304	26.67004	7.73123	3.772023	14.41877	22.63814
ARIMA	22.492391	23.976972	16.790335	7.8018272	13.215895	5.83122	3.406919	5.615153	3.2995957	11.38115

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

To compare the accuracy of our models we predicted the next ten days of stock prices and then compared the RMSE for each stock. In most cases, the ARIMA model performed much better than the regression model. However, in a few cases, the regression model performed about the same or slightly better than the ARIMA model.

3 Conclusion

A sample of our nine favorite tech stocks were analyzed in a portfolio. Simple linear regression and autoregressive integrated moving average models were used to predict the closing prices for each stock as well as the average of all stocks in the portfolio ten days into the future. Based on the average RMSE for each model, we found that the ARIMA model performed better at predicting future closing prices for the next ten days.

References

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

[2] H. Seltman, (2008). http://onlinestatbook.com/Online_Statistics_Education.pdf, p. 227.

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

[4] 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

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

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

```
library(quantmod)
library(tseries)
library(forecast)
library(tspred)

library(quantmod)
library(tseries)
library(forecast)

stocks <- c("NFLX", "NVDA", "FB", "SQ", "SHOP", "RHT", "Z", "TXN", "MTCH")
start_date <- "2016-01-01"


for(i in 1:length(stocks)){
  data.frame(getSymbols(stocks[i],from=start_date))
}

a<- merge(FB, MTCH, by=0, all = TRUE)
a <- merge(a, NFLX, by=0, all = TRUE)
a <- merge(a, NVDA, by=0, all = TRUE)
a <- merge(a, RHT, by=0, all = TRUE)
a <- merge(a, SHOP, by=0, all = TRUE)
a <- merge(a, SQ, by=0, all = TRUE)
a <- merge(a, TXN, by=0, all = TRUE)
a <- merge(a, Z, by=0, all = TRUE)

a <- data.frame(a)

b <- grep("by",colnames(a))

a <- a[,-b]

a.ts <- ts(a)


reg_func <- function(df)
{
check <- as.numeric(df[!is.na(df)])
#regdata <- data.frame(check = check[3:length(check)], index = 1:(length(check)-2),
prev_day = check[1:(length(check)-2)], two_day = check[2:(length(check)-1)])
regdata <- data.frame(check = check[1:(length(check))], index = 1:(length(check)))
regdata_train <- regdata[1:floor(length(check)-10),]
regdata_test <- regdata[floor(length(check)-9):nrow(regdata),]
reg <- lm(check ~ index, regdata_train)
summary(reg)
plot(regdata_train$check)
lines(predict(reg), col = "red")



predicted <- c(predict(reg),predict(reg, regdata_test))
plot(regdata$check)
lines(predicted, col = "red")
```

```
list(rmse = accuracy(predict(reg, regdata_test),regdata_test$check)[2],predicted =
predicted)
}

ghs <- reg_func(a$NFLX.Close)
stocks_predict <- list(a$NFLX.Close, a$NVDA.Close, a$FB.Close, a$SQ.Close, a$SHOP.Close,
a$RHT.Close, a$Z.Close, a$TXN.Close, a$MTCH.Close)

accuracy_scores <- list()
for(i in 1:length(stocks_predict)){
  accuracy_scores[[i]] <- reg_func(stocks_predict[[i]])[1]
}
#accuracy(predict(reg, regdata_test),regdata_test$check)[2]

rmse_scores <- unlist(accuracy_scores)
names(rmse_scores) <- paste(stocks,"rmse", sep="_")
rmse_scores <- data.frame(rmse_scores)


Predicted_scores <- list()
for(i in 1:length(stocks_predict)){
  Predicted_scores[[i]] <- reg_func(stocks_predict[[i]])[2]
}
Predicted_scores_df <- do.call(data.frame, Predicted_scores)
names(Predicted_scores_df) <- paste(stocks,"Reg_Predict", sep="_")

data.train <- window(a.ts[,'NFLX.Close'], start=1, end=554)
plot(data.train)
dim(as.matrix(data.train))
data.test <- window(a.ts[,'NFLX.Close'], start=555, end=564)
plot(data.test)
dim(as.matrix(data.test))

bestmodel1b <- auto.arima(data.train, seasonal = TRUE, stepwise = FALSE,
approximation=FALSE, trace=TRUE, test="kpss", ic="aic")
bestmodel1b

summary(bestmodel1b)
confint(bestmodel1b)
tsdiag(bestmodel1b)

bestmodel1b.forecast <- forecast(bestmodel1b, h=10)
bestmodel1b.forecast
plot(bestmodel1b.forecast, xlab="Day", ylab="Closing Price", xlim=c(1, 564))

plotarimapred(data.test, bestmodel1b, xlim=c(450, 564), range.percent = 0.05)
accuracy(bestmodel1b.forecast, data.test)
plotarimapred(data.test, bestmodel1b.forecast, xlim=c(548, 564), range.percent = 0.05)


arima_predictions <- function(x){
  #Split data into training and testing data set
  data.train <- window(a.ts[,x], start=1, end=554)
  plot(data.train)
  dim(as.matrix(data.train))
  data.test <- window(a.ts[,x], start=555, end=564)
  plot(data.test)
  dim(as.matrix(data.test))

  bestmodel1b <- auto.arima(data.train, seasonal = TRUE, stepwise = FALSE,
approximation=FALSE, trace=TRUE, test="kpss", ic="aic")

  bestmodel1b.forecast <- forecast(bestmodel1b, h=10)
  predicted <- c(bestmodel1b$fitted,bestmodel1b.forecast$mean)
  predicted
```

```
}

stocks_predict_ar <- list("NFLX.Close", "NVDA.Close", "FB.Close", "SQ.Close", "SHOP.Close",
"RHT.Close", "Z.Close", "TXN.Close", "MTCH.Close")

arima_list <- list()
for(i in 1:length(stocks_predict_ar)){
  arima_list[[i]] <- arima_predictions(stocks_predict_ar[[i]])
  print(i)
}
Predicted_scores_ar <- do.call(data.frame, arima_list)
names(Predicted_scores_ar) <- paste(stocks_predict_ar,"AR_Predict", sep="_")
b <- grep("Close",colnames(a))
slimdf <- a[,b]
everything <- data.frame(slimdf, Predicted_scores_df, Predicted_scores_ar)
write.csv(everything,"All_close_predictions.csv")
```