Forecast NFLX stock

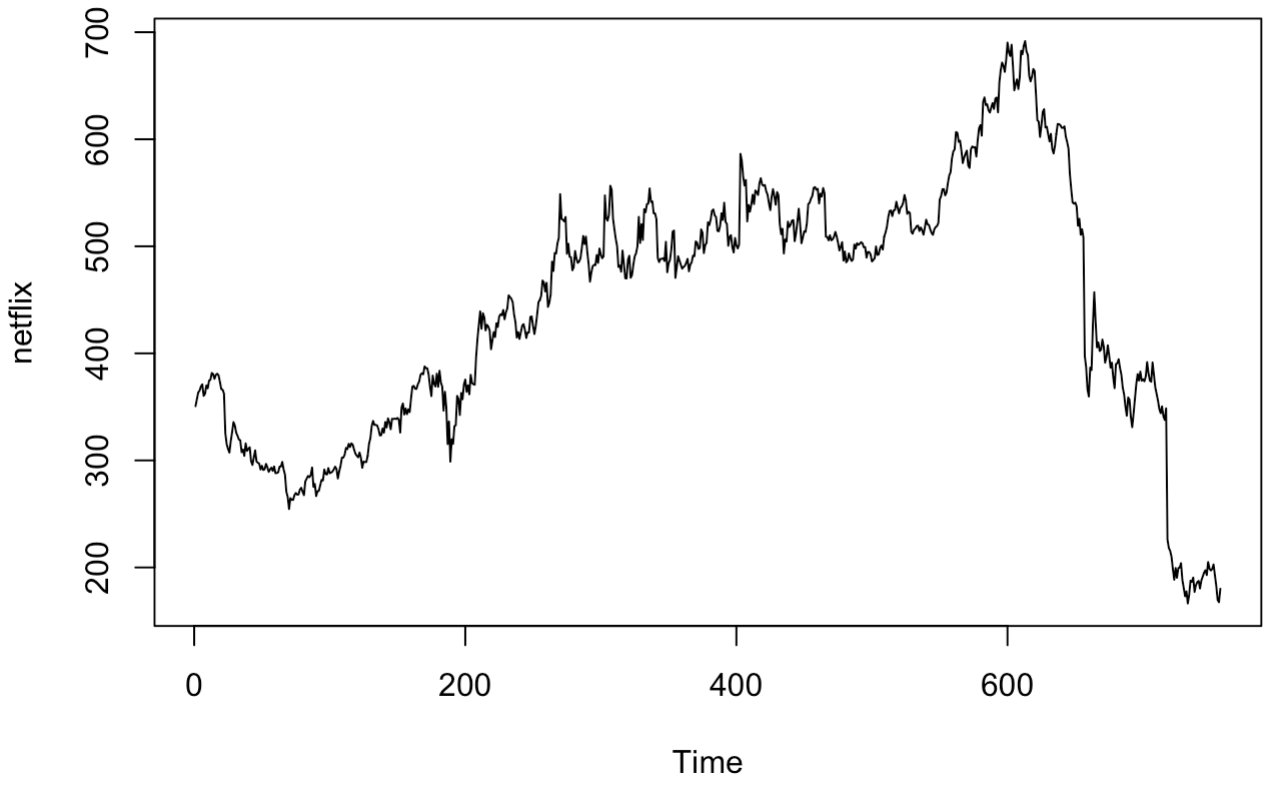
Samarth

```
suppressPackageStartupMessages({
  library(TSA)
  library(ggplot2)
  library(dplyr)
  library(forecast)
  library(tseries)
  library(xgboost)
  library(caret)
  library(Metrics)
})
```

We'll try to forecast Netflix stock price. The dataset contains NFLX stock data from 6/17/2019 to 6/15/2022. Our goal is to forecast the close price for 10 days.

source - https://finance.yahoo.com/quote/NFLX/history?p=NFLX (https://finance.yahoo.com/quote/NFLX/history?p=NFLX)

```
netflix = read.csv("./data/NFLX.csv")
netflix = netflix$Close
T = 758
ts.plot(netflix)
```




```
netflix = ts(netflix)
netflix_train= ts(netflix[1:747], start=1, end = 747)
netflix_test= ts(netflix[748:757], start = 749, end = 757)
```

```
adf.test(netflix_train)
```

```
## Warning in adf.test(netflix_train): p-value greater than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  netflix_train
## Dickey-Fuller = 0.2059, Lag order = 9, p-value = 0.99
## alternative hypothesis: stationary
```

p-value of 0.99 suggests that the data is non-stationary

First differencing

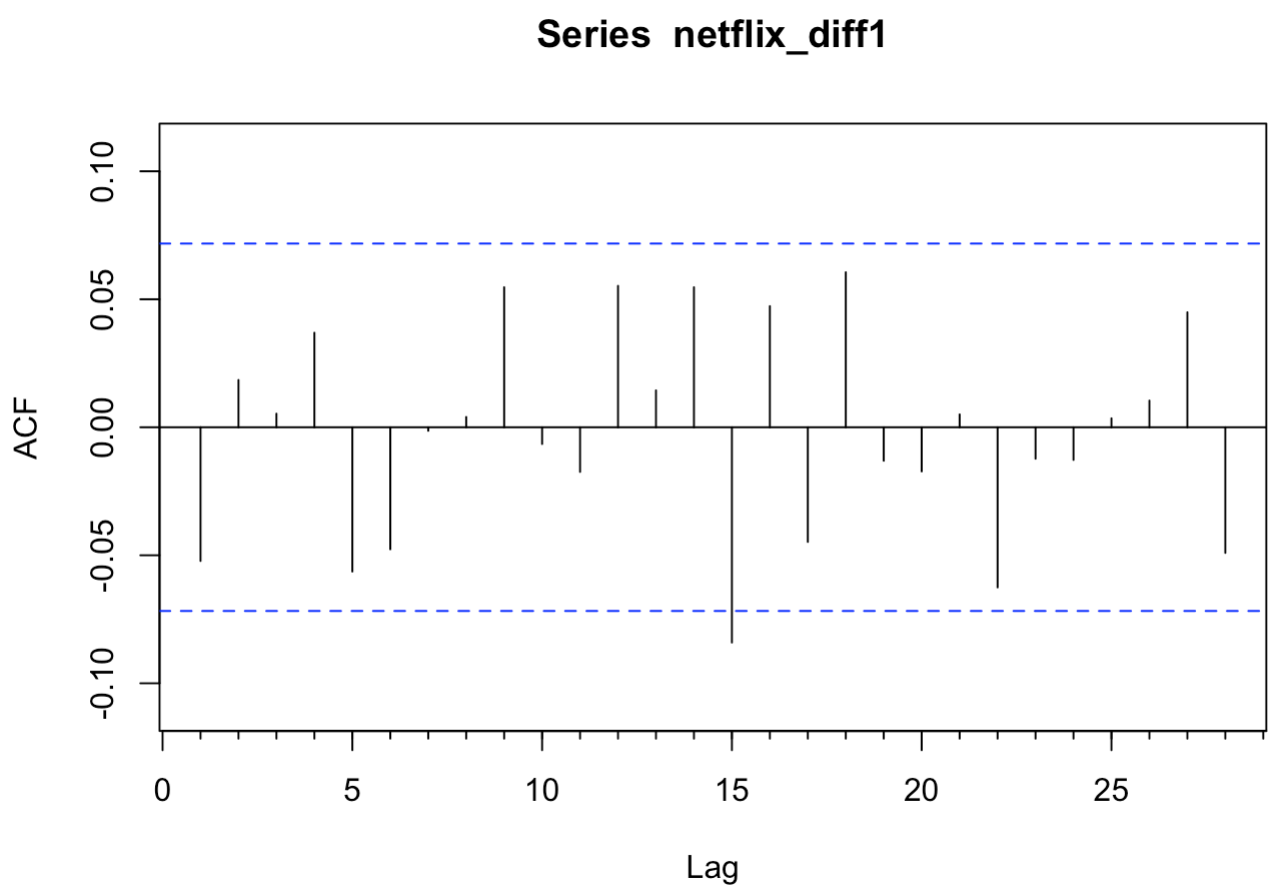
```
netflix_diff1=diff(netflix_train)
adf.test(netflix_diff1)
```

```
## Warning in adf.test(netflix_diff1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  netflix_diff1
## Dickey-Fuller = -8.6126, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

p-value now suggests a stationary series

```
Acf(netflix_diff1)
```



Unclear if tailing or cutting off, no seasonality

```
eacf(netflix_diff1)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 o o o o o o o o o o o  o  o  o
## 1 x o o o o o o o o o o  o  o  o
## 2 x o o o o o o o o o o  o  o  o
## 3 x x x o o o o o o o o  o  o  o
## 4 x x x x o o o o o o o  o  o  o
## 5 x x x x x o o o o o o  o  o  o
## 6 x x x x x x o o o o o  o  o  o
## 7 x o x x o x x o o o o  o  o  o
```

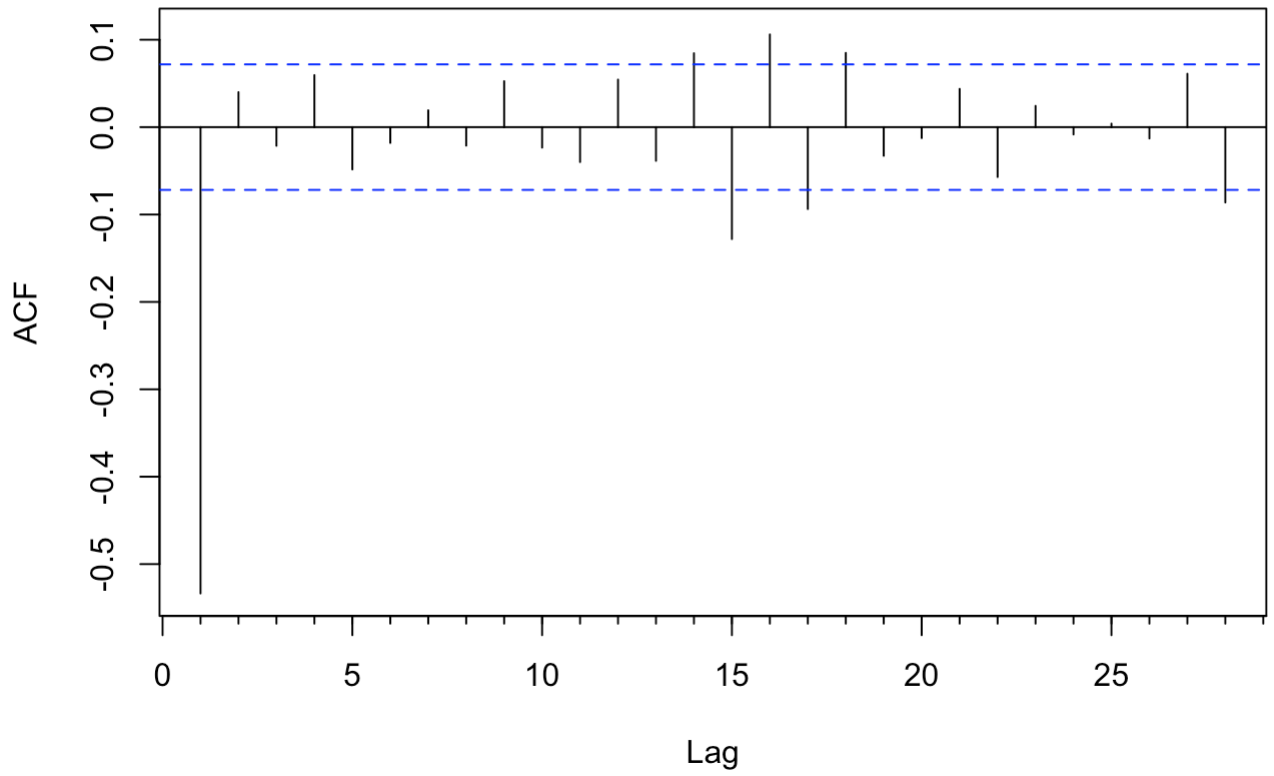
2nd difference

```
netflix_diff2=diff(netflix_train,differences = 2)
adf.test(netflix_diff2)
```

```
## Warning in adf.test(netflix_diff2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  netflix_diff2
## Dickey-Fuller = -14.851, Lag order = 9, p-value = 0.01
## alternative hypothesis: stationary
```

```
Acf(netflix_diff2)
```



With 2nd differencing, the best model is ARIMA(0,2,1). However there isn't much difference between the 1st and 2nd differencing. So we'll proceesd with 1st differencing to ensure we don't over difference.

Check BIC and AIC for serveral models

```
Arima(netflix_train,order=c(0,1,0))
```

```
## Series: netflix_train
## ARIMA(0,1,0)
##
## sigma^2 = 158.8:  log likelihood = -2948.8
## AIC=5899.61   AICc=5899.61   BIC=5904.22
```

```
Arima(netflix_train,order=c(0,1,1))
```

```
## Series: netflix_train
## ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.0501
## s.e.   0.0358
##
## sigma^2 = 158.6:  log likelihood = -2947.83
## AIC=5899.67   AICc=5899.68   BIC=5908.89
```

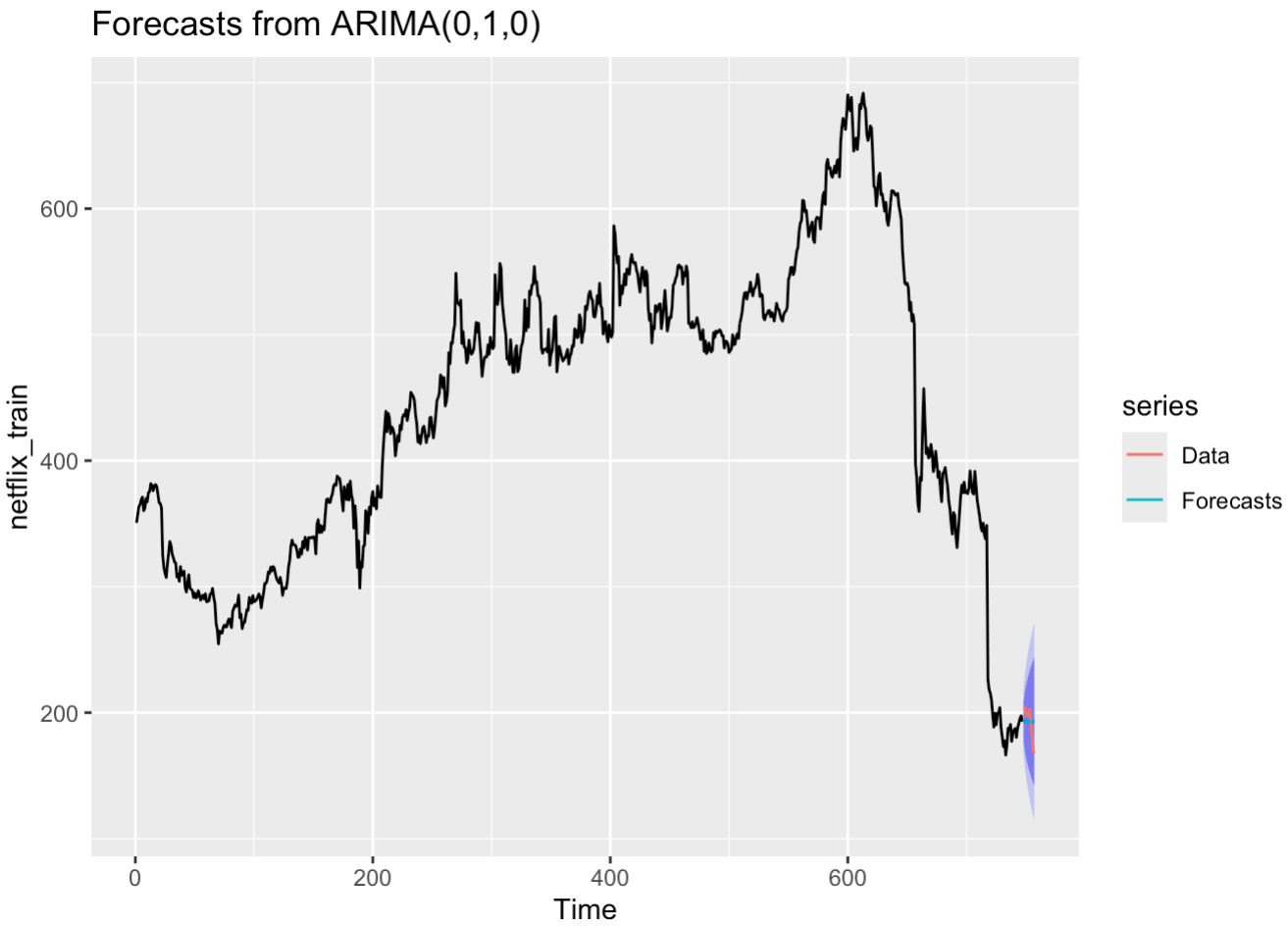
```
Arima(netflix_train,order=c(1,1,0))
```

```
## Series: netflix_train
## ARIMA(1,1,0)
##
## Coefficients:
##           ar1
##       -0.0519
## s.e.   0.0366
##
## sigma^2 = 158.6:  log likelihood = -2947.8
## AIC=5899.6   AICc=5899.61   BIC=5908.83
```

```
arima_est <- Arima(netflix_train,order=c(0,1,0))
arima_est
```

```
## Series: netflix_train
## ARIMA(0,1,0)
##
## sigma^2 = 158.8:  log likelihood = -2948.8
## AIC=5899.61   AICc=5899.61   BIC=5904.22
```

```
arima_pred <- forecast(arima_est,h=10)
autoplot(arima_pred)+
autolayer(netflix_test, series="Data") +
autolayer(arima_pred$mean, series="Forecasts")
```



Calculate RMSE

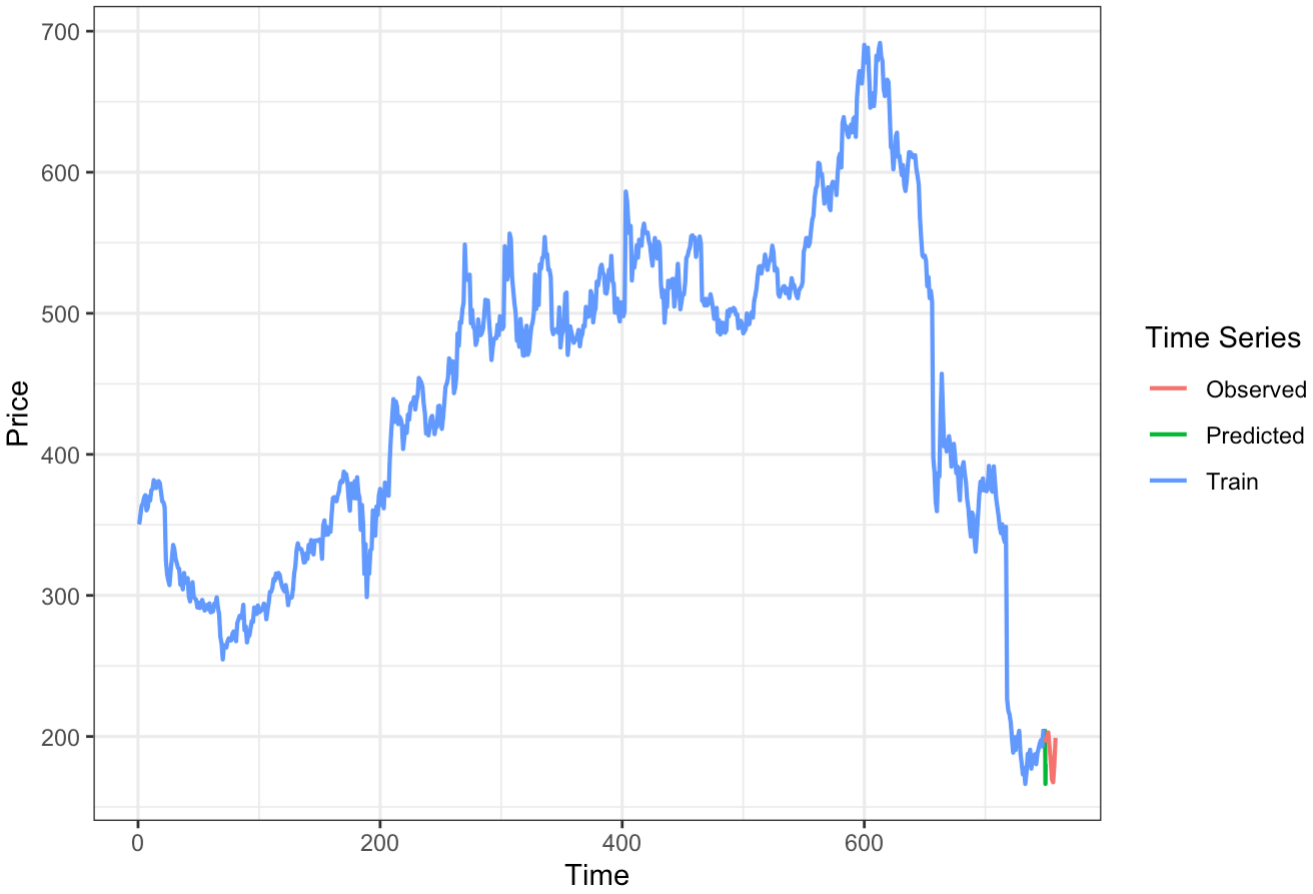
```
rmse(arima_pred$mean,netflix_test)
```

```
## [1] 13.40072
```

Train xgboost for forecasting

Preparing forecast object

```
fitted <- model_xgb %>%
stats::predict(train_Dmatrix) %>%
stats::ts(start=1, end = 748)
data2= data$Close
ts_netflix<- ts(data2[1:748], start=1, end = 748)
forecast_xgb <- model_xgb %>% stats::predict(pred_Dmatrix)
forecast_ts <- ts(forecast_xgb,start=c(749),frequency=10)
forecast_netflix <- list(
model = model_xgb$modelInfo,
method = model_xgb$method,
mean = forecast_ts,
x = ts_netflix,
fitted = fitted,
residuals = as.numeric(ts_netflix) - as.numeric(fitted)
)
class(forecast_netflix) <- "forecast"
observed_values <- ts(test$Close,start = 749, end = 758)
autoplot(forecast_netflix)+
autolayer(forecast_netflix$mean,series="Predicted",size=0.75) +
autolayer(forecast_netflix$x,series ="Train",size=0.75 ) +
autolayer(observed_values,series = "Observed",size=0.75) +
#scale_x_continuous(labels =date_transform,breaks=seq(2013,2021,2))+
guides(colour=guide_legend(title = "Time Series")) +
ylab("Price") + xlab("Time") +
ggtitle("") +
theme_bw()
```




```
rmse(forecast_netflix$mean,test$Close)
```

```
## [1] 1.270181
```

XGBoost's performance is significantly better than ARIMA