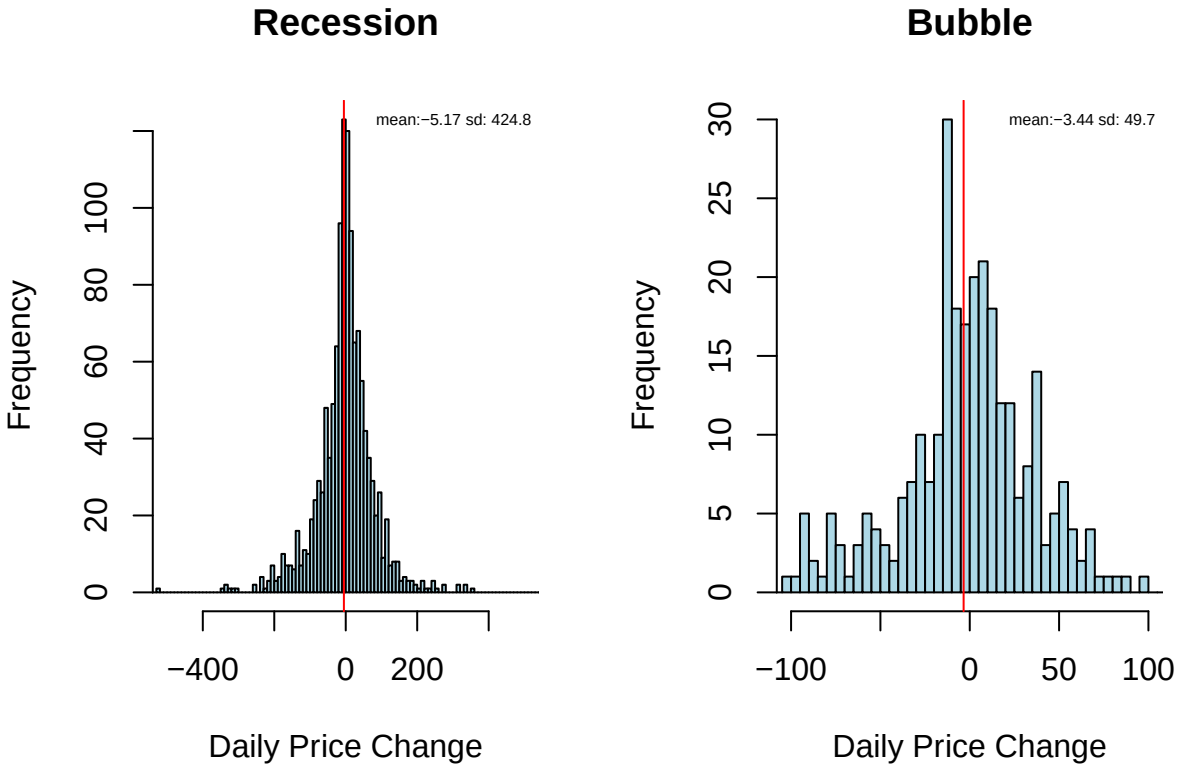
STAT551 Stochastic Processes Project

Daniel Snyder and Yibei Chen

5/4/2017

Introduction

In the stock market, some argue that the periodic returns are normally distributed and depend on underlying market parameters. [4] We may think of these underlying parameter states as Bull or Bear market. In one state the market may have high expected return with low variance. Other times it may have higher variance and negative expected return. Here are two examples of the daily price change of the S&P 500 in two different regimes.



Knowing the current environment of the market gives the better prediction of the stock price. In this project, we would like to construct a model of this market environment and better predict the stock price using this unknown market environment.

We assume each of the stock has Gaussian distribution in a market environment and each market environment is modeled Hidden Markov Model (HMM) to obtain the discrete market regime. In this model, the state of the market regime changes according to a Markov Chain transition probability. We may estimate the parameters of this model from observed data.

Data Analysis

The stock price data was obtained from NASDAQ historical quote tool [3]. The stocks were chosen from a wide range of market segments: AAPL (Apple Computer), GOOG (Alphabet), JNJ (Johnson and Johnson), F (Ford), MSFT (Microsoft), SBUX (Starbucks), UTX (United Technologies). The price history was extracted for the past 10 years and is reported on every business day that the stock exchange was open. The stock history spans the past 10 years, except for the dataset of GOOG stock, which was only available since 2014 after a stock split.

Methods

Here we use a Hidden Markov model to make predictions of stock prices given historical data. Under this model, the unobserved process is a Markov chain whose states represent states of the stock market (Bull market and Bear market for example). Let X_i denote the unobserved Markov Process with discrete state space. Let Y_i denote the observed, Markov dependent process. Let Y_i be the log of price ratio from open to close of a given stock symbol. Assume a normal distribution of Y_i given X_i. Symbolically, $Y_i|X_i \sim N(\mu, \sigma^2)$. The observed process, Y_i depends on the unknown market state, X_i.

Model Parameters Firstly, the initial state of the Markov chain has some inital distribution: $X_0 \sim \pi_0$. This is a distribution of probability to all possible market states at time zero. The number of states of the Markov chain is a parameter that may affect the predictive accuracy of the model.

The Markov chain has some time-homogeneous transition matrix, A, which dictates the probability of transition from one unknown market state to another. The $(i,j)^{th}$ entry is defined to be:

$$A(i,j) = \mathbb{P}[X_{t+1} = j | X_t = i] B(i,j) = \mathbb{P}[Y_t = j | X_t = i] \tag{1}$$

Finally, the model has parameters dictating the distribution of the daily price change Y_i conditional upon the unknown market state X_i. Since the Y_i are assumed to be Gaussian the parameterization will be mean and standard deviation conditional upon market state.[4]

$$\mathbb{P}(Y_t = y_t | X_t = x_t, \theta) \sim N(\mu_X, \sigma^2_{x_t}) \tag{2}$$

where

$$\theta := \{\pi_i, A(i,j), \mu_i, \sigma_i\} \tag{3}$$

Likelihood Function

To estimate the most likely parameters, one must find the maximum of the posterior likelihood of the observed data given the parameters.

$$\max_{\theta} \mathbb{P}\{Y_0 = y_0, \cdots, Y_N = y_N | \theta\} \tag{4}$$

$$
\begin{aligned}
L_c(\theta) =& \mathbb{P}(Y_0 = y_0, \cdots, Y_N = y_N, X_0 = y_0, \cdots, X_N = y_N|\theta) \\
=& \mathbb{P}(X_0 = x_0|\theta)\mathbb{P}(Y_0 = y_0|X_0 = x_0; \theta), \cdots, \\
& \mathbb{P}(X_N = x_N|X_{N-1} = x_{N-1}\theta)\mathbb{P}(Y_N = y_N|X_N = x_N; \theta) \\
=& \mathbb{P}(X_0 = x_0|\theta)\mathbb{P}(Y_0 = y_0|X_0 = x_0; \theta) \\
& \prod_{t=1}^{N} \mathbb{P}(X_t = x_t|X_{t-1} = x_{t-1}\theta)\mathbb{P}(Y_t = y_t|X_t = x_t; \theta) \\
\log L_c(\theta) =& \log \pi_{x_0} + \sum_{t+1}^{N} \log p_{x_t x_{t-1}} + \sum (-\frac{1}{2}\log 2\pi\sigma_{x_t}^2 - \frac{(y_t - \mu_{x_t})^2}{2\sigma_{x_t}^2}) \qquad (5) \\
E[\log L_c(\theta)|Y_1, \cdots, Y_N; \tilde{\theta}] =& \sum_i \mathbb{P}(X_{-0} = i|Y_0, \cdots, Y_N; \tilde{\theta}) \log \pi_i \\
& + \sum_{i,j} \sum_{t=1}^{N} \mathbb{P}(X_t = j, X_{t-1=i}|Y_0, \cdots, Y_N; \tilde{\theta}) \log A(i,j) \\
& + \sum_{i,j} \sum_{t=0}^{N} \mathbb{P}(X_t = i|Y_1, \cdots, Y_N; \tilde{\theta})(-\frac{1}{2}\log 2\pi\sigma_i^2 - \frac{(y_t - \mu_i)^2}{2\sigma_i^2})
\end{aligned}
$$

[4]

Expectation Maximimization To estimate the maximally likely parameters, we will use the Baum Welch algorithm which is a form of Expectation Maximization [1].

Let forward and backward probabilities in the Hidden Markov Model be defined as:

$$
\begin{aligned}
\alpha_t(j) &= \mathbb{P}(Y_0 = y_0, \cdots, Y_t = y_t, X_t = j) \\
\beta_t(j) &= \mathbb{P}(Y_{t+1} = y_{t+1}, \cdots, Y_N = y_N|X_t = j)
\end{aligned} \qquad (6)
$$

Their values may be found recursively:

$$
\begin{aligned}
\alpha_{t+1}(j) =& \mathbb{P}(X_{t+1} = j, Y_0 = y + 0, \cdots, Y_{t+1} = y_{t+1}) \\
=& \sum_i \mathbb{P}(X_{t+1} = j, X_t = i, Y_0 = y + 0, \cdots, Y_{t+1} = l) \\
=& \sum_i \mathbb{P}(Y_{t+1} = l)|X_{t+1} = j, X_t = i, Y_0 = y + 0, \cdots, Y_t = y_t) \\
& \mathbb{P}(X_{t+1} = j|X_t = i, Y_0 = y + 0, \cdots, Y_t = y_t)\mathbb{P}(Y_0 = y_0, \cdots, Y_t = y_t, X_t = j) \\
=& \sum_i B(j,l)A(i,j)\alpha_t(i) \\
\beta_t(j) =& \mathbb{P}(Y_{t+1} = l, \cdots, Y_N = y_N|X_t = j) \\
=& \sum_k \mathbb{P}(X_{t+1} = k, Y_{t+1} = l, \cdots, Y_N =_N |X_t = j) \\
=& \sum_k \mathbb{P}(Y_{t+2} = y_{t+2}, \cdots, Y_N = y_N|X_{t+1} = k) \\
& \mathbb{P}(Y_{t+1} = l|X_{t+1} = k)\mathbb{P}(X_{t+1} = k|X_t = j) \\
=& \beta_{t+1}(k)B(k,l)A(j,k)
\end{aligned} \qquad (7)
$$

where,

$$
\begin{aligned}
A(i,j) &= \mathbb{P}[X_{t+1} = j | X_t = i] \\
B(i,j) &= \mathbb{P}[Y_t = j | X_t = i] \\
\pi_0(i) &= \mathbb{P}[X_0 = i]
\end{aligned} \tag{8}
$$

The following derivation of the Expectation and Maximization steps for likelihood in HMM are provided for reference and is from [4]

$$
\alpha_t(j)\beta_t(j) = \mathbb{P}(X_t = j, Y_0 = y_0, \ldots, Y_N = y_N) \tag{9}
$$

So

$$
d_t(j) = \mathbb{P}(X_t = j | Y_0 = y_0, \cdots, Y_N = y_N) = \frac{\alpha_t(j)\beta_t(j)}{\sum_j \alpha_t(j)\beta_t(j)} \tag{10}
$$

$$
\begin{aligned}
\alpha_t(i)A(i,y_{t+1})B(j,y_{t+1})\beta_{t+1}(j) =& \mathbb{P}(X_t = i, Y_0 = y_0, \ldots, Y_N = y_N) \\
& \mathbb{P}(X_{t+1} = j | X_t = i)\mathbb{P}(Y_{t+1} = y_{t+1} | X_{t+1} = j) \\
& \mathbb{P}(Y_{t+2} = y_{t+2}, \cdots, Y_N = y_N | X_{t+1} = j) \\
=& \mathbb{P}(X_{t+1} = j, X_t = i, Y_0 = y_0, \ldots, Y_N = y_N)
\end{aligned} \tag{11}
$$

Therefore,

$$
\begin{aligned}
e_t(i,j) =& \mathbb{P}(X_{t+1} = j, X_t = i | Y_0 = y_0, \ldots, Y_N = y_N) \\
=& \frac{\alpha_t(i)A(i,y_{t+1})B(j,y_{t+1})\beta_{t+1}(j)}{\sum_{i,j} \alpha_t(i)A(i,y_{t+1})B(j,y_{t+1})\beta_{t+1}(j)}
\end{aligned} \tag{12}
$$

Now we can find the maximum by the steps below.

$$
\begin{aligned}
& \max_\theta E[\log L_c(\theta) | Y_1, \cdots, Y_N; \tilde{\theta}] \\
=& \max_\theta \sum_i d_0(i) \log \pi_i + \sum_{i,j,k} \sum_{t=1}^{N} e_{t-1}(i,j,k) \log Ai, j^k + \sum_i \sum_{t=0}^{N} (d_t(i) - \frac{(y_t - \mu_i)^2}{2\sigma_i^2}) \\
& s.t. \sum_i \pi_i = 1, \sum_j A(i,j) = 1
\end{aligned} \tag{13}
$$

The closed form solution for the optimum parameters are:

$$
\pi_i^* = d_0(i), A(i,j)^* = \frac{\sum_{t=0}^{N-1} e_t(i,j,k)}{\sum_j \sum_{t=0}^{N-1} e_t(i,j,k)}, \mu_i^* = \frac{\sum_{t=0}^{N} d_t(i) y_t}{\sum_{t=0}^{N} d_t(i)}, \sigma_i^* = \frac{\sum_{t=0}^{N} d_t(i)(y_t - \mu_i^*)^2}{\sum_{t=0}^{N} d_t(i)} \tag{14}
$$

[4]

Implementation

The algorithm is implemented using the HiddenMarkov package for R [2].

Prediction of Next Day's Return Here we will predict the next day's stock return (log price ratio) using the Hidden Markov Model. The EM procedure produced estimates of which state the hidden process is in at the all times considered. We will predict the next day's return as the exepecation of the random variable Y_{t+1}. Using the assumed Markov property, we may calculate the marginal distribution of tomorrow's hidden state.

$$\pi_{t+1} = \pi_t \Pi \tag{15}$$

Given this distrbution of tomorrow's hidden state, we may calculate the expectation of tomorrow's daily return as the inner product of the hidden state distribution and the vector of Markov dependent means.

$$\mathbb{E}[Y_{t+1}] = \sum_{i \in S_X} \mathbb{E}[Y_{t+1} | X_{t+1} = i] \cdot \mathbb{P}[X_{t+1} = i] \tag{16}$$

Results

The software package chosen to perform the task of predicting stock returns is the R package *HiddenMarkov*. While some texts refer to R package *depmixS4*, this package was chosen because it is newer and because of its syntactic niceness. The package has capbilites for Markov modulated generalized linear models although these capbilities were not tested during the course of this project.

Code The following are pre processing functions used to import stock market historical quote data. This data was retrieved from the historical quote tool available at NASDAQ.com [3]. The pre processing function here calculates the log price ratio for each day and sorts by date ascending.

```
library(HiddenMarkov)
library(expm)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'expm'
```

```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
pre.process <- function(f.name){
  # Read a csv file of stock price data
  #   having columns: "open", "close", "date"
  # param f.name: string denoting file name to load
  x <- read.csv(f.name)
  x <- x[order(as.POSIXct(x$date)),] # Sort by date
  x$logdiff <- log(x$close) - log(x$open)
  return(x)
}
```

The Baum Welch algorithm, available in *HiddenMarkov*, uses EM algorithm to estimate most likely parameters for the HMM given the data.

```
bwelch <- function(hmm){
  # Perform Baum Welch algorithm
  #   for MLE of model parameters
  # param hmm: dthmm {HiddenMarkov} object
  hmm <- BaumWelch(hmm, control=bwcontrol(prt=F))
  return(hmm)
}
```

To instantiate the dthmm (discrete time hidden markov model) class, provide a time series of data and starting points for the model parameters.

```
get.hmm <- function(x, Tr=nrow(x), s=2){
  # Instantiate HMM object
  # param x: data frame with column 'logdiff'
  # param Tr: Truncation point
  #   (index of last observation included in training set)
  # params s: number of hidden Markov States
  Tr <- min(Tr, nrow(x))
  v <- x[1:Tr, "logdiff"]
  k <- 30.
  p <- 1 / k; q <- 1 - p
  A <- matrix(q, s, s)  # Initial guess at Markov Transition Matrix
  diag(A) <- p
  xi <- matrix(1 / s, 1, s)
  m1 <- mean(v[v < 0]); m2 <- mean(v[v >= 0]);
  s1 <- sd(v[v < 0]); s2 <- sd(v[v >= 0]);
  means <- c(m1,m2)
  stds <- c(s1,s2)
  cond.dist <- list(mean = means,
                    sd = stds)
  hmm <- dthmm(x = v, Pi = A, delta = xi,
               distn = 'norm', pm = cond.dist)
  hmm <- bwelch(hmm)
  return(hmm)
}

update.hmm <- function(hmm, xnew){
  # Update the Hidden Markov model with new observation.
  # Re-solve for parameters using Baum-Welch.
  # param hmm: dthmm {HiddenMarkov} object
  # param xnew: float. New observation to append
  hmm$x <- c(hmm$x, xnew)
  hmm <- bwelch(hmm)
  return(hmm)
}


forecast.hmm <- function(hmm, horizon=1){
  # Forecast tomorrow's return (log dollars)
  # param hmm: dthmm{HiddenMarkov} object
  # param horizon: int. how many steps ahead to forecast
  A <- hmm$Pi  # MLE of Markov Chain Transition Matrix
  Eyx <- hmm$pm$mean  # Expectation of Y given X
  t <- length(hmm$x)  # Series length
```

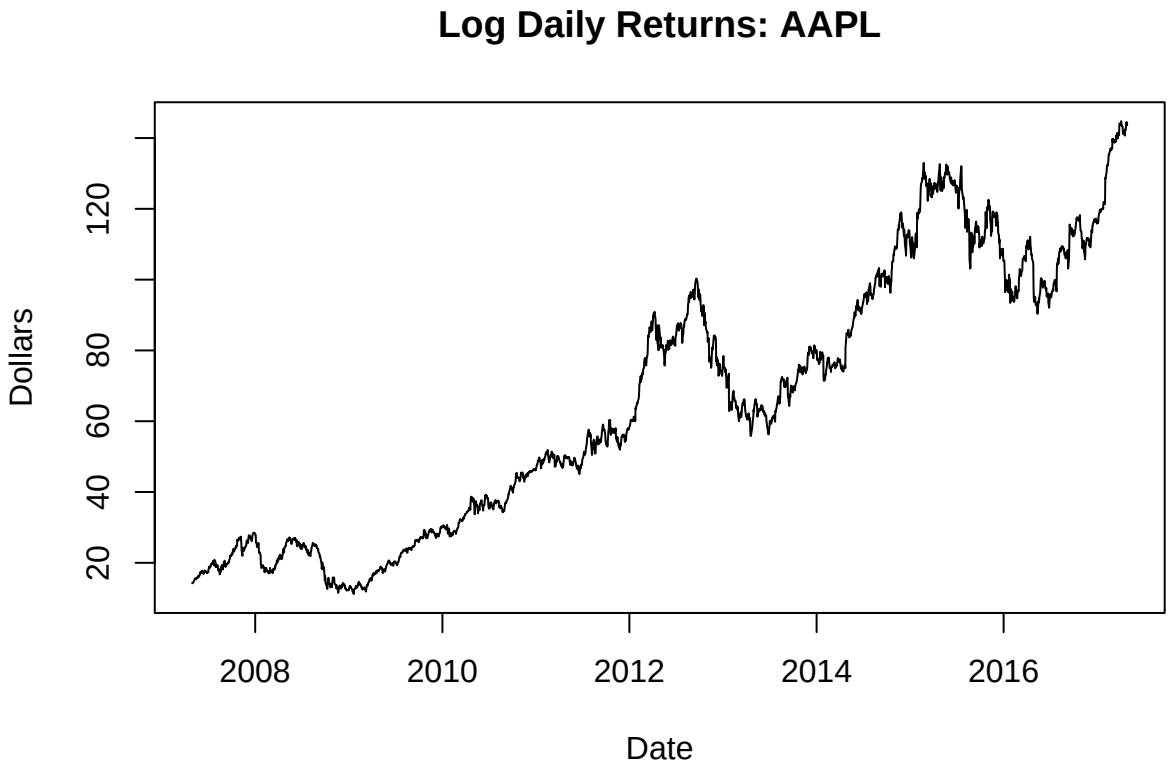
```
  px.T0 <- hmm$u[t,]  # Estimated distribution of Xt
  Ah <- A %^% horizon    # Exponentiate MC transition matrix
  px.Th <- px.T0 %*% Ah  # Estimated distribution of X(t+h)
  Ey.Th <- px.Th %*% Eyx  # Expectation of Y(t+h)
  return(Ey.Th)
}


plot.conditional.dist <- function(hmm){
  # Plot the conditional distributions
  #   of Y given hidden state X
  # param hmm: dthmm {HiddenMarkov} object
  m1 <- hmm$pm$mean[1]
  m2 <- hmm$pm$mean[2]
  s1 <- hmm$pm$sd[1]
  s2 <- hmm$pm$sd[2]
  xlim <- c(min(m1 - 3  * s1,
                m2 - 3  * s2),
            max(m1 + 3  * s1,
                m2 + 3  * m2))
  x <- seq(xlim[1],xlim[2],.001)
  y1 <- dnorm(x, m1, s1)
  y2 <- dnorm(x, m2, s2)
  ymax <- max(max(y1), max(y2))
  plot(x, y1, type="l", col="red",
       ylim=c(0,ymax), main="Conditional Distribution: P[Y|X]",
       ylab="Probability Density", xlab="y")
  abline(v = m1, col='red')
  lines(x, y2, col="blue")
  abline(v = m2, col="blue")
  legend(xlim[1], ymax / 2, c('X=1', 'X=2'), lty=c(1,1),
         lwd=c(2,2), col=c('red','blue'))
}


stock.timeseries.plot <- function(x){
  # Plot the stock's value over time as
  #   well as its daily log returns
  # param x: list of data frames with columns "logdiff" and "close"
  # param symb: stock ticker symbol
  sym <- names(x)
  for(name in sym){
    plot(close ~ as.POSIXct(date), x[[name]], type="l",
         xlab="Date", ylab="Dollars",
         main=paste("Log Daily Returns:", name))
  }
}
```

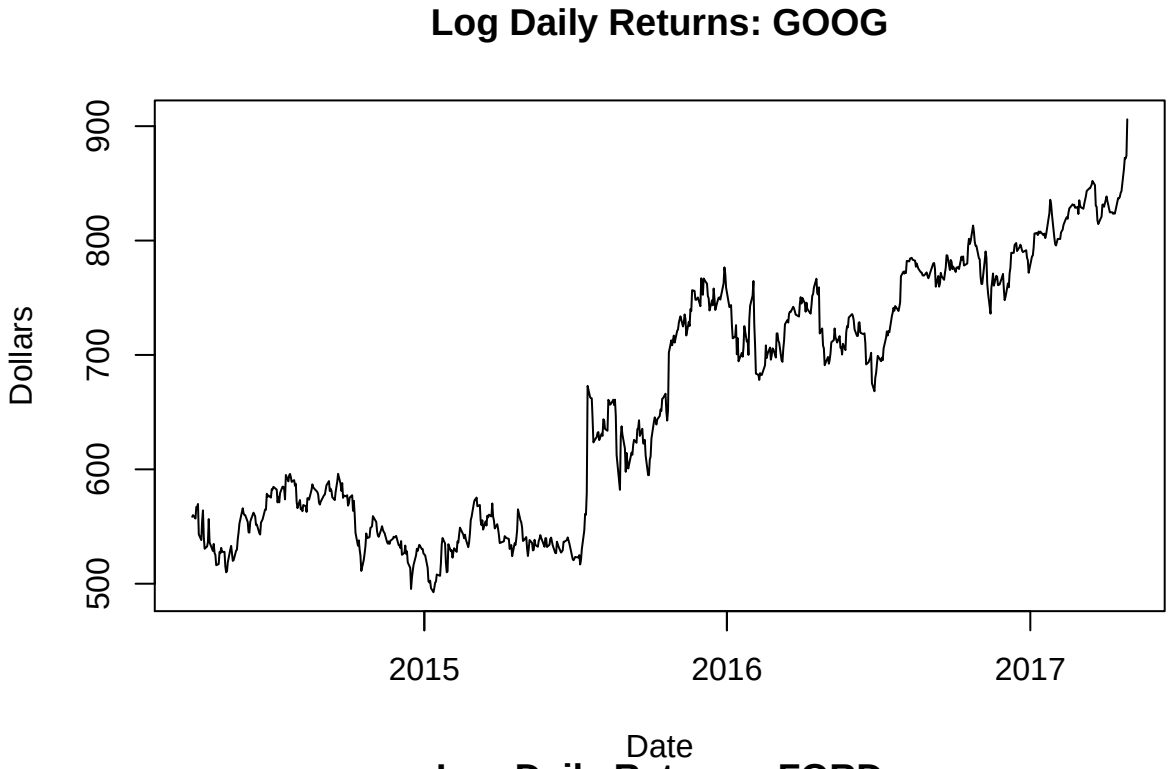
Demonstration

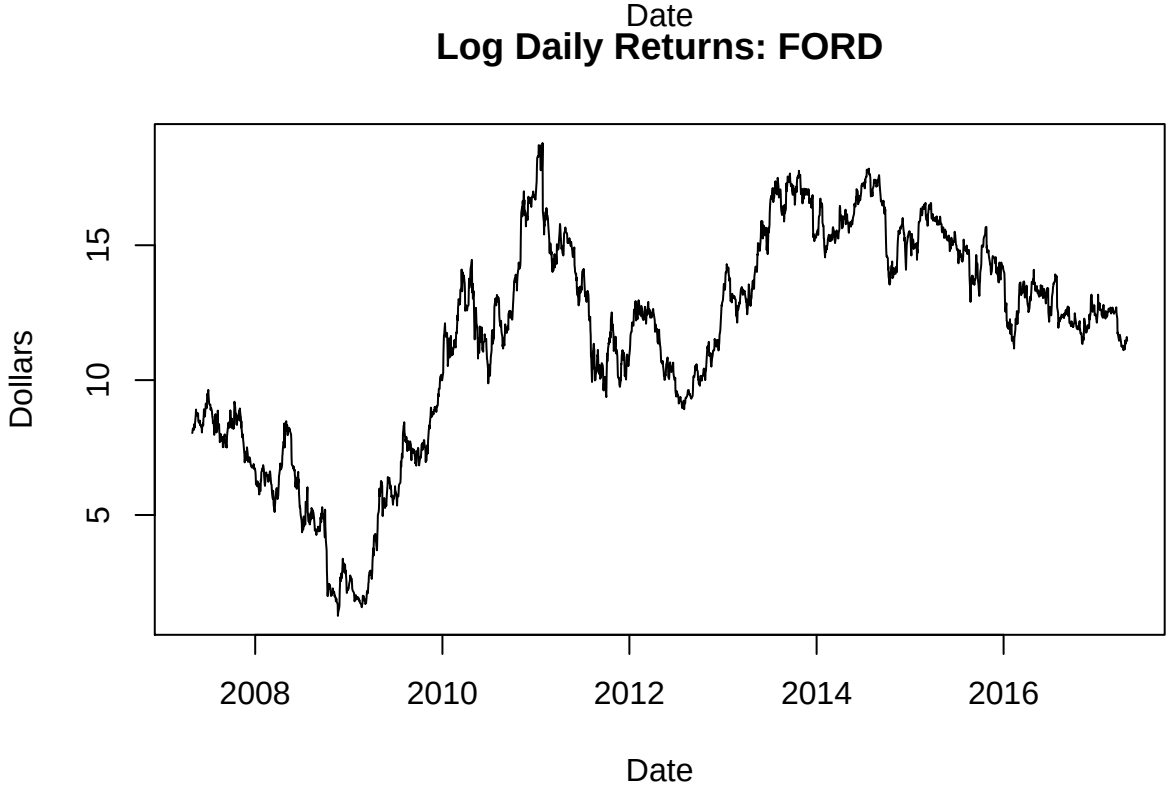
To demonstrate the concept, here we will instantiate a HMM object and train it on a time series of data. Then we will show the estimates for the conditional distribution of Y_i given state of X_i. On the density plots, note how the two states different in both mean and variance. One state is associated with high volatililty

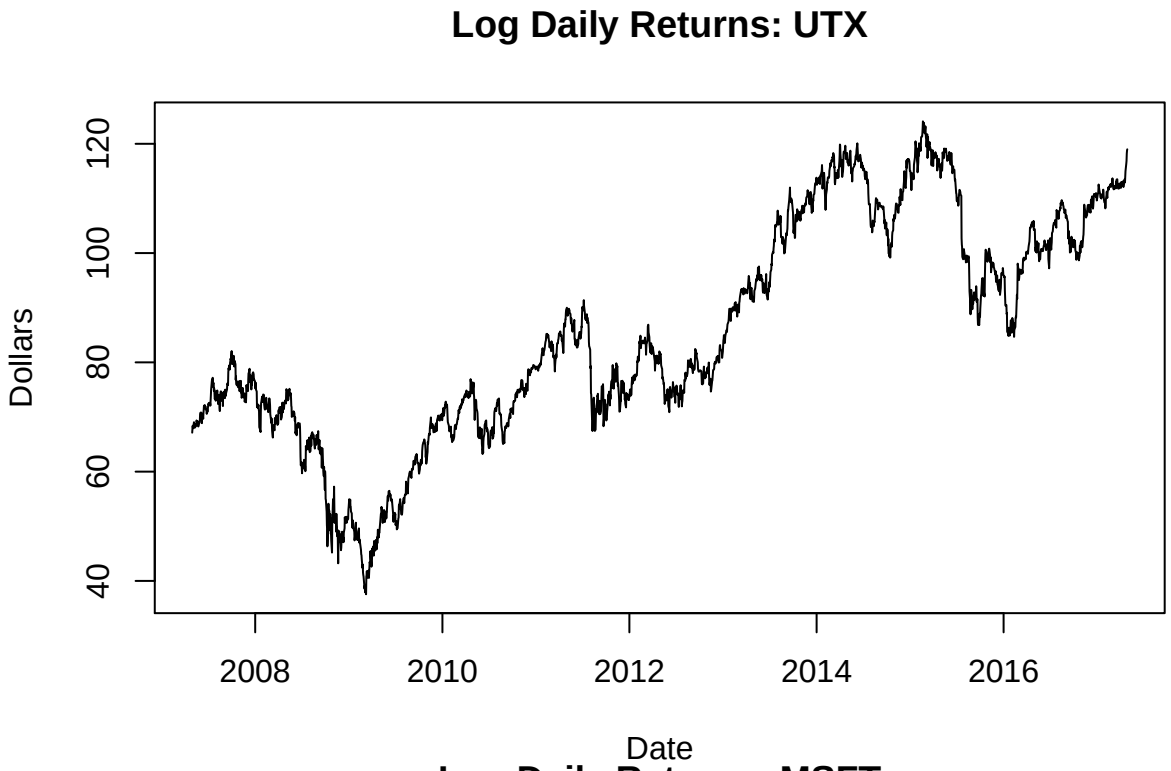
(high variance in log returns) and one state shows lower volatility (lower variance in log returns).

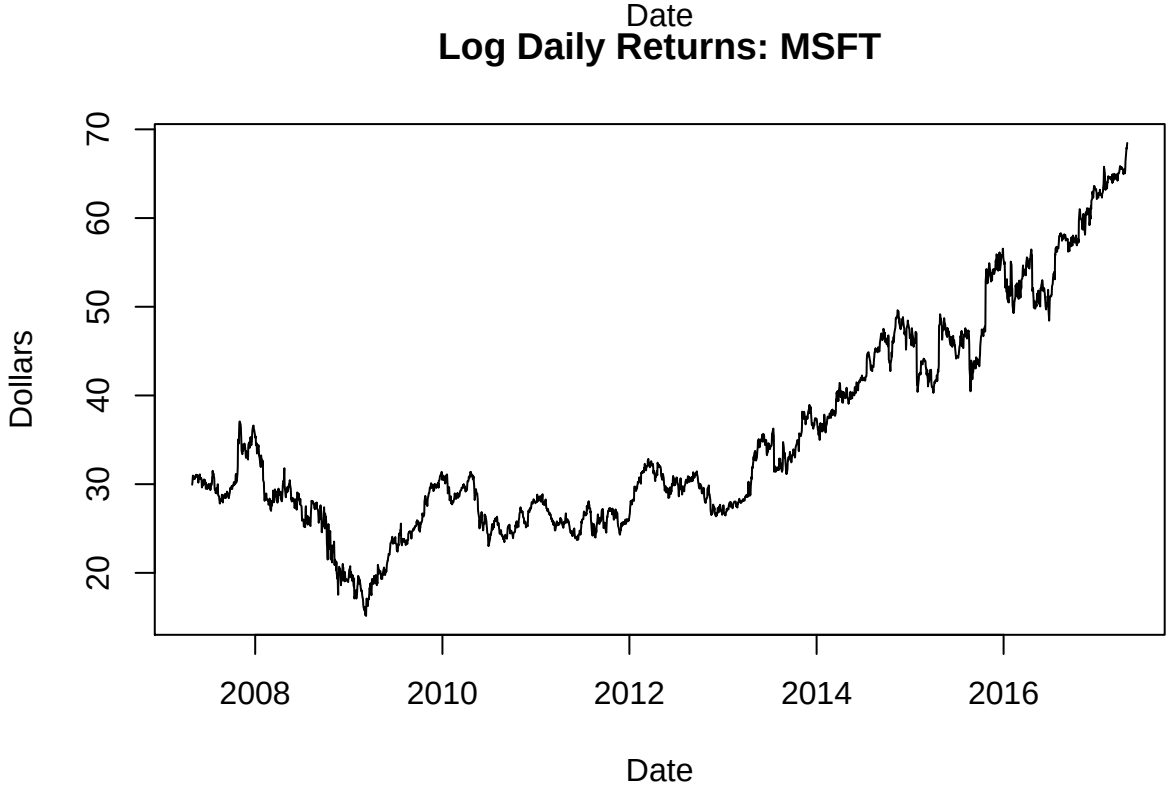
```
# Read stock data (date, open, close, volume)
aapl <- pre.process("./stock_data/aapl_decade.csv")
goog <- pre.process("./stock_data/goog_decade.csv")
ford <- pre.process("./stock_data/ford_decade.csv")
utx <- pre.process("./stock_data/utx_decade.csv")
msft <- pre.process("./stock_data/msft_decade.csv")
sbux <- pre.process("./stock_data/sbux_decade.csv")
jnj <- pre.process("./stock_data/jnj_decade.csv")

# Examine the data as a plot of price and of log returns
stocklist <- list(AAPL=aapl, GOOG=goog, FORD=ford,
                  UTX=utx, MSFT=msft, SBUX=sbux, JNJ=jnj)
stock.timeseries.plot(stocklist)
```

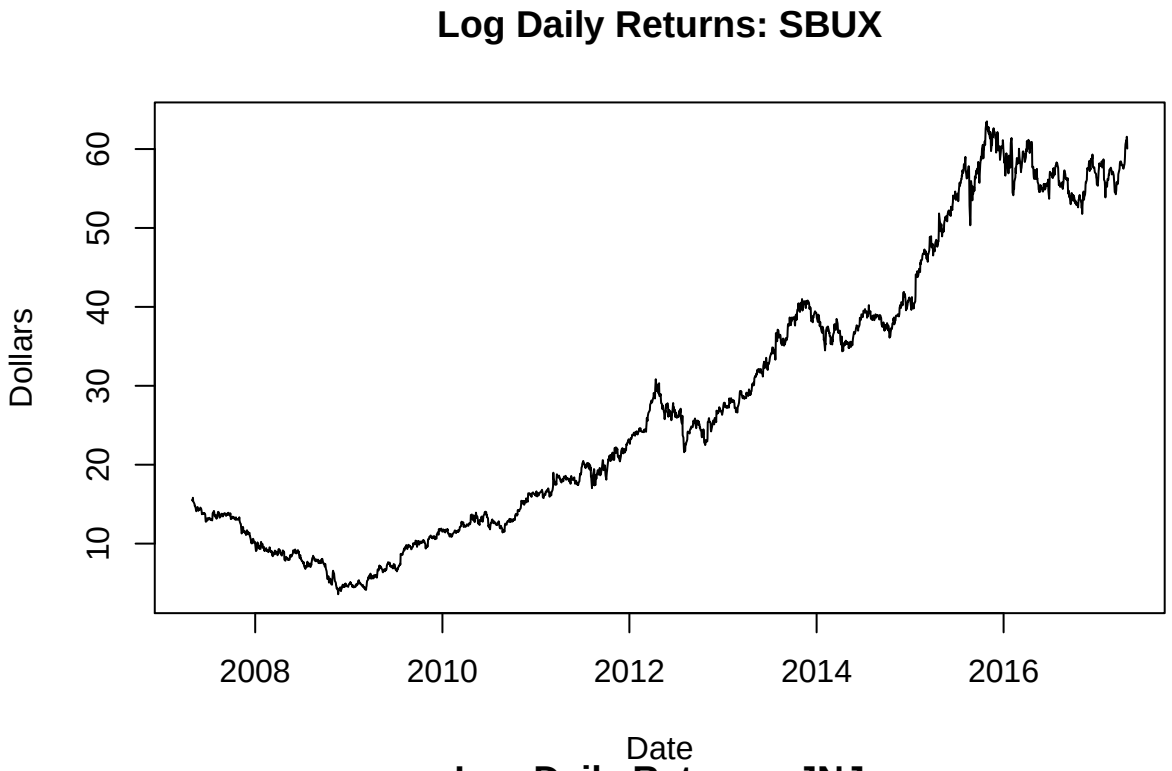


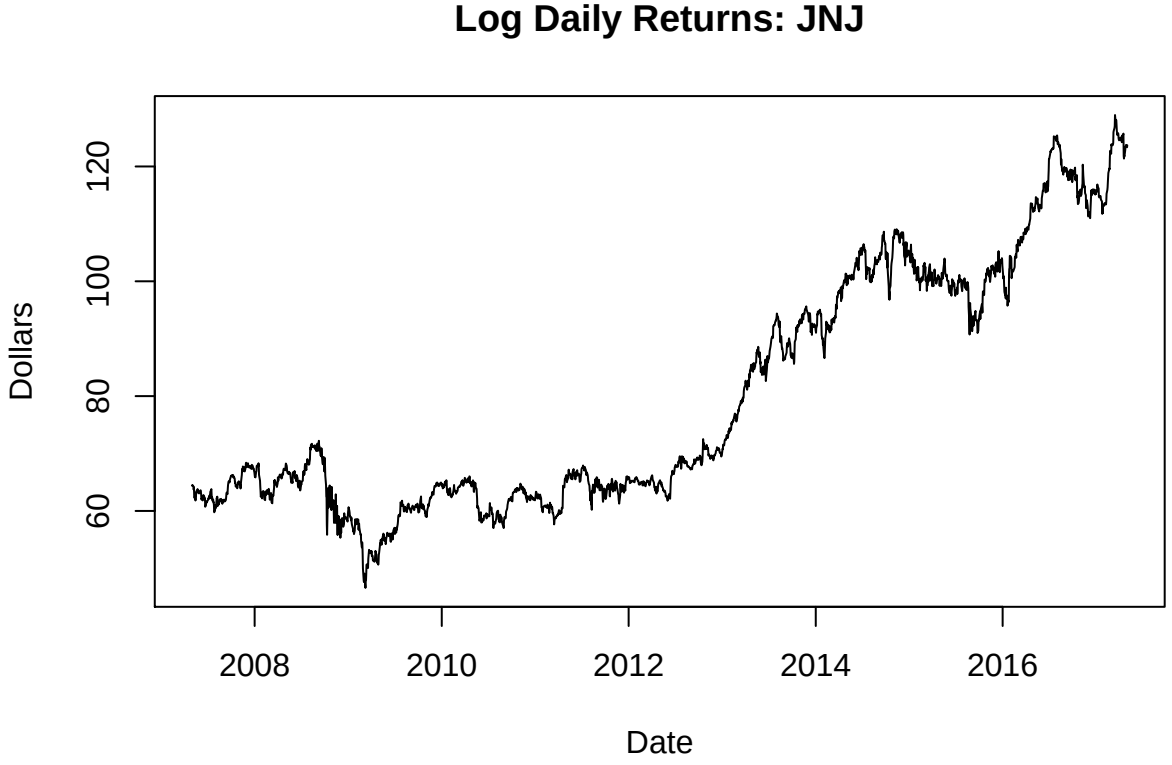

Log Daily Returns: GOOG
Dollars
500
600
700
800
900
2015
2016
2017
Date


Log Daily Returns: FORD
Dollars
5
10
15
2008
2010
2012
2014
2016
Date


Log Daily Returns: UTX
Dollars
40
60
80
100
120
2008
2010
2012
2014
2016
Date


Log Daily Returns: MSFT
Dollars
20
30
40
50
60
70
2008
2010
2012
2014
2016
Date





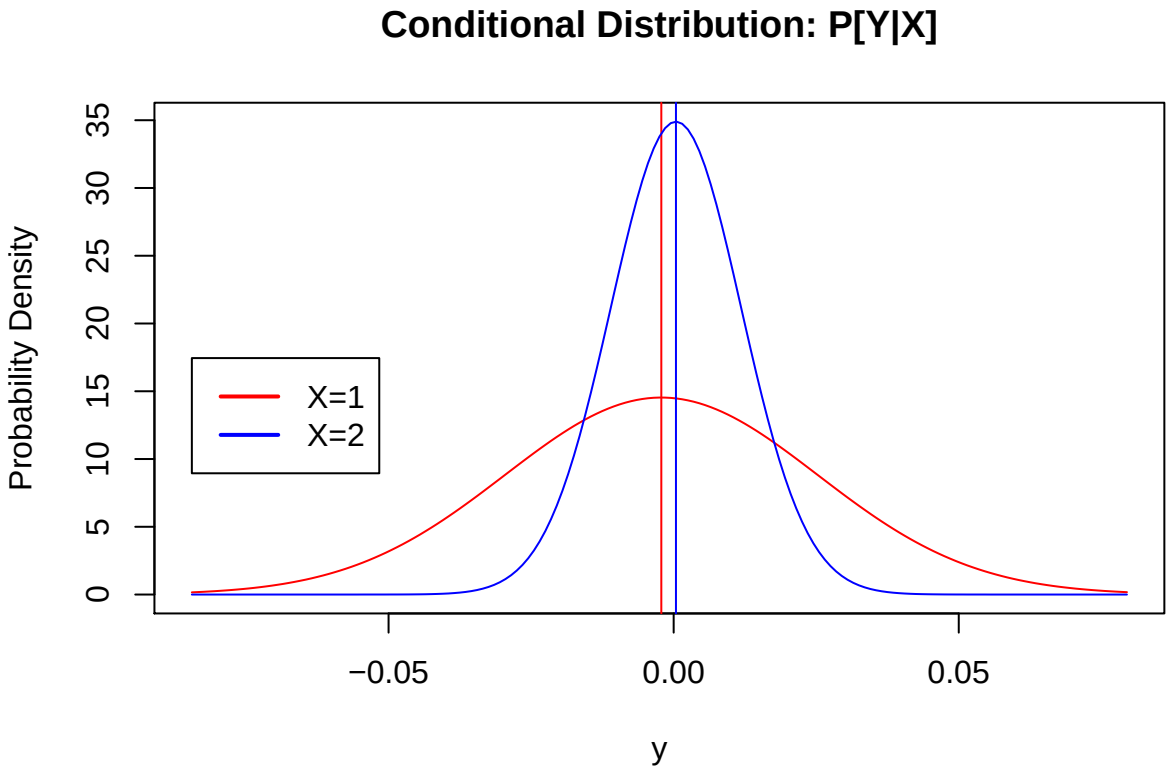

```
# Fit the HMM
hmm <- get.hmm(aapl)

# List the estimated conditional means
hmm$pm$mean
```

```
## [1] -0.0021680570  0.0004014585
```

```
# List the estimated conditional standard deviations
hmm$pm$sd
```

```
## [1] 0.02744515 0.01143210
```

```
# Plot conditional distribution of Y given X (EM estimated parameters)
# The model posits that these are (most likely) the distributions
# that Y (log returns) come from given the underlying state of the market.
plot.conditional.dist(hmm)
```



Sequential Prediction

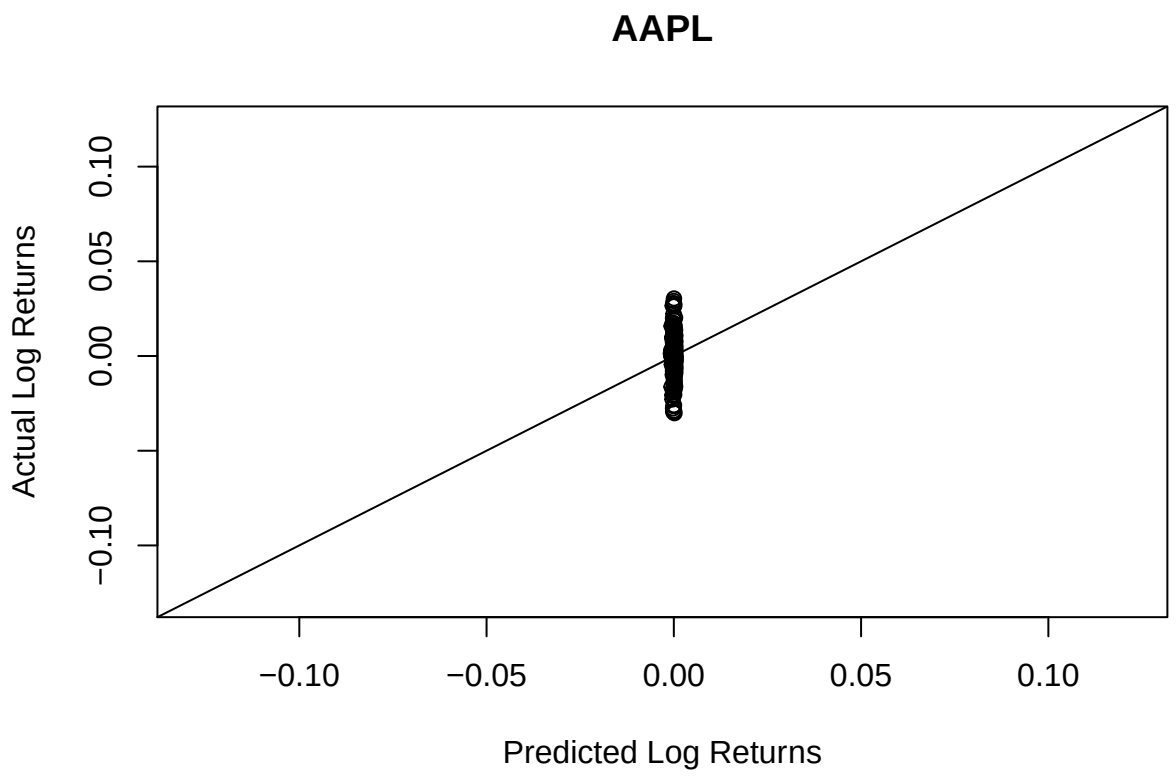
Now we will sequentially predict the next day's returns given a HMM trained on all previous days returns. Then we will define a trading strategy based on the prediction for next day's return. As we will see in the discussion section, the choice of trading strategy (how one takes into account the HMM forecasts) may be just as important as making the forecasts themselves. Here, the trading strategy simply takes a long position if the next day expected return is positive and takes a short position if the next day expected return is negative. The strategy is rudimentary and does not take into account the fixed cost of transacting shares through a broker. It also does not take into account the variance of the prediciton (predicted volatility) when trading. This is just one strategy that could be used and there are many other possible strategies of trading, given predictions of the mean and variance of the next day's returns.

```
employ.trading.strategy <- function(x){
  # Investing rule given the predicted next day returns
  # param x: data frame containing column 'predicted'
  # Creates column "position" representing the number
  #   of shares that a simulated trader would own using
  #   a this trading strategy
  N <- nrow(x)
```

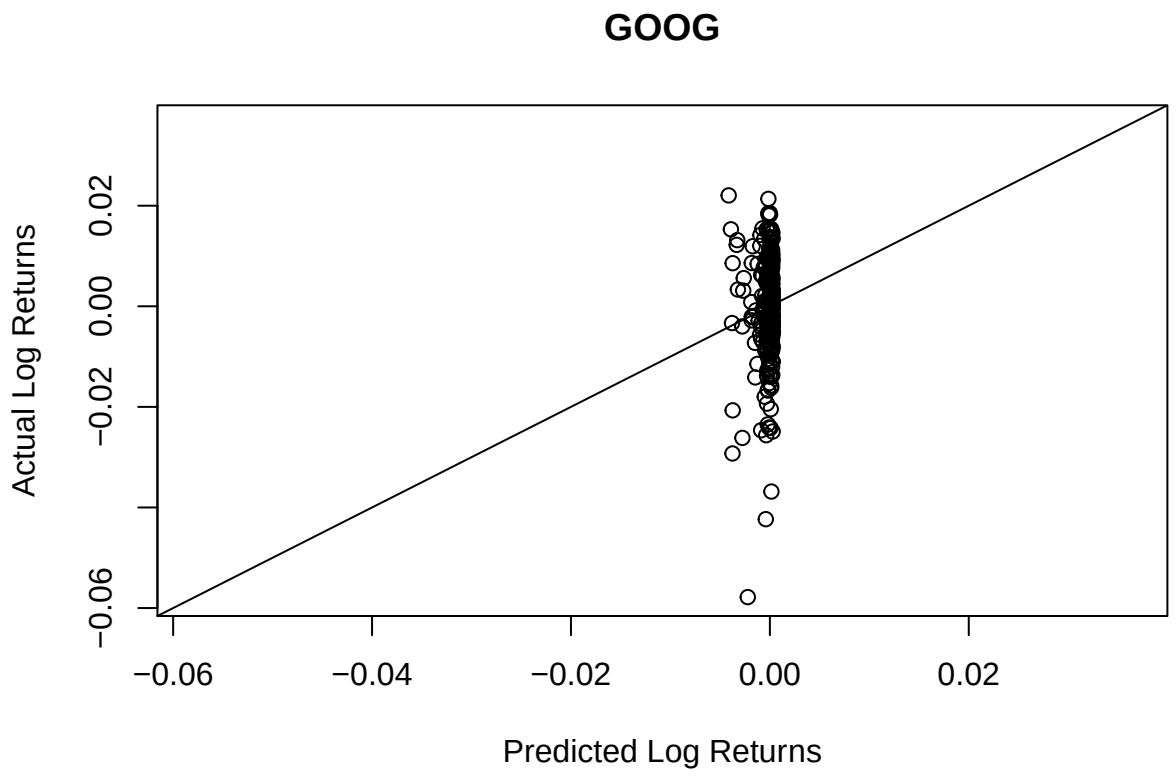
```
  x$position <- rep(0, N)
  a <- min(which(!is.na(x$predicted)))
  for(i in a:N){
    x$position[i] <- 0 + (x$predicted[i] > 0) - (x$predicted[i] < 0)
  }

  return(x)
}


overall.return <- function(x){
  # Compute the overall return from
  #   using the HMM trading strategy
  # param x: data frame containing
  #   columns "open", "close", and "position"
  N <- nrow(x)
  start_ <- min(which(!is.na(x$predicted)))
  D <- 0                  # Initial cash amount
  holdings <- x$position  # How many shares owned, by day
  price <- x$open         # Actual open price
  for(t in start_:N){
    shares <- holdings[t] - holdings[t - 1]
    D <- D - shares * price[t]
  }
  return(D)
}


sequentially.forecast.hmm <- function(x, days, sym){
  # Forecast the log returns for day t using information
  # contained only in days 1 through t - 1
  # param x: dataframe with column <column="logdiff">
  # param sym: string denoting Stock ticker symbol
  N <- nrow(x)
  Tr <- N - days  # Truncation point
  x$predicted <- rep(NA, N)
  for(t in Tr:(N - 1)){
    if(t == Tr){
      hmm <- get.hmm(x, Tr = t)
    } else {
      xnew <- x$logdiff[t]
      hmm <- update.hmm(hmm, xnew)
    }
    x$predicted[t + 1] <- forecast.hmm(hmm)
  }
  x <- employ.trading.strategy(x)
  return(x)
}
```

```
aapl <- sequentially.forecast.hmm(aapl, days=332)
goog <- sequentially.forecast.hmm(goog, days=332)
ford <- sequentially.forecast.hmm(ford, days=332)
utx <- sequentially.forecast.hmm(utx, days=332)
msft <- sequentially.forecast.hmm(msft, days=332)
```

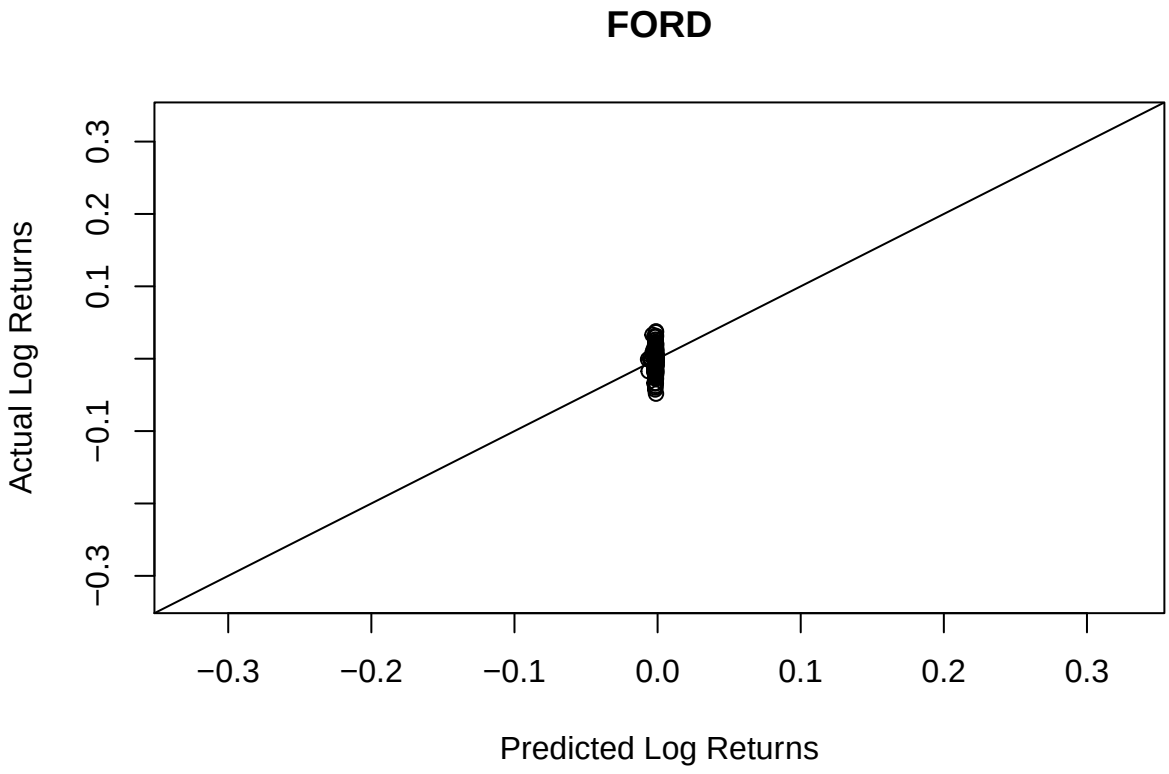
```r
sbux <- sequentially.forecast.hmm(sbux, days=332)
jnj <- sequentially.forecast.hmm(jnj, days=332)

forecast.scatter <- function(df, sym=NULL){
  # Summarize the sequential forecasting
  #   using a actual by predicted plot
  # param df: dataframe with columns "logdiff" and "predicted"
  x <- df$predicted
  y <- df$logdiff
  x <- x[!is.na(x)]
  y <- y[!is.na(x)]
  axis.min <- min(min(x), min(y))
  axis.max <- max(max(x), max(y))
  ax.lim <- c(axis.min,
              axis.max)
  plot(logdiff~predicted, df,
       xlim=ax.lim, ylim=ax.lim,
       xlab="Predicted Log Returns",
       ylab="Actual Log Returns",
       main=sym)
  abline(0,1)
}


prediction.cor <- function(x){
  # Calculate squared correlation between
  #   predictions and actual returns
  y <- x$logdiff
  z <- x$predicted
  y <- y[!is.na(z)]
  z <- z[!is.na(z)]
  return(cor(y, z))
}

forecast.scatter(aapl, "AAPL")
```



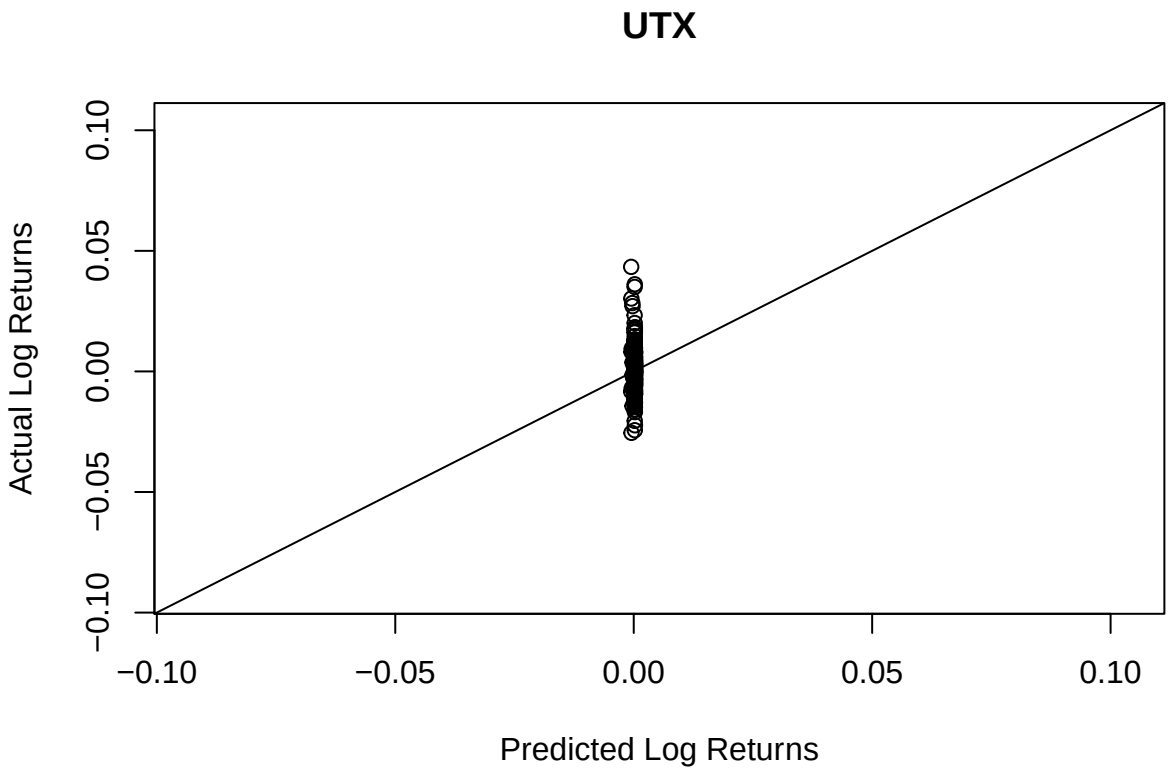

```
forecast.scatter(goog, "GOOG")
```



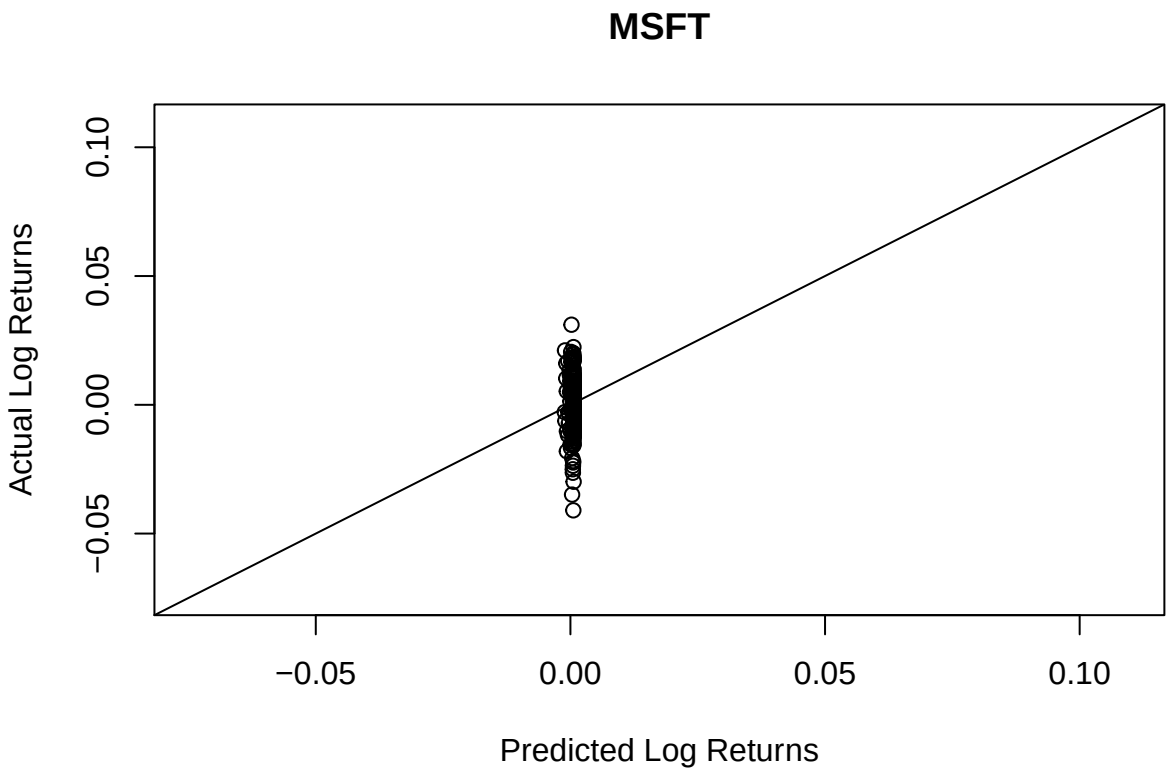

```
forecast.scatter(ford, "FORD")
```



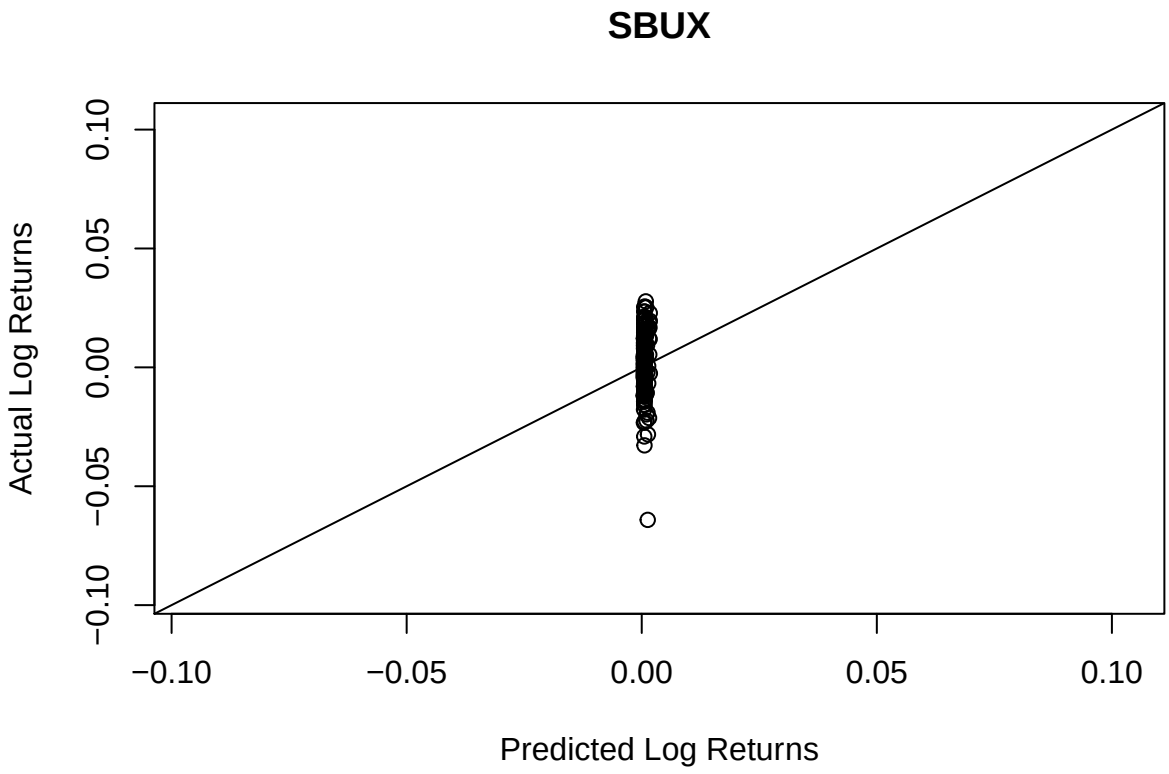

```
forecast.scatter(utx, "UTX")
```



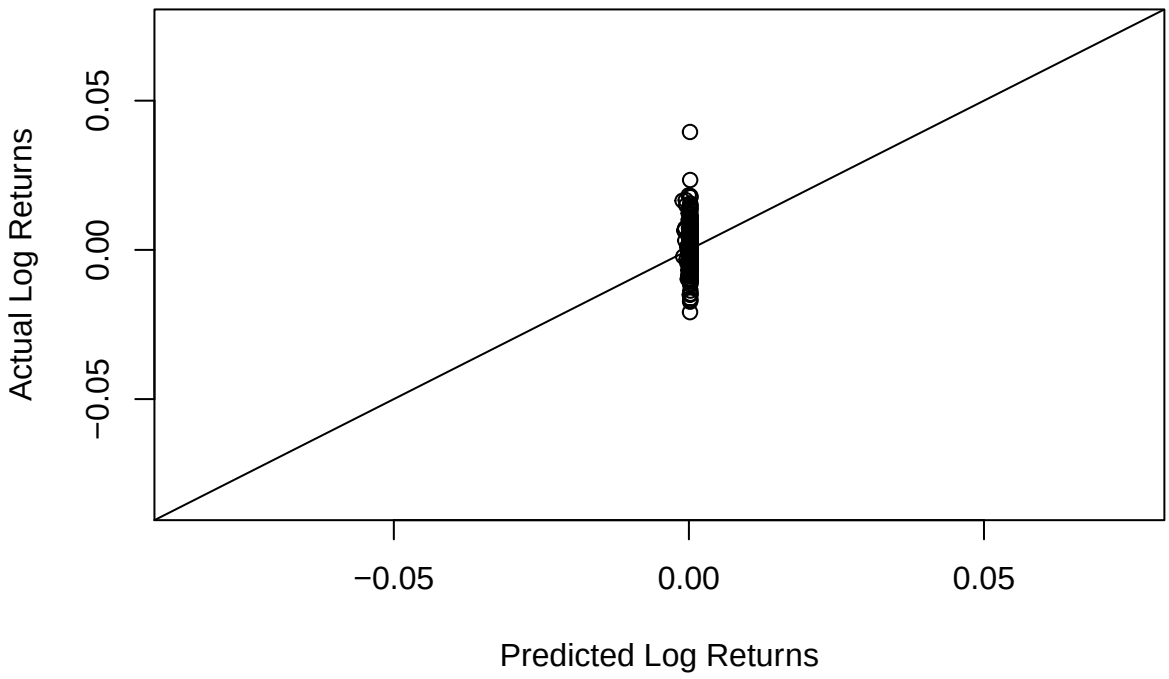

```
forecast.scatter(msft, "MSFT")
```




```
forecast.scatter(sbux, "SBUX")
```




```
forecast.scatter(jnj, "JNJ")
```




```
prediction.cor(aapl)
```

```
## [1] 0.04004898
```

```
prediction.cor(goog)
```

```
## [1] 0.06001654
```

```
prediction.cor(ford)
```

```
## [1] 0.01466092
```

```
prediction.cor(utx)
```

```
## [1] -0.1866004
```

```
prediction.cor(msft)
```

```
## [1] -0.0654455
```

```
prediction.cor(sbux)
```

```
## [1] 0.06420625
```

```
prediction.cor(jnj)
```

```
## [1] -0.1502632
```

Correllation

In some cases the correlation between actual and predicted returns is low but positive. In other cases it is negative. The model, with its current structure, does not predict the stock price well.

Another way to assess the value of these predictions would be to simulate the returns from using an investment strategy based on the predictions. Here we will calculate the returns expected from the rudimentary investment strategy defined above. The result of this simulation is a net profit relative to the first day of investing, 332 days ago.

```
overall.return(aapl)
```

```
## [1] -115.75
```

```
overall.return(goog)
```

```
## [1] -620.43
```

```
overall.return(ford)
```

```
## [1] 13.97
```

```
overall.return(utx)
```

```
## [1] -107.88
```

```
overall.return(msft)
```

```
## [1] -55.31
```

```
overall.return(sbux)
```

```
## [1] -58.79
```

```
overall.return(jnj)
```

```
## [1] -107.51
```

Discussion

In this forecasting endeavor, we found that it is difficult to make accurate predictions about the next day's returns. On the metric of mean squared error or squared correlation between predictions and actual results, the HMM did not preform exceptionally well. However, given that the purpose of such an algorithm would be to make better trading decisions *on average* than other algorithms, the model should be judged on its simulated returns. One would need knowledge of what other prediction and trading algorithms exist in order to tell if this HMM model better. In this project, the predictions and the trading strategy used did not provide a reliable way to always make money on any stock. One stock appears to have positive simulated return and on the rest we would lose money.

Future Work

The HMM provides prediction of both expectation and variance of future returns. Since one of the conditional distributions of Y has higher variance than the other, the best trading strategy must take into account this uncertainty in the forecasts. Perhaps some quantile of the forecast distribution would be an appropriate statistic on which to base trading decisions. Future work could also include knowledge of covariates in the market such as trading volume and other factors that may indicate hidden market state.

References

[1] Zucchini, Walter. Iain L. MacDonald, Ronald Langrock. *Hidden Markov Models for Time Series* An Introduction Using R. CRC Press. Boca Raton, FL. 2016

[2] Harte, David. *HiddenMarkov: Hidden Markov Models*. R package version 1.8-7. Statistics Research Associates. Wellington. ftp://ftp.gns.cri.nz/pub/davidh/sslib/r-repo/. 2016

[3] NASDAQ historical stock quote tool.http://www.nasdaq.com/quotes/historical_quotes.aspx

[4] Lee, Joohyung. Shin, Minyong. *Stock Forecasting using Hidden Markov Processes* http://cs229.stanford.edu/proj2009/ShinLee.pdf.2009