Predicting Stock Market Returns

L. Torgo

ltorgo@fc.up.pt
Faculdade de Ciências / LIAAD-INESC TEC, LA
Universidade do Porto

Aug, 2017

Problem Description

The Context

- Stock market trading is an application area with a large potential for data mining
- Huge amounts of data (in several formats) are available
- Still, there are researchers claiming the impossibility of making money out of forecasting future prices - the *efficient markets hypothesis*
- The goal of trading is to maintain a portfolio of assets based on buy and sell orders, and achieve profit with this

Problem Description (2)

The Concrete Application

- We will use data mining in a slightly more specific trading context
- We will trade a single security - the S&P 500 market index
- Given **historic prices data** of this security and an **initial capital** we will try to maximize our profit over a future testing period **by means of trading actions** - buy, sell, hold

NYU | STERN
MS in Business Analytics

Problem Description (3)

The Concrete Application (cont.)

- Our trading strategy will **base the decisions on the results of a data mining process**
- This process will try to forecast the future evolution of prices based on historical data
- The overall evaluation criteria will be the profit/loss resulting from the trading actions

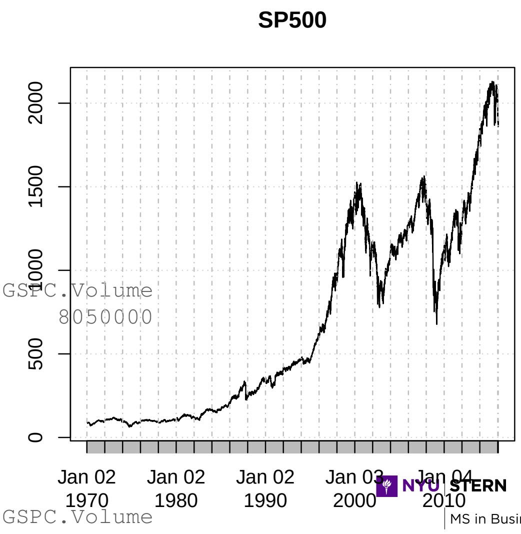
NYU | STERN
MS in Business Analytics

The Data

- We will use a data set available in package `DMwR2`

```
library(xts)  # extra package to install

## Loading required package:  zoo
##
## Attaching package:  'zoo'
## The following objects are masked from
'package:base':
##
##    as.Date, as.Date.numeric

data(GSPC,package="DMwR2")
first(GSPC)

##            GSPC.Open GSPC.High GSPC.Low GSPC.Close GSPC.Volume
## 1970-01-02     92.06     93.54    91.79         93     8050000
##            GSPC.Adjusted
## 1970-01-02            93

last(GSPC)

##            GSPC.Open GSPC.High GSPC.Low GSPC.Close GSPC.Volume
## 2016-01-25   1906.28   1906.28  1875.97    1877.08  4401380000

dim(GSPC)
```

```
plot(GSPC$GSPC.Close,main="SP500")
```



Obtaining Further Prices Data

- The package `quantmod` has facilities for getting more data from the web

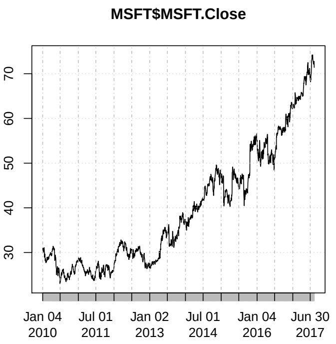
```
library(quantmod)  # extra package you need to install
getSymbols('MSFT',from='2010-01-01')
## If the above fails (Yahoo recent restrictions) try:
## getSymbols.google('MSFT',from='2010-01-01',env=.GlobalEnv)
getFX("USD/EUR")
getMetals("Gold")
getFinancials("AAPL")  # Use viewFin() to view the downloaded financial data
```

```
## [1] "MSFT"
## [1] "USDEUR"
## [1] "XAUUSD"
## [1] "AAPL.f"
```

Obtaining Further Prices Data (2)

```
## [1] "MSFT"
```

```r
plot(MSFT$MSFT.Close)
```




```r
viewFin(AAPL.f)
```

```
## Annual Balance Sheet for AAPL

##                                           2016-09-24 2015-09-26
## Cash & Equivalents                                NA         NA
## Short Term Investments                      58554.00   30212.00
## Cash and Short Term Investments             67155.00   41601.00
## Accounts Receivable - Trade, Net            15754.00   16849.00
## Receivables - Other                               NA         NA
## Total Receivables, Net                      29299.00   30343.00
## Total Inventory                              2132.00    2349.00
## Prepaid Expenses                                  NA         NA
## Other Current Assets, Total                  8283.00   15085.00
## Total Current Assets                       106869.00   89378.00
## Property/Plant/Equipment, Total - Gross     61245.00   49257.00
## Accumulated Depreciation, Total            -34235.00  -26786.00
## Goodwill, Net                                5414.00    5116.00
## Intangibles, Net                             3206.00    3893.00
## Long Term Investments                      170430.00  164065.00
## Other Long Term Assets, Total                8757.00    5422.00
## Total Assets                               321686.00  290345.00
## Accounts Payable                            37294.00   35490.00
```



```
69.00
## Notes Payable/Short Term Debt                8105.00    8499.00
## Current Port. of LT Debt/Capital Leases      3500.00    2500.00
## Other Current liabilities, Total             9156.00    9952.00
10.00
29.00
NA
29.00
28.00
62.00
NA
89.00
90.00
NA
NA
16.00
NA
84.00
NA
17.00
55.00
45.00
NA
78.75
09-28
NA
41.00
46.00
02.00
NA
41.00
64.00
NA
35.00
86.00
19.00
22.00
77.00
79.00
15.00
46.00
00.00
67.00
```


Defining the Predictive Task

What to Predict?

- To make a proper decision concerning our current position we need to be able to anticipate the future trend of the prices
- The following details our approach:
  - If prices vary more than *p*% we consider that worthwhile for trading
  - We want to forecast if this margin is attainable in the next *k* days - note that prices may go up and down during these *k* days
  - This is different from predicting the price for a certain future time tag
  - What we want is a good prediction of the general tendency of the prices in the next *k* days

NYU STERN
MS in Business Analytics


```
## Accrued Expenses                             7689.00    4782.00
```

What to Predict? (2)

- We will propose a variable, calculated with the quotes data, that reflects the tendency of the prices in a set of days
- We will try to forecast the future value of this variable
- Positive values of this tendency indicator will lead us to buy, whilst negative values will lead us to sell

NYU | STERN
MS in Business Analytics

What to Predict? (3)

- Let the daily average price be approximated by:

$$\bar{P}_i = \frac{C_i + H_i + L_i}{3}$$

- Let V_i be the set of k percentage variations of today's close to the following k days average prices (often called arithmetic returns):

$$V_i = \left\{ \frac{P_{i+j} - C_i}{C_i} \right\}_{j=1}^{k}$$

- The proposed indicator is the sum of the variations whose absolute value is above our target margin $p\%$:

$$T_i = \sum_v \{v \in V_i : v > p\% \vee v < -p\%\}$$

NYU | STERN
MS in Business Analytics

What to Predict? (4)

- The following function implements the proposed indicator:

```
T.ind <- function(quotes,tgt.margin=0.025,n.days=10) {
   v <- apply(HLC(quotes),1,mean)
   v[1] <- Cl(quotes)[1]          # correction to the book!

   r <- matrix(NA,ncol=n.days,nrow=NROW(quotes))
   for(x in 1:n.days) r[,x] <- Next(Delt(v,k=x),x)

   x <- apply(r,1,function(x)
                   sum(x[x > tgt.margin | x < -tgt.margin]))
   if (is.xts(quotes)) xts(x,time(quotes)) else x
}
```

NYU | STERN
MS in Business Analytics

Inspecting the Values of the *T* Indicator

```
library(quantmod)
data(GSPC,package="DMwR2")
candleChart(last(GSPC,'3 months'),theme='white',TA=NULL)
avgPrice <- function(p) apply(HLC(p),1,mean)
addAvgPrice <- newTA(FUN=avgPrice,col=1,legend='AvgPrice')
addT.ind <- newTA(FUN=T.ind,col='red',legend='tgtRet')
addAvgPrice(on=1)
addT.ind()
```

STERN
MS in Business Analytics

What to Predict? - summary

- We will forecast the value of the T indicator using data mining models
- If the predicted value is above a certain threshold we will buy our asset
- If the predicted value is below a certain threshold we will sell our asset
- Otherwise we will just hold our current position

Which Predictors to Use?

- Which information should we give to our models (in the form of predictors) to obtain good predictions of the T indicator?
- The main assumption behind trying to forecast the future behavior of financial markets is that it is possible to do so by observing the past behavior of the market
- More precisely, that if in **the past behavior** p **was followed by** f, and **this pattern occurs frequently**, then if we are observing again p **we are confident that** f **will follow**

Which Predictors to Use? (2)

- We are approximating the future behavior using our T indicator
- We need to decide how to describe the recent past behavior of the prices
- We will try to collect a series of indicators that capture the recent dynamics of the prices

Which Predictors to Use? (3)

- Obvious candidates are the recent prices of the asset
- We will focus on the Closing prices, more precisely on the arithmetic h-days returns:

$$R_t^h = \frac{C_t - C_{t-h}}{C_{t-h}}$$

Which Predictors to Use? (4)

- Additional information can be given by calculating relevant statistics on the **recent evolution of the prices**
- Technical indicators are **numeric summaries** that reflect some **properties of the price time series**
- We will select an illustrative set of technical indicators calculated with the recent prices and use them as predictors for our models
  - Package `TTR` contains a huge sample of technical indicators

Which Predictors to Use? (5)

- Auxiliary functions we will use to obtain the predictors

```
library(TTR)
myATR        <- function(x) ATR(HLC(x))[,'atr']
mySMI        <- function(x) SMI(HLC(x))[, "SMI"]
myADX        <- function(x) ADX(HLC(x))[,'ADX']
myAroon      <- function(x) aroon(cbind(Hi(x),Lo(x)))$oscillator
myEMV        <- function(x) EMV(cbind(Hi(x),Lo(x)),Vo(x))[,2]
myMACD       <- function(x) MACD(Cl(x))[,2]
myMFI        <- function(x) MFI(HLC(x),  Vo(x))
mySAR        <- function(x) SAR(cbind(Hi(x),Cl(x))) [,1]
myVolat      <- function(x) volatility(OHLC(x),calc="garman")[,1]
```

The Prediction Task and Data We Will Use

- The following code creates the objects with the data we will use

```
data.model <- specifyModel(T.ind(GSPC) ~ myATR(GSPC) + mySMI(GSPC) +
                  myADX(GSPC) + myAroon(GSPC) + myEMV(GSPC) +
                  myVolat(GSPC) + myMACD(GSPC) + myMFI(GSPC) + mySAR(GSPC) +
                  runMean(Cl(GSPC)) + runSD(Cl(GSPC)))

Tdata.train <- as.data.frame(modelData(data.model,
                                       data.window=c('1970-01-02','2005-12-30')))
Tdata.eval <- na.omit(as.data.frame(modelData(data.model,
                                       data.window=c('2006-01-01','2016-01-25'))))
Tform <- as.formula('T.ind.GSPC ~ .')   # the formula to be used in models
```


The Prediction Task and Data We Will Use (2)

```
head(Tdata.train)

##            T.ind.GSPC myATR.GSPC mySMI.GSPC myADX.GSPC myAroon.GSPC
## 1970-02-18 0.15215888   1.757594 -27.544696   48.67410          -30
## 1970-02-19 0.05307516   1.757766 -22.066236   45.56942          -30
## 1970-02-20 0.05120619   1.765782 -16.483777   42.76333          -25
## 1970-02-24 0.00000000   1.756084 -11.374860   39.93884          -20
## 1970-02-25 0.00000000   1.822792  -4.722482   37.88671           85
## 1970-02-26 0.00000000   1.835450   0.825322   35.98116           85
##              myEMV.GSPC myVolat.GSPC myMACD.GSPC myMFI.GSPC mySAR.GSPC
## 1970-02-18 0.0002233277    0.2144511   -2.124947   68.64115   85.02000
## 1970-02-19 0.0002792395    0.2151897   -1.976305   75.99052   85.02000
## 1970-02-20 0.0001116343    0.2181275   -1.818074   75.65063   85.16720
## 1970-02-24 0.0002632325    0.2184983   -1.658710   74.89682   85.38157
## 1970-02-25 0.0003729626    0.2296881   -1.478420   75.28083   85.66384
## 1970-02-26 0.0003360702    0.2291771   -1.299327   74.16197   86.07746
##            runMean.Cl.GSPC runSD.Cl.GSPC
## 1970-02-18          86.583     0.4582108
## 1970-02-19          86.769     0.5230780
## 1970-02-20          86.939     0.6298933
## 1970-02-24          87.037     0.7129286
## 1970-02-25          87.362     0.9422276
## 1970-02-26          87.558     1.0431437
```

Hands On Data Creation

- Data download and pre-processing
  1. Experiment with data downloading
     - Search for ticker IDs at Yahoo Finance
  2. Create different data sets for modeling (try different predictors and targets)
  3. Create a dynamic document that allows you to obtain a regular report on the evolution of the prices of some stock ticker during the last x days. Note: it should be easy to change (e.g. through variables in the beginning of the document) both the stock ticker and the value of x and thus being able to obtain a different report

NYU | STERN
MS in Business Analytics

How Predictions Will be Evaluated

- Our prediction task means that the models will forecast a value for the T indicator
- We will map these predicted values into a trading signal:

$$signal = \begin{cases} sell & \text{if } T < \text{sell.thr} \\ hold & \text{if sell.thr} \leq T \leq \text{buy.thr} \\ buy & \text{if } T > \text{buy.thr} \end{cases}$$

NYU | STERN
MS in Business Analytics

How Predictions Will be Evaluated (2)

- Function `trading.signals()` in package `DMwR2` does that mapping for us. The result is a factor with possible values: “b”, “s” or “h”.
- Evaluating these signals could be done through a standard Error Rate, given that it is a nominal variable - however, there is a strong imbalance and thus this is not a good idea!
- We will use the Precision/Recall setup, together with the F-measure to unify both into a single score

Trading-related Evaluation

- In the end the trading signals shall be transformed into trading actions
- The consequences of these actions will have an economic impact
- We will also evaluate our trials using trading-related metrics

Trading-related Evaluation (2)

- Trading-related metrics we will consider:
  1. Economic results factors such as:
     - net balance over the trading period
     - percentage return over this period
     - excess return over buy-and-hold
  2. Risk-related metrics such as:
     - the Sharpe ratio
     - the maximum draw-down
  3. Characteristics of the trades such as:
     - number of trades
     - average return per trade
     - percentage of profitable trades

NYU | STERN
MS in Business Analytics

Performance Estimation for Time Series Models

- The usual techniques for model evaluation revolve around resampling.
  - Simulating the reality.
    - Obtain an evaluation estimate for unseen data.
- Examples of Resampling-based Methods
  - Holdout.
  - Cross-validation.
  - Bootstrap.

Time Series Data Are Special!

Any form of resampling changes the natural order of the data!

NYU | STERN
MS in Business Analytics

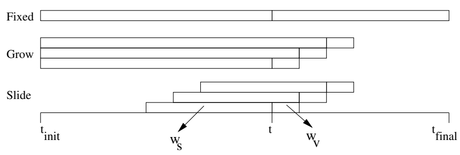
Correct Evaluation of Time Series Models

- General Guidelines
  - Do not "forget" the time tags of the observations.
  - Do not evaluate a model on past data.

- A possible method
  - Divide the existing data in two time windows
    - Past data (observations till a time *t*).
    - "Future" data (observations after *t*).
  - Use one of these three learn-test alternatives
    - Fixed learning window.
    - Growing window.
    - Sliding window.

NYU | STERN
MS in Business Analytics

Learn-Test Strategies for Time Series



Fixed Window

A single model is obtained with the available "training" data, and applied to all test period.

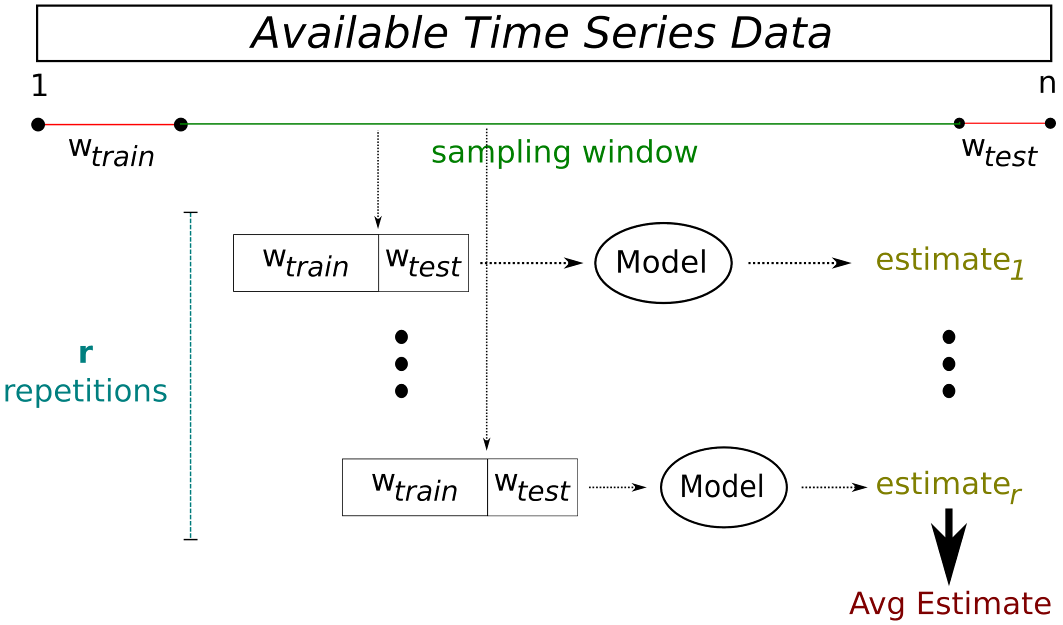
Growing Window

Every w_v test cases a new model is obtained using all data available till then.

Sliding Window

Every w_v test cases a new model is obtained using the previous w_s observations of the time series.

Using Monte Carlo Simulations for Obtaining Reliable Estimates



Use 1.1.1

An illustrative example

The first 2000 rows of our training set

Need The download Perf. Est.

```
library(performanceEstimation)
library(e1071)
exp <- performanceEstimation(
    PredTask(Tform, Tdata.train[1:2000,], 'SP500'),
    c(Workflow('standardWF', wfID="standSVM",
               learner='svm',learner.pars=list(cost=10,gamma=0.01)),
      Workflow('timeseriesWF', wfID="slideSVM",
               type="slide", relearn.step=90,
               learner='svm',learner.pars=list(cost=10,gamma=0.01))
     ),
    EstimationTask(metrics="theil",
                   method=MonteCarlo(nReps=10,szTrain=0.5,szTest=0.25)))
```

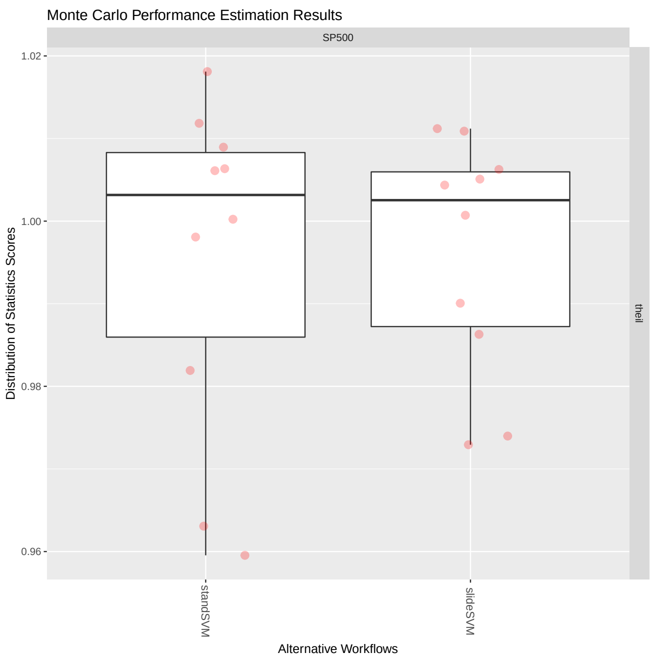
The Results

```
summary(exp)
```

```
##
## == Summary of a  Monte Carlo Performance Estimation Experiment ==
##
## Task for estimating  theil  using
## 10  repetitions Monte Carlo Simulation using:
##   seed =  1234
##   train size =  0.5 x NROW(DataSet)
##   test size =  0.25 x NROW(DataSet)
##
## * Predictive Tasks ::  SP500
## * Workflows  ::  standSVM, slideSVM
##
## -> Task:  SP500
##   *Workflow: standSVM
##               theil
## avg     0.99541918
## std     0.02040426
## med     1.00316807
## iqr     0.02234253
## min     0.95954300
## max     1.01810234
## invalid 0.00000000
##
##   *Workflow: slideSVM
##               theil
## avg     0.99617836
## std     0.01444845
## med     1.00253771
## min     0.97295232
## max     1.01119959
## invalid 0.00000000
```


The Results Graphically

```
plot(exp)
```



NYU | STERN
MS in Business Analytics

Multivariate Adaptive Regression Splines

Additive Models

- Main idea:
  - A complex function may be decomposed in an additive way such that each term has a simpler form.
  - Main advantage/motivation: additive models are very interpretable
- A Generalized Additive Model (GAM) (Hastie and Tibshirani, 1990) can be defined as,

$$r(\mathbf{x}) = \alpha + \sum_{i=1}^{a} f_i(X_i)$$

where the f_i's are univariate functions.

Hastie,T., Tibshirani,R. (1990) : Generalized Additive Models. Chapman & Hall.

Additive Models (cont.)

$$r(\mathbf{x}) = \alpha + \sum_{i=1}^{a} f_i(X_i)$$

- These models can be further generalized over functions with more than one variable.
- The model parameters are usually obtained through the *backfitting* algorithm (Friedman and Stuetzle, 1981).

Friedman,J., Stuetzle,W. (1981) : Projection pursuit regression. Journal of the American Statistical Association, 76 (376), 817-823

Multivariate Adaptive Regression Splines (MARS)

- These are another example of additive models, this time with the form,

$$r(\mathbf{x}) = c_0 + \sum_{i=1}^{p} c_i \prod_{k=1}^{K_i} \left[s_{k,i} \left(X_{v(k,i)} - t_{k,i} \right) \right]_+$$

  where $\left[s_{k,i} \left(X_{v(k,i)} - t_{k,i} \right) \right]_+$ are two-sided trucanted base functions.
- These models can be re-written in an easier to understand format as follows,

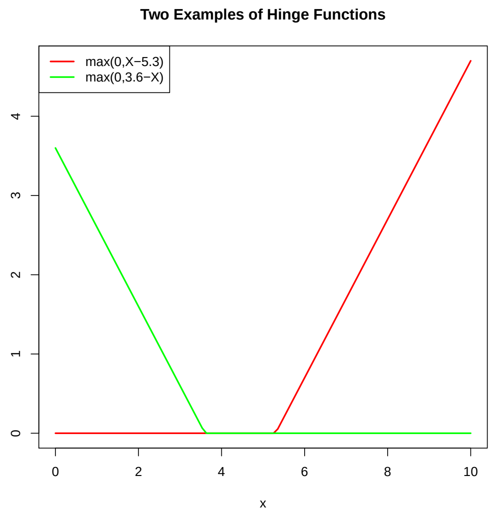
$$r(\mathbf{x}) = c_0 + \sum c_i \cdot B_i(\mathbf{x})$$

  where the B_i's are basis functions and the c_i's are constants.

Friedman,J. (1991) : Multivariate Adaptive Regression Splines. Annals of Statistics, 19:1, 1-141.

Multivariate Adaptive Regression Splines (MARS) - 2

- The basis functions usually take one of the following forms:
  1. the constant 1 (for the intercept)
  2. a hinge function with the form $\max(0, X - k)$ or $\max(0, k - X)$, where k are constants
  3. a product of two or more hinge functions, which try to capture the interactions between two or more variables



MARS - the algorithm

- MARS builds models in two phases: the forward and backward passes
  1. Forward pass
     - start with an intercept (mean of the target)
     - iteratively keep adding new basis function terms
     - this is carried out until a certain termination criterion is met
  2. Backward pass
     - iteratively tries to remove each term in turn
     - use a cross validation criterion to compare and select alternatives

Obtaining MARS Models in R

```
library(earth)  # extra package to install
data(Boston, package="MASS")
sp <- sample(1:nrow(Boston),as.integer(0.7*nrow(Boston)))
tr <- Boston[sp,]
ts <- Boston[-sp,]
mars <- earth(medv ~ .,tr)
preds <- predict(mars,ts)
(mae <- mean(abs(ts$medv - preds)))

## [1] 2.716068
```


Obtaining MARS Models in R (cont.)

```
summary(mars)

## Call: earth(formula=medv~., data=tr)
##
##                 coefficients
## (Intercept)          30.3372
## h(crim-4.22239)      -0.4881
## h(crim-22.0511)       0.4375
## h(nox-0.488)        -23.4996
## h(rm-6.405)           6.6240
## h(rm-7.454)          10.5300
## h(rm-7.923)         -22.6268
## h(dis-2.4298)        -0.7862
## h(2.4298-dis)         6.6821
## h(rad-7)              0.4358
## h(tax-300)           -0.0141
## h(ptratio-14.7)      -0.7006
## h(black-395.5)       -1.3363
## h(395.5-black)       -0.0074
## h(lstat-6.07)        -0.6514
## h(6.07-lstat)         2.5575
## h(lstat-24.56)        0.7851
##
## Selected 17 of 23 terms, and 9 of 13 predictors
## Importance: rm, lstat, ptratio, nox, dis, crim, rad, tax, black, ...
## Number of terms at each degree of interaction: 1 16 (additive model)
## GCV 15.53    RSS 4520    GRSq 0.8263    RSq 0.8564
```

Hands On Modeling and Performance Estimation using Monte Carlo

- Using the data sets you have created previously:
  1. Try different modeling techniques and parameter variants (e.g. MARS and Random Forests)
  2. Estimate their mean squared error on the financial prediction task you have defined before
  3. Try different learn+test variants through the `timeseriesWF()` function. Check its help page to understand how to use it

Not sure of where I did they

The Trading Setup

- We will assume we will trade with futures
  - *Long* and *Short* positions available
- Given a set of signals (derived from the predictions of our models), there are many ways to use them to make trading decisions
- We will describe a few plausible trading strategies, but this is far from exhaustive!

The 1st Trading Strategy

One among many possibilities!

- All **decisions** will be taken **at the end of the day**
- If at the end of the day we have a prediction for a low value of T
  - If we already have a opened position *we ignore this signal*
  - If we do not have a position *we open a short position* issuing a sell order
    - When this order is carried out in the future at a price p we immediately post two extra orders:
      - A **buy limit order** with a limit price of $p - t\%$ ($t\%$ is our target profit) with a deadline of 10 days
      - A **buy stop order** with a limit price of $p + l\%$ ($l\%$ is our maximum accepted loss)

NYU | STERN
MS in Business Analytics

The 1st Trading Strategy (2)

- If the models forecast a high value of T
  - If we already have a opened position *we ignore this signal*
  - If we do not have a position *we open a long position* issuing a buy order
    - When this order is carried out in the future at a price p we immediately post two extra orders:
      - A **sell limit order** with a limit price of $p + t\%$ ($t\%$ is our target profit) with a deadline of 10 days
      - A **sell stop order** with a limit price of $p - l\%$ ($l\%$ is our maximum accepted loss)

NYU | STERN
MS in Business Analytics

The 2nd Trading Strategy

- Similar to the 1st strategy with two exceptions:
  - We will *always open new positions* (as long as we have enough money) even if we have already an opened position
  - We will *wait for ever* (unless the limit loss fires) for positions to reach the target profit
- We will consider only these two strategies, though with several variants in terms of their parameters.
- The two trading strategies are implemented by functions `policy.1()` and `policy.2()` available at the course Web page.

NYU | STERN
MS in Business Analytics

A Trading Simulator

- Function `trading.simulator()` in package `DMwR2` implements a simulation of a trading market
- It accepts several arguments controlling this simulation, namely:
  - The market quotes during the simulation period
  - The signals issued for this period
  - A trading policy function that will transform these signals into market orders
  - The parameters to be passed to this trading policy function
  - The cost of each transaction (defaults to 5 Eur)
  - The initial capital available for trading (defaults to 1M Eur)
- The result of the function is an object of class `tradeRecord` that can be used for obtaining trading evaluation metrics and also for the visualization of the trading performance during the simulation period

NYU | STERN
MS in Business Analytics

An Illustrative example

```
library(DMwR2)
library(quantmod)
data(GSPC, package="DMwR2")
## Train and test periods used in this illustration
start <- 1
len.tr <- 1000  # first 1000 for training models
len.ts <- 500   # next 500 for testing them
tr <- start:(start+len.tr-1)
ts <- (start+len.tr):(start+len.tr+len.ts-1)
## The market quotes during this "testing period"
## This will be used by the simulator
## Note: you need the training data created previously!
date <- rownames(Tdata.train[start+len.tr,])
market <- GSPC[paste(date,'/',sep='')][1:len.ts]
```


An Illustrative example (2)

```
## First we need the code implementing the 2 policies
source("twoPolicies.R")  # file to be downloaded from course site
```

```
library(e1071)
s <- svm(Tform,Tdata.train[tr,],cost=10,gamma=0.01)
p <- predict(s,Tdata.train[ts,])
sig <- trading.signals(p,0.1,-0.1) # predictions to signals
## now using the simulated trader
t1 <- trading.simulator(market,sig,
                        'policy.1',  # the policy function name
                        list(exp.prof=0.05,bet=0.2,hold.time=30))
t1

##
## Object of class tradeRecord with slots:
##
##   trading: <xts object with a numeric  500 x 5  matrix>
##   positions: <numeric  8 x 7  matrix>
##   init.cap :  1e+06
##   trans.cost :  5
##   policy.func :  policy.1
##   policy.pars : <list with  3  elements>
```

Inspecting the Results

```
summary(t1)

##
## == Summary of a Trading Simulation with  500  days ==
##
## Trading policy function :  policy.1
## Policy function parameters:
##   exp.prof  =  0.05
##   bet  =  0.2
##   hold.time  =  30
##
## Transaction costs :  5
## Initial Equity    :  1e+06
## Final Equity      :  1019712   Return :  1.97 %
## Number of trading positions:  8
##
## Use function "tradingEvaluation()" for further stats on this simulation.
```

NYU | STERN
MS in Business Analytics

Inspecting the Results Visually

```
plot(t1,market,theme='white',name='SP500')
```

NYU | STERN
MS in Business Analytics

A More Interesting Alternative with the Same Predictions

```r
t2 <- trading.simulator(market,sig,'policy.2',list(exp.prof=0.05,bet=0.3))
```


A More Interesting Alternative with the Same Predictions (2)

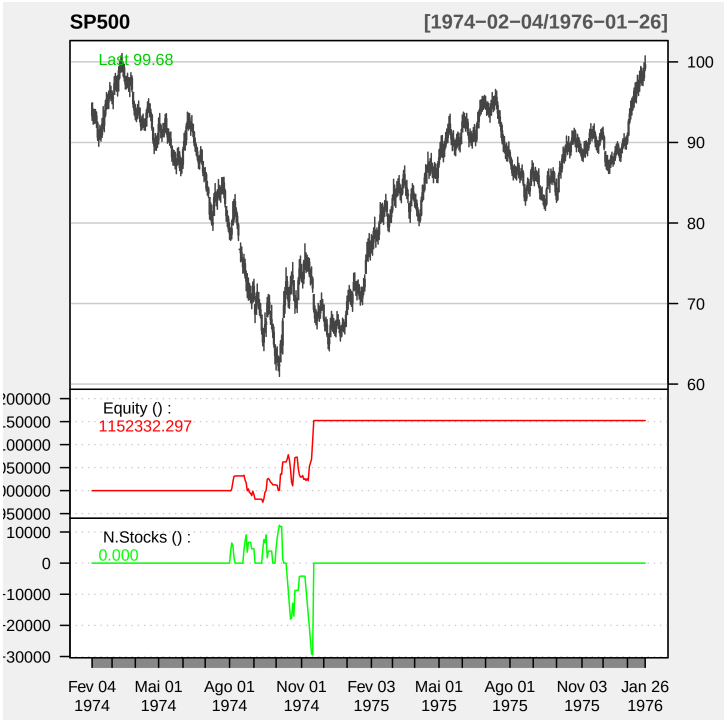
```r
summary(t2)

##
## == Summary of a Trading Simulation with  500  days ==
##
## Trading policy function :  policy.2
## Policy function parameters:
##   exp.prof  =  0.05
##   bet  =  0.3
##
## Transaction costs :  5
## Initial Equity     :  1e+06
## Final Equity       :  1152332   Return :  15.23 %
## Number of trading positions:  37
##
## Use function "tradingEvaluation()" for further stats on this simulation.
```

A More Interesting Alternative with the Same Predictions (3)

```
tradingEvaluation(t2)

##      NTrades        NProf     PercProf           PL          Ret    RetOverBH
##        37.00        26.00        70.27    152332.30        15.23         8.38
##        MaxDD  SharpeRatio     AvgProf      AvgLoss        AvgPL      MaxProf
##     67492.23         0.06         4.99        -4.89         2.05         5.26
##      MaxLoss
##        -5.00
```


Inspecting the Results Visually

```
plot(t2,market,theme='white',name='SP500')
```



Hands On Trading with Model Outcomes

- Using the data you have built previously:
  1. Try to obtain some models and trade with them
  2. Evaluate the results of your models
  3. Experiment with different trading policies and parameter settings of these policies

A Brief Recap

- What we have seen till now
  - How to obtain the required data and how to define a useful predictive task
  - How to obtain predictive models for this task
  - How to transform their predictions into trading actions
  - How to evaluate these models
  - That reliable estimates of the performance should be obtained through Monte Carlo
- What we will see now
  - How to select and compare different trials at this problem

A Brief Recap (2)

- The plan
  - We have split the data in two partitions (`Tdata.train`, 30 years; and `Tdata.eval`, 10 years)
  - We will perform model selection and comparison looking only at `Tdata.train`
  - After we select the "best" model we will use it to trade on `Tdata.eval` - the final evaluation

Doing Monte Carlo Comparisons

- The function `performanceEstimation()` from package `performanceEstimation` will be used for model selection
- The different steps we have described to address this problem include many parameters and alternatives
- Exploring and comparing all possibilities requires far too much computation power and is out of the scope of this course
- We will illustrate the comparison methodology selecting a few of these alternatives
- Namely:
  - In terms of modeling we will consider a few variants of SVM
  - We will also consider some alternatives in terms of trading policies

Doing Monte Carlo Comparisons (2)

- Our approach to solve the trading task consists of the following steps:
  1. Obtain a model and its predictions for the *T* indicator
  2. Transform these prediction into trading signals
  3. Trade using these signals and a certain trading policy
- This **workflow** will have to be executed for different train and test partitions within the Monte Carlo simulation, each time collecting the respective trading results
- As this is a specific workflow we will write a function implementing it

Our User-defined Workflow Output Evaluation

```
tradingEval <- function(trueSigs,predSigs,tradeRec,...)
{
    ## Signals evaluation
    st <- sigs.PR(predSigs,trueSigs)
    dim(st) <- NULL
    names(st) <- paste(rep(c('prec','rec'),each=3),c('s','b','sb'),sep='.')

    ## Trading record evaluation
    tradRes <- tradingEvaluation(tradeRec)
    return(c(st,tradRes))
}
```

Our User-defined Trading Workflow

```
tradingWF <- function(form, train, test,
                      quotes, pred.target="signals",
                      learner, learner.pars=NULL,
                      predictor.pars=NULL,
                      learn.test.type='fixed', relearn.step=30,
                      b.t, s.t,
                      policy, policy.pars,
                      trans.cost=5, init.cap=1e+06)
{
    ## obtain the model(s) and respective predictions for the test set
    if (learn.test.type == 'fixed') {  # a single fixed model
        m <- do.call(learner,c(list(form,train),learner.pars))
        preds <- do.call("predict",c(list(m,test),predictor.pars))
    } else {  # either slide or growing window strategies
        data <- rbind(train,test)
        n <- NROW(data)
        train.size <- NROW(train)
        sts <- seq(train.size+1,n,by=relearn.step)
        preds <- vector()
        for(s in sts) {  # loop over each relearn step
            tr <- if (learn.test.type=='slide') data[(s-train.size):(s-1),]
                  else data[1:(s-1),]
            ts <- data[s:min((s+relearn.step-1),n),]

            m <- do.call(learner,c(list(form,tr),learner.pars))
            preds <- c(preds,
                       do.call("predict",c(list(m,ts),predictor.pars)))
        }
    }
```


Our User-defined Trading Workflow (cont.)

(continuation of the function on the previous slide)

```
## Getting the trading signals
    if (pred.target != "signals") {  # the model predicts the T indicator
        predSigs <- trading.signals(preds,b.t,s.t)
        tgtName <- all.vars(form)[1]
        trueSigs <- trading.signals(test[[tgtName]],b.t,s.t)
    } else {  # the model predicts the signals directly
        tgtName <- all.vars(form)[1]
        if (is.factor(preds))
            predSigs <- preds
        else {
            if (preds[1] %in% levels(train[[tgtName]]))
                predSigs <- factor(preds,labels=levels(train[[tgtName]]),
                                   levels=levels(train[[tgtName]]))
            else
                predSigs <- factor(preds,labels=levels(train[[tgtName]]),
                                   levels=1:3)
        }
        trueSigs <- test[[tgtName]]
    }

    ## obtaining the trading record from trading with the signals
    date <- rownames(test)[1]
    market <- get(quotes)[paste(date,"/",sep='')][1:length(preds),]
    tradeRec <- trading.simulator(market,predSigs,
                                  policy.func=policy,policy.pars=policy.pars,
                                  trans.cost=trans.cost,init.cap=init.cap)

    return(list(trueSigs=trueSigs,predSigs=predSigs,tradeRec=tradeRec))
```

Carrying Out Performance Estimation

Note : the following should take long to run. You can download the `mc.res` object from the course web site.

```
library(DMwR2)
library(performanceEstimation)
library(e1071)
library(quantmod)
mc.res <- performanceEstimation(
    PredTask(Tform,Tdata.train),
    workflowVariants('tradingWF',
                     quotes='GSPC',
                     learner='svm',
                     pred.target="indicator",
                     learner.pars=list(cost=c(1,10),gamma=0.01),
                     b.t=c(0.01,0.05),s.t=c(-0.01,-0.05),
                     policy='policy.2',
                     policy.pars=list(bet=c(0.2,0.5),
                         exp.prof=0.05,max.loss=0.05)
                     ),
    ## Monte Carlo repeating 5 times: 10y for training and 5y for testing
    EstimationTask(method=MonteCarlo(nReps=5,szTrain=2540,szTest=1270,seed=1234),
                   evaluator="tradingEval")
  )
```

NYU | STERN
MS in Business Analytics

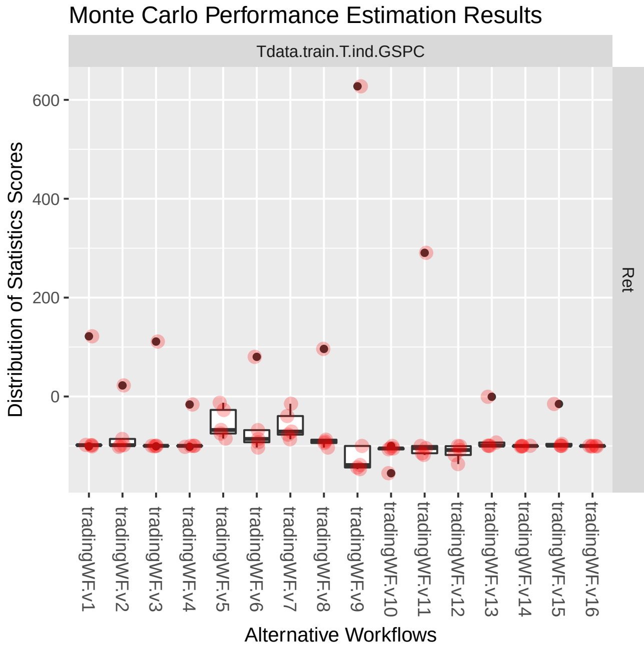
Analyzing the results

```
mc.res2 <- subset(mc.res,
                  metrics=c("PercProf", "Ret",  "MaxDD"),
                  partial=FALSE)

topPerformers(mc.res2, maxs=c(TRUE,TRUE,FALSE))

## $Tdata.train.T.ind.GSPC
##                Workflow   Estimate
## PercProf tradingWF.v6     44.578
## Ret      tradingWF.v9      19.59
## MaxDD    tradingWF.v7 737927.236
```

NYU | STERN
MS in Business Analytics

Analyzing the results (2)

```
plot(subset(mc.res2,metrics="Ret"))
```



NYU | STERN
MS in Business Analytics

Analyzing the results (3)

```
getWorkflow("tradingWF.v5",mc.res)

## Workflow Object:
##  Workflow ID       ::  tradingWF.v5
##  Workflow Function ::  tradingWF
##       Parameter values:
##   learner.pars  -> cost=1 gamma=0.01
##   policy.pars  -> bet=0.2 exp.prof=0.05 max.loss=0.05
##   quotes  -> GSPC
##   learner  -> svm
##   pred.target  -> indicator
##   b.t  -> 0.01
##   s.t  -> -0.05
##   policy  -> policy.2
```

NYU | STERN
MS in Business Analytics

Analyzing the results (3)

```
summary(subset(mc.res2, workflows="tradingWF.v5"))
```

```
##
## == Summary of a  Monte Carlo Performance Estimation Experiment ==
##
## Task for estimating all metrics of the selected evaluation function using
## 5  repetitions Monte Carlo Simulation using:
##    seed =  1234
##    train size =  2540  cases
##    test size =  1270  cases
##
## * Predictive Tasks ::  Tdata.train.T.ind.GSPC
## * Workflows  ::  tradingWF.v5
##
## -> Task:  Tdata.train.T.ind.GSPC
##   *Workflow: tradingWF.v5
##            PercProf        Ret     MaxDD
## avg      43.798000 -53.52000 844258.3
## std       8.452104  31.70547 156811.2
## med      41.210000 -67.75000 911210.9
## iqr       4.050000  47.72000 140256.0
## min      36.250000 -85.10000 598429.9
## max      58.120000 -12.65000 996775.5
## invalid  0.000000   0.00000      0.0
```

NYU | STERN
MS in Business Analytics

Trying the “Best” Model on the Evaluation Period

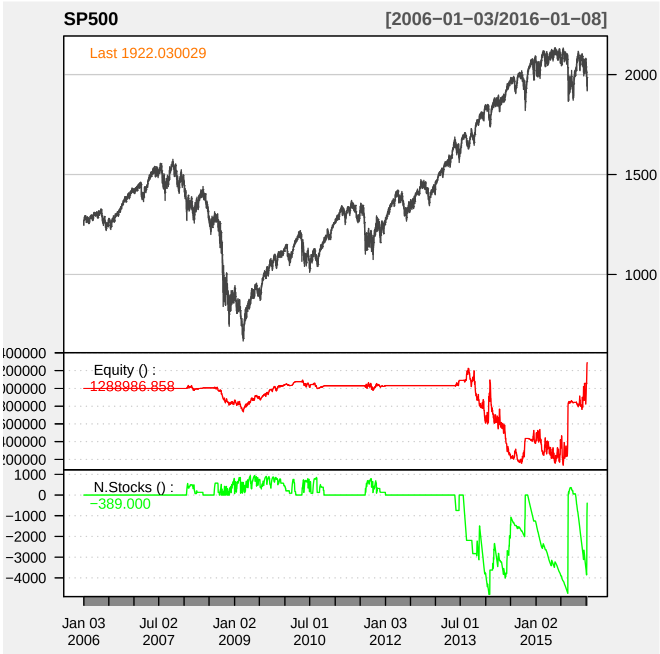
```
t <- tradingWF(Tform,Tdata.train[(nrow(Tdata.train)-2540+1):nrow(Tdata.train),],
               Tdata.eval,pred.target="indicator",
               learner="svm",learner.pars=list(cost=1,gamma=0.01),
               quotes="GSPC",
               b.t=0.01,s.t=-0.05,
               policy="policy.2",
               policy.pars=list(bet=0.2,exp.prof=0.05,max.loss=0.05))
tradingEval(t$trueSigs,t$predSigs,t$tradeRec)
```

```
##         prec.s        prec.b        prec.sb         rec.s         rec.b
##   1.778243e-01  4.731707e-01  3.141892e-01  1.689861e-01  3.139159e-01
##         rec.sb       NTrades          NProf      PercProf            PL
##   2.488849e-01  8.870000e+02  5.500000e+02  6.201000e+01  2.889869e+05
##            Ret     RetOverBH          MaxDD   SharpeRatio       AvgProf
##   2.890000e+01 -2.259000e+01  1.093129e+06  3.000000e-02  5.140000e+00
##        AvgLoss         AvgPL        MaxProf       MaxLoss
## -4.780000e+00  1.370000e+00  5.980000e+00 -5.620000e+00
```

NYU | STERN
MS in Business Analytics

The Performance Visually

```r
library(quantmod)
tr <- t$tradeRec
market <- GSPC[paste(start(tr@trading),end(tr@trading),sep='/'),]
plot(tr,market,theme="white",name="SP500")
```



Hands On Model Selection and Final Evaluation

- Using the results of the model selection process (object `mc.res` provided at the course web page) :
  1. Select other model variant and obtain its characteristics and main evaluation metrics
  2. Apply the selected model to the final evaluation period and check the results
  3. Repeat the previous question using different trading policy settings

Summary

- We have seen another case study of using data mining tools and R for helping in solving a hard decision problem - trading in financial markets
- We have covered further relevant data mining topics in the context of this case study:
    - Time series forecasting
    - Examples of data pre-processing for constructing relevant predictors and target variables
    - How to evaluate properly the predictive performance of time series forecasting models
    - How to move from predictions into decisions
    - How to incorporate a data mining tool into a real time application using a simulator to properly evaluate its potential
    - Specific tools and packages of R for financial analysis

NYU STERN
MS in Business Analytics