Machine Learning for Financial Forecasting

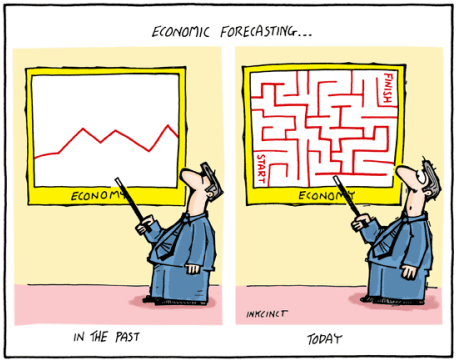
Ali Habibnia

Department of Statistics, LSE

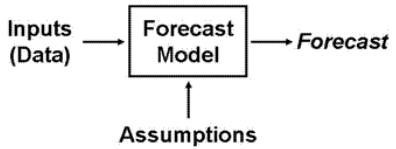
May , 2016

ML in My Academic Works

- **2010** - Forecasting Gold Price Using Optimized Neuro-Fuzzy with Genetic Algorithm (GA-ANFIS) & Smooth Transition Regression with Long Memory (FI-STAR)
- **2011** - Developing Mathematical Models for Forecasting EURJPY in Foreign Exchange Market
- **2012** - Forecasting Financial Volatility By Introducing a GA-Assisted SVR-GARCH Model
- **2012-2016**
- Past, Present and Future of Testing for Nonlinearity in Time Series
- A Nonlinear Generalization of Factor Models for Forecasting Financial Series with Many Predictors
- Nonlinear Forecasting Using a Large Number of Predictors: One-shot Model
- Econometric Modelling of Systemic Risk: Allowing for Nonlinearity & High-Dimensionality



$(X_1, X_2, ..., X_n)$ $f(X;\theta)$ y



$$y_{t+h|t} = f(X_{t-i|t};\theta)$$

- Building accurate forecast models in economics and finance is a complex and challenging task.
- **In this talk:** we will see how to apply appropriate and novel techniques to design data driven forecast models in few steps from data mining and model selection to forecasts evaluation and comparison. Each step has its own tricks!

Outline

- **Stylized facts of financial series** (non-Gaussianity, nonlinearity ...)
- **Input selection and optimal lags** (garbage in, garbage out)
- **Review of forecasting models and benchmark**
- **Foundations of statistical machine learning**
  - Neural Networks & Deep Learning
  - Support Vector Regression
  - Tree-structured models for regression & Random Forest
- **Model Validation and Forecast Comparisons**
  - Time series approach
  - Trading (portfolio) simulation approach
- **Machine Learning and Technical Analysis**
- **ML in MATLAB, R and Python**

Stylized facts of financial returns

- Financial returns present special features and share the following stylised facts: comovements, nonlinearity, non-gausianity (skewness and heavy tails), leverage effect and and volatility clustering which makes the modelling of this variable hard.
- It is very instructive and crucial for building forecast models and input selection to understand and consider these features.
- Market returns have become more correlated during the period of crisis.

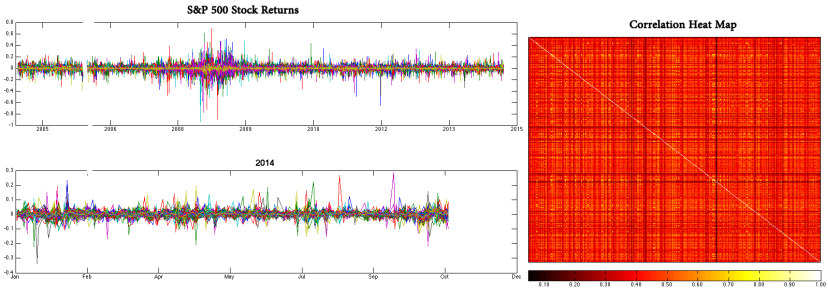


Figure : Daily return observations of the 419 companies in S&P500 index

Nonlinearity: we let the data speak for themselves as much as possible

- A linear stochastic process can be represented in terms of an arithmetic sequence of independent and identically distributed random variables in time domain or the power spectrum in the frequency domain
- Any stochastic process that does not satisfy the condition of the those representations is said to be nonlinear and can be shown with a nonlinear dynamic equation of iid random variables consisting of the current and past shocks.
- Nonlinearity may arise in different ways. The characteristic of nonlinear time series such as higher-moment structures, time-varying variance, asymmetric fluctuations, thresholds and breaks can be only modelled by an appropriate nonlinear function like $f(.)$ and a linear process is not adequate to model these features.
- Before we apply nonlinear techniques, such as those inspired by machine learning theories, to real-world financial data, it is logical to first ask if the use of such techniques is justified by the data. To this purpose, we examine the nonlinear dependencies in return series by applying nonlinearity tests introduced in the literature.
- Nonlinearity test results can assist in terms of choosing the appropriate model.

Some of the well known nonlinearity tests (classified by the author)

ML for complex and nonlinear phenomena

- However linear regression models are adequate to explain many phenomena in the world, most important economic and financial phenomena are complex and nonlinear in nature.

- Parametric nonlinear regression models:
  - The shape of the functional relationships between the response and the predictors are predetermined
  - Can take the form of a polynomial, exponential, trigonometric, power, or any other nonlinear function
- Nonparametric and semiparametric models :
  - In many situations, the relationship is unknown
  - The shape of the functional relationships between variables can be adjusted to capture unusual or unexpected features of the data
  - Artificial Neural Networks, Kernel-based methods & Tree-based regression models

Artificial Neural Networks: one of the oldest and one of the newest areas

- For those who are not familiar with NN models, see Bishop (1995), Hastie, Tibshirani & Friedman (2009), Terasvirta, Tjostheim & Granger (2010)
- Neural Networks: Statistical approach (Varian, 2014; Terasvirta, Van Dijk, Medeiros, 2005; Kuan White, 1994)
- ANNs are flexible functional forms motivated by the way the brain processes information. ANNs consist of a cascade of simple computational units called neurons, which are highly interconnected.
- Neural networks can use a variety of topologies but, based on the universal approximation theorem, a single hidden layer feed forward network architecture with finite number of neurons can approximate arbitrary well any continuous function of n real variables. (poofs have been given by Cybenko (1989), Hornik et al. (1989), White (1990) and Hornik (1991)).
- There has been a resurgence in the field of artificial neural networks in recent years, known as Deep neural networks. Deep neural networks use multiple stages of nonlinear computation and have won numerous contests on an array of complex tasks.

Feedforward Neural Networks

- Multilayer neural networks form compositional functions that map the inputs nonlinearly to outputs. If we associate index i with the input layer, index j with the hidden layer, and index k with the output layer, then an output unit in the network computes an output value y_t given and input $\mathbf{x}_t$ via the following compositional function:

$$y_t = f(X;\theta) = \phi_k\left[\beta_k + \sum_{j}^{L} \phi_j\left(\beta_j + \sum_{i}^{N} x_{it} w_{ij}\right) w_{jk}\right] + \varepsilon_t$$

- where x_{it} is the value of the ith input node, which can be a matrix of lagged values of y_t and some exogenous variables. $\phi_j(.)$ and j are activation functions and number of nodes (L neurons) used at the hidden layer. $\phi_k(.)$ function denotes the output transfer function than can be either linear or a Heaviside step function. β_j and β_k are the biases.

Feedforward Neural Networks

- Formulation of a multilayer feedforward neural network model with more than one hidden layer (i.e. h hidden layers when $h = 1, ..., M$) can be generalized to

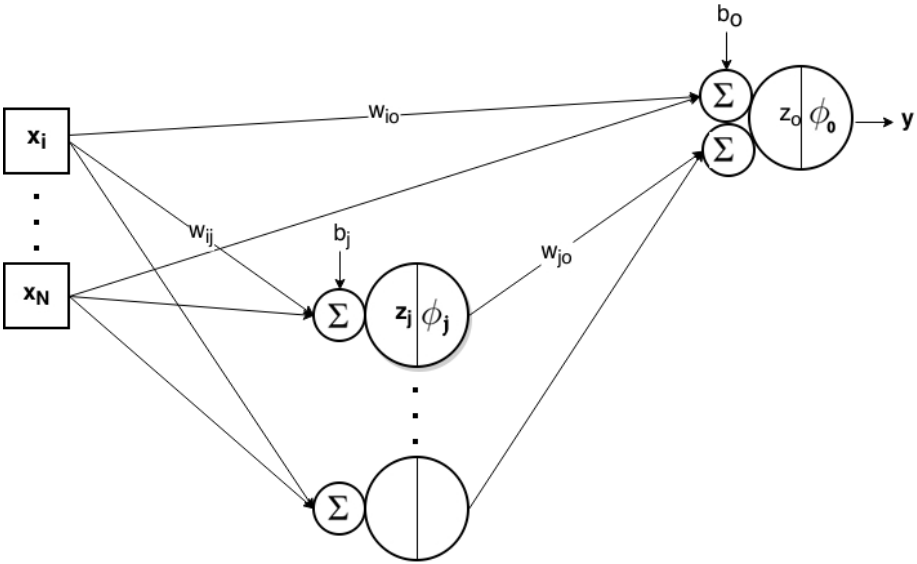
$$y_t = \phi_k \left[\beta_k + \sum_h^M \phi_h \left(...\phi_j \left(\beta_j + \sum_i^N x_{it} w_{ij} \right) \right) w_{hk} \right] + \varepsilon_t,$$

- To show that the neural network models can be seen as a generalization of linear models, we allowed for direct connections from the input variables to the output layer and we assumed that the output transfer function $\{\phi_k(.)\}$ is linear, then the model becomes

$$y_t = \beta_k + \sum_i^N x_{it} w_{ik} + \sum_j^L \phi_j \left(\beta_j + \sum_i^N x_{it} w_{ij} \right) w_{jk} + \varepsilon_t,$$

- Where the first summation represents a linear regression term with constant.

a single-hidden-layer neural network with skip-layer connections



Learning Algorithms

- Estimating the set of network parameters $\theta = \{\mathbf{W}, \beta\}$ in a way that minimize the errors that the network makes is known as training/learning neural network. It is equivalent to finding point in parameter space that makes the height of the error surface small.
- In principle, four classes of supervised learning algorithm (desired outputs are available) has been discussed in literature: steepest descent algorithm (also known as backpropagation), Newtons method, Gauss-Newtons algorithm and Levenberg-Marquardt algorithm.

$$\theta^* = \underset{\theta}{\text{argmin}}\ E$$

Gradient descent learning algorithm:

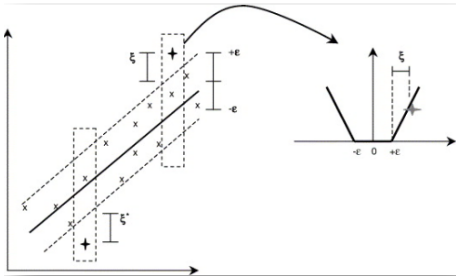
$$\begin{cases} \Delta = -\frac{\partial E}{\partial \theta} \\ \theta^{new} = \theta^{old} + \eta\Delta \end{cases}$$

Kernel-based methods

- Can be viewed as a nonlinear mapping from inputs into higher dimensional feature space in the hope that the data will be linearly separable or better structured.
- Provide a structured way to use a linear algorithm in a transformed feature space. (This so-called kernel trick; See Hofmann, Scholkopf, Smola (2008))
- Perhaps the biggest limitation of the kernel-based methods lie in choice of the kernel and tuning model parameters.
- Best known example are support vector machines (SVM) introduced by Vapnik in 1995.
- In contrast to previous black box learning approaches, SVMs allow for some intuition and human understanding.
- SVM avoids over-fitting: Trade-off between complexity and error can be controlled explicitly.

Support Vector Regression (SVR)

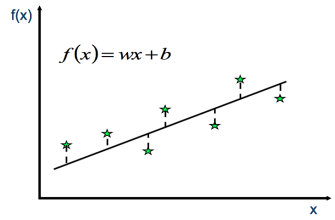
- In SVR we deal with a ε-Insensitive loss function (Vapnik-Chervonenkis VC Loss Function which is similar to the Huber robust loss functions) and the goal is finding the flattest function which has biggest ε deviation from the actual targets. See (Vapnik, 1995; Vapnik, 1998)



- Only points out-side the upper and lower bounds contribute to the cost.
- Nonlinear SVR and Kernel Trick: we can map every input data point into a high-dimensional feature space through a mapping function and then apply the standard SVR algorithm.

Comparition with OLS

- Optimization problem can be solved in a dual formulation. Lagrange Multipliers is a standard dualization method. The optimality will achieve when the partial derivatives of the Lagrange function for the primal variables be equal to zero. See Smola and Scholkopf (2004)



- Solution:

$$Loss = \|(wX + b) - Y\|_2$$

$$\frac{dLoss}{dw} = 0 \Rightarrow$$

$$(X^T X)w = X^T Y$$

- Minimise:

$$\frac{1}{2} w^T w + C \sum_{i=1}^{N} (\xi_i + \xi_i^*)$$

- Constraints:

$$y_i - w^T x_i - b \le \varepsilon + \xi_i$$

$$w^T x_i + b - y_i \le \varepsilon + \xi_i^*$$

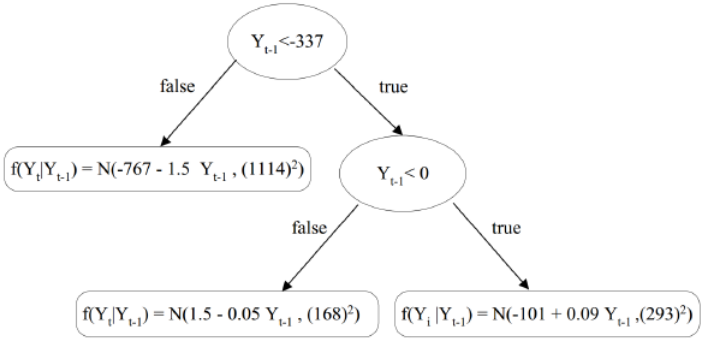
$$\xi_i, \xi_i^* \ge 0$$

Tree-based regression models

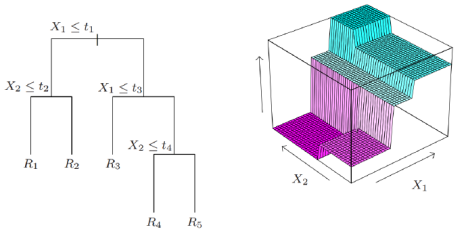
- Are alternative (nonparametric and nonlinear) approaches to regression that are not based on assumptions of normality and user-specified model statements.
- Originated in the 1960s with the development of AID (Automatic Interaction Detection) by Morgan and Sonquist. In the 1980s, statisticians Breiman et al. (1984) developed CART(Classification And Regression Trees)
- Regression Tree:
  - The fundamental idea behind RT is to recursively partition the regressors space in regions (build a tree) until all the subspaces are sufficiently homogenous in order to estimate the regression function with the sample average (or the specific local model employed) in each region.
  - The root is the top node, includes all observations in the learning sample. The splitting condition at each node is expressed as an if-then-else rule that is determined by a specific splitting criterion. (i.e. minimize the mean squared error)
  - Financial dataset has multiple features that interact in complicated and nonlinear ways, So, a single global model for the entire of data may inadequately capture the underlying relationships.

Autoregressive Tree Models: an AR(1) example

- An autoregressive tree (ART) model is a piecewise linear autoregressive model in which the boundaries are defined by a decision tree, and the leaves of the decision tree contain linear autoregressive models. See Meek, Chickering and Heckerman (2002)



ARXT example and Overfitting problem



To prevent overfitting

- Bagging and boosting: is a technique proposed by Breiman (1996) that can be used with many regression methods to reduce the variance associated with prediction, and thereby improve the prediction process. It is a relatively simple idea: many bootstrap samples are drawn from the available data, some prediction method is applied to each bootstrap sample, and then the results are combined.
- Random forest: developed by Breiman that combine many decision(regression) trees constructed using subset of training samples and choose best feature as split point among random subset of features at each split node. (Randomisation in training samples and features)

Comparing the forecastability of alternative quantitative models

- Time series approach (R2, RMSE, MAE, Hit Rate, Mean Profit per Day, DM-test ...)

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=h+1}^{N}(y(t) - \hat{y}(t))^2}$$

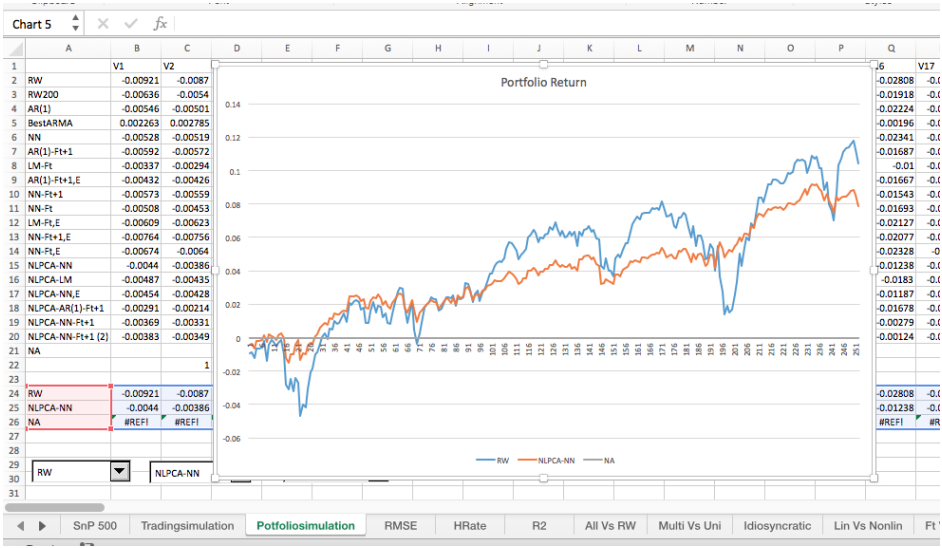
$$Hit - Rate = \frac{|\{t|y(t)\hat{y}(t)>0, t=1,..,N\}|}{|\{t|y(t)\hat{y}(t)\neq 0, t=1,..,N\}|}$$

how often the sign of the return is correctly predicted. HR>0.5 better than RW

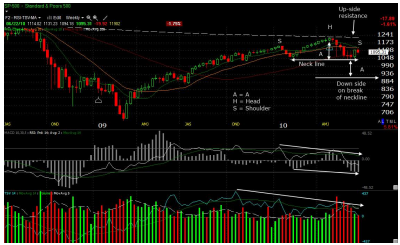
- Trading simulation approach (Profit of a portfolio with one asset or more than one assets)
  - In time series approach, models aim to minimise Out-of-sample forecasting errors, however, the model with minimum statistical errors does not necessarily guarantee maximised trading profits, which is often deemed as the ultimate objective of financial application.
  - Since the ultimate goal of investment is to make profit, the best way to evaluate alternative financial forecast model is therefore to evaluate their trading performance.
- Benchmark for trading simulation:
  - The performance of AR(1), RW or the stock market index during the same out-of-sample period.

Portfolio return

- Sharpe Ratio is higher in ML Model means that the portfolio performance fluctuate less during crisis.



The link between Technical Analysis & Machine Learning



- How ML helps TA:
  - Metaheuristic optimization algorithm (Genetic algorithm, Ant colony optimization, Particle swarm optimization and ...) can be used to evolve Trading Strategies.
  - NNs can be used to create a model for predicting the future values of a TA indicators.
  - Automatically recognizing technical analysis chart patterns by using NNs.
  - Automatically discovering significant candlestick patterns.
  - These are just some of the ways ML can be used for technical analysis. Many other possibilities exist.

- TA indicators as model inputs :
  - Most NN systems are just made to use the raw price time series for input (maybe with some kind of normalization), but an NN with technical indicator inputs (MAs, MACD, even pattern matching for stuff like Head-Shoulders, support levels, etc.) is imaginable.

ML in MATLAB, R and Python

- When you know the math behind ML methods, you can use any programming languages to write your own programms. But, you dont need to reinvent the wheel.
- MATLAB Toolboxes for Machine Learning:
  - Statistics and Machine Learning Toolbox
  - Neural Network Toolbox
  - Global Optimization Toolbox
  - Fuzzy Logic Toolbox
  - Signal Processing Toolbox
  - Computer Vision System Toolbox
- CRAN R Packages for Machine Learning:
  - See: CRAN Task View: Machine Learning Statistical Learning
  - An Introduction to Statistical Learning: with Applications in R (Authors: James, G., Witten, D., Hastie, T., Tibshirani, R.)
- Machine Learning Libraries in Python:
  - There are tons of machine learning libraries already written for Python.
  - Tensorflow, scikit-learn, Theano, Pylearn2, Pyevolve, NuPIC, Pattern, Caffe, ...
- A quick review of ML in Python, R and MATLAB can be downloaded from my LSE page http://personal.lse.ac.uk/habibnia (coming soon)