An Introduction to Machine learning in Quantitative Finance

Hao Ni
UCL, ATI and DataSig

DataSig

A rough path between mathematics and data science

The Alan Turing Institute

Imperial College London

Objectives of the Tutorial

- To introduce a high-level picture of Machine Learning (ML) in Quantitative finance;
- To provide you with a systematic framework of supervised learning;
- To uncover the black-box of deep learning and help you develop the mathematical understanding of neural networks;
- To equip you with hands on experience on applying ML to empirical financial data challenges.

This tutorial will cover

- An overview of Machine Learning in Quantitative finance
- Supervised Learning, including the linear regression and neural networks
- The prediction of future price movement on high frequency limit order book data using neural networks.

Outline

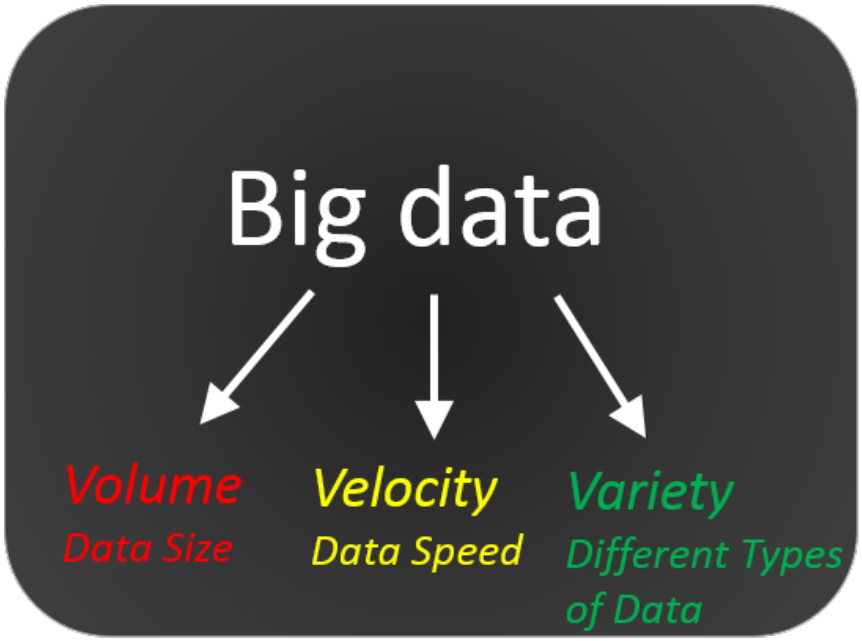
1. A Primer to Machine Learning and its Financial Applications

2. Supervised Learning
   - Linear Regression
   - Non-linear Regression

3. Neural Networks
   - Shallow Neural Network
   - Deep Neural Network
   - Recurrent Neural Network

4. Applications of Neural Networks in Finance

Big Data Era


Big data
Volume
Data Size
Velocity
Data Speed
Variety
Different Types
of Data

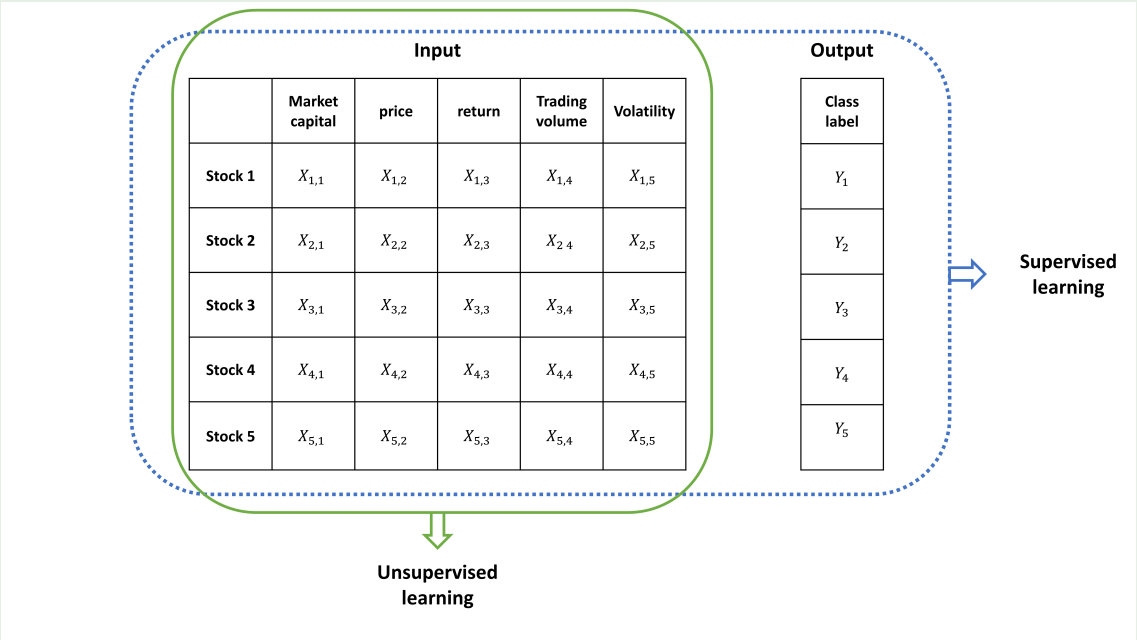
Machine Learning is a field of study to give computer systems the ability to ”learn” with data, without being explicitly programmed.

Machine learning *is ideal for exploiting the opportunities hidden in* ***big data***.

Categories of Machine Learning Tasks

- Supervised Learning
  - Regression (e.g. forecast returns);
  - Classification (e.g. predict the direction of returns).
- Unsupervised Learning
  - Clustering (e.g. identify the most common signs of market stress).
- Reinforcement Learning (e.g. learning trading strategy)

Example (Supervised v.s. Unsupervised Learning)



Opportunities and Challenges of ML in Finance

Industrial Applications in Financial Markets

- Fintech
  - AI Advisor for Personal Customer;
- Factor Investments
  - Digging New Factors (e.g. genetic programming algorithm);
  - Predict Stock Returns (e.g. neural network, ensemble models);
  - Alternative Data (e.g. news data and NLP).
- Timing Strategy
  - High Frequency Data.

Challenges of Financial Data

- Low Signal-to-Noise Ratio;
- Time-heterogeneity;
- Low Availability.

Outline

1 A Primer to Machine Learning and its Financial Applications

2 Supervised Learning
- Linear Regression
- Non-linear Regression

3 Neural Networks
- Shallow Neural Network
- Deep Neural Network
- Recurrent Neural Network

4 Applications of Neural Networks in Finance

Supervised Learning

Supervised Learning

Supervised learning is the machine learning task of learning a function (f) that maps an input (X) to an output (Y) based on example input-output pairs.

Regression analysis

Dataset: $\mathscr{D}_{\text{training}} = \{(x_i, y_i)\}_{i=1}^N$ satisfies that $y_i = f(x_i) + \varepsilon_i$, where $x_i := (x_i^{(1)}, \cdots, x_i^{(d)}) \in \mathbb{R}^d$, ε_i is iid with $\mathbb{E}[\varepsilon_i | x_i] = 0$.
We also adopt the matrix form for $\mathscr{D}_{\text{training}} = (X, Y)$, where

$$Y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \text{ and } X = \begin{pmatrix} x_1^{(1)}, & x_1^{(2)}, & \cdots, & x_1^{(d)} \\ \vdots & \vdots & & \vdots \\ x_N^{(1)}, & x_N^{(2)}, & \cdots, & x_N^{(d)} \end{pmatrix}.$$

Question: For any new input data x^*, what is $\mathbb{E}[\underbrace{y}_{f(x)+\varepsilon} | x = x^*]$, i.e. $f(x^*)$? $\Leftrightarrow$

How to estimate f from $\mathscr{D}_{\text{training}}$?

General Framework

Regression Framework

Dataset: $\mathscr{D} = \{(x_i, y_i)\}_i = \mathscr{D}_{\text{training}} \cup \mathscr{D}_{\text{test}}$

Model: $y = f_\theta(x) + \varepsilon, \quad \forall x \in \mathbb{R}^d. \ (f_\theta \sim f)$

Empirical Loss: $L(\theta|\mathscr{D}_{\text{training}}) \rightarrow$Minimize

Optimization: $\hat{\theta} = \arg\min_\theta(L(\theta|\mathscr{D}_{\text{training}}))$

Prediction: $\hat{y}_* = f_{\hat{\theta}}(x_*).$

Validation: Compute the test metrics on $\mathscr{D}_{\text{test}}$.

Linear Regression

Ordinary Least Squares (OLS)

Model:	$y = f_\theta(x) + \varepsilon = x\theta + \varepsilon.$
Loss Function:	$L(\theta \mid X, Y) = (Y - X\theta)^T (Y - X\theta) \to \min;$
Optimization:	$\hat{\theta} = (X^T X)^{-1} X^T y.$
Prediction:	$\hat{y}_* = x_* \hat{\theta}.$
Validation:	Compute RMSE, R^2, the adjusted R^2, p-value.

The Limitations of OLS

- Even if linear model is a reasonable model, $X^T X$ might not be invertible:
  - There are co-linear dependence between different coordinates of the input variables (Dimension Reduction);
  - The sample size N is smaller than the dimension of the input space. (Overfitting issue)
- Linear model might not be adequate (Underfitting issue);

Overfitting and Regularization

Overfitting

Overfitting is the phenomena of using over-complicated models to fit the data, which results in poor performance on the test data set.

Remark

For OLS, when the sample size is smaller than the input dimension d, there are infinite many θ such that

$$L(\theta|X,Y) = (Y - X\theta)^T(Y - X\theta) = 0,$$

and it leads to the overfitting issue.

Regularization

In order to resolve the overfitting issue here, we consider the constraint optimization problem

$$\min_{\beta}(Y - X\beta)^T(Y - X\beta), \text{ s.t. } ||\beta|| \leq t.$$

Extension of OLS

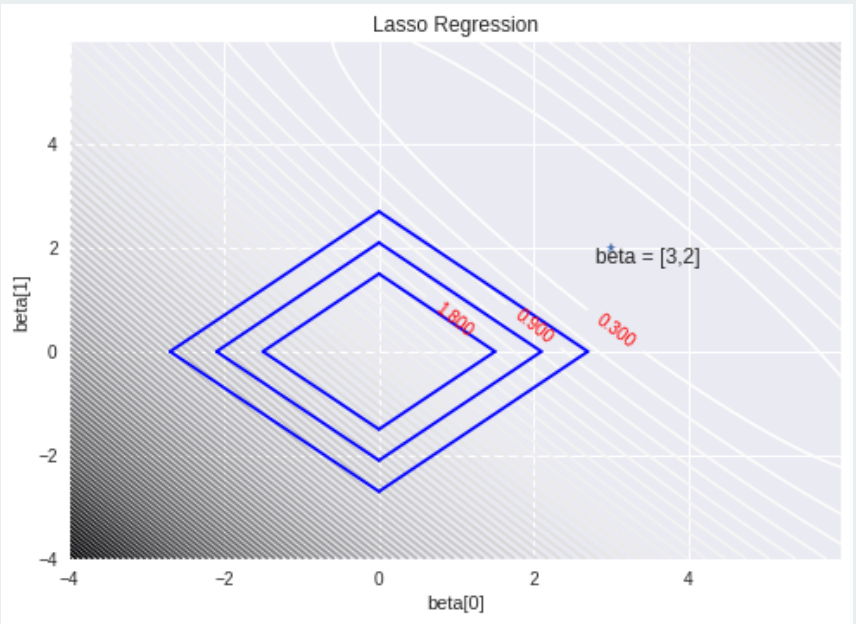
- Regularization (avoid overfitting issue):
  - Lasso Regression: $L(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda||\beta||_1$;
  - Ridge Regression: $L(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda||\beta||_2^2$;
  - Elastic Net:

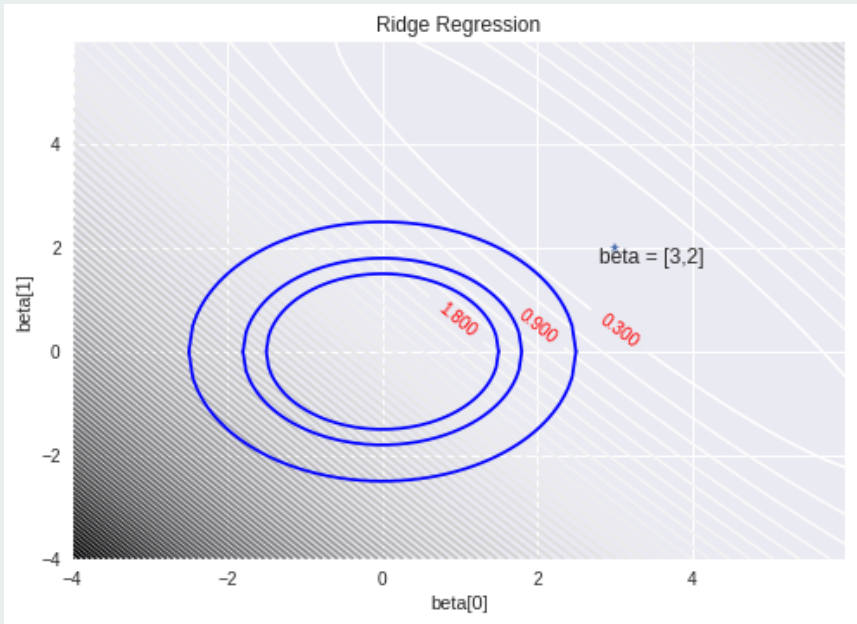
$$L(\beta|X,Y) = (Y - X\beta)^T(Y - X\beta) + \lambda\left(\frac{1-\alpha}{2}||\beta||_2^2 + \alpha||\beta||_1\right);$$

- Robust to outliers:
  - Robust Regression: Modify the Loss function to the weighted sum of the residuals and the weighting is done iteratively.

Comparison of Different Regularization Methods

- Lasso requires numerical techniques to solve, but it can be used for feature selection;
- Ridge regression has the closed form solution for the optimal parameters.
- Elastic nets is a combination of Lasso and Ridge regression.





Example (Sparse Linear Model)

We simulate 1000 samples of the input-output pairs (x_i, y_i) based on the following model:

$$y = x\beta + \varepsilon$$

where $x \in \mathbb{R}^{800}$, $y \in \mathbb{R}$ and β is a sparse vector, i.e. there are only three non-zero coordinates of β.

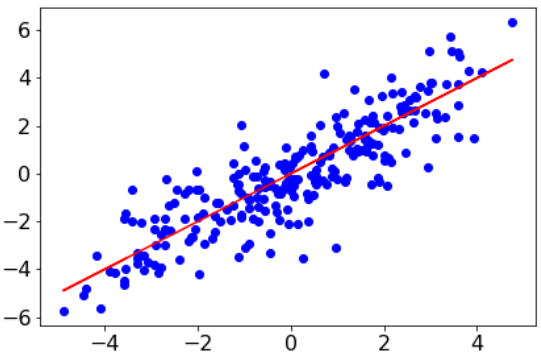


Figure: Linear Regression

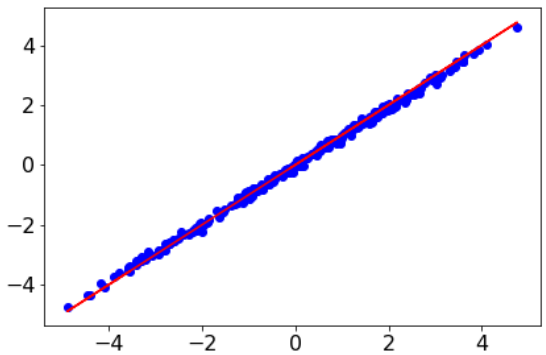


Figure: Lasso Regression

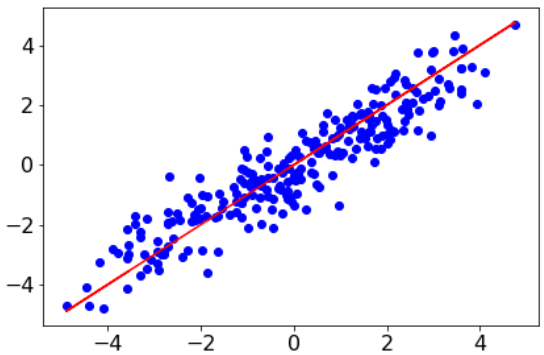


Figure: Ridge Regression

Non-Linear Regression

Underfitting issue

An underfitted model is a model where some parameters or terms that would appear in a correctly specified model are missing. Underfitting would occur, for example, when fitting a linear model to non-linear data.

Basis Expansion

$$y = f(x) + \varepsilon;$$

$$f(x) \approx \mathscr{L}(\phi_1(x), \dots, \phi_n(x)) = \sum_{i=1}^{n} \theta_i \phi_i(x), x \in \mathbb{R}^d.$$

Fixed basis functions (Features), e.g.

- Polynomial basis: $x^0, x^1, x^2, \dots, x^n$;
- Spline basis...

Alternative approach

Propose non-linear parameterized functions, e.g. Neural Network.

Outline

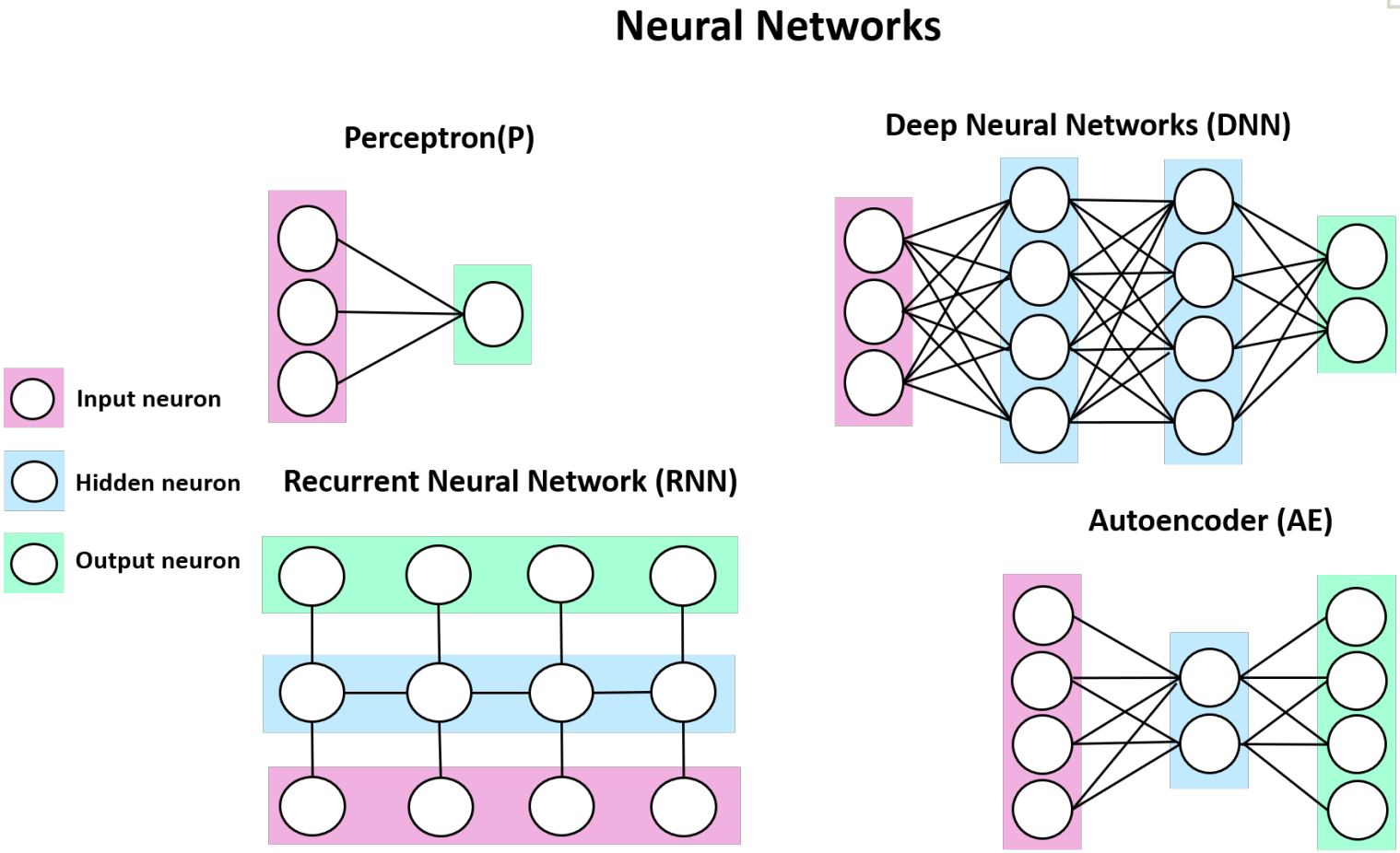
1 A Primer to Machine Learning and its Financial Applications

2 Supervised Learning
- Linear Regression
- Non-linear Regression

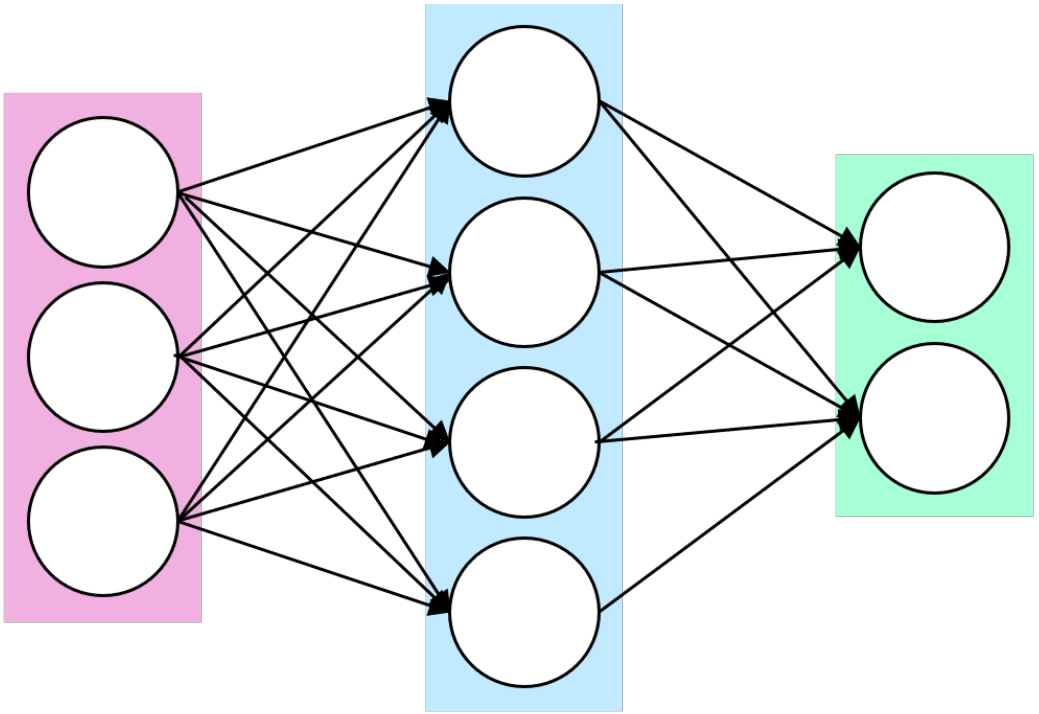
3 Neural Networks
- Shallow Neural Network
- Deep Neural Network
- Recurrent Neural Network

4 Applications of Neural Networks in Finance

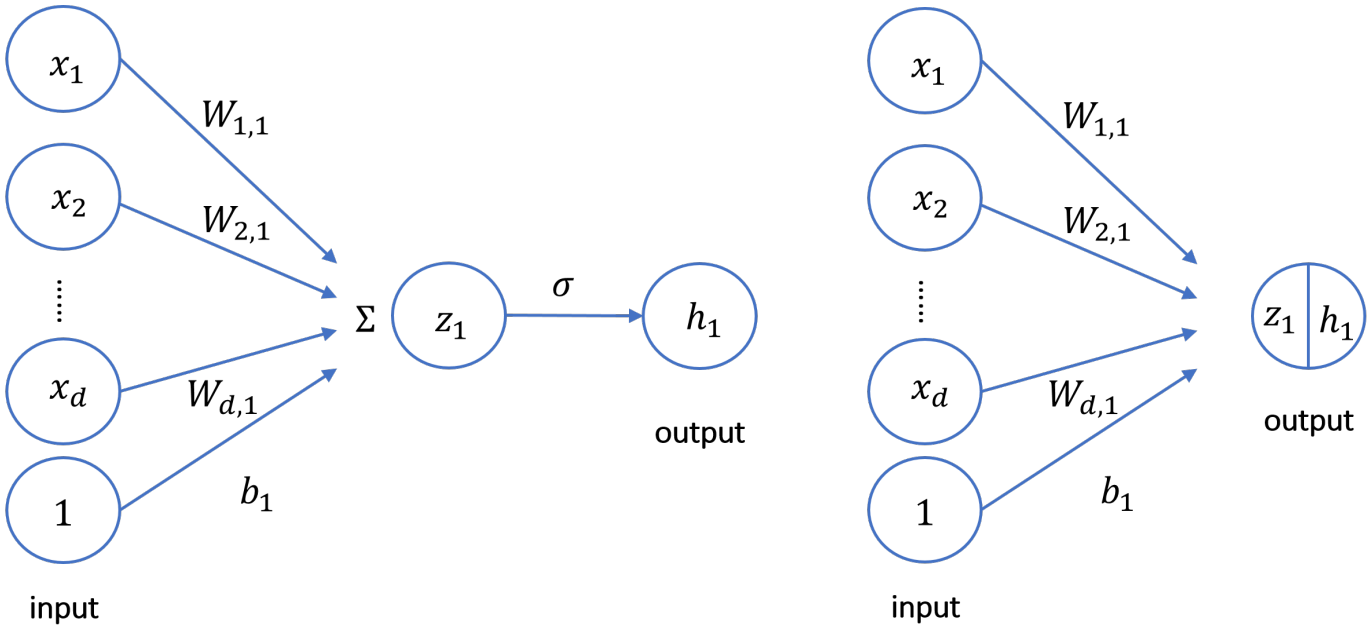
Neural Networks



Shallow Neural Network



Dense layer



$$z_1 = \sum_{i=1}^{n_1} W_{i,j} x^{(i)} + b_1, h_1 = \sigma(z_1)$$

Figure: Dense layer.

Deep Neural Network(DNN)

Model

- Input layer (h^0): $h^0(x) = x$.
- Hidden layer ($\{h^l\}_{l=1}^{L-1}$);
- Output layer (h^L).

$$\begin{aligned} z^{l+1} &= W^l h^l + b^l, \\ h^{l+1} &= \sigma_l(z^{l+1}), \end{aligned}$$

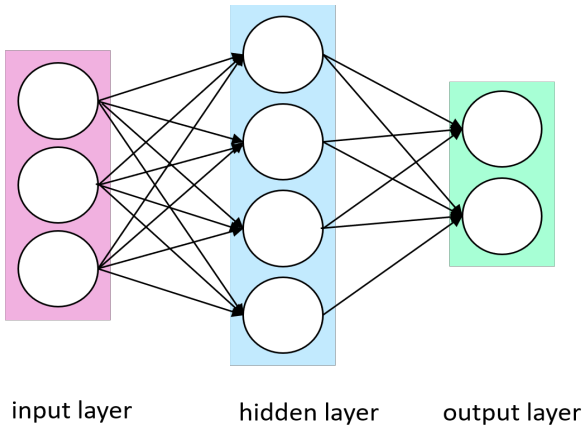
where σ_l is an activation function, which is usually non-linear, e.g. the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. The parameter set $\theta = (W^l, b^l)_{l=1}^{L}$.

Neural Networks

Universality

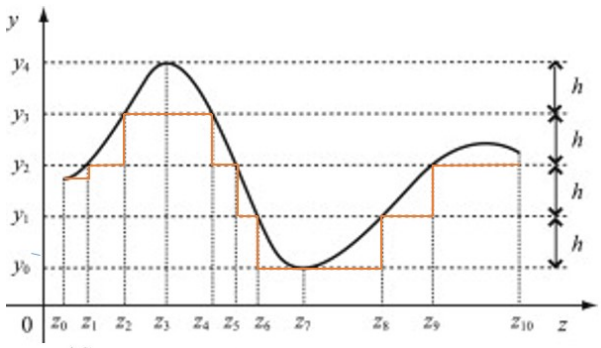
Neural network is NOT a Magic! It is about *function approximation theory*.

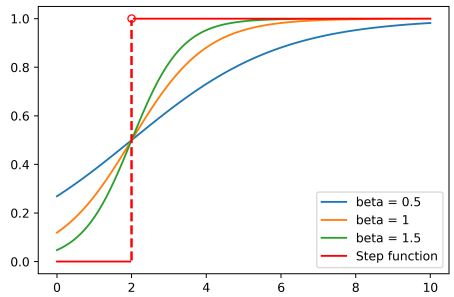
$$f \approx f_\Theta$$



$$f_\Theta(x) = \sum_{i=1}^{M} \alpha_j \sigma\left(\sum_{j=1} \beta_j^T x + \theta_j\right),$$

where $\Theta := \{\alpha_j, \theta_j, \beta_j\}_{j=1}^{N}$.





Neural Networks

Theorem (Universal Approximation Theorem[1])

Let σ be any continuous discriminant function (e.g. sigmoid function). The finite sum of the form

$$\sum_{i=1}^{M} \alpha_j \sigma \left(\sum_{j=1} \beta_j^T x + \theta_j \right)$$

are dense in $C(I_n)$. In other words, given any $f \in C(I_n)$ and $\varepsilon > 0$, there is $f_\Theta(x)$ of the above form, for which

$$|f_\Theta(x) - f(x)| < \varepsilon, \forall x \in I_n$$

where $\Theta := \{\alpha_j, \theta_j, \beta_j\}_{j=1}^{N}$.

Caution

Universality does not mean usefulness. Over-complicated model may lead to the overfitting issue.

Recurrent Neural Network (RNN)

RNN Architecture

- Input Layer $(x_t)_{t=1}^T \in \mathbb{R}^{p_0 \times T}$:
- Hidden Layer $(s_t)_{t=1}^T \in \mathbb{R}^{p_1 \times T}$:

$$s_t = h(\boldsymbol{U} x_t + \boldsymbol{W} s_{t-1}).$$

- Output Layer $(o_t)_{t=1}^T \in \mathbb{R}^{p_2 \times T}$:

$$o_t = g(\boldsymbol{V} s_t).$$

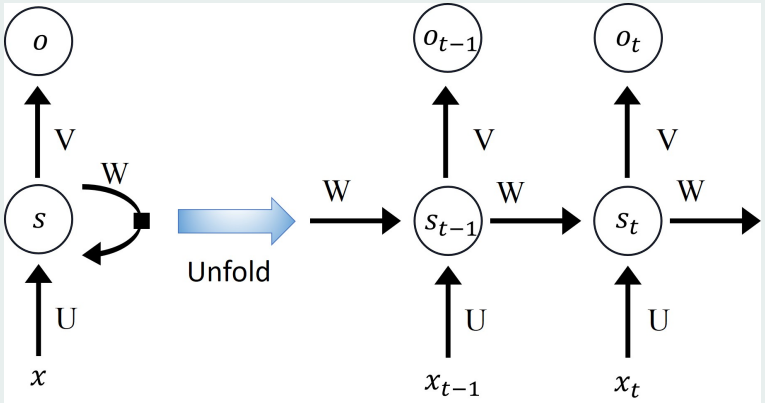


Figure: Recurrent Neural Network

where $(\boldsymbol{U}, \boldsymbol{W}, \boldsymbol{V})$ are RNN parameters to learn from data.

Long short-term memory (LSTM)

LSTM is a variant of RNN models to use the gate mechanism to better capture long term temporal dependency.

Parameter Optimization in Neural Networks

We aim to find optimal θ to minimize the function $L(\theta)$ provided L is differentiable.

- Gradient Descent method: Initialize θ_0. We update θ_n by the $\theta_{n+1} = \theta_n - \eta \nabla L(\theta_n)$, where $\nabla L(\theta)$ is the derivative of $L(\theta)$ and $\eta > 0$ is a learning rate.
- Main idea: $L(\theta_n) \Downarrow n$ and $\lim_{n\to\infty} \nabla L(\theta_n) = 0$.

$$L(\theta_{n+1}) - L(\theta_n) \approx -\eta |\nabla L(\theta_n)|^2$$

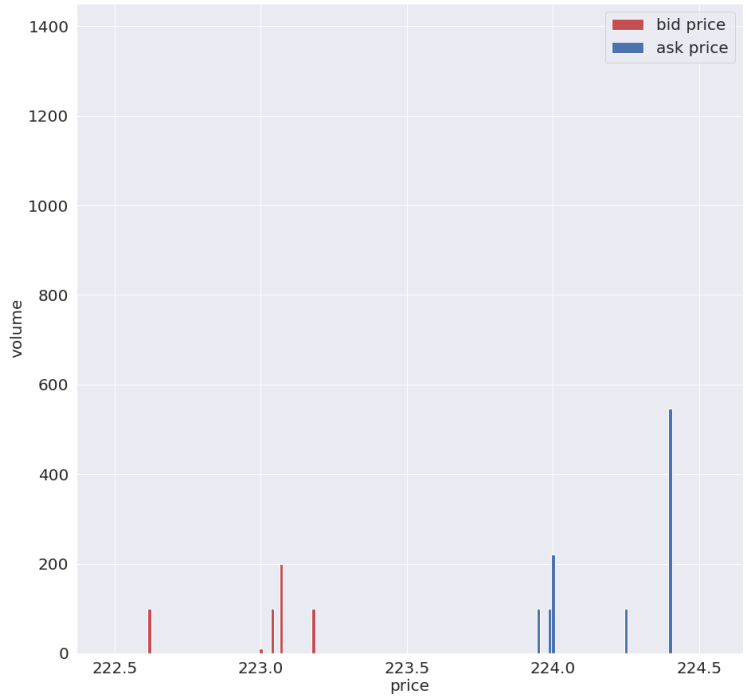
- Mini-batch Gradient Descent: a stochastic approximation of Gradient Descent to reduce the computational burden.
- Gradient calculation ([3, 2]):

	DNN	RNN
Method	Backpropogation	Backpropogation through time

Outline

1. A Primer to Machine Learning and its Financial Applications
2. Supervised Learning
   - Linear Regression
   - Non-linear Regression
3. Neural Networks
   - Shallow Neural Network
   - Deep Neural Network
   - Recurrent Neural Network
4. Applications of Neural Networks in Finance

Application: Predicting future price movement



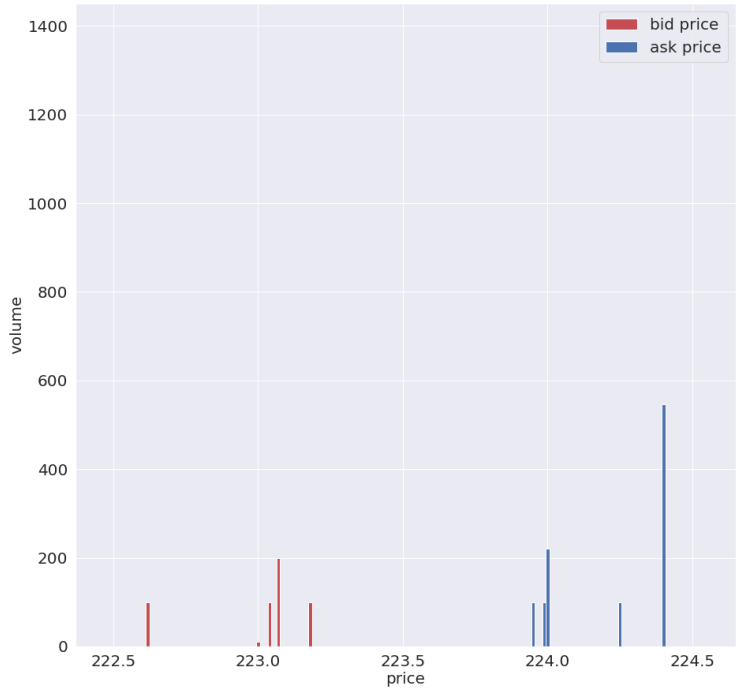


Figure: Left: the video of the Limit order book movements; Right: the snapshot of limit order book, which is composed with the bid/ask prices/volume up to certain depth.

Pipelines of ML application

Question

Given the history of the limit order book of one asset, how to build an accurate model to predict whether the next mid-price is going up or down?

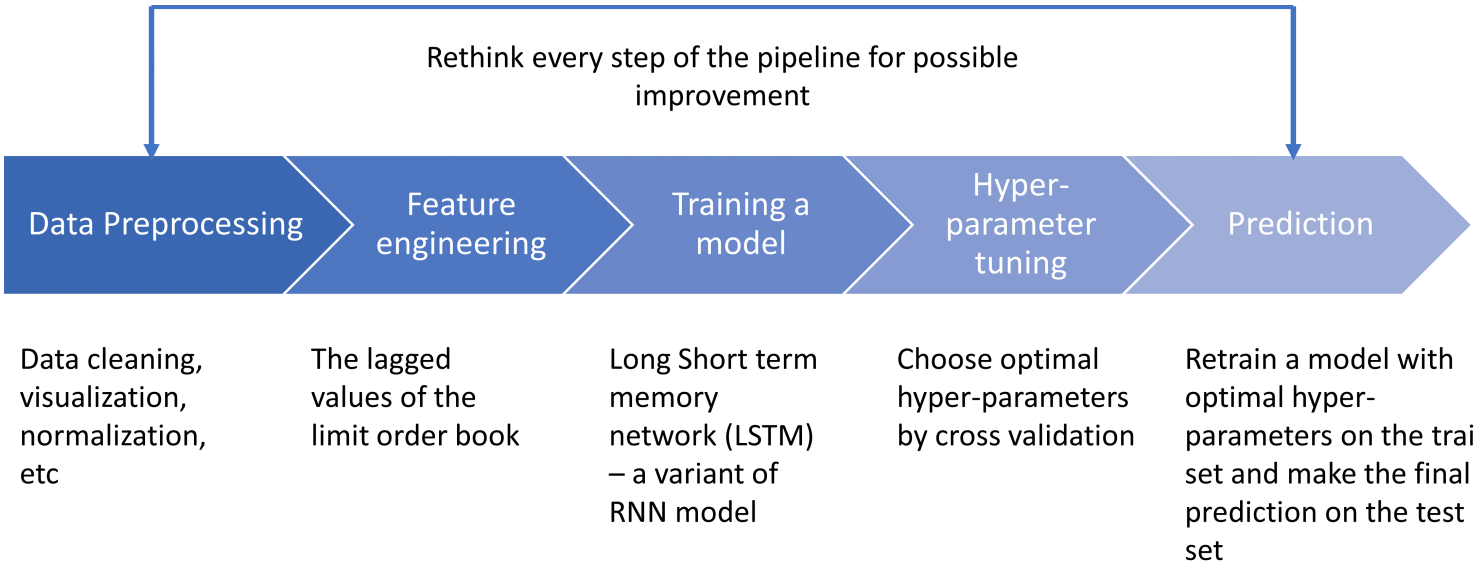


Figure: To tackle this problem, we choose the features and model for this task following [4].

Let LOB_t denote the limit order book information including the bid/ask prices/volumes of limit order up to depth 10 at the t^{th} time when the mid-price changes.The mid-price is the average of the best bid and ask price.

DataSig

3-layer LSTM for binary classification

Data: Input-Output pairs $(X_t, Y_t)_{t=}^{N}$, where
$X_t := (\text{LOB}_i)_{i=t-49}^{t} \in \mathbb{R}^{40\times 50}$ and
$Y_t := \mathbf{1}(\text{MidPrice}_{t+1} > \text{MidPrice}_t) \in \{0, 1\}$

Model: $f_\theta(X) \sim \mathbb{P}(Y|X)$.

Loss function: cross entropy

Optimization: mini-batch Gradient Descent.

Prediction: $\hat{Y} = \arg\max_{i\in\{0,1\}} \underbrace{\hat{P}(Y = i|X)}_{f_{\hat{\theta}}^{(i)}(X)}$.

Validation: Accuracy, Precision, Recall, f1 score.

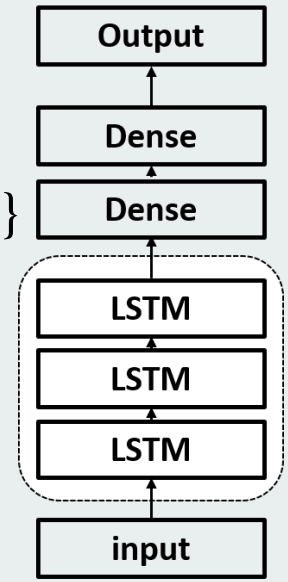


Figure: Model architecture f_θ

Implementation of Recurrent Neural Network

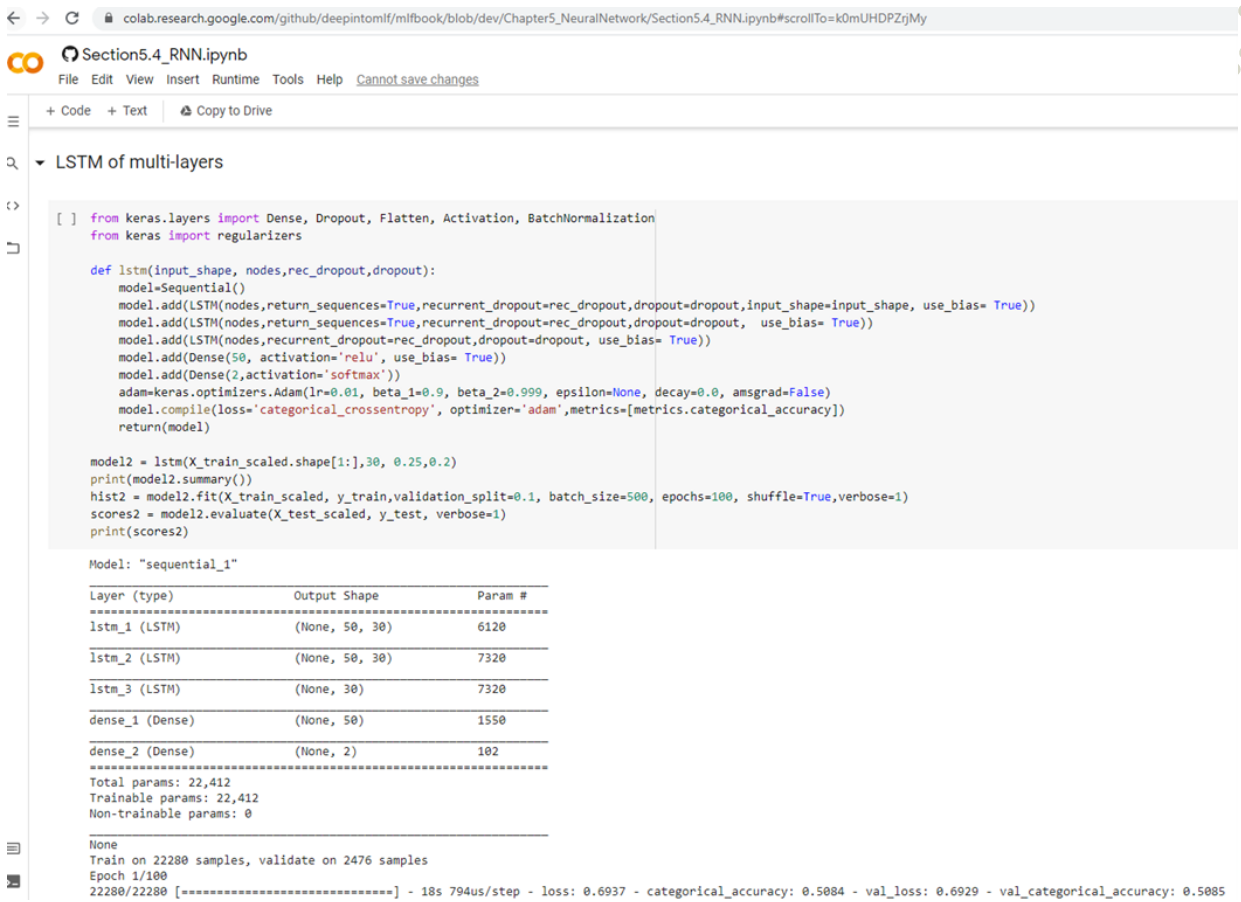


Figure: The snapshot of the Python notebook to showcase how to apply the Recurrent Neural Network to this prediction task step by step. Interested readers can download this notebook as Section5.4_RNN.ipynb under the folder Chapter5_NeuralNetwork/ at the GitHub repository `https://github.com/deepintomlf/mlfbook.git`.

Numerical Results

	accuracy	precision	recall	f1 score
One-Layer LSTM	56.32%	54.12%	76.46%	62.11%
Three-Layer LSTM	57.57%	55.24%	75.27%	62.64%

Table: The performance of the single-layer LSTM model and the three -layer LSTM model for predicting the next price direction in the test data.

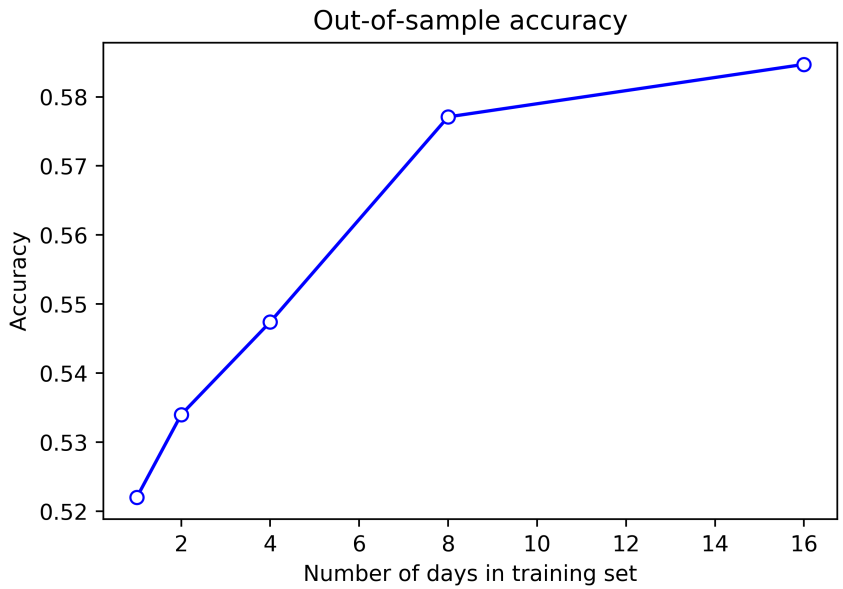


Figure: The plot of the out-of-sample accuracy v.s. the number of days in the training set.

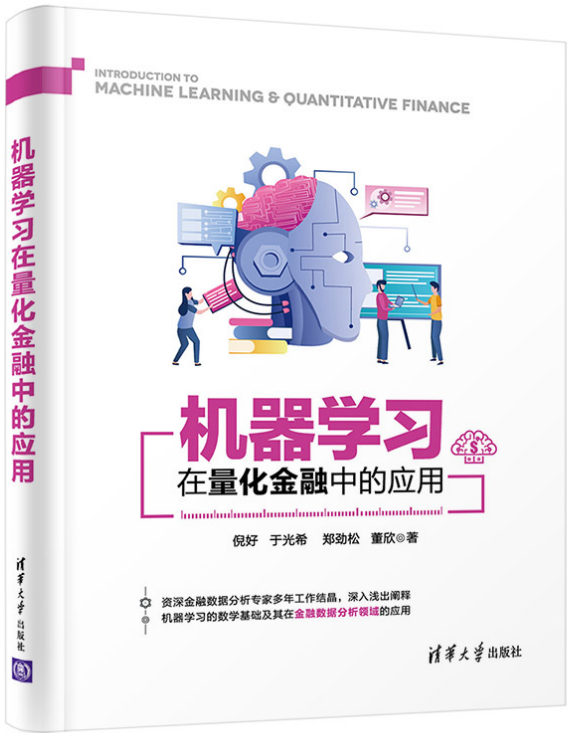
Conclusion

Takeaway Messages

- The general framework of Supervised Learning include data, model, loss function, optimization, prediction, and validation.
- Regulation methods, e.g. Lasso or Ridge regression, can help with the overfitting issues.
- Linear models and Neural networks including shallow/deep neural network and recurrent neural network are discussed.
- The financial application of neural network showcase the pipeline of ML projects in finance.

This tutorial is based on my new book entitled *"An introduction to machine learning in quantitative finance"* jointed with X. Dong, J. Zheng and G. Yu. It provides a systematic and rigorous introduction to machine learning and includes ample examples of financial applications. The GitHub repository for the supplementary python codes of the book is public available at https://github.com/deepintomlf/mlfbook.git.

We hope that this book can help the beginners in ML and quantitative finance to get a quick start and enjoy their journey of applying ML to financial problems!



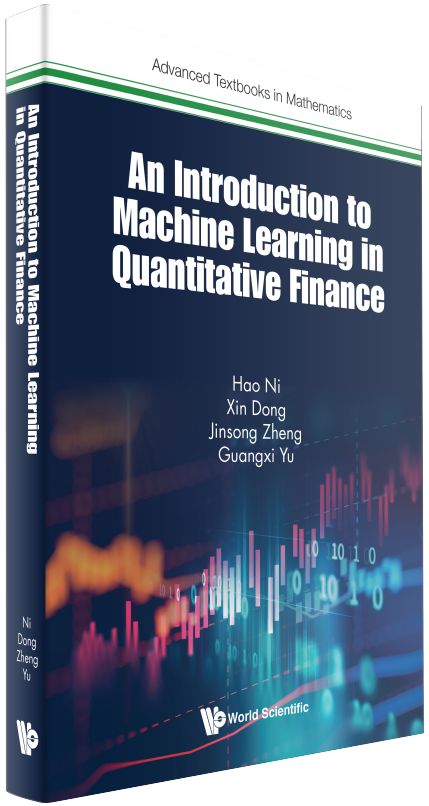


Figure: (Left) Chinese version: Tsinghua University Press [2]; (Right) English version: World Scientific Press [3].

Thanks for your attention!

Now it is time for the Q&A session.

References I

G Gybenko.
Approximation by superposition of sigmoidal functions.
Mathematics of Control, Signals and Systems, 2(4):303–314, 1989.

Hao Ni, Xin Dong, Jinsong Zheng, and Guangxi Yu.
An Introduction to Machine Learning in Quantitative Finance (Chinese version).
Tsinghua University Press, 2021.

Hao Ni, Xin Dong, Jinsong Zheng, and Guangxi Yu.
An Introduction to Machine Learning in Quantitative Finance (English version).
World Scientific, 2021.

Justin Sirignano and Rama Cont.
Universal features of price formation in financial markets: perspectives from deep learning.
Quantitative Finance, 19(9):1449–1459, 2019.