Imperial College London

Department of Mathematics

Fx Trading Strategy Using Rough Path Signature

Author: Lidan Xing (CID: 01935185)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2020-2021

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Acknowledgements

Foremost, I would like to express my warmest thank you to my industrial supervisor from UBS, Gordon Lee, who made this work possible. His expert knowledge and enthusiasm motivate me all the time in this thesis. When I encountered difficulties, he was always there to provide the guidance and encouraged me to keep trying. I will remember every inspiring moment I discuss the paper with him, and this is a very precious journey for me.

Moreover, I would like to express my sincere gratitude to my internal supervisor, Dr Cristopher Salvi, whose academic knowledge, patience, timely feedback gave me the confidence to overcome challenges. It is also my honor to be his first student and learn from him.

In addition, I am very grateful to Dr Jack Jacquier for sharing Fx data and contributing to this MSc programme. At the same time, great thank you for each staff's contribution to this programme which makes me benefit a lot.

Last but not least, I want to express my deepest gratitude to my parents, sisters, and grandparents, who have always supported me. Their encouragement and understanding gave me the courage to overcome difficulties. Thank you, Karen, my close friend, for being with me as always. Thanks to Xiaofu, Xiaoming, and Viola for their daily encouragement, and every moment of studying with them made my Master's journey more precious.

Abstracts

This paper investigates the ability of signature transformation to extract information from time-series data, and take signature as features to forecast the short-term price movement in the largest and most liquid foreign exchange market. The optimal signature features are determined by examining the predictive power of various combinations of time-series data streams and model parameters. The predictive classification models are established by Random Forest (RF), Extreme Gradient Boosting (XGB) and Long-Short term memory (LSTM) neural network, and its practical application is evaluated by backtesting trading strategies on 20 foreign exchange currency pairs.

Contents

List of Figures

List of Tables

Introduction

Currencies markets is an over-the-counter (OTC) global marketplace that transacts 24 hours per day and closing only weekend, where participants, such as central banks, hedge funds, retail forex dealers, can buy, sell, exchange based on the exchange rates. From Triennial Central Bank Survey (2019), the forex market is the largest and most liquid financial market in the world according to its trading volume, and the worth of the entire global forex trading market is estimated to $2409 trillion [1]. Moreover, refer to Yan and Ouyang (2018), forex market prediction is an exciting but challenging topic in the financial field because of the complex characteristics of financial time series, such as non-linearity, non-stationary, and sequence correlation [2]. To construct a scientific prediction financial time series model, an effective method of capturing the complex features of financial data and powerful ability to learn features is necessary.

For extracting information from financial time series data, there are two primary schools based on financial analysis in the market prediction: fundamental analysis and technical analysis. The fundamental analysis in the forex market is to take the economic and political factors for countries and prefer to be used in long term investments. Technical analysis is to evaluate the price by generating statistics based on the past time series data generally including open/high/low/close/volume data streams, which is usually used by traders to determine market future performances, in the meantime, it is sometimes criticized because of subjectivity. Ratto (2018) leveraged technical analysis and machine learning to construct a trend prediction model, which achieved interesting prediction performances [3]. Therefore, we first build a technical analysis based prediction and take it as a benchmark.

Apart from financial analysis based predictions, the signature is a faithful mathematical method to transform the time series data, and Hao Ni and Terry Lyons (2013) pointed out that the possibility of using signature method in financial data information extraction and in machine learning application [4]. Moreover, Gyurko and Lyons (2013) identified atypical market behaviour and characterised the market impact of different parent orders on the FTSE 100 Index by using signature capturing the information from the historical financial time series [5]. Hence, we will try different approaches to establish various signature-based prediction models, and choose the appropriate approach in this scenario, as well as compare it with the benchmark model (technical analysis based).

Several machine learning methods are used in classification problems, such as logistic regression, decision tree, ensemble learning, neural network, and support vector machine. In this paper, Random Forest (RF), eXtreme Gradient Boosting (XGBoost), and Long-short Term Memory (LSTM) machine learning methods are used to establish predictive classification models. RF and XGBoost classifiers are both ensemble tree methods, combining several tree models to an optimal predictive model, but they construct tree structures in different approaches. Random forest classifier builds lots of independent trees by bootstrapping and gathers the results from each tree in the end, while XGBoost method takes the additive training method to optimise the tree at each iteration by adding a weaker classifier, which has been widely used in financial market prediction by Nobre (2019) and Yun (2021) because of its execution speed and model performance [6, 7]. LSTM is a recurrent neural network (RNN) architecture, but it introduces the memory cell, which deals with the short-term memory problem of the RNN, and is suitable to grasp the information of sequential data over time. In recent years, the LSTM network has proved to have a well-achieved performance in financial data prediction, given its ability to distinguish between short-term and long-term information. Kim and Kang (2019) compared various deep learning methods such as multilayer perceptron (MLP) with the attention LSTM framework in financial time series prediction [8], and Nelson (2017) obtained an average of 55.9% of accuracy in 15-minute trend prediction of stock in IBoverspa Index [9].

This paper aims to build a robust predictive classification model to predict the short-term up/down trend of various currency pairs in currency markets. We establish binary classification predictive models by three machine learning methods (Random Forest, XGBoost and LSTM) and evaluate the predictive power for these three methods in terms of their performances. In addition to evaluate the performance of the predictive models, in financial markets, it is crucial to use the predictive models to determine trading signals; we, therefore, take the backtesting of trading strategy to evaluate the practical application of models.

In chapter 1, we introduce the definition and properties of the signature, as well as its application in machine learning, and discuss why we can use signature as a feature set in application.

Chapter 2 shows how we use three machine learning methods (Random Forest, XGBoost, and LSTM) to establish technical analysis based, time series based, and signature-based classification prediction models in the AUDCAD currency pair. Moreover, this chapter elaborates different approaches clearly in the establishment of time series based and signature-based features. In the end, based on the performances of the various predictive classification models, we choose the appropriate model, close signature + technical analysis based prediction, in 5-minute price movement prediction.

Finally, in chapter 3, we implement the simple trading strategy for 20 currency pairs by taking the chosen appropriate model to assess its practical performance in the financial market.

Chapter 1

Methodology

1.1 The Signature of a Path

A path $X_t := X(t) \in \mathbb{R}^d$ in euclidean space could be considered as a continuous function, mapping from $t \in [0,T]$ to $\mathbb{R}^d$. In the context of signature, it is generally assumed that paths are piece-wise differentiable to guarantee the properties and applications of signatures. Moreover, the denotation of a path $X_t \in \mathbb{R}^d$ is as

$$X_t : [0,T] \mapsto \mathbb{R}^d, \quad X_t = \{X_t^1, X_t^2, ..., X_t^d\}, \tag{1.1.1}$$

where $\{X_t^1, X_t^2, ..., X_t^d\}$ are coordinate paths. In quantitative finance, the path X_t is usually the multivariate time series data over the time interval $[0,T]$, where $X_t^1, X_t^2, ..., X_t^d$ represent continuous financial data such as open price, close price, trading volume, mid-price at time t.

1.1.1 The Integral of a Path

Signatures could be considered as a mathematical method to extract the information of paths, and one of the basic mathematical intuitions to a continuous path X_t is integration. For a one-dimension path $X_t : [0,T] \mapsto \mathbb{R}$, the integral of a path X_t is defined by

$$I = \int_0^T dX_t = \int_0^T \dot{X}_t dt = X_T - X_0. \tag{1.1.2}$$

If we consider a two-dimensional path $X_t = \{X_t^1, X_t^2\} \in \mathbb{R}^2$, then we have an iterated integral as

$$I = \int_0^T X_t^2 dX_t^1 = \int_0^T X_t^2 \dot{X}_t^1 dt, \tag{1.1.3}$$

which is essential to introduce the path signature as discussed following.

1.1.2 Signatures

As mentioned in above section, now we extend the path into a d-dimension, i.e., $X_t = \{X_t^1, X_t^2, ..., X_t^d\}$, then we introduce an quantity

$$S(X)_{0,s}^i = \int_{0<t<s} dX_t^i = X_s^i - X_0^i, \quad i = 1, 2, ..., d,$$

which represents a integral with respect to the i-th coordinate path of $X_t \in \mathbb{R}^d$ on $t \in [0,s]$, and it describes the information of a coordinate path over time interval $[0,s]$, i.e., the increment of path X_t along to i-th coordinate.

Moreover, since the path X_t is d-dimensional, we aim to extract the information not only in each coordinate, any pairs and combinations of coordinate paths also provide nontrivial information of the path X_t. First of all, consider the pairs of coordinate paths, X_t^i and X_t^j, where $i, j \in \{1, 2, ..., d\}$, then we define the double-iterated integral as

$$S(X)_{0,s}^{i,j} = \int_{0<t<s} S(X)_{0,t}^i dX_t^j = \int_{0<r<t<s} dX_r^i dX_t^j. \tag{1.1.4}$$

This double-iterated integral figures the information of the pairs of coordinate paths, and it is intuitive to consider a k-fold iterated integral to obtain the information among k coordinate paths, $\{X_t^{i_1}, X_t^{i_2}, ..., X_t^{i_k}\}$ with $i_1, ..., i_k \in \{1, 2, 3, ..., d\}$, which is defined recursively as

$$k = 3, \quad S(X)_{0,s}^{i_1,i_2,i_3} = \int_{0<t<s} S(X)_{0,t}^{i_1,i_2} dX_t^{i_3} = \int_{0<t_1<t_2<t_3<s} dX_{t_1}^{i_1} dX_{t_2}^{i_2} dX_{t_3}^{i_3}, \tag{1.1.5}$$

$$\cdots$$

$$k = n, \quad S(X)_{0,s}^{i_1,i_2,...,i_n} = \int_{0<t<s} S(X)_{0,t}^{i_1,i_2,...,i_{n-1}} dX_t^{i_n} = \int_{0<t_1<...<t_n<s} dX_{t_1}^{i_1} ... dX_{t_n}^{i_n}, \tag{1.1.6}$$

where $n \in \mathbb{N}$, and $S(X)^{i_1,i_2,...,i_n}$ is a mapping: $[0,T] \mapsto \mathbb{R}$, which is a real-valued path, and these quantities are the elements of signature.

Definition 1.1.1 (The signature of a path)**.** The signature of a continuous path $X_t : [0,T] \mapsto \mathbb{R}^d$, is denoted as a collection of all iterated integrals of the path $X_t \in \mathbb{R}^d$, which is a infinite and real-valued sequence as defined by

$$S(X)_{0,T}^W = (1, S(X)_{0,T}^1, S(X)_{0,T}^2, ..., S(X)_{0,T}^d, S(X)_{0,T}^{1,1}, S(X)_{0,T}^{1,2}, ...) \tag{1.1.7}$$

where 1 is the "zeroth" term by conventionally, the subscript of $S(X)_{0,T}^W$ is a infinite collection of all multi-indexes

$$W = \{(i_1, ..., i_k) | k \geq 1, i_1, ..., i_k \in \{1, ..., d\}\}.$$

1.1.3 The Properties of Signatures

Now, we highlight the following properties related to the applications in this paper.

- **Invariance under time reparametrisations** Let consider the path X_t and a reparametrisation $\phi : [0,T] \mapsto [S,U]$, $0 < t \leq T$, $(i_1, \cdots, i_k) \in W$, then we have

  $$\int_{0<t_1<\cdots<t_k<T} dX_{t_1}^{i_1} \cdots dX_{t_k}^{i_k} = \int_{S<\phi(t_1)<\cdots<\phi(t_k)<U} dX_{\phi(t_1)}^{i_1} \cdots dX_{\phi(t_k)}^{i_k}. \tag{1.1.8}$$

  This shows that the element of path signature, $S(X)_{0,T}^{i_1,\cdots i_k}$, is invariant under the time reparametrisations of path $X_t, t \in [0,T]$ by reparametrising a surjective, continuous, non-decreasing function ϕ. Thus, we can claim that the augmentations such as lead-lag transformation remains the signature of the data stream 1.2.3.

- **Shuffle Product** Another fundamental properties of the signature is shuffle product, which shows that any product of iterated integrals could be represented by a linear combination of iterated integrals. Consider a path $X_t : [0,T] \mapsto \mathbb{R}^d$, $0 \leq s < u \leq T$, two multi-indexes $W_1 = (i_1, \cdots, i_k)$, $W_2 = (j_1, \cdots, j_m)$ with $(i_1, \cdots, i_k), (j_1, \cdots, j_k) \in \{1, \cdots, d\}$, and $W_1, W_2 \subset W$, then there exists a $W_3 \subset W$ such taht

  $$S(X)_{s,u}^{W_1} S(X)_{s,u}^{W_2} = \sum_{W_3 \in W} S(X)_{s,u}^{W_3}. \tag{1.1.9}$$

  This property implies that the product of the two terms of signatures could be expressed by the combination of higher order elements of signatures, which enables us to apply the signature in the regression analysis.

- **Chen's identity:** Let $X_t : [0,T] \mapsto \mathbb{R}^d$, there there exists a binary operation $\otimes$ such that for any $0 \leq 0 < s < t < u \leq T$, we have

  $$S_{s,t}(X) \otimes S_{t,u}(X) = S_{s,u}(X).$$

- **Signature of linear paths:** If a linear path $X_t : [0,T] \mapsto \mathbb{R}^d$, which is denoted as $X_t = a + bt$, for some $a, b \in \mathbb{R}^d$, then for all $t \in [0,T]$ and $W = (i_1, \cdots, i_k)$, we have

  $$S(X)_{s,t}^W = \frac{(t-s)^k}{k!} \prod_{j=1}^{k} b_{ik}.$$

- **Uniqueness** Hambly and Lyons (2010) showed that there is a tree-like equivalence between $S(X)_{s,t}$ and the function $u \mapsto X_u - X_s, u \in [s,t]$ [10]. Moreover, the existence of $i \in \{1, \cdots, d\}$ such that the signature term $S(X)^i_{s,t}$ is monotone increasing, which is sufficient to the uniqueness. This property implies a unique relationship between path and its signature, and the truncated signature could be seen as a projection of the path from a infinite dimensional space to a lower dimension, and the first few terms of signatures extracts the most information of the path, this is why we use the truncated signature in the application, which could not only contain the useful information of the path but also reduce the dimensions.

1.2 Practical Application of Signature

In above sections, we discuss how to compute the signature of a continuous path theoretically, while in the real-life problem, we usually observe a sequential data stream $\{X_i\}$. Thus, we need to transform discrete data streams to a continuous path X_t, then to get the signature according to its definition. There are several transformation methods applied in the signature computation as shown below.

1.2.1 Signature Augmentation

The time-series data is a one-dimensional sequence with an index of time order, represented by $\{X_i\}_{i=1}^n$ with n successive equally spaced time points (per minute/day/year). Morrill and Fermanian (2020) grouped four categories for variations in the signature of a path, which are *augmentation, windows, transforms,* and rescaling [11].

- **Augmentations:** These transform time series into one or more new series, and then compute signatures.
- **Windows:** Slide the time series into several windows and apply the signature transform locally.
- **Transform:** Different techniques to extract the information, including signature and logsignature.
- **Rescaling:** Approaches to normalised the terms in signature, we will take pre-signature scaling in this paper, which is to multiply the input data stream by some scaling factor $\alpha \in \mathbb{R}$.

A simple augmentation is to add time to the original data $\{X_i\}_{i=1}^n$, which transforms the original data from a one-dimensional sequence into two-dimensional sequence by adding a new sequence.

Definition 1.2.1 (AddTime Augmentation)**.** Given a sequence data stream $\{X_i\}_{i=1}^n$, then we transform it by taking

$$\mathbf{X}^{\mathbf{AddTime}} = \{t_i, X_i\}_{i=1}^n = \{(0, X_1), (\frac{1}{n}, X_2), (\frac{2}{n}, X_3), ..., (\frac{n-1}{n}, X_n)\}, \tag{1.2.1}$$

where $\{t_i\}_{i=1}^n$ is a time sequence of length n adding to the original data.

Hambly and Lyons (2010) proposed adding time information to the original time series, and makes sure the uniqueness of the signature [10].

Definition 1.2.2 (Basepoint Augmentation)**.** Given a sequence data stream $\{X_i\}_{i=1}^n$, then we tailor it by adding a zero at the beginning of the time series:

$$\mathbf{X}^{\mathbf{Basepoint}} = \{0, X_1, \cdots, X_n\}, \tag{1.2.2}$$

which makes the signature more sensitive to the time series proposed by Kidger and Morrill (2020).

Another augmentation approach is the Lead-Lag transformation, which maps a one-dimensional sequence into two-dimensional sequence by combining lead and lag sequences, which is given by:

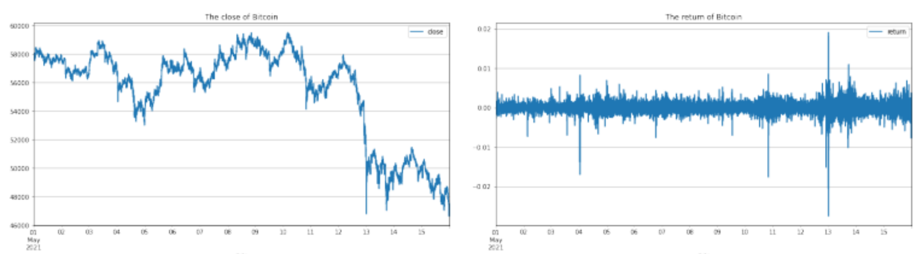
Definition 1.2.3 (Lead-lag augmentation)**.** Consider a one-dimensional sequence data $\{X_i\}_{i=1}^n$, then we take the transformation as

$$\mathbf{X}^{\mathbf{Lead-Lag}} = \{X^{\mathbf{Lead}}, X^{\mathbf{Lag}}\} = \begin{cases} X^{\mathbf{Lead}} = \{X_1, X_2, X_2, X_3, X_3, ..., X_{n-1}, X_{n-1}, X_n, X_n\}, \\ X^{\mathbf{Lag}} = \{X_1, X_1, X_2, X_2, X_3, ..., X_{n-2}, X_{n-1}, X_{n-1}, X_n\}, \end{cases} \tag{1.2.3}$$

where $X^{\mathbf{Lead}}$ and $X^{\mathbf{Lag}}$ are respectively $2(n-1)$ sequences, and comprising of a two-dimensional sequence data stream.

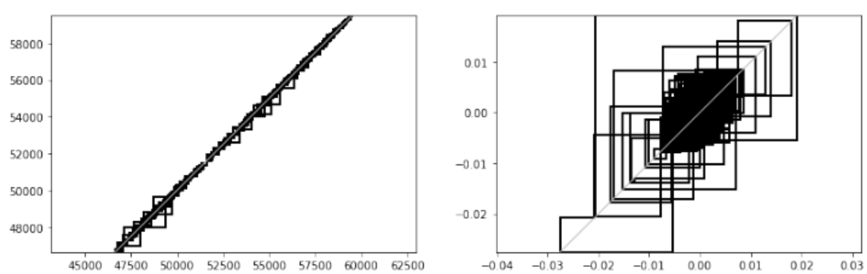
Refer to Chevyrev and Kormilitzin (2016), one key property of Lead-Lag augmentation is to capture the quadratic variation of the time series data, which is a important characteristic for financial data [12].

Given above signature augmentations we discussed, ***piece-wise linear*** interpolation and ***rectilinear*** interpolation are two main approaches to embed discrete data points into a continuous path. Now, we take the close price and log return of Bitcoin from 1st to 15th May as our sequential data stream (minute-by-minute data, see Figure 1.1), and apply the lead-lag augmentation of original data, then take piece-wise linear interpolation to obtain the a continuous lead-lag paths respectively (see Figure 1.2).



(a) Close price of Bitcoin (b) Log-return of Bitcoin

Figure 1.1: Close price and log-return of Bitcoin from 1st to 15th May (minute-by-minute data)



(a) Lead-lag transformation of close price (b) Lead-lag transformation of log-return price

Figure 1.2: Lead-lag transformation of close price and return of Bitcoin from 1st to 15th May (minute-by-minute data)

1.2.2 Signature as a Feature Set in Time Series Analysis

Financial data is a time-varying dynamics data, lots of methods such as Gaussian process regression model introduced in the financial time series analysis and volatility modelling, Han and Zhang (2016) aimed to use the stochastic nature of Gaussian process to capture the time-varying dynamics of financial times series [13]. Rough path theory is a mathematical method which models the interactions between highly oscillatory rough paths, and is recently used in the stochastic analysis fields. Signature is called a certain graded feature of a data stream and is a fundamental object in rough path theory. Lyons and Ni (2013) used the signature as a feature set in linear regression problem [4]. Moreover, the relationship of signature and Brownian motion validates the feasibility of using signature to capture the stochastic nature.

Theorem 1.2.4 (Uniqueness of signature of Brownian motion). *Let W be a standard d-dimensional Brownian motion, and $S(W)_{(0,t)}$ is the Stratonovich signatures of Brownian motion W up to time t, then all Brownian motion sample paths up to time t could be determined by their signature $S(W)_{(0,t)}$ up to time re-parameterisation almost surely.*

The signature could also be approximated by a linear function as the following theorem states:

Theorem 1.2.5 (Signature Approximation). *Let f be a function whose derivative exists in every point, and f is a continuous function from $S_1 \mapsto \mathbb{R}$, where S_1 is a compact subset of $S(V^p(J, \mathbb{R}^d))$, where $V^p(J, \mathbb{R}^d)$ is the space of continuous functions mapping from J to $\mathbb{R}^d$ with finite p- variation, and the element of it is a path. Given these, for every $\epsilon > 0$, there exists a linear function L such that for every $a \in S_1$,*

$$|f(a) - L(a)| \leq \epsilon.$$

The uniqueness of signature of a path we discuss before and Theorem 1.2.4 show the relationship between a path and its signature from deterministic and probabilistic perspectives. Hence, we can roughly says that there is a one-to-one correspondence of the path and its signatures, thus the smooth function on a path space can be regarded as a smooth function on a signature space. Besides, from Theorem 1.2.5 we know that any continuous function f on the signature spaces can be approximated by the a linear function on the signature. In other words, the linear function on the signature can be viewed as the basis functions for the smooth function on the signature locally. Moreover, in time series data analysis, signature could extract the path information effectively if the time reparametrisation does not influence its outputs because of the invariance under time reparametrisation of signatures. It is noticed that, in practical application, we only have the order-sequence time series data point at discrete time stamps. If we want to extract the path information, daily data is not enough for signature to summarise the information, which is why we use the data on a minutely basis in this paper.

1.2.3 Signature in Machine Learning

Machine learning is a computer programming applying statistical theory to optimise performance criterion by using past experience, then we use the model we train to predict the unseen future. In machine learning method, we firstly meet with a data mining problem, which requires us to extract useful information from a large amount of raw data as features, then uses these to train and make a prediction, aiming to determine a model with a high accuracy in out of sample.

As discussed, given a series of discrete data, embedding these data into a continuous path and compute its signatures is a useful approach to extract important information of original data. Thus, involving signature transformation in the data mining problem helps us to determine the features in machine learning process. The workflow is as shown below:

$$\text{discrete data} \xrightarrow[\text{interpolation}]{\text{augmentations}} \text{continuous path} \longrightarrow \text{signature of path} \longrightarrow \text{features of data.} \tag{1.2.4}$$

The sensitivity to the geometric shape of a path is the attribute of signature, which leads to grasp the characteristic features from sequential data, then the extracted features are used in the machine learning applications. In quantitative finance, time-series financial data ideally fits into the signature method since it is natural to determine a path for the time-ordered sequential data.

Chapter 2

FX Rate Movements Prediction

This chapter aims to deal with a classification predictive modelling problem in financial time-series data by using different classifiers, called the sequence classification, where we use some sequence of time-ordered inputs to predict a category for the sequence. Refer to Huynh and Dang (2017), the sequence classification can be more challenging since the inputs we select from historical data can vary in length and may have a high dimensionality [14].

There are three large types of *sequence classification methods*, *feature-based classification*, *sequence distance based classification*, and *model based classification*. What we are dealing with is the feature-based classification problem, in which the feature extraction plays a key role. In our framework, we evaluate the performance of 'signature method' in information extraction by evaluate the performance of signature based prediction models.

This chapter uses different feature extraction techniques (technical analysis and signature transform) to predict the direction movement of foreign exchange (Forex) exchange rates in different length trends. Moreover, by evaluating the statistics metrics, such as out-of-sample accuracy, cross-validation accuracy and $F1$-score, we will have more insights into the predictive power of length trends and the performance of the feature extraction techniques.

2.1 Framework Formalization

This section first discusses how to create features and outputs in a time series framework without and with signature methods. Then, we introduce statistics scores and the cross-validation method to construct an evaluation framework. Finally, we present the theorem of machine learning used (RF, XGBoost, LSTM) briefly.

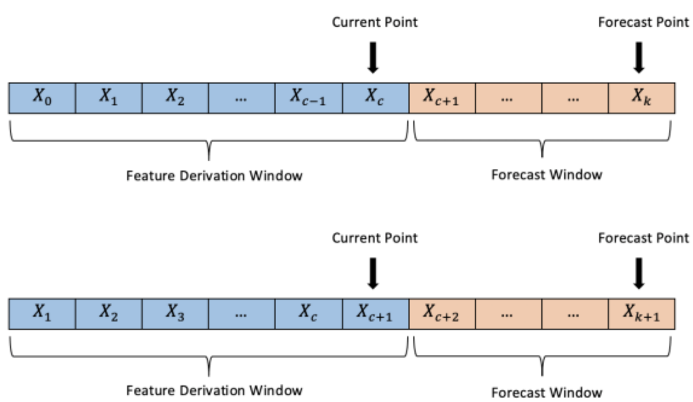


Figure 2.1: Two overlapping windows for a data stream $X = \{X_0, \ldots, X_{k+1}\}$

2.1.1 Predefined Categories

First of all, we slice the minute-by-minute foreign exchange rate data stream into multiples overlapping windows, and each window is composed of three parts, a feature derivation window, a current point and a forecast point, which is as shown in figure 2.1. Then, we label the output variable and extract features in each window to build the corresponding dataset.

Our aim is to predict the direction of the Forex market trend, and the market trend can be divided into upward and downward categories, respectively representing a strong rising and dropping. Considering a N-sample dataset, $\delta_t(l)$ represents a market movement between time $t+l$ and time t as

$$\delta_t(l) = PC(t+l) - PC(t), \tag{2.1.1}$$

where $PC(t)$ is the close price at time t, then we label each output $y_t(l)^i, i = 1, \cdots, N$ as

$$\{y_t(l)\}_{t=1}^{N} = \begin{cases} 1, & \text{if} \quad \delta(t) > 0, \\ -1, & \text{otherwise}, \end{cases} \tag{2.1.2}$$

representing the direction trend between at time $t+l$ and t.

2.1.2 Features Extraction

Features Extraction is a critical part of the feature-based classification, and we will derive feature vectors from feature derivation windows. Many attempts are experimented with for market trend prediction based on raw historical data, technical analysis, fundamental analysis, and sentiment analysis. Ratto and Merello (2018) have shown a relatively high directional accuracy in a 1-week ahead prediction of stock portfolio prediction by taking technical indicators and SVM [3]. Technical indicators are mathematical calculations to describe the information of historical price and helps analyse the market future movement. Moreover, the signature method is also a technique to extract information from multivariate time series data. Inspired by these, we first use technical indicators to predict the l-minute ahead forex rate market trend by taking different machine learning methods, next combining the signature method to evaluate its performance. Hence we aim to take different approaches to involve signature methods in features extraction to see if the signature method extracts valuable information and improves the predictive classification model.

Now, we first discuss how to work out the signature transform on d-dimensional time series denoted as $\bar{X} = (X_1, X_2, \cdots, X_h)$, where h is the length of the time series, and $X_i \in \mathbb{R}^d$ for each $i \in \{1, 2, .., h\}$, then a n sample dataset sliced from this time series is as shown in table 2.1 if we directly use $\bar{X}_{h_i} = (X_{i,1}, X_{i,2}, ..., X_{i,h_i})_{i=1,...,n}$ as features in each sample. Then, we involve the signature method into feature extraction by the following steps, and the data framework is therefore as shown in table 2.2 :

- (optional) take the AddTime/Lead-Lag augmentations for $\{X_{i,j}\}_{j=1}^{j=h_i}$ for each $i = 1, \cdots, n$
- embed $\{X_{i,j}\}_{j=1}^{j=h_i}$ into a continuous path P_i for each $i = 1, \cdots, n$
- compute the truncated signature $S(P_i)|_L$ for each P_i, $i = 1, \cdots, n$, where L is the truncation level
- use the terms of $S(P_i)$ for each path P_i, $i = 1, \cdots, n$, and remove the first constant term 1 and standardise the signatures by columns if needed

After introducing the signature transformation of the financial data, we turn to discuss the construction of technical analysis based features, time-series based features, and signature-based features.

2.1.2.1 Technical Indicators Features

A large number of technical indicators could be chosen to anticipate the future market trend. We choose ten technical indicators as features as shown in Table 2.3, which describes the trend, momentum, volatility and volume of the market. The technical indicators could be denoted as inputs as shown

$$\boldsymbol{TI}(t) = \{TI(t)_1, TI(t)_2, \ldots, TI(t)_f\}, \tag{2.1.3}$$

Table 2.1: Data Frame - Without Signature Transformation

	Features					Outputs
	t_1	t_2	$\cdots$	t_{h-1}	t_h	
$X_{h_1} =$	$X_{1,1}$	$X_{1,2}$	$\cdots$	X_{1,h_1-1}	X_{1,h_1}	y_1
$\bar{X}_{h_2} =$	$X_{2,1}$	$X_{2,2}$	$\cdots$	X_{2,h_2-1}	X_{2,h_2}	y_2
$\bar{X}_{h_3} =$	$X_{3,1}$	$X_{3,2}$	$\cdots$	X_{3,h_3-1}	X_{3,h_3}	y_3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\bar{X}_{h_{n-2}} =$	$X_{n-2,1}$	$X_{n-2,2}$	$\cdots$	$X_{n-2,h-1}$	$X_{n-2,h_{n-2}}$	y_{n-2}
$\bar{X}_{h_{n-1}} =$	$X_{n-1,1}$	$X_{n-1,2}$	$\cdots$	$X_{n-1,h_{n-1}-1}$	$X_{n-1,h_{n-1}}$	y_{n-1}
$\bar{X}_{h_n} =$	$X_{n,1}$	$X_{n,2}$	$\cdots$	X_{n,h_n-1}	X_{n,h_n}	y_n

Table 2.2: Data Frame - With Signature Transformation

Features						Outputs
$S_1^{(1)}$	$S_1^{(2)}$	$S_1^{(1,1)}$	$S_1^{(1,2)}$	$\cdots$	S_1^I	y_1
$S_2^{(1)}$	$S_2^{(2)}$	$S_2^{(1,1)}$	$S_2^{(1,2)}$	$\cdots$	S_2^I	y_2
$S_3^{(1)}$	$S_3^{(2)}$	$S_3^{(1,1)}$	$S_3^{(1,2)}$	$\cdots$	S_3^I	y_3
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$S_{n-2}^{(1)}$	$S_{n-2}^{(2)}$	$S_{n-2}^{(1,1)}$	$S_{n-2}^{(1,2)}$	$\cdots$	S_{n-2}^I	y_{n-2}
$S_{n-1}^{(1)}$	$S_{n-1}^{(2)}$	$S_{n-1}^{(1,1)}$	$S_{n-1}^{(1,2)}$	$\cdots$	S_{n-1}^I	y_{n-1}
$S_n^{(1)}$	$S_n^{(2)}$	$S_n^{(1,1)}$	$S_n^{(1,2)}$	$\cdots$	S_n^I	y_n

where represents f technical indicators computed at time t, and the responding outcome variable is $y_t(l)$ denoted as equation 2.1.2. Furthermore, in order to extract as much information as possible of historical prices, we compute the technical indicators with different time parameters N. For example, Exponential Moving Average (EMA), RSI are computed at $[10, 12, 14, 16, 18, 20, 22, 24, 26, 28]$ periods. Finally, we take the open/high/low/close price and volume at time t, together with the technical indicators as features, thus the matrix form of the dataset is shown as Table 2.4.

2.1.2.2 Time Series

First we start from the simple features, just take the historical close price of different lengths as features, to evaluate its predictive power, the data frame is shown in the Table 2.5. Moreover, technical indicators in Table 2.3 primarily are derived from multivariate time series, including high/low/close/volume time series. Hence, one intuition is to take the raw multivariate time series as features in the predictive model. However, for high-frequency time series forecasting, we generally need a longer timestamp for historical data. At the same time, a large number of inputs would cause the over-fitting problem and poor predictive performance.

2.1.2.3 Signature Based Features

There are five approaches to generate the signature based features as shown below.

1. ***Close price signature based***

   Our goal is to predict the direction movement of close price, and we would like to examine whether the signature method extracts the information of close price well, so we first compute the signatures of close price and take it as features. Suppose we denote the raw forex close price as $\{PC_i\}_{i=1}^h, PC_i \in \mathbb{R}$, where we slice a series of close price into several windows with a length H feature derivation window. In that case, the raw dataset is as shown in table 2.5. Finally, we follow the above signature transformation steps to compute its signature with augmentation and a truncation level. The structure of the final dataset is as shown in Table 2.2.

2. ***Multivariate time series signature based prediction***

Table 2.3: Technical Indicators

Categories	Technical Indicators	Formula
Trend	Simple Moving Average (SMA)	$SMA_N(t) = \frac{1}{N}\sum_{k=1}^{N} PC(t-k)$
	Exponential Moving Average (EMA)	$EMA_N(t) = PC(t)(\frac{2}{N+1}) + EMA_N(t-1)(1-\frac{2}{N+1})$
	Moving Average Convergence Divergence (MACD)	$EMA_N(t) - EMA_M(t), \quad N < M$
Momentum	Stochastic Oscillator (SO)	$100 \times [PC(t) - \min_N(PL)]/[\max_N(PH) - \min_N(PL)]$
	Williams(%R)	$100 \times [\max_N(PH) - PC(t)]/[\max_N(PH) - \min_N(PL)]$
	Relative Strength Index (RSI)	$100 - 100/(1+\frac{AG}{AL})$
Volatility	Bollinger Bands (BB)	$Upper\ Band(t) = SMA_N(t) + 2 \times std_N$ $Middle\ Band(t) = SMA_N(t)$ $Lower\ Band(t) = SMA_N(t) - 2 \times std_N$
	Average True Range (ATR)	$ATR(t) = ATR(t-1) * (N-1) + TR(t)/N$
Volume	Ease of Movement(EMV)	$Distance\ Moved = \frac{PH(t)+PL(t)}{2} - \frac{\max_N(PH)+\min_N(PL)}{2}$ $Box\ Ratio = Volume(t) \times (PH(t) - PL(t))/Scale$ EMV(t) $= Distance\ Moved/Box\ Ratio$
	Force Index (FI)	$FI_1(t) = (PC(t) - PC(t-1)) \times Volume(t)$ $FI_N(t) = EMA(FI_1(t))$

N : the time parameters,
$PO(t)/PH(t)/PL(t)/PC(t)/Volume(t)$: the open/high/low/close/price and volume at time t,
$\max_N(\cdot), \min_N(\cdot)$: the maximum, minimum value in the past N time period.

Table 2.4: Data Frame - Technical Indicators Features

Features						Outputs
$OHLCV(t)^*$			$TI(t)$			
$PO(t)$	$\cdots$	$Volume(t)$	$TI(t)_1$	$\cdots$	$TI(t)_f$	$y(t)$
$PO(t+1)$	$\cdots$	$Volume(t+1)$	$TI(t)_1$	$\cdots$	$TI(t)_f$	$y(t+1)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$PO(t+N-2)$	$\cdots$	$Volume(t+N-2)$	$TI(t)_1$	$\cdots$	$TI(t)_f$	$y(t+N-2)$
$PO(t+N-1)$	$\cdots$	$Volume(t+N-1)$	$TI(t)_1$	$\cdots$	$TI(t)_f$	$y(t+N-1)$

$^*OHLCV(t) = [PO(t), PH(t), PL(t), PC(t), Volume(t)]$

Table 2.5: Close Price Based Prediction

Features					Outputs
t_1	t_2	$\cdots$	t_{h-1}	t_h	
$PC_{1,1}$	$PC_{1,1}$	$\cdots$	$PC_{1,h-1}$	$PC_{1,h}$	y_1
$PC_{2,1}$	$PC_{2,1}$	$\cdots$	$PC_{2,h-1}$	$PC_{2,h}$	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$PC_{m-1,1}$	$PC_{m-1,1}$	$\cdots$	$PC_{m-1,h-1}$	$PC_{m-1,h}$	y_{m-1}
$PC_{m,1}$	$PC_{m,1}$	$\cdots$	$PC_{m,h-1}$	$PC_{m,h}$	y_m

Technical indicators used as features are mainly calculated by the past high/low/close/volume series, so we can compute the signatures for the 5-dimensional multivariate time series of length h. There are two ways to calculate the signature of high-dimensional data, one is to take signature transformation directly on the multivariate streams, the second is to work out the signature for individual one-dimensional data stream.

3. ***Technical indicators signature based prediction***

   Based on the Table 2.4, we can specify the technical indicators as

$$\begin{aligned}\{TI(t)_k\}_{k=1}^{f} = \{&\{SMA_i(t)\}_{i\in\mathbb{I}},\ \{EMA_j(t)\}_{j\in\mathbb{J}},\ \{MACD_j(t)\}_{j\in\mathbb{J}}, \{RSI_j(t)\}_{j\in\mathbb{J}},\ \{SO_j(t)\}_{j\in\mathbb{J}},\\ &\{WI_j(t)\}_{j\in\mathbb{J}}, \{BB_{Hj}(t)_{j\in\mathbb{J}}\},\ \{BB_{Mj}(t)\}_{j\in\mathbb{J}},\ \{BB_{Lj}(t)\}_{j\in\mathbb{J}},\ \{ATR_j(t)\}_{j\in\mathbb{J}},\\ &\{EoM_j(t)\}_{j\in\mathbb{J}},\ \{FI_j(t)\}_{j\in\mathbb{J}}\ \},\end{aligned}$$

$$\text{where}\quad \mathbb{I} = \{2,\ 4,\ 6,\ 8,\ 10,\ 12,\ 14,\ 16,\ 18,\ 20\},$$
$$\mathbb{J} = \{10,\ 12,\ 14,\ 16,\ 18,\ 20,\ 22,\ 24,\ 26,\ 28\}.$$

   Finally, we get the 120 features for technical indicators, and each technical indicator is computed with respect to 10 different time parameters. Thus, we calculate the signatures of each technical indicator stream, then create signature-based technical indicator features. There are also two approaches to compute the technical indicators signature based features, one is to take $\{SMA_i(t)\}_{i\in\mathbb{I}}$ as a stream of data of length 10 and compute its signature, while another one is to select the relative important technical indicators with time parameters, and calculate the signature on its own historical data stream of a length h.

4. ***Technical indicator + time series signature based***

   We select the meaningful technical indicators from the technical indicators based prediction model, then combine the time series signature based features, to see whether the performance of model improves after adding the time series signature-based features.

2.1.3 Evaluation

2.1.3.1 Evaluation Scores

In binary classification, there are four types prediction results based on their actual labels predicted labels as true positive, false positive, true negative and false negative, which as shown in Table 2.6. In order to see whether a classification prediction model solves our goal well, it is not enough to evaluate the performance only based on model accuracy. Thus, we take four standard measures, Accuracy, Recall, Precision, F_1 Score to evaluate the performance of predictive classification

Table 2.6: Confusion matrix of binary classification

		Predicted	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

models, and these statistics scores is calculated as below:

$$\begin{aligned}
\text{Accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TN+FP+FN+TP}}, \\
\text{Recall} &= \frac{\text{TP}}{\text{TP+FN}}, \\
\text{Precision} &= \frac{\text{TP}}{\text{TP+FP}}, \\
F_1 &= \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall+Precision}}.
\end{aligned}$$

Recall is calculated by dividing the number true positive with the total number of true positive and false negative, which measures the completeness of the classification model. Precision is a number of true positive divided by the true positive and false positive, it measures the exactness of the model. F_1 balances the recall and precision, and we expect a high accuracy, recall, precision and F_1 score.

2.1.3.2 Cross-Validation Method

Since the overlapping window we used in time series forecasting is not independent with each other, so the traditional k-fold cross-validation method is not an appropriate evaluation in this case. Thus, we use the 'increasing-window' technique to assess the performance of model, and this technique is as shown in figure. This technique makes sure that the each validation fold will not overlap with each other and hence obtain a reliable result. Moreover, considering the recency problem caused in the time series data proposed by Yao (1995), we set a recency margin between train & validation set and the test set [15]. In this case we set a 5-fold increasing window cross-validation.

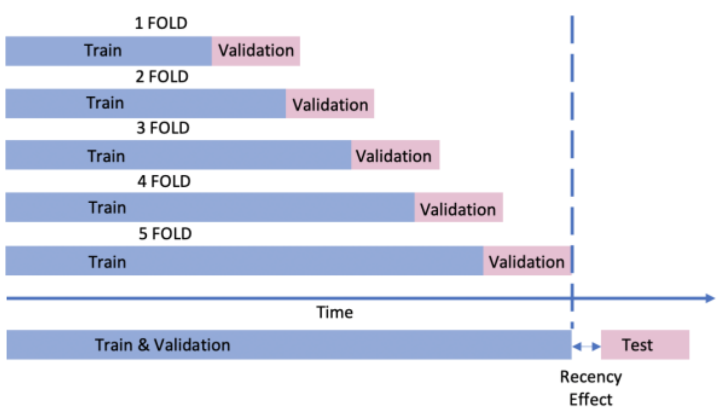


Figure 2.2: Increasing Window Cross Validation

2.1.4 Classifier Algorithms

We are dealing with the supervised learning problem, which means that we use the training data to predict the target variable y_i. We take random forest, XGBoost, and LSTM to build the predictive classification models. According to Liaw and Wiener (2002), Random forest and XGBoost are both constructed from the decision tree algorithms, and both belongs to the "ensemble learning" method, which generate lots of classifiers and combine their results together in the end [16]. Moreover, these two algorithms differ in tree construction and results combinations. In supervised learning, we need to optimise the objective function such that finding the best parameters θ to fit the training data. The objective function consists of the training loss $L(\theta)$, which decides the predictive power in the training set, and regularization term $\Omega(\theta)$ controlling the overfitting:

$$\text{objective}(\theta) = L(\theta) + \Omega(\theta) = \sum_{i}^{N} l(\hat{y}_i, y_i) + \sum_{k}^{N} \Omega(f_k) \tag{2.1.4}$$

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F} \tag{2.1.5}$$

where $\hat{y}_i$ is the predicted outcome, K is the number of trees, N is the sample size, and f is a function in the set $\mathcal{F}$, which is the set containing all possible classification and regression trees. Random forest uses *bagging* method while XGBoost takes *boosting* method to optimised this objective function.

2.1.4.1 Random Forest Classifier

The random forest algorithms is constructed from the decision tree ensembles, which consists of the a set of classification and regressions (CART). The core idea of random forest is a bootstrapping algorithm which bootstraps sample at each iteration to build a CART model, and repeats this multiple times, then finally combines all trees together to make a final prediction. Because of the bootstrapping, then the successive tree is independent of the previous ones. The algorithm is taking bootstrap procedures in each iteration as shown below:

1. Draw n_{tree} bootstrap samples from the data.
2. For each samples, generate an unpruned classification tree by randomly choosing m_{try} of the predictors, and select the best split among these chosen variables.
3. Aggregate the n_{tree} tree grown in the the bootstrap sample and take the majority votes to predict the 'out-of-bag' data.

Random forest mainly has only two parameters, the number of variables in each node, and the number of trees in the forest, also it is not very sensitive for their values. Moreover, the random forest provides the information of variable importance, which helps us to reduce the number of predictors, hence obtain a simpler and more readily interpret able model.

2.1.4.2 XGBoost Classifier

eXtreame Gradient Boosting (XGBoost) is a gradient boosted decision trees method and dominated in machine learning field because of execution speed and model performance. Instead of training models independently as random forest does, XGBoost trains model in a sequential order, and it takes additive training method to optimised the objective function at each step. The additive strategy adds a new weaker classifier to correct the errors of the previous one, and then update the objective function. Let the predicted outcome at step t in XGBoost learning process denoted as $\hat{y}_{i,t}$, then the predicted outcome $\hat{y}_{i,t+1}$ at next step $t+1$ is updated based on the earlier one as

$$\hat{y}_{i,t+1} = \sum_{k=1}^{t+1} f_k(x_i) = \hat{y}_{i,t} + f_{t+1}(x_i). \tag{2.1.6}$$

Hence we update the objective function at step t, and optimise it :

$$\text{objective}(\theta)_t = \sum_{i}^{N} l(\hat{y}_{i,t}, y_{i,t}) + \sum_{i}^{t} \Omega(f_i) = \sum_{i}^{N} l(\hat{y}_{i,t-1} + f_t(x_i), y_{i,t}) + \sum_{i}^{t} \Omega(f_i). \tag{2.1.7}$$

In each iteration, outcomes are weighted based on the prediction power, and in the end to take a weighted average to generate a final outcome. Overall, XGBoost just the push the extreme of the computation limits of machine to obtain a state-of-the-art result based on the additive training strategy proposed by Chen (2016) [17].

2.1.4.3 LSTM Classifier

The LSTM networks are the particular type of recurrent neural network (RNN) which is as shown in Figure 2.3, this network architecture illustrates a direct connection between nodes along to the temporal sequence, which allows it to describe the sequential input $\{X_0, \cdots, X_t\}$, but it is only able to memorise the short sequences. However, the LSTM networks we choose here have proved to distinguish the recent and early information successfully because of introducing memory cell to preserve the time related information and hidden dependencies in the data, therefore it is widely used in Natural Language Processing and time series analysis.

The architecture of LSTM is shown in Figure 2.4, and the memory cell is the fundamental unit of LSTM as shown in Figure 2.4(a). A memory cell includes three data ports, input gate i_t, forget gate f_t, and output gate o_t. These gates control the information interactions among input data and neighbor neurons. Based on the Figure 2.4(b), at time t, suppose we receive the input x_t and the output from neighbours h_{t-1}, and historical information C_{t-1} from time 0 to $t-1$, and the cell generates output h_t and adds the information to update historical information C_t at time t.

- The forget gate $f_t \in [0, 1]$ decides whether to remember or forget the previous information, which is based on the new information x_t and historical information h_{t-1} as shown in 2.1.12, and is wrapped in the sigmoid function.
- ΔC represents how much information will be added to C_t after introducing the new information input x_t, which is wrapped by the tanh function of the historical information h_{t-1} and x_t as shown in 2.1.9.
- The input gate i_t controls the portion information of previous output h_{t-1} and input x_t at time t as shown in 2.1.10, the input gate wrapped in the sigmoid function, hence the value of i_t is in the range [0, 1].
- The state vector C_t containing information from time 0 to t is determined by two part as shown in 2.1.11. One is the previous state C_{t-1}, which is controlled by the forget gate f_t; if $f_t = 0$, then forget all previous information from 0 to t. Another is ΔC controlled by the input gate i_t, meaning how much new information is added in terms of the value of the input gate.
- The output gate o_t is determining by h_{t-1} and x_t as shown in 2.1.12, and it decides how much information from C_t will be used to generate an output h_t of this memory cell as shown in 2.1.13.

$$
\begin{aligned}
f_t &= \sigma(W_f * [h_{t-1}, x_t] + b_f), && (2.1.8)\\
\Delta C_t &= \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right), && (2.1.9)\\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), && (2.1.10)\\
C_t &= f_t * C_{t-1} + i_t * \Delta C_t, && (2.1.11)\\
o_t &= \sum (W_o \cdot [h_{t-1}, x_t] + b_o), && (2.1.12)\\
h_t &= o_t * \tanh\left(C_t\right). && (2.1.13)
\end{aligned}
$$

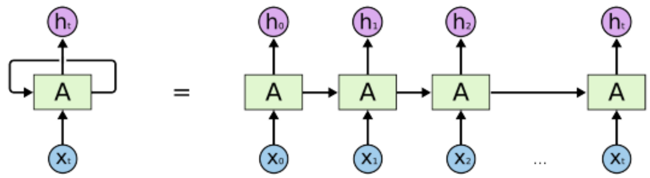
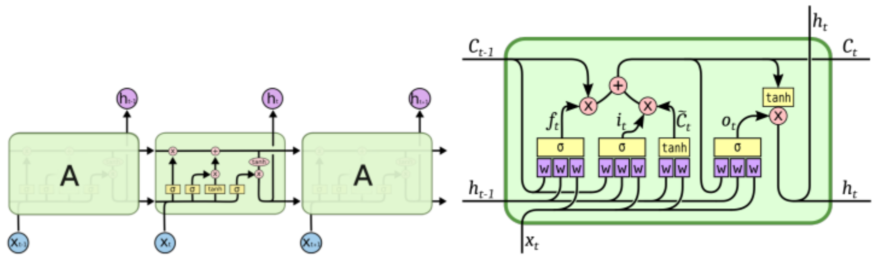


Figure 2.3: RNN



(a) LSTM cell in RNN (b) LSTM memory cell

Figure 2.4: LSTM

2.2 Empirical Investigation

This section constructs various predictive classification models based on different features, and then we choose the optimal features by comparing their performances. In the feature selection process, we construct the models by using random forest and XGBoost, and then use the LSTM networks to build the model only for the optimal features.

2.2.1 Data

2.2.1.1 Data Description

The value of currency currency depends on 'free float' or 'fixed float', free-floating currency is determined by the supply-demand force such as U.S. dollar, Japanese yen, and British pound, while the value of fixed-floating currency is set by the government. The data set we select to examine signature's ability of information extraction is 20 free-floating foreign exchange pairs 1-minute data from 2019 to 2020. We first take the 'audcad' foreign exchange pair to establish predictive classification models given various features and different model parameters. Then, we determine the optimal features by comparing their performances, and then evaluate its generality by applying it to build models for 20 currency pairs.

In features selections process, we split the 'audcad' pair in 2019 into a test set (20%) and a training set (80%), then the resulting training sample size is 284,511 and the test sample size is 70,128.

2.2.1.2 Data Prepossessing

In most machine learning methods, the weights of the model are initialised in the small random sample then updated by optimisation algorithms and estimates error. According to Bishop (1995), it is nearly advantageous to scale the input data and obtain a fast and stable learning process [18]. However, data scaling would be more complex since there is no best way to preprocess input data. There are generally two approaches to preprocess the data, standardisation and normalisation, and standardisation assumes data follows Gaussian normal distribution. Since the financial data is heavy tail and not normally distributed, in order to get reliable results, the Min-Max normalisation is used on data, which scales the raw data into the range of [0, 1].

2.2.2 Machine Learning Establishment

Supervised learning algorithms, including gradient boosting, random forest and neural network we discussed, need to set hyperparameters before running them. We first intuitively initialise hyperparameters for random forest and XGBoost as shown in Table 2.7, 2.8 respectively. After selecting the optimal features and model parameters, we then take the tuning process to find best hyperparameters to improve the performance. Moreover, we tune the hyperparameters for LSTM models after determining optimal features. The tuning process is taken in section 2.3.2.

- **Random Forest:** The hyperparameters of random forest can be categorised as three parts by their functions, and list the essential hyperparameters we considered [19] :
  - **the structure of the individual tree**: 'min_leaf_ leaf' specifies the minimum number of samples in the terminal node, and a lower value leads to generate tress with a large depth of the forest. It would be better to set a higher value in the large dataset from the perspective of computational cost.
  - **the structure of the forest:** 'n_estimators' (the number of trees in the forest), this generally sets a sufficiently high number of trees in the random forest to ensure having more different predictions than obtaining optimal performance.
  - **the randomness of the sampling:** 'max_features' (the number of drawn candidate features in each split) are the central hyperparameters for the random forest. The lower value of m_{try} variables result in generate in different and less correlated trees and hence yield better performance in prediction. At the same time, it could also lead to building a tree based on suboptimal variables and unstable models for the general case. Generally, a large value of m_{try} for high dimensional predictors or many relevant predictors, and a low value for a small number of variables or many irrelevant predictors. Thus in our case, we initially choose a low value of 'n_estimators' because of large dataset we have.
- **XGBoost:** XGBoost provides a large number parameters and could be divide into three distinct categories, **general parameters** (booster, num_features), **booster parameters** (learning_rate,gamma, max_depth,etc), **learning task parameters** (objective, eval_metric, seed). We choose a 'gbtree' booster method, and ['error','logloss'] for binary classification, where error is a binary classification error rate, and logloss is negative log-likelihood.

Table 2.7: Random Forest Hyperparameters (Initial)

Parameters		Parameters	
bootstrap	True	min_samples_leaf	1
criterion	gini	min_samples_split	2
max_depth	4	min_weight_fraction_leaf	0.0
max_features	auto	n_estimators	100
max_leaf_nodes	None	n jobs.	None
max_samples	None	oob score	False
min impurity decrease	0.0	random state	True
min impurity split	1	verbose	0
warm start	False		

After establishing random forest and XGBoost, we take them to build the predictive classification models based on various features.

Table 2.8: XGBoost Hyperparameters (Initial Setting)

Parameter	
booster	'gbtree'
max_depth	4
min_child_weight	None
learning_rate	0.1
n_estimators	100
subsample	0.5
colsample_bytree	None
eval_metric	['error', 'logloss']
gamma	None
seed	1000

2.2.3 Technical Indicator Based Prediction

We take 12 technical indicators with respect to time parameters and current OHLCV (open, high, low, close, volume) price as features shown in Table 2.4 to set up a predictive classification model. In order to evaluate the predictive power in future trend of different lengths, we set the forecast window range from 1 minute to 21000 minutes (around 3 weeks). The results of out-of-sample accuracy and cross-validation accuracy of technical indicators based prediction for forecast windows of different lengths are shown in Table 2.9.

We can see that the 2/3-week (16000/21000 minutes) ahead prediction has a relatively higher cross-validation accuracy (RF:75.64%, Xgboost:72.70%), but with a higher standard deviation (RF:15.22%, Xgboost:16.71%). In addition, the out-of-sample accuracy first decreases when the length of the forecast window range from 1 minute to 1 hour, then start to increase from approximately 6 hours (360 minutes). Moreover, the standard deviation of cross-validation accuracy (RF: 52.66 %, Xgboost: 52.97%) of 1-minute ahead prediction is relatively low (RF: 0.75 %, Xgboost: 0.72%), afterwards it increases as the prediction trend is longer. Although the cross-validation accuracy performs well under a longer forecast trend such as 2/3 weeks (RF: 75.64%/77.02%, Xgboost: 72.70%/71.96%), the resulting standard deviation is comparatively higher (Xgboost: 16.71%/16.55%). Moreover, to better examine the performance of the classification model, the confusion matrices of the 1/5/1000/16000-minute ahead predictive classification model and F_1 scores are shown in Table 2.10 and Table 2.11. The short-term (1-30 minutes) trend and long-term (16000/21000 minutes) trend prediction both have a relatively high F_1 scores, which indicates a satisfactory precision and recall. To summary, the results of the predictive power of different length forecast window suggest that:

- Higher prediction accuracy for a long forecast window but a high volatility of accuracy indicated by a high standard deviation of cross-validation accuracy
- The accuracy of the short window is relatively low, but the standard deviation of the cross-validation accuracy is low, hence its performance is more stable, and it is safer to use in prediction and guide investors to determine trading signals

In the following sections, we evaluate if the signature-based prediction could improve the cross-validation accuracy, lower its standard deviation, and obtain a stable predictive model. Due to the characteristics of the predictive ability of different length trends, for example, high accuracy is accompanied by high standard deviation and low accuracy is accompanied by low standard deviation, we aim to examine the predictive power of short-term trend by taking signature-based prediction. The 5-minute ahead prediction has a relative higher accuracy and a lower standard deviation in cross-validation, therefore, in the following sections, we build predictive classification models to predict the future 5-minute trend movement.

Moreover, we are interested in determining important features of technical indicators based prediction. Then we compute the important features for 5-minute ahead trend prediction in XGBoost model, which are as shown in figure 2.5. In the 5-minute step ahead prediction, the Stochastic Oscillator (SO), Relative Strength Index (RSI), Exponential Moving Average (EMA), Simple Moving Average (SMA) and current trading volume are among the top 10 important characteristics.

Table 2.9: Technical Indicators Based Prediction

Length	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
1	53.24	53.19	52.66	52.97	0.75	0.72
5	52.45	52.43	52.08	52.35	1.25	0.99
15	52.66	52.61	52.25	51.82	1.26	1.15
30	52.48	52.03	52.53	52.44	2.07	1.38
60	50.95	49.35	52.63	53.07	2.54	2.09
360	51.23	53.05	56.17	52.74	6.92	4.67
600	55.00	55.59	56.03	53.51	11.23	7.48
800	56.63	60.05	57.12	50.25	14.39	11.93
1000	57.72	61.41	58.92	53.39	16.22	13.13
3000	62.47	62.57	41.33	50.93	23.82	17.03
5000	65.82	65.58	45.41	50.93	32.67	17.03
8000	63.56	61.21	41.80	61.46	39.54	24.33
16000	68.57	67.12	75.64	72.70	15.22	16.71
21000	53.34	55.83	77.02	71.96	19.22	16.55

Table 2.10: Confusion Matrix of 1/5/1000/16000-minute ahead Technical Indicator based Prediction (XGBoost)

1-minute ahead		Predicted Trend	
Actual	Down	8405	25259
Trend	Up	7714	29059
Directional Accuracy		53.19%	

5-minute ahead		Predicted Trend	
		Down	Up
Actual	Down	16203	17544
Trend	Up	15391	20104
Directional Accuracy		52.43%	

1000-minute ahead		Predicted Trend	
		Down	Up
Actual	Down	30363	7889
Trend	Up	19171	12705
Directional Accuracy		61.41%	

16000-minute ahead		Predicted Trend	
		Down	Up
Actual	Down	27794	6640
Trend	Up	16670	20824
Directional Accuracy		67.12%	

Table 2.11: F1 Scores in Technical Based Prediction

		1	5	15	30	60	360	600	800	1000	3000	5000	8000	16000	21000
$F_1(\%)$	RF	65.17	57.56	57.28	56.77	51.19	19.48	22.72	23.92	25.01	33.84	40.44	49.56	68.15	55.09
	XGB	63.82	56.53	56.85	53.91	18.40	57.32	27.18	53.50	46.61	38.55	40.25	48.75	68.89	59.01

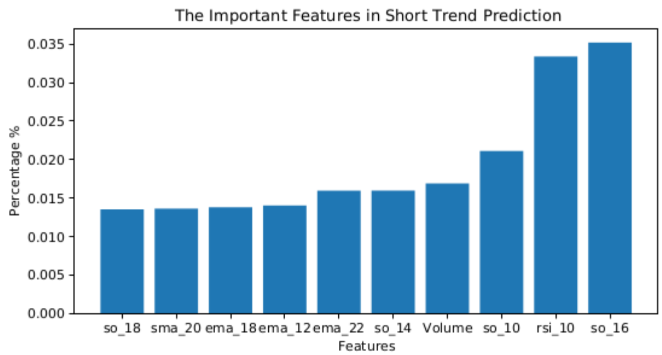


Figure 2.5: The Important Features in Technical Indicator Based Prediction for 5-minute Future trend

2.2.4 Time Series Based Prediction

2.2.4.1 Close Price Based Prediction

We directly use the close price of different lengths as input data in the close price based prediction. According to Let (2005), a large number of features contains much complicated information, which is hard to discern for machine learning algorithms [20], so we set the historical window of lengths ranging from 5 minutes to 360 minutes. The accuracy of this prediction model is as shown as Table 2.12. Compared with the technical indicators based prediction, the accuracy is around one percent lower whatever the lengths of historical window using RF and XGB, moreover the performance decreases slightly as the longer historical data we choose. The results show that technical indicator based prediction performs better than close price based prediction, which is reasonable, since the technical indicators consists of more information. Moreover, Table 2.13 shows that the close price based prediction models constructed by random forest and XGBoost method both have a lower prediction accuracy for future down trend indicated by lower recall scores.

Table 2.12: Close Price Based Prediction

Length	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
5	51.52	51.35	51.01	50.68	0.39	0.58
15	51.27	51.37	50.97	50.68	0.39	0.58
30	51.24	51.39	51.11	50.73	0.40	0.73
60	51.29	51.28	51.06	50.68	0.44	1.02
360	51.29	51.35	50.92	50.98	0.45	0.44
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

2.2.4.2 Multivariate Time Series Based Prediction

This section takes open, high, low, close, volume data streams as features and sets the historical range from 5 minutes to 60 minutes. Table 2.14 shows that the accuracy of multivariate time series

Table 2.13: Precision/ Recall /F_1 Scores in Close Price Based Prediction

	15-minute length (RF)			15-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.48	0.02	0.04	0.51	0.01	0.02
Up	0.51	0.98	0.67	0.51	0.99	0.68

based prediction is much lower than the technical indicated based prediction. Besides, the technical indicators are generated by these five time-series data streams. Hence the results illustrate that, the technical indicators extract the hidden valuable information of historical financial data, which could be used for future trend prediction. The precision/ recall / F1 scores show that the technical analysis based prediction have a better predictive ability in the rising trend than the downward trend.

Table 2.14: Multivariate Time Series Based Prediction

Lengths	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
5	51.30	51.20	51.00	50.50	0.43	0.82
15	51.31	51.21	50.86	50.39	0.66	0.54
30	51.32	51.22	50.83	50.09	0.67	0.72
60	51.34	51.30	51.05	49.87	0.41	0.71
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Table 2.15: Precision/ Recall /F_1 Scores in Multivariate Time Series Based Prediction

	15-minute length (RF)			15-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.45	0.00	0.01	0.47	0.02	0.05
Up	0.51	1.00	0.68	0.51	0.97	0.67

2.2.5 Signature Based Prediction

We aim to see if signature-based prediction extracts valuable information from historical data and obtain an interesting performance. For the signature-based prediction, there are lots of model parameters to determine, as shown in Table 2.16, where we have different model parameters that illustrate various combinations to calculate the signature of the data stream. Moreover, the relatively important model parameter is the length of the historical window since it depends on how much data to be extracted, which directly impacts the quantity and quality of features extraction in this kind of feature-based classification. Thus, first, we try different lengths of historical windows in model construction and choose an appropriate historical window as input data or compute the signatures.

2.2.5.1 Close Price Signature Based Prediction

In signature based close price prediction, we first fix other model parameters in signature transformation, and find an appropriate length of the historical data stream for information extraction. Thus we set the length of historical data stream range from 5 minutes to 1000 minutes, and the results in Table 2.17 show that a longer close price historical window, a lower accuracy we obtained. Table 2.18 also presents a overall performance of the close price signature based prediction in terms of on the Precision/ Recall/ F_1 scores.

- ***Close price signature based vs Technical analysis based:***

Table 2.16: Model Parameters

Parameters	
Historical Window Length	[5, 15, 30, 60, 360, 600, 1000]
Truncation Level	[1, 2, 3, 4, 5, 6, 7, 8]
Scaling Factor	[0.001, 0.01, 0.1, 1, 10, 100, 1000]
Transformation	[Signature, Logsignature]
Augmentation	[AddTime + Basepoint, AddTime, Lead-Lag, Lead-Lag + AddTime, AddTime+Lead-Lag]

To be specific, the cross-validation accuracy drops from 52.95% (XGBoost) to 50.74% (XGBoost) as the historical window increase from 15 minutes to 1000 minutes. Moreover, the 15-minute length closing price signature-based prediction performs a bit better than technical indicators based prediction in 5-minute trend forecast, which has a relatively higher cross-validation accuracy (XGBoost: 52.95% vs 52.06%) and a lower standard deviation (XGBoost: 0.85% vs 0.87%).

- ***Close price signature based vs Close price based:***

  Compared with the short length historical window of close price based prediction in section 2.2.4.1, the same length (15 mins) historical window of close price signature based prediction has approximately two percent higher cross-validation prediction accuracy (RF: 50.97% vs 51.90% , XGB: 50.68% vs 52.95 %). At the same time, the close price signature based prediction model have a better predictive power both for the rising and downward trend indicated by a higher recall and precision scores in Table 2.18

To summarise, the close price signature-based prediction performs better than the close price-based prediction and has a similar performance as the technical indicators-based prediction, further proving the ability of signature transformation in information extraction on the historical data stream.

Table 2.17: Close Price Signature Based Prediction

Lengths	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
5	52.59	52.55	52.05	51.92	0.94	0.95
15	**52.53**	**52.59**	**51.90**	**52.95**	**0.94**	**0.85**
30	52.51	52.24	51.79	52.06	1.21	0.96
60	52.24	52.35	51.85	51.70	1.27	1.34
360	51.36	51.63	50.80	51.43	0.60	0.65
1000	51.01	50.49	50.75	50.74	0.91	0.80
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Parameters		Parameters	
Truncation Level	4	Augmentation	AddTime
Transformation	Signature	Scale of Time Series	1
Forecast Window	5		

2.2.5.2 'Signature Return' Based Prediction

There is another to compute the close price signature-based prediction by considering the close return rather than the close price itself. Now we evaluate which approach provides a better extraction ability in financial data. Inspired by Gyurkó and Lyons's work (2020) [21], we preprocess the financial time series before signature transformation by the following step:

Table 2.18: Precision/ Recall / F_1 Scores in Close Price Signature Based Prediction

	5-minute length (RF)			5-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.52	0.39	0.44	0.52	0.40	0.45
Up	0.53	0.66	0.59	0.53	0.65	0.58

	15-minute length (RF)			15-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.51	0.41	0.46	0.52	0.42	0.46
Up	0.53	0.63	0.58	0.53	0.63	0.58

	1000-minute length (RF)			1000-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.49	0.12	0.19	0.51	0.25	0.34
Up	0.51	0.88	0.65	0.52	0.76	0.62

- Compute log-returns for close price with a time interval Δ_t, which is chosen based on the prediction scenario:
$$r(t, \Delta_t) := \log PC(t + \Delta_t) - \log PC(t), \tag{2.2.1}$$
where $r(t, \Delta_t)$ represents the log-return of close price on $(t + \Delta_t, t)$.

- Calculate the signatures for the generated log-returns data stream.

Results are as shown in Table 2.19. We also establish the models in different lengths of close price historical window from 5 minutes to 10 minutes. Observing that whatever the length of the historical window, the cross-validation accuracy is one per cent lower than technical indicated based and close price signature-based prediction (15 mins, XGB: 51.14% vs 52.35% vs 52.95%), as well as has the lower predictive power for the downtrend compared with the close price signature based prediction. In conclusion, the log-return signature-based prediction does not perform as well as the close price signature-based prediction. Hence this approach of preprocessing financial data is not suitable in this scenario.

2.2.5.3 Multivariate Time Series Signature Based Prediction

We now work out signature extraction of Open/High/Low/Close/Volume time series, and high dimensionality comes with the complexity and computational cost in signature transform. Thus, it is imperative to preprocess data streams properly to ensure that the signature transform extracts information efficiently for our target goal. Consider a d-dimensional time series $X_t = \{X_{ti}\}_{i=1}^n$ of length n in each sliding window, there are three approaches we tried as following:

- **Approach 1:** Calculate the signatures of each time series separately, and then combine them into features.

- **Approach 2:** Calculate the signature for a 5-dimensional time series.

- **Approach 3:** Dimensionality reduction of data. From the technical indicators based predictions, we discover four data streams play important role in prediction, high/low/close/volume data streams. Inspired by Gyurko and Lyons's work in 2013 [5], we do the next following steps for these three data streams:
  - AddTime Augmentation:
  $$u_{t_i} := (t_i - t_0)/(t_n - t_0), \tag{2.2.2}$$
  - the normalised close price: $PC(t_i), i = 1, .., n$
  - the standardised spread at each time :
  $$s(t_i) := PH(t_i) - PL(t_i), \tag{2.2.3}$$

Table 2.19: Log-returns Signature Based Prediction

Lengths	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
5	51.65	51.81	51.80	51.21	0.74	0.59
15	51.64	51.70	51.13	51.14	1.05	1.14
30	51.94	51.74	51.53	51.39	0.95	0.80
60	52.23	52.18	51.42	51.48	0.93	0.83
360	52.12	52.16	51.57	51.70	0.98	1.36
1000	52.18	51.60	51.61	51.43	0.99	0.46
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Parameters		Parameters	
Truncation Level	4	Augmentation	AddTime
Transformation	Signature	Scale of Time Series	1
Forecast Window	5		

Precision/ Recall /F_1 Scores in Close Price Signature Based Prediction

	15-minute length (RF)			15-minute length (XGB)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Down	0.50	0.32	0.39	0.51	0.31	0.40
Up	0.52	0.70	0.60	0.52	0.69	0.60

– the normalised cumulative volume:

$$v_{t_i} := \text{Volume}(t_i)/\sum_i \text{Volume}(t_i). \tag{2.2.4}$$

Thus we use above components to compose a time-series denoted as $\hat{X} = (u_{ti}, p_{ti}, s_{ti}, v_{t_i})_{i=0}^{n}$. For the augmentation, we choose to do the lead transformation of $\hat{X}$ and then pair it with the lag-transformation of log close price p_t, aiming to capture the quadratic variation pattern of price, then the resulting input streams are of the form as

$$(\hat{Z}_{s_i})_{i=0}^{2n} := ((u_{s_i}^{\text{lead}}, p_{s_i}^{\text{lead}}, s_{s_i}^{\text{lead}}, v_{s_i}^{\text{lead}}, p_{s_i}^{\text{lag}}))_{i=0}^{2n}. \tag{2.2.5}$$

Subsequently we compute the signature for each time series up to the truncation level of 4, and take these signature together as feature vector in the multivariate time series prediction.

We follow the same step to select model parameters as in the signature-based close price prediction, and the results for three different signature computation approaches mentioned above are as shown in Table 2.20. We can see that the signature-based multivariate time series prediction does not have a better performance than the technical indicator based forecast at the given model parameters. In addition, the approach of signature computation slightly affects the performance of the model, especially the size of the feature vector. Specifically, if we compute the signature of each time series separately, there is much freedom in choosing model parameters, such as truncation level, augmentation, and the series length, in each signature transform. Moreover, the large size of the features in the model might cause overfitting problem, and impacts of open/high/low/-close/volume series on market trend are distinguished, so it would be better to work out data processing for each time series before signature transform.

Table 2.20: Multivariate Time Series Signature Based Prediction

Parameters Selection			
Parameters		Parameters	
Truncation Level	3/4	Augmentation	AddTime
Transformation	Signature	Scale of Time Series	1
Forecast Window	5		

Approach 1 (Truncation Level = 4)						
Length	Out-of-Sample(%)		CV (%)		Std (%)	
	Random Forest	XGboost	Random Forest	XGboost	Random Forest	XGboost
5	51.32	51.33	50.96	50.49	0.40	0.23
15	51.34	51.22	50.98	50.12	0.40	0.25
30	51.32	51.13	50.91	49.84	0.34	0.88
60	51.26	51.20	50.51	52.03	0.37	0.65
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Approach 2 (Truncation Level = 3)						
Length	Out-of-Sample(%)		CV (%)		Std (%)	
	Random Forest	XGboost	Random Forest	XGboost	Random Forest	XGboost
5	51.34	51.24	51.03	49.91	0.37	0.34
15	51.31	50.82	51.07	50.34	0.39	0.35
30	51.33	51.22	50.84	50.15	0.40	0.51
60	50.31	51.08	50.90	50.25	0.41	0.26
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Approach 3 (Truncation Level = 4)						
Length	Out-of-Sample(%)		CV (%)		Std (%)	
	Random Forest	XGboost	Random Forest	XGboost	Random Forest	XGboost
5	52.17	52.28	51.64	51.33	0.94	0.86
15	52.25	52.26	51.75	51.79	0.95	0.83
30	51.82	51.94	51.66	51.53	1.06	1.06
60	51.26	51.44	51.41	51.40	0.55	0.45
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

2.2.5.4 Technical Indicator Signature Based Prediction

There are two approaches to set up signature-based technical indicators prediction. The results of these two approaches are as shown in Table 2.21.

- **Approach 1:** Calculate the signature for each technical indicator stream ordered by its time parameters
  - Each technical indicator is computed with ten different time parameters, and we have 12-dimensional technical indicator data streams. Thus, for this approach, we have 12-dimensional time series with a fixed length of 10. We do not consider selecting the best historical lengths in this case rather than examining the predictive power of different truncation levels. We set the range of truncation levels as [2,3,4]. We do not choose a higher truncation level due to the computation cost for the high dimensional data stream. There also are difficulties for algorithms to learn models with the high complexity of features. Based on the experiences from the multivariate time series signature-based prediction, we decide to compute the signature for each data stream separately.
  - From the results in Table 2.21, we can see that the accuracy rises slightly as the truncation level increases, but do not perform better than technical indicator based prediction, which shows that the signature transformation in this case do not extract the valuable information but hide some valuable information from technical indicators.
- **Approach 2:** Select the meaningful technical indicators and time parameters, generate a time series of technical indicators at a chosen time parameter, then calculate the signature of the data stream
  - In the second approach, we first choose the meaningful technical indicators based on the important features of technical indicators based prediction. Figure 2.5 shows that stochastic oscillator (SO), relative strength index (RSI), exponential moving average (EMA), simple moving average (SMA) play important roles among all technical indicators features, and we choose to generate $\{SO_{16}, RSI_{10}, EMA_{22}, SMA_{20}\}$ 4-dimensional time series, and then generate the signature for each data stream of increasing historical lengths. In this approach, as the above steps in time series signature based prediction, we compare the predictive power of different lengths historical window ranging from 5 minutes to 360 minutes.
  - Table 2.21 shows that a one percent lower cross-validation accuracy obtained in this approach, and points out that this approach does not extract the valuable information from technical indicators.

To sum up, the technical indicators signature-based prediction results illustrate that the signature transform could not extract valuable information from the technical indicators. One guess for this is that the technical indicators at each time point t are computed from the raw historical data trying to extract the information of the data and use the statistics to represent the past information. At the same time, the signature transformation also tries to extract information for the historical data, the extraction ability might overlap between these two approaches. In other words, if we use the signature method on the technical indicators, some valuable information hidden in the raw historical data might lose, then is not extracted by the signature transform method.

2.2.5.5 Ensemble of Technical Indicators + Close Price Signature Based

Based on the finding so far, the performances of close price signature based prediction and technical indicators based prediction are very similar to each other, what we have to do next is to evaluate whether the information extracted by these two approaches (technical indicators and signature transform) is overlapped and whether there exists an improvement of performances when we combine these two feature extractions. Thus, we take the close price signature based, together with meaningful financial technical indicators, to build a new predictive classification model. We choose the top 10 important features as shown in Figure 2.5, and it is noticeable that the stochastic oscillator has a higher weight among 125 features and it is generated by high/low/close time series together as shown in 2.3. We expect this new model (technical indicators + close price signature

Table 2.21: Technical Indicator Signature Based Prediction

Parameters Selection			
Parameters		Parameters	
Truncation Level		Augmentation	AddTime
Transformation	Signature	Scale of Time Series	1
Forecast Window	5		

Approach 1

Truncation	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
2	51.34	51.20	51.04	51.77	0.35	0.33
3	51.33	51.28	51.04	50.60	0.35	0.18
4(300)	51.33	51.32	51.04	50.74	0.35	0.19
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

Approach 2 (Truncation Level = 4)

Lengths	Out-of-Sample(%)		CV (%)		Std (%)	
	RF	XGB	RF	XGB	RF	XGB
5	51.36	51.25	51.04	51.06	0.35	0.34
15	51.31	51.24	51.04	51.87	0.35	0.35
30	51.31	51.51	51.04	50.66	0.35	0.40
60	51.35	51.32	51.04	50.60	0.35	0.34
360	51.25	50.86	51.12	50.87	0.30	0.85
TI	**52.66**	**52.43**	**52.25**	**52.35**	**1.26**	**0.99**

based) outperforms those models which only consider technical indicators or close price signature based alone.

- **Features Selection:** We mainly compare three models in this section, which are
  - New model: Ensemble of 10 Technical Indicators + Close Price Signature Based
  - Benchmark: 10 Technical Indicators Based (B1) and Close Price Signature Based (B2)
- **Historical Window Length:** We compare the predictive power of historical windows of different lengths range from 5 minutes to 1000 minutes.
- **Results Analysis:** The Table 2.22 shows that the accuracy for different lengths of historical windows is similar to each other. However, the 15-minute length has a relative high out-of-sample accuracy (XGB: 52.74 %)and low standard deviation of cross-validation (0.81%). Moreover, the performance of 'Ensemble of Technical Indicators + Close Price Signature' based prediction has a similar predictive power with the other two benchmarks models. They both achieved around 52% out of sample and cross-validation accuracy. However, we see that the signature-based predictions generally has a lower standard deviation of the cross-validation accuracy (B2: 0.94%, 0.85 %, new:0.81%), which illustrates that a signature-based prediction would be a bit more stable.

  There is no significant improvement in the accuracy after the combination, indicating that the information extracting by the signature transformation and technical analysis is overlapped to a degree. However, the improvement on the standard deviation of the cross-validation accuracy suggests that the signature-based prediction would provide a more stable prediction, which might be because it extracts some information that technical indicators do not grasp.

Table 2.22: Ensemble of Technical Indicators + Close Price Signature Based

Parameters Selection			
Parameters		Parameters	
Truncation Level	4	Augmentation	AddTime
Transformation	Signature	Scale of Time Series	1
Forecast Window	5		

Length	Out-of-Sample (%)		CV(%)		Std(%)	
	RF	XGB	RF	XGB	RF	XGB
5	52.54	52.70	52.51	52.31	1.06	1.29
15	52.47	52.74	52.19	52.14	1.02	0.81
30	52.42	52.31	52.23	52.32	0.98	1.15
60	52.43	52.46	52.25	51.96	1.11	1.29
360	52.50	52.48	52.21	51.66	1.14	0.92
1000	52.50	52.32	52.21	52.50	1.14	1.49
Benchmark						
B1 (15 min)	**52.44**	**52.48**	**52.09**	**52.17**	**1.21**	**1.20**
B2 (15 min)	**52.53**	**52.59**	**51.90**	**52.95**	**0.94**	**0.85**

2.3 Parameters Selection

According to the above feature selection results, we choose the prediction model based on the 'technical indicators and closing price signature ' to forecast future 15-minute trend. Then, we take different model parameters, such as augmentation and truncation levels, in the chosen model to evaluate the impact of different model parameters and determine the appropriate ones.

2.3.1 Model Parameters Selection

In this case, we aim to assess the model's performance under different model parameters and choose appropriate model parameters, the model parameters we need to specify as shown in 2.16 except for the historical window length. For model parameters, generally, the more information extracted, the higher truncation level we set. Besides, we do not know which augmentation, scaling factor, transformation works well in this scenario. The selection process is taken as follow step:

1. Fix other model parameters, try different augmentations as shown in Table 2.23. 'A+B' means that we do 'A' augmentation first and then 'B' augmentation. Table 2.23 shows that 'AddTime' and 'LeadLag + AddTime' augmentations outperform other augmentations. Moreover, these two augmentations have the same result. Additionally, 'LeadLag' and 'AddTime+LeadLag' have the same performance as well. We can conclude that the 'LeadLag' and 'AddTime' combination's order affects the predictive power. We finally choose 'AddTime' augmentation in our model, now fix it to examine other model parameters

2. Fix the optimal augmentation selected by step 1, and try different scaling factor $\alpha \in \mathbb{R}$, where $\alpha = (0.001, 0.01, 0.1, 1, 10, 100, 1000)$. Table 2.24 shows that rescaling does not impact the ability of the signature to extract the information of the data stream a lot. Hence, we do not do any rescaling in our time series data.

3. We fix the rescaling factor as $\alpha = 1$ and 'AddTime augmentation', then we try different transformations, including log-signature and signature. Finally, found that signature performs a better result compared with log-signature in this scenario.

4. Finally, we examine the predictive power in increasing truncation levels. We expect a higher truncation level providing a better performance since it would extract more information. However, the high signature depth also leads to a large number of the signature terms to be considered, and it would be more complicated for the algorithm to learn the hidden relationship between variables. Based on the results of Table 2.25, the accuracy does not change two

much as the truncation level increases, the out of sample is generally around 52.50% and the cross-validation accuracy for all depth levels is above 52% with a 1% standard deviation. This means that there are no many differences in various truncation levels. Moreover, there exists a trade-off between the information extraction and the low-dimensional features. We finally select a medium value of the truncation level, 4, which could extract sufficient information and would not generate high-dimensional features.

Based on the above selections, the optimal model parameters we choose is as shown in Table 2.26.

Table 2.23: Results in Different Augmentations

Augmentation	Out-of-Sample (%)		CV(%)		Std(%)	
	RF	XGB	RF	XGB	RF	XGB
AddTime + Basepoint	52.43	52.51	52.18	52.12	1.04	1.01
AddTime	52.47	52.74	52.19	**52.14**	**1.02**	**0.81**
LeadLag	52.36	52.45	**52.25**	51.98	1.20	0.94
LeadLag + AddTime	52.47	52.74	52.19	**52.14**	**1.02**	**0.81**
AddTime + LeadLag	52.36	52.45	**52.25**	51.98	1.20	0.94

Table 2.24: Results in Different Rescaling Factors

Rescaling Factor	Out-of-Sample (%)		CV(%)		Std(%)	
	RF	XGB	RF	XGB	RF	XGB
0.001	52.42	**52.74**	**52.23**	52.13	1.12	0.94
0.01	52.39	**52.74**	**52.33**	51.96	1.10	0.83
0.1	52.36	52.69	52.24	**52.27**	1.09	0.99
1*	52.47	**52.74**	52.19	52.14	1.02	0.81
10	**52.53**	52.70	52.13	52.22	1.04	**0.79**
100	52.45	52.74	52.18	52.18	1.03	0.92
1000	52.43	52.72	52.14	52.14	**0.97**	0.89

Table 2.25: Results in Different Truncation Levels

Truncation Level	Out-of-Sample (%)		CV(%)		Std(%)	
	RF	XGB	RF	XGB	RF	XGB
1	52.40	52.47	52.22	52.32	1.55	1.03
2	52.47	52.44	**52.28**	**52.37**	1.12	1.13
3	52.42	52.59	52.20	52.34	1.52	0.98
4	52.47	52.74	52.19	52.14	1.02	0.81
5	52.43	52.57	52.22	52.21	1.03	0.63
6	**52.52**	52.54	52.11	52.17	**1.00**	1.01
7	52.44	**52.80**	52.13	52.28	1.09	**0.77**
8	52.46	52.77	52.13	52.13	1.11	0.80

2.3.2 Hyperparameters Tuning

- **Random Forest:** We now do hyperparameter tuning after determining the model parameters. For the random forest classifier, the initial hyperparameters used in the above sections are as shown in Table 2.7. Based on the discussion in the section 2.2.2, we set a relatively

Table 2.26: Optimal Model Parameters

Parameters	
Historical Window Length	15
Truncation Level	4
Scaling Factor	1
Transformation	Signature
Augmentation	AddTime

large node size (‘min_sample_leaf’), 5, to decrease the running time since we have a large sample dataset of 284,511 observations. Moreover, in the tuning process, we mainly consider the hyperparameters that control the structure of the forest and the randomness of the sampling, so we tune the number of trees (n_estimators), the number of features (max_features), and the depth of forest (max_depth). Thus we take the randomized search to tune the Random Forest as shown in Table 2.3.2, the optimal parameters we choose are bolded.

Table 2.27: Results of Hyperparameters Tuning for Random Forest

Parameters	
n_estimators	[50, **100**, 150, 200, 250, None]
max_features	[‘auto’, **‘sqrt’**]
max_depth	[2, 4, 6, **8**, 10, None]
Accuracy	
Out-of-Sample%	52.60 (52.52)
CV%	52.15 (52.11)
Std%	1.09 (1.00)

- **XGBoost:** Table 2.8 illustrates the hyperparameters we set before, and now we do the hyperparameters tuning by taking randomized search. The result of hyperparameters tuning for XGBoost is shown in Table 2.28, and found that the out-of-sample accuracy and cross-validation respectively increase around 0.30% and 0.20% after tuning.

Table 2.28: Results of Hyperparameters Tuning for XGBoost

Parameters	
learning_rate	[0.05, 0.1, **0.15**, 0.2, 0.3]
n_estimators	[50, **100**, 200, 300, 500]
max_depth	[2, 3, 4, 5, **6**]
sub_sample	[**0.5**, 0.6, 0.7, 0.8]
Accuracy	After (Before)
Out-of-Sample %	52.87 (52.54)
CV%	52.35 (52.17)
Std%	0.75 (1.01)

- **LSTM:** We now establish the LSTM prediction model for the optimal features and model parameters. There are five hyperparameters to be determined, and we also take the randomised search to determine the optimal one. The tuning results are as shown in Table 2.29:

Table 2.29: Results of Hyperparameters Tuning for LSTM

Parameters	
learning rate	[0.01, 0.05, **0.08**, 0.1, 0.5, 0.8]
layers	[**1**, 2, 3]
epochs	[5, **10**, 20, 30]
hidden dimension	[10, **20**, 30, 40, 50, 100]
batch size	[100, 250, **500**, 750, 1000]

2.4 Conclusions

The above features selections process is summarised as shown in Figure 2.6 and we built as the following five types:

1. Technical Analysis Based Prediction
2. Time Series Based Prediction
3. Time Series Based Signature Based Prediction
4. Technical Analysis Signature Based Prediction
5. Time Series Based Signature Based Ensemble of Technical Analysis Prediction

We built the technical analysis based prediction model first to evaluate the predictive power of the different length future trends. Then we found that the short-term prediction is much stable compared with the long-term predictions. Hence we chose a 5-minute ahead prediction because of its stability in model performance. Next, we tried various approaches to compute the time series and technical indicators signature-based predictions. By comparing these signature features based predictions with features based predictions (the same features used to compute signature), we found that signature extracts valuable information from the raw historical time series data, but does not grasp the valuable information from technical indicators time series. Moreover, we found that the close price signature-based prediction has a similar performance as the technical indicators-based prediction, indicating that the information extracted from close price only by signature methods has an equivalent ability in future trend prediction as the information extracted by technical indicators from high-dimensional times series (high/low/close/volume).

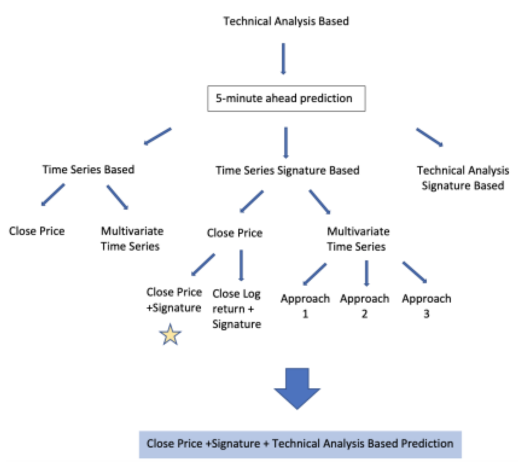


Figure 2.6: Features Selection Process

Chapter 3

Trading Strategies

This chapter uses the 'close price signature ensemble of technical indicators based prediction' to guide trading strategies. The model predicts market movement in the next $l = 5$ minutes by taking technical indicators and close price signature as features. Furthermore, we not only build a model to predict up and down trend in foreign rate market, but also build another specialised model to predict a flat trend, and implement trading strategy based on these two models together.

3.1 Framework Establishment

Specifically, the predictive binary classification model helps us implement the trading strategy by the following two approaches.

- **Approach 1:** The trading signals are determined based on up/down trend prediction model only, which is denoted as

$$\{signal_t(l)\}_{t=1}^{N} = \begin{cases} long, & \text{if} \quad \hat{y}_t(l)^{\text{up/down}} = 1, \\ short, & \text{if} \quad \hat{y}_t(l)^{\text{up/down}} = 0, \end{cases} \tag{3.1.1}$$

  where $\hat{y}_t(l)^{\text{up/down}}$ represents the l -minute up/down trend prediction of the model at time t, and we take a long position in the next l minutes if the model predicts the rising trend, take a short position in the next l minutes if the model predicts the dropping trend.

However, in reality, we have to consider the transaction cost in every trade, which depends on market volatility and the currency pair. Thus the price movement needs to be higher enough to make up for the transaction cost. In the currency market, the transaction cost consists of the commission fees and the bid-ask spread. Now, we estimate the transaction costs by considering the bid-ask spread only. If we let the bid-ask spread as S_t, then we label the output variables in the flat-trend prediction model as

$$\{y_t(l)\}_{t=1}^{N} = \begin{cases} 1, & \text{if} \quad |\delta_t(l)| < S_t, \\ 0, & \text{otherwise}, \end{cases} \tag{3.1.2}$$

where $y_t(l) = 1$ represents a non-change trend in the next l minute when the price movement is smaller than the spread. Then, we take the same features of predicting up/down trends to establish a predictive binary classification model for non-change trend forecasting. However, in this non-change trend classification predictive model, the non-change trend is the majority class in a 5-minute period and hence involves an imbalanced classification challenge, which means the number of samples in each class is not balanced. The imbalanced problem in the dataset could not be ignored since most machine learning algorithms assume a balanced class, and the bias in the dataset would result in a poor prediction in the minority class. The two techniques, oversampling and undersampling, provide a naive idea to rebalance our dataset by randomly resampling the examples of minority class and removing the majority class examples to generate a balanced distribution in the training dataset. We take the undersampling technique to balance the training set since we still have approximately 180,000 samples in the training dataset after undersampling, which is sufficient for the algorithm to learn a reliable model. Thus, we have the following approach to set up a trading strategy.

- **Approach 2:** Establish a specialised model to predict the flat trend, then set up trading signals according to the flat-trend and up/down trend predictions. The trading signals are determined as:

$$\{signal_t(l)\}_{t=1}^{N} = \begin{cases} long, & \text{if} \quad \hat{y}_t(l)^{\text{up/down}} = 1 \text{ and } \hat{y}_t(l)^{\text{non-change}} = 0, \\ short, & \text{if} \quad \hat{y}_t(l)^{\text{up/down}} = 0 \text{ and } \hat{y}_t(l)^{\text{non-change}} = 0, \\ no\ action, & \text{others}, \end{cases} \quad (3.1.3)$$

where $\hat{y}_t(l)^{\text{non-change}}$ represents the prediction of the l-minute non-change trend of the model, and we take the long/short position only when the up/down prediction model has a rising/-dropping trend prediction and non-change prediction model forecast a high volatility trend in the next l minutes.

3.2 Empirical Investigation

We use the Random Forest, XGBoost and LSTM in this implementation. We backtest the above trading strategies on a monthly basis, from January to December 2020, by building predictive classification model in the past twelve months to evaluate the robust performance of the trading strategy. The backtest framework is based on the following rules:

- We assume that the predictions of the model can be generated immediately, and there is no time lag when we execute trading positions.
- Each currency pair has its own pips that measures the change in value between the two currencies.
- We buy/sell one unit currency pairs at each execution.
- No transaction fees considered.
- Assume a fix bid-ask spread in the entire period for **Approach 2**, and the spread is estimated by 2 pips
- Assume that there are no short selling restrictions during the entire period. We start with an initial balance of 100 (base currency), short 10% of the balance to execute the position, and return it after exit the position (5 minutes).
- The initial balance of the next month is the balance of the previous month
- Suppose there is no interest rate in our backtesting framework
- We take the buy-and-hold strategy as a benchmark, to evaluate the profitability of the trading strategies

Moreover, the balance of trading strategy updated at time t as below:

$$\begin{aligned} B_t &= B_{t-1} + \text{Profit}_t, \\ \text{Profit}_t &= \text{Invest Fund} \times [(\frac{PC(t+l)}{PC(t)})^{I_t} - 1], \end{aligned} \quad (3.2.1)$$

where,

$$I_t = \begin{cases} 1 & \text{if } signal_t(l) = long, \\ -1 & \text{if } signal_t(l) = short, \\ 0 & \text{else }, \end{cases}$$

where B_t is the balance at time t and I_t is chosen based on the signals we obtain.

3.3 Results

- Monthly accuracy results for rising/downward trends by Random Forest, XGBoost, and LSTM are as shown in Figure 3.1, 3.2, 3.3. The 12-month average accuracy of each currency pair is shown in the Table 3.1. The prediction performances of three machine learning methods are similar. They all made good predictions on the "xauusd" pair, achieving a high average accuracy rate (RF: 64.64%, XGB: 64.81%, LSTM: 58.33%) per month. Other pairs generally have an average of 52% accuracy per month for these three methods. Moreover, we can see that the performance for Random Forest and XGBoost are quite similar to each other, it might because that they are both ensemble tree methods. Moreover, in most currency pairs LSTM network has the similar performance as the random forest and XGBoost given the hyper parameters we set, except for the xauusd pair, random forest and XGBoost outperforms than LSTM model(RF: 64.64% vs 64.81% vs 58.33%). One possible reason for this is that the hyperparameters setting for LSTM is more complicated for the random forest and XGBoost methods, and the hyperparameters we set might not be suitable for all currency pairs, and it is necessary to tune process in each currency pair individually.

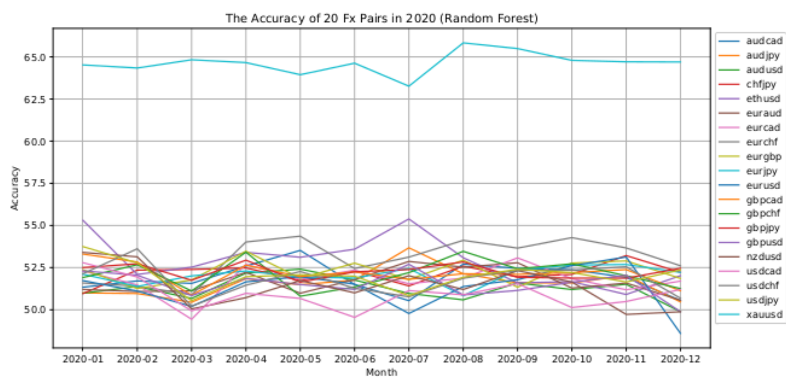


Figure 3.1: The Prediction Accuracy of 20 Fx Pairs from January to December in 2020 (Random Forest)

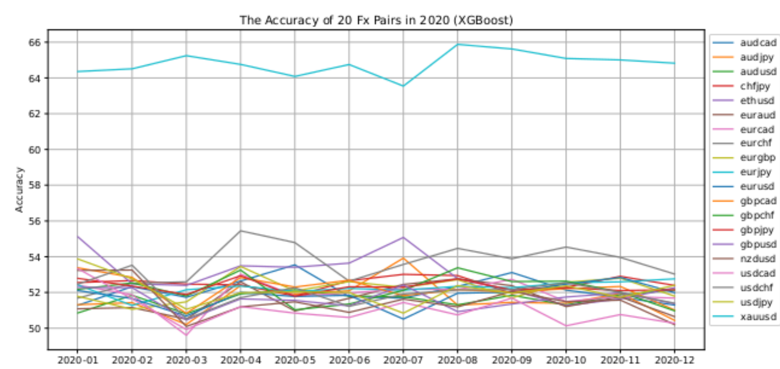


Figure 3.2: The Prediction Accuracy of 20 Fx Pairs from January to December in 2020 (XGBoost)

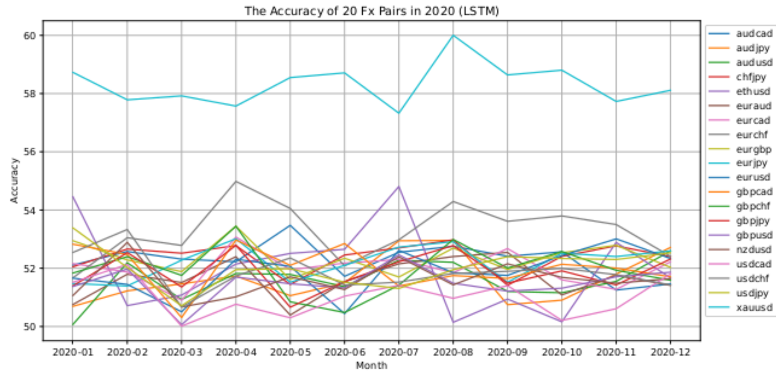


Figure 3.3: The Prediction Accuracy of 20 Fx Pairs from January to December in 2020 (LSTM)

Table 3.1: Average accuracy (%) for 12 months in 2020 (20 foreign exchange pairs)

Forex Pair	RF	XGB	LSTM
audcad-1m	51.90	52.13	52.28
audjpy-1m	51.58	51.65	51.64
audusd-1m	51.26	51.63	51.59
chfjpy-1m	52.14	52.50	52.38
ethusd-1m	52.89	52.97	52.01
euraud-1m	51.67	51.60	51.64
eurcad-1m	51.77	51.90	52.00
eurchf-1m	53.01	53.59	53.26
eurgbp-1m	52.26	52.19	52.48
eurjpy-1m	52.07	52.26	52.20
eurusd-1m	51.29	51.82	51.86
gbpcad-1m	52.13	52.20	52.01
gbpchf-1m	52.11	52.16	51.93
gbpjpy-1m	52.26	52.32	51.87
gbpusd-1m	51.50	51.60	51.49
nzdusd-1m	51.40	51.52	51.61
usdcad-1m	50.94	51.05	51.02
usdchf-1m	51.79	51.99	51.87
usdjpy-1m	52.04	52.12	52.01
xauusd-1m	64.64	64.81	58.33

- We need to evaluate its practical application in the financial market, so we implement the trading strategies by two mentioned approaches. The results of strategy simulations are as shown in Table 3.2 and Table 3.3, where we take Random Forest, XGBoost and LSTM to establish prediction models in Approach 1, and only take Random forest and XGBoost to establish models in Approach 2. The results illustrate that a higher prediction accuracy leads to higher profits in trading strategy implementation. We plot the results taking by Approach 1 and 2 for 'audcad' pair in Figure A.1 and A.2, which shows that the investment payoffs of these two approaches have a similar trend but the profit growth of Approach 2 will be steeper. Moreover, after introducing the flat-trend prediction model for transaction costs consideration, the profits reduce to an extent, which is more reasonable in the real-world financial market. However, it still makes money in most cases and has better performances than the buy and hold strategy. The backtesting results for 'eurgbp' pair taking by three machine learning methods is shown in **??**, it can see that the trend of profits taken by these three methods is similar most of the time, and LSTM performs better in the first quarter. Significantly, they all predict a sharp dropping in December 2020, and the profits grow smoothly throughout the year.

Table 3.2: The Return of 20 Fx Pairs in 2020 (Approach 1)

	Return %			
Forex Pair	Random Forest	XGBoost	LSTM	Buy and Hold
audcad-1m	32.75	34.20	46.56	7.57
audjpy-1m	20.99	19.04	15.76	3.47
audusd-1m	14.55	16.80	10.15	9.29
chfjpy-1m	46.94	50.94	58.56	3.57
ethusd-1m	1637.94	3054.72	83.50	473.18
euraud-1m	41.60	34.49	48.11	0.14
eurcad-1m	38.33	38.98	50.33	7.05
eurchf-1m	38.58	48.79	45.56	-0.61
eurgbp-1m	53.60	57.87	61.47	6.04
eurjpy-1m	30.97	31.13	31.83	3.78
eurusd-1m	6.85	12.10	15.01	9.42
gbpcad-1m	94.96	92.27	111.34	0.70
gbpchf-1m	52.20	52.64	51.05	-5.57
gbpjpy-1m	43.66	50.12	20.17	-2.06
gbpusd-1m	22.31	14.70	10.54	3.09
nzdusd-1m	17.84	24.81	12.66	6.91
usdcad-1m	5.84	4.95	10.09	-1.88
usdchf-1m	14.15	17.93	16.48	-8.58
usdjpy-1m	5.24	7.94	3.62	-6.61
xauusd-1m	22917.96	24589.36	2850.01	25.02

Table 3.3: The Return of 20 Fx Pairs in 2020 (Approach 2)

	Return %		
Forex Pair	Random Forest	XGBoost	Buy and Hold
audcad-1m	11.48	14.79	7.57
audjpy-1m	2.87	7.94	3.48
audusd-1m	-3.68	10.39	9.29
chfjpy-1m	19.51	28.07	3.57
ethusd-1m	1057.70	1086.69	473.18
euraud-1m	12.08	30.67	0.14
eurcad-1m	20.12	22.17	7.05
eurchf-1m	19.55	19.94	-0.61
eurgbp-1m	20.74	37.54	6.04
eurjpy-1m	13.43	14.77	3.73
eurusd-1m	2.96	4.03	9.42
gbpcad-1m	38.76	61.92	0.70
gbpchf-1m	20.07	31.72	-5.57
gbpjpy-1m	16.91	23.68	-2.06
gbpusd-1m	2.49	10.24	3.09
nzdusd-1m	-3.63	8.02	6.91
usdcad-1m	-3.13	1.36	-1.88
usdchf-1m	4.61	2.85	-8.58
usdjpy-1m	-2.22	-3.54	-6.61
xauusd-1m	3252.52	2516.83	25.02

Conclusion and Future Work

In this paper, we try different approaches to select the optimal features used for foreign exchange currency trend prediction, including technical analysis based features, time-series based features, and signature-based features, and show that the close price signature based model performs as well as technical indicators based model in the short-term trend prediction.

The technical indicators based prediction has proved to achieve satisfactory performance in financial market trend forecast. In the future trend prediction, high prediction accuracy is achieved on the long-term (2/3 weeks) predictions but comes with the high volatility in the performances proved by the increasing window cross-validation. Thus, we choose a short-term (5 minutes) trend prediction in the whole paper. In the model selection process, we demonstrate that the close price signature based predictive classification model leads to a decent prediction performance as the technical indicators based prediction does, together with a lower standard deviation of the cross-validation accuracy, which further manifests that the significant ability of signature transformation in extracting information from time-series data. Significantly, technical indicators draw out the information from five financial data streams, including open/high/low/close/volume, while the close price signature based prediction grasps information only from the close price data stream, demonstrating again the powerful ability of information extraction for signature transformation.

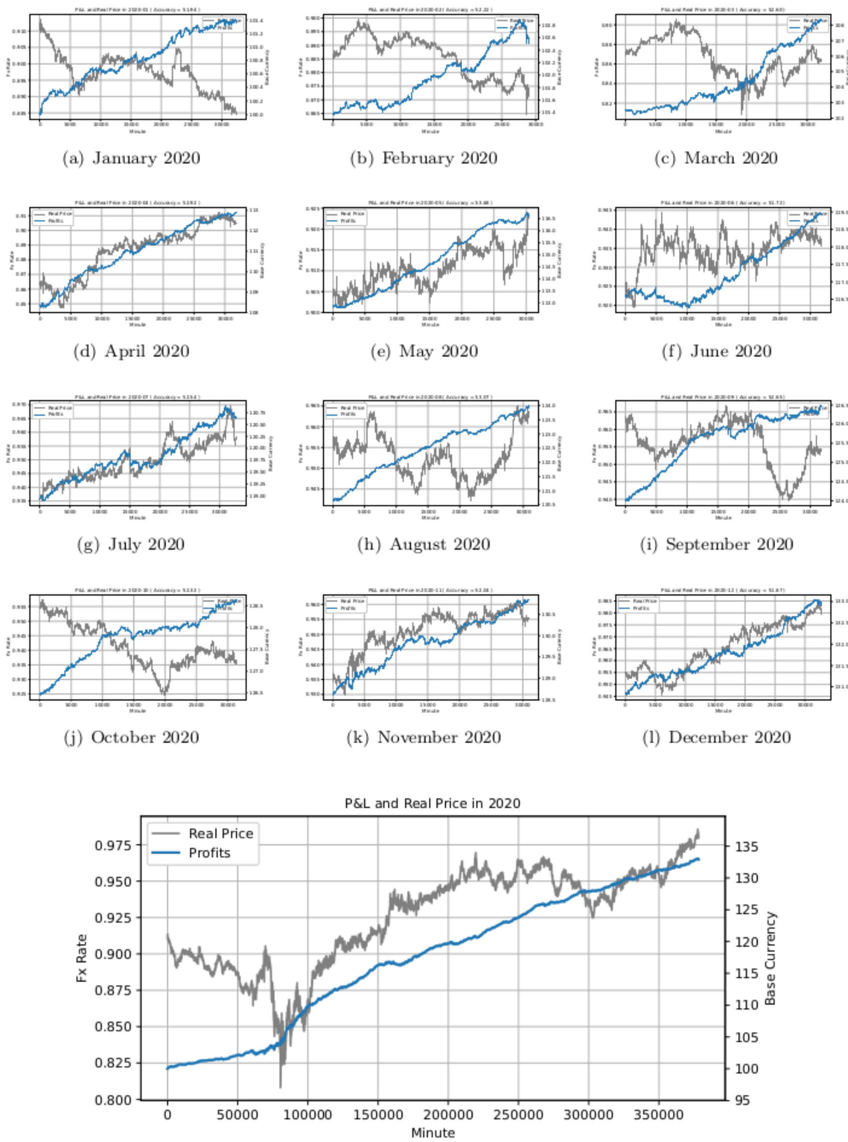
Moreover, the approach to compute the signature of time-series data is also vital for the quality of the extracted information. We make much effort on trying different approaches in data processing, choosing the best historical windows, determining the optimal time series data used for extraction, selecting the optimal truncation levels, signature augmentations, rescaling factors and so on. Through a large number of comparisons, we find out that model parameters choice does not affect the performance of the prediction model to a degree, while the type and the length of time series data are essential for extracting valuable information by signature transformation. Finally, we combine two important features for short-term trend prediction, technical indicators and close price signature based features, where we select the top 10 meaningful technical indicators and 15-minute length historical window to compute the signature, which generates a stable and adequate prediction to assist us in determining the buy/sell signals in the trading strategy. We also evaluate the practical application for this model by taking trading strategy backtesting for 20 foreign exchange pairs in 2020. In the model construction, we use the Random Forest (RF), Extreme Gradient Boosting (XGB) and Long-Short Term Memory (LSTM) network to establish the predictive classification model. We employ two approaches to implement the trading strategy based on the predictive binary classification model in the backtesting framework. One is to identify the trading signals only based on the rising/dropping trend prediction, but in reality, the transaction cost is unavoidable, so we need to ensure that the price movement is sufficiently large enough to make up the transaction cost. Based on this consideration, we construct one more specialised model in flat-trend prediction, then combine this model with the price movement prediction model, hence identifying a sufficiently large price movement. Even though the profits would decrease to a great extent in this case, it is more reasonable in the financial market; at the same time, the return of this trading strategy for 20 currency pairs are still higher than the buy-and-hold strategy. Besides, the LSTM model does not perform better than the other two tree-based algorithms in some specific currency pairs, which might be because its tuning process is more complicated than the other two algorithms, and it might be necessary to tune it for each currency pair individually.

Forecasting financial market trend is a challenging issue in financial areas. What we discussed in this paper provides a reliable way to predict the short-term price movement by introducing the signature methods to dig out the hidden and valuable information from its historical data streams. However, there are many attempts to be made to improve the performance of the predictive models. For example, since there exists a correlation between some currency pairs, such as EURUSD and

GBPUSD, thus more valuable information could be extracted for prediction from a correlated currency pair. Moreover, the combination of volatility prediction and trend prediction model also gives a possible way to identify buy or sell signals in the trading strategy.

Appendix A

Figures



(a) January 2020 (b) February 2020 (c) March 2020

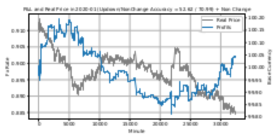
(d) April 2020 (e) May 2020 (f) June 2020

(g) July 2020 (h) August 2020 (i) September 2020

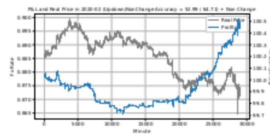
(j) October 2020 (k) November 2020 (l) December 2020

(m) 2020

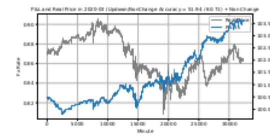
Figure A.1: Approach 1 - AUDCAD P&L and Real Price in 2020 (XGBoost)



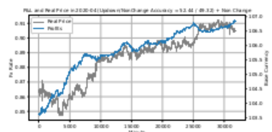
(a) January 2020



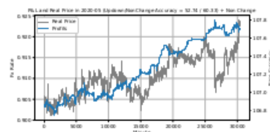
(b) February 2020



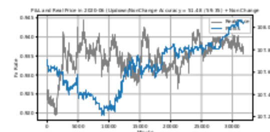
(c) March 2020



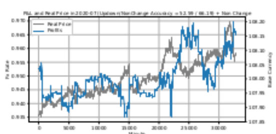
(d) April 2020



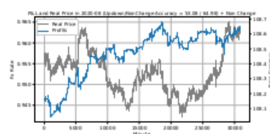
(e) May 2020



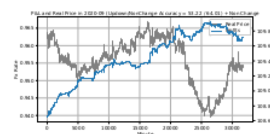
(f) June 2020



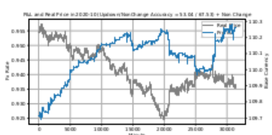
(g) July 2020



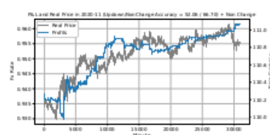
(h) August 2020



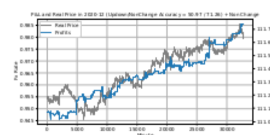
(i) September 2020



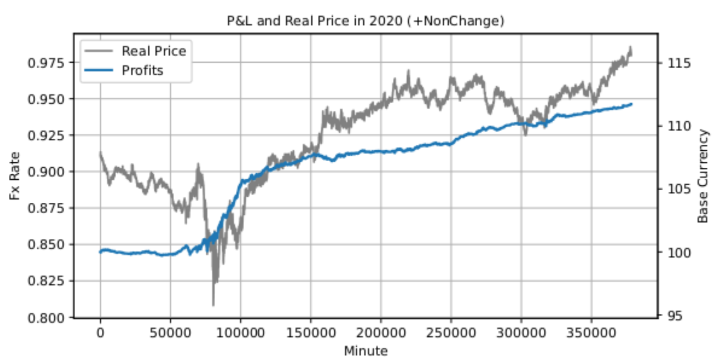
(j) October 2020



(k) November 2020

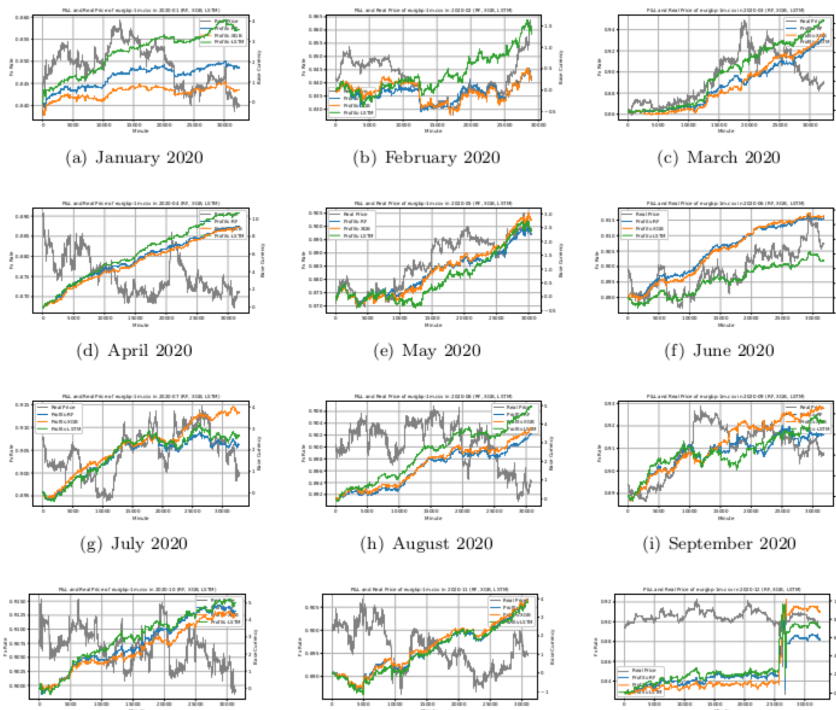


(l) December 2020



(m) 2020

Figure A.2: Approach 2 - AUDCAD P&L and Real Price in 2020 (XGBoost)



(a) January 2020 (b) February 2020 (c) March 2020

(d) April 2020 (e) May 2020 (f) June 2020

(g) July 2020 (h) August 2020 (i) September 2020

(j) October 2020 (k) November 2020 (l) December 2020

Figure A.3: The trading simulations for 'EURGBP' pair in 2020 (RF, XGBoost, LSTM)

Bibliography

[1] Triennial central bank survey of foreign exchange and over-the-counter (otc) derivatives markets in 2019. *The Bank for International Settlements*, Sep 2019.

[2] Hongju Yan and Hongbing Ouyang. Financial time series prediction based on deep learning. *Wireless Personal Communications*, 102(2):683–700, 2018.

[3] Andrea Picasso Ratto, Simone Merello, Luca Oneto, Yukun Ma, Lorenzo Malandri, and Erik Cambria. Ensemble of technical analysis and machine learning for market trend prediction. In *2018 IEEE symposium series on computational intelligence (ssci)*, pages 2090–2096. IEEE, 2018.

[4] Daniel Levin, Terry Lyons, and Hao Ni. Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv preprint arXiv:1309.0260*, 2013.

[5] Lajos Gergely Gyurkó, Terry Lyons, Mark Kontkowski, and Jonathan Field. Extracting information from the signature of a financial data stream. *arXiv preprint arXiv:1307.7244*, 2013.

[6] João Nobre and Rui Ferreira Neves. Combining principal component analysis, discrete wavelet transform and xgboost to trade in the financial markets. *Expert Systems with Applications*, 125:181–194, 2019.

[7] Kyung Keun Yun, Sang Won Yoon, and Daehan Won. Prediction of stock price direction using a hybrid ga-xgboost algorithm with a three-stage feature engineering process. *Expert Systems with Applications*, page 115716, 2021.

[8] Sangyeon Kim and Myungjoo Kang. Financial series prediction using attention lstm. *arXiv preprint arXiv:1902.10877*, 2019.

[9] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. IEEE, 2017.

[10] Ben Hambly and Terry Lyons. Uniqueness for the signature of a path of bounded variation and the reduced path group. *Annals of Mathematics*, pages 109–167, 2010.

[11] James Morrill, Adeline Fermanian, Patrick Kidger, and Terry Lyons. A generalised signature method for multivariate time series feature extraction. *arXiv preprint arXiv:2006.00873*, 2020.

[12] Ilya Chevyrev and Andrey Kormilitzin. A primer on the signature method in machine learning. *arXiv preprint arXiv:1603.03788*, 2016.

[13] Jianan Han, Xiao-Ping Zhang, and Fang Wang. Gaussian process regression stochastic volatility model for financial time series. *IEEE Journal of Selected Topics in Signal Processing*, 10(6):1015–1028, 2016.

[14] Huy D Huynh, L Minh Dang, and Duc Duong. A new model for stock price movements prediction using deep neural network. In *Proceedings of the Eighth International Symposium on Information and Communication Technology*, pages 57–62, 2017.

[15] Jingtao Yao and Hean-Lee Poh. Forecasting the klse index using neural networks. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 2, pages 1012–1017. IEEE, 1995.

[16] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[17] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[18] Christopher M Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

[19] Philipp Probst, Marvin N Wright, and Anne-Laure Boulesteix. Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(3):e1301, 2019.

[20] X Let. Pattern classification. 2005.

[21] Hans Buehler, Blanka Horvath, Terry Lyons, Imanol Perez Arribas, and Ben Wood. A data-driven market simulator for small data environments. *Available at SSRN 3632431*, 2020.

Lidan_Xing_01935185

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53