APPLICATION OF SIGNATURES FOR FORECASTING

Modelling project work 3

Bat-erdene Egshiglen

Supervisor: Tikosi Kinga

Budapest, Hungary

2020, I semester

1 Summary

The signatures of a rough path contain information about its behaviour. In this project we wish to give a brief theoretical overview of this concept and also propose an application. In the next part, we review the definition of the signature of a path and some important properties.

The third section will contain information about the application, and connection of Taylor's polynomial to signature. Theorem 3.2 tells us that computing the signatures of a path gives us a Taylor-polynomial for rough paths, where we substitute the derivatives with the appropriate signatures. As an example to demonstrate the use of signatures, in this project we wish to utilise the information encoded in the signatures of a rough path in order to make statistical predictions about the future behaviour of the time series. The time series we collected is financial information, stock prices, and in that section, we present a way of making predictions using signatures. Stock prices are well known to be volatile, that is the reason why rough path analysis is needed. We take advantage of the fact that these iterated integrals can easily calculated with already implemented "iisignature" package (see [6] for more information). The results of the analysis show that while clearly there is no chance to predict the future exactly, sometimes the method can give us nice approximate results.

2 Introduction to signatures

Before anything else, we should recall the definition of our main concept of the project: signature of a path.

Definition 2.1. Let us assume $X_t : [a,b] \mapsto \mathbb{R}^m$, then the signature of the path X_t is an infinite series of the iterated integrals

$$S(X)_{a,b} = (1, S(X)^1_{a,b}, S(X)^2_{a,b} \ldots, S(X)^m_{a,b}, S(X)^{11}_{a,b}, \ldots).$$

S(X)$^i_{a,t} = \int_{a<s<b} dX^i_s = X^i_t - X^i_0$

S(X)$^{i,j}_{a,t} = \int_{a<s<b} S(X)^i_{a,s}\, dX^j_s = \int_{a<r<s<t} dX^i_r dX^j_s$(1)

Now, we can move on to the properties which will be useful for computational, statistical problems, and analysing time series.

2.1 Reparametrization

The first property states that, the signature $S(X)_{a,b}$ remains invariant under time reparametrizations of X.

Proof. We have two paths $X, Y : [a, b] \to \mathbb{R}$, we denote the reparametrization by $\varphi : [a, b] \to [a, b]$. We assign new paths : $X_\varphi = \bar{X}, Y_\varphi = \bar{Y}$. If we differentiate $\bar{X}$, we get : $(\bar{X}_t)' = (X_{\varphi(t)})' = (X_{\varphi(t)})' * (\varphi(t))'$. It means

$$\int_a^b \bar{Y}_t\, dX_t = \int_a^b \bar{Y}_t\, X_{\varphi(t)})' * \varphi(t)' \tag{2}$$

by substituting $u = \varphi(t)$

$$\int_a^b \bar{Y}_t\, dX_t = \int_a^b \bar{Y}_t\, X_{\varphi(t)})' * \varphi(t)' = \int_a^b \bar{Y}_u\, dX_u \tag{3}$$

□

2.2 Shuffle product

The next fundamental property is useful when we deal with higher dimensions, since we it allows us to describe the product of two signature terms as a summation using other terms. Additionally, the signature satisfies the shuffle product formula for every path X, this was discovered by Chen in 1957.

Definition 2.2. A (k,m) shuffle is a permutation of $\{1,\ldots,k+m\}$ if $\sigma^{-1}(1) < \cdots < \sigma^{-1}(k)$ and $\sigma^{-1}(k+1) < \cdots < \sigma^{-1}(k+m)$. Notation of collection of all (k,m) shuffles : Shuffle(k,m)

Given two multi-indexes, $I = \{i_1,\ldots,i_k\}$ and $J = \{j_1,\ldots,j_m\}$, $(i_1,\ldots,i_k,j_1,\ldots,j_m \in \{1,\ldots,d\})$, the shuffle product of I and J, which we denote by $I \sqcup J = \{(r_{\sigma(1)},\ldots,r_{\sigma(k+m)})|$ $\sigma \in$Shuffle$(k,m)\}$.

Theorem 2.1. We have a path $X : [a,b] \mapsto \mathbb{R}^d$, and $I = (i_1,\ldots,i_k)$ and $J = (j_1,\ldots,j_m)$, $(i_1,\ldots,i_k,j_1,\ldots,j_m \in \{1,\ldots,d\})$, then

$$S^I(X)S^J(X) = \sum_{K \in I \sqcup J} S^K(X) \tag{4}$$

Proof.

$$S^I(X)S^J(X) = \int\cdots\int_{0<u_1<\cdots<u_k<1} dX^{i_1}_{u_1}\ldots dX^{i_k}_{u_k} \int\cdots\int_{0<t_1<\cdots<t_m<1} dX^{j_1}_{t_1}\ldots dX^{j_m}_{t_m} =$$

$$= \sum_{\sigma \in Shuffles(k,m)} \int\cdots\int_{0<v_1<\cdots<v_k+m<1} dX^{r_{\sigma(1)}}_{v_1}\ldots dX^{r_{\sigma(k+m)}}_{v_k+m} == \sum_{K \in I \sqcup J} S^K(X)$$

□

2.3 Chen's identity

Before continuing to define the Chen's identity, we should know what is the formal power series and a concatenation of two paths.

Definition 2.3. We call the vector space of series a non-commuting formal power series if, for all e_l, when $l \in [1,d]$ indeterminates, it has a form of

$$\sum_{k=0}^{\infty} \sum_{i_1,\ldots,i_k \in \{1,\ldots,d\}} \lambda_{i_1,\ldots,i_k} e_{i_1,\ldots,i_k}, \;\; \lambda \in \mathbb{R}. \tag{5}$$

Note that the space of formal power series is an algebra, if it has the usual scalar multiplication, addition and the tensor product.

But how is it related to signatures and paths, one may ask. The answer is simple, we can express our signatures with the help of the previous definition, which becomes clear if we observe the multi indices.

$$S(X)_{a,b} = \sum_{k=0}^{\infty} \sum_{i_1,\dots,i_k \in \{1,\dots,d\}} S(X)_{a,b}^{i_1,\dots,i_k} e_{i_1,\dots,i_k}. \tag{6}$$

Definition 2.4. Let $X : [a,b] \mapsto \mathbb{R}^d, Y : [b,c] \mapsto \mathbb{R}^d$, then the concatenation of X and Y is a path from $[a,c] \mapsto \mathbb{R}^d$:

$$(X * Y)_t = \begin{cases} X_t, & \text{if } t \in [a,b] \\ X_b + (Y_t - Y_b), & \text{if } t \in [b,c]. \end{cases}$$

Theorem 2.2 (Chen's identity)**.** As usual, let us have two paths $X : [a,b] \mapsto \mathbb{R}^d, Y : [b,c] \mapsto \mathbb{R}^d$, then

$$S(X * Y)_{a,c} = S(X)_{a,b} \otimes S(Y)_{b,c}. \tag{7}$$

Proof. To start proving, we should remember what is a signature and what are the coordinate paths:

$$S(X) = (1, X, X^2, \dots) \tag{8}$$

$$X^n = \int \cdots \int_{0<u_1<\cdots<u_n<1} dX_{u_1} \otimes \cdots \otimes dX_{u_n}. \tag{9}$$

Now, lets assign a new path $Z = X * Y$, it means

$$\begin{aligned} Z^n = &\underset{s<u_1<\cdots<u_n<u}{\int \cdots \int} dZ_{u_1} \otimes \cdots \otimes dZ_{u_n} = \\ &= \sum_{k=0}^{n} \underset{s<u_1<\cdots<u_k<t<u_{k+1}<u_n<u}{\int \cdots \int} dZ_{u_1} \otimes \cdots \otimes dZ_{u_n} = \\ &= \sum_{k=0}^{n} \underset{s<u_1<\cdots<u_k<t}{\int \cdots \int} dX_{u_1} \otimes \cdots \otimes dX_{u_k} \underset{t<u_{1+k}<\cdots<u_k<u}{\int \cdots \int} dX_{u_{k+1}} \otimes \cdots \otimes dX_{u_n} = \\ &= \sum_{k=0}^{n} X^k \otimes Y^{n-k} \end{aligned} \tag{10}$$

□

2.4 Uniqueness and Time reversal

Definition 2.5. A path $X : [0,1] \mapsto \mathbb{R}^d$ is tree-like, if $\exists f : [0,1] \mapsto [0,\infty) : f(0) = f(1) = 0$ and $\forall s,t \in [0,1], s \leq t :$

$$\|X_s - X_t\| \leq f(s) + f(t) - 2 \inf_{u \in [s,t]} f(u). \tag{11}$$

Theorem 2.3. Assume $X, Y : [a,b] \mapsto \mathbb{R}^d$, then

$$\forall t \in [a,b] : X_t = Y_t \implies \forall k \in \{1,\dots,d\} : \; S^k(X) = S^k(Y). \tag{12}$$

Theorem 2.4 (Time reversed signature)**.** If we have a path $X : [a,b] \mapsto \mathbb{R}^d$, then the following is true:

$$S(X)_{a,b} \otimes S(\overleftarrow{X})_{a,b} = 1. \tag{13}$$

Here $\overleftarrow{X}$ is the time reversal, meaning $\overleftarrow{X}_t = X_{a+b-t}, \forall t \in [a,b]$.

Definition 2.6. Two paths X and Y are called tree-like equivalent if $X^* \overleftarrow{Y}$ is tree-like (where $\overleftarrow{Y}_t = Y_{1t}$).

Now, since we defined a tree-like path we can state the uniqueness theorem.

Theorem 2.5. Assume X is a continuous path with bounded variation, then $S(X) = 1 \iff X$ is tree-like, additionally $S(X)$ is unique up to a tree-like equivalence. Which means that, the equivalence in Definition 2.5 is the same as Theorem 2.3.

After all these properties, we can conclude, the path cannot be simply reconstructed from its signature in the exact speed it travels, because of the time invariance property. However, it is not an awful thing, since we are unable to reconstruct a Geometric Brownian motion from only its volatility and drift. Furthermore, when X does not cross itself, meaning it is a tree-like path, we can recreate the geometry of the traverse of our path.

3 Application

After reviewing all fundamental information, we have now a grasp on how signature works in theory, but how do we use all these in real life?

The most known theorem for approximating functions is Taylor's theorem (Theorem 3.1), and signature describes the same thing, but for rough paths we do not have partial derivatives.

Theorem 3.1. $f(x,y) \approx f(a,b) + (x-a)f_x(a,b) + (y-b)f_y(a,b)$

Remark. While it is true if we take higher derivatives we get a better approximation even in the signature method (going with higher multi-indices), for now we will only go until second derivative and multi-indices such as $S^1, S^2, S^{1,1}, S^{1,2}, S^{2,1}, S^{2,2}$.

The advantage of using the signature method is that any multivariate distribution of data could be represented as a path in a high-dimensional space $\mathbb{R}^d$.

The process of using signature would be the following:

1. We have a data stream
2. Embed it to $\mathbb{R}^d$
3. Compute the iterated integrals up to a level of truncation
4. Use the resulting set of features for analysing the data/forecasting

In this stage of the project, the data stream is a Geometric Brownian motion. Unfortunately, Theorem 3.2 from [5] states that the Brownian motion is nowhere differentiable, almost surely. Therefore, we have :

Theorem 3.2. $f(X) = c_0 + c_1 S(X)^1_{a,b} + c_2 S(X)^2_{a,b} + c_{1,1} S(X)^{1,1}_{a,b} \dots$

We may have a list of data, which we need to reconstruct as a path. They are many ways to do so, for example, piece-wise linear interpolation, rectilinear interpolation or lead-lag transformation, which we will see in the next example.

Example 3.1. Let us say $\{X^1\} = \{1, 2, 5, 6\}$, $\{X^2\} = \{1, 6, 5, 3\}$, then the embedding with lead-lag would look like this: $X^{1,lead} = \{1, 2, 2, 5, 5, 6, 6\}$,$X^{1,lag} = \{1, 1, 2, 2, 5, 5, 6\}$. Also, after using the definitions and calculating integrals we get $S(X) = \{5, 2, 12.5, -9, 19, 2\}$.

$S^1(X) = X^1_3 - X^1_0 = 6 - 1 = 5$, $S^2(X) = X^2_3 - X^2_0 = 3 - 1 = 2$,

$S^{1,1} = (X^1_3 - X^1_0)^2/2 = 25/2$, $S^{2,2} = (X^2_3 - X^2_0)^2/2 = 4/2$,

$S^{1,2} = \int_0^3 \int_0^3 dX^1_{t_1} dX^2_{t_2} =$

I have computed the slope for X^1 is $(1, 3, 2)$, for X^2 it is $(5, -1, -2)$.

$= \int_0^1 \left[\int_0^{t_2} dX^1_{t_1}\right] 5dt_2 + \int_1^2 \left[\int_0^{t_2} dX^1_{t_1}\right] (-1)dt_2 + \int_2^3 \left[\int_0^{t_2} dX^1_{t_1}\right] (-2)dt_2 =$

$= \int_0^1 [X_{t_2} - X_0]\, 5dt_2 + \int_1^2 [X_{t_2} - X_0]\, (-1)dt_2 + \int_2^3 [X_{t_2} - X_0]\, (-2)dt_2$

$= \int_0^1 [t_2 + 1 - 1]\, 5dt_2 + \int_1^2 [2t_2 + 1 - 1]\, (-1)dt_2 + \int_2^3 [2t_2 - 1]\, (-2)dt_2$

$= -9$

One difficult thing we have encountered while trying to calculate the signature would be the integrals especially without calculators or computers. Hence, we installed "iiisignature" with python which helps us to efficiently and quickly get our results.

```python
pip install iisignature
import iisignature as isig
import numpy as np
data= ([1,1], [2,6], [5,5], [6,3])
isig.sig(data, 2, 1)
output: (array([5., 2.]), array([12.5, -9. , 19. ,  2. ]))
```

Now, let us test it with real life financial data. First, we imported the Google stock data (precisely the weekly data from 2020.10.1 until 2021.9.30) from Yahoo Finance. From now on we will use the adjusting closing price, which takes into account some factors that may affect the price after the market closes. We can have a grasp on how our data looks like from Table 1 and Figure 1.

0	1458.420044	27	2285.879883	47	2891.0100100
1	1515.219971	28	2297.760010	48	2895.500000
2	1573.010010	29	2315.300049	49	2838.419922
3	1641.000000	30	2410.120117	50	2829.270020
4	1621.010010	31	2398.689941	51	2852.659912
5	1761.750000	32	2316.159912	52	2665.310059
...	...	...	...	...	

Table 1: The Adjusting Closed price

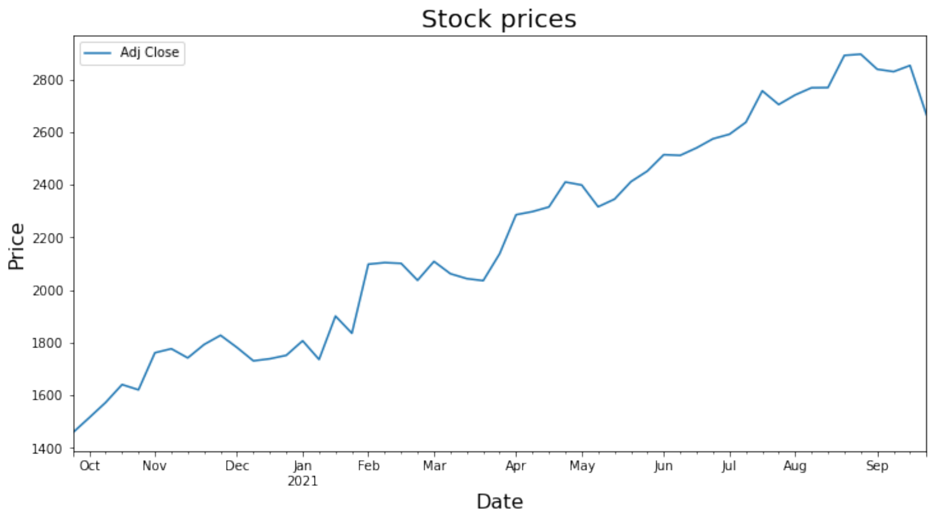

Figure 1: The plot of the Adjusting Closed price

As we have mentioned before and saw in Example 3.1, we will use lead-lag transformation to construct a two dimensional path from our discrete one dimensional data.

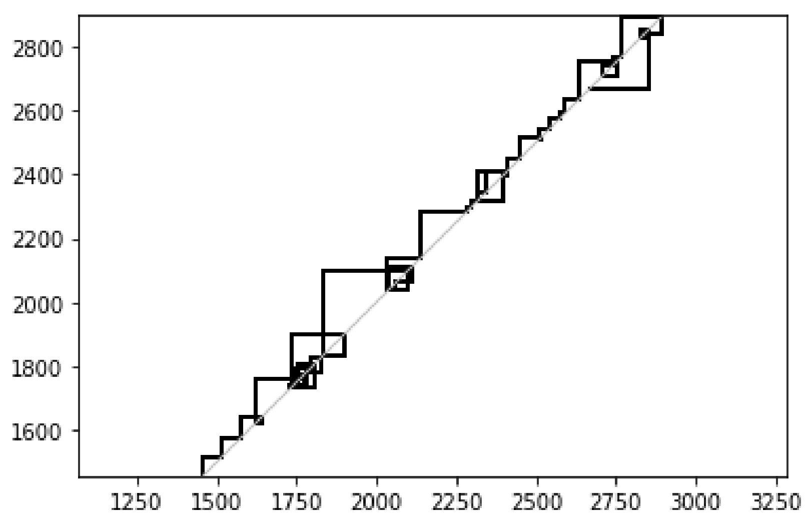

Figure 2: The Lead-Lag

Our aim is to predict the future stock prices using our first 30 weeks of date. For this, firstly, we would like to write a function that will provides us with a dataframe that contains means of 2 consecutive prices, for the rolling window of size 30. Not to forget, we also should consider the time of the given data while we do all these.

Since we concatenated the lead and lag to get our stream, we will calculate the signature (level 2) of the data and stored them in a table.

To make predictions about the future signatures of the time series, we used the past information (i.e. the signatures the program computed for the rolling windows of the past) and fit lasso linear regression. The parameter alpha of the regression was optimised with GridSearchCV.

Last, we use the Train-Test Split Procedure (from sklearn.model), where we split our signature table and next mean dataframe into a train and test, with the objective of predicting the next future prices. The result can be seen below in Figure 3.

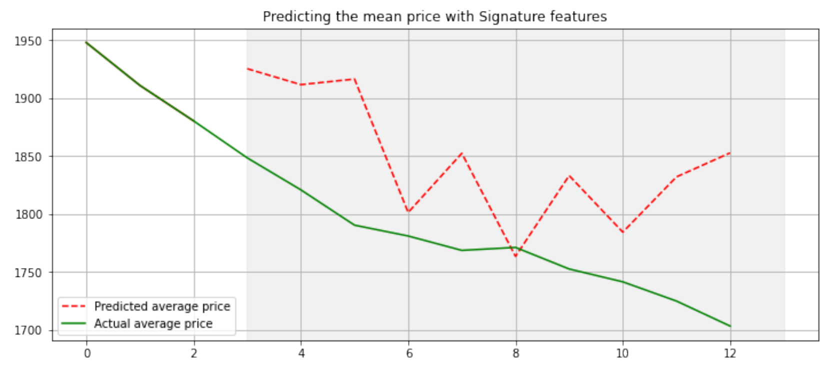

Figure 3: Prediction of Google

Unfortunately, it is not a hundred percent accurate, however the difference varies between $0 - 250$ dollars, which can be considered a lot, but compared to the maximum price of a stock it is not extremely much, additionally, we can see the method predicts the down slope in our data.

Now, let us see another data set, namely Samsung stock prices in the same time, using our method.

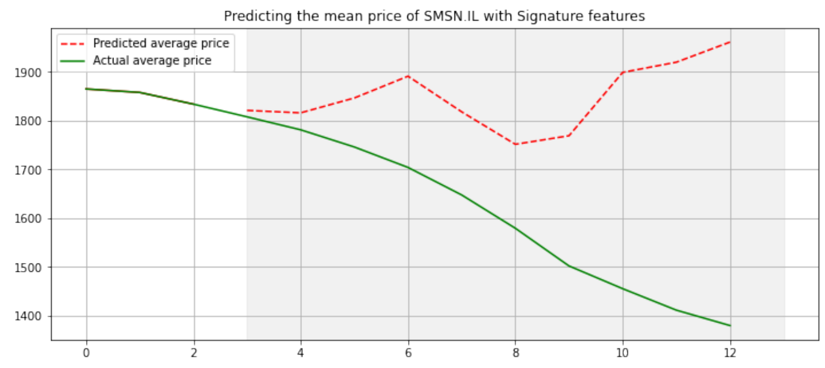

Figure 4: Prediction of Samsung

In conclusion, we can not predict the target every time with a nice accuracy using signature. There may be a few nice forecasts, but, since data such as stock prices can be swayed by a lot of factors, it is still not likely the perfect method for all the time, if there is one.

References

[1] Adeline Fermanian: Signature and statistical learning,
https://afermanian.github.io/docs/thesis_fermanian.pdf

[2] Ilya Chevyreva and Andrey Kormilitzin: A Primer on the Signature Method in Machine Learning,
https://arxiv.org/pdf/1603.03788.pdf

[3] Gérard Biau and Adeline Fermanian: Chapter 4, Learning with Signatures
https://afermanian.github.io/docs/Biau_and_Fermanian_2019.pdf

[4] Chapter 4: Approximating functions by Taylor Polynomials,
http://www.math.smith.edu/~rhaas/m114-00/chp4taylor.pdf

[5] Aaron Mcknight, Some basic properties of Brownian Motion,
http://www.math.uchicago.edu/~may/VIGRE/VIGRE2009/REUPapers/McKnight.pdf

[6] Jeremy Reizenstein, Centre for Complexity Science Ben Graham, Department of Statistics and Centre for Complexity Science University of Warwick : iisignature (version 0.23), August 2018
https://warwick.ac.uk/fac/cross_fac/complexity/people/students/dtc/students2013/reizenstein/iisignature.pdf

[7] Rattana Pukdee: Predict Bitcoin prices by using Signature time series modelling,
https://towardsdatascience.com/predict-bitcoin-prices-by-using-signature-time-series-modelling-cf3100a882cc

[8] Mathanraj Sharma: Grid Search for Hyperparameter Tuning,

`https://towardsdatascience.com/grid-search-for-hyperparameter-tuning-9f63945e8fec`

[9] Imanol Pérez: Rough Path Theory and Signatures Applied To Quantitative Finance - Part 1,

`https://www.quantstart.com/articles/rough-path-theory-and-signatures-applied-to-quantitative-finance-part-1/`