Predicting Foreign Exchange Market by Machine Learning

Xue Zhihuan	3035424982	jeffxue@hku.hk
Kan Lei	3035414054	kanlei@hku.hk
Zhang Mengyun	3035414183	zmy95523@.hku.hk

Abstract

Our project is about Forex market data prediction using machine learning models. In the project, we use a lot of data to train three basic classification models and test the performance and accuracy of them.

The data for our models include market data and fundamental data from 3 different sources. Fundamental data include CFTC Currency Futures holdings. And market data include Forex and Stock Index market data. We also calculate some indicators from our Forex data. There are total 63 features as the inputs to our models. Details can be seen in Methodology part.

The purpose of our study is to get a positive cumulative return and a win rate higher than 50% on the test data. We achieve it by 2 of the 3 models, which are LogisticRegression and LinearSVC. We also use a "thresh" strategy during the test to avoid the effect of market noise. The result is very interesting .

Introduction

We are trying to predict the trend of foreign exchange market by training simple classification models on previous data. It is important for investing and earning in exchange market. There is a former article named "Evaluating machine learning classification for financial trading: An empirical approach" using models of OneR, C4.5, Jrip, LMT, Kstar and NaiveBayes and finally got a positive result.

Based on former experts, we use three models of RandomForestClassifier, LogisticRegression and LinearSVC, and add more features to do the prediction. According to the theory of Technical Analysis, history will recur in the future. Therefore, we believe a model trained on former data can predict the future market data .

Literature Review

4.1 Machine Learning (ML) & Artificial Neural Networks (NNs)

Previous studies have reported effective uses of ML practices in performing financial forecasting (Gerlein et al., 2016; Kumar et al., 2016; Yang & Lin, 2017). Specifically, ML practices associated with financial forecasting are multifaceted, ranging from rule-based systems to genetic computing, with *technical indicators* being employed (Maggini et al., 1997). yet a notable drawback is referred to as the difficulty in retrieving appropriate, applicable indicators, as the rules related to financial forecasting keep changing along financial time series.

Of various ML tools, artificial NNs are regarded frequently practiced for financial prediction. Previous studies, for instance the work by Tenti (1996), examine three recurrent NNs in accordance with respective returns drawn upon financial forecasting simulation about currency futures. A set of *technical indicators* were utilized as NNs' inputs, such as the rate of change and trend movement index. Having considered trading costs, Tenti (ibid) concludes that, with the aid of NNs, reliable forecasting can be achieved.

4.2 Other Useful Tools

Aside from NNs, Maggini et al. (1997) suggested a heuristic strategy with which various inputs associated with a nonlinear ML algorithm are selected; meanwhile, the option related to time series prediction is negated and the issue to identify the class of price variation is limited. *K-nearest neighbours*, with a sliding window dataset, is adopted to retain the model at every time step. The study conducted by Maggini et al (1997) concludes that predicting price variation accurately appears to be infeasible. Thus, such a heuristic method is not recommended, as the results are because of the weaknesses of the model that picks inappropriate inputs.

Interestingly, Li et al (1999) forecasted expected return in the Dow Jones Industrial Average with another model – Financial Genetic Programming; they compared the return with random decisions. Simple *technical indicators*, for instance short- and long-term moving averages and price filters, were employed. Alongside focusing on accuracy, Li et al (1999) also placed a great emphasis upon annualised returns and positive returns derived from investment simulation. The result displays more than 60% in positive returns, along with 40% in annualized returns. Clearly, these estimates present an excellent return, while trading costs are not even accounted. Hence, *technical indicators* would lead to profitable models that can be employed for prediction in complicated time series.

Single agents were used by Barbos and Belo to perform algorithmic trading in the FOREX Market, hedge fund management with a micro-society and a multi-agent system for multiple markets trading. Their works regarded profitability and maximum drawdown as performance metrics. Also, the proposed architecture is divided into three modules with corresponding functions in forecasting next trend via means of an ensemble of binary classifier (Barbosa & Belo, 2008b), establishing a risk-management module to determine the amount necessitated to invest in each trade (Barbosa & Belo, 2010), and setting up a rule-based system that takes human experience into account to improve the trading decision (Barbosa & Belo, 2008a). The result derived from this 3-module system produces a success rate of 66.67% in profitability over the tested period.

Methodology

First, we get our USD/JPY market data (2007.01.01-2007.09.07) from http://www.forextester.com/data/datasources. It's in the frequency of 1min/bar and includes open,close,high,low,volume and so on. The original data is in txt format, but because it is too big to upload to Moodle, we read it using pd.read_csv('USDJPY.txt') and save it to a HDF5 file named "data.h5". So the code we upload now directly read data from the HDF5 file. Then, We remove some beginning and ending values to make sure the data start from a Sunday(market open time) and end at a Friday(market close time). Then we loop through every week's data to add all missing minute data.

In details, for each week's data, we firstly fill or cut the beginning and ending data to the nearest 30 or 60 minutes. For the missing data in the middle of a week, we use the open price of the next minute to complete the filling. This kind of filling is reasonable because Forex market is a global, 24-hr open market. Market shutting is very rare.

Afterwards, we aggregate data from 1min/bar to k min/bar. Then we draw a graph of continuous close prices to verify the competence of data.

Then we use TA-lib to calculate indicators and name it indi_df. The indicators included lag_r, lag_r_MA, SMA, EMA, STOCH, ADX, ATR, STD, SAR, WILLR, STOCHRSI, RSI and TYPPRICE.

we get S&P500 and Nikkei225 data with the help of pandas-datareader module. And the final resource is Yahoo Finance website. We merge the two indexes' returns into a DataFrame called idx_data. We also get CFTC data using modules of requests and

Beautifulsoup from http://www.99qh.com/d/cftc.aspx. And the final source is CME Group. We scrap each week's data with a loop into a DataFrame called cftc_data. During the scrapping, we set a 0.5 seconds interval to obey the scrapping rule. We also set a retry mechanism when facing connection error during the scrapping.

After that, we merge Forex mkt_data, cftc_data, idx_data together into the final_data. During this process, we try our best to avoid "future data". For example:

1. when merging cftc_data to mkt_data, we only have "until date" for cftc_data, not the "publish date". The "publish date" is 4 days later on Saturday so we merged cftc_data with the mkt_data of the next week.

2. when merging idx_data to mkt_data, because S&P 500 and Nikkei 225 only can trade for several hours each day, and the time zones of their markets (one is GMT-5, one is GMT+7) are different from Forex data(GMT+0). So we have to merge them to mkt_data with a shift. This process takes most of the merging time.

At last, we delete some columns from the merged DataFrame that are not needed by our models . And named it as final_df. After the data munging process , we have a panel of clean and tidy final_df. Then we set a series of "thresh" to label our final_df to avoid market noise. More specifically, we label our final_df by this function:

$$\text{label} = \begin{array}{ll} 1 & \text{next\_r>thresh} \\ 0 & \text{-thresh<=next\_r<=thresh} \\ -1 & \text{next\_r<-thresh} \end{array}$$

where next_r represents the return of the next bar.So for different threshes, we may have different labels for the same bar. And the higher the thresh, the more bar is labelled as 0. Then we we divide final_df with labels into train data(the first 80%) and test data(the last

20%).

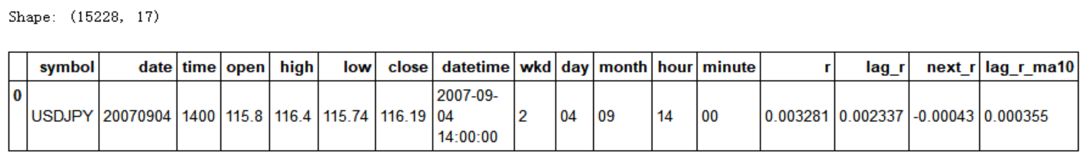
We train models on train data and got the result on test data. We suppose we buy if we predict a label of 1, sell at a predicted label of -1, and do nothing for a label of 0. Several main performance indicators we calculate are:

Not_0_Perc	the percent of data with a label of 1 or -1
Trades	Numbers of data with a label of 1 or -1
Accuracy	Percent of correctly predicting the label
MAXDD(%)	Max Drawdown
Cum_R(%)'	Cumulative return. We calculate with a simple interest mode. (each trade with a fixed lot)

You can see our results in the Discussion and Conclusion part.

Analyses and results

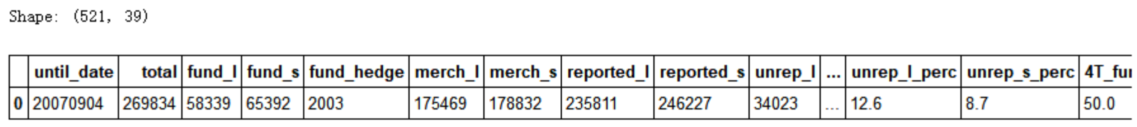
1. final_mktdata: After dealing with the USD/JPY market data, we get final_mktdata and draw a graph of the "close" column to verify the competence of data.



```
Shape: (15228, 17)
```

	symbol	date	time	open	high	low	close	datetime	wkd	day	month	hour	minute	r	lag_r	next_r	lag_r_ma10
0	USDJPY	20070904	1400	115.8	116.4	115.74	116.19	2007-09-04 14:00:00	2	04	09	14	00	0.003281	0.002337	-0.00043	0.000355

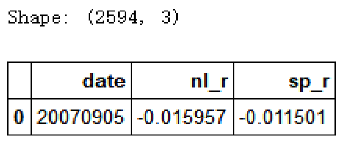
2. cftc_data: A DataFrame recording weekly CFTC Currency Holding Report from CME Group.



```
Shape: (521, 39)
```

	until_date	total	fund_l	fund_s	fund_hedge	merch_l	merch_s	reported_l	reported_s	unrep_l	...	unrep_l_perc	unrep_s_perc	4T_fu
0	20070904	269834	58339	65392	2003	175469	178832	235811	246227	34023	...	12.6	8.7	50.0

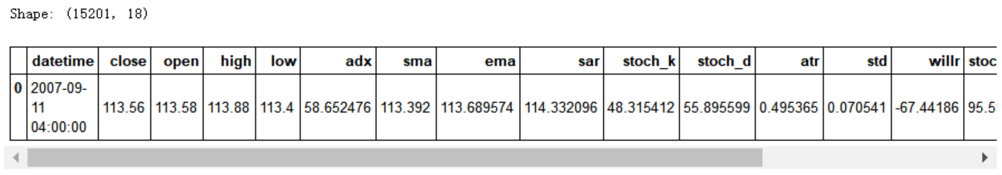
3. Idx_data: a merged DataFrame recording the daily returns of Nikkei225 and S&P 500.



```
Shape: (2594, 3)
```

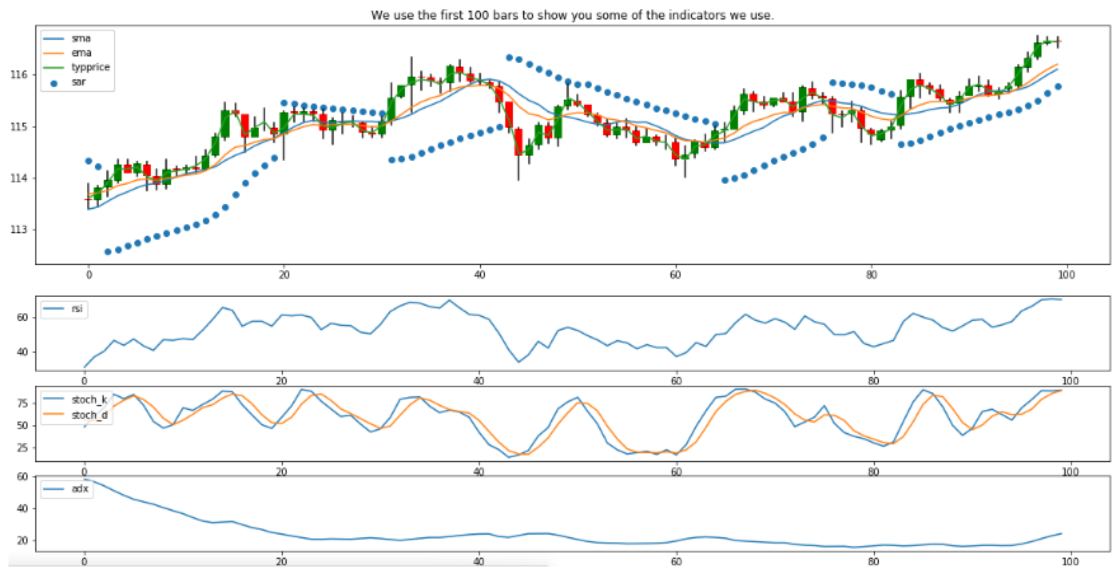
	date	nl_r	sp_r
0	20070905	-0.015957	-0.011501

4. indi_df: A bundle of indicators calculating from the final_mktdata. We also plot a series of graph showing some of them.

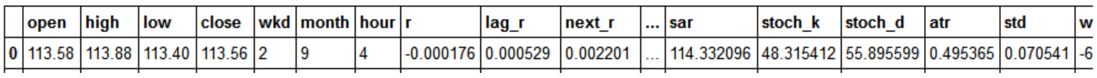


```
Shape:  (15201,  18)
```

	datetime	close	open	high	low	adx	sma	ema	sar	stoch_k	stoch_d	atr	std	willr	stoc
0	2007-09-11 04:00:00	113.56	113.58	113.88	113.4	58.652476	113.392	113.689574	114.332096	48.315412	55.895599	0.495365	0.070541	-67.44186	95.5



5. final_df: The DataFrame we get after merging cftc_data, final_mktdata, idx_data and indi_data together and delete some useless columns for our models, such as 'day', 'month', 'hour' and 'minute'.



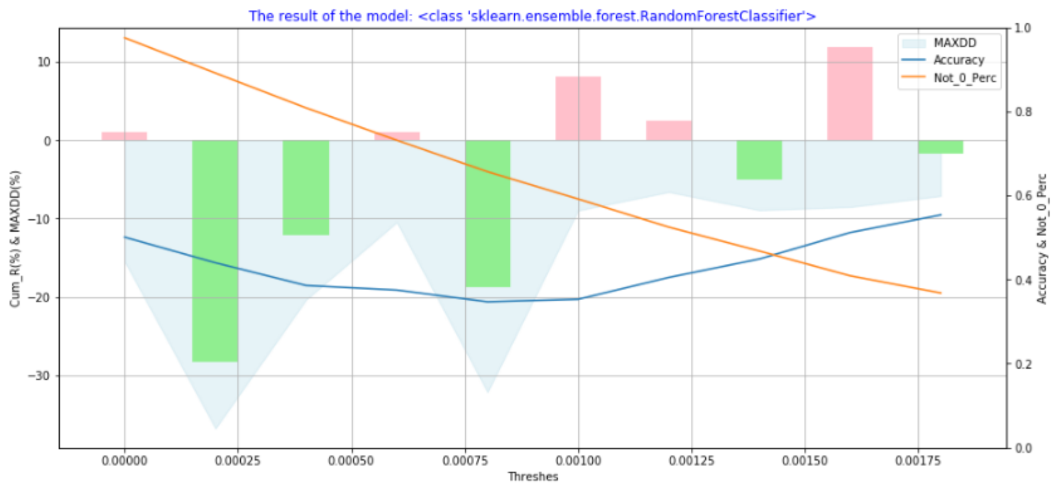
	open	high	low	close	wkd	month	hour	r	lag_r	next_r	...	sar	stoch_k	stoch_d	atr	std	w
0	113.58	113.88	113.40	113.56	2	9	4	-0.000176	0.000529	0.002201	...	114.332096	48.315412	55.895599	0.495365	0.070541	-6

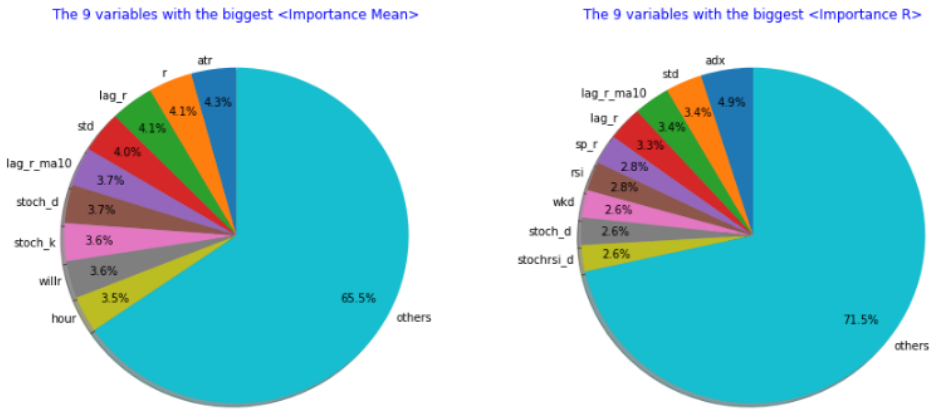
Discussion and conclusions

Below are the results of our 3 simple models, we mainly draw out 4 performance indicators of the results: MAXDD, Accuracy, Not_0_Perc and Cum_R(%).

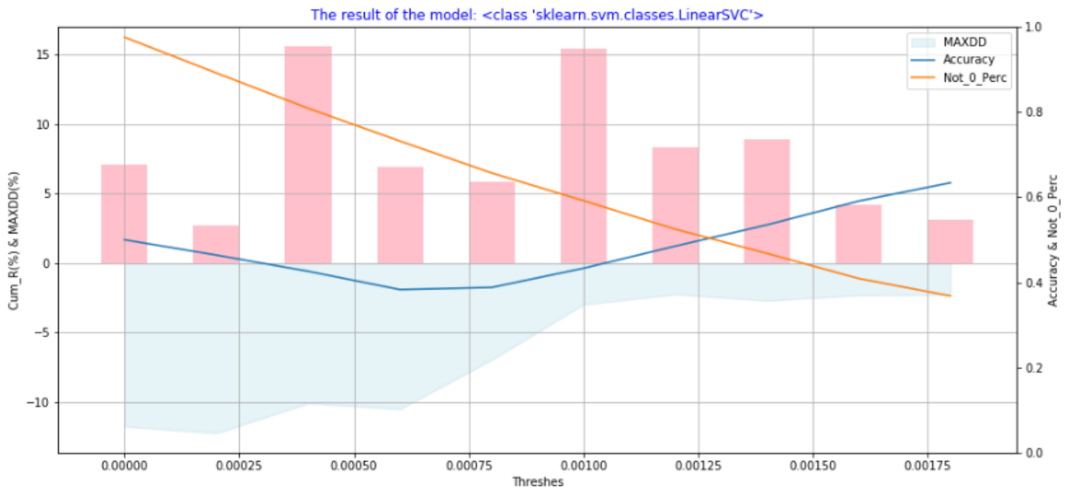
The bar chart stands for final return, corresponding to left axis. The blue line stands for win rate, the red line indicates the percent of labels whose values are not 0, and the light blue area indicates the max drawdown. They three are corresponding to the right axis.

RandomForestClassifier():(You may get a different result from us by this model, because it’s RandomForest!):

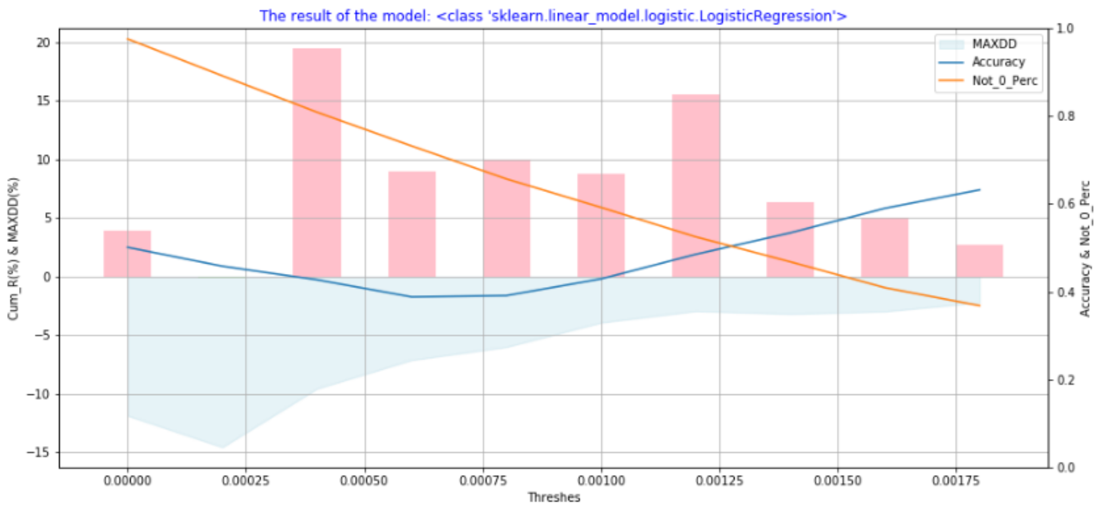




LinearSVC(penalty='l1',dual=False)



LogisticRegression(penalty='l1')



By the results above, we also achieve a positive Cum_R on 2 of the 3 models like former papers' authors, which also kind of proves the correctness of the Technical Theory.

Besides that, we also add a "thresh" mechanism in our research to avoid market noise. The result we got can prove that when ignoring more small variation of the market (market noise), The trades we made become fewer but we can achieve a higher win rate and a lower maximum drawdown. However, as shown in the Cum_R, we don't achieve a higher return, so we need to find a balance between trade numbers and win rate, which also means to find a proper thresh.

Our work still has some limitations and problems, For example, we don't get a positive result on RandomForest, and by looking at the pie chart we can find many of the important indicators are just the same of those used in the paper (r, lag_r, lag_r_10, willr, hour). So why we don't have a similar result on this model and what if we delete the number of features we input using techniques such as PCA? We will continue researching these problems in the future.

References

Barbosa, R. P., & Belo, O. (2008). Autonomous FOREX trading agents. In *Proceedings of the 8th industrial conference on Advances in Data Mining: Medical Applications, E-Commerce, Marketing, and Theoretical Aspects. ICDM'08,* 389-403, Springer-Verlag.

Barbosa, R. P., & Belo, O. (2010). The agent-based hedge fund. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 449-452. IEEE.

Gerlein, E. A., Mcginnity, M., Belatreche, A. & Coleman, S. (2016). Evaluating Machine Learning Classification for Financial Trading: An Empirical Research. *Expert Systems With Applications, 54*, 193-207..

Kumar, D., Meghwani, S. S. & Thankur, M. (2016). Proximal Support Vector Machine-based Hybrid Prediction Models for Trend Forecasting in Financial Markets. *Journal of Computational Science, 17*, 1-13.

Maggini, M., Giles, C. I. & Horne, B. (1997). Financial Time Series Forecasting Using K-Nearest Neighbors Classification. In *Proceedings of the 1st Nonlinear Financial Forecasting Conf. (INFFC97)*, 169-181.

Tenti, P. (1996). Forecasting Foreign Exchange Rates Using Recurrent Neural Networks. *Applied Artificial Intelligence, 10*, 567-581.

Yang, H. & Lin, H. (2017). Applying the Hybrid Model of EMD, PSR, and ELM to Exchange Rates Forecasting. *Computational Economics, 49*(1), 99-116.