

Matteo NOTTARIS and Nicolas BOINAY

Tutor : Maria ZULUAGA

RSI indicator and Machine Learning



Summary

1	Project origin	2
2	Project description	2
3	Extraction of the data sets	3
4	Creation of a new data set	3
5	Use of a Neural Network	4
5.1	Initial thoughts	4
5.2	Scikit-Learn	5
6	What should be improved	6
7	Sharing of the work	6
8	Annexe	6
8.1	Annexe 1	6

1 Project origin

We wanted to invest ourself in a project which is linked with finance because we both want to orient our professional career in this domain. In order to do so we contacted a FOREX trader working for Crédit Agricole Indosuez Wealth Management, with who one of us worked before during an internship. Our main concern was to find a project's subject which was usable for traders once finished and could have a real application in the trading rooms.

We knew that there were already several projects on internet about trying to determine the future evolution of stocks using Machine Learning algorithms. Nevertheless, we wanted for our first ever project in the subject to try something knew, ambitious and above all interesting for traders. Therefore, our contact told us to work on financial indicators which are helping traders to interpret trends on the market.

At the beginning the idea was to right algorithms capable of determining for a given representation of the market the most relevant indicator. Therefore, traders would just need to see which indicator is returned by algorithms and use it in order to interpret a situation. However, we soon understood that it was maybe a bit too complexe for beginners and to develop it in the time we have before the submitting of the project. Therefore, we slightly change the project to have it as it is today.

2 Project description

As explained above the project definition slightly changed from our original thoughts. We focus on one common indicator and used by traders which is called RSI for Relative Strength Index. RSI is an indicator which gives an indication on the trend of an asset, using two different parameters, which we will call the upper and the lower one. If the RSI signal is over the upper parameter it means that the asset we are interested in (EURUSD in our case) is overbought and therefore there will probably have a reversal in the asset trend. Hence the RSI tells us to sell the asset will the price is high and rebuy it after the reversal, this would allow us to make a profit. The lower parameter is working exactly the same, because when the RSI signal is under it the asset is over sold and the asset trend might go up strongly. As a result the RSI tells us to buy while it is low and sold after the reversal.

However, the main problem with this indicator is that it is always used with two parameters like $RSI(80,20)$ or $RSI(70,30)$. We thought that depending on the market situation those two parameters could adapted in order for the RSI

to be more precised.

There were three main steps in order to lead this project to its end. The first was to determine what could be a representative picture of a market and then extract all the data sets corresponding. The second one was to find which couple of parameters suits the best for a given situation. With this last step we were able to develop a new data set which we used in our last step of the project which is the training of a Neural Network and find which regression function suits the best our problem.

3 Extraction of the data sets

For the sake of our project we needed to extract a lot of different data sets (Annexe 1) in order to perfectly determine what represents a market situation. In order to do so we contacted our trader colleague who was far more capable than us to do this list of important representant of the market.

We used two different websites to extract the data sets.

The first one was AlphaVantage in order to extract the RSI signal for our asset the dual currency EUR/USD. The second website we used, in order to download the data from all the other stocks, was Yahoo Finances.

The hard part in it wasn't to download the data sets from the websites and load them in excel files, but it was to succeed on making sure that all of them were over the same dates set. Indeed, we needed for one date to be sure that we had the value of each indicators for this given date. We succeeded in doing so, using the manipulation of Python's dictionnaires and arrays. At the end of this step we had more than 2900 common and different dates for all the data set. We could now use this data set as we wanted.

4 Creation of a new data set

The second step of the project was to generate, using the data sets we downloaded and shaped in the first step, a new data set which would contain for different couple of parameters the most corresponding market situation.

In order to do so we had to determine how to estimate the best situation according to a couple of parameters. We decided, with the trader, that the best way was to calculate a performance using the RSI signal and the two chosen parameters. In fact, we are listening to the instructions of the $RSI(p_1, p_2)$ and we consider that when he tells us to sell we do so and we calculate a performance, in other words we calculate our benefits in percentage, compared to the situation

where we would have not listened to it.

At the end we look for this given couple of parameters which of the 2900 dates has the highest performance. We did that for every parameter from RSI(90,10) to RSI(51,49) incremented by 0.5 each time. For these 80 different couples we attribute 10 different dates which correspond to the 10 best performances.

This new data set will be used in order to train our Neural Network in the next step of the project.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Best date par1	par2	CAC40	DAX	DOLLARIN	DOWJONES	EURUSD	FTSE100	GOLD	HSCJ	NASDAQ	NIKKEI	OIL	SP500	STOX50DE	STOX6000	TAUX10US	TAUX5US	USDCNY	USDJPY	VIX		
2015-03-13	12	88	4997.94	11845.9	99.19	17889.05	1.062398	676.11	1158.2	11589.19	4885.54	19119.58	46.93	2064.56	3649.46	395.79	2.107	1.589	6.252	121.302	15.47	
2014-03-11	13.5	86.5	4900.37	11531.55	98.65	17662.94	1.070939	6702.8	1162	11468.02	4866.94	18604.87	48.77	2044.69	3574.98	390.57	2.151	1.638	6.252	121.06	16.44	
2014-09-08	14	86	4484.4	9767.9	83.8	17131.71	1.295203	6855.1	1268.4	11414.43	4579.06	15718.13	93.49	2007.17	3276.2	346.63	2.437	1.673	6.1305	105.149	12.64	
2015-01-20	14	86	4406.96	10278.55	92.66	17516.96	1.159703	6585.5	1278.9	11611.51	4655.84	17071.65	48.69	2020.76	3229.34	353.74	1.827	1.301	6.208	117.74	20.07	
2010-10-06	14.5	85.5	3750.87	6246.81	77.82	10936.79	1.383604	5635.8	1391.1	12901.51	2395.16	9586.47	82.6	1159.81	2768.47	261.82	2.405	1.138	6.68	83.21	21.82	
2014-09-10	15	85	4442.69	9672.45	84.14	17016.05	1.293996	6829	1256.7	11301.71	4554.15	15688.55	92.79	1988.41	3241.34	344.5	2.529	1.78	6.1265	106.127	13.36	
2015-01-23	15	85	4588.49	10503.33	94.21	17812.5	1.134649	6796.6	1299.8	12223.44	4748.19	17520.63	46.63	2062.98	3345.45	365.5	1.82	1.328	6.199	118.71	16.79	
2008-10-27	15.5	84.5	3035.37	4143.45	86.32	8375.92	1.263807	3883.4	742.8	5699.02	1538.12	7568.36	64.78	874.28	2293.04	195.042	3.652	2.549	6.8276	93.68	79.13	
2013-09-14	15.5	84.5	3573.3	7397.98	79.24	13540.4	1.299376	5819.9	1771	9733.41	3166.24	9097.92	98.04	1460.07	2567.96	273.9	1.826	0.691	6.3301	77.68	13.82	
2014-09-30	15.5	84.5	4375.38	9446.81	85.61	17070.45	1.2691	6646.6	1214.8	10409.68	4512.64	16252.72	94.34	1978.21	3199.44	341.9	2.504	1.778	6.14	109.412	15.49	
2015-01-16	15.5	84.5	4303.55	9985.51	92.1	17320	1.1639	6498.8	1256.1	12102.23	4566.38	16812.96	46.35	1992.25	3147.53	349.31	1.715	1.176	6.1775	116.112	22.8	
2013-08-31	16	84	3028.93	6297.68	83.1	12414.41	1.237471	5297.3	1562.7	9525.39	2837.37	8499.68	87.55	1313.09	2120.86	240.71	1.617	0.694	6.3507	78.966	23.83	
2015-03-10	16	84	4930.06	11555.97	97.78	17889.56	1.083576	6876.5	1166.4	11642.61	4899.51	18891.03	50.08	2076.14	3605.81	393.31	2.158	1.62	6.255	121.427	16.47	
2010-10-05	16.5	83.5	3648.96	6132.12	78.5	10752.78	1.367503	5556	1312.9	12541.38	2368.52	9337.43	81.37	1140.68	2699.63	257.35	2.463	1.213	6.68	83.502	22.52	
2007-09-28	17	83	5728.95	7850.94	78.29	13912.94	1.415208	6486.4	733.7	16750.85	2709.65	16903.64	83.25	1531.24	4389.08	377.77	4.545	4.194	7.5044	115.54	17.23	
2017-08-02	17	83	5122.38	12269.9	93.06	23004.36	1.18093	7423.7	1268.5	11073.15	6391.1	20057.07	48.8	2480.38	3478.21	380.01	2.271	1.816	6.7144	110.458	10.08	
2007-11-07	17.5	82.5	5737.63	7839.06	75.57	13646.72	1.456092	6474.9	830	18975.59	2798.45	16325.18	96.83	1515.46	4416.99	379.02	4.318	3.91	7.4457	114.7	23.15	
2012-05-30	17.5	82.5	3051.25	6360.79	82.5	12579.02	1.247816	5391.1	1553.9	9776.62	2847.27	8637.95	90.86	1331.25	2147.3	243.35	1.688	0.741	6.3415	79.517	22.68	
2014-09-26	17.5	82.5	4361.37	9500.55	85.14	16948.62	1.275705	6639.7	1214.1	10547.21	4476.48	16087.95	92.47	1966.22	3205.34	341.46	2.502	1.752	6.126	108.62	15.77	

Figure 1: New, neural network's training data set

5 Use of a Neural Network

5.1 Initial thoughts

In this last step of the project we are trying to train a neural network. This neural network takes as inputs a vector of shape (19,1), corresponding to the market indicators to define a situation (see Annexe 1). As output the neural network sends back a vector of shape (1,1) corresponding to the first optimal parameter which should be used considering the situation in input. The second parameter can be easily determined :

$$par_1 + par_2 = 100$$

In order to do so, we took an already implemented neural network that we used during on of our previous labs in class, and tried to adapt it to our situation.

The first main difference between the neural network problem in the lab session and our project is that the problem we are tackling is a regression problem while the other one was a classification exercice. We understood that when we found odd results especially between 0 and 1 due to the sigmoid activation function used.

Now that we understood the fact that our problem is a regression one we have changed the activation function on the last layer of our network by a regression function. However, we were struggling with this neural network, and our tutor wisely advised us to use a python library like scikit-learn to implement it, because this new network might be more robust.

5.2 Scikit-Learn

The use of scikit-learn was very helpful because we just had to focus on the choice of parameters and not anymore on the concrete development of the neural network.

Our subject was a regression problem, therefore we compared four different linear functions : *tanh*, *identity*, *logistic* and *relu*. For each of those functions we used a function of scikit-learn called *GridSearchCV*, in order to find the optimal parameters.

However, even with the help of those tools the results were very disappointing. The main issue was that our output vector was composed of only one value, which we understood as a sort of average of all supposed output value. We understood that this issue was generated because our output were very close no matter how different were the inputs. Indeed we understood that the w_i must be close to 0. Therefore, in the prediction formula, the neural network was only returning the constant w_0 in : $y_{pred} = w_0 + \sum w_i * x_i$.

In order to tackle this issue, we decided to make sure to spread the y_{train} values before training the neural network. We used different functions to spread the data, but the most efficient one was to multiply by 1000 and then divide by 1000 at the end of the neural network. Nevertheless, the results were not as expected. The Mean-Squared-Error (MSE) didn't decrease by a lot but for two functions (*relu* and *identity*), the outputs vectors weren't composed of a single value anymore. The neural network was actually working, and was trying for each different input to predict a coherent output.

We were not yet satisfied by the results, and we thought that we could reduce our dataset and only keep the most efficient market situation for each couple of parameter. As a result we trained our neural network with around 70 lines of data instead of 700, and this had a major effect on the result. First of all we saw the MSE decrease for every functions. Secondly, now the *tanh* function was, as *relu* and *identity*, working as expected and returning different outputs for various inputs.

Overall, we can see in the followed table that the result are far better when the dataset was smaller, which probably shedlight a problem of overfitting in the first case. Moreover, the best function in order to tackle our problem is *tanh* because it shows the smallest MSE.

MSE	logistic	tanh	identity	relu
Data (700 lines)	93	85	87	75
Data (70 lines)	10	13	35	30

Figure 2: MSE for different functions and datasets

6 What should be improved

This project was very interesting, especially for beginners like us in machine learning. Obviously, the results we obtained could be far better and there are two main reasons for it.

The first one is that the data we are looking to could be wider. Indeed, we could try to find better situations for the different couples of parameters. For example, we could try to look for datas from previous years. We could even look for the market situation every hours, instead of everyday, to increase the number of potential situation to optimally fit the given parameters.

The second reason is that maybe neural networks from scikit-learn are not powerful or robust enough. As a result by using stronger neural networks we could obtained far better results for some of the functions like *tanh* or *relu*.

Overall, this study is primary and there is still a lot of work to improve it.

7 Sharing of the work

For the beginning of the project we spread the work pretty evenly. Nicolas did most of the extraction of data from the two websites (AlphaVantage and Yahoo Finance). Then Matteo did the shaping of the data and the creation of the new data set.

Concerning the last part, the use of neural network, we shared the work in two and each of us was trying some set of parameters different application functions to find the most suitable situation to our problem.

8 Annexe

8.1 Annexe 1

CAC40	DAX	DOLLAR INDEX	DOW JONES	EURUSD
NIKKEI	OIL	SP500	STOXX 50 E	STOXX 600
FTSE100	GOLD	HSCEI	NASDAQ	VIX
US 10 year treasury rate	US 5 year treasury rate	USDCNY	USDJPY	

Figure 3: All market indicators used in our case