Associative Analysis on ForexData

January 28, 2018

The aim of the project is to apply associative analysis to Forex data (market prices of currency pairs) in order to find sets of currency pairs that correlate. We further try to use these sets to formulate rules that predict when a certain currency will change its course.

1 Preprocessing

We use a dataset consisting of the closing prices of every minute of 9 currency pairs in the time interval 2002-2017 : `http://www.histdata.com/download-free-forex-data/?/excel/1-minute-bar`.

We first need to binarize the data to apply associative analysis. Since we want to find sets of rising/falling currency pairs, the binary indicator shall be if the given currency rises/falls in a certain time interval or not. Certainly, we lose a lot of information here (how much the currency exactly rises/falls), but we still hope to achieve significant results here.

We use two different discretization intervals : 1 minute and 4 hours. We want to try to use the algorithm to make short term predictions (useful for high frequency trading) and long term predictions (long term trends).

More precisely the binarization is in the following manner : at each record, we take the average price between the "high" and "low" value. We then compare this value to the next average value (1 minute discretization) or the average value of the record 4 hours later (4 hour discretization).

As for missing data, to apply the FPGrowth algorithm, it constitutes no problem to replace missing values with zeros. If a value is not available, we

simply say it is not rising/falling. For the evaluation data, this would be a big mistake, since this procedure could make us say a prediction is wrong when in reality we simply don't know because the data is missing.

Since there is no time component in FPGrowth, we can simply discard the date attribute. This is the second big generalization that makes us lose information.

2 Applying the algorithm

We then apply the FPGrowth algorithm to the binarized and 0-filled data to extract associative rules. We use the confidence measure for pruning. For both time intervals (1 minute / 4 hours), we want to formulate 10 rules for rising pairs and 10 rules for falling pairs.

We train on the 2002-2016 data for the 1 minute time interval and the 2002-2015 data for the 4 hour interval. We hold out a bigger datatset to test for the higher timer interval, because in this case we have a lot less data to test on. We then extract respectively the 10 rules with the highest confidence.

The exact rules produced with respective confidence/support values can be found in the rules folder.

3 Parameters

The parameters used in Weka are : $delta = 0,03$, $lowerBoundMinSupport = 0,08$, $metricType = confidence$, $minMetric = 0.8$, $numRulesToFind = 10$.

4 Evaluation Strategy

We say a rule matches a record if the precedent is true and the antecedent is false : eg. we have found a rule

$$EUR/CAD\ rising\ \wedge\ EUR/GBP\ rising \implies\ EUR/JPY\ rising$$

and see a record where both EUR/CAD and EUR/GBP are rising but EUR/JPY is falling, the rule would match the record and the prediction would be that EUR/JPY is likely to rise soon too.

We now want to scan through the hold out test set (2017 for the 1 minute discretization and 2016-2017 for the 4 hour discretization) and look for rules matching the records. If a rule applies, we check if the price actually rose/fell in the next 3 minutes for the 1 minute discretization or in the next 4 hours for the 4 hour discretization. If it did, we say the rule did a correct prediction.

We then compute the precision $P = \frac{TP}{TP+FP}$ to evaluate the correctness of a given set of rules.

5 Results

We obtain a the following results.

	TP	FP	Precision
1 minute discretization with rising pairs	11372	7189	61,3%
1 minute discretization with falling pairs	15713	13233	54.3%
4 hour discretization with falling pairs	564	392	59,0%
4 hour discretization with falling pairs	216	145	59,8%

6 Discussion

The results show a precision of about 60%. This value is not incredibly high, but in the case of the 1 minute discretization, it is absolutely significant over 18561 matches. It could be interesting to try an algorithm that accounts for the exact values (not binarized) and takes into account the time.

7 Analyzing live datastreams

An interesting application would be watching the live forex data, in order to give out alerts when a rule matches the live data. This idea is implemented in the file *livealerts.py*. The tracker fetches the live data from the internet, computes the average price of each pair every minute and compares it to the average price the next minute. It then binarizes the data just like before, and looks for rules that match the live data. If a rule matches the live data, the tracker gives out an alert.