Question:
Can we predict future price movements in financial markets by analyzing prior price and volume data via data science techniques?

More specifically, we will be looking at intraday data for NSE Nifty index futures, and trying to predict returns (in percentage terms, but also as binary / categorical variables) on several time frames. We have price and volume data, and we will construct features from these.

Raw Data:
The dataset is 1-minute open-high-low-close-volume data downloaded from Bloomberg. The data run from August 3, 2015 to February 1, 2016, inclusive.

The NSE Nifty futures trade from 11:45 Singapore time to 18:00 Singapore time. Each row of data represents 1 minute of trading, and there is typically a row of data reported after 18:00, which represents the daily settlement price.

The original fields downloaded from Bloomberg are Date, OPEN, HIGH, LOW, LAST_PRICE, NUMBER_TICKS, and VOLUME.
OPEN: the first traded price in the one-minute period
HIGH: the highest traded price in the one-minute period
LOW: the lowest traded price in the one-minute period
LAST_PRICE: the last traded price in the one-minute period
It is not clear what NUMBER_TICKS represents.
VOLUME: the number of contracts traded in the one-minute period. Do note that the exchange considers the minimum trade to be 75 lots, but Bloomberg reports this as 1. So, if the VOLUME field in our data shows 100, that means that in that minute, we would have seen trades totaling 7500 lots.

As this is a futures contract, there is a monthly roll. i.e., trading is typically most active in the nearest to expiry contract, and then when that expires, trading activity moves to the next contract. As an example, on August 3, 2015, trading was concentrated in the August 2015 contract “NZQ5” (“Q” is the code for August). Then on August 28, 2015, trading moved to the September 2015 contract “NZU5” (“U” is the code for September), because the last trading day of the August 2015 contract was August 27, 2015. Bloomberg takes care of adjusting the reported prices to make a continuous contract, by adjusting the earlier prices. So, the data would be consistent if we wanted to form longer term indicators (e.g., constructing moving averages over more than 1 futures contract cycle such as a 60-day moving average). But, we will focus on short-term (less than 1-day) predictions, and will using indicators that do not look further back than the current day, so the futures contract roll will not have an impact on our analysis even if it was incorrectly done.

Figure 1 - OHLC chart

In our dataset, the Nifty price has ranged between approximately 7000 – 9000, as shown in Figure 1. Note that the tick size of this contract is very small, at 0.05. Thus, the traded price typically fluctuates through many ticks, and is only very rarely unchanged within a one-minute bar.

	OPEN	HIGH	LOW	LAST_PRICE	NUMBER_TICKS	VOLUME
Count	45,609	45,609	45,609	45,609	45,609	45,609
Mean	7939.33	7941.39	7937.24	7939.31	43.12	904.53
Std	303.85	303.72	303.99	303.84	10.96	2008.37
Min	7237.50	7241.30	7234.55	7238.20	1	-
0.25	7773.75	7775.45	7771.65	7773.70	36	239
0.5	7888.60	7890.95	7886.00	7888.45	45	501
0.75	8136.05	8138.00	8134.40	8136.00	52	1,049
Max	8646.00	8647.60	8643.90	8647.00	64	319,857

Table 1 - Summary of Bloomberg data

Table 1 suggests that there may be outliers in the VOLUME data; it does suggest all other fields seem sensible.

quantile	VOLUME
0.75	1049
0.9	2023
0.95	2971.6
0.975	4132
0.99	6154.96
0.999	13213.26
0.9999	32182.85

Table 2 - VOLUME percentiles

Date	
8/25/2015 14:34	319857
8/25/2015 15:07	60292
9/29/2015 13:30	49179
8/24/2015 11:45	47967

8/21/2015 11:45	36381
8/17/2015 11:52	28895
9/9/2015 11:45	27515
8/4/2015 15:12	24862
8/28/2015 11:45	24682
8/4/2015 16:17	23441

Table 3 - Top 10 VOLUME values

Table 2 and Table 3 together suggest that there is an outlier in VOLUME, we'll drop the highest value of 319,857 as implausible and more likely to be a data error.

Nonetheless, VOLUME takes very high and very low values, such that it only appears normally distributed (with extra weight at volume 0) on a log-log plot:

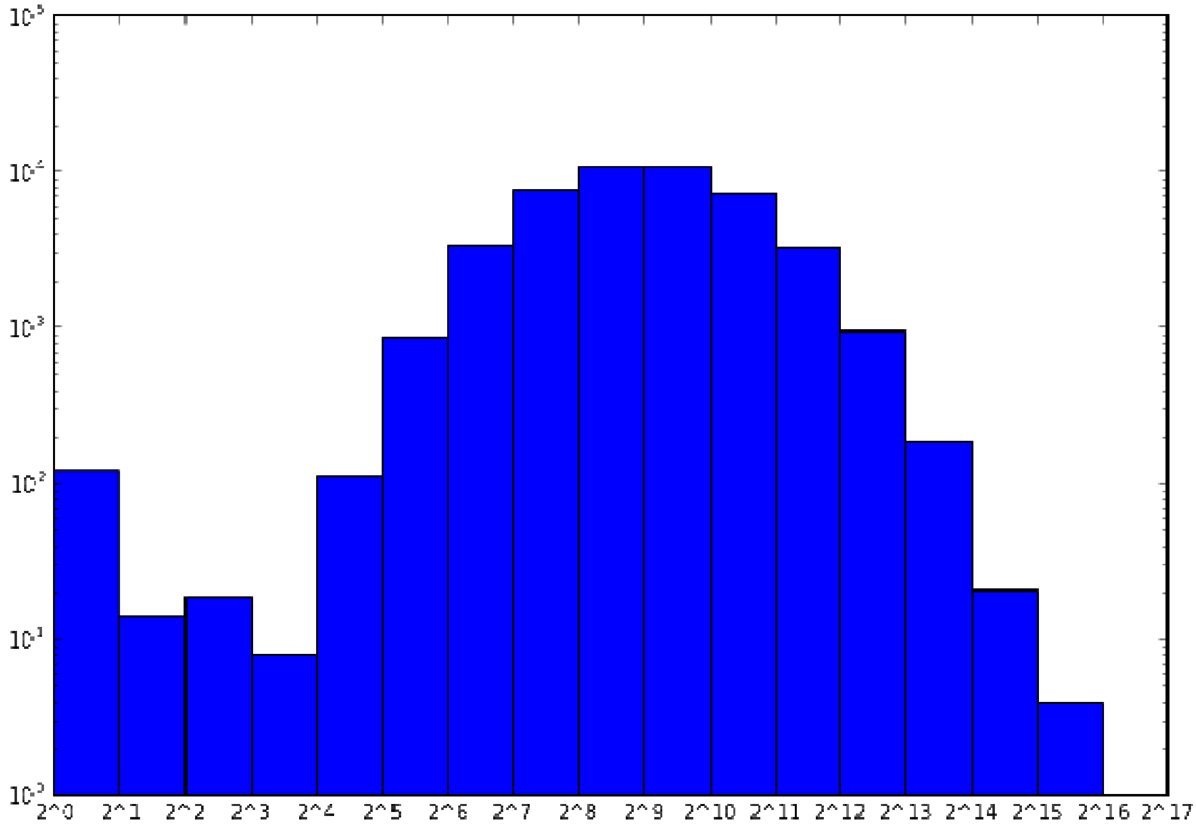

Figure 2- log-log histogram of VOLUME

We may want to consider scaling volume in some cases.

<u>Processing the data</u>

First, we remove the single row where volume was > 300k, as that is likely an outlier. We'll also remove all rows after 1800 (exclusive) as we do not want settlement data – these are not tradable times and these prices are not relevant. We do want the close row (at exactly 1800), because that closing price is important for computing returns on holding periods extending beyond the end of the trading day.

We'll add:

DayOfWeek: 0 for Monday, …, 4 for Friday

Hour: the hour (Singapore time), which ranges from 11 to 18 (but there are few rows at 11 as the market opens at 1145.
We set a time-series index to our dataframe, which will allow us to use 'resample' to create some features on a minute-by-minute basis.

Response variables
We'd like to explore predictability on multiple intraday time frames. We'll standardize this list to [1, 5, 10, 20, 40, 80]. i.e., as responses, we create the 1-minute ahead percentage return, but also the 5-minute, …, and 80-minute ahead percentage returns. This will be, for any given row, the (LAST_PRICE x-minutes ahead) / (current LAST_PRICE) – 1. In cases where the x-minute-ahead period extends beyond the end of the trading session, we'll use the LAST_PRICE at 1800 to compute the return. (This is accomplished by 'ffill' in the resample method.)

	1_min_return	5_min_return	10_min_return	20_min_return	40_min_return	80_min_return
count	45341	45338	45333	45324	45305	45265
mean	-0.0002%	-0.0011%	-0.0022%	-0.0042%	-0.0068%	-0.0153%
std	0.0412%	0.0898%	0.1241%	0.1730%	0.2438%	0.3411%
min	-0.5672%	-0.9208%	-1.1510%	-1.4103%	-1.7253%	-2.1843%
25%	-0.0208%	-0.0464%	-0.0633%	-0.0860%	-0.1179%	-0.1681%
50%	0.0000%	-0.0006%	-0.0006%	-0.0013%	-0.0020%	-0.0094%
75%	0.0208%	0.0453%	0.0611%	0.0815%	0.1092%	0.1459%
max	1.3354%	1.4400%	1.5734%	1.6781%	1.3665%	1.8271%

Table 4 - x-min return summary

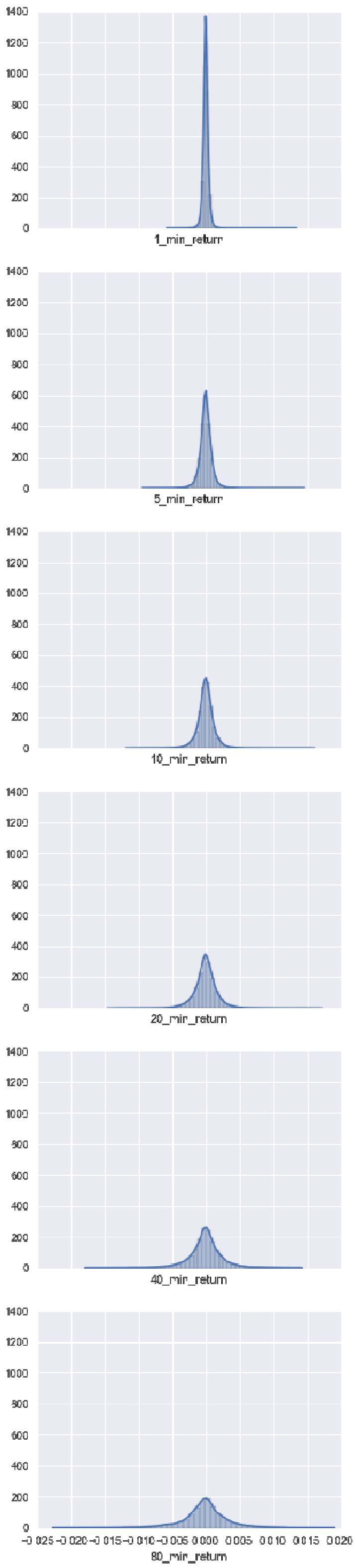

Figure 3 - x-min ahead returns

Table 4 and Figure 3 show the x-minute ahead returns, these appear sensible.

We look at skewness, kurtosis, and normality of the returns, with the following results:

```
We are using Fisher kurtosis, so a sample from a  normal distribution should
report a 0 kurtosis on this measure
For 1_min_return, skewness is 0.548331; kurtosis is 30.846228
For 5_min_return, skewness is 0.000061; kurtosis is 10.517373
For 10_min_return, skewness is -0.138187; kurtosis is 7.741777
For 20_min_return, skewness is -0.158083; kurtosis is 5.673287
For 40_min_return, skewness is -0.205899; kurtosis is 4.020208
For 80_min_return, skewness is -0.257549; kurtosis is 3.373104


For 1_min_return, p-value of rejecting hypothesis that sample is from a norma
l distribution is 0.000000.
For 5_min_return, p-value of rejecting hypothesis that sample is from a norma
l distribution is 0.000000.
For 10_min_return, p-value of rejecting hypothesis that sample is from a norm
al distribution is 0.000000.
For 20_min_return, p-value of rejecting hypothesis that sample is from a norm
al distribution is 0.000000.
For 40_min_return, p-value of rejecting hypothesis that sample is from a norm
al distribution is 0.000000.
For 80_min_return, p-value of rejecting hypothesis that sample is from a norm
al distribution is 0.000000.
```

Figure 4 - distributional tests on x-minute returns

i.e., the x-minute return data are heavy tailed and thus not normal.

We'll also create 'x_min_updown' columns which are coded +1 if the x-min return was positive, -1 if it was negative, and is blank if it was exactly 0. This ignores very few returns:

```
1_min_updown has 1124 (2.48%) missing values
5_min_updown has 538 (1.19%) missing values
10_min_updown has 453 (1.00%) missing values
20_min_updown has 379 (0.84%) missing values
40_min_updown has 356 (0.79%) missing values
80_min_updown has 351 (0.78%) missing values
```

Figure 5 - missing values in x-minute binary variables

	1_min_updown	5_min_updown	10_min_updown	20_min_updown	40_min_updown	80_min_updown
count	44344	44945	45031	45104	45131	45289
mean	0.500113	0.494872	0.495858	0.493814	0.492455	0.482523
std	0.500006	0.499979	0.499988	0.499967	0.499949	0.4997
min	0	0	0	0	0	0
25%	0	0	0	0	0	0

50%	1	0	0	0	0	0
75%	1	1	1	1	1	1
max	1	1	1	1	1	1

Table 5 - x-minute binary variables summary

Table 5 shows no obvious issues with the binary variables.

To get a sense of the baseline, we look at the percent of returns that are up:

```
For 1 min returns: +ve: 22176; -ve: 22167; i.e., 50.01% up
For 5 min returns: +ve: 22235; -ve: 22694; i.e., 49.49% up
For 10 min returns: +ve: 22319; -ve: 22695; i.e., 49.58% up
For 20 min returns: +ve: 22265; -ve: 22823; i.e., 49.38% up
For 40 min returns: +ve: 22220; -ve: 22891; i.e., 49.26% up
For 80 min returns: +ve: 21765; -ve: 23351; i.e., 48.24% up
```

Figure 6 - percent positive binary returns

So, the returns are almost 50% up

We also make categorical variables for the return, where we code the data as ‘2’ if it’s a significantly positive return, ‘1’ if it’s a small return, and ‘0’ if a significantly negative return,. The cut-offs for these classifications are based on percentiles of absolute values, such that approximately 1/3 of the data are ‘2’, 1/3 are ‘1’, and 1/3 are ‘0’. Figure 7 shows the distribution of the categorical variables, there do not seem to be any obvious issues.

```
For 1 min returns: +1: 15169; 0: 15114; -1: 15057
For 5 min returns: +1: 14994; 0: 15113; -1: 15230
For 10 min returns: +1: 14943; 0: 15111; -1: 15278
For 20 min returns: +1: 14876; 0: 15108; -1: 15339
For 40 min returns: +1: 14794; 0: 15102; -1: 15408
For 80 min returns: +1: 14302; 0: 15088; -1: 15874
```

Figure 7 - count of categorical return variables

Feature variables

Our first features are the x-minute prior returns, and their binary versions. i.e., for a given minute, we compare the current LAST_PRICE to the LAST_PRICE x minutes ago, and compute the percent change. That is x_min_prior. The binary version is x_min_prior_updown, and simply is 1 if x_min_prior is positive, 0 if it was negative, and missing otherwise.

	1_min_prior	5_min_prior	10_min_prior	20_min_prior	40_min_prior	80_min_prior
count	45466	45462	45457	45447	45427	45387
mean	-0.000002	-0.000009	-0.000018	-0.000036	-0.000059	-0.000083
std	0.000411	0.000904	0.001257	0.001757	0.00247	0.00341
min	-0.005705	-0.009293	-0.011644	-0.014304	-0.017556	-0.022331
0.25	-0.000207	-0.000466	-0.000638	-0.000873	-0.001207	-0.001685
0.5	0	0	0	-	-	-

				0.000006	0.000007	0.000076
0.75	0.000206	0.000459	0.000625	0.000848	0.001168	0.00161
max	0.013178	0.014195	0.01549	0.016504	0.013481	0.017943

Table 6 - x-minute prior return summary

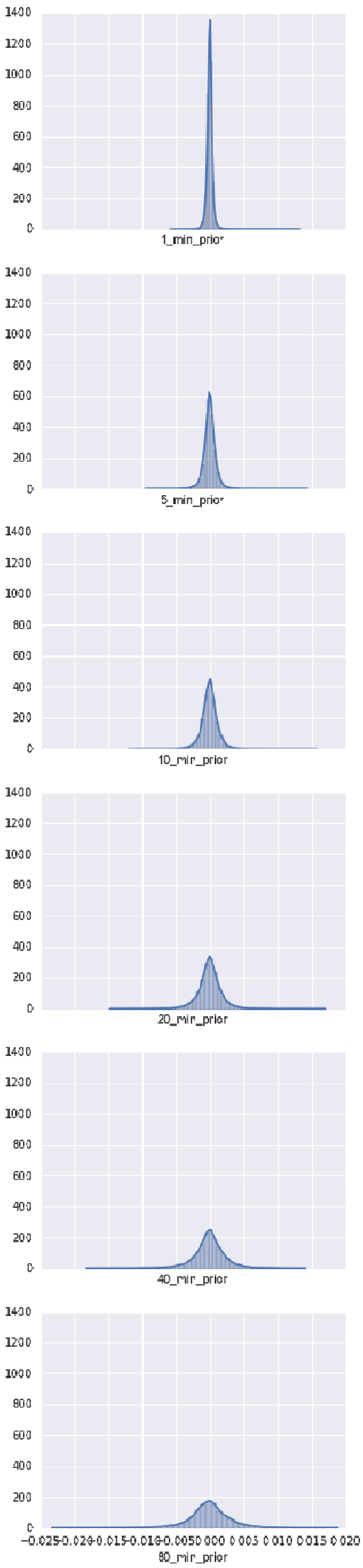

Figure 8 - x-minute prior distribution

Table 6 and Figure 8 show the prior returns, there are no red flags here.

We next make x-minute moving average columns. So, this is the average LAST_PRICE over the last x minutes, including the current minute (because, we assume you would put on a trade at the LAST_PRICE). When we are at times near the start of the day, and there are < x minutes available, we just average over all minutes since the open. So, if it is 1148, the 5-minute moving average would only average over the LAST_PRICE for 1145, 1146, 1147, and 1148. Table 7 summarizes these columns – there seems to be no cause for concern. Figure 9 shows a sample of these series.

	1_min_ma	5_min_ma	10_min_ma	20_min_ma	40_min_ma	80_min_ma
count	45467	45467	45467	45467	45467	45467
mean	7939.34	7939.38	7939.42	7939.52	7939.67	7939.86
std	303.86	303.85	303.85	303.84	303.83	303.85
min	7238.20	7241.47	7242.00	7247.19	7252.31	7264.39
25%	7773.78	7773.52	7773.52	7773.31	7773.77	7773.96
50%	7888.60	7888.72	7888.74	7888.64	7889.38	7890.20
75%	8136.00	8136.17	8135.83	8135.95	8136.53	8139.01
max	8647.00	8644.35	8643.02	8642.07	8641.23	8639.23

Table 7 - x-minute moving average summary

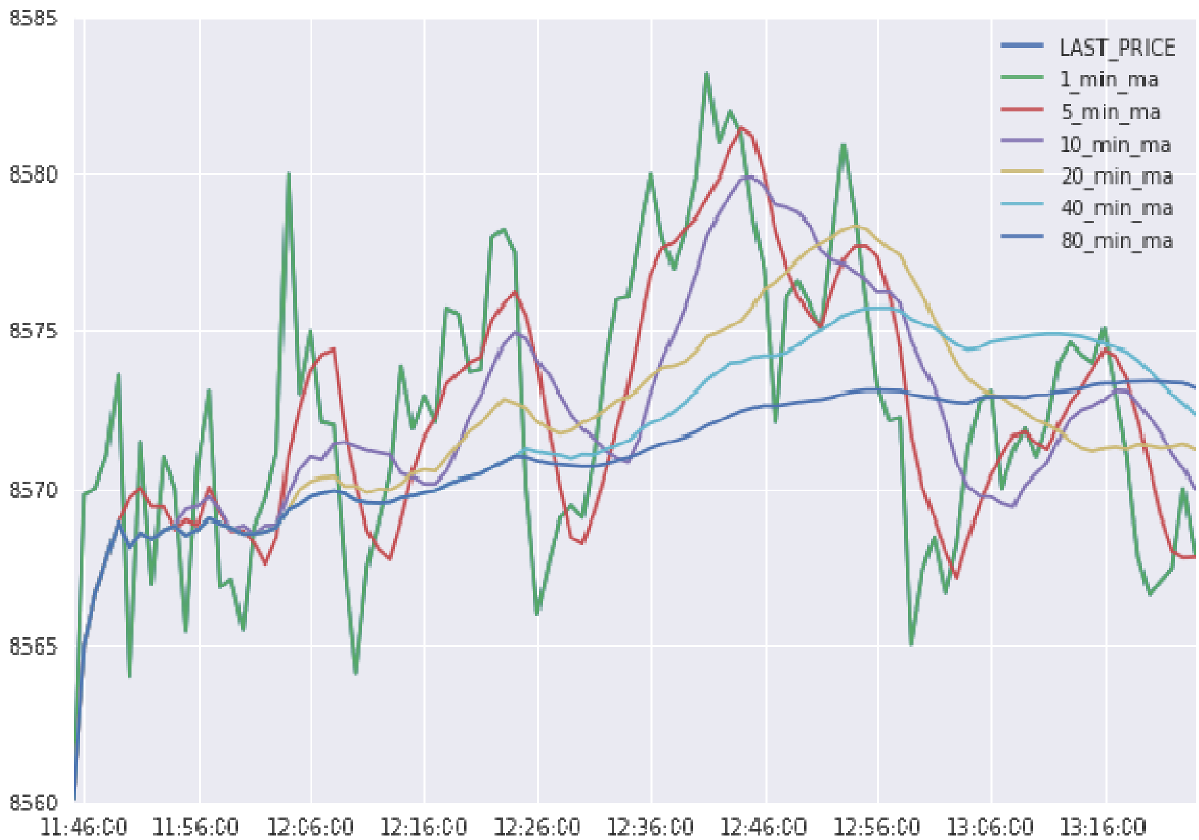

Figure 9 - example of moving averages

We also create x-minute difference from moving average columns, and binary (“updown”) versions of these.

	1_last_vs_ma	5_last_vs_ma	10_last_vs_ma	20_last_vs_ma	40_last_vs_ma	80_last_vs_ma
count	45,467	45,467	45,467	45,467	45,467	45,467
mean	0.00	-0.03	-0.08	-0.17	-0.33	-0.51
std	0.00	3.51	5.30	7.63	10.87	15.18
min	0.00	-51.77	-59.05	-74.44	-92.14	-122.13
25%	0.00	-1.75	-2.66	-3.74	-5.18	-7.11
50%	0.00	0.00	-0.03	-0.04	-0.05	-0.25
75%	0.00	1.73	2.61	3.59	4.89	6.64
max	0.00	82.23	94.20	104.04	112.01	109.29

Table 8 - price vs moving average summary

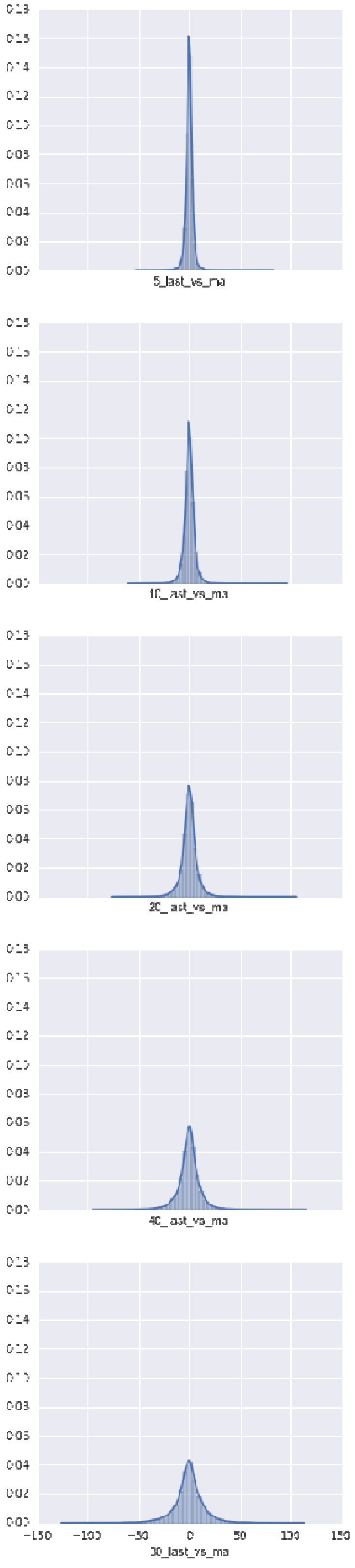

Figure 10 - last price vs moving average distributions

Table 8 summarizes the difference from moving average columns, Figure 10 plots their distributions. This looks fine; we shouldn't use the 1-minute columns (this is just the price minus itself).

We add some features for volume data. These are the rolling 1, 5, …, 80 minute average volume columns. We'll also add some columns where we average the log of (volume + 1) over these periods. We need to add 1 to make sure we get defined numbers in cases where volume = 0. We do these averages by taking rolling sum over rolling count. So, at the beginning of the day, when less than x minutes have elapsed, the x minute average will just be the average over the number of minutes elapsed since the start of the day.

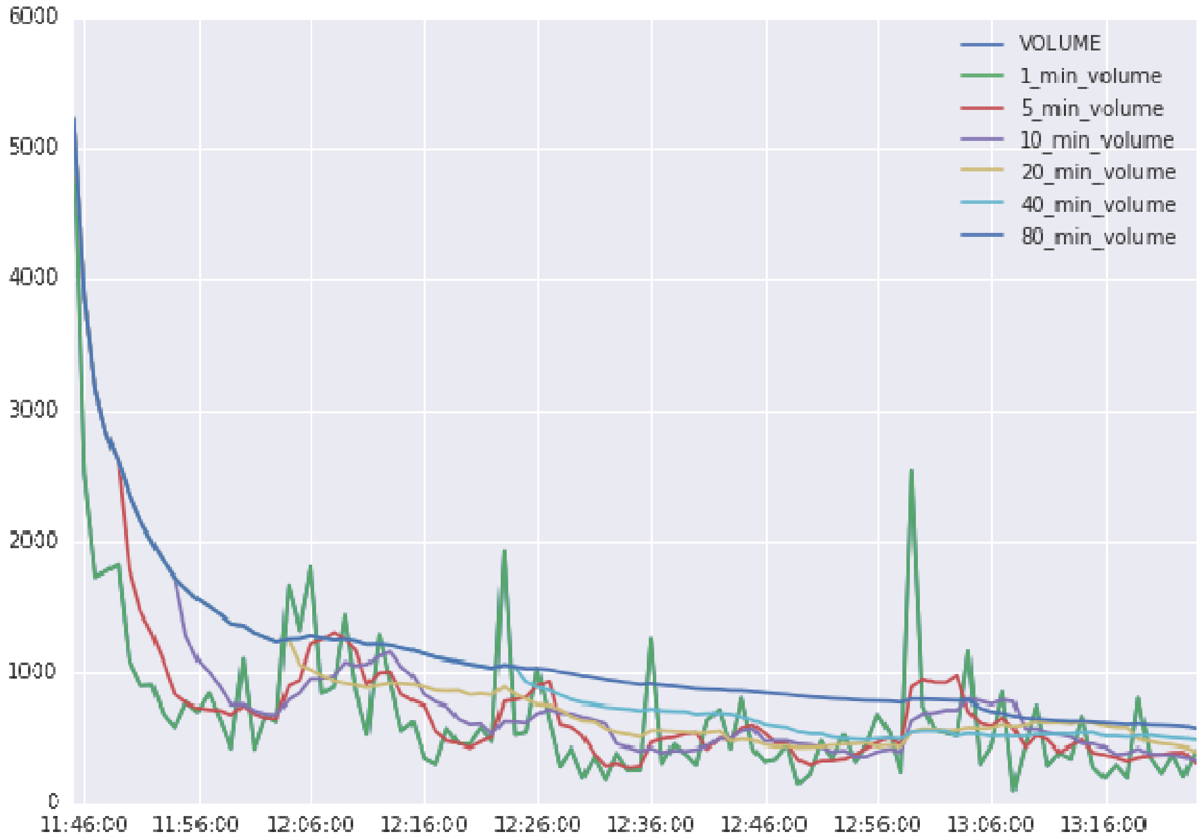

Figure 11 – volume

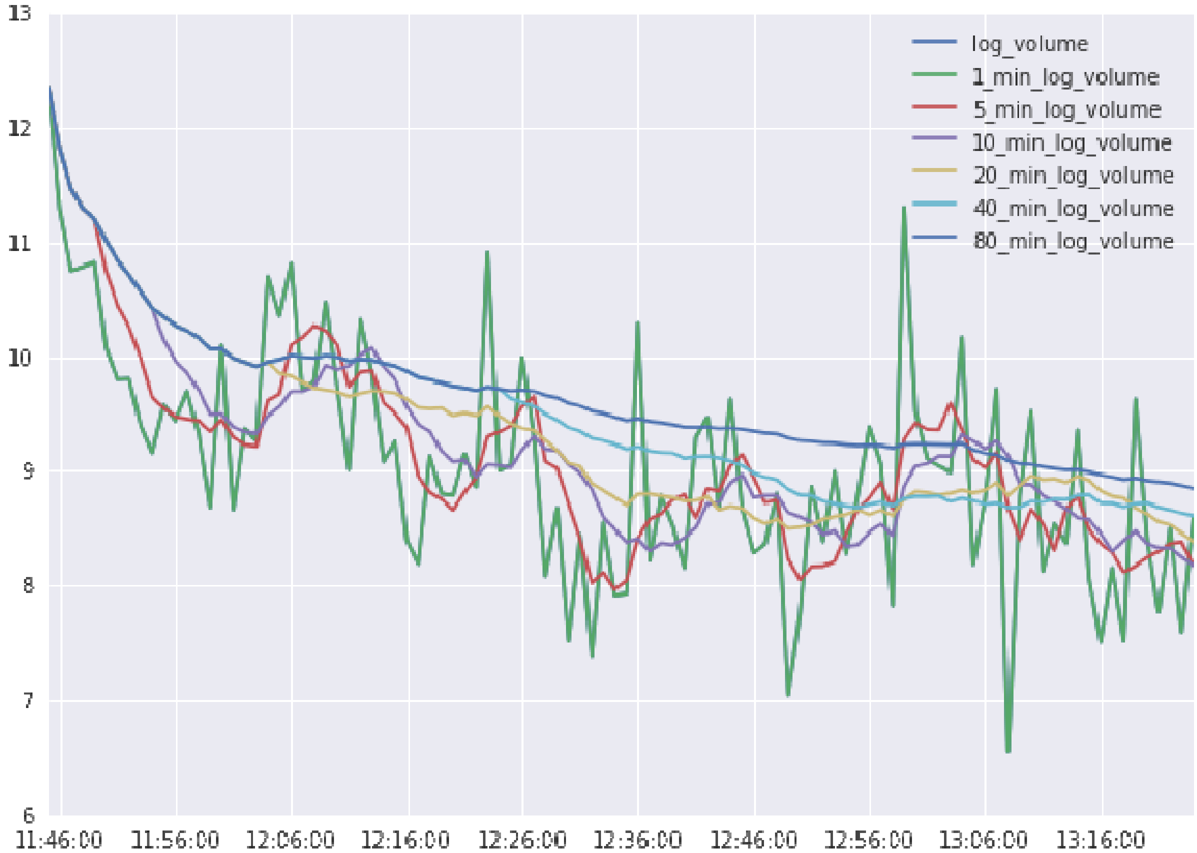

Figure 12 - log volume

Figure 11 and Figure 12 show examples of the volume and log volume columns. As volume varies quite wildly over each minute, log volume could be preferred as a feature as it has less extreme variations which may cause some analyses to be led astray.

The next set of columns represent buying up / selling down behavior. This is, loosely, a measure of how much volume went behind a particular directional move. It's possible that upmoves on high volume are more predictive of subsequent upmoves than upmoves on low volume. So, we'll mark a minute as 'up' if its LAST_PRICE is higher than the LAST_PRICE in the previous minute. For the opening minute of the day, we'll compare LAST_PRICE to OPEN. Then, we sign the volume in that minute as the raw 'busd' value. E.g., if LAST_PRICE at 1148 is 8889 and LAST_PRICE at 1149 is 8888, and VOLUME in the 1149 minute is 1000, then 'busd' is -1000. i.e., we simply return the volume in an up minute, and -1 times the volume in a down minute. Then we average over x minutes. We'll do the same for log volume too.

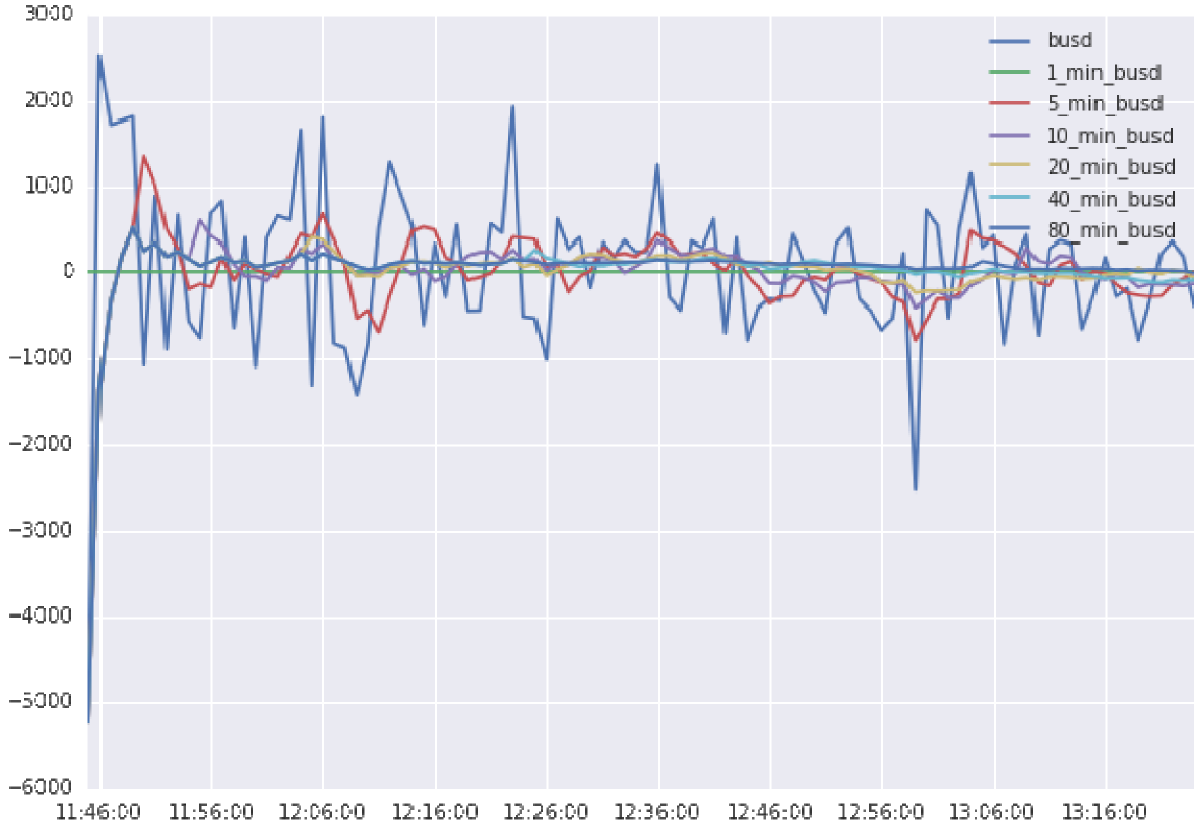

Figure 13 – buying up / selling down example

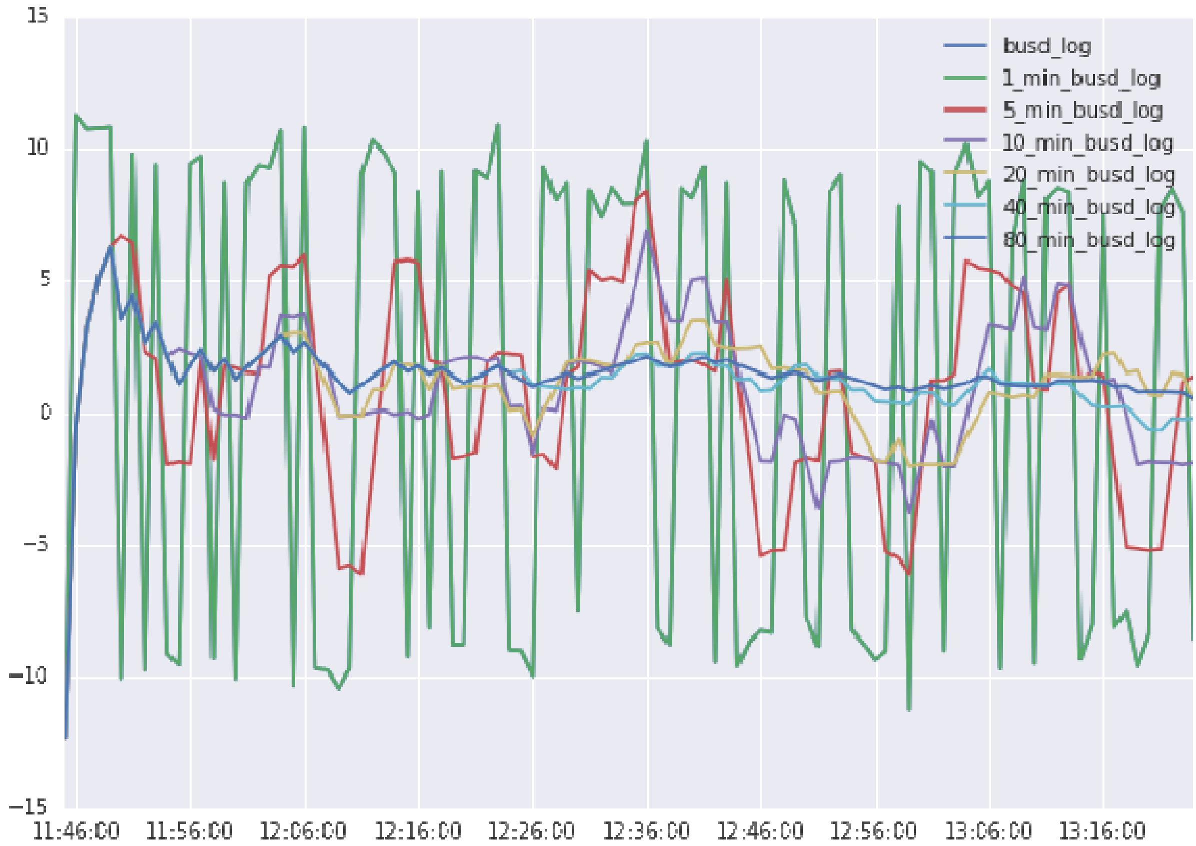

Figure 14 - log buying up / selling down example

Again, log values may be better for some analyses, as seen in Figure 13 and Figure 14.

Next we look at high-low ranges over x-minute prior periods. To construct this we take the difference between the maximum of the x-minute HIGH less the minimum of the x-minute LOW. This may be useful as it proxies the recent level of price volatility. The distributions of these features are shown in Figure 15, they're heavy right tailed.

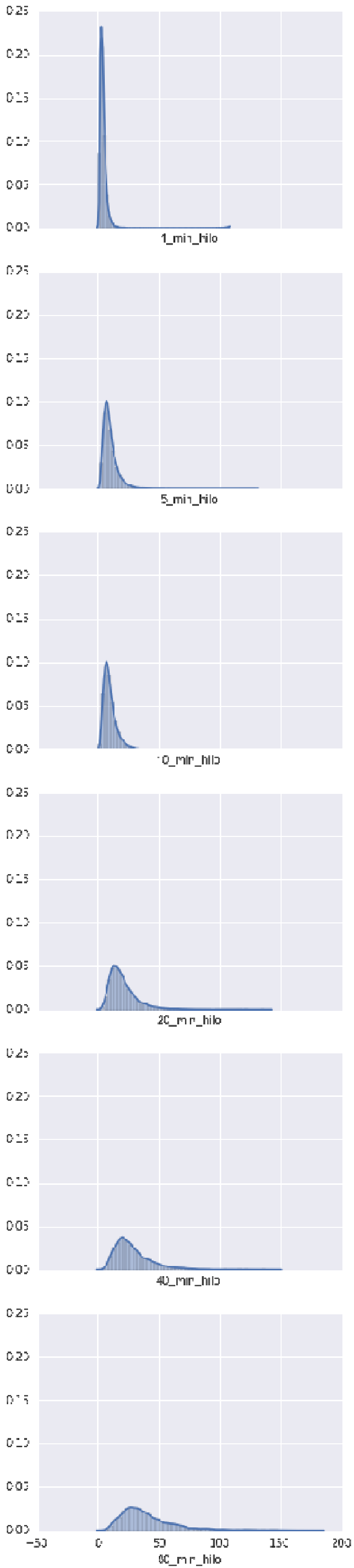

Figure 15 - high-low range distributions

Another set of indicators is the current LAST_PRICE relative to the 1-minute-ago x-minute high-low range. E.g., if at 1241 the LAST_PRICE is 8801, and between 1200 and 1240 the highest price was 8800 and the lowest price was 8700, then we return 1.01. We scale this such that if the current price matches the high, it will be 1; if it matches the low, it will be -1; if it's at the mean of the high and low, it will be 0; and so on. This can be a good proxy for breakouts of a range, which may suggest continuation of previous movements. Figure 16 provides an example at the 10-minute timeframe.

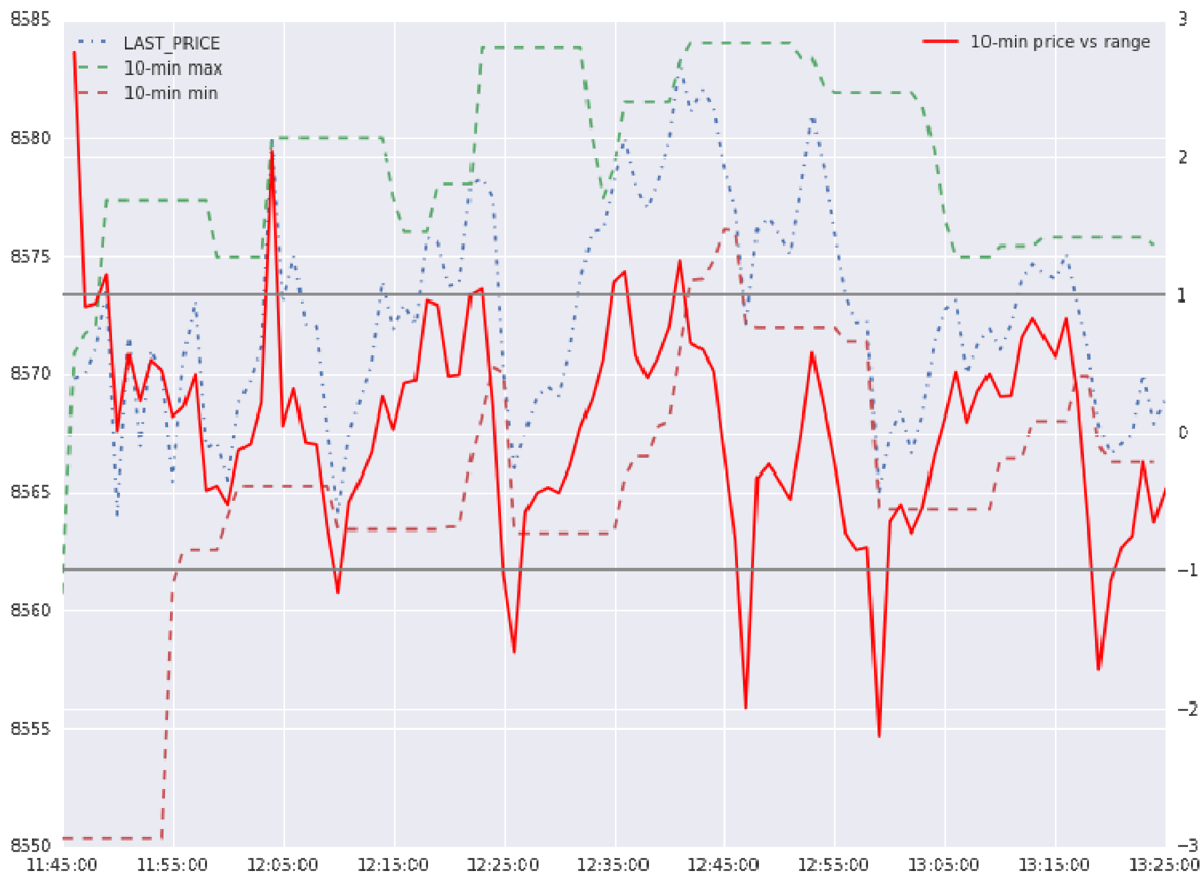

Figure 16 - Price vs range example

Columns to drop:
1_min_ma (same as LAST_PRICE)
1_last_vs_ma
1_min_volumes (same as VOLUME)