Weather data statistics

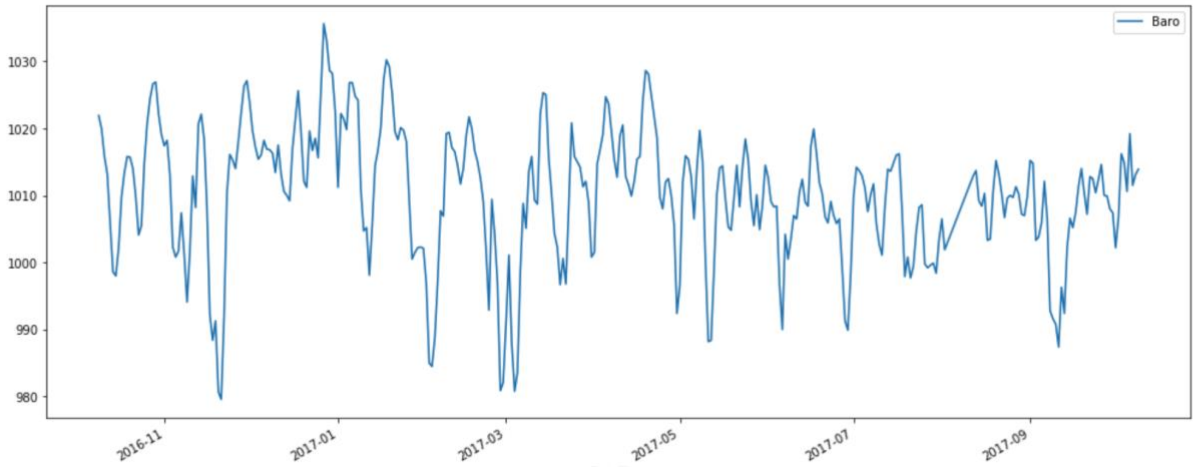
Table of Contents

Introduction

The following weather data statistics was calculated based on the collected weather data in a time span of a year between 2016-10-09 and 2017-10-09. The data is first visualized to get a first feel of it. In the next step essential descriptive statistics is calculated (mean, std dev, min and max). In the second part some outliers were manually inserted into data. Two methods were tested for detecting and removing the outliers. The first method is visual inspection of data using any visualization library like Matplotlib. The second method is called Outlier Labelling Rule (Hoaglin, Iglewitz, Tukeym, 1986, Performance of some resistant rules for outlier labeling). Using this rule of thumb we could attempt to automatically determine what values are outliers based on distance from 25% and 75% percentile. This method has also been tested on the example data set in the last section.

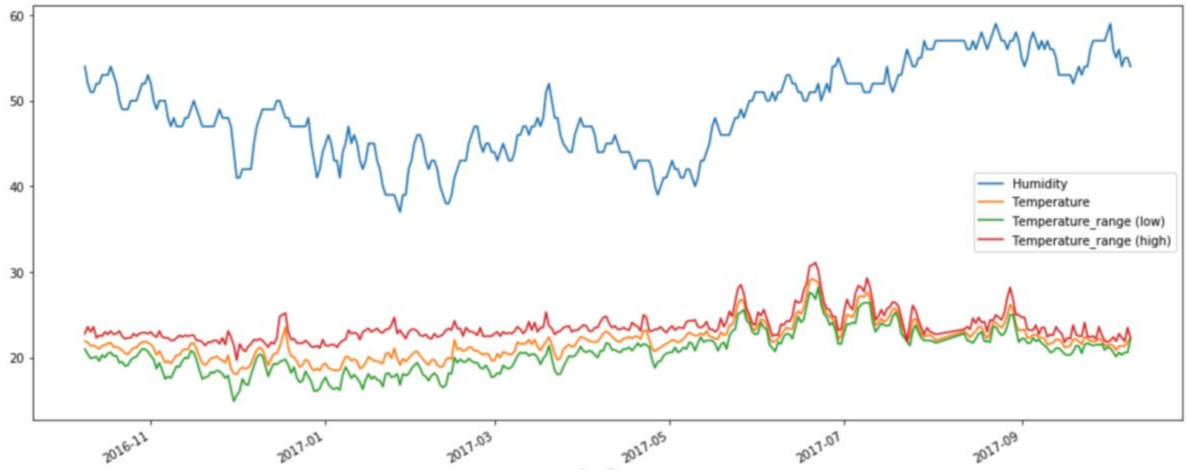
Quick look at the data

Barometric pressure



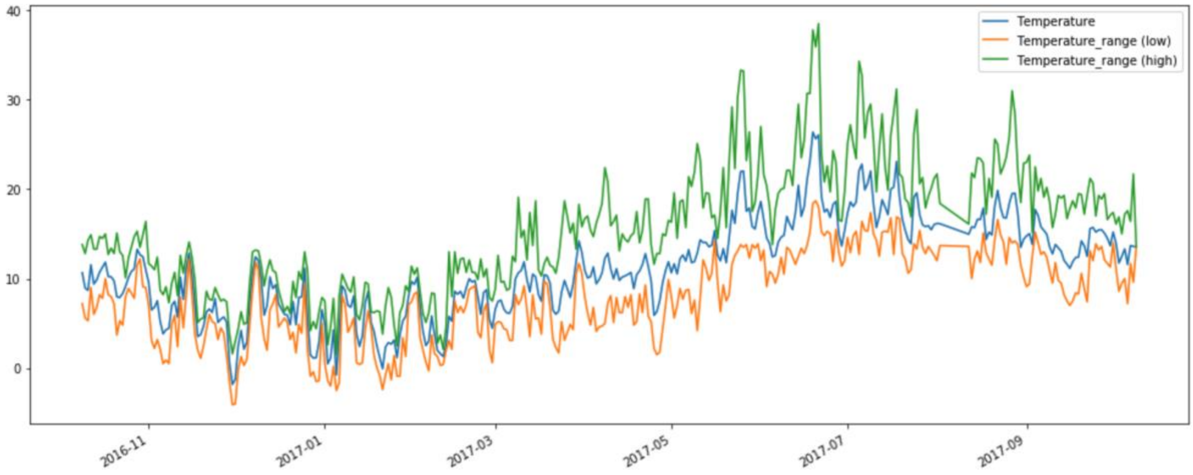
	Baro
count	355
mean	1.010
std	10
min	980
25%	1.005
50%	1.011
75%	1.016
max	1.036

Indoor temperatures



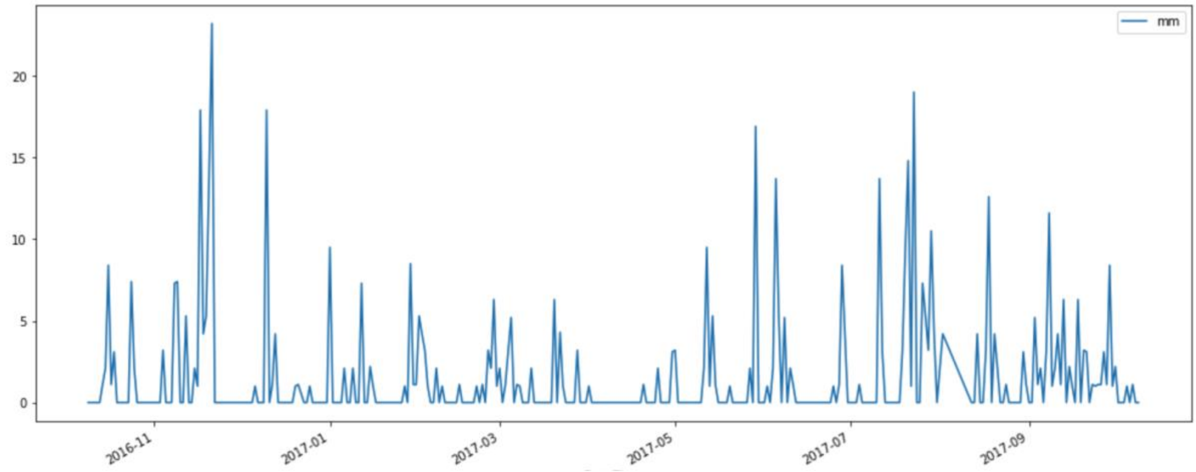
	Humidity	Temperature	Temperature_range (low)	Temperature_range (high)
count	354	354	354	354
mean	48,520	21,828	20,556	23,534
std	5,189	2,058	2,405	1,701
min	37	18,040	14,900	19,700
25%	44	20,345	18,725	22,500
50%	48	21,710	20,600	23,200
75%	52	22,710	21,900	24,100
max	59	29,210	28,200	31,100

Outdoor temperatures



	Temperature	Temperature_range (low)	Temperature_range (high)
count	355	355	355
mean	11,139	7,866	15,520
std	5,355	4,879	7,034
min	-1,810	-4,100	1,500
25%	7,390	4,350	10,250
50%	10,960	8	15,100
75%	15,050	12,050	19,850
max	26,38	18,7	38,500

Rain fall

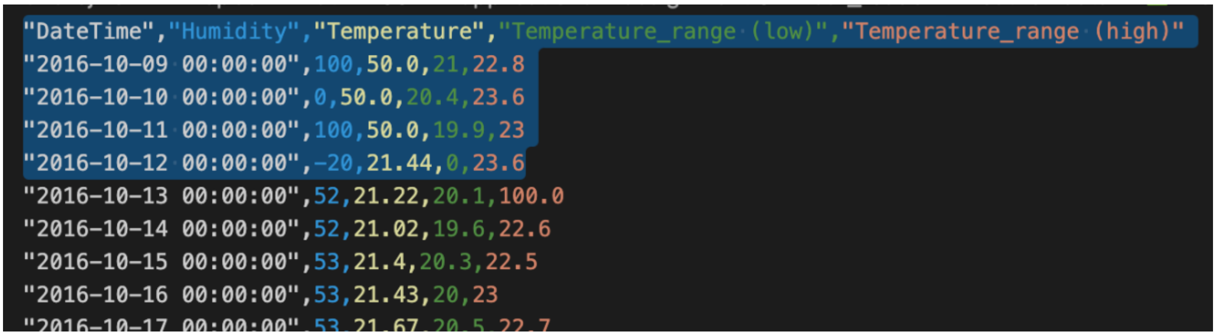


count	353
mean	1,549
std	3,325
min	0
25%	0
50%	0
75%	1,100
max	23,200

Adding outliers to the indoor temperatures data

The indoor temperatures data has been modified with some false values, which could happen in case of sensor malfunction or human error:

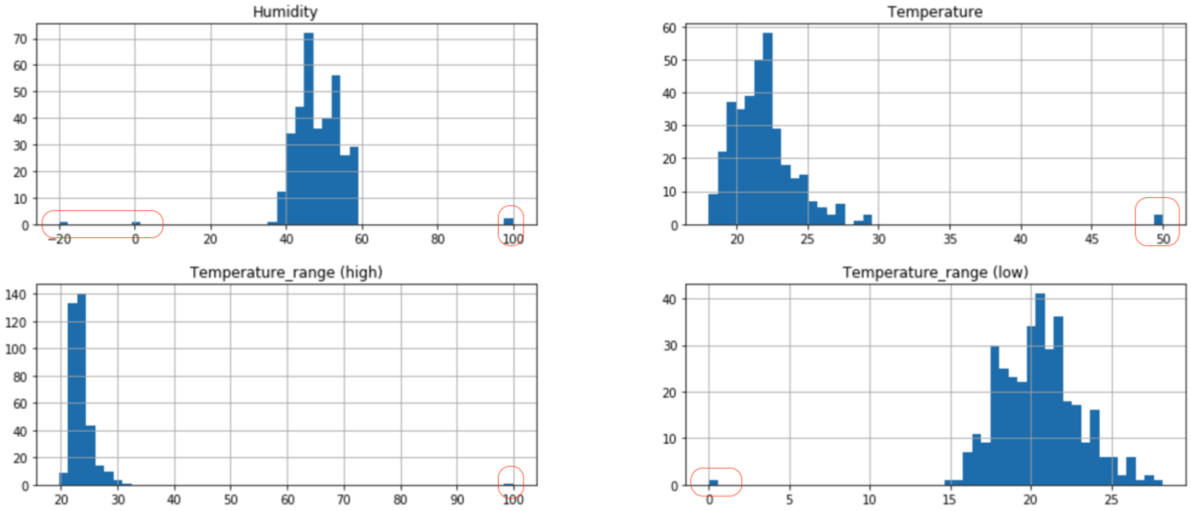
The first 5 lines of data were modified for columns humidity, temperature and temperature low range:

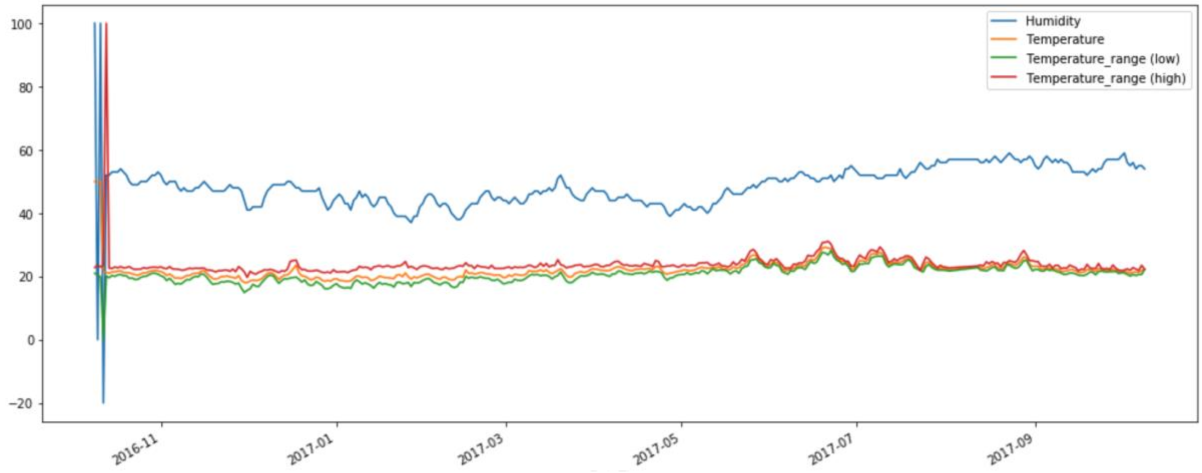


```
"DateTime","Humidity","Temperature","Temperature_range (low)","Temperature_range (high)"
"2016-10-09 00:00:00",100,50.0,21,22.8
"2016-10-10 00:00:00",0,50.0,20.4,23.6
"2016-10-11 00:00:00",100,50.0,19.9,23
"2016-10-12 00:00:00",-20,21.44,0,23.6
"2016-10-13 00:00:00",52,21.22,20.1,100.0
"2016-10-14 00:00:00",52,21.02,19.6,22.6
"2016-10-15 00:00:00",53,21.4,20.3,22.5
"2016-10-16 00:00:00",53,21.43,20,23
"2016-10-17 00:00:00",53,21.67,20.5,22.7
```

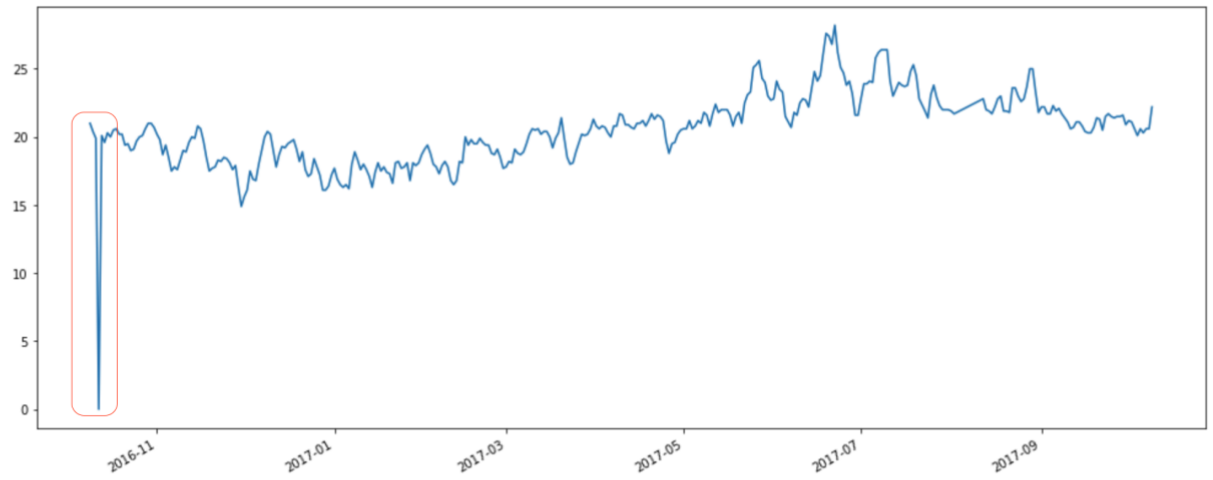
Visual inspection of data

The outliers can be easily spotted just by plotting the data again in Matplotlib. It is clear that the values occurring at the beginning of the time series seem out of place and look like some kind of a sensor malfunction or human error:





It is easier to visually detect outliers by looking at each time series separately:



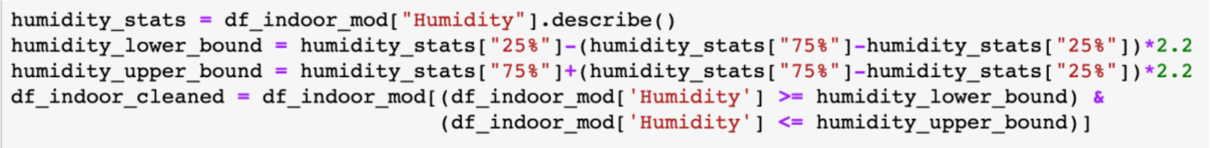
Automatically removing outliers using the Outlier Labeling Rule (Tukey's rule)

We could also attempt to detect and remove outliers automatically. Looking again at the basic descriptive statistics of the modified CSV document we can see that the mean and std dev haven't changed much, but we can see that min and max values reveal it might be worth looking more closely at the data:

	Humidity	Temperature	Temperature_range (low)	Temperature_range (high)
count	354	354	354	354
mean	48,520	21,828	20,556	23,534
std	5,189	2,058	2,405	1,701
min	37	18,040	14,900	19,700
25%	44	20,345	18,725	22,500
50%	48	21,710	20,600	23,200
75%	52	22,710	21,900	24,100
max	59	29,210	28,200	31,100

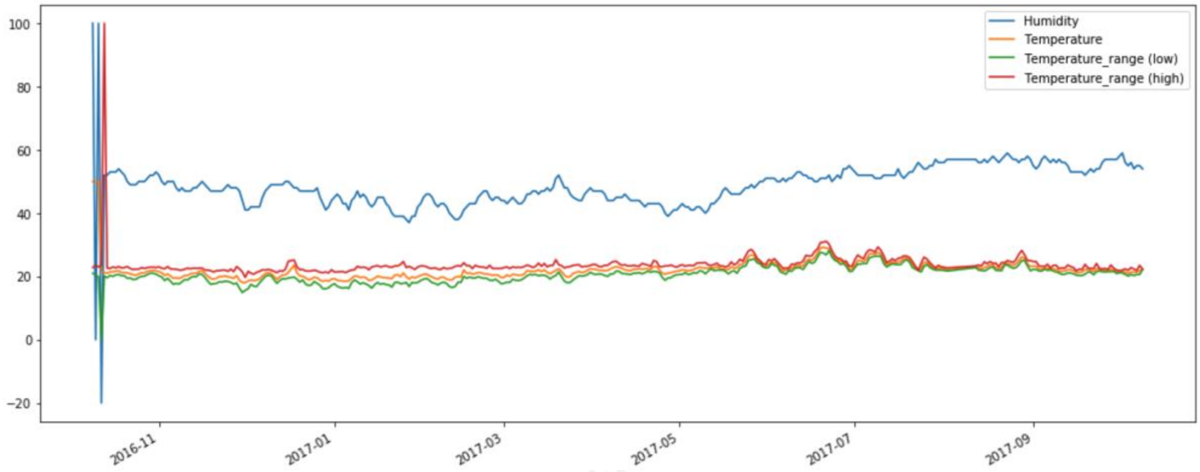
	Humidity	Temperature	Temperature_range (low)	Temperature_range (high)
count	354	354	354	354
mean	48,440678	22,067828	20,499435	23,753107
std	7,858027	3,30504	2,641508	4,40527
min	**-20**	18,04	**0**	19,7
25%	44	20,345	18,7	22,5
50%	48	21,725	20,6	23,2
75%	52	22,765	21,9	24,175
max	**100**	**50**	28,2	**100**

The outlier labelling rule states that the outliers are values more than 1.5 times the interquartile range from the quartiles — either below Q1 − 1.5IQR, or above Q3 + 1.5IQR. However here we will be using a slightly modified factor 2.2 instead of 1.5 as proposed in Hoaglin, Iglewitz, Tukeym, 1986, Performance of some resistant rules for outlier labeling. An example of Python code to apply the Outlier Labeling Rule on one column might look like this:

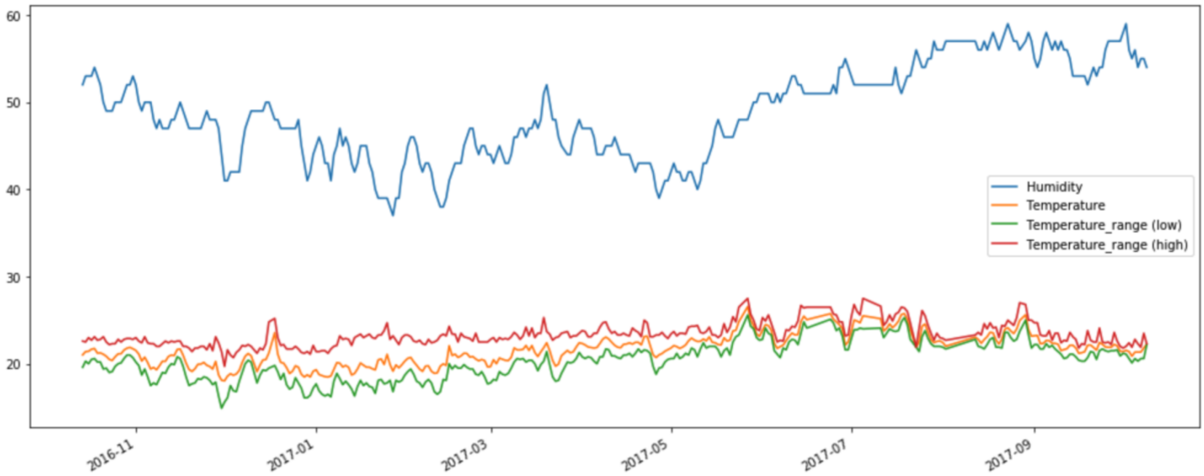


```
humidity_stats = df_indoor_mod["Humidity"].describe()
humidity_lower_bound = humidity_stats["25%"]-(humidity_stats["75%"]-humidity_stats["25%"])*2.2
humidity_upper_bound = humidity_stats["75%"]+(humidity_stats["75%"]-humidity_stats["25%"])*2.2
df_indoor_cleaned = df_indoor_mod[(df_indoor_mod['Humidity'] >= humidity_lower_bound) &
                                  (df_indoor_mod['Humidity'] <= humidity_upper_bound)]
```

Before automatically cleaning outliers:



After automatically cleaning outliers:



Note: the code used to calculate statistical values, plot the data and remove the outliers is included in the accompanying Jupyter notebook file EDA.ipynb.