Statistical analysis of Milano Weather Station Data

By

Ali A. Rahmanian

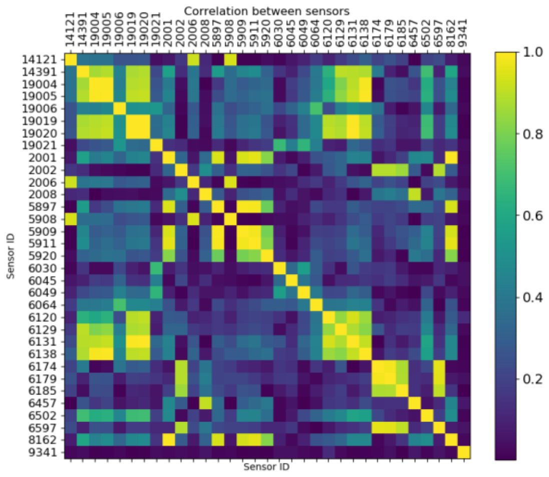
In this assignment, all the experiments and the steps done to do understand some interesting statistics in Milano Weather Station Data[1] are provided in a respective order in the following sections. Python is used for implementation and the source code is available on GitHub[2].

I: Preprocessing

The first step is data cleaning phase. Based on our observation, the data is stored in multiple CSV files which are not sorted, clean, nor normalized. The chosen dataset includes sensing data of different types of sensors geo-distributed in Milan city. These sensors did not necessarily start sensing data at the same time slot and they may have missing data for a number of time slots without any specific patterns. In order to make the data clean and ready to analyze we 1) read the data of individual sensors; 2) sorting the data by their sensing date & time; 3) Normalize the data of individual sensors in scale of 0 to 1.

II: Correlation evaluation between sensors

As the very first experiment, we would like to show if there are any dependencies between sensors with a color map demonstrating correlations[3]. In this experiment we calculate correlation between any two possible sensors in the dataset based on the intersection of their whole activity (time slots). The following image illustrates the correlation map between any two sensors in the data set. There are a lot of information and statistics can be deduced from the image. To name, there is very high correlations between sensors 1904,1905,14391, 19019, 19020, 6129, 6131, and 6138, which all were wind speed sensors. There are dependencies between sensors of different groups such as temperature sensors (5909, 5911, 5920, 8162, 2001, 5897); and even between relative humidity sensors (6174, 6179, 6185, 6597).



III: Outlier analysis

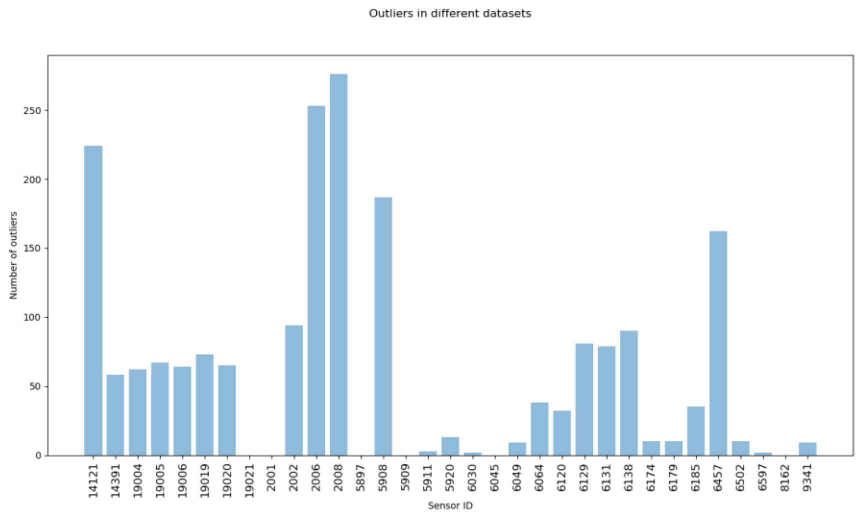
In prior section, we calculated correlation between time series data without analyzing outliers. However, existence of outliers may have negative impact on our analysis and statistical modeling. In this section, we

1 https://dandelion.eu/datagems/SpazioDati/milano-weather-station-data/resource/

2 https://github.com/ali-rahmanian/data_stats

3 File stat1.py in the attached project

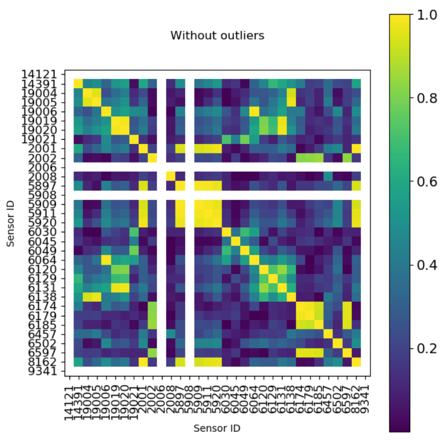
are trying to find observations which do not follow the pattern of the rest of the data, called outliers[4]. We identify outliers based on interquartile change (IQR) score. To do so, observations lower than **Q1-1.5*IQR** and observation higher than **Q3+1.5*IQR** are identified as outlier points. The following image demonstrates number of identified outliers in individual sensors. As it is seen, wind direction sensors and temperature sensors have no outliers while most of identified outliers are found in the data of global radiation, precipitation, and net radiation sensors.

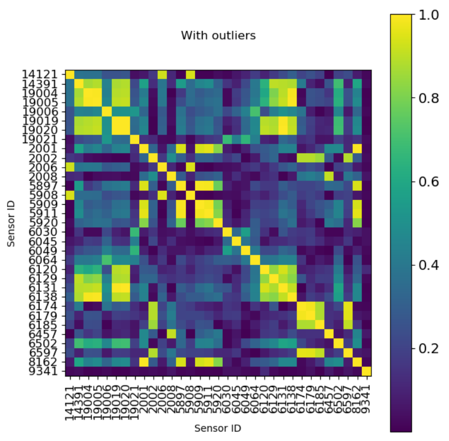


IV: Evaluation of outlier removal on correlation

In this experiment, we calculated the correlation between any two sensors after removing their outlier points in their time-series data[5]. We first read and clean the data of all sensors. We then remove outliers based on the approach provided in section III. Finally, data normalization function is applied. In order to provide a precise evaluation, we have calculated and illustrated correlation map between sensors before and after removing outlier data. The right part of following image illustrates color map of correlation between any two sensors before outlier removal while the left part shows the impact of outlier removal on correlation between any two sensors. Outlier removal remove very interesting results in this dataset which needed to be fully analyzed: Temperature sensors are much more correlated now (5920, 5911, 5909, 5897, 2001, 8162). Amazingly, wind sensors (19004, 19005, 6138) where located in the same place (same latitude and longitude) have very high correlation. Moreover, unlike the experiment without outlier removal, these wind sensors (19004, 19005, 6138) where located in the same place have very low correlation with other wind sensors (14391, 19019, 19020, 6129, 6120, 6131, 6138) located in other places. Furthermore, high correlation between two wind direction sensors (6064, 19006) located in the same place is identified now unlike the scenario without removing outlier. There are lots of more facts illustrated in the image below that proof proficiency of removing outliers. Last but not least, there is no remaining data for all precipitation sensors (14121, 2006, 5908, 9341) after outlier removal which is just happened to these type of sensors. The reason behind this experience is that these datasets are sparse with majority 0s and that is why after removing outliers no data is remaining to be considered for correlation.

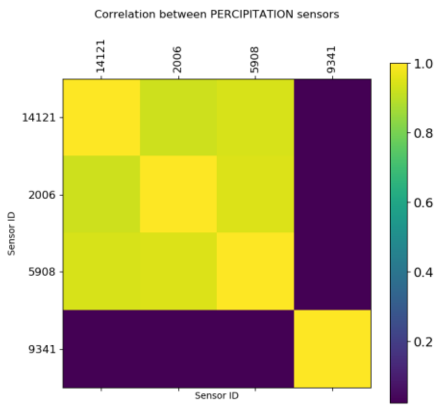
[4] File stat2.py in the attached project
[5] File stat3.py in the attached project





V: Evaluation of correlations between dropped sensors

As we mentioned in the last section, some of sensors which were all of precipitation sensors were dropped from the last experiment because of applying outlier removal technique. The achieved results was expecting since the data of precipitation sensors were sparse including lots of zeros and that is why data of precipitation sensors is fully dropped from the last experiment. In this experiment, we are going to test dependencies between all sensors were dropped from the last experiment[6]. The following image illustrates color map of correlations between these sensors which shows an interesting fact. There are high correlations between all precipitation sensors except sensor 9341 where is located in different location far from the others. Thus, there is very high correlation between precipitation sensors located in different spots in Milan city except sensor 9341.

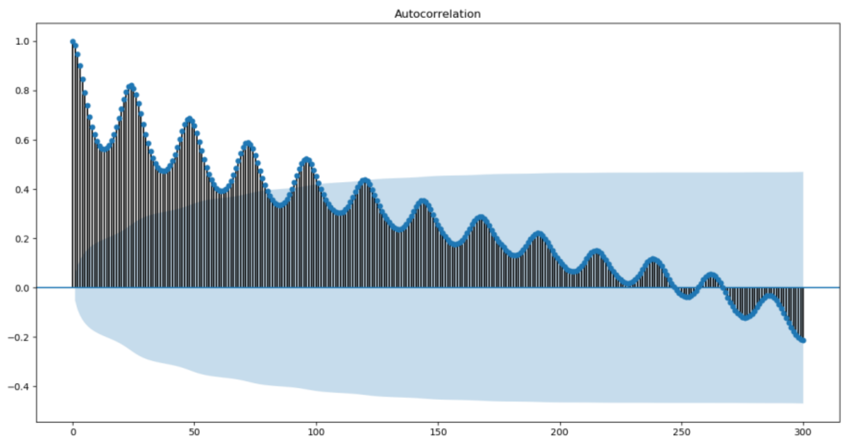


VI: Stationary and seasonality

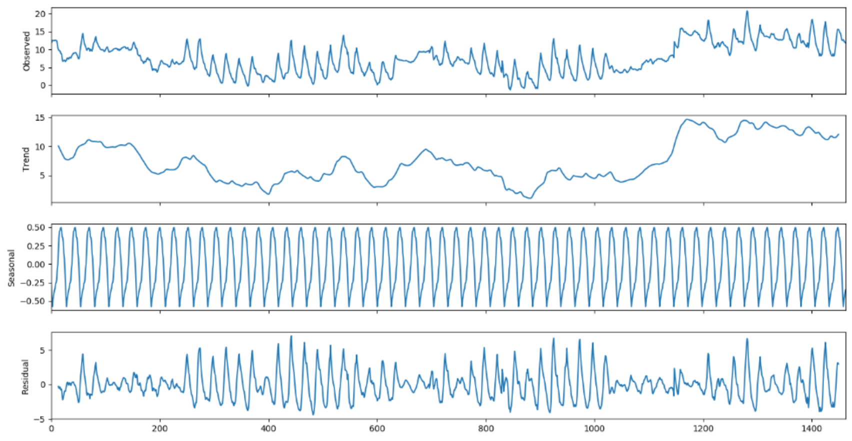
In this experiment, we are going to check whether the data is stationary and check whether the data is seasonal or not. Furthermore, we decide whether the data is stationary and identify the exact seasonal cycles in the data in order to detect trend of the data and finally decompose seasonality in order to achieve the residual data[7]. In this experiment, we chose one of the sensor we feel there may be some sort of seasonality. Since temperature seems highly seasonal, we choose sensor 5911 which is sensing temperature in Milano

[6] File stat4.py in the attached project
[7] File stat5.py in the attached project

- Viale Marche for this experiment. Firstly, we applied autocorrelation to understand probable serial dependence of the time-series to a lagged version of itself. In other words, we use autocorrelation for understanding linear dependent correlation in the evaluating sensor. The reason behind trying to understand autocorrelation in the residuals is that most of forecasting models such as regressions assume that there is no autocorrelation in the data and thus having dependency might be misleading. As illustrated in the following image, autocorrelation function (AFC) for different lags are calculated (in this experiment up to lag=300) and as the outcome the following plot is demonstrated, which clearly shows high seasonal correlation with a trend.



Now, based on the identified lag that we found a full cycle every 24 lags (24 hours) which is a logical for temperature sensor. We then detect the whole cycles in the time-series and the first observation can be decomposed to achieve the residuals as illustrated below. By this way, trend and residual are finally decomposed.



VII: Conclusion

We could have done more analyses such as time-series clustering, decision fusion, regression classification, principal component analysis (PCA) for reducing number of features we have to analyze if we assume any sensor is a feature to determine dependencies and independencies. But it seems they are outside the scope of this assignment since it is supposed to identify a few interesting statistics in the dataset.