A Computational Intelligence Approach to Water Demand Forecasting and Anomaly Detection in Water Consumption Time Series

(Zastosowanie inteligencji obliczeniowej do przewidywania zapotrzebowania na wodę i wykrywania anomalii w szeregach czasowych zużycia wody)

Dominik Samorek

Praca licencjacka

Promotor: dr hab. Piotr Wnuk-Lipiński

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

3 września 2018

Dominik Samorek

.....

.....
(adres zameldowania)

.....

.....
(adres korespondencyjny)

PESEL: .....

e-mail: .....

Wydział Matematyki i Informatyki
stacjonarne studia I stopnia
kierunek: Indywidualne Studia Informatyczno-Matematyczne
nr albumu: 281623

Oświadczenie
o autorskim wykonaniu pracy dyplomowej

Niniejszym oświadczam, że złożoną do oceny pracę zatytułowaną *A Computational Intelligence Approach to Water Demand Forecasting and Anomaly Detection in Water Consumption Time Series* wykonałem/am samodzielnie pod kierunkiem promotora, dr. hab. Piotra Wnuk-Lipińskiego. Oświadczam, że powyższe dane są zgodne ze stanem faktycznym i znane mi są przepisy ustawy z dn. 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (tekst jednolity: Dz. U. z 2006 r. nr 90, poz. 637, z późniejszymi zmianami) oraz że treść pracy dyplomowej przedstawionej do obrony, zawarta na przekazanym nośniku elektronicznym, jest identyczna z jej wersją drukowaną.

Wrocław, 3 września 2018

(czytelny podpis)

Abstract

This work presents methods of computational intelligence for water usage forecasting and anomaly (leaks) detection performed on data from waterworks network of Wrocław. Additionally, the work contains comparison of known methods with proposition of their future improvement.

Celem niniejszej pracy jest zaprezentowanie metod inteligencji obliczeniowej w przewidywaniu zużycia wody i wykrywania anomalii (wycieków) na danych z wrocławskiej sieci wodociągowej. Dodatkowo, praca zawiera porównanie znanych metod i propozycje rozszerzenia ich w przyszłych badaniach.

Contents

Chapter 1

Introduction

Water usage forecasting is very important topic for its urban suppliers. As described in [1], [2] and [3] water schedule optimization is wide field of study, which can bring much savings in energy costs and consumption. Other significant problem in this area is anomalies detection, i.e. finding moments in time, where water usage behaves in singular way.

1.1 Leaks detection problem

Application of tracking down anomalies is a leaks detection problem. In case of water suppliers it is very hard problem, because most of the leaks appears under the ground, therefore sometimes it is impossible to detect it in ordinary methods. Such unnoticed leakage can cause big losses for long period of time.

1.2 Overview of this work

In this thesis all way from raw data to water consumption forecast is explored on data from waterworks of Wrocław, with solutions similar to those in [1], [2] and [3]. In addition, models based on Recurrent Neural Networks are tested.

With water demand prediction, ability of trained models to detect anomalies is tested, with comparison to more basic methods. After such experiments, improvement of used methods is suggested.

Chapter 2

Data Description

Dataset consisted of information from six flowmeters of Wrocław's waterworks, which can be seen on Figure 2.1. Each of them held independent area and provided numerical value of water flow counted from 1 July 2014 with ten minutes frequency. For data analysis information from every flowmeter was treated as a time series.

2.1 Time series

Time series[4] is a sequence of random variables realizations ordered by time and measured with fixed time step.

Time series decomposition

For analysis three components of every time series are considered:

- **trend** - long term direction
- **seasonal** - systematic, time related movements
- **irregular** - short term, often chaotic fluctuations

In our water flow example trend can be related to increasing number of corresponding area residents, so more water is used. Seasonal component is most connected to people weekly lifestyle.

These components can be connected in additive, multiplicative or mixed way. That is, observed series O_t can be decomposed as $O_t = T_t + S_t + I_t$ or $O_t = T_t \cdot S_t \cdot I_t$ or in some mixed way.

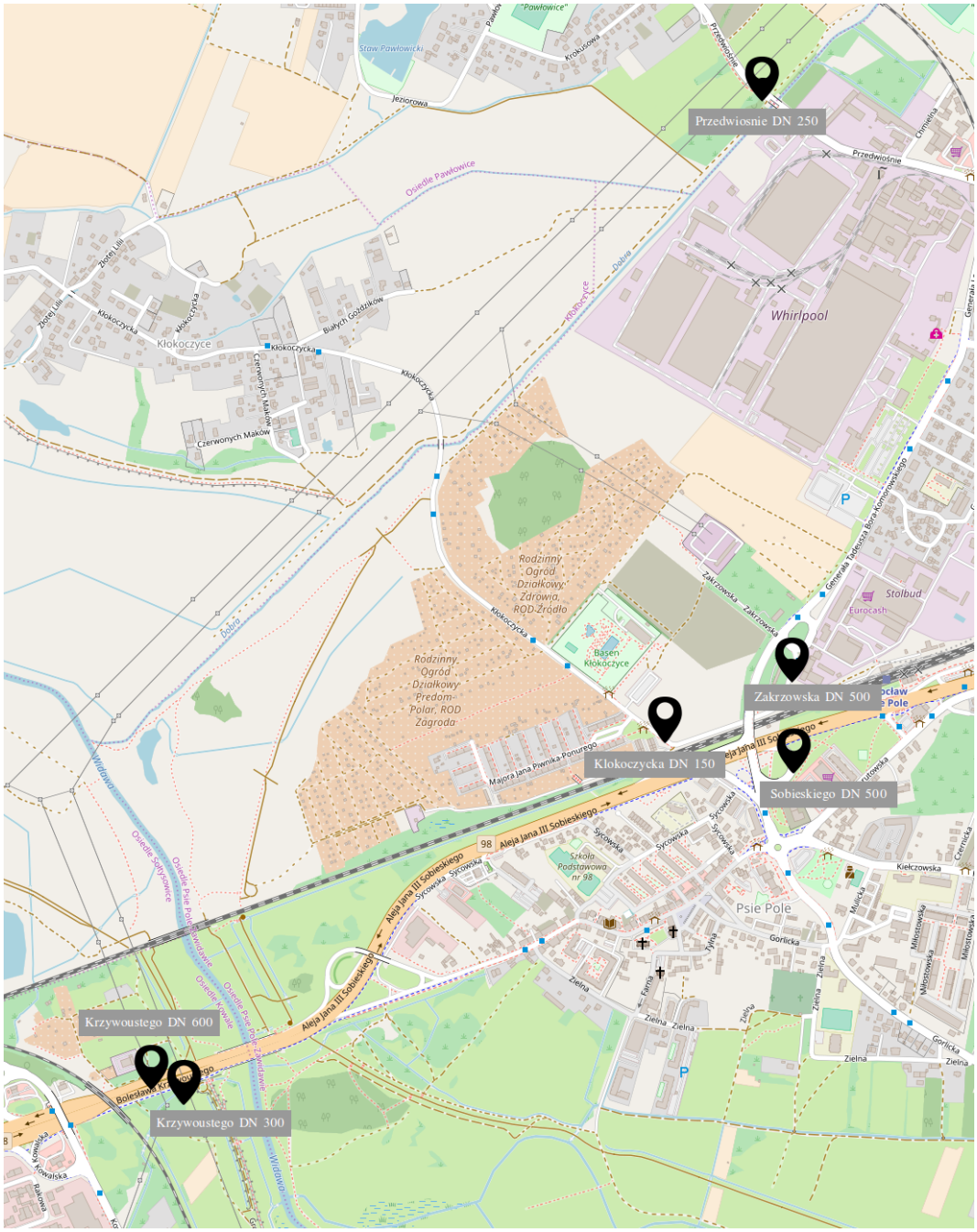

Figure 2.1: Map shows region of Wrocław with places of six flowmeters. These are Kłokoczycka DN 150, Krzywoustego DN 600, Krzywoustego DN 300, Zakrzowska DN 500, Przedwiośnie DN 250 and Sobieskiego DN 500. Number next to the name of the region is a value of nominal diameter of particular pipe. The map was made with usage of OpenStreetMap [9].

Stationarity of time series

It is very important for analysis to decompose time series and work only with irregular component I_t. After such operation one can assume stationarity of time series,

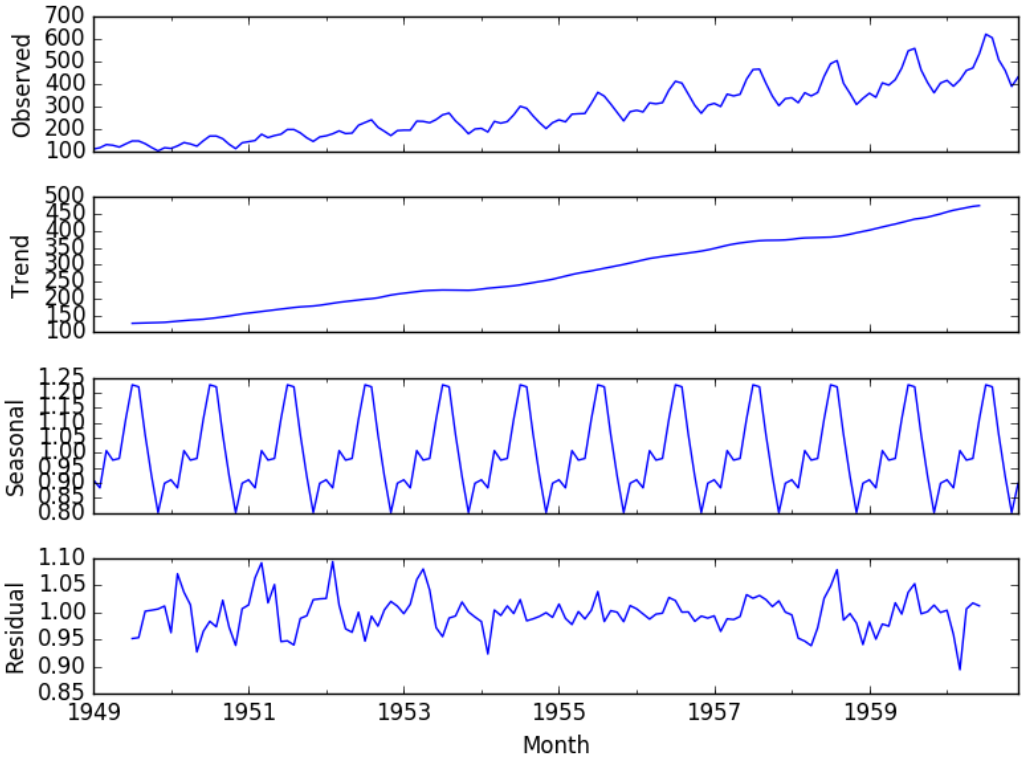

Figure 2.2: On the upper plot example of observed time series is shown. Lower are plotted three components: trend, seasonal and irregular (residual). In this example series was decomposed in multiplicative way.

i.e. every random variable in series has the same probability distribution.

2.2 Regression problem

After decomposing time series it can be assumed, that all random variables in series have the same probability distribution, i.e. all observed values are realizations of the same random variable Y.

For estimating Y by regression approach, group of predictors is needed. Therefore random variables $X_1, ..., X_d$ should be defined.

Let

$$y = [y_1, ..., y_n]$$

be observed values of variable Y, and let

$$\mathbf{X} = \begin{pmatrix} x_1^1 & \dots & x_n^1 \\ \vdots & \ddots & \vdots \\ x_1^d & \dots & x_n^d \end{pmatrix}$$

be observed values of predictors $X_1, ..., X_d$. In such case the goal is to find function

$f(x) = f(x^1, ..., x^d)$ which, for every sample $i = 1, ..., n$ minimize the distance between y_i and $f(x_i^1, ..., x_i^d)$ in terms of some error function. For this thesis, as in [2], mean absolute percentage error (MAPE) function was used.

$$MAPE(f, Y, \mathbf{X}) = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{f(x_i) - y_i}{y_i} \right|,$$

where for simplicity, notation $x_i = [x_i^1, ..., x_i^d]$ was used.

Chapter 3

Description of Algorithms

For water usage forecasting, similarly to [2] and [3], SVR and Gaussian processes models were used. In addition to these, also artificial neural networks models were tried, i.e. LSTM networks.

3.1 SVR

Support Vector Regression ([5], [6]) is kind of SVM (Support Vector Machine) used for regression problem. Suppose $\{(x_1, y_1), ..., (x_n, y_n)\} \subset \mathbb{R}^d \times \mathbb{R}$ are observed data points. The goal is to find a function $f(x)$ that for every data point x_i, observed value of function is near, i.e. $|f(x_i) - y_i| < \varepsilon$ and additionally, f is as flat as possible.

In linear case, f has a form:

$$f(x) = \langle w, x \rangle + b, \text{where } w \in \mathbb{R}^d, b \in \mathbb{R}.$$

$\langle \cdot, \cdot \rangle$ denotes the dot product in Euclidean space. In this case, the flatness of f means, that $||w||^2$ need to be minimized.

In fact, with fixed ε, not always such function exists, or even one could want to allow some errors. In such case, instead of minimizing $||w||^2$, one have to minimize

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i - \xi_i^*)$$

subject to

$$\begin{cases} y_i - \langle w, x \rangle - b & \leq \varepsilon + \xi_i \\ \langle w, x \rangle + b - y_i & \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq 0 \end{cases}$$

Here one can easily see meaning of C parameter, which is some kind of trade-off between the flatness of f and error toleration.

It is important to mention, that for describing w not all x_i are necessary, but only these which are outside the ε-tube. Such points are called **support vectors**.

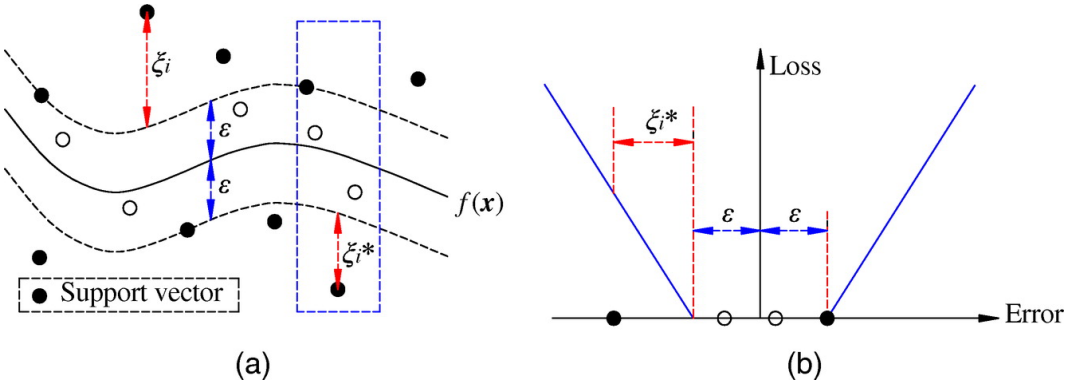

Figure 3.1: Empty dots are in smaller than ε distance to f, so there is no penalty for them. Dark dots are outside of ε-neighborhood so corresponding loss function is greater than 0. They are also called **support vectors**, because predicted function is described only by them.

In nonlinear case observed data can be implicitly mapped via kernel function $\Phi(x)$ to some other, more dimensional space. Such mapping can be computed efficiently, because there is no need to know it explicitly. Values of $\langle \Phi(x), \Phi(x') \rangle$ is enough to find a solution.

3.2 Gaussian processes

In Gaussian process regression (GPR [7]) the point is to describe probability distribution over functions. Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution. Such process is completely described by its mean and covariance functions, $\mu(x)$ and $K(x, x')$.

Let $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\} = (\mathbf{X}, \mathbf{y})$ be observed data points, where $y_i = f(x_i) + \epsilon_i$, $f \sim GP(\cdot|0, K)$ and $\epsilon_i \sim \mathcal{N}\left(\cdot|0, \sigma^2\right)$.

With such prior it is possible to make predictions. Let x_* be new point, and y_* value in this point. Then posterior distribution can be expressed as

$$p(y_*|x_*, \mathcal{D}) = \int p(y_*|x_*, f, \mathcal{D}) p(f|\mathcal{D}) df$$

and the marginal likelihood

$$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|f, \mathbf{X}) p(f) df.$$

In such a case predicted value y_* would be the mean of this posterior distribution.

Marginal likelihood can be treated as a function of K's hyperparameters, so they can be learned from data via likelihood maximization.

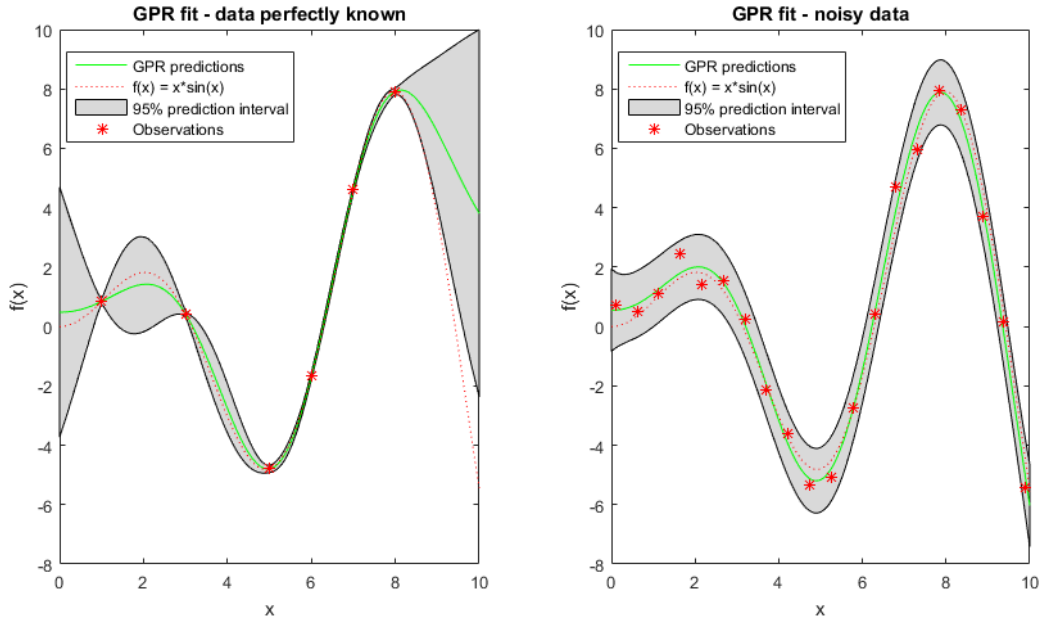

Figure 3.2: In Gaussian process regression it is possible to explicitly set a strength of data noise. In addition to function prediction, such model gives also probabilistic information, e.g. about confidence intervals.

3.3 LSTM neural networks

Long Short Term Memory networks [8] are a special kind of RNNs (Recurrent Neural Networks). Their architecture is designed to work with long-term dependencies, because they have additional memory cell in which they are able to store some data to "remember", or "forget" it and save something new.

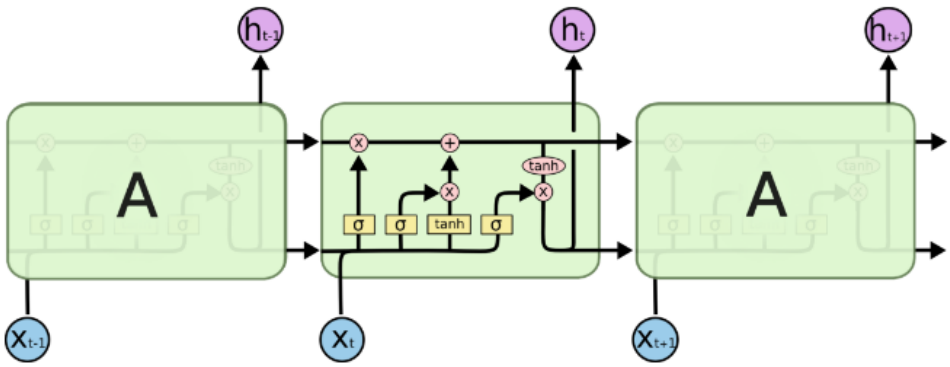

Figure 3.3: Unrolled LSTM layer. Upper horizontal line describe memory cell and its information flow. The lower parts are similar to those in RNN, but with additional elements for interaction with memory cell.

Let x_t be the input to the layer at the time t, C_t be the memory cell value and h_t - the output of the layer. Then the new value of memory cell C_t is computed as

follows:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \end{aligned}$$

and new output value h_t:

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t \odot tanh(C_t). \end{aligned}$$

The σ and $tanh$ activation functions can be replaced, for example by $RectifiedLinearUnit$.

Chapter 4

Data Preparing

The general purpose in this work (as described in previous Chapters) was to create group of regression models, which could predict water consumption. Such forecast, if it is precise enough, should be later compared to real values. When much differences between estimation and true consumption are noticed, it is a sign that leakage might happened.

4.1 Data preprocessing

Before applying any machine learning model three operations needed to be performed on raw data described in Chapter 2. Firstly, series needed to be resampled to lower frequency and any outliers, which could mess analysis, should have been removed. Afterwards, as described in 2.1, time series had to be decomposed to be more stationary. This step was very important for prediction accuracy. As the last part before learning, to perform regression some predictor variables needed to be determined, as described in 2.2.

4.1.1 Frequency change

Ten minutes frequency was to high for this application, because for every day only one value was to be predicted. Aggregating series to days would be to inaccurate, in day rhythm could be hidden useful information. Due to this fact time series was resampled to hours, i.e. six values from every hour with ten minutes frequency were summed up to one value for entire hour. Such operation reduced amount of data, but also made it more stable, as can be seen on the Figure 4.1

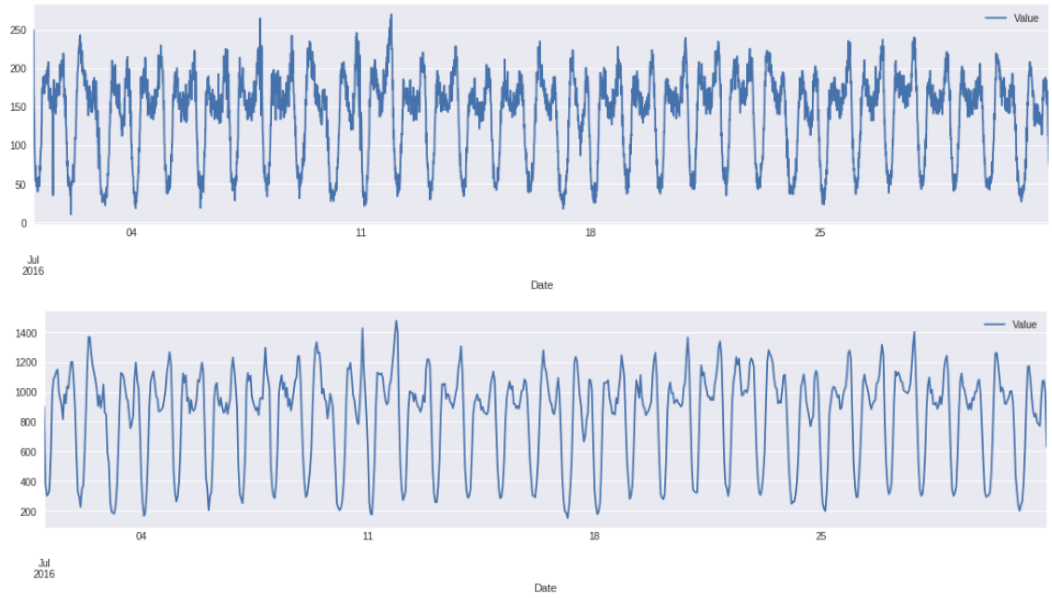

Figure 4.1: Ten minutes frequency was to high for this application, so series was resampled to hours, i.e. six values from every hour with ten minutes frequency were summed up to one value for entire hour.

4.1.2 Outliers removing

The data was collected by electronic devices, so it was susceptible to errors. That is why, before computing any statistics, any values, that were likely to be biased by such errors were removed.

In time series it is important to have sequential data without big gaps. Therefore instead of removing outliers entirely, its values were changed to be more suitable for series. For y, the old value of time series, the new value $\tilde{y}$ was computed according to formula

$$\tilde{y} = min(y,\ mean + c \cdot std),$$

where $mean$ was the mean of entire series, std was its standard deviation and constant c had value between 4 and 8, depending on particular dataset.

4.1.3 Seasonal component removing

As described in Section 2.1, it is important, to decompose time series before forecasting. In this example, trend component was omitted, it was not noticeable, because of too little time period of our series.

With seasonal component the case was much different. Because of people lifestyle two strong seasonal components could be distinguished - weekly and daily. Because only night time values were to predict, so daily seasonality was not important.

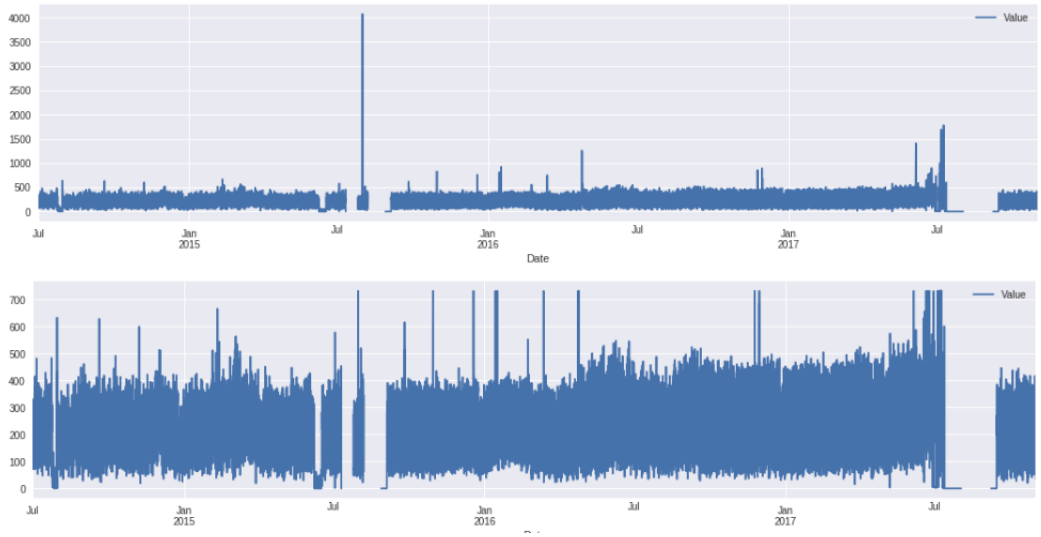

Figure 4.2: Such outliers, as can be seen on the upper plot, make analyze inaccurate. Therefore all values were cut on the level of data mean plus standard deviation multiplied by constant between 4 and 8, depending on particular dataset.

Finally, observed time series was represented as:

$$O_t = S_t^1(S_t^2 + I_t),$$

where S_t^1 and S_t^2 are two seasonal components.

Two extract I_t series was firstly divided by walking seven day mean according to formula

$$\tilde{y_i} = \frac{y_i}{\frac{1}{7}\sum_{j=i-6}^{i} y_j}$$

This operation reduced differences between the same days of week, i.e. all Mondays became more similar to each other, also Tuesdays etc. Later, to reduce disparity between all days of week, mean value of every day of week was computed and subtracted from the series respectively, accordingly to formula

$$\hat{y_i} = \tilde{y_i} - \frac{1}{|W_i|}\sum_{j \in W_i} \tilde{y_j},$$

where W_i is set of indexes of observed values from the same day of week as y_i.

4.2 Features extraction

Till this moment of preprocessing time series was still containing only single sequence of values. The idea was to apply regression models, so it was necessary to find proper features which could explain the series. What is more, series still had hour frequency, but only night time values were to forecast.

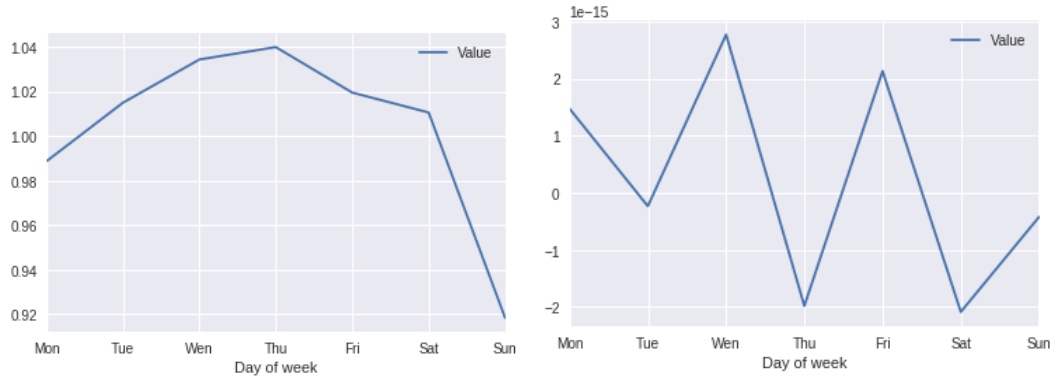

Figure 4.3: Week seasonality causes series to be non stationary. To remove this inconvenience, series was preprocessed to make means of all days of week equal to zero.

4.2.1 Finding proper night time

For this task hour which was most stable was chosen. After computing standard deviations of every hour it turned out (as can be seen on Figure 4.4), that 2 o'clock was the best, because it had the smallest value of this statistic.

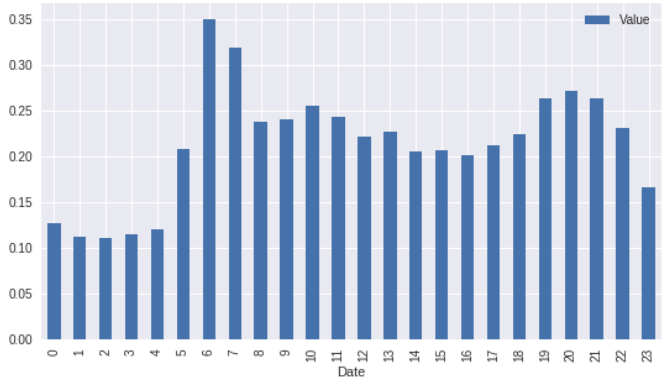

Figure 4.4: It was necessary to find proper night time for forecasting. Standard deviation of values from every hour was used as a measure of stability and the time with the smallest value was chosen, i.e. 2 o'clock.

4.2.2 Determination of predictor variables

For the features shifted back values of time series, which had the strongest autocorrelation response, were chosen. In addition to this, statistics from longer period of time were taken, where was wrapped information about more global behavior of series, which could be a consequence of some special events, weather or season.

Finally, such features were determined:

- histogram of values of twenty three hours before main night hour

- histogram of three values of series shifted by 24, 48 and 72 hours back
- mean of last twenty four hours
- means of values from last seven and fourteen days

This gave twenty nine values plus value from 2 o'clock which was to predict, but with one day frequency.

Last step of preprocessing was normalization or standard scaling of data. This means either scaling data to interval $[0, 1]$ or making its mean equal to 0 and variation to 1. Both were tried, because one models worked better with normalization, others with standard scaling.

Example of such preprocessed time series can be seen on Figure 4.5.

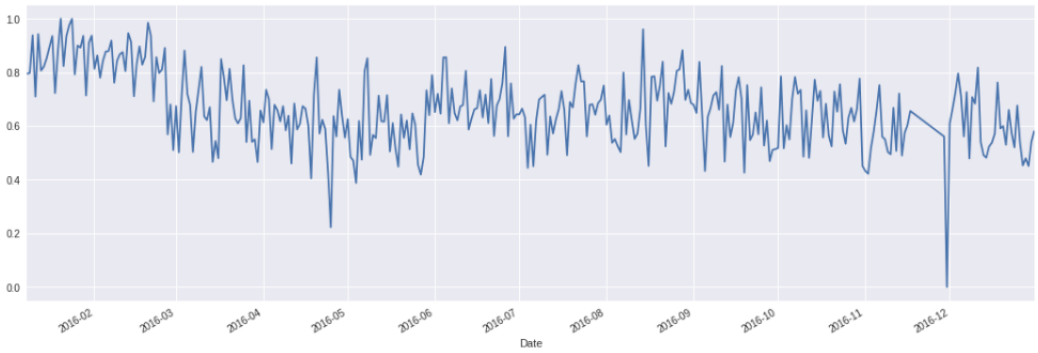

Figure 4.5: Example of preprocessed and normalized time series from Krzywoustego DN 600 flowmeter from year 2016.

4.3 Learning

As described in Chapter 3, regression models based on SVR, Gaussian Processes and LSTM neural network were used. Therefore, for better comparison, they needed to be checked on different dataset.

For time series prediction testing, simple cross validation cannot be used, because it would be possible to train model on data very near in time to testing data, and result would not be reliable. Therefore number of independent enough datasets needed to be created.

4.3.1 Datasets creation

After preprocessing described in Section 4.1 performed on data from all flowmeters separately, it was divided to nineteen periods of 350 days in such way that none two overlapped by more than 150 days. All these new datasets appeared in two forms, one after normalization and one after standard scaling.

Characteristics of all datasets before normalization and standard scaling can be seen in Table 4.1.

Table 4.1: Datasets characteristics

Dataset	Flowmeter	Date from	Date to	Mean	Standard deviation
0	Krzywoustego DN 600	2014-07-15	2015-06-29	-0.635	0.076
1	Krzywoustego DN 600	2014-12-04	2015-11-18	-0.67	0.128
2	Krzywoustego DN 600	2015-08-22	2016-08-05	-0.661	0.115
3	Krzywoustego DN 600	2016-04-12	2017-03-27	-0.7	0.064
4	Krzywoustego DN 600	2016-11-17	2017-12-27	-0.678	0.122
5	Krzywoustego DN 300	2014-07-07	2015-06-21	-0.54	0.285
6	Krzywoustego DN 300	2015-10-02	2016-09-15	-0.74	0.123
7	Krzywoustego DN 300	2016-05-28	2017-05-12	-0.769	0.12
8	Przedwiośnie DN 250	2016-02-06	2017-01-20	-0.746	0.126
9	Przedwiośnie DN 250	2016-10-15	2017-09-29	-0.737	0.152
10	Sobieskiego DN 500	2014-07-12	2015-06-26	-0.57	0.097
11	Sobieskiego DN 500	2015-03-06	2016-02-18	-0.594	0.09
12	Sobieskiego DN 500	2015-10-26	2016-10-09	-0.643	0.111
13	Sobieskiego DN 500	2016-07-19	2017-07-03	-0.717	0.077
14	Zakrzowska DN 500	2014-08-02	2015-07-17	-0.82	0.144
15	Zakrzowska DN 500	2015-05-09	2016-04-22	-0.883	0.134
16	Zakrzowska DN 500	2015-10-01	2016-09-14	-0.872	0.132
17	Kłokoczycka DN 150	2014-09-21	2015-09-05	-0.696	0.279
18	Kłokoczycka DN 150	2015-01-10	2015-12-25	-0.7	0.267

Characteristics of created datasets. Different number of datasets was created from data from different flowmeters, because some periods were not usable due to big gaps of information.

4.3.2 Applying a model

To every so prepared dataset, all models described in Chapter 3 were applied, but used many times with different parameters, which are explained in details in next Chapter.

Chapter 5

Experiments and Results

5.1 Parameters description

In this Section particular models, which were used, are described, including used parameters, neural network architecture etc.

SVR and Gaussian processes regression models were created and trained using scikit-learn library [10], LSTM network with use of Keras library [11] run on TensorFlow [12] backend.

5.1.1 SVR

Number of different parameters were used with SVR models, which can be seen in Table 5.1.

Table 5.1: SVR parameters

C	0.1	0.5	1	1.5	2	2.5	3	
γ	0.01	0.034	0.05	0.1	0.2	0.5	0.7	1
ε	0.05	0.1	0.5	1				

Parameters used with SVR models. All configurations were tried, also with normalized and standard scaled datasets, which gave 448 different models.

ε parameter was responsible for acceptable error tube, C parameter was the trade-off between flatness of function and error toleration. As kernel mapping, Radial Basis Functions were used, with γ parameter. This mapping is computed according to following formula:

$$\langle \Phi(x), \Phi(x') \rangle = e^{\gamma ||x - x'||^2}$$

5.1.2 Gaussian processes

As with SVR, for Gaussian process regression model different parameters values were tried, what can be seen in Table 5.2.

Table 5.2: GPR parameters

α	1e-10	1e-5	1e-2	1e-1	5e-1	1	1.5
c	0.01	0.05	0.1	0.5	1	1.5	2
l	0.01	0.05	0.1	0.5	1	1.5	

Parameters used with GPR models. All configurations were tried, also with normalized and standard scaled datasets, which gave 588 different models.

α parameter described strength of data noise. As a covariance function once again Radial Basis Functions were used, with l and c parameters, according to the formula:

$$K(x, x') = c^2 \cdot e^{-\frac{1}{2}||x/l, x'/l||^2}$$

Like was mentioned in Section 3.2, covariance hyperparameters were optimized during model fitting. For every model ten restarts were performed to find optimal solution.

5.1.3 LSTM networks

Few different network architectures were used:

- network with single LSTM layer with 64 units,
- network with four LSTM layers with 64 units each,
- network with four LSTM layers with 256 units each.

It gave six different models, because of normalized and standard scaled types of datasets.

In every network there was last dense layer with single output. All LSTM layers got input from two time steps. As activation function *Rectified Linear Unit* was tried, to avoid vanishing gradient problem. As a loss function mean absolute percentage error (MAPE) was used with Adam Optimizer [13]. All networks were trained for 15000 epochs with batch size equal to 32.

5.2 Results

Every model was trained separately on nineteen datasets consisting of 250 successive training samples and later tested on further 100 testing samples, as described in

Section 4.3.

5.2.1 SVR

In total, 448 SVR models were trained, because model with fixed set of parameters were tried separately on normalized and standard scaled datasets. For every model MAPE on training and testing data was computed. Later, three best models were chosen, depending on their average error on testing data, which was plotted on Figure 5.1.

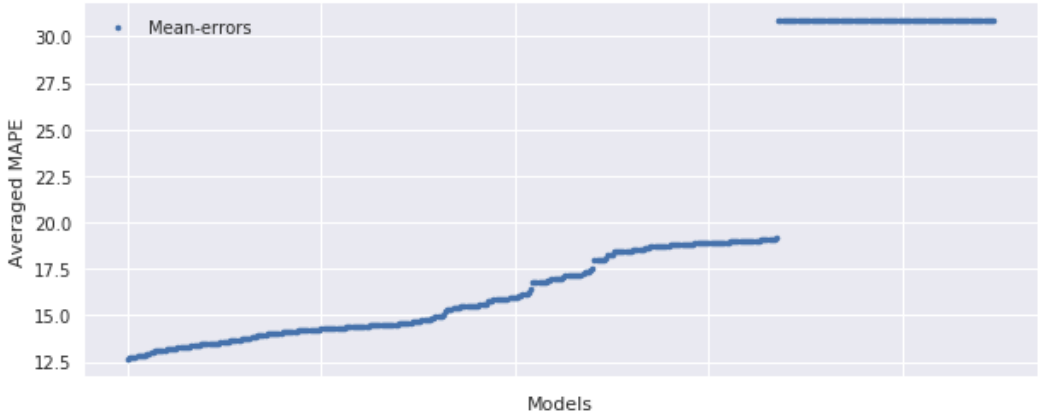

Figure 5.1: MAPE on testing data of all SVR models averaged on all datasets sorted ascending. These values were used to choose three best models.

All these three models were tried on normalized datasets. On Figure 5.2 their MAPEs computed on testing data on all datasets are plotted. Parameters of these models were similar, what can be seen in Table 5.3.

Table 5.3: Best SVR parameters

C	3	2.5	3
γ	0.05	0.034	0.034
ε	0.05	0.05	0.05
Average MAPE	12.66	12.74	12.77

Parameters of three best SVR models.

5.2.2 Gaussian processes

After the same procedure as in previous Section with SVR 5.2.1, three best GPR models were chosen out of 588, depending of averaged MAPE values plotted on Figure 5.3.

Chosen models had similar results on all datasets, as can be seen on Figure 5.4,

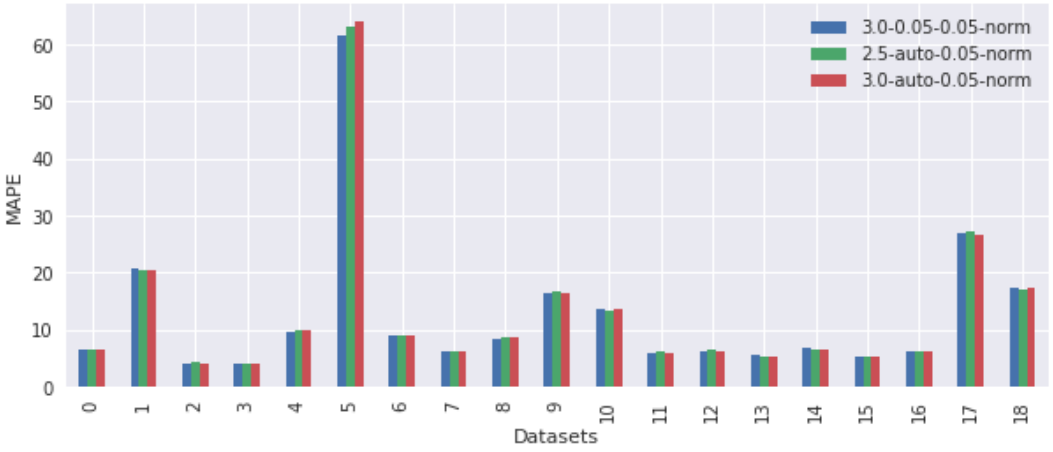

Figure 5.2: MAPE computed on testing data of three best SVR models, depending on dataset. For all three models datasets were in normalized type.

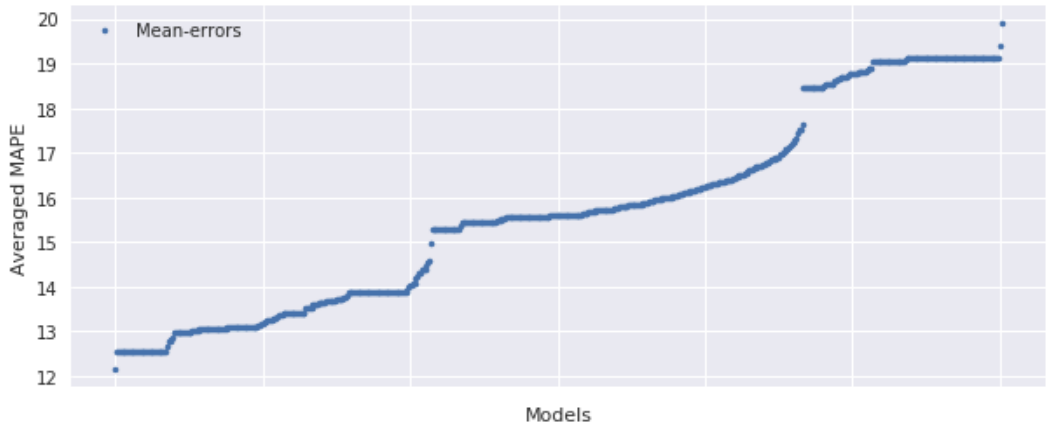

Figure 5.3: MAPE on testing data of all GPR models averaged on all datasets sorted ascending. These values were used to choose three best models.

except datasets number 1 and 5. Especially on datasets number 1 difference was very significant. Parameters of best model can be seen in the Table 5.4.

5.2.3 LSTM networks

Results of LSTM networks from normalized datasets can be seen on the Figure 5.5 and from standard scaled datasets on the Figure 5.6.

Errors on normalized datasets was very big in most of examples, but it was not an effect of overfitting, because error on training samples was comparable. With standard scaled results was much better, but still errors were twice as big as in SVR or GPR models.

To explore problem with neural networks deeper, also Multilayer perceptron model was tried, with similar architecture, i.e. four dense layers with 64 units each

Table 5.4: Best GPR parameters

α	1.5	0.5	0.5
c	1.5	2	1
l	1.5	1	0.05
Average MAPE	12.16	12.54	12.54

Parameters of three best GPR models.

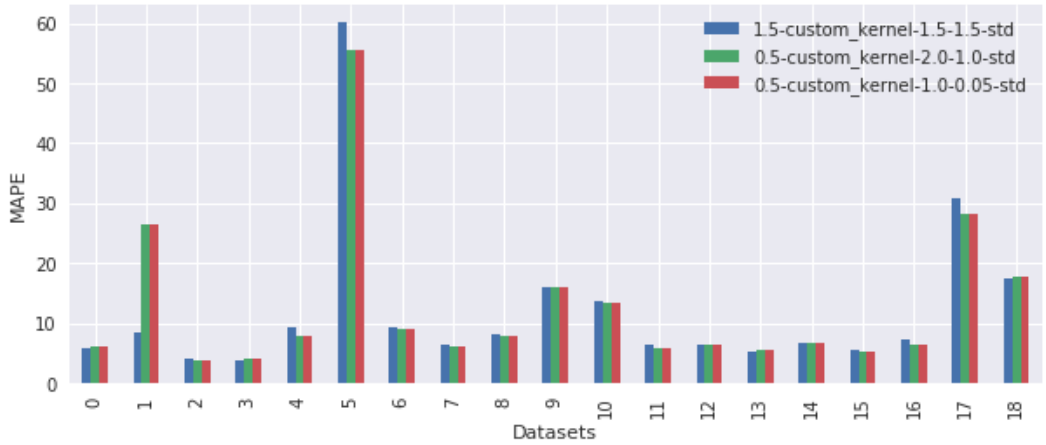

Figure 5.4: MAPE computed on testing data of three best GPR models, depending on dataset. For all three models datasets were in standard scaled type.

and last dense layer with single output. All hidden layers had *Rectified Linear Unit* as activation function. Network was trained with Adam Optimizer for 2000 epochs with batch size equal to 32. Such model had very similar results to LSTM models.

Because of poor results and long time of training, LSTM models were not considered for further experiments with particular datasets.

5.3 Comparing to forecasting by value from previous hour

Regression models from experiments used, among other features, value from previous hour, which could be used also as predictor itself. Therefore it was important for reliability of this thesis, to compare forecasting by value from previous hour with forecasting by machine learning models.

As can be seen on Figure 5.7, on most of the datasets machine learning models had better results, but there were few, where previous hour was more accurate. Most significant difference could be noticed on dataset number 1. This dataset was tested more detailed in next Section.

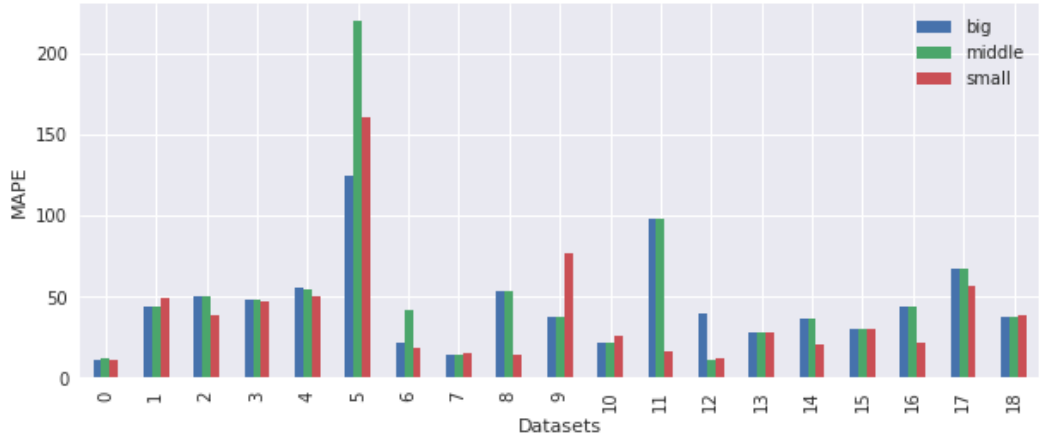

Figure 5.5: MAPE from three LSTM models computed on all datasets in normalized type.

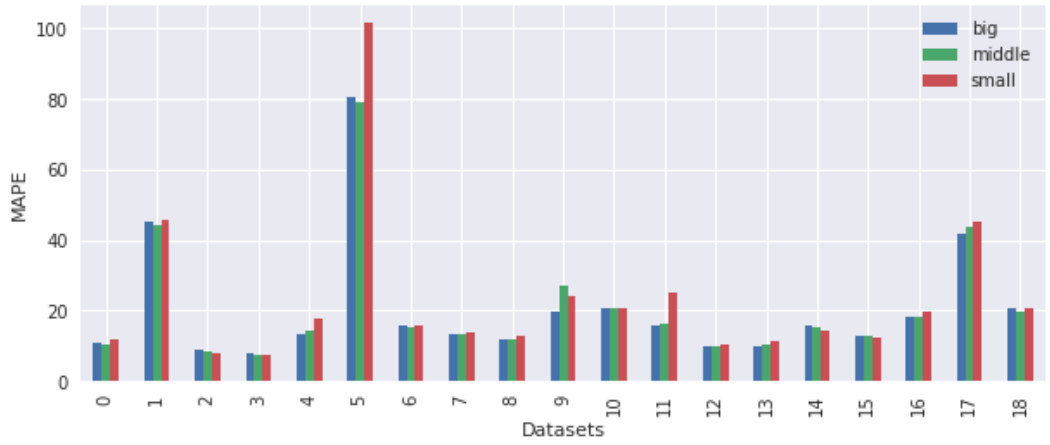

Figure 5.6: MAPE from three LSTM models computed on all datasets in standard scaled type.

5.4 More detailed view on some datasets

On Figures 5.2 and 5.4 can be seen that for dataset number 5 MAPE value was much bigger than for others and that this behavior was similar for all models. In addition, as described in Section 5.3, on dataset number 1, forecasting by value from previous hour would be much more accurate. Therefore, in this Section a more detailed view on these datasets is described.

5.4.1 Dataset with worst result

As can be seen on Figure 5.8, for dataset number 5 forecasting by value from previous hour would not be very accurate, MAPE equal to 72.94%. This could be the reason why all models on this dataset had the worst result, but anyway smaller than 65% for three best SVR models, and one 61% and two even 55% for three best GPR

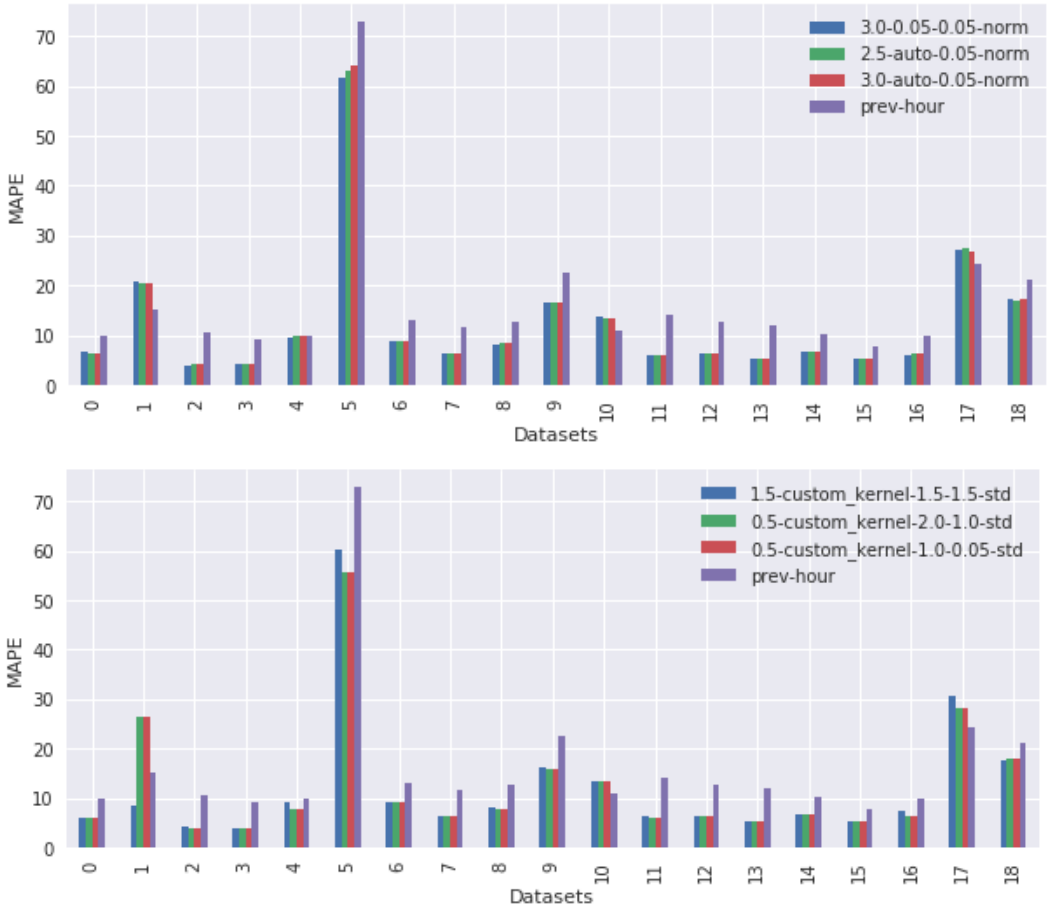

Figure 5.7: On upper plot there are values from SVR models, on the lower from GPR. Blue, green and red columns describe MAPE value of three best models computed on testing data from all datasets. Purple columns describe MAPE computed from forecasting by value from previous hour. On most of the dataset, machine learning models had significant advantage, but there were few, where previous hour had better result.

models.

On the Figure 5.9 one can see that best SVR and GPR models had the biggest errors in places of bigger peaks, which could be desired property for leaks detection. After removing the biggest peak, MAPE from 61.67% became 6.04% for SVR and from 60.17% to 5.29% for GPR.

5.4.2 Dataset with significant previous hour advantage

As can be seen on Figure 5.10, forecasting by previous hour was very accurate on dataset number 1, but probably useless for leaks detection. It can be seen on the plot, that there was one big peak in the series - this could be a leak, but one could not see any anomaly comparing to value from previous hour.

On the Figure 5.11 forecasting by best SVR model is plotted. Here one can

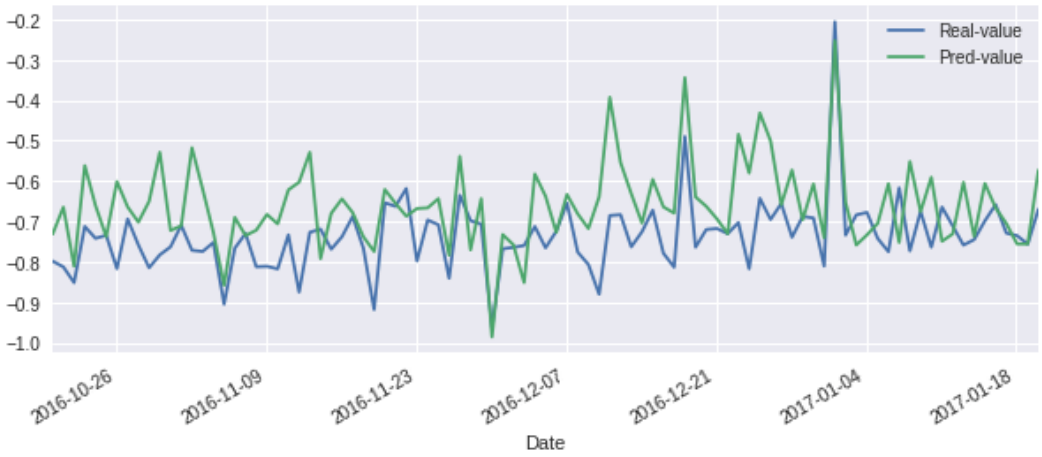

Figure 5.8: Blue line shows real time series values, green shows values from previous hour. There was much difference in most of the points, which could be a reason of inaccurate forecast.

also see accurate forecasting, except this big peak, where much smaller value was predicted. Such difference could be recognized as anomaly and probable leak.

Best SVR error on testing data from this dataset was 20.77%. But after removing this big peak, the error was only 6.86%.

As can be seen of Figure 5.8, best GPR model was significantly better than others for dataset number 1, and even better than forecasting by previous hour, as the only one from best models. Therefore for this dataset two best GPR models were tested more carefully.

As can be see on plot 5.12, best GPR model, similar to forecasting by previous hour, did not notice big anomaly, in contrast to second best GPR model, where anomaly was very clearly visible. Therefore, it turned out, that it is very important to check if models are not overfitted not only by computing forecasting accuracy, but in this application also perceptivity for anomalies detection.

Figure 5.9: Blue line shows real time series values, green shows values from prediction by best SVR model on top and best GPR below. Here the difference between values was smaller than on the plot 5.8, only in places of big peaks error was more significant, which could be a desired property for leaks detection.

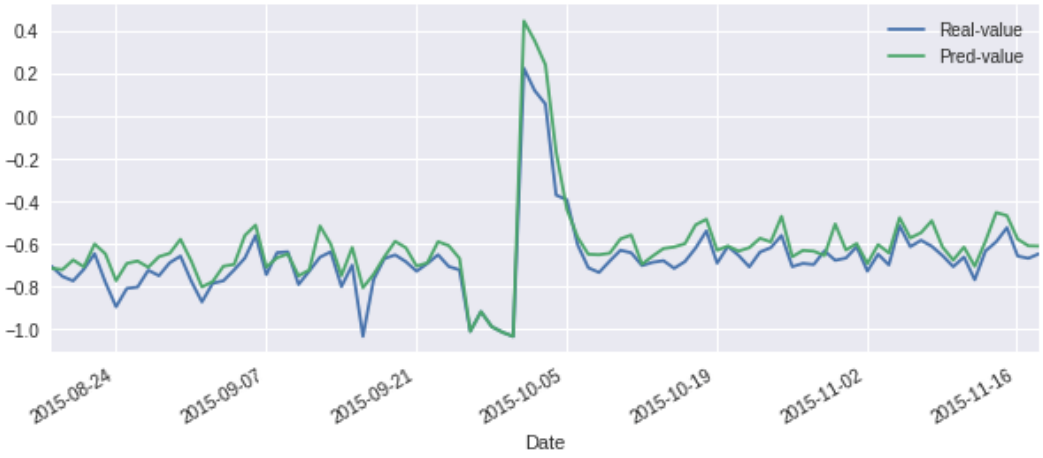

Figure 5.10: Blue line shows real time series values, green shows values from previous hour. It can be easily seen, that lines are very near, even in the biggest peak which is probable leak.

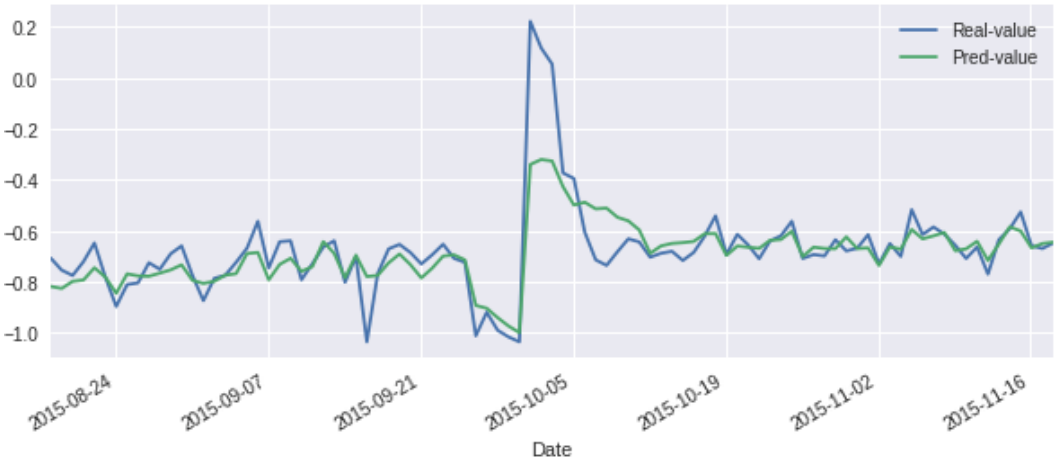

Figure 5.11: Blue line shows real time series values, green shows values from prediction by best SVR model. Here, as on the Figure 5.10 values are very near, except the biggest peak, which is probable leak.

Figure 5.12: Blue line shows real time series values, green shows values from prediction by best GPR model on top and second best below. Upper model, similar to forecasting by previous hour, had well predicted value even for big peak, which could be a leak, therefore lower model was better in this example, because one could see probable anomaly from its prediction.

Chapter 6

Conclusions and Future Work

Experiments described in Chapter 5 showed, that SVR and Gaussian processes models are much better in forecasting water demand than used LSTM neural networks, because they achieved smaller errors and also their training was computationally cheaper.

What is more, Gaussian processes on most of the datasets had significantly lower MAPE level than SVR models and, additionally GPR provides information about probability distribution of every variable in series, which could be useful both in anomalies detection and water demand schedule optimization.

6.1 Leaks detection problem

Both SVR and GPR models were able to detect anomalies in time series, but, as shown in Section 5.4 on GPR example, they could be misled and might predict value close to leakage, in such case anomaly would not be detected. Therefore, it would be useful to have information about points where real leaks happened and use it as validation data.

In such case even more complicated approach could be made. Additionally to water usage forecast, individual classifier could be trained and, with use of estimated and real values, and also some other local information, predict whether there is a leakage or not.

Bibliography

[1] Antonio Candelieri, Ilaria Giordani, Francesco Archetti: *Automatic Configuration of Kernel-Based Clustering: An Optimization Approach.* International Conference on Learning and Intelligent Optimization.

[2] Antonio Candelieri: *Clustering and Support Vector Regression for Water Demand Forecasting and Anomaly Detection.* Water 2017, 9, 224.

[3] A. Candelieri, R. Perego, F. Archetti: *Bayesian optimization of pump operations in water distribution systems.* Journal of Global Optimization 2018.

[4] Robert H. Shumway, David S. Stoffer: *Time Series Analysis and Its Applications.* Springer, Third edition.

[5] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex J. Smola, Vladimir Vapnik: *Support Vector Regression Machines.* Neural Information Processing Systems Conference 1996.

[6] Alex J. Smola, Bernhard Schölkopf: *A tutorial on support vector regression.* Statistics and Computing 14: 199–222, 2004 ©. 2004 Kluwer Academic Publishers. Manufactured in The Netherlands.

[7] C. E. Rasmussen, C. K. I. Williams: *Gaussian processes in machine learning.* The MIT Press, 2006, ISBN 026218253X. ©2006 Massachusetts Institute of Technology.

[8] Sepp Hochreiter, Jürgen Schmidhuber: *Long Short-Term Memory.* Neural Computation 9(8):1735-1780, 1997

[9] OpenStreetMap contributors: `openstreetmap.org`

[10] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E.: *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research 12: 2825-2830, 2011.

[11] Chollet, François and others: *Keras,* 2015. `keras.io`.

[12] Martín Abadi and Ashish Agarwal and Paul Barham and Eugene Brevdo and Zhifeng Chen and Craig Citro and Greg S. Corrado and Andy Davis and Jeffrey Dean and Matthieu Devin and Sanjay Ghemawat and Ian Goodfellow and Andrew Harp and Geoffrey Irving and Michael Isard and Yangqing Jia and Rafal Jozefowicz and Lukasz Kaiser and Manjunath Kudlur and Josh Levenberg and Dandelion Mané and Rajat Monga and Sherry Moore and Derek Murray and Chris Olah and Mike Schuster and Jonathon Shlens and Benoit Steiner and Ilya Sutskever and Kunal Talwar and Paul Tucker and Vincent Vanhoucke and Vijay Vasudevan and Fernanda Viégas and Oriol Vinyals and Pete Warden and Martin Wattenberg and Martin Wicke and Yuan Yu and Xiaoqiang Zheng: *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,* 2015. Software available from `tensorflow.org`.

[13] Diederik P. Kingma, Jimmy Ba: *Adam: A Method for Stochastic Optimization.* arXiv:1412.6980, 2014.