Article

Predicting Heat Meters' Failures With Selected Machine Learning Models

Przemysław Pałasz [1] and Radosław Przysowa [2,*]

1 Rapp Enserv, Hochstrasse 100, 4018 Basel, Switzerland; patryk.palasz@rapp.ch
2 Instytut Techniczny Wojsk Lotniczych, ul. Księcia Bolesława 6, 01-494 Warszawa, Poland; radoslaw.przysowa@itwl.pl
* Correspondence: patryk.palasz@rapp.ch

Version November 13, 2018 submitted to

Abstract: Heat metres are used to calculate the consumed energy in central heating systems. The subject of this article is to prepare a method of predicting a failure of a heat meter in the next settlement period. Predicting failures is essential to coordinate the process of exchanging the heat metres and to avoid inaccurate readings, incorrect billing and additional costs. The reliability analysis of heat metres was based on historical data collected over many years. Three independent models of machine learning were proposed, and they were applied to predict failures of metres. The efficiency of the models was confirmed and compared using the selected metrics. The optimisation of hyperparameters characteristics for each of models was successfully applied. The article shows that the diagnostics of devices does not have to rely only on newly collected information, but it is also possible to use the existing big data sets.

Keywords: machine Learning (ML); artificial neutral network (ANN); bagging decision tree (BDT); SUpport Vector Machines (SVM); no free lunch theorem (NFLT); hyperparameter optimisation; model comparison; heat meter

---

1. Introduction

In many climate zones, heating is the largest cost of operating a building, and its reliable settlement is a crucial issue for social reasons. To account for the consumed heat energy, heat metres are used both in multifamily houses, as well as in single flats or offices heated by heat networks. Smart heat metres ensure that costs are settled based on the actual consumption and contributes to saving heat by inhabitants and to reducing the emission, particularly in buildings with many tenants. A heat meter is a microprocessor measuring device which calculates the consumption of heat in kWh by measuring the flow rate of the heat transfer medium[1] and the difference between supply and return temperature (Fig. 1). The meter has an in-built flow meter, which measures the volume of the flowing medium, and two temperature sensors for the inflow and outflow fluid. Depending on the method applied to measure the volume of the medium, we can distinguish two types of sensors: volumetric and ultrasonic. The first one uses classic volumetric flow meter to calculate the volume of the medium. The meter with an ultrasonic module (Fig. 2) does not measure the volume itself, but calculates it by measuring the velocity of the flowing medium and the pipe's cross-section area [1]. A digital system calculates the amount of heat in kilowatt-hours [kWh] according to the following formulas:

$$\hat{Q} = \hat{m} \cdot c_w \cdot (t_1 - t_2) \tag{1}$$

---

[1] Most often it is water, although in the case when the system is also used for cooling, it can be water with appropriate additives to prevent freezing.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

where $\hat{Q}$ – instantaneous heat, $\hat{m}$ – instantaneous mass, c_w – specific heat of medium, t_1 – input temperature, t_2 – return temperature. In case of counting with a volumetric module, $\hat{m}$ is calculated by the following equation:

$$\hat{m} = \hat{V} \cdot \rho \tag{2}$$

where $\hat{V}$ is the instantaneous volume and ρ is the specific density of medium. If we deal with the ultrasonic meter, then the formula takes the following form:

$$\hat{m} = A \cdot v \cdot \rho \tag{3}$$

where A is the cross-section of the pipe and v is the instantaneous speed. By adding up the instantaneous values of measurements we obtain the values of consumed heat:

$$Q = \int \hat{Q}_t \, \mathrm{d}t \tag{4}$$

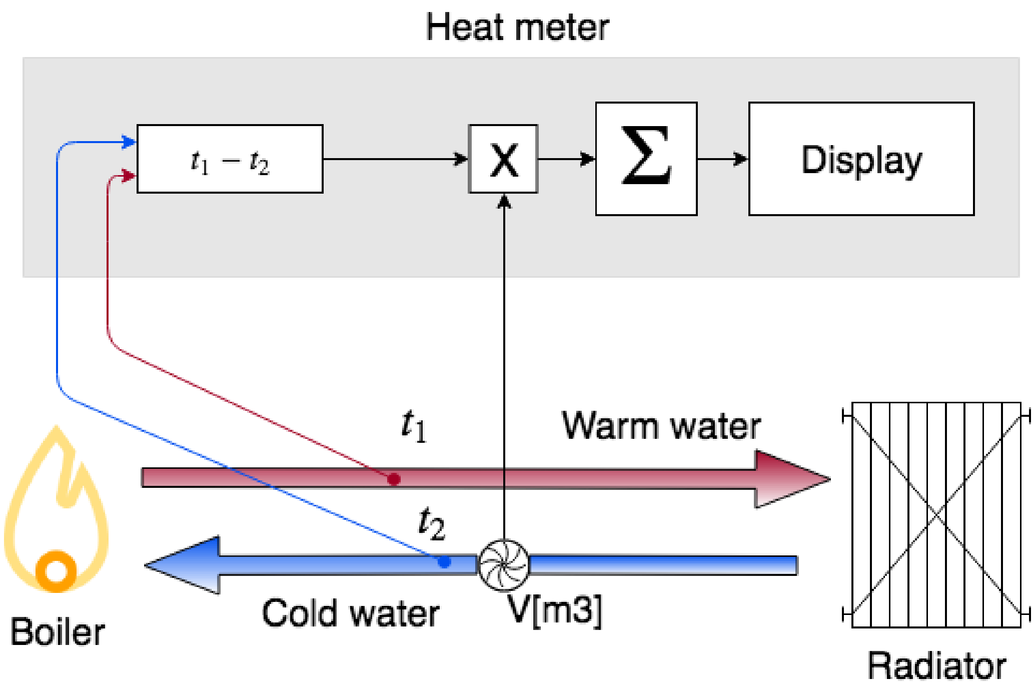


Figure 1. Principle of the heat meter

doi:10.20944/preprints201811.0293.v1

Figure 2. Example heat meter

The nominal lifetime of meters in Switzerland, where our data comes from, is 10 years. In other European countries, such as Germany, France or Poland, it is necessary to calibrate the device every 5 years. In practice, however, there are many cases where heat meters are in operation for more than 10 years. As a member state of the International Organization of Legal Metrology (OIML), Switzerland applies OIML standards and guidelines for the evaluation of technical condition and the calibration of heat meters. In [2], general requirements for metrology, terminology and technical characteristics of heat meters are defined. In addition, based on the maximum permissible error, devices are classified into three distinct categories. The expansion to this document is [3], which describes what test procedures should look like and what values should be obtained, so that a device can be classified appropriately. The procedures for the durability test are also described there. An important element of any heat meter is the control program, which is not mentioned in the OIML test procedures. Unfortunately, a poorly developed driver can also cause incorrect readings or even failure of the meter itself. Papers [4] and [5] present algorithms for testing and assessing the stability of programs controlling heat meters.

The device is exposed to water corrosion and the effects of changes of its temperature and pressure in the central heating system. Heat meters are also subject to damage due to lime deposit from water (especially from warm and hot water), which builds up in the meter and moves with the flow stream through the device causing wear of mechanical parts. Clogging and corrosion of 'windmills' is the most common cause of breakdown of flow-type heat meters. For this reason, ultrasonic meters are considered to be much more durable and reliable. There are fewer problems with electronics or temperature sensors (mainly Pt100 or Pt500 sensors) and in case of breakdown the damaged meter is simply replaced by a new one.

The dismantled meters can be repaired by specialised service centres; they are cleaned, calibrated and sealed, so that they could be put in use again and comply with metrological requirements. The cost of repair is slightly lower from the price of the new meter, especially as regards heat meters with mechanical heat converter. Due to the high degree of wear of movable components and stresses that occur during several years of device's operation, many components of the meter have to be replaced by new ones.

The replacement of meters is a complex logistic process, which must be planned in detail. The old meters have to be removed from the network and delivered to a service centre or destroyed. The installation of new meters is usually done in stages and it is convenient to carry out the service works

 doi:10.20944/preprints201811.0293.v1

in a limited area before proceeding to the next one. It is necessary to arrange meetings with building managers or tenants. The whole procedure can generate high costs related to the work of qualified installers, planning and delivering of materials, final inspection of the installed devices and updating the operational data regarding meters.

Premature meter failures affect all parties, i.e. tenants , building managers and the company clearing the accounts (billing), significantly disturbing the scheduled repair works and causing delays. They may result in incorrect readings, understated invoices and customer complaints. It is more economical to replace a functioning device during a scheduled maintenance check if there are any indications of a possible failure in the next settlement period. Developing a method for predicting the occurrence of the meter failure in the subsequent period is the subject of this article.

In the era of progressing digitalisation, the heat meters, similarly to other devices and sensors, collect data on their operation. The availability of this data has become more widespread, and it encourages in-depth analysis. The following issues and problems are studied: forecasting the consumption of energy, predicting occurrence and location of a failure, analysis of meters' reliability ([6], [7], [8]), or even detecting the occupancy of a flat based on electricity consumption [9]. The mentioned studies of large data sets differ in the used methods and tools.

In forecasting the reliability [10] three approaches can be distinguished:

1. *element stress method*, which is mainly based on theoretical reliability analysis of particular components,
2. *reliability tests* based on tests done in laboratory conditions,
3. *reliability verification* based on the analysis of operational data accumulated during real operations of the already installed meters.

The most accurate and reliable is undoubtedly the third method; however, it involves, in comparison to the first two, collecting, processing and analysing extremely huge volumes of information, the so-called Big Data. Machine Learning (ML) is a natural tool for the analysis of such issues and is widely applied for data from smart meters ([11], [12], [13]). There are various models of ML, which vary in sensitivity to the amount and quality of data, the number of parameters of such data, computational complexity of training algorithms (which has a direct impact on the learning time of the model) or the type of response which we require from the model (classification, regression, pattern recognition). Collecting, preparing and preprocessing of such large amounts of data is not a trivial task and needs an appropriate approach [14].

The previous works on meters focus mainly on the so-called 'smart meters', which most frequently provide large amounts of information concerning the current consumption of electric energy ([15], [16] or [17]). The analysis of this data usually regards the predicted energy consumption ([18], [19]), rarely the reliability of the devices alone ([15], [10]). The tools used for this analysis are usually limited to one selected method or algorithm of machine learning. Due to the certain universality, the most frequently applied ML model is an artificial neural network [18].

The article describes the process of preparing, testing, evaluating and optimising three selected models predicting failures of heat meters, which encompassed:

- preprocessing and preliminary analysis of raw data
- appropriate selection of features
- normalisation of features
- selection of a few independent models suitable to the posed problem
- training models and their evaluation
- hyperparameter optimisation
- re-evaluation and interpretation of results.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

2. Source data and preprocessing

Information on installation, operation and exchange of heat meters was accumulated over the last ten years in a relational database. Operating a meter consists in cyclical reading of its current value necessary to calculate the consumption of energy in a defined settlement period. Potential failure should be detected at the time of meter reading at the latest. The settlement period is usually 12 months long (but there are also periods 6-, 18- or 24-month long) and starts at the beginning of the chosen month (often it is January, June or September). Some modern meters also store the monthly values – although they do not affect the final settlement. After the closure of the period, the data regarding the objects being settled (buildings, flats, meters and such) is copied, and a new version of the same object is created. Such solution has certain advantages when it comes to the business logic of the application; however, as regards the preparation of the data for the analysis, it is a substantial difficulty. The moment the data has been copied, we lose direct continuity of information on the meter and have to reconstruct its history in an iterative or recurrent way (which is not a feature of relational databases).

The discussed database also includes many other items of information used for the settlement of utilities (approximately 150 relational tables – some of them consist of 20 million records – in total, 250 GB of data) as well as data regarding other types of meters (water meters and heat distributors). However, the authors decided to focus on the heat meters elaborated in the introduction. The available data may offer clues to many questions concerning the operation of heat meters. For economic reasons, the most significant problem is detecting and predicting a failure; thus, the data was prepared for this purpose. One of the possible approaches to the problem is statistical analysis, which can reveal some dependencies between certain variables, their mutual relations or potential redundancies. Creating a matrix of correlation and covariance often allows to eliminate the irrelevant features and provide the basis for Principal Component Analysis. Besides, various stochastic models are commonly used to study and analyse the behaviour and durability of the device over an extended period of time. One of such models is the Markov Model, which was constructed and tested for the data mentioned above [20].

Usually preparing data from such a big set is a time-consuming task and cannot be easily automated. According to Forbes [21], data scientists spend up to 60% of their time on the preparation of data and only 20% on building and optimising a model. In our case, not all features of records were found in the catalogue. The work was also hindered by having to use three different languages (German, French and Italian) to describe some parameters of stored objects (the data was collected in Switzerland). Despite the difficulties mentioned above, we managed to distinguish almost 367000 archival records, which corresponds to more than 50000 meters (Fig. 3) along with failures of some of them.

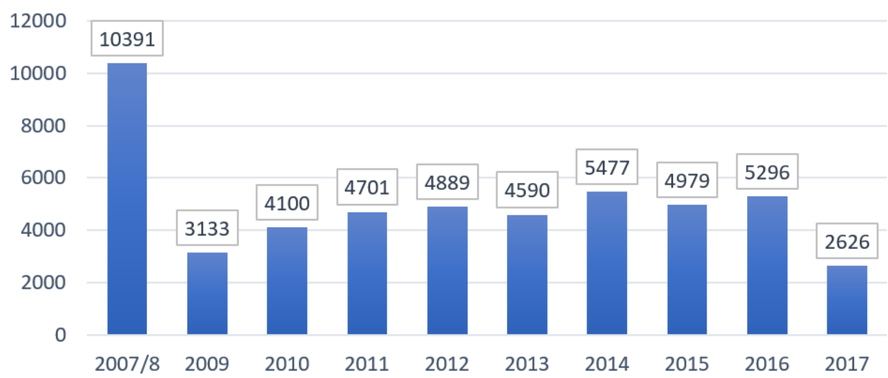


Figure 3. Data collected within 10 years

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

The computer system was introduced in 2008. Therefore, the data spanning the years 2007 and 2008 is cumulated. For 2017 we only have partial data. To facilitate the export, all data was ultimately collected in the single 'Counters' table. Next to the information whether the meter was damaged, we decided to distinguish 16 features, which are depicted in Tab. 1.

Table 1. 16 selected features

Name	Description	Type
Age	Life of the meter in months from the moment of installation to the last readout or failure	ℕ Number
ZIP	Postal code of flat/office, in which the meter is installed	ℕ Number
Floor	Floor where the meter is installed	ℕ Number
FlatType	Type of usable area	String of letters
RoomType	Type of area	String of letters
AccConsumption	Consumption from the moment of registration in the system	ℝ Number
CurrentConsumption	Consumption in the last settlement period	ℝ Number
CommType	Type of communication with the meter	String of letters
Producer	Name of the manufacturer of the meter	String of letters
RatingFactor	Equalization factor	ℝ Number
BillingNo	Next number of the settlement period	ℕ Number
CurrentValue	Last recorded value of the meter	ℝ Number
AvgConsumption	Average consumption in all settlement periods (historical data)	ℝ Number
MaxConsumption	Maximum consumption in the settlement period (historical data)	ℝ Number
MinConsumption	Minimum consumption in the settlement period (historical data)	ℝNumber
CalculatedAge	Calculated age of the meter in months: CurrentValue/AvgConsumption	ℝ Number

The Swiss postal codes consist of four digits, and places located next to each other have very similar values of these codes. Thank to this, we have indirect information on the geographical location of the examined meter.

The floor where the meter is located was introduced into the system based on the convention, that is '04.02' format should be read as the fourth floor, 2nd flat from the left. Conversion of such information to an integer value is relatively easy providing that data is reasonable (e.g. '92.04' is a wrong value because the highest building in Switzerland has 41 floors). Sometimes, to denote the ground floor the 'EG.02' format was used and to denote the basement or floors below the ground floor '-01.02' or 'UG.02' format was used. In all cases when the conversion was impossible or did not make sense, the authors left this value empty.

The type of usable surface (e.g. flat, office or storeroom), as well as the type of room (e.g. room, kitchen, bathroom and corridor), were represented by enumerations. In case of an unclear situation, the value *other* was used. The authors decided to distinguish sixteen types of rooms and five types of usable surface. Due to the fact that some models accept only numbers, each type of room was assigned a successive natural number. The same was done in the case of meters' manufacturers (16 different values). *Rating factor* is a real number from the interval (0-1]. It is used in case of rooms which are characterised by increased consumption of heat, e.g. due to adjacency to external walls of the building. This ensures a fair distribution of the heating costs of the whole building between all tenants – irrespective of the fact whether they have an external flat or not.

An important parameter is also the method of communication with the device (*CommType*). There are four communication types:

- *bus* – meters regularly send their updates to the central panel installed in the same building which also collects data from other meters – cable connection
- *funk* – similar as bus, but the connection of the meter with the control panel does not require the additional wiring system
- *walk by* – on specific days (programmed) the meter sends data, which has to be collected by the technician sent to the neighbourhood and equipped with the receiving device

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

- *without a module* – the reading has to be done *manually* directly on the meter

After entering all data into the *Counters* table, we removed all the records which had unlikely (negative or a few orders of magnitude too big) values in *CurrentValue*, *MinConsumption* and *MaxConsumption* fields.

Despite the availability of the data regarding the users of flats or offices, the authors decided to ignore them, because the change of a flat's tenant should not impact the lifetime of a meter. The database also contains information on the size of the usable surface, but it is incomplete and not always up-to-date; that's why it was not added to the parameters list either.

As we see, despite the availability of big historical data sets, the task to prepare it for the analysis is not trivial and often involves the cooperation of experts in a given area.

3. Machine learning

Recently applications of machine learning in technology have been proliferating, which is even visible in everyday life, e.g. browsing the Internet, filtering spam, detecting frauds, image recognition, predicting heart diseases or algorithms of artificial intelligence in online games. The problem addressed in this paper is a typical problem of binary classification. Based on the vector of data (16 features) we are trying to answer the question whether a given meter will fail or not in the next settlement period. Formally, this problem can be defined in the following way. For a given set of training data $\mathcal{D}$

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{l} : x_i \in \mathcal{X} \subset \mathbb{R}^d \wedge y_i \in \{-1, 1\} \tag{5}$$

we construct an algorithm A (model), which for a given $x \notin \mathcal{X}$ 'correctly' calculates the value $y = A(x)$ and predicts whether the meter with parameters x will break down $(y = 1)$ or not $(y = -1)$. In our case, l (cardinality of dataset) consists of more than 50000 records, but the size of the vector of data d equals 16. As regards expectations concerning the accuracy of predictions of our model it is important to note that it is a generalisation made based on a small sample of all possible combinations of a certain big domain. It was aptly described by George Box in [22], where he wrote: *All models are wrong, but some are useful*. In addition, there is no single universal model that works best in all conditions. Depending on the domain, the amount of data or the type of task, some types of algorithms work better and others worse. This fact is known as *No free lunch theorem* and was formulated by David Wolpert [23]. The consequence of this is the necessity of building various types of models and their correct evaluation ([24], [25], [26]). Due to this, the authors of this paper decided to test 3 independent and substantially different algorithms of machine learning: Support Vector Machine (SVM), Artificial Neural Network (ANN) and Bagging Decision Tree (BDT).

3.1. Performance comparison metrics

Before we start to build models we have to know how to estimate their performance. It will also enable us to compare these models both before and after optimisation. The starting point is the so-called confusion matrix, which is calculated on a testing set[2]. In case of a binary classification, the matrix is 2x2 and includes the following fields: TP – the number of expectations that are true positives, FP – the number of expectations that are false positives, TN – the number of expectations that are true negatives and FN – the number of expectations that are false negatives [24]. The easiest metric is *accuracy* defined as

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

It gives us a general idea of the model's performance, though sometimes it may be inadequate or misleading. It concerns in particular the unbalanced data, where the number of occurrences of one

---

[2] Testing set was previously in no way used to learn a model

 doi:10.20944/preprints201811.0293.v1

class is much more frequent than of the second class. Thus, in addition to *accuracy*, two complementary metrics

$$precision = \frac{TP}{TP + FP} \tag{7}$$

and

$$recall = \frac{TP}{TP + FN} \tag{8}$$

– the latter sometimes also referred to as *sensitivity* [27], are also defined. We certainly want both these metrics to be as close to unity as possible. However, it turns out that we have to seek a compromise between these two. It is due to the fact that increasing the sensitivity of the model to the positive class (by decreasing FP), we automatically reduce the predictability for the negative class (by increasing FN). The metric which combines *accuracy* and *recall*, is the metric

$$f_1 = \frac{2 \cdot precision}{precision + recall} \tag{9}$$

(harmonic mean). Another very popular metric we are going to use is AUC. It is defined as the area under the Receiver Operating Characteristic curve. It takes values between 0 and 1, whereby 0.5 means a random classifier. This metric is crucial as it enables measurement of the ability of the model to distinguish between classes [28]. The metrics described so far focus only on the positive class[3]. If we want to achieve the estimation of the model for both classes at once, we can use the Matthews Correlation Coefficient metric:

$$MCC = \frac{TP \cdot TN + FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

which has this additional property that it is insensitive to unbalanced data [28]. MCC takes values between -1 (worst model) and 1 (best model). As we see, there are a few methods for comparison and estimation of different models. The selection of the appropriate metric (or metrics) shall depend on the quality of data, its balancing as well as the goals to be achieved by the model, that is whether the minimisation of FP or maximisation of TP is more important. Due to the character of the data (high predominance of records without failure – approximately 80%), as well as the goal of the model (equally important as predicting failure is predicting whether the meter will continue working), the authors of this work used two metrics: AUC and MCC. For the more comprehensive image we will provide also accuracy, precision, recall and f_1 for both classes.

3.2. Testing environment

In our work we used the Python language, which offers advanced tools for machine learning and data analysis. The vital feature of this language is its objectivity, code readability, independence from the operating system and availability of specialist libraries for linear algebra and artificial intelligence. The authors, to train and evaluate the selected models, used the following components and applications:

- Python 3.6.3
- Keras 2.2.0 – Open Source library for creating neural networks; it works with the application of one of 3 libraries/engines for linear algebra: TensorFlow, Microsoft CNTK or Theano
- TensorFlow 1.8.0 – Open Source library written by Google Brain Team for linear algebra and neural networks
- Scikit-learn 0.19.1 – advanced library implementing many different methods of machine learning
- Matplotlib 2.2.2 – library for creating charts and graphics in Python

[3] The same calculation can be done for the negative class, but we we will still have information only on one class

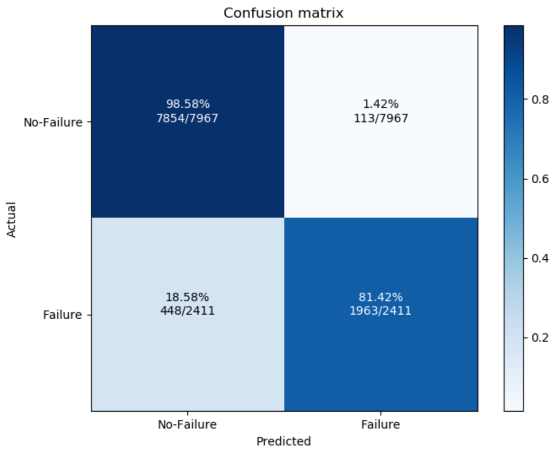
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

- Visual Studio 2017 – integrated development environment supporting various programming languages, including Python

All tests were performed on the computer equipped with the Intel Core i7-6820HQ@2.70GHz processor, 16.0 GB RAM, Toshiba SSD PCIe M.2 512GB, with Windows 7 operating system. Building and evaluation of the models were always based on the same dataset (51890 records), randomly divided into the training set (80% - 41512 records) and the testing set (20% - 10378 records). Because we managed to reduce the number of input data to tens of thousands, the above configuration proved sufficient to train most modern ML models. Only hyperparameter optimisation is so demanding in terms of resources that it can take several hours, but for each model we need to do it only once.

3.3. Support Vector Machines

Support Vector Machine (SVM) is a classifier whose learning aims to determine the hyperplane separating particular classes in order to maximise the margin, that is the distances between this hyperplane and the nearest point of each class, the so-called support vector ([26], [29]). To solve problems which are not linearly separable, we use the so-called *kernel-trick*, which transforms the data space in such a way that it becomes linearly separable ([26], [12]). Kernel functions commonly used include low order polynomials, Radial Basis Function (*rbf*) or *sigmoid* function [30]. A thoroughly significant feature of this classifier is high resistance to overfitting [31]. Unfortunately – SVM training for a considerable amount of data is relatively slow – the computational complexity of this algorithm, depending on the used algorithm, ranges between $O(n^2)$ and $O(n^3)$. For the purpose of heat meters failure forecasting analysed in this paper, we applied a standard SVC (Support Vector Classifier) class from the *Scikit-learn* library with default parameters: penalty parameter=1.0, kernel=*rbf*, gamma = $1/\#$ of features $(1/16)$. The results obtained for this model are presented in (Fig. 4).



(a) Confusion matrix

Metric	Result
Accuracy	94.57%
$Precision_0$	94.72%
$Precision_1$	93.97%
$Precision_{avg}$	94.55%
$Recall_0$	98.49%
$Recall_1$	81.13%
$Recall_{avg}$	94.57%
$f1_0$	96.57%
$f1_1$	87.08%
$f1_{avg}$	94.43%
AUC	89.81%
MCC-S	92.02%

(b) Metrics (avg. 10 tries)

Figure 4. Simple SVM with default parameters

As you can see, we can determine quite well (98.58%) whether the heat meter will survive, but it is much more difficult to predict its failure (81.42%). For the overall estimation of the model, we used metrics mentioned in 3.1. (see Fig. 4b)

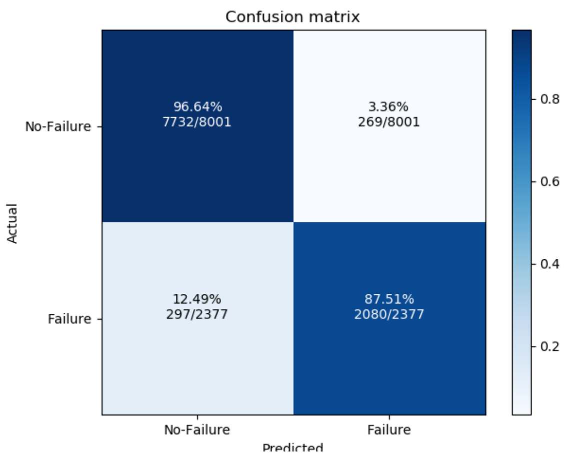
Given that it is a basic model, we obtained a reasonably good result. Further optimisation will undoubtedly improve the results for the class 1, that is predicting a failure.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

3.4. Neural Networks

Neural Networks are currently one of the most popular tools in machine learning. They are applied in almost all problem types: starting from the classification, through regression and pattern recognition to reinforcement ([32], [33], [34]). Thanks to the increase of the computational power, we are no longer so much limited by the number of layers of hidden networks when designing a network – hence the popularity of the DNNs (deep neural networks). The application of specialist filters and image data preprocessing enabled CNNs (convolutional neural networks), which work well in handwriting recognition or image classification ([35], [36]). However, a considerable drawback of neural networks is the interpretation of learning results. The obtained model is frequently so complicated that, even though it gives correct results, we are not able to explain the reasons for such predictions [37].

To construct and train a simple neural network we used the *Keras* library with the *TensorFlow* engine. Our network had the following parameters: input layer (16 features), two hidden layers each with 32 neurons, output layer (binary) with sigmoid activation function. As the optimisation function we used *adam* (adaptive momentum) with default parameters, and as the objective function - binary cross entropy. Learning took 20 epochs. The obtained results are compiled in Fig. 5.



(a) Confusion matrix

Metric	Result
Accuracy	94.78%
$Precision_0$	95.59%
$Precision_1$	91.75%
$Precision_{avg}$	94.73%
$Recall_0$	97.78%
$Recall_1$	84.47%
$Recall_{avg}$	94.78%
$f1_0$	96.67%
$f1_1$	87.94%
$f1_{avg}$	94.70%
AUC	91.13%
MCC-S	92.38%

(b) Metrics (avg. 10 tries)

Figure 5. Simple ANN with default parameters

The results are a bit better than in case of SVM, but also here predicting failure is rather poor. *Keras* provides also information on the learning process of a network in particular epochs. (Fig. 6). As we see – metric accuracy does not differ substantially from the result for the testing set (0.9478 vs 0.9471), which indicates that we do not deal here with the problem of overfitting.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

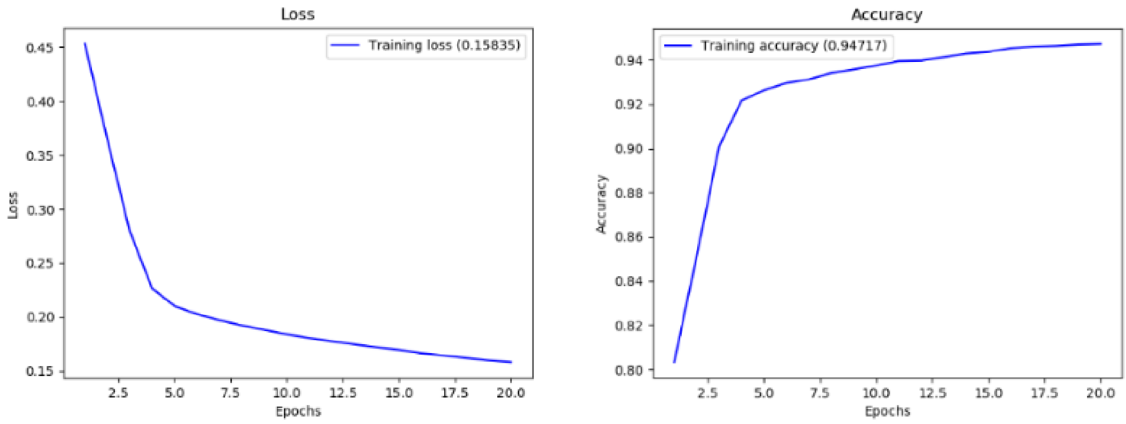
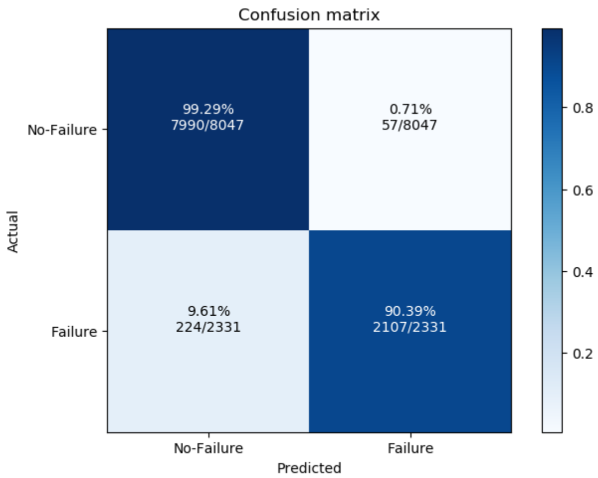


Figure 6. Learning curves for simple ANN with default parameters

3.5. Bagging Decision Trees

Bagging Decision Trees is in fact a meta-classifier which creates a set of decision trees and, on the basis of partial results, it determines the final result of the model. Each decision tree is constructed on a specially selected subset of the training set [24]. There are a few techniques describing how to create subsets for learning and how to calculate the final result. The authors decided to use the Bagging (Bootstrap Aggregating) technique, which partly reduces the problem of overfitting ([38], [39]). It has to be mentioned that decision trees are very fast, that is why they are often used as an introduction to problem analysis. Besides, this model is much easier to interpret and allows to draw conclusions and to build knowledge base from the examined data. The disadvantage of this algorithm is the reduced susceptibility to optimisation – which we will show in the next chapter.

Scikit-learn shares the standard BaggingClassifier class with the default Decision Tree estimator. The results for ten estimators (each for ten trees) are presented in Fig. 7.



(a) Confusion matrix

Metric	Result
Accuracy	97.34%
$Precision_0$	97.27%
$Precision_1$	97.59%
$Precision_{avg}$	97.35%
$Recall_0$	99.35%
$Recall_1$	90.44%
$Recall_{avg}$	97.34%
$f1_0$	98.30%
$f1_1$	93.88%
$f1_{avg}$	97.30%
AUC	94.89%
MCC-S	96.15%

(b) Metrics (avg. 10 tries)

Figure 7. Simple BDT with default parameters

The results are surprisingly good – even failure detectability is at 90%. Fig. 8 shows the comparison of the current results of 3 models – before optimisation. As we see, BDT is the most successful and SVM – the least effective. All models are much better at predicting meter's survival in the next settlement

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

period rather than at detecting its failure. Even though we haven't optimised the examined models yet, the results are excellent. This is mainly due to preprocessing and data normalisation as well as the appropriate selection of features.

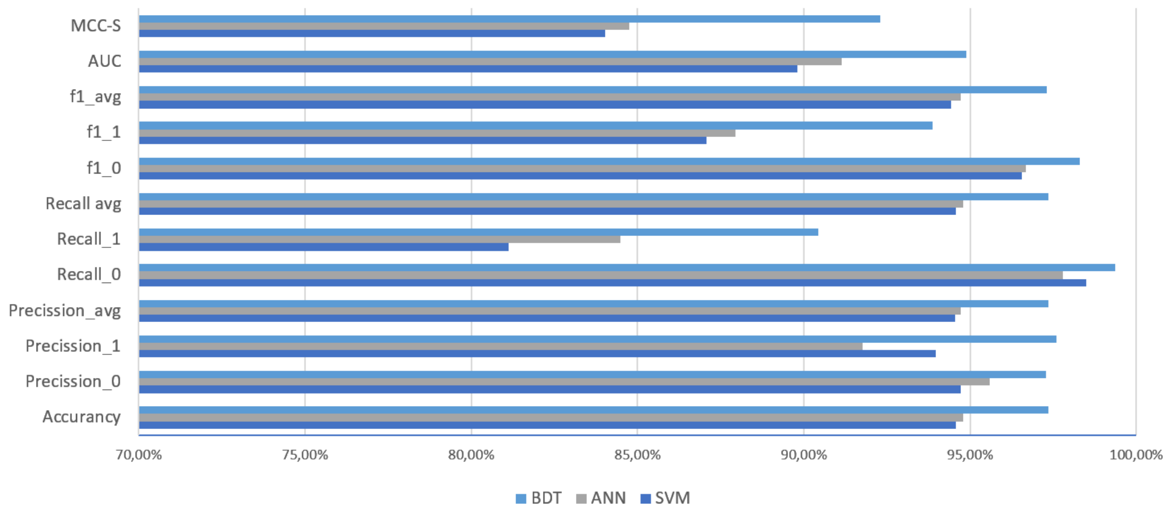


Figure 8. Compared models' performance before optimisation

4. Model optimisation

Even though the results obtained in the first approach were satisfactory, we will show that it is possible to further improve them. Prior to the beginning of the learning phase, a preliminary data analysis is often conducted in order to limit and/or select the most valuable features.This analysis also enables rejection parameters which are irrelevant or have a minimum impact on the result ([27], [40]). Apart from that, almost every mode in machine learning has specific input parameters which determine its operation – the so-called hyperparameters. They affect not only the speed of learning, but also the quality of the model after the learning phase or its susceptibility to overfitting/underfitting ([41], [42]).

4.1. Features selection and dimensionality reduction algorithms

The reduction of dimensions is a procedure which allows to reduce the number of input features and keep only those parameters which are necessary to define the data set. Such reduction can be achieved by eliminating the insignificant elements (Principal Component Analysis, Autoencoders) or by reducing the dimension of the features space (Linear Discriminant Analysis), where our classes remain equally well separable (Linear Discriminant Analysis) - ([43], [44], [27]).

In our study, we used the built-in PCA function of *Scikit-learn* with the implementation of a probabilistic model [45]. It turned out that even the minimum decrease from 16 down to 15 parameters resulted in a considerable deterioration of the quality of all three models – see Tab. 2.

Table 2. Results after PCA: 15 features left

Model	AUC Diff	MCC Diff
SVM	-1.03%	-2.20%
ANN	-2.98%	-1.92%
BDT	-10.12%	-8.11%

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

It means that all parameters have a significant impact on the result of the prediction. It is visible especially in case of the BDT model. It can be concluded that feature selection has the more significant influence on examined models, the larger is the size of input data or when the parameters are strongly dependent on each other and partially redundant. It is well visible in the problems of image recognition, where we deal with spaces of thousands of dimensions. In our case, the optimisation of hyperparameters played a much more important role in the optimisation of models – what we will show in the next chapter.

4.2. Hyperparameter optimisation

Optimisation of hyperparameters is a problem of finding a minimum of a certain objective function, the domain of which is the space of parameters of the examined model. The parameters can be continuous, discrete or categorical and additionally they can be dependent on each other [42]. It is worth highlighting that calculating the objective function is extremely expensive – it involves the full training and evaluation of the model.

There are different strategies of looking for optimum hyperparameters. The easiest way is 'manual' tuning. However, it requires expert knowledge of the model and the data, which does not foster generalisation. The other strategy is either full or random search of parameters' domain, the so-called 'grid search'. Checking all combinations is usually unrealistic due to the high costs. It has been confirmed that random search can work well in the case of a model with many parameters, out of which only some play a key role in its quality [46]. The next method of searching for optimum parameters of a classifier is a SMBO (Sequential Model-Based Optimisation) method. To put it simply, it consists in constructing a surrogate model approximating the objective function, the minimum of which we look for. Most frequently GP (Gaussian Process), RFR (Random Forest Regressions) or TPE (Tree-structured Parzen Estimator) are used as surrogate models.

The selection of subsequent domain points (values of hyperparameters) is calculated in a way to optimise the selection function – here we most frequently use the EI function (Expected Improvement). Such a strategy usually provides the best results and eliminates the element of randomness [44].

To optimise the hyperparameters of models described in this paper the authors decided to use SMBO with TPE model. As the objective function, the AUC metrics was applied. Since training of the SVM model was the most time-consuming, we conducted for it only 100 iterations. In spite of this, we managed to find parameters, for which the value of the AUC metric reached almost 94%. Training a TPE model for SVM was shown in Fig 9.

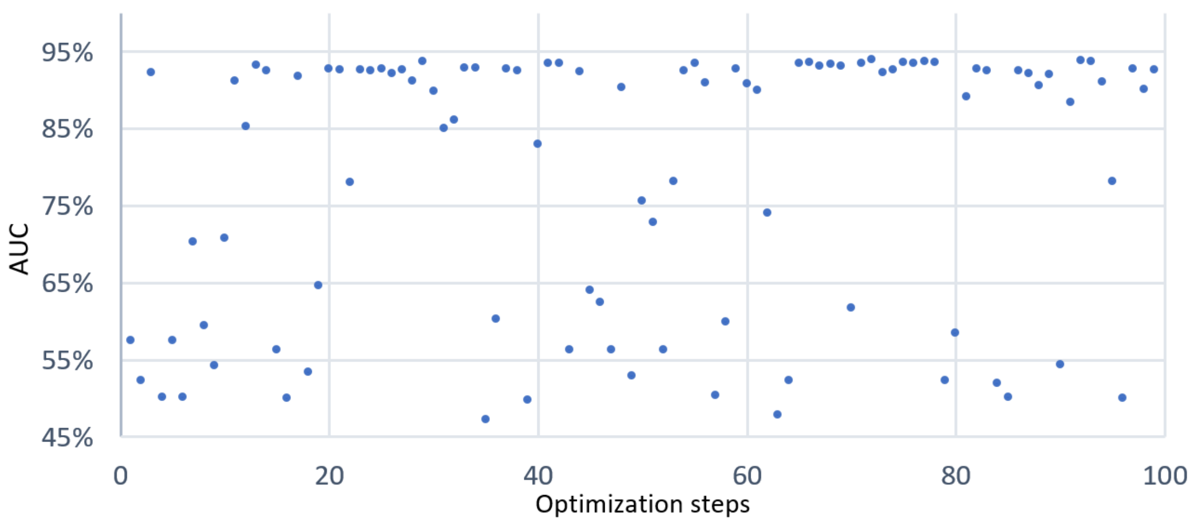


Figure 9. Hyperparameter Optimisation for SVM

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

In case of a neural network, the number of hyperparameters was slightly higher. The algorithm of learning is much faster, that's why we managed to conduct 200 iterations in the comparable time and we found a configuration for which the value of AUC reached 99.28%. Learning a TPE model for ANN was demonstrated in Fig. 10.

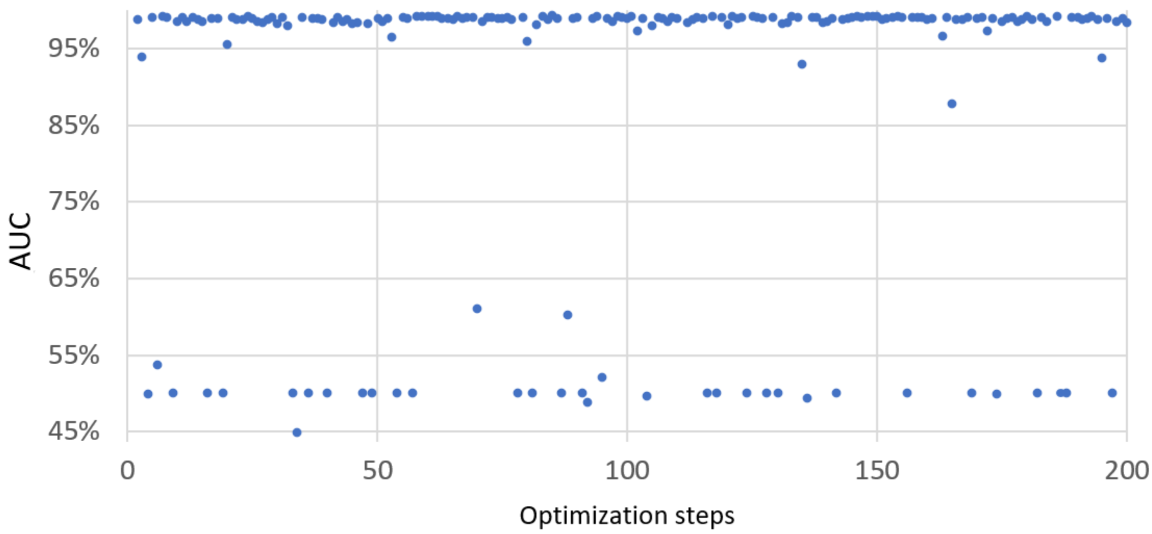


Figure 10. Hyperparameter Optimisation for ANN

BDT model has only three hyperparameters, and already in the basic version it obtained outstanding results. Thus, the optimisation itself did not contribute to significant improvement (max. 95.28%) – Fig. 11.

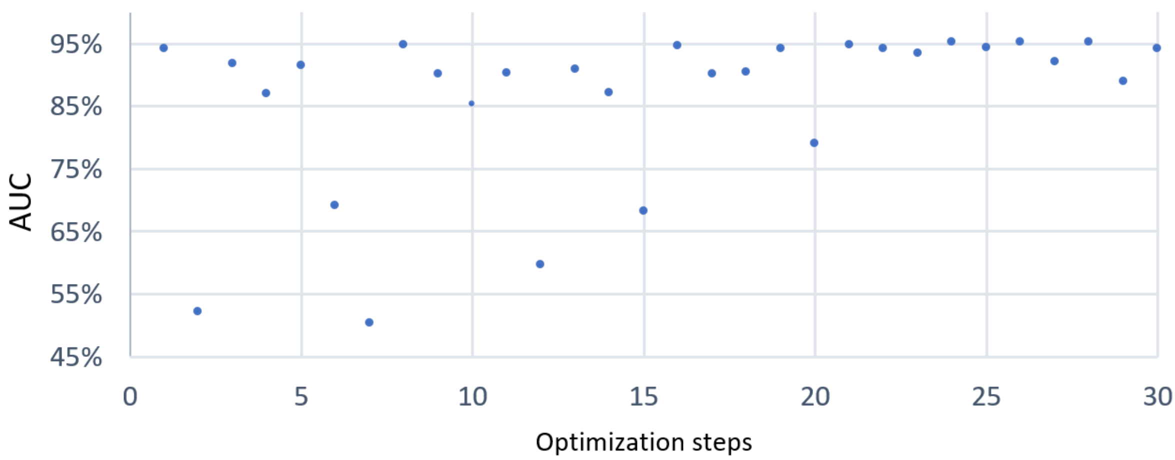


Figure 11. Hyperparameter Optimisation for BDT

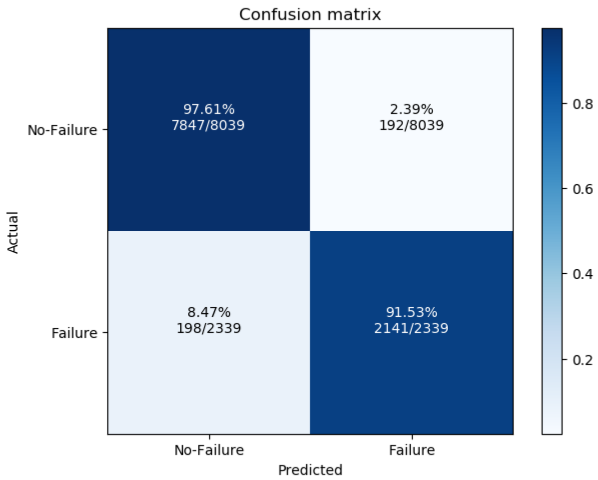
5. Results after optimisation

After finding the optimal hyperparameters, we carried out the same tests as in chapter 3. Below we present the results.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

5.1. SVM optimised

The optimised SVM model has the following hyperparameters: penalty parameter C: 50, class_weight=*balanced* (automatically adjust weights inversely proportional to class frequencies in the input data), kernel function: *rbf*, gamma: 0.1717.



(a) Confusion matrix

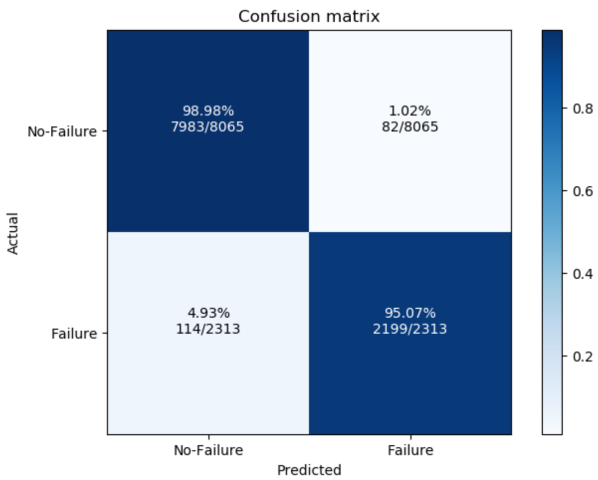
Metric	Result
Accuracy	96.80%
$Precision_0$	97.10%
$Precision_1$	95.70%
$Precision_{avg}$	96.78%
$Recall_0$	98.81%
$Recall_1$	90.00%
$Recall_{avg}$	96.80%
$f1_0$	97.94%
$f1_1$	92.76%
$f1_{avg}$	96.76%
AUC	94.40%
MCC-S	95.39%

(b) Metrics (avg. 10 tries)

Figure 12. Optimised SVM

5.2. ANN optimised

The optimised model of a neural network had the following hyperparameters: 4 hidden layers – each consisting of 112 neurons, activation function: *softsign*, optimisation function: *adadelta*, objective function: *binary crossentropy*, batch size: 158, number of epochs: 123.



(a) Confusion matrix

Metric	Result
Accuracy	97.89%
$Precision_0$	98.34%
$Precision_1$	96.33%
$Precision_{avg}$	97.88%
$Recall_0$	95.95%
$Recall_1$	94.24%
$Recall_{avg}$	97.89%
$f1_0$	98.64%
$f1_1$	95.27%
$f1_{avg}$	97.88%
AUC	96.60%
MCC-S	96.96%

(b) Metrics (avg. 10 tries)

Figure 13. Optimised ANN

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1

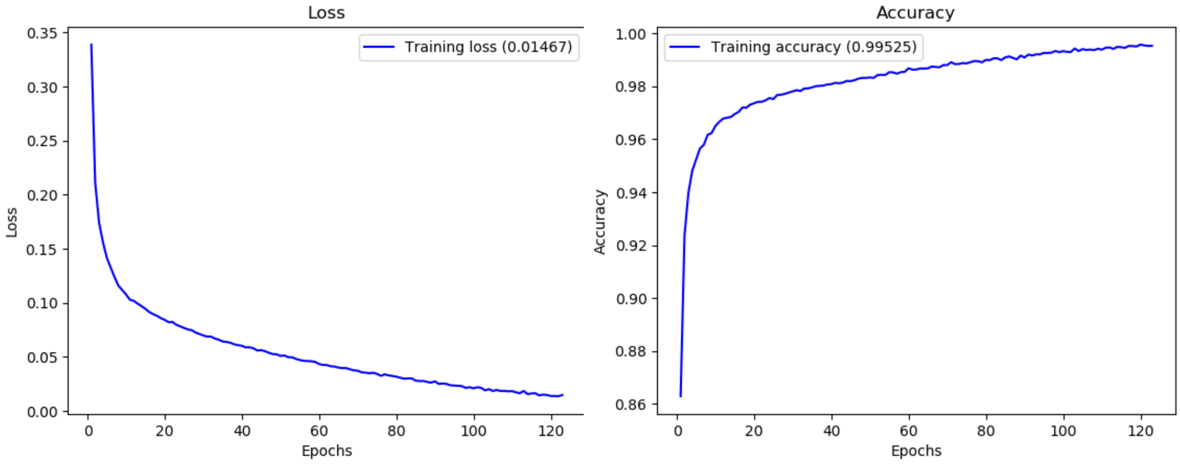


Figure 14. Learning curves for optimised ANN without Dropout

Comparing the metrics' results *accuracy* calculated on the testing set with metrics accuracy in the learning phase, one can observe small overfitting: 99.52% vs 97.89%. For this reason, the authors decided to modify this model and add four alternate *Dropout* layers with parameter 0.3. It is a standard way to minimise overfitting in neural networks [47]. The averaged outcomes of the network improved in this way are only slightly (negligibly) weaker than the original network; however, we managed to limit overfitting (96.99% vs 97.45%) – Fig 15.

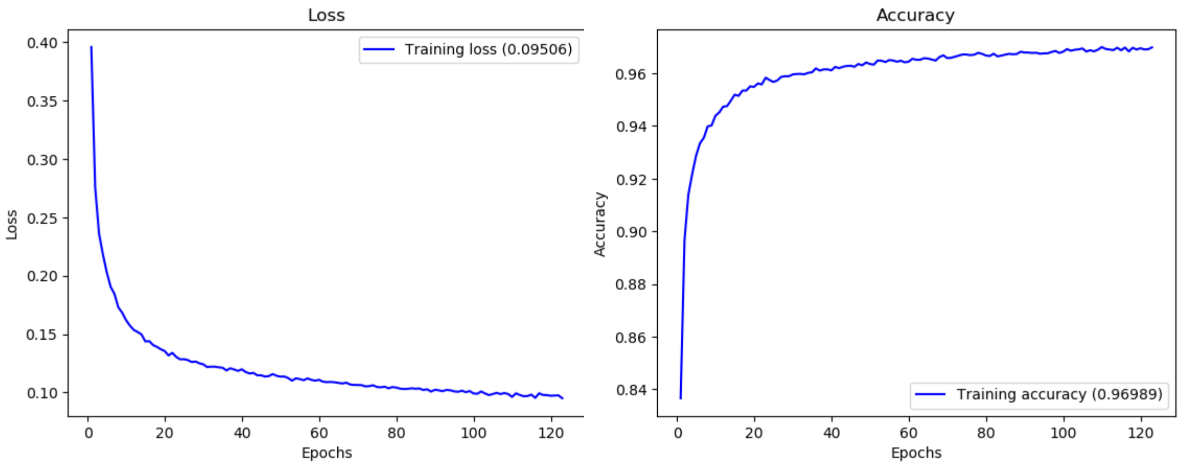
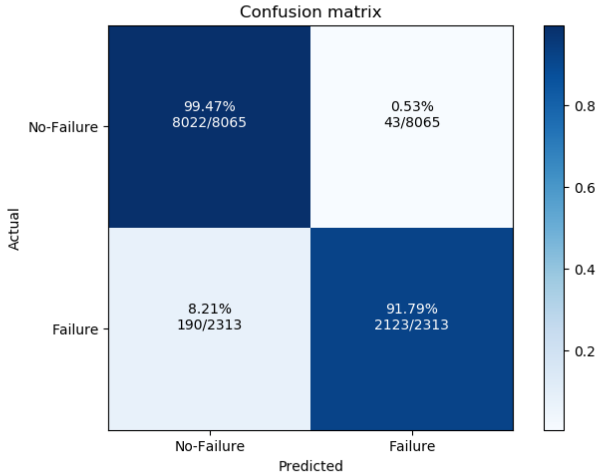


Figure 15. Learning curves for optimised ANN with Dropout (0.3)

5.3. BDT optimised

The optimised BDT model had the following hyperparameters: number of estimators: 20, max features: 0.95, max samples: 0.95. As we see, the sheer number of parameters limits the possibility of optimisation. Besides, having obtained excellent results already in the first stage, we do not observe here such significant progress as in two previous cases. Nevertheless, a small improvement for class 1 can be noticed (Fig.16).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 13 November 2018 doi:10.20944/preprints201811.0293.v1



(a) Confusion matrix

Metric	Result
Accuracy	97.54%
$Precision_0$	97.54%
$Precision_1$	97.68%
$Precision_{avg}$	97.54%
$Recall_0$	99.37%
$Recall_1$	91.25%
$Recall_{avg}$	97.54%
$f1_0$	98.43%
$f1_1$	94.35%
$f1_{avg}$	97.51%
AUC	95.31%
MCC-S	96.44%

(b) Metrics (avg. 10 tries)

Figure 16. Optimised BDT

5.4. *Comparison of optimised models*

In our case, feature selection and dimensionality reduction had almost no influence on models' performance. Nonetheless, this does not mean that such analysis should not be conducted – quite the contrary. Many examples show that it is a significant step toward building and optimising a ML algorithms. With regard to hyperparameter optimisation, each model improved its performance, especially in failure detection. As a matter of fact, it was our primary objective which we managed to attain. It can be observed that the larger is the space of hyperparameters of the investigated model, the better it can be optimised. The most significant progress in each metric was recorded by the neural network, the smallest – by the BDT. The considerable progress is also noted in the SVM model. In metrics selected by us (MCC and AUC), there is progress in the range of 3-5% which is a very good outcome, especially taking into account that the performance of the examined models before optimisation was above 92%. We demonstrated the differences in particular metrics for investigated models in Fig. 17.

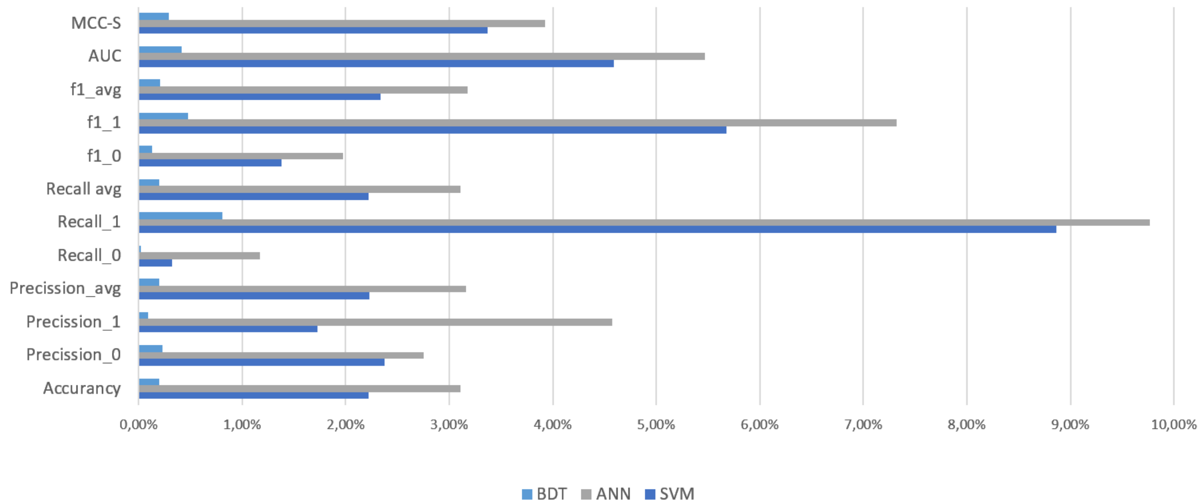


Figure 17. Increase in the value of metrics as a result of optimisation

 doi:10.20944/preprints201811.0293.v1

6. Conclusions

All models developed in this paper have a very high degree of predictability of failures of examined heat meters (Fig. 18). Taking into consideration the NFLT, we chose the models so that they could be as diverse as possible. Hyperparameter optimisation not only improved the general results, but also the failure detection rate for each of the algorithms. In our case, all three models accomplished the given task – though it is not a rule, but rather an exception.

The neural network gives the best results. It has the advantage over other algorithms that it can, instead of a binary response, calculate the probability of affiliation to a particular class, that is the risk of the occurrence of failure expressed in the real number (fuzzy logic). It allows the user to control the threshold of the sensibility of the model. Thresholds higher or lower than 0.5 can be more suitable for operators, which are more or less prone to risk, e.g. in the case of a region with expensive labour.

Even though the BDT and SVM models proved slightly weaker, their simultaneous training is still reasonable. It can turn out that for new data, e.g. the SVM model will perform much better. We demonstrated that the access to historical data would not suffice to construct the functioning model.

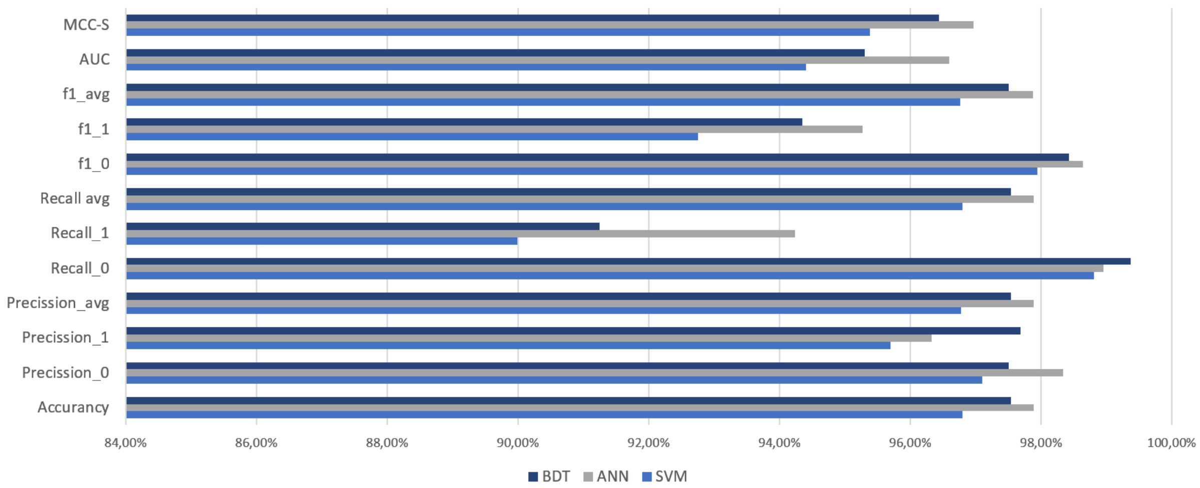


Figure 18. Final results for all models

This article is one of the few which deal with reliability and predictability of heat meters' failures. It is also, according to our knowledge, the first attempt to build three independent ML models based on a single database. Achieving the result above 95% for the AUC metric by the model, while maintaining overfitting at the minimum level, is a remarkable outcome.

It is not certain whether the developed models achieve equally good efficiency for meters and data derived from other sources. Due to the fact that training data supplied by only one meters' operator, the models can be biased. We should be very cautious about attempting to generalise the results of the analysis, e.g. for other types of meters.

The presented methodology of constructing a model shall perform well independent of data sources. The methods applied by us are so universal that they can be utilised to study the reliability and predict failures of other types of meters, e.g. water meters or heat cost allocators. We can imagine building one model (ensemble) containing three models trained by us. Ensemble methods work best when the predictors are as independent of one another as possible. Such a model shall perform better with the new data than a particular ANN model, SVM and BDT individually.

The results of the developed models can be successfully used in practical applications. The optimisation of a large scale replacement projects for big buildings will allow companies to save both time and resources. Such optimisation is also crucial for the tenants, who are the end users of heat meters. It does not only shorten the time needed for their presence during replacement, but it also guarantees accurate meter readings and the fair distribution of heating costs.

Supplementary Materials: Due to the sensitivity and ownership of the data, datasets used will not be publicly available. Under special circumstances the access can be granted by Rapp Enserv AG. In that case please contact Rapp Enserv, Hochstrasse 100, 4018 Basel, +415859577777.

Acknowledgments: We are very grateful to Rapp Enserv AG for allowing us to use and analyse the data. We would like to thank Prof. Józef Żurek from Instytut Techniczny Wojsk Lotniczych for sharing his expertise with us during the course of this research. We are also much obligated to Krzysztof Przysowa for his comments on an earlier version of the manuscript, although any errors are our own and should not tarnish the reputations of these esteemed persons. Special thanks should be given to Magda Tomaszewska and Krzysztof Szymański for English language editing.

Author Contributions: P. Pałasz and R. Przysowa conceived and designed the research; P.P. processed and analysed the data; R.P. verified and evaluated the results. P.P. and R.P. drew conclusions and produced the paper.

Conflicts of Interest: The authors declare no conflict of interest. Rapp Enserv AG did not sponsor this research and had no role in the design, execution, interpretation, or writing the study. The views, information, or opinions expressed herein are solely those of the authors and do not necessarily represent the position of any organization.

Abbreviations

The following abbreviations are used in this manuscript:

adam: Adaptive Momentum
ANN: Artificial Neural Network
AUC: Area Under the ROC Curve
BDT: Bagging Decision Tree
CNN: Convolutional Neural Network
DNN: Deep Neural Network
LDA: Linear Discriminant Analysis
MCC: Matthews Correlation Coefficient
MDPI: Multidisciplinary Digital Publishing Institute
ML: Machine Learning
NFLT: No Free Lunch Theorem
OIML: Organisation Internationale de Métrologie Légale | Organization of Legal Metrology
PCA: Principal Component Analysis
rbf: Radial Basis Function
RFR: Random Forest Regression
ROC: Receiver Operating Characteristic
SMBO: Sequential Model-Based Optimisation
SVM: Support Vector Machine
TPE: Tree-structured Parzen Estimator

References

1. Tawackolian, K.; Büker, O.; Hogendoorn, J.; Lederer, T. Calibration of an ultrasonic flow meter for hot water. Flow Measurement and Instrumentation **2013**, 30, 166–173. doi:10.1016/j.flowmeasinst.2012.05.003.
2. Heat meters Part 1: General requirements OIML R 75-1. International Organization of Legal Metrology **2002**, pp. 1–18.
3. Heat meters Part 2: Type approval tests and initial verification tests OIML R 75-2. International Organization of Legal Metrology **2002**, pp. 1–24.
4. Knyva, V.; Knyva, M. Testing of Data Processing Software in Heat Metering Systems. Electronics and Electrical Engineering **2007**, 7, 11–14.
5. Knyva, V.; Knyva, M. Algorithm for Estimation of Measurement Devices Software's Metrological Reliability. Electronics and Electrical Engineering **2010**, pp. 47–50.

6. Yang, J.; Jiang, Ping Chen, G.; Yuan, T.; Xue, F. Application of C5.0 algorithm in failure prediction of smart meters. 13th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP). IEEE, 2016, pp. 328–333. doi:10.1109/ICCWAMTIP.2016.8079867.
7. Yang, R.; Satou, Y.; Tateno, S.; Matsuyama, H. Fault diagnosis of flow meters utilizing historical data. SICE-ICASE, 2006. International Joint Conference. IEEE, 2006, pp. 4703–4708. doi:10.1109/sice.2006.314733.
8. Wang, Z.; Zhang, M.; Wang, D.; Song, C.; Liu, M.; Li, J.; Lou, L.; Liu, Z. Failure prediction using machine learning and time series in optical network. Optics Express **2017**, 25, 18553–18565. doi:10.1364/OE.25.018553.
9. Vafeiadis, T.; Zikos, S.; Stavropoulos, G.; Ioannidis, D.; Krinidis, S.; Tzovaras, D.; Moustakas, K. Machine Learning Based Occupancy Detection via the Use of Smart Meters. 2017 International Symposium on Computer Science and Intelligent Controls (ISCSIC). IEEE, 2017, pp. 6–12. doi:10.1109/iscsic.2017.15.
10. Yao, H.w.; Wang, X.w.; Wu, L.s.; Jiang, D.; Luo, T.; Liang, D. Prediction Method for Smart Meter Life Based On Big Data. Procedia engineering **2018**, 211, 1111–1114. doi:10.1016/j.proeng.2017.12.116.
11. Azaza, M.; Wallin, F. Evaluation of classification methodologies and Features selection from smart meter data. Energy Procedia **2017**, 142, 2250–2256. doi:10.1016/j.egypro.2017.12.626.
12. Pérez, R.; Vásquez, C. Fault location in distribution systems with distributed generation using support vector machines and smart meters. Ecuador Technical Chapters Meeting (ETCM), IEEE. IEEE, 2016, pp. 1–6. doi:10.1109/etcm.2016.7750857.
13. Makonin, S.; Wang, Z.J.; Tumpach, C. RAE: The rainforest automation energy dataset for smart grid meter data analysis. Data **2018**, 3, 8. doi:10.3390/data3010008.
14. Yang, Z.; Chen, Y.; Li, Y.F.; Zio, E.; Kang, R. Smart electricity meter reliability prediction based on accelerated degradation testing and modeling. International Journal of Electrical Power & Energy Systems **2014**, 56, 209–219. doi:10.1016/j.ijepes.2013.11.023.
15. Zhou, L.; Liao, X.; Li, S.; Yuan, J. Design and development of the reliability prediction software for smart meters. Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE), 2012 International Conference on. IEEE, 2012, pp. 612–616. doi:10.1109/ICQR2MSE.2012.6246308.
16. Zhang, G.; Wang, G.G.; Farhangi, H.; Palizban, A. Data mining of smart meters for load category based disaggregation of residential power consumption. Sustainable Energy, Grids and Networks **2017**, 10, 92–103. doi:10.1016/j.segan.2017.03.006.
17. Khairalla, M.; Abdrabou, A.; Dahir, A.; Gaouda, A. Fault location based on smart meters time synchronized measurements. Smart Grid and Renewable Energy (SGRE), 2015 First Workshop on. IEEE, 2015, pp. 1–6. doi:10.1109/sgre.2015.7208720.
18. Oprea, S.V.; Pîrjan, A.; Căruțașu, G.; Petroșanu, D.M.; Bâra, A.; Stănică, J.L.; Coculescu, C. Developing a Mixed Neural Network Approach to Forecast the Residential Electricity Consumption Based on Sensor Recorded Data. Sensors (Basel, Switzerland) **2018**, 18. doi:10.3390/s18051443.
19. Bansal, A.; Rompikuntla, S.K.; Gopinadhan, J.; Kaur, A.; Kazi, Z.A. Energy Consumption Forecasting for Smart Meters. ArXiv e-prints **2015**, [arXiv:cs.OH/1512.05979].
20. Pałasz, P.; Żurek, J.; Przysowa, R. Markov reliability model for heat meters. Journal of KONBiN **2018**, 45, 83–96. doi:10.2478/jok-2018-0005.
21. Press, G. Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says. Forbes **2016**.
22. Box, G.E.; Draper, N.R. Empirical model-building and response surfaces; John Wiley & Sons, 1987.
23. Wolpert, D.H. The lack of a priori distinctions between learning algorithms. Neural computation **1996**, 8, 1341–1390. doi:10.1162/neco.1996.8.7.1341.
24. Murphy, K.P. Machine Learning: A Probabilistic Perspective; Number Adaptive computation and machine learning series, MIT Press: London, England, 2013.
25. Domingos, P. The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World; Basic Books, 2015; p. 354.
26. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. Deep Learning; Vol. 1, MIT press Cambridge, 2016. http://www.deeplearningbook.org.
27. Bruce, P.; Bruce, A. Practical Statistics for Data Scientists: 50 Essential Concepts; " O'Reilly Media, Inc.", 2017.

28. Davis, J.; Goadrich, M. The relationship between Precision-Recall and ROC curves. Proceedings of the 23rd international conference on Machine learning. ACM, 2006, pp. 233–240. doi:10.1145/1143844.1143874.
29. Tang, Y.; Zhang, Y.Q.; Chawla, N.V.; Krasser, S. SVMs modeling for highly imbalanced classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **2009**, 39, 281–288. doi:10.1109/tsmcb.2008.2002909.
30. Long, Y.; Du, Z.J.; Wang, W.D.; Zhao, G.Y.; Xu, G.Q.; He, L.; Mao, X.W.; Dong, W. PSO-SVM-based online locomotion mode identification for rehabilitation robotic exoskeletons. Sensors **2016**, 16, 1408. doi:10.3390/s16091408.
31. Cawley, G.C.; Talbot, N.L. On over-fitting in model selection and subsequent selection bias in performance evaluation. Journal of Machine Learning Research **2010**, 11, 2079–2107. https://dl.acm.org/citation.cfm?id=1859921.
32. Allison Gray. Practical Deep Learning Examples with MATLAB; MathWorks, 2018.
33. Mazurowski, M.A.; Habas, P.A.; Zurada, J.M.; Lo, J.Y.; Baker, J.A.; Tourassi, G.D. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. Neural networks **2008**, 21, 427–436. doi:10.1016/j.neunet.2007.12.031.
34. du Jardin, P. Predicting bankruptcy using neural networks and other classification methods: The influence of variable selection techniques on model accuracy. Neurocomputing **2010**, 73, 2047–2060. doi:10.1016/j.neucom.2009.11.034.
35. Lee, K.; Yoon, H.; Song, J.; Park, K. Convolutional Neural Network-Based Classification of Driver's Emotion during Aggressive and Smooth Driving Using Multi-Modal Camera Sensors. Sensors **2018**, 18, 957. doi:10.3390/s18040957.
36. Papyan, V.; Romano, Y.; Elad, M. Convolutional neural networks analyzed via convolutional sparse coding. arXiv preprints **2016**, [arXiv:stat.ML/1607.08194].
37. Ribeiro, M.T.; Singh, S.; Guestrin, C. Why should i trust you?: Explaining the predictions of any classifier. ACM, Association for Computational Linguistics, 2016, pp. 1135–1144. doi:10.18653/v1/n16-3020.
38. Zhang, D.; Zhou, X.; Leung, S.C.; Zheng, J. Vertical bagging decision trees model for credit scoring. Expert Systems with Applications **2010**, 37, 7838–7843. doi:10.1016/j.eswa.2010.04.054.
39. Quinlan, J.R.; others. Bagging, boosting, and C4.5. AAAI/IAAI, Vol. 1, 1996, pp. 725–730. https://www.osti.gov/biblio/430734.
40. Tuv, E.; Borisov, A.; Runger, G.; Torkkola, K. Feature selection with ensembles, artificial variables, and redundancy elimination. Journal of Machine Learning Research **2009**, 10, 1341–1366. http://dl.acm.org/citation.cfm?id=1577069.1755828.
41. Bergstra, J.S.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for Hyper-Parameter Optimization. In Advances in Neural Information Processing Systems 24; Shawe-Taylor, J.; Zemel, R.S.; Bartlett, P.L.; Pereira, F.; Weinberger, K.Q., Eds.; Curran Associates, Inc., 2011; pp. 2546–2554.
42. Bergstra, J.; Yamins, D.; Cox, D. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. Proceedings of the 30th International Conference on Machine Learning; Dasgupta, S.; McAllester, D., Eds.; PMLR: Atlanta, Georgia, USA, 2013; Vol. 28, Proceedings of Machine Learning Research, pp. 115–123.
43. Prerana, T.; Shivaprakash, N.; Swetha, N. Prediction of Heart Disease using machine learning algorithms-naïve bayes, introduction to PAC algorithm, comparison of algorithms and HDPS. International Journal of Science and Engineering **2015**, 3, 90–99.
44. Snoek, J.; Larochelle, H.; Adams, R.P. Practical Bayesian Optimization of Machine Learning Algorithms. In Advances in Neural Information Processing Systems 25; Pereira, F.; Burges, C.J.C.; Bottou, L.; Weinberger, K.Q., Eds.; Curran Associates, Inc., 2012; pp. 2951–2959. http://papers.nips.cc/paper/4522-practical-bayesian-optimization.
45. Tipping, M.E.; Bishop, C.M. Mixtures of probabilistic principal component analyzers. Neural computation **1999**, 11, 443–482. doi:10.1162/089976699300016728.
46. Bergstra, J.; Bengio, Y. Random Search for Hyper-parameter Optimization. The Journal of Machine Learning Research **2012**, 13, 281–305. http://dl.acm.org/citation.cfm?id=2188385.2188395.
47. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **2014**, 15, 1929–1958. http://dl.acm.org/citation.cfm?id=2627435.2670313.

doi:10.20944/preprints201811.0293.v1

48. Powers, D.M. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies **2011**, 2, 37–63. http://hdl.handle.net/2328/27165.

49. Choi, H.M.; Yoon, B.R.; Kim, C.G.; Choi, Y.M. Evaluation of flowmeters for heat metering. Flow Measurement and Instrumentation **2011**, 22. doi:10.1016/j.flowmeasinst.2011.08.001.