Linköping University | Department of Computer and Information Science
Master thesis, 30 ECTS | Computer Science
202018 | LIU-IDA/LITH-EX-A--2018/XXX--SE

BEAUTIFUL TITLE, ENGLISH

– SUBTITLE, ENGLISH

VÄLDIGT FIN TITEL

Linus Kortesalmi

Supervisor : Mattias Tiger
Examiner : Fredrik Heintz

External supervisor : Simon Johansson

LINKÖPING UNIVERSITY

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Upphovsrätt

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår. Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns lösningar av teknisk och administrativ art. Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart. För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/.

Copyright

The publishers will keep this document online on the Internet – or its possible replacement – for a period of 25 years starting from the date of publication barring exceptional circumstances. The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/hers own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility. According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement. For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/.

© Linus Kortesalmi

Abstract

Abstract.tex

Acknowledgments

`Acknowledgments.tex`

Contents

List of Figures

Todo list

1 Introduction

Maintenance of devices/vehicles/systems/machines (hereinafter referred to as “units”) is generally done by planned scheduling. Typically this is done when some parameter of the system reaches a threshold value. For example, car maintenance can be scheduled after 30,000 kilometres, after a year or perhaps after a certain number of operating hours. One problem with planned scheduling is the reliance on experience and statistics from many units. For a single unit, the planned scheduling will either be executed too early (could have waited longer before service) or too late (problems encountered before threshold reached).

Internet of Things (IoT) permits the streaming of continuous data from multiple units. The data is typically the state of the unit in the shape of many different variable values. A rule framework can be built incorporating the continuous data. The framework can then give an informed service alert based on the actual state and need of a unit.

Machine Learning (ML) models can be used to, for example, capture dependencies in large-scale data sets (ref needed), anomaly detection (ref), clustering (ref), image recognition (ref), and decision making (ref). Anomaly detection could be used together with continuous real-time data from a system to find unusual changes or behaviour, which could be the basis for a service alert and/or the gathering of new knowledge about a system. ML-algorithms can also be used to perform Predictive Maintenance (PdM).

1.1 Background

Maintenance Techniques

There are different techniques to carry out maintenance work, each with various drawbacks and benefits. Corrective Maintenance (CM) is the technique to identify, isolate, and solve faults. The aim of CM is to restore the faulty equipment to working conditions. CM can only be done after a fault has been detected and is a reactive approach to maintenance. The fault detection can be carried out without the need of expensive, or extensive, equipment to monitor the health of a system. On the other hand, such equipment can help during the fault identification and isolation steps. The faults typically require costly repairs or equipment replacement, as small failures might not be detected until a major fault occurs. source

Another technique is Preventive Maintenance (PM). PM focuses on proactive maintenance of equipment by carefully monitoring and servicing equipment by expert personnel. The goal is to maintain satisfactory operating conditions of the system by solving minor faults before they cause major ones. PM requires good knowledge about the system and regular, systematic inspections to be carried out.

The technique which will be explored in this thesis is Predictive Maintenance (PdM). The aim of PdM is to optimise scheduled maintenance work and minimise, or even prevent, unexpected equipment failures. Equipment is directly monitored and measured to understand the actual maintenance need. Maintenance can then be scheduled for a specific equipment, based on the actual need.

Internet of Things

The rise of Internet of Things (IoT) applications in various industries indirectly solve the problem of data gathering and equipment monitoring. For example, data is typically integrated in IoT applications to provide direct system control, statistical features, localisation or planning.

1.2 Problem Description

Systems with a Corrective Maintenance (CM) policy typically suffer from expensive repairs and system downtime due to faults. Preventive Maintenance (PM) can suffer from premature repairs and replacement of equipment. Premature repairs and replacements do not only lead to greater costs but also a larger environmental stamp. Predictive Maintenance (PdM) aims at solving both issues. By monitoring the actual need of a system, premature maintenance can be avoided and minor faults detected and solved. However, PdM requires direct monitoring of equipment, which might not be easily achievable in traditional industries. Internet of Things (IoT) applications solve the monitoring problem by continuously gathering data and collecting it. This work focuses on the usage of IoT data to realise PdM. The goal is to implement PdM to reduce costs associated with repairs of faulty equipment, premature replacement of equipment, and unnecessary maintenance services.

1.3 Aim

The aim of the work is to perform a case-study on how Machine Leaning (ML) algorithms can be used on Internet of Things (IoT) data for Predictive Maintenance (PdM). The study shall compare at least two different ML algorithms. The goal is to implement the ML algorithms and use real-world data from an existing IoT system. The case-study shall also compare how well PdM performs against the Corrective Maintenance approach used in the IoT application today.

1.4 Research Questions

This work explicitly answers the following questions:

The RQs below are just an initial draft.

1. What methods can be used to detect anomalies in a dynamic real-time system?
2. How can the need for maintenance be detected from real-time processing of domain-specific data?
3. How can Predictive Maintenance be achieved by processing real-time domain-specific data?
4. How many days in advance can maintenance be predicted?
5. How can Predictive Maintenance be evaluated and compared to Corrective Maintenance?
6. How can faults be identified and isolated using Predictive Maintenance?

7. What data is required to realise fault identification and fault isolation in Predictive Maintenance?

1.5 Limitations

The data available is from an existing IoT application mainly used to control units. The available sensors in the application can not be changed and additional sensors can not be added.

2 Theory

2.1 Principal Component Analysis

Principal Component Analysis (PCA) is a dimension reduction technique used to transform a large number of variables into principal components (PCs) [11, 21]. The PCs describe the variation of the data set and are uncorrelated to each other [11]. The PCs are ordered so the first few highest-ranked variables describe most of the variance in the original variables [21].

PCA first finds a linear function $\boldsymbol{\alpha}_1'\boldsymbol{x}$, where $\boldsymbol{x}$ is a vector of p variables, that maximises the variance of $\boldsymbol{x}$. $\boldsymbol{\alpha}_1'$ is a vector of p constants $\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1p}$, where transpose is denoted $'$. The definition for a PC $\boldsymbol{\alpha}_1'\boldsymbol{x}$ is thus given by Equation 2.1 [11].

$$\boldsymbol{\alpha}_1'\boldsymbol{x} = \alpha_{11}x_1 + \alpha_{12}x_2 + \cdots + \alpha_{1p}x_p = \sum_{j=1}^{p} \alpha_{1j}x_j \tag{2.1}$$

The next step is to find the linear function $\boldsymbol{\alpha}_2'\boldsymbol{x}$, which maximises the variance subject to being uncorrelated with $\boldsymbol{\alpha}_1'\boldsymbol{x}$, then to find $\boldsymbol{\alpha}_3'\boldsymbol{x}$ up to k-th PC $\boldsymbol{\alpha}_k'\boldsymbol{x}$ maximising the variance, subject to being uncorrelated with $\boldsymbol{\alpha}_1'\boldsymbol{x}, \boldsymbol{\alpha}_2'\boldsymbol{x}, \ldots, \boldsymbol{\alpha}_{k-1}'\boldsymbol{x}$. In theory p PCs could be found, but the idea is to only find the k PCs that account for most of the variance in the original variables, where $k \ll p$ is preferred [11].

In [21], Wold discuss the implications of outliers in the data set and the type of scaling used. In general, outliers should be removed and unit variance scaling should be applied to the data set [21].

2.2 Gaussian Processes

In [17] everything we need to know about Gaussian Processes (GP) is written, good huh?

Obviously edit this and fill out the section!

2.3 Support Vector Machines

Support Vector Machines (SVMs) use a decision boundary for classifying data points. In the general case, the decision boundary is decided by maximising the margin [3]. The margin is the distance between the decision boundary and all data points [6]. The margin can be expressed by only using a set of the closest data points, which means that the formulation of the SVM is generally sparse [3]. The points in this set are aptly named Support Vectors.

The decision boundary is defined to be the hyperplane $w^T x - b = 0$. The margin hyperplanes for the two classes are thus $w^T x - b = 1$ and $w^T x - b = -1$, respectively. A new data point x_i is then classified by looking at the sign (positive or negative) of $w^T x_i - b$.

Hard margins are used when the data is assumed to be linearly separable. Linearly separable data means that all the data points can be correctly classified using a linear decision boundary. Hard margins have no flexibility to misclassify data points [3].

Soft margins can be used in the case where the data is not linearly separable. The soft margins allow for some *slack* to exist, meaning some points will be wrongly classified [6]. This introduced slack would, if not controlled, allow the SVM to misclassify every single point in order to maximise the margin [3]. The slack can be controlled using regularisation [3]. Regularisation is the concept of minimising a loss function which penalises misclassifications with respect to a given parameter λ. The regularisation parameter λ can be seen as a parameter controlling the size of the margin and the size of the loss function (the number of misclassified data points).

SVMs can also use *kernels* to transform the input space into a generally higher dimensional space [3]. Figure 2.1 illustrates an example using a radial basis function (RBF) kernel to transform the data points. The decision boundary is still linear in the transformed kernel space, but in the original input space it is nonlinear, as shown in Figure 2.1. The nonlinear decision boundary in the input space allows the SVM to correctly classify data points that are not linearly separable in the input space [3].

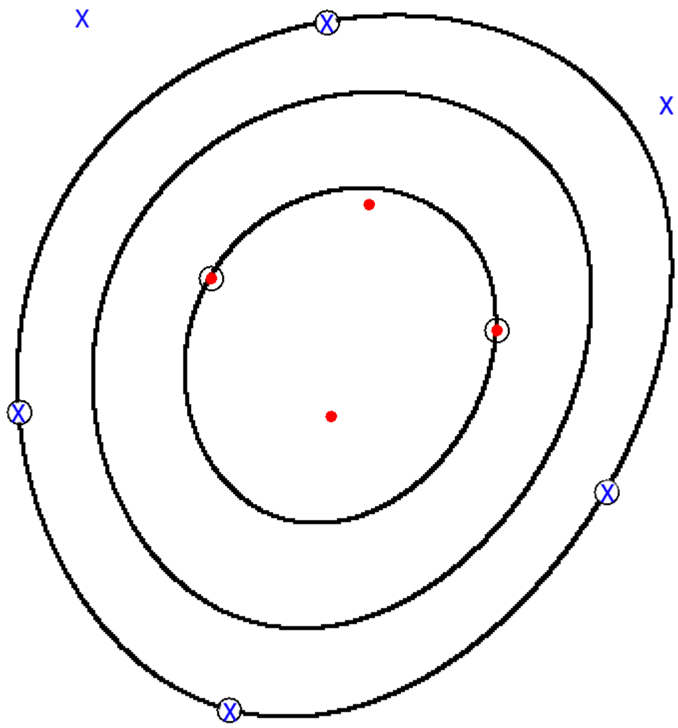

Figure 2.1: A SVM using the RBF kernel. It creates a decision boundary between the two classes that can be used to classify new points. The decision boundary is the middle black line, surrounded by the visual margin. The points with circles are the support vectors which define the margin and decision boundary. For this data set, the RBF kernel creates a nonlinear decision boundary which correctly classifies all data points.

2.4 Random Forest

Random Forest (RF) classifiers use multiple decision trees for classification. Decision trees classify data points by traversing the trained tree until a leaf is reached. The leaf contains the label that shall be assigned to the data point. An example of a simple decision tree would be to look at x_1 in the feature vector and split the tree if $x_1 < a$, for some number a. Then the decision would take into consideration x_2, until a leaf is reached. Dimensional splits of a feature vector is denoted a *subspace* of the feature vector.

Instead of only using one decision tree, trained on a certain subspace of feature vectors, the RF algorithm trains multiple decision trees [5, 8] using random subspaces [9]. In order to combine the classifications of all the trained trees, a *discriminant function* is used. The discriminant function can be seen as every tree voting on the label of the sample [5]. Averaging the output of the decision trees in a RF classifier has been proven to reduce the variance of individual decision trees [5]. The averaging can be seen as removing the correlation between the decision trees [9].

In [5], Breiman show that if too few features are used then the trees grow too large, causing the error to increase. On the other hand, if too many features are used then the correlation increases, which also increases the error [5].

2.5 Neural Networks

The traditional feed-forward Neural Network consists of multiple layers of logistic regression models [3]. Each layer consists of multiple nodes; every node in one layer is connected with links to all other nodes in the next, following, layer. The links are weight parameters that need to be trained for the network, which is typically done with back-propagation [3]. The first (input) layer feeds the features $(x_1, ..., x_D)$ over the links to the first (hidden) layer; the links give a weight to each feature. The linear combination of features and weights is denoted an *activation*. This second (hidden) layer transforms these activations and feeds them to the next, following, layer. This process is repeated until the final (output) layer is reached. The output layer uses the activation from the previous layer, together with an appropriate activation function, to give a network output vector y.

Convolutional Neural Networks

The structure of Convolutional Neural Networks (CNNs) differ from the traditional feed-forward Neural Networks. Instead of every layer containing logistic regression models, CNN layers can be of different types. CNNs typically have one or more convolutional layers and pooling layers [14]. A convolutional layer takes an input area (or an area from a previous layer) and performs convolution on it. This can be seen as applying a kernel function on an area, where the kernel functions need to be learned by the layer [12]. A pooling layer can be seen as applying a non-linear transformation to down-sample the input [14]. A pooling layer is commonly inserted between convolutional layers to control overfitting [12].

2.6 Exploratory Data Analysis

Exploratory data analysis (EDA) is a broad concept containing various techniques [1, 7, 10, 19, 20] for exploring and analysing data. Early popular techniques include *boxplots* and *stem-and-leaf* displays. A stem-and-leaf plot takes numbers and splits them into two groups. The first group contains the leading digit(s) and the second group contains the trailing digit(s). Figure 2.2 is an example of a stem-and-leaf plot with one leading and one trailing digit. The grouping helps when sorting batches of data and visualising important features, without losing the information of every single data point used [20].

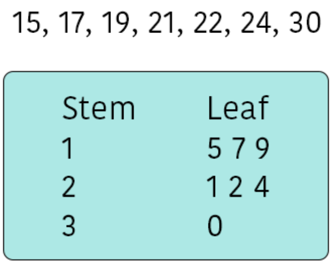

Figure 2.2: Example of a stem-and-leaf plot. The numbers above the plot is the input. The first digit of the number is the *stem*, the following digits are the *leafs*.

EDA can be seen as applying tools and statistics to analyse data in a *meaningful* way. EDA could be applied to the detection of outliers, smoothing the data, and performing a variance analysis [1, 10, 19, 20]. EDA can also reveal subtle practical problems with the chosen model that can easily be missed when performing statistical theory analysis of the model [7].

In [19], Tukey describes how EDA can be used to answer research questions such as ”What is the age distribution for the Vietnamese population?” and ”Are there differences in the annual household per capita expenditures between the rural and urban populations in Vietnam?”. Tukey uses plots to compare different groups and estimators to quantify the difference. For example, the sample mean estimator, or the *winsorised* mean can be used [19]. The winsorised mean handles the case where tails of a distribution dominates the value space. This would cause the sample mean estimator to poorly reflect on the ”typical” data point, as it is skewed by the small tail population [19].

In [20], Velleman et al. present different EDA techniques and highlights four key areas of EDA: displays (plots), residuals, re-expressions and resistance. Residuals is what remains after data analysis is performed. Residuals could, for example, be what remains after fitting data to a model (the error of the fit) [20]. Re-expression is the notion of applying mathematical functions to the data. Re-expressing the data can help with the data analysis [10, 20]. Examples of mathematical functions that can be applied are: logarithm, square root, reciprocal square function or generally raising the data to some power p. Resistance is the concept that outliers should not disproportionately affect the data analysis [10, 20]. For example, the winsorised mean estimator would be less sensitive to localised misbehaviour than the sample mean estimator [19].

Smoothing data is important for many different applications [4, 15, 16, 20]. This can, for example, be done by applying *running median smoothers*. The running median smoothers go through all the data points in sequence and calculate only the median for the n closest values near each point [20]. Another approach is the *running weighted average* [20]. Instead of taking the median of the n values, the average is calculated. The average can also be weighted with different functions, like hanning smoothing [20]. The hanning smoothing for three data points is shown in Eq. 2.2. It is worth noting that a single outlier will heavily affect the hanning smoothing [20]. In practice it is common to first do running median smoothing to remove outliers and then apply hanning smoothing [20].

$$\hat{y}_t = \frac{1}{4}y_{t-1} + \frac{1}{2}y_t + \frac{1}{4}y_{t+1} \tag{2.2}$$

3 Related Work

Predictive Maintenance (PdM) approaches have been studied and tested for many different industries. This chapter covers some of the approaches proposed in the literature.

3.1 Multiple Classifier Approach

In [18], Susto et al. propose a multiple classifier approach to PdM. They use the approach to predict problems which stem from the "wear and tear" effects of equipment used for semiconductor manufacturing. Each classifier is trained on a different failure horizon m, which results in a different classification problem for each classifier. The failure horizon is the number of iterations in a maintenance cycle where the fault has taken place. In a traditional R2F (Run to Fail) environment only the last iteration would be faulty ($m = 1$). Instead, the dataset is transformed for each classifier so that the last m iterations are marked as faulty. A larger m reduces the skewness of the dataset and enables a more conservative PdM policy. The multiple classifier approach thus enables the implementation of a cost optimisation policy as well as a fault prevention policy.

Results of the Multiple Classifier approach?

Requirements in order to use it? (type of data, dimensionality, environment (R2F), supervised/unsupervised etc.

3.2 Framework based on ARMA and data-driven techniques

Baptista et al. propose in [2] a different approach to PdM utilising usage data instead of sensor data. A framework is built in order to predict the next fault event based on previous events. The usage data (past failures and past scheduled events) is given to the Auto-Regressive Moving Average (ARMA) model, which outputs predictions on future failure events. The predictions from the ARMA model is fed to the data-driven model and transformed, using statistics features and PCA, in order to output a more informed prediction. The data-driven model trains five different classifiers: k-nearest neighbours (k-NN), random forest (RF), neural networks (NN), support vector machines (SVM), and generalised linear regression model (GLM).

The framework is then compared against a baseline approach using a standard life usage (LU) model with the Weibull distribution . Baptista et al. show in their case study [2], that almost all data-driven models outperform or perform comparably with the LU model. The only model to perform worse was the NN model, due to over-fitting [2]. The SVM model

source?

source?

achieves the best results in almost all metrics which, according to Baptista et al. [2], proves that it is possible to build more sophisticated and improved models than the LU model.

3.3 Kernel Spectral Clustering

Langone et al. demonstrate in [13] the use of Kernel Spectral Clustering (KSC) to implement PdM of industrial machines. KSC can be seen as spectral clustering formulated as weighted kernel PCA problem [13]. Langone et al. apply KSC to data collected from a Vertical Form Fill and Seal (VFFS) machine, which is used to typically fill and seal packages in the food industry. They show that two clusters are created by the algorithm. They denote one cluster as "normal" operating condition and the other as "critical" operating condition. By changing the output of the KSC they can calculate a "probability" of future faults [13]. The implementation was evaluated by comparing the cluster membership ("normal" or "critical") to logs containing conducted maintenance actions.

Vi kan nog inte använda denna approach, eftersom det inte riktigt är den typ av data som vi har tillgång till. Vi har tillgång till sensor data (och kanske även lite usage data). Dock är usage datan baserad på den tidigare statistiska modellen, så man vet ju inte riktigt ifall alla scheduled events of failure events hände precis då som de står i loggen. Det hade kanske varit intressant att undersöka huruvida man kan kombinera sensor data med usage data?

better source?

Beskriv KSC, PCA, kernel PCA och LS-SVM

4 Method

Bibliography

[1] Luc Anselin. "Interactive techniques and exploratory spatial data analysis". In: *Geographical Information Systems: principles, techniques, management and applications* 1 (1999), pp. 251–264.

[2] Marcia Baptista, Shankar Sankararaman, Ivo. P. de Medeiros, Cairo Nascimento, Helmut Prendinger, and Elsa M.P. Henriques. "Forecasting fault events for predictive maintenance using data-driven techniques and ARMA modeling". In: *Computers & Industrial Engineering* 115 (2018), pp. 41–53. ISSN: 0360-8352. DOI: `https://doi.org/10.1016/j.cie.2017.10.033`. URL: `http://www.sciencedirect.com/science/article/pii/S036083521730520X`.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 0387310738.

[4] Andrew P. Bradley. "The use of the area under the ROC curve in the evaluation of machine learning algorithms". In: *Pattern Recognition* 30.7 (1997), pp. 1145–1159. ISSN: 0031-3203. DOI: `https://doi.org/10.1016/S0031-3203(96)00142-2`. URL: `http://www.sciencedirect.com/science/article/pii/S0031320396001422`.

[5] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 1573-0565. DOI: `10.1023/A:1010933404324`. URL: `https://doi.org/10.1023/A:1010933404324`.

[6] Corinna Cortes and Vladimir Vapnik. "Support-Vector Networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 1573-0565. DOI: `10.1023/A:1022627411411`. URL: `https://doi.org/10.1023/A:1022627411411`.

[7] Andrew Gelman. "A Bayesian Formulation of Exploratory Data Analysis and Goodness-of-fit Testing*". In: *International Statistical Review* 71.2 (2003), pp. 369–382. ISSN: 1751-5823. DOI: `10.1111/j.1751-5823.2003.tb00203.x`. URL: `http://dx.doi.org/10.1111/j.1751-5823.2003.tb00203.x`.

[8] Tin Kam Ho. "Random decision forests". In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. Aug. 1995, 278–282 vol.1. DOI: `10.1109/ICDAR.1995.598994`.

[9] Tin Kam Ho. "The random subspace method for constructing decision forests". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.8 (Aug. 1998), pp. 832–844. ISSN: 0162-8828. DOI: 10.1109/34.709601.

[10] David C. Hoaglin. "John W. Tukey and Data Analysis". In: *Statistical Science* 18.3 (2003), pp. 311–318. ISSN: 08834237. URL: http://www.jstor.org/stable/3182748.

[11] Ian T Jolliffe. "Principal Component Analysis". In: *Principal Component Analysis, Second Edition*. Springer, 2002. ISBN: 0-387-95442-2.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[13] R. Langone, C. Alzate, B. De Ketelaere, and J. A. K. Suykens. "Kernel spectral clustering for predicting maintenance of industrial machines". In: *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. Apr. 2013, pp. 39–45. DOI: 10.1109/CIDM.2013.6597215.

[14] Y. LeCun and Y. Bengio. "Convolutional Networks for Images, Speech, and Time-Series". In: *The Handbook of Brain Theory and Neural Networks*. Ed. by M. A. Arbib. MIT Press, 1995.

[15] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs Up?: Sentiment Classification Using Machine Learning Techniques". In: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*. EMNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 79–86. DOI: 10.3115/1118693.1118704. URL: https://doi.org/10.3115/1118693.1118704.

[16] John R Quinlan et al. "Learning with continuous classes". In: *5th Australian joint conference on artificial intelligence*. Vol. 92. Singapore. 1992, pp. 343–348.

[17] Carl Edward Rasmussen. "Gaussian Processes in Machine Learning". In: *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures*. Ed. by Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 63–71. ISBN: 978-3-540-28650-9. DOI: 10.1007/978-3-540-28650-9_4. URL: https://doi.org/10.1007/978-3-540-28650-9_4.

[18] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi. "Machine Learning for Predictive Maintenance: A Multiple Classifier Approach". In: *IEEE Transactions on Industrial Informatics* 11.3 (June 2015), pp. 812–820. ISSN: 1551-3203. DOI: 10.1109/TII.2014.2349359.

[19] John W Tukey. *Exploratory data analysis*. Reading, Mass., 1977.

[20] Paul F Velleman and David C Hoaglin. *Applications, basics, and computing of exploratory data analysis*. Duxbury Press, 1981. ISBN: 0-87150-409-X. URL: http://hdl.handle.net/1813/78.

[21] Svante Wold, Kim Esbensen, and Paul Geladi. "Principal component analysis". In: *Chemometrics and Intelligent Laboratory Systems* 2.1 (1987). Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists, pp. 37–52. ISSN: 0169-7439. DOI: https://doi.org/10.1016/0169-7439(87)80084-9. URL: http://www.sciencedirect.com/science/article/pii/0169743987800849.