HiPerDNO

High Performance Computing Technologies for Smart Distribution Network Operation

FP7 - 248135

Project coordinator: Dr Gareth Taylor, BU

Consortium Members: BU, EF, IBM ISRAEL, University of Oxford, EENL, UNION FENOSA, INDRA, GTD, KORONA, EG, Fraunhofer IWES

Document Title	Report on new algorithms and proto-type platform for pattern detection in probabilistic data streams and new data mining algorithm developed
Document Identifier	HiPerDNO/2012/D1.3.3
Version	1.0
Work package number	WP1
Sub-Work package Number	WP1.3
Distribution	Public
Reporting consortium member	IBM, Oxford, GTD
Internal reviewer & review date	Yehuda Naveh 30/01/2012

Report on new algorithms and proto-type platform for pattern detection in probabilistic data streams and new data mining algorithm developed

Executive Summary

Future electricity distribution network operators (DNO) with mass deployment of network equipment sensors will generate vast amounts of data, which requires scalable data mining techniques in order to turn the data into actionable information. To meet these challenges DNOs can benefit from the use of techniques recently developed to cost-effectively solve large scale data mining problems using high performance computing platforms. This in turn, will bring operations and maintenance more online moving the industry from *reactive* to *proactive* operations, which can lead to actions that will improve electrical grid reliability.

In this report, we introduce new practical data mining applications and techniques designed to tackle real problems that arise in DNOs operations. Moreover, the new developed applications performance was tested and verified on real data coming from different DNOs consortium partners. By doing so, our goal is to demonstrate that data collected by electrical utilities can be used to create statistical models for proactive maintenance, to exemplify how this can be accomplished through state-of-the–art data mining techniques, and show how DNOs can be most effective in building predictions and decision support application, which in turn, can lead to actions that will improve electrical grid reliability.

The report is organized as follows: We conduct a literature review on data mining applications and techniques for power grids. Thereafter, we describe new data mining and machine learning applications designed to tackle real problems that arise in DNOs industrial operations, which were tested and verified on real data coming from different DNOs project partners. In addition, we study new algorithms to pattern detection in probabilistic data streams. Last, we present our recommendation for data mining platform, which is a key ingredient in the development process of large scale data mining applications, and disccuss our conclusions.

Document Information

Project Number	FP7 – 248135		**Acronym**	HiPerDNO
Full Title	Report on new algorithms and proto-type platform for pattern detection in probabilistic data streams and new data mining algorithm developed			
Project URL	http://www.hiperdno.eu			
Document URL	N/A			
Deliverable Number	D1.3.3	**Title**	Report on new algorithms and proto-type platform for pattern detection in probabilistic data streams and new data mining algorithm developed	
Work Package Number	WP1	**Title**	Research & Development of High Performance Computing (HPC) and Communications for Large-scale Data Processing in Distribution Networks	

Delivery	**Work Plan Date**		**Actual Date**	
Status	Version 1.0			
Nature	Prototype ☐ Internal Report ☒ External Report ☐ Dissemination ☐			
Dissemination Level	Public ☒ Consortium ☐			
Author(s) (Partners)	IBM, Oxford, GTD			
Lead Author	**Name**	Yaacov Fernandess	**E-mail**	yaacov@il.ibm.com
	Partner	**IBM**	**Phone**	972-4-8296324
Abstract				
Keywords				

HiPerDNO
COOPERATION
Energy & ICT

Table of Contents

1 Introduction

According to surveys [40] conducted in Europe and North America power grid reliability will be a key issue as electrical grids transform to smart grids throughout the next several decades, and network maintenance will become even more important than it is currently. As grid parts are replaced gradually and as smart assets are added, the old assets, including cables, switches, sensors, etc., will still need to be maintained. Maintaining a massive grid that is a mix of new and old assets is more difficult than managing a new grid. Thus, the key to making smart grid components cost effective is to use data mining and statistical machine learning for preventive maintenance, and in turn preventing cascade failures. More concretely, the electrical grid data, which includes monitoring information of the distribution network, can be transformed into scalable machine learning and data mining models that aim to predict grid reliability and assisting with maintenance actions. To meet these challenges DNOs can benefit from the use of techniques recently developed to cost-effectively solve large scale computational problems in areas such as Biology, Finance and Web Services. In such systems, increased access to ubiquitous sensing and the web has resulted in an explosion in the size of data mining and machine learning tasks, which in turn, driven the growing demand for *scalable* implementations of data mining algorithms on very large datasets (ranging from 100s of GBs to TBs of data). In doing so, DNOs will bring operations and maintenance more online moving the industry from *reactive* to *proactive* operations, which in turn, can lead to actions that will improve electrical grid reliability.

Generally speaking, data mining applications for power grids include various applications such as the prediction of power security breaches, forecasting, power system operation, control and maintenance, and classification of power system disturbances. In this report, we introduce new practical data mining applications designed to tackle real problems that arise in DNOs operations, which were tested and verified on real data coming from different DNOs project partners. In addition, we propose new algorithms to pattern detection in probabilistic data streams. Our goal is to demonstrate that data collected by electrical utilities can be used to create statistical models for proactive maintenance, to exemplify how this can be accomplished through state-of-the–art data mining techniques, and show how DNOs can be most effective in developing predictions and decision support application.

The report is organized as follows: a literature review on relevant topics in data mining is given in Section 2. More concretely, in Section 2.1 we survey feature extraction and selection methods, in Section 2.2 and 2.3 we review unsupervised clustering techniques and in Section 2.4 we review decision tree classifiers. Section 3 describes new data mining applications designed to tackle real problems that arise in DNOs operations, which were tested and verified on real data coming from different DNOs project partners. More specifically, in Section 3.1 we introduce a general framework for partial discharge (PD) diagnosis in underground cables. We demonstrate our framework performance on real industrial data coming from U.K. power networks PD activity database. In Section 3.2 we introduce a decision tree algorithm for condition monitoring application of overhead lines based on weather conditions, which we demonstrate on real data coming from Union Fenosa and E.G. In Section 3.3 we study clustering uncertain data with arbitrary correlations, which can be used to increase the accuracy of the clustering result, in case sensors in the energy distribution network fail. Finally, in Section 4 we present our recommendation of data mining

platform, which is a key ingredient in the development process of large scale data mining applications and our conclusions is given in Section 5.

2 Literature Review of Data Mining Techniques in Power System

2.1 Feature Extraction

In recent years, there has been growing interest in large-scale machine learning applications for which complex datasets with thousands of features are available. One of the key challenges, when performing such large-scale analysis of complex data, stems from the number of variables involved. The analysis of large number of variables generally requires a large amount of memory and computation resources. In order to tackle this problem, feature extraction and selection methods were recently introduced, which construct and select combinations of the features while still describing the data with sufficient accuracy.

Generally speaking, feature extraction and selection involves simplifying the amount of resources required to describe a large set of data accurately. The goal of feature extraction is to extract information from complex input datasets that captures the relevant information in order to perform the desired task. According to recent literature surveys (i.e. [20]), there are many potential benefits of feature extraction and selection: facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, defying the curse of dimensionality to improve prediction performance. Some methods put more emphasis on one aspect than another; however, in essence, feature extraction and selection objective is three-fold:

1. Improving the prediction performance of the analysis process, as the machine learning algorithms can focus on the relevant information.
2. Providing faster and more cost-effective learning machine by reducing the size of data to be processed, in order to eliminate features or attributes, which are irrelevant or redundant for the task at hand.
3. Providing a better understanding of the underlying process that generated the data. Expressing the data mining model with fewer features allows better visualization and understanding of data.

For a complete and elaborate review on feature extraction and feature selection techniques the reader is referred to the HiPerDNO deliverable 1.3.1.

2.2 Unsupervised Clustering

Unsupervised cluster analysis is used to discover distribution of patterns in data sets. In general terms, the goal of the clustering is to partition a data set into groups (clusters) so that the data elements within a cluster are more similar to each other than data elements in different clusters. Nonetheless, the notion of a cluster varies between algorithms and is one of the many decisions to take when choosing the appropriate algorithm for a particular problem. An elaborate discussion of clustering methods can be found in [11, 12, 13]. Here, we briefly review some typical cluster models.

• **Centroid-based clustering** In centroid-based clustering clusters are represented by a central vector also known as centroid which is not necessarily a member of the data set. Given a priori *k* the

fixed number of clusters, the well-known *k*-means clustering gives a formal definition as an optimization problem: find the *k* centroids and assign the observations to the nearest centroid, such that the squared distances from the cluster observations are minimized.

• **Density-based clustering** Density based algorithms typically regard clusters as dense regions of observations in the data space that are separated by regions of low density. The most popular density based clustering method is DBSCAN [14].

• **Connectivity-based clustering** Connectivity-based or hierarchical clustering [15] creates a hierarchy of clusters which is usually represented in a tree structure called a dendrogram. The tree is not a single set of clusters, but rather a multilevel hierarchy, such that clusters at one level are merged as clusters at the next level. In general, the merges/splits are determined in a greedy manner using linkage criteria, which evaluate the distance between sets of observations as a function of the pair wise distances between observations.

• **Distribution-based clustering** The clustering model most closely related to statistics is based on distribution models (i.e. Gaussian distribution). Clusters can be defined as observations belonging most probably to the same distribution. The most eminent algorithm is known as expectation-maximization clustering algorithm [16].

As outlined above, a clustering algorithm is based on a criterion for assessing the quality of a given partitioning. That is, a clustering method attempts to define the best partitioning of a data set based on certain assumptions, not necessarily the one that fits the data set. That is why, when performing cluster analysis of high dimensional data, it is important to evaluate the quality of the clustering algorithm results using cluster validation techniques. Cluster validation techniques give an indication of the quality of the resulting partitioning, and can be considered as a tool at the disposal of the domain experts in order to evaluate the clustering results. For unsupervised clustering models, clustering results are evaluated based on the data that was clustered itself. This is called *internal evaluation*. The authors in [10] distinguish between two key criteria measures for internal validation:

1. **Compactness** (a.k.a. intra-cluster distance): The members of each cluster should be as close to each other as possible.

2. **Separation** (a.k.a. inter-cluster distance): The clusters themselves should be widely separated.

Compactness assesses cluster homogeneity by looking at the intra-cluster variance, while separation quantifies the degree of separation between clusters by measuring the distance between cluster centers. Since compactness and separation demonstrate opposing trends, i.e., compactness increases with the number of clusters but separation decreases, popular methods combine the two measures into a single score, which we address their usage in Section 3.1 of this report.

2.3 Unsupervised Clustering of Uncertain Data

Both supervised and unsupervised data clustering are well-established and well-studied subfields within data mining. After its introduction in the 1960's, many new techniques have been designed and existing techniques have been improved. About 10 years ago, when research in database started explicitly accounting for *uncertainty* in data, many clustering algorithms (from each of the categories listed in Section 2.2) were adopted to work on uncertain data and new algorithms

were developed specifically for processing this type of data. Examples include (but are not limited to): FOPTICS [33], U-DBSCAN [34], and U-AHC [35].

The algorithms are designed to work for a specific representation of uncertainty. In the literature, uncertainty is represented in many different ways. The most popular representation systems are [37]:

- Existential uncertainty over tuples: the existence of each tuple in the input date is considered to be uncertain. The uncertainty can be specified using a 'simple' probability (implying tuple independence) or using more sophisticated notation with support for correlations (e.g. pc-tables).
- Probabilistic or-set tables: each tuple contains a discrete probability distribution over different possible values.
- Continuous PDF: values are represented using some continuous probability density function (e.g. a Gaussian distribution).

Although research in probabilistic databases often considers pc-tables (correlated existential uncertainty), the literature does not consider correlations in the existence of tuples. We introduce a centroid-based clustering technique for correlated uncertain data in Section 3.3.

2.4 Decision Trees

Decision trees are useful data mining tools designed to face classification problems. They posess a great versatility and they are adequate to be applied to very diverse different real applications. Their key advantage radicates on the easy interpretability of the results, they can extract human readable information about the underlying process.

In the power system domain they have been applied since the late 80s. Wehenkel [27] showed their applicability in electric power system for transient stability assessment of power systems. Swarup et al. [29] show in 1994 their good performance for security assessment. The general decision tree methodology was applied for predicting the robustness of a power system in the occurrence of severe disturbances, and for discovering appropriate control actions. Decision trees were found suitable for classification and identification of the operating states: They were effective in combining real-time possibilities, accuracy and interpretability of the results and robust since the trees showed to adjust well to learning data.

Teeuwsen et al. [28] in 2004 reported the use of decision tree for oscillatory stability assessment. The decision trees were implemented both as classifier for stability or as predictor for the system damping. Decision trees showed high accuracy , were robust to noise and could be used with only a small number of input features, they were constructed in a short period of time and used on-line since the tree evaluation did not require any time-consuming computation.

Zhiyong et Weilin [30] in 2009 applied decision trees for on-line status appraisal of a realistic Chinese power grid model. Using a knowledge database covering all possible pre-fault operating conditions, decision rules in the form of hierarchical trees were developed for on-line assessment. Furthermore, Phasor Measurement Units (PMU) were taken into consideration to improve

decision tree's performance. The results demonstrate that the proposed that decision trees were able to identify crucial security indicators and gave reliable security predictions. Their main advantages were high processing speed and the easy interpretability of the results. This method offered a twofold knowledge to system operator: first to appraise system's capability to withstand major disturbances, and second to suggest remedial actions to enhance this capability.

Lobato et al. [26] showed in 2006 the great versatility of decision trees showing their adequacy to be applied to very diverse different probabilistic real applications such as: prediction of stochastic residual demand curves in the Spanish electricity market, to estimate the daily load pattern of units and to predict the values of reactors and capacitors of the Spanish power system in a short-term time scope. Ma et al. [31] in 2010 report on the use of decision trees for detecting and identifying various transient dynamic events using the characteristic ellipsoid method. The goal was to determine fault types, fault locations and clearance times in the system. The results demonstrated that the proposed approach was capable to detect the fault type, location, and clearance time in up to 99%. Sun et al. [32] in 2011 presented a method for detecting power system islanding contingencies using both the system's topological structure and real-time system dynamic state variables. An islanding severity index concept was given for ranking the severity of the islanding cases. A decision tree algorithm was used to analyze the distinction between islanding contingencies and other operating conditions. For a full scale tree an average predication success of 98% was achieved. Simulation results demonstrated that decision tree algorithm was effective in the islanding judgment for large scare power system models. Furthermore, the important variables and primary splitters could help in deciding the phase measurement unit location.

3 Data Mining Applications

3.1 PD Analysis in Underground Cables

3.1.1 Introduction

Partial discharge (PD) is a localized failure of a small portion of a solid or fluid electrical insulation system under high voltage stress, which does not bridge the space between two conductors. PD is considered to be the main cause of long term degradation of electrical insulation. Off-line PD test is common practice for checking the integrity of the insulation of the assets. During the last decade, on-line PD monitoring techniques have received much attention both in academic and industry. According to [41] the reasons are twofold. As mentioned before, most distribution network operators (DNOs) are facing the problem of aging assets. While many assets are now approaching the end of their original life expectancy, periodic checking is no longer reliable enough for safe operation of the assets. Furthermore, on-line condition monitoring brings more efficient utilization of assets by deferring unnecessary network reinforcement. Through on-line PD monitoring systems, PD can be detected and recorded to form a comprehensive PD database which can be used for further analysis. However, analysis and interpretation of PD data remains an open research topic.

3.1.2 PD Analysis Framework

Recently, there has been an increasing effort to apply unsupervised clustering algorithms for the separation process of PD signals due to multiple sources and noise occurring in industrial environment, to name but few [4,5,8,9,10]. The separation process has been approached under the assumption that the same source generates signals having similar pulse shapes while different sources are characterized by different waveforms. The clustering process main concern is to automatically separate the contribution of the different sources of recorded practical PD activity. This work aims to provide a robust data mining framework for partial discharge (PD) pattern recognition, specifically to classify the PD signals based on their shapes. As shown in Fig. 1, we propose a clustering framework which integrates cluster evaluation techniques at multiple stages along the work-flow of the separation process.

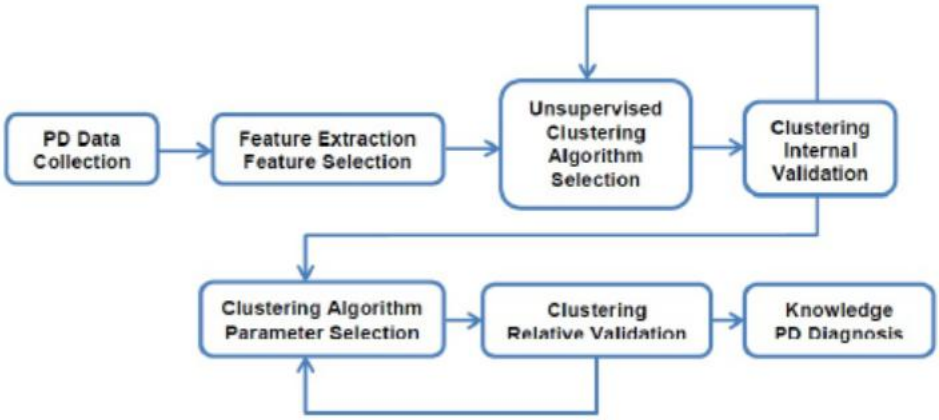


Figure 1: Framework of unsupervised learning of PD data

The framework contains feature extraction (FE), feature selection (FS) descried in Section 3.1.2 and unsupervised clustering analysis and clustering result validation described in Section 3.1.3. In the process of FE, Principal Component Analysis (PCA) is shown to be the suitable dimension reduction technique by extracting the majority of the variation in the original data set. We show that singular value decomposition (SVD) can provide additional insight to understand the results of PCA which are often difficult to interpret. By comparing the patterns of the PD pulses and the Normalized Autocorrelation Functions (NACFs) of the pulses after applying SPCA, the PD pulses are chosen to be the features for cluster analysis. In the process of cluster analysis, the need for cluster validation in unsupervised learning is discussed.

For our experiments we use UK Power Networks (UKPN) Advanced Substation Monitoring (ASM). ASM is an on-line PD monitoring system developed by IPEC Ltd. ASM acquires PD signals from the distributed PD sensors via the multiplexers. The analogue signals are processed and digitized by IPEC's purpose built signal conditioning and acquisition electronics. An integrated PC then analyses the digitized signals applying sophisticated noise reduction. In order to reliably detect the onset of PD in noisy industrial environments and accurately measure its intensity, very sophisticated signal processing is required. The PD segment refers to a 2000-point section of the unprocessed sampled data including the detected PD signal. These recorded PD signals have been aligned by their peaks where the 2000 points represents a 20μs sampling period with 4μs of signal before the PD peak and 16μs after the PD peak. These PD segment data are the raw features used for analysis throughout this Section.

3.1.3 Feature Extraction and Selection

Data dimensionality reduction is an important step in pattern recognition which aims to map high-dimensional patterns into low-dimensional ones. There are three main reasons that dimension reduction is important. First, improving the prediction performance of the analysis process, as machine learning algorithms can focus on the relevant information. Second, provide a faster and more cost-effective learning machine by reducing the size of data to be processed, in order to eliminate features or attributes, which are irrelevant or redundant for the task at hand. Finally, provide a better understanding of the underlying process that generated the data. Expressing the data mining model with fewer features allows better visualization and interpretation of data. The dimension reduction techniques are generally classified into two groups: feature extraction (FE) and feature selection (FS). FE techniques extract a set of new features from the original attributes through some functional mapping. FS is a process that selects a subset of original attributes.

3.1.2.1 Feature Extraction Using PCA

The goal of PCA is to find an orthogonal linear transformation to project the data into a set of uncorrelated variables, i.e., the principle components (PCs). The PCs are arranged in order so that the first few contain the majority of the variation in the original data set. Let $x=[x_1,x_2,\ldots,x_m]^T$ be a column vector of random variables. The covariance matrix of x is denoted by Σ. Mathematically the problem of finding the first PC coefficient a_1 for x can be formulated as

$$a_1 = \arg_{\|a_1\|=1} \max\{ a_1^T \Sigma a_1 \}, \qquad (1)$$

where $a_1=[a_{11}, a_{12},\ldots,a_{1m}]^T$ is a vector with the same number of dimensions as x. The function $a_1^T x$ is defined as PC. Having found a_1, the second PC coefficient a_2 can be defined as the linear transformation having maximum variance where $a_2^T x$ is subjected to $cov(a_1^T x, a_2^T x)=0$, i.e., $a_2^T x$ is uncorrelated to $a_1^T x$. Assume the first $k-1$ PCs have been found. The k^{th} PC coefficient a_k can be defined as

$$\begin{aligned} & a_k = \arg_{\|a_k\|=1} \max a_k^T \Sigma a_k \\ & s.t. \qquad\qquad (2) \\ & \operatorname{cov}(a_1^T x, a_2^T x,\ldots, a_{k-1}^T x) = 0 \end{aligned}$$

Where $k=2,\ldots,m$. It has been shown that (1) and (2) can be solved using the technique of Lagrange multipliers. The solutions are given in the following theorem. The proof can be found in [6], and therefore is omitted.

Theorem 3.1 **(Principle Components)** Let Σ be the covariance matrix of x. Assume the eigenvalues of Σ are distinct. For all $k=2,\ldots,m$, the following statements hold. The vector of coefficients a_k^T for the k^{th} PC is an eigenvector of Σ corresponding to the k^{th} largest eigenvalue λ_k.

- The variance of the k^{th} PC is λ_k.
- From Theorem 3.1 we can see that solving the maximization problems in (1) and (2) involves eigen-decomposition of the covariance matrix. In the next section, we will discuss using the technique of SVD to perform PCA.

3.1.2.2 SPCA

Consider a row vector $x=[x_1,x_2,\ldots,x_m]^T$ consisting of m sampling points taken from a snap shot of PD signal. Suppose there are n such snapshots. We can form a m x n sample matrix $X=[x^T_1,x^T_2,\ldots,x^T_n]$ where the i^{th} sample is $x_i=[x_{i1},x_{i2},\ldots,x_{im}]^T$ $i=1,2,\ldots,n$. Here m represents the number of dimension and n is the number of observations. As mentioned in section 3.1.1, these PD signal snap shots have been aligned by their peaks. PCA is well known to be scale dependent. Before proceeding to the procedure of PCA, data need to be normalized. In this work we normalize the data by subtracting off mean for each dimension. Discussions of other normalization methods can be found in [14]. With a slight abuse of notation, we use X to denote the normalized matrix where $X_{ij}=x_{ij}-\mu_j$ and μ_j is the mean on j^{th} dimension. The PCs defined in section 3.2 can be rewritten in matrix form,

$$P = A^T X. \qquad (3)$$

The matrix $A_{mxm}=[a_1,a_2,\ldots,a_m]$ is the PC coefficient matrix. Thus, the entries of P can be obtained as $P_{ij}=a_i^T x_j$, $i=1,2,\ldots,m$, $j=1,2,\ldots,n$. Thus $a_i^T x$ is defined as the i^{th} PC and $a_i^T x_j$ is the PC score for the j^{th} observation on the i^{th} PC. As given in Theorem 3.1, the vector a_k^T is the eigenvector of the covariance matrix Σ_X for the normalized samples $[x^T_1,x^T_2,\ldots,x^T_n]$ corresponding to the k^{th} largest eigenvalue. It can be shown that the covariance matrix $\Sigma_X=(1/n)XX^T$ when the samples have zero means. Based on matrix theory [8], a matrix Y_{nxm} can be decomposed using SVD into the form

$$Y = USV^T \qquad (4)$$

where *U* and *V* are *n x n* and *m x m* orthonormal matrices and *S* is an *n x m* diagonal matrix. On the leading diagonal of *S* are the non-negative singular values (SVs) σ_j, j=1,2,…,min(n,m), arranged in descending order. The SVs of *Y* are the square roots of the eigenvalue of Y^TY. The columns of *U* and *V* are orthonormal eigenvectors of YY^T and Y^TY respectively. Compare (3) and (4), we can see PCA and SVD are closely related. Define matrix *Y* as $Y = \frac{1}{\sqrt{n}} X^T$. The equalities

$$Y^YY = (\tfrac{1}{\sqrt{n}} X^T)^T(\tfrac{1}{\sqrt{n}} X^T) = \tfrac{1}{n} XX^T \qquad (5)$$

show Y^TY equals to the covariance matrix Σ_X. Thus the columns of *V* are orthonormal eigenvectors of Y^TY if we apply SVD to the matrix *Y*. In other words, the columns of *V* are the PC coefficients of *X* since $Y^TY = \Sigma_X$.

In order to perform dimension reduction, a decision must be made on the number of PCs to be retained to summaries the data. The rules of choosing l_k, the number of PCs, are mostly ad hoc rules-of-thumbs in literature. In this paper, we discuss the choice of l_k based on SVs. The reasons are twofold. First, SVs and variances are related. In Theorem 3.1 the variance of the k^{th} PC is shown to be λ_k. Hence, the variances are simply σ_k^2 since $\sigma_k^2 = \lambda_k$. Secondly, studying the effects of dimension reduction based on SVs can bring additional insights into PCA as an FE technique.

In the field of biomedical signal processing, the SVD technique has been applied successfully in noise filtering as discussed in [7]. The subspace containing the signal information is of a lower rank than the original matrix. Small eigenvalues means less energy for the corresponding eigenvectors. The subspace containing the first *k* largest eigenvalue is considered to be the signal space and the remaining subspace generally contains noise. Hence, SVD can be used to filter noise in signals. The reduced SVD is of the form

$$\tilde{Y} = U_k S_k V_k^T \qquad (6)$$

U_k and V_k are *n x k* and *m x k* matrices computed based on the *k* largest singular values. S_k is a *k x k* diagonal matrix with *k* singular values on the diagonal. $\tilde{Y}$ is the data matrix after SVD filtering.

With regard to choosing the number of truncation rank *k*, we will apply a simple test on the second-order rate of change of σ_j, j=1,2,…,min(n,m),. Consider the vector $\sigma=[\sigma_1, \sigma_2, \ldots, \sigma_{min(m,n)}]$ consisting of all singular values of matrix *Y*, where *min(n,m)>2*. The vector of the second-order rate of change is $\Delta\sigma=[\Delta\sigma_1, \Delta\sigma_2, \ldots, \Delta\sigma_{min(m,n)-2}]$ where $\Delta\sigma_i=|(\sigma_i-\sigma_{i+1})-(\sigma_{i+1}-\sigma_{i+2})|$. The selection rule is: If $\Delta\sigma_i$ is the unique maximum of $\Delta\sigma$, the truncation rank *k* is chosen to be *k = i + 2*. For the case of multiple maximum, *i* is chosen to be the highest subscript of the multiple maximum points. All singular values after this point will be discarded.

The results of applying SVD for PD signal filtering are shown in Fig. 2-. In this example, we have chosen *150* PD pulses for analysis. These pulse snapshots are taken from different months over a one year period. Fig. 2 shows a subset of the chosen PD pulses which are contaminated by noise. Fig. 3 shows the eigenspectrum of the SVD decomposition. Based on the method mentioned above, the `knee' of the curve is at the 5th SV. Fig. 4 shows the filtered signals reconstructed using the first 5 SVs. The effects of different SV components to the reconstructed pulses are shown in Fig. *5-7*. The *9* plots in Fig. 5 are superimposed plots of the reconstructed pulses based on the 1st, 2nd, …, 9th SV. The first plot shows that the pulses reconstructed from the largest SV give the largest

magnitudes. The first 5 plots show different patterns while the patterns of the 6^{th}-9^{th} plots are quite consistent. This shows it is appropriate to choose *k=5* as the truncation rank. In fact, Fig. 7 shows that 5^{th} is the lowest number of SVs where the cumulative variance is more than *80%*. In general, having a *80%* cumulative variance is considered to be good enough to summaries the data. Fig. 6 shows the reconstructed pulses based on the last *4* SVs. It clearly shows the characteristics of noise. For example, the patterns of these plots are very similar and the magnitudes of these components are very small.

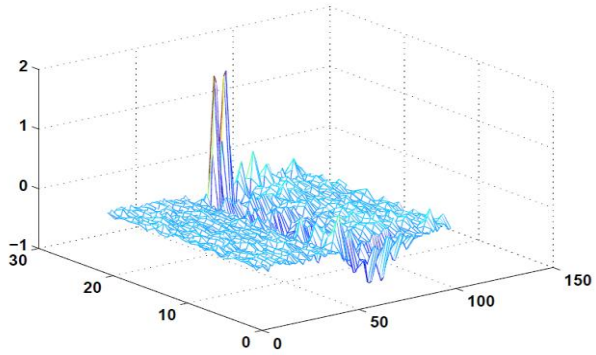


Figure 2: PD signals contaminated by noises.

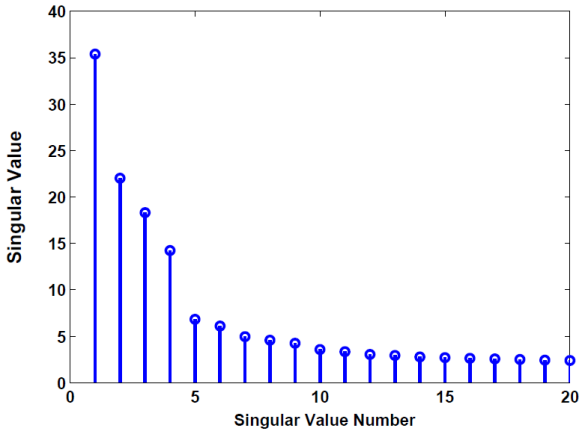


Figure 3: Eigenspectrum of SVD decomposition.

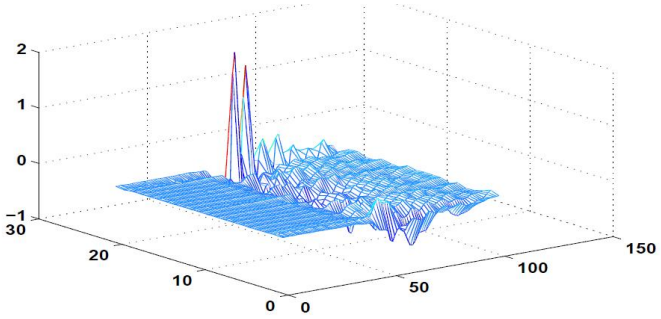


Figure 4: PD signals after SVD filtering.

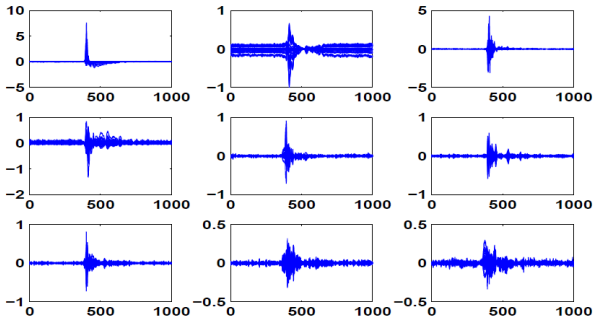


Figure 5: Reconstructed PD pulses based on the first 9 SVs.

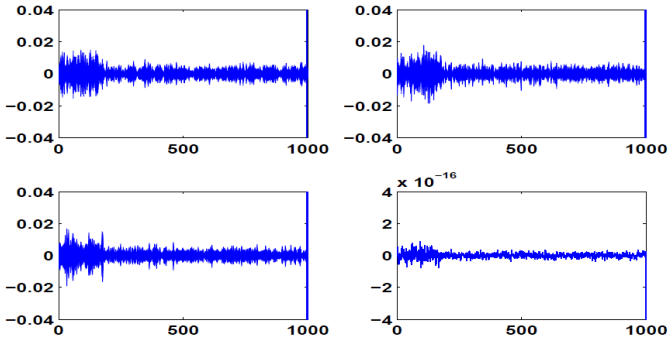


Figure 6: Reconstructed PD pulses based on the last 4 SVs.

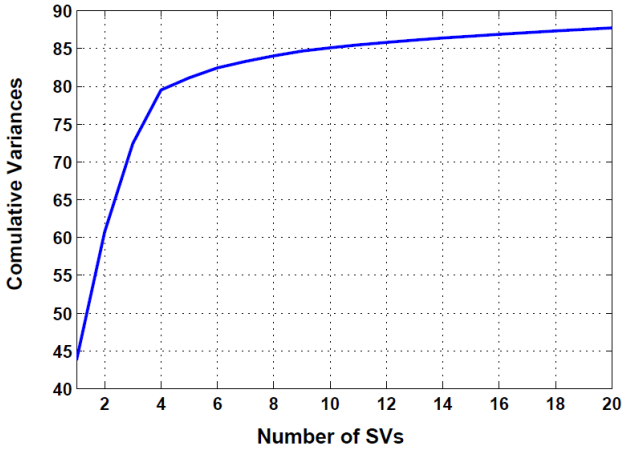


Figure 7: Cumulative variance against the number of SVs.

In this example, we have chosen one-year worth of data for analysis. We first apply SPCA to perform FE on the PD pulses and NACFs. The scatter plots of the first and second principle components for both cases are shown in Fig. 8 and Fig. 9. As discussed in Section 3.1.2.2, the first few PCs summaries the majority of variation in the data. After experimenting with *1-3* PCs, we choose to show the plots of the first two PCs as they give very clear and indicative patterns. In Fig. 8, we can see there are three well-separated clusters. However, the pattern in Fig. 9 is less clear. If unsupervised clustering is

chosen as the next step for pattern recognition, clusters result from Fig. 7 will be expected to have much better quality than the clusters found from Fig. 8. In other words, for clustering purpose, the pulse data are more suitable features than the NACF.

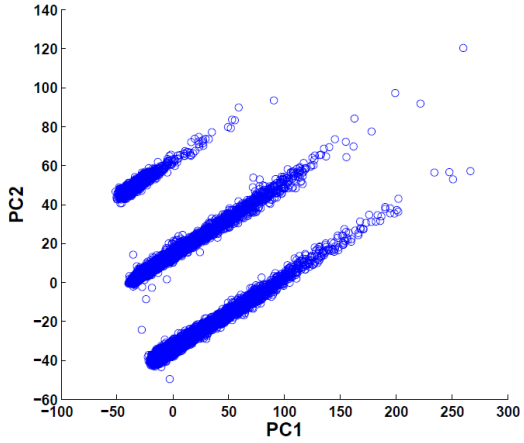


Figure 8: Principle components of PD raw data

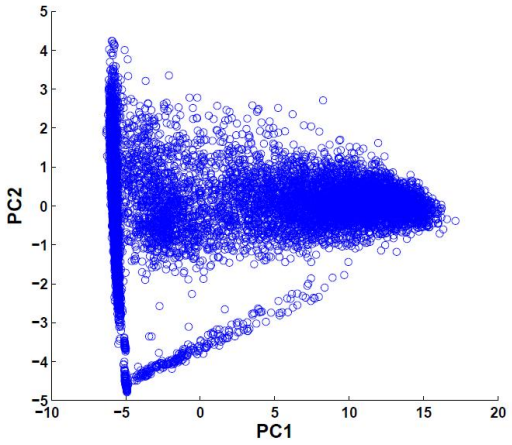


Figure 9: Principle components of NACF

3.1.4 Unsupervised Cluster Analysis

A clustering algorithm is based on a criterion for assessing the quality of a given partitioning. As discussed in Section 2.2, it is important to evaluate the quality of the clustering algorithm results using appropriate criteria and techniques. Hence we propose a clustering framework that integrates *cluster validation* techniques at multiple stages along the work-flow of the separation process of PD signals due to multiple sources. Cluster validation techniques give an indication of the quality of the resulting partitioning, and can be considered as a tool at the disposal of the domain experts in order to evaluate the clustering results. Moreover, by integrating cluster validation in our framework we automatically conduct a sensitivity analysis and investigate two key challenges in unsupervised cluster analysis. The first challenge is to estimate the parameters values of an arbitrary algorithm (i.e. the number of clusters). The second challenge is to investigate the variation in the results when using different clustering algorithms. As a result, we can automatically and systematically choose the cluster algorithm and to estimate the number of clusters for our separation

process of PD signals, and in turn, obtain an efficient diagnosis of PD activity. The general approach is to evaluate the quality of the results from each algorithm and select the algorithm that generated the best partition according to validation method. The evaluation procedure proceeds with the following steps:

1. **Execute**: Each algorithm is executed several times to improve confidence.

2. **Choose**: The algorithm that obtained the best index results will be chosen.

An additional related problem is to estimate the number of clusters that are most appropriate for the data set. Here the basic idea is to evaluate the clustering structure by comparing it to other clustering schemes, resulting by the same algorithm but with different number of clusters. The general framework of estimating the number of clusters is also based on internal validity index. The evaluation procedure proceeds with the following steps:

1. **Execute**: For each possible value of the number of clusters, which are determined a-priori, we execute the algorithm several times in order to improve confidence.

2. **Choose**: We choose the algorithm parameters values that obtained the best index results.

Next, we present a comparative experiment of PD separation process using our clustering framework. For our experimental work, we obtained 3000 pulses from UKPN ASM and applied the SVD-guided PCA feature extraction method. As before, the goal is two fold: to determine the most appropriate unsupervised clustering method and to estimate the number of clusters for the data set. That is, by applying the validation measures, we calculate the scores along with the corresponding cluster method and number of clusters, and in turn, choose the configuration that obtained the best results. For our study, we compared between hierarchical and partitioning algorithms, two commonly discussed clustering approaches. More specifically, we compared between the well-known k-means algorithm and agglomerative hierarchical clustering algorithm, in which we apply three commonly used linkage criteria namely, complete, single and centroid. With reference to validity methods, three indexes has been considered for this application due to there ability to investigate the variation in the results when using different clustering algorithms and estimate the number of clusters that are most appropriate for the data set. As before, the idea, here, is to execute each algorithm a several times for different number of clusters each time. Thereafter, we plot the respective graphs of the validity indexes for the resulting clustering and search for the optimal index value.

The first index is the Silhouette [17], which is the average of the Silhouette value of each observation. The Silhouette value measures the degree of confidence in the clustering assignment of a particular observation, with well-clustered observations having values near *1* and poorly clustered observations having values near *-1*. The Silhouette value for observation *i* is defined as follows:

$$S(i) = \frac{b(i) - a(i)}{\max\{b(i), a(i)\}}$$

where a_i is the average distance between *i* and all other observations in the same cluster and b_i is the average distance between *i* and the observations in the ``nearest neighboring cluster''. The Silhouette average thus lies in the interval *[-1,1]*, and should be maximized. Fig. 10 summarizes the result of the Silhouette index, for different clustering schemes as mentioned

above of our PD signals data set. The graph indicates that estimate number of clusters is *2* for both *k*-means and for variations of linkage criterion of hierarchical clustering, and it is clear that hierarchical clustering outperforms *k*-means clustering results.

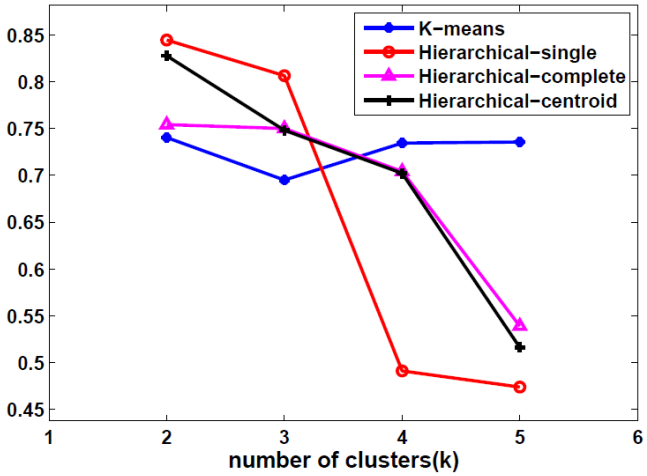


Figure 10: Silhouette Average

The second, is Dunn index, which based on the idea of identifying the cluster sets that are compact and well separated. The Dunn index is the ratio of the smallest distance between observations not in the same cluster to the largest intra-cluster distance. It is computed as follows:

$$D = \min_{i\in[k]}\left\{\min_{j\in[k],i\neq j}\left\{\frac{d(c_i,c_j)}{\max_{r\in[k]}\{diam(c_r)\}}\right\}\right\}$$

Where $d(c_i,c_j)$ denotes the distance between clusters c_i, and c_j (inter-cluster distance); and $diam(c_i)$ is the diameter of cluster c_i (intra-cluster distance), which is the maximum distance between observations in cluster. The Dunn index has a value between zero and *1*, and should be maximized. Figure 11 summarizes the result of the Dunn index, for different clustering schemes as mentioned above of our PD signals data set. The graph indicates that estimate number of clusters is *2* for both *k*-means and for variations of linkage criterion of hierarchical clustering, which reinforce our previous result. Moreover, it is clear that the single linkage criterion obtains the best results. The third, is Davies-Bouldin index, which is a function of the ratio of the sum of within-cluster scatter to between-cluster separation given by

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{i\neq j}\frac{\sigma_i+\sigma_j}{d(c_i,c_j)}$$

where σ_i denotes the average distance of all objects from the cluster to their cluster center, and $d(c_i,c_j)$ denotes the distance between clusters centers. Hence the ratio is small if the clusters are compact and far from each other. Consequently, Davies-Bouldin index will have a small value for a good clustering. Fig. 12 summarizes the result of the Davies-Bouldin Index, for different clustering schemes as mentioned above of our PD signals data set. The graph indicates that

estimate number of clusters is *2* for both *k*-means and for variations of linkage criterion of hierarchical clustering, which reinforce our previous results. However, it is clear that the single linkage criterion obtains the worst results.

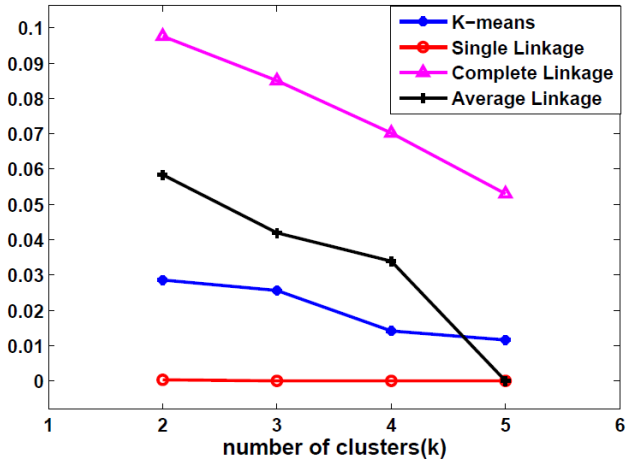


Figure 11: Dunn Index

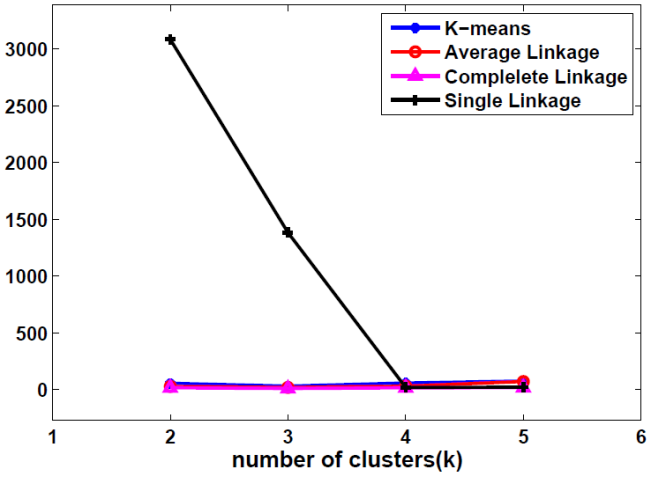


Figure 12: Davies-Bouldin Index

3.2 Correlation of Weather and Fault Data

3.2.1 Introduction

Weather is one of the major factors affecting power distribution systems. On one hand weather conditions affects the reliability of power distribution systems on the other, energy consumption is highly dependent on the weather conditions, specially the temperature. Power quality studies have focused on the source-identification of voltage disturbances occurring in distribution networks [19]. The ability to model weather's impact on overhead distribution lines will allow utilities to take actions and prevent or reduce outages. A decision support system that predicts the probability of failure based on the current weather conditions in each area will allow them to make the right decisions to plan maintenance teams to reduce the impact or consequences of the failures.

Power delivery companies are paying more attention nowadays to reliability of electric service due to increased expectations from customers and regulators. Distribution Network Operators (DNOs) measure their system performance based on reliability indexes, such as CAIDI -

Customer Average Interruption Duration Index or SAISI - System Average Interruption Frequency Index which measures average outage duration for each customer served. In both cases the duration of the failures plays an important role.

Data mining algorithms can be used to describe the correlation between faults and the weather conditions. Characteristic parameters of the weather conditions causing certain faults can be extracted from the knowledge obtained by these techniques and allow a better maintenance plan to reduce the consequences of the failures or even prevent them. Thus, by predicting potential failures based on the nature of the failures (weather), failure duration could be reduced and improve system performance and therefore DNOs' reliability indexes.

In the following section a introduction to decision trees, the basis of the algorithm developed, is presented.

3.2.2 Decision Trees

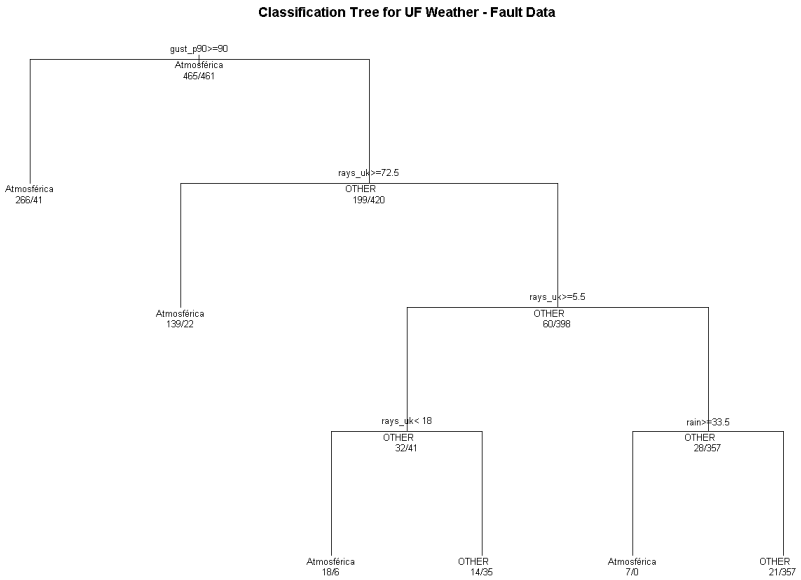
Decision trees are useful data mining tools designed to face classification problems. They possess a great versatility and they are adequate to be applied to very diverse different real applications. Their key advantage radicates on the easy interpretability of the results and the supply of probability values without assuming normal distributions.

A decision tree represents a function that takes as input a vector of attribute values and returns a "decision"—a single output value. The input and output values can be discrete or continuous. A decision tree reaches its decision by performing a sequence of tests. Each internal node in the tree corresponds to a test of the value of one of the input attributes, and the branches from the node are labeled with the possible values of the attribute. Each leaf node in the tree specifies a value to be returned by the function. The ***decision tree representation is natural for humans***; indeed, many "How To" manuals (e.g., for car repair) are written entirely as a single decision tree stretching over hundreds of pages.

A Boolean decision tree is logically equivalent to the assertion that the goal attribute is true if and only if the input attributes satisfy one of the paths leading to a leaf with value true. Writing this out in propositional logic, we have Goal (Path1 V Path2 V ...) , where each *Path* is a conjunction of attribute-value tests required to follow that path. For a ***wide variety of problems, the decision tree format yields a nice, concise result***.

The **decision *tree learning algorithm*** adopts a greedy divide-and-conquer strategy: ***always test the most important attribute first***. This test divides the problem up into smaller subproblems that can then be solved recursively. "Most important attribute" refers to making the most difference to the classification of an example. That way, the correct classification will be achieved with a small number of tests, meaning that all paths in the tree will be short and the tree as a whole will be shallow.

Note that the set of examples is crucial for constructing the tree, but nowhere do the examples appear in the tree itself. A tree consists of just tests on attributes in the interior nodes, values of attributes on the branches, and output values on the leaf nodes.



Decision tree induction is one of the simplest and yet most successful forms of machine learning. The decision-tree learning algorithm splits the example input set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions.

The greedy search used in decision tree learning is designed to approximately minimize the depth of the final tree. The idea is to pick the attribute that goes as far as possible toward providing an exact classification of the examples.

For decision trees, a technique called decision tree pruning combats overfitting. Pruning works by eliminating nodes that are not clearly relevant. The pruning starts with a full tree, then it looks at a test node that has only leaf nodes as descendants, if the test appears to be irrelevant—detecting only noise in the data—then the test is eliminated, replacing it with a leaf node. The process is repeated considering each test with only leaf descendants, until each one has either been pruned or accepted as is.

3.2.3 Feature Extraction Using Decision Trees

As pointed out before the decision tree learning algorithm adopts a greedy divide-and-conquer strategy by always testing the most important attribute first. This means that the first levels of the tree contained the most relevant attributes for the goal classification. Therefore, decision trees can not only be applied for classification but also can be used for the feature extraction and selection task. The relevant attributes can be selected by taking the first attributes used by the decision tree, since the most important attributes, those making the most difference to the classification, are always at the top of the decision tree.

3.2.4 Random Forest

Significant improvements in classification accuracy have resulted from growing an ensemble of trees and letting them vote for the most popular class. In order to grow these ensembles, often random vectors are generated that govern the growth of each tree in the ensemble.

Random forests are a ***combination of tree predictors*** such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them.

Since random forest is an ensemble classifier based on building many decision trees, the classification output is based on the vote of the individual trees. The random forest learning algorithm strategy can be described as follows: to classify a new object from an input vector, the input vector goes down each of the trees in the forest. Each tree gives a classification, and the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Random forest is reported to run efficiently on large datasets giving high accuracy rates. Furthermore, it gives estimates of what variables are important in the classification and can even show variable interactions. Therefore it performs naturally a feature extraction/selection process.

3.2.5 Proposed Algorithm Inspired by Random Forest

One of the major problems to adopt expert systems based on artificial intelligence technologies is the lack of control and understanding about the decision that the system is performing. These systems they are usually seen as black boxes which take decision “magically”. Thus, critical applications forbid the use of such systems since it can not be analyze the result of a decision automatically made by the system. Specially when the system makes a wrong decision, it is often difficult to describe why the system did a bad choice and try correct the behavior. However, there are systems which are descriptive, which means that they can actually justify their decision and the system can be tuned or adapt to include expert knowledge to avoid bad choices. Decision trees are descriptive systems, since their representation is very natural for humans and it is straightforward to derive rules which can be then use to justify the choices made the classification process.

The algorithm proposed is inspired on random forest. It creates several trees based on the training set available, like the random forest algorithm the training set for each tree will be obtained by random sampling with replacement the original training set. The trees grown, unlike random forest, might be pruned to obtain a compact model representation of the knowledge. Once the trees are obtained, there is a transformation step to create the knowledge representation based on rules. The final goal is to be able to transform and fusion the knowledge of the different trees built in the form of IF-THEN rules. For every tree the IF-THEN rules are created, and since all trees are built based on the same training set, but with different random sampling, some trees could potentially lead to similar rules. Thus, a unification step is needed in order to remove potential repeated rules. After that the rules are ranked based on two parameters called confidence level and coverage. Confidence level refers to the performance obtained on a cross-validation evaluation of the

rule, and coverage refers to the percentage of examples (input instances) that they covered. Note that this is not the validation step, since the cross-validation is performed previously on the resulting trees, but a way to select the most relevant rules for the description of the knowledge acquired by the tree. Depending on the number of attributes available on the input data a similar strategy to random forest can be used, where a subset of the original attributes is selected randomly for selecting the best split at each node.

The following learning process can be summarized as follows:

- **Creation of decision trees**: First a tree with all attributes and data will be built. The level of complexity (cp) will be set to a very low value, i.e 0.0001 (this will create a very complex tree which will provide several rules for the target descriptive model). Depending on the number of attributes, a set of smaller trees might be built, following the strategy (inspired in the random forest): First a value m which is m < M (M is the number of attributes available on the data) is selected. This is the minimum number of attributes that will be used to generate trees. Then a value n will be selected with n<N (N is the number of data instances) to build the tree with different training sets.
- **Knowledge representation** step to obtain **IF-THEN rules**.
- **Unification step of the IF-THEN rules**: the trees obtained previously will have different inputs instances and the performance and coverage obtained for those will depend on the training set (this could yield to the same rules as other tree but with different performance and coverage, if the same rule has different values for performance and coverage, the average will be taken for the final rule)
- **Rules Ranking and selection:** Based on coverage and confidence. A minimum of confidence and coverage might be selected and only rules above both of these thresholds will be taken.

Feature extraction is also performed inside the proposed algorithm. The algorithm builds several trees, the variables used in each node for the split are the relevant features for the data mining task. Moreover, since the different trees are converted to IF-THEN rules and they are ranked based on the confidence level and the coverage, and therefore features can be ranked as well based on the attributes used to the rules.

The algorithm is implemented in a three-level approach: the first level generates in parallel different knowledge representations based on decision trees. The result of this first level is a group of models containing the knowledge derived. The second level gets as input the models obtained in the first level. The process on this second level induces IF-THEN rules based on the initial models and processed them to compute a new knowledge representation model based on the coverage and level of confidence of the rules derived. At the third level the model obtained in the previous step is used to perform the classification task.

The expected architecture to run in the HPC platform relies on the fact that communication between the different levels must be warranty. The first levels can be seen as N different pipelines which all get the input data. The output information of each pipeline is sent to the following level. The second level can be seen as a pipeline that waits to get results of the first N pipelines. The system can either wait to get all input from the N previous pipelines or can start processing even if some previous pipelines are not yet finished, but in both cases the second level needs to get all

inputs from the first level before the end of the processing. Once the second level has output its model, the third level can start processing it and update the classification results.

3.2.6 Experiments

The following paragraphs describe the first analysis performed using decision trees and the algorithm proposed in different set of data from the HiPERDNO consortium.

3.2.2.1 UF Data Analysis: Overhead lines condition monitoring based on weather conditions

Weather is one of the major factors affecting power distribution systems. One good example of the problems caused by bad weather conditions are overhead lines faults, since they are mainly caused by atmospheric conditions

DNOs keep a log of the failures suffered by their networks and based on this information they compute several reliability indexes. Since overhead lines suffered from faults mainly caused by atmospheric conditions, by monitoring the current weather conditions that the line is experimenting can help to predict the likelihood that the line is going to experiment a fault based on historical information of weather conditions that in the past caused failure.

A decision support system that studies faults caused by atmospheric conditions (bad weather/storms/lightings) can be implemented to define an index for characterization of storms/weather conditions which allows quantifying the impact of storms/weather condition. Such index will allow comparing different storms/weather conditions and therefore predicting their impact. The comparison can be performed based on historical information about weather conditions and the faults occurred during these situations.

The proposed algorithm in the previous section can generate a descriptive correlation between geographical weather conditions and the probability of a fault in the distribution network. It focuses on using historical fault data, where faults where caused by some atmospheric conditions and historical meteorological information such as rain level, wind force and lightnings per area. The IF-THEN rules model obtained can be applied to current meteorological conditions in different areas to give a probability of having a fault in a certain area based on similar meteorological conditions that in the past caused an atmospheric fault.

UF Weather - Fault Data

The following data has been provided by Union Fenosa. The data consists in two different sets:

- Data regarding faults in the distribution network
- Data about the meteorological conditions

Both sources contain information of 3 different years: 2008, 2009, and 2010.

Faults Data

The data containing the faults is divided to Medium voltage (MV) and High voltage (HV) and have the following information (some of the attributes are only available for MV or HV):

- ID of fault

- Type of fault
- Cause
- If the fault is solved by a reclosing and therefore it is not relevant for the TIEPI index.
- Associated to a market lost
- Affected facilities
- Detection date
- Detection time
- Affected area
- Voltage level
- Resolution date
- Resolution time
- Substation
- Working day
- Duration
- Power
- Contracted power multiplied by the duration of the fault
- TIEPI: Index, similar to SAIDI but related to contracted power and not to number of customers
- Number of customers affected

Meteorological data

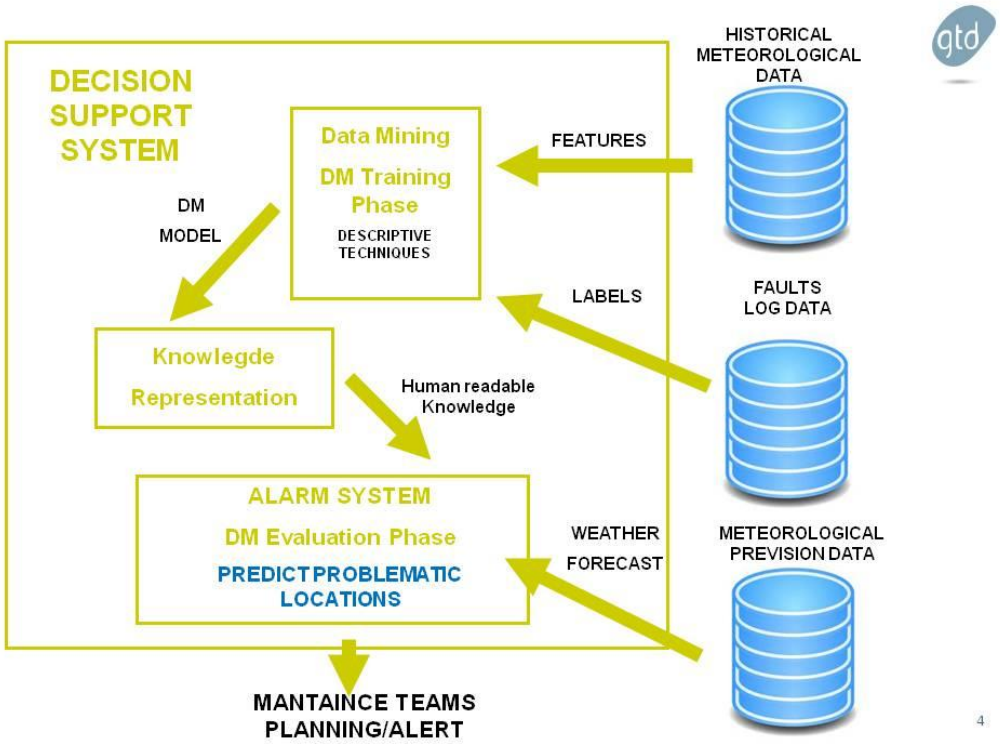
The meteorological data correspond to different provinces in Spain. The available fields are:

- Date
- Number of lightnings/10000km2 (source: UK Metoffice)
- Number of lightnings/10000km2 (source: AEMET)
- Rain in mm
- Maximum gust of wind percentile 50 (km/h)
- Maximum gust of wind percentile 50 (km/h)
- Dominant wind percentile (50 km/h)
- Dominant wind percentile (90 km/h)

UF Weather - Fault Data Application

The application's goal is to define an index for characterization of weather conditions based on the probability of causing a fault in the power distribution system. Such index will allow comparing different weather conditions and therefore predicting their impact. The comparison can be performed based on historical information about weather conditions and the faults occurred during these situations. The system will learn the weather conditions that in the past caused a fault in the system, thus historical examples of faults which were observed to be caused by weather conditions is the basic input for the learning step of the system. The following diagram depicts the overall system. The two source of information are the historical meteorological data and the logs from the faults detected on the system. From these two data sources the system will extract the features and labels to perform the supervised learning and will create in the training phase a model that will lead to a knowledge representation based on IF-THEN rules. This knowledge

representation will then be used to compare to current meteorological data or even metrological forecasts to the data mining model and the system will be able to classify areas according to the probability of having atmospheric faults. This will allow the planification of the maintenance teams in case the faults appears.



Data Preprocessing

The original data from UF was two disjoint sets of data. A preprocessing step was needed in order to work with all input data. The relation between both data sets was made by means of the date. For each fault detected on the system the corresponding weather conditions from the meteorological data was found and match to the faults. The goal was to describe the weather conditions on the very moment of the fault.

Data Mining Model Creation

The analysis was performed initially using decision trees to study different parameter tuning and afterwards the proposed algorithm was used.

Decision Tree Analysis

The initial analysis was performed using decision trees on the meteorological data using as class for this supervised learning a new attribute that distinguish between atmospheric cause and other type causes. Faults labeled as other were the union of all faults except the atmospheric ones.

The first studies were performed using most of the data available. The data was strongly unbalanced, that means that there were far more examples of faults that were not caused by atmospheric conditions. Some of the first decision trees built were based on more than 3000 faults, where only less than 14% correspond to atmospheric faults. Due to the strong difference between

the amounts of instances from non-atmospheric faults, the results obtained were very poor and thus their models could not be applied for the detection of the atmospheric faults. To guarantee a balance in the data a subset of the original data was selected in order to have a similar amount of atmospheric and non atmospheric faults (485 faults were labeled as atmospheric in the original data). It was considered only information regarding meteorological conditions extracted from UF database, no extra data was added. The studied data contained:

- 458 atmospheric faults (only from the Galician region)
- 535 other faults (descargo)
  Total faults: 993

Different experiments were performed using the J48 algorithm tuning the pruning level, which allows to reduce the size of the tree and thus to get a more compact knowledge representation. The best performance obtained with a very compact representation (the evaluation was 10-fold stratified cross-validation) was obtained with the following model:

J48 pruned tree

----

gust_p90 <= 87

| rays_uk <= 5: Other (500.0/23.0)

| rays_uk > 5

| | rays_uk <= 205

| | | rain <= 13: Other (45.0/15.0)

| | | rain > 13: Atmospheric (30.0/4.0) -> 6% atmosfericas 0,7% Other

| | rays_uk > 205: Atmospheric (130.0/7.0) -> 28,38% atmosfericas 1,3% descargo(Other)

gust_p90 > 87: Atmospheric (288.0/17.0) -> 68% atmosfericas 0,03% descargo(Other)

Number of Leaves : 5

Size of the tree : 9

=== Stratified cross-validation ===

Correctly Classified Instances	926	93.2528 %
Incorrectly Classified Instances	67	6.7472 %
Kappa statistic	0.8641	
Mean absolute error	0.1167	
Root mean squared error	0.2434	
Relative absolute error	23.4781 %	
Root relative squared error	48.8339 %	
Total Number of Instances	993	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.921	0.058	0.932	0.921	0.926	0.934	Atmospheric
	0.942	0.079	0.933	0.942	0.938	0.934	Other
Weighted Avg.	0.933	0.069	0.933	0.933	0.932	0.934	

=== Confusion Matrix ===

a b <-- classified as
422 36 | a = Atmospheric
31 504 | b = Other

From this decision tree a set of rules can be derived that will give us an idea about the weather conditions that are usually associated with atmospheric faults, the highlighted lines of the trees represent the information to create the IF-THEN rules.

gust_p90 <= 87
| rays_uk <= 5: Other (500.0/23.0)
| rays_uk > 5
| | rays_uk <= 205
| | | rain <= 13: Other (45.0/15.0)
| | | rain > 13: **Atmospheric (30.0/4.0)**
| | **rays_uk > 205: Atmospheric (130.0***/7.0)* ***-> 28,38% atmosfericas*** *1,3% descargo(Other)*
gust_p90 > 87: Atmospheric (288.*0/17.0) -> **68% atmosfericas** 0,03% descargo(Other)*

Knowledge Representation

From the decision tree model obtained some rules are derived to represent the knowledge obtained in a more straightforward way. The goal is to obtain a synthesis of the knowledge obtain in the model based on the coverage of the atmospheric faults and the level of confidence of the classification. These two attributes are considered to be the most relevant ones to decide on the importance of each rule.
The derived rules are:

Rule1: *IF the 90percentil of wind gust is above 87 THEN Atmospheric*
Rule 2: *(IF the 90percentil of wind gust is less than 87) AND (IF the level of lighthings(UK) is above 205) THEN Atmospheric*
Rule 3: *(IF the 90percentil of wind gust is less than 87) AND (IF rays_uk is between 5 and 205) AND (IF rain is above 13) THEN* Atmospheric

The levels of coverage and confidence are shown in the following table:

	Rule 1	**Rule 2**	**Rule 3**
Level of confidence	94.10%	94.61%	86.67%
Coverage (atmospheric)	59.17%	33.41%	5.67%

These rules can give an idea of the weather conditions that can bring an atmospheric fault. The level of coverage of only these 3 rules reaches around 98,25% of all the atmospheric faults detected.

The selection of the non-atmospheric fault was done by means of selecting another cause with a similar number of faults, this was a biased selection and therefore a new subset selection was needed in order to have a more representative selection of the non-atmospheric faults. Thus a stratified sampling of the non-atmospheric faults data was performed in order to get a more

reliable solution for atmospheric faults. All atmospheric faults were selected and a subset of the remaining faults based on their probability of appearance (stratified sampled) were selected.

The following table shows the number of instances for each type of fault labeled by Union Fenosa.

Cause	Count	Percentage
Own transmission	1	0,03%
Strike	3	0,09%
External transmission	4	0,12%
Generation	13	0,39%
Accidental or intentional	14	0,42%
Other DNO	31	0,92%
External Agent	35	1,04%
Intern	111	3,29%
U unknown	398	11,81%
Atmospheric	465	13,79%
Descargo	546	16,20%
Particular facility	625	18,54%
REE transmission	1125	33,37%
Total	**3371**	**100,00%**

The data studied was:

- 465 atmospheric faults
- 461 other faults (stratified sampled from all possible faults)
  Total faults: 926

The decision tree obtained using the J48 was the following:

J48 pruned tree
---

gust_p50 <= 77
| rays_uk <= 70
| | gust_p90 <= 67: OTHER (363.0/24.0)
| | **gust_p90 > 67**
| | | **wind_p50 <= 26: Atmospheric (72.0/26.0)**
| | | wind_p50 > 26: OTHER (82.0/28.0)
| rays_uk > 70: Atmospheric (164.0/22.0)
gust_p50 > 77: Atmospheric (245.0/20.0)

Number of Leaves : 5

Size of the tree : 9

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	781	**84.3413 %**
Incorrectly Classified Instances	145	15.6587 %
Kappa statistic	0.6868	
Mean absolute error	0.2295	
Root mean squared error	0.3471	
Relative absolute error	45.9058 %	
Root relative squared error	69.4143 %	
Total Number of Instances	926	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.82	0.133	0.859	0.82	0.839	0.894	OTHER
	0.867	0.18	0.829	0.867	0.848	0.894	Atmospheric
Weighted Avg.	0.843	0.157	0.844	0.843	0.843	0.894	

=== Confusion Matrix ===

a	b	<-- classified as
378	83	a = OTHER
62	**403**	**b = Atmospheric**

The results obtained in this case show reliable results for the detection of atmospheric faults.

Knowledge Representation

From the decision tree model the following rules are derived as a synthesis of the knowledge obtain in the model based on the coverage of the atmospheric faults and the level of confidence of the classification.

The derived rules are:

Rule1: *IF the 50percentil of wind gust is above 77 THEN Atmospheric*
Rule 2: *(IF the level of lighthings(UK) is above 70) THEN Atmospheric*

The levels of coverage and confidence are shown in the following table:

	Rule 1	**Rule 2**
Level of confidence	91.84%	86.59%
Coverage (atmospheric)	48.39%	30.54%

There is a third rule that can be derived from the J48 model, but the coverage it is less than 1 % and the level of confidence is less than 75%. In this case this rule is not considered.

Rule 3: *(IF the 90percentil of wind gust is more than 67) AND (IF 50 percentile of wind is less than 26) THEN* Atmospheric

	Rule 3
Level of confidence	63.88%
Coverage (atmospheric)	<1%

It is important therefore to set thresholds for both the coverage and the level of confidence.

Alternating decision tree analysis

An alternating decision tree (ADTree) is a machine learning method for classification that generalizes decision trees introducing connections to boosting. An alternating decision tree consists of decision nodes and prediction nodes. Decision nodes specify a predicate condition. Prediction nodes contain a single number. ADTrees always have prediction nodes as both root and leaves. An instance is classified by an ADTree by following all paths for which all decision nodes are true and summing any prediction nodes that are traversed.
This is different from decision trees in which an instance follows only one path through the tree.

A study using alternating decision trees was performed to see if this variant of the concept of decision tree could improve the results obtained so far. The data used for this study is the same as the previous study:

- 465 atmospheric faults
- 461 other faults (stratified sampled from all possible faults)
  Total faults: 926

The results obtained using alternating decision tree was the following:

Alternating decision tree

: 0.004
| (1)gust_p50 < 77.5: -0.308
| (1)gust_p50 >= 77.5: 1.184
| (2)rays_uk < 32: -0.376
| (2)rays_uk >= 32: 1.099

Legend: -ve = OTHER, +ve = Atmospheric
Tree size (total number of nodes): 7
Leaves (number of predictor nodes): 5

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	774	**83.5853 %**
Incorrectly Classified Instances	152	16.4147 %
Kappa statistic	0.6717	
Mean absolute error	0.36	
Root mean squared error	0.3902	
Relative absolute error	71.9942 %	
Root relative squared error	78.0354 %	
Total Number of Instances	926	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.833	0.161	0.837	0.833	0.835	0.863	OTHER
	0.839	0.167	0.835	0.839	0.837	0.863	Atmospheric
Weighted Avg.	0.836	0.164	0.836	0.836	0.836	0.863	

=== Confusion Matrix ===

a	b	<-- classified as
384	77	a = OTHER
75	**390**	**b = Atmospheric**

In this case we obtained a lower precision in detecting the atmospheric faults compare to the J48 algorithm. We do not go further in this study since different configurations have been tested and all models obtained gave lower performances than the obtained with J48. Furthermore our goal is to represent the knowledge obtained in the form of IF-THEN rules (human readable format), and this technique not only offers lower performance for the data studied but also brings more complexity since as pointed out before the main different between alternating decision trees and decision trees is that an instance follows only one path through the tree for a decision tree and several for alternating decision tree, making more complex the transformation to a IF-THEN rules knowledge representation.

Knowledge representation: Algorithm proposed

As introduced before the learning process of the algorithm proposed can be summarize as follows:

- Creation of decision trees: This was explained in the previous paragraphs with the studies performed with J48. The current prototype of the algorithm is implemented in R language and the rpart library of R is used during the process of creation of the decision trees.
- Knowledge representation step to obtain IF-THEN rules.
- Unification step of the IF-THEN rules

- Rules Ranking and selection: based on level of confidence and coverage.

The following table shows an example of the knowledge representation obtained by running the implemented algorithm by selecting only rules which offer a level of confidence above 80% and covers more than 10% of the atmospheric faults.

covered	confidence	Rules
48,5411141	100	IF gust_p90 >= 90 AND rain < 23.5 AND gust_p50 >= 78.5 THEN Atmospheric
48,2288828	93,15789474	IF gust_p90 >= 90 AND gust_p50 >= 77.5 THEN Atmospheric
46,8319559	100	IF gust_p50 >= 78.5 AND rain < 23.5 THEN Atmospheric
45,5913978	100	IF gust_p90 >= 90 AND rain < 23.5 AND gust_p50 >= 77.5 THEN Atmospheric
23,8709677	97,36842105	IF gust_p90 < 44.5 AND rays_uk >= 190 THEN Atmospheric
22,0385675	98,7654321	IF gust_p50 < 67.5 AND rays_uk >= 190.5 AND gust_p90 < 44.5 THEN Atmospheric
19,346049	98,61111111	IF gust_p90 < 44.5 AND 3229 > rays_uk >= 72.5 THEN Atmospheric
11,8918919	100	IF gust_p90 < 42.5 AND rain >= 2.5 AND gust_p50 >= 31.5 THEN Atmospheric

Index creation

The final index for the level of criticality of the current weather conditions will be based on the rules selected. A weighted sum will be taken based on the coverage and confidence of the rules. Several indexes are being tested for the final system. The first index is test as follows:

- The first index is a very basic one and represents the percentage of the rules that evaluates positive to the current meteorological conditions. Therefore it performs a simple division between the counts of positive rules and the number of rules of the model.
- The second index is a weighted index taking into account the level of confidence of the rules: rules with higher level weight more than the ones with less. The operation performed for this index is the sum of all levels of confidence of the rules that applied for the current weather condition divided by the sum of all rules contained in the model.
- The third index is a weighted index taking into account not only the level of confidence but also the coverage. Therefore a rule having higher coverage will have a bigger impact on the result.

3.2.2.2 EG Data Analysis

Weather is one of the major factors affecting power distribution systems Not only weather conditions affects the reliability of power distribution systems as discussed previously when describing the Weather Fault Application but also energy consumption is highly dependent on the weather conditions, specially the temperature.

EG is performing certain studies to find information regarding the correlation of the weather conditions to the load level in the network to be able to do some predictions regarding this influence.

EG Weather - Load Data

The following data has been provided by EG. The data consists in a set of attributes relating load levels in the network with some meteorological conditions. The different information contained in the data are:

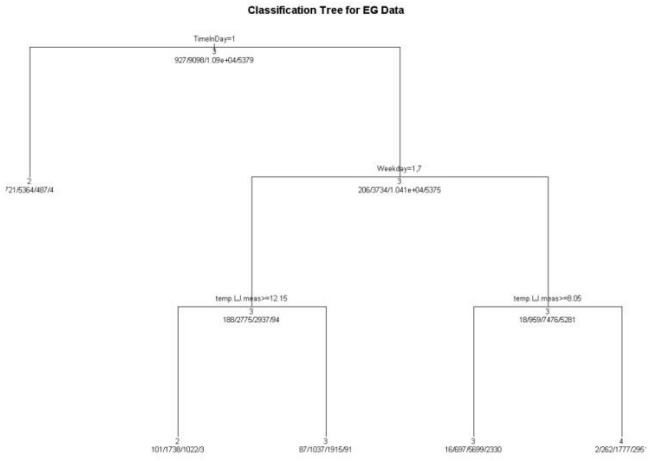
- Date
- Year
- Month
- Hour
- Weekday
- Season
- TimeInDay
- Network Load
- Temperature
- Solar irradiance

Weekday and TimeInDay are especially useful attributes since it is known that depending of the type of day (Monday, Saturday, Sunday,…) and the time of the day consumption patterns are very different.

UF Weather - Fault Data Application

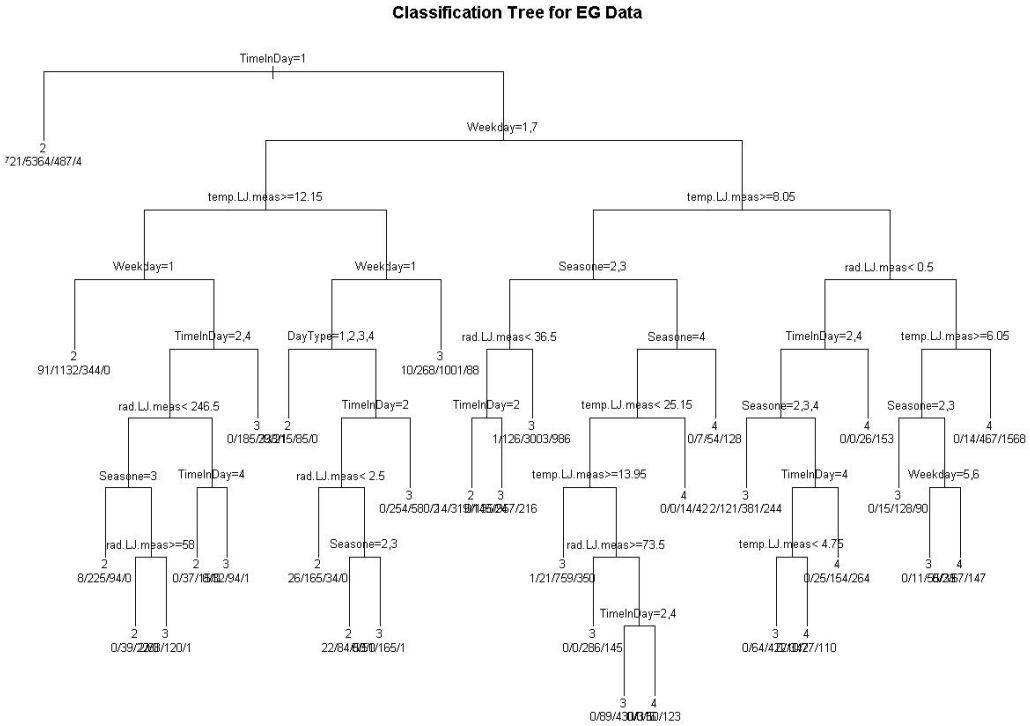
The goal of the application is to explain the correlations between the load level in the network and the type of day and weather conditions.

The first analysis performed with this data was the creation of simple decision trees to explain the dependencies/correlations. The load level attribute was discredited in 4 levels: 1 –very low, 2- low, 3–medium, 4-High. The underlying idea was to obtain a first explanation regarding the most important variables influencing the load levels in the network.



The first experiments showed that the most important variables were TimeInday, Weekday and the temperature (as seen in the previous figure), which are actually known variables to have an important dependency on the load of the network.

Further analysis was performed were bigger decision trees were built to show other dependencies. The following figure shows a more complex tree. Notice that the most important variables, the first ones to appear on the tree, there are still the ones found before, since decision trees always show first splits that are more important.



Knowledge representation: Algorithm proposed

The implemented algorithm introduced in the previous section was applied to extract the knowledge obtained by several decisions trees as IF-THEN rules, which will make easier the interpretation of the results.

Covered	confidence	Rules
58.96	81.57	IF TimeInDay = 1 THEN Low Load
19.10	60.68	IF TimeInDay = 2,3,4 AND Weekday = 1,7 AND temp.LJ.meas >= 12.15 THEN Low Load
17.57	61.18	IF TimeInDay = 2,3,4 AND Weekday = 1,7 AND temp.LJ.meas < 12.15 THEN Medium Load
52.28	65.19	IF TimeInDay = 2,3,4 AND Weekday = 2,3,4,5,6 AND temp.LJ.meas >= 8.05 THEN Medium Load
54.86	59.11	I IF TimeInDay = 2,3,4 AND Weekday = 2,3,4,5,6 AND temp.LJ.meas < 8.05 THEN High Load

For example the first rule establish that during the first hours of the day (from 00:00 – 6:00) the load level is low, since it is the time of the day that most customers are sleeping and therefore their consumption is low.

3.3 Clustering Correlated Uncertain Sensor Data

3.3.1 Introduction

Smart Grids heavily rely on sensor data: millions of smart meters (essentially sophisticated sensors) are being rolled out into households throughout the European Union to increase our understanding of dynamics of power demands [21], whilst the power networks are monitored using sensors on switchgear and cables, both in- and outside substations [22], [23].

Sensor data is well-known to be uncertain. Large sensor networks are often constructed using cheap hardware, which is prone to malfunctioning. In addition, communication channels used to transmit sensor data introduce an additional risk factor: some sensor readings are lost, or transmitted incorrectly. Ignoring this uncertainty when processing data from such sources can be harmful. Hence, analysis and mining of uncertain data is one of the components of the HiPerDNO project.

Additionally, uncertain data often contains strong correlations. For example: two readings of the same sensor are heavily correlated, as are readings from different closely located sensors. Also, features are strongly correlated, e.g. increasing power consumption will often lead to an increasing temperature.

In the first year of HiPerDNO, we investigated the correlation between load and partial discharge, which led to a classification algorithm for the early detection of suspicious partial discharge activity [24]. In the past year, we worked on a more generic data mining technique: clustering correlated uncertain data. This report contains a high-level description of this approach, referencing sections, expressions and figures from a paper submission (currently under review) titled “Clustering Correlated Uncertain Data”, which has been included as Appendix A.

3.3.2 Quantifying Uncertainty

Although it is clear that data from sensors in Smart Grids should be considered “uncertain” and processed as such using techniques for data processing that have been proposed during the last decade, quantifying the uncertainty and the correlations in play has not been investigated in very much detail.

In our paper submission, we introduce an approach to quantifying uncertainty which is geared towards a setting in which multiple sensors are located reasonably close to each other (monitoring the same or similar hardware), for example multiple temperature sensors. Each sensor is assumed to function correctly with a probability, represented by a Boolean probabilistic random variable which can be used in a propositional formula (as illustrated in Figure 1, Appendix A). Each measurement is compared to the values reported by the other sensors. If two sensors agree (i.e., report similar values), the measurement values are aggregated into a single data point, annotated by a *disjunction* of the respective sensor variables. The disjunction represents the event: “at least

one of the sensors is functioning properly". Alternatively, if two (or more) sensors disagree, the two values are included as separate data points, annotated by a *conjunction* of a variable together with *negated* variables of the sensors that are conflicting. The conjunction represents the event "this sensor is working and the others are malfunctioning". Arguably, if a sensor is functioning properly at time *t*, we can assume it will still report accurate measurements at time *t+1*. We can store the expressions (in the domain of propositional calculus) alongside the data using so-called "pc-tables" (probabilistic conditioned tables).

In our experiments (Section 6, Appendix A), we used both data generated using random Gaussian distributions and sensor data from smart meters, generated using occupancy models from project work package 1.4 (also see Richardson et al. [38]).

As expressed, the approach to quantifying uncertainty and correlations described above is specific to a specific configuration of sensor networks. In order to be able to assess the uncertainty and correlations in data in a more generic setting, we are currently investigating the possibility to leverage Markov Logic Networks [25]. This approach is well-established in the field of artificial intelligence and multiple algorithms exist to construct Markov Logic Networks (learn correlations), and to infer the probabilities inside such networks.

Note that [24] contains an approach to quantifying uncertainty and correlations in streams of load and partial discharge data.

3.3.3 The ϕ k-medoids Algorithm

In Appendix A, we introduce the ϕk-medoids algorithm, which is based on the widely-used k-medoids algorithm. The k-medoids algorithm endeavours to find *k* clusters in the input data, using only a dissimilarity (or distance) function $d(o_a,o_b)$ defined over all pairs of data points. Each of the *k* clusters is represented by its *medoid*: the data point which has the minimum total distance to all other data points in the cluster. The algorithm consists of three phases:

1.*Initialisation* phase: initial cluster medoids are chosen (randomly, or using a heuristic). These medoids are not necessarily close to the centres of the clusters the algorithm eventually finds.
2.*Assignment phase*: all data points are assigned to the closest medoid.
3.*Update phase*: based on the assignment of data points, the new cluster medoids are determined.

After the initialisation, the algorithm repeats the assignment and update phases until either (1) convergence or (2) the preset maximum number of iterations has been reached. The traditional k-medoids algorithm is unable to deal with existential uncertainty: it assumes that all data points exist with full certainty.

Traditional algorithms (like k-medoids) can be run on probabilistic databases by instantiating the exponentially many (in terms of the number of data points) *possible worlds* and running the algorithm on each and every single world. A probability is attached to each possible world, based on the subset of tuples from the probabilistic database D it contains. This probability can be used to weigh the algorithm results into an aggregated, probabilistic result. Note that in a pc-tables setting,

the number of possible world is exponential in the total number of variables. Furthermore, the existence of tuples in each world depends on the total valuation of variables (and hence, depends on correlations).

The ϕk-medoids algorithm does not explicitly enumerate the exponentially many possible worlds in the probabilistic database, but yields the same probabilistic result. The input of the algorithm is a probabilistic database based on pc-tables, and it will produce a probabilistic result using the same pc-tables. Hence, our approach is closed under these semantics, and the output can be used for further processing in probabilistic databases. Alternatively, one can perform sensitivity analysis on the probabilistic clustering result.

Instead of assigning objects to clusters and selecting cluster medoids in a deterministic way, the ϕk-medoids algorithm constructs probabilistic events that describe the assignment of objects to clusters and the selection of new medoids. In each iteration, two *n* x *k* matrices are constructed containing expressions that represent these two types of probabilistic events:

- $\phi[o_i \in C_j]$: represents the event that data point o_i is assigned to cluster C_j
- $\phi[c_j = o_i]$: represents the event that data point o_i is chosen as cluster medoid of C_j

These events are represented using expressions of propositional logic. However, in order to be able to construct expressions $\phi[c_j = o_i]$ (which depend on the minimum distance sum of uncertain objects), we had to apply propositional logic on the $B \otimes R$ semimodule generated by the variables used in the input pc-table. More details on the use of the $B \otimes R$ semimodule can be found in Appendix A.

The output of the algorithm consists of two n x *k* matrices with expressions constructed during the last iteration of the algorithm. These matrices represent the assignment of objects to clusters (which can be seen as a per-object discrete probability distribution over all clusters) and the selection of cluster medoids (per-cluster discrete probability distribution over objects). This output can be used in many ways, which are discussed in great detail in Appendix A. In our work, we introduce the "pairwise similarity" measure $s(o_a,o_b)$ of two data points, using the expressions in the matrix $\phi[o_i \in C_j]$. The pairwise similarity can be expressed using a propositional formula which is the disjunction of all possible events in which the two objects o_a,o_b are assigned to the same cluster.

The expressions that represent the events (including the pairwise similarity measure) can be compiled into probabilities using an extended version of Shannon's expansion, the details of which can be found in Appendix A.

3.3.4 Experiments

The ϕk-medoids algorithm is the first algorithm with full support for correlations in the existential uncertainty in the input data. Data points that are negatively correlated have a small (or zero, in case of mutually exclusive objects) probability of being assigned to the same cluster, whereas data points that are positively correlated are more likely to end up in the same cluster. Unfortunately, there are – to the best of our knowledge – no other approaches which deal with uncertainty in this way. This means evaluation of our algorithm (both in terms of time performance and

quality/accuracy) is not straightforward. Other clustering algorithms for uncertain data assume independence over the input tuples. Comparing to such algorithms is impossible, as the goal of both types of algorithms is fundamentally different:

– ϕk-medoids is designed to adhere to the possible world semantics and will hence endeavour not to put two negatively correlated objects (however close they are to each other in the feature space) in the same cluster.
–Any other clustering algorithm will only consider the dissimilarity function $d(o_a,o_b)$ and will ignore correlations.

Comparing the output of ϕk-medoids to any other clustering algorithm for uncertain data would yield a significant disadvantage for either one of the algorithms (depending on how the clustering result is evaluated). If the evaluation method does not take correlations into account, ϕk-medoids will be penalised for not putting two closely located, but negatively correlated data points into the same cluster. On the other hand, if the evaluation method does take correlations into account, this will be a disadvantage for the algorithm that is oblivious with respect to correlations: it will be penalised for putting two negatively correlated objects in the same cluster.

We have, however, compared ϕk-medoids to a naïve approach which respects correlations by explicitly enumerating all possible worlds. The results of this experiment are documented in Appendix A.

4 DM Platform

Future electricity distribution network operators (DNO) with mass deployment of network equipment sensors will generate vast amounts of data, which requires analysis in order to turn the data into actionable information. To meet these challenges DNOs can benefit from the use of techniques recently developed to cost-effectively solve large scale computational problems in areas such as Biology, Finance and Web Services. In such systems, increased access to ubiquitous sensing and the web has resulted in an explosion in the size of data mining and machine learning tasks, which in turn, driven the growing demand for *scalable* implementations of machine learning algorithms on very large datasets (ranging from 100s of GBs to TBs of data).

In the meantime, physical and economic limitations have forced computer architecture towards parallelism and away from exponential frequency scaling. In general, parallel computing - often called distributed computers - deals with hardware and software for computation in which many calculations are carried out simultaneously. There are different types of existing architectures and technologies for parallel computing but in this report we focus on *commodity computing cluster*. A computer cluster is a group of shared individual computers, linked by high-speed communications in a local area network, and incorporating system software which provides an integrated parallel processing environment for applications with the capability to divide processing among the nodes in the cluster.

In order to benefit from current and future trends in parallel computing there are several attempts at building scalable distributed data mining platforms on top of commodity computing clusters. More concretely, a *scalable data mining platform* is a parallel computing application that includes a collection of fundamentals algorithms in machine learning. That is the algorithm's computation is distributed on large set of cluster's nodes rather than processed on a single core machine. Scalable data mining platforms are a key ingredient in the development process of large scale data mining applications. Such software platforms includes a collection of fundamentals algorithms in machine learning (e.g. clustering, classification, dimension reduction, regression analysis and pattern mining), which are implemented in a parallel programming paradigm in order to run on top of commodity computing clusters. As a result, these scalable data mining platforms are indispensable to the data miner as they aim to assist building large-scale intelligent systems easier and faster.

Currently, scalable data mining platforms are expensive and selection of the wrong platform can be costly in many ways. The cost of selecting an improper scalable data mining platform for a particular application is even more costly in terms of personnel resources, development time, and the potential for acting on spurious results. Moreover, evaluating scalable data mining platforms is not simply a matter of selecting the best tool for all purposes. Instead a DNO must consider the platforms with respect to their particular environment, and analysis needs.

To better understand and to evaluate the different scalable data mining platforms that are available, we adopted three major categories of criteria for evaluating scalable data mining platforms: functionality, usability, scalability and performance. For a complete and elaborate review on scalable data mining platforms the reader is referred to the HiPerDNO deliverable 1.3.2, which compares two

leading public and open source scalable data mining platforms coming from two different approaches using the above well established categories for evaluating scalable data mining software within a smart gird environment.

In deliverable 3.1.1 we address compatibility issues of the HPC platform suggested in HiPerDNO deliverable 1.2.1 and 1.2.2 to the data mining platforms introduced in HiPerDNO deliverable 1.3.2. In general terms, it should be noted here that all candidate data mining platforms suggested HiPerDNO deliverable 1.3.2 are open source scalable data mining platforms, which are developed under public license. Therefore, deploying them on top of HPC platform will not incur additional cost to the DNOs. However, given the additional security requirements adopted after project 1st year review and the restriction to available schedulers in PelicanHPC it seems that Parallel GNU R is the most suitable for the HiPerDNO project. With reference to resource tool manager (a.k.a. scheduler), Parallel GNU R jobs can be easily scheduled with various commodity HPC scheduler, therefore, it should be straight forward to manage Parallel GNU R jobs on HiPerDNO computational platform. GNU R is ideally suited to the many challenging tasks associated with data mining. In fact, according to Rexer's Annual Data Miner Survey in 2010, GNU R has become the data mining tool used by more data miners (43%) than any other. Furthermore, the GNU R platform has become a de facto standard among statisticians for developing statistical software, and is widely used for statistical software development and data analysis. Not surprisingly, GNU R is already adopted in utilities companies for their analysis needs. For example, IPEC and National Grid in the UK both use R platform for their data analysis process. Taking all of this into account, it seems that GNU R perfectly fits the needs and requirements of the HiPerDNO project.

5 Discussion and Conclusion

Data mining techniques have been successfully applied in power systems in several areas such as security assessment, fault detection, power system control, load forecasting, load profiling. In this report, we introduced new practical machine learning applications designed to tackle real problems that DNOs face in their operations. Our objective is to demonstrate that data collected by power companies can be used to create statistical models for proactive maintenance, to exemplify how this can be accomplished through state-of-the–art data mining techniques, and show how DNOs can be most effective in building and developing predictions and decision support applications.

We introduced a data mining framework for automatic pulse separation, including FE, FS, unsupervised clustering analysis and clustering result validation. In the process of FE, PCA has been shown to be the suitable dimension reduction technique by extracting the majority of the variation in the original data set. In addition, we explained the relation between PCA and SVD explicitly and the filtering effects of SVD on the data. A simple test on the second-order rate of change of the singular values was used to decide the number of PCs needed for a sufficient summary of the data. In the process of FS, we show NACF, the previously suggested pulse shape feature, and the raw pulse data show different patterns in the similarity matrix. After applying FE technique on both data sets, the clusters found from the raw data have much better quality, i.e., the clusters are more compact and well separated. Hence, we have chosen the raw data as the feature for cluster analysis. In the process of cluster analysis, we have stressed the need for cluster validation in order to discover the most appropriate unsupervised clustering method and to estimate the number of clusters for the separation process PD signals. Experimental results have shown that using several indexes gives greater confidence in choosing the appropriate unsupervised clustering and determining the correct number of clusters.

We developed a condition monitoring application of overhead lines based on weather conditions using decision tree analysis. In general terms, decision trees are data mining tools designed to face classification problems which have proven to be very useful in several power systems applications [39]. They possess a great versatility and they are adequate to be applied to very diverse different real applications. Their key advantage radicates on the easy interpretability of the results, they can extract human readable information about the underlying process. In power systems decision trees have been found suitable for diverse classification tasks where they were proven to be effective in combining real-time possibilities, accuracy, robustness to noise and interpretability of the results. Furthermore decision trees showed high accuracies also with a small number of input features. They can be constructed in a short period of time and use on-line since the tree evaluation does not require any time-consuming computation. Based on decision trees an algorithm for knowledge representation has been implemented, which extract the knowledge learnt by several decision trees in the form of IF-THEN rules which can be easily interpretated by domain experts. This new algorithm has been applied for two different task: correlation between weather conditions and faults in the distribution network and relation between network load levels and day and weather conditions. In both applications the algorithm is a powerful tool to automatically detect the main variables involved in the problem and offer a descriptive explanation which can be contrasted with expert knowledge. Furthermore, it can be used to automatically warn about possible situations that in the past led to faults or to high loads in the distribution network. For example in the case of Union Fenosa faults and weather data, if the current weather

conditions are similar to weather conditions that led in the past to faults maintenance teams can be warned and prepared in case a contingency occurs. In the case of EG data, if the current day and weather conditions are similar to the ones that caused in the past very high load levels in the distribution network, the system can be warned about a possible higher load in the network.

Last, our work on clustering uncertain data with arbitrary correlations can be used to increase the accuracy of the clustering result, in case sensors in the energy distribution network (or the transmission channels used to transfer sensor data) fail. Using our newly designed ϕk-medoids algorithm, it is also possible to perform efficient retrospective corrections and sensitivity analysis using newly available data on the (mal)functioning of sensors. Although it has become clear that uncertainty introduces a non-trivial extra layer of complexity, we endeavour to use the HPC platform to speed up the data mining process. Also, alternative clustering techniques (and in general: alternative data mining techniques) will be investigated to see whether these can be adapted for efficient use in a probabilistic setting.

6 References

[1] A. Contin and G.C.Montanari and G.Pasini and F.Puletti, “Digital Detection and Fuzzy Classification of Partial Discharge Signals”. IEEE Trans. Dielectr. Electr. Insul., vol. 9, no. 3, pp. 335–348, Jun 2002.

[2] A.Cavallini, A.Contin, G.C.Montanari, and F. Puletti, “Advanced pd interference in on-field measurements. part i: Noise rejection ”, IEEE Trans. Dielectr. Electr. Insul., vol. 10, no. 2, pp. 216-224, Apr 2003.

[3] A. Contin and S. Pastore, “Classification and separation of partial discharge signals by means of their auto-correlation function evaluation”, IEEE Trans. Dielectr. Electr. Insul., vol. 16, no. 6, pp. 1609 -1622, Dec 2009.

[4] L. Hao, P. L. Lewin, and S. G. Swingler, “Use of machine learning for partial discharge discrimination”, in The 11th International Electrical Insulation Conference, 2009.

[5] L. Hao, P. Lewin, J. Hunter, D. Swaffield, A. Contin, C. Walton, and M. Michel, “Discrimination of multiple pd sources using wavelet decomposition and principal component analysis”, IEEE Trans. Dielectr. Electr. Insul., vol. 18, no. 5, pp. 1702 -1711, october 2011.

[6] I. Jolliffe, Principal Componnet Analysis, 2nd ed. New York, USA:Springer-Verlag New York, 2002.

[7] J. Jackson, A user’s guide to principal components. USA: Wiley, 1991.

[8] D Evagorou, A Kyprianou, G E Georghiou, L Hao, P Lewin, and A Stavrou. “Multisource PD identification based on phase synchronous and asynchronous data”, IEEE Conference on Electrical Insulation and Dielectric Phenomena. 2011.

[9] H. Al-Marzouqi and A. Contin. “Separation of multiple sources in PD measurements using an intensity based clustering algorithm”, IEEE Conference Electrical Insulation (ISEI). 2010.

[10] L Hao, A Contin, Jack Hunter, D J Swaffield, P L Lewin, C Walton, and M Michel. “A new method for automatic multiple partial discharge classification”, In 17th International Symposium on High Voltage Engineering, 2011.

[11] G. Brock, V. Pihur, S. Datta, and S. Datta. “clvalid: An r package for cluster validation”. Journal of Statistical Software, 2008.

[12] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, “On clustering validation techniques”, Journal of Intelligent Information Systems, vol. 17, pp. 107-145, 2001.

[13] R. Xu and D. Wunsch, “Survey of clustering algorithms”, IEEE Transactions on Neural Networks, pp. 645-678, 2005.

[14] M. Ester, H. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise”, in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996.

[15] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning. Springer New York Inc., 2001.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm”, Journal of The Royal Statistical Society, 1977.

[17] P. Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” Journal of Computational and Applied Mathematics, pp. 53–65, 1987.

[18] J. C. Dunn, “Well separated clusters and fuzzy partitions,” Journal of Cybernetics, pp. 95–104, 1974.

[19] Y. Zhou, A. Pahwa, S.S. Yang, “Modeling of Weather-Related Failures of Overhead Distribution Lines”, IEEE Trans. on PWRS, Vol. 21-4, November 2006, pp. 1683-1690.

[20] I. Guyon, A. Elisseeff, “An introduction to variable and feature selection”, Journal Machine Learning, 2003.

[21] S. Attree, J. Kay, “The value of reducing distribution losses by domestic load-shifting: a network perspective”, Energy Policy, 2009.

[22] M. Michel, “Innovative asset management and targeted investments using on-line partial discharge monitoring and mapping techniques”, CIRED, 2007.

[23] M. Michel, C. Eastham, “Improving the management of MV underground cable circuits using automated on-line cable partial discharge mapping”, CIRED, 2011.
[24] D. Olteanu, S. Van Schaik “A data mining approach to fault detection in uncertain measurement data”, (technical report) 2010.
[25] P. Domingos, D. Lowd “Markov Logic – An Interface Layer for Artificial Intelligence”, Morgan & Claypool publishers, 2009.
[26] E. Lobato, A. Ugedo, L. Rouco, M. Echavarren, “Decision Trees Applied to Spanish Power Systems Applications”, 9th International Conference on Probabilistic Methods Applied to Power Systems KTH, 2006
[27] L. Wehenkelt and M. Pavella, “Decision Trees and Transient Stability of Electric Power Systems”, Automatica, Volume: 27, Issue: 1, 1991, pp. 115-134
[28] S. P. Teeuwsen, I. Erlich, M. A. El-Sharkawi, “Decision Tree based Oscillatory Stability Assessment for Large Interconnected Power Systems”, IEEE Power Systems Conference and Exposition, 2004.
[29] K.S. Swarup, R. Mastakar, K.V Reddy, “Decision tree for steady state security assessment and evaluation of power systems”, Conference on Intelligent Sensing and Information Processing, 2005.
[30] L. Zhiyong, W. Weilin, “Phasor Measurements-Aided Decision Trees for Power System Security Assessment”. Second International Conference on Information and Computing Science, 2009.
[31] J. Ma, R. Diao, Y.V Makarov, P.V Etingo, N. Zhou, J.E Dagle, “Building decision trees for characteristic ellipsoid method to monitor power system transient behaviors”. IEEE Power and Energy Society General Meeting, 2010.
[32] R. Sun, W. Zhongyu, V. Centeno, “Power system islanding detection & identification using topology approach and decision tree”, IEEE Power and Energy Society General Meeting, 2011.
[33] H. Kriegel, M. Pfeifle, “Hierarchical Density-Based Clustering of Uncertain Data”, Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM ’05), 2005.
[34] A. Tepwankul, S. Maneewongwattana, “U-DBSCAN: A density-based clustering algorithm for uncertain objects”. IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010), 2010.
[35] F. Gullo, G. Ponti, A. Tagarelli, S. Greco, “A Hierarchical Algorithm for Clustering Uncertain Data via an Information-Theoretic Approach”. 2008 Eighth IEEE International Conference on Data Mining (ICDM ’08), 2008.
[36] D. Suciu, D. Olteanu, C. Ré, C. Koch, “Probabilistic Databases”. Morgan & Claypool Publishers, 2011.
[37] C. Aggarwal (editor), “Managing and Mining Uncertain Data”. Springer, 2009
[38] I. Richardson, M. Thomson, D. Infield, C. Clifford, “Domestic electricity use: a high-resolution energy demand model”. Energy and Buildings, 42, 2010.
[39] H. Mori, “State-of-the-Art Overview on Data Mining in PowerSystems” Power Systems Conference and Exposition (PSCE ‘06), 2006.
[40] C. Rudin, D. Waltz, R. N. Anderson, A. Boulanger, A. Salleb-Aouissi, M. Chow, H. Dutta, P. Gross, B. Huang, S. Ierome, D. Isaac, A. Kressner, R. J. Passonneau, A. Radeva, L. Wu. "Machine Learning for the New York City Power Grid." In TPAMI, 2011.
[41] R. Liao, G. A. Taylor, M. R. Irving, “Statistical Analysis of Partial Discharge Data Based on Master Equation”. In UPEC 2011.