5-1-2013

Fault analysis using state-of-the-art classifiers

Girish Chandrashekar

Fault Analysis using State-of-the-art Classifiers

by

Girish Chandrashekar

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in Electrical Engineering

Approved by:

Dr. Ferat Sahin, *Thesis Advisor*

Dr. Athimoottil Mathew, *Committee Member*

Dr. Vincent Amuso, *Committee Member*

Department of Electrical and Microelectronic Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology,
Rochester, New York.
May 2013

Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Fault Analysis using State-of-the-art Classifiers

I, Girish Chandrashekar, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Girish Chandrashekar

Date

Acknowledgments

I would like to express my deepest appreciation to my advisor Dr. Ferat Sahin for his continued support and encouragement over the past three years at RIT. Without his guidance and persistent help this thesis would not have been possible.

I am very thankful to my family members and friends for their endless motivation to help me pursue my M.S. degree. I would also like to thank Dr. Daniel Philips and Dr. Sergey Lyshevski for their guidance during my study at RIT. I am also very thankful to my colleagues Vincent Baier, Eyup Cinar and Ryan Bowen for their support in my academic life and research. Lastly I would like to thank Aaron Radomski and Dan Sarosky from MKS Instruments who helped me in providing data and feedback towards my work for MKS Instruments.

List of Publications

[1] Girish Chandrashekar and Ferat Sahin, “In-Vivo Fault Prediction for RF Generators using Variable Elimination and State-of-the-art Classifiers ”, 2012 *IEEE International Conference on Systems, Man, and Cybernetics,* October 14-17, 2012, COEX, Seoul, Korea.

[2] Girish Chandrashekar and Ferat Sahin, “A Survey on Feature Selection Methods ”, under review in *Journal of Computers & Electrical Engineering*, Elsevier.

[3] Girish Chandrashekar and Ferat Sahin, “In-vivo Fault Analysis and real-time Fault Prediction for RF Generators using state-of-the-art classifiers ”, under review in 2013 *IEEE International Conference on Systems, Man, and Cybernetics.*

Abstract

Fault Analysis is the detection and diagnosis of malfunction in machine operation or process control. Early fault analysis techniques were reserved for high critical plants such as nuclear or chemical industries where abnormal event prevention is given utmost importance. The techniques developed were a result of decades of technical research and models based on extensive characterization of equipment behavior. This requires in-depth knowledge of the system and expert analysis to apply these methods for the application at hand. Since machine learning algorithms depend on past process data for creating a system model, a generic autonomous diagnostic system can be developed which can be used for application in common industrial setups.

In this thesis, we look into some of the techniques used for fault detection and diagnosis multi-class and one-class classifiers. First we study Feature Selection techniques and the classifier performance is analyzed against the number of selected features. The aim of feature selection is to reduce the impact of irrelevant variables and to reduce computation burden on the learning algorithm. We introduce the feature selection algorithms as a literature survey. Only few algorithms are implemented to obtain the results. Fault data from a Radio Frequency (RF) generator is used to perform fault detection and diagnosis. Comparison between continuous and discrete fault data is conducted for the Support Vector Machines (SVM) and Radial Basis Function Network (RBF) classifiers.

In the second part we look into one-class classification techniques and their application to fault detection. One-class techniques were primarily developed to identify one class of objects from all other possible objects. Since all fault occurrences in a system cannot be

simulated or recorded, one-class techniques help in identifying abnormal events. We introduce four one-class classifiers and analyze them using Receiver-Operating Characteristic (ROC) curve. We also develop a feature extraction method for the RF generator data which is used to obtain results for one-class classifiers and Radial Basis Function Network two class classification.

To apply these techniques for real-time verification, the RIT Fault Prediction software is built. LabView^{TM}environment is used to build a basic data management and fault detection using Radial Basis Function Network. This software is stand alone and acts as foundation for future implementations.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past years in the applications of machine intelligence or pattern recognition, the domain of features have expanded from ten's to hundreds of variables or features used in those applications. Several techniques are developed to address the problem of reducing irrelevant and redundant variables which are a burden on challenging tasks. Variable elimination (feature selection) helps in understanding data, reducing computation requirement, reducing the effect of curse of dimensionality and improving the predictor performance. In this paper we look at some of the methods found in literature which use a particular measurement to find a subset of variables which can be used to build a good predictor.

The focus of variable elimination is to select a subset of variables from the input which can efficiently describe the input data while reducing effects from noise or irrelevant variables and still provide good prediction results [2]. One of the applications would be in gene micro array analysis [2, 3, 4, 5, 6]. The standardized gene expression data can contain hundreds of variables of which many of them could be highly correlated with other variables. Hence by eliminating the dependent variables the amount data can be reduced which can lead to improvement in the classification. To remove an irrelevant feature, a feature selection criterion is required which can measure the relevance of each feature with the output. From a machine learning point if a system uses irrelevant variables, it will use this information for new data leading to poor generalization. Removing irrelevant variables

must not be compared with other dimension reduction methods such as Principal Component Analysis (PCA) [7] since good features can be independent of the rest of the data [8]. Feature elimination does not create new features since it uses the input features itself to reduce their number. Once a feature selection criterion is selected, a procedure must be developed which will find the subset of useful features. Directly evaluating all the subsets of features (2^N) for a given data becomes an NP-hard problem as the number of features grows. Hence a suboptimal procedure must be used which can remove redundant data with tractable computations.

A generic fault diagnostic framework is given in figure 1.1. From figure 1.1 we can observe the various components in a typical plant/machine. Feature selection helps in fault

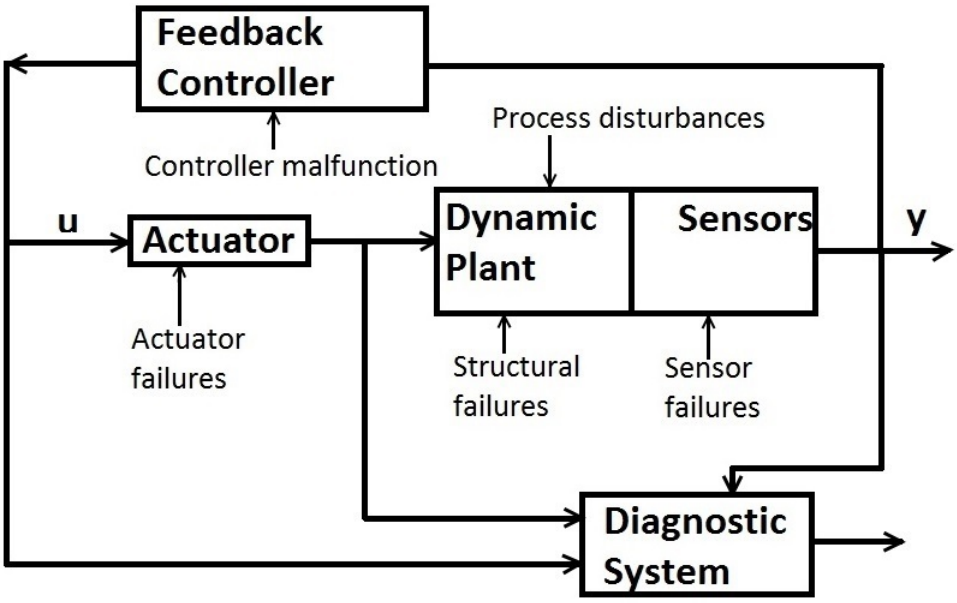


Figure 1.1: A generic fault diagnostic framework.

analysis to identify irrelevant process variables. This can be common in fairly complex operations since not all operations are dependent on each other. Independent abrupt faults can occur due to wear and tear of machine parts or unforeseen events. By identifying irrelevant variables, more insight into the cause of the fault can be gained and a more compact

model representation is obtained. Fault diagnosis is made simpler and fault prognosis becomes efficient by observing fewer relevant variables. For autonomous unsupervised fault detection systems, speed is important to prevent faults and abrupt fault progression. With fewer sensors to record, data collection and learning is made efficient. If suitable feature extraction is applied, the classifier performance can be increased due to maximization of information content.

One disadvantage of machine learning algorithms is their dependence on data. For multi-class classification a minimum number of observations is required for each class. Most classifiers rely on linear or non-linear separability of the classes in the feature space to distinguish between the classes. With partial data, the classifier will not be able to correctly learn or identify the pattern if the classes are similar which will lead to poor classifier performance. Insufficient data is a potential problem in fault detection using multi-class classifiers depending on the complexity of the process and the application. An expert with knowledge of the system can identify and classify the possible faults in the system for the application at hand. This method of manual classifying of faults will become unreliable as the number of observed variables increase. Abrupt faults can occur due to malfunction during the operation of the system. Due to wear and tear, the operating range of the observed variables can change over time. Hence the diagnostic system must be robust to identify these fault which have not been observed before and capability to adapt. This leads to Novel identification wherein the diagnostic system must be capable of identifying unknown faults.

One solution to novelty fault detection is one-class classification. In one-class classification, a target class is learned using objects from target class only using which all other possible objects are rejected. One-class techniques were mainly developed for cases where

data for a single class was abundantly available and insufficient or no data was available for other classes. A target data description model is obtained by learning from target objects wherein new objects are accepted into the model or rejected as outlier objects. In fault analysis, one-class model can be obtained by learning from normal operation data. By detecting deviations from normal operation, fault detection is achieved. Once a fault is detected, further diagnosis can be done to determine the cause of the fault. In this thesis we only study fault detection using one-class classifiers.

The rest of the thesis is organized as follows. In chapter 2 we discuss feature selection methods followed by the two classifiers in chapter 3. In chapter 4 we discuss the one-class classifiers. One-class techniques originated to account for the lack of data in certain applications. One-class techniques revolve around identify and separating one class from all other possible data. In chapter 5 we look into some fault detection techniques and the results are presented in chapter 6. In chapter 7 we provide an overview of the LabView^{TM}software implementation done for fault detection based on the algorithm presented in this thesis and the conclusions are given in chapter 8.

Chapter 2

Feature Selection

In [9] the variable elimination methods were broadly classified into filter and wrapper methods. Filter methods act as preprocessing to rank the features wherein the highly ranked features are selected and applied to a predictor. In wrapper methods the feature selection criterion is the performance of the predictor i.e. the predictor is wrapped to a search algorithm which will find a subset which gives the highest predictor performance. A search algorithm must be used to avoid large computations. Another method called embedded [2, 10, 11] methods are developed in literature. Embedded methods include variable selection as part of the training process without splitting the data into training and testing sets. We will focus on feature selection method using supervised learning algorithms and very brief introduction to feature selection methods using unsupervised learning will be presented. In the following sections we discuss each method and their approach.

2.1 Filter Methods

Filter methods use variable ranking techniques as the principle criteria for variable selection by ordering. Ranking methods are used due to their simplicity and good success is reported for practical applications. A suitable ranking criterion is used to score the variables and a threshold is used to remove variables below the threshold. Ranking methods are filter methods since they are applied before the classifier to filter out the less relevant

variables.

Here the issue of relevancy of a feature has to be raised i.e. how do we measure the relevancy of a feature to the data or the output. Several publications [2, 9, 10, 11, 12] have presented various definitions and measures for the relevance of a variable. One definition that can be mentioned which will be useful for the following discussion is that "A feature can be regarded as irrelevant if it is conditionally independent of the class labels." [8]. It essentially states that if a feature is to be relevant it can be independent of the input data but cannot be independent of the class labels i.e. the feature that has no influence on the class labels can be discarded. For the rest of the paper we use a standard notation to represent data and the variables. The input data $[x_i, y_{ik}]$ consists of N samples *i = 1 to N* with D variables *j = 1 to D*, x_i is the i^{th} sample and y_{ik} is the class labels *k = 1 to Y*.

2.1.1 Correlation Criteria

One of the simplest criteria is the Pearson correlation coefficient [2, 13] defined as:

$$R(i) = \frac{cov(x_i, Y)}{\sqrt{var(x_i) * var(Y)}} \tag{2.1}$$

where x_i is the i_{th} variable, Y is the output (class labels), $cov()$ is the covariance and $var()$ the variance. Correlation ranking can only detect linear dependencies between variable and target.

2.1.2 Mutual Information (MI)

Information theoretic ranking criteria [2, 9, 13, 14, 15, 6] use the measure of dependency between two variables. To describe MI we must start with Shannon's definition for entropy

given by:

$$H(Y) = -\sum_{y} p(y)log(p(y)) \tag{2.2}$$

Equation (2.2) represents the uncertainty (information content) in output Y. Suppose we observe a variable X then the conditional entropy is given by:

$$H(Y|X) = -\sum_{x}\sum_{y} p(x,y)log(p(y|x)) \tag{2.3}$$

Equation (2.3) implies that by observing a variable X, the uncertainty in the output Y is reduced. The decrease in uncertainty is given as:

$$I(Y,X) = H(Y) - H(Y|X) \tag{2.4}$$

This gives the MI between Y and X meaning that if X and Y are independent then MI will be zero and greater than zero if they are dependent. This implies that one variable can provide information about the other thus proving dependency. The definitions provided above are given for discrete variables and the same can be obtained for continuous variables by replacing the summations with integrations. The MI can also be defined as a distance measure given by:

$$K(f,g) = \int f(y)log(\frac{f(y)}{g(y)}) \tag{2.5}$$

The measure K in (2.5) is the Kullback-Leibler divergence [16, 17] between two densities which can also be used as a measure of MI. From the above given equations we need to know the probability density function (pdf) of the variables to calculate MI. Since the data we obtain is of finite samples, the pdf cannot be calculated accurately. Several methods have been developed for estimating the MI in [13, 16, 17]. Once a particular method is chosen for calculating MI then one of the simplest methods for feature selection is to find the MI between each feature and the output class labels and rank them based on this

value. A threshold is set to select $d < D$ features. This is a simple method and the results can be poor [14] since inter-feature MI is not taken into account. But MI is an important concept and is used in embedded methods which will be presented in a later section. In [18] the author develop a feature ranking criteria based on Conditional Mutual Information for binary data. A score table is updated as features are selected to the subset using the conditional mutual information criteria which is to be maximized. The score at each iteration is calculated using (2.6) given by:

$$s[n] = min_{l<k} \hat{I}(Y; X_n | X_{\nu(l)}) \tag{2.6}$$

where $s[n]$ is the score which is updated at each iteration, X_n is the current evaluated feature, $X_{\nu(l)}$ is the set of already selected features.

Other statistical tests found in literature can be used for feature ranking. In [14] twelve feature selection metrics are considered for the text classification problem [2, 14, 19]. All the features are ranked using each metric and a threshold is set which would select 100 words which are then applied to the predictor. Filter approaches applied to various applications can be found in [20, 21, 22, 19, 6]. Earlier comparisons for text classification using ranking methods can be found in [23]. In [24] the authors develop a ranking criteria based on class densities for binary data. A two stage algorithm utilizing a less expensive filter method to rank the features and an expensive wrapper method to further eliminate irrelevant variables is used. This two stage approach is also found in [25]. The RELIEF algorithm [26, 27] is another filter based approach wherein a feature relevance criterion is used to rank the features. Using a threshold a subset of features is selected. The drawback of the RELIEF algorithm is in selecting a threshold. Authors in [27] compare the RELIEF and other wrapper methods for different datasets. In [20] discarded variables are used to perform multitask learning (MTL).

The advantages of feature ranking are that it is computationally light and avoids over fitting and is proven to work well for certain datasets [2, 28, 6]. Filter methods do not rely on learning algorithms which are biased which is equivalent to changing data to fit the learning algorithm. One of the drawbacks of ranking methods is that the selected subset might not be optimal in that a redundant subset might be obtained. Some ranking methods do not discriminate the variables in terms of the correlation to other variables. The variables in the subset can be highly correlated in that a smaller subset would suffice [12, 28]. This issue of redundant vs. relevant variables is addressed in [2] with good examples. In [29] a random variable called probe is used to rank the features using Gram-Schmidt orthogonalization. In feature ranking important features that are less informative on their own but are informative when combined with others could be discarded [2, 30]. Finding a suitable learning algorithm can also become hard since the underlying learning algorithm is ignored [12]. Also there is no ideal method for choosing the dimension of the feature space.

2.2 Wrapper Methods

As mentioned above the wrapper methods use the predictor as a black box and the predictor performance as the objective function to evaluate the variable subset. Suboptimal subsets are found by employing search algorithms. A number of search algorithms can be used to find a subset of variables which is decided based on the objective function. The search algorithm evaluates different subsets and obtains the subset which maximizes the objective function. The Branch and Bound method [9, 31] used tree structure to evaluate different subsets for the given feature selection number. But the search would grow exponentially [9] for higher number of features. Exhaustive search methods are computationally intensive and might become a NP hard problem for larger datasets. Therefore simplified algorithms such as sequential search or evolutionary algorithms (GA or PSO)

which yield local optimum results are employed which can produce good results and are computationally feasible.

2.2.1 Sequential Selection Methods

The Sequential Feature Selection (SFS) algorithm [32, 33] starts with a null set and adds one feature for the first step which gives the highest value for the objective function. From the second step onwards the remaining features are added one at a time to the included set and the new subset is evaluated. The process is repeated until the required number of features is added. This is a nave SFS algorithm since the dependency between the features is not accounted for. A Sequential Backward Selection (SBS) algorithm can also be constructed which is similar to SFS but the algorithm starts from the complete set of variables and removes one feature at a time whose removal gives the lowest decrease in predictor performance.

The Sequential Floating Forward Selection (SFFS) [32, 33] algorithm is more flexible than the nave SFS because it introduces an additional backtracking step. The basic flowchart is given in figure 2.1 where k is the current subset size and d is the required dimension. The first step of the algorithm is the same as the SFS algorithm which adds one feature at a time based on the objective function. The SFFS algorithm adds another step which excludes one feature at a time from the subset obtained in the first step and evaluates the new subsets. If excluding a feature increases the value of the objective function then that feature is removed and goes back to first step with the new reduced subset or else the algorithm is repeated from the top. The process is repeated until the required number of features is added or required performance is reached. These two methods suffered from producing nested subsets since the forward inclusion was always unconditional which

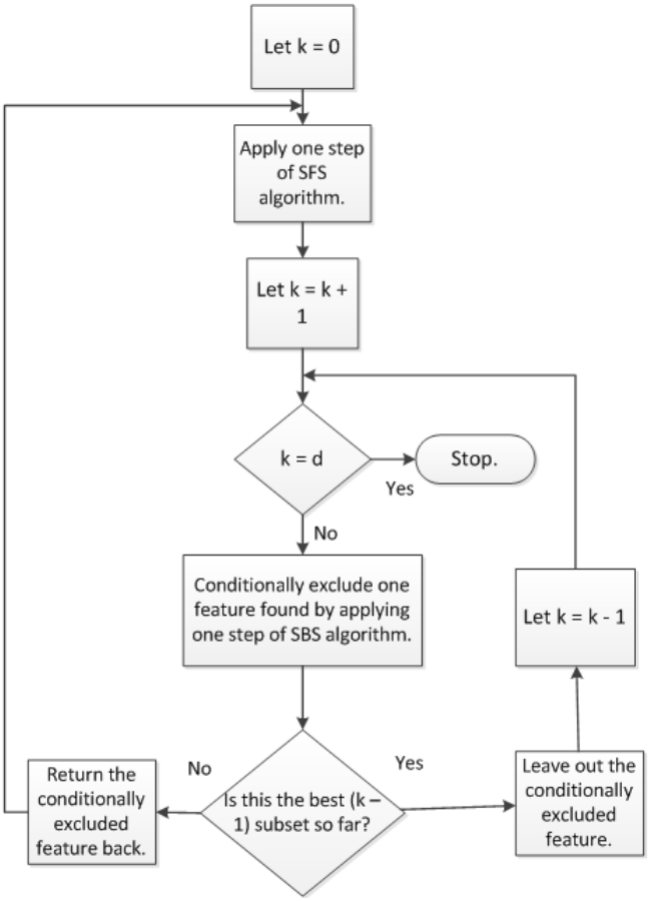


Figure 2.1: SFFS flow chart.

means that two highly correlated variables might be included if it gave the highest performance in the SFS evaluation. To avoid the nesting effect adaptive version of the SFFS was developed in [34, 35]. The ASFFS algorithm used a parameter r which would specify the number of features to be added in the inclusion phase which was calculated adaptively. The parameter o would be used in the exclusion phase to remove maximum number of features if it increased the performance. The ASFFS attempted to obtain a less redundant subset than the SFFS algorithm. It can be noted that a statistical distance measure can also be used as the objective function for the search algorithms as done in [10, 11, 32, 34]. Theoretically the ASFFS should produce a better subset than SFFS but this is dependent on the objective function and the distribution of the data. The Plus-L-Minus-r search method

[34, 36, 37] also tried to avoid nesting. In the Plus-L-Minus-r search, in each cycle L variables were added and r variables were removed until the desired subset was achieved. The parameters L and r has to be chosen arbitrarily. In [36] the authors try to improve the SFFS algorithm by adding an extra step after the backtracking step in the normal SFFS in which a weak feature is replaced with a new better feature to form the current subset.

2.2.2 Heuristic Search Algorithms

Genetic Algorithm (GA) [38] can be used to find the subset of features [35, 39, 40, 41, 42] wherein the chromosome bits represent if the feature is included or not. The global maximum for the objective function can be found which gives the best subset. Here again the objective function is the predictor performance.

The GA parameters and operators can be modified within the general idea of an evolutionary algorithm to suit the data or the application to obtain the best performance or the best search result. A modified version of the GA called the CHCGA [43, 44] can be used for feature selection [35]. The CHCGA is a nontraditional GA which differs from GA in the following ways:

- The best N individuals are chosen from the pool of parents and offspring i.e. better offspring replaces lesser fit parents.
- A highly disruptive HUX crossover operator is used which crosses over exactly half of the non-matching alleles, wherein the bits to be crossed over are selected at random.
- During reproduction step, each member of the parent population is randomly selected without replacement and paired for mating. Not all the pairs are crossed over but before mating the Hamming distance between the parents are calculated and if half this distance does not exceed a threshold d, they are not mated. The threshold is usually initialized to L/4 where L is the chromosome length. If no offspring is obtained in

the generation, the threshold is decremented by one. Due to these mating criteria of mating only diverse parents, the population converges as the threshold decreases.

- If there are no offspring generated and the threshold drops to zero, a cataclysmic mutation is introduced to create a new population. The best individual in the current parent population is taken as the template to create the new population. The rest N-1 individuals are obtained by randomly flipping a percentage (35%-40%) of bits of the template. The regular mutation after crossover step is skipped each time and the above mentioned mutation is carried if required.

The CHCGA converges on the solution faster and provides a more effective search by maintaining the diversity and avoiding stagnation of the population. In [45] multi-objective GA is used for hand written digit recognition. In [46, 47] several wrapper methods some of which mentioned above are compared with different datasets. They derive various fitness functions with weighting/penalty imposing characteristics. A binary PSO [48, 49, 5] algorithm can also be used for wrapper implementation. In [50] comparison between GA and PSO using SVM for gene selection can be found.

2.3 Embedded Methods

Embedded methods [2, 10, 11] want to reduce the computation time taken up for reclassifying different subsets which is done in wrapper methods. Embedded methods go through different approaches to incorporate the feature selection as part of the training process. There is no defined approach for embedded methods. In section 2.1.2 we mentioned that MI is an important concept but the ranking using MI yielded poor results since the MI between the feature and the class output only was considered. In [13] a greedy search algorithm is used to evaluate the subsets. The objective function is designed such that choosing

a feature will maximize the MI between the feature and the class output while the MI between the selected feature and the subset of so far selected features is a minimum. This is formulated as:

$$I(Y, f) - \beta \sum_{s \in S} I(f; s) \tag{2.7}$$

where Y is the output, f is the selected feature, s is the feature in the subset S and controls the importance of the MI between the current feature f and the features in the subset S. The output subset is applied to a Neural Network classifier. Equation (2.7) will select better subset since the inter-feature MI is used in the calculation to select the non-redundant features. An improvement of this method is presented in [15] which estimates the MI using Parzen window method.

The mRMR (max-relevancy, min-redundancy) [25] is another method based on MI. It uses similar criteria as in (2.7) given as:

$$I(x_j; C) - \frac{1}{m-1} \sum_{x_l \in S_{m-1}} I(x_j; x_l) \tag{2.8}$$

where x_i is the the m^{th} feature subset S and the set S_{m-1} is the so far selected subset with m-1 features. Instead of a greedy algorithm a two stage approach is implemented. First the criterion (2.8) is used to select a number k which is the optimal number of features which gives the lowest cross-validation classification error. In the second stage wrapper methods are used to evaluate different subsets of size k or direct evaluations are done on different subsets to find the subset which consistently yields the smallest classification error. The application of mRMR can be found in [51, 4] wherein the simplest incremental search method is used with four different classifiers for gene classification.

Another method used in literature is to use the weights of a classifier [2, 3, 51] to rank

the feature for their removal. Let w_j be defined as:

$$w_j = \frac{\mu_j(+) - \mu_j(-)}{\sigma_j(+) + \sigma_j(-)} \tag{2.9}$$

where $\mu_j(+)$ and $\mu_j(-)$ are the mean of the samples in class (+) and (-) and σ_j is the variance of the respective classes and *j = 1 to D*. Equation (2.9) [3] can be used as a ranking criteria to sort the features. The rank vector w can be used to classify since features rank proportionally contributes to the correlation. A voting scheme given as:

$$D(x) = w(x - \mu) \tag{2.10}$$

where w is the rank of the features or weight, $D(x)$ is the decision and μ is the mean of the data. Hence the weights (rank) of the features can be used as classifier weights. By conducting sensitivity analysis [2, 3] of the weights, feature selection can be done i.e. the change in the weight wj can be viewed as removing a feature j. In [3] it is suggested to use the change in the objective function, a linear discriminant function J which is a function of w_j. This concept of using the weights as the ranking and the search is done using the change in the objective function is applied to the SVM classifier [3, 1, 51] to perform Recursive Feature Elimination or known as the SVM-RFE method. The objective function of the SVM is presented later in Section 3.1. The SVM-RFE method in [3] is proposed for binary class classification, its multi class classification technique can be found in [52]. In [53] the authors use the concept of RFE to derive a modified algorithm for selecting features in hyper spectral image data. In the SVM-RFE method, the l_2 norm is used in the SVM minimization problem. It is shown in literature that other norms can be used which help in feature selection. In [54] the authors provide a comprehensive review of the different SVM based feature selection methods. The paper also presents non-linear classification and feature selection approaches using SVM.

Similar to optimizing the SVM equation and weighting features, the same can be done using Neural Networks. Multilayer perceptron networks are trained and feature weights are calculated using a saliency measure calculated from the trained network [55, 56]. In [55] a penalty is applied for features with small magnitude at the node and the nodes connecting to these input features are excluded. This type of node removal is also called Network Pruning [55, 56] commonly used to obtain the optimum network architecture for Neural Networks. In [57] authors derive a cost function for random variable elimination. In chemometric applications [58, 59, 60] regression models are developed to reduce the number of variables which helps in analyzing the chemical properties. In [20] the discarded variables are used to improve the prediction performance using multi task learning (MTL) approach. Lazy Feature Selection (LFS) approach is developed in [61] where the authors take advantage of the sparseness in the feature space as a feature selection method for text categorization problem. The LFS ranks the test samples against all training samples and k-NN [19, 7] type algorithm is used to determine the class.

2.4 Other Feature Selection Techniques

In this paper we present feature selection methods based on the supervised learning methods i.e. the output labels or the output pdf is known or can be calculated. Unsupervised and Semi-supervised learning methods are beyond the scope of this paper and will not be discussed but we refer to other papers which present more detail about unsupervised feature selection methods. There may be certain examples in which only the data and no other information is available. Using feature selection based on unsupervised methods can provide better description and reliability of the data than just unsupervised learning. Several papers attempt to solve the unsupervised learning feature selection can be found in [62, 63, 8, 64, 65].

Semi-supervised learning is another class wherein both labelled and unlabelled data are used for learning [30, 66, 67]. It uses both labelled data (less number of samples) and unlabelled data (abundantly available) to modify a hypothesis obtained from labelled data alone. In [67] the authors use a clustering indicator construction to score a set of features. In [30] the authors use the maximum margin principle (SVM) using manifold regularization problem optimization similar to SVM-RFE [3].

Ensemble feature selection [68, 69] is a relatively new technique used to obtain a stable feature selection. A single feature selection algorithm is run on different subsets of data samples obtained from bootstrapping method. The results are aggregated to obtain a final feature set. Authors in [68, 69] use filter methods to rank the genes/features and use different aggregation methods such as ensemble-mean, linear aggregation, weighted aggregation methods to obtain the final feature set.

2.5 Stability of Feature Selection Algorithms

For a particular application various feature selection algorithms can be applied and the best one can be selected which meets the required criteria. An overlooked problem is the stability of the feature selection algorithms. Stability of an algorithm can be viewed as the consistency of an algorithm to produce a consistent feature subset when new training samples are added or when some training samples are removed [70, 71, 72, 73, 68]. If the algorithm produces a different subset for any perturbations in the train data then that algorithm becomes unreliable for feature selection. Examples of instabilities are demonstrated in [70, 73] which can be verified by ourselves by changing the training set and running the algorithm again. In [70] wrapper techniques are used to study their instability and stability measures are introduced along with possible solutions to alleviate the problem. Various measures are established in [70, 71, 72, 68] to evaluate different subsets obtained for a

certain number of runs. Using these measures a more robust subset is found for different datasets. In [73] multi-criterion fusion algorithm is developed which uses multiple feature selection algorithms to rank/score the features which are combined to obtain a robust subset based on combining multiple classifiers to improve the accuracy [73]. In [74] the author also suggests dividing the input features (based on their feature extraction procedures) to obtain different classifiers and combine the predictions to obtain a final decision.

Chapter 3

Classifiers

In this section we want to provide a brief introduction to two classifiers in literature which can be used for feature selection tasks. We will present the SVM and RBF classifiers due to their wide range of applications found in literature.

3.1 Support Vector Machine (SVM)

SVM [1, 3, 25, 19] is a marginal classifier which maximizes the margin between the data samples in the two classes. An optimal hyperplane boundary is drawn which will separate the data. In SVM kernels are used to map the input data to a higher dimensional space where a decision boundary can be constructed. The decision function is given as:

$$D(x) = w\phi(x) + b \tag{3.1}$$

where w and b are the SVM parameters and $\phi(x)$ is a kernel function that maps the input data into the new M dimension. Equation (3.1) defines the hyperplane and

$$\frac{D(x)}{\|w\|} \tag{3.2}$$

is the distance between hyperplane and pattern x. In figure 3.1 (taken from [1]) a graphical illustration of the boundary and the samples is provided. The objective of the training

algorithm is to find w such that M^* is maximized. For the linear decision function in (3.1), the parameters are given as:

$$w = \sum_k a_k y_k x_k \tag{3.3}$$

$$b = (y_k - w * x_k) \tag{3.4}$$

The vector w is a linear combination of training patterns wherein α_k is non-zero for the marginal patterns which forms the support vectors. The value b is an average over the support vectors. The objective function of the training algorithm is:

$$J = (\frac{1}{2})\|w\|^2 \tag{3.5}$$

For the SVM-RFE method described in section 2.2, (3.3) gives the feature ranking criteria.

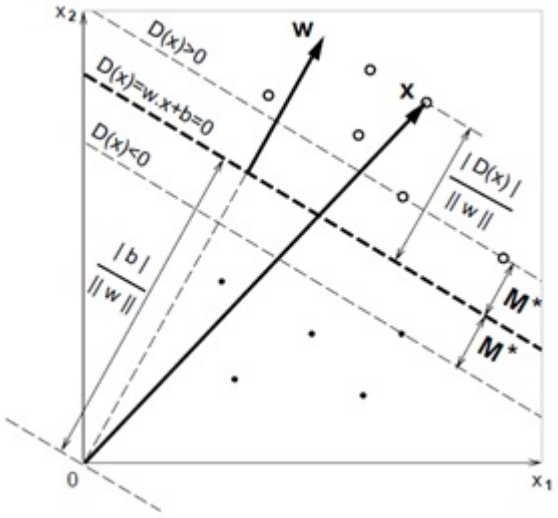


Figure 3.1: Maximum margin for linear decision function D(x) [1]

To solve the objective in (3.5) quadratic programming methods have to be used. These methods are not easy to implement when compared to other classifier algorithms. Libraries like LIBSVM [75] can be used to implement wrapper techniques. Since SVM is a binary

classifier, in this paper we use the one-vs-all classification technique for multi-class problems. In one-vs-all approach different binary classifiers are trained wherein each one is trained to distinguish samples of a single class from all the remaining samples. It is a simple method which is proven to be robust and accurate when compared to other approaches [53, 76].

3.2 Radial Basis Function Network

A Radial Basis Function Network [77, 7] (RBFN or RBF) is a type of feed-forward Neural Network. The network structure consists of three layers: input layer, hidden layer and output layer as shown in figure 3.2. The input layer consists of a single D dimensional input. The hidden layer is composed of radially symmetric Gaussian kernels given by:

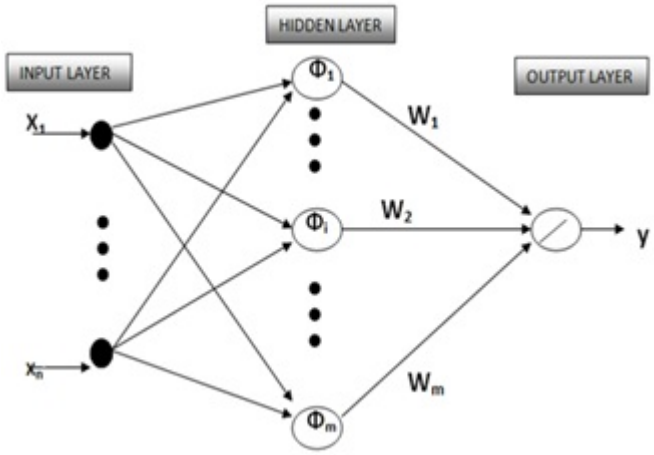


Figure 3.2: Radial Basis Function Network structure.

$$\phi_j = e^{-\frac{\|x - x_j\|^2}{\sigma^2}} \tag{3.6}$$

where *j = 1 to M*, *M* being the number of kernels, x_j the j^{th} kernel centroid and ϕ_j is calculated for every input data vector. The output layer node is connected to each kernel in

the hidden layer with a weight W_j. The output Y can be calculated using the equation:

$$Y = \sum_{j=1}^{M} W_j \phi_j \tag{3.7}$$

In the training phase the weight vector is calculated for all the samples in the training set. The weight vector can be calculated using:

$$W = \phi^{-1}.Y \tag{3.8}$$

where the ϕ matrix element ϕ_{ij} gives the value of ϕ_j for the i^{th} sample. For the testing set the output values can be calculated for each data using (3.8) since the weight vector is known in the training phase. To find the centroids of the kernel function clustering algorithms [77, 78, 79, 80, 81] are employed to enhance the generalizing capability of the classifier. In [78] the RBF network is used to EEG data classification wherein three clustering methods for the RBF are explored.

3.3 Validation Methods

It is also important to choose a method of validating the method or the classifier chosen. This method is known as cross-validation in which the input data is split into training and testing sets and the test set (unseen by the method or classifier) is validated against the training set to check if the classifier can reproduce the known output. Several types of cross-validation are used in literature. One of the simplest methods is the 2-fold cross-validation wherein the data is randomly split into training and test sets. An extension of the 2-fold cross-validation is the K-fold cross-validation in which the data is randomly split into K subsets. For training K-1 subsets are chosen and the remaining subset is used for testing. This process is repeated until all the subsets are used for testing. Another version of the

K-fold cross-validation is the Leave-one out cross-validation (LOOCV) wherein K is equal to the number of samples i.e. each sample is used for testing and the rest of the samples are used for training. This process is done until all the samples are tested. For model selection (feature selection or classifier) the training set may be further split into training and validation sets. The prediction of the validation set is used to reinforce the model selection. In [33] the authors address the problem of over fitting found when comparisons of the feature selections are made. The authors argue the cross-validation methods can contribute to over estimating the model performance. The author tests the SFS and SFFS method on various datasets with k-NN [19, 7] classifier as the wrapper. They use the 2-fold and LOOCV cross-validation methods to compare the results of the two feature selection algorithms.

Chapter 4

One-class Classifiers

4.1 Introduction

One class classification is a unary classification technique that tries to distinguish a class objects from all other possible objects, by learning from objects of that class only. This method is more difficult than traditional classification methods in which we distinguish two or more classes from the available samples. An examples of one class classification is the classification of the operational status of an RF Generator as ”normal”. In this scenario we can generate faulty data by seeding of faults, but doing so would highly restrict the fault operating range since we would be characterizing the data into predefined categories. Hence ”normal” operating data is collected and trained as the target class using which we can detect any deviations from the ”normal” operating range and identify machine faults.

One class classification methods exploit different characteristics of data. Some of the important characteristics are probability distribution and object clusters. Next we will look at the characteristics of some of the one class classifiers.

4.2 Characteristics of One class classifiers

In all one class methods two distinct measurement quantities can be defined. The first quantity is a measure of probability $p(z)$ of an object z or distance $d(z)$ to the target class. The second quantity is a threshold θ on the measurement [82] . From the data description new objects are accepted if the distance is smaller or if the probability is greater than the threshold given by:

$$f(z) = I(d(z) < \theta_d) \tag{4.1}$$

$$f(z) = I(p(z) > \theta_p) \tag{4.2}$$

where $I(.)$ is the indicator function. One class classification methods differ in the optimization of $p(z)$ or $d(z)$ and defining the appropriate threshold. First a probability or distance model is optimized and the threshold is obtained on the target class.

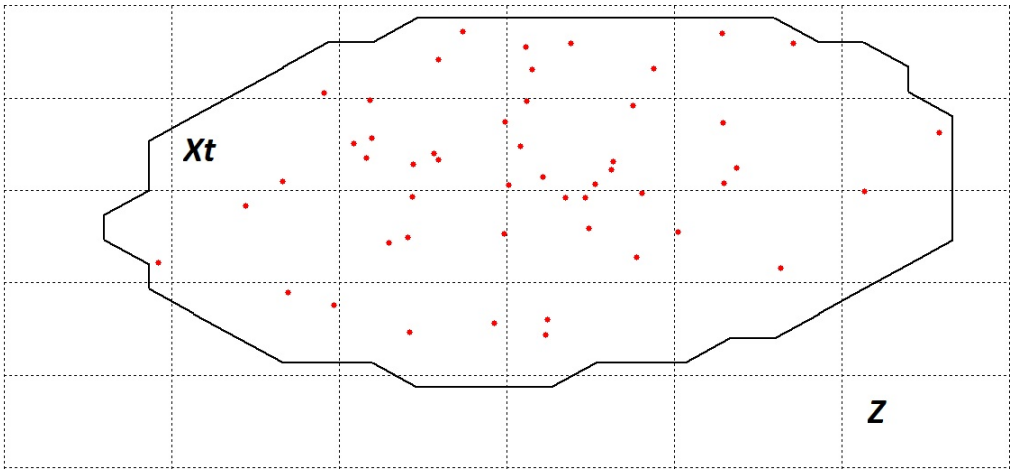


Figure 4.1: Regions in one class classification

The most important feature of one class classifiers is the trade-off between the fraction of the target class that is accepted, f_{T+}, and the fraction of outliers that is rejected, f_{O-}. The f_{T+} is measured using an independent test set drawn from the same target distribution. The f_{O-} is measured on data obtained from an assumed outlier density. For experiments of

artificial data, we assume that the outliers are drawn from a bounded uniform distribution. An example is given in figure 4.1. The region Xt is the target class region and Z is the region from which outlier objects are drawn. The boundary depicted is the target data description.

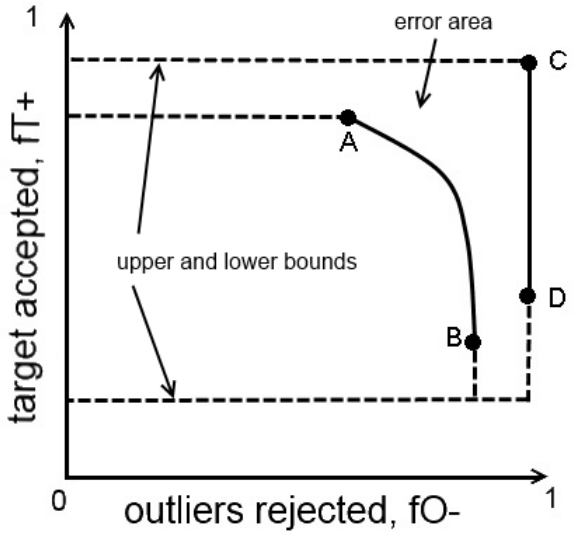
4.2.1 Error Definition

To make comparisons between different one class classifiers which use either definition, f_{T+} is fixed and f_{O-} is measured. Optimizing (or choosing) different thresholds gives different trade-off's between f_{T+} and f_{O-}. A method which results the lowest outlier rejection rate f_{O-} is preferred. Choosing different thresholds will result in different definition of $p(z)$ or $d(z)$. Once the model is chosen, the threshold is chosen. As done in [82], in this thesis we set the threshold such that a predefined fraction of the target class is accepted. The threshold $\theta_{f_{T+}}$ [82] is defined as:

$$\theta_{f_{T+}} : \frac{1}{N}\sum_{i} I(p(x_i)) \geq \theta_{f_{T+}}) = f_{T+}, \text{ where } x_i \in Xt, \forall_i \tag{4.3}$$

An equivalent threshold for distance based model ($d(z)$) can also be defined. To avoid over fitting, the threshold can be calculated using an independent set. But in some cases the data available and the training set itself is used to determine $\theta_{f_{T+}}$.

Using (4.3) we measure f_{O-} for varying f_{T+} by calculating $\theta_{f_{T+}}$ on the training set. A Receiver-Operating Characteristic (ROC) curve [82], [90], [88] can be obtained when f_{O-} is obtained for all values of f_{T+}. An example ROC curve is given in figure 4.2.1. The line C-D is the line obtained for an ideal one-class classifier which can reject 100% outliers f_{O-} for a pre-set target acceptance rate f_{T+}. In practise some outliers will always be accepted and the line A-B will be obtained realistically depending on the distribution of the data and the model assumption.



One-class classifiers are broadly classified into Density, Boundary and Reconstruction methods. In the next sections we provide details of at least one algorithm for each type of one-class classifier.

4.3 Density Methods

In density methods, target density is estimated to obtain a one-class classifier. Standard distributions such as Gaussian, Parzen or a Poisson distribution can be assumed on the target data. The probability threshold can be obtained empirically from the training data using (4.3). As with assuming probability distribution, sufficient training data should be available to fit the data to avoid the curse of dimensionality. In this thesis we study the Gaussian model which is presented next.

4.3.1 Mixture of Gaussians

A Mixture of Gaussians (MoG) is a parametric probability density function represented as a weighted sum of Gaussian component densities. MoG is simple and powerful due to its

ability to form smooth approximations to arbitrarily shaped densities. A Gaussian mixture model is given by the equation,

$$p(x) = \frac{1}{N} \sum_{i} \alpha_i g(x; \mu_i, \Sigma_i) \tag{4.4}$$

where *i = 1 to N*, α_i is the mixing weights, *x* is the *D*-dimensional data vector, μ_i and Σ_i are the mean and covariance matrix of the i^{th} Gaussian component. The Gaussian function is given by the equation,

$$g(x; \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} exp\left\{-\frac{1}{2}(x-\mu_i)'\Sigma_i(x-\mu_i)\right\} \tag{4.5}$$

For a given number of Gaussian components, the standard Expectation Maximization (EM) [83] algorithm can be used to calculate the other variables of the Gaussian mixture. To avoid assuming the number of components in the data, we use the Greedy EM algorithm [84], [83] which can automatically determine the number components depending on the probability distribution. When using EM algorithm singularities in calculation of covariance matrix must be avoided or compensated.

4.4 Boundary Methods

If the available data is less then density estimation is prone to over fitting. In boundary methods a closed boundary around the target data is optimized. Boundary methods rely on distances between objects to obtain a distance based description which cannot be interpreted as a probability measure. Using (4.1) the distance resemblance measure $d()$ and distance threshold θ_d is calculated. By using a smaller threshold, the boundary can be tightened but does not guarantee that the high density areas are captured. In this thesis we will study the k-centers and Support Vector Data Description (SVDD) boundary methods which are presented next.

4.4.1 K-centers

The k-center method is the problem of placing k balls with equal radii to minimize the maximum distances of all minimum distances between training objects and the centers. The ball centers μ_k are placed on the training objects itself. The error minimized is given by,

$$\varepsilon_{k-center} = \max_i(\min_k \|x_i - \mu_k\|^2) \tag{4.6}$$

The radius is determined by the maximum distance to the objects that the corresponding ball should capture. By placing the centers on the target objects, the data description is sensitive to outliers. The distance between a test object z and the target set can be calculated using,

$$d_{k-center}(z) = \min_k \|z - \mu_k\|^2 \tag{4.7}$$

To avoid suboptimal solutions for training, several random initializations are used to place the centers and the best solution for (4.6) is used. The user must specify the number of tries and the number of centers to be used.

4.4.2 Support Vector Data Description

In support vector machine, an optimal hyperplane margin is computed which separates the data from the origin [82], [85]. The support vector data description method involves computing an optimal hypersphere which contains all target objects. To minimize the error on accepting outliers, the volume of this hypersphere is minimized. Flexible description can be obtained by using kernel functions which map the input dimension to a higher dimensional space with no additional computational load on minimizing the volume of the sphere.

First we will present the linear SVDD derivation which can be extended using 'kernel-trick' to obtain a better description.

4.4.2.1 Linear SVDD

The hypersphere is characterized by a center $\mathbf{a}$ and radius R which contains all the training objects. The minimum hypersphere is obtained by solving the following constrained optimization problem [82]:

$$\|x_i - \mathbf{a}\|^2 \leq R^2, \forall_i \tag{4.8}$$

To allow for a more robust definition and outliers in training set, a slack variable ξ and parameter C is introduced. Using this the objects to the center $\mathbf{a}$ are not strictly lesser than R^2 while larger distances are penalized. The constrained problem (4.8) is rewritten as:

$$\|x_i - \mathbf{a}\|^2 \leq R^2 + C\sum_i \xi_i, \xi_i \geq 0, \forall_i \tag{4.9}$$

The constraints (4.9) can be found by optimizing the following Lagrangian:

$$L(R, \mathbf{a}, \xi, \alpha) = R^2 + C\sum_i \xi_i \\ - \sum_i \alpha_i R^2 + \xi_i - (x_i.x_i - 2\mathbf{a}.x_i + \mathbf{a}.\mathbf{a}) \tag{4.10}$$

with Lagrange multipliers $\alpha_i \geq 0$, and $x_i.x_j$ is the inner product. The radius R is to be minimized with respect to α.

By setting the partial derivatives of (4.10) to 0, the following constraints are obtained:

$$\sum_i \alpha = 1 \tag{4.11}$$

$$\mathbf{a} = \sum_i \alpha_i x_i \tag{4.12}$$

$$C - \alpha_i = 0, \forall_i \tag{4.13}$$

The final Lagrangian equation L is given by:

$$L = \sum_i \alpha_i (x_i.x_i) - \sum_{i,j} \alpha_i \alpha_j (x_i.x_j) \tag{4.14}$$

with the reformulated constraint:

$$0 \leq \alpha_i \leq C \tag{4.15}$$

The optimization of the above Lagrangian is a well known quadratic programming problem for which standard algorithms exist. The Lagrange multipliers α are found which can be interpreted as follows. The Lagrange multiplier α will be equal to zero if an object x_i satisfies the inequality of (4.9). If an object satisfies the equality of (4.9) then the object is at or outside the boundary with $\alpha_i > 0$. The constraint (4.15) limits the values of α_i and an object with $\alpha_i > 0$ is on or outside the boundary. From (4.12) the center of the sphere can be calculated by considering objects with $\alpha_i > 0$. The Lagrange minimization will lead to a small number of objects with positive α_i using which the data description is obtained. These objects are called the support vectors.

To test a new object $\mathbf{z}$, the distance of the object from the center $\mathbf{a}$ is calculated using:

$$\|\mathbf{z} - \mathbf{a}\| = (\mathbf{z}.\mathbf{z}) - 2\sum_i \alpha_i (\mathbf{z}.x_i) + \sum_{i,j} \alpha_i \alpha_j (x_i.x_j) \tag{4.16}$$

The test object $\mathbf{z}$ is accepted if (4.16) is less than or equal to the radius. The radius can be calculated from the definition as the distance from center of the sphere to one of the support vectors given by:

$$R^2 = (x_k.x_k) - 2\sum_i \alpha_i (x_i.x_k) + \sum_{i,j} \alpha_i \alpha_j (x_i.x_j) \tag{4.17}$$

where x_k is the support vector for which $0 < \alpha_k < C$.

The one-class support vector data description can be completely defined as:

$$f_{SVDD}(\mathbf{z};\alpha,R) = I(\|\mathbf{z}-\mathbf{a}\|^2 \leq R^2) \tag{4.18}$$

$$f_{SVDD}(\mathbf{z};\alpha,R) = I\left((\mathbf{z}.\mathbf{z}) - 2\sum_i \alpha_i(\mathbf{z}.x_i) + \sum_{i,j}\alpha_i\alpha_j(x_i.x_j) \leq R^2\right) \tag{4.19}$$

The indicator function *I* is 1 if the object is accepted otherwise it is 0.

An example SVDD boundary was already shown in figure 4.1. In the next section we will discuss the kernel based SVDD wherein kernel functions are used to replace the inner products $(x.y)$.

4.4.2.2 Kernel-based SVDD

Additional flexibility can be added by mapping the input space to a higher dimension using $\Phi(x)$. Solving the SVDD in the feature space allows for a tighter and flexible minimum enclosing hypersphere. By applying the mapping the Lagrangian obtained is:

$$L = \sum_i \alpha_i\Phi(x_i).\Phi(x_i) - \sum_{i,j}\alpha_i\alpha_j\Phi(x_i).\Phi(x_j) \tag{4.20}$$

Using the 'kernel-trick', the inner products of the mappings $\Phi(x)$ can be defined as:

$$K(x_i,x_j) = \Phi(x_i).\Phi(x_j) \tag{4.21}$$

The kernel-based SVDD equation now can be defined as:

$$f_{SVDD}(\mathbf{z};\alpha,R) = I\left(K(\mathbf{z},\mathbf{z}) - 2\sum_i \alpha_i K(\mathbf{z},x_i) + \sum_{i,j}\alpha_i\alpha_j K(x_i,x_j) \leq R^2\right) \tag{4.22}$$

The 'kernel-trick' is a well known technique and used in SVM's [1] so that the input space becomes linearly separable in the feature space using which better descriptions can be obtained. The only overhead in the using the kernel method is the calculation of the kernel functions $K(x_i, x_j)$ and the optimization will not suffer any overhead calculations and remains the same as when the original input space was used.

4.5 Reconstruction Methods

In reconstruction methods, prior knowledge of the data is used to make assumptions about the generating process and model is chosen to fit the data. The information content is retained by using compact object representation and reducing sensitivity to noise. Clustering characteristics of the data is used to identify prototypes or subspaces using which a reconstruction error is minimized.

Methods such as k-means, Principal Component Analysis, learning vector quantization [82] are some of the examples which involve optimizing prototypes and their reconstruction error. Using target object representation, the reconstruction error of the outlier objects should be high. We will study only Fuzzy C-means one-class method in this thesis which is presented in the next section.

4.5.1 Fuzzy C-means

In Fuzzy C-means (FCM) algorithm [86], the data is characterized by few prototype objects and Euclidean distance measure is used to represent the data. The data is partitioned into the clusters according to their membership values whose values change between zero and one and represent the degree of how close the data to each cluster center. First the FCM

algorithm is briefly explained followed by the adaptation to the one-class technique will be explained.

The Fuzzy C-means (FCM) model is represented by the constrained optimization problem as,

$$\min J_m(U, V; X) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m \|x_k - v_i\|^2 \tag{4.23}$$

where $1 \leq m < \infty$, u_{ik} is the degree of membership of x_k in cluster i, x_k is the k^{th} data and v_i is the i^{th} cluster center. From [86], the following u_{ik} and v_i formulas will minimize the objective function.

$$u_{ik} = \left(\sum_{j=1}^{c} (\frac{D_{ik}}{D_{jk}})^{\frac{2}{m-1}} \right)^{-1}; \qquad 1 \leq i \leq c, 1 \leq k \leq n \tag{4.24}$$

$$v_i = \frac{\sum_{k=1}^{n} u_{ik^m} x_k}{\sum_{k=1}^{n} u_{ik}^m}; \qquad \sum u_{ik} = 1 \tag{4.25}$$

where $D_{ik} = \|x_k - v_i\|$ represents the distance between the k^{th} data and the i^{th} cluster center. The algorithm updates the cluster centers according to (4.25) and calculates the membership matrix based on the new cluster centers. It terminates when $\|v_t - v_{t-1}\| \leq \epsilon$ that is the cluster centroids do not change any more. More crisp cluster are obtained when m gets closer to one. As m gets larger, the overlapping of the clusters is also increased.

The k-center method focusses on worst case objects to accept all data whereas in FCM optimization, the centers are averaged depending on the membership values of each data object which makes it less sensitive to outliers. The k-center method also places the prototypes on the target object whereas in FCM algorithm, an averaged object is calculated.

The distance of an object $\mathbf{z}$ to the target set is defined as the distance to the nearest prototype [82] given as:

$$d_c(\mathbf{z}) = \min_c \|\mathbf{z} - v_c\|^2 \tag{4.26}$$

By empirically determining the threshold from the training set, outlier objects can be rejected using (4.1).

Chapter 5

Fault Analysis

5.1 Introduction

Fault analysis is a broadly classified discipline which involves fault detection and fault diagnosis in process engineering. Fault detection deals with timely detection of abnormal events in a process and fault diagnosis deals with isolation and identification of the origin of the fault [87]. Conventional fault detection approaches are based on detecting deviations from normal behavior of the machine operation. Various approaches used a variety of techniques such as quantitative modelling, qualitative or process history based models [87], [84], [88]. Quantitative approaches used mathematical modelling to express relations between inputs and outputs. In history based methods, a large amount of operational data is available or collected to build a model. These models were typically built for high critical systems where personnel safety or preventing hazards were given utmost importance such as in nuclear power plants. Due to advancements in sensor technology and computational capabilities, Fault Detection and Diagnosis is gaining awareness among academic researchers and industry professionals for Condition Based Maintenance (CBM) [84] of industrial machines/equipment. Condition Based Maintenance involves continuous health monitoring with predictive and diagnostics capability. Using prognostics, wear and degeneration of machine parts can be tracked and appropriate maintenance can be performed to minimize drastic faults [87], [84]. Novelty detection [87], [84], [89] plays an important role

in such systems where the diagnosis system must identify between normal, known fault and unknown faults. Typically normal operating data is sufficiently available to model the process but sometimes there may be no data or very few data points are available to model the faulty conditions.

Relying on human operators is becoming an increasingly difficult task due to the broad scope of process control and diagnostic activity. Modern process plants are complicated in size and operational complexity. Some operations can have more than 1000 variables that can be observed at a time. Diagnostic activity can encompass variety of malfunction such as process unit malfunction, unit degradation, sensor malfunction, etc. Human error can result in major accidents due to a simple malfunction which was not detected on time or frequent minor accidents leading to personnel injury and increasing downtime costs. Utilizing timely detection and performing predictive maintenance can help in avoiding major disaster due to petrochemical industries or minimizing cost of ownership of semiconductor wafer manufacturing plant.

Automation of regulatory control has led to increased efficiency, safety, and quality in process control. The next challenge is the automation of detection of abnormal events using intelligent systems. In [87], Venkatasubramanium et al study a detailed description of the characteristics of a fault diagnosis system such as quick response, isolability, robustness, adaptability is presented. For mass adoption of fault diagnosis for common industrial machines, the techniques must be generic, easily configurable, calibrated and autonomous. Incorporating these characteristics is the major challenge for a fault detection and diagnosis system. In this thesis we focus on fault detection using machine learning techniques.

5.2 Fault Detection

Fault analysis can be distinctly divided into fault detection and fault diagnosis steps depending on the application and the complexity of the operation. Fault detection step involves identifying the fault when the plant/machine operation deviates from its normal behavior. Fault diagnosis is the following step that is responsible in identifying the cause of the fault so the maintenance or replacement can be performed. These two steps can be combined which increases the complexity of the fault analysis system. Fault detection systems focus on quick response and novelty detection. Speed is required so that the operation can be stopped and the damage can be minimized. Novelty detection deals with identifying any abnormal behavior even if the fault was previously not observed. Fault diagnosis then determines the exact cause and whether the fault condition was progressive or an abrupt fault.

In quantitative modelling, a priori knowledge of the actual process is needed whereas in process history based methods, only past process data is needed. Machine learning methods are used to model only normal operation or both normal and faulty operation. These methods can be broadly classified into statistical and non-statistical learning methods. In machine learning feature extraction is an important step when using machine learning methods wherein, process data is transformed into new feature space possibly with fewer dimensions. Articles in [84], [88], [90], [91] extract various features for vibration analysis of machine bearing or gear functioning. It is not necessary to strictly implement the feature extraction step since machine learning algorithms depend on the data that they are provided. If the decision function of the machine learning algorithm can sufficiently discriminate the machine processes, then feature extraction in not required. In our previous work [92] we used raw data from a different (from the above mentioned) RF generator to train and validate two machine learning algorithms. This would work for simple machine processes and is biased by the data collection method since simulated faults may not

capture the abrupt behavior in the machine when an actual fault occurs during its field operation. These problems can be alleviated by using a suitable feature extraction method.

This step can also be preceded by feature selection where process variables are examined so that important variables can be identified for the process. Feature selection can lead to reduction in data and influence of noise or redundant variables which increases the overall efficiency in the machine learning and classification process. In our previous study we have demonstrated the application of feature selection methods for fault detection [92].

Using Radial Basis Function Networks, we can identify between different faults using process data. But this fails at novelty detection since the RBF is data dependent and will provide ambiguous results if data which is not in the training set is used. This is important in a fault detection system since every fault condition or unit failure cannot be simulated or recorded. Hence we take a one-class approach to model the normal operation. One-class techniques suit fault detection due to the inherent rejection capability of an outlying object. Using normal operation data, a one-class model is generated which is capable in identifying any abnormal operation. In the next section we present the Modified Novelty Detection Framework algorithm which is used for fault detection using one-class classification principles.

5.2.1 Novelty Detection Framework

As mentioned above traditional fault detection systems used physical modelling to establish relationship between variables and monitored any deviations from normal behavior. These techniques were often reserved for high critical systems due to high development cost. Due increasing awareness in need of a diagnostic system, fault detection systems are being developed and adopted for common industrial equipments for health monitoring

which benefit from maximizing production time and reducing the cost of ownership. Due to vastness and mass customization of industrial equipment, a generic fault detection system is necessary for autonomous diagnostics and prognostics which is cost effective for common industrial equipment [84].

To address the goal of developing a generic fault detection system, Novelty Detection Framework (NDF) was developed by the authors in [84]. The Novelty Detection Framework has various capabilities such as automatic detection of a number normal operating conditions based on past process data, novel fault detection and online learning which facilitates health monitoring. Using unsupervised techniques autonomous monitoring is achieved for health monitoring and diagnostics capability. Since the data used in this thesis is not suitable for applying the NDF algorithm directly we modify the NDF algorithm to operate in one-class context for fault detection which is presented next.

5.2.1.1 Modified NDF Algorithm

In NDF, the operating modes [84] are automatically identified using a clustering algorithm such as Greedy EM algorithm [83] defined by Gaussian mean and variance. This is an unsupervised technique applied to previously collected data which identifies the various regions of operations. By doing this a pre-set input is eliminated and the parameters remain free to be identified by the algorithm. Depending on the number of variables, feature extraction can be done before so that a more compact representation of the data is obtained. Once the operating modes are established, we adopt only the continuous fault detection part for our purposes.

Since a distance based measure is available, we ue the following equations used in NDF

similar to (4.26) and (4.1) to decide if the new sample belongs to a normal operating cluster.

$$J(\mathbf{z}) = \min_{c}(\mathbf{z} - \mu_c)\Sigma_c^{-1}(\mathbf{z} - \mu_c)^{-1} \tag{5.1}$$

$$f(\mathbf{z}) = I(J(\mathbf{z}) < \chi^2_{n,\beta}) \tag{5.2}$$

where μ_c and Σ_c are the mean and variance of the cluster c, $\mathbf{z}$ is the new sample, n is the dimension of the feature, and $\chi^2_{n,\beta}$ is the Chi-squared distribution with n degrees of freedom and probability β equal to 0.9973 which relates to the $\pm 3\sigma$ concept in process control.

The NDF algorithm incorporates online learning technique wherein each new data sample modifies the existing operating mode cluster parameters which is the backbone for health monitoring. We adopt a offline learning technique so that new data is incorporated into the data model. The NDF is applied to an individual unit/machine and the algorithm learns from the data collected from that particular unit. Since we do not have the facility to implement the NDF directly for an individual unit, we create a one-class data description model for a particular family of machines. By doing this we can learn from an old or a new generator to improve the model and incorporate any parameter differences due to physical differences of the machines. The above one-class model defined by the operating mode clusters is updated offline using new normal data. Using similar procedure as fault detection, a new cluster is created with the samples value (new normal data) as mean and the variance of the previous cluster if it did not belong to any existing cluster. By performing offline updates, we ensure that data from an old or new generator can be incorporated and a robust data description model is obtained. The same model can be used for the family of generators for fault detection.

5.3 Feature Extraction Methods

Feature extraction is the process of extracting information which contains discriminatory information while common information is reduced. In literature several feature extraction methods are used for fault analysis applications. The NDF algorithm [84] uses principal component analysis (PCA) [7] for reduce the dimension of the data to two. Articles in [88], [90], [91] extract various features for vibration analysis of machine bearing or gear functioning. It is not necessary to strictly implement the feature extraction step since machine learning algorithms depend on the data that they are provided. If the decision function of the machine learning algorithm can sufficiently discriminate the machine processes, then feature extraction in not required. For the results in sections 6.1 and 6.3.2 we used raw data from a RF generator to train and validate two machine learning algorithms. This would work for simple machine processes and is biased by the data collection method since simulated faults may not capture the abrupt behavior in the machine when an actual fault occurs during its field operation. These problems can be alleviated by using a suitable feature extraction method.

Chapter 6

Experimental Results

In this section we provide the experimental results for the algorithms described in this thesis. We divide the results into three sections for feature selection, one-class classification techniques and fault analysis. Different data sets are used to test the algorithms which will be detailed before presenting the results.

6.1 Feature Selection Results

We apply some of the feature selection algorithms on seven data sets using the two classifiers mentioned in chapter 3. Comparisons between the classifiers and the used algorithms is made.

6.1.1 Datasets and Parameter Settings

Seven data sets were utilized to verify the feature selection algorithms. Out of the seven data sets, four are form the UCI Machine Learning Repository [93] such as *ionosphere, diabetes, liver disorders* and *breast cancer* (cancer). The *Medical* dataset contains statistical data related to patients coming to the clinic provided by Bellevue Hospital in New York. In this dataset, there are 19 features that include demographical attributes of each patient such as ethnicity, sex, current diseases and additional binary class label representing whether the patient came to the scheduled appointment or not. The sixth is Fault Mode data form MKS

Instruments, NY. The Fault Mode data is obtained from a 5kW, 40MHz RF Power generator from MKS Instruments, NY. The generator consists of several subsystems including an AC/DC converter, RF Power Amplifier, RF Sensor, and Digital Control System given in figure 6.1. Sensors are designed into the RF generator to provide accurate power delivery and facilitate health monitoring during critical semiconductor/thin film processing steps. The data used in the study was created by intentionally seeding various faults and collecting generator sensor data during an automated RF power sweep designed in LabView^{TM}.The seventh dataset is the discrete Fault Mode data obtained by applying histogram to the continuous data. The seventh dataset is the discrete Fault Mode data obtained by applying

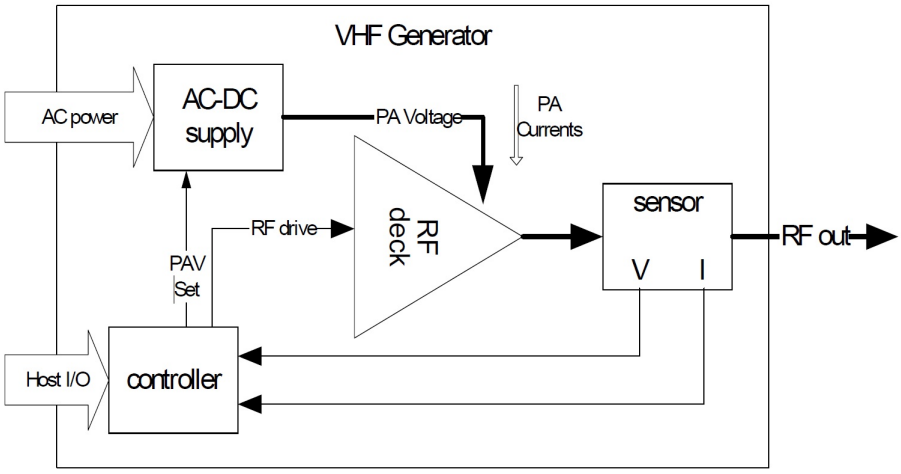


Figure 6.1: VHF RF generator block diagram

histogram to the continuous data. The number of variables in each dataset is given in table 6.1. The Fault Mode data contains 17 variables and 13860 samples. The output contains six classes of which one class is the no fault class which indicates no faults in the system. The other five classes are the various faults that can be detected in the system.

The datasets are divided into 50% training and 50% testing sets and the performance measure was the prediction accuracy of the test set. For all the results the test set was not shown to the feature selection algorithm. LIBSVM [75] library was used to implement

Table 6.1: Number of Variables in the Datasets used

Dataset	No. of Features
Breast Cancer	10
Diabetes	8
Ionosphere	34
hline Liver Disorder	6
Medical	19
Fault Mode	16

the SVM classifier. For the RBF classifier clustering must be performed to determine the optimum number of kernels, variance and their respective centers for (3.6). As suggested in [78] we perform an exhaustive search using K-means [7] technique to determine the RBF kernel centres. Since the input to the classifier varies according to the subset selected by the feature selection algorithm, using the parameters obtained from the whole dataset will bring down the efficiency of the classifier and results will not be accurate. Due to this clustering must be done for each subset which would increase the number of computations significantly. To reduce the number of computations we first find the number of kernels using the exhaustive search in range D to 2*D. The variance in (3.6) is found each time by searching in the range [0.1, 0.2, 0.3, ... 10]. By doing this we obtain better classification. To avoid indirectly learning the test data, the RBF exhaustive search is done on the training data itself. For the CHCGA algorithm, the GA parameters are: population size = 40, mutation rate = 0.4, maximum number of generations = 800.

6.1.2 Results and Comparison for Feature Selection Methods

In figure 6.2 the result for the first ranking method is given. The x-axis is the number of features and the y-axis is the performance of the subset containing the specified number of features. Figure 6.3 gives the result for the MI criteria (2.4), figure 6.4 gives the result of applying SFFS algorithm with SVM as the wrapper and figure 6.5 gives the result of applying SFFS with RBF as the wrapper. In figure 6.2 we can see that the performance for the first ranked feature for *Cancer* and *Ionosphere* gives low performance this is due

to the calculation of MI. Since we are approximating the pdf of a single feature and the output class distribution, calculation of MI will not be accurate and is easily influenced by marginal densities [23]. The irregular graph for the filter methods also proves that the ranking methods are trivial.

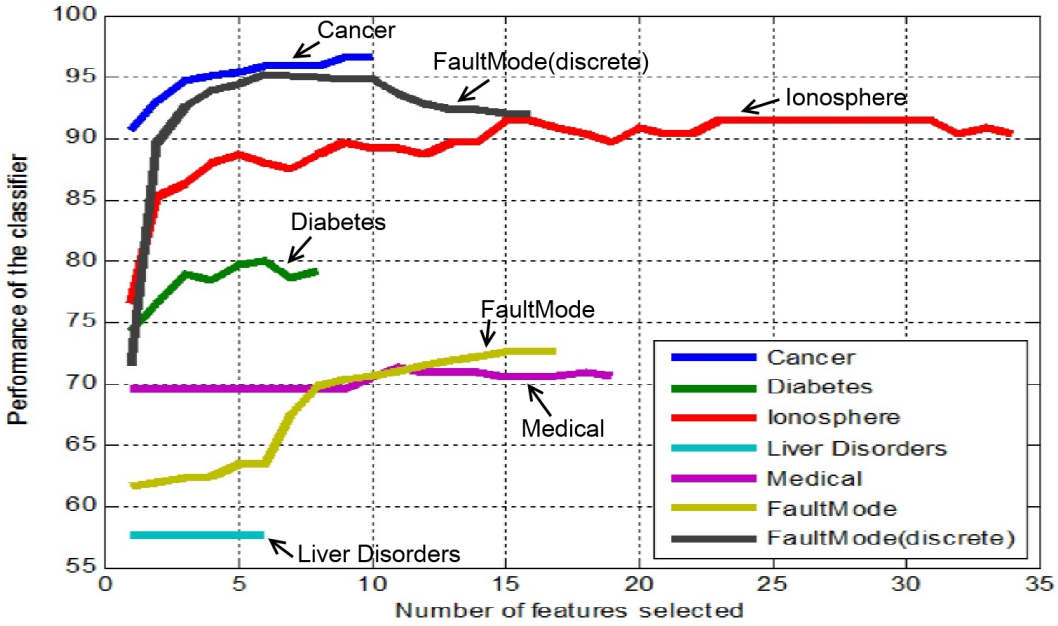


Figure 6.2: Result for correlation criteria using SVM

Since SFFS can produce only one subset, we generate subsets of lower size by backward elimination or a subset with larger number of features by forward addition to the subset selected by SFFS. By doing this we can see that the maximum performance is always obtained from the SFFS algorithm and the performance of all other subsets are either equal to or lower than the maximum performance. Figure 6.5 gives the results of applying SFFS using RBF classifier with the optimization procedure stated above. It can be seen in figure 6.5 that the graph does not give a clear idea of the feature selection due to optimization of RBF parameters for each subset. If the sample values are close to the center of the kernel, the weight contribution will be higher. It can be noticed in figure 6.4 that for *Liver Disorder* data the performance is the same for any subset which is interesting since

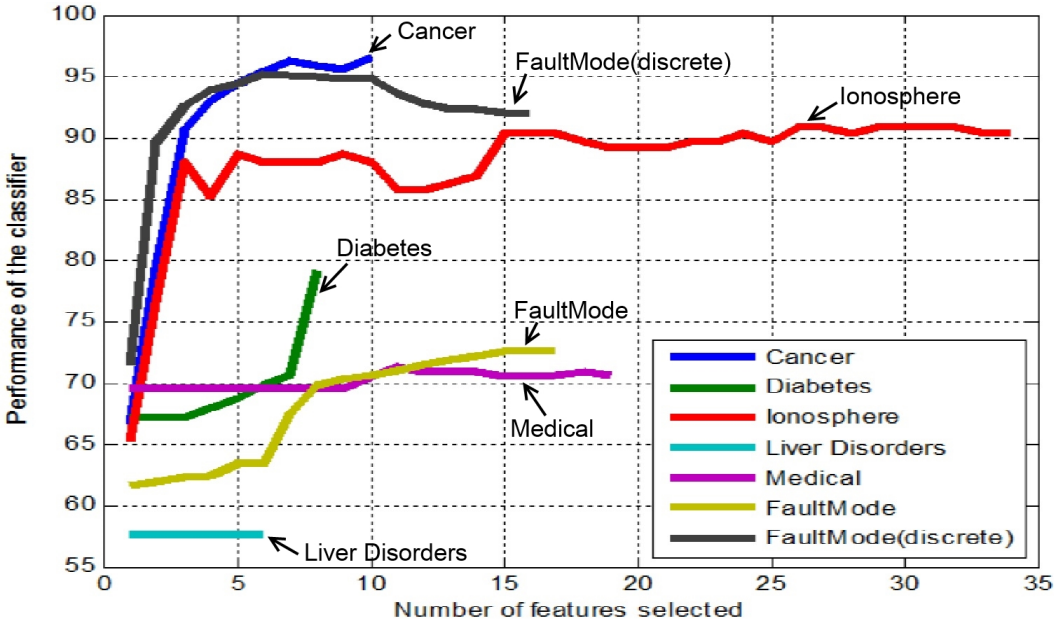


Figure 6.3: Result for MI using SVM

the same result is not obtained from RBF due to the optimization of the RBF classifier.

Table 6.2: Experimental Results for CHCGA with SVM

Dataset	Maximum Performance	No. of Selected Features
Breast Cancer	97.361	5
Diabetes	80.469	7
Ionosphere	94.286	16
Liver Disorder	57.558	4
Medical	73.2	8
FaultMode	76.392	6
FaultMode(discrete)	98.829	6

In table 6.2 we give the results of running the CHCGA algorithm with objective to find the subset that has the maximum classifier performance using SVM. We can see that the results in figure 6.4 and table 6.2 are close. Table 6.3 gives the results of running the CHCGA algorithm with RBF as the wrapper.

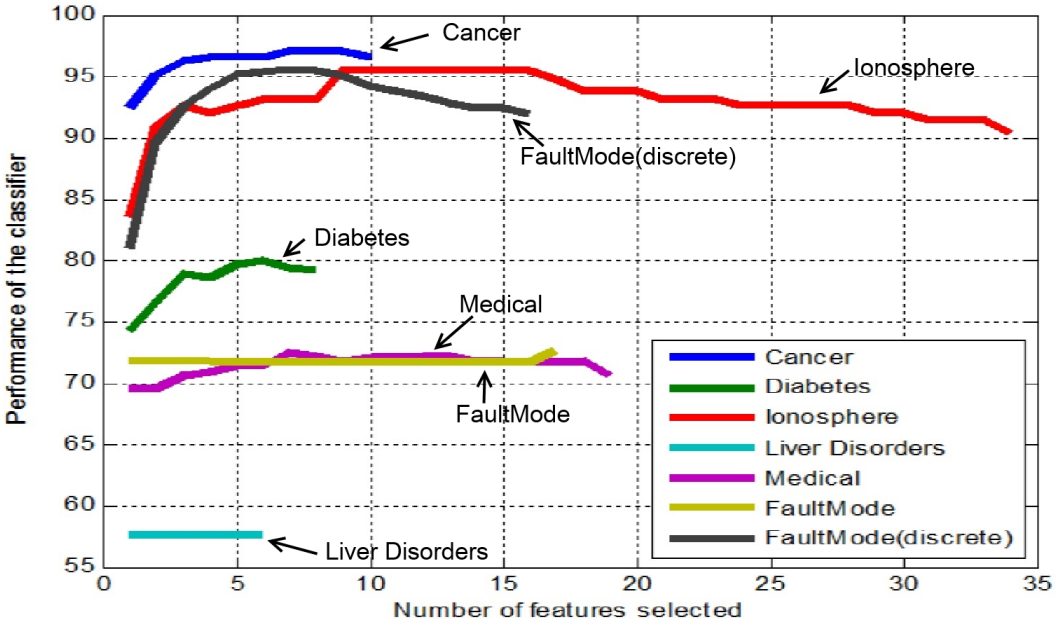


Figure 6.4: Results for SFFS using SVM

Table 6.3: Exprimental Results for CHCGA with RBF

Dataset	Maximum Performance	No. of Selected Features
Breast Cancer	96.774	5
Diabetes	76.822	7
Ionosphere	94.285	16
Liver Disorder	72.675	4
Medical	70	8
FaultMode	79.639	6
FaultMode(discrete)	82.799	6

6.2 One-class Classification Results

In this section we will provide the results for the one-class algorithms presented in this thesis. The results are generated using only artificial data sets. The application of one-class techniques for fault analysis will be provided in the following section. As before we will start with data set description and the results will be presented next.

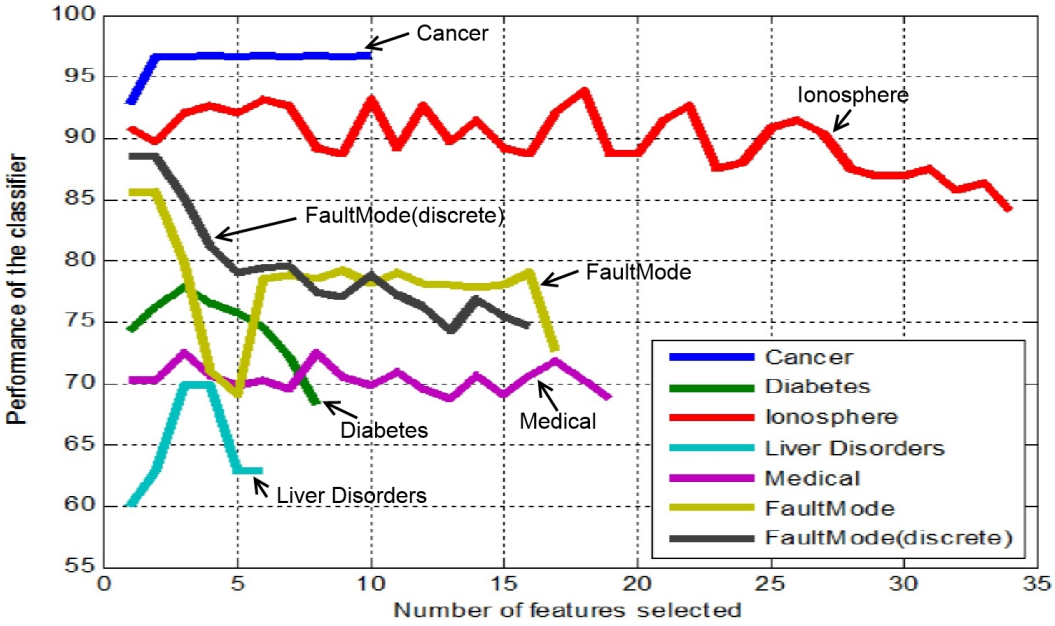


Figure 6.5: Results for SFFS using RBF

6.2.1 Datasets and Parameter Settings

We use three datasts to verify the one-class algorithms presented in this thesis. The first data set is an artificial banana distribution data set used in [82] and generated using the Matlab toolbox [94]. The other two datasets are *Cancer* and *Diabetes* datasets used above for verifying feature selection algorithms. In figure 6.6 the 2-dimensional banana distribution used to obtain results is given. The green ”+” symbols are target objects and red ”x” symbols represent outlier objects. The distribution contains 50 target samples and 150 outlier samples.

For MoG no user parameters are required and the algorithm automatically determines the number of Gaussian mixtures. For k-center algorithm, the number of balls to be placed are fixed to twice the dimension of the dataset. For SVDD, a new classifier must be trained for each target acceptance rate since the center and radius of the sphere is calculated using the support vectors lying on or at the boundary. The objects at the boundary will have

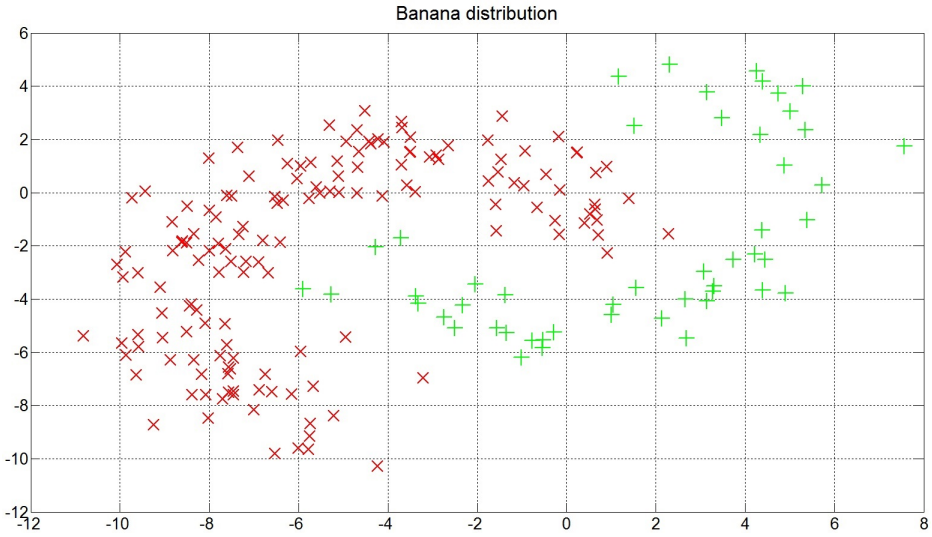


Figure 6.6: Banana distribution

positive α_i and the boundary is made tighter by restricting the upper bound on α_i. The constraint (4.15) is used to restrict α_i such that a predetermined percentage target objects are on or inside the boundary. The parameter C is the fraction of the target objects accepted which is used to tighten the boundary and reject some of the target objects. Since the FCM one-class is a clustering method, we fix the number of cluster to twice the dimension of the dataset.

For one-class classifier results in this thesis, we use a standard procedure to set the target acceptance rate f_{T+} and outlier rejection rate f_{O-}. We start with 50% target acceptance rate with an increment of 10% and test the complete outlier set with the obtained thresholds.

6.2.2 Results for One-class Classifiers

In figure 6.7 we present the receiver operating characteristic (ROC) curve for mixture of Gaussians for the above mentioned datasets. The trend in one-class classifiers can be observed from the three curves. At 100% target acceptance rate, the outlier rejection rate is less than 100%. This is due to the assumption made in each model about the data. For the

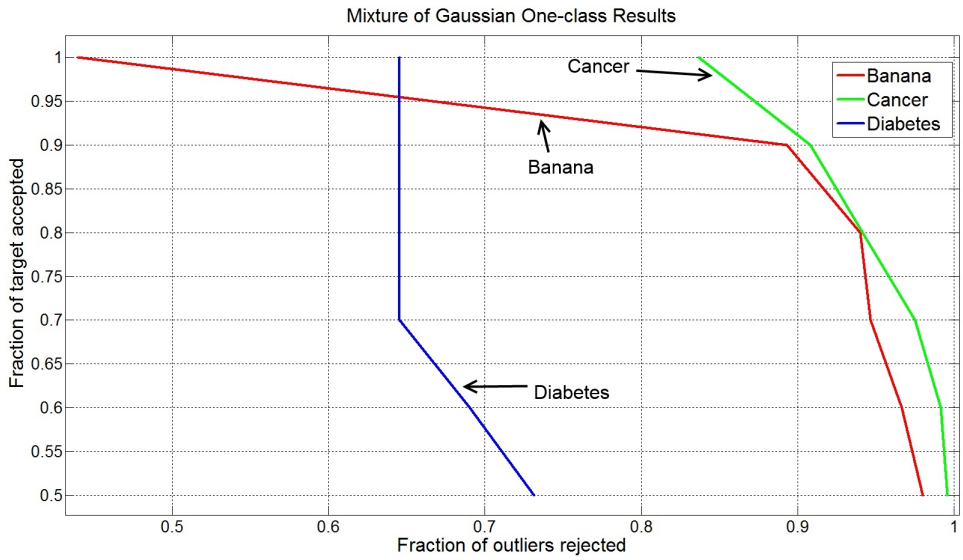


Figure 6.7: Mixture of Gaussian ROC curves.

banana dataset the MoG is sensitive to outliers due to a Gaussian assumption which defines a mean and variance. The MoG is also influenced by the greedy EM algorithm [83] which depends on the distribution of data to obtain clusters.

In figure 6.8 we give the results of applying k-center one-class algorithm. The k-center algorithm does well for banana distribution since the balls are placed on the target objects itself and depending on the radius the outliers are rejected better than MoG. Similar results are obtained for *cancer* and *diabetes* datasets.

The results for SVDD one-class experiments are given in figure 6.9. With tighter boundaries the SVDD will perform well. For the *cancer* data, the SVDD performs really well to obtain an almost ideal curve. Since its real data, the description will accept some outliers. The banana data ROC is consistent and varies slightly as new target objects are included in the training. The *diabetes* data ROC is still inconsistent and is difficult to obtain a suitable description.

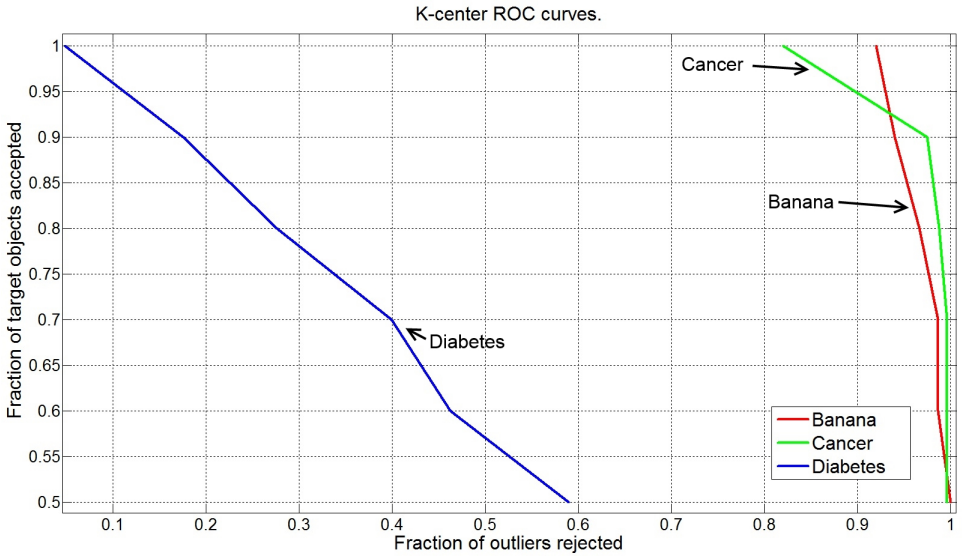


Figure 6.8: K-center ROC curves.

In figure 6.10 the results for the FCM one-class is given. The FCM one-class does well for both banana and *cancer* datasets with good rejection rates. The number of clusters and the distribution of the data influences the data description model. Good results are obtained if the number of selected cluster match the actual subspace distribution of the data. As usual the *diabetes* dataset acceptance/rejection rates are not acceptable due to failing in the assumption of the data distribution.

6.3 Fault Analysis Results

In this section we present the results for fault detection obtained using feature selection and one-class algorithms. The data was obtained from two different families of RF generators from MKS Instruments.

6.3.1 Datasets and Feature Extraction

The first generator data is described in section 6.1.1. No feature extraction was performed on this dataset. Some of the results are already presented in section 6.1 which was relevant

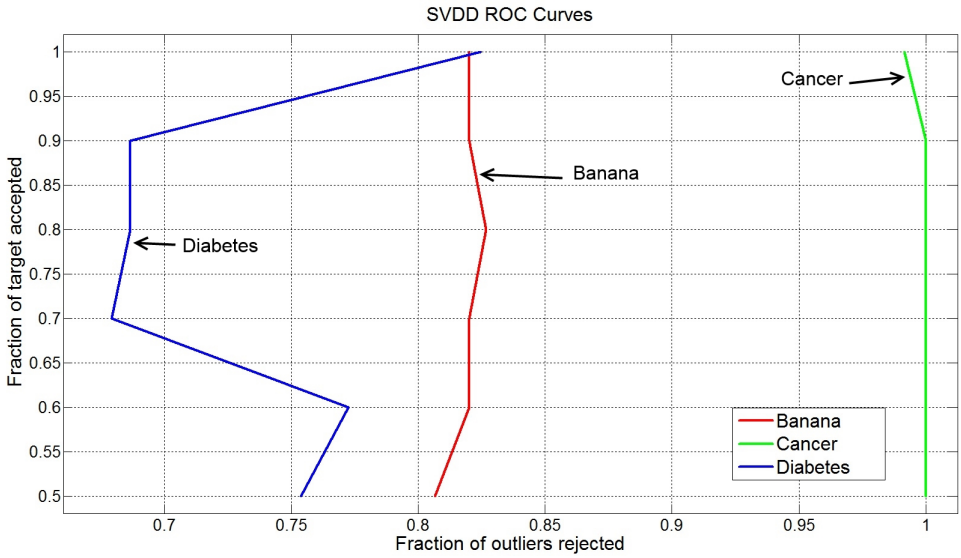


Figure 6.9: SVDD ROC curves.

for verifying the feature selection algorithms. In section 6.3.2 we present further results which are specific to fault detection.

Data from second generator belonging to model GHW50A is applied to one-class algorithms for fault detection is presented in section 6.3.4. The second generator is a 5kW, 13.56MHz RF Power generator from MKS Instruments. The block diagram is similar to figure 6.1. The data used in the study was created by intentionally seeding various faults and collecting generator sensor data during an automated RF power sweep designed in LabView^{TM}. Each test run consists of power sweeps over three frequencies 12.88MHz, 13.56MHz and 14.24MHz. For each frequency the required output power is set from 0W to 5KW in steps of 500W. A total of nine variables can be observed from the generator control unit. An extra variable of 0 or 1 is added to account for open or 50 load for each sample. For each test run we obtain 165 samples with 10 variables accounting for power sweep measurements and load condition. Only one type of load is set for each test run. Two datasets are constructed using data from six different generators belonging to the GHW50A model. The functioning capabilities of all the generators are the same, but they

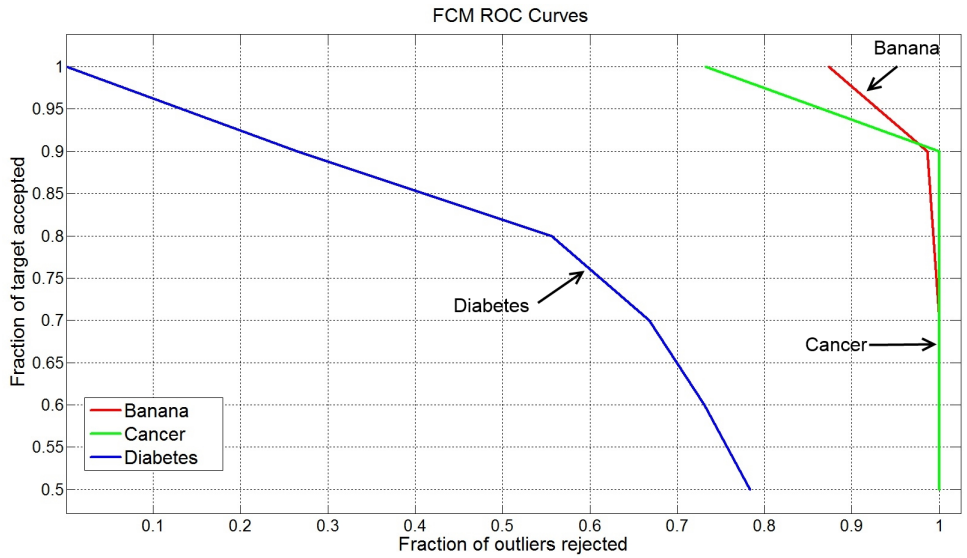


Figure 6.10: FCM ROC curves.

differ slightly in electrical performance. The first data set contains normal operation with only one high operating temperature fault which makes up for 96% of all fault data. The second data set adds the remaining 4% of fault data which contains five other faults.

For the second generator data, we apply the multi-dimensional Fast Fourier Transform (FFT) for each individual test run. The Fourier coefficients (absolute values) of each variable (column) are concatenated to form a single row feature vector. To reduce the FFT data and to get rid of coefficients with very small values, we divide 165 coefficients of each variable into 49 bins with the number of coefficients in each bin rounded down to the nearest integer. The coefficients in each bin are summed to obtain a new feature vector. To account for the redundancies in FFT and adding a constant load variable, we further apply Principal Component Analysis (PCA) [7] to obtain a much more compact representation of the data. We apply PCA such that 99% of the variance is retained.

6.3.2 Results for Fault Analysis using Feature Selection

Only the Fault Mode data from the first generator is used for feature selection algorithms. We use the confusion matrix given in table 6.4 to calculate the performance of the of the feature selection methods. We use three metrics Fault Detection Rate (FDR), Performance and Diagnosis defined as,

$$FDR = \frac{TP}{TP + FP} \tag{6.1}$$

$$Performance = \frac{TP + TN}{TP + FN + FP + TN} \tag{6.2}$$

$$Diagnosis = \frac{\text{no. of correct faults detected}}{\text{no. of faults}} \tag{6.3}$$

Table 6.4: Confusion Matrix

	Actual Output		
Estimated Output		Fault	No Fault
	Fault	True positive (TP)	False positive (FP)
	No Fault	False negative (FN)	True negative (TN)

In figure 6.11 we give the feature selection plot similar to figures 6.4 and 6.5 for the Fault Mode data only. In table 6.5 we present the experimental results obtained for the original data without variable elimination for SVM and RBF classifiers. The number of faults in the test set for continuous data is 1967 and 1982 for discrete data. For continuous data the RBF gives slightly better performance than SVM with more faults detected and better diagnosis. FDR tells us if any of the faults are misclassified as no fault. Performance value gives us the whole accuracy of fault detection and no fault detection. With Diagnosis we can measure if the fault detected is misclassified as other faults. This value is generally low since the number of faults detected is low. For discrete data the SVM performs better with high accuracy, FDR, Performance and Diagnosis values. This is probably because the SVM can find better support vectors with discrete data than continuous data.

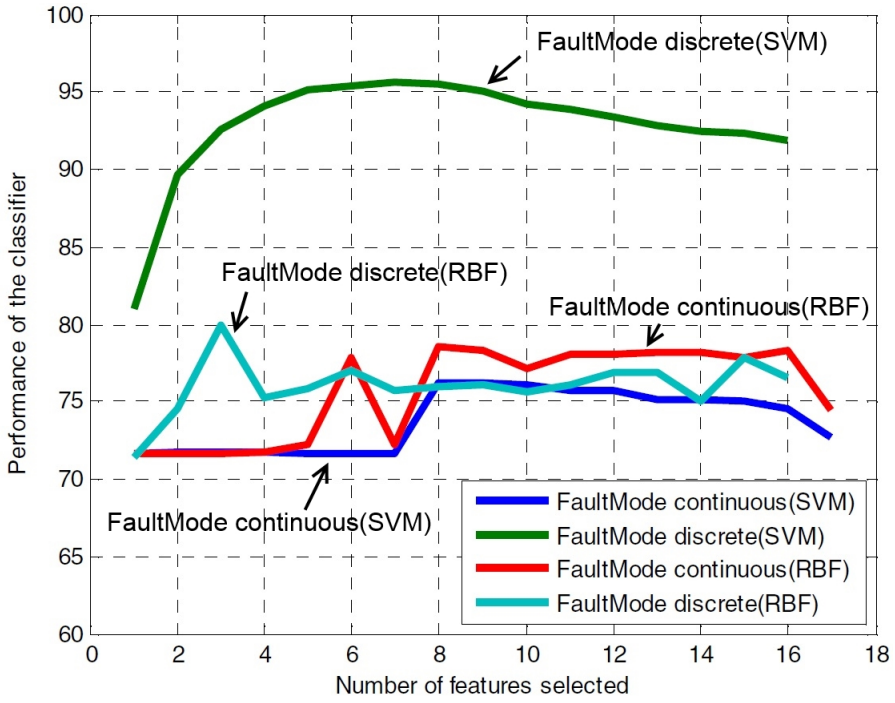


Figure 6.11: Results for Fault Mode data only.

Table 6.5: Results for original Fault mode data.

	Continuous		Discrete	
	SVM	RBF	SVM	RBF
Classifier Accuracy	72.67	78.543	91.876	76.595
Detected Faults	222	567	1520	322
FDR	1	0.9788	0.9967	0.9410
Performance	0.7482	0.7945	0.9319	0.7550
Diagnosis	0.1129	0.2883	0.7669	0.1625

In table 6.6 we present the results for SFFS algorithm. The RBF gives slightly better performance for continuous data when compared to SVM as in table 6.5 with the number of features reduced to eight by both classifiers. For discrete data, the SVM again performs better with high Diagnosis. For the RBF the maximum performance was obtained with three features only. This is because of optimization of the parameters for each subset. If the sample values are close to the center of the kernel, the weight contribution will be higher. In table 6.7 we present the details for CHCGA algorithm. The CHCGA reduces both the features equally for both classifiers. The RBF results are influenced by the cluster centers obtained from k-means and variance parameters. Again we can observe RBF doing slightly better than SVM for continuous and SVM for discrete data.

Table 6.6: Results Fault mode data using SFFS.

	Continuous		Discrete	
	SVM	RBF	SVM	RBF
Classifier Accuracy	72.234	78.543	95.556	80
No. of features	8	8	7	3
Detected Faults	320	492	1830	430
FDR	1	0.9715	0.9995	1
Performance	0.7623	0.7831	0.9778	0.7760
Diagnosis	0.1627	0.2501	0.9233	0.2170

Table 6.7: Results Fault mode data using CHCGA.

	Continuous		Discrete	
	SVM	RBF	SVM	RBF
Classifier Accuracy	76.392	79.004	95.368	77.561
No. of features	6	12	6	12
Detected Faults	331	402	1849	356
FDR	1	0.9726	1	0.9663
Performance	0.7639	0.7710	0.9808	0.7619
Diagnosis	0.1683	0.2044	0.9329	0.1796

6.3.3 RBF Training for GHW50A data

We apply RBF for the first dataset constructed from GHW50A data. It functions as a two class classifier to differentiate between normal and the high temperature fault condition. Due to the flexibility in tuning the RBF parameters such as number of kernels and variance of the kernels, we use three performance metrics to evaluate different RBF networks. We search for a suitable network by keeping the variance of the Gaussian kernel constant to a very high value (e.g. 100000) and vary the number of kernels. The K-means [7] algorithm is used to find the kernel centers in the range from D to 2*D where D is the dimension of the data. The data set is split into three categories with 50% for training, 25% for testing and 25% robustness testing. All three sets contain both fault and normal data. The robustness testing data set contains data from a single generator whose data is not in the training and testing set. The performance metrics used are training cross validation performance, testing performance and robustness test performance. The robustness test set was created to ensure that the classifier can generalize the normal and a single fault condition across different generators of the same model. With this procedure a classifier with 100% accuracy for all three metrics can be obtained.

6.3.4 Results for Fault Analysis using One-class Techniques

In this section we apply the one-class algorithms and Modified NDF to the second generator data as described in section 6.3.1. For one-class training normal data is used to obtain the one-class data description model. The first dataset with all faults is used. In figure 6.12 we give the results for MoG and FCM one-class. The other two results for k-center and SVDD classifiers is given in figure 6.13 . Except SVDD all methods perform well with very good rejection rates. The SVDD performs poorly with highly varying ROC curve is due to the fact that the data contains subspaces (clusters). The SVDD can only construct a tight boundary around the target data and if disconnected clusters are present such as in

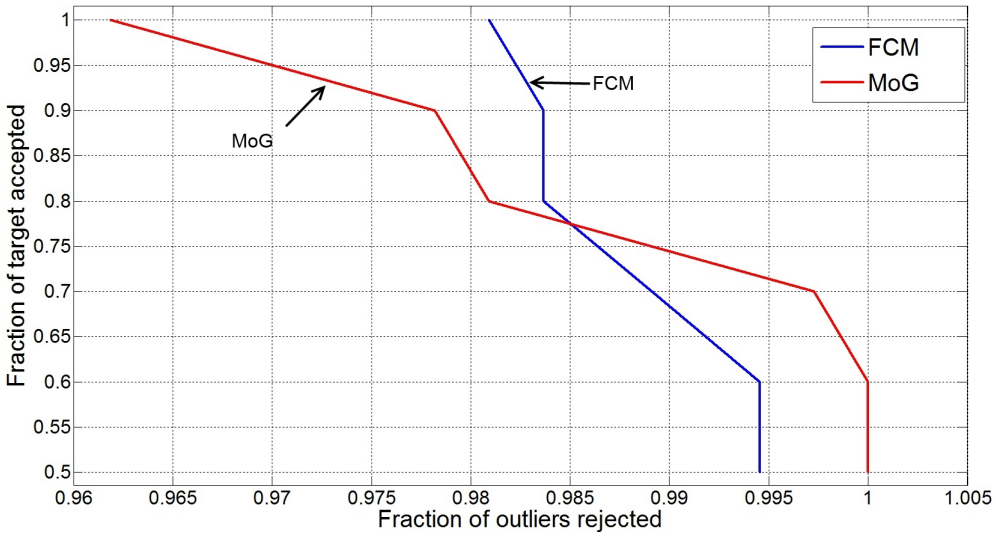


Figure 6.12: MoG and FCM ROC curves for GHW50A data.

GHW50A data, the SVDD will encompass any fault data that are adjacent to the clusters. The k-center ROC curve produces an ideal curve due to the worst case placement of ball

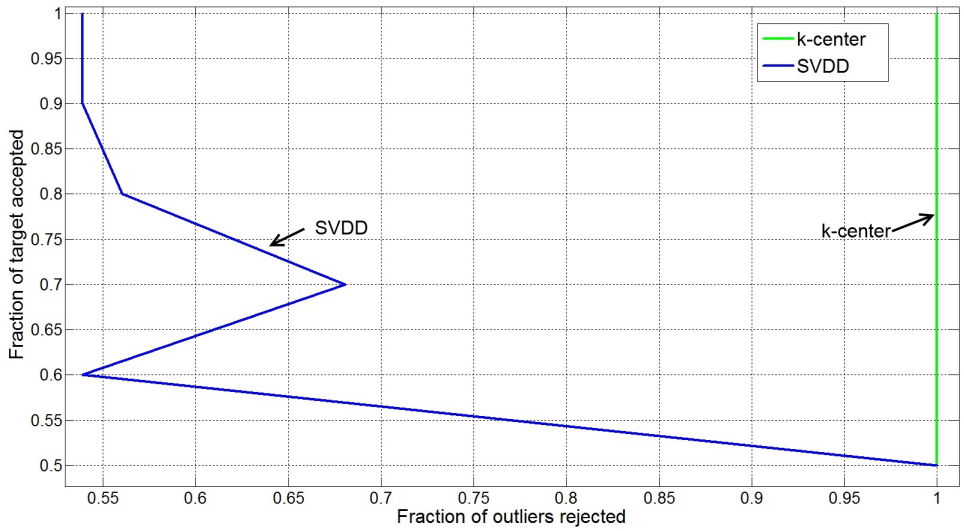


Figure 6.13: k-center and SVDD ROC curves for GHW50A data.

on target data. The k-center and FCM algorithm rejection rates for 100% target acceptance rates are very close. Even though we have 100% outlier rejection it is because we are also

classifying a small percentage of target objects as outlier objects. The reason that the k-center method can perform very well is due to the success of the feature extraction method to separate the normal and fault data.

The NDF algorithm [84] is not a one-class classifier hence we do not obtain an ROC curve for the Modified NDF algorithm (section 5.2.1.1). To evaluate the modified NDF model similar to RBF robust testing (section 6.3.3), we remove one generator data from the training set consisting of normal and fault data. We refer to the dataset of five generators as dataset 3 and dataset 4 which contains the sixth generator data. After training, the model is tested with fault data of the dataset 3, normal data of dataset 4 and fault data of dataset 4. The existing model is updated using the normal data from dataset 4. A final test is done for the dataset 4 fault data using the updated model. The results given in Table 6.8 are average performance of ten runs. From Table 6.8, we get very good rejection for fault data of dataset 3. The normal data of dataset 4 has a high rejection rate since the normal operation is not available in the operating mode clusters. The fault data of dataset 4 has a high rejection rate since the model rejects every object except the objects similar to the normal operation. We update the model using the above mentioned procedure to learn the new generator normal operation. After updating, the model parameters change and we observe a rejection rate close to before update. The goal of updating the operating mode clusters is to avoid classification of normal operation from different generators (of the same model) as fault.

We have concluded the experimental results section for this thesis. In the next chapter we provide details LabView^{TM}software implementation built for MKS Instruments for fault detection.

Table 6.8: Results for modified NDF.

Dataset	Rejection performance (% avg. of 10 runs)	Standard deviation of rejection performance
Dataset 3 fault	98.02	3.32
Dataset 4 normal (before update)	55	15.81
Dataset 4 fault (before update)	80.73	17.26
Dataset 4 fault (after update)	79.573	16.11

Chapter 7

Software Implementation for Fault Detection

7.1 Introduction

In section 6.3.3 we analyze the application of RBF classifier to fault data from GHW50A generator for fault detection. The algorithms in this thesis are developed and tested using the MATLAB^{TM}environment. Due to the adoption of LabView^{TM}software by MKS Instruments, a software interface for a fault detection system using RBF classifier is developed. The software supports database management, RBF training, testing new data and online data collection. The RBF classifier developed is a two class classifier which can distinguish between normal and one fault condition as previous results are presented. The LabView^{TM}environment enables the deployment of software over a wide range of indusrial products and factory sites world-wide. The implementation details of the fault detection software is presented next.

7.2 RIT Fault Prediction

The software is named **RIT Fault Prediction**. The main menu interface for version 1.1 is given in figure 7.1. From the figure 7.1 we can see that the interface supports six options:

- Add new generator/files to the database.
- Train the RBF Classifier.

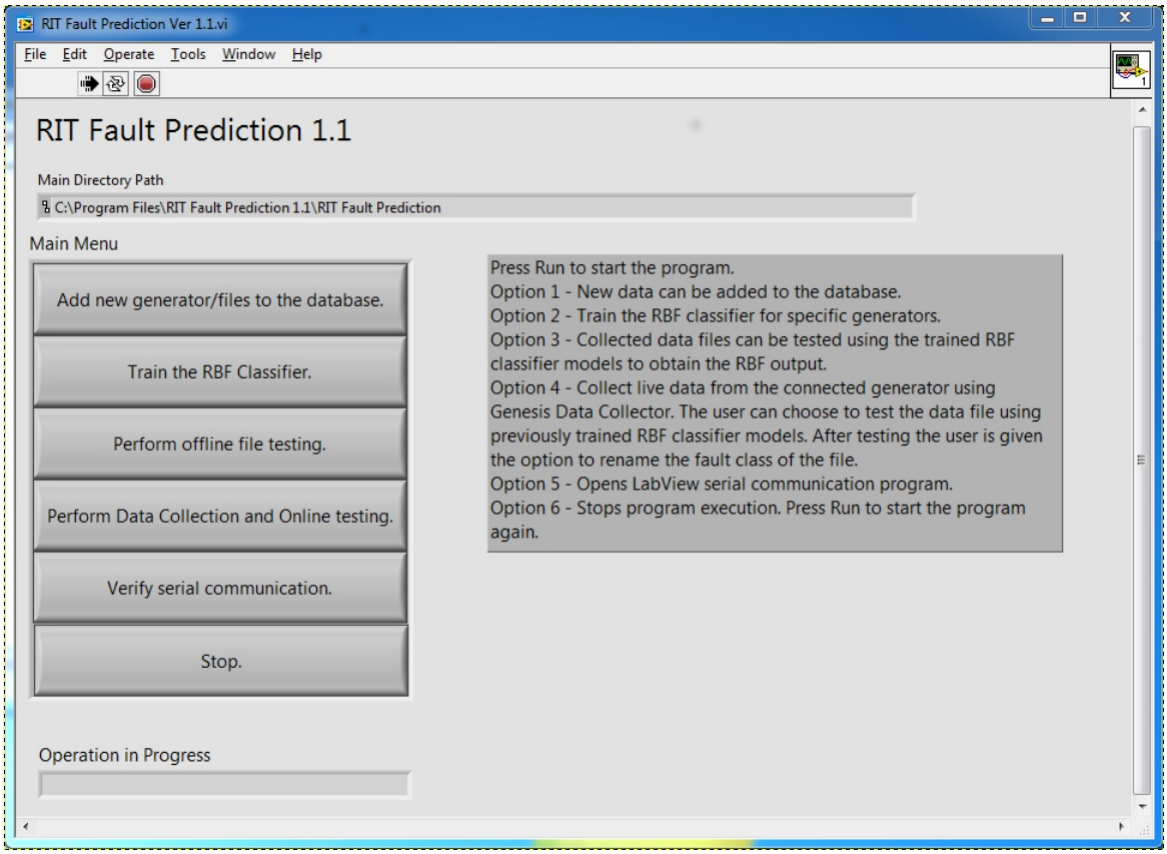


Figure 7.1: Fault prediction software interface.

- Perform offline file testing.
- Perform Data Collection and Online Testing.
- Verify serial communication.
- Stop.

Each option is designed so that an engineer can operate the software with ease and with knowledge of only the software documentation. Each option has its own interface which hides the inner workings and only user input and relevant information is displayed to the user. The software is a standalone install which can be installed on a Windows^{TM}operating system and can operate independently without requiring any other extra software libraries. The details of each option are presented next.

7.2.1 Option 1: Database Management

This option was developed for efficient storage of generator data. Since there are several RF generator models in use and manufactured by MKS Instruments, there was a need to collect and segregate the data from different generators so that different RBF classifiers can be trained for each model. The interface is given in figure 7.2. Any input data is copied

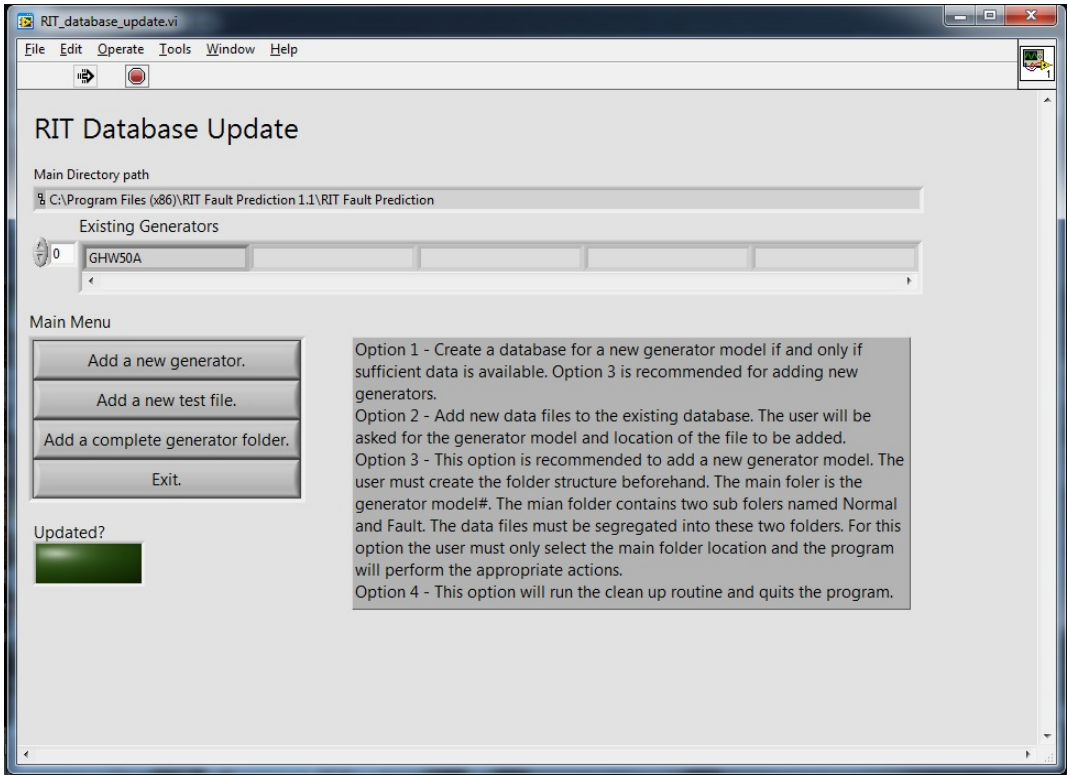


Figure 7.2: RIT fault prediction database management interface.

into designated folders created in the software install directory which is displayed on the interface. In each generator folder two sub folders are created in which the normal and fault data are stored.

The interface contains four options:

- Add a new generator: Using this option, a new folder can be created for a new generator. The program creates the normal and fault folders for each new generator. After

successful creation of the folder, .dat files can be added.

- Add a new test file: For an existing generator, new collected .dat files can be added to update the database for the required generator.
- Add a complete generator folder: Since the file directory is simple, the user can create the required folder structure and add a folder which contains all data at once using this option. Depending on the users wish, either the first option or this option can be used for adding new generator data. The user will need to point to the main generator folder.
- Exit: This option will close the database management interface and returns to the main menu.

Since only generator name is input from the user for adding subsequent files, we follow a standard file naming scheme to recognize .dat files. The data collected from this software conforms to below file naming format. The file name must contain the following fields:

generator_model# serial# load_value load **fault_class** ***date time***.dat

This naming scheme is adopted to minimize user input. The first two fields indicates the model number and serial number of the generator. Each field must be a single word and the underscore can be used to merge two different words so that no white spaces are in each field. The **load_value** field contains the load name used to obtain the data e.g. 50_ohm. The date and time fields indicate the date and time at file creation. This helps to generate a unique file identifier for the data files. The **fault_class** is not a numeric value but a single word (case sensitive) representing the condition of the generator e.g. Normal. For this software, generator model# is represented as the parent and the generators belonging to the same generator model# will be treated as its children under the parent model.

The input data files are raw and are copied directly into the respective folders. In order to create valid RBF classifiers, both normal and fault data must be present for training. Hence database maintenance is performed when the database management interface is closed. If any generator folder is missing normal or fault data files, the generator folder is deleted from the database. After this as mentioned in section 6.3.1, the feature extraction procedure is run on the stored raw data. A single .dat file is compiled for each generator which contains the feature extracted data which is stored in the respective generator folder.

7.2.2 Option 2: Train RBF Classifier

Multiple RBF classifier models can be created using this option. The user interface is given in figure 7.3. The available generators are displayed in the second field. The user must

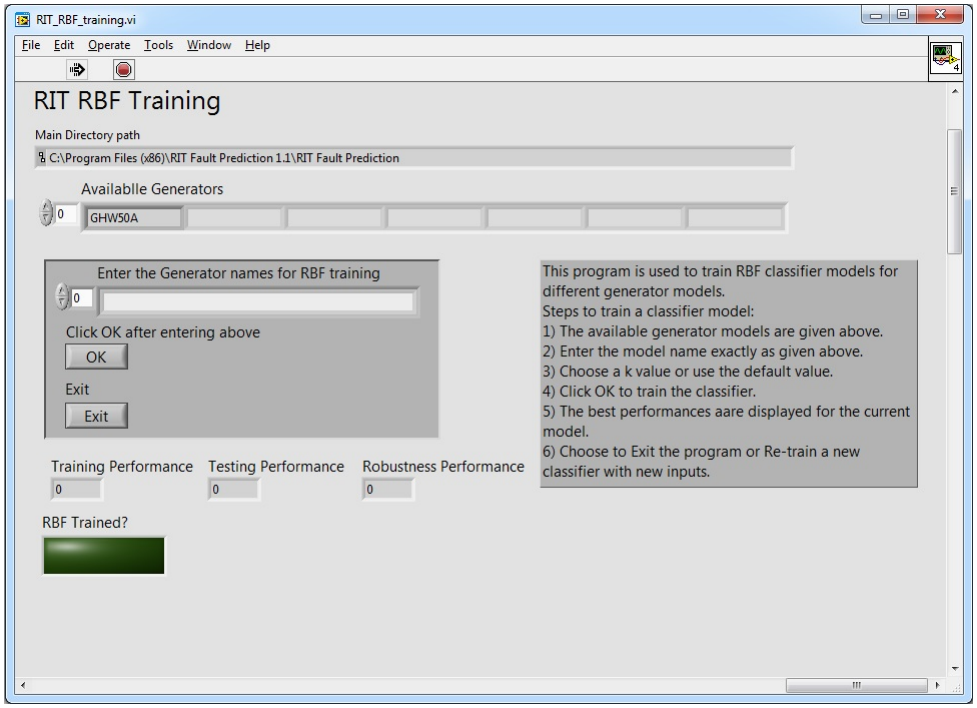


Figure 7.3: RIT fault prediction RBF training interface.

type in the name of the generator to be used to train the RBF classifier. The consolidated feature extracted file for the input generator is read to obtain the training, testing and robust

testing data sets. The RBF parameters and training procedure is the same as mentioned in section 6.3.3. The three performance measurement metrics for the current trained classifier are also displayed. The model file name is generated using the following template:

generator_model# date time

As the name of each field indicate the generator model, date and time are used to generate unique file names. The program will exit if there are no generators available in the database. Since date and time are used to generate a unique file name, multiple models for a single generator can be created. The trained RBF classifier is stored in the software directory.

7.2.3 Option 3: Offline File Testing

This option can be used to test/validate a data file using any previously trained RBF classifier models. The interface is given in figure 7.4. This is referred to as **"offline"** file test which means that a data file collected elsewhere can be tested using the models present on the user system.

When the interface opens, the user is asked to select the raw data file. If no trained models are present, the interface will quit displaying the reason. The available classifiers are displayed. After the test file is input, the user must enter the name of the model to be used for testing. The input file must also conform to the naming format used by the software. This is done so that the generator model of the classifier and the input file match. If the input file belongs to a different generator, no testing is done and the user is asked to make the appropriate adjustments. If the test is successful, the RBF decision and the actual RBF output is displayed. The RBF output is usually threshold to the nearest integer to get an integer class value (0 or 1).

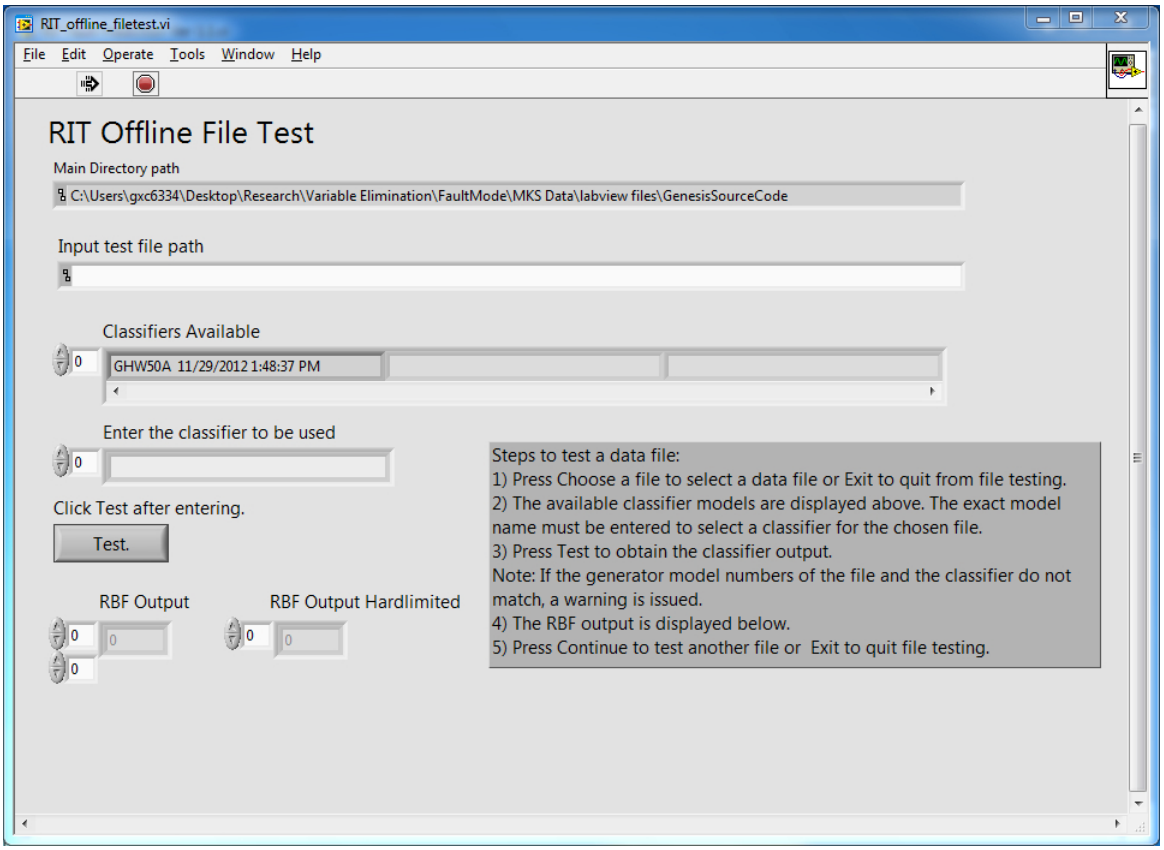


Figure 7.4: RIT Fault Prediction offline file test interface.

7.2.4 Option 4: Online Data Collection and Online Testing

Using this option, data from a generator connected to the computer running the software can be collected and stored. The generator communicates with the software using a standard RS232 serial communication. The data collection LabView^{TM}program is called the **Genesis Data Collector** which is developed by MKS Instruments. The interface used in the fault prediction software is given in figure 7.5. As mentioned above we adopt a standard file naming format for the data files. The original **Genesis Data Collector** is modified to accommodate the naming scheme. The interface provides clear inputs to setup the data collection run. The program asks the user for the location to store the raw data.

After the data collection has successfully completed, the user is given an option to test

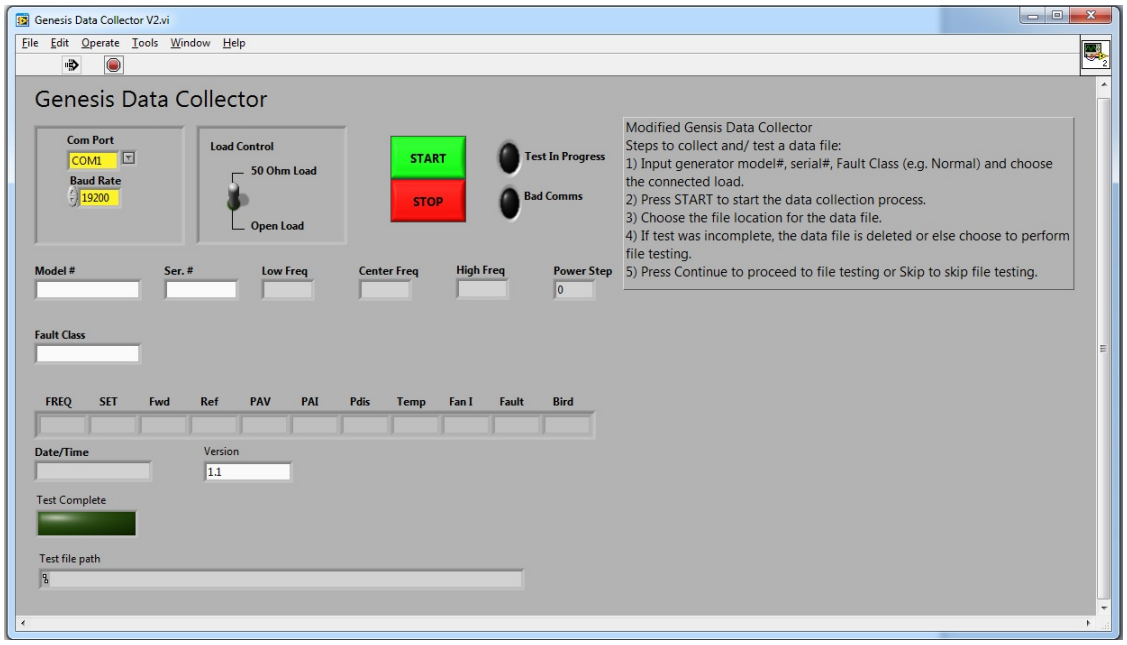


Figure 7.5: Fault prediction Genesis Data Collector interface.

the collected file with the available classifiers. If **"online"** test option is chosen, the offline test interface is brought up with the current file already provided as input to the testing program.

7.2.5 Option 5: Verify Serial Communication

This sub routine was added to provide a LabView^{TM}serial communication interface for the software. This is used to verify if the communication between the software and the generator is working since the **Genesis Data Collector** directly starts the data collection procedure and ensuring the communication between the software and the generator is an important setup step. The interface is given in figure 7.6. The interface emulates a generic serial terminal but with very limited functionality built to communicate with the RF generators from MKS Instruments. The serial port and baud rate can be set by the user. Only one command can be sent to the generator at once using the *Write* command. The program will wait until the input *timeout* period expires or if data is read back from the generator. Typically one

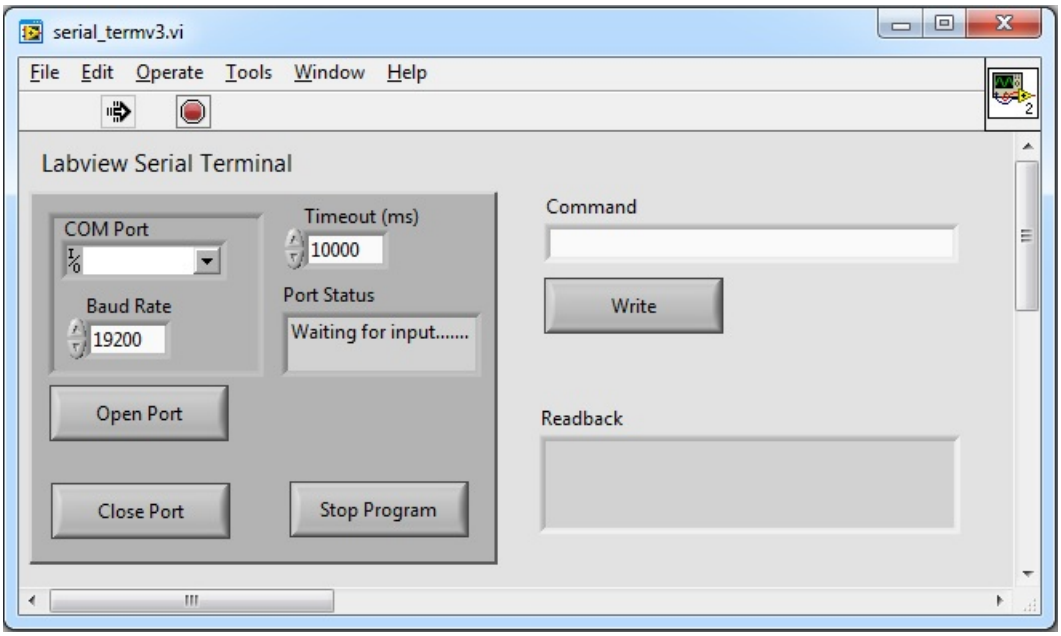


Figure 7.6: Fault prediction serial communication interface.

line commands are used to invoke simple requests from the generator control unit. Since the software uses the LabView^{TM}environment, this also ensures that the LabView^{TM}serial interface routine can function with the installed serial port on the computer.

7.2.6 Option 6: Stop

This is the final option whose function is to quit from the fault prediction main program which handles the above mentioned interfaces.

Chapter 8

Conclusions

In thesis we cover few focus areas in Machine Learning and provides an introductory approach to learning and applying these techniques. During the scope of this study feature selection, one-class classification techniques and their application to fault analysis are studied, verified and applied to standard datasets and data collected from RF generators. Also a survey study on feature selection algorithms is also presented.

Using feature selection techniques and state-of-the-art classifiers, generator data is analyzed to study the effect of unnecessary sensor variables. Few feature selection algorithms are applied to SVM and RBF classifiers to compare the classifier performance and the number of features selected. The effect of using continuous and discrete data is also investigated which provides insight into classifier mathematics. SVM classifier does well for discrete data due to discrete support vectors which provide better class separation.

Fourier transform is successfully applied as feature extraction technique for RF generator data. Using RBF optimization techniques, a robust two-class classifier is built for data from six generators. One-class techniques are studied to provide an alternative to multi-class classifiers which depend on data to distinguish between different classes. Various techniques are studied to detect if the machine deviates from its normal operation which

can function as a preliminary technique for fault detection. Due to the nature of data distribution and SVDD model assumption, the SVDD fails to capture a valid model. The FCM and MoG one-class perform well for fault detection after applying our feature extraction method. The modified NDF algorithm also provides results comparable to formal one-class techniques in terms of classifier performance.

To move forward from academic research to applying these algorithms for industrial use, the **RIT Fault Prediction** software is implemented. Even though only a two-class capable classifier is currently implemented, the software can be used as a learning tool to introduce machine learning based applications in an industrial setting. The software also supports standard data collection program using which data from different machines can be collected and unified for continuous study and future development.

Bibliography

[1] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *In Fifth Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.

[2] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *JMLR*, vol. 3, pp. 1157–1182, 2003.

[3] I. Guyon, J.Weston, S. Barhill, and V. Vapnik, "gene selection for cancer classification using support vector machines," in *Mach. Learn*, vol. 46, 2002, pp. 389–422.

[4] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," in *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 2, 2005, pp. 185–205.

[5] L.-Y. Chuang, H.-W. Chang, C.-J. Tu, and C.-H. Yang, "Improved binary pso for feature selection using gene expression data," in *Computational Biology and Chemistry*, vol. 32, 2008, pp. 29–38.

[6] C. Lazar, J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, hughes Bersini, and A. Nowe, "A survey on filter techniques for feature selection in gene expression microarray analysis," in *IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS*, vol. 9, no. 4, July/August 2012.

[7] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, 2004.

[8] M. H. Law, M. rio A.T. Figueiredo, and A. K. Jain, "Simultaneous feature selection and clustering using mixture models," in *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 26, no. 9, September 2004.

[9] R. Kohavi and G. H. John, "Wrappers for feature subset selection," in *Artificial Intelligence*, vol. 97, 1997, pp. 273–324.

[10] P. Langley, "Selection of relevant features in machine learning," in *AAAI Fall Symp. Relevance*, 1994.

[11] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," in *Artificial Inteligence*, vol. 97, 1997, pp. 245–270.

[12] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," in *Proc. 11th Int. Conf. Mach. Learn.*, 1994, pp. 121–129.

[13] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," in *IEEE Transactions on Neural Networks*, vol. 5, no. 4, 1994.

[14] G. Forman, "An extensive empirical study of feature selection metrics for text classification," in *JMLR*, vol. 3, 2003, pp. 1289–1306.

[15] N. Kwak and C.-H. Choi, "Input feature selection for classification problems," in *IEEE Trans. Neural Networks*, vol. 13, no. 1, 2002, pp. 143–159.

[16] P. Comon, "Independent component analysis, a new concept?" in *Signal Processing*, vol. 36, 1994, pp. 287–314.

[17] K. Torkkola, "On feature extraction by non-parametric mutual information maximization," in *JMLR*, vol. 3, 2003, pp. 1415–1438.

[18] F. Fleuret, "Fast binary feature selection with conditional mutual information," in . *Machine Learning Research*, vol. 5, 2004, pp. 1531–1555.

[19] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter, "Distributional word clusters vs. words for text categorization," in *JMLR*, vol. 3, 2003, pp. 1245–1264.

[20] R. Caruana and V. de S, "Benefitting from the variables that variable selection discards," in *JMLR*, vol. 3, 2003, pp. 1245–1264.

[21] D. Koller and M. Sahami, "Towards optimal feature selection," in *ICML*, vol. 96, 1996, pp. 284–292.

[22] J. L. Davidson and J. Jalan, "Feature selection for steganalysis using the mahalonobis distance," in *Proc. SPIE 7541, Media Forensics and Security II*, vol. 7541, 2010.

[23] Y. Yang and J. O. Perdersen, "A comparative study on feature selection in text categorization," in *International Conference on Machine Learning*, 1997.

[24] K. Javed, H. A. Babri, and M. Saeed, "Feature selection based on class-dependent densities for high-dimensional binary data," in *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, vol. 24, no. 3, March 2010.

[25] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," in *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, August 2005.

[26] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Proceedings of Tenth National Conference on Artificial Intelligence*, 1992, pp. 129–134.

[27] E. Acuna, F. Coaquira, and M. Gonzalez, "A comparison of feature selection procedures for classifier based on kernel density estimation," in *Proceedings of Computer, Communication and Control Technologies*, vol. 1, 2003, pp. 468–472.

[28] H. Liu and R. Setiono, "A probabilistic approach to feature selection a filter solution," in *International Conference on Machine Learning - ICML*, 1996, pp. 319–327.

[29] H. Stoppiglia, G. Dreyfus, R. Dubios, and Y. Oussar, "Ranking a random feature for variable and feature selection," in *Journal of Machine Research*, vol. 3, 2003, pp. 1399–1414.

[30] Z. Xu, I. King, M. R.-T. Lyu, and R. Jin, "Discriminative semi-supervised feature selection via manifold regularization," in *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 21, no. 7, 2010.

[31] P. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," in *IEEE Trans. Computers*, vol. 6, no. 9, September 1977, pp. 917–922.

[32] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," in *Pattern Recognition Letters*, vol. 15, 1994, pp. 1119–1125.

[33] J. Reunanen, "Overfitting in making comparisons between variable selection methods," in *JMLR*, vol. 3, 2003, pp. 1371–1382.

[34] P. Pudil, J. Novovicova, J. Kittler, and P. Paclik, "Adaptive floating search methods in feature selection," in *Pattern Recognition Letters*, vol. 20, 1999, pp. 1157–1163.

[35] Y. Sun, C. Babbs, and E. Delp, "A comparison of feature selection methods for the detection of breast cancers in mammograms: adaptive sequential floating search vs. genetic algorithm." *Conf Proc IEEE Eng Med Biol Soc*, vol. 6, 2005.

[36] S. Nakariyakul and D. P. Casasent, "An improvement on floating search algorithms for feature subset selection," in *Pattern Recognition*, vol. 42, 2009, pp. 1932–1940.

[37] S. Stearns, "On selecting features for pattern classifiers," in *Proceedings of the 3rd International Conference on Pattern Recognition*, 1976, pp. 71–75.

[38] D. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[39] A. Alexandridis, P. Patrinos, H. Sarimveis, and G. Tsekouras, "A two-stage evolutionary algorithm for variable selection in the development of rbf neural network models," in *Chemometrics and Intelligent Laboratory Systems*, vol. 75, 20025, pp. 149–162.

[40] D. Jouan-Rimbaud, D. L. Massart, R. Leardi, and O. E. D. Noord, "Genetic algorithms as a tool for wavenumber selection in multivariate calibration," in *Anal. Chem., Elsevier Science*, vol. 67.

[41] J. Yang and V. Honavar, "Feature subset selection using a genetic algorithm," in *IEEE Intelligent Systems and their Applications*, vol. 13, no. 2, 1998, pp. 44–49.

[42] W. Puch, E. Goodman, M. Pei, L. Chia-Shun, P. Hovland, and R. Enbody, "Further research on feature selection and classification using genetic algorithm," in *International Conference on Genetic Algorithm*, 1993, pp. 557–564.

[43] L. Eshelman, "The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination." in *In Foundations of Genetic Algorithms, G.J.E. Rawlins (Ed.)*. Morgan Kauffman, 1991.

[44] O. Cordon, S. Damas, and J. Santamaria, "Feature-based image registration by means of the chc evolutionary algorithm," in *Image and Vision Computing*, vol. 24, 2006, pp. 525–533.

[45] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen, "A methodology for feature selection using multiobjective genetic algorithms for handwritten digit sting recognition," in *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, no. 6, 2003, pp. 903–929.

[46] F. Ferri, P. Pudil, M. Hatef, and J. Kittler, "Comparative study of techniques for large-scale feature selection," in *Pattern Recognition in Practice*, 1994, pp. 403–413.

[47] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," in *Pattern Recognition*, vol. 33, no. 1, 2000, pp. 327–336.

[48] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *In Proc. IEEE Int'l. Conf. on Neural Networks, IV*, 1995, pp. 1942–1948.

[49] C.-J. Tu, L.-Y. Chuang, J.-Y. Chang, and C.-H. Yang, "Feature selection using pso-svm," in *International Journal of Computer Science*, vol. 33, 2006.

[50] E. Alba, J. Garca-Nieto, L. Jourdan, and E.-G. Talbi, "Gene selection in cancer classification using pso/svm and ga/svm hybrid algorithms," in *Evolutionary Computation*, 2007, pp. 284–290.

[51] P. A. Mundra and J. C. Rajapakse, "Svm-rfe with mrmr filter for gene selection," in *IEEE Transactions on Nanobioscience*, vol. 9, no. 1, 2010.

[52] O. Chapelle and S. S. Keerthi, "Multi-class feature selection with support vector machines," in *booktitle*, 2008.

[53] R. Archibald and G. Fann, "Feature selection and classification of hyperspectral images with support vector machines," in *IEEE GEOSCIENCE AND REMOTE SENSING LETTERS*, vol. 4, no. 4, October 2007.

[54] J. Neumann, C. Schnorr, and G. Steidl, "Combined svm-based feature selection and classification," in *Machine Learning*, vol. 61, 2005, pp. 129–150.

[55] R. Setiono and H. Liu, "Neural-network feature selector," in *IEEE Trans. Neural Netw*, vol. 8, no. 3, May 1997, pp. 654–662.

[56] E. Romero and J. M. Sopena, "Performing feature selection with multilayer perceptrons," in *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 19, no. 3, March 2008.

[57] D. J. Stracuzzi and P. E. Utgoff, "Randomized variable elimination," in *Journal of Machine Learning*, vol. 5, 2004, pp. 1331–1364.

[58] D. Wu, Z. Zhou, S. Feng, and Y. He, "Uninformation variable elimination and successive projections algorithm in mid-infrared spectra wavenumber selection," in *Image and Signal Processing*, 2009.

[59] V. Centner, D.-L. Massart, O. E. de Noord, S. de Jong, B. M. Vandeginste, and C. Sterna, "Elimination of uninformative variables for multivariate calibration," in *Anal. Chem*, vol. 68, 1996, pp. 3851–3858.

[60] B. K.Alsberg, A. M. Woodward, M. K. Winson, J. J. Rowl, and D. B. Kell, "Variable selection in wavelet regression models," in *Analytica Chimica Acta, Elsevier Science BV*, vol. 368, 1998, pp. 29–44.

[61] Y. Peng, Z. Xuefeng, Z. Jianyong, and X. Yunhong, "Lazy learner text categorization algorithm based on embedded feature selection," in *Journal of Systems Engineering and Electronics*, vol. 20, no. 3, 2009, pp. 651–659.

[62] E. Xing and R. Karp, "Cliff: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts," in *In 9th International Conference on Intelligence Systems for Molecular Biology*, 2001.

[63] P. Pudil, J. Novovicova, and J. Kittler, "Feature selection based on the approximation of class densities by finite mixtures of the special type," in *Pattern Recognition*, vol. 28, no. 9, 1995, pp. 1389–1398.

[64] P. Mitra, C. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," in *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 24, no. 3, March 2002.

[65] S. K. Pal, R. K. De, and J. Basak, "Unsupervised feature evaluation: A neuro-fuzzy approach," in *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 11, no. 2, March 2000.

[66] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[67] Z. Zhao and H. Liu, "Semi-supervised feature selection via spectral analysis," in *Proc. 7th SIAM Data Mining Conf. (SDM)*, 2007, pp. 641–646.

[68] A.-C. Haury, P. Gestraud, and J.-P. Vert, "The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures," in *PLoS ONE*, vol. 6, no. 12, December 2011, p. e28210.

[69] A. T, H. T, V. de Peer Y, D. P, and S. Y, "Robust biomarker identification for cancer diagnosis with ensemble feature selection methods," in *Bioinformatics*, vol. 26, no. 3, 2010, pp. 392–398.

[70] K. Dunne, P. Cunningham, and F. Azuaje, "Solutions to instability problems with sequential wrapper-based approaches to feature selection," Trinity College, Tech. Rep., 2002.

[71] A. Kalousis, J. Prados, and M. Hilario, "Stability of feature selection algorithms: A study on high dimensional spaces," in *Knowledge and Information Systems*, vol. 2, no. 1, May 2007, pp. 95–116.

[72] P. Somol and J. Novovicova, "Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality," in *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 32, no. 11, November 2010, pp. 1921–1939.

[73] F. Yang and K. Mao, "Robust feature selection for microarray data based on multicriterion fusion," in *IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS*, vol. 8, no. 4, 2011.

[74] T. Dietterich, "Machine learning research: Four current directions," in *Artificial Intelligence Magazine*, vol. 18, no. 4, 1997, pp. 97–136.

[75] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," in *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, 2011, pp. 1–27.

[76] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," in *Journal of Machine Learning Research*, vol. 5, 2004, pp. 101–141.

[77] S. Haykin, *Neural Networks: a comprehensive foundation*. Prentice Hall, 2008.

[78] E. Cinar and F. Sahin, "A study of recent classification algorithms and a novel approach for eeg data classification," in *IEEE 2010 International Conference on Systems, Man and Cybernetics*, October 2010.

[79] A. S. Loong, O. H. Choon, and L. H. Chin, "Criterion in selecting the clustering algorithm in radial basis functional link nets," in *WSEAS Transactions on Systems*, vol. 7, no. 11, 2008, pp. 1290–1299.

[80] J. V. Marcos, R. Hornero, and D. Alvarez, "Radial basis function classifiers to help in the diagnosis of the obstructive sleep apnoea syndrome from nocturnal oximetry," in *Medical and Biological Engineering and Computing*, vol. 46, no. 4, 2008, pp. 323–332.

[81] L. Hongyang and J. He, "The application of dynamic k-means clustering algorithm in the center selection of rbf neural networks," in *In Proc 3rd International Conference on Genetic and Evolutionary Computing*, vol. 177, no. 23, 2009, pp. 488–491.

[82] D. Tax, "One-class classication: Concept-learning in the absence of counter-examples," Ph.D. dissertation, University of Delft, 2001.

[83] J. J. Verbeek, N. Vlassis, and B. Krose, "Efficient greedy learning of gaussian mixture models," *Neural Computing*, vol. 5, no. 2, pp. 469–485, 2003.

[84] D. P. Filev, R. B. Chinnam, F. Tseng, and P. Baruah, "An industrial strength novelty detection frameowrk for autonomous equipment monitoring and diagnostics," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, 2010.

[85] A. Banerjee, P. Burlina, and C. Diehl, "A support vector method for anomaly detection in hyperspectral imagery," *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING*, vol. 44, no. 8, August 2006.

[86] N. R. Pal, K. Pal, and J. C. Bezdek, "A mixed c-means clustering model," in *Proceesing of the Sixth IEEE International Conference on Fuzzy Systems*, vol. 1, 1997, pp. 11–21.

[87] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri., "A review of process fault detection and diagnosis part i: Quantitative model-based methods," *Computers and Chemical Engineering*, vol. 27, pp. 293–311, 2003.

[88] A. Ypma, D. M. J. Tax, and R. P. W. Duin, "Robust machine fault detection with independent component analysis and support vector data description," in *PROCEEDINGS OF THE 1999 IEEE WORKSHOP ON NEURAL NETWORKS FOR SIGNAL PROCESSING*.

[89] M. Markou and S. Singh, "Novelty detection: a review part 1: statistical approaches," *Signal Processing*, vol. 83, pp. 2481–2497, December 2003.

[90] D. M. J. Tax, A. Ypma, and R. P. W. Duin, "Support vector data description applied to machine vibration analysis," 1999.

[91] D. Wang, P. W. Tse, W. Guo, and Q. Miao, "Support vector data description for fusion of multiple health indicators for enhancing gearbox fault diagnosis and prognosis," *Measurement Science and Technology*, vol. 22, no. 2, 2011.

[92] G. Chandrashekar and F. Sahin, "In-vivo fault prediction for rf generators using variable elimination and state-of-theart classifiers," in *2012 IEEE International Conference on Systems, Man, and Cybernetics October 14-17, 2012, COEX, Seoul, Korea*.

[93] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[94] D. Tax, "Ddtools, the data description toolbox for matlab," May 2012, version 1.9.1.