Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Statistics and Machine learning
2020 | LIU-IDA/STAT-A–20/032–SE

Clutter Detection in Radar Applications

Pedram Kasebzadeh

Supervisor : Hao chi kiang
Examiner : Oleg Sysoev

External supervisor : Peter J. Hessling

LINKÖPING
UNIVERSITY

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

Upphovsrätt

Copyright

Abstract

Radars have been used for detection purposes in safety applications (i.e., blind spot detection radar in cars) extensively. The existing detection methods, however, are not flawless. So far, the main focus of these methods is on detecting an object based on its reflectiveness.

In this thesis, the limitation of conventional methods are addressed, and alternative approaches are proposed. The main objective is to model/identify the noise with statistical and machine learning approaches as an alternative to conventional methods that focus on the object. The second objective is to improve the time efficiency of these methods.

The data for this thesis contains measurements collected from radars at ABB AB, Sweden. These measurements reflect the received signal strength. These radars are meant to be used in safety applications, such as in industrial environments. Thus, the trade-off between accuracy and complexity of the algorithms is crucial.

One way to ensure there is nothing but noise in the surveillance field of the radar is to model the noise only. A new input can then be compared to this model and be classified as noise or not noise (object). One-class classifiers can be employed to approach this problem as they only need noise for training; hence they have been one of the initial proposals in this thesis. Alternatively, binary classifiers are investigated to classify noise and object given a new input data. Moreover, a mathematical model for noise is computed using the Fourier series expansion. While the derived model holds useful information in itself, it can be used, e.g., for hypothesis testing purposes. Furthermore, to make the classification m ore time-efficient, dimension reduction methods are considered. Feature extraction has been performed for this purpose with the help of the derived noise model.

In order to evaluate the performance of the considered methods, three different datasets have been formed. In the first dataset, the collected raw data has been preprocessed and used as the input to the considered algorithms. The second dataset consists of the features extracted from the preprocessed data. Finally, in the last dataset, the derived mathematical noise model is used to calculate the features. The methods are then carried out on these datasets for comparison over time efficiency and accuracy.

The One-class SVM seems to be the best candidate for this application considering the trade-off between accuracy and time efficiency. There has been a significant improvement in the time efficiency of all the methods after using dimension reduction techniques. However, that has been achieved with the cost of a small accuracy degradation. The question of if it is worth to use dimensionality reduction is best answered by the application needs considering the trade-off between accuracy and time efficiency.

Acknowledgments

I would like to thank ABB Jokab AB for giving me the opportunity to work with them. I want to express my sincere thanks and appreciation to my supervisor, Peter Hessling, for his guidance, ideas, and support throughout an exciting project.

Then, my special thanks are extended to my supervisor from Linköping University, Hao Chi Kiang, for the support during the project and his great suggestions.

I would also like to express my appreciation to my examiner and course leader, Oleg Sysoev, for his valuable guidance and support not only throughout the thesis period but for the past two years.

Thanks also to my opponent Jiawei Wu for the useful comments provided at the revision meeting, and in general to all my classmates, students of Statistics and Machine Learning of Linköping University for the unforgettable memories during the Master's studies.

Finally, I must say thanks to my mother, my father, my sister, and last but not least, my brother for their love, unconditional support, and continuous encouragement throughout this Master's degree and my whole life.

Contents

List of Figures

List of Tables

1 Introduction

The detection reliability of a radar system is critical, especially when it comes to detecting humans. For instance, industrial applications where safety is mandatory, require reliable sensing of the presence of humans. This could be a scenario in a factory where operators are working closely with an automatic sharp blade. The radar shall guarantee there is no human in the field so the blade can operate safely, to avoid any coincidences.

When there is *nothing* within the surveillance volume, it could be denoted as *empty state*. *Empty state* would still reflect noise or clutter. For objects whose radar reflections are not strong enough, such as humans, extra care must be taken to distinguish them with noise-only scenarios. In short, radars should ideally guarantee an *empty state*.

This thesis provides a survey of some of the existing radar detection methods and highlights the drawbacks of the conventional approaches. Additionally, some approaches to the considered detection problem are introduced.

This thesis was conducted through a paid collaboration with ABB AB. In the process of data collection and throughout the whole thesis, no humans/animals were used, and no personal or critical information is involved.

1.1 Background

Traditionally, radar signal detection relies on Constant False Alarm Rate (CFAR) detectors, a common form of algorithm used in radar systems to detect targets [34]. The role of the CFAR processor is to determine a threshold, above which any value is considered as an object. If this threshold is too low, then more targets will be detected at the expense of an increased number of false alarms. Conversely, if the threshold is too high, the number of false alarms will be low, but fewer targets will be detected.

In most CFAR schemes, an estimation of the level of the noise floor around the cell under test (CUT) is done to determine the threshold level. This estimation can be done by calculating the average power level of cells around the CUT. However, cells immediately adjacent to the CUT (also known as guard cells) are not considered. The reason is to avoid corrupting this estimate with power from the CUT itself.

The estimate of the local power level is increased slightly to allow for the limited sample size, which forms the threshold. A signal is declared as an object in the CUT if it is both greater than all its adjacent cells and greater than the threshold. This simple approach is called a cell-averaging CFAR (CA-CFAR).

Figure 1.1 [41] illustrates the procedure of CA-CFAR. It shows how CUT exceeds the slightly raised threshold, which can be interpreted as an object.

There are other approaches to calculate averages of each side of the cut separately. These will then use the greatest-of (GO-CFAR) or least-of (LO-CFAR) these levels to define the local threshold.

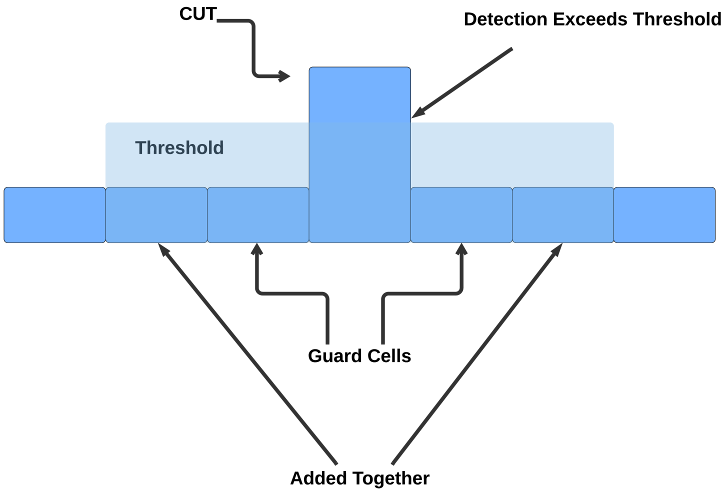


Figure 1.1: Constant false alarm rate. The two guard cells are ignored, and the rest are added and multiplied by a constant to establish a threshold.

Using a fixed threshold leads to a constant error, also known as a systematic error, for any situation which does not meet the assumptions made for setting the threshold [16]. Hatem et al. [24] compares different types of CFAR and shows how using one method is not reliable in all situations.

Jalil et al. [2] investigates two scenarios where CFAR might not perform properly. The first scenario is when the clutter power received within a signal suddenly changes. Regions, where this clutter power transition happens, are known as clutter edges. The presence of clutter edges result in performance decay; for instance, it might increase false alarms.

The second situation happens when there is more than one object present. This will cause a raise of the threshold level leading to missing the weak echoes (received signals) of distant targets by the primary object, also known as masking effect.

Jalil et al. in [2], then investigate different CFAR methods such as CA-CFAR, GOCA-CFAR, and Smallest of Cell Average CFAR (SOCA-CFAR) in challenging situations, such as in clutter edges or with multiple objects. These methods will not be discussed here as they are shown to be unreliable by Jalil.

Chen et al. [8] investigate an adaptive CFAR detection for clutter-edge using Bayesian inference, and show the threshold setting to maintain a specific false alarm rate is fairly insensitive.

The goal of this work is to improve the reliability of radars detection. This would be done with the help of statistical methods and machine learning algorithms. So far limitation of conventional methods has been introduced. Now some of the alternatives to the CFAR are discussed.

The process of deciding whether or not a measurement represents an empty state could be formulated as statistical hypothesis testing [46]. Given a set of observations, statistical hypothesis testing can be employed to choose between the null hypothesis (H_0), which is that no object exists, and observed values are noise contributions, and the alternative (H_1) is that there is an object in the surveillance area.

This could be done with the help of various approaches. Santoso et al. [22] investigate multiple machine learning algorithms, such as support vector machine (SVM) and neural networks, for detection purposes, where SVM shows promising results. Another approach is considering the similarity of the distribution of train and test datasets. Based on the similarity of these distributions, noise and object could be distinguished, as any significant deviation from the noise distribution (or noise model) could be considered as something (object) within the surveillance volume. This could be a potential slow/stop order depending on the application in hand. Filzmoser [42] defines a cut-off to identify outliers by a measure of the deviation of the empirical distribution function of the Mahalanobis distance from the theoretical distribution of the data.

Additionally, one-class classification (OCC) algorithms are investigated. There is a rich literature on OCC, Khan [47] claims that OCC algorithms aim to build classification models when one class is present and well defined, but other classes are not. This could be extended to the problem referred to in this thesis. It is possible to collect data for noise and consider any other type of input as not noise, or negative as khan puts it.

The term one-class classification originates from Moya [36], but other terms have been used. Ritter et al. [23], refer to it as outlier detection, and Bishop [11] uses the term novelty detection. The different terms originate from the different applications to which one-class classification can be applied.

The One-class approach has been successfully applied to various problems, [33], [12], [1], however, to the best of author knowledge, it has not been used to have noise as positive class so far.

Bartkowiak [3] divides the system behavior to normal and abnormal and tries to find the space in the data where the bounds permit to distinguish between the normal and abnormal items.

Wenzhu et al. study the OCC in [51] and illustrate the advantages of using OCC by experiments. The authors then compare the learning and generalization ability of OCC algorithms, as well as classification accuracy and algorithm complexity, to find the most efficient method.

One of the most important factors in a safety application is processing time. Any delay in detection could be crucial. The data used in this thesis has high dimensionality (will be explained in Chapter 2); Hence, using features instead of raw data is considered in order to improve time efficiency. Some widely known features like signal variance and signal energy were extracted [32].

High dimensionality of the data and the need for fast response in a safety application was the motivation to investigate feature extraction.

1.2 Objective

This thesis is set to investigate the alternatives to conventional methods for detection (CFAR). So far, the focus of detection has been on detecting the object. The goal here is to explorer the capabilities of using a noise model for detection purposes instead of looking for an object. In other words, by learning the characteristics of noise, an empty state might be guaranteed.

The main objectives of this thesis are:

1. Noise modeling using statistical and machine learning approaches.
2. Investigating feature extraction to improve time efficiency.
3. Performance evaluation of different machine learning algorithms.

The rest of this thesis is organized as follows. Chapter 2 describes the data and data collection processes as well as preprocessing steps. Chapter 3 investigates different methods and their implementation. Results are presented in Chapter 4 and then discussed in Chapter 5 followed by a conclusion in Chapter 6.

2 Data

The data for this thesis was generated by radars manufactured by Texas Instruments (TI). TI develops multi-purpose sensors that are used in industry, medicine, and autonomous driving. In this thesis, industrial millimeter-wave (mmWave) sensors (IWR) are used. IWR mmWave solutions detect range, velocity, and angle of objects with unique accuracy. Samples are generated in a designed empty space, with a radar reflector, and a human as an object to form a diverse dataset. Data is then processed with multiple functions, which will be discussed furthermore.

This chapter starts by introducing the module used to collect the data in Section 2.1 then moves to explain some basics of the signals and how they are processed in Section 2.2. Finally, a short description of the dataset and sampling process is presented in Section 2.3.

2.1 Module

The modules used were equipped with IWR6843 intelligent mmWave sensors developed by TI, which operates in the spectrum between 60GHz and 64GHz. The IWR6843 sensor is an integrated single-chip mmWave sensor based on frequency modulated continuous wave (FMCW) radar technology [14] .

The IWR6843 sensor is an ideal solution for self-monitored, low power, ultra-accurate radar systems in the industrial space [45].

Each radar is equipped with four receivers and three transmitters, which will result in twelve *channels* of data in total. Each *channel* is a combination of one transmitter and one receiver.

There are several other parts in the module such as, mixers (a component that combines two signals to create a new signal with a new frequency), amplifiers, converters, and filters, which will not be discussed as they are out of the scope of this thesis.

2.2 Signal

Figure 2.1, summarizes the process in a FMCW radar. As the first step, Synthesizer generates a sinusoid signal whose frequency will increase with time, also known as a chirp. TX antenna

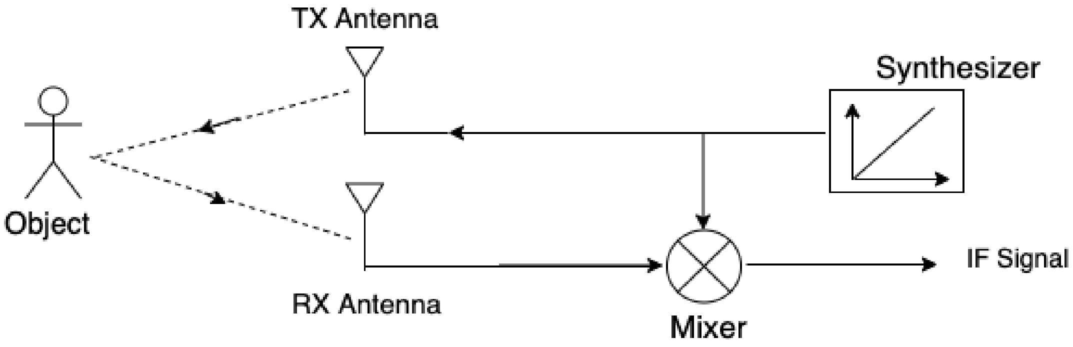


Figure 2.1: Summary of how a radar works

then transmits this chirp (x_T). The signal reflected (x_R) from the object is then received by RX antenna. These signals can be formulated as:

$$\begin{aligned} x_T(t) &= sin(\omega_T t + \phi_T) \\ x_R(t) &= sin(\omega_R t + \phi_R), \end{aligned} \tag{2.1}$$

where ω denotes frequency, ϕ denotes phase while t denotes time.

These two chirps are passed to the mixer, as shown in Figure 2.1. A mixer contains three ports, two inputs, and one output. For two sinusoids x_T and x_R inputs in Equation 2.1, the output is:

$$x_{out}(t) = sin[(\omega_T - \omega_R)t + (\phi_T - \phi_R)]. \tag{2.2}$$

The mixture of x_T and x_R signals is called an Intermediate Frequency (IF) signal, which is the difference of the transmitted and received frequency at a certain time [48]. The IF signal obtained from the radar is in the form of complex numbers.

A Fast Fourier Transform (FFT) is then performed on the IF signal. The location of each peak in the frequency spectrum corresponds to the distance of objects [44].

Fast Fourier Transform

Fourier transforms are an essential part of FMCW radar signal processing.

A Fourier transform converts a time-domain signal into a frequency domain, so a better analysis is possible.

Initially Discrete Fourier Transform (DFT) were used, which is defined by:

$$x_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N} \quad where \;\; k = 0, ..., N-1, \tag{2.3}$$

where $x_0, ..., x_{N-1}$ are complex numbers generated by the radar as shown in Equation 2.2, and i is the imaginary unit. There are N, x_k outputs, and each requires the sum of N terms. An N-point transformation by this method takes time up to N^2 proportionally; hence they are computationally expensive.

$O(N^2)$ operations requirement for evaluating Equation 2.3, resulted in the development of computer algorithms, called the Fast Fourier Transform (FFT). FFT denotes to any algorithm that computes the same results with $O(NlogN)$ operations. The most common FFT is the Cooley–Tukey algorithm [10].

For this purpose, a simple available function in MATLAB is used. The output of the FFT function in MATLAB based on distance is called a range-FFT.

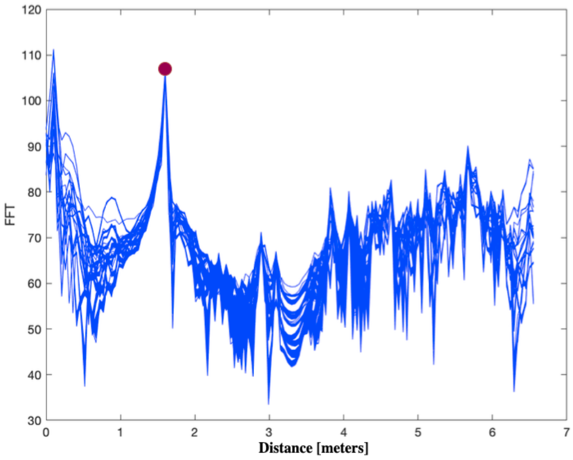


Figure 2.2: FFT over range plot. The red dot in the peak represents the object in 1.5 meters.

Figure 2.2 illustrates range-FFT, for 1500 observations and 128 features (distances). As objects have stronger reflections than noise, in case of presence of an object in the radar field of view, a peak in the FFT value is expected. Figure 2.2 shows a peak at approximately 1.5 meters; this yields the presence of an object in 1.5 meters from the radar supposedly, which indeed was the situation of data collection for this plot. This figure also reveals that there might be a potential pattern in the range-FFT of an object at a specific distance, considering this figure is plotted using 1500 observations with the object was in the same location.

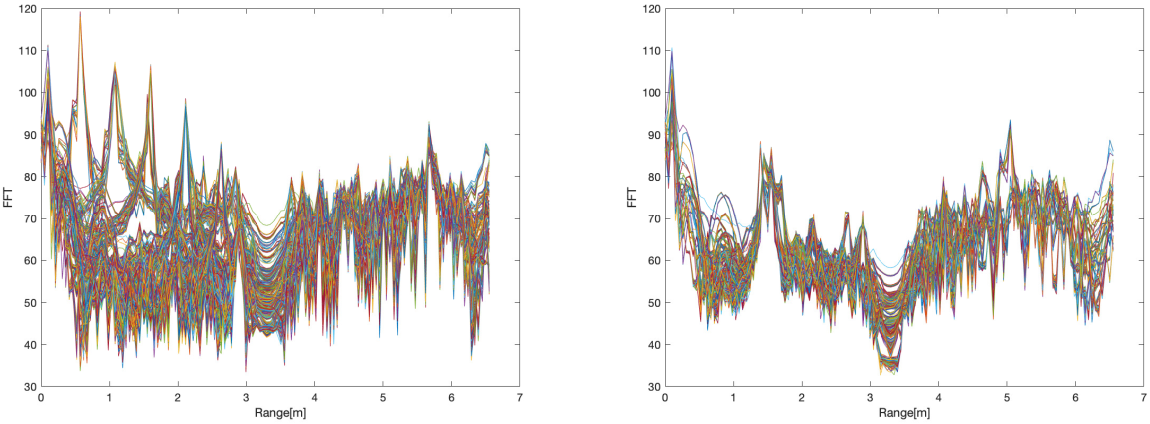
The data collected for this thesis was produced by taking multiple samples with and without an object, and then the output of the radars was processed to range-FFT.

2.3 Dataset

The data collected for this thesis was first collected using a reflector, which gave unrealistic results. In other words, since the reflector provides ideal (strong) reflections compare to an ordinary object (i.e., a human), it was easy for implemented algorithms to distinguish between noise and object, and they reached perfect accuracy (100%). This is not the case in real world applications. Therefore, a new dataset was collected with a human to have a more realistic and challenging dataset.

Figure 2.3 represents two plots, one with a human (Figure 2.3b) and one with a reflector (Figure 2.3a) as the object to compare the two sets of collected data. As the figure illustrates, reflections captured while using a reflector as an object reveal relatively higher FFT compared to when a human was used as an object. Furthermore, these two collections are combined to one dataset to have the most diversity possible.

The sampling process was done in a setting with one module, as explained in Section 2.1. This process involved placing a reflector at different distances (d) from radar and capturing the reflected signal for each distance, where $d = 0.5, 1, 1.5, \ldots, 3$. The number of observations collected for each distance was 1500, which sums up to a total of 9000 observations with a reflector as an object. Later on, a human was used instead of a reflector, and the reflections were captured the same way. Finally, 9000 observations were obtained when there was nothing in the radar field of view, to represent the noise. The data of these scenarios were then combined and processed through rang-FFT, as explained in Section 2.2. Finally, a combination of these scenarios range-FFT formed the initial dataset for this thesis, with 27000 observations and 128 features for each observation. The dataset also had an additional column to label the class of each observation as *object* (reflector or human), and *noise*.



(a) Reflections of radar reflector located in different ranges

(b) Reflections of a human located in 1.5 meters

Figure 2.3: Represents two plots with reflector and human as object.

Lastly, two more datasets are conducted. One using feature extraction (which is explained in detail in Section 3.7), and the other one is features calculated with the mathematical noise model introduced in Section 3.6.

3 Method

This chapter covers the scientific methods used in the thesis. Section 3.1 is dedicated to the theoretical background. The following Section 3.2 describes the methodology used in the thesis. Section 3.3 investigates the methods of binary classification while one-class classifiers are discussed in 3.4. Section 3.5 is devoted to non-parametric hypothesis testing method. Mathematical noise model is presented in Section 3.6 and Section 3.7 is devoted to feature extraction. Finally this chapter ends with evaluation methods presented in Section 3.8.

3.1 Theoretical Background

Classical statistics and *machine learning* (ML) are two main approaches used for statistical modeling to conclude data [9]. *Classical modeling* assumes that the data are generated by a given stochastic data model, and uses a variety of functions to model relations between dependent and independent variables. Mathematical models are then generated. The goal is to find the properties of the underlying distribution from which the data is generated and obtain meaningful statistical inference.

Machine learning goal is to develop algorithms that learn from the examples to predict and identify (classify) future unknown data, without relying on any formal statistical assumptions. The learner then analyzes the *training data* to find a pattern between different features. The statistical model of the data is then produced. The model is then used for prediction, classification, etc.

In this thesis, classical statistics are investigated as well as ML methods to see if it is possible to increase the reliability of conventional techniques for detections, which are mainly thresholding base. As discussed in Chapter 1, the goal of this thesis is to provide an approach to guarantee an *empty state* in the sense that there is nothing in the surveillance area. This can be formulated as a one-class classification problem (which is explained in detail in Section 3.4).

The classification problem is one of the oldest problems in machine learning. The goal is to decide which category a new data point belongs to based on its features. Binary classification algorithms assign an unknown object into one of the pre-defined categories. The limitation of these algorithms is that for a new unknown signal, which does not belong to any of the pre-defined classes in the training phase, these algorithms might fail to act properly.

3.2 Methodology

In this thesis, the limitations of binary classification are very critical. Considering each object has a different reflection pattern based on its reflectiveness and distance to the radar, it is challenging to have all the possible classes in our dataset. There are countless different objects which could be placed in many different positions in the radar field of view. For instance, the signal received by radar for a metal object in 3 meters, is fundamentally different from a wooden object placed in 1 meter from the radar. Hence, a new input that represents an object with different patterns regarding what is available in the training dataset could be miss-classified as noise. This is not acceptable in a safety detection application.

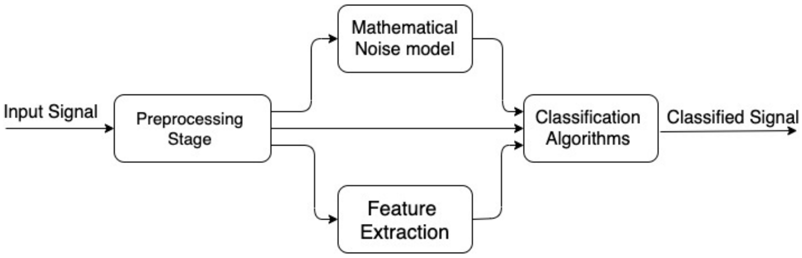


Figure 3.1: An overview of the thesis.

Figure 3.1 shows the workflow of this thesis. As it appears in the block diagram, after prepossessing the signal (which was explained in Chapter 2), a mathematical noise model is obtained using the Fourier series. Furthermore, feature extraction is performed in order to reduce dimensionality. Finally, different classification algorithms are carried out on preprocessed data (referred to as raw data from here on), features extracted, and mathematical noise model.

Classification algorithms used in this thesis are presented first. These methods can be categorized as:

- Binary classification.
- One-class classification (OCC).
- None-parametric Hypothesis testing.

Each method is explained in detail further on. Then the mathematical noise model is presented in Section 3.6. The feature extraction is defined in Section 3.7. The mathematical noise model is also used for feature extraction, which forms another dataset that is used for comparison purposes later.

These methods are then evaluated based on their accuracy, training and testing time, *safety*, and *availability.* Safety refers to the number of false-negative classified observations. In this application, false negative refers to when the signal is generated from an object; however, the method is classifying it as an empty state (noise). This could result in a fatal accident in a safety application and hence is not acceptable. Availability refers to false positive, which happens when the method is misclassified and an actual empty state for an object. This results in an unnecessary interruption (slow/stop) in the system functionality. The evaluation methods are discussed in more detail in Section 3.8.

There are other methods presented in classification, for instance, Chamidah [17], presents a hybrid K-Means and support vector machine for Fetal state classification.

3.3 Binary Classification

Binary classification refers to classification algorithms that assign an input to one of two available categories, considering its attributes. These categories can be denoted as positive and negative. As mentioned in Chapter 1, the goal of this thesis is to model the noise, hence positive will refer to noise and negative in this thesis is used to address not noise (object).

The goal of binary classification is to learn a function $g(\boldsymbol{x})$ which minimizes the following misclassification probability [37]:

$$P\{yg(\boldsymbol{x}) < 0\}, \tag{3.1}$$

where $\boldsymbol{x}$ is the new input while y denotes its class label. When positive $y = +1$ and for negative class label $y = -1$. There are many popular binary classification methods. This thesis investigates two widely used binary classification methods K-Nearest Neighbor (KNN) and Support Vector Machine (SVM).

3.3.1 K-Nearest Neighbor

The K-Nearest Neighbor (KNN) is a simple, non-parametric classifier that computes the distance (e.g., Euclidean distance) between a new (unseen) input and the training data points; the output is a class membership. Selecting the K training points with the closest distance to the input, the algorithm computes the plurality of members of each class. The input is then classified based on the most common class in its K neighboring points. K denotes the number of neighbors to take into account, e.g., $K = 1$ means that a new input would have the same class as its closest neighbor.

A useful technique for improving KNN can be to assign weights to the classes in the sense that nearer neighbors get higher weights compared to the neighbors further away, hence have a higher contribution. This is known as a weighted-KNN. A common way to do so is giving each neighbor a weight of $\frac{1}{d}$, where d is the distance to the neighbor. There are different approaches to calculate the distance between two points in KNN. Rosa [5] achieves good results using an adaptive Mahalanobis distance. However, this method requires two parameters to be set. In this thesis, a more popular Euclidean distance is used, which can be shown as:

$$d(p,q) = \sqrt{\sum_{l=1}^{L}(q_l - p_l)^2}, \quad l = 1,2,\ldots,L, \tag{3.2}$$

where p and q are two points, and L denotes the number of dimensions (we can also say features in machine learning terms).

The class will be assigned to the new input signal based on its membership probability. For a new input in the test set, $\boldsymbol{y_t}$, membership probability is estimated as:

$$Class_t = P(\boldsymbol{y_t} = c|K) = \frac{1}{K}\sum_{i=1}^{K} F(\boldsymbol{y}_{k_i} = c), \tag{3.3}$$

where $\boldsymbol{y_k}$ are the K-nearest points to $\boldsymbol{y_t}$ in the training set. c is the possible classes, $c \in \{0,1\}$ or $c \in \{object, noise\}$ in this thesis case and $F(v)$ is the indicator function defined as

$$F(v) = \begin{cases} 1 & if\ v\ is\ true \\ 0 & otherwise \end{cases}. \tag{3.4}$$

KNN was implemented using kknn function in kknn package in R [30].

3.3.2 Support Vector Machine

Support Vector Machine is considered to be a state-of-the-art-method in classification and regression. In this thesis, features of SVMs when applied to binary classification are investigated. SVMs is a supervised ML algorithm that introduces the concept of margin as a measure of the distance between the separation boundaries for each class [31]. The observations on the margin are known as support vectors.

SVM attempts to find the separation hyperplane, which maximizes this margin (i.e., maximizes the distance between the closest data points at the edge of each class). SVM represents observation points in space, mapped in such a way that observations of the separate categories are divided by a clear hyperplane that is as wide as possible.

New observations are then mapped into that same space and classified based on the side of the hyperplane on which they lie. This is shown in Figure 3.2.

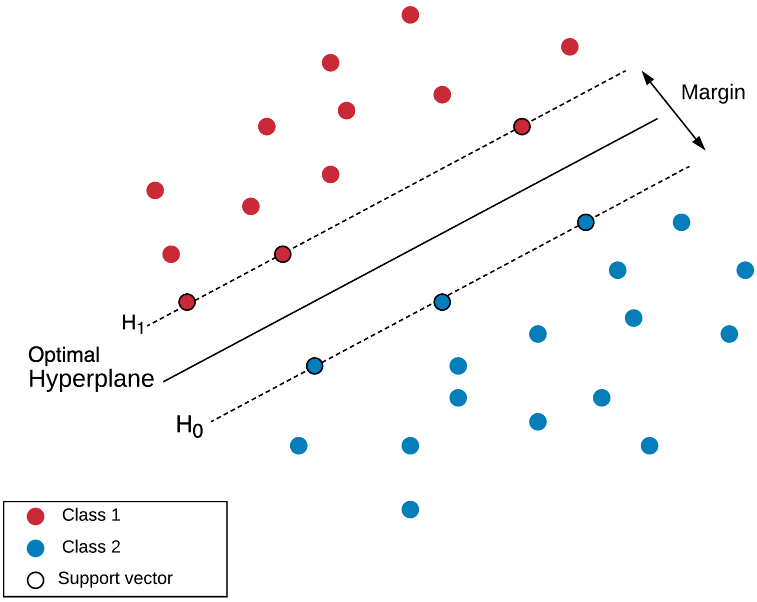


Figure 3.2: Mapped observations divided by a hyperplane

Assuming,

$$(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n), \tag{3.5}$$

denotes a set of labeled training data, where $y_i \in \{-1, 1\}$, with $i = 1, \ldots, n$. It is considered to be linearly separable if there exists a vector $\boldsymbol{w}$ and a scalar b such that:

$$\begin{aligned} \boldsymbol{w}\boldsymbol{x}_i + b &\geqslant 1 \quad if \quad y_i = 1, \\ \boldsymbol{w}\boldsymbol{x}_i + b &\leqslant -1 \quad if \quad y_i = -1. \end{aligned} \tag{3.6}$$

Equation 3.6 is valid for all the training points in Equation 3.5. If we rewrite Equation 3.6 as

$$y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1, \tag{3.7}$$

then the optimal hyperplane

$$\boldsymbol{w}\boldsymbol{x}_i + b = 0, \tag{3.8}$$

would be the one that separates the training data with the maximal distance between the projections of two different classes. To calculate the maximum distance between two hyperplanes, m (shown as Margin in Figure 3.2) should be obtained.

Assuming $\boldsymbol{u} = \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$ is the unit vector of $\boldsymbol{w}$, and $\|\boldsymbol{w}\|$ is the Euclidean norm of $\boldsymbol{w}$; $\boldsymbol{g} = m\boldsymbol{u}$ is a vector which is perpendicular to both hyperplanes, while m is the distance between the two hyperplanes. If $\boldsymbol{x}_0$ denotes a point on hyperplane H_0, then a point $\boldsymbol{z}_0$ on H_1 can be shown as $\boldsymbol{z}_0 = \boldsymbol{x}_0 + \boldsymbol{g}$. Since $\boldsymbol{z}_0$ is a point on H_1 support vector:

$$\boldsymbol{w}\ \boldsymbol{z}_0 + b = 1, \tag{3.9}$$

as assumed before $\boldsymbol{z}_0 = \boldsymbol{x}_0 + \boldsymbol{g}$, and $\boldsymbol{g} = m\boldsymbol{u}$ hence,

$$\boldsymbol{w}\ (\boldsymbol{x}_0 + m\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}) + b = 1, \tag{3.10}$$

which is equal to:

$$\boldsymbol{w}\ \boldsymbol{x}_0 + m\frac{\boldsymbol{w}\boldsymbol{w}}{\|w\|} + b = 1. \tag{3.11}$$

Equation 3.11 can be written as:

$$\begin{aligned} \boldsymbol{w}\boldsymbol{x}_0 + m\frac{\|\boldsymbol{w}\|^2}{\|\boldsymbol{w}\|} + b = 1, \\ \boldsymbol{w}\boldsymbol{x}_0 + m\|\boldsymbol{w}\| + b = 1, \\ \boldsymbol{w}\boldsymbol{x}_0 + b = 1 - m\|\boldsymbol{w}\|\,. \end{aligned} \tag{3.12}$$

As mentioned $\boldsymbol{x}_0$ is assumed to be on H_0 hyperplane, hence, $\boldsymbol{w}\boldsymbol{x} + b = -1$, so 3.12 will become:

$$\begin{aligned} &-1 = 1 - m\|\boldsymbol{w}\|\,, \\ &m\|\boldsymbol{w}\| = 2, \\ &m = \frac{2}{\|\boldsymbol{w}\|}. \end{aligned} \tag{3.13}$$

Hence the distance between to hyperplane is shown as $m = \frac{2}{\|\boldsymbol{w}\|}$; this shows the maximized distance is obtained by minimizing $\|\boldsymbol{w}\|$. The optimization problem to solve is presented as:

$$\begin{aligned} &min\frac{\|\boldsymbol{w}\|}{2}, \\ &such\ that:\ \ y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1, \\ &(for\ i = 1, \ldots, n). \end{aligned} \tag{3.14}$$

For computational efficiency equation 3.14 is then shown as [4]:

$$\begin{aligned} &min\frac{\|\boldsymbol{w}\|^2}{2}, \\ &such\ that:\ \ y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1. \end{aligned} \tag{3.15}$$

However, this is For separation of data without any errors [13], which is referred to as a *hard* margin. Figure 3.2 is showing a hard margin SVM. If it is not possible to separate the data without error, then a $soft$ margin hyperplane will be used to minimize the error. Soft margin refers to when there are data points beyond their class hyperplane, for instance, between H_0 and H_1, as shown in Figure 3.3.

Slack variables denoted as ξ_i in Figure 3.3, measure the amount of error of the constraints, considering the misclassified data point. If ξ_i, denotes a slack variable to relax the constraints in the optimization problem (Equation 3.15). Then the constraints would only have to satisfy:

$$\begin{aligned} &y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i, \\ &for\ \xi_i \geqslant 0. \end{aligned} \tag{3.16}$$

.

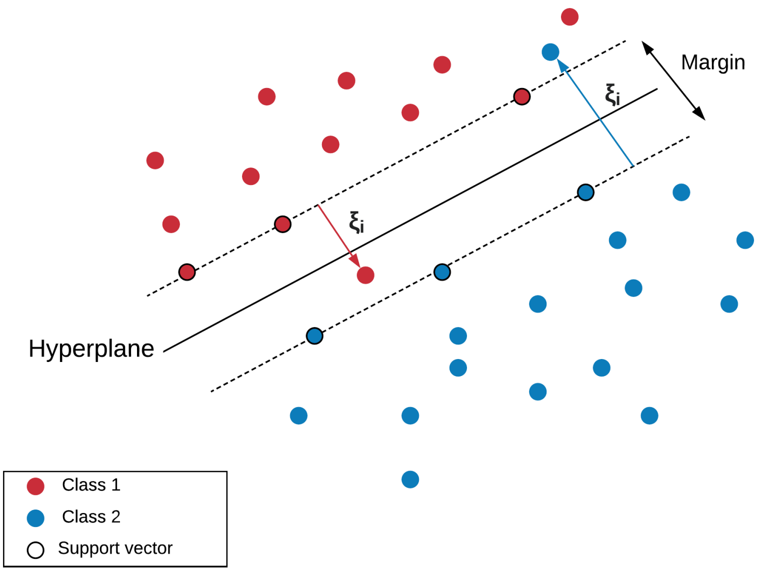


Figure 3.3: Soft margin SVM

If these slack variables are too large, the relaxed constraints would be trivially satisfied, and hence one has to add safeguards against such behavior. One way to do so is to add the regularization parameter to the objective function [25]:

$$\begin{aligned} &\min \frac{||\boldsymbol{w}||^2}{2} + C \sum_{i=1}^{l} \xi_i, \\ &such\ that:\ y_i(\boldsymbol{w}\boldsymbol{x}_i + b) \geqslant 1 - \xi_i, \\ &with\ \xi_i \geqslant 0, \end{aligned} \tag{3.17}$$

where $C > 0$ is the regularization parameter[18]. The regularization parameter relaxes the constraints and allows some flexibility to the number of errors made by the hyperplane margin.

To perform the binary SVM, e1071 package [38] in R was used, where ν parameter is introduced as a rate of C and ξ

3.4 One Class Classifier

One-class classification (OCC) includes two classes defined as [47]:

1. *Positive* class (noise) is referred to the only class present in the training dataset.

2. *Negative* class (not noise/object) which either has very few or no samples in the training dataset.

OCC defines a classification boundary around the positive class, and will consider anything out of this boundary as negative, and tries to minimize the chance of miss-classification. Since in the training data, we only have positive class, only one side of the boundary can be determined, which makes OCC more challenging compared to multi-class classification. The properties of the boundary, which indicates how tight the boundary around the positive class should be, are critical as they affect the miss-classification rate. This could result in false positives, which are not tolerable. False-positive means there is an object in the radar surveillance area; however, the signal is estimated as an empty state. In other words, a safe environment is declared falsely, which can be fatal. This will happen in case a negative class data point is too close to the positive class; hence this negative class ends up being classified as a positive class.

One of the challenges of OCC is obtaining *clean data*, which refers to a training dataset of only one class. Eskin [21] introduces a technique to overcome this challenge. He develops a mixture model to explain the presence of outliers in the training data. However, the data used in this thesis was collected in an observed process; hence it is *clean*.

There are many different approaches to implementing OCC. Wenzhu [51] categorized, the OCC approaches into three groups: $Density - based$, $Reconstruction - based$ and $Boundary - based$.

1. $Density - based$ method estimates the distribution of the positive class and sets a threshold based on training data to form an acceptance domain for testing.

2. $Reconstruction - based$ method uses training data to establishes a sample generation model. Furthermore, test samples can be considered as a result of the generated model. K-center and K-means [35] fall into this category.

3. $Boundary - based$ optimizes the boundary description of the positive samples based on prior knowledge and then classifies the test samples based on the described boundaries. K-nearest neighbor method [6] and one-class support vector machine(OCSVM) are considered as boundary-based OCC.

Support vector based approaches are good for classification, despite their offline training time. In this thesis, a boundary-based one-class SVM is investigated. The performances of one class classifiers are then compared with binary SVM, KNN, and a non-parametric method.

3.4.1 One Class SVM (OCSVM)

OCSVM is a supervised learning technique. It is an extension of SVM. To identify negative class, OCSVM estimates a distribution that encompasses positive class observations and will consider any observation far from this distribution as a negative class.

Support vector data description (SVDD) proposed by Tax and Duin [19], is an enhancement to OCSVM. This model aims at finding a spherically shaped boundary around a data set, which will not be discussed in this thesis.

OCSVM algorithm maps the positive class into a feature space using an appropriate kernel function (radial kernel in this thesis (Equation 3.19), chosen by cross-validation), and then attempts to find the hyper-plane that separates the mapped data from the origin with maximum margin. let $\boldsymbol{\Phi} : \boldsymbol{X} \rightarrow \boldsymbol{\Psi}$ be the kernel map which transforms training data ($\boldsymbol{X}$) to another space. To separate the positive class from the origin the following equation which is obtained from Equation 3.17 needs to be solved :

$$\begin{aligned} & \min \frac{||\boldsymbol{w}||^2}{2} + \frac{1}{\nu l} \sum_{i=1}^{l} \epsilon_i - b, \\ & where: \ \nu \in (0,1], \epsilon_i \geqslant 0, \ \ and, \\ & (w\Phi(x_i)) \geqslant b - \epsilon_i, \ \forall i = 1, \ldots, l, \end{aligned} \tag{3.18}$$

where ϵ_i are nonzero slack variables which allow the procedure to incur in errors.The parameter ν sets an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors [49].

Gaussian Radial Base kernel chosen is presented as:

$$\Phi(\boldsymbol{x}_i, \boldsymbol{x}_j) = exp(-\frac{1}{2\sigma}||\boldsymbol{x}_i - \boldsymbol{x}_j||^2), \tag{3.19}$$

where σ is a kernel parameter and $||\boldsymbol{x}_i - \boldsymbol{x}_j||^2$ is the dissimilarity measure, which Euclidean distance was used in this thesis.

Heller et al. [29] compares three different kernels for OCSVM and shows the "optimal kernel" heavily depends on the dataset by comparing the results. Hence, a cross-validation method was performed to choose the kernel in this thesis.

The drawbacks of the OCSVM are that it requires to solve a quadratic problem, which means the training phase could be time-consuming. The variance of the training data in each feature direction is not considered either in this approach.

3.4.2 One class Mahalanobis distance

Drawbacks of popular approaches such as SVDD and Kernel Principal Component Analysis (KPCA) for one class classification were the motivation to pursue a new approach. Nader et al. [40] discusses the limitation of each method and proposes a Mahalanobis distance based one-class classification.

Mahalanobis distance measures the distance between a point and a distribution. As a given point p moves further away from the mean of an assumed distribution the Mahalanobis distance increases. Mahalanobis distance takes the covariance in each feature direction and the different scaling of the coordinate axes into account [40]. Assuming $x_1, \dots, x_N \sim N_p(\mu, \Sigma)$ is a multivariate normal i.i.d. (a sequence of independent, identically distributed random variables) sample, where μ is the mean and Σ is the variance, Mahalanobis distance is computed as:

$$d_i^2(\mu, S) = (x_i - \mu)^T S^{-1}(x_i - \mu), \tag{3.20}$$

where $i = 1, \dots, N$ and N is the number of observations. S is the covariance matrix given as:

$$S = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T. \tag{3.21}$$

After calculating the Mahalanobis distance between each training sample x_i and the mean μ, testing sample are compared against the $95th$ percentile of a $F - distribution$ obtained by the degrees of freedom of the data. Hardin [26] shows that given S and x_i are independent:

$$\frac{n-p}{(n-1)p} d_i^2(\mu, S) \sim F(p, n-p) \tag{3.22}$$

where p is the number of the features and N is the number of observation. $F(p, n-p)$ denotes Snedecor's F distribution or the Fisher–Snedecor distribution. It is a continuous probability distribution and was applied in this case, since independent data distances have an F distribution [26]. For a random variable ζ this probability density function (PDF) is defined as:

$$f(\zeta : d_1, d_2) = \frac{\sqrt{\frac{(d_1\zeta)^{d_1} d_2^{d_2}}{(d_1\zeta + d_2)^{d_1+d_2}}}}{\zeta B\left(\frac{d_1}{2}, \frac{d_2}{2}\right)}, \tag{3.23}$$

where d_1 and d_2 are the degrees of freedom, and B is the beta distribution [26].

The independence of data points is investigated with the help of a *normality test*. Furthermore, *power transforms* were acquired to make the noise distribution more 'normal looking' if required. These steps are explained in section 3.4.2.

Algorithm 1 shows the procedure of Mahalanobis distance one class classification.

Algorithm 1: Mahalanobis distance one class classification

```
Data: Training Dataset $D_{tr}$, Testing Dataset $D_{te}$
Result: List of class labels for test dataset $C_{te}$
initialization;
$N$ = Number of observations($D_{tr}$);
$p$ = Number of samples in each observation($D_{tr}$);
instructions;
for $i \in D_{te}$ do
    $D_i$ = Mahalanobis distance(i,mean($D_{tr}$),covariance($D_{tr}$));
    if $D_i \in F_{0.95}(df_1 = p, df_2 = N - p)$ then
        $C_{te}$ =Noise;
    else
        $C_{te}$ = Object;
    end
end
return $C_{te}$
```

Normality

Normality test is used to determine if a random variable (in this case, noise) is well-modeled by a normal distribution and to compute how likely it is for a random variable underlying the dataset to be normally distributed.

One of the most popular normality test is the Shapiro–Wilk test which is shown as:

$$W = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \tag{3.24}$$

where $x_{(i)}$ is the ith-smallest number in the sample, $\bar{x}$ is the sample mean, and a_i are the coefficients.

If the data is not normally distributed enough, one approach is to make the data distribution more 'normal looking' by **transformations** of the data, which is a reexpression of the data in different units. A suitable family of transformations for our purpose are the *power transformations*. For a given observation x, x^{λ} denotes the power transformation of x with the power of λ. For instance, for $\lambda = -1$ and $\lambda = 1/4$, x would be x^{-1} and $\sqrt[4]{x}$, respectively. Box-cox [27], considers modified family of power transformations. For $x > 0$ it will perform the following transformation :

$$x^{\lambda} = \begin{cases} \frac{x^{\lambda}-1}{\lambda} & \lambda \neq 0, \\ \ln x & \lambda = 0. \end{cases} \tag{3.25}$$

Johnson [27] shows for an $n \times d$ dimensional dataset, d, λs are required to perform power transformation. These λs, denote as, $\lambda_1, \ldots, \lambda_d$ maximize the following equation [27]:

$$l(\lambda_d) = \frac{-n}{2} ln \left[\frac{1}{n} \sum_{i=1}^{n} (x_{id}^{\lambda_d} - \mu_d^{\lambda_d}) \right] + (\lambda_d - 1) \sum_{i=1}^{n} \ln x_{id}, \tag{3.26}$$

where $\mu_d = \frac{1}{N} \sum_{i=1}^{N} x_{id}^{\lambda_d}$. Furthermore any new input is transformed with obtained λs before validation with any method.

3.5 None-parametric Hypothesis testing

A none-parametric hypothesis testing without any prior assumption on the underlying distribution is carried out in this section, with the use of the Kruskal-Wallis test. This test offers a distribution-free alternative to the one-way analysis of variance (ANOVA). One-way ANOVA is a parametric method for comparing k independent samples. The null hypothesis (H_0), that the data belongs to class noise, is accepted if the $p - value$ obtained from the Kruskal-Wallis test is higher than 0.03, and rejected otherwise [50]. A confusion table and accuracy score is then calculated.

Kruskal [50] argues using the ranks could be more beneficial and presents the test statistic H, in case there are no ties (that is, if no two observations are equal) as:

$$H = \frac{12}{M(M+1)} \sum_{i=1}^{P} \frac{R_i^2}{s_i} - 3(M+1), \tag{3.27}$$

where P is the number of samples, s_i is the number of observations in ith sample, M denotes the number of observations in all samples combined, R_i denotes the sum of the ranks in the ith sample. Large values of H lead to rejection of the null hypothesis. The null hypothesis is that the new input has the same distribution as the training data (which is noise). The rejection of null hypothesis due to a large value of H would result in accepting the alternative, which means the new input does not have the same distribution as the training data. Since only noise was used as the training set of this method, rejecting the null hypothesis, in this case, means the test sample is most probable, not noise.

3.6 Mathematical Noise model

In this section, a mathematical model for noise is presented. Fourier transform (explained in Section 2.2) is a way to represent not periodic signals; However, a Fourier series is a way of representing a periodic signal as a sum of sine and cosine functions. A low order approximation of the noise data mean is obtained using Fourier Series (FS) expansion. Moreover, the best model order is estimated using AIC and BIC.

Since the signal was converted to the frequency domain using FFT as explained in Section 2.2, a reverse FFT function was applied to the data to convert frequency domain to time domain in order to apply Fourier expansions. A function called *ifft* in MATLAB was used for this purpose.

3.6.1 Fourier Series

To find a parametric model, a linear regression framework, of the noise is used. In order to obtain a low dimensional feature vector, Fourier Series (FS) expansions are applied.

A Fourier series is a periodic function consists of sinusoids, combined by a weighted summation. The sine-cosine form of Fourier series is:

$$S_n(y_l) = \frac{a_0}{2} + \sum_{t=1}^{T} (a_t cos(2\pi t y_l) + b_t sin(2\pi t y_l)), \tag{3.28}$$

where the parameter set $\{a_t, b_t\}_{t=1}^{T}$ forms the feature space used to identify noise signal and T is the model order. y denotes the input signal, and $l = 1, 2, ..., L$ is the number of samples in one signal.

The trade-off between model complexity and accuracy cannot be ignored. In order to estimate the FS coefficients, for each model order T, the Fourier series expansion (Equation 3.28), is considered as a linear model [43]:

$$\bar{\boldsymbol{y}}_s = \boldsymbol{M_T \theta_T}, \tag{3.29}$$

where

$$\boldsymbol{\theta_T} = [a_0, a_1, ..., a_T, b_0, b_1, ..., b_T], \tag{3.30a}$$

$$\boldsymbol{M_T}^\top = \begin{bmatrix} 1 & \cos(2\pi y_1) & \dots & \cos(2\pi y_L) \\ \vdots & \vdots & \dots & \vdots \\ 1 & \cos(2\pi y_1(T)) & \dots & \cos(2\pi y_L(T)) \\ 0 & \sin(2\pi y_1) & \dots & \sin(2\pi y_L) \\ \vdots & \vdots & \dots & \vdots \\ 0 & \sin(2\pi y_1(T)) & \dots & \sin(2\pi y_L(T)) \end{bmatrix}. \tag{3.30b}$$

The solution to the linear model $\bar{\boldsymbol{y}}_s$ is obtained by solving the following optimization problem:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{T}} = \underset{\boldsymbol{\theta_T}}{argmin}\ V^{LS}(\boldsymbol{\theta_T}), \tag{3.31}$$

where V^{LS} is:

$$V^{LS}(\boldsymbol{\theta_T}) = (\bar{\boldsymbol{y}}_s - \boldsymbol{M_T}\boldsymbol{\theta_T})^\top(\bar{\boldsymbol{y}}_s - \boldsymbol{M_T}\boldsymbol{\theta_T}). \tag{3.32}$$

Finally the closed form solution for $\hat{\theta}$ is:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{Ts}} = (\boldsymbol{M_T}^\top\boldsymbol{M_T})^{-1}\boldsymbol{M_T}^\top\bar{\boldsymbol{y}}_s. \tag{3.33}$$

Best model order in FS expansion is then obtained by modeling the noise for $T \in \{1, ..., T_{max}\}$, where T_{max} is the maximum order set to be considered. Furthermore, the FS coefficients for each order are obtained and evaluated by two well-known model selection criteria: Akaike information criterion (AIC) and Bayesian information criterion (BIC).

3.6.2 AIC

The Akaike information criterion (AIC) is an estimator of relative quality (distance) between the unknown true likelihood function of the data and the fitted likelihood function of the model so that a lower AIC means a model is considered to be closer to the truth. Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models. It is defined as [7]:

$$AIC = (-2)log(L) + 2k, \tag{3.34}$$

where L is the likelihood function which can be denoted as: $p(\bar{\boldsymbol{y}}_s|\hat{\boldsymbol{\theta}}_T) = \boldsymbol{\mathcal{N}}(\boldsymbol{M}_T\hat{\boldsymbol{\theta}}_T, \boldsymbol{M}_T(\boldsymbol{M}_T^\top\boldsymbol{M}_T)^{-1}\boldsymbol{M}_T^\top)$. k is the number of independently adjusted parameters in the model.

The penalty for AIC is less than for BIC, which results in more complex models using AIC compared to BIC [39].

3.6.3 BIC

The Bayesian information criterion (BIC) is a criterion for model selection among a set of models; the model with the lowest BIC is preferred. It is based, on the likelihood function and it is closely related to the Akaike information criterion (AIC), with a greater penalty for the number of parameters and is shown as:

$$BIC = (-2)log(L) + klog(n), \tag{3.35}$$

where n is the number of data points.

3.7 Feature Extraction

The complexity of any machine learning algorithm depends on the number of inputs. In this chapter, feature extraction as a way to exclude inherent information and reduce the dimensions of the data is used. These methods choose a subset of essential features and form fewer, new features from the original inputs.

Assuming that we have a p dimension dataset with N samples, the goal is to reduce the dimensionality of the dataset to reduce the computation. Simpler models are more robust on small datasets and have less variance.

In *feature selection*, some of the p dimensions that give the most information is chosen, and the other dimensions will be discarded[20].

In *feature extraction*, some dimensions are created by combining the original p dimensions. which is the main approach here. To select the most informative features, a survey on related work was done.

Selected features for this thesis are:

1. *Signal Energy* is a useful feature to distinguish fixed or moving objects [32].
2. *Signal Max* obtains the maximum FFT in a given signal.
3. *Signal Variance* highlights the variations in a given signal.
4. $Cross - Correlation$ is used to obtain the correlation between any new signal and noise mean.

The rest of this section is devoted to explaining the selected features in more detail as well as a demonstration of their ability, that was the motivation for them to be chosen.

3.7.1 Signal Energy

The total energy of a signal $\boldsymbol{x}$ is defined as:

$$E_s = \sum_{i=1}^{p} |x_i|^2, \tag{3.36}$$

where p is the number of dimensions (samples), the obtained energy is a useful feature to distinguish fixed or moving objects [32].

Figure 3.4 shows a clear difference in the signal energy level when there is an object present in the 1.5 meter range (Figures 3.4a, 3.4b and 3.4c) compare to when there is no object. In the distance further than 2.5 meters (Figure 3.4f), when there is no object, the energy signal level relatively close to when there is an object, which can indicate that it is harder to distinguish noise and object.

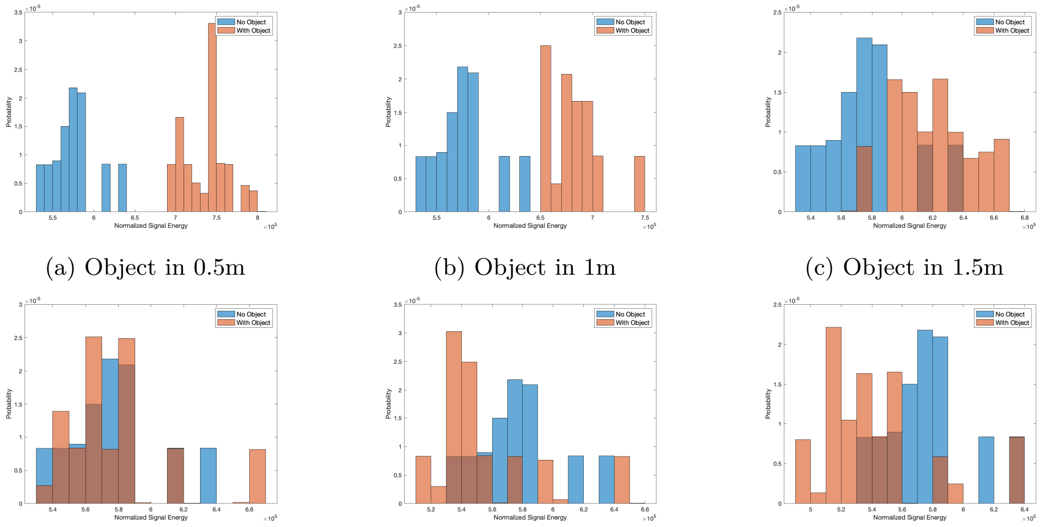
3.7.2 Signal Max

Signal max for a signal $\boldsymbol{x}$ is defined as:

$$S_{max} = max(\boldsymbol{x}). \tag{3.37}$$

Even though Signal Max is a simple feature, it holds valuable information. Signal Max captures the essential tool that CFAR thresholding based methods use.

Figure 3.5 shows the signal max of signals with and without an object in different situations. As it appears, it is an excellent feature to distinguish noise from objects at least within a 2-meter range. The reason is that presence of an object results in stronger reflection; hence, higher maximums in the signal FFT.



(a) Object in 0.5m

(b) Object in 1m

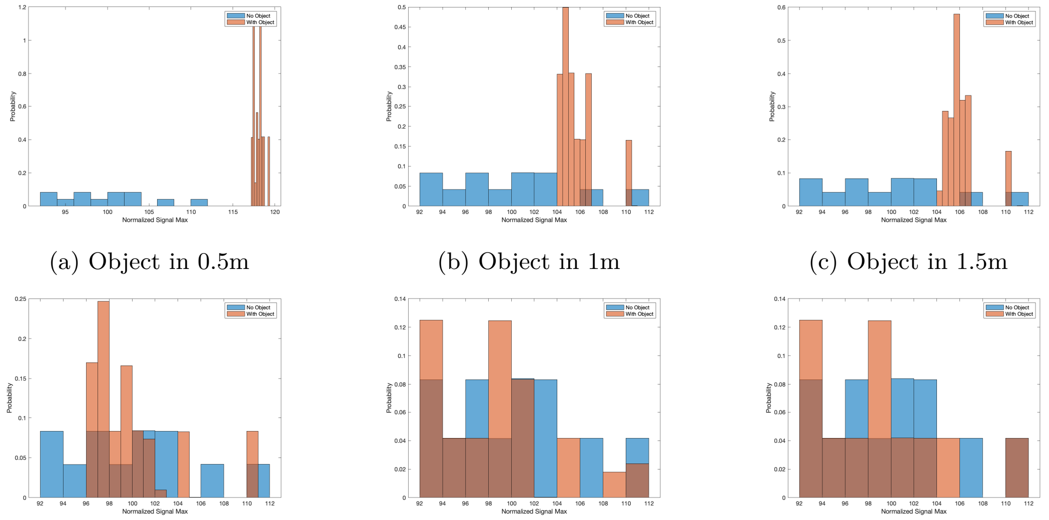
(c) Object in 1.5m

(d) Object in 2m

(e) Object in 2.5m

(f) Object in 3m

Figure 3.4: Normalized signal energy with respect to object's distance. Red bars indicate presence of an object while blue bars mean no object was present while sampling



(a) Object in 0.5m

(b) Object in 1m

(c) Object in 1.5m

(d) Object in 2m

(e) Object in 2.5m

(f) Object in 3m

Figure 3.5: Normalized Signal Norm with respect to object's distance.Red bars indicate presence of an object while blue bars mean no object was present while sampling

3.7.3 Signal Variance

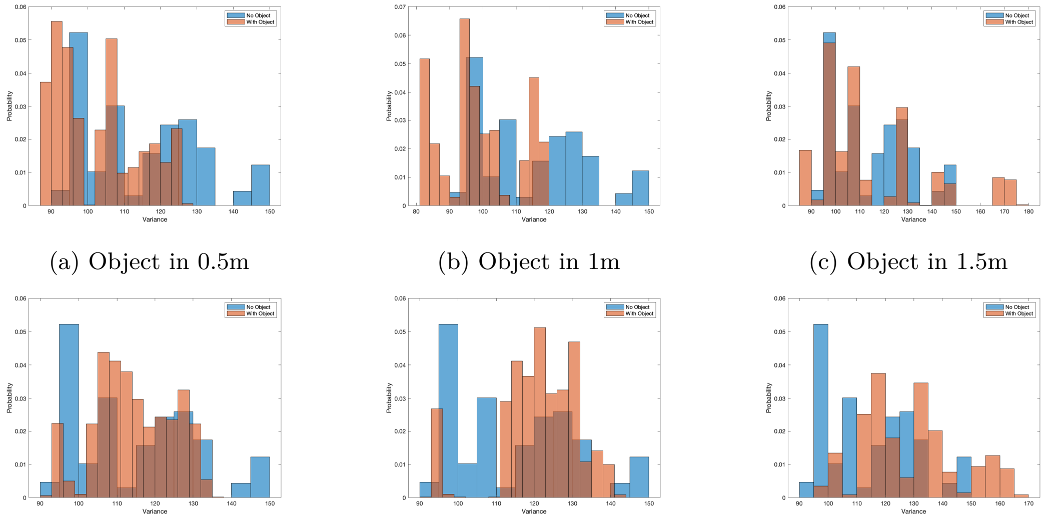
The variance is a tool to characterize the dispersion among the measures in a given dataset. In a given signal, high variance means high amplitude variation of the signal [15]. The variance of a signal also is a good measure to discriminate between high and low movements [43]. The variance (sample variance) of the signal $\boldsymbol{x}$ is defined as the power of the signal with its mean removed:

$$\sigma_x^2 = \frac{1}{p}\sum_{i=1}^{p} |x_i - \mu_x|^2, \tag{3.38}$$

where μ is

$$\mu_x = \frac{1}{p}\sum_{i=1}^{p} x_i. \tag{3.39}$$

. while p denotes the number of samples



(a) Object in 0.5m

(b) Object in 1m

(c) Object in 1.5m

(d) Object in 2m

(e) Object in 2.5m

(f) Object in 3m

Figure 3.6: Normalized variance with respect to object's distance.Red bars indicate presence of an object while blue bars mean no object was present while sampling.

Figure 3.6 illustrates variance in the signals for six different situations, in each, the variance obtained is compared with the variance of no object. As it appears when the object is close to the radar, i.e., figure 3.6a, the variance is relatively low. This could be due to systematic errors such as the peak visible in Figure 2.2, close to the radar, approximately 0.1 meter. As the object moves further away, the variance increases as in the case of figure 3.6f, where the object is in 3 meters, the variance is quite high, almost 3 times higher than when the object is in 0.5 meters.

Hence this feature would be useful when the object is in a closer range.

3.7.4 Cross-Correlation

The mean of all signals classified as noise is computed as:

$$\mu_d = \frac{1}{N}\sum_{i=1}^{N} x_i^{(d)}, \tag{3.40}$$

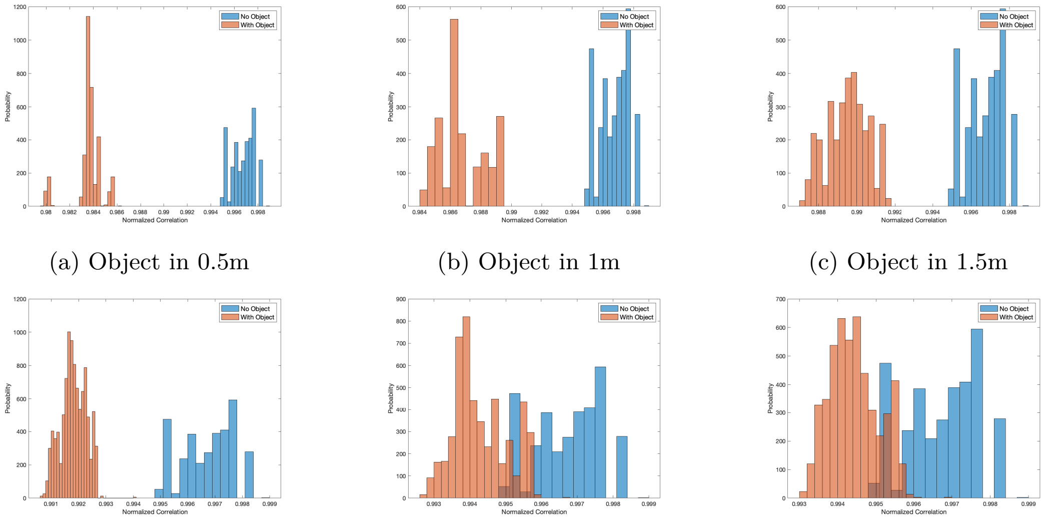
where N is the number of observations, $\boldsymbol{x}^{(d)}$ refers to dth sample, and $d = 1, \dots, p$ while p denotes the number of samples. Hence $\boldsymbol{\mu}$ is a p length vector.

Then cross correlation between the noise mean, and a new input signal $\boldsymbol{y}_n$ is computed as

$$corr(\boldsymbol{y}_n, \boldsymbol{\mu}) = \frac{\boldsymbol{y}_n^{\top}\boldsymbol{\mu}}{\sqrt{(\boldsymbol{y}_n^{\top}\boldsymbol{y}_n)(\boldsymbol{\mu}^{\top}\boldsymbol{\mu})}}. \tag{3.41}$$

.

As it appears in Figure 3.7, correlation is one of the best features so far in distinguishing objects from noise. Almost in all six plots in Figure 3.7, the correlation between a new noise signal and the noise mean is higher than the correlation between an object generated signal and noise mean.



(a) Object in 0.5m

(b) Object in 1m

(c) Object in 1.5m

(d) Object in 2m

(e) Object in 2.5m

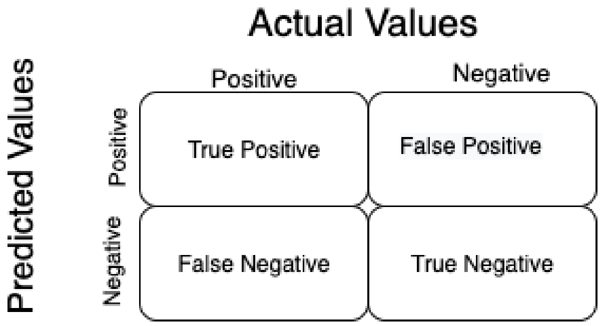
(f) Object in 3m

Figure 3.7: Normalized Correlation of signals with signature signal with respect to object's distance.Red bars indicate presence of an object while blue bars mean no object was present while sampling.

3.8 Method Evaluation

To evaluate the performance of our methods, some classic measures of performance for classification methods such as *precision*, *recall*, *accuracy* and *F*1 *score* are used. This section will describe them in short.

As stated before, in this thesis, positive is noise, and negative is used to refer to anything but noise. Given a confusion matrix (Table 3.8) *True Positive* (TP) refers to when the actual signal was generated from noise and the obtained classification is noise as well, while *True Negative* (TN) means the signal was generated from not noise (object) and was classified as objected correctly. *False Positive* (FP) refers to when a signal was generated from an object; however, it was classified as noise; this is the worst possible outcome that can be fatal. FP has the most effect on the accuracy of safety detection. Finally, *False Negative* (FN) refers to when the signal is generated from an empty state; however it is classified as an object, this will result in unnecessarily slow/stop in the system, which will damage the availability of the system.



Actual Values

Predicted Values

	Positive	Negative
Positive	True Positive	False Positive
Negative	False Negative	True Negative

Figure 3.8: A binary class confusion matrix.

Precision is then defined as:

$$Precision = \frac{TP}{TP + FP}, \tag{3.42}$$

which reflects the number of correct classification rate in all of the positive class classifieds. Precision is important to determine how safe the method is in terms of not missing any object. A low Precision shows that a system is not safe enough and might classify an object as empty state.

Recall is defined as:

$$Recall = \frac{TP}{TP + FN'} \tag{3.43}$$

which reflects the number of correct classification rate in all of the true positive classes. Recall is important to determine how available is the system. Low recall score means many unnecessarily slow/stop in the system, hence a low availability.

Accuracy is simply denote how many true classifications has been done with respect to the whole data:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.44}$$

Finally, since it can be hard to compare two models with low precision and high recall or vice versa an $F1 - score$ is presented as:

$$F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}. \tag{3.45}$$

4 Results

This chapter is devoted to the results of implemented methods; their evaluation is presented in Chapter 5. First, the noise model obtained from the Fourier series is presented in Section 4.1, then the power transformation results are shown in Section 4.2. Section 4.3, and, Section 4.4 present the results of the implemented algorithms on the raw dataset and the features dataset, respectively, for any method they were applicable.

4.1 Generated Noise Model

As explained in Section 3.6, a noise model was generated using the Fourier series. The goal was to have a mathematical model for noise so it can regenerate the noise signal whenever needed. The generated noise model was used as $\boldsymbol{\mu}$ to calculate the cross correlation in Section 3.7.4 . Furthermore the Kruskal-Wallis test was done with the help of this signal.

Figure 4.1 shows the estimated order of AIC and BIC obtained using Equation 3.34 and Equation 3.35 plotted together. Based on this plot (Figure 4.1) Estimated model orders are $T = 9$ and $T = 11$ for BIC and AIC respectively. Furthermore the noise signal was generated using the estimations obtained with the procedure described in Section 3.6; which is shown in Figure 4.2. 95% confidence bounds were computed as $\bar{\boldsymbol{y}}_s \pm (1.96\ \sigma)$ where σ is the standard deviation and $\bar{\boldsymbol{y}}_s$ is the generated noise model. Figure 4.3 shows the approximation error between the generated signal ($\bar{\boldsymbol{y}}_s$) by Fourier series and the original mean ($\boldsymbol{\mu}$) of noise signals. Narrow confidence bounds which were obtained by $(\bar{\boldsymbol{y}}_s - \boldsymbol{\mu}) \pm 1.96\ \sigma$ indicate a small error.

4.2 Power Transformation

In order to achieve a more *normal looking* data distribution, power transformation was performed. Obtained λs from Equation 3.26 were used to transform the data as shown in Equation 3.25.

Figure 4.4 illustrates a small part of the data before transformation using scatter plot matrices. While Figure 4.5 shows the same data after transformation. Scatterplots of each pair of samples are shown under the diagonal. Pearson correlations are displayed over the diagonal. while each sample distribution is available on the diagonal.

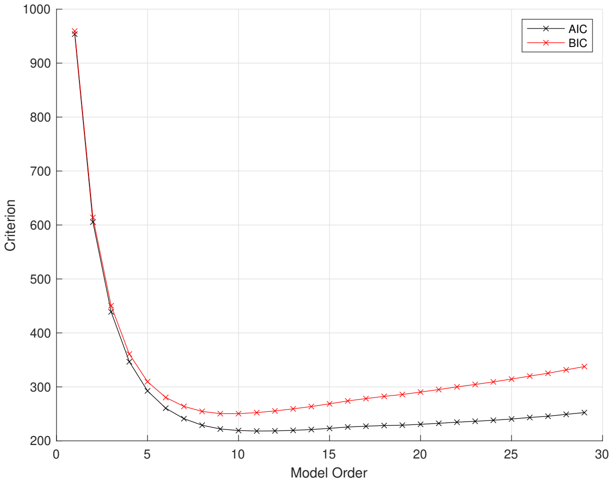


Figure 4.1: Model order selection.

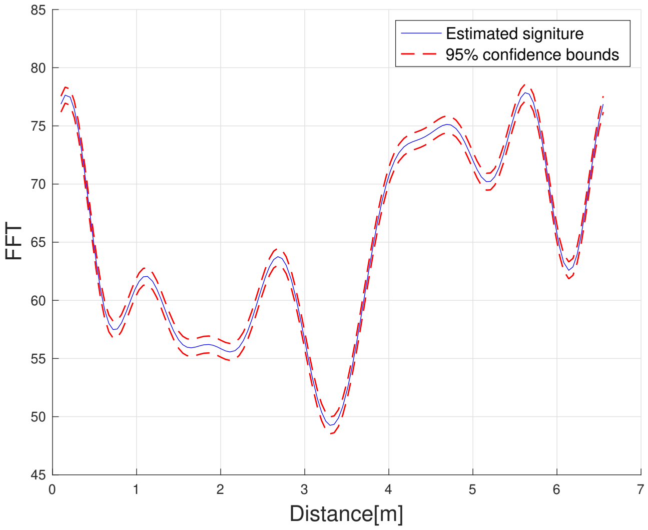


Figure 4.2: Estimated signature using Fourier series with 95% confidence bound

Shapiro-Wilk test of normality for raw and transformed data is shown in Table 4.1. For both cases the $P-value$ is significantly small, however, there is in increase in $P-value$ after transformation.

Table 4.1: Shapiro-Wilk test of normality for raw and transformed data.

Data	P-value
Raw	< 2.2e-16
Transformed	0.0001493

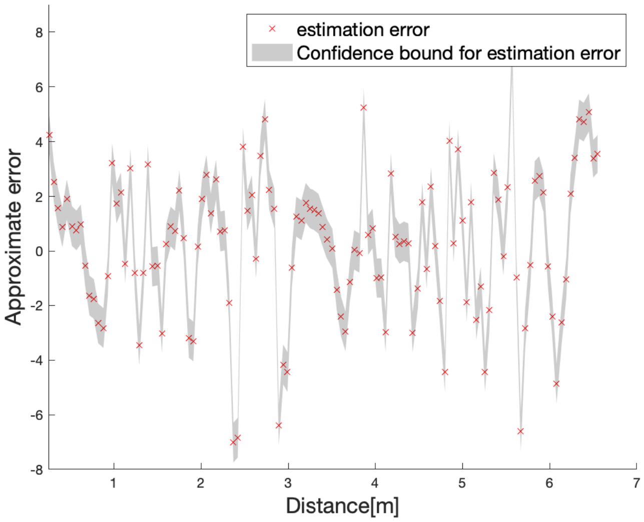


Figure 4.3: The approximation error and the 95% confidence bound

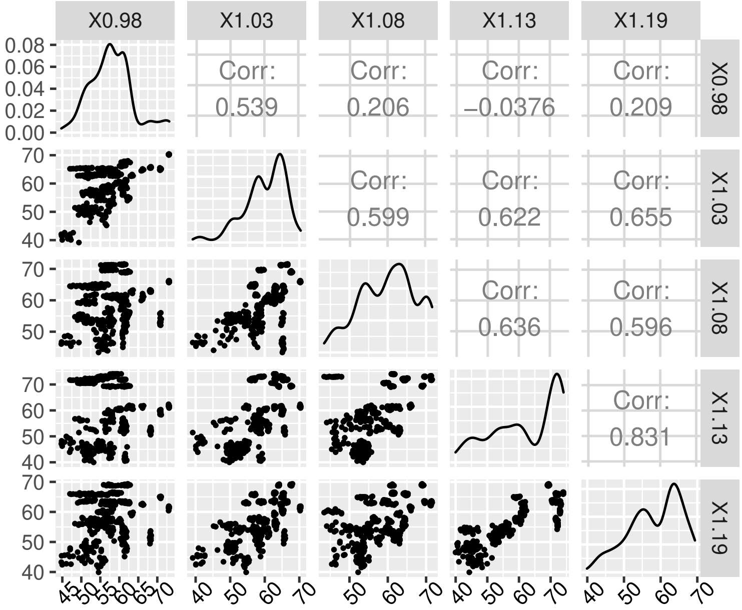


Figure 4.4: Scatter matrix plot of raw data

4.3 Results for raw data

To evaluate binary class classification methods (SVM and KNN) the dataset is divided into train and test. Training consists of 70% of the data set while 30% of it is devoted for testing. K parameter for KNN and ν parameter for SVM approach where chosen based on a grid search. A cross validation was performed to obtain the best kernel for OCSVM.

To train OCC (OCSVM, OC Mahalanobis) only the noise labeled observation of the training dataset were used which included 6300 observations. The mean of these observations was then used to perform Kruskal-Wallis test.

The accuracy, training time, testing time, precision, recall, and $F1-score$ of used methods are shown in Table 4.2.

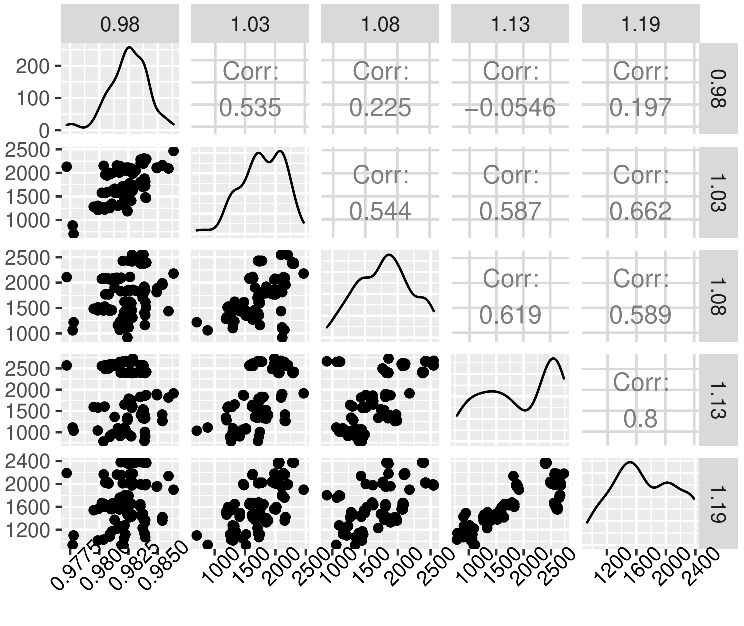


Figure 4.5: Scatter matrix plot of transformed data

Table 4.2: Represents the classification accuracy, training time, testing time, precision, recall, and $F1-score$ of different methods for raw data

Method	Accuracy	Train (s)	Test (s)	Precision	Recall	$F1$
KNN	98.39%	13.21	5.7	1	0.94	0.969
SVM	99.09%	6.6	0.9	0.96	1	0.979
OCSVM	98.86%	4.1	0.50	1	0.95	0.974
Mahalanobis	92.55%	-	116	1	0.704	0.826
Kruskal-Wallis	75.01%	-	34.49	0.479	0.038	0.07

In addition, to test the performance of the Fourier Series generated signal, Kruskal-Wallis results were obtained using the noise model regenerated using Section 4.1 procedure, which is shown in Figure 4.2. The results (Accuracy, Testing time, Precision and Recall which were explained in Section 3.8) are presented in Table 4.3. Since Kruskal-Wallis was the only method that was applicable with only one signal as a reference and did not need any training data, this table has only one row.

Table 4.3: Represents the classification accuracy, training time, testing time, precision, recall, and $F1-score$ of Kruskal-Wallis method obtained with Fourier series

Method	Accuracy	Train (s)	Test (s)	Precision	Recall	$F1$
Kruskal-Wallis	75.30%	-	44.7	0.353	0.012	0.023

4.4 Results for features

Algorithms were also implemented using features extracted in chapter 3.7 to decrease the computation time. Table 4.4 shows the Accuracy, computation time and a summary of used methods confusion matrix for the features.

Table 4.4: Represents the classification accuracy, training time, testing time, precision, recall, and $F1 - score$ of different methods for features

Method	Accuracy	Train (s)	Test (s)	Precision	Recall	$F1$
KNN	98.94%	6.1	0.4	1	0.95	0 .974
SVM	93.06%	2.1	0.099	0.784	1	0.878
OCSVM	93.17%	0.3	0.067	0.807	0.95	0.872
Mahalanobis	88.53%	-	3.64	0.70	0.91	0.791
Kruskal-Wallis	63.93%	-	9.15	0.311	0.35	0.32

In addition to test the performance of the Fourier series, features were obtained using the Fourier series generated noise model instead of the mean for the correlation score, results obtained using the features generated with Fourier series estimations are shown in Table 4.5.

Table 4.5: Represents the classification accuracy, training time, testing time, precision, recall, and $F1 - score$ of different methods for Fourier obtained features

Method	Accuracy	Train (s)	Test (s)	Precision	Recall	$F1$
KNN	90.75%	5.4	0.3	0.996	0.733	0.84
SVM	89.75%	2.5	0.095	0.712	0.997	0.830
OCSVM	89.20%	0.3	0.024	0.70	0.993	0.821
Mahalanobis	81.48%	-	1.7	0.577	0.959	0.72
Kruskal-Wallis	65.11%	-	3.14	0.32	0.37	0.34

5 Discussion

The results reported in Chapter 4 are analyzed and further discussed in this part. The rest of this chapter is organized as follows. Section 5.1 evaluates the proposed noise model in terms of estimation accuracy. Discussions on power transformation results are presented in 5.2. The achieved classification accuracy using the raw data is analyzed in Section 5.3 followed by performance evaluations corresponding to feature-based classification methods presented in Section 5.4. Finally possible future work is discussed in Section 5.5.

5.1 Noise model

In section 4.1, the noise model is presented. Figure 4.1 shows the estimated model order for BIC and AIC. Considering a trade-off between estimation accuracy and model complexity, $AIC = 11$ was chosen as the model order to obtain the highest resemblance of the original signal possible; even though BIC had a lower order. Figure 4.2 shows the generated model using AIC model order together with its associated 95% confidence bound. The confidence narrow bounds are added to better visualize the goodness of the estimated model with respect to the original data.

Figure 4.3 shows the approximation error and its confidence bound. As the Figure suggests, the residuals are within the estimated confidence interval. Additionally, the estimation error terms are not following any pattern and are fairly random, which means there is no systematic error. They have a mean around zero which illustrates that the proposed model is representing the original data fairly well, which means, the approximation error of the low order Fourier series expansion is negligible.

5.2 Power transformation

Power transforms were used to make the data more normal looking. Figure 4.4 illustrates the normality of raw data with the use of scatter plot matrices. The diagonal of the scatter plots matrices shows the density curve which reveals the raw data is extremely not normal. This was the motivation for using power transforms.

Figure 4.5 illustrates the scatter plot matrix for data after transformation. Even though the data distribution is more normal looking, it still can not be considered as normal. The correlations shown on the plot are also relatively higher than the non transformed scatter plot matrix. The scatter plots would indicate normality in case they converge to a line (diagonal) which is not the case here and they do not follow any pattern, which supports the idea of nonnormality.

Shapiro-Wilk test on a sample of the data supports this statement. The null hypothesis was that the sample is normally distributed. The alternative was that the data points are not normally distributed. The critical value for significance was assumed to be 0.01. Table 4.1 illustrates the result for both raw and transformed data. For raw data, it shows a significantly low $p - value$ on the order of 10^{-16} that strongly rejects the null hypothesis. After the transformation, the $p - value$ increases up to 10^{-3}, which is still significantly low; however, it shows an improvement. Considering the improvement, the $p - value$ is still not high enough. Hence transformations are not useful in the case of this thesis.

5.3 Raw Data

Results for implemented algorithms on the raw data were presented in Section 4.3 and summarized in Table 4.2. This table compares five different algorithms used for classification and outlier detection purposes. As it appears on the table, the highest accuracy is obtained with a binary class SVM. However, high accuracy with a relatively good implementation time comes with a price of false-positive happening, which is indicated by precision lower than 1. False positives are critical misclassifications in this application. It means that even though there is an object in the surveillance field, the system is declaring an empty state. Hence, while SVM has the highest accuracy, it might not be the best algorithm tailored for the considered application. In general, precision in this thesis indicates the safety of the classification of the methods. Precision equal to 1 refers to a safe classification in terms of no objects has been missed; the lower the precision gets, the method becomes unreliable. Recall, on the other hand, reflects the availability of the system; in other words, it indicates how many unnecessary slow/stop orders were given based on false detection of the presence of an object. $F1 - score$ is used to show the trade-off between recall and precision for a better understanding of the method performance regarding both safety and availability. Accuracy in this table refers to the total classification accuracy, which is relatively less critical in this application. Finally, training and testing times are presented. While both times are important in this thesis, since their performance is evaluated on offline data, the test time is relatively more important.

While SVM shows great recall, and the highest $F1 - score$, the fastest algorithm for training is OCSVM, with 0.15% less accuracy than SVM and 2.5 seconds faster training time, an a full score precision. The accuracy was expected to be a bit lower since OCSVM only has noise in training but that makes the algorithm training faster than binary SVM. The training time and the accuracy with no false positive in classifications makes OCSVM a great candidate for raw data classification/detection.

KNN could be a good option as well, with a high accuracy and great precision; however, its training and testing time is relatively insufficient, and the recall is lower than OCSVM. Tuning the K parameter is also critical and is not necessarily a trivial task when using KNN. In this specific case, $K = 10$ was chosen after a grid search for the best K.

Mahalanobis distance base OCC method gives relatively low accuracy while consuming the most time among other alternatives. It worth mentioning that for two methods, OCC Mahalanobis distance and Kruskal-Wallis there was no training time since they are simply comparing new signals with the predefined noise. The OCC Mahalanobis distance accuracy was expected to be lower than other algorithms, since this method has a Gaussian assumption which is rejected in section 5.2. Considering the rejected assumption, the accuracy seems to be high which is due to the properties of our data; in the sense that noise and object have fundamentally different patterns. A more challenging dataset where the noise samples and object samples are more similar, would damage the accuracy of this method.

The none-parametric hypothesis testing approach (Kruskal-Wallis) seems to have the worst accuracy among others. This was expected since non-parametric tests are more suitable for analyzing ordinal data and outliers. The Kruskal-Wallis approach was also applied to Fourier series generated mean and the achieved results are presented in Table 4.3. The accuracy is a little bit higher but is still low compared to other algorithms. In both cases $F1 - score$ is significantly low, and also considering the testing time, this method is not suitable for this application.

5.4 Features

Results for implemented algorithms on features obtained using feature extraction (Section 3.7) are presented in Table 4.4. The implementation times of the algorithms for features are considerably lower than the time it takes for the algorithms to classify raw data. The accuracy, however, for most methods, are slightly affected. However, the gain in time is orders of magnitude more than accuracy loss.

Table 4.4 also indicates that the KNN algorithm resulted in the highest accuracy. This is due to different dimensionality between these two datasets (raw dataset and feature dataset). While the raw test dataset had a size of 8100×128, the features data set is a matrix of the size 8100×5. As the dataset size increases, KNN computation time increases faster than SVM [28], this is visible when comparing Table 4.2, which presents the raw data and Table 4.4, which shows the results for features which have fewer dimensions. KNN performers better when there are fewer samples in the dataset. KNN is also the only algorithm with no false positives ($Precision = 1$). Even though OCSVM has much better testing time, and they have the same recall score, the approximately 0.2 difference in precision score can not be neglected. However, the training time is still higher than OCSVM implemented on raw data. Overall, OCSVM is showing an exceptional testing time, but a low precision makes it a not very reliable method.

Table 4.5 reveals the results for all methods on features obtained with the Fourier series. The results are less accurate in general, which was expected since an estimation of the noise mean is used instead of the original mean of the data.

KNN gives the highest accuracy here as well, with a more efficient timing, however, still much less accurate than previously discussed methods.

All methods have some amount of false positive (none had a $precision = 1$); this indicates that even though the timing has improved using features with Fourier series is not a reliable approach for this application, unless in a case where speed is more demanding than accuracy.

5.5 Future work

One important future extension on this work is using multiple sensors instead of one by aggregating their received signals using coordinate transformations. This will be more challenging and realistic to real world situations.

Another interesting approach for future work is considering Neural networks. Neural networks can be used to find and learn patterns in noise and/or in object. These patterns can then be used for classification purposes. Tuning the NN hyperparameters can be a real challenge.

6 Conclusion

In this thesis, several approaches for reliable radar detection were investigated. The machine learning algorithms used can be categorized into three categories: $Binary\ classification$, $One\ class\ classification$, and $None - parametric\ Hypothesis\ testing$. KNN and SVM were the methods used in binary classification. SVM was also used in one-class classification alongside with a Mahalanobis distance-based approach. Furthermore, a None-parametric Hypothesis testing approach (Kruskal-Wallis test) was considered. Lastly, a mathematical noise model was presented in order to be used for feature extraction purposes as well as the Kruskal-Wallis test.

Furthermore, feature extraction was investigated. Features were chosen based on their ability to highlight the important information in the signals. One of the most important features was the cross-correlation score between the mean of all noise signals and a new unknown signal. This feature was then obtained with the help of the previously presented mathematical noise model. These features were then used to form two datasets for evaluating the performance of the algorithms. Using the features instead of raw data helped a great deal with time efficiency. Even though the accuracy was slightly decreased, depending on the application, they could be a good choice

These machine learning and statistical approaches were carried out on a raw dataset, feature dataset, and features constructed by the mathematical noise model dataset. In Binary classification, KNN and SVM were investigated to classify both noise and object in an incoming signal. The study of these two methods demonstrated that for raw data using a hyperplane (SVM) to distinguish noise from object is more efficient in terms of accuracy and time consumption compared to relying on nearest neighbors methods. Furthermore, One-class SVM showed slightly lower accuracy compared to binary SVM. However, better test time and no false positives ($precision = 1$) in classification, made this method the better one for this application. One other benefit of OCSVM is that it only relies on noise data for training, which is proved to be easier to obtain. OCC Mahalanobis distance method and None-parametric Hypothesis testing (Kruskal-Wallis test) did not show any sufficient benefits regarding the time or accuracy compared to other methods.

Bibliography

[1] A.Ganapathiraju, J.Hamaker, and J.Picone. "Support vector machines for speech recognition." In: *International Conference on Spoken Language Processing (ICSLP), Sydney, Australia* (1998).

[2] A.Jalil, H.Yousaf, and M.Baig. "Analysis of CFAR techniques". In: *2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, Jan. 2016.

[3] A.M.Bartkowiak. "Anomaly, novelty, one-class classification: a comprehensive introduction". In: *International Journal of Computer Information Systems and Industrial Management Applications* (2011).

[4] A.Ng. "CS229 Lecture notes - Supervised learning". Support vector machines. 2012.

[5] A.Rosa. "Adaptive Mahalanobis Distance and k -Nearest Neighbor Rule for Fault Detection in Semiconductor Manufacturing". In: *IEEE Transactions on Semiconductor Manufacturing* (2009).

[6] S. Agarwal and A . Sureka. *Using KNN and SVM Based One-Class Classifier for Detecting Online Radicalization on Twitter*. Springer International Publishing, Jan. 7, 2015. 476 pp.

[7] H. Akaike. "A new look at the statistical model identification". In: *IEEE Transactions on Automatic Control* 19.6 (1974), pp. 716–723.

[8] B.Chen, P.K.Varshney, and J.H. Michels. "Adaptive CFAR detection for clutter-edge heterogeneity using Bayesian inference". In: *IEEE Transactions on Aerospace and Electronic Systems* (2003), pp. 1462–1470.

[9] Leo Breiman. "Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author)". In: *Statistical Science* 16.3 (2001), pp. 199–231.

[10] E. O. Brigham and R. E. Morrow. "The fast Fourier transform". In: *IEEE Spectrum* 4.12 (Dec. 1967), pp. 63–70.

[11] C.Bishop. "Novelty detection and neural network validation". In: *IEE Proceedings-Vision, Image and Signal processing* 217–222 (1994).

[12] C.Campbell and K.P.Bennett. "A linear programming approach to novelty detection". In: 2001, pp. 395–401.

[13] C.Cortes and V.Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20.3 (1995), pp. 273–297.

[14] C.Iovescu and S.Rao. “The fundamentals of millimeter wave sensors”. In: *Texas Instruments* (2017).

[15] C.Yanmei, R.Jinchang, Z.Rongchun, and J.Jingping. “Automatic Gait Recognition using Dynamic Variance Features”. In: *IEEE Computer Society* (2006).

[16] M. V. Carretero, R. I. A. Harmanny, and R. P. Trommel. “Smart-CFAR, a machine learning approach to floating level detection in radar”. In: *2019 16th European Radar Conference (EuRAD)*. IEEE. 2019.

[17] Nurul Chamidah and Ito Wasito. “Fetal state classification from cardiotocography based on feature extraction using hybrid K-Means and support vector machine”. In: *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. IEEE, Oct. 2015.

[18] Chih-Chung Chang and Chih-Jen Lin. “LIBSVM”. In: *ACM Transactions on Intelligent Systems and Technology* 2.3 (Apr. 2011), pp. 1–27.

[19] D.Tax and R. Duin. “Support Vector Data Description”. In: *Machine learning* (2004).

[20] E.Alpaydin. *Introduction to Machine Learning*. The MIT Press, Aug. 29, 2014. 640 pp. ISBN: 0262028182.

[21] E.Eskin. “Anomaly detection over noisy data using learned probability distributions”. In: *the 17th International Conference on Machine Learning.* (2000).

[22] F.Santoso, F.Baqai, M.Gwozdz, J.Lange, M.G. Rosenberger, J.Sulzer, and D.Paydarfar. “Applying Machine Learning Algorithms for Automatic Detection of Swallowing from Sound”. In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, July 2019.

[23] G.Ritter and M.Gallegos. “Outliers in statistical pattern recognition and an application to automatic chromosome classification”. In: *Pattern Recognition Letters* 18.6 (June 1997), pp. 525–539.

[24] G. M Hatem, J. W Abdul Sadah, and T. R. Saeed. “Comparative Study of Various CFAR Algorithms for Non-Homogenous Environments”. In: *IOP Conference Series: Materials Science and Engineering* 433 (Nov. 2018), p. 012080.

[25] I.Steinwart and A.Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[26] J.Hardin and D.Rocke. “The Distribution of Robust Distances”. In: *Journal of Computational and Graphical Statistics* 14.4 (Dec. 2005), pp. 928–946.

[27] Johnson, R.Arnold, and Dean W Wichern. *Applied multivariate statistical analysis*. Prentice hall Upper Saddle River, NJ, 2002.

[28] JS.Raikwal and K.Saxena. “Performance evaluation of SVM and k-nearest neighbor algorithm over medical data set”. In: *International Journal of Computer Applications* (2012).

[29] K.Heller, K.Svore, A.Keromytis, and S.Stolfo. “One class support vector machines for detecting anomalous windows registry accesses”. In: *In Proceedings of the Workshop on Data Mining for Computer Security* (2003).

[30] K.Schliep, K.Hechenbichler, and M.Schliep. *Package ‘kknn’*. 2016.

[31] K.Veropoulos, C.Campbell, and N.Cristianini. “Controlling the Sensitivity of Support Vector Machines”. In: *Proceedings of International Joint Conference Artificial Intelligence* (1999).

[32] Parinaz Kasebzadeh, Gustaf Hendeby, Carsten Fritsche, Fredrik Gunnarsson, and Fredrik Gustafsson. “IMU dataset for motion and device mode classification”. In: *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, Sept. 2017.

[33] L.Manevitz and M.Yousef. “One-class SVMs for document classification”. In: *Journal of machine Learning research* (2001), pp. 139–154.

[34] L.Scharf. *Statistical Signal Processing*. Pearson Education (US), Jan. 8, 1991. 544 pp. ISBN: 0201190389.

[35] Tingli Li and Lan Du. “Target Discrimination for SAR ATR Based on Scattering Center Feature and K-center One-Class Classification”. In: *IEEE Sensors Journal* 18.6 (Mar. 2018), pp. 2453–2461.

[36] M.Moya, M.W.Koch, L.D.Hostetler, and D.Larry. “One-class classifier networks for target recognition applications”. In: *NASA STI/Recon Technical Report N* (1993), pp. 797–801.

[37] M.Sugiyama. *Introduction to Statistical Machine Learning*. Elsevier Science & Technology, Sept. 25, 2015.

[38] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics, ProbabilityTheory Group (Formerly: E1071), TU Wien*. 2019.

[39] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. MIT Press Ltd, 2012.

[40] P. Nader, P. Honeine, and P. Beauseroy. “Mahalanobis-based one-class classification”. In: *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. 2014.

[41] Nanoatzin. “Constant false alarm rate (CFAR)”. In: *CC BY-SA 3.0* (2011).

[42] P.Filzmoser. *A multivariate outlier detection method*. 2004.

[43] P.Kasebzadeh. *Learning Human Gait*. Linköping: Linköping University Electronic Press, 2019. ISBN: 9789175190143.

[44] R.Sandeep. “Introduction to mmWave sensing: FMCW radars”. In: *Texas Instruments* (2017).

[45] R.Sandeep. “IWR6843 Sensor Manual”. In: *Texas Instruments* (2018).

[46] M. A. Richards. *Fundamentals of radar signal processing*. New York: McGraw-Hill Education, 2014. ISBN: 9780071798327.

[47] S.Khan and M.Madden. “A Survey of Recent Trends in One Class Classification”. In: *Irish conference on artificial intelligence and cognitive science*. Springer. 2009.

[48] S.Rao, A.Ahmad, J.C.Roh, and S.Bharadwaj. “77GHz single chip radar sensor enables automotive body and chassis applications”. In: *Texas Instruments* (2017).

[49] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. “Advances in neural information processing systems”. In: 2000. Chap. Support vector method for novelty detection, pp. 582–588.

[50] W.Kruskal and W.Wallis. “Use of Ranks in One-Criterion Variance Analysis”. In: *Journal of the American Statistical Association* (1952), pp. 583–621.

[51] S. Wenzhu, H.Wenting, X.Zufeng, and C.Jianping. “Overview of one-Class Classification”. In: *2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP)* (2019), pp. 6–10.