Machine learning for condition monitoring in power converters

Department of Electrical Engineering

Written by: Marc Saade

September 2019

Abstract

Switch mode power supplies are used in many applications such as in railways, aeronautics and hybrid vehicles. The degradation of the components plays an important role in the reliability of these devices. Degradation of different components in the switch mode power supply can be characterized. For instance, the degradation of MOSFET can be identified by the increase of its on-state resistance, while the degradation of the capacitor can be characterized by the decrease of its capacitance value. By identifying which component is degrading and whether their are in critical condition or not, is crucial in order to apply a good cost effective maintenance.

In this report, a set of classification algorithms, based on machine learning algorithm, are trained through data collected from a Buck converter that was simulated in Matlab\Simulink in order to identify which component is degrading and to determine its condition. These algorithms are then tested after training in order to assess their accuracy. The results are presented to observe how each algorithm is capable of predicting the component and its condition correctly. Moreover, the report will provide information on how to use filters in order to reduce noise from collected sensor data since pre-processing is needed in practice before using the data as training for the algorithms.

The report starts with a theory chapter that provides information on the sensitivity analysis, used to identify which features can be useful in the training data for the machine learning algorithms. Moreover, it provides simple explanation of how each algorithm works. The chapter ends with the explanation of the importance of noise reduction in the data and the sampling rate during the collection of the data. The next chapter will show how the data is collected and used for training the algorithms and how noise is created in the signal in order to be filtered. In the end of the report, a conclusion will be drawn based on the results and will also provide what other experiments could have been done for future work.

Dedication

This project is dedicated to my family and friends whose solid love, encouragement and support have enriched my soul and motivated me to pursue and complete this research.

Declaration

I, Marc Saade hereby declare that the project work entitled "Machine learning for condition monitoring in power converters" is a record of an original work done by me under the guidance of Prof. Massimo Bongiorno in partial fulfillment for the award of degree of MSc in Electric Power Engineering at Chalmers university of technology. The results of this thesis have not been submitted to any other university for the award of any degree or diploma. I would also like to declare that, to the best of my knowledge, my thesis does not transgress on anyone else copyright and that my thesis is a true copy.

Acknowledgements

This thesis was made possible because of the help and support of many individuals that I would like to extend my thanks to them.

First, I would like to thank my supervisor Prof. Massimo Bongiorno for his suggestions and guidance about how to do the work. I would like to give thanks to Prof. Jan Svensson for his comments on my work which gave me a good perspective of how to complete the project. And last but not least, I am also grateful for the two machine learning expert, Dr. Azam Bagheri and Dr. Hannes Hagmar for their valuable information on machine learning.

Nomenclature

ANN	Artificial Neural Network
DUT	Device under Test
FFT	Fast Fourier Transform
IC	Capacitor Current
IC_max	Capacitor maximum Current value
IC_mean	Capacitor mean Current value
IC_median	Capacitor median Current value
IC_min	Capacitor minimum Current value
IC_RMS	Capacitor Current root mean square value
IC_STD	Capacitor Current standard deviation value
IC_var	Capacitor variance Current value
IL	Inductor Current
IL_max	Inductor maximum Current value
IL_mean	Inductor mean Current value
IL_median	Inductor median Current value
IL_min	Inductor minimum Current value
IL_RMS	Inductor Current root means sqaure value
IL_STD	Inductor Current standard deviation value
IL_var	Inductor variance Current value
IM	MOSFET Current
IM_max	MOSFET maximum Current value
IM_mean	MOSFET mean Current value
IM_median	MOSFET median Current value
IM_min	MOSFET minimum Current value

IM_RMS	MOSFET root mean square Current value
IM_STD	MOSFET Current standard deviation value
IM_var	MOSFET variance Current value
KNN	Kernel Nearest Neighbor
ML	Machine Learning
PC	Principle Component
pc	principle component
PCA	Principle Component Analysis
RUP	Remaining Useful Performance
SVM	Support Vector Machine
VD	Diode Voltage
VD_max	Diode maximum Voltage value
VD_mean	Diode mean Voltage value
VD_median	Diode median Voltage value
VD_min	Diode minimum Voltage value
VD_RMS	Diode Voltage root mean square value
VD_STD	Diode Voltage standard deviation value
VD_var	Diode variance Voltage value
VL	Inductor Voltage
VL_max	Inductor maximum Voltage value
VL_mean	Inductor mean Voltage value
VL_median	Inductor median Voltage value
VL_min	Inductor minimum Voltage value
VL_RMS	Inductor Voltage root means square
VL_STD	Inductor Voltage standard deviation
VL_var	Inductor variance Voltage value
VM	MOSFET Voltage
VM_max	MOSFET maximum Voltage value
VM_mean	MOSFET mean Voltage value

VM_median	MOSFET median Voltage value
VM_min	MOSFET minimum Voltage value
VM_RMS	MOSFET Voltage root mean square value
VM_STD	MOSFET Voltage stadard deviation
VM_var	MOSFET variance Voltage value
VO	Output Voltage
VO_max	Output maximum Voltage value
VO_mean	Output mean Voltage value
VO_median	Output median Voltage value
VO_min	Output minimum Voltage value
VO_RMS	Output Voltage root mean square value
VO_STD	Output Voltage standard deviation
VO_var	Output variance Voltage value

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Power electronic converters are used in many applications, spacing from hybrid vehicles to wind turbines. They play a key role in these systems and, therefore, should always have a high reliability. Generally, the fault in a converter is of two types. The first type is called a catastrophic fault which is created due to a rapid change in the circuit like a short or open-circuit happens. The second type is referred to as parametric fault, which means that components of a circuit drift away from their original or nominal value (like the degradation of a capacitor or semiconductor switch). The parametric fault can take long time to affect the overall performance of the converter. For both fault types, maintenance will be needed. But when it comes to parametric fault, different type of maintenance can be applied. Before explaining the basic type of maintenance applied in the industry, in order to have a clear understanding of the importance of maintenance, we will demonstrate by an example. The figures 1.1 and 1.2 below, show an electricity generating wind turbine that was operating close to Hornslet Aarhus (Denmark) during a windy day and how it broke down.

Figure 1.1: Wind Turbine Normal Operation [1]

Figure 1.2: Wind Turbine Destroyed [1]

The destruction of the wind turbine occurred because of the breaking system inside the turbine, whose role is to prevent the wind turbine's blades from spinning too fast during high speed, was not functioning. The insufficient maintenance led to the failure of the breaking system and the blades started to spin so fast that

they tore themselves apart from the rotation forces. This resulted in a complete destruction of the turbine(the debris of the destruction was found a half mile away from the destroyed turbine)[1]. This pose a serious problem considering the fact that wind turbine are expensive to built and their failures could injure people. Their are many types of maintenance strategy in the industry [7]. The first type is called the corrective maintenance which is done after the breakdown of a device. Clearly, in the case such as wind turbine this is not a good cost efficient type of maintenance. The second type is to perform more scheduled maintenance and this is called the preventive maintenance. However, preventive maintenance can be very expensive and in the case of the wind turbine, it can be dangerous due to the fact that the maintenance crew might have to go up to the top of the wind turbine which can reach a height of 70 meters. Moreover, there is also a risk that the team might perform maintenance when it is not needed. However, if there is a system that can avoid such catastrophic failures and reduce the amount of maintenance, than this could be a huge improvement from both economic and safety point of view. Therefore, industries are now using a new maintenance method called the predictive maintenance which involve machine learning to predict possible failures in a system.

1.2 Motivation for the work

Most people uses Machine Learning (ML) without knowing. The ML is not just used for predictive maintenance purposes, it is also applied in other applications such as in Websites. For instance, whenever we watch a set of video in our YouTube account we are actually feeding input to the ML. The ML will learn from these input and provides the user with recommended videos that might interest us [8]. This is also valid for Netflix. Whenever the user watches a film or TV show in his account, the ML will see a certain pattern and provides us with recommended movies or TV shows that might interest us [9]. In the same way we can simulate certain parametric faults in a converter so that our ML can see a pattern and predict which component in our circuit that is prone to failure in order to apply a predictive maintenance.

1.3 Literature review

The predictive maintenance is not an unfamiliar concept in the field of power electronic converters. In fact in [10], the authors varied a different component value such as capacitor and inductor in a super-buck converter and at each variation they collected the mean and standard deviation of the output voltage and stored it as a vector. After that they calculated the mahalanobis distance between the stored vector in order to observe the difference when a certain component value degrades compared to its optimal value and used it to train the ML algorithm. The result shows that the trained ML is capable of predicting the RUP of the entire converter. In [11] they used a set of circuits (four operational amplifiers biquad highpass filters, nonlinear rectifier circuit and Sallen–Key bandpass filter circuit) and extracted the output voltage response and calculated the cosine distance which in turns is fed to the relevance vector machine algorithm in order to estimate the remaining useful life of the entire circuit. In [12], they used a Buck converter and collected the inductive current, voltage across capacitor, capacitor current, and output voltage along with

their ripple value whenever they change a value in the component and used them as inputs to train an artificial neural network in order to estimate the current parameter value of a certain degrading component (such as inductor and capacitor). The overall goal is to make sure that the value of the component that is degrading, does not exceed its critical values. The ML method was also used for fault diagnostics in the power electronic converter. In [13], they used three-parallel power conversion system for a wind turbine in a simulation as a test set, and collected the three phase measured current and converted them into the dq-frame. In the dq-frame, the current and voltage showed a unique kind of shape whenever a specific switch had stopped working. Hence, a neural network was used in order to identify which switches were faulty by using these dq plots as features.

Unlike previous research, this thesis will focus on using a different set of ML algorithms in order to identify which component is about to fail. This report will apply sensitivity analysis in order to see how the features for the ML is influenced by the certain change in a component. We also observe how accuracy change for different sets of conditions. For testing the ML algorithms, we used an open loop Buck converter model in Simulink as a DUT.

1.3.1 Thesis structure

The report is structured as follows:

- Chapter 2: Theory;
- Chapter 3: Investigating system;
- Chapter 4: Results;
- Chapter 5: Conclusion and future work.

All the chapters (Except for Chapter 5) will have a small introduction in the beginning and a summary in the end.

Chapter 2

Theory

This chapter will provide information about:

- *The basic concept of sensitivity analysis. What it does and how does it work?*
- *Provide information on the machine learning. What are the different types? What are different kind of classification algorithms and how they work?*
- *Data retrieval, which involve the importance of sampling and the reduction of noise in the signal.*

2.1 Sensitivity analysis

In any electrical circuit, the values of its components are most likely to change over time [14] [15]. These changes will affect the output response of the circuit, particularly the output voltage for example. It will also affect other voltages and currents across the entire circuit. In order to determine the "strength" of the effect of one parameter change over a certain current or voltage (or other responses that we will show in a later chapter), a sensitivity analysis is needed. Hence, sensitivity analysis is used in order to observe how a certain output of interest in a system is affected when one of its component value deviates from its normal condition. The sensitivity analysis method is used in many other fields such as in ecological, chemical, semiconductor material and economics for decision making. In the case of power electronic converters the sensitivity analysis is used in order to optimize the design of the electrical circuit. In our case, however, we will use the sensitivity method in order to see how certain features are influenced by a certain change in one of the parameter components in the Buck converter. This could be helpful in order to find out how the features (or inputs) are changing whenever a component is changed in order to create training data for the ML.

Cross correlation

In sensitivity analysis, the correlation method is one of many that is used in order to observe how a certain parameter variation has an affect on a specific function. The cross correlation method is often used in sensitivity analysis to analyse how a model parameter and the cost function are correlated. In order to calculated the

cross correlation, the following equations are used [16]:

$$R(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}} \tag{2.1}$$

where R is correlation coefficients and C is the covariance matrix and can be expressed as:

$$C = cov(x,y) = E[(x - \mu_x) \cdot (y - \mu_y)] \tag{2.2}$$

$$\mu_x = E[x] \tag{2.3}$$

$$\mu_y = E[y] \tag{2.4}$$

The μ_y and the μ_y are the mean value of the x and y. The x value contains the number of samples in the model parameters. The y represents a set of cost function evaluation for a sample in x. The maximum and minimum value of R is 1 and -1 and the (i,j) entry of R indicates the correlation between x(i) and y(i). if R(i,j) is positive, the variables increases together. if R(i,j) is negative, the input parameter value of a component will be inversely proportional to the output of interests. if the value of R is zero, then their is no correlation between the input parameter and the output of interests.

2.2 ML

Basically, any engineer is capable of writing a certain code in order to program a software to do a certain activity. Hence, the program learns from an explicit code that the programmer writes. What makes the ML or Artificial Intelligence (AI) different, is that it enables any kind of system to learn from data instead from explicit programming [17]. The ML algorithm takes the input data in order to be trained. Once the training is complete, it will then be capable of providing the output response. Hence, if the algorithm is of predictive algorithm, the ML will give a predictive model. This means that whenever a user provides a recently trained predictive ML model with data, the user will receive a prediction based on the data that trained the model. However, the data that we use to provide the machine for training, must provide a good information. To better understand, imagine if you want to train the ML that is able to tell the difference between cat and a dog by providing certain features which are a set of descriptions of the animal. The selection of these sets of features, affect the accuracy of the ML model. For example using the weight of the animal as the only input, will result with a very bad prediction of the model given the fact that there are dogs and cats that have the same weight. If you add more features such as eye colour and the shape of the ear, it could perform much better. However, that does not mean that the more feature the better, the features must provide a good distinction between the two animals. In this case we could use just only one feature that is very power full which is the voice of the two animals since their is a difference between the two animals voices. Another problem one should consider is the size of the data. If the input data is big, it will take long time for the ML algorithm to train and also a long time to provide a response during its application. An innovative business with a fast changing market for example would prefer to have an AI that can provide an answer within seconds . Hence, it is important to identify the size of data and the type of features in order to train an AI so that it can provide a fast and accurate response.

2.2.1 Types of ML

Except selecting meaningful data, the user must also know what type of ML algorithm to use. Generally, the most used types of learning models are [17]:

- Supervised learning;
- Unsupervised learning.

These models will be briefly explained in order to clarify the difference between them.

Supervised learning

In the supervised learning, the ML establishes the data in which the user already knows what the correct response of the trained machine should be. Hence, in supervised learning, the data that is fed to the machine for training will contain the desired outputs (or labels) so that a certain function can calculate an error for a given prediction, as can be shown in the Figure 2.1 below. The supervision comes

when a prediction is made and an error produced (actual vs. desired) to alter the function and learn the mapping.

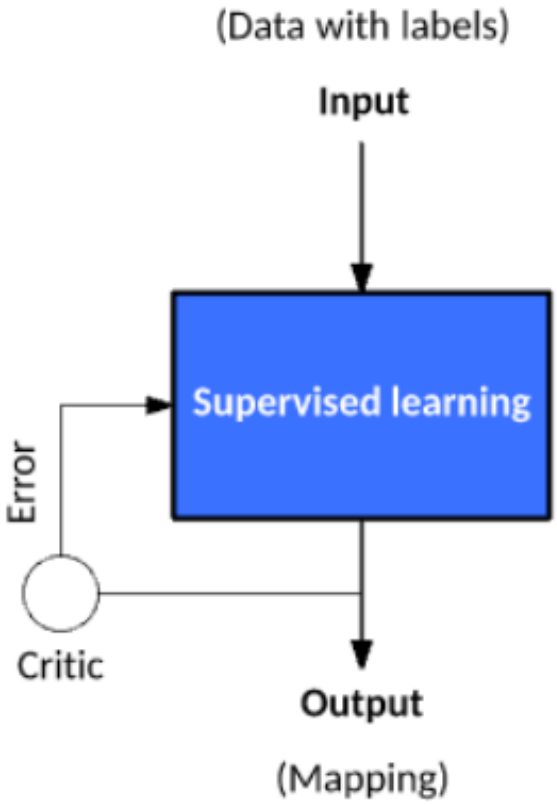


Figure 2.1: Supervised Learning

The response of the ML in supervised learning comes in two ways. If the response is discrete, as the previous example of cat and dog, then it is called classification. Otherwise, if the response of the ML is continuous, for example when a certain feature is given to the ML in order to estimate the value of the inductor, then it is called regression.

Unsupervised learning

The unsupervised learning is mostly used when an individual does not know the correct answer of a certain data and, therefore, the data will be unlabeled. Hence, in unsupervised learning the data does not have a desired output(or label) as can be seen in Figure 2.2 and there's no way to supervise the function. Instead, the function attempts to segment the data set into groups of features that are, in a way, have some similarities. Hence, unsupervised learning of ML analyses the data without human intervention.

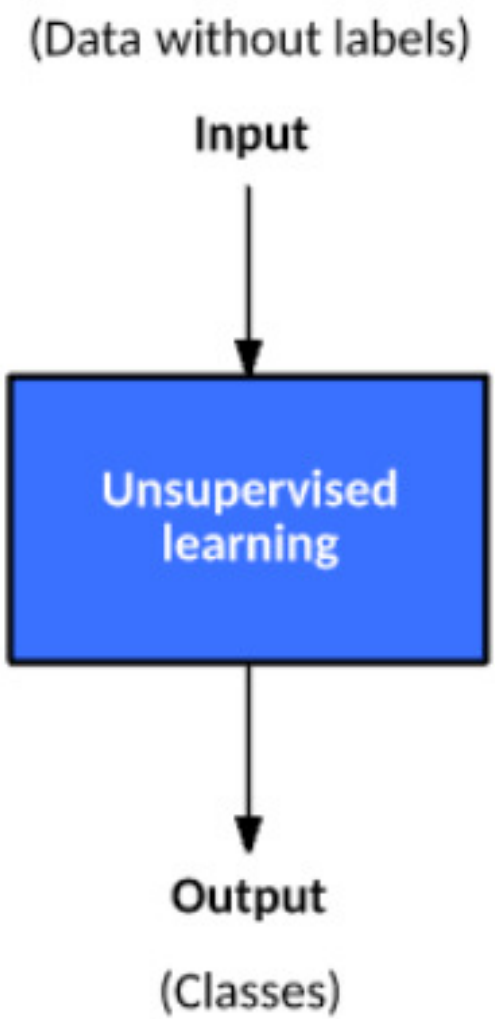


Figure 2.2: Unsupervised Learning

For example, unsupervised learning is used in detecting e-mail spam. Given the fact that there are too many variables in legitimate and spam emails for an analyst to flag, the unsupervised ML will group the spam email based on certain familiarity's and association are applied in order to identify unwanted email.

Since, our project is about predicting which component of the Buck converter is about to fail, than the ML will be using the supervised classification learning.

2.2.2 Under-fitting and over-fitting problems

Before explaining the under-fitting and over-fitting curve[17], one needs to understand how the basic ML provides a model for a classification problem.

Lets assume that there is a completed data with certain features as input for the ML and a set of discrete output responses. The entire data is used for the training. During the training phase, and in the case of supervised classification, the ML will try to create a statistical model in order to separate the different labels (answers) from each other based on the features.

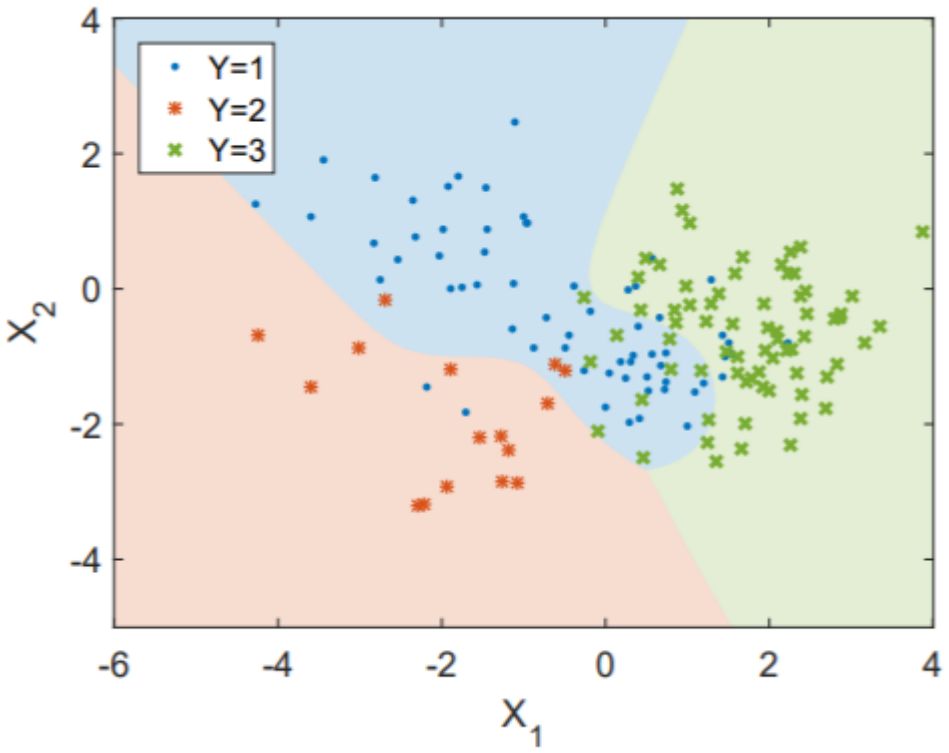


Figure 2.3: Separation of different output responses

Figure 2.3 shows a plot of three different output responses that are separated by different colored zones. The blue zone belongs to responses for Y=1, the red zone belongs Y=2 and the green zone to Y=3. The X_1 axis is a certain feature (input for the machine) type and the X_2 is another feature that is different from the last one. It can be seen from the figure that sometimes, there are occasions when the blue dots are located in the red zone and same thing for the other dots. This indicates that the ML is capable of generalizing the model in order to have a good accurate prediction given the fact that most of the colored dot are located in the correct zone. This is what is called a good fit.

Under-fitted model means that the ML algorithm is not capable of modeling the training data nor generalizing the new data. Figure 2.4 shows an example of how an under-fitting model would look like.

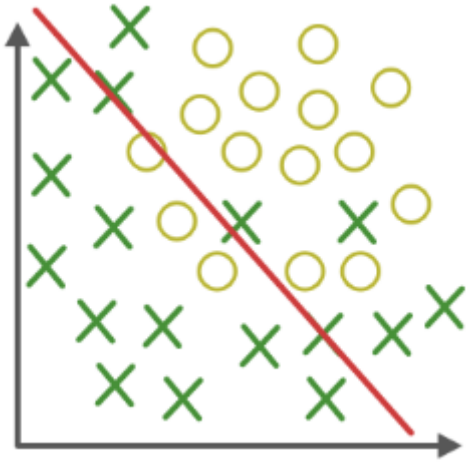


Figure 2.4: Under-fitting

The linear red line is the statistical model that is separating the 'X' responses and the 'O' responses. This would result in a poor accuracy of the ML due to the fact the model does not fit the data well and will result in the model providing wrong response. Under-fitting problems generally occurs when we have a less data in order to train the ML or if the input data does not provide good information for the ML. Hence, a solution is to provide bigger size data with good features or simply to add more input data. An over-fitting model however, means that the trained model fit the training data perfectly but has poor accuracy on the test data (new data which was not part of the training), meaning that after the training of the ML is complete and when new input data that was not part of testing is used in order to predict a response, the ML does not provide a correct answer.

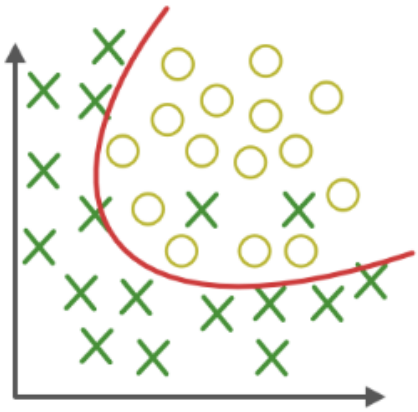


Figure 2.5: Appropriate Fitting

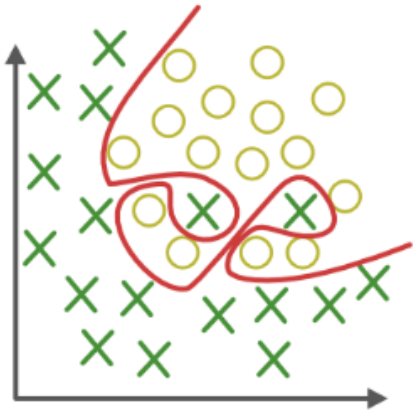


Figure 2.6: Over-Fitting

Figures 2.5 and 2.6 show how a generalized fitting would look like compare to over-fitting. The generalize fitting will make small mistakes as can be seen from Figure 2.5 where the red line curve that separates the 'X' from the 'O' does have two 'X' located on the other side of the curve. But mostly all the 'X' are on the correct side of the curve. In the over-fitting one can observe that the curve is not made to generalize the response, but to simply create a model that is only perfect for the data that was given in the training. Unlike in under-fitting, the over-fitting

occurs when one uses a lot of data during the training. Over-fitting will also occur because of non-parametric and non-linear methods for which the ML algorithms will have more freedom in building the model based on the data set resulting in unrealistic models. There are many ways to reduce the possibility of over-fitting. One of the most known methods is cross validation [18].

Cross validation

Cross validation is a statistical method that is used to evaluate and compare ML algorithms by diving data into two parts: One part will be used to train the ML and the other part will be used to validate the ML by testing the trained ML on the rest of data that was not part of the training. Usually in cross-validation, the training and validation sets have to cross-over in successive rounds so that each data point has a chance of being validated against. There are many types of cross validation, but the basic one is K-fold cross validation.

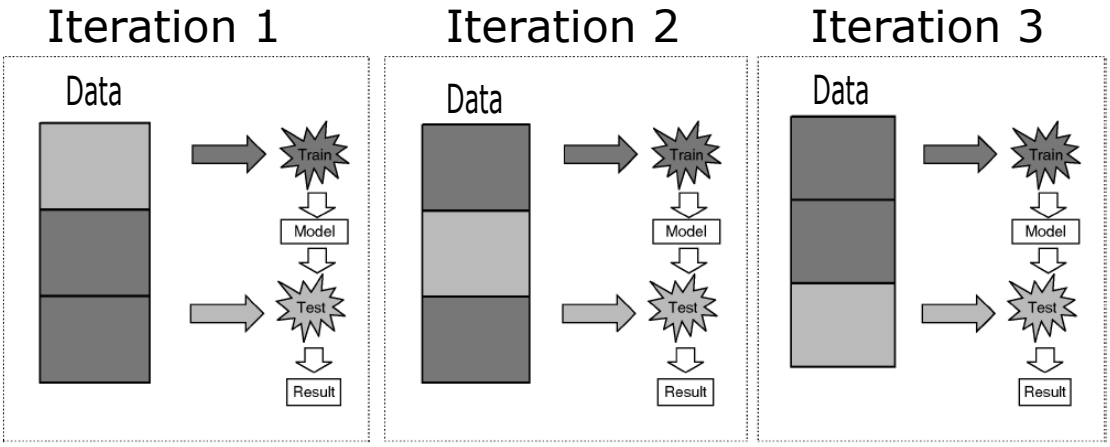


Figure 2.7: Procedure of three-fold cross-validation [2]

Figure 2.7 shows an example of a three fold cross validation. The dark gray colored square represents data that are used for training while the light colored gray square represents data that is used for validation. In the three fold cross validation, the data is first partitioned into 3 size segments or folds. As a result, three iterations of training and validation are made. Within each iteration, a different part of the data will be held out for validation while the (k-1) will be used for learning. In most cases, for the cross validation a 10th fold is chosen [19].

2.2.3 Confusion matrix

When the ML algorithm has been completed and tested on new data, we can observe its accuracy by plotting the confusion matrix. The confusion matrix summarizes the performance of the trained classifier. It is considered to be a useful method since it provides us with an idea of what the trained classifier is getting right and what is getting wrong. In other words, it shows how the model get confused when it makes prediction. To get a better picture of what a confusion matrix would look like, we have taken the result of a trained classifier from [2] which is shown in the Figure 2.8 below.

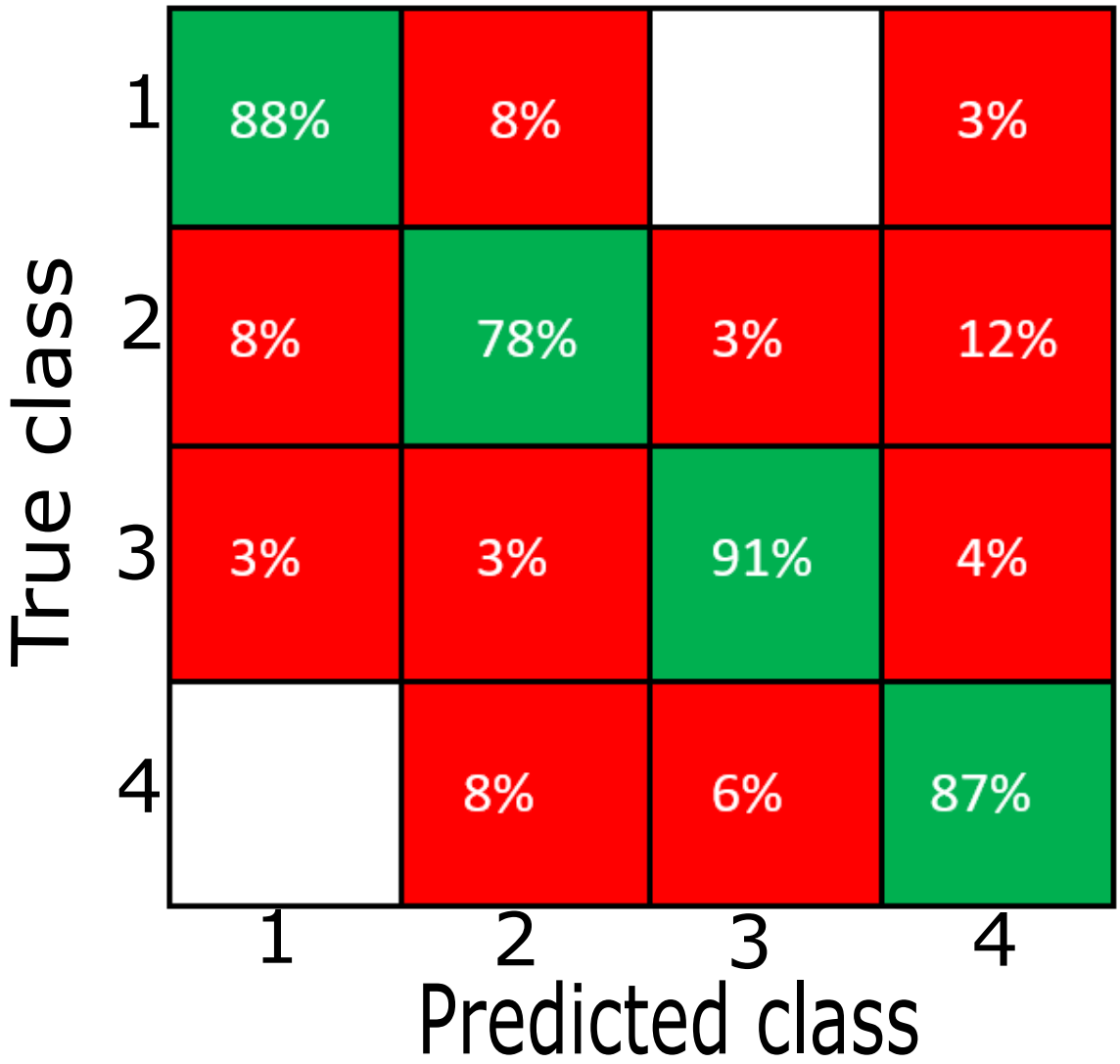


Figure 2.8: confusion matrix

Based on the figure, we can see that we have four classes (1, 2, 3 and 4). The confusion matrix will be built by putting the true class, which shows the correct labels versus the predicted class, in order to visualize how the model is capable of predicting the correct answers. The main goal is to have high percentage (high accuracy) on the diagonal (colored in green) since this will results in a good prediction of the ML. The way to read the confusion matrix is simple. By looking at the label number 1 on the true class for example, it shows that 88 percent of the time the model will predict a correct class. However, their 8 percent chance that it will output a response of 2 while the correct is one, 3 percent of chance that it will predict a class 4 instead of 1 and it will never get output a class 3. Same thing can be read for the other classes. Note that the white colored squares indicates that the probability of prediction is 0% (meaning that the ML will note output such response).

2.2.4 Classification algorithms

There are many type of classification algorithms. Hence, in this section we will talk about the classification algorithm that were used in the simulation.

Decision trees

Decision trees are one of the most famous type of classification algorithm and are also considered to be extremely powerful [3]. It contains a number of nodes that form a rooted tree. Hence, the algorithm will predict a label based on an input data 'X' that is given to the algorithm. The output response of the algorithm comes from the root node of a tree to a leaf. To better understand how a decision tree works imagine that the tree (from root to leaf) is constituted by a number of flow-charts divided in levels, where each level has question which has a yes or no answer in order to move on to a lower level. If we consider the scenario in Figure 2.9 where the algorithm needs to classify based on certain questions to tell whether the animal is mammals, bird or fish. The figure shows the decision tree splitting the data into two questions.

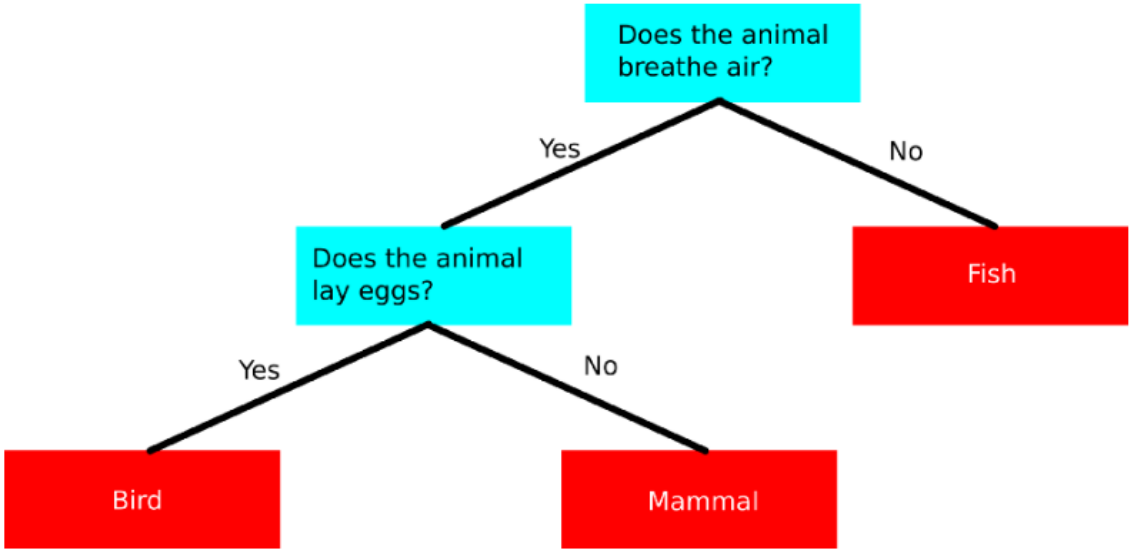


Figure 2.9: Decision Tree [3]

Hence, if we provide the model with a new animal such as a cat, it will classify it. Question 1: Does the cat breath air? Answer 1: yes. Question 2: Does the cat lay eggs. Answer 2: No. The lowest chart is reached which is the final answer and it is mammal.

Support vector machine (SVM)

Like the Decision tree, the SVM (also referred to as "road machine") can be used for both classification and regression problems. If we look at any road for example[20], the cars on the right side of the road is separated by cars on the left side using a yellow line. This is the same principle of operation of the SVM (that is why it is referred to as a road machine) in the case a line separation is possible. To better understand, Figure 2.10 shows a geometric representation of the SVM.

Assume that there are two sets of features (inputs) X1, X2 and two labels 1 for green and -1 for red [20][21]. The SVM will create a hyper-plane that separates the two labels from each other. Hence, the first step, is to calculate all the distances between the dots for the two different classes. From the figure, support vectors are

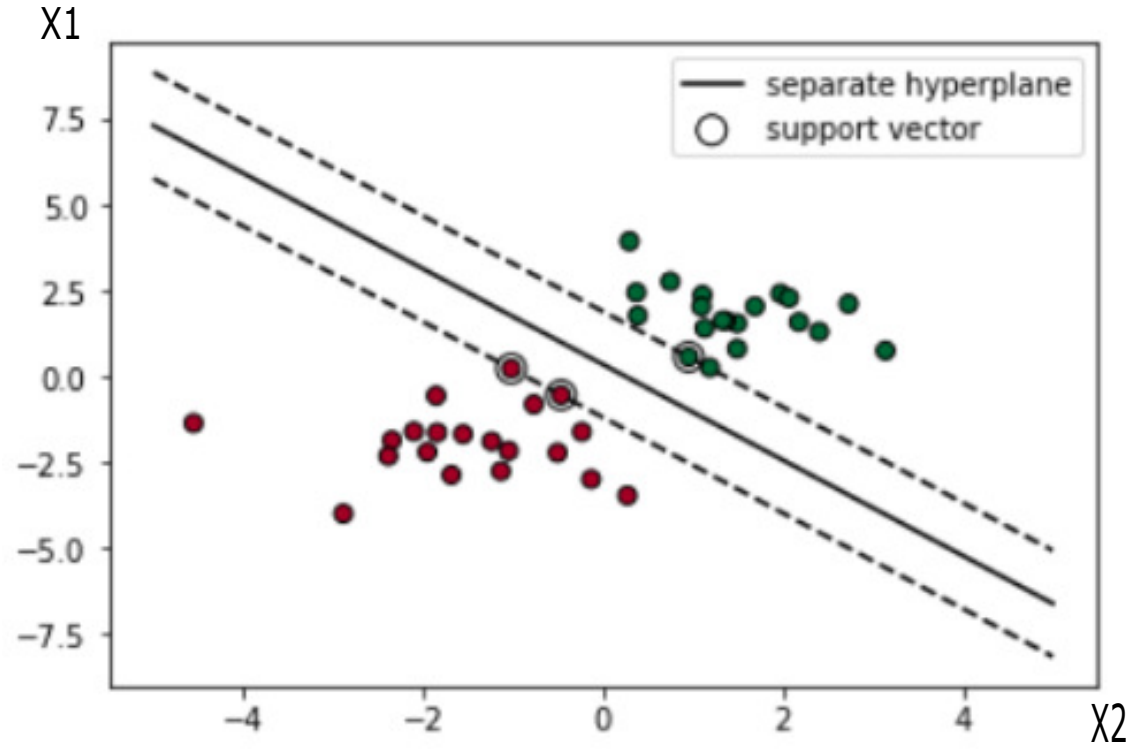


Figure 2.10: Geometric representation of the SVM

the data points that are closest to the hyper-plane. Identifying the support vectors is crucial in order for the SVM to be able to plot the two margins that can be seen as dashed lines in the figure. In other words, the distance between the hyper-plane and the nearest support vectors from either labels, are known as margins. Therefore, the hyper plane will be the greatest possible margin to any point within the training set, thereby giving a greater chance of new data being classified correctly. Hyper plane is an (n-1) subspace for an n-dimensional space. If we have a 2-dimension space (by space we mean features or the number of input data), its hyper-plane will be 1-dimension, which is just a line. if it is a 3-dimension space, its hyperplane will be 2-dimension. For the case of 1D hyperplane, its mathematical representation will be express as so:

$$\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 = 0 \tag{2.5}$$

From Figure 2.5, the β are coefficient that are calculated in order to get the best hyper-plane. In our previous example, once the hyper-plane is created the SVM will label new data as follows:

- For the new data point, by using its features X1, X2 and inserting them in Eq.(2.5). If the result is positive then the label will be green (1);
- Otherwise, if the result is negative it will be red (-1).

Note that in some cases the labels cannot be separated by a line and, therefore, the hyper-plane might therefore be a none linear.

K-nearest neighbor (KNN)

The KNN is another kind of supervised learning algorithm that can be used for both classification and regression. Unlike the previous algorithms, the KNN is a

non-parametric kind of supervised learning which uses the training set in order to classify new inputs based on the similarities in the training data [4]. Hence, the prediction of a new input data is made by comparing it with data that was used for training in order to find the k most similar cases and summarize the output variable for those k cases. We will clarify with a simple example. Imagine that we have a training set that contains the weight and height of a customer and outputs is what T-shirt size they wear. The data array is shown in Table 2.1.

Table 2.1: Training data set [4]

Weight (kg)	Height (cm)	T-Shirt Size
58	158	Medium
59	158	Medium
63	160	Medium
59	160	Medium
60	160	Medium
60	163	Medium
62	168	Large
63	168	Large
66	168	Large
63	170	Large
64	170	Large
68	170	Large

The first thing the KNN algorithm will do is to use the data in order to calculate the similarities by using the distance function. There are many types of distance functions that can be used in order to calculate similarities, the most common one is the Euclidean distance which can be expressed as:

$$d(x, y) = \sqrt{(\sum_{i=1}^{m}(x_i - y_i)^2)} \tag{2.6}$$

d(x,y) will be the euclidean distance between a point or a vector x (which usualy is the first observation in the data) and y (another observation in the data). With the Eq.(2.6), the algorithm will be able to find similarities between new sample and training cases and then finds the k-closest customers to new customer in terms of height and weight. When a new customer with a height of 161 cm and weights 61 kg and by using Eq.2.6, the Euclidean distance between first observation in the table and new observation is:

$$\sqrt{((161 - 158)^2 + (61 - 58)^2)} = 4.24 \tag{2.7}$$

We will then calculate the distance of all the training cases with the new case and calculate the rank in terms of distance. The smallest distance value will be ranked 1 and considered as nearest neighbor. The second step if finding the k nearest neighbor. Suppose we put k=5. This means that the algorithm will search for the 5 customers closes to the new customer and see in which categories they are in. If four out of five of the customers have medium size then the algorithm will label the new customer as a medium size. Figure 2.11 shows a graphical representation of

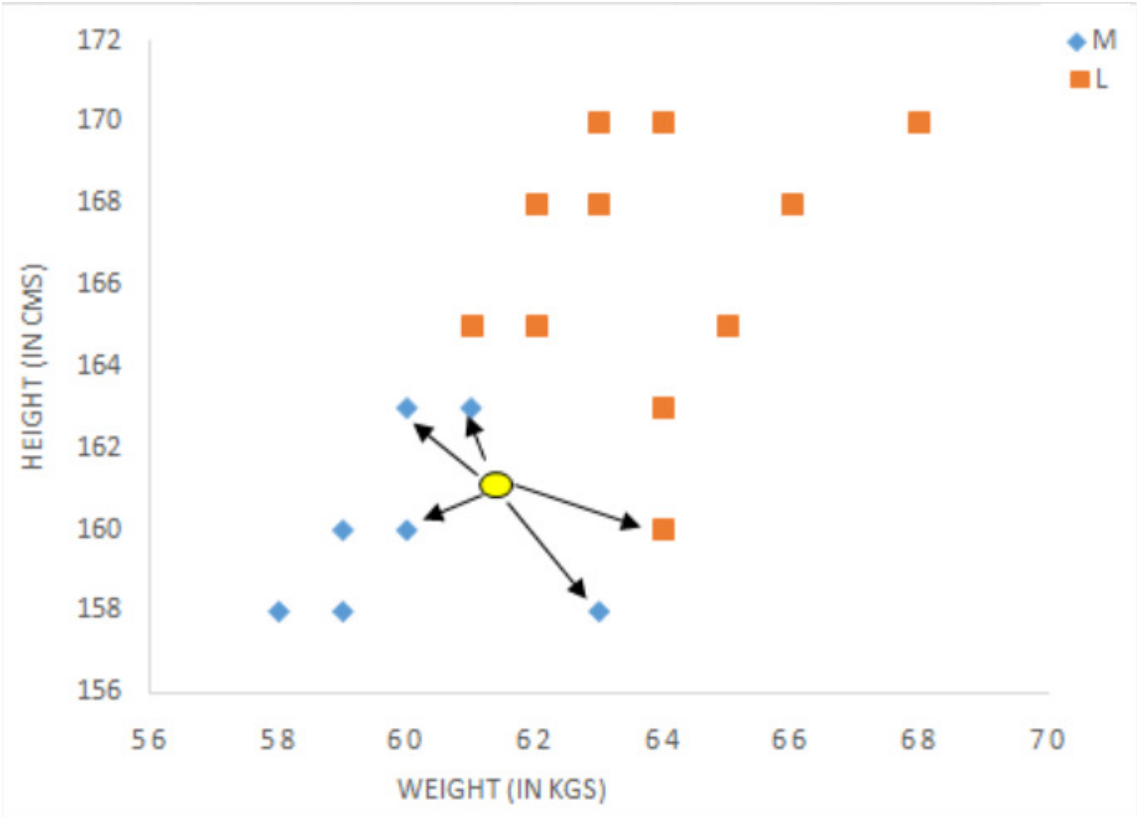


Figure 2.11: Geometric representation of the KNN [4]

the KNN. The y axis is the features representing the height, while the x features representing the weight. The labels are colored. For the Large size it is represented as orange while blue for the medium size. The circled yellow is the new customer, and as can be seen from the Figure four blue highlighted data points and one orange highlighted data point are close to yellow circle. Therefore, the prediction for the new case is blue highlighted data point which is medium T-shirt size.

Ensemble classifiers

Lets say that you want to buy a car, you'll probably would not walk to the first car shop and buy a car immediately based on the dealer's advice, but you would rather search websites and observe people's comment on cars and prices. You would also prefer to hear suggestion from friends. Thus, you will not jump to conclusion but you make decision based on other peoples opinion. The ensemble has the same behaviour[22]. The way it works is that it takes different type of models, SVM, KNN or decision tree and then calculate the overall accuracy. The one with the best accuracy, will be the model that is chosen by the ensemble. Hence, ensemble method combine multiple ML techniques in order to improve the prediction. In most cases the ensemble are preferably used when we have an unbalanced training data (when the number of the different labels are not equal).

2.2.5 Principal component analysis (PCA)

Suppose there is a dataset in order to be used for training the ML algorithm. The dataset contains features and what the output response should be for every set of features. If the size of the data is large (if we have to many features) many problems could occur such as over-fitting and taking long time for the ML to predict a response. The long duration of the prediction exists because the multiple features in the dataset is modeled at a high dimensionality [5]. By dimensionality, we mean the number of features (the number of columns present in the training data). The PCA counteracts these problems by reducing the dimensionality of the data while maintaining almost the same important information that exists in the data. In other words, PCA removes the least beneficial feature that improves in the distinction between labels. Before explaining how the PCA works, some mathematical formulas need to be explained.

Orthogonal: Two features are considered to be ortogonal if they are uncorrelated. Eigenvectors and Eigenvalues: Any vector plotted in an XY chart has a direction. A vector is called eigenvector if a transformation is applied on it (such as multiplying it by a scalar) and does not affect its direction. Hence, eigenvector is a non-zero vector that changes by only a scalar factor when a linear transformation is applied to it. Eigenvectors are the main tools for reducing the dimensionality in PCA. They remove features that have a strong correlation between each other and also help in reducing over-fitting.

Covariance matrix: Covariance provide the level of relationship between two variables.

The steps of the PCA are as follows:

1. Normalization of the data: The purpose of this first step, is to standardize the range of the input features so that each has an equal contribution to the analysis, before using the data to calculate the covariance and the eignevalues. To clarify, if one feature varies between 0 and 100 while another varies between 0 and 1 the feature with a range between 0 and 100 will dominate the other during the calculation of the eignevalues resulting in a bad evaluation from the PCA. The standardization can be done on each value in a feature vector by subtracting the mean value of the feature vector and then divide it by the standard deviation of the feature vector. Eq.(2.8) shows the expression of the normalization.
$$z = \frac{x - mean}{standarddeviation} \tag{2.8}$$
When the standardization is done on every feature, the features will be set to the same scale.

2. Covariance calculation: When the standardization is complete, the covariance matrix will be created. The covariance matrix is needed in order to calculate the relationship level between the features. Highly correlated features usually have unimportant information. Hence by creating the covariance matrix, it will identify the correlations. To understand how the matrix would look like, if we have a 2 features X1 and X2, the covariance matrix will have a size of [2x2]. the matrix below represent the covariance matrix in case of 2 features:

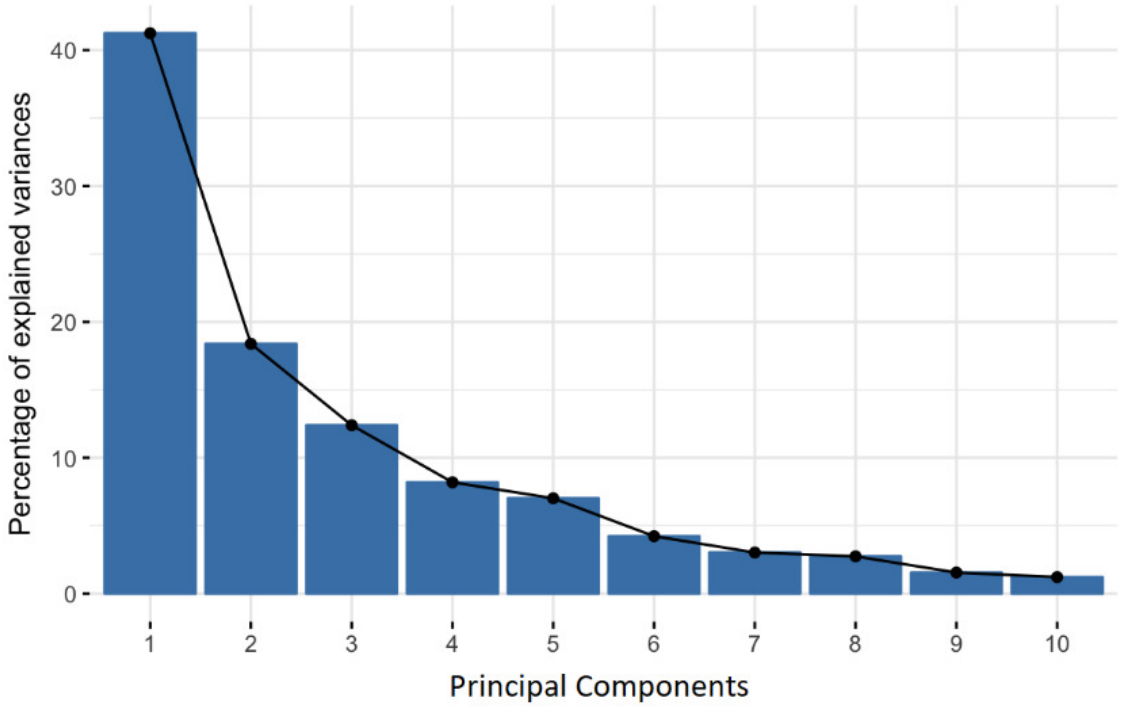


Figure 2.12: Percentage of information in each pc [5]

$$\begin{bmatrix} Cov(X1,X1) & Cov(X1,X2) \\ Cov(X2,X1) & Cov(X2,X2) \end{bmatrix}$$

However, the main importance of the covariance is the sign. If the covariance is positive, then both features are proportional. Otherwise, if the covariacne is negative then the feature will be inversely proportional.

3. Calculate the eigenvector and eigenvalues of the covariance matrix: From the covariance matrix, the eigenvalue and eigenvectors are calculated in order to get the principal components, which are new variables. These new principal components are uncorrelated and the information in the initial feature from the training data are "squeezed" into the first PC. The PCA will insert as much information as possible from the main data into the first component. Next step is to distribute the remaining information in the second and so on. Figure 2.12 shows an example of how the information is distributed across the PCs

4. Ranking the eigenvalues: In this step the PCs are ranked based on the amount of information exist in each PC. Hence, the PCs are ranked from high to low.

5. Select the number of component, to keep: This step involves selecting the number of PCs to be used as new training data. For example, if we look back at the Figure 2.12 above, if the user decide to use 5 PCs, then the user will only be using the 5 first PCs as input to the ML.

6. Create the new feature vector: After selecting the number of PCs to keep, a feature vector that is a matrix which has columns, that contain the eigenvectors of the components that the user decided to keep.

7. Recasting the data along the principal components axes: So far we selected the number PCs to use and formed a vector from it. However, the input data is still in terms of the inital features. Hence, in the final step, the feature vector that was formed using the eigenvectors is used to reorient the data from the initial input to the ones that is represented by the principal components. This is done by applying the following formula:

$$FinalData = Transpose(FeatureVector) \cdot Transpose(NormalizedOriginalData) \tag{2.9}$$

2.3 Issues regarding the collection of data from sensors

In practice, whenever we collect data from a sensor, there are two main principles to keep in mind before creating the training data. They are the sampling rate and de-noising of the data.

Sampling rate

The sampling rate is how fast we are collecting the data from the sensors [6]. Both high and low sampling rate have advantages and disadvantages in ML. For the case of high sampling, the training data will be large, resulting in more effort for training the ML and longer time to provide a response whenever a new input is given to the trained ML algorithm. However, the data we collected will provide more information which could result in increase of accuracy of the ML. On the other hand, low sampling results in a smaller data size and, therefore, quicker training of the ML and faster response on the new test data. However, with low sampling rate we might not collect all the important information, which could result in loss of accuracy in the ML.

Figure 2.13 shows two different results of sampling. If we have a sinusoidal waveform having frequency of 60 Hz and every 1ms (or 1000 Hz) we sample the signal, we would have retrieved a data that that is similar to the analog sinusoidal waveform as can be observed from the white circles in Figure 2.13. If we reduce the sampling rate to 20ms(or 50 Hz), the collection of the data will be the one that is colored in red in the picture. So if we compare calculation of features such as minimum, maximum points or the mean, then the fast sampling will be more accurate than the low sampling rate. The perfect sampling rate, when it comes to ML, is the one that gives a great distinctive information between the classes while using the smallest amount of data, to provide fast and accurate response.

Pre-processing data

For the case of noise, whenever we collect data from any sensor, the data has to be pre-processed in order to reduce the noise before using it as training data for the ML. Hence, when we want to clean the data, a filter must be applied before collecting the data. Filters are blocks that process signals in the frequency domain [23]. They allow certain frequency to pass through while rejecting others by using cut-off frequencies. There are different classes of filters. The low and band-pass

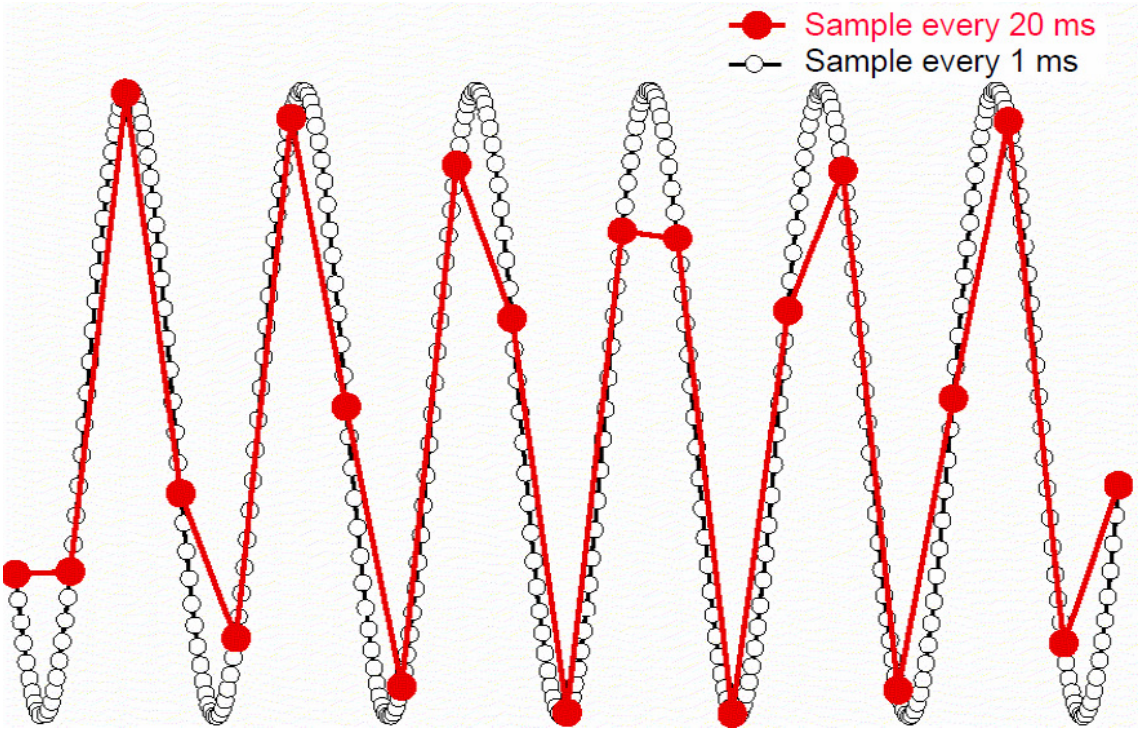


Figure 2.13: Different Sampling speed for data retrieval [6]

filters can be used in order to reduce the noise in the signal. The Figure 2.14 shows how the ideal transmission of a band-pass and low-pass filters. The band pass filter is also used in order to see if the perturbation can still be visible, so that it can be used for training the ML.

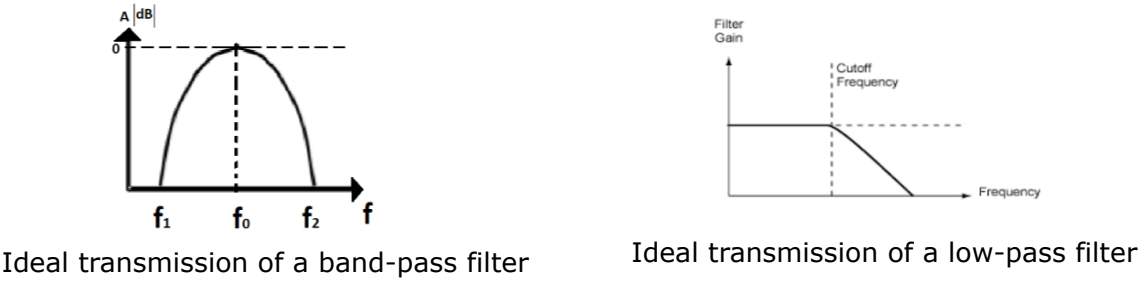


Figure 2.14: Band-pass and low-pass filters

By the selecting the middle frequency f_0, the high and the low frequencies f_1 and f_2, the band-pass filter will only allow frequencies between f_1 and f_2 to pass and will attenuate the others and thereby reducing the noise in the signal. As for the low-pass filter, it is used to allow only low frequencies to pass will cutting off other frequencies at the cut-off frequency.

2.4 Summary

This chapter started with the discussion of the sensitivity analysis and why are we going to need it in the project. The chapter also has provided a clear and simple explanation of the basic concept of the different type of classification algorithm that are going to be used along with the PCA. Finally, the importance of a clean data retrieval and the pros and cons of high and low sampled data have been discussed. In the next chapter we will discuss how we created our test circuit and which steps have been made in order to train the ML algorithms.

Chapter 3

ML applied on Buck converter

This chapter will explain how we used the Buck converter in Simulink to extract data for training. It will also show how perturbation was created in the PWM before collecting data and then how the data was structured. Moreover, an explanation on how noise influence the results will be provided.

3.1 Test circuit

Figure 3.1 shows the implemented Buck converter while Table 3.1 reports the new data for the converter. The same model has been implemented in Simulink. The

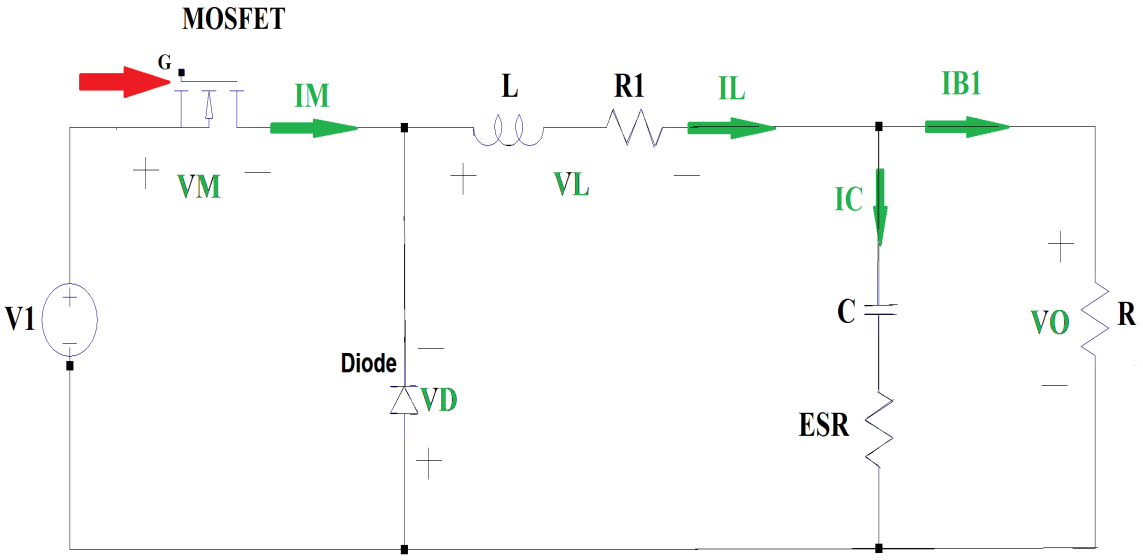


Figure 3.1: Buck converter circuit

green arrows and green colored text in the circuit represent the voltages and currents collected from the Buck converter circuit. In Simulink, They are blocks that are used in order to export the values of sensors (both voltage and current) to the workspace at a given sampling rate during parametric sweep. The sensors are used to collect the following signals:

- MOSFET Voltage and Current (VM an IM);
- Diode Voltage (VD);

- Inductor Voltage and Current (VL and IL);
- Capacitor Current (IC);
- Current between inductor and output voltage (IB1);
- Output Voltage (VO).

Table 3.1: Value of components of the Buck converter

Parameter	Value
V1	15 V
L	103.1 μH
R1	0.5 Ω
V_f	1 V
Rd	1 mΩ
C	680μF
ESR	0.39 Ω
R	10 Ω
D	0.5
Sampling Speed	1000 Hz

Note that for the training of the ML, we are assuming that there is no noise in the signals. However, we will demonstrate how noise can be filtered by using band-pass filters. The red arrow in the circuit is pointing at the gate of the MOSFET, where a PWM was applied with three different types of reference signals (one without a fault perturbation and two with different kind of fault perturbation) that can be seen in the Figures 3.2, 3.3 and 3.4. The idea is to show, later on, that injecting a very small perturbation signal can affect the performance of the machines(Tree, SVM, KNN and ensemble algorithms).

Figures 3.2 to 3.4 show the PWM with different types of reference signals that were used during the simulations. The first one in Figure 3.2 shows a block for designing a normal PWM, where no perturbation is used. The block consist of a triangular waveform whose max and min are 1 and -1. Hence, in order to change the max and min value of the triangular waveform, it is added with 1 and then divided by 2 so the min and max are set to 0 and 1. After that it is compared with a constant value (the duty cycle), in order output a value of 1 if it is higher or equal than 0.5 or 0 if it is lower. The plots (A) and (B) show the constant value and the output response of the PWM. As can be seen, the pulse width are the same size when the triangular waveform is lower and higher than the duty cycle.

The second figure (Figure 3.3) shows the PWM design block when a perturbation of a sinusoidal shape takes place. The additional blocks is used, in order to add a temporary sinusoidal perturbation. The sinusoidal frequency is randomly set to 200 Hz (in order to see if it has an affect on the ML algorithm) with an amplitude of 5% and will only last for 0.01 seconds (between 0.2s and 0.21s). The plots (A) and (B) show the scopes of the injected fault and the output response of the PWM. From plot (B), we can see that there is basically no influence on the width of the PWMs and, therefore, it will not affect the overall performance of the Buck converter.

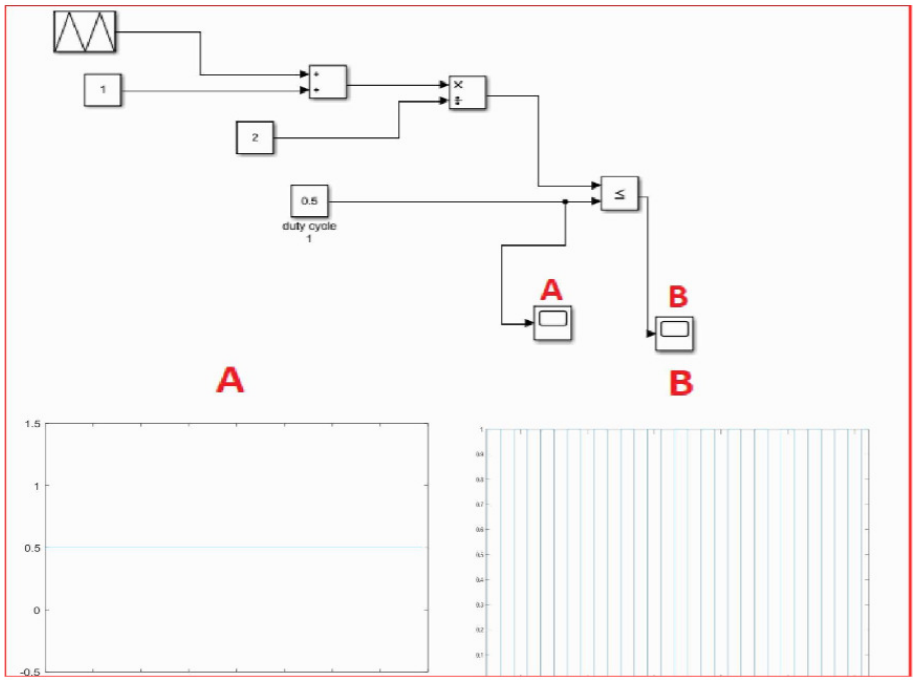


Figure 3.2: PWM with no perturbation

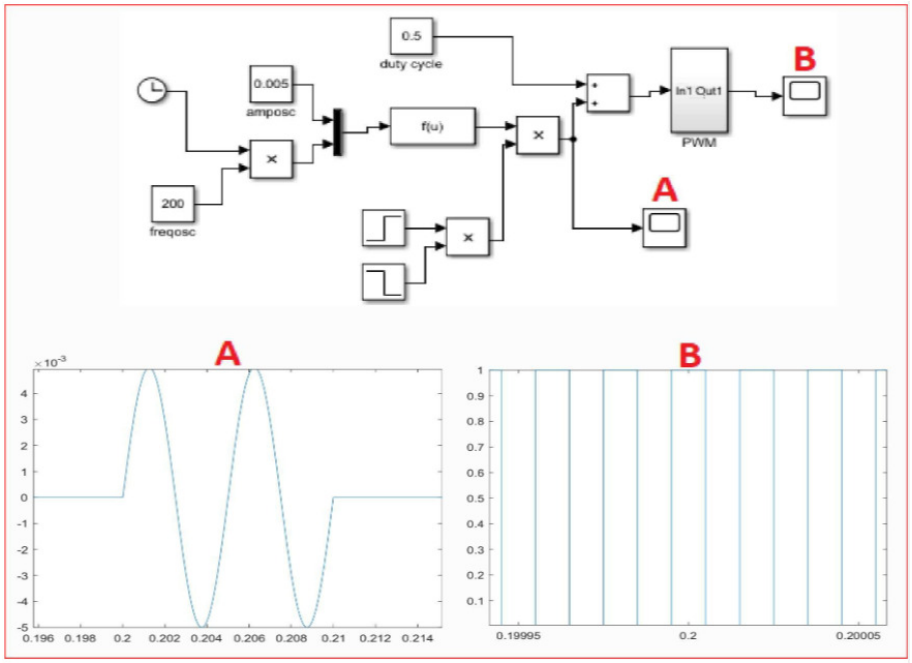


Figure 3.3: PWM with sinusoidal perturbation

The third plot in Figure 3.4 shows another type of perturbation injected in the PWM. Instead of injecting a sinusoidal waveform, a step shape is used. The perturbation amplitude is set to 5% and it is injected at 0.2 s and goes back to 0 at 0.21s as can be seen from the Figure (A). Similar to previous PWM, because the perturbation value and duration is small, it has little affect on the output response of the PWM as can be seen from Figure (B).

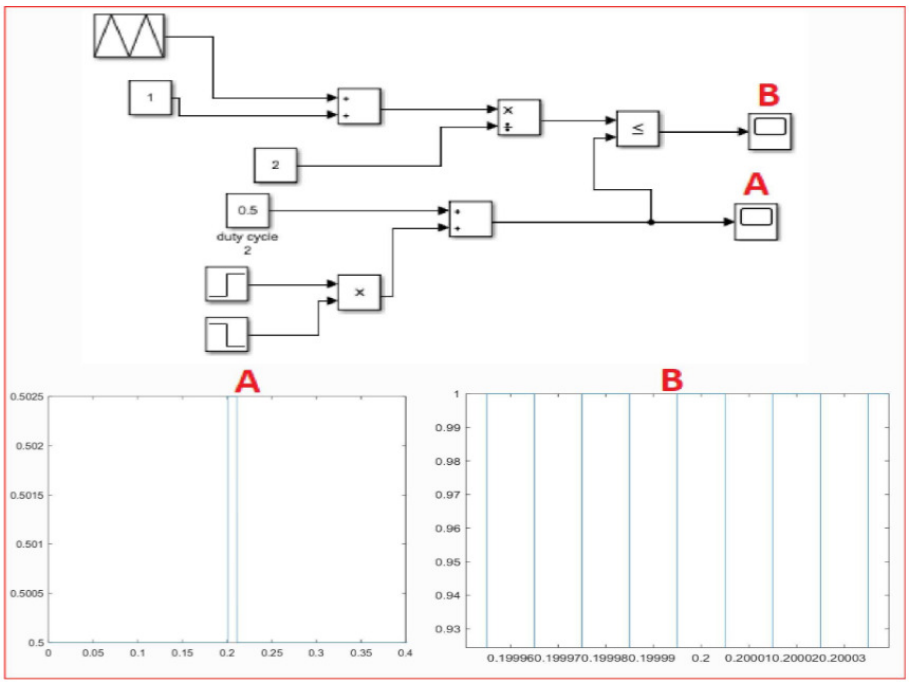


Figure 3.4: PWM with step perturbation

3.2 Analysing sensors

For the the parametric sweep, the components that were changed were inductance, capacitance, MOSFET on resistance and diode forward voltage. These components were varied 40 times for the case of no PWM perturbation and with perturbation used in the PWM and then data from the sensors were collected. The following figures (From Figure 3.5 to Figure 3.12) show the data collected from sensors with a sampling rate of 1000 Hz for different PWM conditions. The plots with label (A) shows the plot for the case of No perturbation used in the PWM, while the (B) and (C) shows the plots when sinusoidal and step perturbations were used.

From the figures, we can see that in case of the sinusoidal injection, for a certain value of components the sinusoidal shape seems, for different values of C,L,Vf and Rm starts to disappear. This also can be seen in the square shape perturbation case in the plots labeled (C). However, for the square perturbation, the shape is less clearer than that of the sinusoidal shape. Hence, adding a small perturbation to the circuit can be useful, since it can provide a good distinction in the signals whenever we vary a component. Note that in the figures, the values of a component that was varied during simulation, is not of importance since we only want to observe if there is a difference when using a fault perturbation. Moreover, signals alone cannot be the only features for the training data. Other signal properties could be used as features for the training data. Hence, sensitivity analysis had to be done in order to see how properties of a signal changes whenever a component's value has changed.

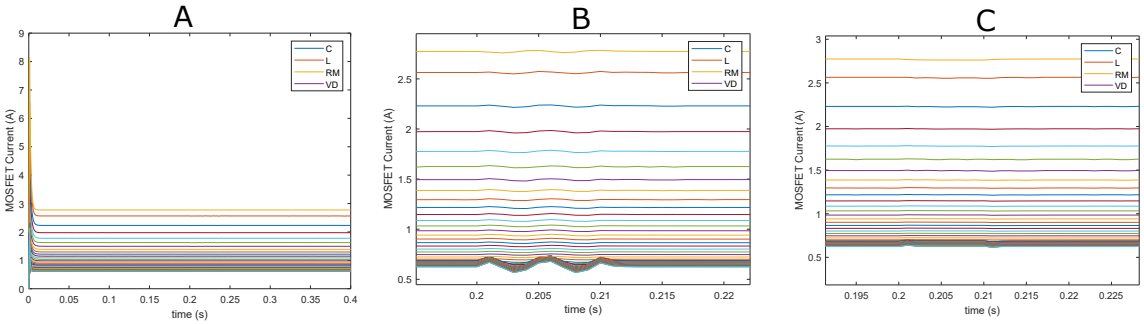


Figure 3.5: MOSFET Current

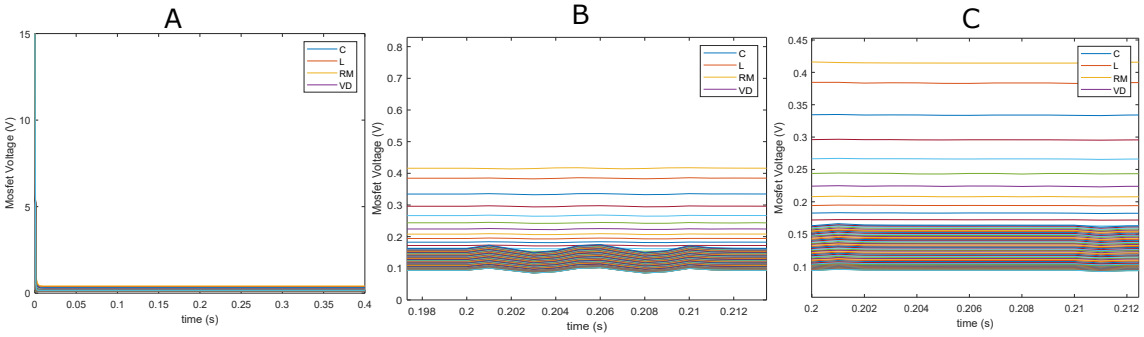


Figure 3.6: MOSFET Voltage

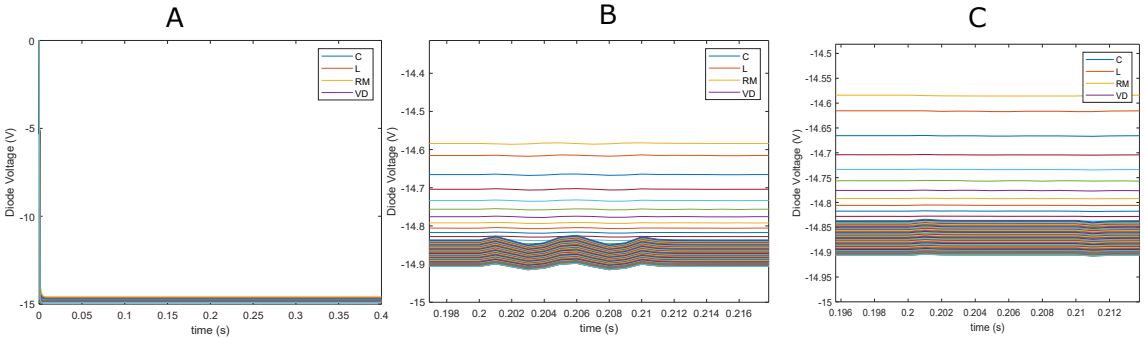


Figure 3.7: Diode Voltage

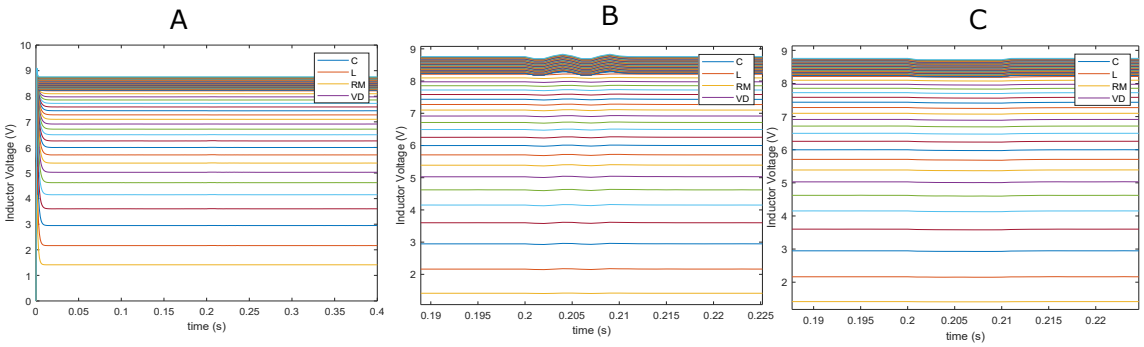


Figure 3.8: Inductor Voltage

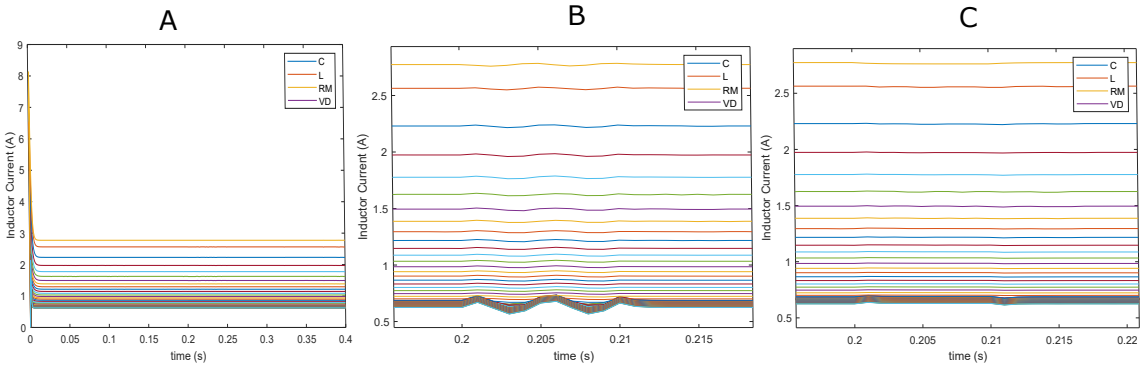


Figure 3.9: Inductor Current

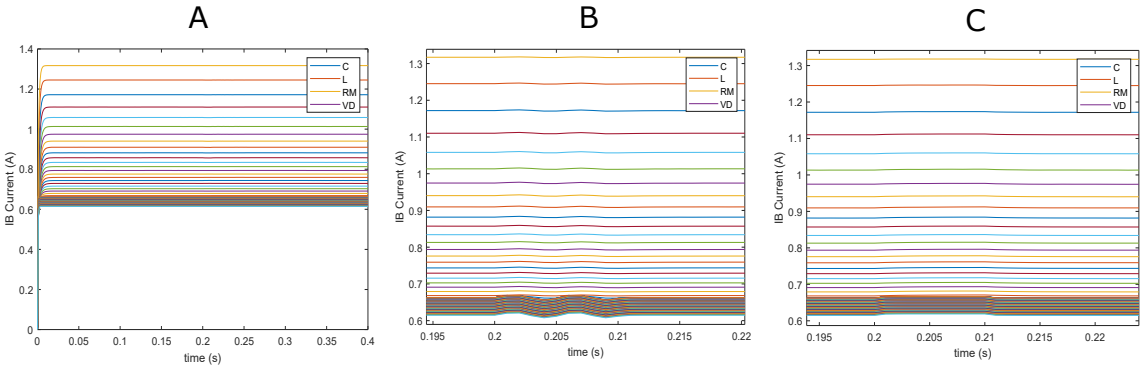


Figure 3.10: IB

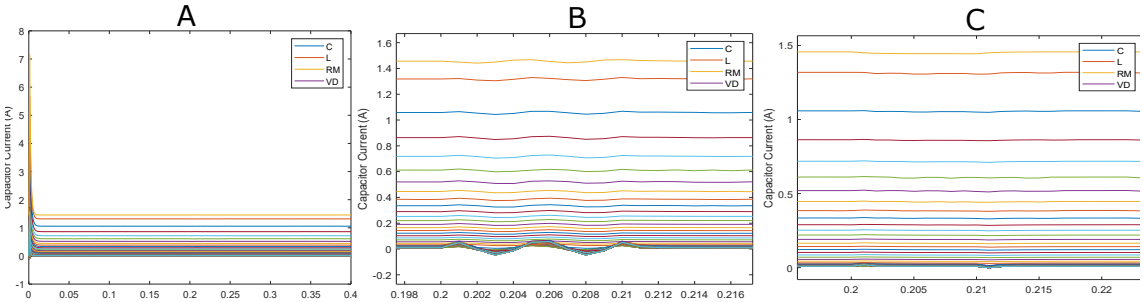


Figure 3.11: Capacitor Current

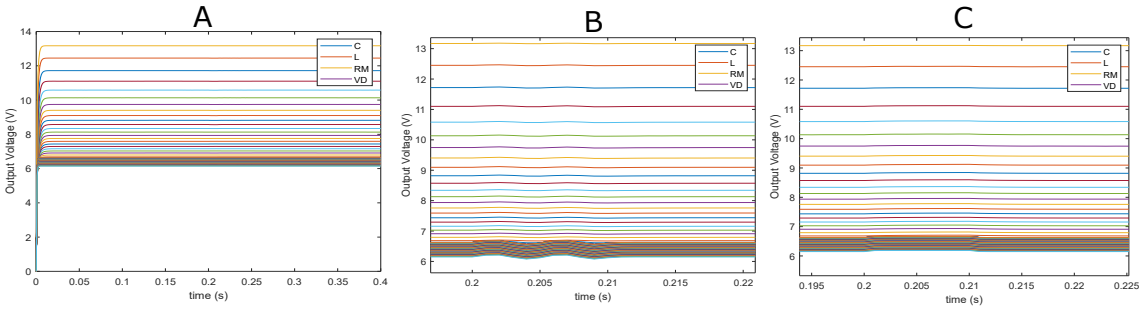


Figure 3.12: Output Voltage

3.3 Applying sensitivity analysis

For each signal collected we took their properties and applied sensitivity analysis on them. The signal properties that are considered are the root mean square, the

variance, the minimum, the maximum, the mean, the median and the standard deviation of the signal. After selecting the properties of each signal, 10 random samples of C,L,Rm and Vf are chosen for applying sensitivity analysis. The values of the 10 samples are shown in the Table 3.2. Note that for each sample, only

Table 3.2: parameter set

C[μF]	L[μH]	R[Ω]	VD[V]
722.8	96.03	0.154	1.041
735.1	112.8	0.136	0.906
629.2	112.5	0.160	0.955
736.2	102.7	0.163	0.909
698	109.24	0.155	0.919
625.2	95.71	0.157	1.064
649.8	101.4	0.157	1.038
686.3	111.6	0.146	0.963
742.2	109.1	0.154	1.090
743.26	112.5	0.140	0.906

one component is considered. Thus, we only apply sensitivity for one component variation at a time. Figures below show the results of the sensitivity analysis on the signal properties of inductor current (IL), IB1, diode voltage(VD), inductor Voltage (VL), output voltage (VO), capacitor current (IC), MOSFET voltage (VM) and MOSFET current (IM). The x axis in each plot shows how much the signal properties are influenced by the each parameter on the y axis. If it is negative, it means that it is inversely proportional to the parameter change otherwise it will be proportional. The plots are ranked from the component in which the property is mostly sensitive, to the weakest.

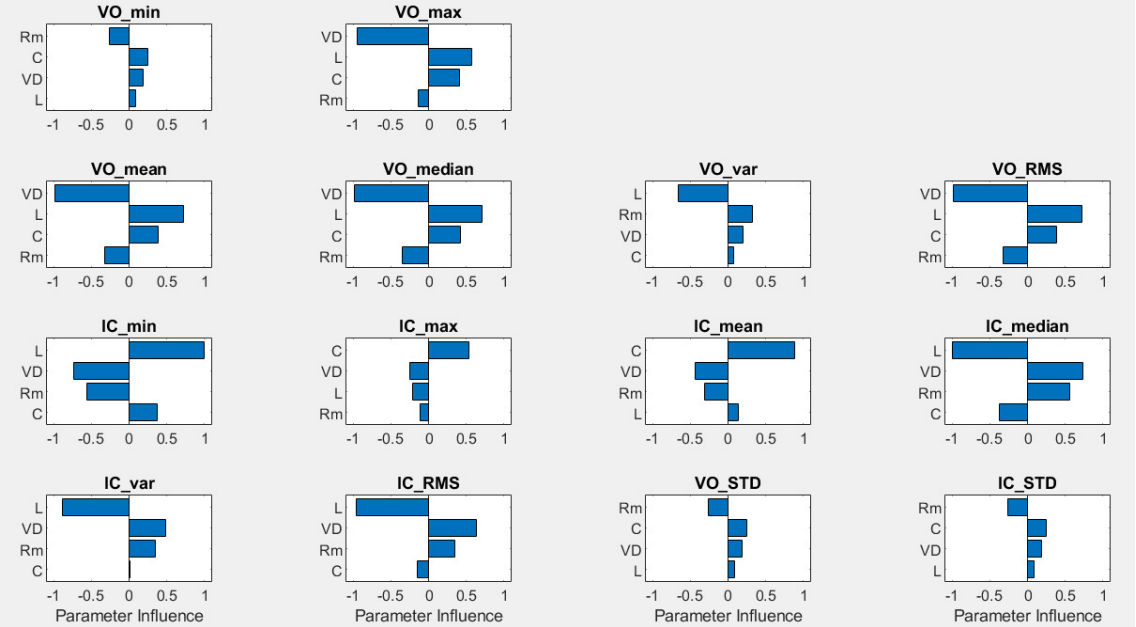


Figure 3.13: Sensitivity results of capacitor current and output voltage

Output voltage (VO) and capacitor current (IC)

- From Figure 3.13, we can see that the minimum values of VO and IC (VO_min and IC_min) are highly influenced by the Rm and L and least influenced by L and C . For the VO_min the correlation is close to -0.4 while for the IC_min the correlation is 1.
- The maximum values of VO and IC (VO_max and IC_max) are highly influenced by diode forward voltage variations and C while least influenced by Rm. The correlation between the VO_max and VD is -1 while the correlation of IC_max and C is close to 0.5.
- The mean values of VO and IC (VO_mean and IC_mean) are greatly affected by VD and C while Rm and L has the least affect on VO and IC. The correlation between the VO_mean and VD is -1 and the correlation between IC_max and C is approximately 0.5.
- The median values of VO and IC (VO_median and IC_median) are very sensitive to variation of VD and L and least sensitive to variation of Rm and C. The VO_median sensitivity value to VD variation is -1 which is same for the case of IC_median.
- the variance values of the signals VO and IC (VO_var and IC_var) are greatly affected by L while the parameter C has little affect on the two signals. The correlation for the VO_var case is close to -0.6 while for the case of IC_var, it is -1.
- The RMS values of VO and IC (VO_RMS and IC_RMS) are very correlated to the variation of VD and L and low correlation to the variation of Rm and C.
- The STD values of VO and IC (VO_RMS and IC_RMS) are mostly affected by the variation of Rm and less affected by the variation of Rm and C.

Inductor current and IB current

From the inductor and IB currents (IL, IB):

- From Figure 3.14, the minimum values of IL and IB (IL_min and IB_min) are highly influenced by the Rm and least influenced by the L. The correlation value for IL_min and IB_min is 0.4.
- The maximum values of IL and IB (IL_max and IB_max) are highly influenced by C and VD while least influenced by Rm. For the IL_max it is 0.5 and for the IB_max it is -1.
- The mean values of IL and IB (IL_mean and IB_mean)are greatly affected by VD while Rm has the least affect on both.
- The median values of IL and IB (IL_median and IB_median) are very sensitive to variation of VD and least sensitive to variation of Rm.

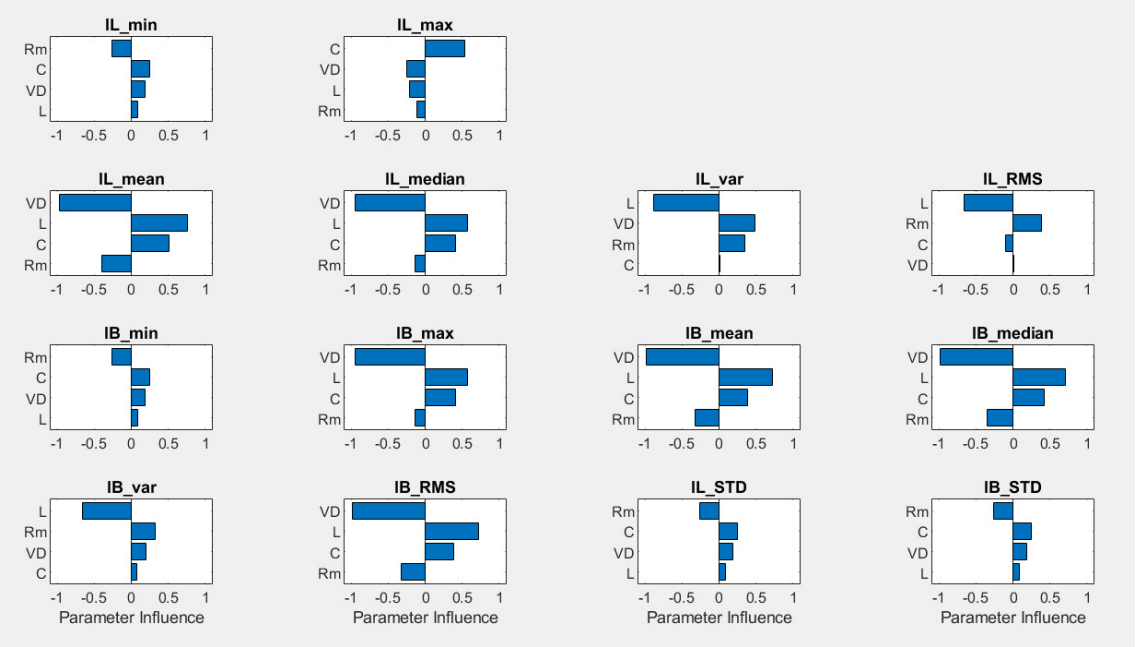


Figure 3.14: Sensitivity results of inductor current and IB1

- The variance values of the signal IL and IB (IL_var and IB_var) are greatly affected by L while the parameter C has almost no affect on the two signals.
- The RMS values of IL and IB (IL_RMS and IB_RMS) are very correlated to the variation of L and VD and low correlation to the variation of VD and Rm.
- The STD values of IL and IB (IL_STD and IB_STD) are most affected by the variation of Rm and less affected by the variation of L.

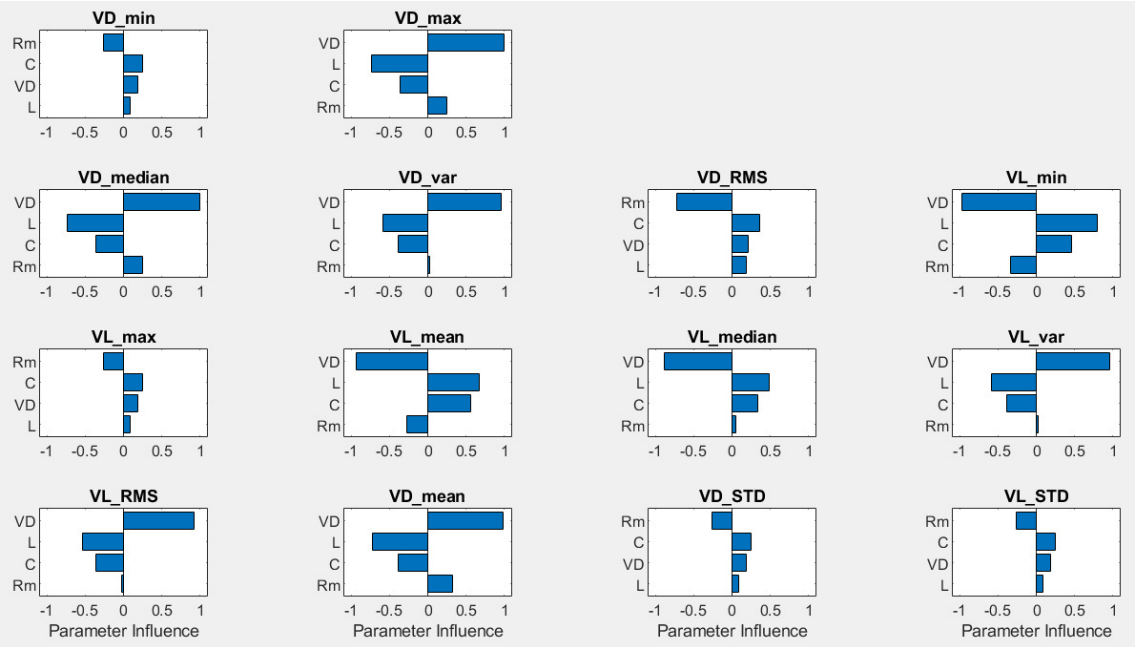


Figure 3.15: Sensitivity results of diode voltage and inductor voltage

Diode voltage (VD) and inductor voltage (VL)

- From Figure 3.15, the minimum values of VD and VL (VD_min and VL_min) are highly influenced by the Rm and L and least influenced by L and Rm . The correlation for the case of VD_min is 0.4 while for the case of VL_min it is -1.
- The maximum values of VD and VL (VD_max and VL_max) are highly influenced by diode forward voltage variations and Rm while the VD_max and VL_max are least influenced by Rm and L.
- The mean values of VD and VL (VD_mean and VL_mean) are greatly affected by VD while Rm has the least affect on both.
- The median values of VD and VL (VD_median and VL_median) are very sensitive to variation of VD and least sensitive to variation of Rm.
- The variance values of the signals VD and VL (VD_var and VL_var) are greatly affected by VD while the parameter Rm has little affect on the two signals.
- The RMS values of VD and VL (VD_RMS and VL_RMS) are very correlated to the variation of Rm and VD and low correlation to the variation of L and Rm.
- The STD values of VD and VL (VD_STD and VL_STD) are mostly affected by the variation of Rm and VD and less affected by the variation of L.

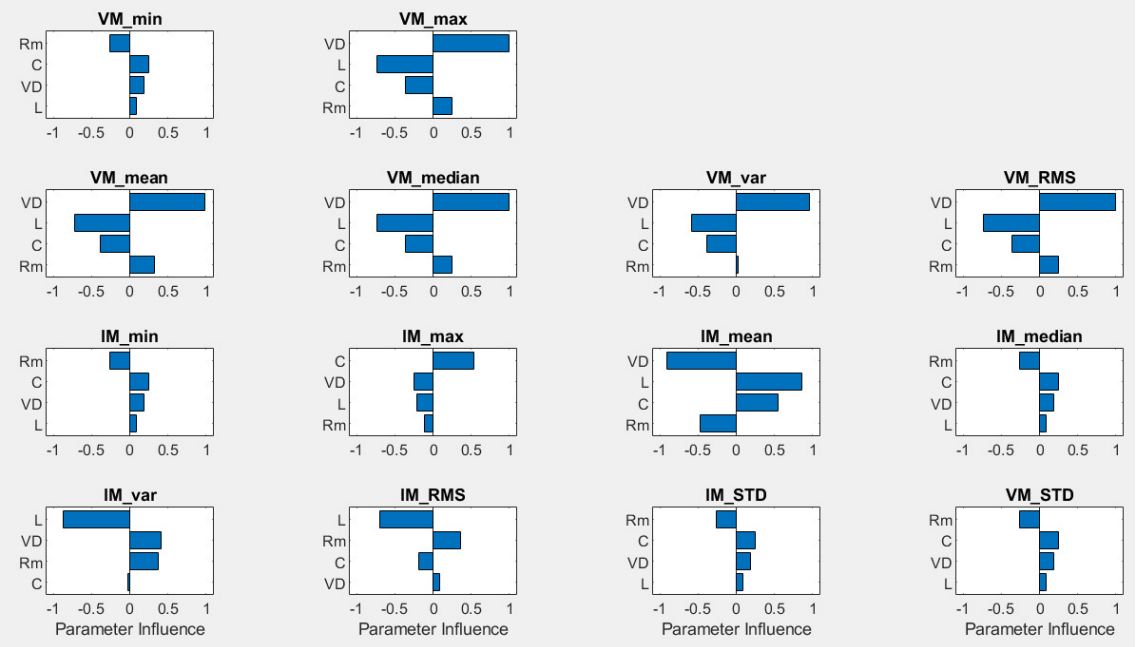


Figure 3.16: Sensitivity results of MOSFET voltage and MOSFET current

MOSFET voltage (VM) and MOSFET current (IM)

- From Figure 3.16, the minimum values of VM and IM (VM_min and IM_min) are highly influenced by the Rm and least influenced by L.
- The maximum values of VM and IM (VM_max and IM_max) are highly influenced by diode forward voltage variations and C while least influenced by Rm.
- The mean values of VM and IM (VM_mean and IM_mean) are greatly affected by VD while Rm has the least affect on VM and IM.
- The median values of VM and IM (VM_median and IM_median) are very sensitive to variation of VD and Rm and least sensitive to variation of Rm and L.
- The variance values of the signals VM and IM (VM_var and IM_var) are greatly affected by VD and L while the parameter Rm and C has little affect on VM and IM.
- The RMS values of VM and IM (VM_RMS and IM_RMS) are very correlated to the variation of VD and L and low correlation to the variation of Rm and VD.
- The STD values of VM and IM (VM_STD and IM_STD) are mostly affected by the variation of Rm and less affected by the variation of L.

Based on these results we can conclude that all the properties (min, max, mean, median, var and RMS) can also be used as features due to the fact that they are influencing the signals. Even for the ones that are less influential, this can be considered as good inputs for the ML algorithm in which it will be able to distinct between which component is degrading. For example assuming that we have the output voltage and its properties as inputs to the machine. If the inductor is degrading, then the variance will have a huge variation compare to rest of the signals properties, and from that the ML can predict that it is the inductor that is degrading. The standard deviation is not considered since it has the same affect for different component variation in every current and voltage sensors and will not provide a good information during the training of the machines. Hence, the variance, RMS, min, max, mean and median are used as inputs along with the small part of a signal.

3.4 Structuring data

After completing the sensitivity analysis and selecting the features, we need to create the data to be used for the training. In the previous section, a PWM with three different reference signal were used in order to extract the signals from the sensors at a certain sampling rate. This is also done in order to collect data for the three different cases (when there is no perturbation in the reference signal and when there is sinusoidal or square perturbation in the reference signal). Hence, for each scenario, a data is structured for creating the training set. The first step is to use all the sensors (sensors which we conducted sensitivity analysis on), in the circuit in order

to create the training set. The charts below show how the data is structured for each signal in case of PWM with normal reference signal and PWM with perturbation in the reference signal. Figure 3.17 represents how each data is structured for each

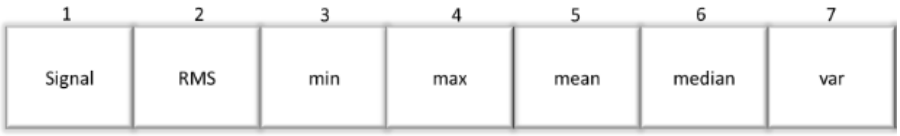


Figure 3.17: Data structuring of one signal

signal. Column 1 involves the retrieval of a part of the signal. For the case of no perturbation in the reference signal of the PWM, the signal is taken from the steady state while for the case of perturbation in the reference signal of the PWM, the signal is taken during the beginning of the fault until the end of it. As for the properties of the signals they are calculated in steady state and collected for both case (perturbation and no perturbation). This kind of structure is done for every signal collected after the parametric sweep. The final data array is created by inserting the new structured data for each signal in the chart 3.18 below.

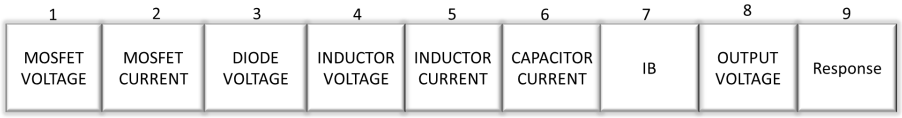


Figure 3.18: Final data

After that, another data which has only the output voltage sensor and features as inputs along with labels. This is done in order to see how each machine performance changed when there is less information in the input. The response in Figure 3.18 represents the labels that are given whenever a component is varied. The labels are numbered from 1 to 8 as seen in Table 3.3.

Table 3.3: Labels

Label	Definition
1	Capacitor is Degrading
2	Inductor is Degrading
3	Diode is Degrading
4	MOSFET is Degrading
5	Capacitor is in critical condition
6	Inductor is in critical condition
7	Diode is in critical condition
8	MOSFET is in critical condition

The selection of the labels in Table 3.3 are as follows:

- In case of capacitor degradation, if the capacitor value is between 680 μF and 588 μF, the machine should give the output a value of 1 to indicate that the capacitor is degrading but it is still in acceptable condition. However, if the capacitance is below 588 μF, then the machine should give the output a value of 5 in order to indicate that the capacitor is in critical condition.

- For the inductor, if the inductance is between 103.1 μH and 85 μH then the ML sets the outputs to the value 2. Below 85 μH it will output a value of 6.
- If the MOSFET Rm has increased, the response should be 4 if it is still lower or equal to 0.1668 Ω otherwise the correct output should be 8.
- For diode forward voltage variation, the response should be 3 if it is still between 1 and 1.179V otherwise the output should be 7.

The selection of the values of components in which the ML indicates whether the component is in critical condition or not is based on the usual standard of deciding whether the component can no longer be used (for the case of capacitor for example, if its value is reduced by almost 20%, the component can no longer be used). Note that once the data is created, only 80% of each label is taken for training and cross validation, where the cross validation is set to 15th fold. The 20% of the different conditions are used as test data (for the test data the labels are removed in order to see if the machines are capable of predicting correctly).

3.5 Training data with load variation

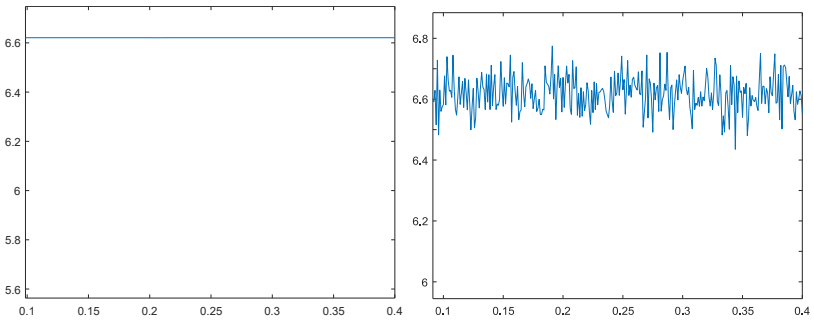
In reality, in every DC/DC converter, the load will not be a constant value, and therefore it can also affect the accuracy of the trained ML algorithm. Hence, a new data set will be added to the previous one by a varying the load (40 samples) during parameter sweep and adding a label 0 to the responses of the machine in order to indicate a normal conditions. The structure of the data is same as before. the load was varied from one to 10 Ω.

3.6 Adding noise to measurements

Another thing we will demonstrate is how noise affects the signal and how it can be filtered. Before exporting the data to the workspace a gaussian white noise block, as seen in the Figure 3.19 below, was used in order to simulate a random noise in the output voltage.

Figure 3.19: White Gaussian noise block

The two Figures 3.20a and 3.20b show the output voltage when there is no noise and when the white Gaussian noise block is added to the signal.

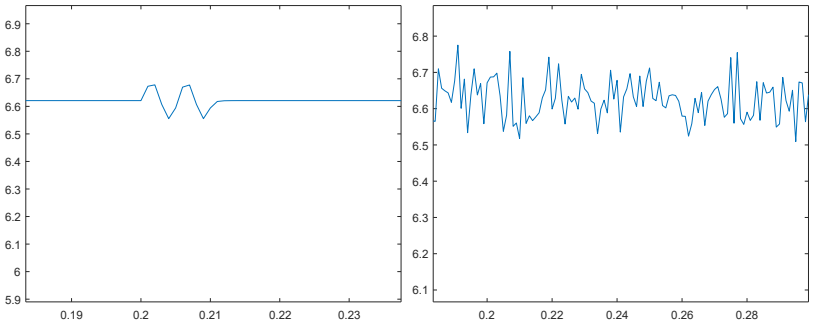


(a) Output voltage without noise

(b) Output voltage with added white Gaussian Noise

Figure 3.20: Output voltage for the case of no fault injected to the PWM

At steady state, the noisy signal will no longer be a straight line and the features that are calculated for condition of no noise will be completely different in case of noise being added to the signal. This also can be seen when their is perturbation in the PWM reference signal.

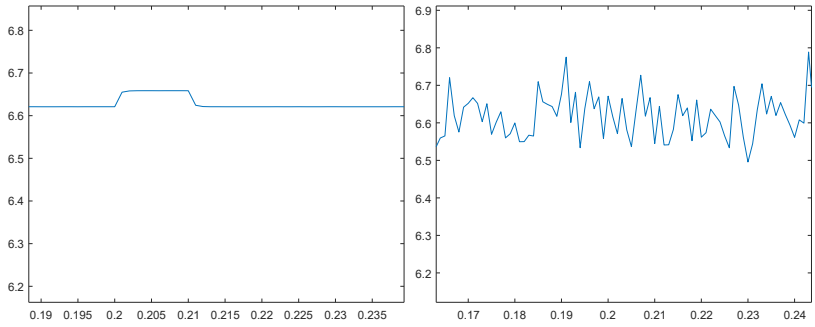


(a) Output voltage without noise

(b) Output voltage with added white Gaussian Noise

Figure 3.21: Output voltage for the case of sinusoidal injected to the PWM

As can be seen, in case of step perturbation or sinusoidal perturbation in Figures 3.21 and 3.22, the shape of the perturbations disappear when noise is applied.



(a) Output voltage without noise

(b) Output voltage with added white Gaussian Noise

Figure 3.22: Output voltage for the case of a step fault injected to the PWM

For the case of no perturbation and step perturbation, a band pass filter is needed in order to observe only the dc components of the signal. However, for the case of sinusoidal fault two band pass filter had to be used in combination, one for the dc component and one for the sinusoidal fault which has a frequency of 200 Hz. This will be shown in the results.

3.7 Summary

After reading this chapter, the reader should be able to understand how the two different data were created. The first data will involve using all sensors that were analysed in the sensitivity analysis section along with their features. The second data will only use the output voltage and its features as input the different machines. This is done for the case of no perturbation and with perturbation. The last thing that was explained in the method is how we added noise to the signal. In the next chapter we will show the confusion plots for different machines and for different scenarios. We will also demonstrate how band pass filter can reduce noise in the signal.

Chapter 4

Results on ML applied on Buck converter

Once the data is created with the labels, 20% of each labels is used as testing set to see how machines are performing. For the labels number '1','2','3' and '4' the number of test sets are 3 while for the labels '0','5','6','7' and '8' the number of test sets are 5 . In this chapter, for each algorithm, we will show the confusion matrix in case of all sensors taken into consideration as inputs. Then we will show the confusion matrix for the case when only the output voltage and its features are taken as inputs. And then we will do it again when we have load variations. The last section will show how the band-pass filter reduces the noise on the output voltage.

4.1 KNN

Before showing the results, it is important to understand how to read the results from the confusion matrix result for the different ML algorithms that were used. Therefore, Figure 4.1 shows the a result of the KNN when no perturbation was applied in the reference signal in the PWM. The y-axis shows the predicted answer of the ML algorithm while the x-axis shows what the actual answer should be. Hence, the circle green classes indicate degradation labels for the different components, while the red circles represent the critical conditions. The encircled green label on the y-axis should match the ones on the x-axis (same for the encircled red case). In order to tell whether they are matching the labels on x-axis or not, one must look at the diagonal of the confusion matrix. As for the colored squares, each one contains a statistical number (the yellow circle) and a number beneath it (the blue circle). The number indicates the number of times for which the ML algorithm outputs this respond while the statistical number indicates the probability for the occurrence of that response.

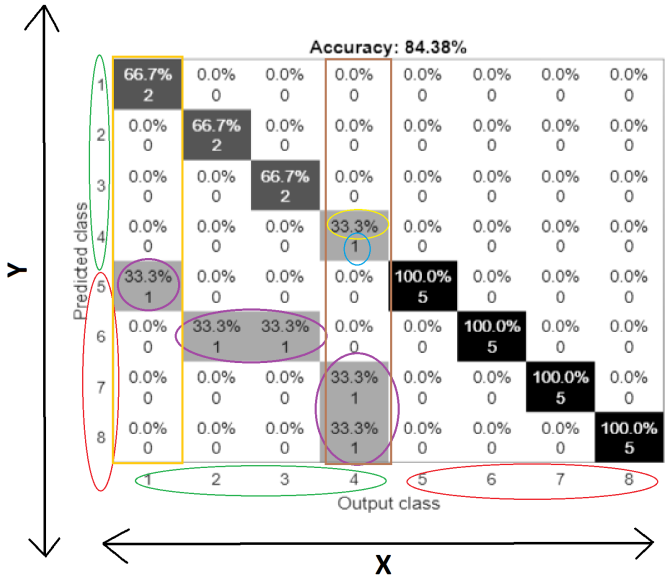
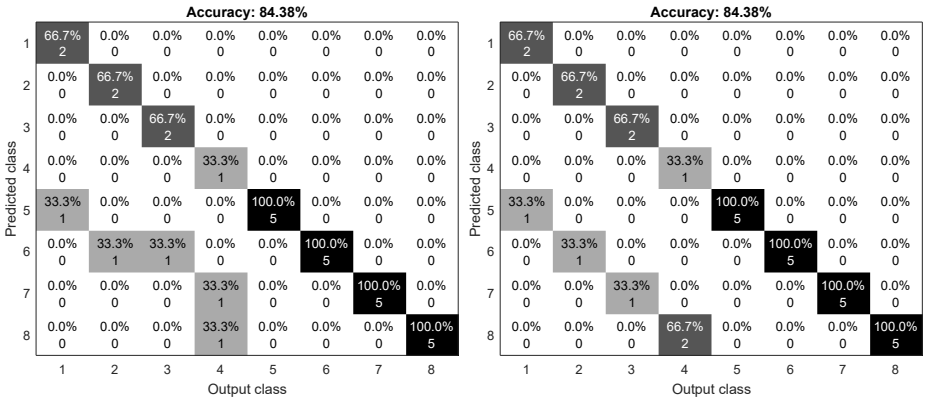


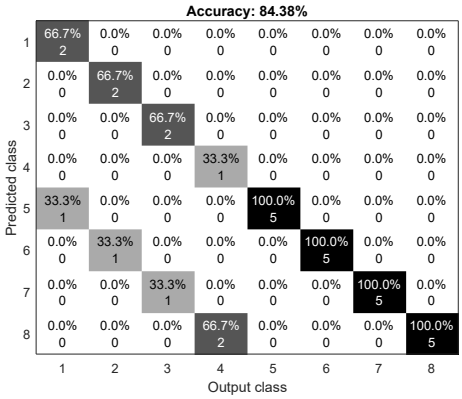
Figure 4.1: Confusion matrix example

If one looks at the orange and the brown squares that represent the results for the testing the Label 1 (degradation of capacitor) and 4 (degradation of MOSFET). Based on the figure, one can see for the capacitor case, two out of three the Label 1 is correct while one out of three, the algorithm outputs the label number 5 which is capacitor in critical condition instead of 1 (the purple circle that is inside the orange square). This means that 66.7% (2*100/3= 66.7%) out of the time the Label 1 is predicted correctly while 33.3 % the algorithm will give a wrong response for the Label 1 case. This can be read in the same manner for the case of the Label 4 in the brown square.

4.1.1 Results of the KNN for large size data



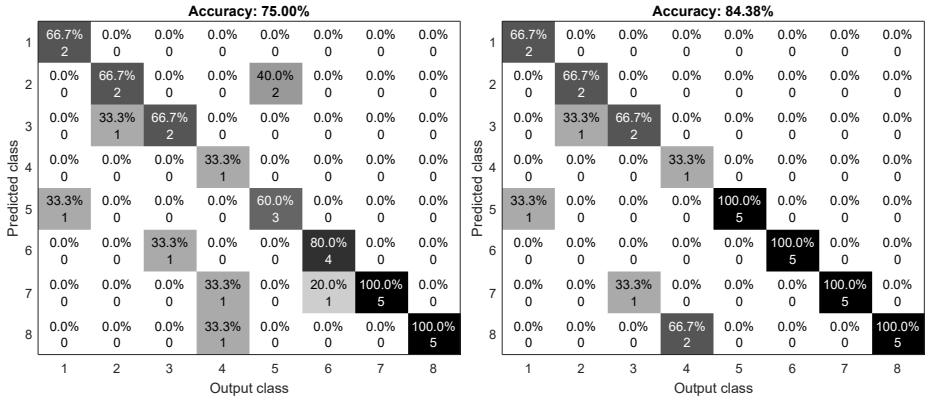
(a) With no perturbation (b) With sinusoidal perturbation



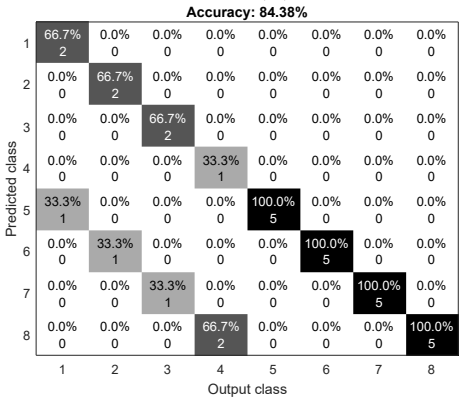
(c) With step perturbation

Figure 4.2: KNN without PCA

Figures 4.2a, 4.2b and 4.2c above show the confusion matrix of the KNN when multiple sensors were used as inputs. Figure 4.3a shows the results when there is no fault perturbation at the PWM while Figure 4.3b and 4.3c shows the confusion matrix of the KNN in case of sinusoidal and step perturbation. As mentioned in the theory section, the diagonal part of the confusion matrix shows how much in percent the machine algorithm is capable of predicting the label correctly. Note that in this type of confusion matrix, to count the total number of labels, one should look at the columns. For example for the no perturbation case the label '1' is predicted by the machine 2 out of 3 correctly while one of the prediction is false and predicts '5' instead. Though the overall accuracy is the same for different conditions (84.3 percent), having a perturbation in the signal improves the prediction a little by at least predicting the correct component while giving a false condition. For example in the case of MOSFET degradation, when the correct label should be '4', the KNN with perturbation simply specifies that the condition of the MOSFET is critical instead of simply it is degrading. However, with no perturbation, the machine is confused between the critical condition of the inductor as well. Hence, this is for the case without using the PCA. Therefore, 5 principal components are used for the new confusion matrix scene in the confusion plots below.



(a) With no perturbation (b) With sinusoidal perturbation

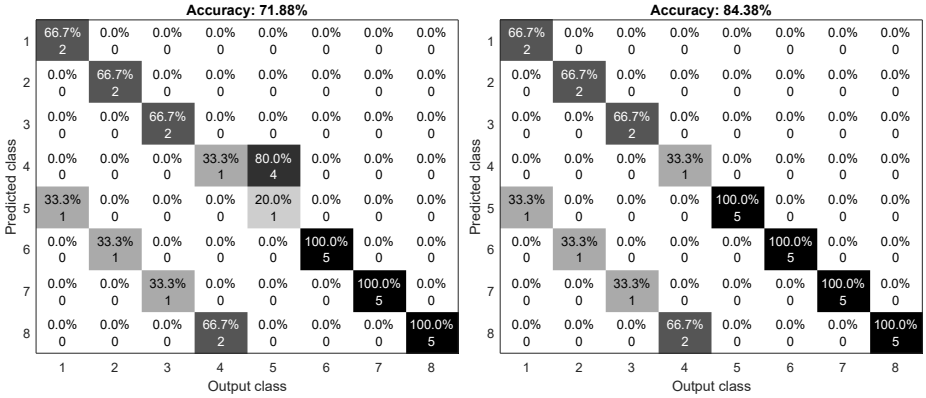


(c) With step perturbation

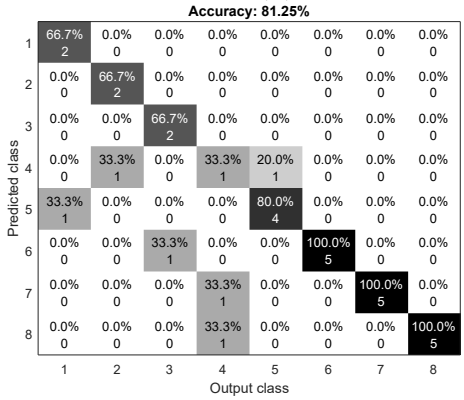
Figure 4.3: KNN with PCA

As can be seen the PCA in this case has reduced the size of the data, but have lost valuable information with the case of no perturbation. For the sine perturbation condition, it only affected the Label '2' where it is confused 33 percent of the time with diode degradation instead of inductor critical condition indicating also a slight reduction in the performance. For the step perturbation case, no change occurred when the PCA was used.

4.1.2 Results of the KNN when only the output Voltage and its features are considered as inputs



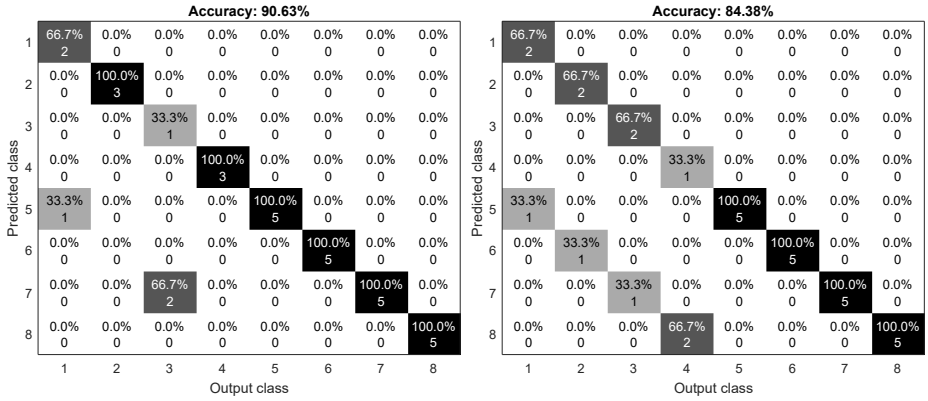
(a) With no perturbation (b) With sinusoidal perturbation



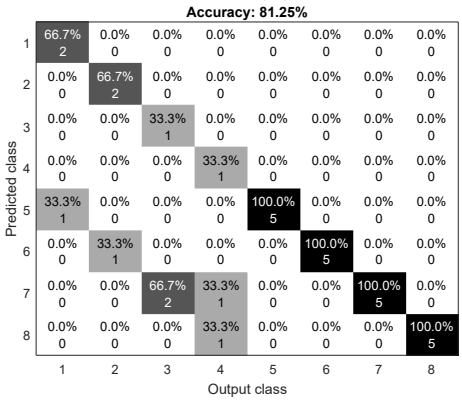
(c) With step perturbation

Figure 4.4: KNN without PCA

When only the output voltage is considered along with its features, the machine will have less information and therefore for the case of no perturbation the overall accuracy decreases to 75 percent. The step perturbation accuracy has been slightly reduced to 81.25 percent while for the case of sinusoidal perturbation it had no affect on it. Once the PCA is used as seen in the Figure 4.5, one can observe a high increase in accuracy for the case of no perturbation to a 90.63 percent accuracy. However, the two perturbation have not changed.



(a) With no perturbation (b) With sinusoidal perturbation



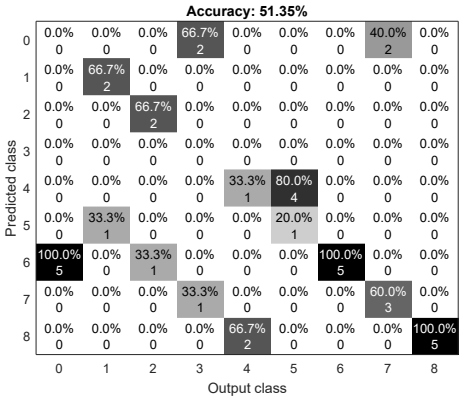
(c) With step perturbation

Figure 4.5: KNN with PCA

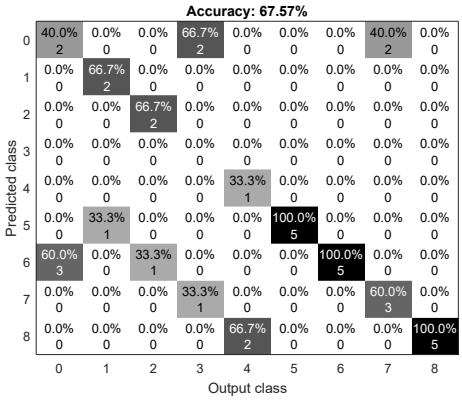
Adding another label for the case of load variation

As mentioned before, in practice the load varies and therefore could lead to false prediction from the machine. Hence, the confusion matrix below is same as the one in Figure 4.4 and 4.5 but with added label '0' indicting load variation.

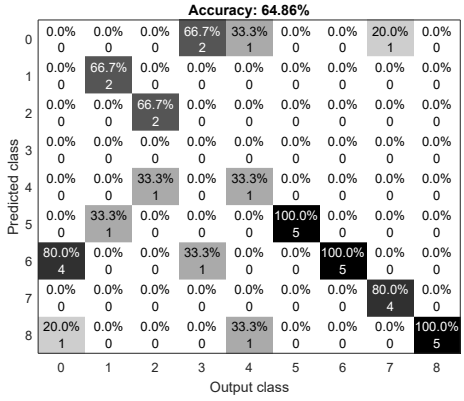
As can be seen from Figure 4.6, adding an additional label has drastically reduced the accuracy of the KNN in all cases, especially in the case of no perturbation where the machine was never able to predict correctly the Labels '0' and '3'(this is same for the step perturbation). In the sinusoidal perturbation the accuracy is a bit better than the no perturbation case where the machine was never able to correctly predict the Label '3'.



(a) With no perturbation



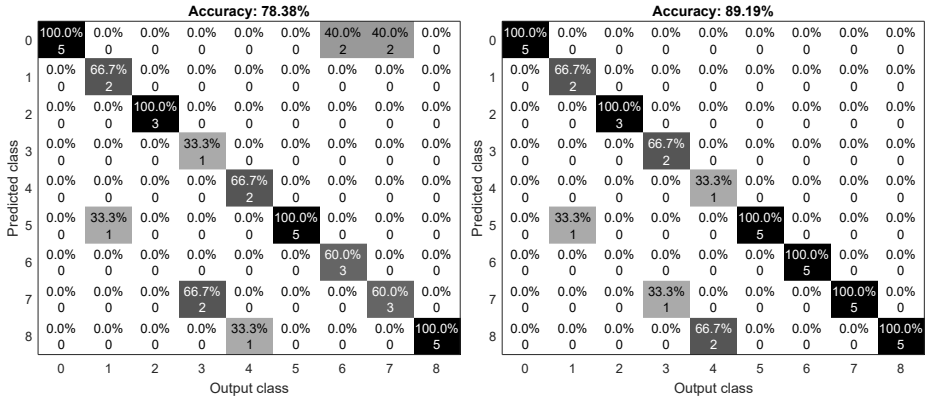
(b) With sinusoidal perturbation



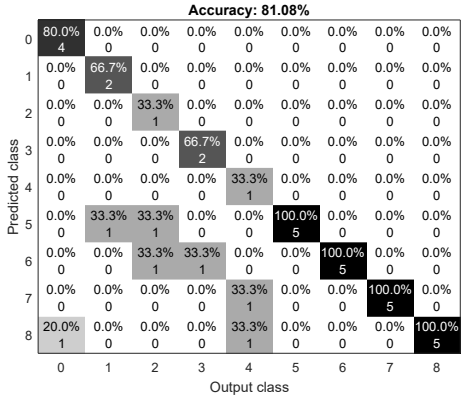
(c) With step perturbation

Figure 4.6: KNN without PCA

When the PCA was used, the machines accuracy increased for all cases as can be seen in the Figure 4.7.



(a) With no perturbation (b) With sinusoidal perturbation



(c) With step perturbation

Figure 4.7: KNN with PCA

By adding PCA the dimensionality is reduced and therefore, the ML was able to better separate the Labels which increased its overall performance.

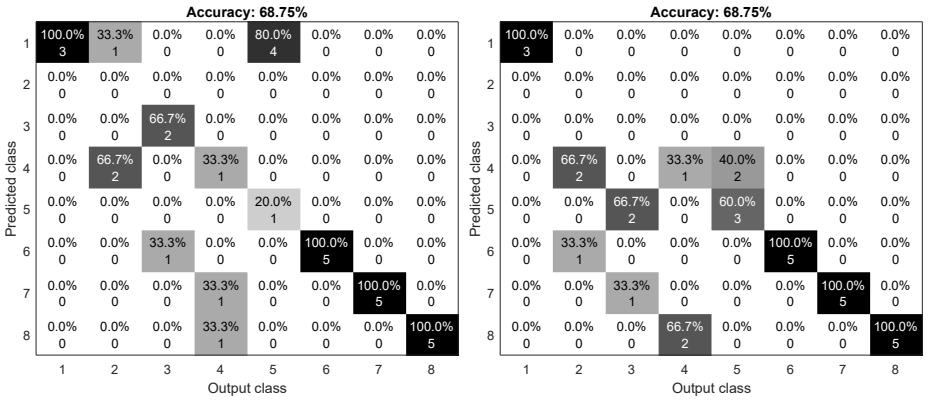
4.1.3 KNN summary

Based on these results, the KNN works adequately, especially if one needs to use a signal sensor to collect data. With combination of PCA and sinusoidal perturbation, the KNN still have errors but only in the degradation of the components and note in the critical condition which seems to be acceptable given the fact that maintenance will only be applied in case of critical conditions.

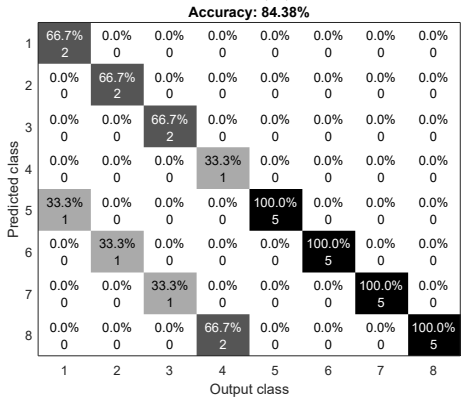
4.2 Tree

4.2.1 Results of Tree for large size data

The three confusion plots in Figure 4.8, shows the performance of the tree algorithm.



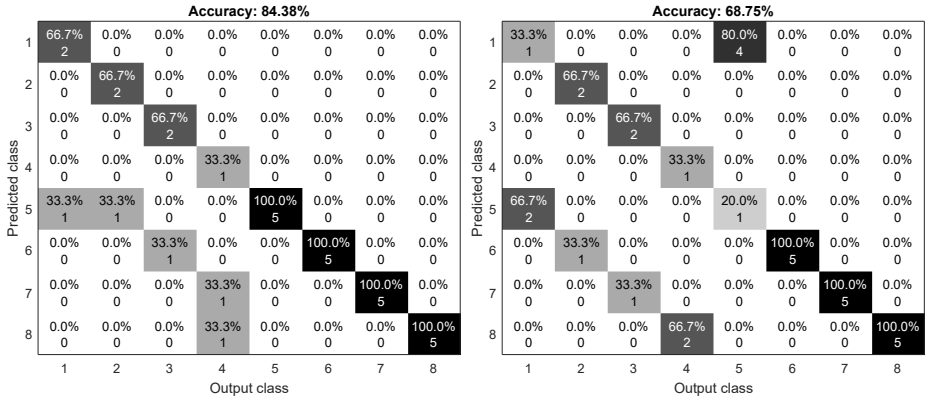
(a) With no perturbation (b) With sinusoidal perturbation



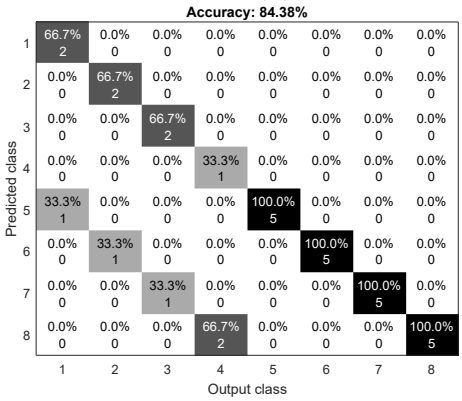
(c) With step perturbation

Figure 4.8: Tree without PCA

As can be seen from the confusion matrices, the case for no perturbation and sinusoidal perturbation the overall accuracy is low (68.75 percent). Plus there are labels were both cases always gets it wrong. For example in the no perturbation condition in Figure 4.3a, two out of three times the Label '2' is predicted by the machine as '4' which is the wrong component and one out of three times it predicts '1' which is a capacitor degradation instead an inductor. On the other hand the sinusoidal perturbation is worse. Two labels in which the machine never predicted correctly, the Label '2' which 2 out of three scenarios is predicted as '4' and one out of three is predicted as '6'. And also the Label '3' is predicted two out of three times as '5' and one out of three times as '7'. For the case of step perturbation, it is much more accurate with an accuracy of 84.3 percent. After that a PCA is used. By Creating 5 PCs we obtained the confusion matrices in the Figure 4.9 below.



(a) With no perturbation (b) With sinusoidal perturbation



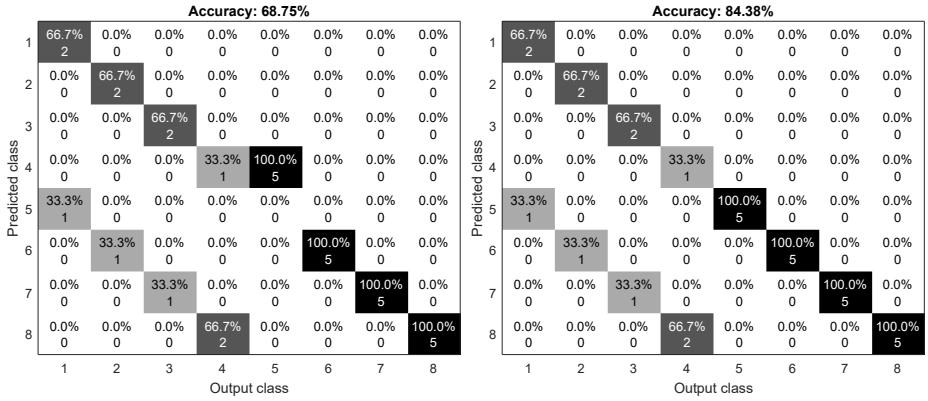
(c) With step perturbation

Figure 4.9: Tree with PCA

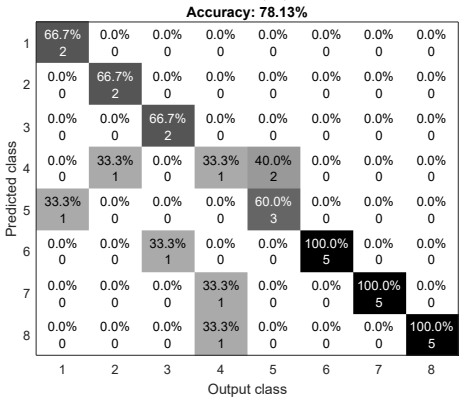
As can be seen in Figure 4.9a, the PCA have increased the overall accuracy to 84.38 percent and in most cases it has a high percentage of correct predictions except for the Label '4' where one out of three predicts it as '7' and one out of three predicts it as '8'. The PCA has also given a slight increase in the performance in the step perturbation case as can be observed in the Figure 4.9c. However, in the sinusoidal case, the overall accuracy remains the same, although the error distribution is much better than before. Since it can be seen that the only reason where the accuracy decreased is that, for the label '5', four out of five it predicts it as '1'. Meaning that it is still the correct component (the capacitor) but it provides the wrong conditions.

4.2.2 Results of Tree when only the output Voltage and its features are considered as inputs

The following confusion matrices represent the performance of the tree where the part of the output voltage signal and its properties are taken as inputs.



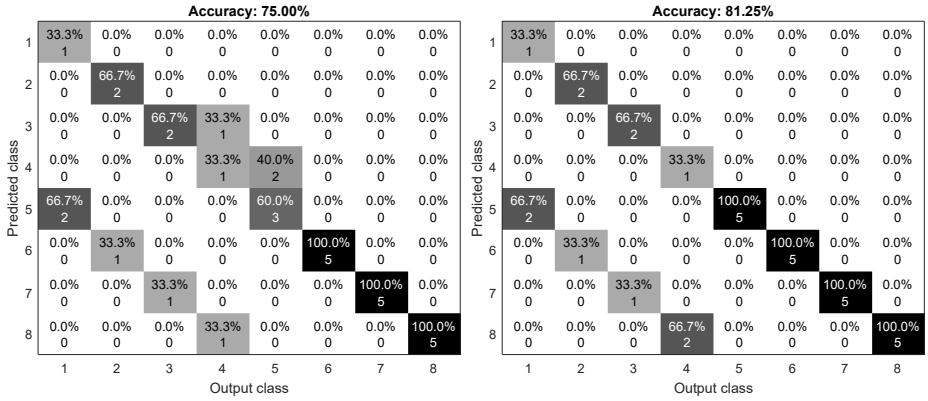
(a) With no perturbation (b) With sinusoidal perturbation



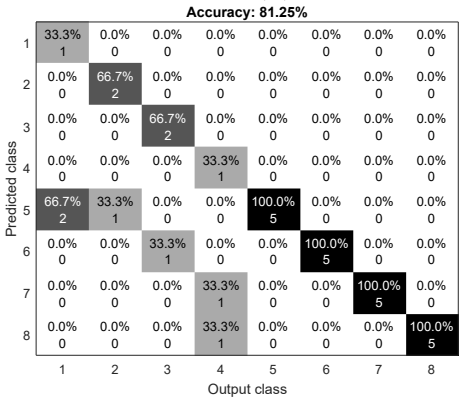
(c) With step perturbation

Figure 4.10: Tree without PCA

As can be seen in Figures 4.10a, 4.10b and 4.10c, the sine perturbation case has a the best overall accuracy and performance due to the fact that when error is made it is only the condition and not the component that the machine provide a wrong answer. In the no perturbation case, the Label '5' which indicates a critical condition of the capacitor is predicted by the classification learner as '4' which is a degradation of the MOSFET. The step perturbation case however, has some slight random errors. The step perturbation also predicts two out of 5 times the critical condition of the capacitor as MOSFET degredation. Figure 4.11 below represents the tree After using the PCA.



(a) With no perturbation (b) With sinusoidal perturbation



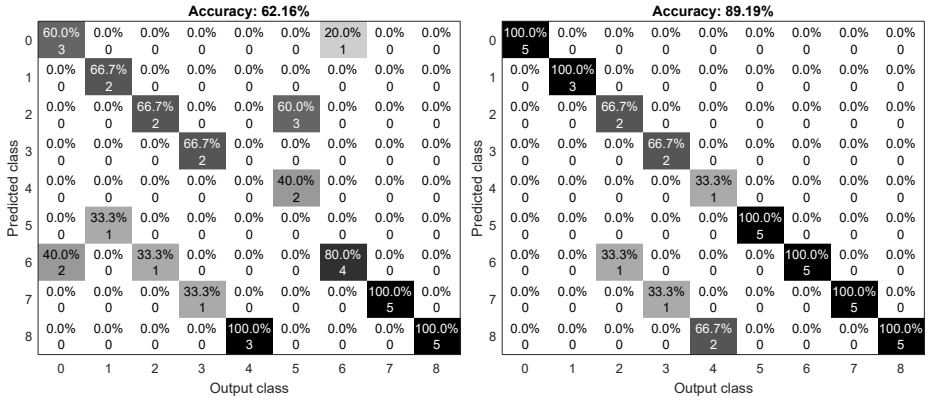
(c) With step perturbation

Figure 4.11: Tree with PCA

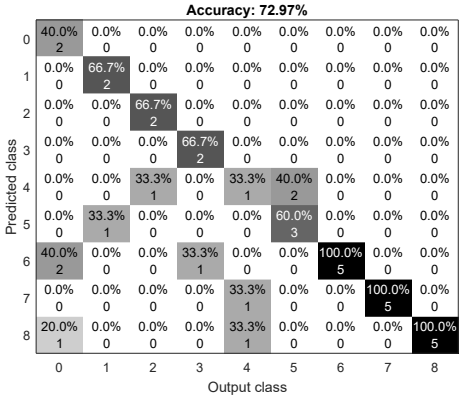
As can be observed with the PCA, in all cases, the accuracy increases. The step and sinusoidal perturbation cases have better performance than the first one; however, we can easily see that the sinusoidal performs better than the step due to the fact that it only predict the wrong conditions and not the wrong components.

Adding another label for the case of load variation

For the case of load variation, by adding an additional label '0' indicating variation of load, the performance of the tree for the different conditions are plotted in the Figure 4.12 matrices.



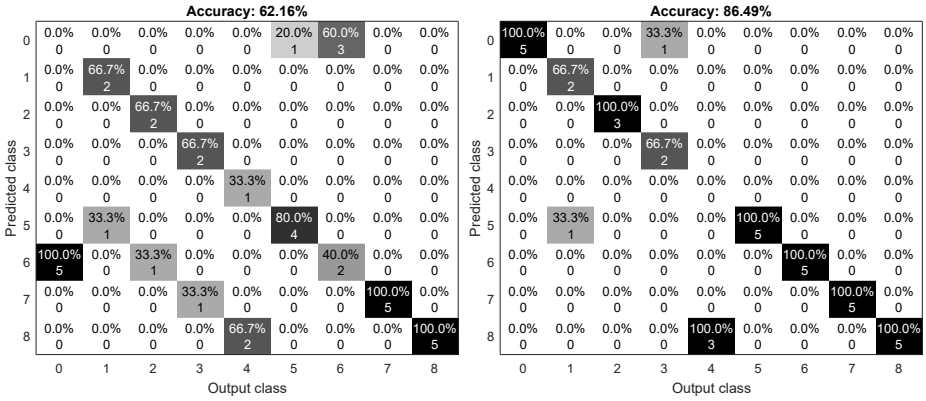
(a) With no perturbation (b) With sinusoidal perturbation



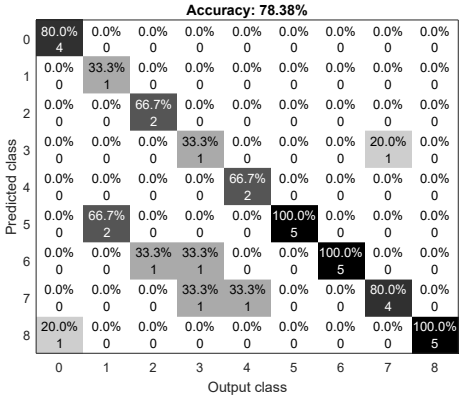
(c) With step perturbation

Figure 4.12: Tree without PCA

As can be seen in the confusion plots, the no perturbation condition has low accuracy. The step perturbation condition has an overall accuracy of 72.97 percent and the sinusoidal perturbation has a very high accuracy of 89.19 percent. By adding the PCA as seen in the Figure 4.13, the accuracy in the no perturbation and sine perturbation case are slightly reduced while in the step perturbation case is slightly increased compared to that without PCA.



(a) With no perturbation (b) With sinusoidal perturbation



(c) With step perturbation

Figure 4.13: Tree with PCA

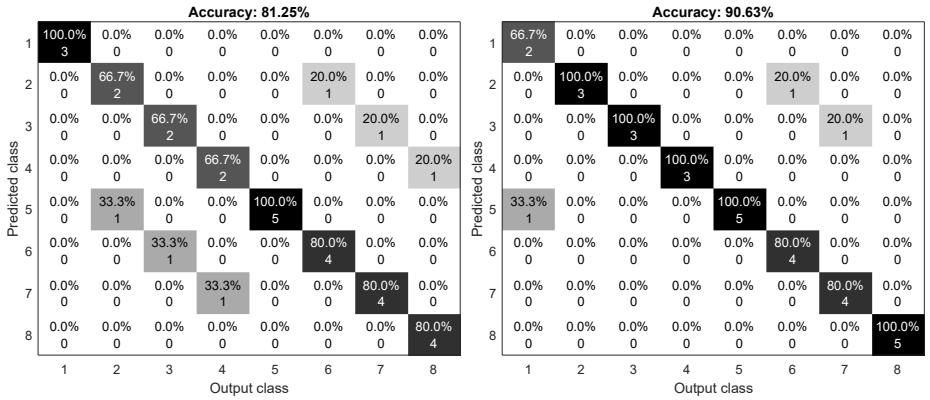
4.2.3 Tree summary

Based on the results of the Tree algorithms, it is preferable to use the sinusoidal perturbation without PCA, given the fact that with the use of PCA and sinusoidal perturbation resulted in under-fitting and therefore reduction of accuracy especially when the load variation is considered. Moreover, using the sinusoidal perturbation without PCA has an overall good accuracy while capable of accurate prediction of components in their critical conditions.

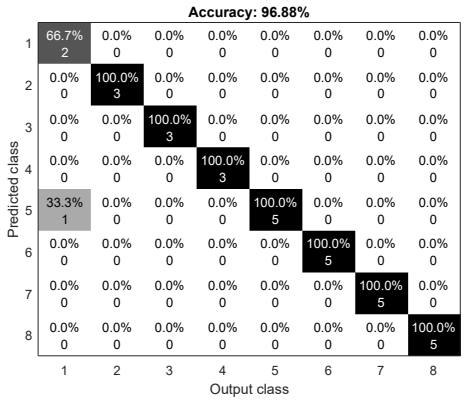
4.3 SVM

4.3.1 Results of SVM for large size data

As can be seen by the Figure 4.14, in all cases the SVM performs well.



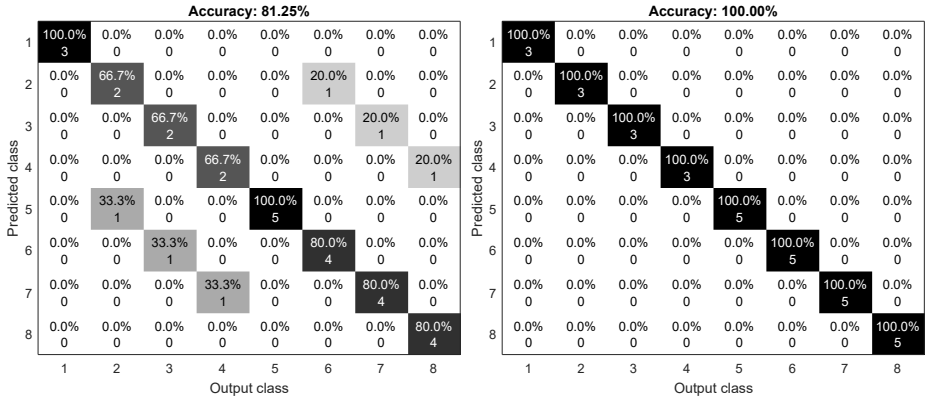
(a) With no perturbation (b) With sinusoidal perturbation



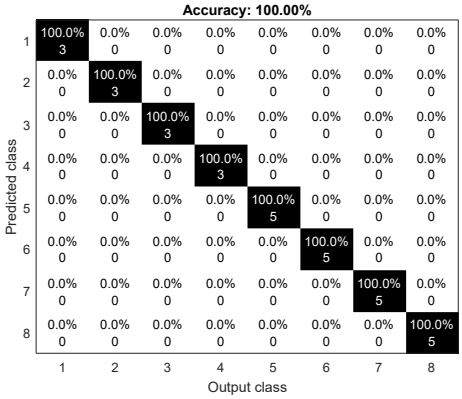
(c) With step perturbation

Figure 4.14: SVM without PCA

The step perturbation case has the highest accuracy than that of the no perturbation case and sine perturbation case. Only one out of 3 cases the algorithm predicts a label of '5' instead of 1. In the sinusoidal perturbation, the SVM wrongfully outputs one out of three, a label of '5' instead of '1', one out of 5 a label of '2' instead of '6' and one out of 5 a label of '3' instead of '7'. By using the PCA, the no perturbation case did not improve while in the fault condition cases the algorithm managed to predict all the test data correctly as can be seen in the Figure 4.15.



(a) With no perturbation (b) With sinusoidal perturbation

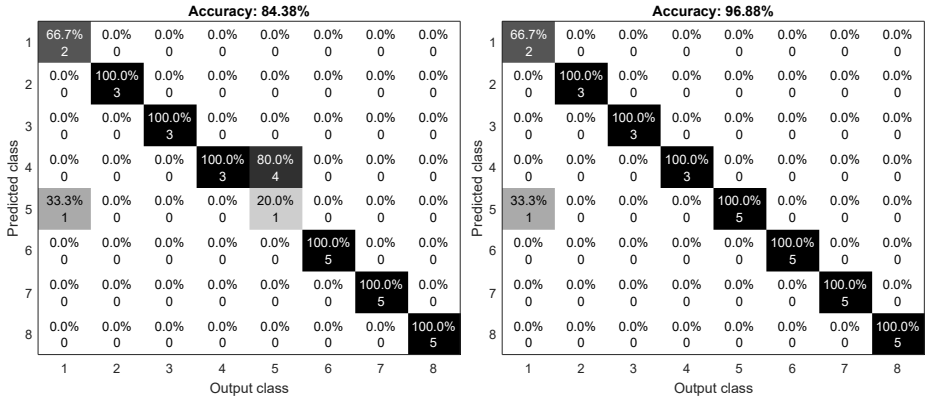


(c) With step perturbation

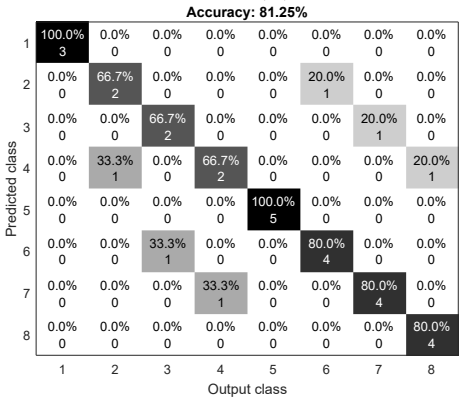
Figure 4.15: SVM with PCA

4.3.2 Results of SVM when only the output Voltage and its features are considered as inputs

When input data was removed from the training set while maintaining only the output voltage signal and its properties the SVM looses accuracy for the step perturbation condition but increases its accuracy in the no perturbation and sine condition as can be seen from the Figure 4.16. This can be explained due to the fact that with less data in the training set the SVM manages to separate the different labels correctly for the case of no perturbation and sine perturbation conditions and less able to do the same in the step perturbation case.



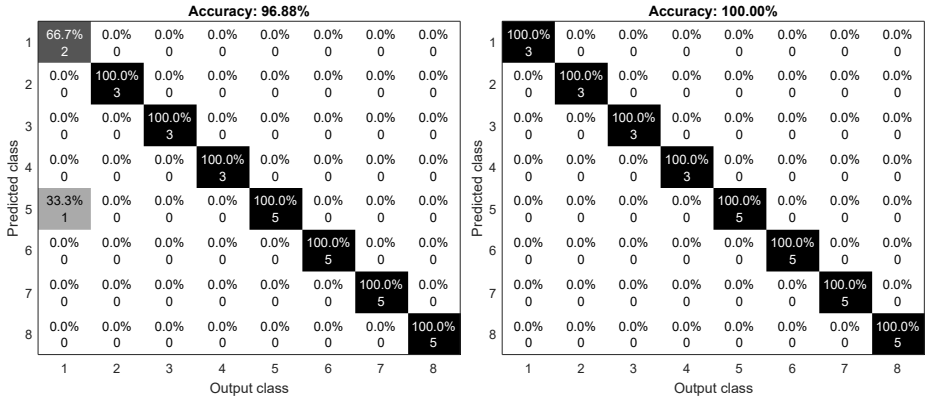
(a) With no perturbation (b) With sinusoidal perturbation



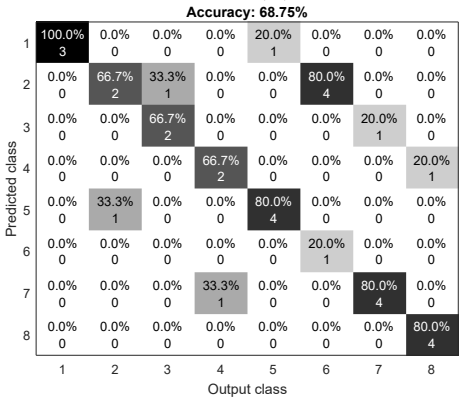
(c) With step perturbation

Figure 4.16: SVM without PCA

By further using the PCA, the no perturbation and sine perturbation condition has been improved even further while in the step perturbation cases, it has further reduced its accuracy indicating a loss of important information in the data.



(a) With no perturbation (b) With sinusoidal perturbation

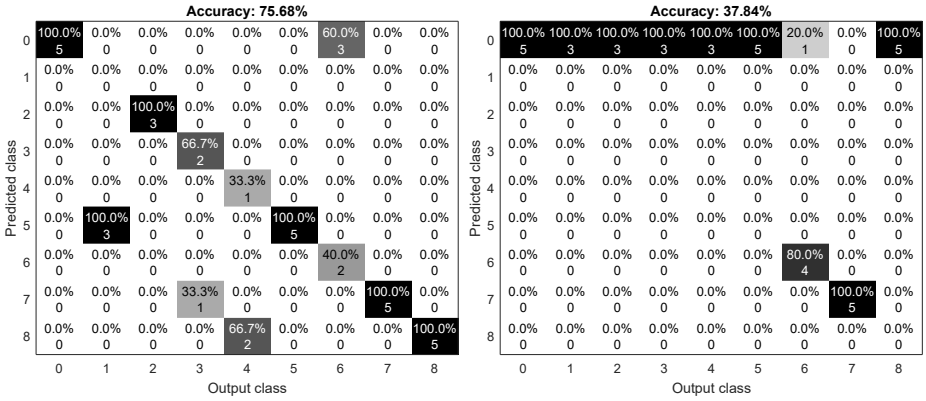


(c) With step perturbation

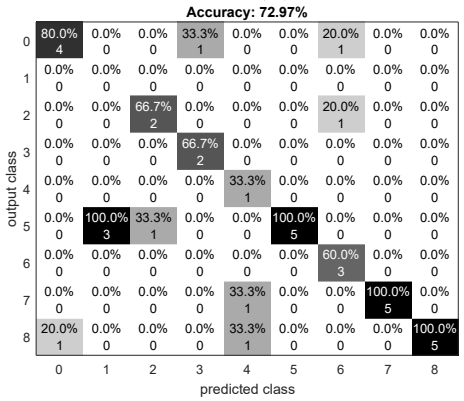
Figure 4.17: SVM with PCA

Adding another label for the case of load variation

Once the label '0', it resulted in drastic reduction of accuracy especially for the sinusoidal perturbation condition where the accuracy was reduced to 37.84 percent. As can be seen from the Figure 4.18, in the sine perturbation case, the label '0'confuses the SVM, and incorrectly predicts most of the conditions which indicates that the ML was not abel to seperate the Labels from one another..



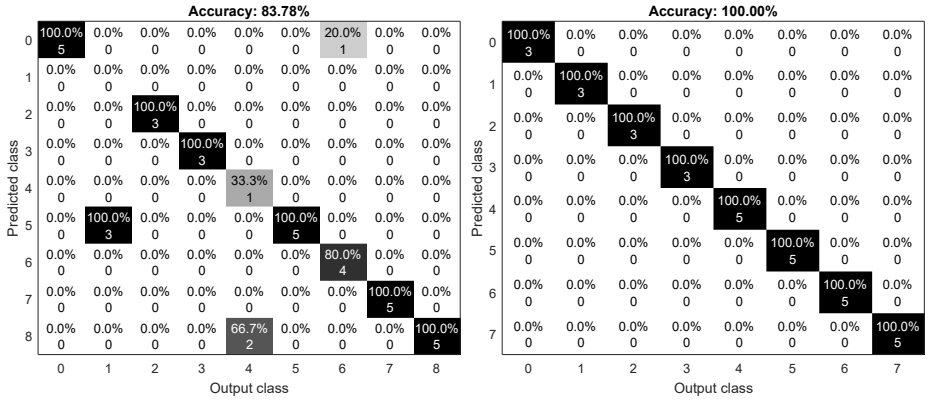
(a) With no perturbation (b) With sinusoidal perturbation



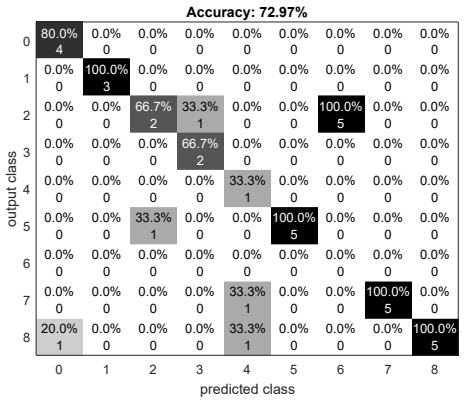
(c) With step perturbation

Figure 4.18: SVM without PCA

However, by using the PCA, the result shows an increase in accuracy in sinusoidal and no perturbation condition while the step perturbation condition remains the same.



(a) With no perturbation (b) With sinusoidal perturbation



(c) With step perturbation

Figure 4.19: SVM with PCA

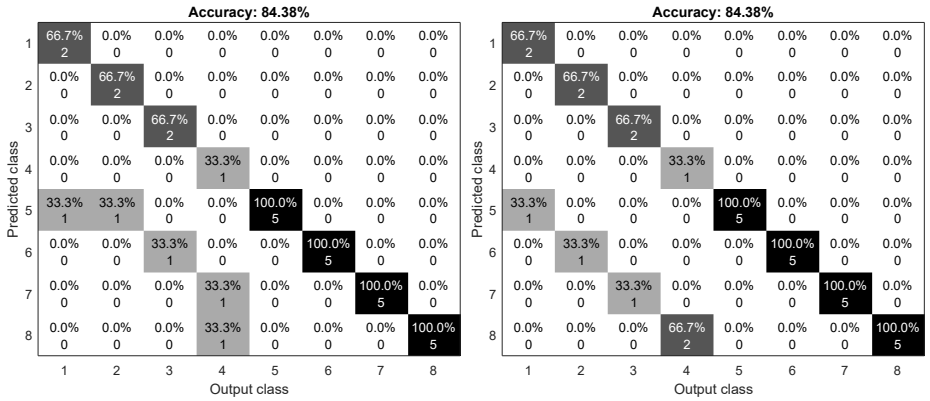
4.3.3 SVM summary

The SVM has great accuracy in all components conditions (degrading or critical conditions) when using the PCA with sinusoidal perturbation. Even when adding the load variation it still has great accuracy.

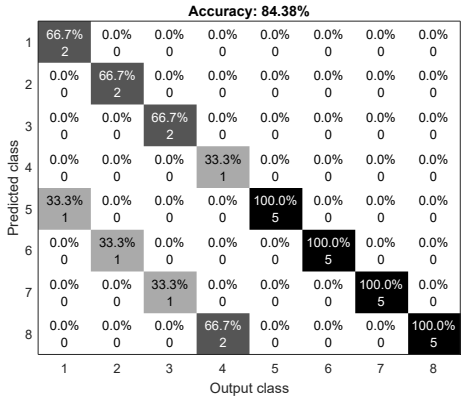
4.4 Ensemble

4.4.1 Results of Ensemble for large size data

The Figures 4.20 and 4.21 show the result of the ensemble algorithm with and without PCA for different cases. As can be seen by the two Figures the accuracy in all cases are high and by using the PCA the overall accuracy increases in all conditions. In the no fault condition the algorithm predicts the two the wrong components twice. The first component is for the case of inductor degradation one out the times the algorithm outputs a label of '5' which indicates a capacitor in critical condition. The second one is when the label should be '4' which indicates a MOSFET degradation, the algorithm predicts a diode in critical condition. These two errors disappears when the PCA is used. With the sinusoidal fault case, by



(a) With no perturbation (b) With sinusoidal perturbation



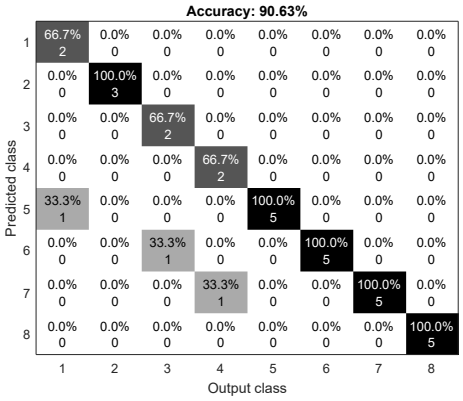
(c) With step perturbation

Figure 4.20: Ensemble without PCA

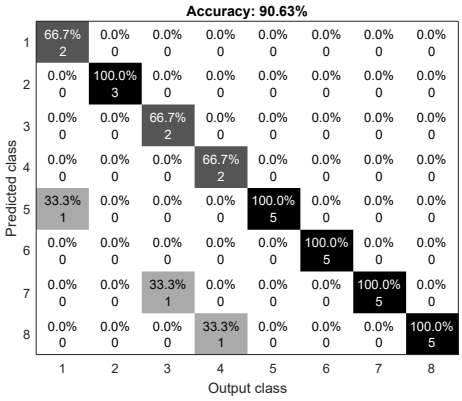
using the PCA the label '2' will always be predicted correctly and for the step fault condition the label '1' will be correctly predicted after using the PCA.

4.4.2 Results of Ensemble when only the output Voltage and its features are considered as inputs

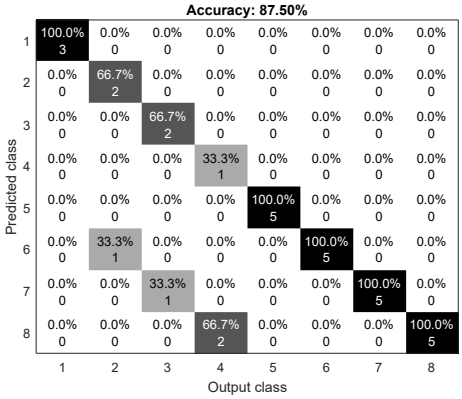
By using only the output voltage, the accuracy in all cases has been reduced. The no perturbation case accuracy has dropped to 71.88% while the sinusoidal and step perturbation cases dropped to 84.38% and 75% of the accuracy, respectively.



(a) With no perturbation

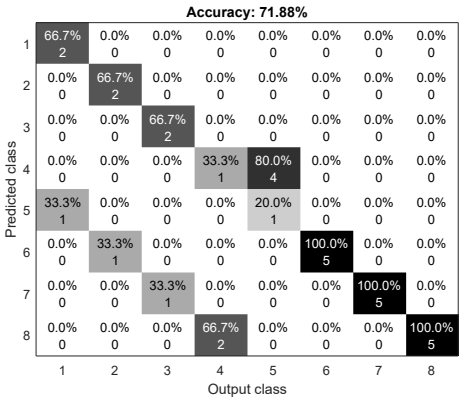


(b) With sinusoidal perturbation

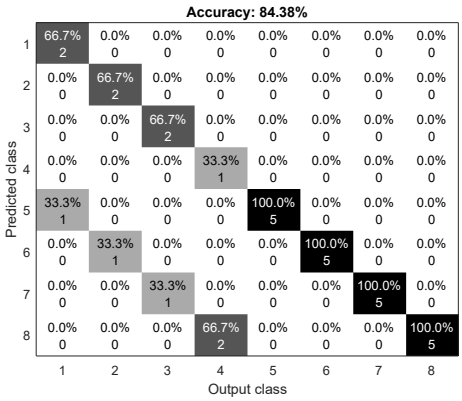


(c) With step perturbation

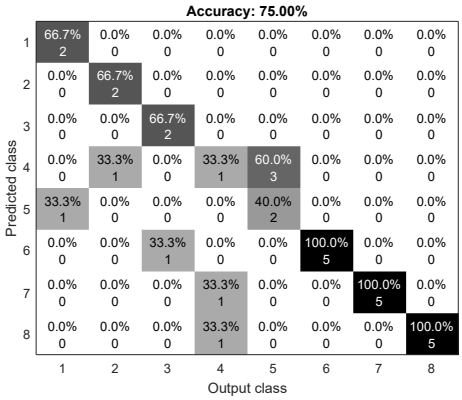
Figure 4.21: Ensemble with PCA



(a) With no perturbation



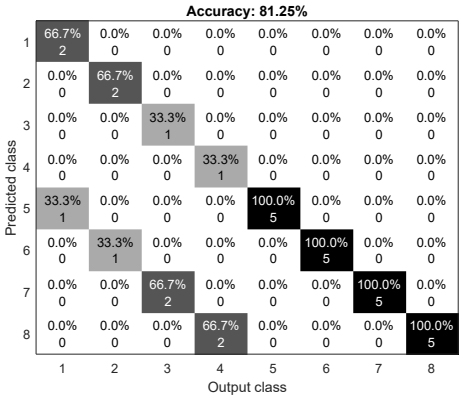
(b) With sinusoidal perturbation



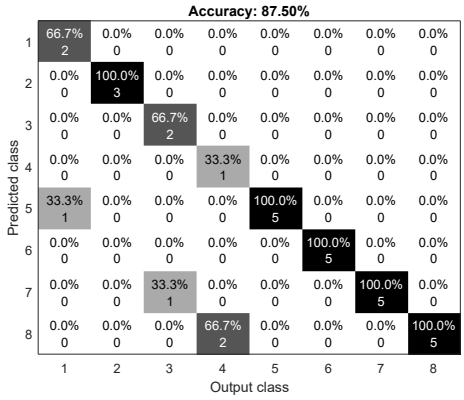
(c) With step perturbation

Figure 4.22: Ensemble without PCA

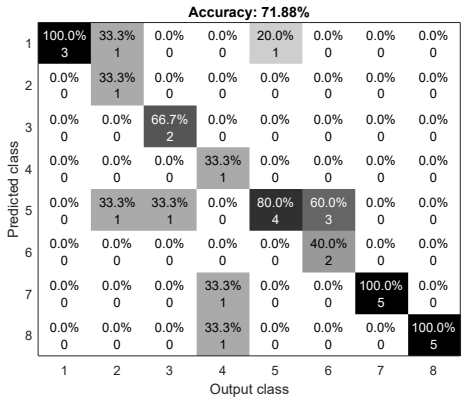
However, with the use of the PCA, the accuracy in the sine and no perturbation case has increased while the one in the step case has decreased. Plus in both the no perturbation and sine perturbation condition, the algorithm predict the wrong condition of the components only and not the components itself.



(a) With no perturbation



(b) With sinusoidal perturbation

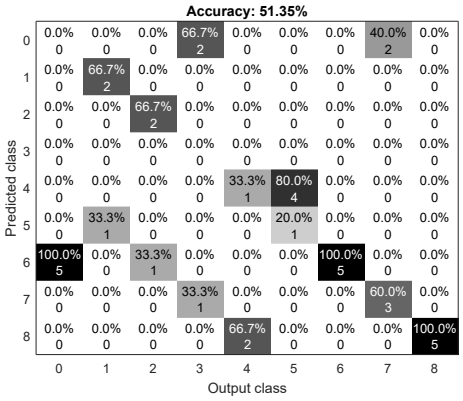


(c) With step perturbation

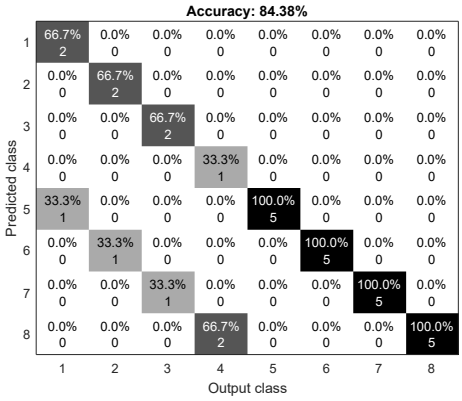
Figure 4.23: Ensemble with PCA

Adding another label for the case of load variation

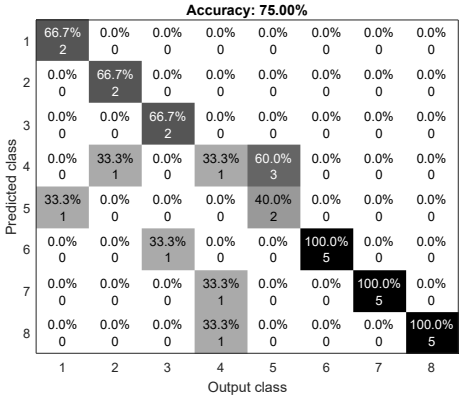
After adding the label '0', it resulted in lower accuracy of the no perturbation condition, slight reduction in the sinusoidal perturbation and a slight increase in the step perturbation case.



(a) With no perturbation



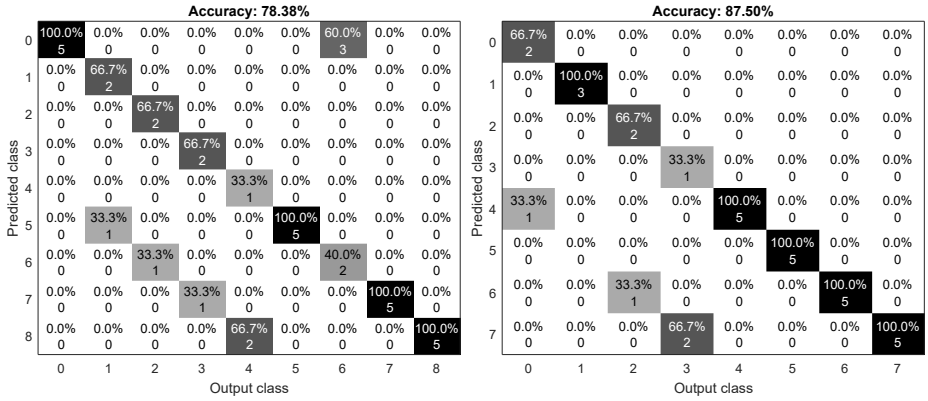
(b) With sinusoidal perturbation



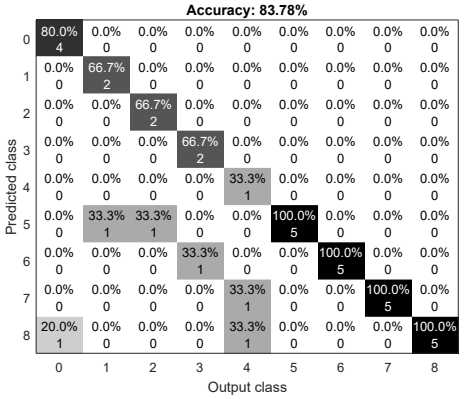
(c) With step perturbation

Figure 4.24: Ensemble without PCA

By adding the PCA, it resulted in the increase in accuracy and acceptable errors from the algorithm.



(a) With no perturbation (b) With sinusoidal perturbation



(c) With step perturbation

Figure 4.25: Ensemble with PCA

Based on these results, we can conclude that the best algorithm is the SVM with a combination of sinusoidal perturbation and use of PCA. The reason is because with less data and less sensor that is needed to be pre-processed, the SVM has high accuracy of predicting witch component is degrading and it is also capable of correctly predicting the condition of the component

4.4.3 Ensemble summary

Like the Tree and the KNN, the Ensemble works well when sinusoidal perturbation is used with the PCA. It has great accuracy for the critical conditions but poor accuracy when it comes to degradation.

4.5 Using Filter

As mentioned in the Theory Chapter, in reality when data are collected from an actual sensors, there will be noise in the data and therefore, must be processed before using it as training data for the machine. By using the equation of the transfer function for the band-pass and low-pass filters, the block seen in the Figure 4.26 is created in Simulink in order to be used for filtering out the noise. α_{f1} in the Figure 4.26 is equal to 2000π, α_{f2} is equal to 278π, α_{f3} is 333π and α_{f4} is 160000π [2]. The block A, represents the transfer function of the low-pass filter for the DC case while the B is used for the sinusoidal perturbation (the band pass is calculated in order to include the frequency of the sinusoidal perturbation). For the case of no perturbation and the step perturbation, the block B is removed. Note that in the filter test the sinusoidal perturbation amplitude is set to 0.1 (peak to peak) which is also same for the step perturbation. This is done in order to see if it is possible to observe the perturbation after applying filters.

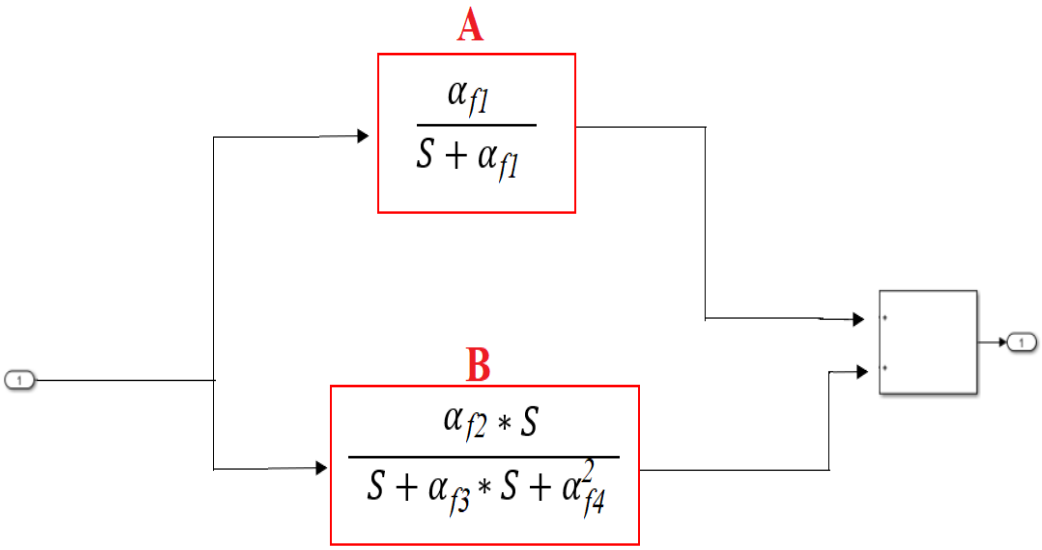


Figure 4.26: Block of the transfer function in simulink

Figure 4.27 and 4.28 show the bode plots of the low-pass filter, used for the case of no perturbation, and step perturbation and band-pass filter, used when sinusoidal perturbation is applied, as it can be observed from the plots, the low-pass filter allows signals with low frequency to pass while cutting of the high frequency signals while the band-pass filter will allow a set of frequencies to pass (between 0.01 and 100) while reducing others.

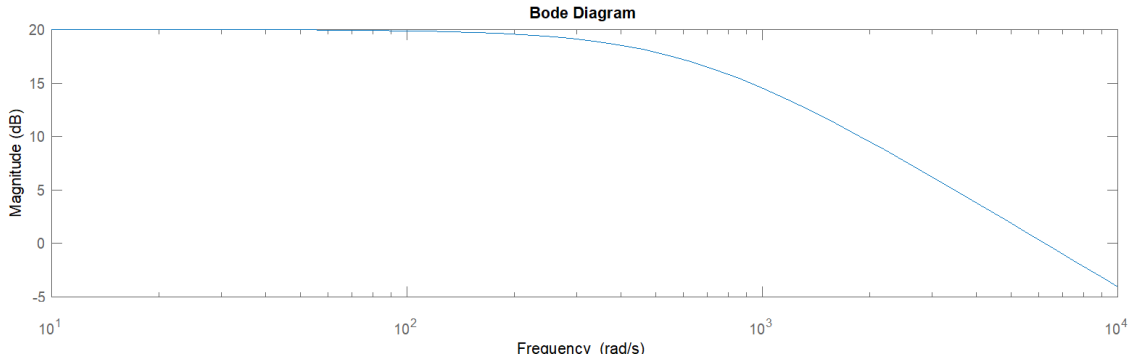


Figure 4.27: Bode plot of Low-pass filter for the DC and step perturbation

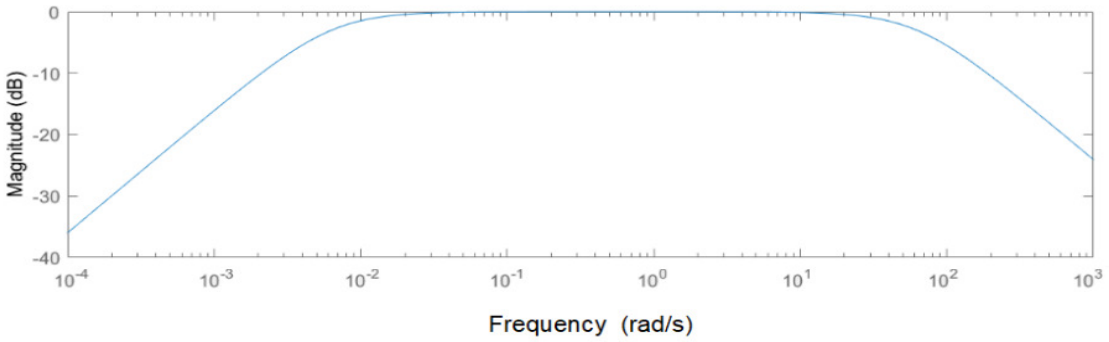
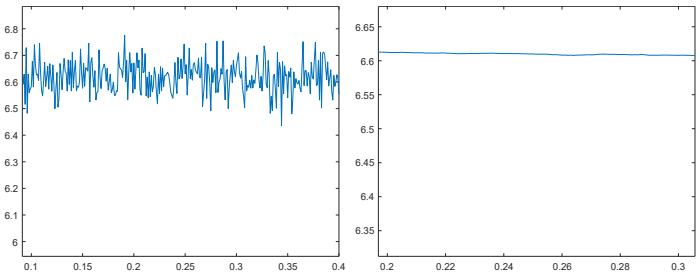


Figure 4.28: Bode plot of band-pass filter for the sinusoidal perturbation

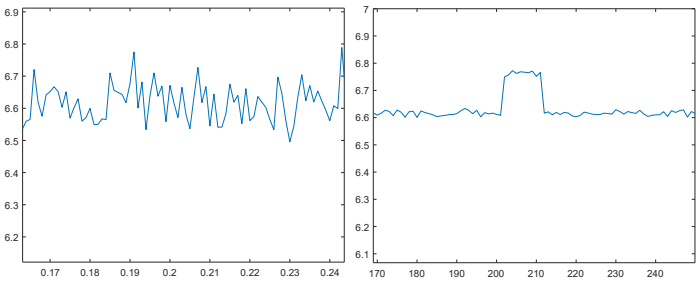
The Figure 4.29 and 4.30 show the result of output voltage before and after adding the DC low-pass filter to it. Based on the result the low-pass filter did reduce noise for no perturbation and for the case of the step perturbation.



(a) Output voltage with noise

(b) Output voltage with Band Pass filter

Figure 4.29: Output Voltage for no perturbation



(a) Output voltage with noise

(b) Output voltage with low-pass filter

Figure 4.30: Output Voltage for step perturbation

By using the band pass filter for the DC components and for the sinusoidal fault in combination, it resulted with the following bode plot in Figure 4.31.

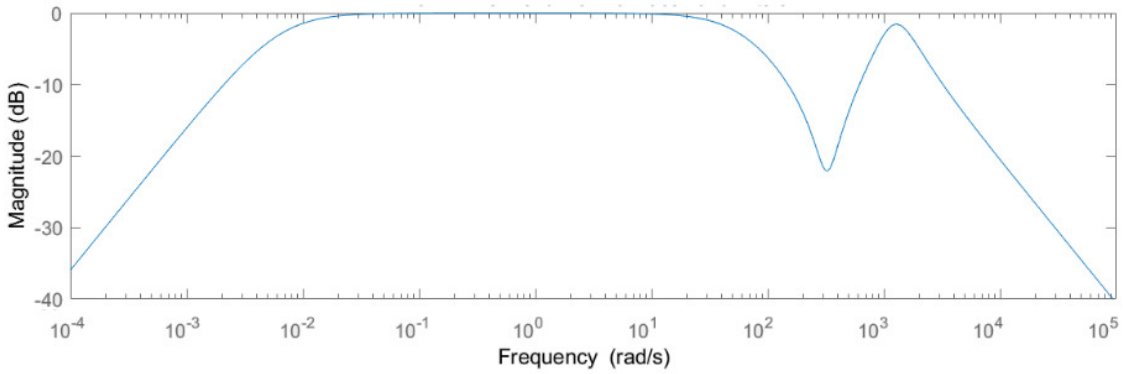
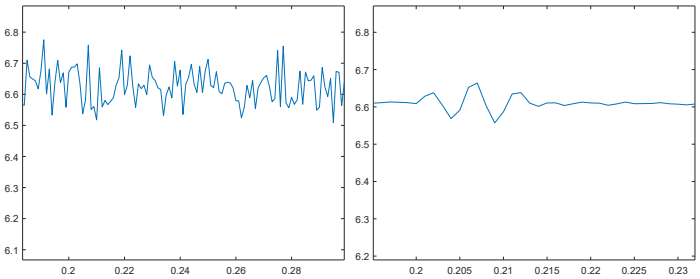


Figure 4.31: Bode Plot of the filters for the sinusoidal perturbation

As seen in the figure, the band pass allows the DC components to pass and the sinusoidal component to pass (between 150 and 300 Hz). The plot in Figure 4.32 shows the result before and after adding the noise to the output voltage signal. The result shows that after using the two filters for DC and sinusoidal perturbation, one is able to observe the sinusoidal perturbation again on the output voltage.



(a) Output voltage with noise

(b) Output voltage with Band Pass filter

Figure 4.32: Output Voltage for sine fault injection case

4.5.1 Filter summary

Both sinusoidal and step function seems to be visible after de-noising and therefore, one must use both perturbation and test the algorithm then see in which case the ML performs better.

4.6 Summary

This chapter is completed and it has shown that most of the algorithm have good performance depending on the conditions. The chapter has also shown that filter works well in all cases.

Chapter 5

Conclusion and future work

5.1 Conclusion

Training a classification learner has proved to be challenging. The performance of every algorithm differs from each other based on the different cases, such as injection perturbation to the system, reducing the size of the training data or using PCA. However, assuming that there is no noise in the sensor (ideal case like when we simulated and tested the ML), we have observed that the combination of SVM with sinusoidal perturbation and using PCA has proven to be the best solution, given the fact that the algorithm is capable of predicting the condition of the components with great accuracy in all cases (degradation and critical cases). Plus, by adding the load variation the SVM algorithm is still capable of correctly predicting the condition and which component is degrading compared to when we applied the step perturbation (based on Figure 4.19). This is probably because the SVM is capable of separating the labels much better when using sinusoidal perturbation compared to step perturbation. Moreover, compared to other algorithms, the SVM is capable of correctly predicting the components in both conditions, in the degradation and critical condition, which provides the possibility of monitoring the component during its degradation phase. Hence, we will only need to filter one sensor and the sinusoidal perturbation shape can be observed after using the filters.

5.2 Future work

More studies are needed before actually applying the classification learner. For example trying to use the data after filter as training data and see if the algorithm still performs well. Other feature could have provided more distinctive information for the variation of different component such as using FFT or wavelet transform. Another problem to take into consideration is the aging affect. In actual converters, if one component degrades, it could accelerate the degradation of other components as well. A larger data set could have been used to train the machine to improve accuracy and a deep learning algorithm such as the Artificial Neural Network (ANN) could have been used as classification learner for training and testing.

Bibliography

[1] A. Filion, "Predictive Maintenance with MATLAB: A Prognostics Case Study." https://www.mathworks.com/videos/predictive-maintenance-with-matlab-a-prognostics-case-study-118661.html?elqsid=1560752939251potential_use=Student, 2019.

[2] A. Adler, F. L. Podio, F. Herr, K. Cao, J. Tian, Y. Zhang, X. Yang, J. W. M. Campbell, and G. Zektser, "Cross-Validation," *Encyclopedia of Biometrics*, pp. 206–206, 2009.

[3] C. Bickerton, "A beginner's guide to decision tree classification." https://towardsdatascience.com/a-beginners-guide-to-decision-tree-classification-6d3209353ea.

[4] Deepanshu Bhalla, "K NEAREST NEIGHBOR : STEP BY STEP TUTORIAL." https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html.

[5] Zakaria Jaadi, "A step by step explanation of Principal Component Analysis,"

[6] D. Acquisition, "Data Acquisition ," *Network*, pp. 1–5, 2019.

[7] EERE, "O&M Best Practices Guide,: Chapter 5 Types of Maintenance Programs," pp. 1–9, 2010.

[8] K. Park, J. Lee, and J. Choi, "Deep Neural Networks for News Recommendations," pp. 2255–2258, 2017.

[9] X. Amatriain, "Big & personal," *Proceedings of the 2nd International Workshop on Big Data, Streams and Heterogeneous Source Mining Algorithms, Systems, Programming Models and Applications - BigMine '13*, pp. 1–6, 2013.

[10] L. Wang, J. Yue, Y. Su, F. Lu, and Q. Sun, "A novel remaining useful life prediction approach for superbuck converter circuits based on modified greywolf optimizer-support vector regression," *Energies*, vol. 10, no. 4, 2017.

[11] C. Zhang, Y. He, L. Yuan, and F. Deng, "A novel approach for analog circuit fault prognostics based on improved RVM," *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 30, no. 3, pp. 343–356, 2014.

[12] A. Luchetta, S. Manetti, M. C. Piccirilli, A. Reatti, F. Corti, M. Catelani, L. Ciani, and M. K. Kazimierczuk, "MLMVNNN for Parameter Fault Detection in PWM DC-DC Converters and Its Applications for Buck and Boost DC-DC Converters," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 2, pp. 439–449, 2019.

[13] J.-M. Kim, K.-B. Lee, D.-C. Lee, and Y.-J. Ko, "Fault diagnosis of three-parallel voltage-source converter for a high-power wind turbine," *IET Power Electronics*, vol. 5, no. 7, pp. 1058–1067, 2012.

[14] A. Report, "Sensitivity Analysis for Power Supply Design," no. September, pp. 1–7, 2011.

[15] D. M. Hamby, "A Review of Techniques for Parameter Sensitivity Analysis of Environmental Models," *Environmental Monitoring and Assessment*, vol. 32, no. 32, pp. 135–154, 1994.

[16] MathWorks, "Analyze Relation Between Parameters and Design Requirements." https://se.mathworks.com/help/sldo/ug/interpreting-analysis-results.html.

[17] J. Hurwitz, D. Kirsch, and J. Wiley, *Machine Learning Machine Learning For Dummies*. No. june, 2018.

[18] S. N. Kasturi, "Underfitting and Overfitting in machine learning and how to deal with it !!!."

[19] MathWorks, "Cross-Validation."

[20] L. Chen, "Support Vector Machine — Simply Explained." https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496.

[21] N. Bambrick, "Support Vector Machines for dummies; A Simple Explanation,"

[22] Vadim Smolyakov, "Ensemble Learning to Improve Machine Learning Results,"

[23] H. Zumbahlen, "BASIC LINEAR DESIGN Chapter 8: Analog Filters," *Linear Circuit Design Handbook*, p. 943, 2008.

Appendix A

Train the classification algorithm in Matlab

1. load the training data to the works-pace.
2. In the app section, click on the classification learner app.
3. Click on new session and select 'From works-pace' as seen in the picture A.1 below.

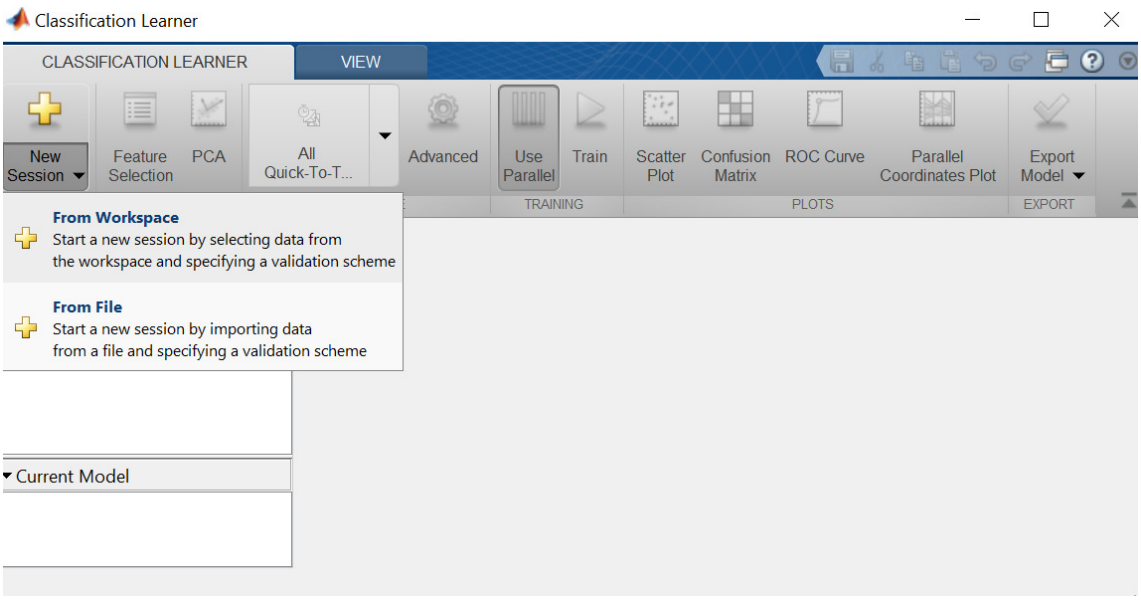


Figure A.1

4. After that select the cross validation hold out and start session.
5. Finally select the model type and PCA if needed and click on train as seen in figure A.2 below. Once the training is complete export the trained model to work-space or save it in a file.

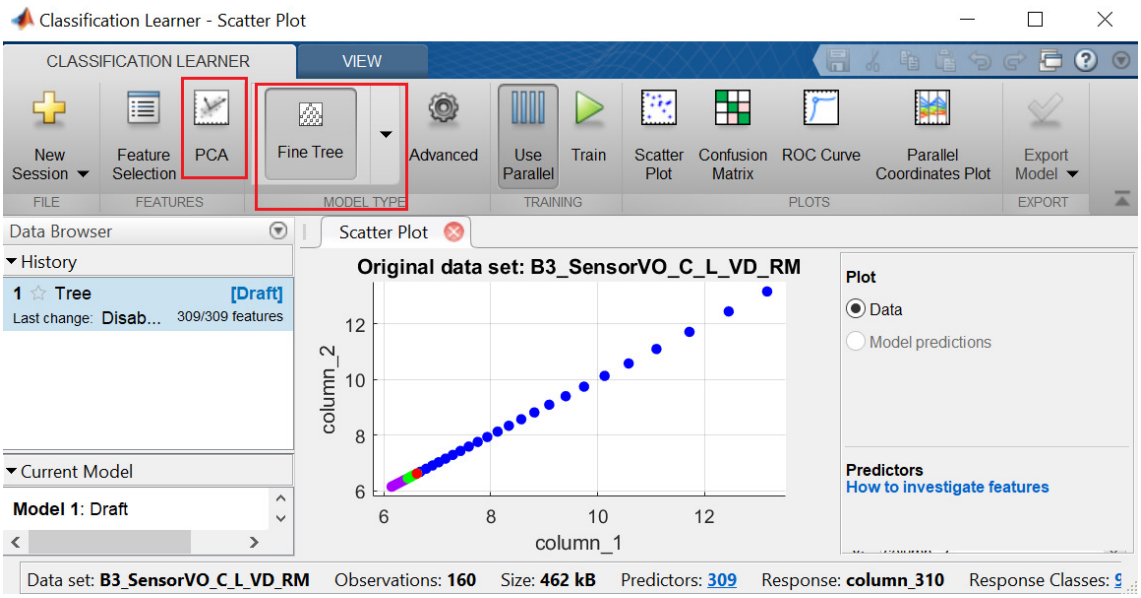

Classification Learner - Scatter Plot
CLASSIFICATION LEARNER
VIEW
New Session
Feature Selection
PCA
Fine Tree
Advanced
Use Parallel
Train
Scatter Plot
Confusion Matrix
ROC Curve
Parallel Coordinates Plot
Export Model
FILE
FEATURES
MODEL TYPE
TRAINING
PLOTS
EXPORT
Data Browser
History
1 Tree [Draft]
Last change: Disab... 309/309 features
Current Model
Model 1: Draft
Scatter Plot
Original data set: B3_SensorVO_C_L_VD_RM
column_2
column_1
6
8
10
12
Plot
Data
Model predictions
Predictors
How to investigate features
Data set: B3_SensorVO_C_L_VD_RM
Observations: 160
Size: 462 kB
Predictors: 309
Response: column_310
Response Classes:

Figure A.2