UiT

THE ARCTIC UNIVERSITY OF NORWAY

Faculty of Engineering Science and Technology

Total Productive Maintenance in An Industry 4.0 Framework

Explanation or subtitle

—

Rami Noureddine

Master's thesis in [Industrial Engineering] … June 2019

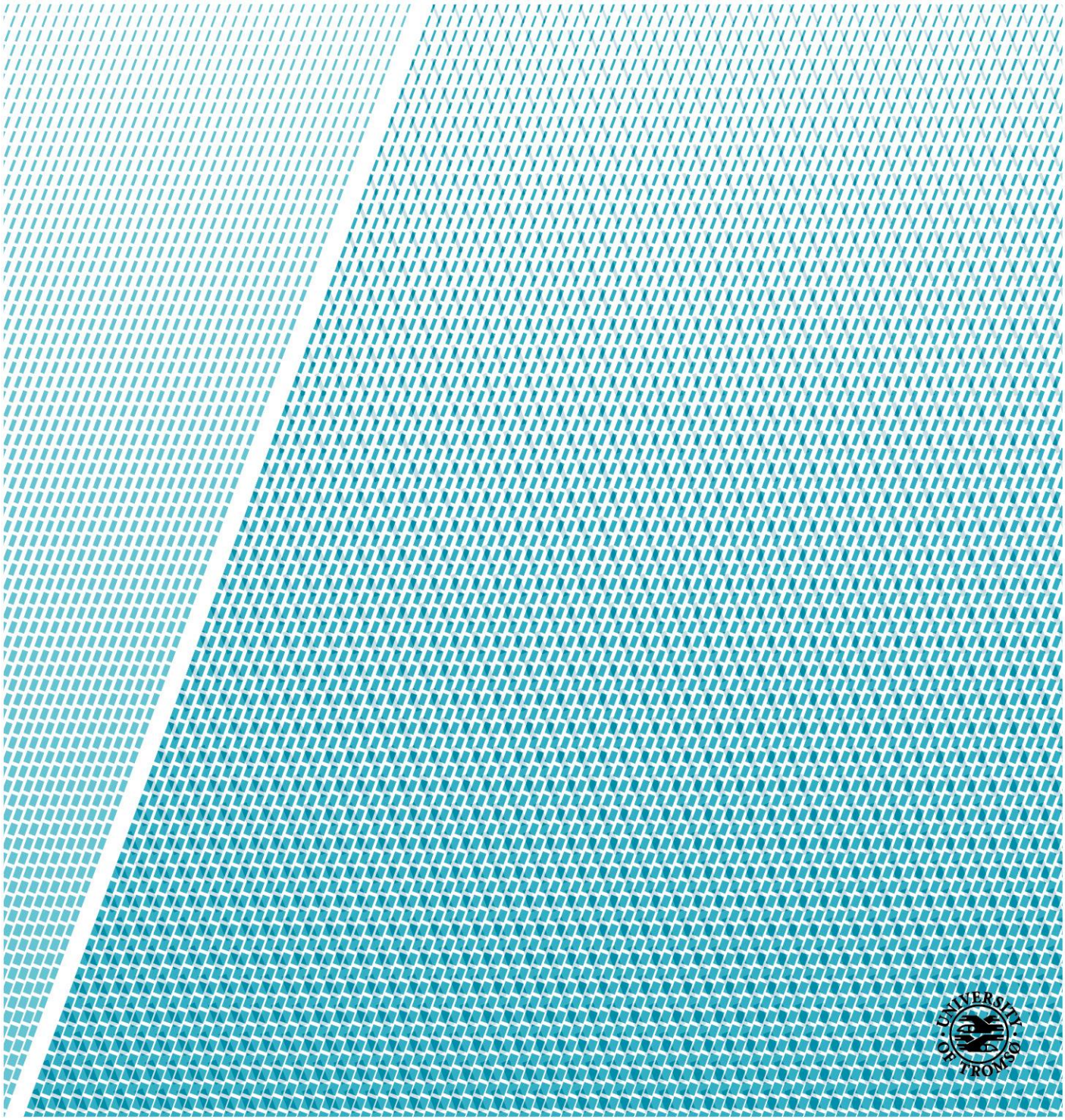


Table of Contents

List of Tables

List of Figures

Abstract

Maintenance is a key operation function that is required to improve business performance by avoiding equipment breakdown. In 1971, Total Productive Maintenance (TPM), a lean manufacturing approach, has been developed and widely used as a maintenance strategy to gain a competitive advantage in industry. However, with the advent of new technology and the internet of things, manufacturing process are subject to evolve from the old traditional ways of manufacturing to digitalized manufacturing. In this stage, the utilization of data for understanding current operating conditions and detecting faults and failures is an important topic to research. However, that alone is not enough to ensure long term survival and success in the market. Today, with the applications and technologies of Industry 4.0, components and systems are able to gain self-awareness and self-predictiveness which will provide management with more insight on the status of the factory. Systems are able to make use of both historical and live data which was not possible before. In this context, this thesis aims on developing a framework for productive and efficient maintenace with the use of Industry 4.0 technologies. The thesis discusses the new benefits that predictive maintenance has the potential of providing and it discusses several machine learning algorithms that are promising in the field of maintenacne. Throughout the thesis several models are developed and discussed to provide a framework that would ease the transition of mainentance from the old traditional ways to the newly emerging concept of a smart factory in Industry 4.0.

Keywords: Predictive Maintenance, Big Data Analytics, Machine Learning, Prognostics, Industry 4.0, Total Productive Maintenance .

1 Introduction

Maintenance plans and policies are strategic decisions for all production and manufacturing processes. Companies have been implementing different maintenance activities and strategies to improve their overall process in terms of productions costs, wastes, flexibility, time, reliability, and customer satisfaction. Nevertheless, these strategies are strongly correlated with the maturity level of companies in a sense that in order for a company to reach a new level of maturity, a good maintenance strategy is a vital decision to make and define.

The traditional ways of maintenance have evolved over time with the constant introduction of new technologies. The earliest maintenance activities are known as reactive maintenance activities where management or workers dealt with problems only after they have taken place. With developing a higher maturity level, companies switched to what is known as preventive maintenance where frequent visualization by team members are scheduled and routine inspections for the system are essential to help prevent failure in equipment or process. With the introduction of electronics and the widespread of sensors and processors, many companies have adopted to even a higher level of maturity by using what is describes as rule-based predictive maintenance strategies. In this rule-based maintenance, sensors are installed in some areas to measure specific parameters and a condition or a rule is coded to the sensor such that if the monitored parameters reach a predefined point, the system sends an alert to notify management of the process present status.

However, with today's new technologies and the advent of industry 4.0 smart factory, maintenance is expected to reach much higher dimensions in terms of maturity and efficiency than ever was possible. The main concept behind maintenance in industry 4.0 is the ability to make use of historical and live data and the ability to make predictions for future states of our process. Nevertheless, data visualization, digital twins, and augmented reality are also new technologies and concepts that provide companies with highly advanced efficient systems in maintenance and other activities in a production process.

The main goal behind this thesis is not to abolish the traditional methodologies of maintenance but rather to provide a framework of how the concepts of these developed methodologies can be implemented and driven by industry 4.0 technologies in order to reach new limits and provide better results for the production performance.

The scope includes:

1) Revising the old traditional ways of maintenance using TPM methodology and conducting a literature review on how these methods are used and implemented

2) Conducting a literature review on industry 4.0 enablers and key concepts defining a smart factory.

3) Discussing predictive maintenance and making the connection between traditional maintenance concepts and smart factory concept

4) Creating a framework for TPM in industry 4.0

5) Devolving models and case studies for applying machine learning in maintenance related tasks

6) Creating two projects of maintenance in industry 4.0 and discussing some opportunities possible to achieve in them.

With the wide scope and applications of industry 4.0 this thesis is confined to predictive maintenance in industry 4.0 and its applications by exploiting historical and live data in order to predict behavior, send alters, and enable self-awareness and autonomous decision making throughout the system. Little emphasis is done on Augmented reality and Digital twins which can be another separate research topic for future work.

The thesis is divided into ten chapters fitting the scope. In *chapter one*, a brief feedback on what is TPM and what is the concept of Lean Six Sigma in terms of reducing process variations. In *chapter two*, Industry 4.0 is discussed and its effects on traditional production process is discussed from the point of view of experts and other researches made. In *Chapter Three*, I discuss what is prognostic analysis and what are the steps for conducting it. In *chapter four*, I discuss the use of data mining and the steps for it. In *chapter five*, an example of feature extraction is presented to show how signals can be analyzed in different domains and how vital it can be to be able to analyze data in real time. In *chapter six*, I discussed predictive maintenance, emphasized on its benefits, and constructed a structure that shows how does it work. In *chapter seven*, Machine learning methods are discussed and models are developed to approach industrial cases. In *chapter nine*, Project one showing lean six sigma methodology application live data. Project two discusses a project I created with a smart algorithm that learns from historical data, reads live data and based on that sends an alert email with attachment to the optimal person to call for checkup or system review. In *chapter nine* conclusions and future work suggested are discussed.

2 *Chapter* 1. Methodology behind traditional maintenance techniques.

For at least the recent decade, many companies have been influenced to adopt Lean Six Sigma methodology to increase their profit and efficiency. Although some companies have failed due to lack of knowledge and experience in Lean Six Sigma techniques, many companies applying Lean Six Sigma projects have succeeded in reducing loses and securing a high share of the market.

Lean Six Sigma strategies are customer driven strategies that are mainly concerned about waste elimination and process improvement. Although this methodology is a comprehensive methodology encompassing the culture of the company as a whole, i.e. in maintenance, team performance, production methods, etc., Maintenance and search for root cause analysis to ensure production efficiency, stability and capability is the main core of it since the ultimate goal of the methodology is to reduce process variation and waste and to ensure continuous improvement by the DMAIC cycle.[1]

One fundamental approach in lean manufacturing is known as Total Productive Maintenance (TPM) which is a used for optimizing maintenance to achieve an efficient production system. The main aim of TPM is to prevent defects, stoppages, downtime, and accidents in a production system.

TPM is a strategic tool used in lean manufacturing that enables manufacturing industries to achieve efficient maintenance activities and decisions. By successfully implementing TPM, industry can significantly increase their competitiveness and effectiveness in the field of maintenance and thus increase profits by satisfying customer demand, just in time deliveries, production capacity, and by cutting on downtime and disruption costs or equipment failure incurred costs.

The different aspects of maintenance are all incorporated in the 8 pillars of TPM know as the elements of TPM. [2]

1) Autonomous Maintenance - requires routine maintenance activities such as cleaning, lubricating and inspection.

2) Focused improvement- focused on eliminating wastes and quality loses by incremental improvements in the operation of the equipment therefore improving overall equipment effectiveness OEE.

3) Planned maintenance – concerns areas of preventive maintenance. This area requires scheduled maintenance tasks based on failure rate or predictive analytics from observations.

4) Quality maintenance – aims to have zero quality defects by identifying root causes and monitoring causes of variation. As part of the Jidoka pillar in Lean systems, errors are detected and stopped from entering into the production system.

5) Education and training – focused on providing knowledge required to implement TPM successfully. This requires training and educational programs to staff and maintenance personnel in an industry.

6) Early equipment Management. Focused on providing better design of new equipment. This is based on previous experiences to help ensure better that lead times and other performance targets in manufacturing are reached faster.

7) Safety, Health and Environment – concerned about sustaining a healthy and safe working environment.

8) TPM in Administration - required to ensure support and services with effective communication and increased transparency across departments.

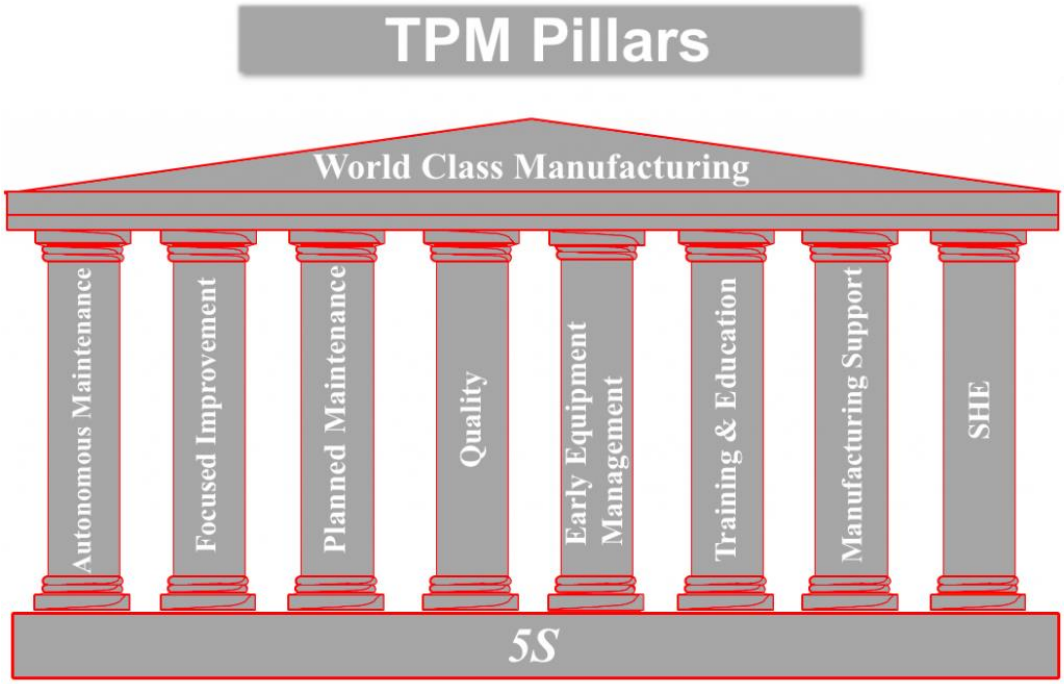


Figure 1 TPM Pillars [2]

TPM has founded the way to an important KPI that is widely used in manufacturing known as Overall Equipment Efficiency (OEE)

The similarities between the goal of TPM and OEE are too close to each other in which TPM aims for no downtime, no short stops or slow production rates, and no defects, while OEE similarly aims for availability performance and quality assurance.

3 Chapter 2. Industry 4.0

Industry 4.0 refers to the fourth revolution happening in today's modern world by the introduction of Big Data, Internet of things, and Cyber Physical Systems (CPS).

Since its introduction, many companies are competing to apply industry 4.0 methodologies to enhance business performance and work experience. Industrial businesses are investing in building global networks to connect their machinery, factories and facilities to enable efficient communication and application of cyber physical systems.

The internet of things in industry 4.0 otherwise known as the industrial internet of things (IIOT) has become attractive to many businesses due to the reduction in costs of modern-day computations, storage, and network systems as a result of the cloud computing model. In IIOT systems, big data can be analyzed online on a cloud with advanced analytics at a very high speed. This Big Data can be used by process engineers to transfer information lying in this data to valuable knowledge.
While the concept behind CPS in industry 4.0 is that they are intelligent systems containing embedded circuits that are connected to their environment. They do not only respond to specific stimulus predefined to them as in embedded systems, but also, they are able to communicate and interact with the surrounding environment. CPS systems are networked and thus are able to send and receive data from different locations. CPS allows the constructions of application that can autonomously interact with environment and execute actions accordingly.
Finally, it's important to know that the cloud in industry 4.0 provides everything as a service. Three main categories are the Infrastructure as a service (IaaS) where hardware needed and server rooms are presented as a service rather than buying it. Platform as a Service (PaaS)

which gives access to development languages, libraries, APIs, etc. Finally, the Software as a service (SaaS) which provides services by providing a new way of accessing software instead of accessing a local private server hosting a copy of the application. It provides users with web server-based shared application.

Certain technologies and protocols have been developed that enables systems in industry 4.0 to quickly access, realize, and analyze data of main interest. One such protocol is the publish /subscribe protocol which allow applications to individually subscribe to published services that are of an interest. [3]

The key technologies of industry 4.0 are listed in figure 2 below.

Figure 2 Industry 4.0 Key technologies [4]

Through the rest of this section, the impacts of industry 4.0 on the traditional methodologies discussed earlier will be emphasized. The results are obtained from research papers that have conducted surveys across different production companies and from different experts feedback.

Research done by [5] Can be summarized with the following impact matrix in table 1.

	Data Acquisition and Data Processing				Machine to Machine Communication (M2M)		Human-Machine Interaction (HMI)	
	Sensors and Actuators	Cloud Computing	Big Data	Analytics	Vertical integration	Horizontal integration	Virtual Reality	Augmented Reality
5S	+	+	+	+	+	+	++	+++
Kaizen	+	++	+++	+++	+++	+++	+++	+++
Just-in-Time	++	++	+++	+++	+++	++	+	++
Jidoka	+	+++	+++	+++	++	++	+	+
Heijunka	++	++	+++	+++	+++	++	++	+
Standardisation	++	+++	+++	+++	++	++	+++	+++
Takt time	+	+	+++	+++	+++	+++	+	+
Pull flow	++	+	+	+	+++	+++	+	+
Man-machine separation	+	+	+	+	+	+	+++	+++
People and teamwork	+	+	+	+	+	+	+++	+++
Waste reduction	+	+	++	+++	+++	+++	+	+

Table 1 Industry 4.0 impact matrix on lean production systems [5]

The above matrix considers the impact of industry 4.0 on different elements of the Lean Six Sigma methodology. Although the topics encompass areas more than maintenance, Most of the elements studies are directly related to the maintenance strategy. For instance, for the concept of man machine separation in terms of having machine being self-aware, self-diagnostic, etc. Others elements that require standardizing work procedures, conducting scheduled maintenance inspections, calling for system stop when defect or failure is noticed, etc. are all factors that can be greatly altered to become more efficient with predictive maintenance.

A more maintenance-oriented approach that considered both qualitative and quantitative measures regarding the effect of industry 4.0 on total productive maintenance pillars is described in work of [2]. Figure 3 shows the final findings and the results are discussed below.

Key technologies of Industry 4.0 (Predictors)	Significance (Sig.) value							
	Pillar 1	Pillar 2	Pillar 3	Pillar 4	Pillar 5	Pillar 6	Pillar 7	Pillar 8
AM	.497	.069	.016	.029	.102	.017	.010	.018
HMI	.000	.833	.002	.002	.001	.358	.000	.000
AR	.005	.000	.392	.110	.868	.447	.001	.356
BDA	1.000	.000	.000	.010	.001	.097	.069	.644
CS	1.000	.000	.193	1.000	.836	.016	.807	.000
M2M	1.000	.011	.008	.376	.740	.282	.696	.293
SIM	.000	.000	.004	.042	.005	.000	.000	.190
IoT	.000	.000	.014	.081	.005	.177	.016	.000

Figure 3 Industry 4.0 technologies impact on TPM Pillars. (values <0.005 represent very strong impacts)

From the figure above we can see for example how BDA (Big Data Analytics) have a strong positive impact on Pillar 2,3, and 5. We can conclude from this that using Industry 4.0, many advantages will be achieved and provided for an industry. For instance, BDA makes it feasible to research huge amount of statistic process data from sensors to detect instable processes and avoid system breakdown or quality flaws.

In general, the several ways industry 4.0 have impact on TPM pillars are discussed below

1) In autonomous maintenance (pillar 1), dashboards will be easily read and interpreted, monitoring will be enhanced, and the automation level of machines is increased.

2) In planned maintenance (pillar 2), using machine learning will give way to predictive maintenance that will reduce failures and downtime.

3) In quality management (pillar 3), sensors provide data that can be analyzed using machine learning to detect variances in machine performance and therefore alert operators directly to run root cause analysis.

4) In focused improvement (pillar 4), AI can help operators in root cause analysis for discovered defects and information and results are easily shared across functional teams allowing for better communication and collaboration.

5) In new equipment management (pillar 5), historic data of previous equipment applications, designs and installations is analyzed to help manage new equipment.

6) In Education and training (pillar 6), learning about machines, components, production lines, and facility management is enhanced using digital twin visualization and online access to data.

7) In Safety and Environment (pillar 7), harmful radiations or gases, dangerous temperatures, electric failures or power surges can be detected by sensors to maintain a healthy and safe environment.

8) In administration (pillar 8), AI will help enhance decision making.

Table 2 shows the challenges in traditional maintenance practices and what and how this thesis aims to provide solutions to them using Industry 4.0.

Dimensions of Lean Manufacturing	Challenges for lean implementation from integration perspective	Solutions provided by Industry 4.0
Total Productive Maintenance	No control of machine breakdown	Machine worker communication
	Unknown problem-solving time	Self-maintenance assessment
		Predictive maintenance system
Statistical process control	Ignorance of operators	Workpiece-machine communication
	Inability to track process variations	Improved user interface
		Process tracking, integration and control

Table 2

4 Chapter 3. Prognostics and Health management for maintenance.

Prognostics and health management (PHM) can be applied on different manufacturing processes to increase reliability, availability, and safety in the system. PHM enables operators and managers to view the overall health state of the system and to make proper decision on machine maintenance that in return would decrease maintenance costs in manufacturing.

Before building a predictive maintenance system, technicians should have a comprehensive understanding of the system/machine/component degradation mechanism and behavior under different conditions. A tool that is widely used in Lean six sigma is the FMMEA (Failure mode, mechanism, and effect analysis). This tool provides the means to analyze failures and identify root causes and the effects of different failures on the system/component. The failure mode describes how failures happens, the failure mechanisms defines the causes of the failure in the

system/component, and finally the effect analysis describes the effect the failure has on the system/component.

The steps required to successfully implement PHM are 1) Data acquisition. 2) Data processing. 3) Detection. 4) Diagnostics ad prognostics. 5)Decision making. 6) Human-machine interface. [6]

[Atamuradov et al, 2017][6]. Discussed several applications and researches for each of the above approaches and the pros and cons for each tool used in the Model-based and Data-driven approach. The results were summarized in the following tables.

Prognostics Tools	Advantages	Disadvantages	Application
Paris' law (PL)	-Model parameters are adaptable for conditional changes	-Linear correlation with defect size and vibration RMS level -Empirical determination of material constants is needed	-Bearings, gearbox, fatigue crack propagation
Forman law (FL)	-Links both monitored data and crack growth physics to life models	- Poor results for complex systems	-Fatigue crack propagation
Fatigue spall initiation/progression model (FSI/P)	-Calculates time up to initiation and from initiation up to failure -Damage is cumulative	-Many parameters to be determined	- Bearings, fatigue crack propagation
Kalman Filter (KF)	-Estimates current/future states -Estimation error is corrected with the latest measurement	-System/measurement model need to be defined -Sensitive to noise -Applicable to linear systems with Gaussian noise	- Gearbox bearings, PEMFC, batteries
Particle Filter (PF)	-Applicable to non-linear systems with non-Gaussian noise -Better accuracy -Avoids degeneracy problem by resampling	-System/measurement model need to be defined -High dimensional data increases computational cost	-PEMFC, batteries, filter clogging, fatigue crack propagation

Table 3 Model-based prognostics pros and cons

Prognostics Tools	Advantages	Disadvantages	Application
ARIMA Models	-Applicable to linear systems with stationary behavior -Uses less amount of data	-Short term prediction -Not useful for non-stationary processes	- Rotating machinery
Match Matrix (MM)	-Deals with high dimensional data -Provides long term prediction -Suitable for non-stationary processes	-Needs sufficient historical data -Data should have degradation trend	- Rotating machinery
Gaussian Mixture (GM)	-Many Gaussian functions can be used to approximate an arbitrary distribution and accuracy	-Initialization methods are important in parameter optimization -Determining number of mixtures is difficult	- Bearings, CNC machines
Gaussian Process Regression (GPR)	-Adaptable to environment and can learn from experience	-Needs covariance function determination -Suitable for Gaussian likely hood	- Nuclear power plants, batteries
Artificial Neural Networks (ANN)	-Applicable for complex systems and which have non-linear behavior -Adaptable to the system	-Network structure is not determinable -Needs resources for computation	- Bearings, batteries, turnout point machines
Fuzzy Logic (FL)	-Inputs can be imprecise noisy/incomplete -Appropriate for complex systems	-Needs rule development based on expert knowledge	- Bearings
Bayesian Networks (BN)	-The number of structure parameters are reduced by conditional probability distribution -Visualizes variable pair dependency links	-Has complex and costly learning -Prior knowledge is needed	- Bearings

Table 4 Data-driven prognostics pros and cons

5 Chapter 4. Data Mining

One framework for data mining tasks that has been developed and funded by the European community to standardize the processes in data mining is the CRISP-DM.

As seen from figure below, CRISP-DM which stands for cross-industry process for data mining presents a structured approach for data mining tasks that encourages interoperable tools across the entire data mining process [7]

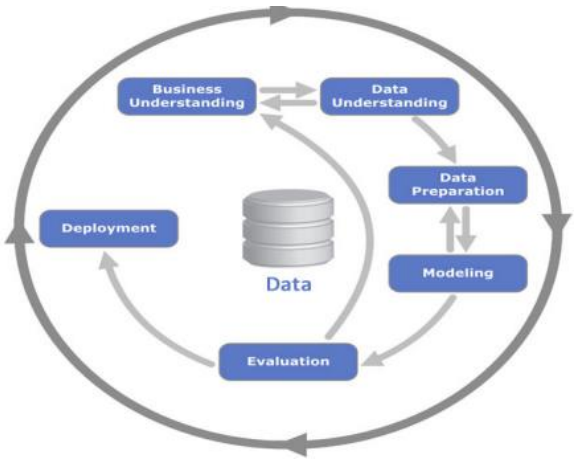


Figure 4 CRISP-DM model

The model shows a sequence of events with information flowing back and forth through different stages in the CRISP-DM methodology. These information backflows are necessary to improve functions executed in each stage.

In the first stage (Business Understanding), activities such as defining desired outputs of the project, developing project plan, defining business success criteria, and assessing current situation by defining resources_ (personnel, data, hardware platforms, and software), requirements and constraints, risks and contingencies, assumptions, costs, and terminologies are essential to develop a high level of business understanding.
For instance, the main goal of the project might be to decrease system maintenance routines and downtime by predicting when is maintenance required. In this stage, project teams might ask questions such as how different activities or conditions affect the performance or health of the system or how are the process inputs correlated with each other or with the output of the system.

In the second stage (Data understanding). Data listed in the resources for the project should be acquired and loaded into the data mining tool used for understanding it. A data collection report that defines which data is collected and from where is produced. Problems encountered and solutions accomplished should also be recorded in this report to help with improvement and predictions for similar future projects.

The Data format, quantity, identities and features in it should then be reported in the data description report.

Afterwards, data is explored and several data mining inquiries are addressed using data visualization and reporting techniques and the results are documented in the data exploration report.
Finally, in the data quality report, the quality of the data is examined to check whether the data is complete and correct, if errors are present and data is missing it should be noted how common are these errors, where do they occur, etc.

In the third stage (Data Preparation) after understanding the data. Data included and excluded are listed and reasons for the taken decision are presented. Later on, the selected data is cleansed and the quality is amplified using chosen analysis techniques. For instance, using probabilistic or deterministic models to estimate missing values in the data. All of these decisions and actions made are reported in the data cleaning report.

After new features are derived or transformed from existing ones, wanted data is constructed and information is listed in records for data integration. These new data might be discovered attributes or behavior of certain parameters that would be used for modelling.

In the fourth stage (Modeling). The modelling technique to be used is selected. These techniques, as shown previously in tables 3 and 4, might be model-based, data-driven, or hybrid approaches.
The modelling technique chosen should be documented along with the required assumptions about the data using the chosen technique. For instance, when using Kalman filters, the system is assumed to be linear with known variances in gaussian noise.
The last step required before building the model is to generate test design. This involves developing a procedure to examine the model's quality and validity. In this procedure, data is separated into train and test sets to achieve desired error rates that in turn define the quality of the model used.
Afterwards, the model is built by applying the modelling tool on an arranged dataset and results, outcomes, and experiences are listed.
Finally, the model is assessed and revised and required necessary actions are implemented. The end results along with actions executed are then all documented.

In the fifth stage (Evaluation), the degree to which the model fulfills the business objective is assessed using business success criteria and possible reasons for model failure is discussed. In addition to that, any additional knowledge given by the model is studied since it can present valuable information or potential projects for the future.
The process is then reviewed to highlight certain actions and activities that were missing or overlooked in the process and next steps required are decided.

In the final stage of the CRISP-DM model (Deployment). The strategy for the deployment of the evaluation results is determined and the proven successful procedures are documented. It is the stage where the predictive analysis results from previous data mining stages can help improve the process by providing valuable information that where undiscovered in the system.
To efficiently apply data mining and achieve successful results, the system should always be monitored and maintained to ensure correct flow and update of data.
A final report that summarizes and encompasses all previous deliverables is written and finally a meeting with the client to present the final results is scheduled.

Figure 5 summarizes the Generic tasks and outputs of the CRISP-DM model

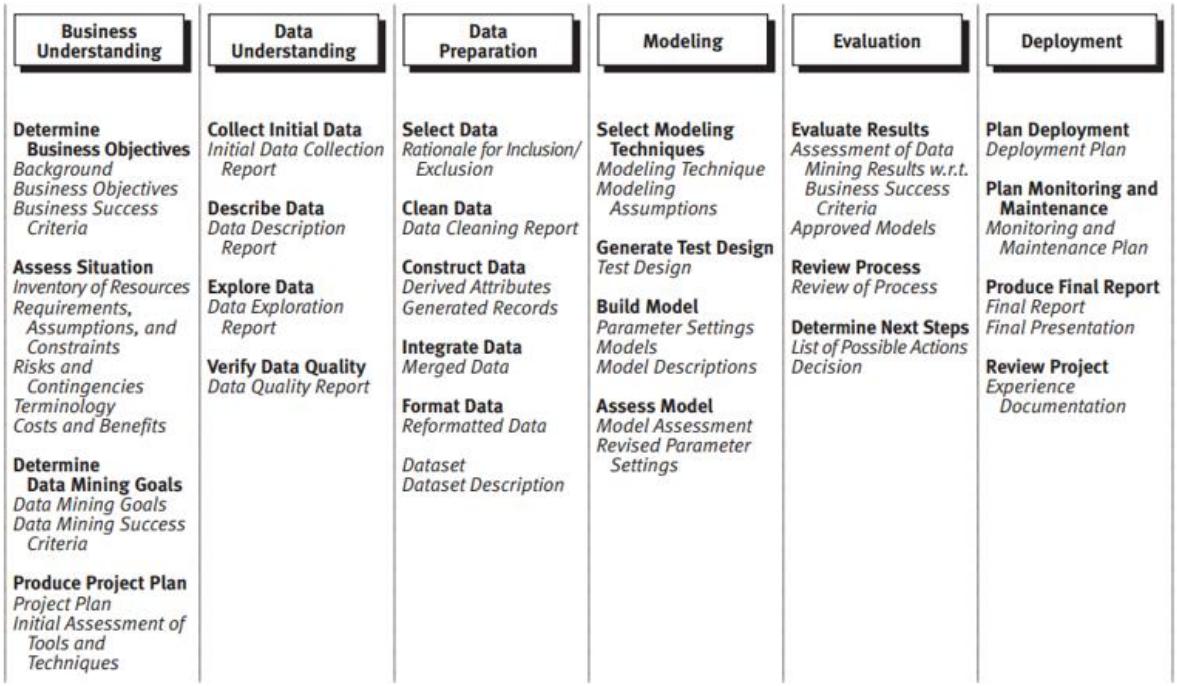


Figure 5 Generic tasks (bold) and outputs (italic) of the CRISP-DM model [7]

6 Chapter 5. Feature extraction using MatLab example

Data can be collected from different sources such as images, videos, or sensor signals. For instance, consider a noisy signal coming from an engine. Using available computer software, this signal can be studied in different domains_ time, frequency, and time-frequency domain to show different hidden information in it. In my example, the only data I was able to find online is EEG signal which I'll assume it's some engine signal with some noise in it and run a quick analysis on it using MatLab (Code available in appendix). In time domain analysis, we can visualize how does the system respond to certain inputs and calculate the mean, standard deviation, autocorrelation and cross correlations analysis on it. This data can be very useful in understanding the process of many engineering problems. The data (check appendix) consists of 23 sensors located in different places with more than 15 thousand observations. The figure below shows the signal of each sensor in time domain. Experts can have a quick idea about the signal behavior, peak values, cycles, changes vs time, etc.

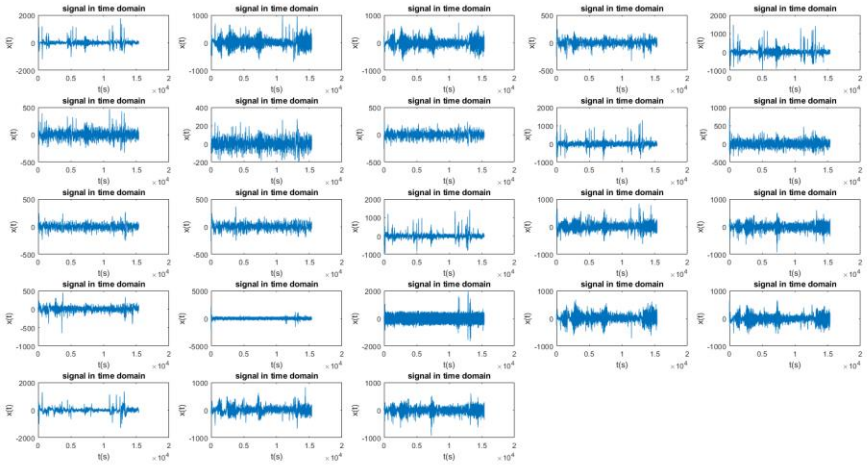


Figure 6 Signal representation in Time domain

In addition to plotting signals, the means and variances in the signals represent very useful information about the process. Experts can calculate confidence intervals, control limits, and evaluate process performance, capability, and yield from these data. These are typical practices in Lean Six Sigma through which quality managers can tell if a process is running efficiently and how to approach undesired variation root causes. Techniques and equations for calculating process performance and constructing control graphs have been discussed in my literature review in leans six sigma methodologies in industry**.** To understand this data and extract important information from it, experts makes use of histograms, box plots, baseline sigma, SIPOC analysis, and other techniques that sets a framework and a way to continuously improve a process in the cycle of DMAIC.

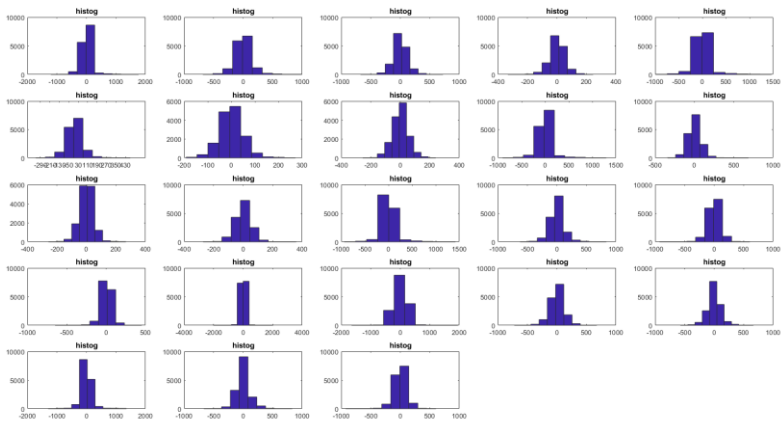


Figure 7 Histogram plotting of sensors signal

Figure 7 shows the histogram presentation of the signals. Histogram usage are necessary to estimate the density function of the Data and have an idea of the probability distribution function. Experts can then set confidence intervals, predict outcomes and other important deductions from Histograms (this will be elaborated more in the machine learning section).

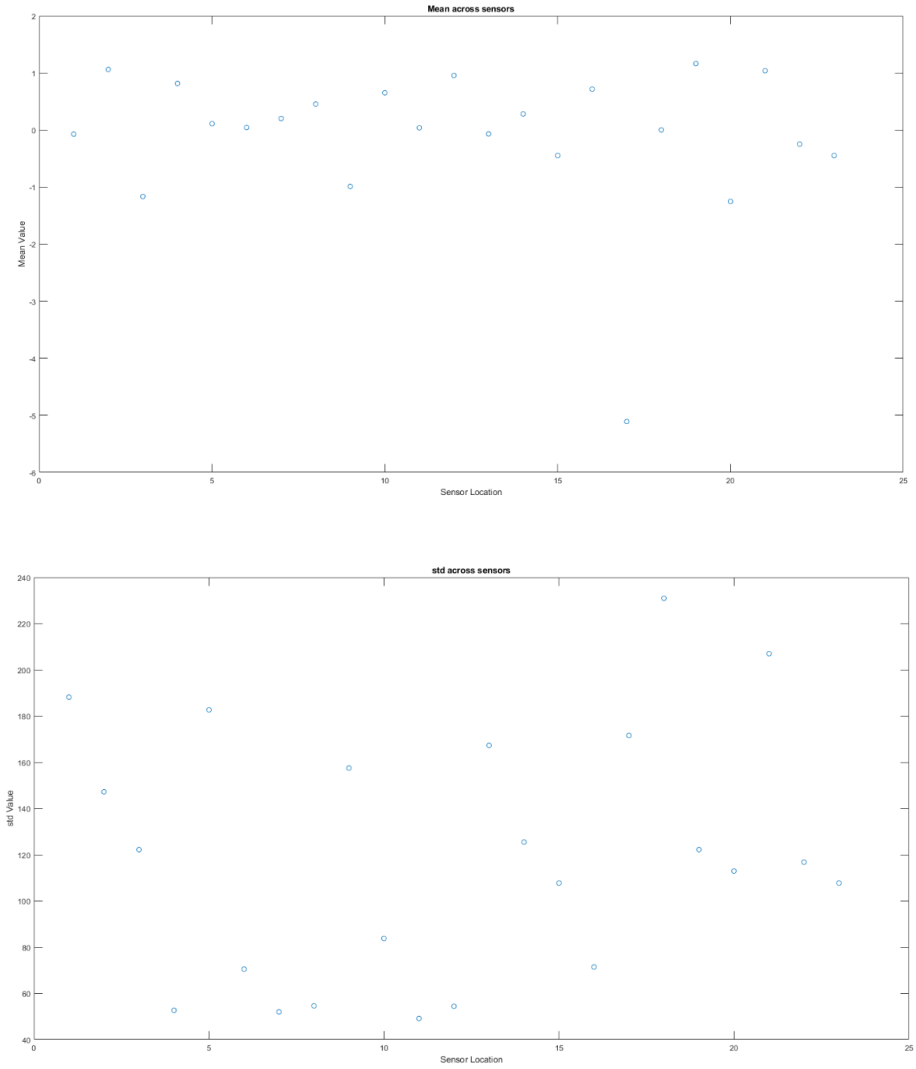


Figure 8 (top plot) shows the expected values of the signals. (Bottom plot) the standard deviation for each sensor signal.

In addition to time domain, analysis in the frequency domain can help engineers understand a lot of what's going in the process. According to Fourier series, any signal can be decomposed into a spectrum of frequencies over a continuous range. The power spectral density which is a Fourier transform of the autocorrelation function, shows the different energies at different frequencies in the signal. These energy levels hold valuable information for experts who can extract important information on the process. Figure 9 shows the power spectral density graph for each sensor signal.

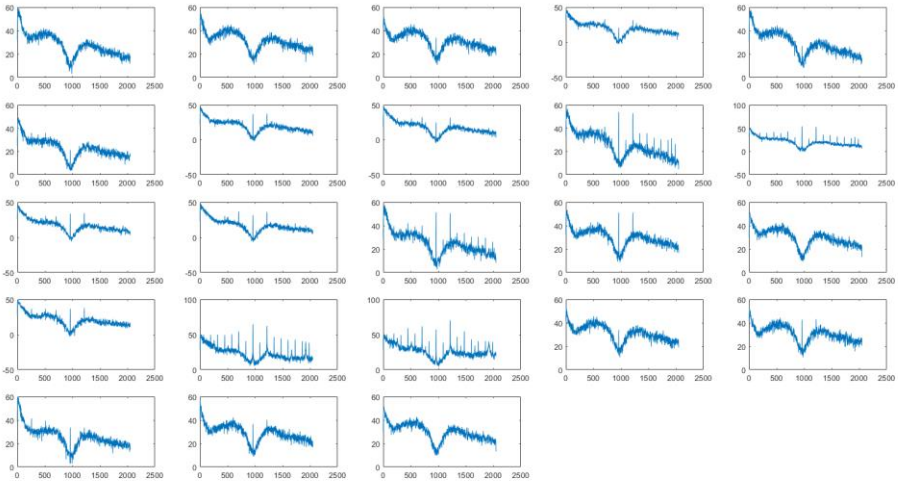


Figure 9 Power spectral density graphs for each sensor signal (Frequency domain)

Finally, the time-frequency domain analysis which combines both domains, shows how the spectrum of frequency varies with time. This has a lot of useful application such as its usage for analyzing results of passing a test signal through a signal processor. Below is the spectrograms of our studied signals.

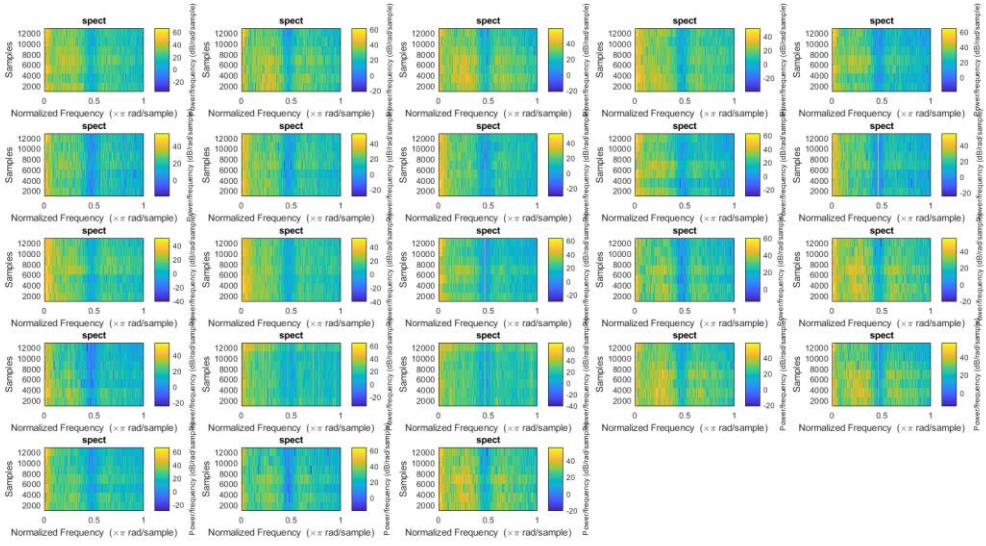


Figure 10 Spectrograms for each sensor signal (Time-Frequency domain)

With the presence of modern technology and the ability to store and analyze big data along with the internet of things where all systems are interconnected with each other, these calculations can be made easily and in real-time allowing machines to detect abnormalities in behavior and to execute actions that must be done while also updating users of the system

performance. For instance, if a certain vibration is indicating a hazardous effect on engine safety, the system might automatically stop and signal for maintenance and with data visualization, root cause analysis would become much more efficient and overall process performance improved.

7 Chapter 6. Predictive maintenance

Predictive maintenance in industry 4.0 is, as been described earlier, a method that can be used to prevent failure in process or machines by analyzing operational data and identifying patterns to predict issues before they take place allowing for just-in-time maintenance.

The main inspiration behind this thesis is that until now predictive maintenance have not been applied as a practice in most of the companies and there exist no published framework for applying this new methodology in current businesses. Previous prognostic tools applications exist mostly in other fields that require prediction such as in navigation, supply chain, space travel, etc. While in maintenance, the practice is still on small scale and in the research phase. Some big and advanced companies such as Equinor are using predictive maintenance and investing more in it, yet the practices and results are kept private without a developed structure that can guide rising companies and provide them a clear vision of the potentials in predictive and smart maintenance

Therefore, the traditional method that is widely used until today in maintenance is defined by assigning scheduled check-ups and repairs to the production process and deploying programmed sensors with predefined conditions without making use of big data stored or live data streaming in

This thesis highlights the need of predictive maintenance as a feature of industry 4.0 to enable companies increase uptime, lower service costs, and improve the production quantity and quality. It’s supposed to serve as a guideline or an informative research paper to companies to encourage them on the transition to a smart factory and to give them an idea of the potentials that can be achieved by Industry 4.0 data-driven analytics.
In industry 4.0, maintenance becomes machine-learning based for smart decisions. This is made possible by using cyber physical systems and IIoT to monitor systems, send alerts, share information and create smart and optimized maintenance schedules.

Below are some benefits guaranteed with efficient use of predictive analytics

1) Reduced maintenance time. Maintenance will only be done when necessary.
2) Increased efficiency. Unnecessary maintenance is reduced, root cause analysis becomes easier and even automatic.
3) Improved customer satisfaction. Customer can be sent alerts to inform them of product status or suggest actions for them regarding product health.
4) Competitiveness. Companies will gain a competitive advantage in the market by differentiating the products and brand.

In order to transition a company to a smart factory, companies would have to prepare an appropriate structure that can sustain the concept of it. As a result, companies should invest in several basic components and tools that can make the manufacturing system operate in a smart factory.

These components and tools include:

- sensors that should be installed in the system to monitor behavior and encode system performance, efficiency, and status.
- Data-analysis tools that are then needed to allow for root cause analysis
- Analytic algorithms that should be used to allow for predictive maintenance and smart diagnostics
- A communication system that allows for data to be safely stored and transferred across different machines and team.
- A central place for data storage, this can be indoors or can be cloud based.

The structure should allow data to flow from production process into the central data storage area where data from different systems and devices are gathered. Afterwards, this data should be sent into Machine learning algorithms for extracting knowledge, features, patters, classes, and relations in data. After data is processed out of machine learning algorithms, results are sent to dashboards for visualization of system status and predicted behavior for the future, in addition to that messages or alarms are sent to the right people at the right time to notify them of an outcome that has happened or about to happen in the production process. Data should also be able to flow in the reverse direction where the output of the machine learning algorithms can become as inputs for autonomous decision making and actions executed.

Figure 11 showing industry 4.0 predictive maintenance structure, developed in this thesis, is an updated structure to existing models that consider a one-way direction flow of information between the different levels in a company.

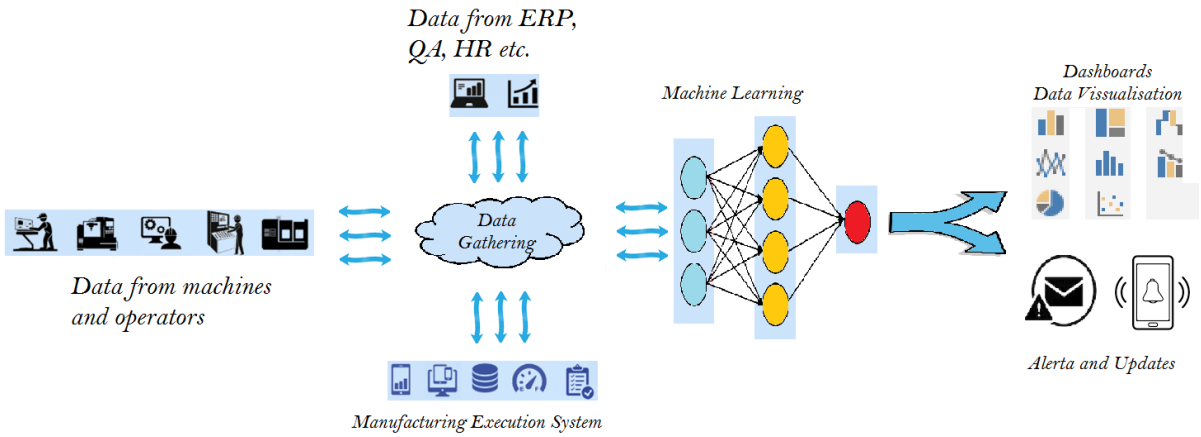


Figure 11 Industry 4.0 predictive analytics structure

Furthermore, to support industry 4.0 predictive analytics, it is necessary that the company develops blue prints on system map and connections, failure data and any observed behavior. This is important since these labeled data are useful for many machine learning algorithms as will be seen later in the thesis.

As mentioned earlier, in addition to machine learning algorithms, maintenance in industry 4.0 is revolutionized with the new emerging technology of Augmented Reality (AR). With AR, system information, maintenance procedures, technical instructions, process mapping, decisions, sensor values, and information gathered from machine learning algorithms can exist as a shadow accompanying every equipment in the system. This can cause decrease in training time, provide better transparency to the process to help in identifying root causes, improve work environment, enhance information sharing, and support maintenance operations across all technicians. Furthermore, with virtual reality, maintenance can also be optimized by testing solutions and running simulations in the virtual world by digitally representing the system in a digital twin. However, AR and VR make use of the structure provided in the figure above since, for instance, for a machine to have shadow information about its remaining useful life and present it in a graph, the machine should be able to gather this data from machine learning algorithms that can make predictions on its future status. Nevertheless, data-driven decisions should also be reflected in the digital twin software.

In this thesis as defined by its scope, the focus will be just on modeling systems and implementing machine learning algorithms that will enable machines and production equipment to be smart devices to gather, analyze, and output data and actions.

8 Chapter 7. Machine Learning

Machine learning is an intelligent methodology that can serves as a valuable tool for classification and prediction purposes.

In the machine learning process, the steps are to collect data and extract features as described before, then create a model to use for classification or predictions to data, afterwards the model should be validated and test for accuracy on test data. Finally, if the model is approved it is deployed to the process.

The main power and benefit behind machine learning is that it uses statistical learning to be able to estimate the process.
The statistical learning is about finding an estimate, ŷ, for y=f(x)+ε such that the expected value of the residual E(y- ŷ) is as small as possible.
ε here is a random error that is independent of x and has an expected value equal to 0 and f(x) is the mathematical function of the process studied.

The equation E(y- ŷ) has both a reducible and irreducible terms as shown by the equation below

$$\text{E(y- ŷ)= } [f(x) + \epsilon - \hat{f}(x)]^2 + Var(\epsilon),$$

where $[f(x) + \epsilon - \hat{f}(x)]^2$ is reducible and $Var(\epsilon)$ is irreducible.

The irreducible term will always be a limiting bound for the accuracy of the prediction of y and is usually unknown in practice. Such an irreducible error may result from errors in data measuring, missing parameters, etc. While the reducible error is what should be minimized to achieve an efficient model that represents the actual process.

In statistical learning there are parametric and nonparametric learning techniques and are either inferential or predictive for the function $f(x)$. For the parametric machine learning algorithms and hence inferential such as in linear regression analysis, the relationship between the input **X** and output **Y** should be understood and the form of the function f(x) should be

assumed to reduce the number of possibilities that would greatly simplify the process. The steps included to implement a parametric machine learning algorithm includes first selecting a form for the function f and then using the training data to learn the optimal coefficients of that function f. For instance, Algorithms for linear regression analysis are some examples for parametric machine learning algorithms. Say Tool-life= B0+ B1*age + B2*load (we need to know 3 parameters here and thus function f for the problem has been reduced significantly)
On the other hand, in non-parametric algorithms that are predictive, no strong and obvious assumptions about the shape of the mapping function f are made, therefore we have more open possibilities and hence they can learn any functional form for the available training data. This makes non-parametric algorithms more flexible, however, they require more training data and time to learn the relationship of the system variables and constrain the model space. Neural networks, decision trees, and Support vector machines that will be discussed later are examples of non-parametric machine learning algorithms.
These different algorithms provide different levels of flexibility and interpretability where flexibility is about the degrees of freedom and its goal is to have an accurate prediction of the function f behavior from the trained data and interpretability is about the ability for us to be able to interpret the function f. Figure below represents the tradeoff that exists between these two factors given different machine learning algorithms.

Figure 12 Different machine learning algorithms flexibility/interpretability tradeoff [8]

An important aspect in machine learning is the train vs test error and bias vs variance tradeoff. Training error is the error that results between the training set of data and the resulting model whereas the test error is the error between the developed model and a new data set. The quality of fit can then be measured by the mean squared error (MSE).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i)^2$$

If a statistical method has a high error between the deterministic function and the estimate function, it is said to be highly biased and therefore it is not able to replicate the reality between input and output. This bias results from the simplifying assumptions made by the model to make the target function simpler to learn and thus less flexible. The highly biased algorithms are not powerful tools for predicting highly complex problems that can hardly fit simplifying assumptions corresponding to known functional forms. However, there exist a tradeoff between bias and variance which measures the variation due to different data sets. i.e. If bias is increased, variation is decreased and vice versa.
The main objective for machine learning in predictive modeling is to decrease both bias and variance to an optimal level.

The figure below shows an example of the difference in MSE and the bias/variance trade-off using different statistical methods based on their flexibility (number of epochs). Note that the shape varies across different data sets.

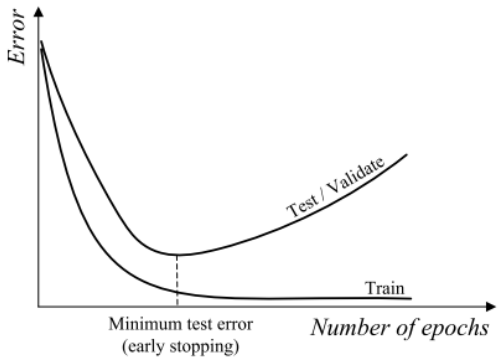


Figure 13 Bias/Variance tradeoff using different statistical methods [8]

The test error decreases until a certain point when the model accurately replicate the trained data yet loses its ability to accurately predict new data, therefore, overfitting shot be avoided by ideally stopping the training of data set at the minimum value of test error.

The expected test error (squared) is given by:

$$\mathrm{E}[(\mathrm{y} - \hat{\mathrm{y}})^2] = [f - E(\hat{\mathrm{y}})]^2 + Var(y) + Var(\hat{\mathrm{y}}) = \sigma^2 + Var(\hat{\mathrm{y}}) + Bias,$$

Where y = f + ε with **ε ~(0, σ),** Bias =$[f - E(\hat{y})]^2$, and Var[y] =Var[**ε]=** σ^2. Note that here $\sigma^2$is not reducible while $Var(\hat{y}) + Bias$ is the reducible part.

Most of the common machine learning algorithm are also classifies to be either supervised or non-supervised. Supervised machine learning algorithms require labeled data as discussed earlier with the need for companies to keep blue prints of their failure data. While in the Unsupervised machine learning algorithms, data is not needed to be labeled, the algorithm is able to classify it based on feature detection and measuring anomalies and differences between different data sets. Figure 14 shows the different algorithms that will be discussed and modeled in python in this thesis to present applications of machine learning in maintenance and serve as a guideline for any future work in predictive maintenance.

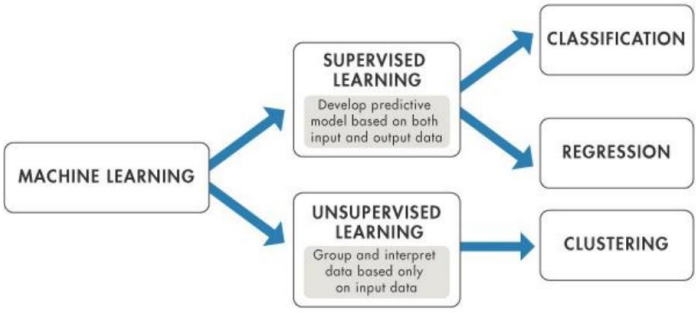


Figure 14 Machine learning Supervised vs Unsupervised [9]

8.1 Regression analysis in machine learning.

Regression analysis is typically used to model the relationship between inputs and outputs, make prediction for future behavior of the system, or testing of hypotheses. When estimating the unknow parameters in a regression model, the model is often not the true representation of the real system but rather an estimation of it, in addition to that, since a regression model is a statistical model, it is subject to change due to new data gathered, Therefore, a regression model needs to keep analyzing the current model for possible changes and updates.
The framework is described in a way where there is a specific variable that we seek to understand or model. Since this variable is a result of certain actions or other inputs that we are targeting, it is referred to as the target, response, or dependent variable and it is represented by "y". The input variables that might or might not be directly correlated to the target variable y are known as the independent, or predicting variables and are represented by $x_1, x_2, \ldots\ldots x_n$.

8.1.1 The linear-regression model.

The linear regression model describes a linear relationship between dependent and independent variables. Since a simple linear regression model is seldom the case in real world application, this paper will discuss multiple linear regression models in which the data consists of n sets of observation that represent a random sample from the population. Thus, given n sets of observation $\{X_{1i}, X_{1i}, X_{1i}, \ldots, X_{pi}, Y_i\}$ where p is the number of independent variables and i is the ith observation. **y** in a multiple linear regression model is said to satisfy the following linear relationship

$$y = X\beta + \varepsilon$$

Where **y, x, β, and ε** are all vectors

$$X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{p1} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \cdots & x_{pn} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

β are the unknown parameters that need to be found and ε are the random error terms [10] This is said to be a linear relationship since y can be represented by a linear combination of *p* number of β.

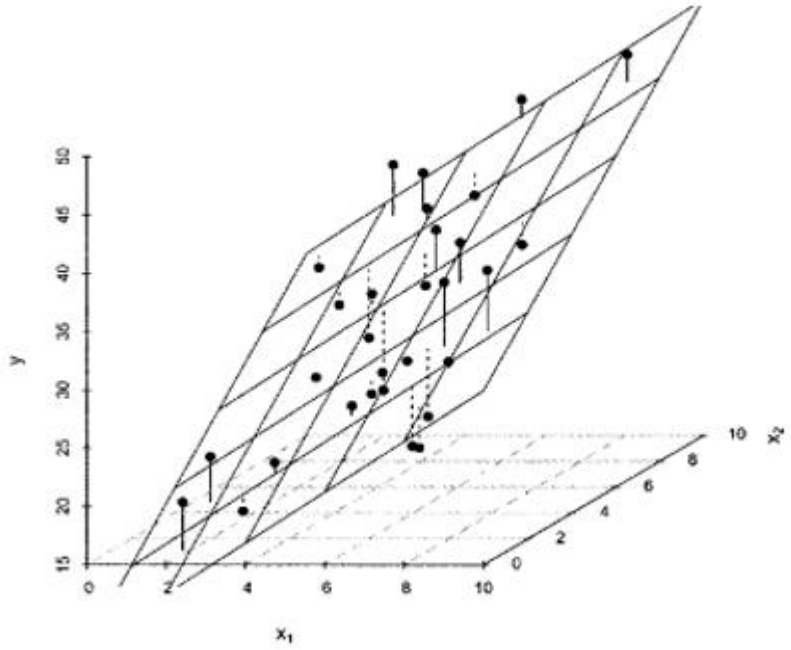


Figure 15 Least square estimation for three-dimensional linear regression models, with increasing dimensions, the estimated regression becomes a hyperplane as will be seen later in Support Vector Machines

The estimates are selected in a way that minimizes the summation of the square error between real value "**y**" and the estimated value of "$\hat{\mathbf{y}}$".

$$\sum_{i=1}^{n}[y_i - (\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi})]^2.$$

Where $\hat{y}_i$called "fitted value" is an estimation of y given by

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \cdots + \hat{\beta}_p x_{pi},$$

In the matrix form, $\hat{\boldsymbol{y}}$ can be re-written as

$\hat{y} = X\hat{\beta}$

Where $\hat{\beta}$ is a matrix of the estimated coefficients calculated by:

$\hat{\beta} = (X'X)^{-1}X'y$

Finally, writing $X(X'X)^{-1}X = \boldsymbol{H}$, the difference between the **yi** and $\hat{\mathbf{y}}$, known as the residual, is calculated by the following equation:

$$\mathbf{e} = (I - H)\mathbf{y}.$$

There are several assumptions that should hold true in order to be able to gain sensible results. First, the error expected value should be equal to 0. Second, errors should not be correlated with each other, i.e. error from a previous observation cannot predict any information about the error in future observations. Third, the errors are normally distributed and have a constant variance, thus there is a confidence interval that we would expect the observed value y to be in.

The variance of this error called **residual mean square** is given by the following equation:

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n - p - 1}.$$

The square root of the residual mean square is called the **standard error of the estimate.**

The **coefficient of determination**, R^2, and the **adjusted** $\boldsymbol{R^2}$ denoted as Ra^2 is used to measure the strength of the regression relationship.
R^2 is calculated as the correlation between the target variable and estimated variable R^2= corr(y_i, $\hat{y}_i$), while Ra^2 is just an adjusted R^2 that is used to compensate for the bias in R^2.

$$R_a^2 = R^2 - \frac{p}{n-p-1}\left(1 - R^2\right).$$

P is the number of independent variables, and n is the number of observations.

The above equation shows how there exists a tradeoff between the strength of the fit and the complexity of the system and is used to help chose the model in regression analysis.

Furthermore, hypothesis test in regression analysis are made by applying the F-test and t-test. The F-test is used to answer the question is there any independent variable that provides predictive power for the dependent variable and therefore it addresses the overall significance of the regression.
This is tested based on the null hypotheses that all **β** are equals to 0. While the alternative hypothesis states that there exists some **β** that is not equal to 0.
The t-test on the other hand tests the significance value of a specific independent variable **βj.** For the equations of these tests refer back to regression analysis handbook [10]

It is important to calculate the confidence interval for our analysis results. for instance, in a simple regression analysis a person might say we are 95% confident that increasing system temperature by one unit causes a change in the system stability between 0.02 and 0.05.

This confidence interval used to describe the degree of precision of the parameter **β** is calculated by the following formula:

$$\hat{\beta}_j \pm t_{\alpha/2}^{n-p-1}\widehat{\text{s.e.}}(\hat{\beta}_j)$$

Where the t term is the critical value using two-sided values and s.e(βj) is the standard error of βj.

Finally, it is important to state the **prediction interval** for describing interval estimate for a predicted value given a particular value in the independent variables, and the and **confidence interval for a fitted value** for describing the probability that the true best-fit in of the

regression line lies within the confidence interval.
The equation for calculating **prediction interval** is:

$$\hat{y}_0 \pm t_{\alpha/2}^{n-p-1}\widehat{\text{s.e.}}(\hat{y}_0^P)$$

Where the standard error of the predicted variable is

$\hat{V}(\hat{y}_0^P) = [1 + \mathbf{x}_0'(X'X)^{-1}\mathbf{x}_0]\hat{\sigma}^2$: **x0** is a certain value of **x** chosen.

$$\widehat{\text{s.e.}}(\hat{y}_0^P) = \sqrt{\hat{V}(\hat{y}_0^P)}$$

The equation of the **confidence interval for a fitted value** is the same as that for the prediction interval however instead of the standard error for the predicted variable it is the standard error for a fitted value and is given by:

$\hat{V}(\hat{y}_0^F) = \mathbf{x}_0'(X'X)^{-1}\mathbf{x}_0\hat{\sigma}^2$: **x0** is a certain value of **x** chosen

$$\widehat{\text{s.e.}}(\hat{y}_0^F) = \sqrt{\hat{V}(\hat{y}_0^F)}$$

Figure 16 below shows difference between confidence and prediction interval

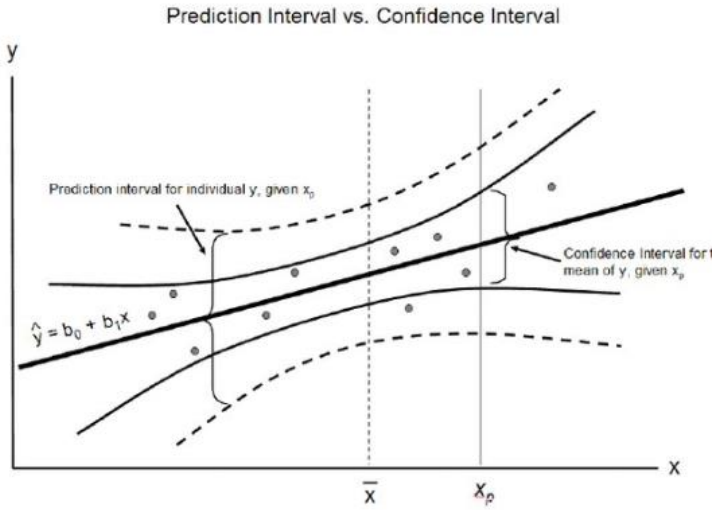


Figure 16 Difference between confidence and prediction interval [11]

Consider for example an engineer responsible of the maintenance of feeding pipes for aquaculture pens needs to predict the amount of debris in a feeding pipe that he/she might observe given a specific temperature, feeding rate, food density, and time. The prediction interval in this sense provides a better approximation than by simply using the standard error of the estimate. Hence, the engineer in that case can have an idea of what to expect under a certain level of confidence. However, if the engineer needs to know the average output of the system given these specific inputs, he/she is required to find the confidence interval of the fitted value.

Model selection is a necessary decision to make in order to avoid overfitting and underfitting situations that in turn results in poor prediction performance. Therefore, a fit model must be simplified to the point that it only encompasses the significant relationships in the data.

The F and t test along with the strength of fit R^2 value alone cannot be sufficient to test model fitness since values might be misleading when collinearity between predicting variables exist. As collinearity increases towards ± 1, the variation in parameters tend to increase to infinity without significantly affecting the value of R^2 and the overall F test. It is thus necessary to check for collinearity and simplify the regression model by omitting unnecessary inputs or assuming certain restrictions. This decision can be made after conducting a test on the variance inflation factor (VIF) for each predicting variable or by certain knowledge and experience related to a specific case of study. A VIF that is close to 1 indicates low multicollinearity while a VIF that lies between 5 and 10 indicates high correlation that should be examined, and finally, if VIF is greater than 10, it is assured that the regression coefficients will be poorly estimated because of the multicollinearity. In addition to that, F statistics test is only effective on nested models and is misleading when considering either large sample size of data or small sample size of data.

Therefore, useful techniques that reflects the need for simplicity and are used for model selection includes analyzing the system at the best combination of predictors over the different number of predictors to be modeled in the system. This is done be assuming different number of predictors to be studies and then calculating the adjusted R^2 for each model and choosing the model with highest adjusted R^2 , or calculating the Mallows C_p and choosing the model with lowest C_p, or calculated the Corrected AIC and choosing the model with lowest value, or finally by calculating the residual mean square and choosing the model with lowest value that has a simple form.

The following data obtained from existing packages in a statistical analysis software illustrates how a model is evaluated. Since it is rare to have a maintenance problem that has simple linear relationship between all input and output variables. In this example we illustrate how to interpret results since it will help us interpret results from more complicated regression models that will be modeled and explained later on in the chapter. For now, home pricing from [10] is studied given number of bedrooms, bathroom's, living area, lot size, year built, and property tax.

```
Coefficients:
              Estimate Std.Error t value Pr(>|t|)    VIF
(Intercept)  -7.149e+06 3.820e+06  -1.871 0.065043       .
Bedrooms     -1.229e+04 9.347e+03  -1.315 0.192361 1.262
Bathrooms     5.170e+04 1.309e+04   3.948 0.000171 1.420 ***
Living.area   6.590e+01 1.598e+01   4.124 9.22e-05 1.661 ***
Lot.size     -8.971e-01 4.194e+00  -0.214 0.831197 1.074
Year.built    3.761e+03 1.963e+03   1.916 0.058981 1.242 .
Property.tax  1.476e+00 2.832e+00   0.521 0.603734 1.300
---
Signif. codes:
   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 47380 on 78 degrees of freedom
Multiple R-squared: 0.5065,     Adjusted R-squared: 0.4685
F-statistic: 13.34 on 6 and 78 DF,  p-value: 2.416e-10
```

Vars	R-Sq	R-Sq(adj)	Mallows Cp	AICc	S	Bedrooms	Bathrooms	Living.area	Lot.size	Year.built	Property.tax
1	35.3	34.6	21.2	1849.9	52576		X				
1	29.4	28.6	30.6	1857.3	54932			X			
1	10.6	9.5	60.3	1877.4	61828						X
2	46.6	45.2	5.5	1835.7	48091		X	X			
2	38.9	37.5	17.5	1847.0	51397			X		X	
2	37.8	36.3	19.3	1848.6	51870		X				X
3	49.4	47.5	3.0	1833.1	47092		X	X		X	
3	48.2	46.3	4.9	1835.0	47635	X	X	X			
3	46.6	44.7	7.3	1837.5	48346		X	X			X
4	50.4	48.0	3.3	1833.3	46885	X	X	X		X	
4	49.5	47.0	4.7	1834.8	47304		X	X		X	X
4	49.4	46.9	5.0	1835.1	47380		X	X	X	X	
5	50.6	47.5	5.0	1835.0	47094	X	X	X		X	X
5	50.5	47.3	5.3	1835.2	47162	X	X	X	X	X	
5	49.6	46.4	6.7	1836.8	47599		X	X	X	X	X
6	50.6	46.9	7.0	1836.9	47381	X	X	X	X	X	X

As seen from the data above, the VIF for predicting variables is low thus they imply low collinearity between them. The significant values of the predicting variables shows that bathrooms, and living areas are most statistically significant, and living area along with bedrooms are fairly significant.
By calculating the R-sq adjusted, Mallows Cp, AICc, and S. we can see that the best option for the R-sq is with four predicting variables, while looking at the lowest Co, AICc, we see that with three predicting variables provides a good candidate. Finally, the residual mean square S has best solutions on 4 and three predicting variables given that the model should be simplified as much as possible. A model is thus selected based on these criteria and tested for verification.

In case of categorical data, dummy variables that are binary variables are used to designate membership in a particular group that has or lacks a specific attribute or characteristic. Thus, the logic of the regression equation remains the same. [12] For example, the quality level of a certain product given that it is either of recycled or non-recycled material. By adding these dummy variables in the regression equation one can study and test whether it is better to use the pooled model where the data category is ignored or the constant-shift model where parallel regression equations exists, or the full model that represents diverse regression equations.

8.1.2 Dealing with outliers

To have an efficient regression model, the model should behave in a consistent way even when unusual observations takes place. Such unusual observations, known as Outliers, can cause significant distortion and misrepresentations that would not allow for an efficient model. One way to locate and eliminate the outliers since they are presumed to be normally distributed is to divide the residual by its standard deviation and filter all values outside ± 2.5 which represents about 1% of the possibilities. Other techniques such as using robust regression analysis is discussed later on.

8.1.3 The Ridge and Lasso for reduced regression

The ordinary linear regression as discussed before can also loose accuracy when high dimensional data is studied relative to data size *n*. Suppose for instance, in testing the overall health of a certain machine, the number of explanatory variables studied is too large relative

to the number of observations available. This causes the problem of overfitting and the model becomes highly variant with approximately 0 bias. For linear regression, their exists reduced regression techniques, such as the Ridge or Lasso regression analysis, that can deal with such situations to increase the accuracy of the model.

For *Ridge regression*.
Instead of only minimizing the sum of squared residuals, the Ridge regression minimizes the sum of squared residuals plus some penalty value $\lambda * slope^2$. Thus, the ridge regression shrinks the coefficients and allows us to reduce model complexity.
Increasing λ causes the response variable *y* to be less sensitive to predictor as the variance of the model decreases and the bias increases.
The optimal value for λ to improve the accuracy of the model for prediction is chosen by cross validation_ a technique which splits a data set into K classes and iterates through the classes choosing each time one class for validation. The value that yields the best result is chosen. [12]

The *Lasso regression* on the other hand is very similar to the Ridge regression, however, the penalty value used is $|\lambda * slope|$. Although they have similar effects by reducing complexity in the model, increasing λ in lasso to a certain value pulls the coefficient values for some predictor variables to 0 while the Ridge regression can only get them close asymptotically to 0. Given that, although they can be used interchangeably in many occasions, Ridge regression can be a better choice in cases where all data used are useful whereas Lasso regression can be a better choice if some data is need to be excluded in a case study that has lots of useless data.[13]

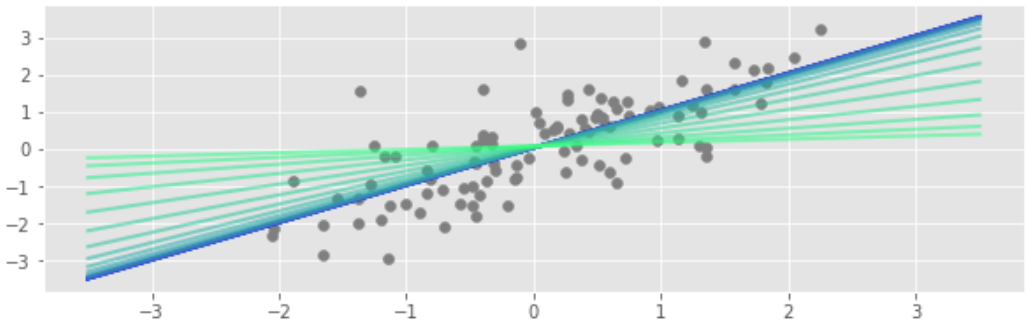


Figure 17 Shows how the slope or coefficient of the predicting variables can be decreased or even excluded by ridge or lasso regression respectively

8.1.4 Nonlinear regression analysis

If the relation between dependent and independent variables are non-linear, there are several approaches to model the system. For instance, in case of a quadratic relation between input (height "h") variable, and output (weight "w") variable the model would be a multiple linear regression model adding h^2 to the equation as shown below [14].

$$w_i = \beta_1 + \beta_2 h_i + \beta_e h_i^2 + \varepsilon_i.$$

Other cases might require different techniques such as using non-linear regression or by the transformation of linearizable models that transforms the non-linear relationships to linear relationships. One common technique for such transformation is the logarithmic transformation_ either using a log-log model or a semi log model.

The log-log model means both the response and predictor variables are logged and it represents a multiplicative-multiplicative relationship where 1% change in x_i holding other variables makes a proportional β_i% change in y. The form of the equation is given by:

$$\ln(y) = \beta_0 + \beta_1 \ln(x).$$

In the semi log model, if the response variable alone is logged as in the equation below

$$\ln(y) = \beta_0 + \beta_1 x \rightarrow y = e^{\beta_0 + \beta_1 x} \text{ or } = e^{\beta_0} e^{\beta_1 x},$$

It represents an additive-multiplicative relationship in which a single unit increase in x_i, holding others stable, causes a e^{β_i} multiplicative change in y.

While if the predictor variable is logged, having the form

$$y = \beta_0 + \beta_1 \ln(x),$$

The relation is multiplicative-additive in which multiplying x_i by 10 or e; weather using log or ln respectively, and holding all other variables stable, cases β_i additive change in y. An illustration of this will be seen in the logistic regression example later on.

However, some problems are not linearizable and therefore non-linear regression is used. Non-linear regression has more flexibility than the linear regression in general and can be more appropriate to use than linearizable semi-log problems. However, for many problems it

is hard to generate the correct mathematical model of the physical process.
The nonlinear regression model is defined by the following equation as before:

y_i=f (x_i, θ)+ε_i

If the error ε satisfies a gaussian distribution, the nonlinear regression uses the principals of the Maximum likelihood where parameters are chosen to have values that provides the data observed the greatest probability of happening.

The non-liner least square estimation is don't by minimizing the sum of squares given by

F=$\sum_{i=1}^{n}[yi - \text{f}\ (x_i, \theta)]2$

The partial derivatives of F for each parameter is taken and set to equal 0. This will generate **j** number of formulas with j unknowns. Newton-Raphson method or other methods may be used to find solutions to this minimization problems.

As discussed before, outliers can cause the regression model loose efficiency by giving biased results that cannot be interpreted. For some problems however, outliers are needed to be kept inside the data and not removed. In such case, **robust regression** is used to overcome the influence of extreme observations that are bound to happen in measuring physical processes. This is done by using loss function ρ for the residual i.e. $\rho(r)$. The asymptotically optima form for ρ corresponds to the classical least square's regression in the following equation:

L2: $\rho(r_i)$= ${r_i}^2$

Other loss functions used in robust regression are the following. [15]

Absolute deviation Loss (L1): $\rho(r_i) = |r_i|$

Huber loss: $\rho(r_i)$=$\begin{cases} \frac{1}{2}{r_i}^2 & |r_i| \le 1 \\ \delta\left(|{r_i}^2| - \frac{1}{2}\delta\right) & Otherwise \end{cases}$, for any positive real δ

Smooth Approximation to Absolute Value Loss: $\rho(r_i)$= 2*$\sqrt{1 + r_i} - 1$

Cauchy Loss: $\rho(r_i)$= ln (r_i +1)

Arctan Loss: $\rho(r_i)$= arctan (r_i)

Figure 18 below show how different loss functions make attenuations for outliers.

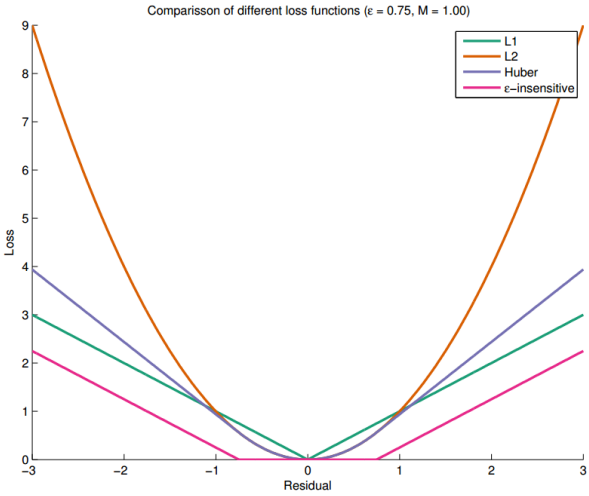


Figure 18 different loss functions effect[16]

For example, consider a physical process that has the following oscillating response over time and a certain normally distributed error ε and its required to model the process. The non-linear equation let's say is

$f(t, \beta, \theta, \omega) = \beta e^{-\theta t}\cos(\omega t)$, where t is time and $\beta, and\ \theta$ are parameters to be estimated.

Code (Python)

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import rcParams
from scipy.optimize import least_squares

rcParams['figure.figsize'] = (11, 7)
rcParams['axes.labelsize'] = 12
rcParams['legend.fontsize'] = 12
r = np.linspace(0, 6, 50) #generates data points 0-6
linear = r**2 # the linear effect
huber = r**2
huber[huber > 1] = 2 * r[huber > 1] - 1 #Huber effect
soft_l1 = 2 * (np.sqrt(1 + r**2) - 1)
cauchy = np.log1p(r**2)
arctan = np.arctan(r**2)
plt.plot(r, linear, label='linear')
plt.plot(r, huber, label='huber')
plt.plot(r, soft_l1, label='soft_l1')
plt.plot(r, cauchy, label='cauchy')
plt.plot(r, arctan, label='arctan')
plt.xlabel("$r$")
plt.ylabel(r"$\rho(r^2)$")
plt.legend(loc='upper left');
plt.show()
def dataGeneration(time, B, theta, omega, Outliers_p=0, disturbance=0,
Randomness=0):
    y = B * np.exp(-theta * time) * np.cos(omega * time)
```

```
    random = np.random.RandomState(Randomness)
    Error_e = disturbance * random.randn(time.size)
    outliers = random.randint(0, time.size, Outliers_p)
    Error_e[outliers] *= 35
    return y + Error_e
B = 2
theta = 0.1
omega = 0.1 * 2 * np.pi
x_true = np.array([B, theta, omega])
disturbance = 0.1
time_min = 0
time_max = 30
time_train = np.linspace(time_min, time_max, 30)
y_train = dataGeneration(time_train, B, theta, omega, disturbance=disturbance,
Outliers_p=4)
print(time_train.size)
def res_fun(x, time, y):
    return x[0] * np.exp(-x[1] * time) * np.cos(x[2] * time) - y
x_i = np.ones(3)

res_Least_square = least_squares(res_fun, x_i, args=(time_train, y_train))
res_robust_regression = least_squares(res_fun, x_i, loss='huber', f_scale=0.1,
args=(time_train, y_train))

time_test = np.linspace(time_min, time_max, 300)
y_test = dataGeneration(time_test, B, theta, omega)
y_Least_square = dataGeneration(time_test, *res_Least_square.x)
y_robust_regression = dataGeneration(time_test, *res_robust_regression.x)
plt.plot(time_train, y_train, 'o', label='data')
plt.plot(time_test, y_Least_square, label='Least_square')
plt.plot(time_test, y_robust_regression, label='robust_regression')
plt.plot(time_test, y_test, label='The True fit')
plt.xlabel('time(t)')
plt.ylabel('y')
plt.legend()
plt.figtext(0.7, 0.65, "Distrubance=0.1",fontsize=16)
plt.show()
```

Output

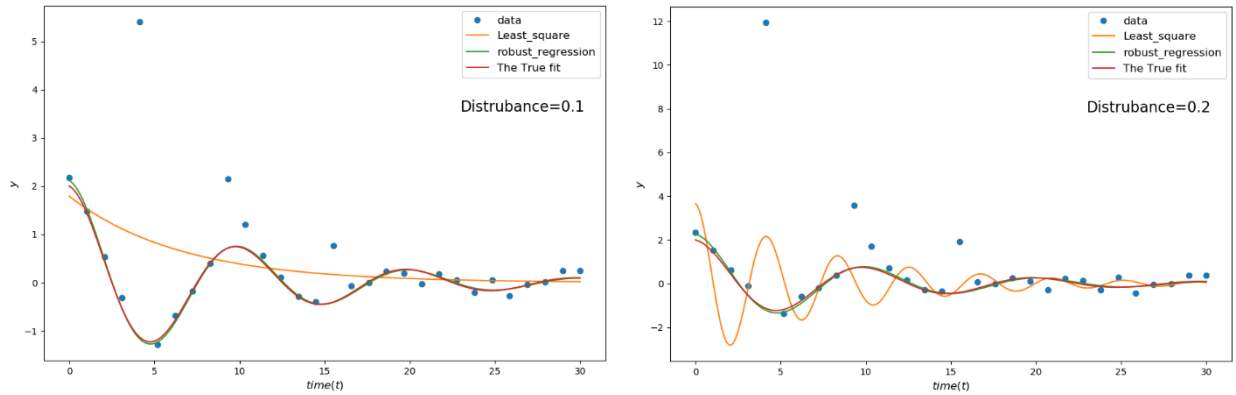


Figure 19 (a) shows output with 0.1 disturbance to data. (b) shows result with 0.2 disturbance

Figure 19 shows how with different outlier points the non-linear regression becomes misleading also note how a robust non-linear regression model compensates for the effect of existing outliers by decreasing bias.

In figure 20, Replacing the function above by the quadratic function by $f(t,\beta) = \beta t^2 + \frac{t}{2}$ and putting system disturbance to 50 with 10 outliers. The following graph illustrates the difference between robust regression, linear regression, and the true model fit.

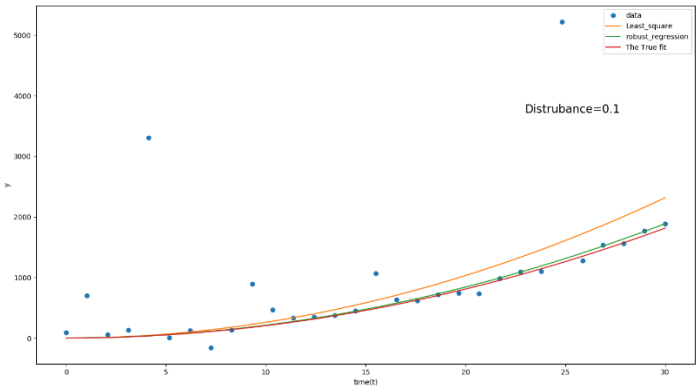


Figure 20 results for f(t,β)=βt^2+t/2

8.1.5 Logistic regression

One special type of regression, known as logistic regression, covers the case where the response variable is categorical, i.e. having a qualitative response. Therefore, a generalized linear model that is an extension of linear regression models for non-normal response variables, is used to run logistic regression. For example, if the production equipment is either in a healthy state (y = 1) or a not in a healthy state (y = 0). For such problems, the expected value of the response variable will be the conditional probability of the event. Such probabilities have an S-shaped curve as seen by figure 21 unlike the normal distribution.

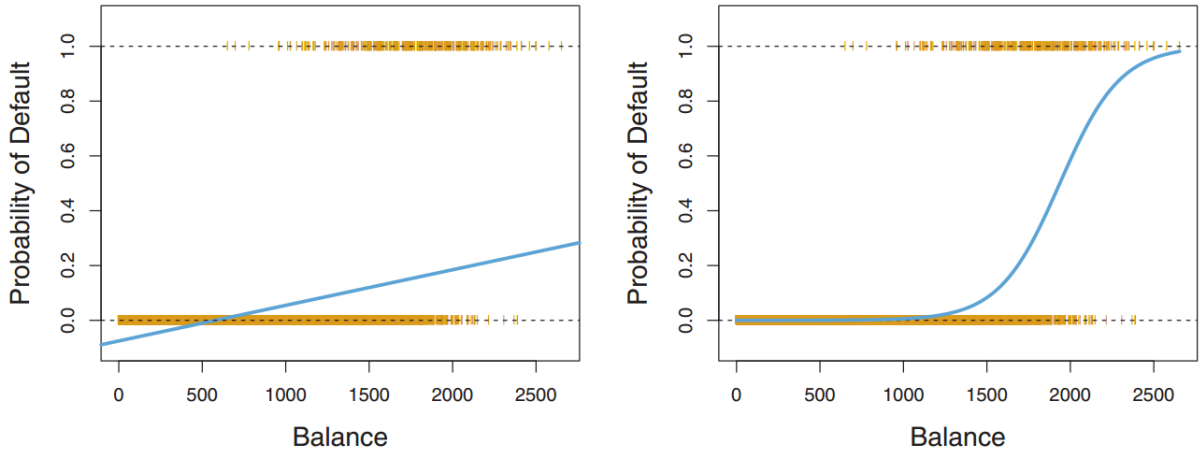


Figure 21 Sigmoid function on left [17]

The figure on right shows output of linear regression with negative probabilities for classifying data, while b shows how all probabilities lies between 0 and one and thus can be used as a classifier for distinguishing data classes.

$p(x)$ represents the probability of success for predictor variables observed values and the *odds* is the ratio of the success to failure probability represented by. $\frac{\boldsymbol{p(x)}}{\boldsymbol{1-p(x)}}$

To make it suitable for linear fitting, the logit function which is the inverse of the logistic function is defined as having a linear relationship with predictor variables such that it can vary between negative and positive infinity given probabilities between zero and one. The equation is given by:

$$\mathrm{L(x)} = \ln\frac{\boldsymbol{p(x)}}{\boldsymbol{1-p(x)}} = \beta_0 + \beta_1 x_{1i} + \cdots \beta_p x_{pi}$$

This is a semi log model that states an increase in one unit in x_j holding all else constant results in a multiplicative factor of $\boldsymbol{e^{\beta j}}$ to the odds of success.

From the equation above, predictions for $\boldsymbol{p(x)}$ is calculated to be:

$$\boldsymbol{p(x)} = \frac{\boldsymbol{e^t}}{\boldsymbol{1+e^t}}\text{, where t = } \beta_0 + \beta_1 x_{1i} + \cdots \beta_p x_{pi}$$

In logistic regression there are two ways to gather data by sampling. Either by prospective sampling where data is gathered from known categories of the predicting variables or retrospective where data is gathered given response variable value. For example, a problem where an engineer is needed to define the probability of having a cutting tool failure given

rotational speed of material (high or low). In prospective sampling, data is sampled out of processes with either high or low rotational speeds and therefore such sampling can have a better presentation of the actual process yet if failure percentage is low it will not be able to accurately model the failure probability. On the other hand, the retrospective sampling approach will gather data from known failure and non-failure processes.

The different approaches result in different conditional probabilities and inputs are related to outputs if the odds of process success given a certain predictor category is different from the odds of process success given another category for the predictor variable.

For the prospective case, this is given by:

$\frac{\boldsymbol{p}[\boldsymbol{s}|\boldsymbol{l}]}{\boldsymbol{p}[\boldsymbol{f}|\boldsymbol{l}]} \neq \frac{\boldsymbol{p}[\boldsymbol{s}|\boldsymbol{h}]}{\boldsymbol{p}[\boldsymbol{f}|\boldsymbol{h}]}$ where **s** represents a successful event, **f** represents a non-successful event, **l** in our example represents low rotational speed and **h** represents a high rotational speed. $\boldsymbol{p}[\boldsymbol{s}|\boldsymbol{l}]$ for instance, is the conditional probability of having a successful event given low rotational speed in our example.

For the retrospective case, it is given by: $\frac{\boldsymbol{p}[\boldsymbol{l}|\boldsymbol{s}]}{\boldsymbol{p}[\boldsymbol{h}|\boldsymbol{s}]} \neq \frac{\boldsymbol{p}[\boldsymbol{l}|\boldsymbol{f}]}{\boldsymbol{p}[\boldsymbol{h}|\boldsymbol{f}]}$

The estimates for $\boldsymbol{\beta}$ in logistic regression are found by using the maximum likelihood estimation that maximizes the log likelihood function given by

$L(\beta_0, \beta_1, , \beta_p) = \sum_{i=1}^{n} [y_i \ln p_i + (n_i - y_i) \ln(1 - p_i)$

The deviance which is the given by $-2\ln\frac{Likelihood\ of\ the\ fitted\ model}{likelihood\ of\ the\ saturated\ model}$ is compared to x^2 distribution and used to show how good the fit is_ smaller values indicate a better fit.

This deviance is also used to determine if a certain predictor or set of predictors improved the model (known as the likelihood ratio test **LR**). The LR is evaluated by subtracting the null deviance from the model deviance and evaluating the results on x^2 distribution with p degrees of freedom. If the model deviance is greatly smaller than the null deviance, it means the added predictor(s) has improved the model, i.e. refusing the null hypothesis

$$H_0: \beta_i = 0 \ for \ i \ \in 1, \dots, p$$

And accepting the alternative hypothesis:

$$H_a: \beta_i \neq 0 \; for \; atleast \; one \; value \; of \; i$$

Just as the t-test discussed earlier in the linear regression, a similar test exists for logistic regression. The Wald test statistic is used to test the significance of coefficients and is given by,

$$W_j = \frac{B_j^2}{SE_{B_j}^2}.$$

AIC is used for deciding on model selection. The equation is given by

$$AIC = -2L + 2(p+1)$$

Finally, the model should be tested for the goodness to check if the logistic regression model fits the data well (null hypotheses) or not (alternative hypothesis). A common test to measure the goodness of fit is known as Deviance Statistic test

$$G^2 = 2\sum_{i=1}^{n}[y_i \log\frac{y_i}{n_i p_i} + (n_i - y_i)\log(\frac{n_i - y_i}{n_i(1-p_i)})]$$

Or the Pearson goodness-of-fit statistic

$$X^2 = \sum_{i=1}^{n}\frac{(y_i - n_i p_i)^2}{n_i p_i(1-p_i)}$$

The result is checked with the x^2 distribution given n-p-1 degrees of freedom.

For example, suppose from the example before, cutting tools are examined after 10 weeks of observation to check whether they are healthy or not healthy. During observation, two predictors were recorded for analysis, predictor 1 is the age of the tools and predictor 2 is whether the tool is used for high speed cutting or low speed cutting. The following table shows the results

Age category (In weeks)	High speed	Healthy	Tools
16	1	48	50
16	0	56	57
24	1	116	119
24	0	147	152
36	1	90	104
36	0	109	116
44	1	98	125
44	0	61	73
54	1	59	110
54	0	76	116
59	1	42	114
59	0	51	112
64	1	2	31
64	0	23	124
76	1	0	8
76	0	0	59

Initially, the data might be misleadingly showing that after 10 weeks 68.8% of the tools used in high cutting speed where healthy while only 64.6% of the tools used in low cutting speed is healthy. However, this should be studied because of presence of other variable age category which might be correlated with the cutting speeds.

Code (R)

```
HealthyTool=read.csv("HealthyTool.csv")#reading file
PercentageH= HealthyTool$Healthy/ HealthyTool$Tools #visualize data
relation
logit = log(PercentageH/(1- PercentageH))# creating logit function
par(mfrow=c(1,2))# to allow us putting multiple graphs in one plot on
separate columns and same raw

plot(HealthyTool$Age.category[HealthyTool$High.speed==0],
PercentageH[HealthyTool$High.speed==0], pch=2, xlab="Age category",
  ylab="Proportion Healthy", main="(a)")#create first graph with points
corresponding to 1 high speed

points(HealthyTool$Age.category[HealthyTool$High.speed==1],
PercentageH[HealthyTool$High.speed==1], pch=3) #append graph of high speed
0 to same plot

legend(20, .09, c("High cutting speed", "Slow cutting speed "), pch=c(3,
2), cex=0.8) #legend to label points

plot(HealthyTool$Age.category[HealthyTool$High.speed==0],
logit[HealthyTool$High.speed==0],pch=2, xlab="Age category",
  ylab="logit", main="(b)") #create the logit graph
points(HealthyTool$Age.category[HealthyTool$High.speed==1],
logit[HealthyTool$High.speed==1], pch=3)
legend(20, -1, c("High cutting speed", "Slow cutting speed"), pch=c(3, 2),
cex=0.8)
```

Output

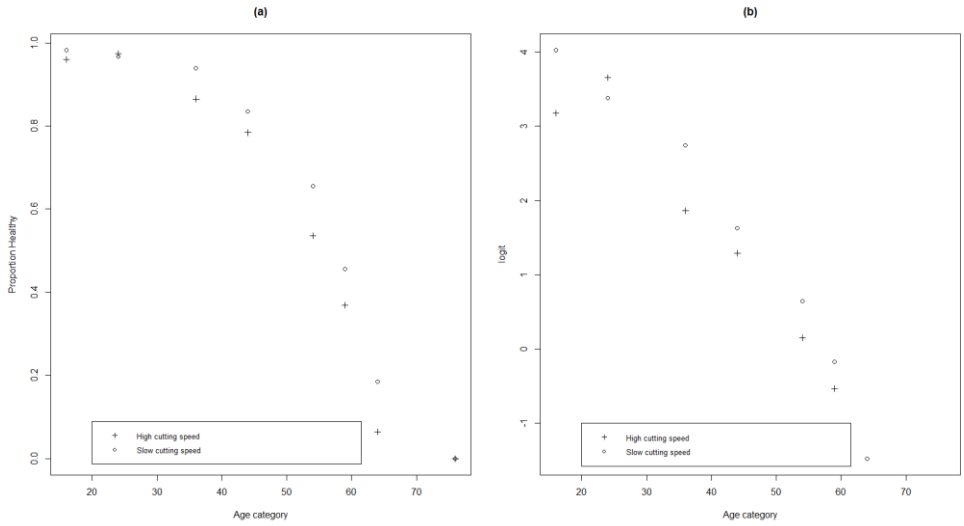


Figure 22 Logistic regression example

Figure 22 on the left shows the healthy proportion of tools after 10 weeks given the age of the tool during observation. The left figure shows the logit function which is the logarithm of the odds.

By plotting these graphs, it clearly states that the given age of tool, a tool being in healthy state is higher for low cutting speeds than for high cutting speeds.

Code R #model 1

```
HealthyTool1.logit = glm(Healthy/Tools ~ Age.category + High.speed,
weights=Tools, family=binomial, data= HealthyTool)#uses generalized linear
model for logistic regression
summary(HealthyTool1.logit)#Get results for regression analysis B,T,P
LR = HealthyTool1.logit$null.deviance - deviance(HealthyTool1.logit)

c(LR, 1-pchisq(LR,2))# Likelihood ratio and Significance Value

c(deviance(HealthyTool1.logit), 1-pchisq(deviance(HealthyTool1.logit),11))#
deviance statistics test and p value

pearson_residual1 = residuals(HealthyTool1.logit, type="pearson")
pearson_test = sum(pearson_residual1^2)
c(pearson_test, 1-pchisq(pearson_test,11))#Pearson test result and P value
```

Output

```
Call:
glm(formula = Healthy/Tools ~ Age.category + High.speed, family = binomial,
    data = HealthyTool, weights = Tools)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9825  -1.5276  -0.5745   0.5404   1.9432

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   7.180103   0.408053   17.60   <2e-16 ***
Age.category -0.127991   0.007163  -17.87   <2e-16 ***
High.speed   -0.323258   0.146274   -2.21   0.0271 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 704.966  on 15  degrees of freedom
Residual deviance:  34.067  on 13  degrees of freedom
AIC: 96.051

Number of Fisher Scoring iterations: 5

> LR = HealthyTool1.logit$null.deviance - deviance(HealthyTool1.logit)
> c(LR, 1-pchisq(LR,2))# LR and Significance Value
[1] 670.8994   0.0000
> c(deviance(HealthyTool1.logit), 1-
pchisq(deviance(HealthyTool1.logit),11))
[1] 3.406680e+01 3.527047e-04
> pearson_residual1 = residuals(HealthyTool1.logit, type="pearson")
> pearson_test = sum(pearson_residual1^2)
> c(pearson_test, 1-pchisq(pearson_test,11))
[1] 3.221144e+01 7.053555e-04
```

The model here shows that the overall regression is statistically significant. The LR which is the difference between the Null and Residual defiance is found to be 670.8994 with 2 degrees of freedom, corresponding to a significance value p ≈ 0. The Wald test which is given by the z value shows that the age category is highly significant while the cutting speed less significant.

The exponential of the slopes ($\beta 1, \beta 2$) gives the odd ratio and is given by 0.8798 and 0.7237

respectively. What this means is that, given a certain cutting speed category, an increase in 1 week in equipment age will result in 12.02% smaller odds of the tool to be healthy after 10 weeks and given equipment age, if the equipment is used in high cutting speeds it will have a 27.63% smaller odds to be healthy after 10 weeks.

However, the $G^2$and X^2 test shows the there is no goodness of fit with the selected model (34.06 with p<0.0031 for G^2 and 32.2 with P<0.0007 for X^2. This is obvious if we look at the relationship between tool age and proportional value of being healthy in the left graph of figure 22

In the following codes a quadratic relation and a categorical relation is tested and the results will be discussed below the final output.

Code model 2 quadratic

```
Age.category.sq = HealthyTool$Age.category^2 # squared assuming quadratic r
HealthyTool2.logit = glm(Healthy/Tools ~ Age.category + Age.category.sq +
High.speed, weights=Tools, family=binomial,
  data= HealthyTool) #semi log model, linear transformation case

summary(HealthyTool2.logit)

c(deviance(HealthyTool2.logit), 1-pchisq(deviance(HealthyTool2.logit),10))
pearson_residual2 = residuals(HealthyTool2.logit, type="pearson")

pearson_test = sum(pearson_residual2^2)
c(pearson_test, 1-pchisq(pearson_test,10))
c(deviance(HealthyTool1.logit)-deviance(HealthyTool2.logit),
  1-pchisq(deviance(HealthyTool1.logit)-deviance(HealthyTool2.logit),1))
```

Output

```
Call:
Call:
glm(formula = Healthy/Tools ~ Age.category + Age.category.sq +
    High.speed, family = binomial, data = HealthyTool, weights = Tools)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.49845  -0.87152  -0.05126   0.49672   1.24947

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      3.1123732  0.8558659   3.637 0.000276 ***
Age.category     0.0633009  0.0398825   1.587 0.112471
Age.category.sq -0.0020609  0.0004439  -4.643 3.44e-06 ***
High.speed      -0.4453333  0.1509959  -2.949 0.003185 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 704.966  on 15  degrees of freedom
Residual deviance:  12.982  on 12  degrees of freedom
```

```
AIC: 76.966

Number of Fisher Scoring iterations: 4

> c(deviance(HealthyTool2.logit), 1-
pchisq(deviance(HealthyTool2.logit),10))
[1] 12.9820091  0.2246794
> pearson_residual2 = residuals(HealthyTool2.logit, type="pearson")
> c(pearson_test, 1-pchisq(pearson_test,10))
[1] 11.4179514  0.3258956
> c(deviance(HealthyTool1.logit)-deviance(HealthyTool2.logit),
+   1-pchisq(deviance(HealthyTool1.logit)-deviance(HealthyTool2.logit),1))
[1] 2.108479e+01 4.394021e-06
```

Code model 3 categorical

```
HealthyTool3.logit = glm(Healthy/Tools ~ factor(Age.category) + High.speed,
weights=Tools, family=binomial,
  data= HealthyTool) #assigning age data as categorical

summary(HealthyTool3.logit)

c(deviance(HealthyTool3.logit), 1-pchisq(deviance(HealthyTool3.logit),6))

pearson_residual3 = residuals(HealthyTool3.logit, type="pearson")
pearson_test = sum(pearson_residual3^2)
c(pearson_test, 1-pchisq(pearson_test,6))

c(deviance(HealthyTool1.logit)-deviance(HealthyTool3.logit),
  1-pchisq(deviance(HealthyTool1.logit)-deviance(HealthyTool3.logit),5))

c(deviance(HealthyTool2.logit)-deviance(HealthyTool3.logit),
  1-pchisq(deviance(HealthyTool2.logit)-deviance(HealthyTool3.logit),4))
```

Output

```
Call:
glm(formula = Healthy/Tools ~ factor(Age.category) + High.speed,
    family = binomial, data = HealthyTool, weights = Tools)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.95108  -0.29699  -0.00008   0.25445   0.73490

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)               3.79461    0.59262   6.403 1.52e-10 ***
factor(Age.category)24   -0.06662    0.68739  -0.097 0.922791
factor(Age.category)36   -1.29820    0.62958  -2.062 0.039208 *
factor(Age.category)44   -2.07317    0.61317  -3.381 0.000722 ***
factor(Age.category)54   -3.16326    0.60186  -5.256 1.47e-07 ***
factor(Age.category)59   -3.91748    0.60184  -6.509 7.55e-11 ***
factor(Age.category)64   -5.36002    0.62895  -8.522  < 2e-16 ***
factor(Age.category)76  -26.50630 6480.95408  -0.004 0.996737
High.speed               -0.47556    0.15183  -3.132 0.001735 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 704.9662  on 15  degrees of freedom
Residual deviance:   3.2506  on  7  degrees of freedom
AIC: 77.235
```

```
Number of Fisher Scoring iterations: 19

> c(deviance(HealthyTool3.logit), 1-pchisq(deviance(HealthyTool3.logit),6))
[1] 3.2505949 0.7768008
> pearson_residual3 = residuals(HealthyTool3.logit, type="pearson")
> pearson_test = sum(pearson_residual3^2)
> c(pearson_test, 1-pchisq(pearson_test,6))
[1] 3.1119907 0.7946653
> c(deviance(HealthyTool1.logit)-deviance(HealthyTool3.logit),
+   1-pchisq(deviance(HealthyTool1.logit)-deviance(HealthyTool3.logit),5))
[1] 3.081620e+01 1.018336e-05
> c(deviance(HealthyTool2.logit)-deviance(HealthyTool3.logit),
+   1-pchisq(deviance(HealthyTool2.logit)-deviance(HealthyTool3.logit),4))
[1] 9.73141418 0.04520335
```

The results show that for a quadratic model and categorical model the G^2 *and* X^2 are much lower and thus have a higher P value showing that the model relation represents a better fit for the data. For the Quadratic model, G^2(and the P value,) X^2(and the P value)*are given as* 12.9820091 (0.2246794) and 11.4179514 (0.3258956) respectively and for Categorical model they are given by 3.2505949(0.7768008) and 3.1119907(0.7946653) respectively. Also note that values of the LR between quadratic and linear is 21.08479 corresponding to a P value <0.00001 showing that the model is improved. Same goes for the LR of categorical versus Linear with a value of 30.816 and a corresponding p value<0.0001. The AIC for the quadratic and categorical are also lower ,76.966 and 77.235 respectively < 96.051, showing a better model selection.

The choice between quadratic and categorical model however is not easily defined since LR has a value of 9.731 with 4 degrees of freedom corresponding to a p value=0.0452. However, in both models the Speed of cutting has a value roughly equal to -0.475, taking the exponential of it tells that given tool age a tool used in high cutting speeds has 37.81% lower odds to be in a healthy state.

Such information might be important for a company to set activities and design processes.

An extension to logistic regression is the multinomial regression where the response variables can take more than binary categories. However, other popular techniques are used for such problems that will be discussed later.

8.1.6 Kernel density regression for non-parametric regression. (pattern analysis)

There exist non-parametric regression techniques in which the shape of the function is not assumed when running regression analysis. The Kernel regression is a common tool used for this purpose. The concept behind it is that it applies some weights and a smoothing function to make the density estimation in a continuous form.

Density estimation

In density estimation, the density function is estimated from data without making functional assumptions

The density function is calculated as the derivative of the cumulative distribution function $F_x(x)$:

$$f_x(x_0) = \frac{d}{dx} F_x(x_0)$$

This equation can then be written as the probability that a given observation x is within a given interval

$$f_x(x_0) = \lim_{h \to 0} \frac{Prob[x_0 - h < x < x_0 + h]}{2h}$$

Histograms are used as a discreet approximation for the density function as it counts the number of observations that fall in a certain interval/bin-width. Note that different interval length yields different sensitivity results for the density function.

The height *H(x)* of the histogram on a kth interval is given by the following equation [18]

$$H(x) = \frac{1}{Nh} \sum_{m=0}^{N-1} I_{\{|x - x_m| \leq \frac{h}{2}\}}$$

N is the number of measurements, and indicator function $I = \begin{cases} 1 & if\ x\ is\ found\ in\ interval \\ 0 & otherwise \end{cases}$

The kernel function extends the density estimation in order to get an estimated continuous density function rather than discreet. This can be done by placing unequal weights on the values in intervals and thus replacing *I* in equation above with a smooth function.

The kernel density estimator is given by:

$$f(x_0) = (\frac{I}{Nh}) \sum_i K \frac{x_i - x_0}{h}$$

Where K has the following properties:

$$\begin{cases} K(t) & \geq & 0 \\ K(-t) & = & K(t) \\ \int_{-\infty}^{\infty} K(t)dt & = & 1. \end{cases}$$

These properties ensure that the weight decrease as we move away from the reference point, i.e. the bin width. It also ensures symmetrical effect in both directions. Finally, the total sum of the weights equal 1 represents a weighted average that integrates to 1. [18] There are two method where K(t) equals zero.

1) Absolute cutoff on some point t_0. $K(t) = 0$ when $t \geq t_0$.

2) Smooth cutoff. $K(t) \rightarrow 0 \; as \; |t| \rightarrow \infty$.

There exist common kernel functions for K that satisfy the criteria. The plot below shows different outputs with different kernel functions

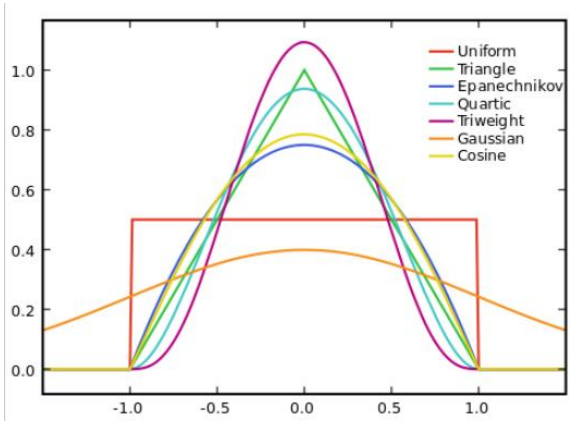


Figure 23 different outputs of Kernel functions

To find the optimal estimated density function, the tradeoff between bias and variance is important as discussed earlier to avoid under smoothed or oversmoothed models as seen by figure [18] Kernel function choice need to be considered and the optimal value of the bin width need to be calculated. One technique to get a good value of the bin width is by minimizing the mean squared error (MSE) of the kernel density function since MSE is a function of the bin width. reference for kernel]called kernel]. Whereas for the kernel function

choice, a kernel function resulting in a good tradeoff between bias and variance is given by the Epanechnikov Kernel equation when the noise is under a desired level or the Gaussian Kernel equation when the noise is greater than desired. An illustration of this is shown in figure 24put [18]

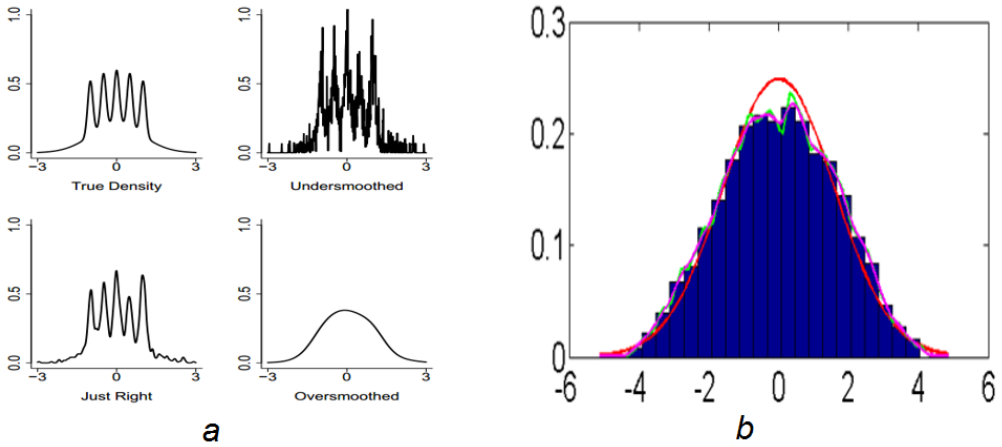


Figure 24 Kernel regression

The figure shows under smoothed estimation (h value is smallest with respect to the rest), good estimation (good h value), and oversmoothed estimation (h has largest value with respect to rest] figure b plot shows density functions estimations, Using a bin width according to the minimum MSE technique, true pdf(red), Epanechnikov Kernel (green), and Gaussian Kernel (pink).

This is extended into the bivariate and multivariate context where in the univariate the interest in in the overall density function while in the, bivariate and multivariate, the interest is in modeling the conditional mean

$$E(y_i|x_i) = f(x_i), \qquad where\ f(x_i)\ is\ unspecified\ and\ y_i = f(x_i) + \epsilon.$$

If the values for *x* are discreet, $E(y|x_0)$ is the kernel weighted average of y_i observed at x_0. if the *x* is continuous, we find the kernel weighted average of y_i for *x* close to x_0.

The above steps are calculated moving along $\boldsymbol{X}$ axis.

So, the regression works by calculating how many x_i are there such $x_i - x_o$ is within the bin width range, and then for those cases, the kernel weighted average of $y_i$values is calculated.

In the following model, we use data generated from the nonlinear regression example to try to estimate response variable y given only data points without assuming functional form.

Code

```
from math import exp,sqrt,pi
import numpy as np
import matplotlib.pyplot as plt

def dataGeneration(time, B, theta, omega, Outliers_p=0, disturbance=0,
Randomness=0):
    y = B * np.exp(-theta * time) * np.cos(omega * time)
    random = np.random.RandomState(Randomness)
    Error_e = disturbance * random.randn(time.size)
    outliers = random.randint(0, time.size, Outliers_p)
    Error_e[outliers] *= 35
    return y + Error_e

B = 2
theta = 0.1
omega = 0.1 * 2 * np.pi
x_true = np.array([B, theta, omega])
disturbance = 0.1
time_min = 0
time_max = 30
time_train = np.linspace(time_min, time_max, 100)
y_train = dataGeneration(time_train, B, theta, omega, disturbance=disturbance,
Outliers_p=4)


def fit(test_X, train_X, train_y, bandwidth=1.0, kn='box'):
    kernels = {
        'gs': lambda x: 1/sqrt(2*pi)*exp(-x**2/2),
        'ep': lambda x: 3/4*(1-x**2) if (x<=1 and x>=-1) else 0
    }
    PredictorY = []
    for entry in test_X:
        nks = [np.sum((j-entry)**2)/bandwidth for j in train_X]
        k = [kernels['gs'](i) for i in nks]
        Numerator = sum([k[i]*train_y[i] for i in range(len(k))])
        Denom_devide = sum(k)
        predict = Numerator/Denom_devide
        PredictorY.extend(predict)
        # print(entry)
    return np.array(PredictorY)[:,np.newaxis]

my_dpi=80
plt.figure(figsize=(600/my_dpi, 600/my_dpi), dpi=my_dpi)

plt.style.use('seaborn-whitegrid')

a = time_train
train_a = a[:,np.newaxis]
b=y_train[:,np.newaxis]
test_a = np.linspace(1,30,100)
formed_a = test_a[:,np.newaxis]
pred_b = fit(formed_a,train_a,b,0.3,'gs')
plt.scatter(train_a,b,color='black',label=" Original Points")
plt.plot(test_a,pred_b,color='blue',linewidth=3,label="Estimated response")
Denom_devide = lambda x: np.sum([k(np.sum((j-x)**2)/bindwidth) for j in train_X])
plt.xlabel('time(t)')
plt.title("Kernel Regression")
plt.ylabel('y')
plt.legend()
```

```
plt.figtext(0.15, 0.8, "Non-Parametric estimation of nonlinear model",fontsize=15)
plt.show()
```

Output

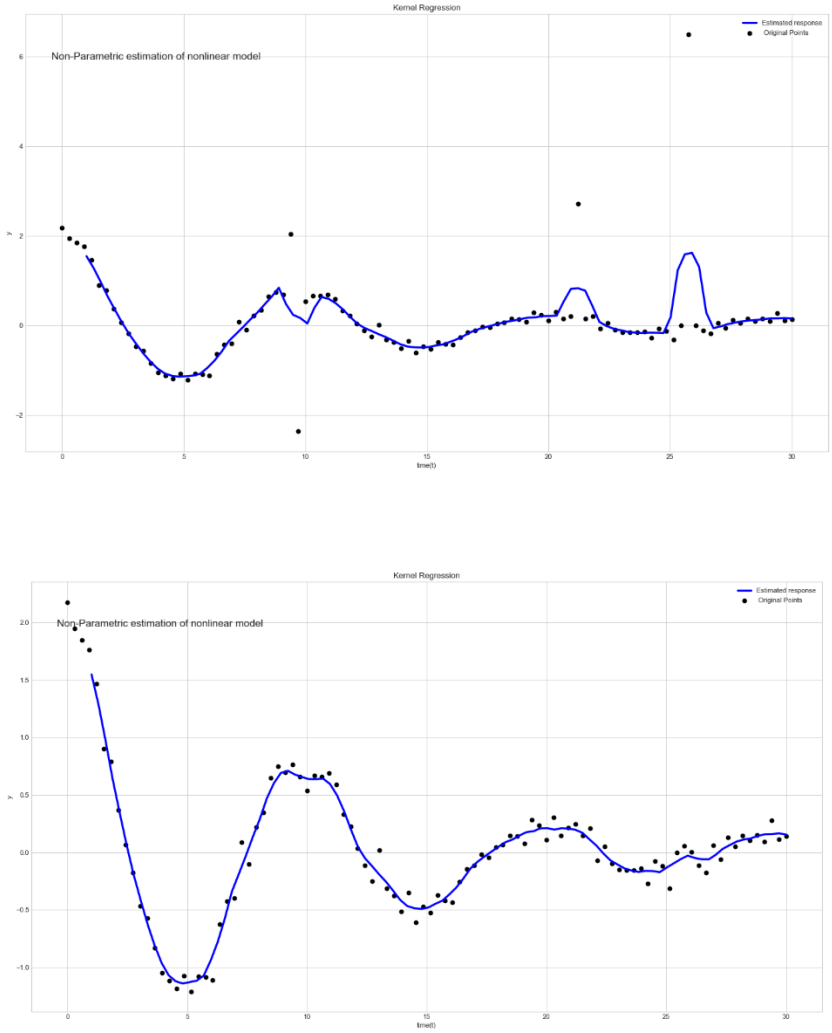


Figure 25 Results of Kernel regression model example

The results above show how the system was able to estimate the behavior of the function without prior knowledge of the functional form. Plot (a) show how outliers can affect the system, plot (b) shows how the model performs when we remove the outliers before system modeling.

This approach can be helpful in industrial uses where the mathematical relation between inputs and the response variable is not known. By estimating system behavior, a person can

infer process features and/or process distribution function in order to have a better understanding on the system and on how it will behave on certain set of input values.

8.2 Classification and Clustering techniques

To classify data the typical approach is to understand the problem addressed and identify some features and check whether data are labeled or not to decide on what algorithm to choose.

There are two phases in classification, train and testing, as is the case with other machine learning algorithms. After that data is split into training and testing, the model is developed then tested for accuracy to confirm its deployment. As will be seen in examples and model to come, there are different techniques to search for best split of data and best parameters of models, for instance the grid search method or cross validation technique and K-fold.

8.2.1 Bayesian Classifier

The Bayesian Classifier assigns to each observation the most likely class, given its predictor values. Thus, given some Data X=x0, we compute the probability that it belongs to a certain class Y=j and assign x0 to the class that has the highest probability. This method has the smallest test error rate that limits the accuracy, however, in reality, the conditional distribution of Y is unknown and therefore we should create models that estimates the probability distribution.

8.2.2 K-nn classifier

One way of doing this is by the K-Nearest Neighbors method which tries to estimate the Bayesian ideal.

P(Y=j|X=x0) = $\frac{1}{K}\sum_{i\varepsilon \mathbf{N0}} I(yi = j)$, K is the number of nearest points chosen to X0 and are represented by N_0.

This simply means that according to a specified number of K, the K-Nearest neighbors counts the nearest K points (based on either the Euclidean, Manhattan, Hamming or other distance finding algorithms) and checks what their classes are and provides the probability of belonging to different classes. To have good results, the value of K should neither be small to

avoid having high variation nor big to avoid being highly biased. Also, splitting of data to train and test is optimized by cross validation technique such as the K-fold cross validation which splits data into k groups and with each iteration it takes one of the k groups as the test data while others as training data to test for classifier accuracy value, it keeps on iterating until all k groups of data are tested for results and the best choice is hence chosen.

However, for industrial use, their exists drawbacks for using KNN algorithm because of it being computationally expensive when testing new data. In addition to that, KNN algorithm is subject to loses accuracy when a certain class dominates the data set. However, depending on problem addressed some measures can be implemented to account for these drawbacks. For instance, applying a weight value on data classes that is inversely proportional to the distance between the test point and class points, or by normalizing data points to rescale data values. Another important technique that is used to make algorithms more efficient is the Principal Component Analysis (PCA) that reduces the data dimension into the main principal componentnts or data features that can make the algorithm increase accuracy and efficiency. [19] shows how the principal components from a data set are derived to form a new data set. For python modeling, we will use library from sklearn.decomposition import PCA and call the function PCA(n_components=N).fit_transform(data) where N is the number of features desired and data is the original set of data present This is shown by the example below. Data assumed for this example is weather healthy or not the system is based on age, width, strain factor, vibrations, temperature.

The first model takes all the data gathered and searches for the optimal number of K-nearest neighbors.

Code

```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
#Access file
df = pd.read_csv('KnnClassifier2Data.csv')
#Drop the ouput data from the input values and chose the target value for Y
X = df.drop(columns=["Outcome"])

Y = df["Outcome"].values

#split data to training and testing

from sklearn.model_selection import GridSearchCV
#create Knn filter and search for appropriate K value
knn = KNeighborsClassifier()
param_grid = {"n_neighbors": np.arange(1, 20)}
knn_Optimized = GridSearchCV(knn, param_grid, cv=4)
```

```
#Training Knn
knn_Optimized.fit(X, Y)
#define K and Measure performance
print("Optimal value for K",knn_Optimized.best_params_)
print("Accuracy is %",100*knn_Optimized.best_score_)
#test
from sklearn.model_selection import train_test_split
X_train_data, X_test_data, Y_train_data, Y_test_data = train_test_split(X, Y,
test_size=0.2, random_state=1, stratify=Y)
print("predicted values \n",knn_Optimized.predict(X_test_data)[12:20], "\n compare
it to test data \n", Y_test_data[12:20])
```

Output

The following models outputs an optimal value for K {'n_neighbors': 3} with an accurace of % 88.3636

Some Predicted values in the Test phase

Actual	Predicted
Not Healthy	Not Healthy
Not Healthy	Healthy
Not Healthy	Not Healthy
Not Healthy	Not Healthy
Not Healthy	Not Healthy
Healthy	Healthy
Healthy	Healthy
Healthy	Healthy

In the following model it shows how the principal components analysis improves both accuracy and efficiency. The model plots the first three principal components in 3d for visualizing how different data classes tend to gather in clusters and finally, for data visualization, the figure 26 shows the zones where Knn classify test points to either class (differentiated by different colors)

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

```
from matplotlib.colors import ListedColormap
from sklearn.decomposition import PCA
import numpy as np
from sklearn import neighbors
import xlrd
loc=r'C:\Users\Computer Clinic\PycharmProjects\Haitham\KnnClassifier3Data.xlsx'
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)
col1=[]
col2=[]
col3=[]
col4=[]
col5=[]
target=[]
for i in range(1,sheet.nrows):
    col1.append(sheet.cell_value(i, 0))
    col2.append(sheet.cell_value(i, 1))
    col3.append(sheet.cell_value(i, 2))
    col4.append(sheet.cell_value(i, 3))
    col5.append(sheet.cell_value(i, 4))
    target.append(sheet.cell_value(i, 5))
    #print(sheet.cell_value(i, 0))
X=np.c_[col1,col5]
y=target
dataAll=np.c_[col1,col2,col3,col4,col5]
cmap_background= ListedColormap(['#A7FFA4', '#5F8FF4']) # for coloring zone
cmap_points = ListedColormap(['green','blue']) # for coloring points
#Plotting 3 dimensional plot for principal components
fig = plt.figure("Principal Components scatter plot", figsize=(10, 7))
ax = Axes3D(fig, elev=-140, azim=100)
X_PC = PCA(n_components=3).fit_transform(dataAll)
ax.scatter(X_PC[:, 0], X_PC[:, 1], X_PC[:, 2], c=y,cmap= cmap_points,
edgecolor='k', s=40)
ax.set_title("Principal Components scatter plot")
ax.set_xlabel("principal comp1 vector")
ax.w_xaxis.set_ticklabels([])
ax.set_ylabel("principal comp2 vector")
ax.w_yaxis.set_ticklabels([])
ax.set_zlabel("principal comp3 vector")
ax.w_zaxis.set_ticklabels([])
plt.show()
# train Knn filter for 2 dimensional points and check accuracy
knn = neighbors.KNeighborsClassifier(n_neighbors=3)
knn.fit(X, y)
x_min, x_max = X[:, 0].min() - .1, X[:, 0].max() + .1
y_min, y_max = X[:, 1].min() - .1, X[:, 1].max() + .1
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100), np.linspace(y_min, y_max,
150))
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape) #need to fit shape for plt.pcolormesh

# Plots
plt.figure("Knn Decision zones and scatter plot", figsize=(10, 7))
plt.clf()
plt.pcolormesh(xx, yy, Z, cmap=cmap_background)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_points, edgecolor='k')
plt.title("Knn Decision zones and scatter plot")
plt.xlabel('age')
plt.ylabel('Temp')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
from sklearn.model_selection import train_test_split
X_train_data, X_test_data, Y_train_data, Y_test_data = train_test_split(X, y,
test_size=0.2, random_state=1, stratify=y) #gets 20 percent for test data.
```

```
print(knn.score(X_test_data,Y_test_data))


plt.show()
```

Ouput

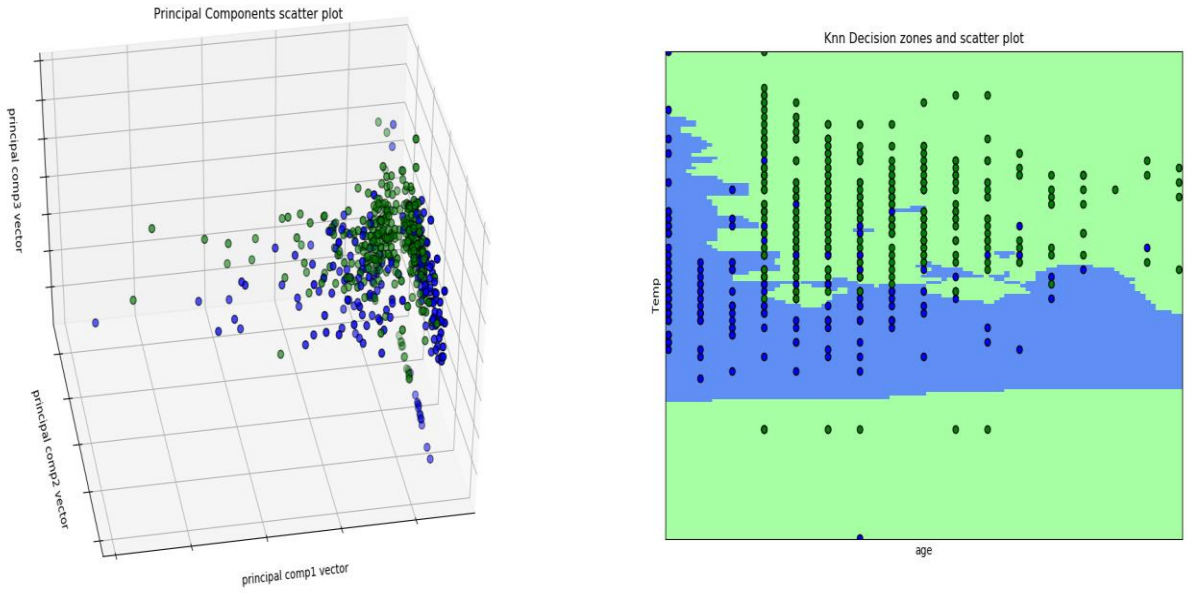


Figure 26 Knn model Example

Algorithm Accuracy = %93.63636363636364

8.2.3 Naïve Bayes Classifier

Another classification algorithm based on the Bayes' algorithm is known as the Naïve Bayes classifier.

In contrast to the discriminative approach of logistic regression in which a sigmoid function estimates the probability of outcome *y* belonging to class *k* given predictor variables **x**, the Naïve Bayes classifier is a generative model meaning it captures probability of predictors for each class as it is based on the Bayesian theorem.

Although its performance is limited by strictly assuming independent variables and assuming to have all classes needed are present in the training data set, the Naïve Bayes classifier has several advantages as it requires little amount of training data to estimate the mean and variances of the variables. Therefore, they have fast speeds during training and testing phase and can be interpreted easily as they provide direct probabilistic prediction with few tunable parameters if needed. This has its application in maintenance problems where real time

prediction is made possible by such technique. However, data should be carefully examined for correlation between them.

The equation is given by

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

Where C_k is the class and **x** is vector $x_1,\ldots, x_n$ representing feature variables.

$p(C_k) = prior\ probability, i.e. the\ probability\ of\ C_k\ being\ true\ regardless$ of data
$p(\boldsymbol{x}) = evidence, i.e. the\ probability\ of\ data\ being\ true\ regardless$ of C_k
$p(C_k|\boldsymbol{x}) = posterior, i.e. the\ probability\ of\ C_k\ being\ true\ given\ data$
$p(\boldsymbol{x}|C_k) = likelihood, i.e. the\ probability\ of\ data\ given\ being\ true\ given\ C_k$

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

By using chain rule [20]

$$p(C_k, x_1, , , , x_n) = p(x_1|x_2, , , , x_n, C_k)p(x_2|x_3, , , , x_n, C_k) \ldots p(x_{n-1}|x_n, C_k)p(x_n|C_k)p(C_k)$$

Since the Naïve Bayes classifier assume independence between variables, the chain above equation reduces to

$$p(C_k, x_1, , , , x_n) = p(x_1|C_k)p(x_2|C_k) \ldots p(x_{n-1}|C_k)p(x_n|C_k)p(C_k)$$

And $p(C_k|\boldsymbol{x})$ is proportional to $p(C_k, x_1, , , , x_n)$

$$p(C_k|\boldsymbol{x}) \propto p(C_k, x_1, , , , x_n)$$

Therefore, the conditional distribution of class C_k given **x** is given by

$$p(C_k|\boldsymbol{x}) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i|C_k), \qquad where\ Z\ is\ a\ scaling\ factor$$

Finally, the Naïve Bayes classifier takes decisions for classifying variable x_i by picking the class that maximizes the equation above.

The model for Naïve Bayes classifier depends on the assumptions made on the distributions of the feature. i.e. whether Gaussian, Bernoulli, or Multinomial distributions.

If the feature distribution is Gaussian, data is segmented by the class and the mean and variance is calculated for the features in each class. After that, the conditional probability density of the feature given class C is given by

$$P(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

If the feature distribution has a Bernoulli distribution, the features are binary variables representing presence/occurrence of a feature or its absence. The likelihood of the data given class C is given by

$$p(F_1, \dots, F_n|C) = \prod_{i=1}^{n} [F_i p(w_i|C) + (1 - F_i)(1 - p(w_i|C))]$$

Finally, if the features embody the rate of occurrence with certain events, a multinomial Naïve Bayes is used and the likelihood equation is given by:

$$p(F|C) = \frac{(\sum_i F_i)!}{\prod_i F_i!} \prod_i p_i^{F_i}$$, where p_i is the probability that event i occurs

To illustrate how Naïve Bayes classifier is used in industries, Gaussian Naïve Bayes is tested on the data from the example before.

Code

```
from sklearn.decomposition import PCA
import numpy as np
from sklearn import metrics
import xlrd
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from matplotlib.colors import ListedColormap
import matplotlib.pyplot as plt
```

```
cmap_points = ListedColormap(['green','blue',"red"])# for coloring points
loc=r'C:\Users\Computer Clinic\PycharmProjects\Haitham\KnnClassifier3Data.xlsx'
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)
col1=[]
col2=[]
col3=[]
col4=[]
col5=[]
target=[]
for i in range(1,sheet.nrows):
    col1.append(sheet.cell_value(i, 0))
    col2.append(sheet.cell_value(i, 1))
    col3.append(sheet.cell_value(i, 2))
    col4.append(sheet.cell_value(i, 3))
    col5.append(sheet.cell_value(i, 4))
    target.append(sheet.cell_value(i, 5))
    #print(sheet.cell_value(i, 0))
X=np.c_[col1,col5]
y=target
dataAll=np.c_[col1,col2,col3,col4,col5]
X_PC = PCA(n_components=3).fit_transform(dataAll)
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(dataAll, y,
test_size=0.3,random_state=109)#takes 30% of Data for testing and 70% fpr training
#Gaussian Naive Bayes model
gnb = GaussianNB()
#Train phase
gnb.fit(X_train, y_train)
#Prediction
y_pred = gnb.predict(X_test)
#check accuracy calculation
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

cmap_background= ListedColormap(['#A7FFA4', '#5F8FF4']) # for coloring zone
gnb.fit(X, y)
x_min, x_max = X[:, 0].min() - .1, X[:, 0].max() + .1
y_min, y_max = X[:, 1].min() - .1, X[:, 1].max() + .1
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100), np.linspace(y_min, y_max,
150))
Z = gnb.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape) #need to fit shape for plt.pcolormesh

# the following is just to show representation of decision boundaries in 2d
# Plots
plt.figure("Naive Bayes", figsize=(10, 7))
plt.clf()
plt.pcolormesh(xx, yy, Z, cmap=cmap_background)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_points, edgecolor='k')
plt.title("Naive bayes zones and scatter plot")
plt.xlabel('principal comp 1')
plt.ylabel('Principal comp 2')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```

Output

Accuracy: %88.485 (**When all of the features used**)

Note that different sampling methods of training and test data can yield different accuracy outcomes for the classifier.

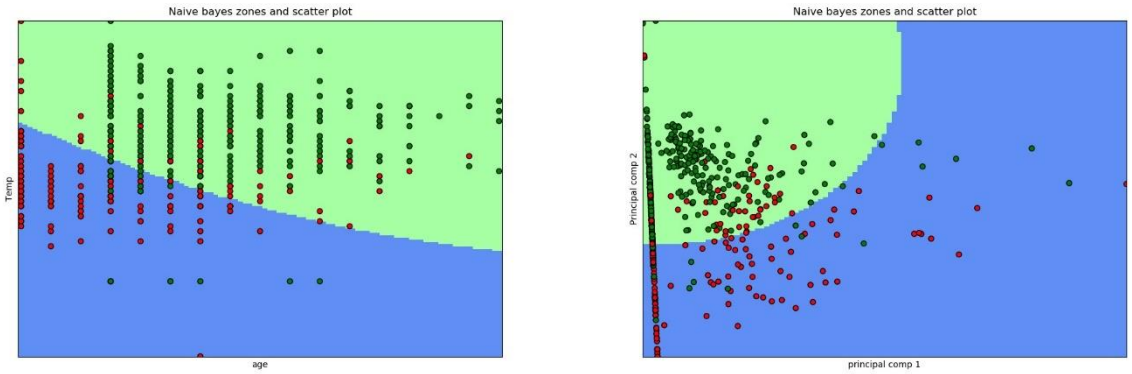


Figure 27 Naive Bayesian Classifier Model Example

Figure 27 shows decision zones if two-dimensional data is chosen for classifier model. Note that accuracy here is decreased when decreasing data dimension, %82.4 and %83.63 respectively, this also shows advantage for using Naïve Bayes classifier as it performs well on large data sets with high dimensional data.

Comparing the Naïve Bayes classifier to the Knn classifier, Knn is a non-parametric technique used in machine learning where no assumptions about the decision boundaries are made where generally for the Naïve Bayes classifier there is an interest to know the function of the distribution in the data. Knn can also yield better result when that data is highly non-linear making it more flexible than other classification techniques.

The optimal choice between these techniques depend solely on the case or application considered.

8.2.4 Support Vector Machines (SVM)[SCM]2d observation of SVM and also]

SVM are very powerful and useful supervised machine learning algorithms that can be applied to many cases for industrial use.

It works by separating data points into classes using a hyperplane. The best hyperplane for the model is the one that has the largest margin, i.e. the distance between the hyperplane and the closest data points (called support vectors) As seen in figure 27.

The algorithm learns by estimating the coefficients that produces best split of classes by that hyperplane.

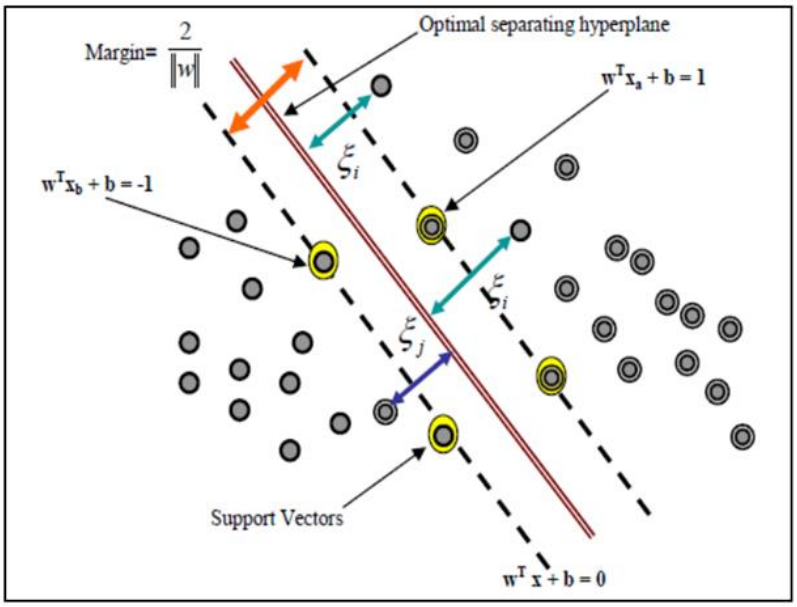


Figure 28 Support Vector Machine separating hyperplane [21]

There are two forms for SVM, linear or non-linear.

For linear:

The equation of the Hyperplane is given by

$$y_i(\boldsymbol{w}.x_i + \boldsymbol{b}) \geq 1 - \epsilon_i \; i = 1, \ldots, m \,,$$

where $\boldsymbol{w}$ is a weighting vector, $\boldsymbol{b}$ is a constant, and ϵ_i *is a nonnegative slack variable*

The margin is given by:

$$\frac{1}{||\boldsymbol{w}||}$$

Therefore, to maximize the margin, the goal is to:

$$\text{Minimize: } \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} \epsilon_i$$

$$\text{Subject to: } y_i(\boldsymbol{w}.x_i + \boldsymbol{b}) \geq 1 \,. \quad i = 1, \ldots, m$$

Capacity *C* is a tuning parameter that weights the in-sample classification error, The smaller is *C*, the wider is the margin and the more and larger in-sample classification errors are permitted. [22]

Using Karush-Kuhn-Tucker theorem, the optimization problem is solved using Lagrange multipliers γ_i.

Finally, the equation can be solved by:

$$\boldsymbol{w} = \sum_{i=0}^{N} \gamma_i y_i \, x_i$$

$$\boldsymbol{b} = \frac{1}{2}(x_{+1}^T + x_{-1}^T).w$$

$x_{+1} and\ x_{-1}\ are\ any\ two\ support\ vectors\ belonging\ in\ different\ classes$

For non-linear form:

The hyperplane in the nonlinear form has a nonlinear region that can separate groups more efficiently in many situations.

This is don't by what is known as the Kernel trick seen in figure 29 [21][22] where data is transformed by the Kernel function into a higher-dimensional feature space to make it possible to perform the linear separation.

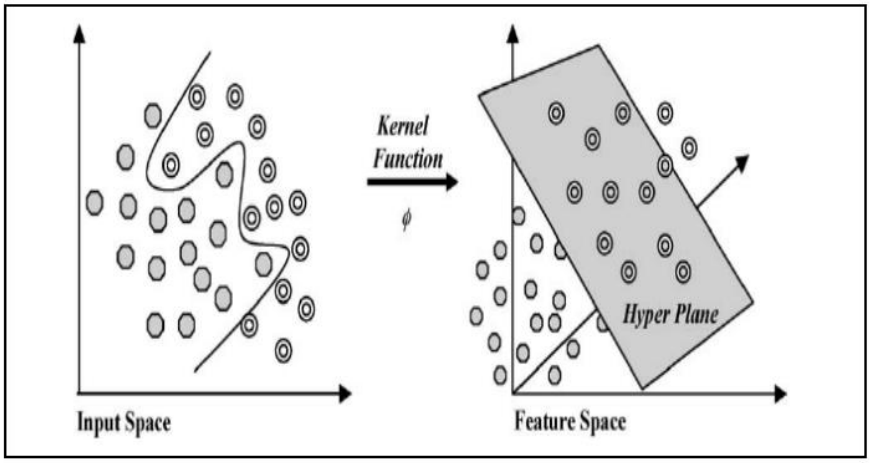


Figure 29 Nonlinear Support Vector Machine (dimension transformation)

The following equation present some classes of Kernel functions

- **Polynomial:** $K(x, x') = (x \cdot x' + c)^q$
- **RBF (radial basis function):** $K(x, x') = e^{-\frac{||x-x'||^2}{2\sigma^2}}$
- **Sigmoide:** $K(x, x') = tanh(\alpha x \cdot x' - b)$

The example below illustrates a SVM model to classify the data used in Knn and Naïve Bayesian filter. To find the optimal values of Parameters, Grid search technique is applied to search for optimal values from given list. (Note that this method is computationally expensive).

Code (Python)

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
import numpy as np
import xlrd
from sklearn import svm

loc=r'C:\Users\Computer Clinic\PycharmProjects\Haitham\KnnClassifier3Data.xlsx'
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)
col1=[]
col2=[]
col3=[]
col4=[]
col5=[]
target=[]
for i in range(1,sheet.nrows):
    col1.append(sheet.cell_value(i, 0))
    col2.append(sheet.cell_value(i, 1))
    col3.append(sheet.cell_value(i, 2))
    col4.append(sheet.cell_value(i, 3))
    col5.append(sheet.cell_value(i, 4))
    target.append(sheet.cell_value(i, 5))
    #print(sheet.cell_value(i, 0))
X=np.c_[col1,col5]
y=target
dataAll=np.c_[col1,col2,col3,col4,col5]

# data split for trainin\testing
X_train, X_test, y_train, y_test= train_test_split(dataAll, y, test_size=0.3,
random_state=0)
# Import the `svm` model

# Support Vector Machine (SVM)
# the following are different parameters to run grid search or cross-validation to
fing optimal values
parameter_candidates = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']},]

clf = GridSearchCV(estimator=svm.SVC(), param_grid=parameter_candidates, n_jobs=-1)
#learning algorythom for determining best parameters  from Parameter_Candidates
clf.fit(X_train, y_train)
print("Optimal Paramters Found\n", 'C',clf.best_estimator_.C ,"\n", "Gamma",
clf.best_estimator_.gamma, "\n", 'Kernel',clf.best_estimator_.kernel)
print('Accuaracy', clf.best_score_)
```

Outcome

Optimal values for parameters are

C: 100

Gamma:0.0001

Kernel: Rbf

Accuracy achieved: 93.45454545454546%

It is seen how the accuracy of SVM and thus this tool is a strong tool to be used in many different classification problems

8.2.5 Other Classifying techniques.

There are many other classification algorithms that this paper will not cover. However below is a brief description of some other techniques used in classification.

In an unsupervised learning algorithm called *linear discriminate algorithm,* data is assumed to have a gaussian distribution and thus outliers are better removed from the data. The algorithm calculates the mean and variance for each class and therefore prediction is by calculating a discriminate value for each class.

In *decision Trees* algorithm which can either be supervised or unsupervised models, the Trees have a binary representation where each node represents a single input variable with a split point on the variable. The leaf nodes or end nodes of the tree represent the output variable and decision are hence made by following the splits in the tree until arriving to the leaf node and outputting the variable or class value in it. This technique is handy when we don't worry about multi-collinearity or feature selection, however for better predictive power on test data, better approaches exist such as the *random forest algorithm* which instead of building a single tree on all the data set, it randomly selects observations and features to create multiple decision trees and finally it averages the result.

In K-Means classifiers (another unsupervised learning algorithm), data are identified in different K clusters representing different classes. In step1, K is defined a value that improves model efficiency considering the bias/variance tradeoff. In step 2, randomly k distinct data are selected as initials clusters. In step 3, the distances between for each point ***x,*** in *n* dimensions, is measured against the three initial clusters. In step 4 we assign the point to the nearest cluster. In step 5, after we assigned the points to the initial clusters, we calculate the mean for

each cluster and afterwards update the location of the clusters to the calculated means. In step 6, we keep on iterating and repeating steps 3 to 6 until the clustering stops changing or we set a stop point.
K-means might not yield accurate results for clustering because of different cluster sizes in original data. To improve the quality of K-means we add the variation within each cluster and the algorithm then chooses the clusters in a way that shows somehow balanced variation among clusters.
Another disadvantages for k-means is that it might be hard to find optimal value for K depending on the case studied.

8.3 Some techniques in Time series forecasting

Time series predictions and forecasting is very important to consider since many problems have a time component related to it. For instance, for our interest, an engine sound output signal, *y(t*), can be strongly related to the measured value *y(t-1) and y(t-2).* The time factor usually complicates problems and thus different approaches are required to predict future behavior. As discussed earlier, data is checked for autocorrelations and some predictive techniques had assumptions that errors are not correlated and that knowing system previous behavior doesn't infer anything about systems future behavior. Fortunately, existing tools and techniques exist to manage time series forecasting.

Visualizing time series data usually reveals a lot of useful information from which we can notice trends, seasonality, cycles, noise, and levels [23]. Techniques like Auto-regression analysis, Moving Average analysis, or the combined Autoregression Moving Average, or Exponential Smoothing, etc. are used to allow for future prediction in such occasions. One way, for instance, to make future predictions is using the most simple form for the "persistence model" which simply assumes that the conditions for the data between time *t* and time *(t+h)* remain unchanged and considering a stationary time series problems the outcome at y(t+h)= y(t). However, for our interest in industrial use in maintenance, we are more interested with data that has changing conditions and variances. One approach to this can include de-seasonalizing and de-trending to data and then modeling to the relationship with time by using regression analysis before finally using a multiplicative factor to estimate future predictions to the process.
In the MR technique, however, we choose the number of previous outcomes we desire for our process, apply different weights on them, and finally we calculate the average of them in

order to predict the outcome in future or we can apply. Whereas, in the Exponential smoothing techniques, we smooth the time series data by applying an exponential function to predict future outcomes.

8.4 Neural networks

Neural networks are deep learning method that emulate the learning process of neurons in the brain. The main concept behind them is that a number of neurons exist to take the values of the inputs and an output layer of neurons exist to output the result of the model. Between these two layers there are hidden layers that we cannot know for sure how do they operate in order to transfer inputs to outputs. The structure of neural networks is shown in the figure below.

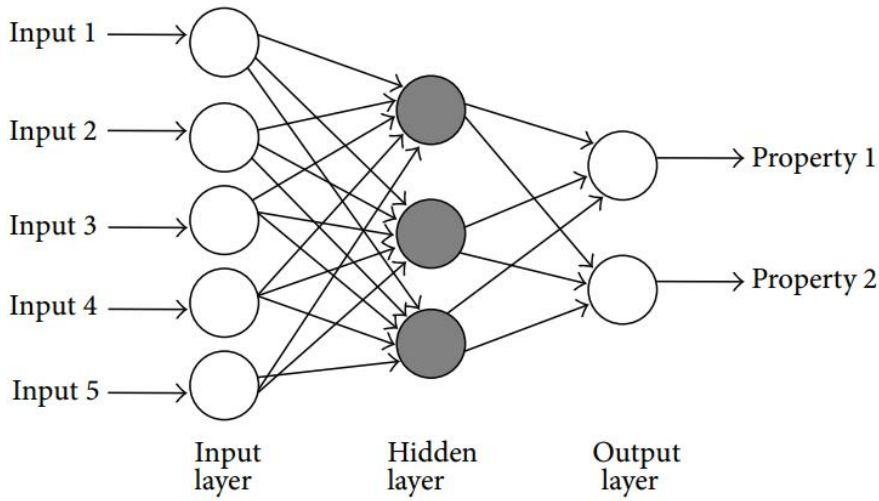


Figure 30 Neural network representation [24]

The values these neural cells can hold are between one or 0 and 1 (this value is referred to as the activation of the neural cell), This can be implemented by applying a sigmoid function to the data as was shown in logistic regression in which the values are pulled to range between 1 and 0. A simple illustration of that is shown in figure 30 showing a single neural representation.

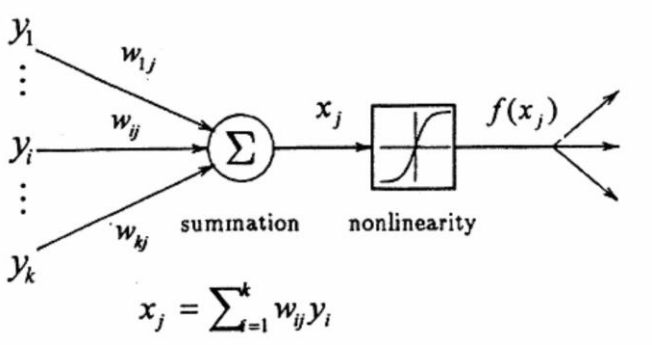


Figure 31 Single neuron function [25]

Neural networks are computationally expensive to learn since all possible connection between neuron layers are tested in the training phase to update the weights *w* to values that would best transfer inputs to the desired outputs. The following equation shows the equation of a neuron at layer *l+1*, were *W* is a weight vector connected between layer *l+1* and *l* and *b* is a bias vector connected between layer *l+1* and *l*.

$$a_{l+1} = \sigma(W_l a_l + b_l)$$

The learning method for the neural function starts by identifying a cost function and measuring the distances between the outputted value in each neuron in the output layer and the desired values that one wished to have seen as an output.

The goal of the machine learning algorithm is then to minimize this cost function using gradient descent concept, one common method for finding the appropriate weights and bias is known as back propagation

The following example I made will illustrate a neural network in Matlab that tries to learn the shape of the function $t = 1 + \sin(\pi * \frac{p}{4})$

Code used to build model (Matlab)

```
p=0:0.05:4; %input
t=1+sin(pi*p/4);% target
i=0;
j=1;
alpha=0.1;

% forward propagation
W1=[-0.27;-0.41;0.2];
b1=[-0.48;-0.13;01];
W2=[0.09 -0.17 0];
b2=0.48;

for i=1:2000 % iteration used
  for j=1:length(p)

a1=logsig(W1(1)*p(j)+b1(1));
a2=logsig(W1(2)*p(j)+b1(2));
```

```
a3=logsig(W1(3)*p(j)+b1(3));
a=purelin(W2*[a1;a2; a3]+b2);
e=t(j)-a;

%back propagation

S2=-2*e;
S1=[(1-a1)*a1 0 0;0 (1-a2)*a2 0; 0 0 (1-a3)*a3]*W2'*S2;
% update of the weight

W2=W2-alpha*[a1;a2; a3]'*S2;
b2=b2-alpha*S2;

W1=W1-alpha*S1*p(j);
b1=b1-alpha*S1;
   end

end
w2f=W2
w1f=W1
b2f=b2
b1f=b1

for k=1:length(p)
    a1(k)=logsig(W1(1)*p(k)+b1(1));
a2(k)=logsig(W1(2)*p(k)+b1(2));
a3(k)=logsig(W1(3)*p(k)+b1(3));
a(k)=purelin(W2*[a1(k);a2(k); a3(k)]+b2);
end
plot (p,t);
hold on
plot(p,a,'r')
```

Output

w2f = 2.0950 -0.5784 2.3384

b2f = -1.7477

w1f =

1.4648

0.0520

-1.3332

b1f =

-0.6869

-0.2637

4.9442

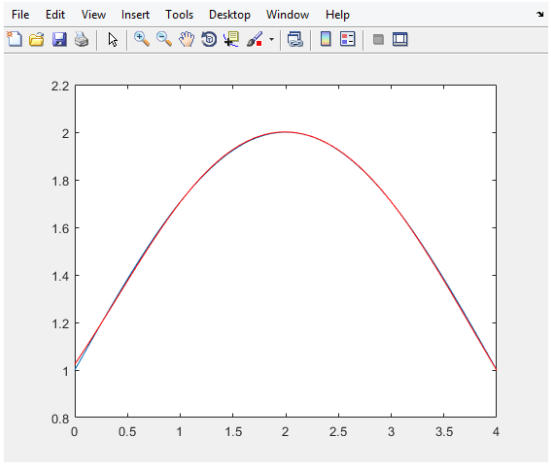


Figure 32 Neural network example learning function shape

There are many different forms and uses for neural networks, come more advanced form are known as conventional neural networks that are good for image recognition, however this will be considered as potential future work in this topic.

Neural networks although have a disadvantage by the time needed to learn in the training phase, especially when dealing with big data, they however can perform very fast in the testing or deployment phase for real time results.

The Other big advantage of Neural networks is that they can learn any function based on updated weights in their hidden layers. Whether simulating Kalman filter in linear conditions or more complex functions.

9 Chapter 8. Two conducted projects to implement the framework proposed in thesis.

9.1 Project 1

In this example, a sensor sends live data and the algorithm is able to read this live data. The model here has historical data uploaded that it reads and analyses using the Naïve Bayesian classifier explained before. The algorithm learned by itself the classification criteria and is able to classify new incoming live data into classes. Based on a certain number of deviations detected in the system, the algorithm automatically sends an email message with an attachment file.

What can be achieved by simple additions to this code is very big with little effort once the model is working. For instance, consider the case of alert message being sent to a list of maintenance engineers and each engineer with time makes actions to repair or fix the process and a score is graded to each person based on system performance. The algorithm can then learn how different issues or classes in data detected are best optimized by a specific group of engineers or individual. As a response with time, the algorithm can choose autonomously according to the problem detected which maintenance guy is more fir to check for the solution.

Criteria for measuring individual fitness can be any criteria imagined as long as outcome is measurable. This also applies to finding root causes and detecting them without personnel inspections. Each time a variation in process is recorded, the algorithm studies the inputs of parameters related to the process, and with time the algorithm can then know if that specific problem occurred and the parameters have that specific value, then the root cause of the problem will be highlighted by the algorithm to notify engineers where to check for the problem.

```
import serial
import matplotlib.pyplot as plt
import atexit
import smtplib, ssl
from email import encoders
from email.mime.base import MIMEBase
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from sklearn.naive_bayes import GaussianNB
import matplotlib.pyplot as plt
#Training Naive Bayesian FIlter
import numpy as np
from sklearn import neighbors
import xlrd
import time
loc=r'C:\Users\Computer Clinic\PycharmProjects\Haitham\KnnClassifier4Data.xlsx'
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)
col1=[]
target=[]
warning=[]
info=[]

for i in range(1,sheet.nrows):
    col1.append(sheet.cell_value(i, 0))
    target.append(sheet.cell_value(i, 5))
X=np.c_[col1]
y=target

gnb = GaussianNB()
gnb.fit(X, y)

# Training GNB
```

```
values = []
plt.ion() #keep ploting live
cnt = 0
serialArduino = serial.Serial('COM3', 9600)



# pre-load dummy data
for i in range(0, 26):
    values.append(0)

while True:


    valueRead = serialArduino.readline(500)

    # Check for valid values from serial
    try:
        valueInInt = int(valueRead)


        print("Distance is", valueInInt)
        value = valueInInt
        if value>200:
            print("observation error")
            pass
        else:
            use = np.c_[value]
            outcome = gnb.predict(use)
            plt.scatter(valueInInt, 6)
            plt.pause(0.05)
            plt.gca().axes.get_yaxis().set_visible(False)
            plt.title("Real-time Ploting of Range values")
            plt.xlabel("Rang values")
            plt.show()
            print(outcome[0])

        if value>200:
            value=np.c_[value]
            outcome=value
            info.append(outcome[0])
            outcome[0]=0
        else:
            use = np.c_[value]
            outcome = gnb.predict(use)
            plt.scatter(valueInInt, 6)
            info.append(outcome[0])

        if outcome[0]==0:
            warning.append(0)
            if warning.count(0)==3:
                #send email.
                subject = "Machine abnormal behavior has been detected"
                body = "Sensor A has detected a bad behavior in machine B. An Alert
has been sent to you as programmed "
                sender_email = "Thesis.Project1992@gmail.com"
                receiver_email = "rami_nd@live.com"
                # Create a multipart message and set headers
                message = MIMEMultipart()
                message["From"] = sender_email
                message["To"] = receiver_email
                message["Subject"] = subject
                message["Bcc"] = receiver_email  # Recommended for mass emails
                # Add body to email
                message.attach(MIMEText(body, "plain"))
                filename = "UiT.png"
                # binary opening of file
                with open(filename, "rb") as attachment:
```

```
                    # Add file as application/octet-stream
                    # Email client can usually download this automatically as
attachment
                    part = MIMEBase("application", "octet-stream")
                    part.set_payload(attachment.read())

                # Encode file in ASCII characters to send by email
                encoders.encode_base64(part)

                # Add header as key/value pair to attachment part
                part.add_header(
                    "Content-Disposition",
                    f"attachment; filename= {filename}",
                )

                # Add attachment to message and convert message to string
                message.attach(part)
                text = message.as_string()

                # Log in to server using secure context and send email
                context = ssl.create_default_context()
                with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as
server:
                    server.login(sender_email, "MyThesis1992")
                    server.sendmail(sender_email, receiver_email, text)

                del warning[:]
                del info[:]

        if len(info)>=10:

            if info[8]==0 and info[9]==0:
                pass
            else:
                del warning[:]
                del info[:]

    except ValueError:
        print("not valid")
```

Outcome

Distance is 111

1.0

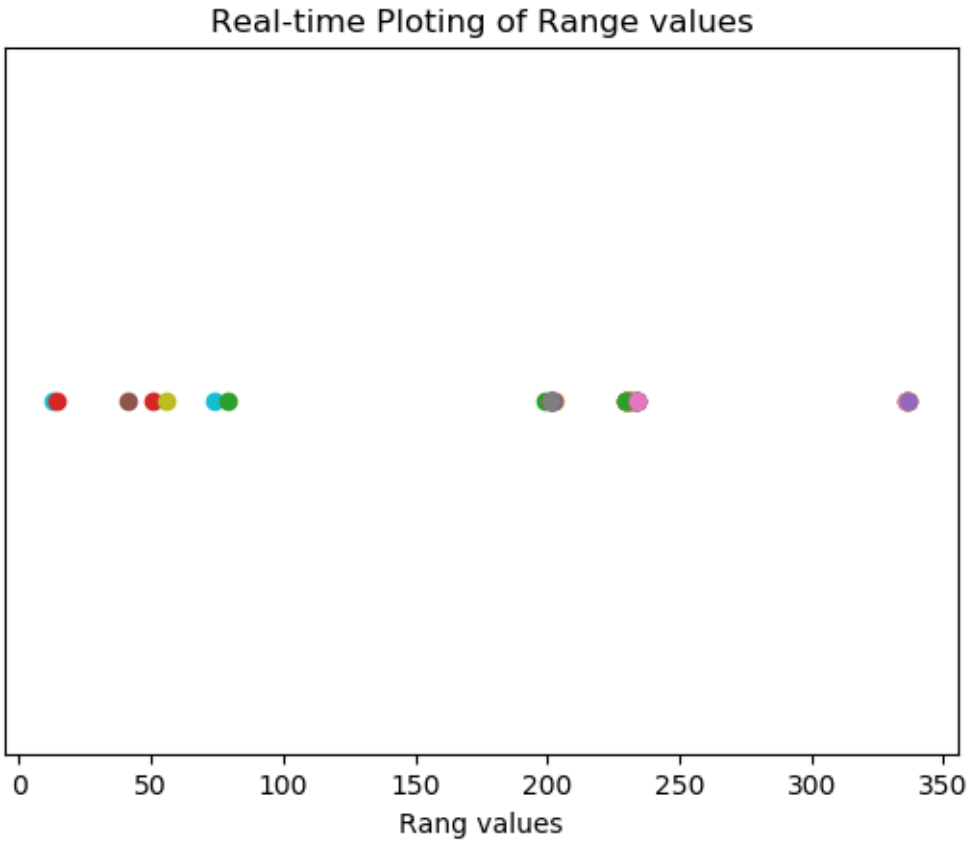
Distance is 115

1.0

Distance is 10

0.0

Real-Time Data Plot



9.2 Example 2

This Example also uses sensor signal from previous example in real time and shows the application of lean six sigma methodolgy through developing control charts to monitor the production process. The devloped model is able to aumotaically create the contol charts and update them in real time while process is running. The algorithm is able to detect when the system has exceeded the specification or control limits, and when a trend or a number of values below the average point is detected to send a message informing management of process status. The update of control limits can be used when testing process improvemnet and setting a new standart for the process. This live data can also show with real time how the process shifts with time with a value of 1.5 sima as described by motorola. By being able to consntanly monotr the process in real time. Causes of variations might be understood and even modeled to understand how the system is behaving. Belowe is the code for constructing the algorithm. (notice that updating system behavior in industry 4.0 can be done as it is done

here, simply by playing the new algorithm on the same component and circtuit, this alone has alot of benefits when it come to SMED and Chaning programs for machines in real time)

```
import serial
import matplotlib.pyplot as plt
import numpy as np
import xlrd

loc=r'C:\Users\Computer Clinic\PycharmProjects\Haitham\KnnClassifier5Data.xlsx'
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
sheet.cell_value(0, 0)
col1=[]
target=[]
warning=[]
info=[]


for i in range(1,sheet.nrows):
    col1.append(sheet.cell_value(i, 0))
    target.append(sheet.cell_value(i, 1))
X=np.c_[col1]
y=target


from statistics import mean, stdev
import scipy.stats as st


succ=col1[275:552]
average_succ=mean(succ)
print("mean in original data is",average_succ)
std_succ=stdev(succ)
print("the std of succ is (original data) is", std_succ)

USL=100
LSL=50
z_val=(USL-average_succ)/std_succ
z_va2=(LSL-average_succ)/std_succ
Probability=st.norm.cdf(z_val)
Probability2=st.norm.cdf(z_va2)
print("probability to go above USL=", 1-Probability)
print("probability to be beyond LSL=", Probability2)


moving_Range=[]
count=0
for i in range(274):
    #print(succ[i+1], "-",succ[i])
    count=abs(succ[i+1]-succ[i])
    #print(count)
    moving_Range.append(count)
MR_average=mean(moving_Range)
UCL=average_succ+2.66*MR_average
LCL=average_succ-2.66*MR_average

Moving_range_UCL=3.27*MR_average
print("strata =", Moving_range_UCL)
#end

values = []
serialArduino = serial.Serial('COM3', 9600)



iterationNum=0
```

```
t=0
initialize1=[]
initialize2=[]
line_remove=0

checkNum=0 # to be used for averages
checkNum2=0
while True:


    valueRead = serialArduino.readline(500)
    # Check for valid values from serial
    try:
        valueInInt = int(valueRead)
        initialize1.append(valueInInt)
        plt.scatter(t, valueInInt)

        print("distance is,", valueInInt)
        if valueInInt > UCL or valueInInt < LCL:
            print("process witnessed deviations, it is not statistically under
control")


        if valueInInt > USL or valueInInt < LCL:
            print("process don't meet specification requirements, process must be
reviewed")

        if MR_average>Moving_range_UCL:
            print("average moving range of the process is above control limit")


        if valueInInt>average_succ:
            checkNum+=1
            checkNum2=0
        elif valueInInt<average_succ:
            checkNum2+=1
            checkNum=0
        if checkNum2>6 or checkNum >6:
            print("process witnessed deviations, it is not statistically under
control")


        plt.plot([0,t+10],[UCL,UCL],'r-', lw=2,label="Control Limits")
        plt.plot([0, t + 10], [LCL, LCL], 'r-', lw=2)

        plt.plot([0, t + 10], [100, 100], 'b-.', lw=1, label= "Upper Specification
Limit")
        plt.plot([0, t + 10], [50, 50], 'b--', lw=1, label= "Lower Specification
Limit")

        plt.plot([0, t + 10], [average_succ, average_succ], 'y-', lw=1 , label=
"Average")
        if iterationNum==0:

            plt.gca().legend(loc = 'upper right')
            iterationNum=1

        t += 1
        plt.pause(0.05)
        #plt.gca().axes.get_yaxis().set_visible(False)

        plt.title("Control Chart")
        plt.ylabel("Sensor Value")
        plt.xlabel("Time")

        if t==25:

            for i in range(24):
```

```
                # print(succ[i+1], "-",succ[i])
                count = abs(initialize1[i + 1] - initialize1[i])
                # print(count)
                initialize2.append(count)
            new_average=mean(initialize1)
            average_succ=new_average
            MR_average = mean(initialize2)
            UCL = new_average + 2.66 * MR_average
            LCL = new_average - 2.66 * MR_average
            Moving_range_UCL = 3.27 * MR_average
            print("strata =", Moving_range_UCL)
            t=0
            stdN=stdev(initialize1)
            initialize1=[]
            initialize2=[]
            line_remove=1
            iterationNum=0
            plt.clf()
            z_val = (USL - average_succ) / std_succ
            z_va2 = (LSL - average_succ) / std_succ
            Probability = st.norm.cdf(z_val)
            Probability2 = st.norm.cdf(z_va2)
            print("the new mean is", new_average)
            print("the new std is", stdN)
            print("probability to go above USL=", 1 - Probability)
            print("probability to be beyond LSL=", Probability2)

    except ValueError:
        print("not valid")
```

Output

mean in original data is 75.07272727272728

the std of success data is (original data) is 8.0605703042482

probability to go above USL= 0.000992408150170987

probability to be beyond LSL= 0.0009337276453392951

strata = 29.179379562043792

With one of the Updates

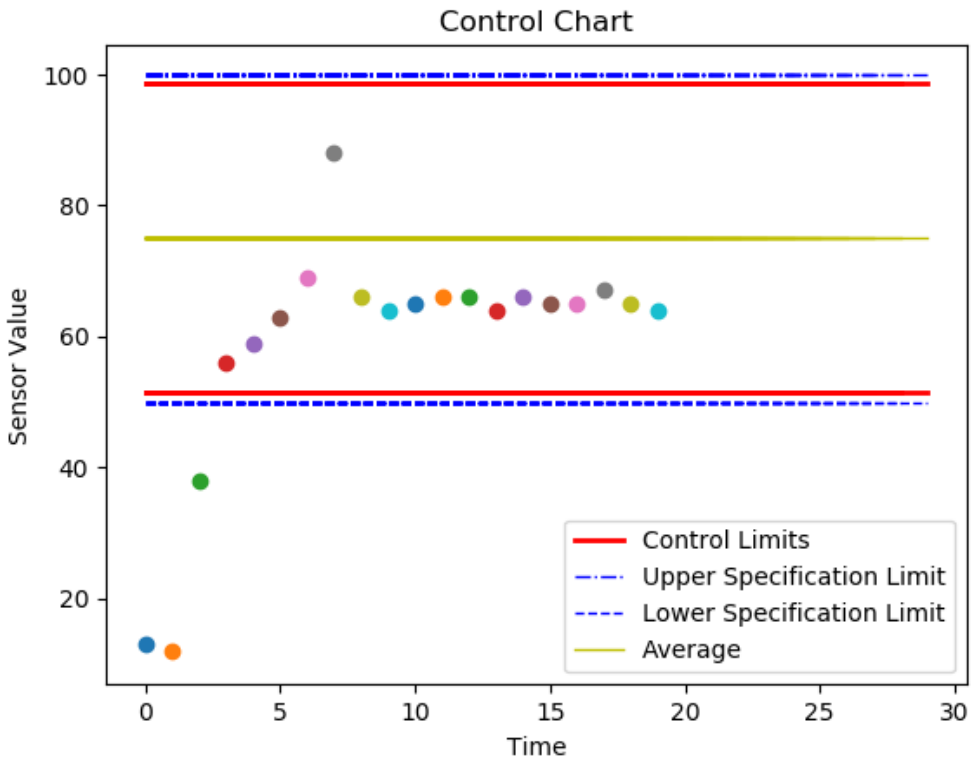
strata = 11.445

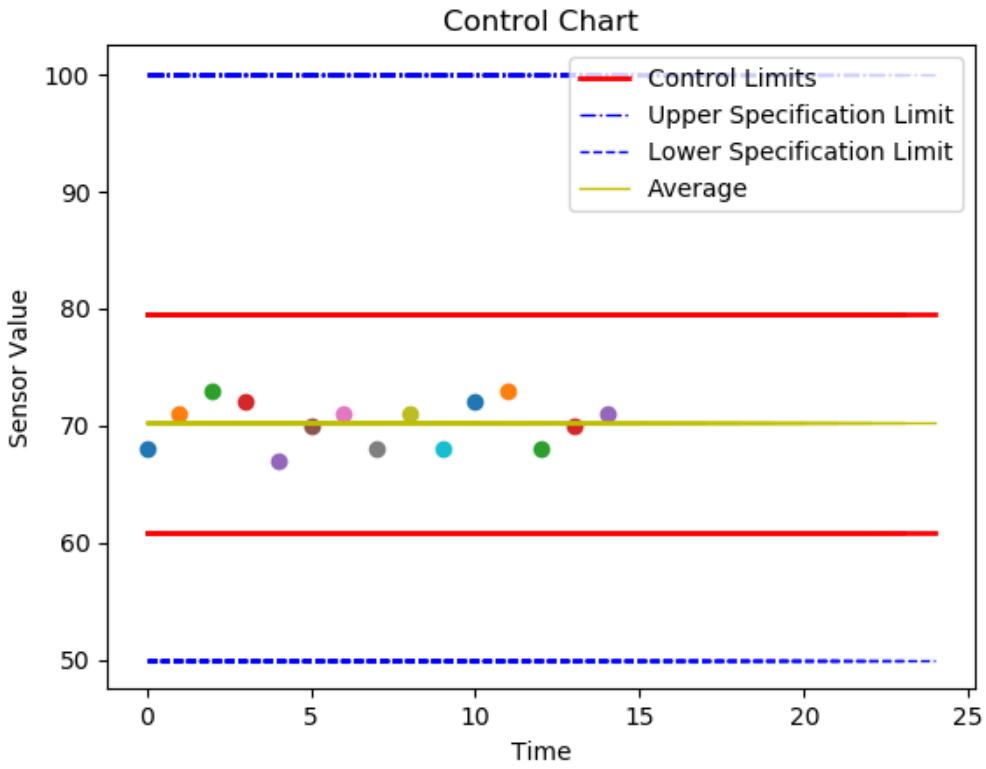
the new mean is 70.24

the new std is 4.657252408878008

probability to go above USL= 0.00011122840981225846

probability to be beyond LSL= 0.006019679545571346





10 Chapter 9 Conclusion

In this thesis, we have seen how new technologies in Industry 4.0 are paving the way for new practices and techniques to be used in maintenance. The proposed concept of industry 4.0 maintenance is characterized by the ability to derive knowledge from historical data or by the ability to learn from new live data in real time.
Since until now, traditional methods of maintenance are being applied in most of the companies, this thesis focused on providing a framework for assisting in the transition of traditional maintenance activities into smart predictive maintenance corresponding to the concept of a smart factory in Industry 4.0.

The concept behind this transition is not to abolish the methodology of traditional maintenance tools but to rather introduce the technologies of industry 4.0 and drive solutions to challenges and optimize the maintenance procedure. Since the main goal of TPM and correspondingly OEE and Lean Six Sigma is to ensure a production process with no wastes,

variations, quality defects and downtime, we can make use of the technologies available in industry 4.0 to provide solutions to many problems and drawbacks faced with the applications of the outdated ways of maintenance, i.e. reactive, preventing and rule-based maintenance.

By providing a deep and detailed view on machine learning algorithms and their usage and applications in maintenance issues, we can be sure that companies need to start thinking of the transition into smart factories not just because of the highly promised profits and benefits guaranteed by appropriately implementing Industry 4.0 technologies, but also to guarantee survival and competency in the market against other competitors.

- The benefits concluded in this report include:
- Reducing maintenance time and unneeded checkups that might be costly and disrupting.
- Increasing efficiency by extending the asset life
- Driving new revenue streams by offering analytic-driven services to customer and thus improving customer satisfaction.
- Improving work environment and achieving a higher level of maturity in operations.

As seen also from the project work of the machine learning algorithm and the potential of industry 4.0 in maintenance, systems can be able to detect root causes that is considered one of the tough tasks experienced engineers face often when issues in the production process appears.

We have also seen an application of Lean Six Sigma methodology to decrease variation in process and ensure continuous process improvement with live data. The major contribution to this methodology is that traditionally the tools in lean six sigma are used on observed data after process execution. The new practive presented was tailoring the toold of Lean Six Sigma to assure quality control and detect process behavior in real-time.

The only challenge as seen in this thesis is that applying Industry 4.0 predictive maintenance requires 3 hard steps that are often challengin

1) Identifying what data exactly need to be studied and defined to describe desired outcome
2) After defining the data, the challenge would be, is it possible to gather the data needed?
3) The third challenge is that skilled data scientist are required to handle these data structure, chose machine learning models, update parameters, and verify before deployment.

Finally, future work that can be suggested to be done as a second step is to study the uses of stochastic based models, reinforcement learning where machine can transfer knowledge to other machines and decrease the training time, application of fuzzy logic where experienced engineers can input their knowledge as part of the autonomous decision making process.

Works cited

[1] Pyzdek, T., & Keller, P. A. (2014). *The six sigma handbook*(Vol. 4). New York, NY: McGraw-Hill Education.

[2] Bekar, E. T., Skoogh, A., Cetin, N., & Siray, O. (2018, August). Prediction of Industry 4.0's Impact on Total Productive Maintenance Using a Real Manufacturing Case. In *The International Symposium for Production Research* (pp. 136-149). Springer, Cham.

[3] Gilchrist, A. (2016). *Industry 4.0: the industrial internet of things*. Apress.

[4] Saturno, M., Pertel, V. M., Deschamps, F., & Loures, E. D. F. (2017). Proposal of an automation solutions architecture for industry 4.0. In *Proceedings of the 24th International Conference on Production Research*. Poznan: ICPR.

[5] Wagner, T., Herrmann, C., & Thiede, S. (2017). Industry 4.0 impacts on lean production systems. *Procedia CIRP*, *63*, 125-131.

[6] Atamuradov, V., Medjaher, K., Dersin, P., Lamoureux, B., & Zerhouni, N. (2017). Prognostics and health management for maintenance practitioners-Review, implementation and tools evaluation. *International Journal of Prognostics and Health Management*, *8*(060), 1-31.

[7] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). CRISP-DM 1.0 Step-by-step data mining guide.

[8] Santos, J. M. F. D. (2007). Data classification with neural networks and entropic criteria.

[9] Bunker, R. P., & Thabtah, F. (2017). A machine learning framework for sport result prediction. *Applied computing and informatics.*

[10] Chatterjee, S., & Simonoff, J. S. (2013). *Handbook of regression analysis* (Vol. 5). John Wiley & Sons.

[11] Gard, D. R. (2011). *Teacher Certification Exams: Predicting Failure on the TExES History (8-12) Content Exam (A Nonparametric Approach using Classification Trees)* (Doctoral dissertation, Texas Tech University).

[12] Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55-67.

[13] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267-288.

[14] Strickland, J. (2015). *Predictive analytics using R*. Lulu. com.

[15] Mohammadi, S., Zuckerman, N., Goldsmith, A., & Grama, A. (2016). A critical survey of deconvolution methods for separating cell types in complex tissues. *Proceedings of the IEEE*, *105*(2), 340-366.

[16] Motulsky, H. J., & Brown, R. E. (2006). Detecting outliers when fitting data with nonlinear regression–a new method based on robust nonlinear regression and the false discovery rate. *BMC bioinformatics*, *7*(1), 123.

[17] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112, p. 18). New York: springer.

[18] Barbé, K., Fuentes, L. G., Barford, L., & Lauwers, L. (2014). A guaranteed blind and automatic probability density estimation of raw measurements. *IEEE Transactions on Instrumentation and Measurement*, *63*(9), 2120-2128.

[19] Mishra, S. P., Sarkar, U., Taraphder, S., Datta, S., Swain, D. P., Saikhom, R., ... & Laishram, M. (2017). Multivariate statistical data analysis-principal component analysis (PCA). *Int. J. Livest. Res.*, *7*(5), 60-78.

[20] *Bou Ammar, H., Tutunov, R., Reinforcement Learning as Probabilistic Modelling: A Variational Inference Formulation (Part II), Blog 2019.*

[21] Auria, L., & Moro, R. A. (2008). Support vector machines (SVM) as a technique for solvency analysis.

[22] Mammass, D., Elhassouny, A., & Ducrot, D. (2013). SVM classification of urban high-resolution imagery using composite kernels and contour information. statistics, 4(7).

[23] Meindl, P., & Chopra, S. (2001). *Supply chain management: Strategy, planning, and operation*. Prentice Hall.

[24] Djuriš, J., Medarević, D., Krstić, M., Vasiljević, I., Mašić, I., & Ibrić, S. (2012). Design space approach in optimization of fluid bed granulation and tablets compression process. *The Scientific World Journal*, *2012*.

[25] Jha, G. K. (2007). Artificial neural networks and its applications. *IARI, New Delhi, girish_iasri@ rediffmail. com*.

Appendix

Matlab Code (Signal Processing)

```
function signal_analysiss(signal,fs)
for i=1:23
    ExpV(i,:)=mean(signal(i,:));
    stdv(i,:)=std(signal(i,:));
end
ExpV
stdv

figure
for i=1:23
subplot(5,5,i)
plot(signal(i,:))
title('signal in time domain')
xlabel('t(s)')
ylabel('x(t)')
end

figure
for i=1:23
subplot(5,5,i)
hist(signal(i,:))
title('histog')
end

figure
for i=1:23
subplot(5,5,i)

pxx=pwelch(signal(i,:)')
plot(10*log10(pxx))

end

figure
for i=1:23
subplot(5,5,i)
 spectrogram(signal(i,:)')
 title('spect')
end

figure
plot (ExpV,'o')
title('Mean across sensors')
xlabel('Sensor Location')
ylabel('Mean Value')

figure
plot (stdv,'o')

title('std across sensors')
xlabel('Sensor Location')
ylabel('std Value')

var=stdv.^2

pause

for j=1:23
for i=1 :23
corcoeff=corrcoef(signal(j,:), signal(i,:));
matrixcoef(j,i)=corcoeff(1,2);
end

end
matrixcoef
for i=1:23
    i
    matrixcoef(:,i)
end
```

```
 figure
 for i=1:23    % Autocorrelation of all signalss
subplot(5,5,i)
[corrval,lags]=xcorr(signal(i,:));
plot(lags,corrval)
xlabel('lag')
ylabel('Autocorrelation')
end
figure

for i=1:23
Hzerocross = dsp.ZeroCrossingDetector;
signalnumber=i
NumZeroCross = step(Hzerocross,signal(i,:)')
end

for i=1:23

    signalnumber=i
energy=sum(abs(signal(i,:)).^2)
end
figure
m = length(signal(i,:));      % Window length
n = pow2(nextpow2(m));  % Transform length
y = fft(signal(i,:)',n);       % DFT of signal
f = (0:n-1)*(fs/n)/15360;  % Frequency range
%p = y.*conj(y)/n;       % Power of the DFT
p=abs(y);
plot(f(1:length(f)/2),p(1:length(f)/2))
xlabel('Frequency (Hz)')
ylabel('Power')
freq=f(1:length(f)/2);
[valpow,posfreq]=max(p(1:length(f)/2));
maximumFrequecny=freq(posfreq)

end
```

Arduino Code

```
//Range Finder
// define pins to be used
const int echoPin = 10;
const int trigPin = 9;
const int LedPin = 13;
// define variables to be used
long Time_duration;
int Range; // I used Integer to avoid decimal values

// define Pin Modes
void setup() {
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(LedPin, OUTPUT);
Serial.begin(9600); //Needed to Starts the serial communication that should be communicating with Python program
digitalWrite(LedPin, HIGH);   // turn the LED on
}
void loop() {
// Set Trig To 0 to initialize
digitalWrite(trigPin, LOW);
delayMicroseconds(3); // give some time
// Triger signal
digitalWrite(trigPin, HIGH);
delayMicroseconds(10); //some time to trigger
digitalWrite(trigPin, LOW);
// Read travel time from echoPin
Time_duration = pulseIn(echoPin, HIGH);
// Range
Range= Time_duration*0.034/2;
// Send Value to Serial Port COM3 in this case
Serial.println(Range);
delay(1000); // read data each second
}
```