TALLINN UNIVERSITY OF TECHNOLOGY
SCHOOL OF ENGINEERING
Department of Mechanical and Industrial Engineering

IMPLEMENTATION OF MACHINE LEARNING ALGORITHMS IN INDUSTRY

MASINÕPPE ALGORITMIDE RAKENDAMINE TÖÖSTUSES

MASTER THESIS

Üliõpilane: Reyhaneh Joodattabrizi

Üliõpilaskood: 184696MARM

Juhendaja: Professor Jüri Majak

Tallinn 2020

(*On the reverse side of title page*)

AUTHOR'S DECLARATION

Hereby I declare, that I have written this thesis independently.
No academic degree has been applied for based on this material. All works, major viewpoints and data of the other authors used in this thesis have been referenced.

"07" May 2020.

Author: Reyhaneh Joodattabrizi
/signature /

Thesis is in accordance with terms and requirements

"....."..... 20….

Supervisor: prof. Jüri Majak
/signature/

Accepted for defence

"....."....20… .

Chairman of theses defence commission: .....
/name and signature/

Non-exclusive Licence for Publication and Reproduction of GraduationTthesis[1]

I, Reyhaneh Joodattabrizi (date of birth: 12.12.1991) hereby

1. grant Tallinn University of Technology (TalTech) a non-exclusive license for my thesis

IMPLEMENTATION OF MACHINE LEARNING ALGORITHMS IN INDUSTRY,

MASINÕPPE ALGORITMIDE RAKENDAMINE TÖÖSTUSES

supervised by prof. Jüri Majak,

1.1 reproduced for the purposes of preservation and electronic publication, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright;

1.2 published via the web of TalTech, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright.

1.3 I am aware that the author also retains the rights specified in clause 1 of this license.

2. I confirm that granting the non-exclusive license does not infringe third persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

---

[1] *Non-exclusive Licence for Publication and Reproduction of Graduation Thesis is not valid during the validity period of restriction on access, except the university's right to reproduce the thesis only for preservation purposes.*

_____ (*signature*)

_____ (*date*)

Department of Mechanical and Industrial Engineering

THESIS TASK

Student: Reyhaneh Joodattabrizi (184696MARM)

Study programme, Industrial Engineering and Management (MARM)

main speciality:

Supervisor(s): Prof. Jüri Majak, +372 620 3265

Consultants: ……..(name, position)

…… (company, phone, email)

Thesis topic:

(in English) *IMPLEMENTATION OF MACHINE LEARNING ALGORITHMS IN INDUSTRY*

(in Estonian) MASINÕPPE ALGORITMIDE RAKENDAMINE TÖÖSTUSES

Thesis main objectives:

1. Review on the utilizing machine learning approach in industry
2. Designing a Graphical User Interface (GUI)
3. Applying machine learning algorithms on a generator of a wind turbine

Thesis tasks and time schedule:

No	Task description	Deadline
1.	Review on the utilizing machine learning approach in industry	01.03.2020
2.	Designing a Graphical User Interface (GUI)	01.04.2020
3.	Applying machine learning algorithms on a generator of a wind turbine	01.05.2020

Language: English **Deadline for submission of thesis:** "25" May 2020

Student: Reyhaneh Joodattabrizi ..... "....."..... 2020
/signature/

Supervisor: Prof. Jüri Majak ..... "....."..... 2020
/signature/

Consultant: ..... "....."..... 2020
/signature/

Head of study programme: **Prof. Kristo Karjust** ..... "....."..... 2020
/signature/

Terms of thesis closed defence and/or restricted access conditions to be formulated on the reverse side

TABLE OF CONTENT

TABLE OF FIGURES 7
TABLE OF TABLES 9
PREFACE 10
LIST OF ABBREVIATIONS AND SYMBOLS 11
INTRODUCTION 12
Problem statement 12
Thesis structure 13
1. THEORY 14
1.1 General Background 14
1.1.1 Stage one: setting the research goal 15
1.1.2 Stage two: retrieving data 16
1.1.3 Stage three: data preparation 19
1.1.4 Stage four: predictive analysis and machine learning 23
1.1.5 Stage five: presentation and automation 33
1.2 Samples of machine learning applications 34
1.2.1 Machine Learning Framework for Predictive Maintenance in Milling 34
1.2.2 Use of machine learning in quality assurance of the fermentation process of black tea 36
1.2.3 A supervised machine learning approach to data-driven simulation of resilient supplier selection in digital manufacturing 37
1.2.4 Intelligent traffic control for autonomous vehicle systems based on machine learning 39
1.2.5 Neural Networks for Wind Turbine Fault Detection via Current Signature Analysis 41
2. PRACTICAL APPLICATION 43
2.1 Development of Graphical User Interface (GUI) 43
2.1.1 Inputs 44
2.1.2 Regression 50
2.1.3 Random Forest (RF) 53
2.1.4 Neural Network (NN) 56
2.1.5 Conclusion 61
2.2 Implementation of machine learning techniques 61
2.2.1 Data 62
2.2.2 Applying machine learning techniques 64
2.2.3 Conclusion 72
SUMMARY 73

LIST OF REFERENCES 75
APPENDIX...79

TABLE OF FIGURES

Figure 1.1 Data science process [2] 14
Figure 1.2 An example of applying OKRs to a project [3] 15
Figure 1.3 MySQL Sample Database Schema [5] 18
Figure 1.4 Linear versus nonlinear classification problems [11].....21
Figure 1.5 Prediction of an employee resigning, modelled by random forest technique [15].....23
Figure 1.6 In (a) we have some example (input/output) pairs. In (b), (c) and (d) we have three hypotheses for functions from which these examples could be drawn [18]. 25
Figure 1.7 Linear and logistic regression lines [22].....27
Figure 1.8 Schematic illustration of the random forest for label ranking [24].....28
Figure 1.9 The flowchart of the algorithm of d-NN [25].....29
Figure 1.10 Prediction for analyzing the offering loan to an employee by decision tree method [27] 30
Figure 1.11 Outline of a multilayer neural network [28].....31
Figure 1.12 Linear support vector machine example [30].....32
Figure 1.13 The accuracy of the k-NN algorithm [35] 38
Figure 1.14 Data preparation procedure [36] 40
Figure 1.15 ANN fault degree detection of a linear increasing fault [40].....42
Figure 1.16 ANN fault detection of a transient fault [40].....42
Figure 2.1 Tabs in the GUI.....44
Figure 2.2 Generating data from equation panel 46
Figure 2.3 Visualization of generated data using an equation 46
Figure 2.4 Importing data from Workspace, (left): Workspace, (Right): GUI's panel..47
Figure 2.5 Visualization of imported data from MATLAB Workspace.....48
Figure 2.6 My_Example, generated data.....49
Figure 2.7 Augmented data using full factorial design.....50
Figure 2.8 Regression tab.....51
Figure 2.9 Regression, Poly32 52
Figure 2.10 Regression, Poly43 53
Figure 2.11 Random forest tab.....54
Figure 2.12 Random Forest with 5 bags.....55
Figure 2.13 Random Forest with 100 bags 56
Figure 2.14 Neural Network tab.....57
Figure 2.15 Neural Network structure.....58
Figure 2.16 Neural Network with 2 neurons and 0,04 as the learning rate.....60

Figure 2.17 Neural Network with 50 neurons and 0,06 as the learning rate 60
Figure 2.18 Generator eccentricity, Or and Os are rotor and stator symmetry centers, g is the width of the air-gap. 63
Figure 2.19 Multivariate linear regression for case study 66
Figure 2.20 Random forest for the case study with 5 bags 68
Figure 2.21 Random forest for the case study with 10 bags 68
Figure 2.22 Neural Network for the case study with 10 neurons and 0,05 as the learning rate 71
Figure 2.23 Neural Network for the case study with 50 neurons and 0,04 as the learning rate 71

TABLE OF TABLES

TABLE 1.1 Results of regression algorithms on V B and RUL [31] 35
TABLE 1.2 Results of classification algorithms [31] 36
TABLE 1.3 A classification sample evaluated through the LR algorithm [35] 38
TABLE 1.4 Predicted results for 545 and 652 sections in terms of volume and speed [36] 40
TABLE 1.5 The model parameters [40] 41
TABLE 2.1 Regression for My_Example with different polynomial degrees 52
TABLE 2.2 Random Forest for My_Example with different number of bags 55
TABLE 2.3 RMSE of Neural Network for My_Example with different number of neurons and learning rates 58
TABLE 2.4 Elapsed time of learning in Neural Network for My_Example with 10 neurons and different learning rates 59
TABLE 2.5 Random Forest for the case study with different number of bags 67
TABLE 2.6 RMSE for Neural Network for the case study with a different number of neurons and learning rates 69
TABLE 2.7 Elapsed time of learning in Neural Network for the case study with 10 neurons and different learning rates 70

PREFACE

The thesis topic was provided by the Department of Mechanical and Industrial Engineering of Tallinn University of Technology.

I am deeply grateful to my supervisor, Professor Jüri Majak, who throughout my thesis, helped me to find all answers to my questions. Under his guidance, I was able to successfully complete the task and gain a lot of new knowledge.

I would like to thank Professor Kristo Karjust, Head of Department of Mechanical and Industrial Engineering of Tallinn University of Technology, for helping and guiding me in difficulties throughout my studies at Tallinn University of Technology.

This thesis tries to firstly give an overview of the utilization of machine learning in industry. Secondly, the author provided a Graphical User Interface (GUI) for comparing the result of different machine learning techniques. Finally, three machine learning approaches are applied to a dataset belongs to a generator of a wind turbine.

Keywords: Machine Learning, Artificial Intelligence, Data Science, Industry 4.0, Master Thesis.

LIST OF ABBREVIATIONS AND SYMBOLS

OKR	Objectives and Key Result
DBMS	Database Management System
SQL	Structured Query Language
ML	Machine Learning
LR-RF	Label Ranking – Random Forest
KNN	K-Nearest Neighbours
ANN	Artificial Neural Network
SVM	Support Vector Machine
TCM	Tool Condition Monitoring
LogR	Logistic Regression
RF	Random Forest
DF	Decision Forest
DJ	Decision Jungle
BDT	Boosted Decision Tree
NN	Neural Network
LCR	Inductor Capacitor Resistor
MLP	Multilayer Perceptron Neural Network
HCA	Hierarchical Clustering Analysis
PCA	Principle Component Analysis
AVS	Autonomous Vehicle System
CM	Condition Monitoring
O&M	Operation and Maintenance
WT	Wind Turbine
CSA	Calculus of Self-Modifiable Algorithms
GUI	Graphical User Interface
MLR	Multivariate Linear Regression

INTRODUCTION

In industry 4.0 generation, a revolution has emerged in the manufacturing industry by using Artificial Intelligence technologies. These technologies have been used in various fields, and still, researchers are trying to uncover more aspects of the power of AI technologies. Some examples of using AI tools would be quality control, forecasting of failure modes, predictive maintenance, supply chain management, generative design, price prediction, robotics, etc. with the adoption of Artificial Intelligence, Manufacturing industry is able to make rapid, data-driven decisions, optimize the manufacturing processes, minimize the production costs, and improve the way it serves the customers.

One of the most important features in Artificial Intelligence is using machine learning techniques to predict an event by evaluating the past data collected. Here, the data and its quality are the most valuable assets for reaching the goal of a project.

As it is mentioned, one of the uses of Artificial Intelligence technology is generative design. Engineers are able to use machine learning techniques to generate alternatives of a product design according to defined criteria such as different producing methods, various materials and cost limitations. This study aims to compare machine learning techniques for estimation of the pattern between inputs and outputs in the case of particular practical application and select the best techniques based on obtained results.

Problem statement

This study focuses mainly on two problems in the Machine Learning (ML) area:

1. Designing a graphical user interface (GUI): A GUI is designed for comparing the implementation of different machine learning approaches and visualize the results. One of the objectives to design the GUI is to give the ability to an engineer or a manager who is new to ML field with basic knowledge of programming to be able to visualize the data and compare the results of different machine learning methods. Also, this GUI can be helpful for students in the beginning steps of understanding machine learning concepts and gives them the chance to understand the influential parameters on the results of each learning method, e.g. the number of neurons in neural network structure.

2. Implementation of the machine learning techniques on a particular practical problem: The dataset considered in modelling is associated with the parameters which have the most influence on stress and deformation of electromagnet generator of a wind turbine. The aim of this section is to choose the best machine learning technique with the least error and the highest accuracy to estimate the pattern of the data. The importance of choosing the best performing technique will be followed in the optimization process of designing the structure of the electromagnet generator, which has been discussed in O.Pabut doctoral thesis [1].

Thesis structure

This thesis is organized in two main chapters, and followed by a summary, references and an appendix for GUI code in MATLAB, at the end.

Chapter 1 is composed of two subchapters. The first subchapter presents the general background/literature overview, explaining the theoretical basis of data analysis processes and introduces different machine learning techniques. The second subchapter presents a number of practical samples, where the machine learning techniques are applied in industry and manufacturing systems.

Chapter 2 is composed of two subchapters. The first subchapter provides a design of a Graphical User Interface (GUI) to compare the results of different machine learning methods. The second subchapter includes evaluation comparison of the three widely used machine learning techniques in a dataset related to electromagnet generator of a wind turbine. The aim of this subchapter is to determine the best ML technique for getting the most fitted pattern for the considered dataset.

1. THEORY

In this chapter, an overview of the theoretical basis of the data analysis process and machine learning methods has been presented.

1.1 General Background

In overall, data analysis can be done through five stages which are shown in Figure 1.1. Each stage can be iterated to reach the desire results.

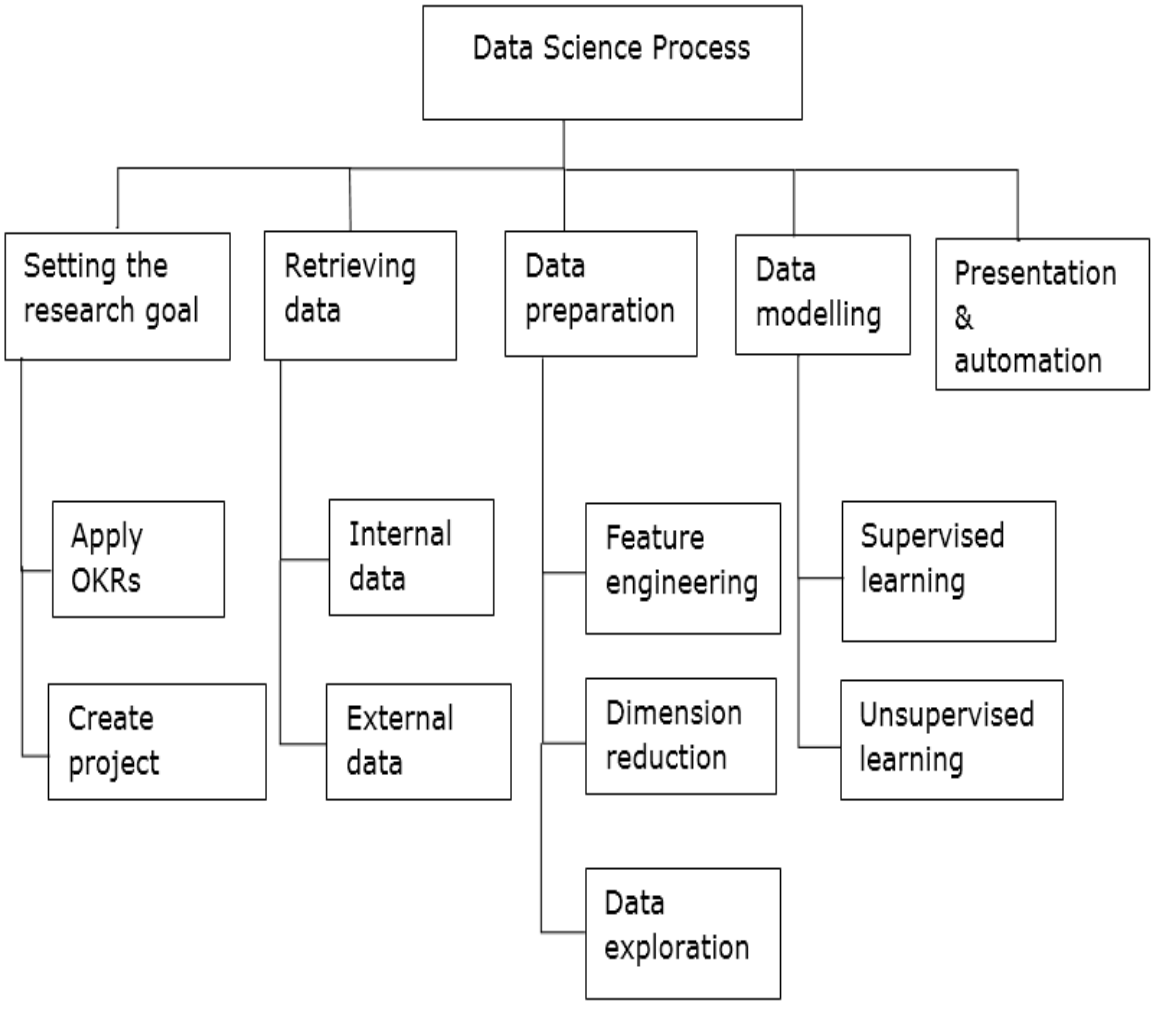

Figure 1.1 Data science process [2]

In the next sections of this chapter, each stage will be explained with more details.

1.1.1 Stage one: setting the research goal

Setting a goal is a crucial stage for doing any project, and the data analysis process is not an exemption. The main purpose of this stage is making sure that the what, how and why of the project is clear for the stakeholders [2].

1. Applying OKRs method to a data science project

OKRs stands for Objectives and Key Results. This method will be used to line and prioritize the resources of a common goal in a company, team and a project. OKRs are mostly used to execute strategic plans across companies in Silicon Valley and Europe[3].

Metrics such as precision, recall, accuracy, or F1-score are the most common metrics in Data Science. Each metrics has advantages and disadvantages, relying upon the nature of the business case. For instance, when there is a highly level imbalanced dataset, accuracy is not the best metric. The most meaningful measurements need to be selected to define whether the business case is solved or not [3].

In many projects, setting metrics and measurements are a part of the objective. It defines where the project wants to go and what is desire output which is demanded. After defining metrics and objectives, the key results need to be defined to show how to achieve the objectives [3]. An example of applying OKRs to a data science project is shown in Figure 1.2.

Detect pedestrians from truck chassis camera with 98% accuracy(IoU 0.5)
Gather data set of 10.000 bounding–box labeled images of pedestrains by 31/01/2019
Implement first prototype of pedestrains detection model by 28/02/2019
Improve model to reach 98% accuracy by 15/03/2019

Figure 1.2 An example of applying OKRs to a project [3]

2. Creating project charter

The objectives and the procedure of the project need to be shown in project charter.

A project charter requires group work, and it should cover at least the below points [2]:

- A clear research goals
- Mission and context of the project
- How to perform the analysis
- What resources are needed
- Proof of concepts and proof that it is an achievable business case.
- Measures and deliverables
- A timeline

1.1.2 Stage two: retrieving data

The aim of this stage is to find the suitable raw data and to get access from the owner of the data. For moving forward in the data analysis process, these raw data might need polishing and transformation in order to make it usable [2].

There are two approaches to access to the data; internal and external:

1. Internal data

In many companies, data is stored in their database in forms of simple text and tables. Here the goal is to get all the relevant data which is needed according to the objective of the project [2].

Database management system (DBMS)

A software which helps companies to organize data that allows easy access to the suitable data is Database Management System (DBMS). Basically, the software acts as an efficient and elaborate file system. With a database program companies are able to retrieve, store, insert and modify data in different ways. Also, the data can be reported and exported in any format that during the time, it will become precious information for the company [4].

Example of database

A concert promoter can store and change data about upcoming concert dates, seating, ticket prices, and sales. After this is done, the promoter can use the software to retrieve information, such as the number of tickets sold in each price range or the percentage of tickets sold the day before the concert [4].

Fourth-generation query language to extract data

In order to retrieve information from a database, a fourth-generation can be used to retrieve information. Data is usually added to databases according to a schedule, and scheduled reports also can be produced. But in case a user needs an unscheduled report, Fourth-generation query language can be used to retrieve the data [4].

MySQL Sample Database Schema

One of the examples of Fourth-generation query language is MySQL, SQL stands for structured query language and can be used to retrieve the unscheduled data. The MySQL sample database schema consists of the following tables [5] (Figure 1.3):

- Customers: stores data related to the customer
- Products: stores a list of scale model cars
- Product Lines: stores a list of product line categories
- Orders: stores sales orders placed by customers
- Order Details: stores sales order line items for each sales order
- Payments: stores payments made by customers based on their accounts
- Employees: stores all employee information as well as the organization structure such as who reports to whom
- Offices: stores sales office data

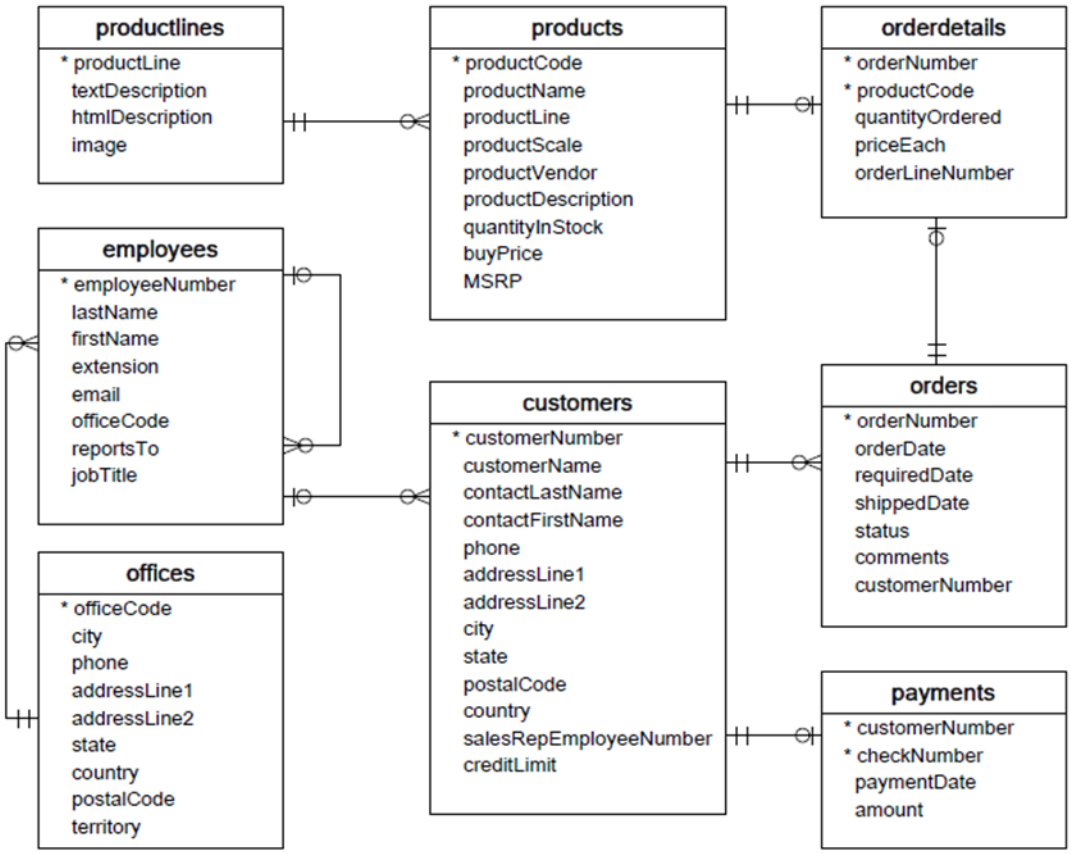

Figure 1.3 MySQL Sample Database Schema [5]

The advantages of the relational database systems can be outlined as [5]:

- enable users to easily categorize and store data that can later be queried and filtered to extract specific information for reports
- are easy to extend and are not reliant on physical organization
- data is stored just once, eliminating data deduplication
- complex queries are easy for users to carry out
- multiple users can access the same database

A NoSQL database is an alternative to relational databases that's especially useful for working with large sets of distributed data and different data formats.

2. External data

Most data warehouses focus only on retrieving data from internal systems. Although this is valuable, many other data are available externally that can significantly increase the value of the data storage.

All data outside the company's operating systems are defined as External data. Data from the spreadsheets of the company are able to be included through SAS/ACCESS directly [6].

The intelligence community emphasizes particularly on the extracting of raw data and information which forms the basis of finished intelligence products, creating agencies assigned exclusively to the collection, processing and exploitation, and analysis of specific intelligence sources. Five basic modalities of the collection have been defined by the CIA(Confidentiality, integrity and availability) model defined [7]:

1. Signals Intelligence (SIGINT): SIGINT is a general category that includes data related to intercepted signals.
2. Imagery Intelligence (IMINT): IMINT consists of data obtained from satellite, aerial, and ground-based collection methods.
3. Measurement and Signature Intelligence (MASINT): MASINT consists of technical information which is not SIGINT or IMINT. MASINT is Mostly related to biological intelligence, nuclear, seismic, and chemical and radar.
4. Human-Source Intelligence (HUMINT): HUMINT includes intelligence data obtained from human sources.
5. Open-Source Information (OSINT): OSINT is information which is available in public and related to newspapers, radio, television, and the Internet.

1.1.3 Stage three: data preparation

When the raw data is gathered, it's time to prepare it for modelling and visualization. For this purpose, first, different kinds of errors in the data must be detected and corrected; raw data needs to be transformed and cleaned. Also, this is in this stage that the test and train sets need to be defined. Then it can be progressed in the subsequent modelling phases of the process. Data mining is a process that consists of analysis, math and statistics. In overall, the process can be divided into 80 % preparation and 20 % analysis [2][7].

In order to execute the data preparation stage, three phases should be processed according to the need.

1. Feature engineering
2. Dimension reduction

3. Data exploration

Each phase is explained in the following sections with more details.

1. Feature engineering

In machine learning, feature engineering is a crucial task for data preparation. It directly affects the performance of the predictive analysis by preparing the features(variables). Feature engineering applies the transformation functions such as arithmetic and aggregation operators on the given variables to produce new ones. In order to scale a feature or convert a non-linear relation between a variable and an output class, into a linear relation, transformation functions will be used; then the pattern will be easier to learn [8]. In order to run the feature engineering process, three steps need to be done [2]:

- Data cleaning (from noises, outliers and null values)
- Transforming data (e.g. converting qualified data to quantified)
- Combining data from different data sources

2. Dimension reduction

In machine learning, "dimensionality" is defined as the number of variables (predictors, features, inputs) in the data store. When the number of variables is very larger than the number of observations and experiments, specific algorithms conflict to train suitable patterns. This is called the "Curse of Dimensionality," and it's specifically related to clustering algorithms that depend on distance calculations. There are two methods to encountered with the dimensionality issues [9]:

- Feature extraction
- Feature selection

Feature extraction

The process of dimensionality reduction is called feature extraction. In this process, an initial set of raw inputs is reduced to more manageable groups in order to be processed. A characteristic of these large data sets is that they include a large number of inputs which require a lot of calculating resources to go through the process. "Feature extraction is the name for methods that select and /or combine variables into features,

effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set" [10].

Feature extraction helps to decrease the number of resources needed for processing without losing significant or relevant information. Feature extraction can also decrease the amount of unneeded data for a given analysis. Also, by reducing the dimensionality of the features, the speed of the learning process will be faster [10].

Feature extraction makes new features by combining of others to decrease the dimension of the selected variables. There are two methods to proceed with the extraction algorithms: linear and nonlinear. The difference between linear and nonlinear model is shown in Figure 1.4 [11].

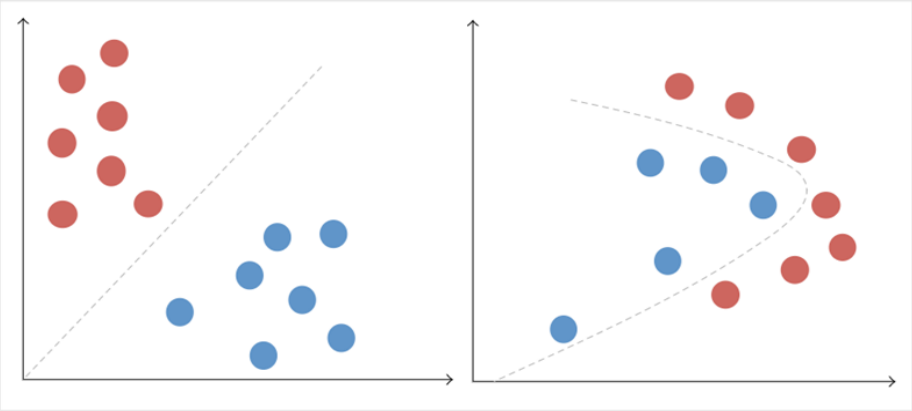

Figure 1.4 Linear versus nonlinear classification problems [11]

Feature selection

In the past few years, the number of high-dimensional data which is publicly available on the internet has greatly increased. This issue makes some difficulties for the machine learning algorithms in dealing with a large number of variables, which is posing an interesting challenge for researchers. In order to run an optimized machine learning process, the preprocessing of the data is crucial. Feature selection is one of the most common and important techniques in data preprocessing and has become an essential component of the machine learning process. It is also called as variable selection, attribute selection, or variable subset selection in statistics. It is the process of detecting relevant features and removing irrelevant, redundant, or noisy data. This process helps to increase the speed of data analysis algorithms, improves predictive accuracy, and increases understandability. [12]

Irrelevant variables refer to those data that provide no useful information, and redundant variables provide no more information than the currently selected variables. In terms of supervised inductive learning, feature selection gives a set of candidate features using one of the three approaches [13]:

- Optimizing the evaluation measures by the specification of the subset's size of the features
- Subsets with a smaller size that satisfies a certain specification on evaluation criterions
- In general, the subset with the most efficient among size and evaluation measure

Difference between feature selection and feature extraction

There is a key difference between feature extraction and selection. Basically, feature selection maintains a subset of the original variables or features, while feature extraction builds brand new ones [9].

3. Data exploration

The goal of data exploration is to gain a deep concept of the data. In this process, data science will look for patterns, correlations, and deviations based on visual and descriptive techniques. The insights are gained from this phase will enable us to start modelling [2].

During this step, the data are gathered, and the analyst begins to search and get familiarity with the data, including form, content, and structure. During the data preparation process, knowledge and understanding of the numeric variables and specifications of the data (e.g., categorical versus continuous data) will be significant. Selection of suitable statistical tools and algorithms used during the modelling phase is necessary. Finally, it is through this primary step that the analyst is able to gain a concept of and familiarity with the data that will be helpful in the following steps to the analytical process, including any modelling, evaluation of the results, and preparation the output and reports [7].

1.1.4 Stage four: predictive analysis and machine learning

Predictive Analytics is a branch of advanced data analysis that uses the various techniques such as machine learning, statistical algorithms and other data mining techniques to predict future events based on historical information. The model is then applied to current data to forecast what would be the next phase of action or suggestion for the output [14][2].

Figure 1.5 illustrates an example of predictive analysis to forecast the resigning of an employee in the next 12 month according to their attributes and specifications [15].

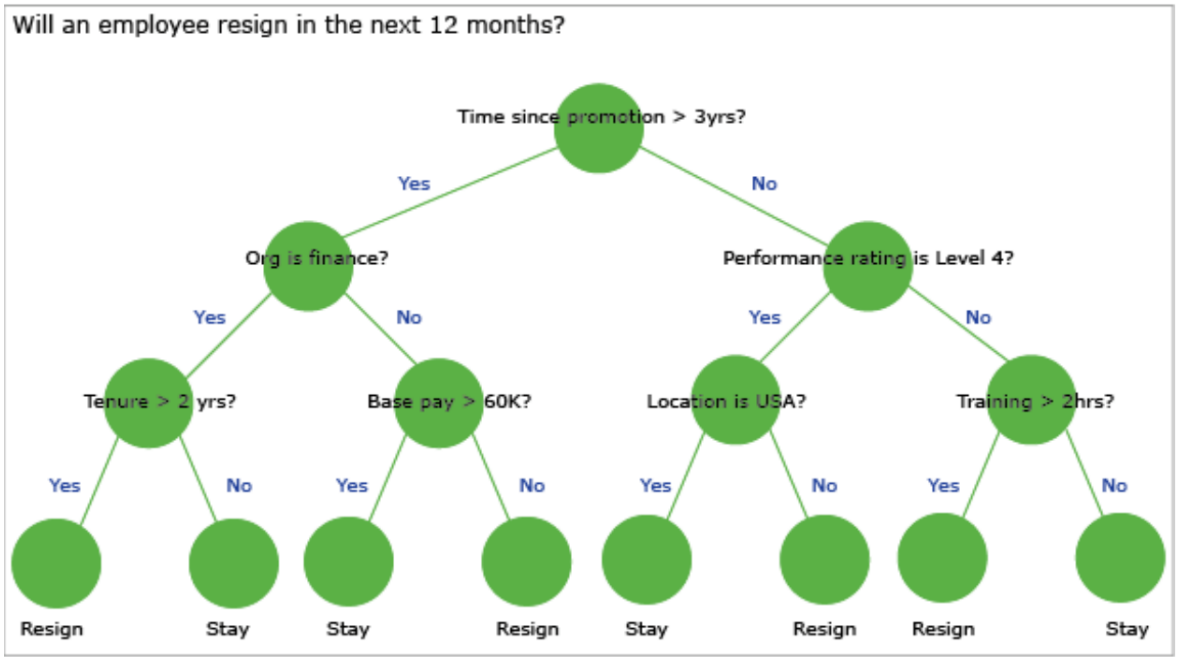

Figure 1.5 Prediction of an employee resigning, modelled by random forest technique [15]

The techniques which are using in the predictive analysis are borrowed from machine learning, data mining and statistics [2].

Machine Learning (ML)

One of the branches of artificial intelligence is machine learning which is dealing with applying various algorithms to learn from data. In the concept of machine learning, "Learning "refers to being able to predict or classify data based on previous data. For example, in network security, machine learning is used to assist with classifying email as a legitimate or spam [16]. Statistical learning and predictive analysis play an

important role in many areas of science, finance and industry. Here are some examples of learning problems [17]:

- Predicting whether a patient hospitalized due to a heart attack will have a second heart attack. The prediction is to be based on demographic, diet and clinical measurements for that patient
- Forecasting the price of a stock in 6 months from now, according to the performance of the company measurements and financial information
- Detecting the numbers in a handwritten ZIP code, from a digitized image
- Estimating the amount of glucose in the blood of a diabetic person, from the infrared absorption spectrum of that person's blood
- Identify the risk factors for prostate cancer based on clinical and demographic variables.

In a typical scenario, there is an output measurement (target), usually quantitative (such as a stock price) or categorical (such as heart attack/no heart attack), which will be forecasted according to a set of attributes (variables such as diet and clinical measurements). There must be a training set of data, in which the output and feature measurements for a set of objects (such as people) will be observed. Using this data, the analyst will create a prediction model, or learner, which will enable the system to predict the outcome for new unseen objects. A good learner is one that accurately predicts such an outcome. The examples above describe what is called the supervised learning problem. It is called "supervised" because of the existence of the output variable to guide the learning process. In the unsupervised learning problem, only the features will be observed, and there are no measurements of the outcome [17].

As mentioned above, statistic learning will be classified into two groups [17]:

1. Supervised learning
2. Unsupervised learning

There is an intersection area between techniques used in supervised and unsupervised learning; some techniques such as ANN can be applied in both.

1. Supervised learning

Supervised learning refers to a learning function that creates an input to an output according to the example of previous input-output pairs [18]. It induces a function from labelled training data, including a set of training examples [19].

Any situation in which both the inputs and outputs of a component can be perceived is called supervised learning. In supervised learning, the learning element is given the correct (or approximately correct) value of the function for particular inputs and changes its representation of the function to try to match the information provided by the feedback. More formally, we say an example is a pair (x, f(x)), where x is the input and f(x) is the output of the function applied to x. The task of pure inductive inference (or induction) is this: given a collection of examples of 'f', return a function 'h' that approximates 'f' as closely as possible. The function h is called a hypothesis [18][20] (Figure 1.6).

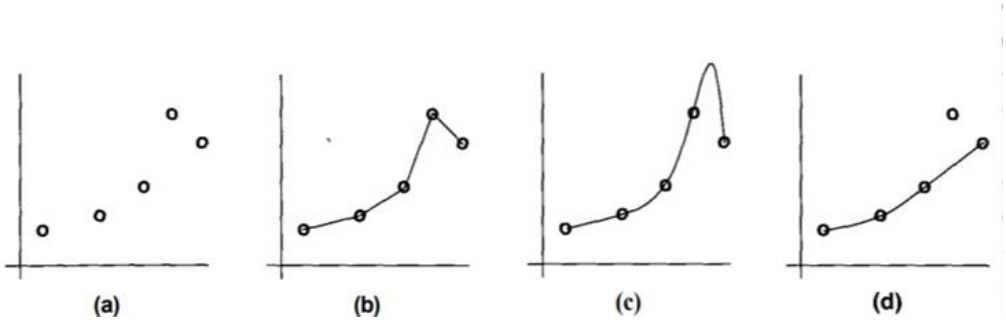

Figure 1.6 In (a) we have some example (input/output) pairs. In (b), (c) and (d) we have three hypotheses for functions from which these examples could be drawn [18].

In supervised learning, each experiment consists of a pair including an input object (typically a vector) and the desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and generates a guesstimated function, which can be used for mapping new experiments [16].

There are two important techniques in supervised learning [16]:

- Linear regression
- Classification techniques

Linear regression

One of the supervised learning techniques is linear regression which is mostly used in predicting, forecasting, and finding relationships between quantitative data. It is one of

the earliest learning techniques, which is still widely used. Some examples of applying these techniques are: to examine if there was a relationship between a company's advertising budget and its sales, to determine if there is a linear relationship between specific radiation therapy and tumour sizes. Linear regression is mostly used when the output is continuous and quantitative [16].

Classification technique

Classification techniques mostly focus on predicting a qualitative output by mining in the data and recognizing patterns. For example, this type of technique is used to classify whether a credit card transaction is fraudulent or not. There are many different classification techniques or classifiers; most important ones are as below [16]:

- Logistic regression
- Random forest
- K-nearest neighbours
- Classification trees
- Artificial neural networks
- Support vector machines

Logistic regression

Logistic regression is a multivariate statistical technique that can be applied to analyze the relationship between independent variables (inputs/predictors), and dependent variables (output/target) to find a pattern which is the best fitting model to describe the relationship between inputs and outputs. Logistic regression is used when the dependent variable is binary, dichotomous, nominal or sequential means there are only two possibilities for the output (yes vs no, positive vs negative, died vs alive, etc.) and there are no constraints for input/predictors [21]. In a simple word, the logistic regression is a nonlinear alteration of the linear regression. The "logistic" distribution is an S-shaped distribution function that the logit distribution possesses the estimated probabilities to lie between 0 and 1. Figure 1.7 demonstrates the logit distribution [22].

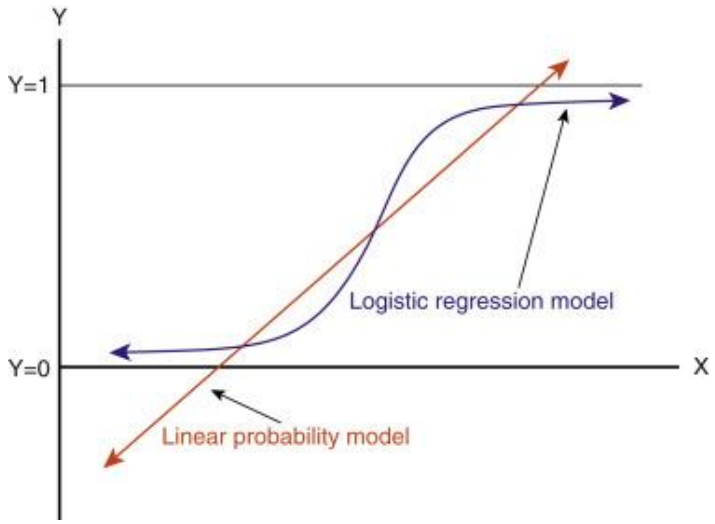

Figure 1.7 Linear and logistic regression lines [22]

Logistic regression generates the coefficients (and its standard errors and significant levels) of a formula to forecast the probability of the presence of the demanded characteristic in the form of a logit transformation:

$$Logit(p) = b_0 + b_1X_1 + b_2X_2 + b_2X_2 + \ldots + b_iX_i \qquad 1.1)$$

where p is the probability of the presence of the characteristic of interest and b_i is the regression coefficient for X_i [22].

Random forest

Random forest is a strong learning algorithm introduced in Breiman [23], which combines several randomized decision trees and cumulates their predictions by their average. It has been one of the most successful general-purpose algorithms in modern times. Here, to describe the random forest algorithm, a label ranking technique as LR-RF has been denoted. The proposed LR-RF (label ranking-random forest) will be processed in two stages. First, at the construction stage, it creates multiple decision trees by using different training instances. After that, at the prediction stage, the query instance goes through all trees. A two-step rank aggregation strategy is used to cumulate the neighbouring rankings into a final predicted ranking. Figure 1.8 demonstrates the entire process from a query instance 'x' to finally obtain the predicted ranking '$\hat{\pi}$' [24].

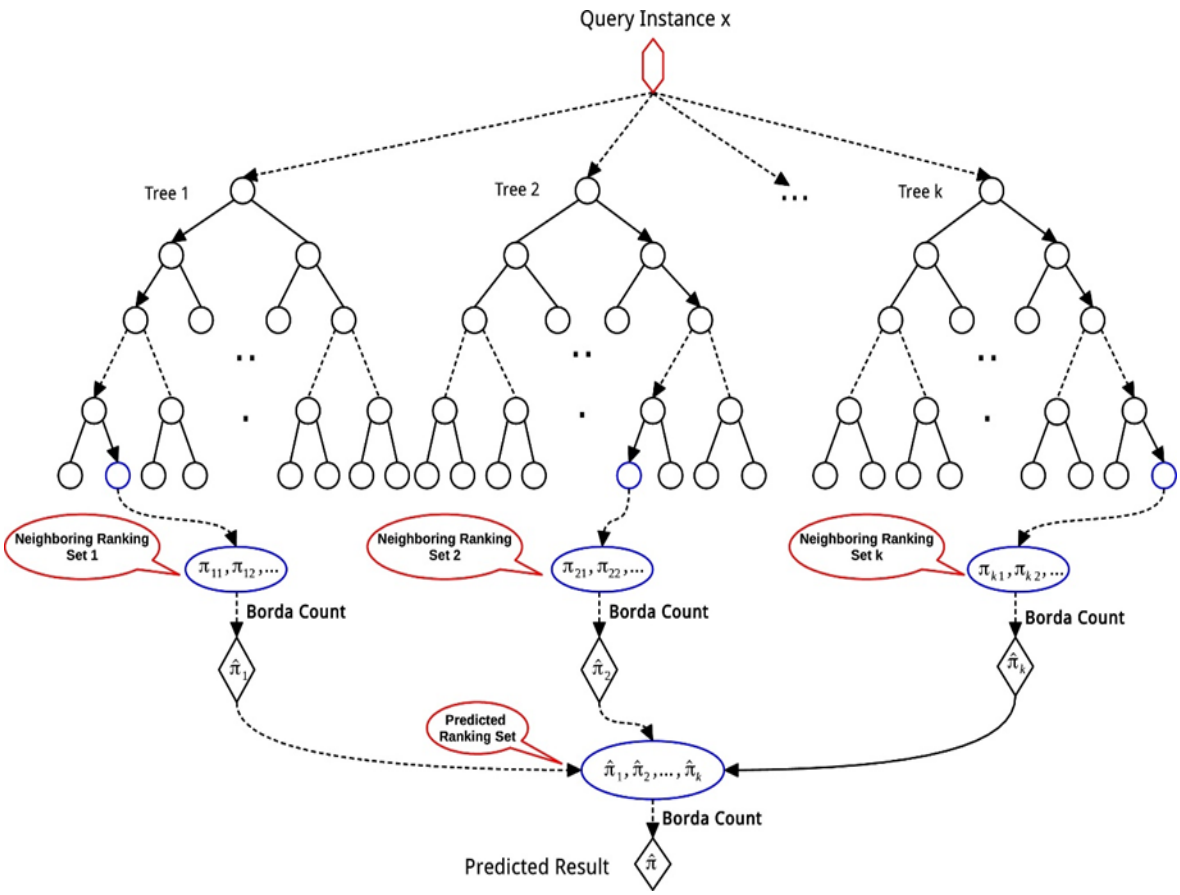

Figure 1.8 Schematic illustration of the random forest for label ranking [24]

K-nearest neighbours

K-nearest neighbour (KNN) is one of the primary techniques behind various machine learning methods. In KNN, a similarity metric, for example, Euclidean distance will be used to analyze the relationship of a query to a neighbouring sample. This process begins with mapping the training dataset onto a one-dimensional distance space based on the computed similarities, and then labelling the query in according to the most dominant or mean of the labels of the k nearest neighbours, in classification or regression issues, respectively. Here, 'K' refers to the number of nearest neighbours, and it is chosen based on the desired limit of success. Nonetheless, two different samples may have equal distances to query, but, their angles may be different in the feature space. In order to differentiate between the two distances in reference to angular information, the similarity of the query to these two samples needs to be weighted based on the angle going between the query and each of the samples. This opinion can be analyzed in the context of dependency and can be optimized to increase the accuracy of the classifier. With this point of view, instead of KNN, the query is labelled according to its nearest dependent neighbours that are defined by a joint function, which is created

based on the similarity and the dependency. This method, therefore, may be called dependent NN (d-NN).

To illustrate the d-NN method, it has been used in combinatorial datasets, which consists of different statistical distributions, and four benchmark datasets, which are Pima Indian, Hepatitis, approximate Sinc and CASP datasets. Results stated that d-NN in terms of accuracy and computation costs has more advantages in comparison to other employed popular machine learning methods [25]. The flowchart of the algorithm of d-NN is shown in Figure 1.9.

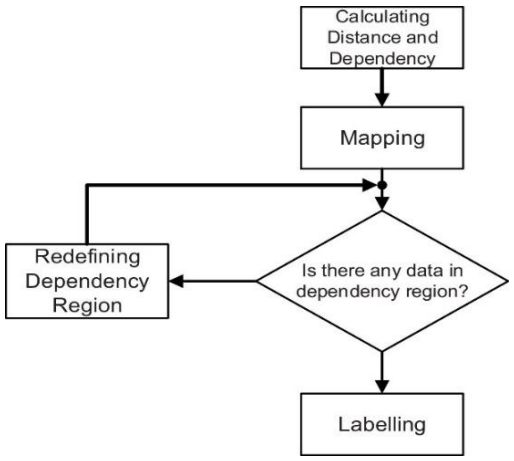

Figure 1.9 The flowchart of the algorithm of d-NN [25]

Classification tree

Classification trees are applied to forecast the membership status of the cases or objects into classes of a categorical dependent variable from their measurements on one or more predictor variables. Classification tree technique has traditionally been one of the main analysis method applied in machine learning and data analysis. The Classification Trees module in STATISTICA Data Miner is a full-featured implementation of techniques for computing binary classification trees based on univariate splits for categorical predictor variables, ordered predictor variables (measured on at least an ordinal scale), or a mix of both types of predictors. Also, there are several options for computing classification trees according to the combination of linear splits for interval scale predictor variables [26].

The flexibility of classification trees makes it a very attractive analysis option, but it doesn't mean that its use is suggested to the exclusion of other techniques. In the opinion of many researchers, classification trees are unsurpassed. Classification trees simply lend themselves to being displayed graphically, helping to make them easier to

interpret than they would be if only a strict numerical interpretation were possible [26]. For example, Figure 1.10 shows that we might have a decision tree in order to decide whether an employee should be offered a loan or not [27].

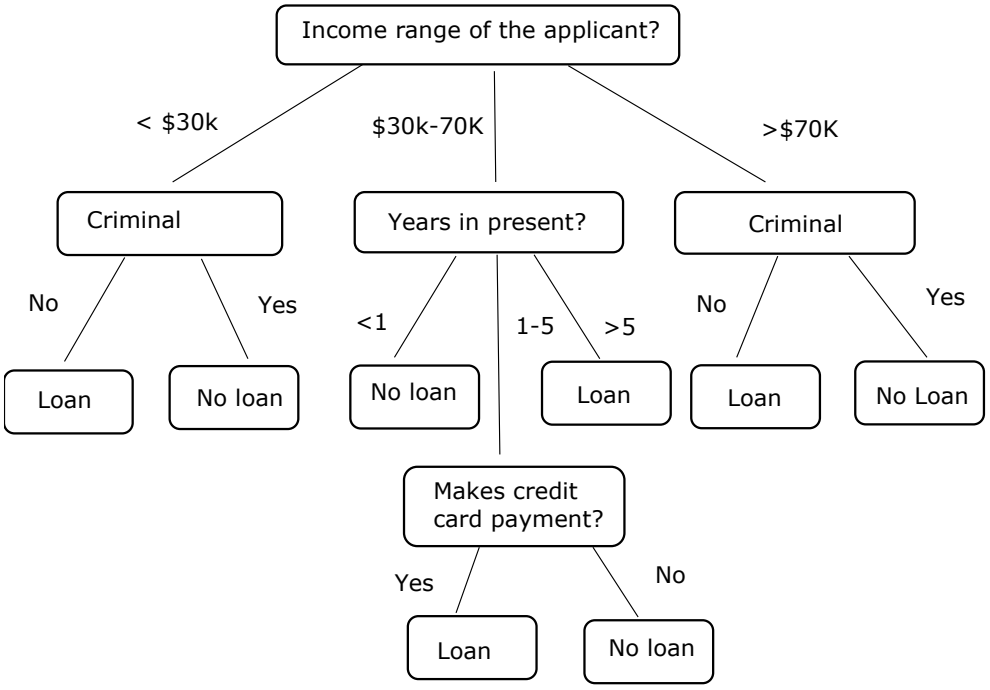

Figure 1.10 Prediction for analyzing the offering loan to an employee by decision tree method [27]

Artificial Neural Network

Artificial Neural Networks (ANNs) are one of the different data analysis methods used to predict the power output of a wind farm using meteorological information forecasted by NWP models.

ANNs algorithm is a simulation of the behaviour of biological neural networks. In comparison to the brain structure, ANNs include single processing units called neurons. In the network structure, the neurons are placed in layers. Each of the neurons in the input layer receives one of the variables (e.g., wind speed and direction, humidity, temperature, and atmospheric pressure) which depend on those variables that we want to predict. Then, in the output layer, neurons return the values of the variables that we wish to predict (e.g., the power output of the wind farm at subsequent instants). Also, there can be several intermediate layers which are known as hidden layers. Connectivity pattern or architecture of the network is the way in which the neurons interconnect.

The examples enable the network to learn and to generalize the acquired knowledge [28]. Figure 1.11 displays an outline of a multilayer neural network.

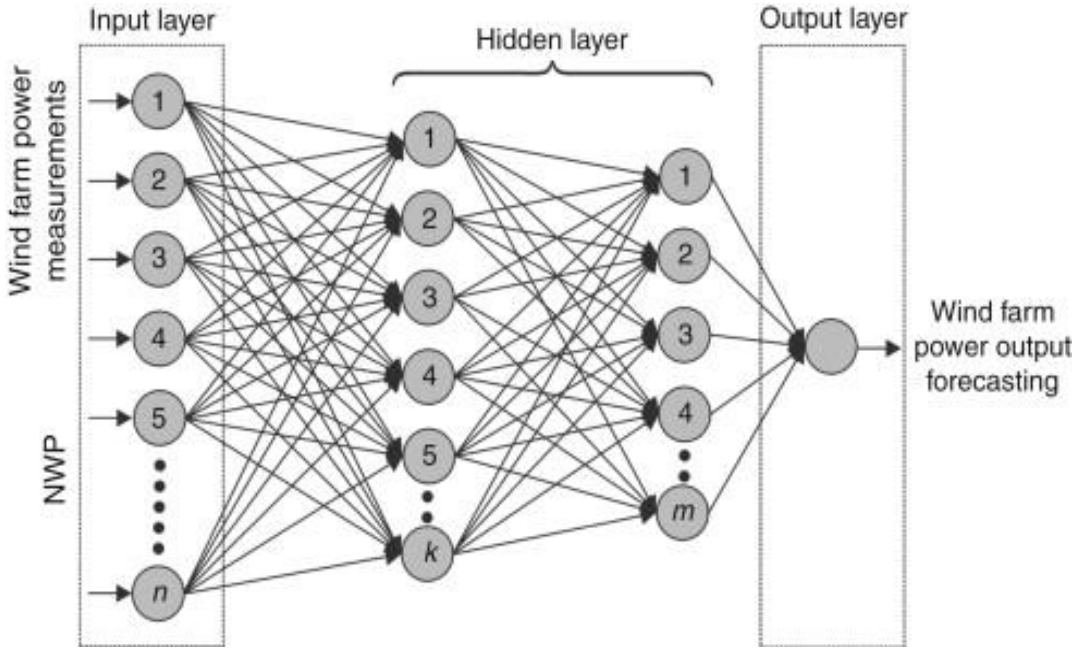

Figure 1.11 Outline of a multilayer neural network [28]

Support Vector Machine

Support vector machines (SVMs) is a supervised non-parametric statistical learning technique. Therefore there is no assumption made on the underlying data distribution. In its original formulation, the method is presented with a set of labelled data instances, and the SVM training algorithm aims to find a hyperplane that separates the dataset into a discrete predefined number of classes in a fashion consistent with the training examples. The term optimal separation hyperplane is used to refer to the decision boundary that minimizes misclassifications, obtained in the training step. Learning refers to the iterative process of finding a classifier with optimal decision boundary to separate the training patterns (in potentially high-dimensional space) and then to separate simulation data under the same configurations (dimensions)[29].

In its simplest form, SVMs are linear binary classifiers that assign a given test sample a class from one of the two possible labels. An instance of a data sample to be labelled in the case of remote sensing classification is normally the individual pixel derived from the multispectral or hyperspectral image. Such a pixel is represented as a pattern vector, and for each image band, it consists of a set of numerical measurements. Elements of the feature vector may also include other discriminative variable measurements based on pixel spatial relationships such as texture. Figure

1.12 illustrates a simple scenario of a two-class separable classification problem in two-dimensional input space. An important generalization aspect of SVMs is that frequently not all the available training examples are used in the description and specification of the separating hyperplane. The subset of points that lie on the margin (called support vectors) is the only ones that define the hyperplane of maximum margin [30].

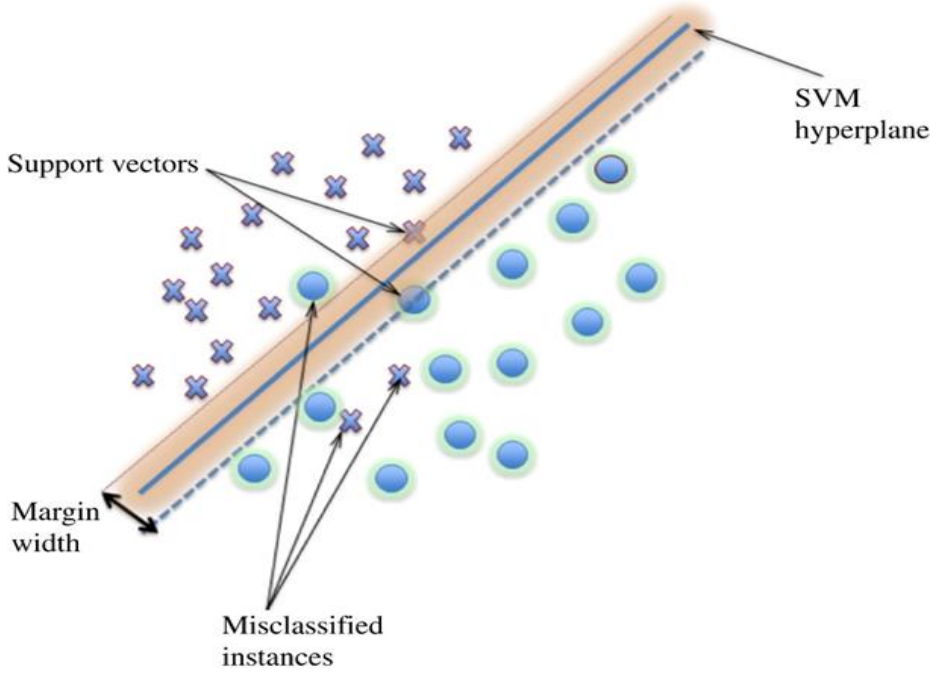

Figure 1.12 Linear support vector machine example [30]

2. Unsupervised learning

Unsupervised learning is the opposite of supervised learning, where unlabeled data is used because a training set does not exist. None of the data can be presorted or pre-classified beforehand, so the machine learning algorithm is more complex, and the processing is time-intensive. With unsupervised learning, the machine learning algorithm classifies a data set by discovering a structure through common elements in the data [16].

Two popular unsupervised learning techniques are as below [16]:

- Clustering technique
- Principal Components technique

Clustering technique

Clustering or cluster analysis is a type of Unsupervised Learning technique used to find commonalities between data elements that are otherwise unlabeled and uncategorized. The goal of clustering is to find distinct groups or "clusters" within a data set. Using a machine language algorithm, the tool creates groups where items in a similar group will, in general, have similar characteristics to each other.

A few of the popular clustering techniques include [16]:

1. K-Means Clustering
2. Hierarchical Clustering

Principle component technique

Analysis Principal components analysis is an Unsupervised Learning technique summarizing a large set of variables and reducing it into a smaller representative variable, called "principal components". The objective of this type of analysis is to identify patterns in data and express their similarities and differences through their correlations [16].

1.1.5 Stage five: presentation and automation

The last step of the data science model is presenting your results and automating the analysis if needed. One goal of a project is to change a process and/or make better decisions. You may still need to convince the business that your findings will indeed change the business process as expected. This is where you can shine in your influencer role. The importance of this step is more apparent in projects on a strategic and tactical level. Certain projects require you to perform the business process over and over again, so automating the project will save time. In reality, one will not progress in a linear way from step 1 to step 5. Often you'll regress and iterate between the different phases [2].

1.2 Samples of machine learning applications

In this chapter, several previous works which have been done in the field of application of machine learning techniques in the industry, are presented.

1.2.1 Machine Learning Framework for Predictive Maintenance in Milling

In the Industry 4.0 era, artificial intelligence is transforming the manufacturing industry. Artificial intelligence has been used in many fields of manufacturing area. Some examples of applying this technology are reducing equipment downtime, spotting production defects, improving the supply chain, and shortening design times by using machine learning technologies which learn from past experiments. One of the uses of these technologies is the development of Predictive Maintenance systems. In order to forecast the precise time of maintenance, industrial IoT (internet of things) will be combined with machine learning techniques.

Predictive Maintenance solution of a milling Cutting-tool solution (including Wear Monitoring), applied to a real milling database as the framework validation, has been discussed in this study. Generally, to improve and optimize the human-machine interaction, a basic framework for creating a tool to control the wear level and preventing the breakdown, has been provided.

Basically, the TCM is composed of elements that can be divided into two categories: a-prior and a-posteriori. The a-prior ones are those which are known at the beginning of the milling activity, such as machine parameters. Conversely, the output of monitoring activities of the machine and specific tool is defined as the a-posterior factors and will be gained by using the sensors. These parameters are selected during the manufacturing processes, and some of them could be the cutting speed, related to the speed of rotation of the tool, the spindle speed, i.e. the rotational frequency of the spindle, the feed rate, that refers to the velocity at which the cutting tool is advanced through the work-piece to remove material, and the depth of cut, that is the penetration of the cutting in the work-piece. The a-prior variables are all related to the factor variable for the ML tool, as they identify a specific case due to the choice of definite circumstances. The a-posteriori factors are time series investigated by ML algorithms to forecast the trend of the wear level during the time.

TABLE 1.1 demonstrates the summary of the application and the proposed framework results for the first regression task and the one about the RUL variable (expected time during which the tool is likely to operate before it needs maintenance and reparations). The best performing algorithm for the regression on V B (the width of the wear band) is the NN regression. This model gives the minimum error values, as well as the highest R2; it means that it can explain a significant percentage of the variance of V B. In the same way, also for the second one regression the best algorithm is the NN regression with a high value for R2. TABLE 1.2 shows the results for the classification task. It shows the best performing algorithm is the Two-Class improved Decision Tree. The most top evaluation metrics, with an accuracy of almost 96%, is demonstrated [31].

For the regression task, the following ML algorithms are trained, tested and evaluated:

- Linear Regression (LR);
- Decision Forest regression (DF);
- Bayesian Linear Regression (BLR);
- Boosted Decision Tree regression (BDT);
- Neural Network regression (NN);

For the classification task, five ML algorithms are trained, tested and compared against each other. Since the classes of the label are two, binary classification algorithms are chosen to train the model. The choices fall for:

- Logistic Regression (LogR);
- Decision Forest (DF);
- Decision Jungle (DJ);
- Boosted Decision Tree (BDT);
- Neural Network (NN).

TABLE 1.1 Results of regression algorithms on V B and RUL [31]

	VB Regression			RUL regression		
Algo.	RMSE	RelSE	R^2	RMSE	RelSE	R^2
LR	0.110	0.182	0.817	1.671	0.178	0.822
DF	0.123	0.225	0.781	1.517	0.134	0.866
BLR	0.116	0.194	0.813	1.640	0.174	0.826
BDT	0.122	0.218	0.794	1.615	0.156	0.844
NN	0.110	0.179	0.821	0.581	0.022	0.979

TABLE 1.2 Results of classification algorithms [31]

Algo.	SP	WP	ACC
LogR	0.960	0.900	0.941
DF	0.936	0.917	0.930
BLR	0.952	0.917	0.941
BDT	0.960	0.950	0.957
NN	0.936	0.900	0.924

1.2.2 Use of machine learning in quality assurance of the fermentation process of black tea

The quality of black tea is directly determined by the fermentation process. This work aimed to describe a rapid way in order to detect the fermentation degree of black tea based on electrical specifications of tea leaves. During the fermentation process, an LCR meter employed to recognize eleven electrical factors of tea leaves, and by using HPLC and UV-Vis spectrometer, respectively, the content of catechins and tea pigments in tea leaves were measured. To classify samples into different classes in the degree of fermentation, principal component analysis, and hierarchical clustering analysis have been used. Correlation analysis applied to characterize the responding strength of electrical parameters on the variation of catechins and pigments. Finally, multilayer perceptron neural network (MLP), support vector machine algorithm and random forest used electrical specification as independent variables, and the HCA clustering results as dependent variables, to build discrimination models of fermentation degree, and the average accuracy rate on the testing set reached to 88.90%, 100%, and 76.92%, respectively [32].

Fermented tea samples can be efficiently grouped in chronological order and fermentation degree by PCA and HCA, based on the changes in catechins and pigments. During the fermentation process of black tea, the high correlation between chemical components (catechins and pigments) and electrical specifications was revealed. Random forest (RF) presented satisfactory prediction accuracy as a pattern recognition method based on "ensemble learning" strategy; it is a good deal with the classification problem in unbalanced and small sample data sets. By the combination of electrical properties of tea leaves and machine learning algorithms to analyze the fermentation degree of black tea, a potential application in the tea industry can be achieved [32].

1.2.3 A supervised machine learning approach to data-driven simulation of resilient supplier selection in digital manufacturing

Companies whose suppliers are prone to disruption risks have a common question to ask. How do firms obtain better performance than others if similar suppliers are affected by disruptions? Recent research hypothesized that some of that success is attributable to the resilient supplier selection and development [33][34].

Selection of resilient suppliers has received a lot of attention in recent years, much of it focusing on predicting the Probability of disorders. Digital manufacturing particularly challenges the supplier selection by the dynamic order allocations and opens new opportunities to extract the digital data to improve decisions related to sourcing. A hybrid technique has been developed in this study by combining simulation and machine learning and examining its applications to data-driven decision-making support in resilient supplier selection. On-time delivery has been defined as an indicator for supplier reliability, and the formation of resilient supply performance profiles surrounded by some sort of conditions, have been evaluated.

The results show that the prediction of delivery reliability can be improved by combining supervised machine learning and simulation (if it is utilized properly). The results of this study advance our conception about how and when simulation and machine learning can be combined to build digital supply chain twins, resilience can be improved through these twins. The proposed data-driven decision-making model for resilient supplier selection can be further exploited for the design of risk mitigation strategies in supply chain disruption management models, re-designing the supplier base or investing in most important and risky suppliers [35].

In TABLE 1.3, a classification sample evaluated through the LR algorithm is shown. The algorithm indicates the probability of each supplier delivering each order based on the time, particularly on the past data. This approach includes an important aspect of human bias avoidance and the potential to support the decision-making process using other quantitative and qualitative approaches. In this paper, the LR performs the supplier selection both singularly and in combination with k-NN. *Rlr*1 and *Rlr*2 represent the results for the first and second suppliers most likely to meet the demand on time according to the LR classifier, respectively. Both pseudo-codes are presented as follows [35].

TABLE 1.3 A classification sample evaluated through the LR algorithm [35]

Order	S_1	S_2	S_3	S_4	R_{lr1}	R_{lr2}
1	0.4361	0.1972	0.4980	0.1887	S_3	S_1
2	0.3319	0.3402	0.6014	0.3642	S_3	S_4
…	…	…	…	…	…	…
454	0.6236	0.2755	0.3248	0.2475	S_1	S_3
455	0.3313	0.9232	0.6015	0.9493	S_4	S_2
…	…	…	…	…	…	…
729	0.3230	0.0730	0.6108	0.0672	S_3	S_1
730	0.4156	0.2006	0.5178	0.1947	S_3	S_1

A set of five different simulation seeds, as presented in Figure 1.13, illustrate the accuracy of the k-NN algorithm in this model. In this study, two predictions are made regarding the k-NN model using (i) the on-time delivery categorization and (ii) the delayed delivery categorization. In other words, the model suggests the supplier most likely to deliver a specific order on time and also the supplier most likely to perform a delayed delivery, respectively. The experiment results suggest that a higher number of suppliers leads to a more resilient system, which can cope with disruptions and recurrent risks [35].

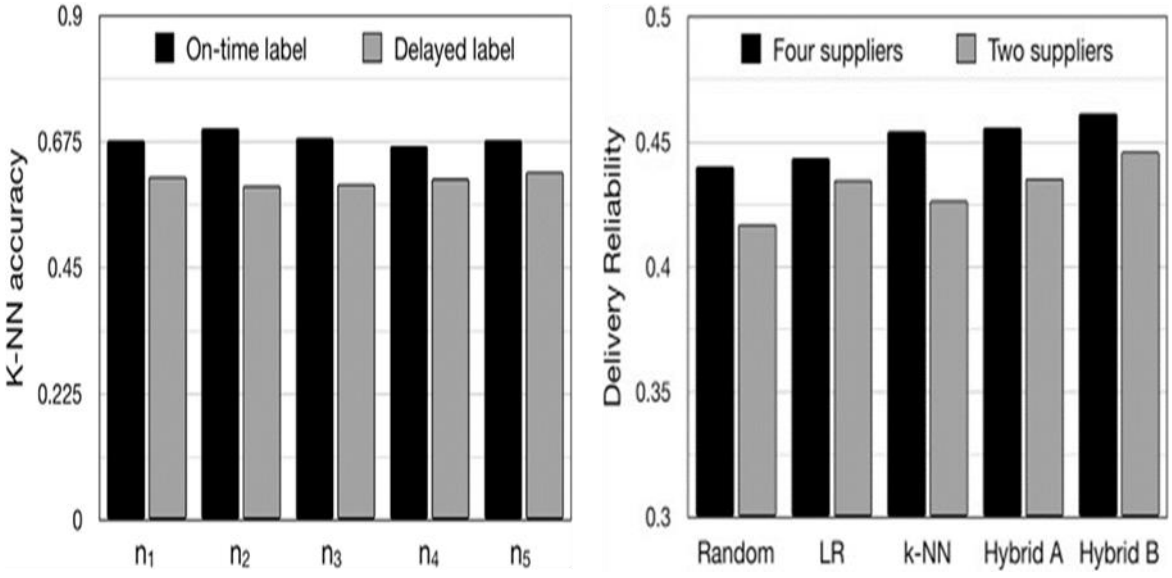

Figure 1.13 The accuracy of the k-NN algorithm [35]

1.2.4 Intelligent traffic control for autonomous vehicle systems based on machine learning

This study aimed to resolve a real-world traffic problem in a large-scale plant. Autonomous vehicle systems (AVSs), which are designed to use multiple vehicles to transfer materials, are widely used to transfer wafers in semiconductor manufacturing. Traffic control is a significant challenge with AVSs because all vehicles must be monitored and controlled in real-time, to cope with uncertainties such as congestion. However, existing traffic control systems, which are primarily designed and controlled by human experts, are insufficient to prevent heavy congestion that impedes production. In this study, we developed a traffic control system based on machine learning predictions and a routing method that dynamically determines AVS routes with reduced congestion rates. We predicted congestion for critical bottleneck areas and utilized the predictions for adaptive routing control of all vehicles to avoid congestion. We conducted an experimental evaluation to compare the predictive performance of four popular algorithms. We performed a simulation study based on data from semiconductor fabrication to demonstrate the utility and superiority of the proposed method. The experimental results showed that AVSs with the proposed approach outperformed the existing approach in terms of delivery time, transfer time, and queuing time. We found that adopting machine learning-based traffic control can enhance the performance of existing AVSs and reduce the burden on the human experts who monitor and control AVSs [36][37].

Traffic data were collected for 626 sections, including two sections of special interest (section IDs: 545, 652), in which frequent traffic congestion was known to occur. First, as shown in Figure 1.14-a, we combined information about the railway network structure and driving logs of vehicles to produce traffic speed and volume data sets. The average values of the three indices measured in intervals of 5, 10, and 15 min were calculated for all sections to construct data sets Figure 1.14-b. Three data sets were constructed for each index with rows and columns representing measured times and sections. The column values of a section of interest were set as the dependent (output) variable, and the remaining column values of the 625 sections were set as independent (input) variables. To predict congestion after one unit of time in a section of interest, we adjusted the data so that we mapped the independent variables and the dependent variable after one unit of time (Figure 1.14-c) [36].

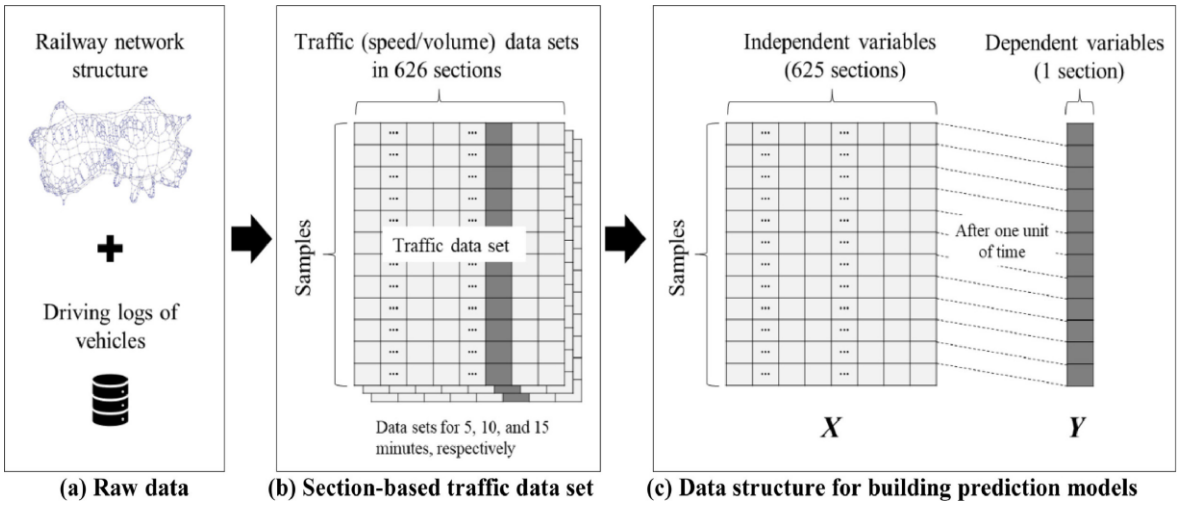

Figure 1.14 Data preparation procedure [36]

We present a machine learning-based predictive model to predict two traffic indexes (speed and volume). To select the final model, we compared the predictive performance of four widely-used models: random forests, support vector machines, gradient boosting machines, and deep neural networks [23][36].

TABLE 1.4 shows the predicted results for 545 and 652 sections in terms of volume and speed. We used three data sets measured at intervals of 5, 10, and 15 min, which are sufficiently short for an effective traffic control system. In practice, short-term predictive models are required to prevent congestion because the vehicle system is vulnerable to a few minutes of congestion [36].

TABLE 1.4 Predicted results for 545 and 652 sections in terms of volume and speed [36]

Section		**545**				**652**			
Index	**Minute**	**RF**	**SVM**	**GBM**	**DNN**	**RF**	**SVM**	**GBM**	**DNN**
Volume	**5**	**0.833**	**0.833**	0.817	0.801	**0.833**	0.726	0.703	0.668
	10	**0.897**	0.894	0.890	0.855	**0.897**	0.855	0.843	0.804
	15	0.906	**0.911**	0.906	0.904	0.888	0.872	**0.892**	0.866
Speed	**5**	**0.919**	0.913	**0.919**	0.889	**0.838**	0.830	0.832	0.767
	10	**0.944**	0.943	0.943	0.943	0.890	0.868	**0.894**	0.821
	15	**0.957**	0.923	0.951	0.941	0.888	0.872	**0.892**	0.866
Average		**0.909**	0.903	0.904	0.889	**0.872**	0.837	0.843	0.799

1.2.5 Neural Networks for Wind Turbine Fault Detection via Current Signature Analysis

Condition monitoring (CM) involves observing the components of a wind turbine to identify changes in operation that can be indicative of a developing fault. It is clear that predicting faults before they occur, through robust CM, should lead to a significant reduction in Operation and Maintenance (O&M) costs [38][39].

This research aims to develop a reliable technique to detect mechanical faults in a WT via the generator current signal. An ANN technique is proposed to automate the fault detection in a variable speed machine. The main purpose of using an ANN is to identify changes in the current signal which have nonstationary characteristics due to the variable speed operating conditions of WTs and to provide online fault detection in advance of catastrophic failures [40].

The data used in this work is based on a WT simulation model. The model is developed and validated with operational data of five 2.5MW turbines recorded by the SCADA system over a period of 1 year. The measured data recorded at 32Hz sampling frequency included wind speed, wind direction, pitch angle, rotational speed and three-phase power output. The model parameters used are detailed in TABLE 1.5 [40].

TABLE 1.5 The model parameters [40]

Parameter	Value
Cut-In, Rated, Cut-Out Wind Speed	3 m/s, 12 m/s, 25 m/s
Rated Tip Speed	80 m/s
Rotor Diameter	90 m
Gearbox Ratio	1:77.4
Line-Line Voltage (RMS)	690V
Frequency	50Hz
Pole Pairs	3
Rated Generator Speed (RPM)	1000

A technique to detect mechanical faults in variable speed WTs via the CSA and ANN is proposed. A framework is discussed for the training of fault detection with simulated signals from faults for later online detection in real WTs. For each set of limited rotational speed variation, a separate ANN will detect the fault [40].

Results of the transient and variable fault detection are presented in Figure 1.15 and Figure 1.16. Although the significant differences between three ANNs trained with the

same input indicate that further optimization of training and algorithm settings might be reasonable, the general fault development is successfully detected. Unsurprisingly, fault degree detection is less accurate than the simple healthy or faulty classification. Regardless, even the rough detection of the strength of a fault enables better monitoring of condition changes [40].

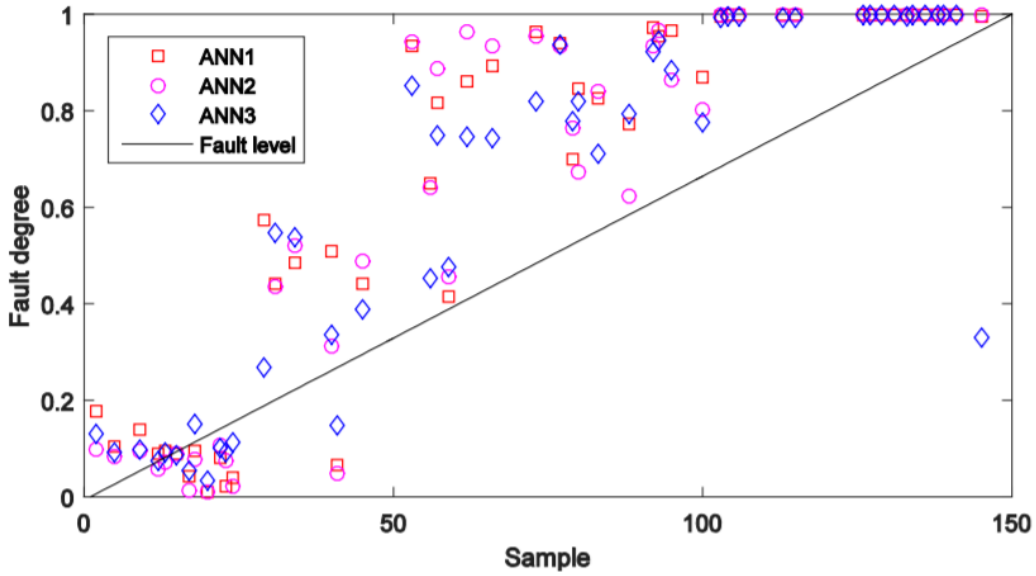

Figure 1.15 ANN fault degree detection of a linear increasing fault [40]

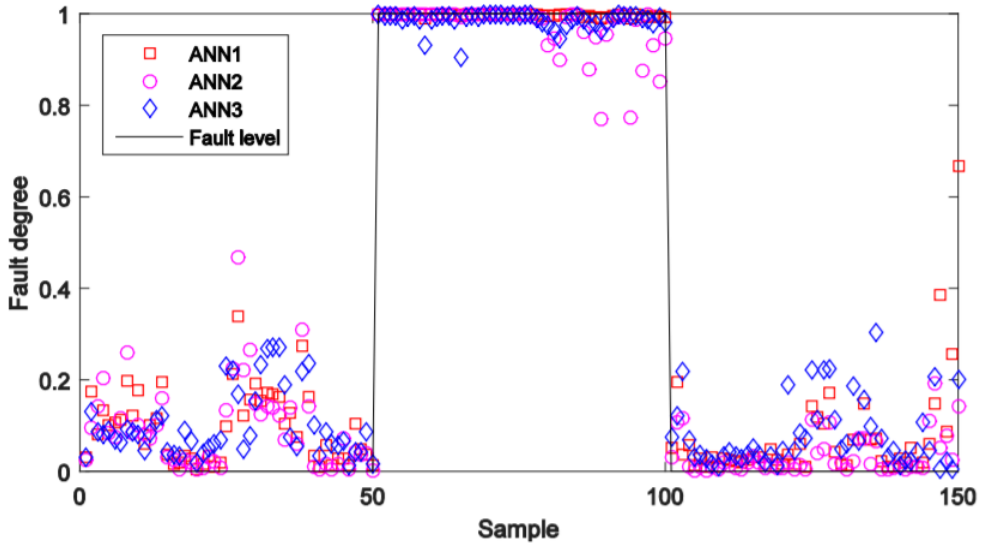

Figure 1.16 ANN fault detection of a transient fault [40]

2. PRACTICAL APPLICATION

Practical application of machine learning methods is utilized for modelling relations between input and output parameters of the ultra large-diameter electromagnet generator of a wind turbine. First, the graphical interface is developed. Next, machine learning methods are implemented for the considered problem. Finally, the results obtained are compared and discussed.

2.1 Development of Graphical User Interface (GUI)

In this chapter, the design of a Graphical User Interface (GUI) to compare the results of different machine learning methods is discussed. One of the objectives to design the GUI is to give the ability to an engineer or a manager with the basic knowledge of programming to visualize the data and be able to compare the results of different machine learning methods. Also, this GUI can be helpful for students in the beginning steps of understanding machine learning concepts and gives them the chance to understand the important parameters which have an influence on the results of each learning method, e.g. the number of neurons in neural network structure.

The GUI is comprised of four tabs: (1) Input, (2) Regression, (3) Random Forest, (4) Neural Network (Figure 2.1).

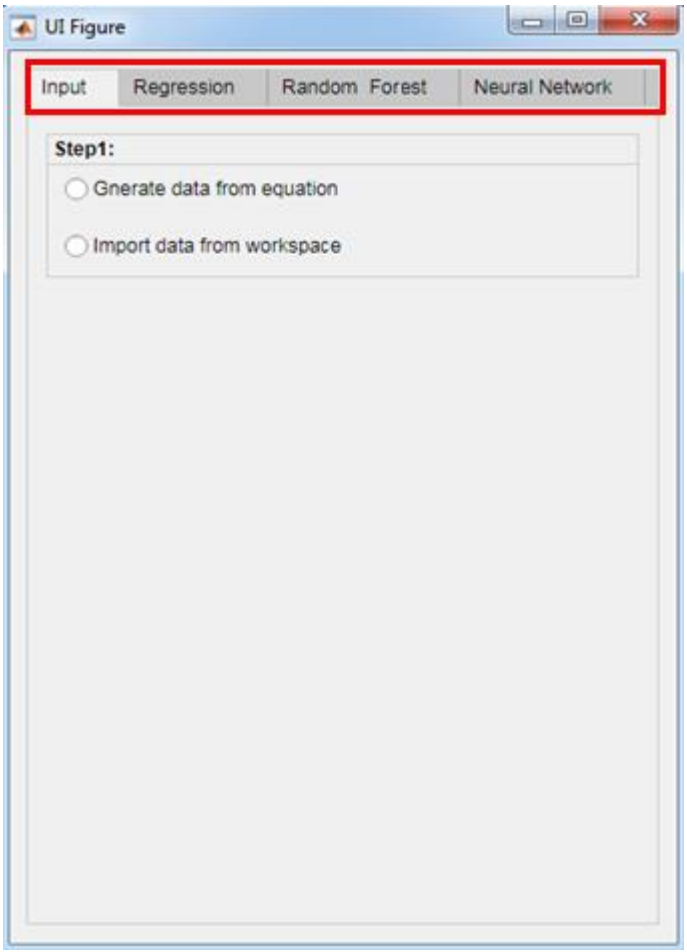

Figure 2.1 Tabs in the GUI

2.1.1 Inputs

The first step in any learning application is to provide the data. For this purpose, there are two options for the user in *Input* tab of GUI: (1) Generating Data using a custom equation, (2) Importing data from MATLAB workspace.

1. Generating data

Generating data using a custom function is added to GUI since, in some application in industry, the model has been investigated and can be presented as an equation with some error. In this case, the user has this opportunity to find which machine learning algorithms work better. Also, in some cases, for students similar to the author, generating data using an equation is helpful to start with the comparison of methods in times of lacking real data.

By selecting *Generate data from equation* button, a new panel named *Step2: Generating data from Equation* will be visible to the user. In this panel, the user should specify some parameters such as Number of Inputs, Number of Levels, beginning and the end range of each input, and Equation (Figure 2.2).

1. Number of Inputs: User should specify the number of variables for generating the data. For example, in the case of having two inputs, the equation should be presented with two variables, x1 and x2.

2. Number of Levels: It determines the number of samples for each input. For example, in case of selecting 5, for an input, e.g. x1 which the beginning and the end of the range are selected 1 to 2, respectively, will generate a vector with 5 elements as follow(2.5):

$$x1 = [1{,}00 \quad 1{,}25 \quad 1{,}50 \quad 1{,}75 \quad 2{,}00] \tag{2.1}$$

3. Beginning and the end of the range: As it is mentioned above, in this section, the user should determine the beginning and the end of each input separately. It worth to mention that the options for the range are only available according to the number of the inputs provided previously in this panel. For example, in case that user selects two inputs, he/she can choose the range only for two variables, and there is no option for selecting the range for third, fourth and fifth inputs.

4. Equation: In this part, the GUI needs the user to enter a valid custom function with the number of variables equal to the number of inputs. A valid custom function in this GUI is a function in which the variables can be x1, x2, x3, x4 and x5.

One example of filling this panel is presented in Figure 2.2

By pressing *Done* button, the program generates the dataset, including inputs and output according to the user's demand. Also, in case of having one or two inputs, the GUI builds a new figure which visualizes the data generated in this stage. Figure 2.3 shows the visualized Inputs versus Output for the above example.

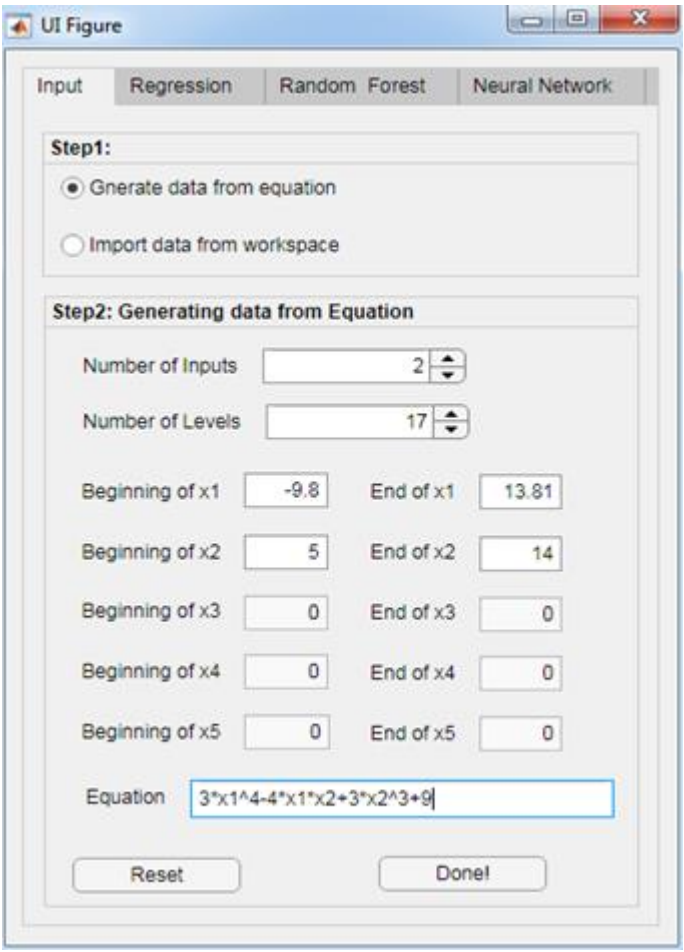

Figure 2.2 Generating data from equation panel

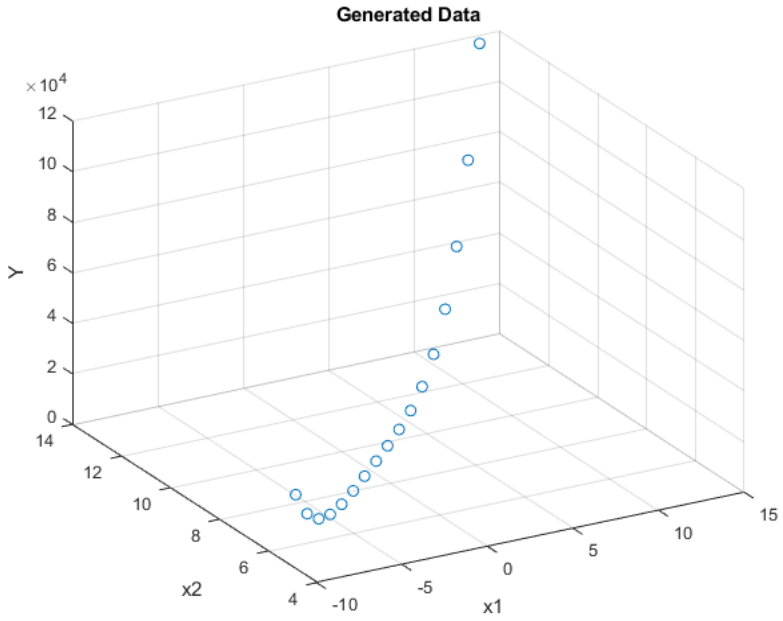

Figure 2.3 Visualization of generated data using an equation

2. Importing data

One of the most important benefits of using machine learning algorithms is to find the relationship between the data, most especially for the cases that this relation might be complex or consume more time for human to do it. Therefore, in most of the applications in industry, the real data is stored, and there is only a need to feed it to a machine learning algorithm and compare the results.

For this purpose, the GUI, allows user to import the data from the Workspace panel of MATLAB. By selecting *import data from workspace* button, a new panel named *Step2: import data from workspace* will be visible to the user. In this panel, the user should choose the inputs and output separately. The options in the Drop Downs are the name of variables in the workspace of MATLAB. Figure 2.4 shows an example of MATLAB workspace as well as the options in the GUI.

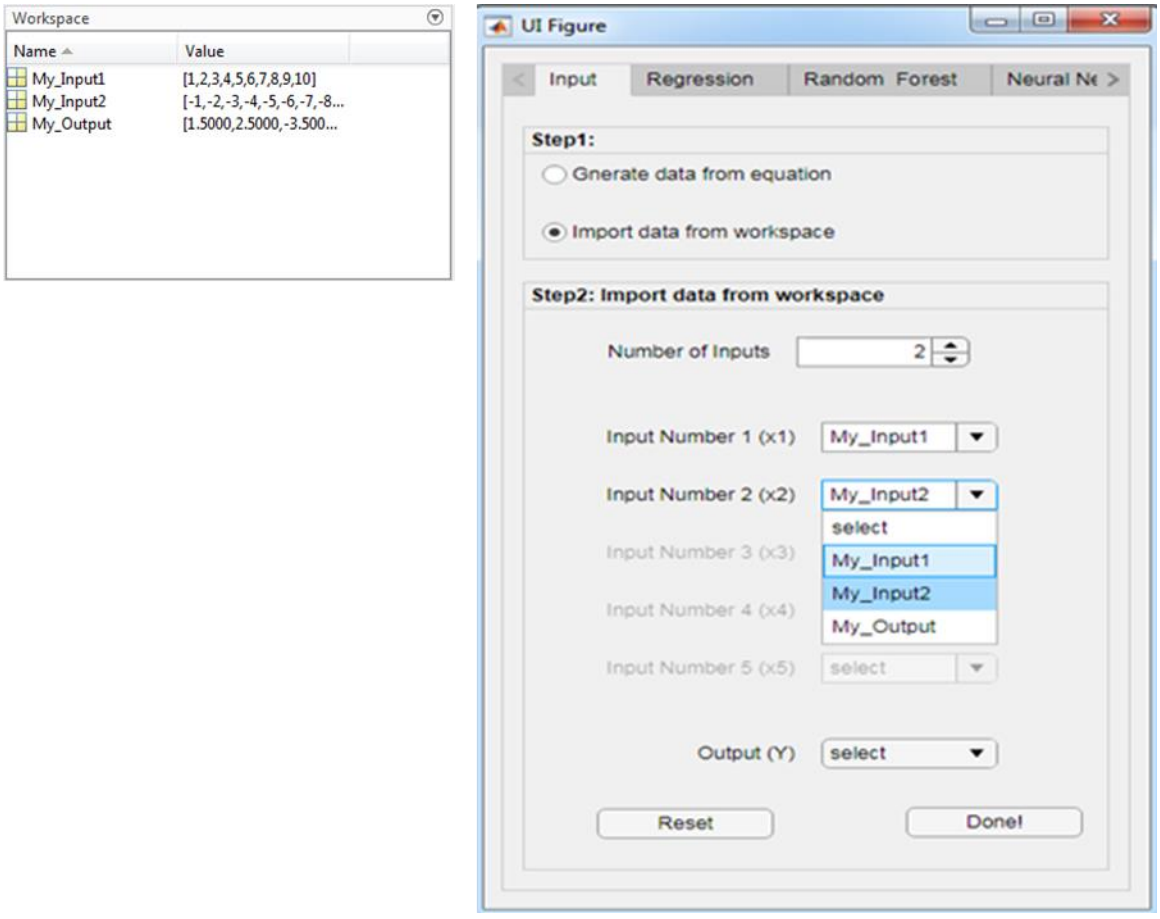

Figure 2.4 Importing data from Workspace, (left): Workspace, (Right): GUI's panel

By pressing *Done* button, the program generates the dataset, including inputs and outputs according to the user's selections. Also, in case of having one or two inputs, the

GUI build a new figure which visualizes the data generated in this stage. Figure 2.5 shows the visualized Inputs versus Output for the above example.

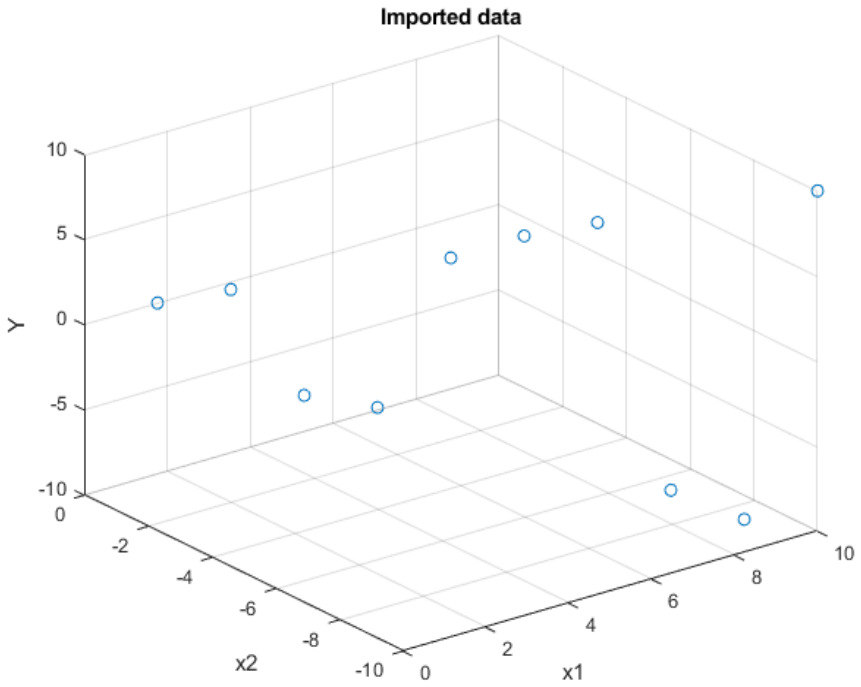

Figure 2.5 Visualization of imported data from MATLAB Workspace

3. My_Example data generation

For a better explanation of machine learning methods used in this GUI, a dataset is introduced and named as My_Example to be used for explaining further tabs.

My_Example has two inputs, X1 and X2, and an output which has shown as Y. The X1 has 100 points in the range of -1 to +1, and X2 in the range of 0 to 1. The equation for generating output (Y), and conditions are as follow (2.2):

$$Y = 3{X_1}^4 - 2{X_2}^3 + {X_1}^2 X_2 + 4X_2 + 5 \qquad (2.2)$$

$$\text{Number of Levels: } 100$$

$$X_1 \in [-1,1]$$

$$X_2 \in [0,1]$$

Using the GUI, the data is visualized and can be seen in Figure 2.6.

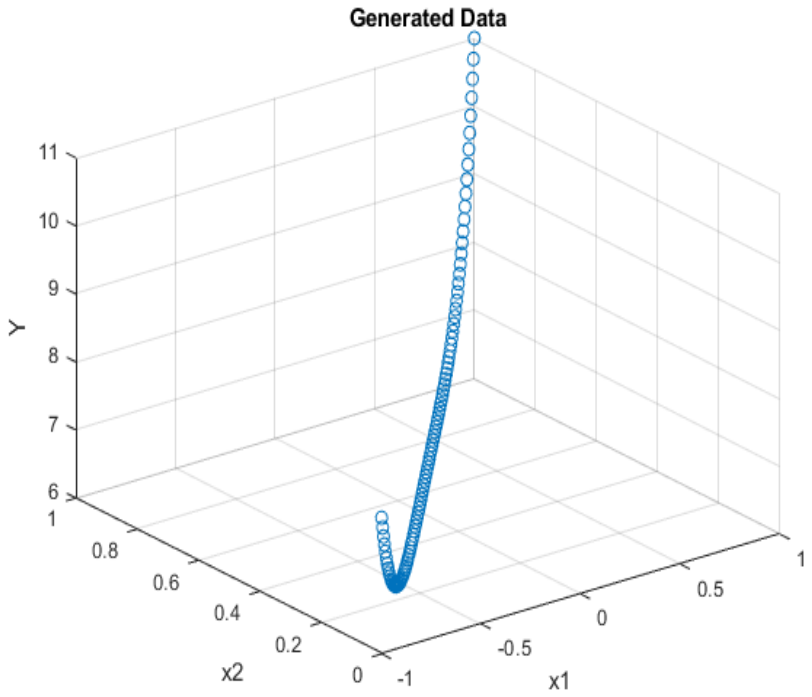

Figure 2.6 My_Example, generated data

Data augmentation

Since in most of machine learning algorithms especially for neural networks, having more data is beneficial, in this GUI, performing Full-Factorial Design is done to increase the number of samples from n to $n \times n$. A full factorial designed experiment consists of all possible combinations of levels for all factors.

In the case of My_Example, with the help of full factorial design, the number of samples increases from 100 to 10000, which is shown in Figure 2.7.

The GUI performs full factorial design on the data in case of generating the data using an equation. Then, the machine learning algorithms in the GUI will be applied to the augmented data with the goal to find the best surface fitted to the data. Also, the accuracy of the methods is calculated based on the Root-Mean-Square Error (RMSE) of the fitted surface.

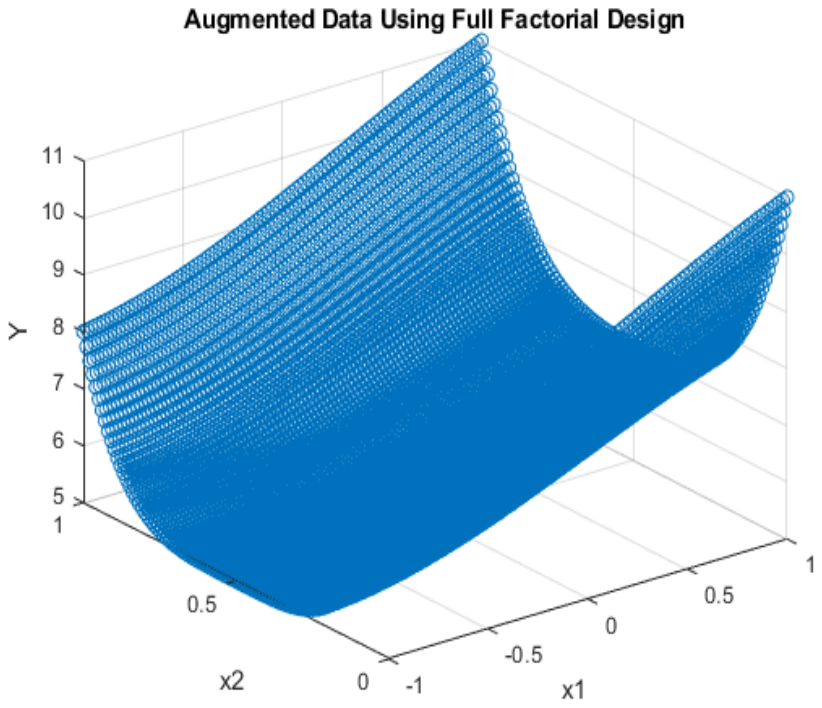

Figure 2.7 Augmented data using full factorial design

2.1.2 Regression

The options in *Regression* tab are available in case of having maximum two inputs. In case of that, after pressing *Done* button in *input* tab, the GUI will automatically change the active tab to *Regression.*

The Regression tab is shown in Figure 2.8, when there are two variables (x1,x2). In this tab, the user can choose the polynomial degree from 1 to 5 for each input separately and compare the results by looking at the Root-Mean-Square Error of each fitted surface. Also, the GUI provides the estimated equation and fitted surface.

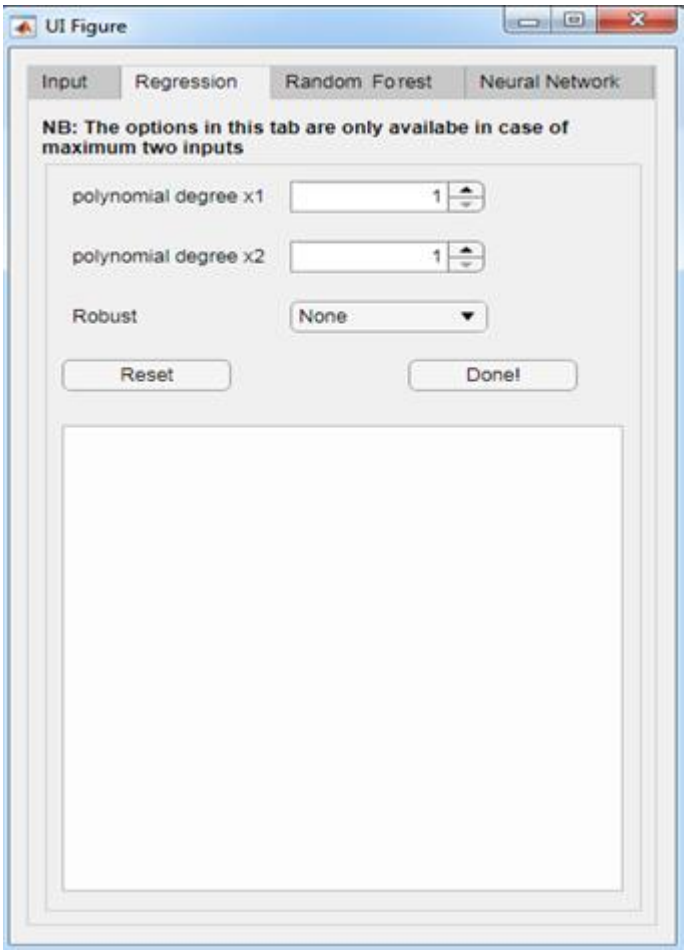

Figure 2.8 Regression tab

My_Example Regression

As it is mentioned above, the goal is to find the best equation which will represent the data, and fit it on the data. Therefore, in the case of My_Example, the equations with different selections of polynomial degree for x1 and x2 is done, and some of the results as an example are presented in TABLE 2.1. The term $polyn_1n_2$ means n_1 and n_2 have been chosen for the polynomial degree of x1 and x2, respectively.

TABLE 2.1 shows poly43 works excellent for My_Example case with RMSE equal to 5,27e-14, and there is no need to make the model more complex with going with poly44.

TABLE 2.1 Regression for My_Example with different polynomial degrees

Equation Name	Estimated Equation	RMSE
poly23	$Y = 2.66{X_1}^2 - 2{X_2}^2 + {X_1}^2X_2 + 4X_2 + 4.37$	0,24
poly32	$Y = 2.62{X_1}^2 - 3{X_2}^2 + {X_1}^2X_2 + 5.19X_2 + 4.63$	0,24
poly33	$Y = -2{X_2}^3 + 2.62{X_1}^2 + {X_1}^2X_2 + 4X_2 + 4.73$	0,23
poly34	$Y = -2{X_2}^3 + 2.62{X_1}^2 + {X_1}^2X_2 + 4X_2 + 4.73$	0,23
poly43	$Y = 3{X_1}^4 - 2{X_2}^3 + {X_1}^2X_2 + 4X_2 + 5$	5,27e-14
poly44	$Y = 3{X_1}^4 - 2{X_2}^3 + {X_1}^2X_2 + 4X_2 + 5$	5,25e-14

Also, the GUI fits the estimated surface on the data. The results are shown for poly32 and poly43 as two examples in Figure 2.9 and Figure 2.10, respectively.

As it can be seen in Figure 2.9, the surface fitted on the data has some error, in contrast with poly43 which the surface perfectly has been fitted on the data.

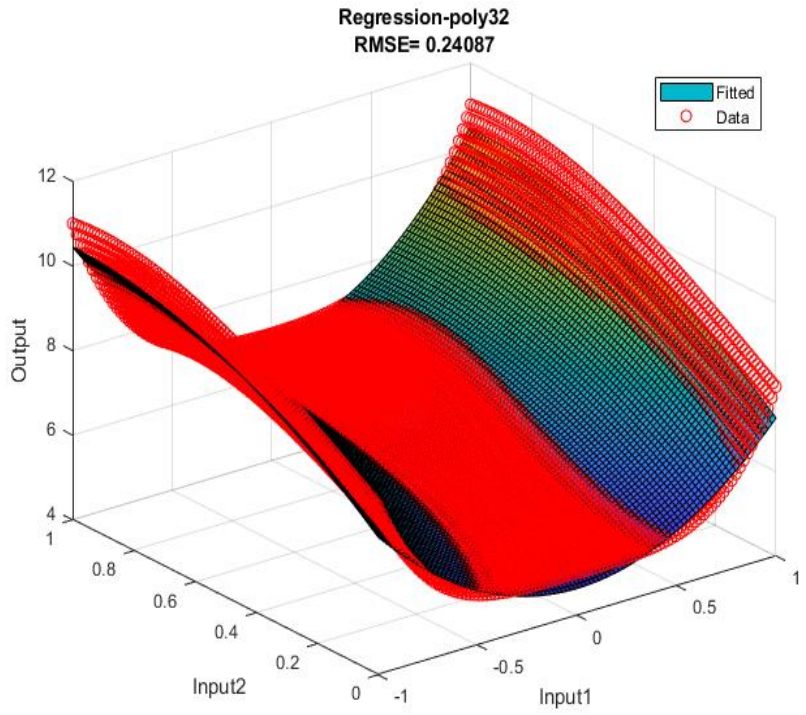

Figure 2.9 Regression, Poly32

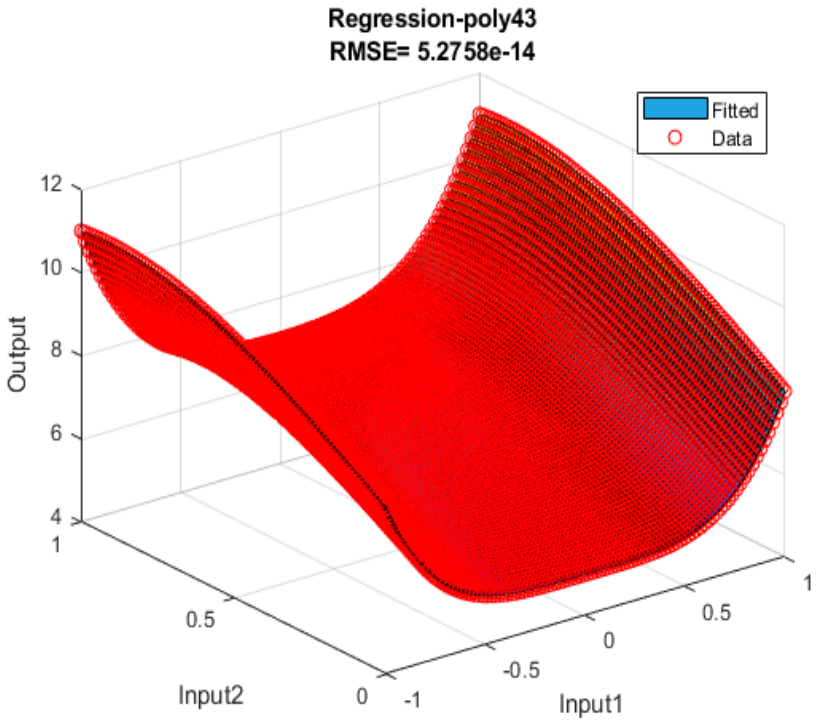

Figure 2.10 Regression, Poly43

2.1.3 Random Forest (RF)

The options in *Random Forest* tab are available in any cases with the number of variables between one to five. However, the visualization of the data is only possible in case of having maximum of two inputs.

Unlike the regression method, the primary purpose of Random Forest and Neural Network is not to find necessarily the equation which represents the data. The main goal is to build a model which represents the data in the best way, and in the next step, to use this model to predict the results in the case of new data. However, the same as Regression, Root-Mean-Square Error (RMSE) is a good criterion to evaluate the results of both Random Forest and Neural Network models.

The *Random Forest* tab is shown in Figure 2.11 when there are two variables (x1,x2). In this tab, the user can choose the *Number of the bags* and compare the results by looking at the RMSE of each fitted surface. Also, the GUI provides the visualized data and fitted surface.

It is worth to mention; there are some other factors than the number of the bags which may influence on the performance of Random Forest; however, it was tried to keep the GUI simple enough that as a person who is new to machine learning area can easily understand the concepts of each method and the most import factors.

Figure 2.11 Random forest tab

My_Example random forest

The results of Random Forest on My_Example dataset with the different number of bags is presented in TABLE 2.2.

TABLE 2.2 shows increasing the number of bags not necessarily cause decreasing the error. This fact can be seen in My_Example dataset when the number of bags increases from 100 to 200. Also, increasing the number of bags uses more memory in the PC, which causes a lower speed. Since there is no formula to determine the best number of bags for Random Forest, it is suggested that user run the Random Forest several times and choose the best model according to both RMSE and the time consumption.

TABLE 2.2 Random Forest for My_Example with different number of bags

The number of bags	RMSE
5	0,11
10	0,08
20	0,07
50	0,07
100	0,05
200	0,06

Also, the GUI fits the estimated surface extracted from the model on the data. The results are shown for 5 and 100 number of bags as two examples in Figure 2.12 and Figure 2.13, respectively.

As it can be seen in Figure 2.12, the surface fitted on the data has some error, which in Figure 2.13, this error decreases dramatically.

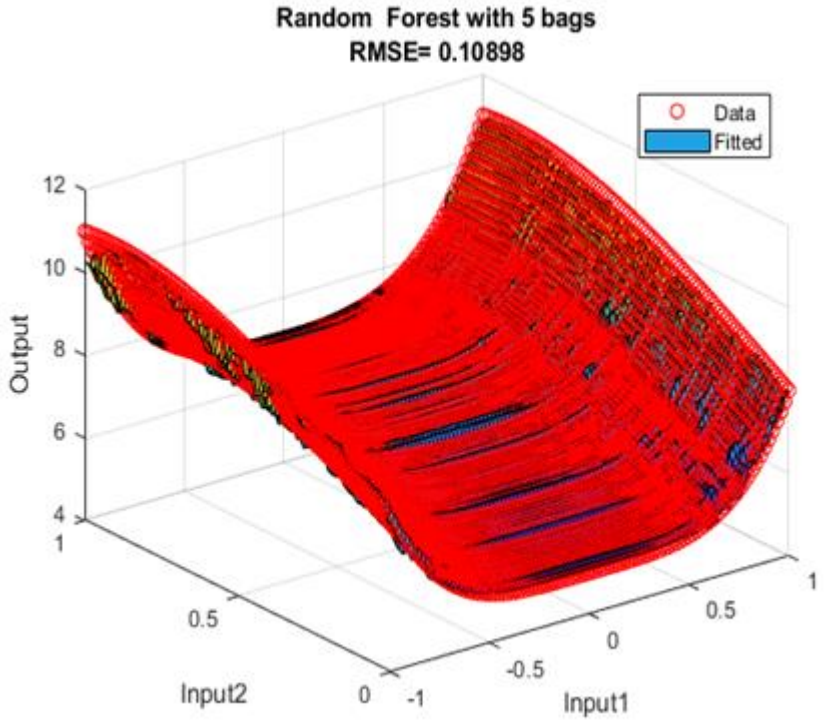

Figure 2.12 Random Forest with 5 bags

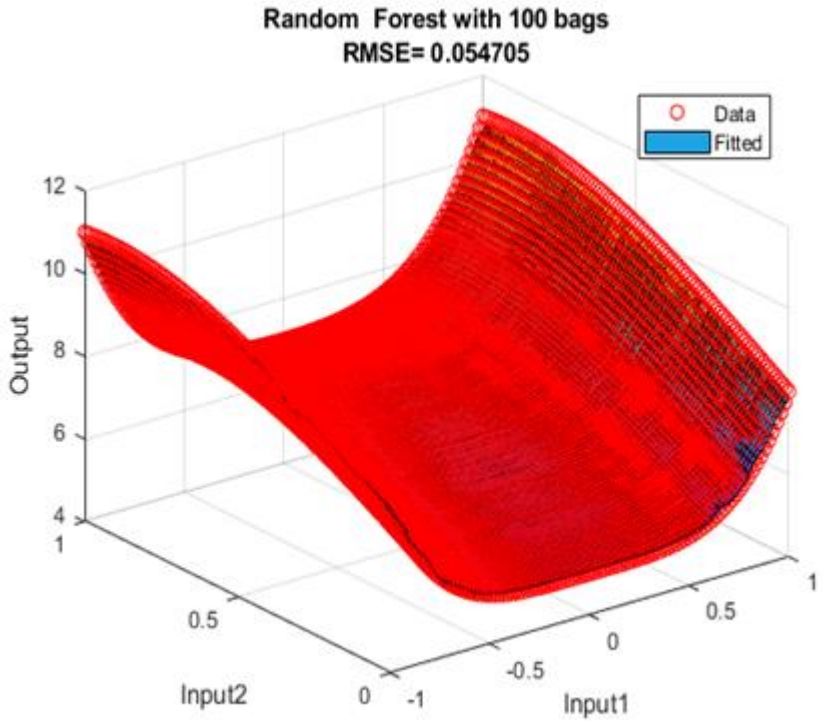

Figure 2.13 Random Forest with 100 bags

2.1.4 Neural Network (NN)

Similar to *Random Forest*, the options in *Neural Network* tab are available in any cases with the number of variables between one to five. However, the visualization of the data is only possible in case of having maximum of two inputs.

As it is mentioned in Random Forest as well, unlike Regression, the main goal of both Random Forest and Neural Network is to build a model which represents the data in the best way, and in the next step, to use this model to predict the results in case of new data. However, the same as Regression, Root-Mean-Square Error (RMSE) is a good criterion to evaluate the results of both Random Forest and Neural Network models.

Herein feed-forward back-propagation neural network with 1 hidden layer is utilized since it is widely known as a satisfactory choice for solving function approximation problems [1]. The *Neural Network* tab is shown in Figure 2.14, when there are two variables (x1,x2). In this tab, the user can choose the *Number of Neurons* and *Learning Rate,* and compare the results by looking at the RMSE of each fitted surface. Also, the GUI provides the estimated equation and fitted surface.

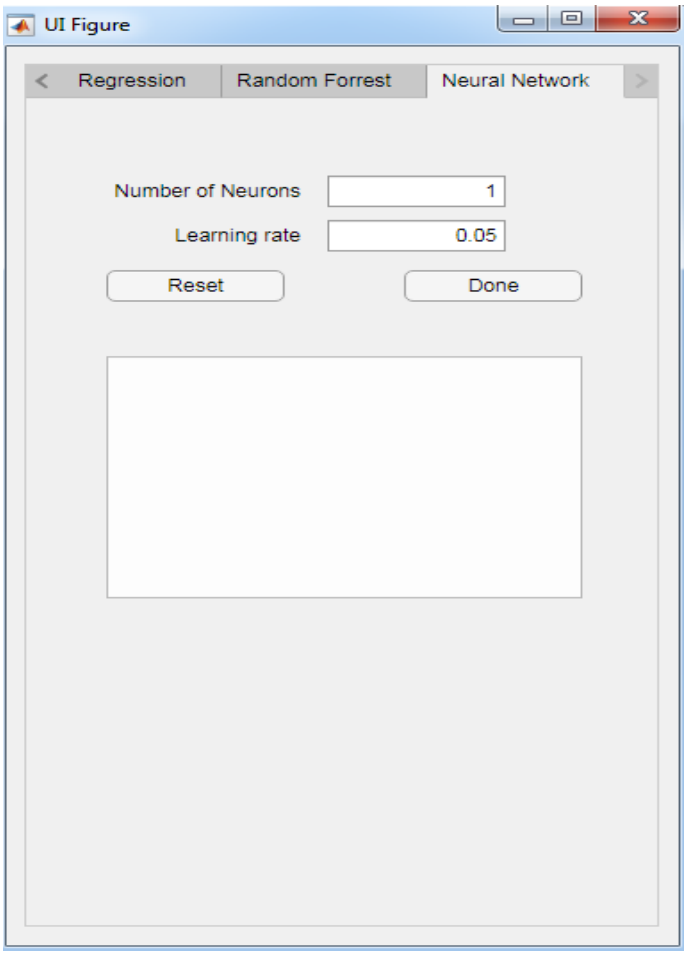

Figure 2.14 Neural Network tab

It is important to know; there are some other factors than the number of neurons which may influence the performance of Neural Network such as the number of the hidden layer, activation functions, etc. However, it was tried to keep the GUI simple enough that as a person who is new to machine learning area can easily understand the concepts of each method and the most import factors. Therefore, in this case, the options for changing the number of neurons as well as learning rate are provided for the user.

The GUI uses the neural network structure provided by MATLAB for fitting applications. The structure has n number of inputs, one hidden layer with m neurons which will be determined by the user in *Neural Network* panel and 'tansig' as its activation function and one output layer with one neuron and 'purelin' as the activation function and one output (Y). This structure is shown in Figure 2.15.

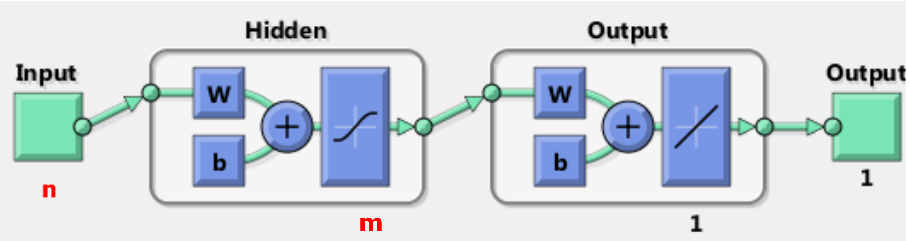

Figure 2.15 Neural Network structure

Also, it should be mentioned that the training algorithm which is used in the GUI is Levenberg-Marquardt backpropagation ('trainlm') and 70% of data is used for Training step, and other 30% is divided between validation and test steps equally.

My_Example Neural Network

The results of the Neural Network on My_Example dataset with the different number of the neurons and learning rate is presented in TABLE 2.3.

TABLE 2.3 RMSE of Neural Network for My_Example with different number of neurons and learning rates

		Number of neurons				
		2	**3**	**5**	**10**	**50**
Leaning rate	**0,04**	0,34	0,05	3,24e-3	6,60e-04	1.93e-05
	0,05	0,34	0,05	3,21e-3	6,12e-04	5.55e-05
	0,06	0,34	0,05	2,34e-3	2,72e-04	8.31e-05
	0,07	0,34	0,05	1,92e-3	6,28e-04	1.47e-05

It is important to know that increasing the number of neurons not necessarily cause decreasing the error in test step (new data). This fact can be seen in smaller datasets when with increasing the neurons, neural networks try to build a complex model which follow all the noises in the dataset and not necessarily works fine with a new dataset. This phenomenon is known as overfitting.

Also, increasing the number of neurons uses more memory in the PC, which causes a lower speed. In My_Example, although the training step with 5 neurons is done in less than 5 seconds, this step took more than one minutes for 50 neurons. These experiments are performed in a PC with Windows 7 as the operating system, with 4GB RAM and Intel(R) Core (TM) i5, 2,67 GHz as the processor.

Also, from TABLE 2.3, it can be interpreted that changing the learning rate did not have a significant influence on the error (RMSE). The reason behind it is because that the learning rate has mainly influence on convergence speed/ computing time. The result of this experiment with the fixed number of neurons (10) and changing the learning rate to measure the elapsed time for learning procedure can be seen in TABLE 2.4. From this table, it can be understood that in general, increasing the learning rate cause decreasing the elapsed time. However, the user should keep in mind a large number (close to 1) for the learning rate might influence the error, and may disrupt the optimization process.

TABLE 2.4 Elapsed time of learning in Neural Network for My_Example with 10 neurons and different learning rates

Learning rate	0,04	0,05	0,06	0,07
Time elapsed (*s*)	19,684	16,035	15,980	15,450

Since there is no formula to determine the best number of neurons and learning rate for Neural Network; it is suggested that user run the Neural Network several times and choose the best model according to both RMSE and the time consumption.

Also, the GUI fits the estimated surface extracted from the model on the data. The results are shown for 2 and 50 number of neurons with learning rate equal to 0.04 and 0.06, respectively (the worst and the best results) as two examples in Figure 2.16 and Figure 2.17, respectively.

As it can be seen in Figure 2.16, the surface fitted on the data has some error, which in Figure 2.17, this error decreases dramatically, and the surface fitted perfectly on the data.

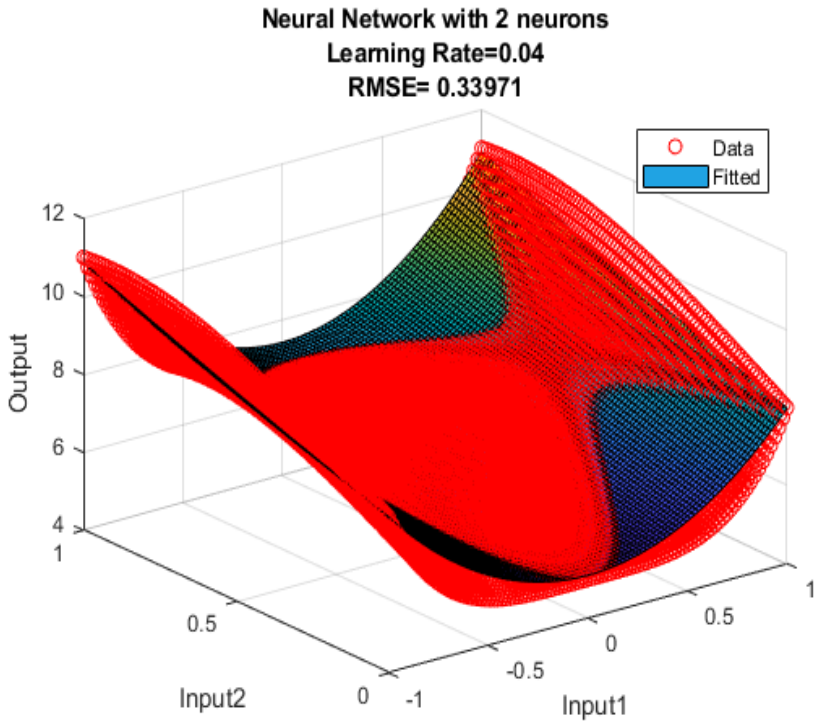

Figure 2.16 Neural Network with 2 neurons and 0,04 as the learning rate

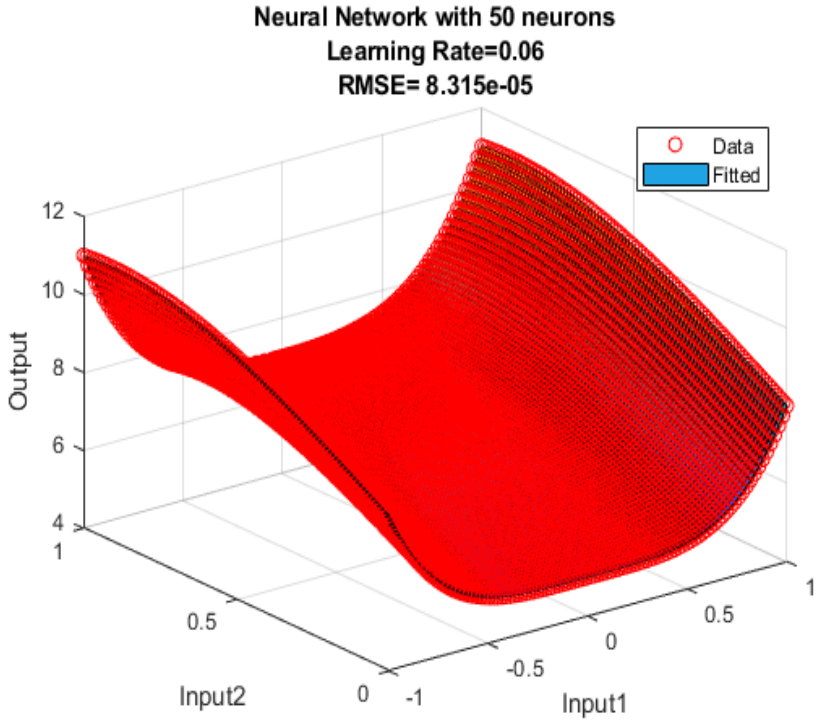

Figure 2.17 Neural Network with 50 neurons and 0,06 as the learning rate

2.1.5 Conclusion

In this chapter, the design of a GUI for comparing Regression, Random Forest, and Neural Network results for a dataset is discussed.

With the help of this GUI, engineers and managers with the basic knowledge in the field of machine learning and programming are able to see and compare the results of different algorithms.

In the first step in this GUI, the user should provide the data using one of two options: (1) generating data using a custom function, (2) importing data from MATLAB Workspace. Then, by determining some import parameters for each method (e.g. for regression: polynomial degree, random forest: the number of bags, neural network: the number of neurons in the hidden layer), GUI is able to implement each method on the given dataset.

Finally, the user is able to see the results in the visualized format, and decide which algorithms work better for him/her according to the RMSE of each fitted curve or surface on the data, as well as the time consumed by the computer.

2.2 Implementation of machine learning techniques

The aim of this chapter is a comparison of different machine learning techniques to demonstrate which ML method is the best for fitting the data and predict the pattern of the dataset. The criteria for choosing the technique with the best performance is Root-Mean-Square Error (RMSE) (2.3) [41].

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)} \tag{2.3}$$

In (2.3), the parameters y_k ,$\hat{y}_k$ and N refers to the predicted values , observed values and the number of data, respectively [41].

2.2.1 Data

With the growth of wind energy, the relative price of the active components has steadily declined, and at higher turbine capacities cost composition is dominated by the mechanical part [42][43]. Therefore, a more structured approach, in connection with novel technologies and optimization procedures tailored for this particular problem, is necessary to reduce the cost of direct-drive generators and price tag of the produced energy. Known methods employed in other industries have already been used at a small scale to aid in the process of selecting initial design parameters. Isfahani et al. [44] have utilized a genetic algorithm to optimize the electromagnetic layout of a direct-drive permanent magnet generator. The same method is used by Li and Chen [45] to match generators with different power ratings and rotational speeds to proposed wind turbine sites.

For comparison of the different machine learning techniques, a data set related to an ultra large-diameter electromagnet generator of a wind turbine has been provided by O. PABUT [1]. PABUT study aimed to optimize the mechanical configuration of the electromagnet generator by finding the influential variables for minimizing the stress and deformation of the stator and rotor of the generator. His study focuses on the development process of an electrical generator for wind turbine applications in order to optimise the machine layout in terms of maximum energy yield and minimal system cost [1]. The data are collected from *Goliath Wind OÜ* company.

O.PABUT selected (1) air-gap (2) magnet thickness (3) magnet length, and (4) coil thickness as variables (inputs), and (1, 2) maximum rotor and stator deformations ε_{rot}^{max}, ε_{stat}^{max} and (3, 4) stresses σ_{rot}^{max}, σ_{stat}^{max}, characterizing stiffness and strength of the structure as the objectives (outputs) [1].

To implement an optimization method for the mentioned objective, estimating the relation between the input variables and output (fitness function) is the first step. Therefore, the scope of the current study is to compare different machine learning algorithms results and find the best approach, which represents the pattern of the data.

Although the parameters are explained in detail in O.PABUT work, a brief explanation of them are as follow:

1. Inputs-variables

There are four inputs for this case study: (1) air-gap (2) magnet thickness (3) magnet length, and (4) coil thickness as variables.

Air-gap

The wind turbine generator includes a core and a plurality of stator windings circumferentially spaced about a generator longitudinal axis, a rotor rotatable about the generator longitudinal axis wherein the rotor includes a plurality of magnetic elements coupled to a radially outer periphery of the rotor such that an air-gap is defined between the stator windings and the magnetic elements and the plurality of magnetic elements including a radially inner periphery having a first diameter (Figure 2.18) [46]. The generator air-gap diameter needs to be achieved with high accuracy to avoid the rotor-stator collision, unbalanced magnetic forces and mass distribution [1].

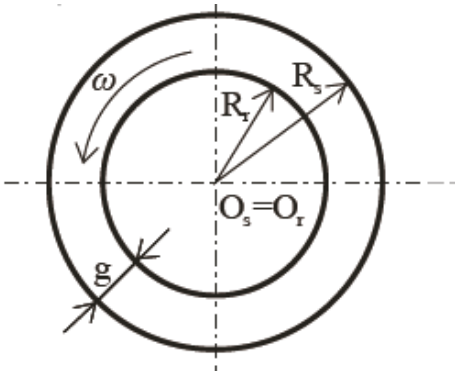

Figure 2.18 Generator eccentricity, O_r and O_s are rotor and stator symmetry centers, g is the width of the air-gap.

Magnet

In every wind turbine and generator, you will find one or more incredibly strong magnets. Simplified, the rotating shaft of a wind turbine is connected to one or more strong magnets, usually neodymium magnets, these magnets turn relative to an assembly of coiled wire, generating voltage in the coil [47].

Coil

An electromagnet uses a rotating coil of wire with a current running through it to induce a magnetic field in the iron bar that is placed in the middle of the coil [47].

2. Outputs-objectives

The optimization problem considered in [1] is a complex multicriteria optimization problem involving four objectives (ε_{rot}^{max}, ε_{stat}^{max}, σ_{rot}^{max}, σ_{stat}^{max}). In order to simplify the solution, the objectives considered are combined into one objective by applying the compromise programming technique (2.4). Latter technique is applicable for non-conflicting objectives. This requirement is checked by performing a pairwise analysis.

$$Y = \left[\sum_{i=1}^{4} (w_{SRi} f_{SRi})^c\right]^{1/c} \tag{2.4}$$

In (2.4) the parameter $c \geq 1$ $i = 1, \ldots, n$ and in the case of $c = 1$, the compromise programming technique reduces to weighted summation technique. The objectives $f_{SR1}(\bar{x})$,..., $f_{SRn}(\bar{x})$ describe normalised structural response of the construction (maximum rotor and stator deformations ε_{rot}^{max}, ε_{stat}^{max} and stresses σ_{rot}^{max}, σ_{stat}^{max}, characterizing stiffness and strength of the structure). The weight coefficients of the structural response characteristics w_{SRi} are given constants, providing that sum of these constants is equal to one.

The simplification performed allows to consider in the current study one output Y given by (2.4).

2.2.2 Applying machine learning techniques

In this case study, to find the related function between four inputs and one output, three machine learning techniques are used:

1. Multivariate Linear Regression (MV Regression)
2. Random Forest (RF)
3. Artificial Neural Network (ANN)

1. Multivariate linear regression

As discussed previously in this chapter, in this dataset, the number of inputs and output is equal to four and one, respectively. Since there are more than two inputs, there is a need to use a multivariate regression. For simplicity purposes, the variables are assumed to be independent, and their relation to output is linear.

Therefore, the goal in this section is to estimate the coefficient vector ($\beta = [\beta_0, \beta_1, \beta_2, \beta_3, \beta_4]$) in the following equation (2.5), where Y is the output, and X_i are the inputs (X_1: air-gap (mm), X_2: magnet thickness (mm), X_3: magnet length (mm), and X_4: coil thickness (mm).

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 \tag{2.5}$$

By using *regress* function in MATLAB, the result of multivariate linear regression is as follow (2.6):

$$\beta = [0{,}0957 \quad -0{,}0749 \quad +0{,}0660 \quad +0{,}1058 \quad -0{,}0621] \tag{2.6}$$

Now it is possible to write the relation between variables as an equation (2.7):

$$Y = 0{,}0957 - 0{,}0749 X_1 + 0{,}0660 X_2 + 0{,}1058 X_3 - 0{,}0621 X_4 \tag{2.7}$$

Since, in this case, there are more than two inputs, the best way to visualize the result, is to put real output versus predicted output in a figure (Figure 2.19). Also, to evaluate the accuracy of the method, calculating RMSE between real and predicted output is used.

As it can be interpreted from Figure 2.19, The closer the data is to the black line, the lower the error rate. For multivariate linear regression, the RMSE was 0,014, and the data gathered around the line well.

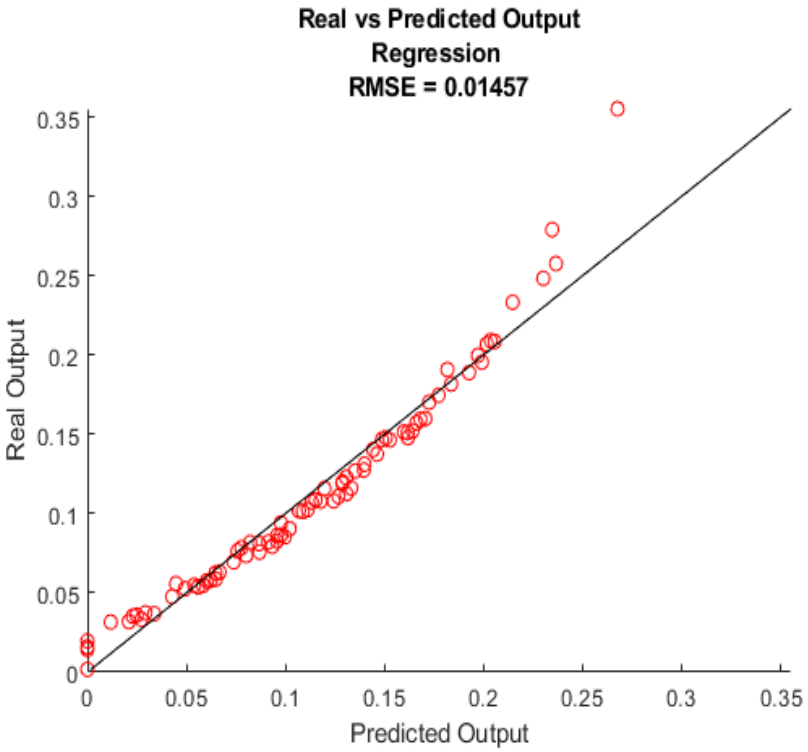

Figure 2.19 Multivariate linear regression for case study

2. Random Forest

Unlike the regression method, the primary purpose of Random Forest and Neural Network is not to necessarily find the equation which represents the data. The main goal is to build a model which represents the data in the best way, and in the next step, to use this model to predict the results in the case of new data. However, the same as Regression, Root-Mean-Square Error (RMSE) is a good criterion to evaluate the results of both Random Forest and Neural Network models.

In this case study, finding the relation between four inputs (X_1: air-gap (mm), X_2: magnet thickness (mm), X_3: magnet length (mm), and X_4: coil thickness (mm), and Y as the output is the main goal.

Random forest experiments are done with a different number of bags, and the result of this case study are shown in TABLE 2.5.

TABLE 2.5 Random Forest for the case study with different number of bags

The number of bags	RMSE
5	0,033
10	0,025
20	0,026
50	0,026
100	0,026
200	0,025

As discussed before, increasing the number of bags not necessarily cause decreasing the error. This fact can be seen in this dataset when the number of bags increases from 10 to 20. Also, increasing the number of bags uses more memory in the PC, which causes a lower speed. Since there is no formula to determine the best number of bags for Random Forest, it is suggested that user run the Random Forest several times and choose the best model according to both RMSE and the time consumption.

From TABLE 2.5, it can be interpreted that the best results among random forest experiments belong to a tree with 10 bags, and increasing the number of bags to 200 to achieve the same result seems not logical.

Furthermore, the visualized result (real versus predicted output) for two random forest experiments as an example are shown in Figure 2.20 and Figure 2.21.

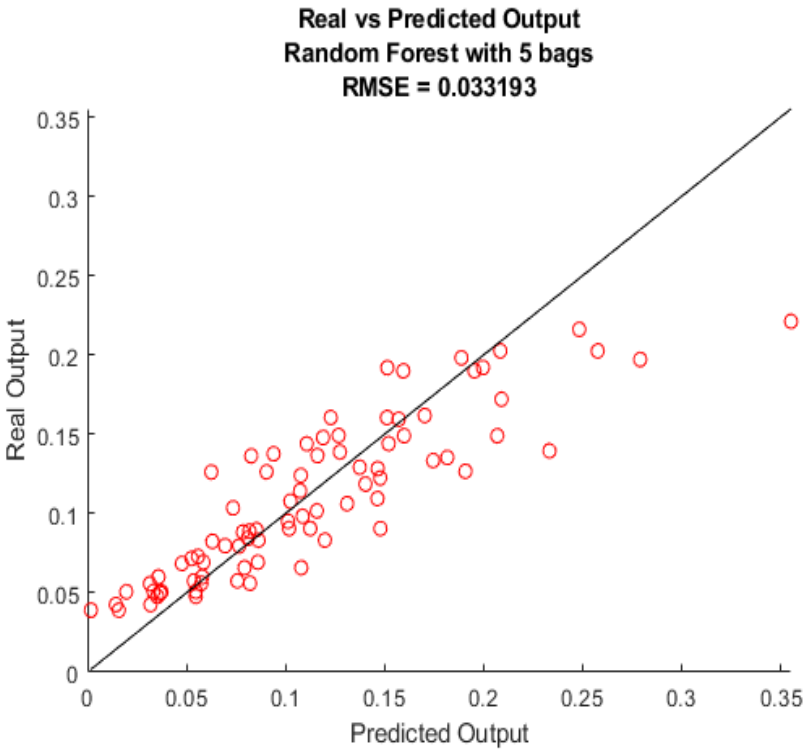

Figure 2.20 Random forest for the case study with 5 bags

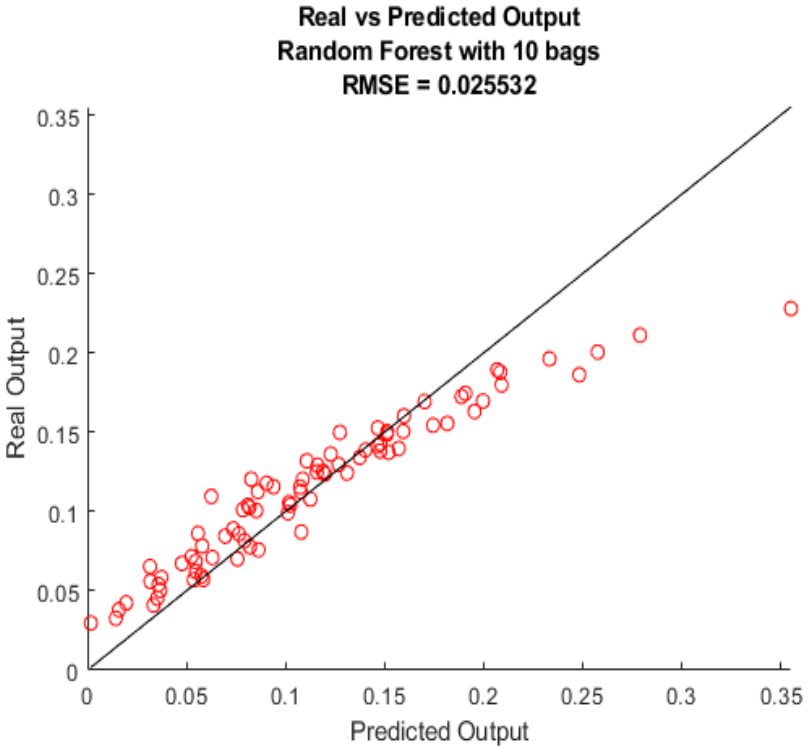

Figure 2.21 Random forest for the case study with 10 bags

As can be seen, data are scattered around the line with more density in Figure 2.21 than Figure 2.20, which shows less error in case of 10 bags.

3. Neural Network

Similar to the random forest, in the neural network, unlike Regression, the main goal is to build a model which represents the data in the best way, and in the next step, to use this model to predict the results in case of new data. However, the same as Regression, Root-Mean-Square Error (RMSE) is a good criterion to evaluate the results of both Random Forest and Neural Network models.

As it is mentioned beforehand in this case study, finding the relation between four inputs (X_1: air-gap (mm), X_2: magnet thickness (mm), X_3: magnet length (mm), and X_4: coil thickness (mm), and Y as the output is the main goal.

As a rule, the neural network needs fine-tuning to perform efficiently, i.e. the number of layers, number of neurons in the hidden layer, learning rate, number of epochs etc. should be determined for the particular problem considered. However, in this case, for simplification, it seems enough to only consider the primary parameters such as the number of neurons and learning rate.

Neural network experiments are done with the different number of neurons in the hidden layer, and different learning rates and the result for this case study are shown in TABLE 2.6. It worth to mention the neural network structure is the same as the structure used for designing the Graphical User Interface (Figure 2.15).

TABLE 2.6 RMSE for Neural Network for the case study with a different number of neurons and learning rates

		Number of neurons					
		2	**3**	**5**	**10**	**20**	**50**
Leaning rate	**0,04**	7,94e-3	6,21e-3	1,35e-3	5,78e-3	0,01	0.08
	0,05	3,57e-3	8,42e-3	2,00e-3	1,50e-3	0,02	0.04
	0,06	5,92e-3	5,38e-3	2,16e-3	3,89e-3	0,02	0.05
	0,07	5,31e-3	4,72e-3	2,46e-3	5,71e-3	0,02	0.06

It is important to know that increasing the number of neurons not necessarily cause decreasing the error in test step (new data). This fact can be seen in this dataset with increasing the number of neurons from 10 to 20, and 20 to 50. With increasing the neurons, neural networks try to build a complex model which follow all the noises in the dataset and not necessarily works fine with a new dataset. This phenomenon is known as overfitting. Also, increasing the number of neurons uses more memory in the PC,

which causes a lower speed. In this case, although the training step with 5 neurons is done in less than 5 seconds, this step took more than 30 seconds for 50 neurons. These experiments are performed in a PC with Windows 7 as the operating system, with 4GB RAM and Intel(R) Core (TM) i5, 2,67 GHz as the processor.

From TABLE 2.6, it can be interpreted that the best results among neural network experiments belong to a network with 10 neurons, and increasing the number of neurons to 20, and 20 to 50 caused overfitting.

Also, from TABLE 2.6, it can be interpreted that changing the learning rate did not have a significant influence on the error (RMSE). The reason behind it is because that the learning rate has mainly influence on convergence speed/ computing time. The result of this experiment with the fixed number of neurons (10) and changing the learning rate to measure the elapsed time for learning procedure can be seen in TABLE 2.7. From this table, it can be understood that in general, increasing the learning rate cause decreasing the elapsed time. However, the user should keep in mind a large number (close to 1) for the learning rate might influence the error, and may disrupt the optimization process.

TABLE 2.7 Elapsed time of learning in Neural Network for the case study with 10 neurons and different learning rates

Learning rate	0,04	0,05	0,06	0,07
Time elapsed (s)	0,290	0,250	0,243	0,225

Since there is no formula to determine the best number of neurons for Neural Network, it is suggested that user run the Neural Network several times and choose the best model according to both RMSE and the time consumption.

Furthermore, the visualized result (real versus predicted output) for two neural network experiments (worst and the best cases) as an example are shown in Figure 2.22 and Figure 2.23.

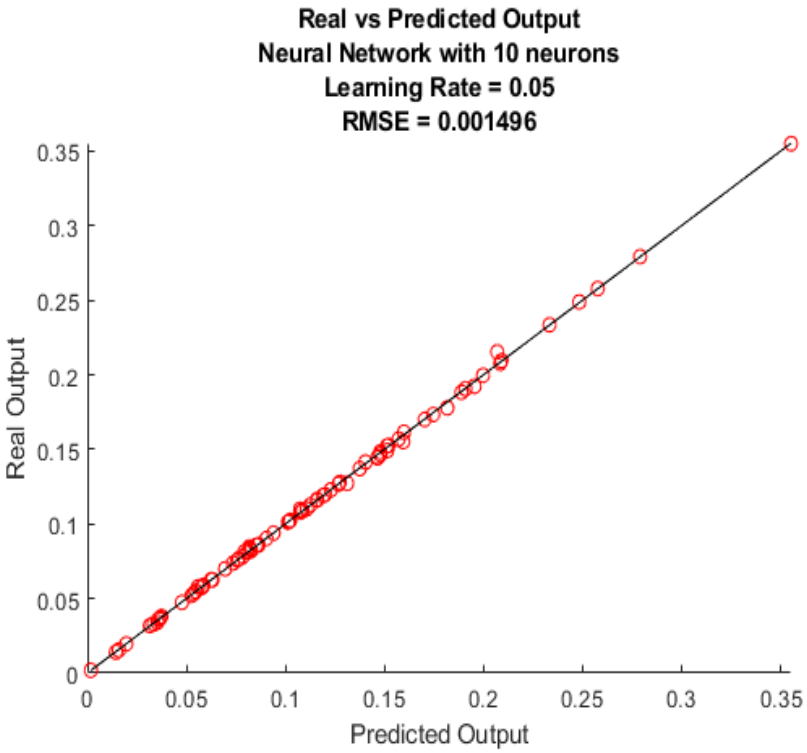

Figure 2.22 Neural Network for the case study with 10 neurons and 0,05 as the learning rate

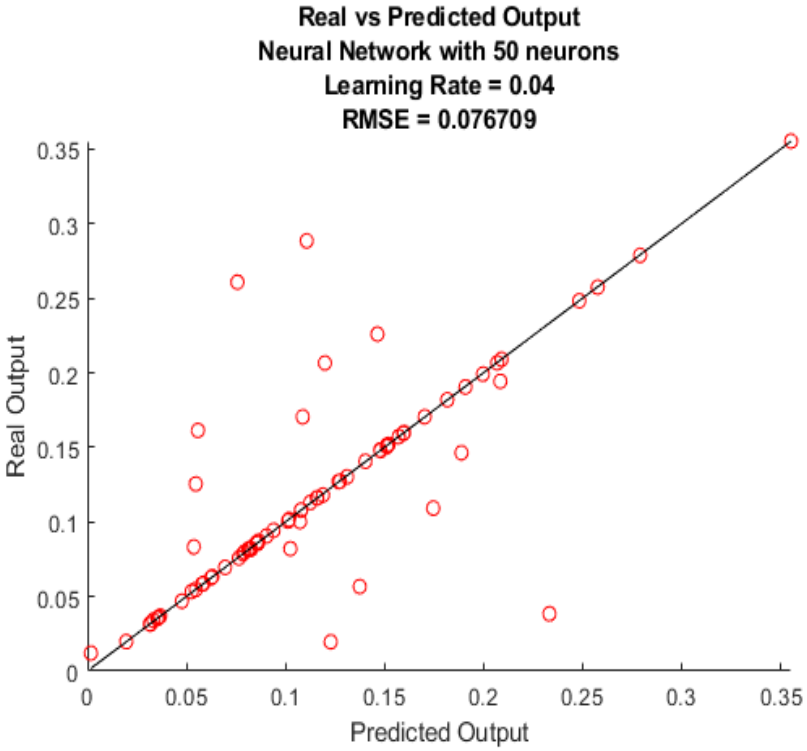

Figure 2.23 Neural Network for the case study with 50 neurons and 0,04 as the learning rate

As can be seen, data are scattered around the line with more density in Figure 2.22 than Figure 2.23, which shows less error in case of 10 neurons.

2.2.3 Conclusion

There were four inputs and one output in this case study, and the goal was to compare the result of implementing different machine learning methods to find the best pattern represents the data.

For this dataset, the best results in term of less RMSE belonged to a neural network structure with 10 neurons in the hidden layer. This RMSE was equal to 0,002. Also, in general, other structures of the neural network gave better results than random forest and regression. Therefore, the best method to represent the data among the mentioned approaches for this dataset was neural network.

However, in case of a need to an equation for visualization of the relationship between inputs and output, using regression is a straightforward solution, which in this case study gave the RMSE equal to 0,014.

SUMMARY

In the first chapter of the thesis, an overview of the theoretical basis of the data analysis process and machine learning methods has been presented. The importance of utilizing machine learning approach in industry and its benefits has been outlined. It has been pointed out that by using machine learning techniques, managers and engineers are able to make better decisions through the knowledge acquired by predictive analysis. The strength of machine learning techniques in addition to relative and qualified data enables engineers to find the hidden relationship between different variables that have not been facilitated for the engineers before. Several examples of using machine learning techniques, covering a wide class of problems, like finding the relationship between weather temperature and flood, prediction of financial fraud by evaluating the behaviour of thousands of transaction records, prediction of failure modes, predictive maintenance, traffic forecasting, etc., are presented.

The second chapter of the thesis includes the practical application of the machine learning techniques and comparison/analysis of results.

Firstly, a graphical user interface (GUI) has been designed to facilitate the using of machine learning methods and visualizing data for the managers and engineers with basic knowledge of programming. This GUI covers three major machine learning techniques: Regression, Random Forest, Artificial Neural Network. This GUI is able to get the variables and objectives dataset as input and output in its related tabs and apply three mentioned machine learning methods on them.

Secondly, three machine learning techniques have been applied for the modelling of the electromagnet generator of a wind turbine with an aim to determine the best method to estimate the pattern and the relationship between air-gap, magnet thickness, magnet length and coil thickness as variables, and combined objective (rotor and stator mechanical characteristics are combined into one objective). In the case of the particular problem considered the ANN approach appears most suitable technique, providing the highest accuracy. The ANN and random forest can be preferred to traditional modelling techniques in the case of dynamic problems since they support incremental training capabilities.

KOKKUVÕTE

Antud uurimistöö esimeses osas antakse lühiülevaade andmeanalüüsi protsessidest ja masinõppe meetoditest. Välja on toodud masinõppe olulisus tööstuses ja sellest tulenevad eelised. Tõdetud on, et masinõppe tehnikate rakendamine tehnika valdkonnas võimaldab saada uusi teadmisi eelanalüüsi tulemusena ning suurendab inseneride ja juhtivtöötajate võimekust võtta vastu asjakohaseid otsuseid. Samuti võimaldavad masinõppe tehnikad leida peidetud seoseid parameetrite vahel ning rakendada neid nii analüüsis ja otsuste vastuvõtmisel.
Töös on kirjeldatud mitmed masinõppe rakendamise näited nagu seoste leidmine õhutemperatuuri ja üleujutuste vahel, finantspettuste kindlaks tegemine tuhandete ülekandekirjete põhjal, purunemise põhjuste/viiside määramine, ennustav hooldus, liikluse prognoosimine, jne.

Uurimistöö teine peatükk sisaldab masinõppe tehnikate praktilist rakendamist, samuti saadud tulemuste analüüsi ja võrdlust.

Kõigepealt töötatakse välja graafiline kasutajaliides haldamaks erinevaid masinõppe meetodeid tagades nii vajalike andmete sisestamise kui saadud tulemuste graafilise visualiseerimise. Arvesse on võetud kolm erinevat tehnikat: regressioon, otsustusmets ja tehisnärvivõrk.

Seejärel rakendatakse masinõppe meetodeid tuule turbiini uurdevaba püsimagnet-generaatori modelleerimiseks eesmärgiga teha kindlaks parim/sobivaim meetod kirjeldamaks seoseid sisendparameetrite (õhupilu, magneti pikkus ja paksus, mähise paksus) ja väljundparameetrite vahel (kombineeritud sihifunktsioon). Vaadeldud konkreetse probleemi korral osutus sobivaimaks tehisnärvivõrke kasutava lahendus, mis tagas suurima täpsuse. Tehisnärvivõrkude ja otsustusmetsa meetodite eeliseks on nende võimekus teostada õppimist järkjärguliselt (ainult uute andmetega), mis on eriti oluline dünaamiliste probleemide korral.

LIST OF REFERENCES

[1] O. Pabut, "Optimal Design of Slotless Permanent Magnet Generators," Tallinn University of Technology, 2015.

[2] D. Cielen, A. Meysman, and M. Ali, *Introducing data science : big data, machine learning, and more, using Python tools*. 2016.

[3] J. Zawadzki, "The Power of Goal-Setting in Data Science - Towards Data Science." [Online]. Available: https://towardsdatascience.com/the-power-of-goal-setting-for-your-data-science-project-9338bf475abd. [Accessed: 12-Jan-2020].

[4] H. L. Capron, *Computers : tools for an information age*. Prentice Hall, 2000.

[5] MySQLTUTORIAL, "MySQL Sample Database." 2017.

[6] T. Walters and I. Marketing, "Incorporating External Data Into the Data Warehouse."

[7] C. McCue, "Process Models for Data Mining and Analysis," in *Data Mining and Predictive Analysis*, Elsevier, 2007, pp. 45–66.

[8] F. Nargesian, H. Samulowitz, U. Khurana, E. B. Khalil, and D. Turaga, "Learning feature engineering for classification," in *IJCAI International Joint Conference on Artificial Intelligence*, 2017, pp. 2529–2535.

[9] Elite Data Science, "Dimensionality Reduction Algorithms: Strengths and Weaknesses," 2019. [Online]. Available: https://elitedatascience.com/dimensionality-reduction-algorithms. [Accessed: 17-Jan-2020].

[10] DeepAI Inc., "Feature Extraction Definition | DeepAI," 2019. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/feature-extraction. [Accessed: 15-Jan-2020].

[11] Z. M. Hira and D. F. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Adv. Bioinformatics*, vol. 2015, 2015.

[12] R. James, G., Witten, D., Hastie, T., Tibshirani, "An Introduction to Statistical Learning - with Applications in R | Gareth James | Springer," 2013. [Online]. Available: https://www.springer.com/gp/book/9781461471370. [Accessed: 14-Jan-2020].

[13] V. Kumar, "Feature Selection: A literature Review," *Smart Comput. Rev.*, vol. 4, no. 3, 2014.

[14] "PREDICTIVE ANALYTICS- MEANING AND IMPORTANT ALGORITHMS TO LEARN." [Online]. Available: https://www.proschoolonline.com/blog/predictive-analytics-meaning-important-algorithms-learn. [Accessed: 18-Apr-2020].

[15] "Fact or Hype: Validating Predictive People Analytics and Machine Learning."

[Online]. Available: https://www.visier.com/clarity/predictive-people-analytics-machine-learning/. [Accessed: 18-Apr-2020].

[16] M. Talabis, R. McPherson, and I. Miyamoto, *Information security analytics : Finding security insights, patterns and anomalies in big data.* Elsevier Science, 2014.

[17] T. et. all. Hastie, *Springer Series in Statistics The Elements of Statistical Learning*, vol. 27, no. 2. 2009.

[18] A. Ligeza and P. Norvig, "Artificial Intelligence: A Modern Approach," *Neurocomputing*, vol. 9, no. 2, pp. 528–529, 1995.

[19] A. Singh, "Foundations of Machine Learning," *SSRN Electron. J.*, 2019.

[20] Margaret A. Boden, Ed., *Artificial Intelligence*. Elsevier, 1996.

[21] K. Ghazvini, M. Yousefi, F. Firoozeh, and S. Mansouri, "Predictors of tuberculosis: Application of a logistic regression model," *Gene Reports*, vol. 17, p. 100527, Dec. 2019.

[22] L. Liu, "Biostatistical Basis of Inference in Heart Failure Study," in *Heart Failure: Epidemiology and Research Methods*, Elsevier, 2018, pp. 43–82.

[23] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[24] Y. Zhou and G. Qiu, "Random forest for label ranking," *Expert Syst. Appl.*, vol. 112, pp. 99–109, Dec. 2018.

[25] Ö. F. Ertuğrul and M. E. Tağluk, "A novel version of k nearest neighbor: Dependent nearest neighbor," *Appl. Soft Comput. J.*, vol. 55, pp. 480–490, Jun. 2017.

[26] S. Lahoti, K. Mathew, and G. Miner, "Predictive Process Control," in *Handbook of Statistical Analysis and Data Mining Applications*, Elsevier, 2009, pp. 513–530.

[27] "Induction of Decision Trees." [Online]. Available: http://www.cse.unsw.edu.au/~billw/cs9414/notes/ml/06prop/id3/id3.html. [Accessed: 13-Apr-2020].

[28] J. A. Carta, "Wind power integration," in *Comprehensive Renewable Energy*, vol. 2, Elsevier Ltd, 2012, pp. 569–622.

[29] G. Zhu and D. G. Blumberg, "Classification using ASTER data and SVM algorithms: The case study of Beer Sheva, Israel," *Remote Sens. Environ.*, vol. 80, no. 2, pp. 233–240, May 2002.

[30] G. Mountrakis, J. Im, and C. Ogole, "Support vector machines in remote sensing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 66, no. 3. Elsevier, pp. 247–259, 01-May-2011.

[31] E. Traini, G. Bruno, G. D'Antonio, and F. Lombardi, "Machine learning framework for predictive maintenance in milling," *IFAC-PapersOnLine*, vol. 52,

no. 13, pp. 177–182, Sep. 2019.

[32] H. Zhu *et al.*, "Application of machine learning algorithms in quality assurance of fermentation process of black tea— based on electrical properties," *J. Food Eng.*, vol. 263, pp. 165–172, Dec. 2019.

[33] S. Y. Gao, D. Simchi-Levi, C.-P. Teo, and Z. Yan, "Disruption risk mitigation in supply chains: The risk exposure index revisited Disruption Risk Mitigation in Supply Chains-The Risk Exposure Index Revisited," *pubsonline.informs.org*, vol. 67, no. 3, pp. 831–852, 2019.

[34] J. Yoon, S. Talluri, H. Yildiz, and W. Ho, "Models for supplier selection and risk mitigation: a holistic approach," *Int. J. Prod. Res.*, vol. 56, no. 10, pp. 3636–3661, May 2018.

[35] I. M. Cavalcante, E. M. Frazzon, F. A. Forcellini, and D. Ivanov, "A supervised machine learning approach to data-driven simulation of resilient supplier selection in digital manufacturing," *Int. J. Inf. Manage.*, vol. 49, pp. 86–97, Dec. 2019.

[36] S. Lee *et al.*, "Intelligent traffic control for autonomous vehicle systems based on machine learning," *Expert Syst. Appl.*, vol. 144, p. 113074, Apr. 2020.

[37] S. A. Bagloee, K. H. Johansson, and M. Asadi, "A hybrid machine-learning and optimization method for contraflow design in post-disaster cases and traffic management scenarios," *Expert Syst. Appl.*, vol. 124, pp. 67–81, Jun. 2019.

[38] J. Helsen, "SCADA analysis for condition monitoring Wind Turbine Drivetrain Reliability Collaborative View project LoCoMot-Low Cost Monitoring for Wind Turbines View project."

[39] A. Stetco *et al.*, "Machine learning methods for wind turbine condition monitoring: A review," *Renewable Energy*, vol. 133. Elsevier Ltd, pp. 620–635, 01-Apr-2019.

[40] R. K. Ibrahim, J. Tautz-Weinert, and S. J. Watson, "Neural networks for wind turbine fault detection via current signature analysis," *Present. Wind Eur. Summit 2016*, 2016.

[41] R. Dash and P. K. Dash, "MDHS-LPNN: A Hybrid FOREX Predictor Model Using a Legendre Polynomial Neural Network with a Modified Differential Harmony Search Technique," in *Handbook of Neural Computation*, Elsevier Inc., 2017, pp. 459–486.

[42] "Asian Metal - The World Metal Information Center." [Online]. Available: http://www.asianmetal.com/. [Accessed: 14-May-2020].

[43] Z. Zhang, A. Chen, A. Matveev, R. Nilssen, and A. Nysveen, "High-power generators for offshore wind turbines," in *Energy Procedia*, 2013, vol. 35, pp. 52–61.

[44] A. H. Isfahani, A. H.-S. Boroujerdi, and S. Hasanzadeh, "Multi-objective design optimization of a large-scale directdrive permanent magnet generator for wind energy conversion systems," *Front. Energy*, vol. 8, no. 2, pp. 182–191, Jun. 2014.

[45] H. Li and Z. Chen, "Design optimization and site matching of direct-drive permanent magnet wind power generator systems," *Renew. Energy*, vol. 34, no. 4, pp. 1175–1184, Apr. 2009.

[46] J. J. G. Grant, B. S. Bagepalli, P. L. Jansen, P. S. DiMascio, A. D. G. Gadre, and R. Qu, "Method and apparatus for wind turbine air gap control," 2007.

[47] J. P. Paredes-Sánchez, E. Villicaña-Ortíz, and J. Xiberta-Bernat, "Materials use in electricity generators in wind turbines," *J. Clean. Prod.*, vol. 87, no. 1, pp. 501–504, 2015.

APPENDIX

Appendix 1 GUI code in MATLAB

```
classdef GUI_Reyhaneh_Joodattabrizi_exported < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        TabGroup                        matlab.ui.container.TabGroup
        InputTab                        matlab.ui.container.Tab
        Step1ButtonGroup                matlab.ui.container.ButtonGroup
        GneratedatafromequationButton   matlab.ui.control.RadioButton
        ImportdatafromworkspaceButton   matlab.ui.control.RadioButton
        ExtraRadio                      matlab.ui.control.RadioButton
        Step2GeneratingdatafromEquationPanel  matlab.ui.container.Panel
        NumberofInputsSpinnerLabel      matlab.ui.control.Label
        NumberofInputsGenSpinner        matlab.ui.control.Spinner
        NumberofLevelsSpinnerLabel      matlab.ui.control.Label
        NumberofLevelsSpinner           matlab.ui.control.Spinner
        Beginningofx1EditFieldLabel     matlab.ui.control.Label
        Beginningofx1EditField          matlab.ui.control.NumericEditField
        Endofx1EditFieldLabel           matlab.ui.control.Label
        Endofx1EditField                matlab.ui.control.NumericEditField
        Beginningofx2EditFieldLabel     matlab.ui.control.Label
        Beginningofx2EditField          matlab.ui.control.NumericEditField
        Endofx2EditFieldLabel           matlab.ui.control.Label
        Endofx2EditField                matlab.ui.control.NumericEditField
        Beginningofx3EditFieldLabel     matlab.ui.control.Label
        Beginningofx3EditField          matlab.ui.control.NumericEditField
        Endofx3EditFieldLabel           matlab.ui.control.Label
        Endofx3EditField                matlab.ui.control.NumericEditField
        Beginningofx4EditFieldLabel     matlab.ui.control.Label
        Beginningofx4EditField          matlab.ui.control.NumericEditField
        Endofx4EditFieldLabel           matlab.ui.control.Label
        Endofx4EditField                matlab.ui.control.NumericEditField
        Beginningofx5EditFieldLabel     matlab.ui.control.Label
        Beginningofx5EditField          matlab.ui.control.NumericEditField
        Endofx5EditFieldLabel           matlab.ui.control.Label
        Endofx5EditField                matlab.ui.control.NumericEditField
        EquationEditFieldLabel          matlab.ui.control.Label
        EquationEditField               matlab.ui.control.EditField
        ResetButton_Gen                 matlab.ui.control.Button
        DoneButton_Gen                  matlab.ui.control.Button
        Step2ImportdatafromworkspacePanel  matlab.ui.container.Panel
        NumberofInputsSpinner_2Label    matlab.ui.control.Label
        NumberofInputsWSSpinner_2       matlab.ui.control.Spinner
        InputNumber1x1DropDownLabel     matlab.ui.control.Label
        InputNumber1x1DropDown          matlab.ui.control.DropDown
        InputNumber2x2DropDownLabel     matlab.ui.control.Label
        InputNumber2x2DropDown          matlab.ui.control.DropDown
        InputNumber3x3DropDownLabel     matlab.ui.control.Label
        InputNumber3x3DropDown          matlab.ui.control.DropDown
        InputNumber4x4DropDownLabel     matlab.ui.control.Label
        InputNumber4x4DropDown          matlab.ui.control.DropDown
```

```
        InputNumber5x5DropDownLabel    matlab.ui.control.Label
        InputNumber5x5DropDown         matlab.ui.control.DropDown
        OutputYDropDownLabel           matlab.ui.control.Label
        OutputYDropDown                matlab.ui.control.DropDown
        ResetButton_WS                 matlab.ui.control.Button
        DoneButton_WS                  matlab.ui.control.Button
        RegressionTab                  matlab.ui.container.Tab
        NBText                         matlab.ui.control.Label
        Regression1InputPanel          matlab.ui.container.Panel
        polynomialdegreeSpinnerLabel   matlab.ui.control.Label
        polynomialdegreeSpinner1       matlab.ui.control.Spinner
        RobustDropDownLabel            matlab.ui.control.Label
        RobustDropDown                 matlab.ui.control.DropDown
        DoneButton_Reg1                matlab.ui.control.Button
        ResetButton_Reg1               matlab.ui.control.Button
        Result_Reg1                    matlab.ui.control.TextArea
        Regression2InputPanel          matlab.ui.container.Panel
        polynomialdegreex1Label        matlab.ui.control.Label
        polynomialdegreeSpinnerx1      matlab.ui.control.Spinner
        RobustDropDown_2Label          matlab.ui.control.Label
        RobustDropDown2                matlab.ui.control.DropDown
        DoneButton_Reg2                matlab.ui.control.Button
        ResetButton_Reg2               matlab.ui.control.Button
        Result_Reg2                    matlab.ui.control.TextArea
        polynomialdegreex2Label        matlab.ui.control.Label
        polynomialdegreeSpinnerx2      matlab.ui.control.Spinner
        RandomForrestTab               matlab.ui.container.Tab
        RandomForrestPanel             matlab.ui.container.Panel
        ResetButton_RF                 matlab.ui.control.Button
        DoneButton_RF                  matlab.ui.control.Button
        Result_RF                      matlab.ui.control.TextArea
        NumberofthebagsEditFieldLabel  matlab.ui.control.Label
        NumberofthebagsEditField       matlab.ui.control.NumericEditField
        NeuralNetworkTab               matlab.ui.container.Tab
        NNPanel                        matlab.ui.container.Panel
        NumberofNeuronsEditFieldLabel  matlab.ui.control.Label
        NumberofNeuronsEditField       matlab.ui.control.NumericEditField
        ResetButton_NN                 matlab.ui.control.Button
        DoneButton_NN                  matlab.ui.control.Button
        Results_NN                     matlab.ui.control.TextArea
        LearningrateLabel              matlab.ui.control.Label
        LearningRate_NN                matlab.ui.control.NumericEditField
    end


    properties (Access = public)
        % Regression results
        fitresult;
        gof;
        Reg_Result_Str;
        X;
        Y;
        X_ffd;
        Y_ffd;
        Inputs;
        Outputs;
        Data_Generation_Type;
```

```
    end

    methods (Access = public)

        function Regression(app)
            %             global Inputs;
            %             global Outputs;
            switch size(app.X,1)
                case 1
                    [xData, yData] = prepareCurveData( app.X(1,:), app.Y(1,:) );
                    ft = fittype( ['poly', num2str(app.polynomialdegreeSpinner1.Value)] );
                    %ft = fittype( ['poly', num2str(2)] );
                    %opts = fitoptions( 'Method', 'LinearLeastSquares' );
                    if ~isequal(app.RobustDropDown.Value,'None')
                        opts.Robust = app.RobustDropDown.Value;
                    end
                    [app.fitresult, app.gof] = fit( xData, yData, ft);
                    RMSE_Reg = app.gof.rmse;
                    figure( 'Name', ['Regression','-
poly',num2str(app.polynomialdegreeSpinner1.Value)] );
                    h = plot( app.fitresult, xData, yData );
                    legend( h, 'Outputs', 'Fitted_Curve', 'Location', 'NorthEast',
'Interpreter', 'none' );
                    % Label axes
                    title(['Regression','-
poly',num2str(app.polynomialdegreeSpinner1.Value),...
                        char(10), 'RMSE= ', num2str(RMSE_Reg)])
                    xlabel( 'Input', 'Interpreter', 'none' );
                    ylabel( 'Output', 'Interpreter', 'none' );
                    grid on

                case 2
                    [xData, yData, zData] = prepareSurfaceData( app.X_ffd(1,:),
app.X_ffd(2,:), app.Y_ffd(1,:) );
                    ft = fittype( ['poly' , num2str(app.polynomialdegreeSpinnerx1.Value),
num2str(app.polynomialdegreeSpinnerx2.Value)]);
                    %ft = fittype('poly32')
                    opts = fitoptions( 'Method', 'LinearLeastSquares' );
                    if ~isequal(app.RobustDropDown2.Value,'None')
                        opts.Robust = app.RobustDropDown2.Value;
                    end
                    % Fit model to data.
                    [app.fitresult,app.gof] = fit( [xData, yData], zData, ft, opts );
                    figure( 'Name', ['Regression','-
poly',num2str(app.polynomialdegreeSpinnerx1.Value),
num2str(app.polynomialdegreeSpinnerx2.Value)] );
                    [Reg_surface]=app.fitresult(app.X_ffd(1,:),app.X_ffd(2,:))
                    if isequal(app.Data_Generation_Type,'Equation')

surf(vec2mat(app.X_ffd(1,:),size(app.X,2)),vec2mat(app.X_ffd(2,:),size(app.X,2)),vec2mat(Re
g_surface,size(app.X,2)));
                    else
                        plot( app.fitresult, [xData, yData], zData );
                    end
                    hold on;
                    scatter3(app.X_ffd(1,:), app.X_ffd(2,:),app.Y_ffd(1,:),'red')
                    %h = plot( app.fitresult, [xData, yData], zData );
                    legend('Fitted' ,'Data', 'Location', 'NorthEast', 'Interpreter', 'none'
```

```
);
                    RMSE_Reg = app.gof.rmse

%MSE_Reg=mse(app.Y_ffd(1,:),app.fitresult(app.X_ffd(1,:),app.X_ffd(2,:)))
                    title(['Regression','-
poly',num2str(app.polynomialdegreeSpinnerx1.Value),
num2str(app.polynomialdegreeSpinnerx2.Value),...
                        char(10), 'RMSE= ', num2str(RMSE_Reg)])
                    %legend( h1, 'Data', 'Data', 'Location', 'NorthEast', 'Interpreter',
'none' );
                    % Label axes
                    xlabel( 'Input1', 'Interpreter', 'none' );
                    ylabel( 'Input2', 'Interpreter', 'none' );
                    zlabel( 'Output', 'Interpreter', 'none' );
                    grid on
            end
            Formula = strrep(formula(app.fitresult),'x','x1');
            Formula = strrep(Formula,'y','x2');

            app.Reg_Result_Str = ['Estimated
Equation:',char(10),Formula,char(10),char(10)];
            app.Reg_Result_Str = [app.Reg_Result_Str,'RMSE:',char(10),
num2str(RMSE_Reg),char(10),char(10)];

            CoeffsNames = coeffnames(app.fitresult);
            CoeffVals = coeffvalues(app.fitresult);
            app.Reg_Result_Str =
[app.Reg_Result_Str,'Coefficients',char(10),CoeffsNames{1},char(9),num2str(CoeffVals(1)),ch
ar(10)];
            if size(CoeffsNames,1)>1
                for i = 2: size(CoeffsNames,1)
                    app.Reg_Result_Str =
[app.Reg_Result_Str,CoeffsNames{i},char(9),num2str(CoeffVals(i)),char(10)];
                end
            end

        end
    end


    % Callbacks that handle component events
    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app)
            close all;
            app.ExtraRadio.Value = 1;
            app.Data_Generation_Type = '';

            app.X=[];
            app.Y=[];
            app.X_ffd=[];
            app.Y_ffd=[];

        end

        % Selection changed function: Step1ButtonGroup
        function Step1ButtonGroupSelectionChanged(app, event)
```

```
            selectedButton = app.Step1ButtonGroup.SelectedObject;
            if selectedButton.Text == 'Gnerate data from equation'
                app.Step2Importdatafromworkspacepanel.Visible = false;
                ResetButton_GenPushed(app);
                app.Step2GeneratingdatafromEquationPanel.Visible = true;
            elseif selectedButton.Text == 'Import data from workspace'
                app.Step2GeneratingdatafromEquationPanel.Visible = false;
                app.Step2Importdatafromworkspacepanel.Visible = true;
                % Import data from workspace
                if isempty(regexp(which('whos'), 'built-in *.'))
                    error('The built-in whos()function is being overshaddowed. Do not
proceed')
                end
                % Get base workspace variable info
                baseVarInfo = evalin('base', 'whos');
                baseVarInfo = rmfield(baseVarInfo,{'nesting','global','sparse',...
                    'complex','persistent','bytes'});
                % Remove not double variables
                i=1;
                while true
                    if size(baseVarInfo,1)< i
                        break;
                    end
                    if ~isequal(baseVarInfo(i).class,'double')
                        baseVarInfo(i)=[];
                    else
                        i=i+1;
                    end
                end
                global rowNames;
                rowNames = {baseVarInfo.name};
                % To add select option to the dropdown items
                for i = 1: size(rowNames,2)
                    rowNames_Toshow{i+1}=rowNames{i};
                end
                rowNames_Toshow{1}='select'
                app.InputNumber1x1DropDown.Items=rowNames_Toshow;
                app.InputNumber2x2DropDown.Items=rowNames_Toshow;
                app.InputNumber3x3DropDown.Items=rowNames_Toshow;
                app.InputNumber4x4DropDown.Items=rowNames_Toshow;
                app.InputNumber5x5DropDown.Items=rowNames_Toshow;
                app.OutputYDropDown.Items = rowNames_Toshow;
                ResetButton_WSPushed(app);
            end
        end

        % Button pushed function: DoneButton_Gen
        function DoneButton_GenPushed(app, event)
            % X refers to initial data (raw data entered by user)
            app.DoneButton_Gen.Enable = 'off';
            app.RandomForrestPanel.Visible = true;
            app.NNPanel.Visible = 'on';
            switch app.NumberofInputsGenSpinner.Value
                case 1
                    X1 = linspace(app.Beginningofx1EditField.Value,
app.Endofx1EditField.Value,app.NumberofLevelsSpinner.Value);
                    app.X = X1;
                    EquationHandle='@(x1)'
```

```
                    EvalEq= 'Equation = @(x1)'
                    app.TabGroup.SelectedTab = app.RegressionTab;
                    app.Regression1InputPanel.Visible = true;
                    app.Regression2InputPanel.Visible = false;
                case 2
                    X1 = linspace(app.Beginningofx1EditField.Value,
app.Endofx1EditField.Value,app.NumberofLevelsSpinner.Value);
                    X2 = linspace(app.Beginningofx2EditField.Value,
app.Endofx2EditField.Value,app.NumberofLevelsSpinner.Value);
                    app.X = [X1;X2];
                    EquationHandle='@(x1,x2)'
                    EvalEq= 'Equation = @(x1,x2)';
                    app.TabGroup.SelectedTab = app.RegressionTab;
                    app.Regression1InputPanel.Visible = false;
                    app.Regression2InputPanel.Visible = true;
                case 3
                    X1 = linspace(app.Beginningofx1EditField.Value,
app.Endofx1EditField.Value,app.NumberofLevelsSpinner.Value);
                    X2 = linspace(app.Beginningofx2EditField.Value,
app.Endofx2EditField.Value,app.NumberofLevelsSpinner.Value);
                    X3 = linspace(app.Beginningofx3EditField.Value,
app.Endofx3EditField.Value,app.NumberofLevelsSpinner.Value);
                    app.X = [X1;X2;X3];
                    EquationHandle='@(x1,x2,x3)'
                    EvalEq= 'Equation = @(x1,x2,x3)'
                    app.TabGroup.SelectedTab = app.RandomForrestTab;
                case 4
                    X1 = linspace(app.Beginningofx1EditField.Value,
app.Endofx1EditField.Value,app.NumberofLevelsSpinner.Value);
                    X2 = linspace(app.Beginningofx2EditField.Value,
app.Endofx2EditField.Value,app.NumberofLevelsSpinner.Value);
                    X3 = linspace(app.Beginningofx3EditField.Value,
app.Endofx3EditField.Value,app.NumberofLevelsSpinner.Value);
                    X4 = linspace(app.Beginningofx4EditField.Value,
app.Endofx4EditField.Value,app.NumberofLevelsSpinner.Value);
                    app.X = [X1;X2;X3;X4];
                    EquationHandle='@(x1,x2,x3,x4)';
                    EvalEq= 'Equation = @(x1,x2,x3,x4)';
                    app.TabGroup.SelectedTab = app.RandomForrestTab;
                case 5
                    X1 = linspace(app.Beginningofx1EditField.Value,
app.Endofx1EditField.Value,app.NumberofLevelsSpinner.Value);
                    X2 = linspace(app.Beginningofx2EditField.Value,
app.Endofx2EditField.Value,app.NumberofLevelsSpinner.Value);
                    X3 = linspace(app.Beginningofx3EditField.Value,
app.Endofx3EditField.Value,app.NumberofLevelsSpinner.Value);
                    X4 = linspace(app.Beginningofx4EditField.Value,
app.Endofx4EditField.Value,app.NumberofLevelsSpinner.Value);
                    X5 = linspace(app.Beginningofx5EditField.Value,
app.Endofx5EditField.Value,app.NumberofLevelsSpinner.Value);
                    app.X = [X1;X2;X3;X4;X5];
                    EquationHandle='@(x1,x2,x3,x4,x5)';
                    EvalEq= 'Equation = @(x1,x2,x3,x4,x5)';
                    app.TabGroup.SelectedTab = app.RandomForrestTab;
            end

            EquationVectorized = vectorize(char(app.EquationEditField.Value));
            Equation = str2func([EquationHandle EquationVectorized]);
```

```
            eval([EvalEq EquationVectorized]);

            switch size(app.X,1)
                case 1
                    app.Y = Equation(X1);
                    figure( 'Name', 'Generating data from equation' );
                    scatter(X1,app.Y);
                    title('Generated Data');
                    xlabel('x1');
                    ylabel('Y');
                    app.X_ffd=app.X;
                    app.Y_ffd = app.Y;
                case 2
                    app.Y = Equation(X1,X2);
                    figure( 'Name', 'Generating data from equation' );
                    scatter3(X1,X2,app.Y);
                    title('Generated Data');
                    xlabel('x1');
                    ylabel('x2');
                    zlabel('Y');
                    Levels = zeros(1,size(app.X,1));
                    Levels(:)=size(app.X,2);
                    ffd = fullfact (Levels);
                    app.X_ffd = zeros(size(ffd,2),size(ffd,1));
                    for  i = 1 : size (ffd,2)
                        app.X_ffd(i,:) = app.X(i,ffd(:,i)); %%
                    end
                    app.Y_ffd(1,:)=Equation(app.X_ffd(1,:),app.X_ffd(2,:)) ;
                case 3
                    app.Y= Equation(X1,X2,X3);
                    app.X_ffd=app.X;
                    app.Y_ffd = app.Y;
                case 4
                    app.Y= Equation(X1,X2,X3,X4);
                    app.X_ffd=app.X;
                    app.Y_ffd = app.Y;
                case 5
                    app.Y= Equation(X1,X2,X3,X4,X5);
                    app.X_ffd=app.X;
                    app.Y_ffd = app.Y;
            end
            app.Data_Generation_Type = 'Equation';

        end

        % Value changed function: NumberofInputsGenSpinner
        function NumberofInputsGenSpinnerValueChanged(app, event)
            value = app.NumberofInputsGenSpinner.Value;
            switch value
                case 1
                    app.Beginningofx1EditField.Editable  = 'on';
                    app.Endofx1EditField.Editable  = 'on';
                    app.Beginningofx2EditField.Editable  = 'off';
                    app.Endofx2EditField.Editable  = 'off';
                    app.Beginningofx3EditField.Editable  = 'off';
                    app.Endofx3EditField.Editable  = 'off';
                    app.Beginningofx4EditField.Editable  = 'off';
                    app.Endofx4EditField.Editable  = 'off';
```

```
            app.Beginningofx5EditField.Editable  = 'off';
            app.Endofx5EditField.Editable  = 'off';
            app.Beginningofx2EditField.Value = 0;
            app.Endofx2EditField.Value = 0;
            app.Beginningofx3EditField.Value = 0;
            app.Endofx3EditField.Value = 0;
            app.Beginningofx4EditField.Value = 0;
            app.Endofx4EditField.Value = 0;
            app.Beginningofx5EditField.Value = 0;
            app.Endofx5EditField.Value = 0;
        case 2
            app.Beginningofx1EditField.Editable  = 'on';
            app.Endofx1EditField.Editable  = 'on';
            app.Beginningofx2EditField.Editable  = 'on';
            app.Endofx2EditField.Editable  = 'on';
            app.Beginningofx3EditField.Editable  = 'off';
            app.Endofx3EditField.Editable  = 'off';
            app.Beginningofx4EditField.Editable  = 'off';
            app.Endofx4EditField.Editable  = 'off';
            app.Beginningofx5EditField.Editable  = 'off';
            app.Endofx5EditField.Editable  = 'off';
            app.Endofx2EditField.Value = 1;
            app.Beginningofx3EditField.Value = 0;
            app.Endofx3EditField.Value = 0;
            app.Beginningofx4EditField.Value = 0;
            app.Endofx4EditField.Value = 0;
            app.Beginningofx5EditField.Value = 0;
            app.Endofx5EditField.Value = 0;
        case 3
            app.Beginningofx1EditField.Editable  = 'on';
            app.Endofx1EditField.Editable  = 'on';
            app.Beginningofx2EditField.Editable  = 'on';
            app.Endofx2EditField.Editable  = 'on';
            app.Beginningofx3EditField.Editable  = 'on';
            app.Endofx3EditField.Editable  = 'on';
            app.Beginningofx4EditField.Editable  = 'off';
            app.Endofx4EditField.Editable  = 'off';
            app.Beginningofx5EditField.Editable  = 'off';
            app.Endofx2EditField.Value = 1;
            app.Endofx3EditField.Value = 1;
            app.Endofx5EditField.Editable  = 'off';
            app.Beginningofx4EditField.Value = 0;
            app.Endofx4EditField.Value = 0;
            app.Beginningofx5EditField.Value = 0;
            app.Endofx5EditField.Value = 0;
        case 4
            app.Beginningofx1EditField.Editable  = 'on';
            app.Endofx1EditField.Editable  = 'on';
            app.Beginningofx2EditField.Editable  = 'on';
            app.Endofx2EditField.Editable  = 'on';
            app.Beginningofx3EditField.Editable  = 'on';
            app.Endofx3EditField.Editable  = 'on';
            app.Beginningofx4EditField.Editable  = 'on';
            app.Endofx4EditField.Editable = 'on';
            app.Beginningofx5EditField.Editable  = 'off';
            app.Endofx5EditField.Editable  = 'off';
            app.Endofx2EditField.Value = 1;
            app.Endofx3EditField.Value = 1;
```

```
                app.Endofx4EditField.Value = 1;
                app.Beginningofx5EditField.Value = 0;
                app.Endofx5EditField.Value = 0;
            case 5
                app.Beginningofx1EditField.Editable  = 'on';
                app.Endofx1EditField.Editable  = 'on';
                app.Beginningofx2EditField.Editable  = 'on';
                app.Endofx2EditField.Editable  = 'on';
                app.Beginningofx3EditField.Editable  = 'on';
                app.Endofx3EditField.Editable = 'on';
                app.Beginningofx4EditField.Editable  = 'on';
                app.Endofx4EditField.Editable  = 'on';
                app.Beginningofx5EditField.Editable  = 'on';
                app.Endofx5EditField.Editable  = 'on';
                app.Endofx2EditField.Value = 1;
                app.Endofx3EditField.Value = 1;
                app.Endofx4EditField.Value = 1;
                app.Endofx5EditField.Value = 1;
        end
    end

    % Value changed function: EquationEditField
    function EquationEditFieldValueChanged(app, event)
        value = app.EquationEditField.Value;
    end

    % Button pushed function: ResetButton_Gen
    function ResetButton_GenPushed(app, event)
        app.DoneButton_Gen.Enable = 'on';
        app.Beginningofx1EditField.Value = 0;
        app.Endofx1EditField.Value = 1;
        app.NumberofInputsGenSpinner.Value = 1;
        app.NumberofLevelsSpinner.Value = 1;
        NumberofInputsGenSpinnerValueChanged(app);
        app.X = [];
        app.Y = [];
        app.X_ffd=[];
        app.Y_ffd=[];
    end

    % Value changed function: NumberofInputsWSSpinner_2
    function NumberofInputsWSSpinner_2ValueChanged(app, event)
        value = app.NumberofInputsWSSpinner_2.Value;
        switch app.NumberofInputsWSSpinner_2.Value
            case 1
                app.InputNumber1x1DropDown.Enable = 'on'
                app.InputNumber1x1DropDownLabel.Enable = 'on';
                app.InputNumber2x2DropDown.Enable = 'off';
                app.InputNumber2x2DropDownLabel.Enable = 'off';
                app.InputNumber3x3DropDown.Enable = 'off';
                app.InputNumber3x3DropDownLabel.Enable = 'off';
                app.InputNumber4x4DropDown.Enable = 'off';
                app.InputNumber4x4DropDownLabel.Enable = 'off';
                app.InputNumber5x5DropDown.Enable = 'off';
                app.InputNumber5x5DropDownLabel.Enable = 'off';

                app.InputNumber1x1DropDown.Value = 'select';
                app.InputNumber2x2DropDown.Value = 'select';
```

```
                app.InputNumber3x3DropDown.Value = 'select';
                app.InputNumber4x4DropDown.Value = 'select';
                app.InputNumber5x5DropDown.Value = 'select';
                app.OutputYDropDown.Value = 'select';

            case 2
                app.InputNumber1x1DropDown.Enable = 'on'
                app.InputNumber1x1DropDownLabel.Enable = 'on';
                app.InputNumber2x2DropDown.Enable = 'on';
                app.InputNumber2x2DropDownLabel.Enable = 'on';
                app.InputNumber3x3DropDown.Enable = 'off';
                app.InputNumber3x3DropDownLabel.Enable = 'off';
                app.InputNumber4x4DropDown.Enable = 'off';
                app.InputNumber4x4DropDownLabel.Enable = 'off';
                app.InputNumber5x5DropDown.Enable = 'off';
                app.InputNumber5x5DropDownLabel.Enable = 'off';
                app.InputNumber3x3DropDown.Value = 'select';
                app.InputNumber4x4DropDown.Value = 'select';
                app.InputNumber5x5DropDown.Value = 'select';
            case 3
                app.InputNumber1x1DropDown.Enable = 'on'
                app.InputNumber1x1DropDownLabel.Enable = 'on';
                app.InputNumber2x2DropDown.Enable = 'on';
                app.InputNumber2x2DropDownLabel.Enable = 'on';
                app.InputNumber3x3DropDown.Enable = 'on';
                app.InputNumber3x3DropDownLabel.Enable = 'on';
                app.InputNumber4x4DropDown.Enable = 'off';
                app.InputNumber4x4DropDownLabel.Enable = 'off';
                app.InputNumber5x5DropDown.Enable = 'off';
                app.InputNumber5x5DropDownLabel.Enable = 'off';
                app.InputNumber4x4DropDown.Value = 'select';
                app.InputNumber5x5DropDown.Value = 'select';
            case 4
                app.InputNumber1x1DropDown.Enable = 'on'
                app.InputNumber1x1DropDownLabel.Enable = 'on';
                app.InputNumber2x2DropDown.Enable = 'on';
                app.InputNumber2x2DropDownLabel.Enable = 'on';
                app.InputNumber3x3DropDown.Enable = 'on';
                app.InputNumber3x3DropDownLabel.Enable = 'on';
                app.InputNumber4x4DropDown.Enable = 'on';
                app.InputNumber4x4DropDownLabel.Enable = 'on';
                app.InputNumber5x5DropDown.Enable = 'off';
                app.InputNumber5x5DropDownLabel.Enable = 'off';
                app.InputNumber5x5DropDown.Value = 'select';
            case 5
                app.InputNumber1x1DropDown.Enable = 'on'
                app.InputNumber1x1DropDownLabel.Enable = 'on';
                app.InputNumber2x2DropDown.Enable = 'on';
                app.InputNumber2x2DropDownLabel.Enable = 'on';
                app.InputNumber3x3DropDown.Enable = 'on';
                app.InputNumber3x3DropDownLabel.Enable = 'on';
                app.InputNumber4x4DropDown.Enable = 'on';
                app.InputNumber4x4DropDownLabel.Enable = 'on';
                app.InputNumber5x5DropDown.Enable = 'on';
                app.InputNumber5x5DropDownLabel.Enable = 'on';
        end
    end
```

```
        % Button pushed function: ResetButton_WS
        function ResetButton_WSPushed(app, event)
            app.DoneButton_WS.Enable = 'on';
            app.NumberofInputsWSSpinner_2.Value = 1;
            NumberofInputsWSSpinner_2ValueChanged(app);
            app.X = [];
            app.Y = [];
            app.X_ffd=[];
            app.Y_ffd=[];
        end

        % Button pushed function: DoneButton_WS
        function DoneButton_WSPushed(app, event)
            app.DoneButton_WS.Enable = 'off';

            global rowNames;
            switch app.NumberofInputsWSSpinner_2.Value
                case 1
                    app.X(1,:) = evalin('base', app.InputNumber1x1DropDown.Value);
                    app.Y= evalin('base', app.OutputYDropDown.Value);
                    figure( 'Name', 'Imported data from workspace');
                    scatter(app.X(1,:),app.Y);
                    title('Imported Data');
                    xlabel('x1');
                    ylabel('Y');
                    app.ResetButton_Reg1Pushed(app);
                    app.TabGroup.SelectedTab = app.RegressionTab;
                    app.Regression1InputPanel.Visible = true;
                    app.Regression2InputPanel.Visible = false;
                case 2
                    app.X(1,:) = evalin('base', app.InputNumber1x1DropDown.Value);
                    app.X(2,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.Y= evalin('base', app.OutputYDropDown.Value);
                    figure( 'Name', 'Imported data from workspace' );
                    scatter3(app.X(1,:),app.X(2,:),app.Y);
                    title('Imported data');
                    xlabel('x1');
                    ylabel('x2');
                    zlabel('Y');
                    app.ResetButton_Reg2Pushed(app);
                    app.TabGroup.SelectedTab = app.RegressionTab;
                    app.Regression1InputPanel.Visible = false;
                    app.Regression2InputPanel.Visible = true;
                case 3
                    app.X(1,:) = evalin('base', app.InputNumber1x1DropDown.Value);
                    app.X(2,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.X(3,:) = evalin('base', app.InputNumber3x3DropDown.Value);
                    app.TabGroup.SelectedTab = app.RandomForrestTab;
                case 4
                    app.X(1,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.X(2,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.X(3,:) = evalin('base', app.InputNumber3x3DropDown.Value);
                    app.X(4,:) = evalin('base', app.InputNumber4x4DropDown.Value);
                    app.TabGroup.SelectedTab = app.RandomForrestTab;
                case 5
                    app.X(1,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.X(2,:) = evalin('base', app.InputNumber2x2DropDown.Value);
                    app.X(3,:) = evalin('base', app.InputNumber3x3DropDown.Value);
```

```
                app.X(4,:) = evalin('base', app.InputNumber4x4DropDown.Value);
                app.X(5,:) = evalin('base', app.InputNumber5x5DropDown.Value);
                app.TabGroup.SelectedTab = app.RandomForrestTab;

            end
            app.Y= evalin('base', app.OutputYDropDown.Value);
            app.X_ffd = app.X;
            app.Y_ffd = app.Y;
            app.Data_Generation_Type = 'Imported';
            app.RandomForrestPanel.Visible = 'on';
            app.NNPanel.Visible = 'on';
        end

        % Button pushed function: DoneButton_Reg1
        function DoneButton_Reg1Pushed(app, event)
            Regression(app);
            app.Result_Reg1.Value= app.Reg_Result_Str;
        end

        % Button pushed function: DoneButton_Reg2
        function DoneButton_Reg2Pushed(app, event)
            Regression(app);
            app.Result_Reg2.Value= app.Reg_Result_Str;
        end

        % Button pushed function: ResetButton_Reg2
        function ResetButton_Reg2Pushed(app, event)
            app.Result_Reg2.Value = '';
            app.RobustDropDown2.Value = 'None';
            app.polynomialdegreeSpinnerx1.Value= 1;
            app.polynomialdegreeSpinnerx2.Value= 1;
        end

        % Button pushed function: ResetButton_Reg1
        function ResetButton_Reg1Pushed(app, event)
            app.Result_Reg1.Value = '';
            app.RobustDropDown.Value = 'None';
            app.polynomialdegreeSpinner1.Value= 1;
        end

        % Value changed function: Result_Reg1
        function Result_Reg1ValueChanged(app, event)

        end

        % Close request function: UIFigure
        function UIFigureCloseRequest(app, event)
            delete(app)
            close all;
        end

        % Button pushed function: DoneButton_RF
        function DoneButton_RFPushed(app, event)
            %app.DoneButton_RF.Enable = 'off';
            %app.Result_RF.Value = 'Please Wait ...' ;
            switch size(app.X,1)
                case 1
                    BaggedEnsemble =
```

```
TreeBagger(app.NumberofthebagsEditField.Value,(app.X(1,:))',(app.Y(1,:))',...
                        'OOBPred','On','Method','regression');
                    Random_Forrest_Output(1,:) = predict(BaggedEnsemble,app.X(1,:)');
                    RMSE_RF = sqrt(mean((app.Y_ffd-Random_Forrest_Output).^2));
                    figure ('Name',['Random Forrest',' with ',
num2str(app.NumberofthebagsEditField.Value),' bags']);
                    scatter(app.X(1,:),app.Y(1,:),5,'blue',"filled");
                    hold on;
                    h = plot(app.X(1,:),Random_Forrest_Output(1,:));
                    legend( h, 'Outputs', 'Fitted_Curve', 'Location', 'NorthEast',
'Interpreter', 'none' )
                    title(['Random Forrest',' with ',
num2str(app.NumberofthebagsEditField.Value),' bags',...
                        char(10), 'RMSE= ', num2str(RMSE_RF)])
                    xlabel( 'Input', 'Interpreter', 'none' );
                    ylabel( 'Output', 'Interpreter', 'none' );
                    grid on
                case 2
                    BaggedEnsemble =
TreeBagger(app.NumberofthebagsEditField.Value,(app.X_ffd)',(app.Y_ffd)',...
                        'OOBPred','On','Method','regression');
                    Random_Forrest_Output(1,:) =
predict(BaggedEnsemble,[app.X_ffd(1,:)',app.X_ffd(2,:)']);
                    figure ('Name',['Random Forrest',' with ',
num2str(app.NumberofthebagsEditField.Value),' bags']);
                    scatter3(app.X_ffd(1,:),app.X_ffd(2,:),app.Y_ffd(1,:),'red');
                    hold on;
                    if isequal(app.Data_Generation_Type,'Equation')

surf(vec2mat(app.X_ffd(1,:),size(app.X,2)),vec2mat(app.X_ffd(2,:),size(app.X,2)),vec2mat(Ra
ndom_Forrest_Output,size(app.X,2)));
                    else
                        [Mesh_X1,Mesh_X2] =
meshgrid(linspace(min(app.X_ffd(1,:)),max(app.X_ffd(1,:)),size(app.X_ffd,2)),...

linspace(min(app.X_ffd(2,:)),max(app.X_ffd(2,:)),size(app.X_ffd,2)));
                        Mesh_ffd_X1 =Mesh_X1(:);
                        Mesh_ffd_X2 = Mesh_X2(:);
                        RF_ffd_Surface = predict(BaggedEnsemble,[Mesh_ffd_X1,Mesh_ffd_X2]);
                        surf(Mesh_X1,Mesh_X2,vec2mat(RF_ffd_Surface,size(app.X_ffd,2)));
                    end
                    legend('Data', 'Fitted', 'Location', 'NorthEast', 'Interpreter', 'none'
);
                    RMSE_RF = sqrt(mean((app.Y_ffd-Random_Forrest_Output).^2));
                    title(['Random Forrest',' with ',
num2str(app.NumberofthebagsEditField.Value),' bags',...
                        char(10), 'RMSE= ', num2str(RMSE_RF)])
                    xlabel( 'Input1', 'Interpreter', 'none' );
                    ylabel( 'Input2', 'Interpreter', 'none' );
                    zlabel( 'Output', 'Interpreter', 'none' );
                    grid on
                case 3
                    BaggedEnsemble =
TreeBagger(app.NumberofthebagsEditField.Value,(app.X_ffd)',(app.Y_ffd)',...
                        'OOBPred','On','Method','regression');
                    Random_Forrest_Output(1,:) =
predict(BaggedEnsemble,[(app.X_ffd(1,:))',(app.X_ffd(2,:))',(app.X_ffd(3,:))']);
                case 4
```

```
                    BaggedEnsemble =
TreeBagger(app.NumberofthebagsEditField.Value,(app.X_ffd)',(app.Y_ffd)',...
                        'OOBPred','On','Method','regression');
                    Random_Forrest_Output(1,:) =
predict(BaggedEnsemble,[(app.X_ffd(1,:))',(app.X_ffd(2,:))',(app.X_ffd(3,:))',(app.X_ffd(4,
:))']);
                case 5
                    BaggedEnsemble =
TreeBagger(app.NumberofthebagsEditField.Value,(app.X_ffd)',(app.Y_ffd)',...
                        'OOBPred','On','Method','regression');
                    Random_Forrest_Output(1,:) =
predict(BaggedEnsemble,[(app.X_ffd(1,:))',(app.X_ffd(2,:))',(app.X_ffd(3,:))',(app.X_ffd(4,
:))',(app.X_ffd(5,:))']);
            end
            RMSE_RF = sqrt(mean((app.Y_ffd-Random_Forrest_Output).^2));
            app.Result_RF.Value = ['RMSE= ', num2str(RMSE_RF)];

        end

        % Button pushed function: ResetButton_RF
        function ResetButton_RFPushed(app, event)
            app.DoneButton_RF.Enable = 'on';
            app.Result_RF.Value = '';
            app.NumberofthebagsEditField.Value = 1;
        end

        % Button pushed function: DoneButton_NN
        function DoneButton_NNPushed(app, event)
            tic;
            x = app.X_ffd;
            t = app.Y_ffd;
            trainFcn = 'trainlm';
            hiddenLayerSize = app.NumberofNeuronsEditField.Value;
            net = fitnet(hiddenLayerSize,trainFcn);
            net.trainParam.lr=app.LearningRate_NN.Value;
            net.divideParam.trainRatio = 70/100;
            net.divideParam.valRatio = 15/100;
            net.divideParam.testRatio = 15/100;
            [net,tr] = train(net,x,t);
            Neural_Network = net(x);
            RMSE_NN = sqrt(mean((app.Y_ffd-Neural_Network).^2));
            performance = perform(net,t,Neural_Network);
            app.Results_NN.Value = ['RMSE= ', num2str(RMSE_NN)];
            switch size(app.X,1)
                case 1
                    figure ('Name',['Neural Network',' with ',
num2str(app.NumberofNeuronsEditField.Value),' neurons']);
                    scatter(app.X(1,:),app.Y(1,:),5,'blue',"filled");
                    hold on;
                    h = plot(app.X(1,:),Neural_Network(1,:));
                    legend( h, 'Outputs', 'Fitted_Curve', 'Location', 'NorthEast',
'Interpreter', 'none' )
                    title(['Neural Network',' with ',
num2str(app.NumberofNeuronsEditField.Value),' neurons',...
                        char(10), 'RMSE= ', num2str(RMSE_NN)])
                    xlabel( 'Input', 'Interpreter', 'none' );
                    ylabel( 'Output', 'Interpreter', 'none' );
                    grid on
```

```
                case 2
                    figure ('Name',['Neural Network',' with ',
num2str(app.NumberofNeuronsEditField.Value),' neurons',char(10),'Learning
Rate=',num2str(app.LearningRate_NN.Value)]);
                    scatter3(app.X_ffd(1,:),app.X_ffd(2,:),app.Y_ffd(1,:),'red');
                    hold on;
                    if isequal(app.Data_Generation_Type,'Equation')

surf(vec2mat(app.X_ffd(1,:),size(app.X,2)),vec2mat(app.X_ffd(2,:),size(app.X,2)),vec2mat(Ne
ural_Network,size(app.X,2)));
                    else
                        [Mesh_X1,Mesh_X2] =
meshgrid(linspace(min(app.X_ffd(1,:)),max(app.X_ffd(1,:)),size(app.X_ffd,2)),...

linspace(min(app.X_ffd(2,:)),max(app.X_ffd(2,:)),size(app.X_ffd,2)));
                        Mesh_ffd_X1 =Mesh_X1(:);
                        Mesh_ffd_X2 = Mesh_X2(:);
                        NN_ffd_Surface = net([(Mesh_ffd_X1)';(Mesh_ffd_X2)']);
                        surf(Mesh_X1,Mesh_X2,vec2mat(NN_ffd_Surface,size(app.X_ffd,2)));
                    end
                    legend('Data', 'Fitted', 'Location', 'NorthEast', 'Interpreter', 'none'
);
                    %RMSE_NN = sqrt(mean((app.Y_ffd-Random_Forrest_Output).^2));
                    title(['Neural Network',' with ',
num2str(app.NumberofNeuronsEditField.Value),' neurons',...
                        char(10),'Learning Rate=',num2str(app.LearningRate_NN.Value),...
                        char(10), 'RMSE= ', num2str(RMSE_NN)])
                    xlabel( 'Input1', 'Interpreter', 'none' );
                    ylabel( 'Input2', 'Interpreter', 'none' );
                    zlabel( 'Output', 'Interpreter', 'none' );
                    grid on
            end
            toc;
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 377 627];
            app.UIFigure.Name = 'UI Figure';
            app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest,
true);

            % Create TabGroup
            app.TabGroup = uitabgroup(app.UIFigure);
            app.TabGroup.Position = [11 14 356 602];

            % Create InputTab
            app.InputTab = uitab(app.TabGroup);
            app.InputTab.Title = 'Input';

            % Create Step1ButtonGroup
```

```
            app.Step1ButtonGroup = uibuttongroup(app.InputTab);
            app.Step1ButtonGroup.SelectionChangedFcn = createCallbackFcn(app,
@Step1ButtonGroupSelectionChanged, true);
            app.Step1ButtonGroup.Title = 'Step1: ';
            app.Step1ButtonGroup.FontWeight = 'bold';
            app.Step1ButtonGroup.Position = [12 459 332 97];

            % Create GneratedatafromequationButton
            app.GneratedatafromequationButton = uiradiobutton(app.Step1ButtonGroup);
            app.GneratedatafromequationButton.Text = 'Gnerate data from equation';
            app.GneratedatafromequationButton.Position = [11 51 169 22];
            app.GneratedatafromequationButton.Value = true;

            % Create ImportdatafromworkspaceButton
            app.ImportdatafromworkspaceButton = uiradiobutton(app.Step1ButtonGroup);
            app.ImportdatafromworkspaceButton.Text = 'Import data from workspace';
            app.ImportdatafromworkspaceButton.Position = [11 9 171 22];

            % Create ExtraRadio
            app.ExtraRadio = uiradiobutton(app.Step1ButtonGroup);
            app.ExtraRadio.Enable = 'off';
            app.ExtraRadio.Visible = 'off';
            app.ExtraRadio.Text = 'Button3';
            app.ExtraRadio.Position = [262 9 65 22];

            % Create Step2GeneratingdatafromEquationPanel
            app.Step2GeneratingdatafromEquationPanel = uipanel(app.InputTab);
            app.Step2GeneratingdatafromEquationPanel.Title = 'Step2: Generating data from
Equation';
            app.Step2GeneratingdatafromEquationPanel.Visible = 'off';
            app.Step2GeneratingdatafromEquationPanel.FontWeight = 'bold';
            app.Step2GeneratingdatafromEquationPanel.Position = [13 15 331 428];

            % Create NumberofInputsSpinnerLabel
            app.NumberofInputsSpinnerLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.NumberofInputsSpinnerLabel.HorizontalAlignment = 'right';
            app.NumberofInputsSpinnerLabel.Position = [17 369 98 22];
            app.NumberofInputsSpinnerLabel.Text = 'Number of Inputs';

            % Create NumberofInputsGenSpinner
            app.NumberofInputsGenSpinner =
uispinner(app.Step2GeneratingdatafromEquationPanel);
            app.NumberofInputsGenSpinner.Limits = [1 5];
            app.NumberofInputsGenSpinner.ValueChangedFcn = createCallbackFcn(app,
@NumberofInputsGenSpinnerValueChanged, true);
            app.NumberofInputsGenSpinner.Position = [130 369 100 22];
            app.NumberofInputsGenSpinner.Value = 1;

            % Create NumberofLevelsSpinnerLabel
            app.NumberofLevelsSpinnerLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.NumberofLevelsSpinnerLabel.HorizontalAlignment = 'right';
            app.NumberofLevelsSpinnerLabel.Position = [17 331 100 22];
            app.NumberofLevelsSpinnerLabel.Text = 'Number of Levels';

            % Create NumberofLevelsSpinner
            app.NumberofLevelsSpinner =
```

```
uispinner(app.Step2GeneratingdatafromEquationPanel);
            app.NumberofLevelsSpinner.Limits = [1 100];
            app.NumberofLevelsSpinner.Position = [132 331 100 22];
            app.NumberofLevelsSpinner.Value = 1;

            % Create Beginningofx1EditFieldLabel
            app.Beginningofx1EditFieldLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Beginningofx1EditFieldLabel.HorizontalAlignment = 'right';
            app.Beginningofx1EditFieldLabel.Position = [17 280 88 22];
            app.Beginningofx1EditFieldLabel.Text = 'Beginning of x1';

            % Create Beginningofx1EditField
            app.Beginningofx1EditField =
uieditfield(app.Step2GeneratingdatafromEquationPanel, 'numeric');
            app.Beginningofx1EditField.Position = [119 281 38 22];

            % Create Endofx1EditFieldLabel
            app.Endofx1EditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Endofx1EditFieldLabel.HorizontalAlignment = 'right';
            app.Endofx1EditFieldLabel.Position = [177 280 56 22];
            app.Endofx1EditFieldLabel.Text = 'End of x1';

            % Create Endofx1EditField
            app.Endofx1EditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'numeric');
            app.Endofx1EditField.Position = [247 280 38 22];
            app.Endofx1EditField.Value = 1;

            % Create Beginningofx2EditFieldLabel
            app.Beginningofx2EditFieldLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Beginningofx2EditFieldLabel.HorizontalAlignment = 'right';
            app.Beginningofx2EditFieldLabel.Position = [17 238 88 22];
            app.Beginningofx2EditFieldLabel.Text = 'Beginning of x2';

            % Create Beginningofx2EditField
            app.Beginningofx2EditField =
uieditfield(app.Step2GeneratingdatafromEquationPanel, 'numeric');
            app.Beginningofx2EditField.Editable = 'off';
            app.Beginningofx2EditField.Position = [119 238 38 22];

            % Create Endofx2EditFieldLabel
            app.Endofx2EditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Endofx2EditFieldLabel.HorizontalAlignment = 'right';
            app.Endofx2EditFieldLabel.Position = [177 238 56 22];
            app.Endofx2EditFieldLabel.Text = 'End of x2';

            % Create Endofx2EditField
            app.Endofx2EditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'numeric');
            app.Endofx2EditField.Editable = 'off';
            app.Endofx2EditField.Position = [247 237 38 22];
            app.Endofx2EditField.Value = 1;

            % Create Beginningofx3EditFieldLabel
            app.Beginningofx3EditFieldLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
```

```
            app.Beginningofx3EditFieldLabel.HorizontalAlignment = 'right';
            app.Beginningofx3EditFieldLabel.Position = [17 196 88 22];
            app.Beginningofx3EditFieldLabel.Text = 'Beginning of x3';

            % Create Beginningofx3EditField
            app.Beginningofx3EditField =
uieditfield(app.Step2GeneratingdatafromEquationPanel, 'numeric');
            app.Beginningofx3EditField.Editable = 'off';
            app.Beginningofx3EditField.Position = [119 196 38 22];

            % Create Endofx3EditFieldLabel
            app.Endofx3EditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Endofx3EditFieldLabel.HorizontalAlignment = 'right';
            app.Endofx3EditFieldLabel.Position = [177 195 56 22];
            app.Endofx3EditFieldLabel.Text = 'End of x3';

            % Create Endofx3EditField
            app.Endofx3EditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'numeric');
            app.Endofx3EditField.Editable = 'off';
            app.Endofx3EditField.Position = [247 195 38 22];
            app.Endofx3EditField.Value = 1;

            % Create Beginningofx4EditFieldLabel
            app.Beginningofx4EditFieldLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Beginningofx4EditFieldLabel.HorizontalAlignment = 'right';
            app.Beginningofx4EditFieldLabel.Position = [17 154 88 22];
            app.Beginningofx4EditFieldLabel.Text = 'Beginning of x4';

            % Create Beginningofx4EditField
            app.Beginningofx4EditField =
uieditfield(app.Step2GeneratingdatafromEquationPanel, 'numeric');
            app.Beginningofx4EditField.Editable = 'off';
            app.Beginningofx4EditField.Position = [119 154 38 22];

            % Create Endofx4EditFieldLabel
            app.Endofx4EditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Endofx4EditFieldLabel.HorizontalAlignment = 'right';
            app.Endofx4EditFieldLabel.Position = [177 153 56 22];
            app.Endofx4EditFieldLabel.Text = 'End of x4';

            % Create Endofx4EditField
            app.Endofx4EditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'numeric');
            app.Endofx4EditField.Editable = 'off';
            app.Endofx4EditField.Position = [247 153 38 22];
            app.Endofx4EditField.Value = 1;

            % Create Beginningofx5EditFieldLabel
            app.Beginningofx5EditFieldLabel =
uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Beginningofx5EditFieldLabel.HorizontalAlignment = 'right';
            app.Beginningofx5EditFieldLabel.Position = [17 112 88 22];
            app.Beginningofx5EditFieldLabel.Text = 'Beginning of x5';

            % Create Beginningofx5EditField
            app.Beginningofx5EditField =
```

```
uieditfield(app.Step2GeneratingdatafromEquationPanel, 'numeric');
            app.Beginningofx5EditField.Editable = 'off';
            app.Beginningofx5EditField.Position = [119 112 39 22];

            % Create Endofx5EditFieldLabel
            app.Endofx5EditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.Endofx5EditFieldLabel.HorizontalAlignment = 'right';
            app.Endofx5EditFieldLabel.Position = [177 111 56 22];
            app.Endofx5EditFieldLabel.Text = 'End of x5';

            % Create Endofx5EditField
            app.Endofx5EditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'numeric');
            app.Endofx5EditField.Editable = 'off';
            app.Endofx5EditField.Position = [247 111 38 22];
            app.Endofx5EditField.Value = 1;

            % Create EquationEditFieldLabel
            app.EquationEditFieldLabel = uilabel(app.Step2GeneratingdatafromEquationPanel);
            app.EquationEditFieldLabel.HorizontalAlignment = 'right';
            app.EquationEditFieldLabel.Position = [19 65 53 22];
            app.EquationEditFieldLabel.Text = 'Equation';

            % Create EquationEditField
            app.EquationEditField = uieditfield(app.Step2GeneratingdatafromEquationPanel,
'text');
            app.EquationEditField.ValueChangedFcn = createCallbackFcn(app,
@EquationEditFieldValueChanged, true);
            app.EquationEditField.Position = [87 63 229 24];

            % Create ResetButton_Gen
            app.ResetButton_Gen = uibutton(app.Step2GeneratingdatafromEquationPanel,
'push');
            app.ResetButton_Gen.ButtonPushedFcn = createCallbackFcn(app,
@ResetButton_GenPushed, true);
            app.ResetButton_Gen.Position = [17 9 100 22];
            app.ResetButton_Gen.Text = 'Reset';

            % Create DoneButton_Gen
            app.DoneButton_Gen = uibutton(app.Step2GeneratingdatafromEquationPanel,
'push');
            app.DoneButton_Gen.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_GenPushed, true);
            app.DoneButton_Gen.Position = [199 10 100 22];
            app.DoneButton_Gen.Text = 'Done!';

            % Create Step2ImportdatafromworkspacePanel
            app.Step2ImportdatafromworkspacePanel = uipanel(app.InputTab);
            app.Step2ImportdatafromworkspacePanel.Title = 'Step2: Import data from
workspace';
            app.Step2ImportdatafromworkspacePanel.Visible = 'off';
            app.Step2ImportdatafromworkspacePanel.FontWeight = 'bold';
            app.Step2ImportdatafromworkspacePanel.Position = [12 15 332 428];

            % Create NumberofInputsSpinner_2Label
            app.NumberofInputsSpinner_2Label =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.NumberofInputsSpinner_2Label.HorizontalAlignment = 'right';
```

```
            app.NumberofInputsSpinner_2Label.Position = [42 366 98 22];
            app.NumberofInputsSpinner_2Label.Text = 'Number of Inputs';

            % Create NumberofInputsWSSpinner_2
            app.NumberofInputsWSSpinner_2 =
uispinner(app.Step2ImportdatafromworkspacePanel);
            app.NumberofInputsWSSpinner_2.Limits = [1 5];
            app.NumberofInputsWSSpinner_2.ValueChangedFcn = createCallbackFcn(app,
@NumberofInputsWSSpinner_2ValueChanged, true);
            app.NumberofInputsWSSpinner_2.Position = [155 366 100 22];
            app.NumberofInputsWSSpinner_2.Value = 1;

            % Create InputNumber1x1DropDownLabel
            app.InputNumber1x1DropDownLabel =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber1x1DropDownLabel.HorizontalAlignment = 'right';
            app.InputNumber1x1DropDownLabel.Position = [42 304 112 22];
            app.InputNumber1x1DropDownLabel.Text = 'Input Number 1 (x1)';

            % Create InputNumber1x1DropDown
            app.InputNumber1x1DropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber1x1DropDown.Items = {'Option 1', 'Option 2', 'Option 3', 'Option
4', 'Option 5', 'Option 6', 'Option 7', 'Option 8', 'Option 9', 'Option 10'};
            app.InputNumber1x1DropDown.Editable = 'on';
            app.InputNumber1x1DropDown.BackgroundColor = [1 1 1];
            app.InputNumber1x1DropDown.Position = [169 304 100 22];

            % Create InputNumber2x2DropDownLabel
            app.InputNumber2x2DropDownLabel =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber2x2DropDownLabel.HorizontalAlignment = 'right';
            app.InputNumber2x2DropDownLabel.Enable = 'off';
            app.InputNumber2x2DropDownLabel.Position = [42 262 112 22];
            app.InputNumber2x2DropDownLabel.Text = 'Input Number 2 (x2)';

            % Create InputNumber2x2DropDown
            app.InputNumber2x2DropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber2x2DropDown.Items = {'Option 1', 'Option 2', 'Option 3', 'Option
4', 'Option 5', 'Option 6', 'Option 7'};
            app.InputNumber2x2DropDown.Editable = 'on';
            app.InputNumber2x2DropDown.Enable = 'off';
            app.InputNumber2x2DropDown.BackgroundColor = [1 1 1];
            app.InputNumber2x2DropDown.Position = [169 262 100 22];

            % Create InputNumber3x3DropDownLabel
            app.InputNumber3x3DropDownLabel =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber3x3DropDownLabel.HorizontalAlignment = 'right';
            app.InputNumber3x3DropDownLabel.Enable = 'off';
            app.InputNumber3x3DropDownLabel.Position = [42 220 112 22];
            app.InputNumber3x3DropDownLabel.Text = 'Input Number 3 (x3)';

            % Create InputNumber3x3DropDown
            app.InputNumber3x3DropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber3x3DropDown.Editable = 'on';
            app.InputNumber3x3DropDown.Enable = 'off';
            app.InputNumber3x3DropDown.BackgroundColor = [1 1 1];
            app.InputNumber3x3DropDown.Position = [169 220 100 22];
```

```
            % Create InputNumber4x4DropDownLabel
            app.InputNumber4x4DropDownLabel =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber4x4DropDownLabel.HorizontalAlignment = 'right';
            app.InputNumber4x4DropDownLabel.Enable = 'off';
            app.InputNumber4x4DropDownLabel.Position = [42 178 112 22];
            app.InputNumber4x4DropDownLabel.Text = 'Input Number 4 (x4)';

            % Create InputNumber4x4DropDown
            app.InputNumber4x4DropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber4x4DropDown.Editable = 'on';
            app.InputNumber4x4DropDown.Enable = 'off';
            app.InputNumber4x4DropDown.BackgroundColor = [1 1 1];
            app.InputNumber4x4DropDown.Position = [169 178 100 22];

            % Create InputNumber5x5DropDownLabel
            app.InputNumber5x5DropDownLabel =
uilabel(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber5x5DropDownLabel.HorizontalAlignment = 'right';
            app.InputNumber5x5DropDownLabel.Enable = 'off';
            app.InputNumber5x5DropDownLabel.Position = [42 136 112 22];
            app.InputNumber5x5DropDownLabel.Text = 'Input Number 5 (x5)';

            % Create InputNumber5x5DropDown
            app.InputNumber5x5DropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.InputNumber5x5DropDown.Editable = 'on';
            app.InputNumber5x5DropDown.Enable = 'off';
            app.InputNumber5x5DropDown.BackgroundColor = [1 1 1];
            app.InputNumber5x5DropDown.Position = [169 136 100 22];

            % Create OutputYDropDownLabel
            app.OutputYDropDownLabel = uilabel(app.Step2ImportdatafromworkspacePanel);
            app.OutputYDropDownLabel.HorizontalAlignment = 'right';
            app.OutputYDropDownLabel.Position = [93 74 61 22];
            app.OutputYDropDownLabel.Text = 'Output (Y)';

            % Create OutputYDropDown
            app.OutputYDropDown = uidropdown(app.Step2ImportdatafromworkspacePanel);
            app.OutputYDropDown.Position = [169 74 100 22];

            % Create ResetButton_WS
            app.ResetButton_WS = uibutton(app.Step2ImportdatafromworkspacePanel, 'push');
            app.ResetButton_WS.ButtonPushedFcn = createCallbackFcn(app,
@ResetButton_WSPushed, true);
            app.ResetButton_WS.Position = [42 23 100 22];
            app.ResetButton_WS.Text = 'Reset';

            % Create DoneButton_WS
            app.DoneButton_WS = uibutton(app.Step2ImportdatafromworkspacePanel, 'push');
            app.DoneButton_WS.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_WSPushed, true);
            app.DoneButton_WS.Position = [217 24 100 22];
            app.DoneButton_WS.Text = 'Done!';

            % Create RegressionTab
            app.RegressionTab = uitab(app.TabGroup);
            app.RegressionTab.Title = 'Regression';
```

```
            % Create NBText
            app.NBText = uilabel(app.RegressionTab);
            app.NBText.FontWeight = 'bold';
            app.NBText.Position = [9 532 324 37];
            app.NBText.Text = {'NB: The options in this tab are only availabe in case of ';
'maximum two inputs'};

            % Create Regression1InputPanel
            app.Regression1InputPanel = uipanel(app.RegressionTab);
            app.Regression1InputPanel.Visible = 'off';
            app.Regression1InputPanel.Position = [10 10 336 523];

            % Create polynomialdegreeSpinnerLabel
            app.polynomialdegreeSpinnerLabel = uilabel(app.Regression1InputPanel);
            app.polynomialdegreeSpinnerLabel.HorizontalAlignment = 'right';
            app.polynomialdegreeSpinnerLabel.Position = [10 468 104 22];
            app.polynomialdegreeSpinnerLabel.Text = 'polynomial degree';

            % Create polynomialdegreeSpinner1
            app.polynomialdegreeSpinner1 = uispinner(app.Regression1InputPanel);
            app.polynomialdegreeSpinner1.Limits = [1 9];
            app.polynomialdegreeSpinner1.Position = [129 468 100 22];
            app.polynomialdegreeSpinner1.Value = 1;

            % Create RobustDropDownLabel
            app.RobustDropDownLabel = uilabel(app.Regression1InputPanel);
            app.RobustDropDownLabel.HorizontalAlignment = 'right';
            app.RobustDropDownLabel.Position = [10 426 44 22];
            app.RobustDropDownLabel.Text = 'Robust';

            % Create RobustDropDown
            app.RobustDropDown = uidropdown(app.Regression1InputPanel);
            app.RobustDropDown.Items = {'None', 'LAR', 'Bisquare', ''};
            app.RobustDropDown.Position = [129 426 100 22];
            app.RobustDropDown.Value = 'None';

            % Create DoneButton_Reg1
            app.DoneButton_Reg1 = uibutton(app.Regression1InputPanel, 'push');
            app.DoneButton_Reg1.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_Reg1Pushed, true);
            app.DoneButton_Reg1.Position = [215 364 100 22];
            app.DoneButton_Reg1.Text = 'Done!';

            % Create ResetButton_Reg1
            app.ResetButton_Reg1 = uibutton(app.Regression1InputPanel, 'push');
            app.ResetButton_Reg1.ButtonPushedFcn = createCallbackFcn(app,
@ResetButton_Reg1Pushed, true);
            app.ResetButton_Reg1.Position = [10 364 100 22];
            app.ResetButton_Reg1.Text = 'Reset';

            % Create Result_Reg1
            app.Result_Reg1 = uitextarea(app.Regression1InputPanel);
            app.Result_Reg1.ValueChangedFcn = createCallbackFcn(app,
@Result_Reg1ValueChanged, true);
            app.Result_Reg1.Editable = 'off';
            app.Result_Reg1.Position = [10 14 315 326];
```

```
            % Create Regression2InputPanel
            app.Regression2InputPanel = uipanel(app.RegressionTab);
            app.Regression2InputPanel.Visible = 'off';
            app.Regression2InputPanel.Position = [11 10 336 523];

            % Create polynomialdegreex1Label
            app.polynomialdegreex1Label = uilabel(app.Regression2InputPanel);
            app.polynomialdegreex1Label.HorizontalAlignment = 'right';
            app.polynomialdegreex1Label.Position = [10 489 120 22];
            app.polynomialdegreex1Label.Text = 'polynomial degree x1';

            % Create polynomialdegreeSpinnerx1
            app.polynomialdegreeSpinnerx1 = uispinner(app.Regression2InputPanel);
            app.polynomialdegreeSpinnerx1.Limits = [1 5];
            app.polynomialdegreeSpinnerx1.Position = [145 489 100 22];
            app.polynomialdegreeSpinnerx1.Value = 1;

            % Create RobustDropDown_2Label
            app.RobustDropDown_2Label = uilabel(app.Regression2InputPanel);
            app.RobustDropDown_2Label.HorizontalAlignment = 'right';
            app.RobustDropDown_2Label.Position = [10 405 44 22];
            app.RobustDropDown_2Label.Text = 'Robust';

            % Create RobustDropDown2
            app.RobustDropDown2 = uidropdown(app.Regression2InputPanel);
            app.RobustDropDown2.Items = {'None', 'LAR', 'Bisquare', ''};
            app.RobustDropDown2.Position = [145 405 100 22];
            app.RobustDropDown2.Value = 'None';

            % Create DoneButton_Reg2
            app.DoneButton_Reg2 = uibutton(app.Regression2InputPanel, 'push');
            app.DoneButton_Reg2.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_Reg2Pushed, true);
            app.DoneButton_Reg2.Position = [215 364 100 22];
            app.DoneButton_Reg2.Text = 'Done!';

            % Create ResetButton_Reg2
            app.ResetButton_Reg2 = uibutton(app.Regression2InputPanel, 'push');
            app.ResetButton_Reg2.ButtonPushedFcn = createCallbackFcn(app,
@ResetButton_Reg2Pushed, true);
            app.ResetButton_Reg2.Position = [10 364 100 22];
            app.ResetButton_Reg2.Text = 'Reset';

            % Create Result_Reg2
            app.Result_Reg2 = uitextarea(app.Regression2InputPanel);
            app.Result_Reg2.Editable = 'off';
            app.Result_Reg2.Position = [10 14 315 326];

            % Create polynomialdegreex2Label
            app.polynomialdegreex2Label = uilabel(app.Regression2InputPanel);
            app.polynomialdegreex2Label.HorizontalAlignment = 'right';
            app.polynomialdegreex2Label.Position = [10 447 120 22];
            app.polynomialdegreex2Label.Text = 'polynomial degree x2';

            % Create polynomialdegreeSpinnerx2
            app.polynomialdegreeSpinnerx2 = uispinner(app.Regression2InputPanel);
            app.polynomialdegreeSpinnerx2.Limits = [1 5];
            app.polynomialdegreeSpinnerx2.Position = [145 447 100 22];
```

```
            app.polynomialdegreeSpinnerx2.Value = 1;

            % Create RandomForrestTab
            app.RandomForrestTab = uitab(app.TabGroup);
            app.RandomForrestTab.Title = 'Random Forrest';

            % Create RandomForrestPanel
            app.RandomForrestPanel = uipanel(app.RandomForrestTab);
            app.RandomForrestPanel.Visible = 'off';
            app.RandomForrestPanel.Position = [15 278 326 267];

            % Create ResetButton_RF
            app.ResetButton_RF = uibutton(app.RandomForrestPanel, 'push');
            app.ResetButton_RF.ButtonPushedFcn = createCallbackFcn(app,
@ResetButton_RFPushed, true);
            app.ResetButton_RF.Position = [18 172 100 22];
            app.ResetButton_RF.Text = 'Reset';

            % Create DoneButton_RF
            app.DoneButton_RF = uibutton(app.RandomForrestPanel, 'push');
            app.DoneButton_RF.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_RFPushed, true);
            app.DoneButton_RF.Position = [207 172 100 22];
            app.DoneButton_RF.Text = 'Done';

            % Create Result_RF
            app.Result_RF = uitextarea(app.RandomForrestPanel);
            app.Result_RF.Editable = 'off';
            app.Result_RF.Position = [18 1 289 123];

            % Create NumberofthebagsEditFieldLabel
            app.NumberofthebagsEditFieldLabel = uilabel(app.RandomForrestPanel);
            app.NumberofthebagsEditFieldLabel.HorizontalAlignment = 'right';
            app.NumberofthebagsEditFieldLabel.Position = [26 231 111 22];
            app.NumberofthebagsEditFieldLabel.Text = 'Number of the bags';

            % Create NumberofthebagsEditField
            app.NumberofthebagsEditField = uieditfield(app.RandomForrestPanel, 'numeric');
            app.NumberofthebagsEditField.Limits = [1 Inf];
            app.NumberofthebagsEditField.Position = [152 231 100 22];
            app.NumberofthebagsEditField.Value = 1;

            % Create NeuralNetworkTab
            app.NeuralNetworkTab = uitab(app.TabGroup);
            app.NeuralNetworkTab.Title = 'Neural Network';

            % Create NNPanel
            app.NNPanel = uipanel(app.NeuralNetworkTab);
            app.NNPanel.BorderType = 'none';
            app.NNPanel.Position = [21 210 314 355];

            % Create NumberofNeuronsEditFieldLabel
            app.NumberofNeuronsEditFieldLabel = uilabel(app.NNPanel);
            app.NumberofNeuronsEditFieldLabel.HorizontalAlignment = 'right';
            app.NumberofNeuronsEditFieldLabel.Position = [25 293 110 22];
            app.NumberofNeuronsEditFieldLabel.Text = 'Number of Neurons';

            % Create NumberofNeuronsEditField
```

```
            app.NumberofNeuronsEditField = uieditfield(app.NNPanel, 'numeric');
            app.NumberofNeuronsEditField.Limits = [1 Inf];
            app.NumberofNeuronsEditField.Position = [150 293 100 22];
            app.NumberofNeuronsEditField.Value = 1;

            % Create ResetButton_NN
            app.ResetButton_NN = uibutton(app.NNPanel, 'push');
            app.ResetButton_NN.Position = [26 227 100 22];
            app.ResetButton_NN.Text = 'Reset';

            % Create DoneButton_NN
            app.DoneButton_NN = uibutton(app.NNPanel, 'push');
            app.DoneButton_NN.ButtonPushedFcn = createCallbackFcn(app,
@DoneButton_NNPushed, true);
            app.DoneButton_NN.Position = [193 227 100 22];
            app.DoneButton_NN.Text = 'Done';

            % Create Results_NN
            app.Results_NN = uitextarea(app.NNPanel);
            app.Results_NN.Editable = 'off';
            app.Results_NN.Position = [26 20 267 169];

            % Create LearningrateLabel
            app.LearningrateLabel = uilabel(app.NNPanel);
            app.LearningrateLabel.HorizontalAlignment = 'right';
            app.LearningrateLabel.Position = [59 262 76 22];
            app.LearningrateLabel.Text = 'Learning rate';

            % Create LearningRate_NN
            app.LearningRate_NN = uieditfield(app.NNPanel, 'numeric');
            app.LearningRate_NN.Limits = [0 1];
            app.LearningRate_NN.Position = [150 262 100 22];
            app.LearningRate_NN.Value = 0.05;

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = GUI_Reyhaneh_Joodattabrizi_exported

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            % Execute the startup function
            runStartupFcn(app, @startupFcn)

            if nargout == 0
                clear app
            end
        end
```

```
        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Published with MATLAB® R2019b