University of Piraeus
Department of Digital Systems
Piraeus, Greece

A Comparative Evaluation of Machine Learning Algorithms: Binary Classification on Medical Data

A master's thesis for the postgraduate programme

"Digital Systems and Services: Big Data and Analytics"

Panagiotis Goumenakis

ME1709

Supervisor
Prof. Dr. Andriana Prentza

Piraeus, September 2019

Abstract

Nowadays Machine Learning (ML) has been well applied and recognised as an effective tool to handle a wide range of real situations, including medical implementations. As the amount of data in the field of healthcare grows year by year, there is a remarkable development in disease forecasting with the help of ML applications. From the prediction of epidemic outburst and several diseases to contributing with better means of labelling and storing healthcare data, implementation of ML in the field of healthcare indicates accurate results.

This thesis focuses mainly on two major aspects of ML areas. Firstly, on analysing a medical dataset providing visualisations together with invaluable information on dataset's variables. Secondly, it emphasizes on implementing the appropriate algorithms to execute binary classification in order to determine whether a person is labelled as infected or not infected based on feature values of the sample set. Choosing the most suitable approach is crucial as it could potentially improve the clinical decisions as well as patients' survival time when applied to real world problems.

The research is based on the mesothelioma disease dataset, allocated on the UCI repository, containing 324 examples with 35 attributes. Regarding the unsupervised learning part, in order to deduct results and conclusions, various ML classification algorithms are used to perform the analysis such as Decision Tree, Support Vector Machines (SVM), Naive Bayes Classifier, Logistic Regression, k Nearest Neighbours (kNN), and Artificial Neural Networks (ANN).

Concerning the techniques for evaluation, the reader can expect several methods as for example statistical measures like accuracy, sensitivity, specificity, f1-score, confusion matrix, AUC (Area Under Curve), and ROC (Receiver Operating Characteristic) curve.

Keywords - *Machine Learning, Binary Classification, SVM, Naive Bayes, Decision Tree, Logistic Regression, ANN, Mesothelioma Dataset, UCI*

Περίληψη

Στις μέρες μας ο τομέας της μηχανικής μάθησης έχει εφαρμοστεί και αναγνωριστεί ως ένα αποτελεσματικό εργαλείο που μπορεί να διαχειριστεί ένα ευρύ φάσμα πραγματικών καταστάσεων συμπεριλαμβανομένων και αυτών των ιατρικών εφαρμογών. Καθώς ο όγκος των δεδομένων στον τομέα της υγείας αυξάνεται χρόνο με το χρόνο, η εξέλιξη της πρόγνωσης μιας νόσου με τη χρήση εφαρμογών της μηχανικής μάθησης είναι αξιοσημείωτη. Οι εφαρμογές ακόμα της μηχανικής μάθησης στον τομέα της υγείας παρουσιάζουν ακριβή αποτελέσματα τόσο στην πρόβλεψη μίας επιδημίας ή διαφόρων ασθενειών όσο και στη συνεισφορά της βελτίωσης των τρόπων με τους οποίους σημειώνονται και αποθηκεύονται τα ιατρικά δεδομένα.

Αυτή η διπλωματική εργασία δίνει έμφαση αρχικά στην ανάλυση ιατρικών δεδομένων παρουσιάζοντας οπτικοποιήσεις αλλά και μετρικές σχετικά με τις πληροφορίες που παρουσιάζουν τα δεδομένα. Έπειτα, επικεντρώνεται στην υλοποίηση των κατάλληλων αλγορίθμων ικανών να ταξινομήσουν τα δεδομένα με σκοπό να καθορίσουν εάν ένας άνθρωπος έχει προσβληθεί από τη νόσο ή όχι. Η επιλογή της καταλληλότερης μεθόδου κρίνεται ως καθοριστικής σημασίας καθώς η εφαρμογή της σε πραγματικές καταστάσεις θα μπορούσε ενδεχομένως να βελτιώσει τόσο τις κλινικές αποφάσεις όσο και το προσδόκιμο ζωής του ασθενή.

Η συγκεκριμένη έρευνα βασίζεται στο σύνολο δεδομένων "Νόσος Μεσοθηλίωμα" που βρίσκεται στην αποθήκη συνόλων δεδομένων UCI και περιέχει 324 παρατηρήσεις με 35 χαρακτηριστικά. Σχετικά με τον τομέα της ανάλυσης που ασχολείται με τη μη επιβλεπόμενη μάθηση χρησιμοποιούνται αλγόριθμοι μηχανικής μάθησης για κατηγοριοποίηση όπως Δέντρα Απόφασης (Decision Trees), Μηχανές Διανυσμάτων Στήριξης (SVM), Λογιστική Παλινδρόμηση (Logistic Regression), k Πλησιέστεροι Γείτονες (kNN) και Νευρωνικά Δίκτυα (ANN) με σκοπό να ολοκληρωθεί η ανάλυση και να οδηγηθεί κανείς σε αποτελέσματα και συμπεράσματα.

Όσον αφορά στις τεχνικές αξιολόγησης ο αναγνώστης μπορεί να περιμένει μεθόδους όπως για παράδειγμα τις στατιστικές μετρικές ακρίβεια (accuracy), ευαισθησία (sensitivity) και προσδιοριστικότητα (specificity), f1-score, την μήτρα σύγχυσης (confusion matrix) και τη χαρακτηριστική καμπύλη λειτουργίας (AUC/ROC).

Table of Contents

List of Figures

List of Tables

Acronyms

ALP	Alkaline Phosphatise
ANN	Artificial Neural Network
AUC	Area Under Curve
CART	Classification and Regression Tree
CHAID	Chi-Square Automatic Interaction Detection
DT	Decision Tree
EDA	Exploratory Data Analysis
EM	Expectation Maximisation
FN	False Negative
FP	False Positive
ID3	Iterative Dichotomiser 3
KDD	Knowledge Discovery in Databases
kNN	k Nearest Neighbours
LDH	Lactic Dehydrogenise
LR	Logistic Regression
MAR	Missing at Random
MCAR	Missing Completely at Random
ML	Machine Learning
MLE	Maximum Likelihood Estimator
MLP	Multilayer Perceptron
MNAR	Missing not at Random
NB	Naïve Bayes
PCA	Principal Component Analysis
PLT	Platelet Count
PNN	Probabilistic Neural Network
RF	Random Forest
ROC	Receiver Operating Characteristic
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
UCI	University of California Irvine

Chapter 1

Introduction

1.1 Motivation

During the last decades ML has been an emerging research field reaching remarkably successful results not only in medicine but in many other applications as well. So far, practitioners run procedures which inevitably include subjectivity as well as plenty of time to be carried out. But with the usage of ML applications, these procedures can be automated and simplified. In that way, the use of ML solutions creates repeatable results free of bias and it also reduces time consumption to a minimum level leaving practitioners more time for their numerous duties. Moreover, excessive number of features in medical datasets together with usually small samples size makes advanced ML of critical importance for clinical interpretation and analysis.

Typically, it requires many experts from various areas to scrutinise and interpret pathological conditions. There are times however, when experts are not enough, or they contradict each other, making decision making process even harder. Hence, in a field which involves people with a great amount of competence and attentiveness, ML techniques can assist in improving diagnostic accuracy, standardization among clinicians, and it aids in developing computer-based appliances to model experts' methods.

Furthermore, as we live in the Big Data world of late, rapid increase in data volumes available for analysis is so obvious on healthcare as in any other field. Insurance companies, research groups and laboratories, healthcare suppliers, government agencies and especially hospitals can produce voluminous digital data as never seen before. This is the reason why ML approaches should step in to manipulate the data concurrently and give light to issues like compact or unlabelled data with which scientists and researchers had been struggling over the years to resolve.

ML consists of many different areas that could be adequately used to serve healthcare. Some of these include regression, classification, clustering and dimensionality reduction. Clustering belongs to a branch of ML which is called unsupervised learning and it deals with situations when the target is to group together examples of the given dataset that exhibit some kind of similarity. This method is particularly effective when for example the different types of patients that lie in the dataset are missing. Next, regression involves the procedure of providing numerical predictions for features by considering the rest of the dimensions for the specific sample. Such models are systematically

used in financial forecasting, trend analysis, marketing or even drug response modelling. In addition, with dimensionality reduction patients' significant characteristics can be extracted by implementing several methods and thus lead the applied model to more accurate results.

Even if the techniques explained above are widely chosen from analysts and researchers, they are beyond the scope of this work and should be omitted but not get past. Yet, one of the most famous and useful areas of supervised learning that will be the main axis of this thesis is classification, which takes into account features of an example to predict a discrete class variable. It has an immediate effect on daily life with an example of that being the problem of separating people who are infected with a virus from those that are not infected.

Based on this background, the goal of this thesis is to specify, implement and finally to compare the outcomes of the most applicable ML classification algorithms. These algorithms should detect a healthy/unhealthy person or a malignant/benign tumour on a patient. Thus, lastly the ML algorithm is to assign the correct label to each sample and reach high performance evaluation measures.

1.2 Related Work

Over the last few years, ML implementations have been notably popular on different domains like social media services, online customer support, fraud detection etc. Hence, healthcare sector could not have been an exception to the above rule. There has heretofore been a numerous amount of surveys, experiment papers, theses, and reports published concerning the applications of ML in medicine. From binary or multiclass classification to complicated methods for labelling patients' unstructured records, ML procedures have been of great interest for analysts.

Nonetheless, concerning the mesothelioma dataset selected for this research, various researches have been carried out executing binary classification. Regular Naive Bayes (NB), Logistic Regression (LR), Decision Trees (DT) and even more complex algorithms such as Logic Learning Machine are some of the techniques that have been used to achieve fascinating results. One could distinguish some of them though for their innovative and compelling perspective.

In central Cappadocia, Turkey, mesothelioma was causing 50% of all deaths and thus in recent years this could not have been ignored by local scientists to examine. Indeed, mesothelioma disease diagnosis was conducted using Artificial Intelligence Methods by Hamza Osman Ilhan and Enes Celik [1]. There, methods like Support Vector Machine (SVM), DT, Artificial Neural Network (ANN) and Ensemble Learning for combining the above had been used to classify the data with an accuracy of 100% even reached. Linear Kernel SVM and Multilayer Perceptron (MLP) scored high accuracy but Linear SVM would be able to still perform well in case of data been generalised as authors state.

Additionally, Orhan Er, Tanrikulu, Abakay and Temurtas [2] approached the problem with 3 different ANNs to detect malignant tumours in patients. Probabilistic Neural Network (PNN) happened to succeed the best results compared to Learning Vector Quantization NN and Multilayer NN. With a usage of a 3-fold cross-validation technique, which performed better than other

conventional validation algorithms, PNN managed a classification accuracy score of 96.3%. According to the writers all 3 of the algorithms used in the analysis were capable and most of all safe to satisfactorily classify the data. Nevertheless, PNN was outperformed by Random Forest (RF) classifier that was examined by Chicco and Rovelli [3] in their paper. Except from the classifiers they also presented feature selection methods such as Mean Square Error decrease and Gini node impurity decrease to determine an even more correlated subset of attributes. These approaches pointed that the most important and relevant features were "lung side" alongside with "platelet count" (PLT). High point of the study was the success of RF to predict mesothelioma patients' diagnosis regardless of the choice made; either complete imbalanced dataset or the under-sampled balanced database.

Apart from these perspectives on classifying the mesothelioma malady database, another interesting survey handles the data with NB classifier [4]. Nilashi, Roudbaraki and Farahmand scrutinised an intelligent system in their paper which except from NB for classification it uses Expectation Maximisation (EM) for clustering. This algorithm concerns a simple and efficient iterative procedure in computing the Maximum Likelihood. After maximising likelihood via Akaike Information Criterion model, they applied a 10-fold cross validation technique to receive the most unbiased result and at the end 7 clusters. It appears that the combination of EM with NB obtained an overall accuracy of 93.21% outcompeting certainly all other algorithms.

Lastly a recent study from Xue Hu and Zebo Yu [5] from Chongqing Medical University, China has brought to light state-of-the-art feature selection methods as well as classification processes. An implemented diagnostic model based on Stacked Sparse Autoencoder algorithm was set up to detect malignant mesothelioma on patients. The latter together with a genetic algorithm exhibited the highest overall performance, namely accuracy, specificity, F-measure and AUC were all 100%. Moreover, these two demanded the smallest number of variables compared to other methods applied in the paper concluding that this approach could assist pathologists by providing them an optimal performance diagnostic system.

1.3 Objectives

This work was conducted for the purpose of fulfilling the postgraduate studies towards the programme "Digital Systems and Services: Big Data and Analytics" offered by the Department of Digital Systems and Services at the University of Piraeus, Piraeus Greece. It uses ML classification algorithms inside the branch of unsupervised learning and aims to:

- First, clarify and give inside to every concept one should familiarise himself/herself with so that all steps of the analysis from preprocessing phase to predictions stage are fully understood;
- Second, implement the most applicable algorithms on mesothelioma disease dataset to carry out binary classification;

- ➢ Last, perform an evaluation overview of the algorithms, providing their merits-demerits together with a comparative analysis on their performance rates.

1.4 Approach and Methodology

The analysis performed in this thesis concentrates mainly on the implementation of ML classification algorithms on medical data provided from the UCI Repository. Apart from that, emphasis is given as well to preprocessing phase by investigating metrics together with visualisations to extract necessary information regarding features of the dataset.

The outlined methodology of this master thesis is split into 3 phases, depending on the target of the analysis and it can be concisely summarised as follows:

At first, starting with the preprocessing phase of the study in which the mesothelioma dataset from the UCI ML repository is acquired and afterwards actions on preprocessing stage of the study are performed. Component wise, the methods included are outlier removal, scaling, label encoding, and feature selection/ extraction. Throughout these steps some plots assist the whole procedure in order to highlight the most valuable information.

Secondly, now that the data is ready to be fed into the training process it will be time to implement the most suitable ML classification algorithms. Having that in mind, it is fairly obvious in this phase that each algorithm comprises of several other substeps and perhaps different preprocessing techniques adapted to the requirements of the algorithm. This stage demands besides repeatedly implementations to optimise hyperparameters of the model and thus making the algorithm to achieve a better performance.

Finally, the last stage of this work plays a principal role. Here, statistical quantities are presented which encapsulate the success and effectiveness that the algorithms managed to reach. These metrics aggregated and combined with visualisations have the power to divulge a faulty model's fit. It is a necessary condition a combination of these quantities to be taken into account and the reason for this is that although for example an algorithm can predict correctly many target attribute values, these may not be of great importance. For instance, suppose a dataset has a binary target attribute of values "1" for a person who survives and "0" for a person that does not survive. Let's say furthermore that examples in total are ten, nine of them being "1" s and the last one "0" s. If an algorithm predicts successfully nine "1"-valued instances but fails to predict the "0"-valued instance, that is to say the patient whose life is in danger, then this algorithm is not efficient for the purpose. So, this chapter shall interpret interesting outcomes and their potential that will help to compare the selected ML algorithms.

1.5 Structure

For the remaining of this thesis in Chapter 2, all the necessary theoretical background is presented concerning preprocessing phase, classification algorithms insight and evaluation measures that surround the methods implemented later. Chapter 3 explains the mesothelioma disease dataset in addition to domain knowledge for the reader to get familiarised. Chapter 4 consists of the techniques used to classify the data presenting the associate details of the analysis. Carrying on with Chapter 5 where the results of each algorithm are described along with performance metrics and a comparative evaluation of the selected algorithms. At the end, Chapter 6 contains conclusions reached and possible future targets.

Chapter 2

Theoretical Background

This chapter concerns the presentation of the necessary theoretical background demanded in order to execute the process of the analysis in total. From preprocessing techniques to evaluation methods, this section shall provide concrete definitions, details and information for every phase. Specifically, the structure contains subchapters as:

- Machine Learning and Data Mining
- Knowledge Discovery in Databases (KDD)
- Preprocessing Techniques
- Machine Learning Algorithms
- Evaluation Methods

2.1 Machine Learning and Data Mining

In general, the term Machine Learning (ML) denotes the scientific study of algorithms and statistical models that computer systems to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It basically concerns the knowledge extraction from data, and it can be identified as the research field at the intersection of statistics, artificial intelligence and computer science. Outside of commercial applications, ML has had a tremendous influence on the way data driven research is done today [6]. Although one could argue that ML has been developed in recent years, the truth is that it has been around for decades in some specialised applications, such as Optical Character Recognition (OCR). But the first ML application that really became mainstream, improving the lives of hundreds of millions of people, took over the world back in the 1990s: it was the spam filter. It was followed by hundreds of ML applications that now quietly power hundreds of products and features that people use regularly, from better recommendations to voice search [7].

ML methods differ in their approach, the type of data they input and output, and the type of the task or problem they are intended to solve. Thus, ML algorithms can be categorised into 4 major groups depending on the above parameters and these are:

- *Supervised Learning*: it is used when the aim is to predict a certain outcome from a given input, and only examples of input-output pairs are given. An ML model from these input-output pairs is then constructed, which comprise the training set. The goal is to make accurate predictions to new, never seen before data. Supervised learning regularly requires human effort to build the training set, but afterwards automates and often speeds up an otherwise laborious or infeasible task.

- *Unsupervised Learning*: this type of learning subsumes all kinds of ML where there is no known output, no teacher to instruct the learning algorithm leaving the algorithm to find a structure in its input. *Unsupervised learning* can be a goal in itself discovering hidden patterns in data. Most common method of it is clustering which is a common technique for statistical data analysis and aims to separate data into groups according to similarities detected [8].

- *Semi-Supervised Learning*: it falls somewhere in between supervised and unsupervised learning, since they use both labelled and unlabelled data for training - typically a small amount of labelled data and a large amount of unlabelled data. The systems that use this method can considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labelled data requires skilled and relevant resources to train it / learn from it. Otherwise, acquiring unlabelled data generally doesn't require additional resources.

- *Reinforcement Learning*: this learning method interacts with its environment by producing actions and discovers errors or rewards. Trial/error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behaviour within a specific context in order to maximise its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as reinforcement signal.

Data mining on the other hand is a subset of ML or even better stated it is the process of discovering patterns in large datasets involving methods at the intersection of ML, statistics, and database systems. It has an overall goal to extract information with intelligent methods from a dataset and transform the information into comprehensible structure for future use. In data mining, association rules are if-then statements that help to show the probability of relationships between data items within large datasets in various databases. They have several applications and they are widely used to help discover

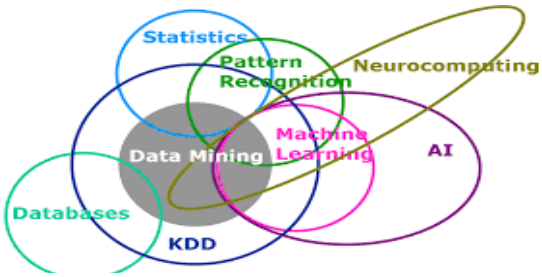

Figure 2.1 - Venn diagram: how AI intersects with other branches[1]

[1] SAS Institute 2014, and PwC, 2016 https://bit.ly/2tA5BT0

correlation in medical datasets among others. These association rules are created by analysing data for frequent if/then patterns, then using the support and confidence criteria to locate the most important relationships within data. Support is how frequently the items appear in the database, while confidence is the number of times if/then statements are accurate.

Data mining is the analysis step of the "knowledge discovery in databases" process or KDD. Aside from the raw analysis step, it also involves database and data management aspects, data preprocessing, model and inference considerations, interestingness metrics, complexity considerations, post processing of discovered structures, visualisations, and online updating.

2.2 Knowledge Discovery in Databases (KDD)

Knowledge discovery in databases (KDD) could be defined in various ways. One of them states that it is the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. This widely used data mining technique is a process that includes data preparation and selection, cleansing, incorporating prior knowledge on datasets and interpreting accurate solutions from the observed results. It is an active research area with promise for high payoffs in many business and scientific domains. The grand challenge of *KDD* is to automatically process large quantities of raw data, identify the most significant and meaningful patterns, and present these as knowledge appropriate for achieving the user's goals [9]. *KDD* process is commonly defined with the stages:

1. *Problem specification*, which comprises the design, the arrangement of the application domain, the relevant prior knowledge obtained by experts and the final objectives pursued by the end-user;

2. *Problem understanding,* that includes the comprehension of both the selected data to approach and the expert knowledge associated to achieve a high degree of reliability;

3. *Selection*, whose main role is to create a target dataset from the original data, i.e., selecting a subset of variables or data samples, on which discovery must be performed;

4. *Preprocessing,* which aims to clean data by performing various operations, such as noise modelling and removal, defining proper strategies for handling missing data fields, accounting for time-sequence information;

5. *Transformation,* that is in charge of reducing and projecting the data, in order to derive a representation suitable for the specific task to be performed; it is typically accomplished by involving transformation techniques or methods that are able to find invariant representations of the data;

6. *Data mining*, deals with extracting interesting patterns by choosing *(i)* a specific data-mining method or task, *(ii)* proper algorithm(s) for performing the task in hand, and *(iii)* an appropriate representation of the output results;

7. *Interpretation/evaluation,* is exploited by the user to interpret and extract knowledge from the mined patterns; this interpretation is typically carried out by visualising the patterns, models, or the data given such model and, in case, iteratively looking back at the previous steps of the process.

8. *Results exploitation*, the last stage may involve using the knowledge directly; incorporating the knowledge into another system for further processes or simply reporting discovered knowledge through visualization tools [10] [11].

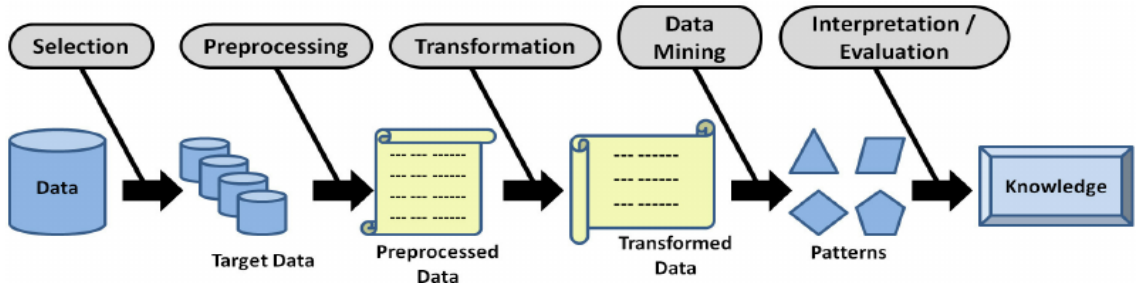

Figure 2.2 - The KDD process[2]

2.3 Preprocessing Techniques

In the data mining procedure data preprocessing is an important step for a successful algorithm implementation later. This data mining technique involves transforming raw data into an understandable format. It is highly likely that real-world data contains errors or is incomplete, inconsistent, and/or lacking in certain behaviours or trends. Data preprocessing is a proven method of resolving such issues. Some of the tasks that it is split together with the approaches used to handle them are the following:

- *Data cleaning*: it is the process of detecting and correcting (or removing) corrupt or inaccurate records from a dataset and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

[2] F. Gullo. From Patterns in Data to Knowledge Discovery: What Data Mining Can Do. 3rd International Conference Frontiers in Diagnostic Technologies, ICFDT3 2013

- *Missing data*: this situation arises when some data is missing for different reasons in the data. It can be solved by data imputation which uses several methods to fill these variables with some intuitive data. Three mechanisms of missing data are:

    - *Missing Completely at Random (MCAR)*, the propensity for a data point to be missing is completely random. It can be handled by substituting the missing value with the mean/median/mode of the feature.

    - *Missing At Random (MAR)*, when the probability of missing data on a variable is related to some other measured variable in the model, but not to the value of the variable with missing values itself. For instance, managers are more likely not to share their income if someone collects data on the profession of subject. Regression or classification techniques are used to handle this issue.

    - *Missing Not At Random (MNAR)*, the missing values on a variable are related to the values of that variable itself, even after controlling for other variables. For example, when data are missing on IQ and only the people with low IQ values have missing observations for this variable.

- *Noisy data*: meaningless data that can't be interpreted by machines. It can be generated due to faulty data collection, data entry errors etc. Handling methods include:

    - *Binning method*: this method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.

    - *Regression*: noisy data can be transformed to smooth by fitting a regression function to them. The regression function can be linear having a single independent variable or multiple otherwise.

    - *Clustering*: this approach groups similar data in a cluster. The outliers may be undetected, or it will fall outside the clusters.

- *Data transformation*: it is the procedure of converting data from one format or structure to another more suitable for mining process. It includes the following ways:

- *Normalisation*: it's used to scale the data values in a specified range [-1, 1] or [0,1] to more easily compare data from different places. Normalizing the data attempts to give all attributes equal weight and it is particularly useful in statistical learning methods.

- *Discretisation*: its main goal is to transform a set of continuous attributes into discrete ones, by associating categorical values to intervals and thus transforming quantitative data into qualitative data.

- *Concept hierarchy generation*: here attributes are converted from level to higher level in hierarchy. For Example, the attribute "city" can be converted to "country".

- *Data reduction*: it comprises the set of techniques that obtain a reduced representation of the original data. It aims to increase the storage efficiency and reduce data storage and analysis costs. The various steps to data reduction are:

  - *Instance selection*: consists of choosing a subset of the total available data to achieve the original purpose of the DM application as if the whole data had been used. It constitutes the family of oriented methods that perform in a somewhat intelligent way the choice of the best possible subset of examples from the original data by using some rules and/or heuristics.

  - *Attribute selection*: achieves the reduction of the dataset by removing irrelevant or redundant features (or dimensions). The goal of FS is to find a minimum set of attributes, such as the resulting probability distribution of the data output attributes, (or classes) is as close as possible to the original distribution obtained using all attributes.

  - *Dimensionality reduction*: this reduces the size of data by encoding mechanisms. It can be lossy or lossless. If after reconstruction from compressed data, the original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction. The two effective methods of dimensionality reduction are: Wavelet transforms and PCA (Principal Component Analysis) [12].

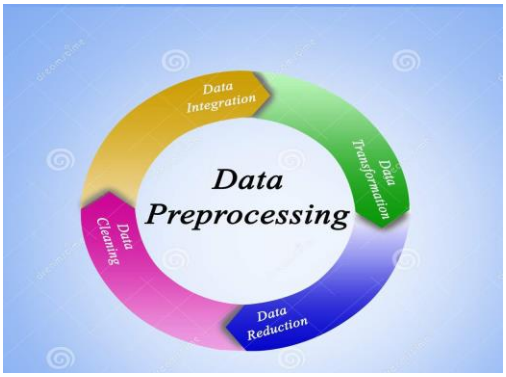

Figure 2.3 - The Data preprocessing circle [12].

2.4 Machine Learning Algorithms

The theory associated with each ML algorithm studied in this thesis is widely spread on the web. Nevertheless, the intuitive idea together with the necessary mathematical notation that stays behind those techniques are presented here. At the end of each subchapter that follows, a table with the most applicable advantages and disadvantages of each algorithm is presented.

2.4.1 Decision Tree

One of the most intuitive tools for data classification is the decision tree. It hierarchically partitions the input space until it reaches a subspace associated with a class label. Decision trees are appreciated for being easy to interpret and easy to use. They are enthusiastically used in a range of business, scientific, and health care applications because they provide an intuitive means of solving complex decision-making tasks. For example, in business, decision trees are used for everything from codifying how employees should deal with customer needs to making high-value investments. In medicine, decision trees are used for diagnosing illnesses and making treatment decisions for individuals or for communities.

A *decision tree* is a rooted, directed tree similar to a flowchart. Each interior node corresponds to a partitioning decision, and each leaf node is mapped to a class label prediction. To classify a data item, imagine the data item to be traversing the tree, beginning at the root. Each interior node is programmed with a splitting rule, which partitions the domain of one (or more) of the data's attributes. Based on the splitting rule, the data item is sent forward to a node's children. This testing and forwarding are repeated until the data item reaches a leaf node.

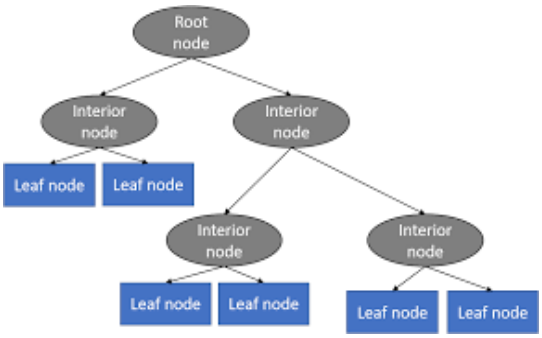

Figure 2.4 - Example of a Decision Tree

Decision trees can be used with both numerical (ordered) and categorical (unordered) attributes. There are also techniques to deal with missing or uncertain values. Typically, the decision rules are univariate. That is, each partitioning rule considers a single attribute. Multivariate decision rules have also been studied. They sometimes yield better results, but the added complexity is often not justified. Many decision trees are binary, with each partitioning rule dividing its subspace into two parts. Even binary trees can be used to choose among several class labels. Multiway splits are also common, but if the partitioning is into more than a handful of subdivisions, then both the interpretability and the stability of the tree suffers.

Let's now introduce mathematical notation to describe the data, its attributes, the class labels, and the tree structure. A data item x is a vector of d attribute values with an optional class label y. In addition, denote the set of attributes as $\boldsymbol{A} = \{A_1, A_2, \ldots, A_d\}$. Thus, define now x as

$\{x_1, x_2, \ldots, x_d\}$, where $x_1 \in A_1, x_2 \in A_2, \ldots, x_d \in A_d$. Let $Y = \{y_1, y_2, \ldots, y_m\}$ be the set of class labels. Each training item x is mapped to a class value y where $y \in Y$. Together they constitute a data tuple (x, y). The complete set of training data is X. The following table illustrates a summary of notation for data and partitions.

Table 2.1 - Data and Partitions for DT

Symbol	Definition
X	Set of all training data $= \{x_1, \ldots, x_n\}$
A	Set of all attributes $= \{A_1, \ldots, A_d\}$
Y	Domain of class values $= \{y_1, \ldots, y_m\}$
X_i	A subset of X
S	A splitting rule
X_S	A partitioning of X into $\{X_1, \ldots, X_k\}$

A partitioning rule S subdivides dataset X into a set of subsets collectively known as X_S; that is, $X_S = \{X_1, X_2, \ldots, X_k\}$ where $\bigcup_i X_i = X$. A *decision tree* is a rooted tree in which each set of children of each parent node corresponds to a partitioning (X_S) of the parent's dataset, with the full dataset associated with the root. The number of items in X_i that belong to class y_j is $|X_{ij}|$. The probability that a randomly selected member of X_i is of class y_j is $p_{ij} = \frac{|X_{ij}|}{|X_i|}$.

The decision tree algorithm can be written almost entirely as a single recursive function which can be widely found in the literature. The idea behind it, is that given a set of data items, which are each described by their attribute values, the function builds and returns a subtree. First, the function checks if it should stop further refinement of this branch of the decision tree. If so, it returns a leaf node, labelled with the class that occurs most frequently in the current data subset X'. Otherwise, it proceeds to try all feasible splitting options and selects the best one. A *splitting rule* partitions the dataset into subsets. What constitutes the "best" rule is perhaps the most distinctive aspect of one tree induction algorithm versus another. The algorithm creates a tree node for the chosen rule.

If a splitting rule draws all the classification information out of its attribute, then the attribute is exhausted and is ineligible to be used for splitting in any subtree. For example, if a discrete attribute with k different values is used to create k subsets, then the attribute is "exhausted". As a final but vital step, for each of the data subsets generated by the splitting rule, recursively call the function which builds the subtree. Each call generates a subtree that is then attached as a child to the principal node. The produced tree is returned now as output of the function [13] .

There exist many specific decision tree algorithms but the most notable of them include:

- ID3 (Iterative Dichotomiser 3)
- C4.5 (successor of ID3)
- CART (Classification and Regression Tree)
- Chi-Square automatic interaction detection (CHAID). Performs multi-level splits when computing classification trees.

Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items. Different algorithms use different metrics for measuring "best". These generally measure the homogeneity of the target variable within the subsets. These metrics are applied to each candidate subset, and the resulting values are combined (e.g., averaged) to provide a measure of the quality of the split. Some examples of them are given below.

- *Gini impurity*: used by the CART algorithm for classification trees, Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset. The Gini impurity can be computed by summing the probability p_i of an item with label i being chosen times the probability $\sum_{k \neq i} p_k = 1 - p_i$ of a mistake in categorizing that item. It reaches its minimum when all cases in the node fall into a single target category.

- *Information gain*: used by the ID3, and C4.5 tree-generation algorithms. Information gain is based on the concept of entropy and information content from information theory. Entropy is defined as:

  $$H(T) = I_E(p_1, p_2, \ldots, p_J) = -\sum_{i=1}^{J} p_i log_2 p_i$$

  Where $p_1, p_2, \ldots$ are fractions that add up to 1 and represent the percentage of each class present in the child node that results from a split in the tree.
  $IG(T, a) = H(T) - H(T|a)$ in other words, information gain equals entropy (of the parent) minus weighted sum of entropy (of the children).

The following table demonstrates some of the most common advantages and disadvantages when using decision tree classifier [14]:

Table 2.2 - DT Merits and Demerits

Decision Tree Classifier	
Merits	*Demerits*
Able to handle both numerical and categorical data	Not well suited for multivariate partitions
Easy to interpret and to produce understandable rules	Complex calculations if many values are uncertain
Runs fast even with lots of observations and variables	Less appropriate when the goal is to predict continuous attribute
Can learn incrementally	Computationally expensive to train

2.4.2 k-Nearest Neighbour (kNN)

Nearest Neighbour algorithms are among the simplest of all ML algorithms. The idea is to memorise the training set and then to predict the label of any new instance based on the labels of its closest neighbours in the training set. The rationale behind such a method is based on the assumption that the features that are used to describe the domain points are relevant to their labelling in a way that makes close-by points likely to have the same label. Furthermore, in some situations, even when the training set is immense, finding a nearest neighbour can be done extremely fast (for example, when the training set is the entire Web and distances are based on links). In contrast with other classification algorithms nearest neighbour method figures out a label on any test point without searching for a predictor within some predefined class of functions.

Assume X is an instance domain with a metric function ρ. That is, $\rho: X \times X \to \Re$ is a function that returns the distance between any two elements of X. For example, if $X = \Re^d$ then ρ can be the Euclidean distance,

$$\rho(x, x') = \left||x - x'|\right| = \sqrt{\sum_{i=1}^{d}(x_i - x_i')^2}.$$

Let $S = (x_1, y_1), \ldots, (x_m, y_m)$ be a sequence of training examples. For each $x \in X$ let $\pi_1(x), \ldots, \pi_m(x)$ be a reordering of $\{1, \ldots, m\}$ according to their distance to x, $\rho(x, x_i)$. That is, for all $i < m$,

$$\rho(x, x_{\pi_i(\chi)}) \leq \rho(x, x_{\pi_{i+1}(\chi)}).$$

For a number k, the k −NN rule for binary classification is defined as follows:

k −NN

Input: a training sample $S = (x_1, y_1), \ldots, (x_m, y_m)$

Output: for every point $x \in X$, return the majority label among $\{y_{\pi_i(x)} : i \leq k\}$

For $k = 1$, the 1-NN rule is established:

$$h_S(x) = y_{\pi_1(x)}.$$

On the right an illustration of the decision boundaries of the 1-NN rule is presented. The points depicted are the sample points, and the predicted label of any new point will be the label of the sample point in the centre of the cell it belongs to. These cells are called a *Voronoi Tessellation* of the space [15].

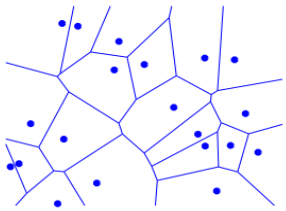

Figure 2.5 - DT boundaries of the 1-NN rule. The points depicted are the sample points, and the predicted label of any new point will be the label of the sample point in the centre of the cell it belongs to [15].

Curse of dimensionality

The kNN classifier is simple and can work quite well, provided it is given a good distance metric and has enough labelled training data. In fact, it can be shown that the kNN classifier can come within a factor of 2 of the best possible performance if N → ∞ (Cover and Hart 1967). However, the main problem with kNN classifiers is that they do not work well with high dimensional inputs. The poor performance in high dimensional settings is due to the *curse of dimensionality*.

To explain the curse, let's look at an example. Consider applying a kNN classifier to data where the inputs are uniformly distributed in the D-dimensional unit cube. Suppose there is a need to estimate the density of class labels around a test point x by "growing" a hyper-cube around x until it contains a desired fraction f of the data points. The expected edge length of this cube will be $e_D(f) = f^{1/D}$. If $D = 10$, and the goal is to base the estimate on 10% of the data then $e_{10}(0.1) = 0.8$, so the need now is to extend the cube 80% along each dimension around x. Even if only 1% of the data is used, it turns out $e_{10}(0.01) = 0.63$. Since the entire range of the data is only 1 along each dimension the method is no longer very local, despite the name "nearest neighbour". The trouble with looking at neighbours that are so far away is that they may not be good predictors about the behaviour of input-output function at a given point [16].

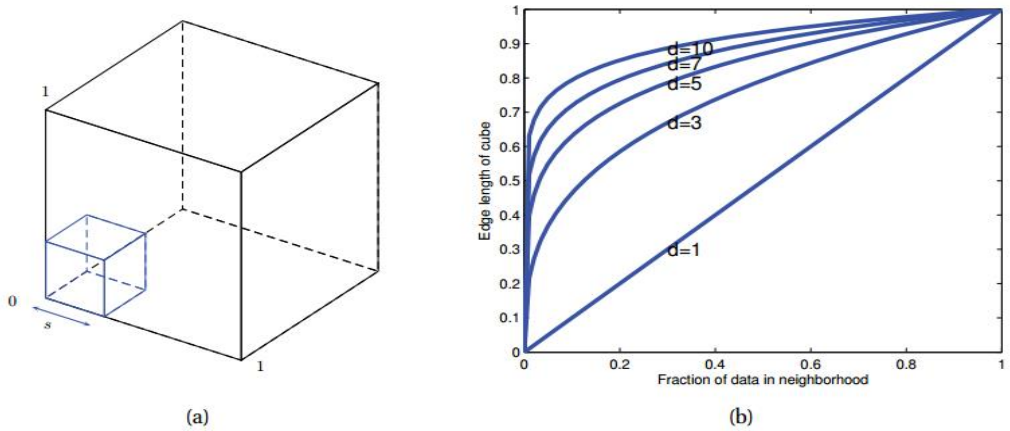

Figure 2.6 - Illustration of the curse of dimensionality. In (a) a small cube of side s inside a larger unit cube is embedded. In (b) the edge length of a cube needed to cover a given volume of the unit cube as a function of the number of dimensions [16].

Table 2.3 - kNN Merits and Demerits

kNN	
Merits	*Demerits*
Easy to implement and debug	Sensitive to irrelevant or redundant features
Robust regarding the search space	Expensive to find the nearest neighbour in a large training set
Sensitive to the local structure of the data	Dimensionality reduction preferred in a high dimensional space

2.4.3 Logistic Regression

Logistic regression is a classification algorithm used extensively in numerous disciplines, including web, engineering, economics, marketing applications and medical and social science fields. In *logistic regression* a family of functions h is mapped from $\Re^d$ to the interval $[0,1]$. However, logistic regression is used for classification tasks: One can interpret $h(x)$ as the probability that the label of x is 1. The hypothesis class associated with logistic regression is the composition of a sigmoid function (name "sigmoid" means "S-shaped", referring to the plot of the function shown) $\varphi_{sig}: \Re \to [0,1]$

over the class of linear functions L_d. In particular, the sigmoid function used in logistic regression is the logistic function, defined as

$$\varphi_{sig}(z) = \frac{1}{1+exp(-z)}.$$

The hypothesis class is therefore (where for simplicity homogeneous linear functions are used):

$$H_{sig} = \varphi_{sig} \circ L_d = \{x \to \varphi_{sig}(w \cdot x) : w \in \Re^d\}$$

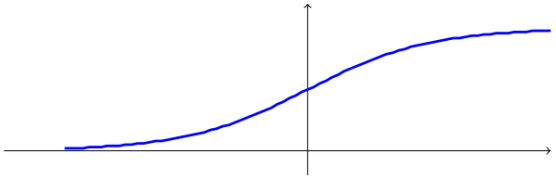

Figure 2.7 - The sigmoid function

Note that when $w \cdot x$ is very large then $\varphi_{sig}(w \cdot x)$ is close to 1, whereas if $w \cdot x$ is very small then $\varphi_{sig}(w \cdot x)$ is close to 0. Recall that the prediction of the halfspace corresponding to a vector w is $sign(\, w \cdot x)$. Therefore, the predictions of the half space hypothesis and the logistic hypothesis are very similar whenever $|w \cdot x|$ is large. However, when $|w \cdot x|$ is close to 0 then $\varphi_{sig}(w \cdot x) \approx \frac{1}{2}$. Intuitively, the logistic hypothesis is not sure about the value of the label so it guesses that the label is $sign(\, w \cdot x)$ with probability slightly larger than 50%. In contrast, the half space hypothesis always outputs a deterministic prediction of either 1 or -1, even if $|w \cdot x|$ is very close to 0.

Next, the goal is to specify a loss function. That is, define how bad it is to predict some $h_w(x) \in [0,1]$ given that the rule label is $y \in \{\pm 1\}$. Clearly, it would be preferable $h_w(x)$ to be larger if $y = 1$ and $1 - h_w(x)$ (i.e., the probability of predicting −1) to be large if $y = -1$. Note that

$$1 - h_w(x) = 1 - \frac{1}{1+exp(-w \cdot x)} = \frac{exp(-w \cdot x)}{1+exp(-w \cdot x)} = \frac{1}{1+exp(w \cdot x)}.$$

Therefore, any reasonable loss function would increase monotonically with $\frac{1}{1+exp[y(w \cdot x)]}$, or equivalently, would increase monotonically with $1 + exp[(-y(w \cdot x)]$. The logistic loss function used in logistic regression penalises h_w based on the log of $1 + exp[(-y(w \cdot x)]$ (recall that log is a monotonic function). That is,

$$l(h_w, (x, y)) = log(1 + exp[(-y(w \cdot x)]).$$

Therefore, given a training set $S = (x_1, y_1), \ldots, (x_m, y_m)$, the ERM (Empirical Risk Minimisation) problem associated with logistic regression is

$$argmin_{w \in \Re^d} \frac{1}{m} \sum_{i=1}^{m} log(1 + exp[(-y_i(w \cdot x_i)]).$$

The advantage of the logistic loss function is that it is a convex function with respect to w; hence the ERM problem can be solved efficiently using standard methods. The ERM problem associated with logistic regression is identical to the problem of finding a *Maximum Likelihood Estimator (MLE)*, a

well-known statistical approach for finding the parameters that maximise the joint probability of a given dataset assuming a specific parametric probability function [17].

Some other algorithms that are used to calculate the parameters of a logistic regression model include:

- *Steepest descent (or gradient descent)*: $\theta_{k+1} = \theta_k - \eta_k g_k$, where η_k is the step size or learning rate. The tricky part in gradient descent is how to set the step size. If someone uses a constant learning rate but makes it too small convergence will be very slow, but if it is made too large, the method can fail to converge at all;

- *Newton's method*: an iterative algorithm which consists of updates of the form $\theta_{k+1} = \theta_k - \eta_k H_k^{-1} g_k$. It is used for minimising a strictly convex function making thus H_k positive definite.

- *Quashi-Newton (variable metric) methods*: the mother of all second-order optimization algorithm is Newton's algorithm. Unfortunately, it may be too expensive to compute H explicitly. QuasiNewton methods iteratively build up an approximation to the Hessian using information gleaned from the gradient vector at each step.

Table 2.4 - LR Merits and Demerits

Logistic Regression	
Merits	*Demerits*
Easy to implement and efficient to train	Requires large sample size to achieve stable results
It outputs well-calibrated predicted probabilities	Limited to linear relationships between variables
No parameters to tune or scale	Sensible to outliers

2.4.4 Naive Bayes Classifier

A widely used framework for classification is provided by a simple theorem of probability known as Bayes' theorem or Bayes' rule. Based on the product rule, together with the symmetry property on probabilities it is easy to obtain the following Bayes' theorem,

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} \text{ (*)},$$

which plays a central role in ML, especially classification. Using the sum rule, the denominator in Bayes' theorem can be expressed in terms of the quantities appearing in the numerator

$$p(X) = \sum_Y p(X|Y)p(Y).$$

The denominator in Bayes' theorem can be regarded as being the normalisation constant required to ensure that the sum of the conditional probability on the left-hand side of equation (*) over all values of Y equals one.

The Naive Bayes classifier is based on Bayes' theorem, and is particularly suited when the dimensionality of the inputs is high. Despite its simplicity, the Naive Bayes classifier can often achieve comparable performance with some sophisticated classification methods, such as decision tree and selected neural network classifier. Naive Bayes classifiers have also exhibited high accuracy and speed when applied to large datasets.

Let us first define the problem setting as follows: Suppose a training set $\{(x^{(i)}, y^{(i)})\}$ is given consisting on N examples, each $x^{(i)}$ is a d −dimensional feature vector, and each $y^{(i)}$ denotes the class label for the example. Assume furthermore random variables Y and X with components $X_1, \ldots, X_d$ correspond to the label y and the feature vector $x = (x_1, x_2, \ldots, x_d)$. Note that the superscript is used to index training examples for $i = 1, \ldots, N$, and the subscript is used to refer to each feature or random variable of a vector. In general, Y is a discrete variable that falls into exactly one of K possible classes $\{C_k\}$ for $k \in \{1, .., K\}$, and the features of $X_1, \ldots, X_d$ can be discrete or continuous attributes.

Our task is to train a classifier that will output the posterior probability $p(Y|X)$ for possible values of Y. According to Bayes' theorem, $p(Y = C_k|X = x)$ can be represented as

$$p(Y = C_k|X = x) = \frac{p(X=x|Y=C_k)p(Y=C_k)}{p(X=x)} = \frac{p(X_1=x_1,X_2=x_2,\ldots,X_d=x_d|Y=C_k)p(Y=C_k)}{p(X_1=x_1,X_2=x_2,\ldots,X_d=x_d)} \quad (**)$$

One way to learn $p(Y|X)$ is to use the training data to estimate $p(X|Y)$ and $p(Y)$. Then use these estimates, together with Bayes' theorem, to determine the probability $p(Y|X = x^{(i)})$ for any new instance $x^{(i)}$.

It is typically intractable to learn exact Bayesian classifiers. Considering the case that Y is Boolean and X is a vector of d Boolean features, the need then is to estimate approximately 2^d parametres $p(X_1 = x_1, X_2 = x_2, \ldots, X_d = x_d|Y = C_k)$. The reason is that, for any particular value C_k, there are 2^d possible values of x, which need to compute $2^d - 1$ independent parameters. Given two possible values for Y, a total of $2(2^d - 1)$ such parameters are to be estimated. Moreover, to obtain reliable estimates of each of these parameters, observe each of these distinct instances multiple times, which is clearly unrealistic in most practical classification domains. For example, if X is a vector with 20 Boolean features, then more than 1 million parameters must be computed.

To handle the intractable sample complexity for learning the Bayesian classifier, the Naive Bayes classifier reduces this complexity by making a conditional independence assumption that the features $X_1, \ldots, X_d$ are all conditionally independent of one another, given Y. For the previous case, this

conditional independence assumption helps to dramatically reduce the number of parameters to be estimated for modelling $p(Y|X)$ from the original $2(2^d - 1)$ to just $2d$. Consider the likelihood $p(X = x|Y = C_k)$ of equation (**), then

$$\begin{aligned} & p(X_1 = x_1, X_2 = x_2, \ldots, X_d = x_d|Y = C_k) \\ &= \prod_{j=1}^{d} p(X_j = x_j|X_1 = x_1, X_2 = x_2, \ldots, X_{j-1} = x_{j-1}, Y = C_k) \\ &= \prod_{j=1}^{d} p(X_j = x_j|Y = C_k) \end{aligned} \quad (***)$$

The second line follows from the chain rule, a general property of probabilities, and the third line follows directly from the above conditional independence, that the value for the random variable X_j is independent of all other feature values, $X_{j'}$ for $j' \neq j$, when conditioned on the identity of the label Y. This is the *Naive Bayes* assumption. It is a relatively strong and very useful assumption. When Y and X_j are Boolean variables, $2d$ parameters are required to define $p(X_j|Y = C_k)$.

After substituting (***) in Equation (**), one obtains the fundamental equation for the Naive Bayes classifier

$$p(Y = C_k|X_1 \ldots X_d) = \frac{p(Y = C_k) \prod_j p(X_j|Y = C_k)}{\sum_i p(Y = y_j) \prod_j p(X_j|Y = y_i)}$$

If concentrating on the most probable value of Y, then the Naive Bayes classification rule becomes:

$$Y \leftarrow arg\ max_{C_k} \frac{p(Y=C_k) \prod_j p(X_j|Y=C_k)}{\sum_i p(Y=y_j) \prod_j p(X_j|Y=y_i)},$$

Because the denominator does not depend on Ck, the above formulation can be simplified to the following

$$Y \leftarrow arg\ max_{C_k} p(Y = C_k) \prod_j p(X_j|Y = C_k).$$

Maximum-Likelihood Estimates for Naive Bayes Models

Next, once MLE is the most common technique used to determine the parameters of the Naive Bayes classifier, let's shortly go over its outline. The Naive Bayes model has two types of parameters that must be estimated. The first is

$$\pi_k \equiv p(Y = C_k)$$

for any of the possible values C_k of Y. The parameter can be interpreted as the probability of seeing the label C_k, under the constraints $\pi_k \geq 0$ and $\sum_{k=1}^{K} \pi_k = 1$. Note there are K of these parameters, $(K - 1)$ of which are independent.

For the d input features X_i, suppose each can take on J possible discrete values, and use for that reason $X_i = x_{ij}$ to denote that. The second is

$$\theta_{ijk} \equiv p(X_i = x_{ij}|Y = C_k)$$

For each input feature X_i, each of its possible values x_{ij}, and each of the possible values C_k of Y. The value for θ_{ijk} can be interpreted as the probability of feature X_i taking value x_{ij}, conditioned on the underlying label being C_k. Note that they must satisfy $\sum_j \theta_{ijk} = 1$ for each pair of i, k values, and there will be dJK such parameters, and note that only $d(J-1)K$ of these are independent [18].
As in any previous algorithm seen, NB classifier comes with its advantages and disadvantages, most important of which are presented in the following table:

Table 2.5 - NB Merits and Demerits

Naive Bayes Classifier	
Merits	*Demerits*
Entire covariance matrix needs to be calculated	Strong feature independence assumptions
Easy to deal with missing attributes	Precision and recall keep low on small datasets
Empirically successful	Low in accuracy

2.4.5 Support Vector Machine (SVM)

Another approach to classification is achieved by using a family of algorithms called support vector machines (SVM). They can work with both linear and non-linear scenarios, allowing high performance in many different contexts. Together with neural networks, SVMs probably represent the best choice for many tasks where it's not easy to find a good separating hyperplane. For example, for a long time, SVMs were the best choice for MNIST dataset classification, thanks to the fact that they can capture very high nonlinear dynamics using a mathematical trick, without complex modifications in the algorithm [19].

The goal of SVM is to produce nonlinear boundaries by constructing a linear boundary in a large, transformed version of the feature space. But let's have a better inside by considering first separable classes and a training dataset consisting of N pairs $(x_1, y_1), \ldots, (x_N, y_N)$, with $x_i \in \Re^p$ and $y_i \in \{-1,1\}$. Define a hyperplane by

$$\{x: f(x) = x^T\beta + \beta_0 = 0\},$$

Where β is a unit vector: $||\beta|| = 1$. A classification rule induced by $f(x)$ is

$$G(x) = sign[x^T\beta + \beta_0].$$

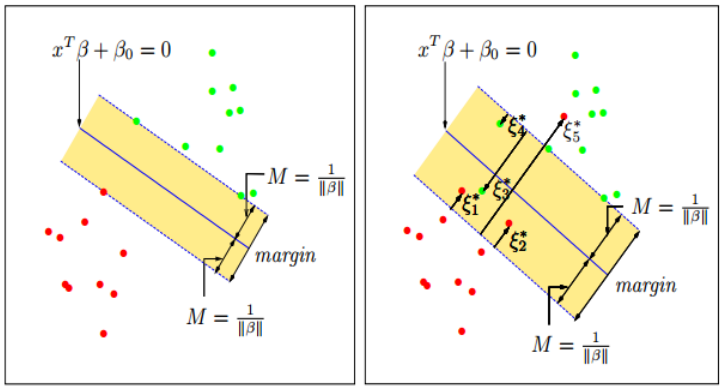

Figure 2.8 - Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2M = 2/||\beta||$. The right panel shows the nonseparable (overlap) case. The points labelled ξ_j^* are on the wrong side of their margin by an amount $\xi_j^* = M\xi_j$;points on the correct side have $\xi_j^* = 0$ [20]

Since the classes are separable, it is possible to find a function for which $f(x) = x^T\beta + \beta_0$ with $y_i f(x_i) > 0\ \forall i$ such that $f(x)$ gives the signed distance from a point x to the hyperplane $f(x) = x^T\beta + \beta_0 = 0$. Hence, one can establish a hyperplane that creates the biggest *margin* between the training points for class 1 and -1. The optimisation problem

$$max_{\beta,\beta_0,||\beta||=1} M$$
$$\text{subject to } y_i({x_i}^T\beta + \beta_0) \geq M, i = 1, \ldots, N \quad (\$),$$

captures this concept. The band in the figure is M units away from the hyperplane on either side, and hence $2M$ units wide. It is called the *margin*.

This problem can be more conveniently rephrased as

$$min_{\beta,\beta_0} ||\beta||$$
$$\text{subject to } y_i({x_i}^T\beta + \beta_0) \geq 1, i = 1, \ldots, N \quad (\#),$$

where the norm constraint on β is dropped. Note that $M = 1/||\beta||$. Expression (#) is the usual way of writing the support vector criterion for separable data and it is characterised as a convex optimisation problem (quadratic criterion, linear inequality constraints).

Suppose now that the classes are not separable and that they overlap in feature space. One way to deal with the overlap is to still maximise M, but allow for some points to be on the wrong side of the margin. Define the slack variables $\xi = (\xi_1, \ldots \xi_N)$. There are two natural ways to modify the constraint in ($):

$$y_i({x_i}^T\beta + \beta_0) \geq M - \xi_i,$$
$$\text{or}$$
$$y_i({x_i}^T\beta + \beta_0) \geq M(1 - \xi_i),$$

$\forall i, \xi_i \geq 0, \sum_{i=1}^{N} \xi_i \leq constant$. The two choices lead to different solutions. The first choice seems more natural, since it measures overlap in actual distance from the margin; the second choice measures the overlap in relative distance, which changes with the width of the margin M. However, the first-choice results in a nonconvex optimization problem, while the second is convex; thus (12.6) leads to the "standard" support vector classifier, which is extensively used.

Here is the idea of the formulation. The value ξ_i in the constraint $y_i({x_i}^T\beta + \beta_0) \geq M(1 - \xi_i)$ is the proportional amount by which the prediction $f(x_i) = {x_i}^T\beta + \beta_0$ is on the wrong side of its margin. Hence by bounding the sum $\sum \xi_i$, one bounds the total proportional amount by which predictions fall on the wrong side of their margin. Misclassifications occur when $\xi_i > 1$, so bounding $\sum \xi_i$ at a value K say, bounds the total number of training misclassifications at K.

We can then drop the norm constraint on β define $M = 1/||\beta||$, and write (#) in the equivalent form

$$min||\beta|| \text{ subject to } y_i({x_i}^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i$$
$$\text{and } \xi_i \geq 0, \ \sum \xi_i \leq constant. \text{ (##)}$$

This is the usual way the support vector classifier is defined for the non-separable case.

Lagrangian multipliers on SVM

The (##) formulation introduced above is quadratic with linear inequality constraints, hence it is a convex optimization problem. A quadratic programming solution is described using Lagrange multipliers. Computationally it is convenient to re-express (##) in the equivalent form

$$min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N} \xi_i \ (\text{^})$$
$$\text{Subject to } \xi_i \geq 0, y_i({x_i}^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i,$$

Where the "cost" parameter C replaces constraint in (##); the separable case corresponds to $C = \infty$.

The Lagrange (primal) function is

$$L_P = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i[y_i({x_i}^T\beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i\xi_i \ (\$\$),$$

Which is minimised w.r.t β, β_0 and ξ_i. Set the respective derivatives to zero, to get

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \ (1),$$
$$0 = \sum_{i=1}^{N} \alpha_i y_i \ (2),$$
$$\alpha_i = C - \mu_i, \forall i \ (3),$$

As well as the positivity constraints $\alpha_i, \mu_i, \xi_i \geq 0 \ \forall i$. By substituting the above 3 equations to ($$), the Lagrangian dual objective function is produced

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} \alpha_i\alpha_{i'}y_iy_{i'}{x_i}^Tx_{i'} \ (4),$$

Which gives a lower bound on the objective function (^) for any feasible point. Then, maximise L_D subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$. In addition to (1)-(3) the Karush–Kuhn–Tucker conditions include the constraints

$$\alpha_i[y_i({x_i}^T\beta + \beta_0) - (1 - \xi_i)] = 0 \quad (5)$$
$$\mu_i \xi_i = 0 \quad (6)$$
$$y_i({x_i}^T\beta + \beta_0) - (1 - \xi_i) \geq 0 \quad (7)$$

For $i = 1, \ldots, N$. Together these equations (1)-(7) uniquely characterise the solution to the primal and dual problem [20].

Table 2.6 - SVM Merits and Demerits

SVM	
Merits	*Demerits*
Can handle multiple continuous and categorical variables	Parameters of a settled model are hard to decipher
Supports both regression and classification tasks ranking problems	Do not specifically give probability estimates
Works good with imbalanced data	Lack of transparency of results

2.4.6 Artificial Neural Network (ANN)

An artificial neural network is a model of computation inspired by the structure of neural networks in the brain. In simplified models of the brain, it consists of a large number of basic computing devices (neurons) that are connected to each other in a complex communication network, through which the brain is able to carry out highly complex computations. *Artificial Neural Networks (ANN)* are formal computation constructs that are modelled after this computation paradigm.

Learning with neural networks was proposed in the mid-20th century. It yields an effective learning paradigm and has recently been shown to achieve cutting edge performance on several learning tasks. A neural network can be described as a directed graph whose nodes correspond to neurons and edges correspond to links between them. Each neuron receives as input a weighted sum of the outputs of the neurons connected to its incoming edges [21].

The purpose of a mathematical model to describe this is that it extracts only the bare essentials required to accurately represent the entity being studied, removing all of the extraneous details. McCulloch and Pitts produced a perfect example of this when they modelled a neuron as:

- a set of weighted inputs w_i that correspond to the synapses
- an adder that sums the input signals (equivalent to the membrane of the cell that collects electrical charge)
- an activation function (initially a threshold function) that decides whether the neuron fires ('spikes') for the current inputs

On the left of the picture are a set of input nodes labelled $x_1, \ldots, x_m$. These are given some values, and if $x_i = 0$ it means neuron i didn't fire and 1 means it did. Each of these other neuronal firings flowed along a synapse to arrive at the neuron, and those synapses have strengths, called *weights*. The strength of the synapse affects the strength of the signal, so multiply the input by the weight of the synapse. Now when all these signals arrive into the neuron, it adds them up to see if there is enough strength to make it fire. Thus, write that as

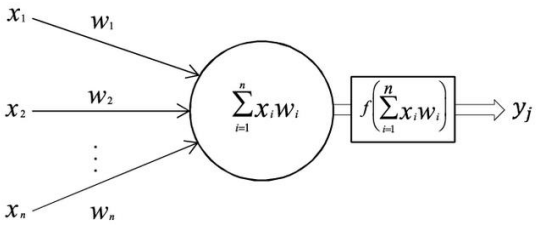

Figure 2.9 - A picture of McCulloh and Pitts' mathematical model of a neuron. The inputs x_i are multiplied by the weights w_i, and the neurons sum their values. If this sum is greater than the threshold ϑ then the neuron fires; otherwise it does not [22].

$$h = \sum_{i=1}^{m} w_i x_i$$

which just means sum all the inputs multiplied by their synaptic weights. Then the need is to decide a *threshold* value θ so that if $h > \theta$ then neuron fires, whereas if $h < \theta$ it does not. Having said that the McCulloch and Pitts neuron is a binary threshold device one writes the decision whether a neuron fires (which is known as an *activation function*) as:

$$g(h) = 1, if\ h > \theta \text{ or}$$
$$g(h) = 0, if\ h \leq \theta.$$

Perceptron

The Perceptron is nothing more than a collection of McCulloch and Pitts neurons together with a set of inputs and some weights to fasten the inputs to the neurons. On the left of the figure, shaded in light grey, are the input nodes. These are not neurons, they are just a nice schematic way of showing how values are fed into the network, and how many of these input values there are (which is the dimension in the input vector). The neurons are shown on the right with both the additive part

(shown as a circle) and the thresholder. Although in the picture neurons are as much as inputs this is not always the case and it could be m inputs and n neurons.

When looking at the McCulloch and Pitts neuron, the weights were labelled as w_i, with the i index running over the number of inputs. Here, there is also a need to work out which neuron the weight feeds into, so labelling them as w_{ij}, where the j index runs over the number of neurons. Now, to compute the activation function and determine which neuron fires the two above equations are used for each neuron to receive a vector with zeros and ones.

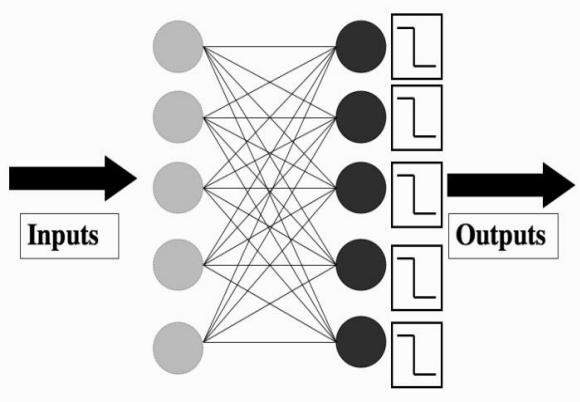

Figure 2.10 - The perceptron network, consisting of a set of input nodes connected to neurons using weighted connections.

The interesting part comes in when a neuron fired when it shouldn't and vice versa. Then, it is obvious that the values of the weights should be changed so neuron gets it right next time. Let's label the neuron that is wrong as k, then the weights that are of interest are w_{ik}, where i runs from 1 to m and one has to compute $y_k - t_k$ (the difference between the output y_k, which is what the neuron did, and the target for that neuron, t_k, which is what the neuron should have done. This is a possible *error function*). To get around with different signs of inputs' elements compute $\Delta w_{ik} = -(y_k - t_k) \times x_i$ so the new value of the weight is the old value plus this value.

Note that there are cases when the threshold value of some neuron must change. Take for example an input with value 0. In that case, even if a neuron is wrong, changing the relevant weight doesn't do anything so the need now is to change the threshold. This is done by multiplying the value above by a parameter called the *learning rate*, usually labelled as η. The value of the learning rate determines how fast the network learns. Hence, the final rule for updating a weight w_{ij} is:

$$w_{ij} \leftarrow w_{ij} - \eta(y_j - t_j) \cdot x_i$$

Multi-layer Perceptron

We previously saw that basically the learning in the neural network happens in the weights. So, to perform more computation it seems sensible to add more weights. There are two things that someone can do: add some backwards connections, so that the output neurons connect to the inputs again or add more neurons. The first approach leads into recurrent networks. These have been studied but are not that commonly used. Consider instead the second approach, in which the neurons between the input nodes and the outputs are added. This will make more complex neural networks, as in the figure. Nevertheless, this approach will generate an issue, namely, how to train this network so that the weights are adapted to generate the correct (target) answers? If considering the method used for the Perceptron one needs to compute the error at the output. That's fine, since the targets there are already known, so the difference between the targets and the outputs is to be computed.

But now the question becomes which weights were wrong: those in the first layer, or the second? Worse, what are the correct activations for the neurons in the middle of the network? This fact gives the neurons in the middle of the network their name; they are called the *hidden layer (or layers)*, because it isn't possible to examine and correct their values directly.

It took a long time for people who studied neural networks to work out how to solve this problem. In fact, it wasn't until 1986 that Rumelhart, Hinton, and McClelland managed it. However, a solution to the problem was already known by statisticians and engineers but they just didn't know that it was a problem in neural networks! Here only the neural network solution proposed by Rumelhart, Hinton, and McClelland is considered, and that is the Multilayer Perceptron (MLP), which belongs to the group of the most commonly used ML methods around.

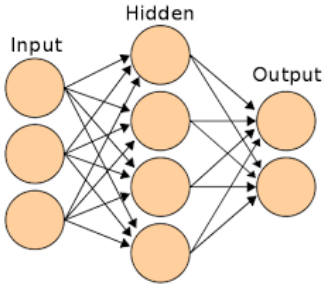

Figure 2.11 - Example of an Artificial Neural Network.

Back-propagation of error

Training the MLP consists of two parts: working out what the outputs are for the given inputs and the current weights, and then updating the weights according to the error, which is a function of the difference between the outputs and the targets. Computing the errors at the output is no more difficult than it was for the Perceptron but working out what to do with those errors is more difficult. The method that is considered here is s form of gradient descent and it is called back-propagation of error, which makes it clear that the errors are sent backwards through the network.

The error function that was used for the Perceptron was $\sum_{k=1}^{N} E_k = \sum_{k=1}^{N} y_k - t_k$, where N is the number of the output nodes. To get around though again with the different signs of these errors, it makes sense to consider the *sum-of-squares* error function, which calculates the difference between y and t for each node, squares them, and adds them all together:

$$E(t, y) = \frac{1}{2}\sum_{k=1}^{N}(y_k - t_k)^2$$

Another issue that has to be overcome is the discontinuity of the threshold function which doesn't allow to differentiate at the point of discontinuity. To avoid this, let's use instead the *sigmoid function* whose most commonly used form is:

$$a = g(h) = \frac{1}{1+exp(-\beta h)}\left(= tanh(h) = \frac{exp(h)-exp(-h)}{exp(h)+exp(-h)}\right).$$

Where the expression in the parenthesis is the hyperbolic tangent function sometimes included in texts. This is a different but similar function; it is still a sigmoid function, but it saturates (reaches its constant values) at ± 1in contrast with 0 and 1, which is sometimes useful.

To sum it up, the key thing of the procedure of the algorithm is to understand that the gradients of the errors with respect to the weights are computed, so that the weights are changed in order to

go downhill, which makes the errors get smaller. To do this try differentiating the error function with respect to the weights, by applying the chain rule and differentiate with respect to known things. This leads to two different update functions, one for each of the sets of weights, and then just apply these backwards through the network, starting at the outputs and ending up back at the inputs.

Multi-layer Perceptron (continued)

Assume L input nodes, plus the bias, M hidden nodes, also plus the bias, and N output nodes, so that there are $(L+1) \times M$ weights between the input and the hidden layer and $(M+1) \times N$ between the hidden layer and the output. The sums will start from 0 if they include the bias nodes and 1 otherwise, and run up to L, M, or N, so that $x_0 = -1$ is the bias input, and $a_0 = -1$ is the bias hidden node. The algorithm that is described could have any number of hidden layers, in which case there might be several values for M, and extra sets of weights between the hidden layers. Let's also use i, j, k to index the nodes in each layer in the sums, and the corresponding (ι, ζ, κ) for fixed indices.

Here is a quick summary of how the algorithm works, and then the full MLP training algorithm using back-propagation of error is described [22].

1. an input vector is put into the input nodes
2. the inputs are fed forward through the network
    2.1. The inputs and the first-layer weights (here labelled as v) are used to decide whether the hidden nodes fire or not. The activation function $g(\cdot)$ is the sigmoid function that is given above
    2.2. The outputs of these neurons and the second-layer weights (labelled as w) are used to decide if the output neurons fire or not
3. the error is computed as the sum-of-squares difference between the network outputs and the targets
4. this error is fed backwards through the network in order to
    4.1. first update the second-layer weights
    4.2. and then afterwards, the first-layer weights

Table 2.7 - MLP Merits and Demerits

MLP	
Merits	*Demerits*
Capability to learn non-linear models	Requires tuning several hyper parameters (hidden neurons, layers)
Capability to learn models in real-time	Sensitive to feature scaling

The Multi-layer Perceptron Algorithm

- Initialisation
  - Initialise all weights to small (positive and negative) random values
- Training
  - Repeat
    - For each input vector:

      Forward phase:

      ➔ Compute the activation of each neuron j in the hidden layer(s) using:

      $$h_\zeta = \sum_{i=0}^{L} x_i v_{i\zeta}$$

      $$a_\zeta = g(h_\zeta) = \frac{1}{1+exp(-\beta h_\zeta)}$$

      ➔ work through the network until you get to the output layer neurons, which have activations:

      $$h_\kappa = \sum_j a_j w_{j\kappa}$$

      $$y_\kappa = g(h_\kappa) = \frac{1}{1+exp(-\beta h_\kappa)}$$

      Backwards phase:

      ➔ compute the error at the output using:

      $$\delta_o(\kappa) = (y_k - t_k) y_k (1 - y_k)$$

      ➔ compute the error in the hidden layer(s) using:

      $$\delta_h(\zeta) = a_\zeta (1 - a_\zeta) \sum_{k=1}^{N} w_\zeta \delta_o(k)$$

      ➔ update the output layer weights using:

      $$w_{\zeta\kappa} \leftarrow w_{\zeta\kappa} - \eta \delta_o(\kappa) a_\zeta{}^{hidden}$$

      ➔ update the hidden layer weights using:

      $$v_\iota \leftarrow v_\iota - \eta \delta_h(\kappa) x_\iota$$

    - (if using sequential updating) randomise the order of the input vectors so that you don't train in exactly the same order each iteration
  - until learning stops
- Recall
  - use the Forwards phase in the training section above

2.5 Evaluation Methods

After applying the algorithm to the test subset of the data an essential part of the project follows; evaluating the effectiveness of the algorithm. For that reason, some of the existing binary classification model evaluation techniques follow:

- *Accuracy*: it is one of the most common metrics used and it refers to the fraction of correctly classified samples of the model

- *Confusion matrix*: a comprehensive way to represent the result of evaluating binary classification. It is a two by two array, where the rows correspond to the true classes, and the columns correspond to the predicted classes. Each entry counts for how many data points in the class given by the row the prediction was the class given the column. Usually, this measure reveals more accurately the performance of the algorithm in a class imbalanced dataset, where a significant disparity between the number of positive and negative labels occurs. There also exist additional metrics based on the actual versus the predicted classes by the model [23]. These are:

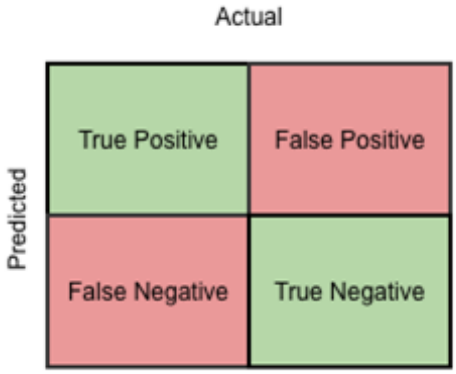

Figure 2.12 - A confusion matrix.

  - *Precision*: the ratio of the correctly positive labelled by the algorithm to all positive labelled;

  - *Sensitivity (or recall or true positive rate)*: the ability of a model to find all the relevant cases within the dataset. It's the number of correctly positive labelled divided by the number of correctly positive labelled plus the number of incorrectly negative labelled;

  - *Specificity*: measures the proposition of actual negatives that are correctly identified as such;

  - *False positive rate*: used in building the ROC curve among others and is calculated as the ratio between the number of negative events wrongly categorised as positive and the total number of actual negative events

- ➢ *f1-score*: a measure that considers both precision and recall. Specifically, it equals the harmonic mean of precision and recall and consequently it achieves best value if the two above measures are balanced

In the next table the exact mathematical formula for each measure is presented:

Table 2.8 - Performance Evaluation Measures

Measure	Formula
accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
precision	$\frac{TP}{TP + FP}$
recall	$\frac{TP}{TP + FN}$
specificity	$\frac{TN}{TN + FP}$
fp rate	$\frac{FP}{FP + TN}$
f1-score	$\frac{2 * recall * precision}{recall + precision}$

- ❖ *Logarithmic loss*: this evaluation metric quantifies the accuracy of a classifier by penalising false classifications. The target consists of minimising the *logarithmic loss (or log loss)* which equivalently maximises the accuracy of the model. Mathematically the *log loss* is defined as:

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} log p_{ij}$$

Where N is the number of samples, M is the number of possible labels, y_{ij} is a binary indicator of whether or not label j is the correct classification for instance i, and p_{ij} is the model probability of assigning label j to instance i. As it can easily implied from the above, an ideal classifier would have a log loss of precisely zero. In the case of binary classification, the *log loss* is simplified to the following expression

$$-\frac{1}{N}\sum_{i=1}^{N}[y_i log p_i + (1 - y_i) log(1 - p_i)].$$

As a note only the term for the correct class actually contributes to the sum.

❖ *ROC curve & AUC*: an essential task of any classification problem is to take advantage of the *ROC (Receiver Operating Characteristics)* curve in order to visualise, organise, and select classifiers based on their performance. It is a probability curve which interprets how much the model is capable of distinguishing between classes. *ROC* graphs are two-dimensional graphs in which recall (or true positive rate) is plotted on the y-axis and false positive rate is plotted on the x-axis. Such a graph depicts relative trade-offs between benefits (true positive) and costs (false positives). Several points in *ROC* space are important to note. The lower left point (0,0) represents the strategy of never issuing a positive classification; such a classifier commits no false positive errors but also gains no true positives. The opposite strategy, of unconditionally issuing positive classifications, is represented by the upper right point (1,1) and the point (0,1) represents perfect classification.

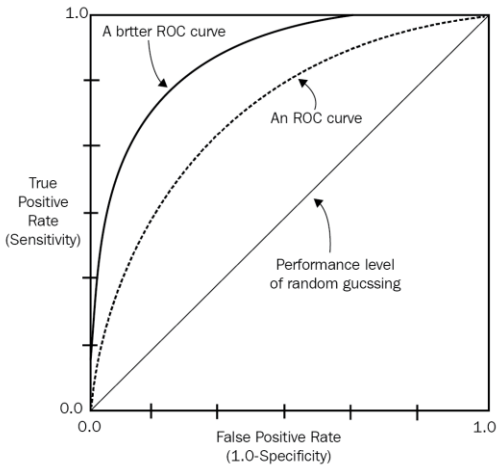

Figure 2.13 - ROC curve illustrating an accurate and a less accurate model.

AUC (Area Under Curve) represents degree or measure of separability. Higher the *AUC,* the better the model is at predicting 0s as 0s / 1s as 1s and in analogy the better the model is at distinguishing between patients with disease and no disease. An excellent model has *AUC* near to 1 which means it has good measure of separability. On the other hand, a poor model has *AUC* near to 0 which means it has worst measure of separability and in fact it means it is reciprocating the result, namely predicting 0s as 1s and 1s as 0s. And when *AUC* is 0.5, it means model has no class separation capacity whatsoever [24].

❖ *k-fold cross validation*: probably the simplest and most widely used method for estimating prediction error. Ideally, if enough data is given, one would set aside a validation set and use it to assess the performance of the prediction model. Since data are often scarce, this is usually not possible. To finesse the problem, *k-fold cross validation* uses part of the available data to fit the model, and a different part to test it. So split the data into k roughly equal sized parts and for the kth part, fit the model to the other (k−1) parts of the data. Then, calculate the prediction error of the fitted model when predicting the kth part of the data. Do this for k = 1, 2, …, K and combine the k estimates of prediction error [25].

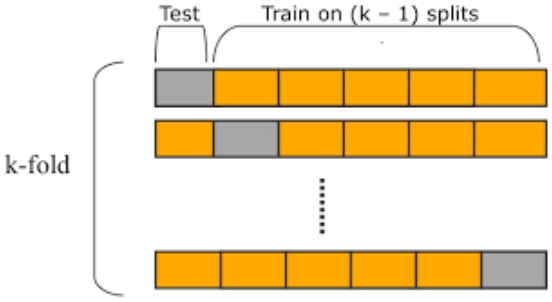

Figure 2.14 - k-fold cross validation interpretation

2.6 Underfitting/Overfitting and trade-off

The main goal when using an ML algorithm is to fit it to the data in a way that it is easy to generalise any new input sample from the problem domain in a proper way. Following this tactic leads to results free of bias but first two commonly seen step backs namely *underfitting* and *overfitting* must be overcome.

- *Overfitting*: building a complex model that does well on the training set but does not generalise to new unseen data.

- *Underfitting*: stopping a model from satisfactorily comprehend some meaningful relations in the data which causes it to learn an estimation that is not as precise as hoped

- *Capacity*: the ability of an algorithm to model the complexity of the data

- *Generalisation*: the models' ability to predict values that it has not seen in the training set

- *Bias-Variance trade-off* is the problem of simultaneously minimizing the error originating from two sources:

    - *Bias* which is an error caused by false assumptions held by the predictor. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

    - *Variance* that occurs when the model is overly sensitive to small fluctuations in the training set. This can cause an algorithm to model the random noise in the training data, impacting its ability to generalize on unseen data (overfitting).

This is referred to as a *trade-off* because reducing one of these two error types might raise the other. In practice achieving both low variance and low bias is possible but difficult. A method to achieve that is by first choosing a model with high enough capacity of modelling the data so that it reduces the bias to low levels. Then regularise the model to reduce the variance. R*egularisation* means the technique that puts a constraint on the model during its training phase, so that is isn't so sensitive to the variance of the data.

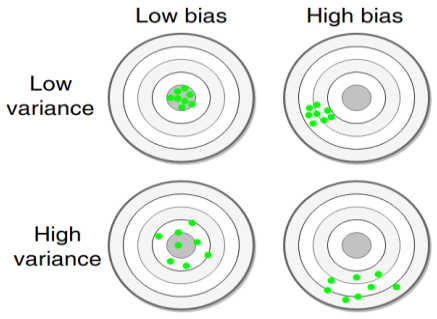

Figure 2.15 - Understanding bias-variance trade-off.

Chapter 3

Dataset Insight

3.1 Mesothelioma Disease

Mesothelioma is an aggressive, malignant cancer caused when inhaled asbestos fibres lodge in the lining of the lungs, abdomen or heart. More than 80% of mesothelioma cases are caused by exposure to asbestos. As of 2013, about 125 million people worldwide have been exposed to asbestos at work. High rates of disease occur in people who mine asbestos, produce products from asbestos, work with asbestos products, live with asbestos workers, or work in buildings containing asbestos. There exist four different types of mesothelioma disease which are named after the position of the body where they develop. These are: Pleural Mesothelioma (soft tissue covering the lungs), Peritoneal Mesothelioma (lining surrounding the abdomen), Pericardial Mesothelioma (soft tissue around the heart) and Testicular Mesothelioma (lining of the testes). The first two are the most common whereas the rest occur with less than or equal to 1% of the cases.

Symptoms of the disease can include shortness of breath, difficulty breathing, persistent cough, significant weight loss and chest pain. After a doctor confirms a mesothelioma diagnosis and determines the stage of the disease, they will be able to provide a prognosis. Treatment for mesothelioma is similar to other types of cancer. The most aggressive treatment is a multimodal approach of surgery, chemotherapy and radiation, though these treatments may be used individually as well. Because mesothelioma latency is 20-50 years many people with mesothelioma are in their 60s or 70s. The life expectancy for most mesothelioma patients is approximately 12 months after diagnosis but nonetheless early diagnosis may lead to patient's life expectancy improvement.

3.2 UCI and Mesothelioma Disease Dataset

The dataset selected for this thesis was pulled from the University of California Irvine (UCI) ML Repository. The UCI ML Repository is a collection of databases, domain theories, and data generators that are used by the ML community for the empirical analysis of ML algorithms. The archive was created as an ftp (file transfer protocol) archive in 1987 by David Aha and fellow graduate students at UC Irvine. Since that time, it has been widely used by students, educators, and researchers all over

the world as a primary source of ML datasets. As an indication of the impact of the archive, it has been cited over 1000 times, making it one of the top 100 most cited papers in all of computer science.

Mesothelioma Disease Dataset specifically consists of 324 observations with 35 attributes. Each example corresponds to a single patient with potential mesothelioma symptoms. One of the total 35 features is the diagnosis label “class of diagnosis”. Concerning the imbalance of the dataset 96 patients (29.63%) suffer from mesothelioma whereas 228 patients (70.37%) do not suffer from mesothelioma. The dataset was prepared by Abdullah Cetin Tanrikulu from Dicle University, Faculty of Medicine, Department of Chest Diseases and Orhan Er from Bozok University, Faculty of Engineering, Department of Electrical and Electronics Eng. both in Turkey. The first analysis of the dataset was published in October 2011 and later in January 2016 it was released publicly on the UCI ML Repository.

3.3 Attributes Explanation

The following table displays information about value range and measurement units for all attributes of the mesothelioma disease dataset.

Table 3.1 - Attributes range and value type

DATA ATTRIBUTES		
INPUT (34 features)	**VALUE RANGE**	**MEASUREMENT UNIT**
age	[19, 85]	years
gender	0, 1	category
city	[0, 8]	category
asbestos exposure	0, 1	Boolean
type of MM	0, 1, 2	category
duration of asbestos exposure	[0, 70]	years
diagnosis method	0, 1	Boolean
keep side	0, 1, 2	category
cytology	0, 1	Boolean
duration of symptoms	[0.5, 52]	years
dyspnoea	0, 1	Boolean

ache on chest	0, 1	Boolean
weakness	0, 1	Boolean
habit of cigarette	0, 1, 2, 3	category
performance status	0, 1	category
white blood	[742, 21500]	cells per mcL (microlitre)
cell count (WBC)	[4, 22]	cells per mcL (microlitre)
hemoglobin (HGB)	0, 1	Boolean
platelet count (PLT)	[111, 3335]	kilo platelets per mcL (microlitre)
sedimentation	[7, 129]	mm/hr (millimetres per hour)
blood lactic dehydrogenise (LDH)	[55, 1128]	IU/L (international units per litre)
alkaline phosphatise (ALP)	[41, 489]	IU/L (international units per litre)
total protein	[3.1, 8.5]	g/dL (grams per decilitre)
albumin	[1.5, 6.9]	g/dL (grams per decilitre)
glucose	[60, 421]	mg/dL (milligrams per decilitre)
pleural lactic dehydrogenise	[110, 7541]	IU/L (international units per litre)
pleural protein	[0, 6.7]	g/L (grams per litre)
pleural albumin	[0, 4.4]	g/dL (grams per decilitre)
pleural glucose	[2, 151]	mg/dL (milligrams per decilitre)
dead or not	0, 1	Boolean
pleural effusion	0, 1	Boolean
pleural thickness on tomography	0, 1	Boolean
pleural level of acidity (pH)	0, 1	Boolean
C-reactive protein (CRP)	[11, 103]	mg/L (milligrams per litre)
OUTPUT		
class of diagnosis	1, 2	(1: Healthy, 2: Mesothelioma)

https://doi.org/10.1371/journal.pone.0208737.t001

In the next table a more descriptive list of attributes is given together with a detailed explanation with supplementary information of each feature.

Table 3.2 - Attributes meaning

ATTRIBUTE	MEANING
ache on chest	presence or absence of pain in the chest area
asbestos exposure	if a patient has been exposed to asbestos during life
cytology exam of pleural fluid	test to detect cancer cells and certain other cells in the area that surrounds the lung
dead or not	if a patient is still alive
diagnosis method	if the patient has had a mesothelioma diagnosed by a common diagnosis method
dyspnoea	shortness of breath
hemoglobin normality test	test that measures how much hemoglobin is in blood
pleural effusion	presence of effusion, common symptom that can inhibit the normal function of the organ
pleural level of acidity (pH)	if the pleural fluid pH is lower than the normal pleural fluid pH, that it's neutral
pleural thickness of thickness	any form of thickening involving either the parietal or visceral pleura
weakness	lack of strength
city	place of provenance of the patients
gender	female or male
habit of cigarette	four categories for the habit of smoking
lung side	the side of the lungs which is experiencing pleural plaques or mesothelioma traces
performance status	patient's ability to perform normal tasks
type of malignant mesothelioma	mesothelioma stage to which the symptoms seem to belong, according to the TNM Classification of Malignant Tumours
age	the age of the patients
duration of asbestos exposure	how long has been the environmental exposure to asbestos
duration of symptoms	the time period, in years, in which the patients show symptoms
albumin	level of blood albumin
alkaline phosphatase (ALP)	test used to help detect liver disease or bone disorders
C-reactive protein (CRP)	acute phase reactant, significantly elevated in patients with pleural mesothelioma (MPM)

glucose	test which measures the amount of glucose in a sample of blood
lactate dehydrogenase test (LDH)	protein that helps produce energy in the body
platelet count (PLT)	test to measure how many platelets patients have in the blood
pleural albumin	level of albumin in the pleural fluid
pleural fluid WBC count	the count of leukocytes in the pleural fluid
pleural fluid glucose	low level can be linked to infection or malignancy
pleural lactic dehydrogenase	its levels indicate if the fluid is exudate or transudate
pleural protein	pleural effusions are classified as transudates or exudates on the basis of the fluid protein level
sedimentation rate	test to measure how quickly erythrocytes settle in a test tube in one hour
total protein	biochemical test for measuring the total amount of protein in serum
white blood cells (WBC)	test measures the number and quality of white blood cells

https://doi.org/10.1371/journal.pone.0208737.t002

Chapter 4

Implementation Pipeline

This chapter presents, the concrete steps in order to carry out the desired analysis on the mesothelioma disease dataset. For that reason, Chapter 4 is split into three subchapters, namely Exploratory Data Analysis (EDA), Data Preprocessing, and Predictive Modelling. In 4.1 the data is explored and potential correlation between dataset's attributes is investigated. This is mainly achieved by plots and visualisations together with statistical measures. Next, 4.2 encapsulates all the necessary procedures demanded so that prepare the data as an appropriate input for the ML algorithms. Finally, 4.3 provides the implementation of the tools used to execute binary classification along with evaluation and performance metrics.

4.1 Exploratory Data Analysis (EDA)

Beginning the analysis, the mesothelioma malignant disease dataset from the UCI Repository is loaded. For a quick recap, it consists of 324 observations with 35 features. There are no missing values or faulty entries, so each attribute contains 324 numeric values representing either a categorical/Boolean variable or a numerical one. The next table splits the features into three categories depending on the type of their values.

Table 4.1 - Features classification by value type

Numerical	*Boolean*	*Categorical*
age	asbestos exposure	gender
duration of asbestos exposure	diagnosis method	city
duration of symptoms	cytology	type of MM
white blood	dyspnoea	keep side
cell count (WBC)	ache on chest	habit of cigarette
platelet count (PLT)	weakness	performance status
sedimentation	hemoglobin (HGB)	

blood lactic dehydrogenise (LDH)	dead or not	
alkaline phosphatise (ALP)	pleural effusion	
total protein	pleural thickness on tomography	
albumin	pleural level of acidity (pH)	
glucose		
pleural lactic dehydrogenise		
pleural protein		
pleural albumin		
pleural glucose		
C- reactive protein (CRP)		

As one may observe the dataset has 17 numerical, 11 Boolean, and 6 categorical variables. Boolean as well as categorical attributes take integer values with some of them being ordinal (i.e. attributes have natural ordered categories) and some not. As a matter of fact, from categorical attributes those that are of ordinal type are: type of MM, and habit of cigarette. Performance status takes values 0 and 1 but it is suggested as categorical, so it is considered as such. Let's first inspect though the balance of the dataset before proceeding to the next steps.

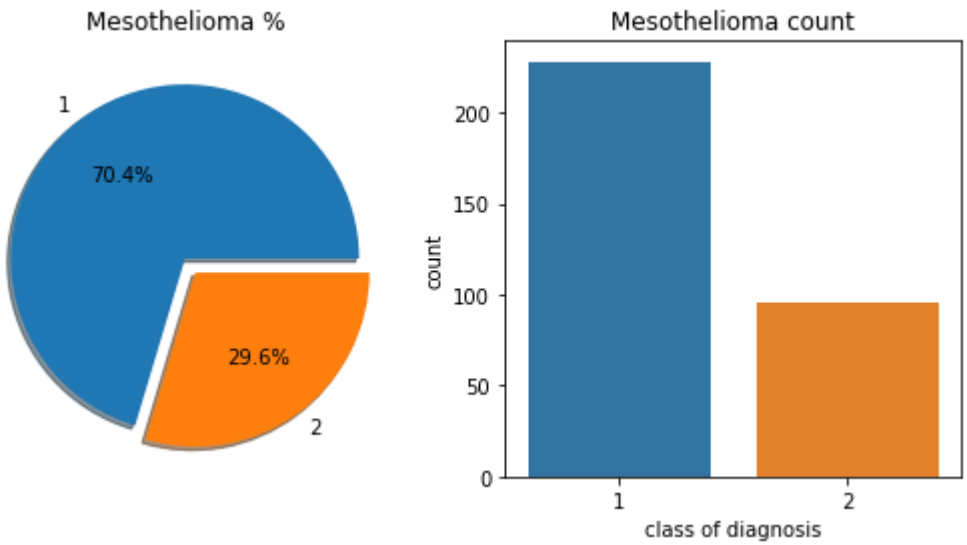

Figure 4.1 - Inspect the balance in the dataset.

The majority of people are healthy (70.4%) and to be more precise 228 people are healthy (class of diagnosis = 1) and 96 people suffer from mesothelioma disease (class of diagnosis = 2). Now, proceed with taking deeper dives into the attributes.

class of diagnosis	1	2	All
gender			
0	83	51	134
1	145	45	190
All	228	96	324

class of diagnosis	1	2	All
type of MM			
0	222	88	310
1	6	5	11
2	0	3	3
All	228	96	324

class of diagnosis	1	2	All
habit of cigarette			
0	126	57	183
1	27	10	37
2	35	19	54
3	40	10	50
All	228	96	324

	dead or not	0	1	All
gender	class of diagnosis			
0	1	7	76	83
	2	3	48	51
1	1	7	138	145
	2	1	44	45
All		18	306	324

Figure 4.2 - Count samples by class of diagnosis, gender, type of MM, habit of cigarette, and dead or not.

The following observations from the above tables can be made:

- ➔ Only 23.6% of the men (gender=1) are infected whereas 38% of the women suffer from mesothelioma. That's strange because usually men are more likely to be exposed to asbestos considering that it is a construction material;

- ➔ 91% of the infected people are diagnosed with the early stage0 of the disease whereas there are 6 people surprisingly with stage1 of the disease that are not infected. On the other hand, there are no healthy people with stage2 of the disease from which one can perhaps suppose that at this level there is no doubt cancer is malignant;

- ➔ From those that are infected 40.6% of them smoke (habit of smoke=0 assumed non-smokers) which seems a reasonable percentage;

- ➔ Only 5% of the people in the dataset in total are alive. From people diagnosed with mesothelioma the possibility of a man and a woman being dead is 97.7% and 94.1% respectively. This comes to reassure the fact that on an average value patients' life expectancy after diagnosed with the disease is unfortunately under a year as experts state.

To continue with let's look at the distribution of numerical attributes in the dataset that lie inside the knowledge of a person with no medical background. Specifically, consider the distribution of age variable, next the duration of symptoms that he/she experienced and finally the duration of asbestos exposure he/she had.

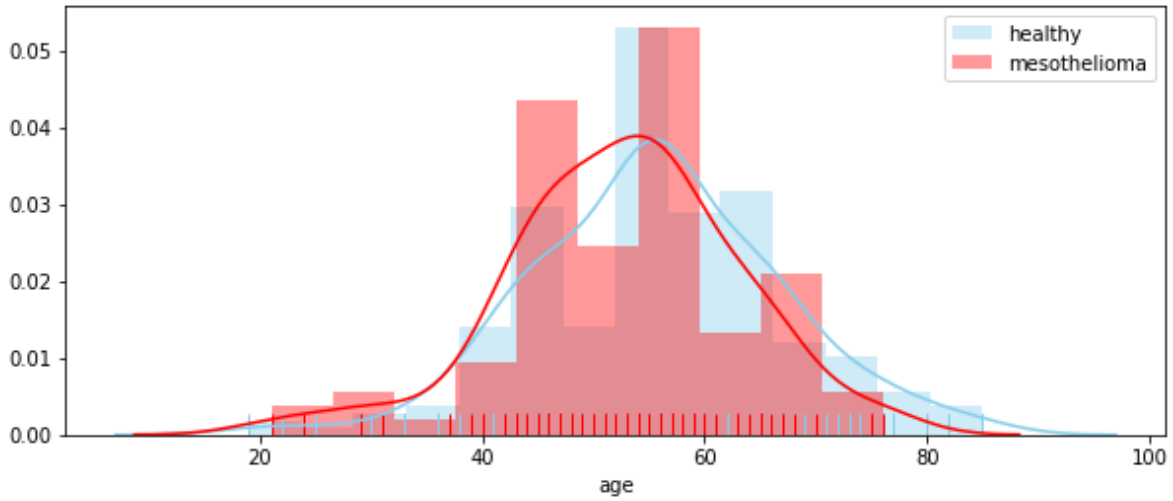

Figure 4.3 - Distribution of age for healthy and infected people in the dataset.

Most people in the dataset are between 45 and 65 years of age regardless of the disease existence. Perhaps this is explained by the attribute duration of asbestos exposure which is illustrated right below.

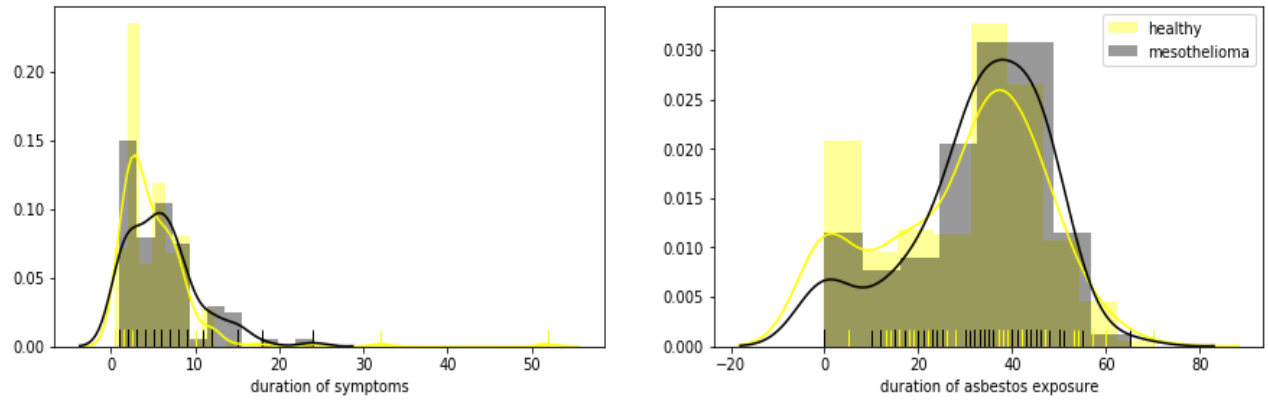

Figure 4.4 - Distribution of duration of symptoms and duration of asbestos exposure in the dataset.

One can clearly observe both patients diagnosed with malignant disease and those not, have experienced asbestos exposure for approximately a mean value of 40 years, a fact that gives an explanation to the mean of age values that takes place in the dataset. On the other hand, feature duration of symptoms is unequally distributed among its values.

Next, have a look to the following plots to gain more information.

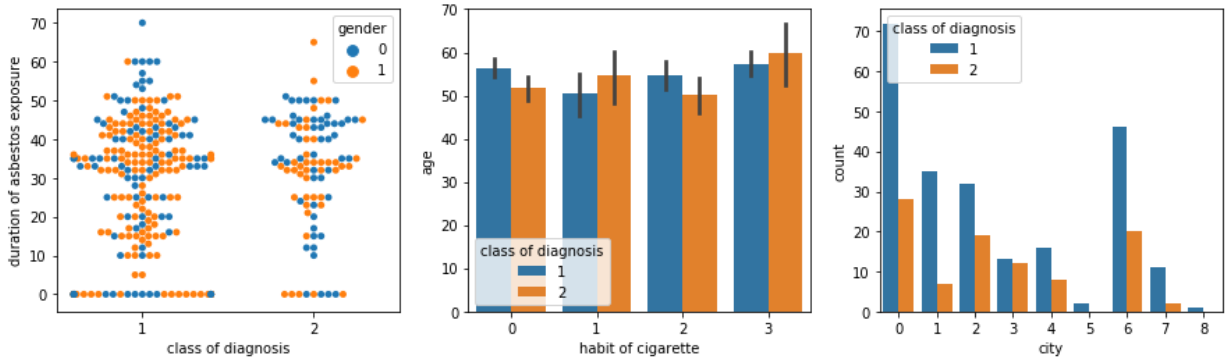

Figure 4.5 - Data exploration for features duration of asbestos exposure, habit of cigarette, city and the target attribute class of diagnosis.

Again, one can notice that:

- ➔ The mean value of the duration of asbestos exposure for people suffering from mesothelioma is 40 years and from that group women seem to experience more years to asbestos exposure;
- ➔ It's interesting that the group of ill people that are excessive smokers has the greatest age variance among other groups. But in general, it looks like habit of cigarette doesn't contain much information about the mesothelioma disease;
- ➔ Most of the measurements have been taken from cities 0 and 6 and from those infected most of them come from city 0.

Consider afterwards some categorical attributes to count their discrete values and observe that:

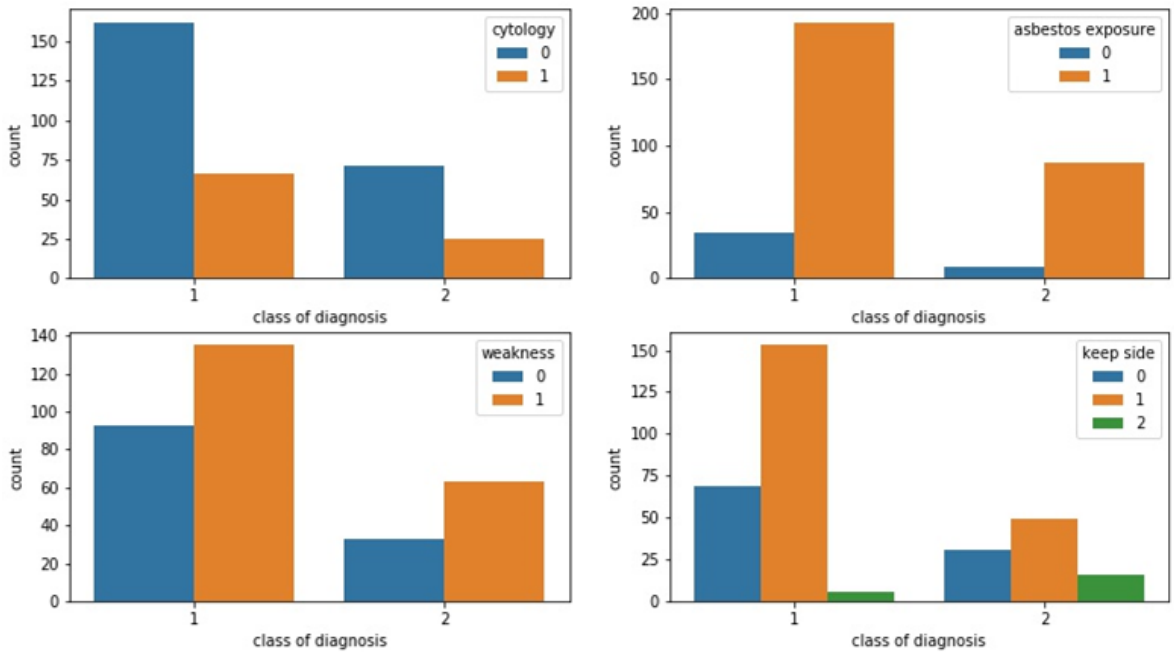

Figure 4.6 - Examples count by cytology, asbestos exposure, weakness and keep side separated by class of diagnosis.

- ➔ only 27% of the people that suffer from mesothelioma have taken a cytology exam of pleural fluid (it is a test to detect cancer cells and certain other cells in the area that surrounds the lungs) which is an interesting fact and the reason might be that patients were infected on a different part of their body;
- ➔ 84.6% of healthy people were exposed to asbestos when 90.6% of people with malignant tumours were exposed to asbestos as expected;
- ➔ 62.5% of the mesothelioma group have experienced a lack of strength, but again weakness doesn't seem yet to be correlated with class of diagnosis;
- ➔ 73% of the people with keep side (the side of the lungs which is experiencing pleural plaques or mesothelioma traces) 2 suffer from the disease.

Carrying on with class of diagnosis versus continue as well as discrete attributes.

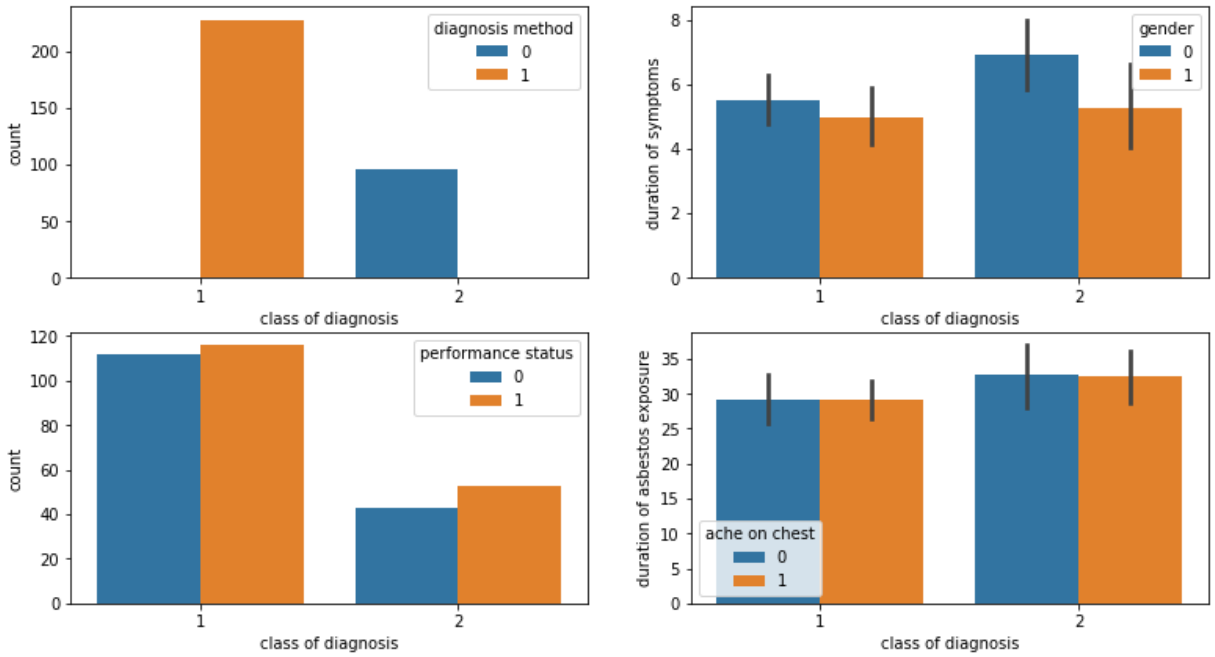

Figure 4.7 - Count and bar plots for categorical and continuous attributes respectively.

Here the following remarks can be made:

- ➔ everyone that took a common diagnosis method had a joyful outcome, whereas unexpectedly not a single person diagnosed with mesothelioma took one of these tests;
- ➔ females suffering from mesothelioma seem to experience more years the symptoms of the disease than men;
- ➔ 50% from the people of class 2 experience problems with performing daily tasks. This also applies to healthy people with class 1 so it may be safe to disregard attribute performance status;
- ➔ from people suffering from mesothelioma both those that experience pain in the chest and those that are not have been exposed to asbestos for a mean value of 33 years.

There are many people in the dataset under 45 years of age that suffer from mesothelioma disease even if there have not been exposed to asbestos. In addition to that people that experience dyspnoea and are infected are more likely to be between 40 and 60 years of age as the next plots illustrate.

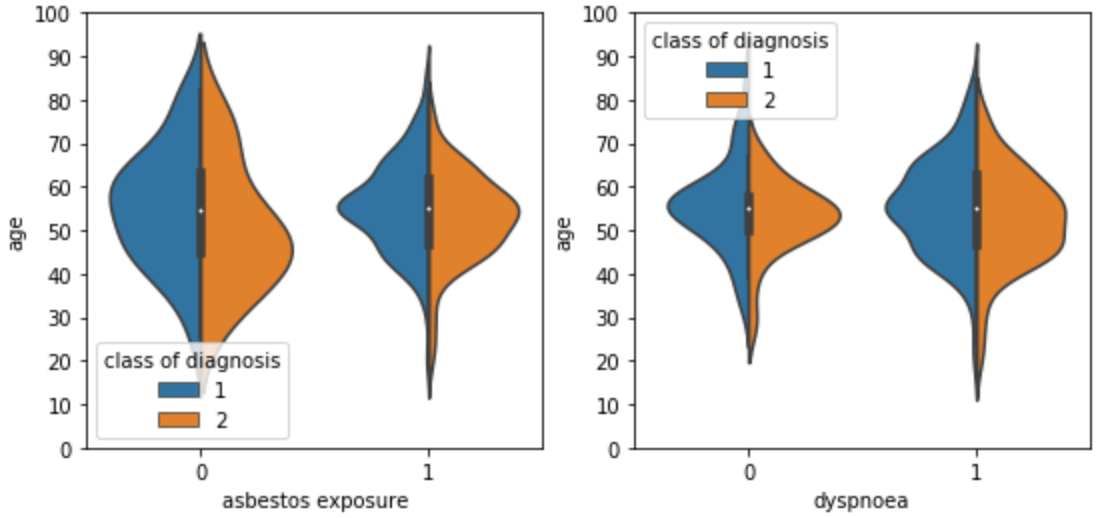

Figure 4.8 - Violin plots for asbestos exposure and dyspnoea versus age.

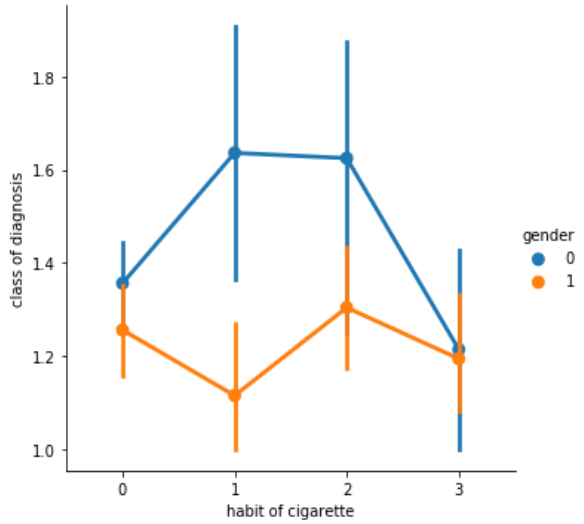

Figure 4.9 - How class of diagnosis differs while varying habit of cigarette and gender.

Before proceeding to take a glance at the numerical attributes, let's observe how class of diagnosis is affected by patients' habit of smoke depending on their gender. Notice that, coloured with blue, women that don't smoke are very likely not to suffer from mesothelioma disease. Nevertheless, surprisingly even though smoking regularly increases the probability of being infected, being a female excessive smoker reduces dramatically the probability of suffering from mesothelioma.

Almost half of the attributes in the dataset are numerical and the vast majority of them involve medical information. Thus, a useful approach would be to look at the correlation matrix between those attributes once lack domain knowledge in the subject exists. Even if it wouldn't though this can be a good technique to extract helpful knowledge and produce afterwards a more effective classification model. In the next figure a heatmap of a correlation matrix between all numerical attributes in the mesothelioma dataset is provided to find out that indeed there exists a special close connection.

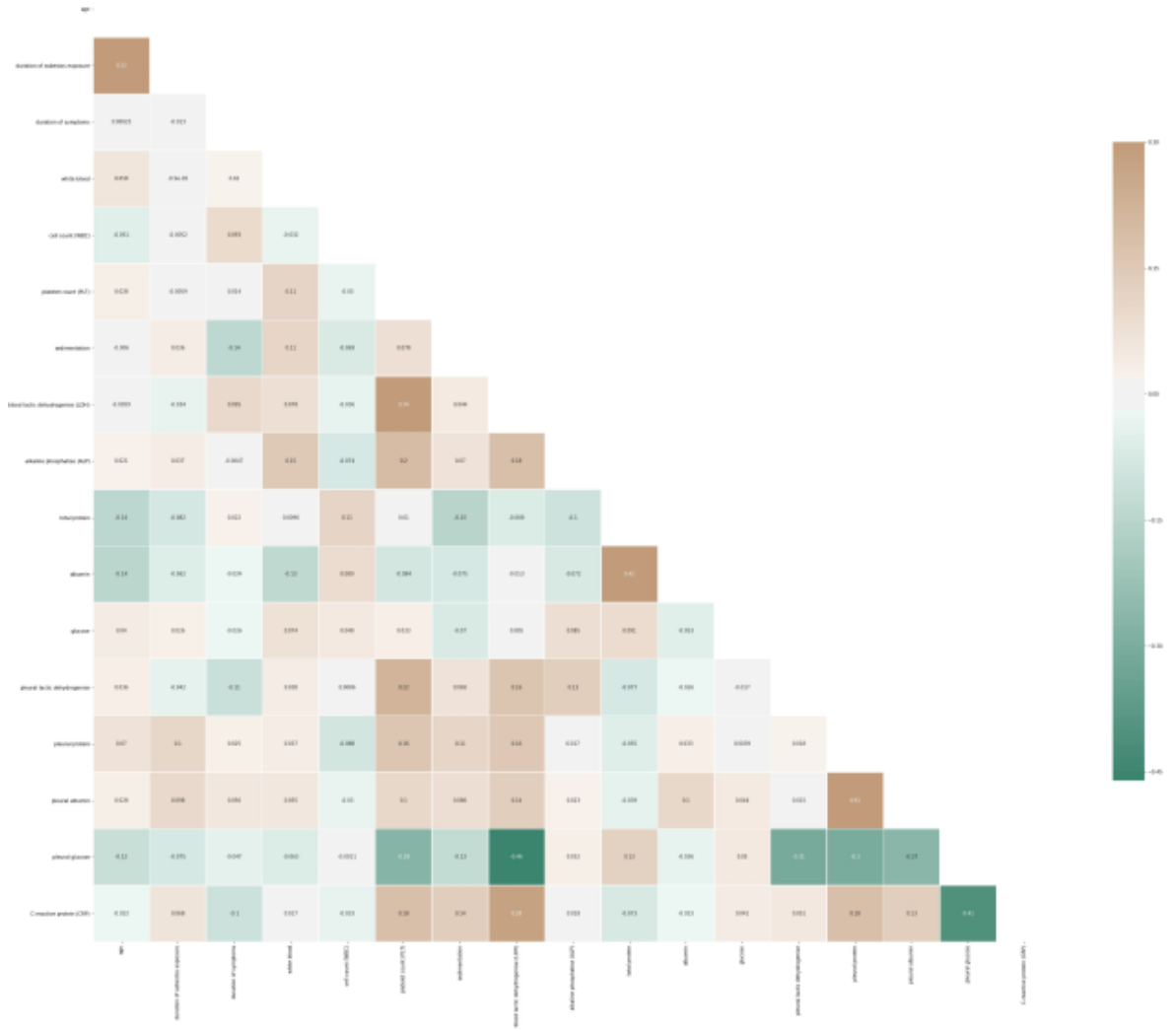

Figure 4.10 - Heatmap for continuous attributes. The browner a cell is the more correlated the associate attributes are.

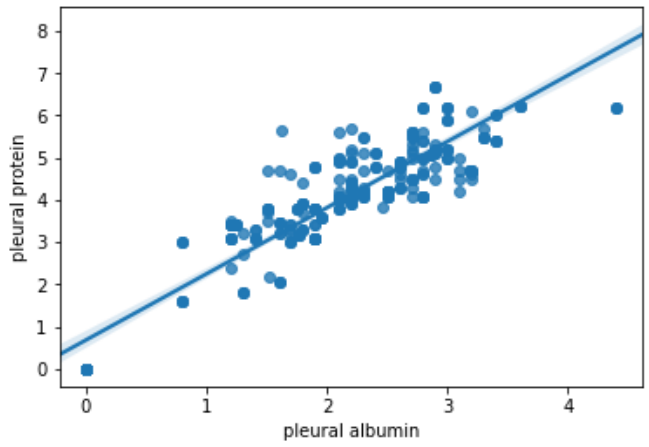

Figure 4.11 -A Linear Regression fit to pleural albumin and pleural protein.

Attributes pleural albumin (level of albumin in the pleural fluid) and pleural protein (pleural effusions are classified as transudates or exudates on the basis of fluid protein level) are highly correlated and as proof of this a simple linear regression model is fit to this data.

4.2 Data Preprocessing

The mesothelioma disease dataset allocated on the UCI Repository has no missing values, it is clean, and it already has label encoded all 6 categorical attributes that is contains. Nevertheless, as stated above, not all categorical features are ordinal and for that reason dummy encoding is used to attributes keep side and city. This method automatically increases the dimension of the dataset which might cause weakness to the models and hence a feature selection and/or extraction method eventually should be handy.

So, for now let's gradually explore all numerical attributes to detect outliers and shape a more delegate dataset.

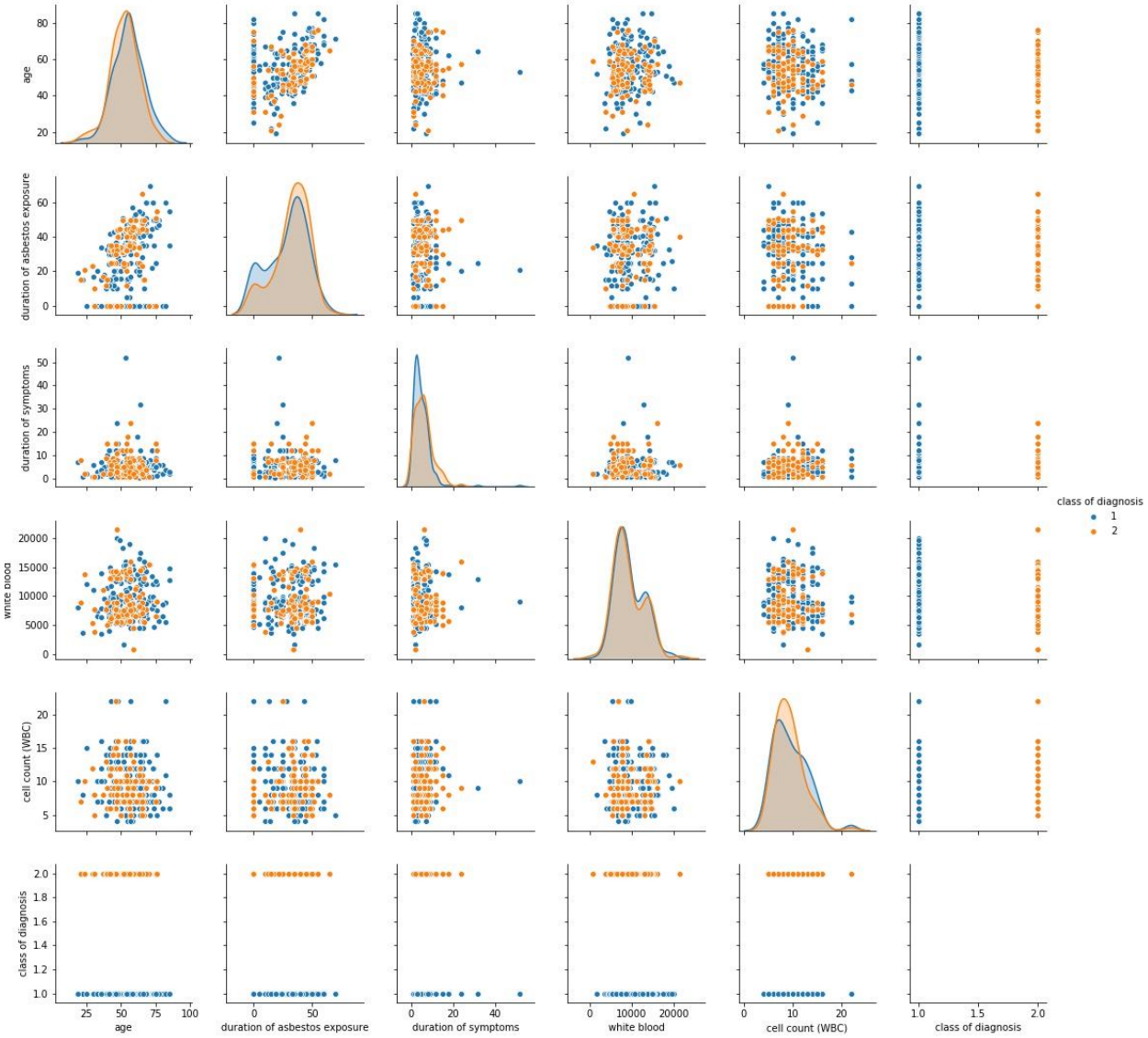

Figure 4.12 - Pair plot of numerical attributes separated by class of diagnosis (part 1).

To avoid overfitting not all outliers are excluded from the dataset. As it can be seen from above attributes duration of symptoms and cell count (WBC) contain outliers, so remove samples for which:

Duration of symptoms ≥ 20, and cell count (WBC) ≥ 20

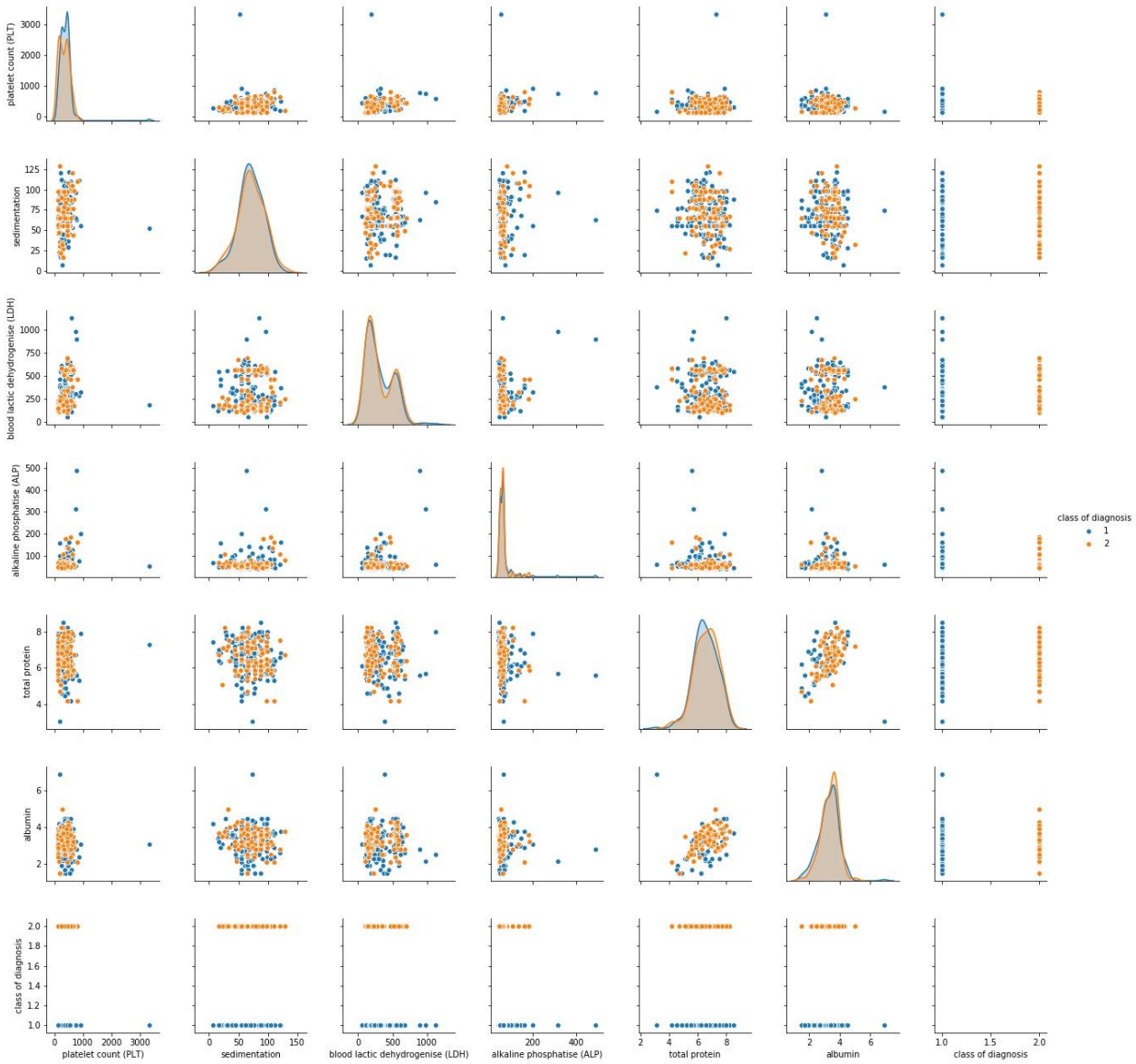

Figure 4.13 - Pair plot of numerical attributes separated by class of diagnosis (part 2).

Next several outliers appear on the next numerical attributes. Precisely to erase instances for which:

platelet count (PLT) > 1000, blood lactic dehydrogenise (LDH) $\geq$ 850,
alkaline phosphatise (ALP) > 400, total protein < 4, and albumin > 6.

So far, the number of observations in the dataset is reduced by 4.3%. Let's have a look at the remaining numerical variables of the mesothelioma dataset.

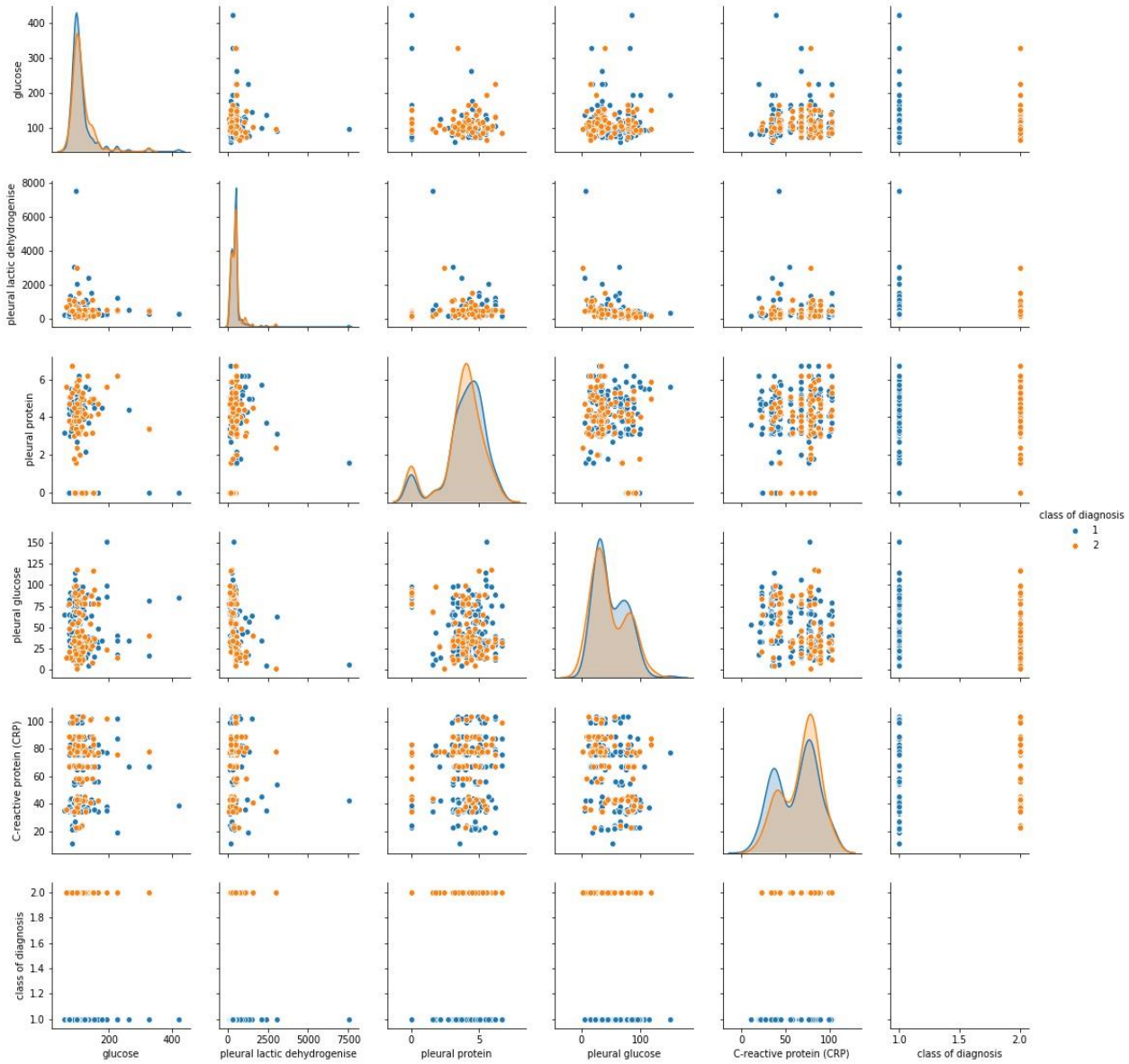

Figure 4.14 - Pair plot of numerical attributes separated by class of diagnosis (part 3).

Some small eliminations can be done here and particularly for instances which hold values that have:

glucose > 300, pleural lactic dehydrogenise > 5000, and pleural glucose > 130

At the end one ends up with a new dataset of 304 examples and 45 features (remember an extra 11 new dimensions were added when dummy encoded attributes keep side and city that contained 3 and 8 discrete values respectively). As a result, the size of the dataset has been shrunk for about 6.1% in total.

Some exploration next on the new smoothed dataset. Let's observe for example the duration of asbestos exposure a patient experienced versus the attribute habit of cigarette, separated by the class of diagnosis as most of the times. Also, look at the feature duration of symptoms against the type of malignant mesothelioma a patient experienced again, separated by the class of diagnosis. The distribution as monitored in both plots is uniformly expanded.

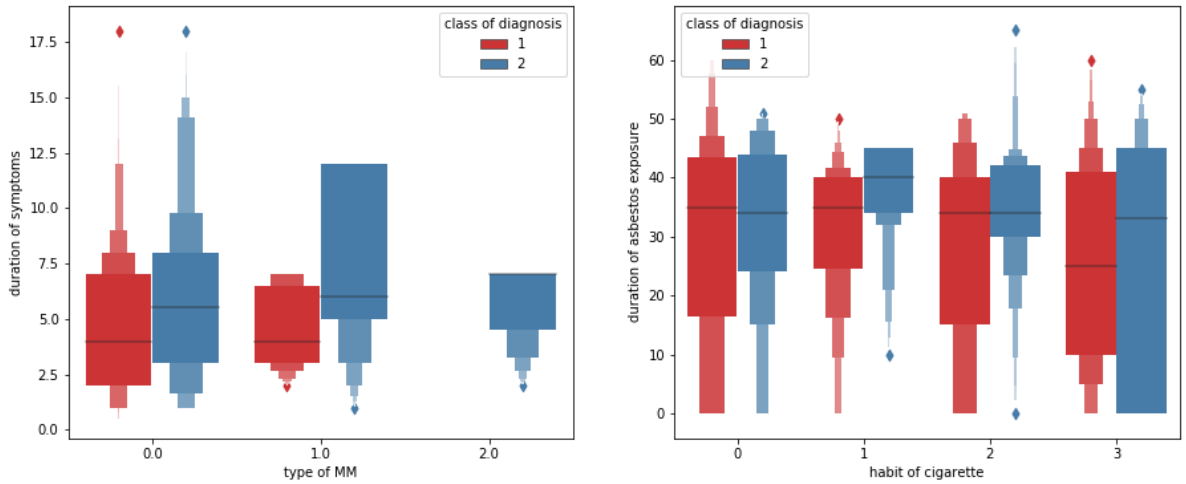

Figure 4.15 – Box plots applied on the processed dataset.

Then, consider two shaded bivariate densities for healthy people and for people suffering from mesothelioma disease. For both cases plot the duration of symptoms patients experienced versus their age. Most healthy people have faced symptoms of the disease for at least 2 years while their age is around 58 years. Whereas people with malignant mesothelioma disease have encountered the disease symptoms mostly for at least 5 years and at the same time their larger part are at about 45 years of age.

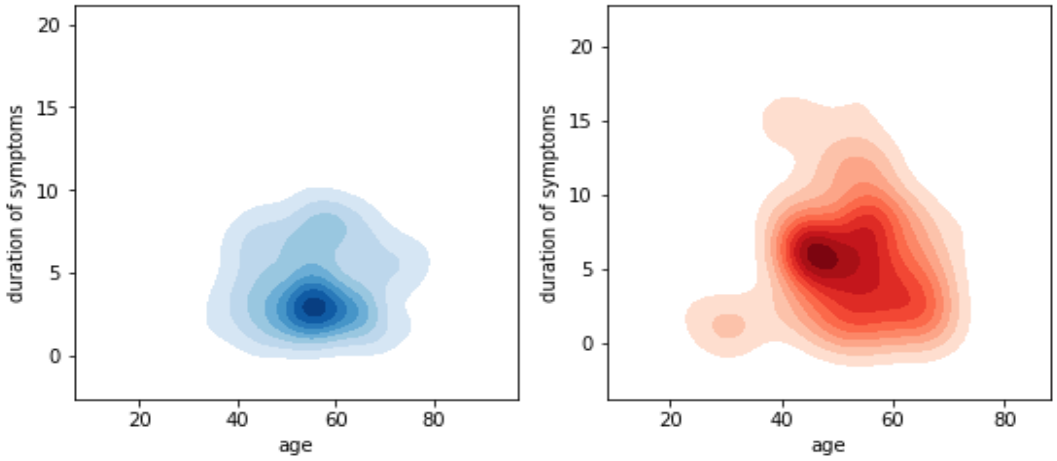

Figure 4.16 - Two shaded bivariate densities.

Now, as a last step before proceeding to implement the classification algorithms one must scale the data. This is a usual procedure for many ML estimators especially when dealing with attributes on a different scale as it happens here, where for example there exist attributes counting years and at the same time others counting the size of the cells. To that end, it is mandatory to use standardisation to scale the independent variables of the data and in this procedure the standard score of a sample x is calculated as $z = \frac{x-u}{s}$, where u is the mean of the training samples, and s is the standard deviation of the training samples. This method removes the mean and scales the data to unit variance. It is also important to compute the mean and the standard deviation considering only the training set and then use that for later scaling. In addition, it is obvious that scaling is needed in both train and test sets or otherwise the algorithm will produce misleading output. The process of scaling the data in the implementation is embedded in a pipeline and it is followed by the classification process.

Standardization:

$$z = \frac{x - \mu}{\sigma}$$

with mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i)$$

and standard deviation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Figure 4.17 - Standardisation formula

4.3 Predictive Modelling

Reaching the final and most important part of the analysis, the time has come to implement the appropriate ML classification algorithms to predict malignant mesothelioma disease. This is achieved by gradually getting to the desired point of high-performance rates with the use of preprocessing methods, hyperparameter optimisation techniques, as well as validation processes. For the sake of the analysis the selected mesothelioma dataset must be split into a train and a test subdataset. The first as the name reveals, is used to train the algorithm and form its necessary parameters so that get ready next to be fed with the unseen data, namely the test dataset. And always keeping in mind that the aim is to classify each patient as healthy or unhealthy depending on the prediction.

Figure 4.18 - The processes that the predictive model lifecycle consists of[1].

So to begin with, choose at first to fit each algorithm purely to the train dataset without any

[1] S. Balanchine, "Challenges & Requirements for Building a Predictive Analysis Model" https://bit.ly/2J5bb7R

preprocessing technique applied or any other alteration made and thereupon attempt to predict the target labels on the test dataset. By approaching the problem in this way, it becomes easier later to notice and highlight any potential performance improvement after properly preparing the data according to each algorithm's needs. Nevertheless, evaluation measures are not always increased after some data handling as long as there exist some occasions eventually, as presented afterwards, when an ML algorithm reaches high accuracy measures regardless of any prior data manipulation. The reason behind this can vary from the ability of the algorithm to easily generalise, to its robustness to outliers.

Carrying on with the next stage of the predictive modelling part of the analysis, the natural procedure of executing classification tasks on a given dataset is followed. Hence, collect the new dataset formed after performing the preprocessing tasks suggested in the previous sub chapter. Particularly to sum them up those were first drop features that are proven to be correlated with other variables to reduce the dimensions of the dataset. Next, label encode all the categorical attributes that were not already label encoded in the first place. This is done by executing dummy encoding which is nothing more than defining a new dimension in the dataset for each possible value of a categorical feature, and as a result this new feature becomes a binary variable easy and safe to use. Going on with scaling the data on both train and test datasets by means of standardisation methods as previously introduced. Thus, now it is time to fit each algorithm to the new dataset and investigate through several performance metrics how effective the established predictions were. Some of these metrics and methods include cross validation, macro average accuracy score, ROC curve and confusion matrix. Furthermore, an attempt to perform hyperparameter optimisation as well as feature extraction and oversampling is performed in order to handle the class imbalance and the small size of the dataset.

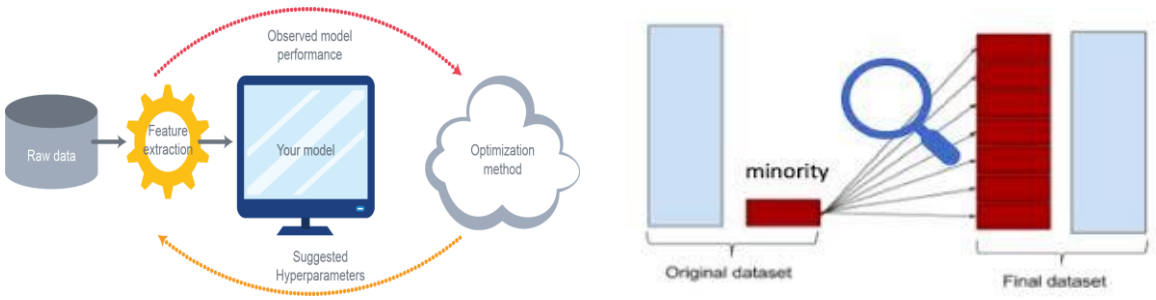

Figure 4.19 - Hyperparameter optimisation process[1] and oversampling the underrepresented class[2] .

[1] P. Barba, "Hyperparameter Theft" https://bit.ly/2VWfZ4F
[2] A. Patil, "Dealing with Imbalanced Data" https://bit.ly/2BspejS

Chapter 5

Results and Discussion

5.1 Algorithms Individual Results

In this chapter, the individual results of each algorithm are presented as well as an aggregate comparative analysis on the outcomes received from the classification algorithms. This includes statistical measures, plots as well as confusion matrices both prior and after the preprocessing methods that were applied to the mesothelioma dataset. In addition, information is included concerning the results of specific classification algorithms such as MLP or SVM for which supplementary techniques like oversampling, and/or feature extraction were added to enhance their efficiency and demonstrate the contrast.

5.1.1 Decision Tree

Decision tree classification algorithm belongs to the group of those algorithms that reached high performance rates regardless of any preprocessing method or any other data manipulation applied to the mesothelioma dataset. This also includes the case where the algorithm was fit to the purely imported dataset without even encoding the necessary variables. The decision tree drawn on the left was produced after applying the algorithm on the preprocessed dataset which consists of 304 examples with 45 features. As it indicates attribute X4, which is type of MM, was able to completely distinguish patients' diagnoses for each of 182 samples that are contained on the train dataset. Precisely, decision tree classifier scored a 100% balanced accuracy score. For a reminder balanced

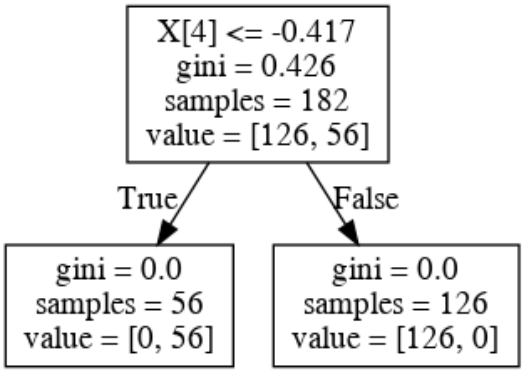

Figure 5.1 - DT for mesothelioma dataset

accuracy score is defined to be the average of recall obtained on each class. As a matter of fact, when plotting the confusion matrix of the DT classifier using a 10-fold cross validation evaluation method, the algorithm manages to correctly predict 211 healthy patients and 93 patients that suffer from malignant mesothelioma. Remember by removing the outliers this immediately reduces the original size of samples for about 6.2% of the total examples.

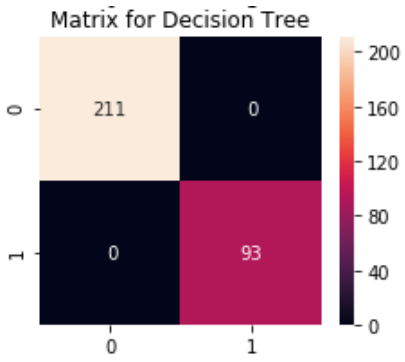

Figure 5.2 - Confusion matrix for DT.

5.1.2 k-Nearest Neighbour (kNN)

There were two classification algorithms during the prediction analysis step of the work that appeared an absorbing performance because they gradually exhibited an increase of their evaluation metrics, while executing hyperparameter optimisation and other tuning methods. kNN classifier was indeed one of the above algorithms specified and thus let's consider its achievements and assess the success of the model. At first, just like in any other case investigate how well the model performs on the pure subset of the given dataset as it is offered on the UCI library. It is immediately noticed a low accuracy of value around 60% on the test dataset but as the dataset is not properly prepared for the required predictive modelling, it is demanded next to proceed with the preprocessing phase. The performance of the algorithm rose 11.6% to reach a balanced accuracy score of around 65%, which encouraged later new approaches to be considered. But for now, highlight the accuracy above which was managed with k=3, the distance between points being the Euclidean metric and all points in each neighbourhood were assigned the same weight. The confusion matrix can give a better insight but even better observe the ROC curve, AUC and precision-recall curve to have a more solid overview of the classifier.

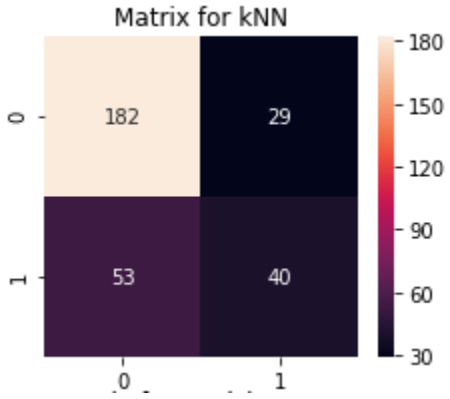

Figure 5.3 - Confusion matrix for kNN.

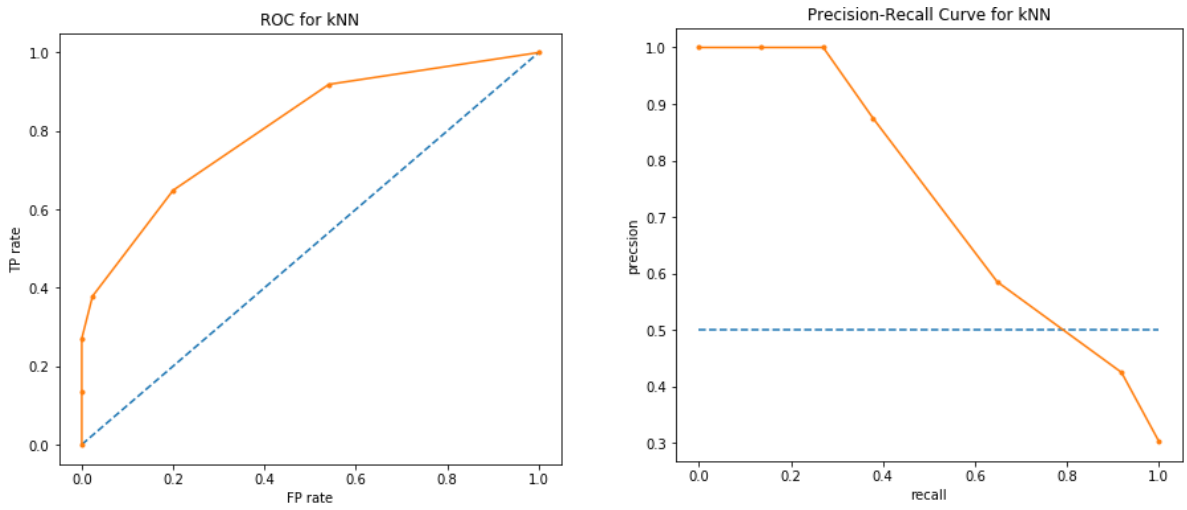

Figure 5.4 - ROC and precision-recall curves for kNN (k=5).

At first glance the model exhibits some skill with the AUC being equal to 0.806. On the right of the ROC the precision-recall curve is specialised to binary classification problems and it clarifies that the model needs extra manipulations once a model with perfect skill is depicted as a point at [1.0, 1.0], and a skilful model is represented by a curve that bows towards [1.0, 1.0] above the flat line of no skill. Therefore, next execute a best fit exploration starting by examining the performance of kNN for different odd values of k between 1 and 21, keeping the rest of parameters fixed. Thus, on the left plot one monitors how accuracy is affected by different values of k and observes that best accuracy is 72.56% which is achieved for k=3. Nevertheless, there is more that can be done and let's try to detect that straight away by considering all those possible combinations for k=[3, 5, 7, 9, 11, 13] and metric=['Minkowski', 'Euclidean', 'Manhattan', 'Chebyshev']. After a 10-fold cross validation as usual the accuracy rose to 82.96% accomplished by k=7 and the metric being Manhattan with p=2.

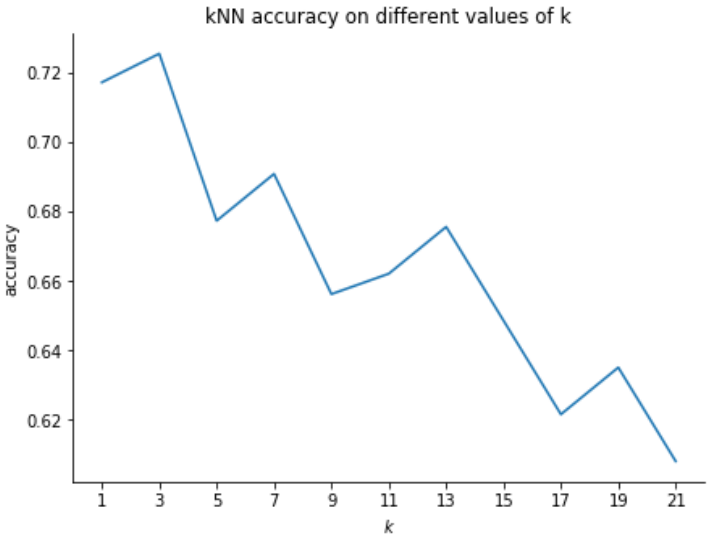

Figure 5.5 - Accuracy behaviour while varying k.

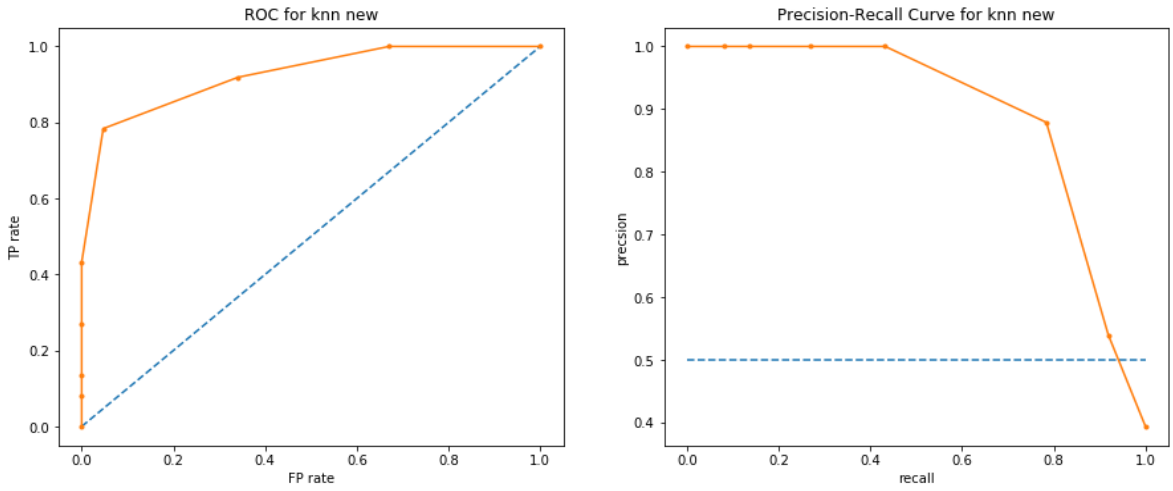

Figure 5.6 - - ROC and precision-recall curves for kNN (k=3).

It can directly be seen the improvement the model experienced both on the ROC and the precision-recall curve approaching the points [0.0, 1.0] and [1.0, 1.0] respectively. In the following table the most important metrics obtained for the kNN classification algorithm are summed up.

Table 5.1 - kNN parameters

kNN	f1-score	precision	recall	AUC
k=3, metric=Euclidean	69.87%	82.9%	77.97%	0.806
k=7, metric=Manhattan	74.7%	90.09%	71.62%	0.924

5.1.3 Logistic Regression

Another algorithm that belongs to the group of those that reached exceptional performance metrics is logistic regression. The confusion matrix as shown next can reassure, that similarly with the DT classifier case, LR successfully classified all samples of both classes. In particular, the model took only 7 iterations to converge when the maximum number of them was set to 100. Moreover, the tolerance parameter which makes the optimisation algorithm to terminate was decided to be 1e-4 and the inverse of regularisation strength parameter C, which is responsible for avoiding the risk of overfitting, was given the value 1. Finally, for the LR classifier it was used the L2-norm for penalisation and that is:

$$||v||_2 = \sqrt{||v_1||^2 + ||v_2||^2 + \ldots + ||v_n||^2}.$$

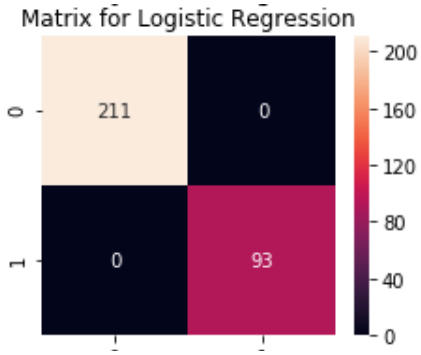

Figure 5.7 - Confusion matrix for LR.

5.1.4 Naive Bayes Classifier

Although it is well known that NB classifier assumes strong independence between features, it can in practice produce realistic results and work efficiently. Applying NB to the mesothelioma disease dataset produced 100% balanced accuracy score whatever preprocessing method had been applied to the data earlier. Particularly, the likelihood of the features was assumed to be Gaussian:

$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_i-\mu_y)^2}{2{\sigma_y}^2})$, where parameters σ_y and μ_y were estimated using maximum likelihood. Also, the portion of the largest variance of all the features that is added to variances for calculation stability and named var_smoothing was determined to be 1e-09 in order to improve prediction strength and enhance the overall evaluation metrics.

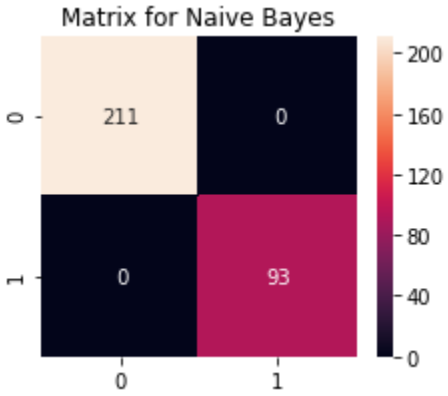

Figure 5.8 - Confusion matrix for NB.

5.1.5 Support Vector Machine (SVM)

When it comes to the SVM classifier, two different kernels were tested to examine the difference of their performance. The first kernel that was used is linear where the classifier didn't face any difficulty in predicting exceptionally all samples of each class just like DT, NB, and LR. Analogously with the LR classifier the parameters of the model were C=1, tolerance=1e-4, and the norm for penalisation was again the L2-norm. Nevertheless, in this case the algorithm took 498 iterations to converge in contrast with the LR classifier. On the other hand, rbf (radial basis function) kernel, which as a reminder is given from the formula $K(x, x') = exp(-\frac{||x-x'||^2}{2\sigma^2})$, where x, x' denote two samples of the dataset, after a 10-fold cross-validation method reached a balanced accuracy score of 97%. The algorithm wrongly classified 8 patients as healthy and 1 patient as sick. In addition, during the hyperparameter optimisation there were different values tested for parameters C, gamma (the kernel coefficient), and the kernel itself of course between rbf and linear in order to examine the best combination among them. It turned out after 720 different fits that the penalty parameter C of the error term could be reduced to a half and the tolerance for stopping criterion to be increased to value 1e-2, all these under the constraint of a linear kernel.

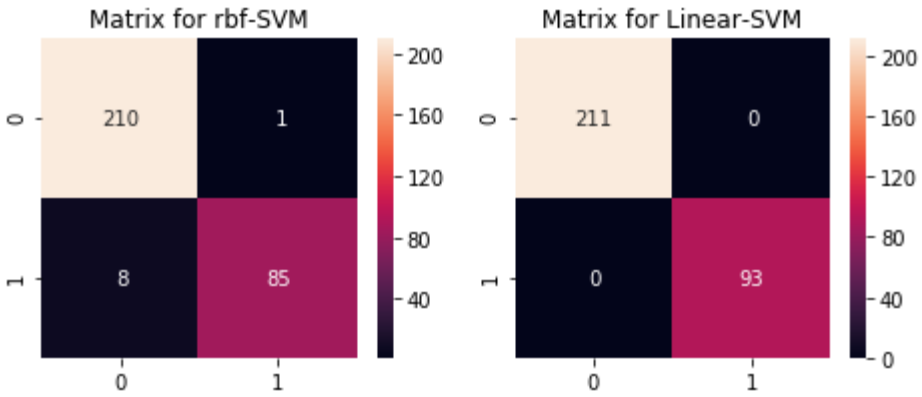

Figure 5.9 - Confusion matrices for SVM using radial and linear kernels respectively.

5.1.6 Artificial Neural Network (ANN)

Finally, the last but undoubtedly the most interesting classification method turned out to be the Multilayer Perceptron algorithm with back propagation learning method. Progressively, the escalation that the performance of MLP developed was outstanding starting from around 50% and reaching at the end 96%. Initially, the algorithm was applied to the purely imported dataset to establish a rough idea. MLP scores around 55% accuracy and carrying on with the more interesting part: the algorithm applied to the processed dataset. There, the model reaches a balanced accuracy score of 76% with the parameters being as follows:

Table 5.2 - Parameters of MLP applied on processed dataset

parameter	*value*	*description*
Activation function	Relu - the rectified linear unit function, returns f(x) = max(0, x)	Activation function for the hidden layer
alpha	1e-04	L2 penalty (regularization term) parameter
solver	Adam i.e. a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba	The solver for weight optimization
beta1	0.9	Exponential decay rate for estimates of first moment vector in adam, should be in [0, 1)
beta2	0.999	Exponential decay rate for estimates of second moment vector in adam, should be in [0, 1)
epsilon	1e-08	Value for numerical stability

Hidden layer sizes	(100,)	The ith element represents the number of neurons in the ith hidden layer
Learning rate	constant	Learning rate schedule for weight updates (constant, invscaling, or adaptive)
Learning rate constant	1e-03	
Maximum iterations	200	this determines the number of epochs (how many times each data point will be used), not the number of gradient steps
tol	1e-04	Tolerance for the optimization. When the loss or score is not improving by at least tol for n_iter_no_change consecutive iterations
n_iter_no_change	10	Maximum number of epochs to not meet tol improvement

Looking next at the ROC and precision-recall curves to explore the model a bit more. The AUC equals 0.806 and the algorithm exhibits some skill but as later be seen the model accommodates a bit more modifications so that to achieve better results.

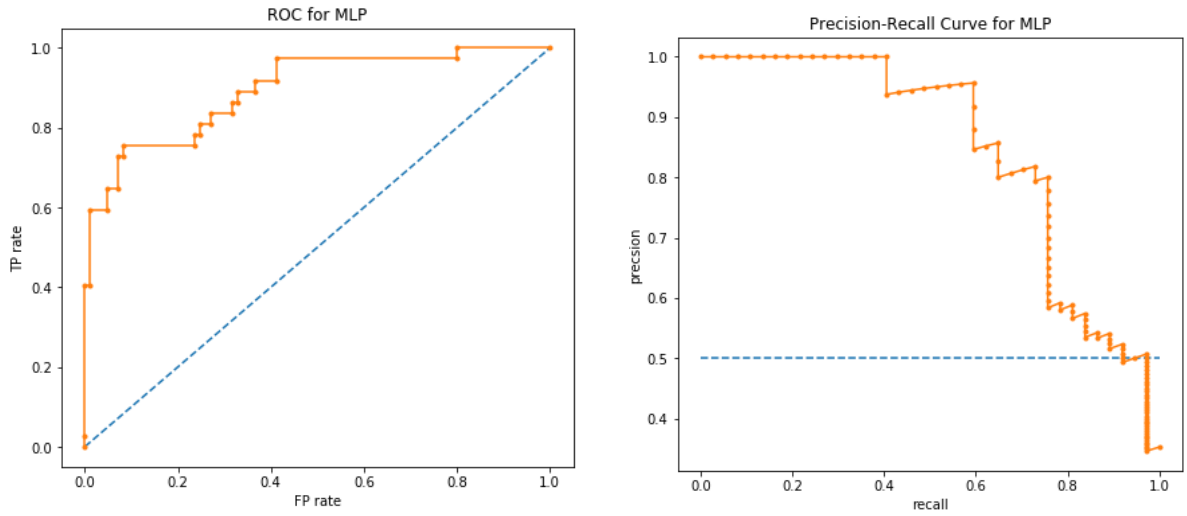

Figure 5.10 - ROC and precision recall curves for MLP applied on the processed dataset (hidden layers size = 100).

For that reason, let the size of hidden layers as well as the activation functions to vary to test each possible combination and find the best fit. These values were: hidden_layer_sizes= [(10,), (20,), (50,), (100,), (200,)], and activation fn = ['identity', 'logistic', 'tanh', 'relu']. It turned out that for hidden_layer_sizes = (200,0) and activation function = 'identity' the model reached the fascinating percentage in balanced accuracy score of 98.65% and the new corresponding ROC and precision-recall plots are:

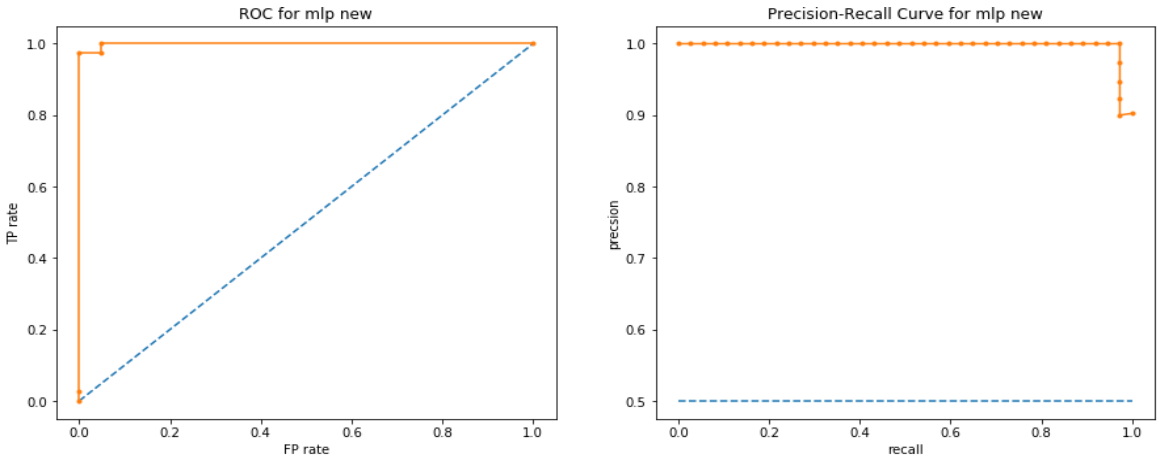

Figure 5.11 - ROC and precision recall curves for MLP (hidden layer size =200).

It is already noticed and highlighted the imbalance occurred in the dataset and due to the small number of samples, for the last part on the MLP classifier case, an attempt to conduct two different techniques simultaneously is performed in order to elaborately examine the problem of binary classification on the mesothelioma dataset. The first of these is oversampling which attempts apparently to generate more samples for the under-represented class, that is people suffering from mesothelioma (target class = 2), so that it increases its population. For this purpose, the SMOTE (Synthetic Minority Over-sampling Technique) method to resample the data is used.

To illustrate how this commonly used algorithm works consider some training data which has s samples, and f features in the feature space of the data. Note that these features, for simplicity, are continuous. As an example, consider a dataset of birds for classification. The feature space for the minority class for which one wants to oversample could be beak length, wingspan, and weight (all continuous). To then oversample, take a sample from the dataset, and consider its k nearest neighbours (in feature space). To create a synthetic data point, take the vector between one of those k neighbours, and the current data point. Multiply this vector by a random number x which lies between 0, and 1. Add this to the current data point to create the new, synthetic data point. Many modifications and extensions have been made to the SMOTE method ever since its proposal. For the mesothelioma dataset specifically the parameters of the algorithm were: $k = 5$, sampling_strategy= resample all classes but the majority class and at the end it increased overall 38.4% the size of the train dataset.

On the other hand, while an attempt was made to bring some balance to the dataset, it seems essential to reduce its feature space once after label encoding the necessary attributes the dimension of the dataset has risen to 44 features. This is done by applying the PCA method, which after all reduced in this case the features to a total number of 15. Testing different parameters for the new balanced and shrunk dataset the new modified MLP algorithm scored 95.6% accuracy after a usual 10-fold cross validation method. The selected parameters for the best fit were variance_threshold=0.005, pca_components=5, mlp hidden layer sizes= (20,), and mlp activation fn='identity'.

5.2 Comparative Analysis

Now that the discovery of the individual performances of each classification method has ended, it is safe to continue with the comparative evaluation of algorithms' accuracies. This is accomplished under the constraint that the mesothelioma dataset is processed in the way that each algorithm demands so there is no room left to make faulty deductions. To that end, conduct a 10-fold cross validation technique in order to on the one hand receive a better estimate of the models' performances and on the other hand to handle the imbalance of the dataset. Hence, the following table displays the aggregate mean as well as the standard deviation of all accuracies obtained from each fold for every algorithm. Observe that there exist particular algorithms that did not achieve exceptional figures and these were SVM using a radial kernel, kNN (with k=3), and MLP. Nevertheless, the two latter methods have even experienced performance improvements based on feature extraction and oversampling the under-represented class. Next, take a glance at the bar plot which encapsulates the information of the above table in a more friendly and colourful way. But never forget that one must take into account the confusion matrices that present a more comprehensive way of the results. These together with the statistical measures contained in the table at the end of this subchapter will point us to the most successful and effective binary classification procedure that was used on the selected dataset.

	CV Mean	Std
Linear Svm	1.000000	0.000000
Radial Svm	0.996667	0.010000
Logistic Regression	1.000000	0.000000
KNN	0.775914	0.135085
Decision Tree	1.000000	0.000000
Naive Bayes	1.000000	0.000000
MLP	0.921720	0.091506

Figure 5.12 - Cross validation mean and standard deviation scores for all algorithms.

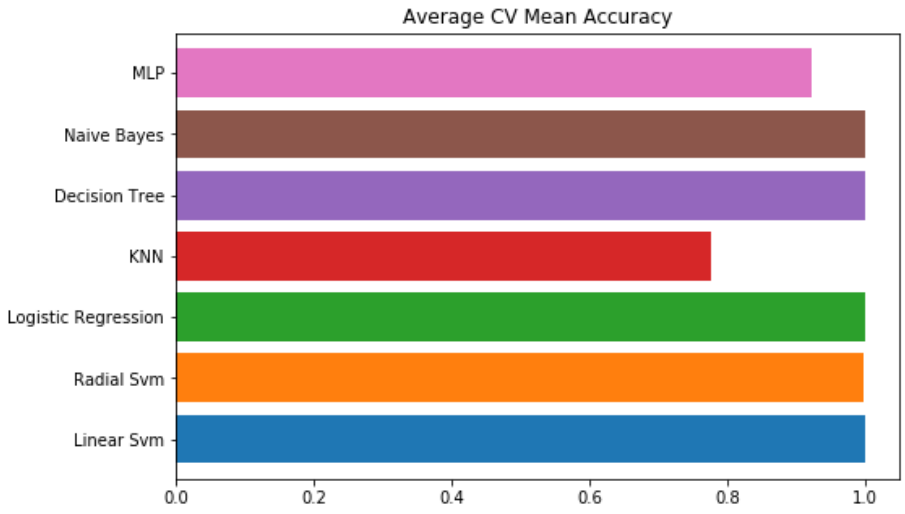

Figure 5.13 - Average CV Mean Accuracy

Let’s glance now at all confusion matrices of the 7 algorithms applied to the mesothelioma disease dataset and always keeping in mind they were produced using a 10-fold cross validation technique.

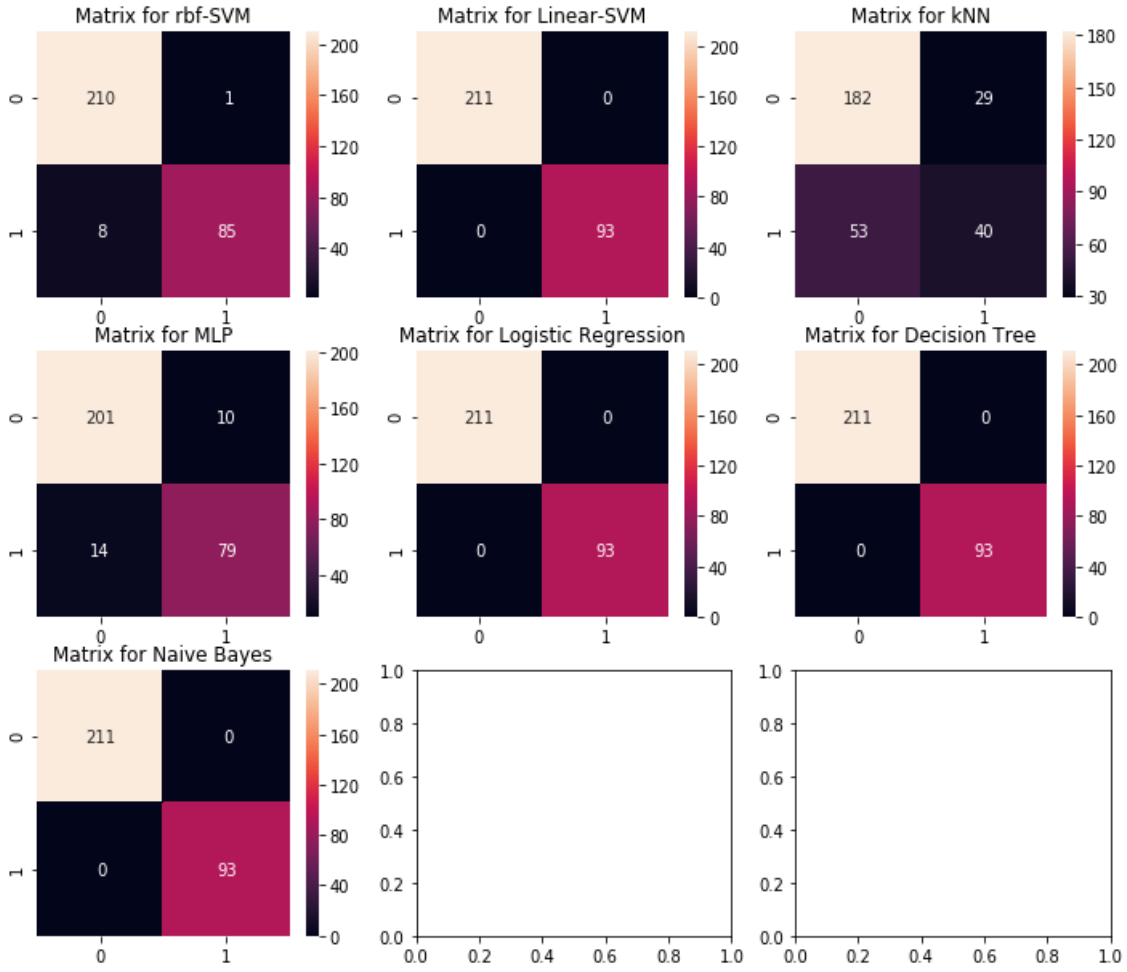

Figure 5.14 - Confusion matrices for every algorithm applied on the dataset.

For a final comparison let’s present the measures precision, recall, f1-score as well as AUC to confirm the results escalation. This includes the cases of the algorithms collocated with default parameters, with hyperparameter optimisation and with feature extraction wherever applied.

Table 5.3 - Aggregate performance table

algorithm(s)	*accuracy*	*precision*	*recall*	*f1-score*	*AUC*
NB, SVM, LR, DT	100%	100%	100%	100%	100%
kNN (k=3)	72.56%	82.9%	67.74%	69.87%	80.6%
kNN (k=7)	82.96%	90.09%	71.62%	74.69%	92.4%
MLP (default)	78.73%	85.22%	78.73%	80.94%	92.9%
MLP (grid search)	98.65%	99.42%	98.65%	99.02%	100%
MLP (PCA, SMOTE)	95.6%	97.01%	95.22%	96.05%	99.36%

It is evident that the algorithm that features the most impressive results is the MLP classifier. Starting from the implementation to the purely imported mesothelioma dataset and ending to the full processed dataset with balancing and feature extraction methods, MLP distinguishes itself from other classification techniques for its unique property to adapt to new data and successfully predict target labels. On the other hand, though kNN algorithm also exhibited an interesting performance while altering parameters to make the model discover its best fit. As a matter of fact, even if by testing different values of k, k=3 was discovered to be the best choice for the number of nearest neighbours, it turned out that k=7 was producing even better balanced accuracy score (namely 82.96%) as long as the distance metrics were varied with the best being Manhattan with p=2. Lastly, the group of those algorithms reaching exceptional results from the first place unexpectedly have proven to be unaffected from several oversampling techniques that were used to bring back balance in the dataset. Particularly, only LR algorithm revealed a minor decrease in its performance while executing Random Over Sampler (over-sample the minority class by picking samples at random with replacement) or even SMOTE. The rest displayed undeviating outcomes that compensate the opinion that due to their simplicity they cannot generalise effectively to new unseen data.

Chapter 6

Conclusions and Proposals for future work

6.1 Conclusions

In this thesis, the problem of executing binary classification on the malignant mesothelioma disease dataset located on the UCI ML Repository was addressed. The dataset consists of 324 instances each of which corresponds to measurements of a single patient. Features vary among different medical characteristics and the target attribute “class of diagnosis” indicates whether a patient suffers from the disease or not. The pipeline of analysing the selected dataset was achieved by using a knowledge discovery process consisting of the following phases: exploratory data analysis (EDA), preprocessing techniques on the data, choosing an appropriate data mining approach to classify the instances and finally evaluate the performances that the ML classification algorithms reached together with a comparative evaluation of these techniques.

At first, concerning the EDA part of the analysis the data was explored in whole and tried to investigate hidden correlations. The most common approach was through plots illustrating behaviour of attributes themselves or even against each other both concerning continuous and discrete values. Interesting facts were revealed, like for example that it is more likely a woman to be infected with the disease, or that the habit of cigarette does not considerably affect the class of diagnosis. Another useful point was that attributes pleural albumin and pleural protein were highly correlated with each other which let us ignore one of them to make the model more efficient and converge faster.

In the next phase, concentration was given on processing the dataset in a way that prepares algorithms to successfully fit and later predict the target attribute. Although the dataset was clean in the first place, dummy encoding was performed producing a new dimension for each possible value of a categorical feature. This way, all attributes became either binary (0 or 1) or taking continuous values which enabled the models to produce more realistic results. Furthermore, even though outliers did not affect dramatically the performance of the majority of classification algorithms as proven, taking a legit threshold a small percentage of them was removed so that allow and simplify further methods like oversampling to be applied.

When it comes to classifying the instances, the mesothelioma disease dataset was split into 2 subdatasets from which the first of total 182 samples (56.1%) was used to train each algorithm and form its necessary shape and parameters. The second of total 122 examples (43.9%) was used to test the performance of the algorithm, that was already been shaped, onto new unseen data. From the

selected classification algorithms there were those that performed well without any preprocessing technique applied, and those that as the manipulation steps proceeded the evaluation metrics were steadily increasing and reaching almost exceptional levels. In the first group of algorithms belong decision tree classifier, logistic regression, Gaussian Naive Bayes, and linear SVC. Whereas, kNN and ANN using back propagation learning algorithm took several steps to enhance their performance. Such steps were hyperparameter optimisation techniques as well as feature extraction/selection methods. Constructing an implementation pipeline, the parameters as well as kernel functions were let to vary in order to find the best fit the algorithms could achieve.

Cross-validation, balanced accuracy score, confusion matrices, ROC/AUC and precision/recall curves were broadly employed to test the success of the models and come to safe deductions. Despite the simplicity of the models that reached excellent evaluation measures, they also managed to perform equally well on the resampled data with reduced dimensions. On the other hand, there is no room for someone not to admit that MLP with back propagation learning method is the algorithm which displayed the most interesting and accretive performance. And this is the reason why such a technique has the best potential to generalise good enough and the safest for a future realistic diagnosis.

6.2 Future Work

Obviously, the work undertaken in this thesis does not cover all the approaches one could follow to accommodate every possible concept or limitation of the subject. In addition, there is no doubt there are enough different adaptations, tests, and experiments that are left for the future to be examined. Some of them could concern deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity. Nevertheless, whatever a future method will be chosen to be, when dealing with analysing medical data and in particular with carcinoid tumours, the analyst should keep in mind the seriousness of faulty classifying an instance as this case could probably interpret a realistic classifying problem. In other words, computing the accuracy of a model alone does not necessarily hold all the demanding information needed to make safe deductions and conclude to a data driven decision. After exploring the area of the analysis, both from domain as well as from the ML perspective, a few proposals and concepts have come across that would potentially include some interest for fellow students, researchers or data analysts.

From the early stage of this work the small dataset was identified to be an issue. A first suggestion for future work is to try these methods and features on a larger dataset. With such a dataset, it would also be interesting to have completely separated data for testing, that is not even used in the k-fold cross validation technique. This might lead to a minor decrease of the values of evaluation metrics but on the other hand it also raises the adaptation to the real-world problem. The data gathering process could be fulfilled by getting in touch with medical centres or other organisations that are

capable of providing such amount and quality of data, unless more sophisticated approaches of producing reasonable data are considered.

Another plausible perspective which also applies to most of the ML problems is raising the domain knowledge of the analyst. It is not an exaggeration to say that domain knowledge on the subject can have equal value as the technical skills because apart from others, it empowers collaboration with domain experts and as a result better models and techniques are chosen. Having clarified the meaning and the physiology of all the features along with their values in a dataset together with the nature of the problem can have positive effects to the results. They enable analysts to take an active part in the choice of the model, as well as the feature selection and data cleaning phases of the analysis. Thus, when dealing with complex areas like medicine it is always preferable to get a deeper dive in the field so that to produce effective models with accurate and interpretable results.

Finally, the last proposition has to do with the so-called ensemble learning in ML. This technique is widely used in many prestigious ML competitions and it comprises a combination of several algorithms. Those models are gathered to enhance predictions while reducing the variance and the bias of the algorithm. There exist numerous fixed such mixtures and they are split into two categories depending on the generation of the base learners whether they are produced in parallel or sequential. However, both groups of methods aim to improve the stability and the accuracy of the algorithm.

Bibliography

[1] H. O. Ilhan, E. Celik. *The Mesothelioma Disease Diagnosis with Artificial Intelligence Methods*. IEE 10th International Conference on Application of Information and Communication Technologies (AICT) 2016.

[2] O. Er, A. C. Tanrikulu, A. Abakay, F. Temurtas. *An approach based on probabilistic neural network for diagnosis of Mesothelioma's disease*. Journal of *Computers and Electrical Engineering* Volume 38 Issue 1 2012 p75-81.

[3] D. Chicco, C. Rovelli. *Computational prediction of diagnosis and feature selection on mesothelioma patient health records*. PLOS ONE 14(1): e0208737. https://doi.org/10.1371/journal.pone.0208737 2019.

[4] M. Nilashi, M. Z. Roudbaraki, M. Farahmand. *A Predictive Method for Mesothelioma Disease Classification Using Naive Bayes Classifier*. Journal of *Soft Computing and Decision Support Systems EISSN 2289-8603* Volume 4, No6 2017.

[5] X. Hu, Z. Yu. *Diagnosis of mesothelioma with deep learning*. Oncology Letter 17(2) 1483-1490 10.3892/ol.2018.9761 2018.

[6] Andreas C. Mueller and Sarah Giudo. *Introduction to Machine Learning with Python (1^{st}ed. O'Reilly)* 2016. p9

[7] Aurelien Geron. *Hands-On Machine Learning with Scikit-Learn & TensorFlow (1^{st}ed. O'Reilly)* 2017. p16

[8] Mueller, Giudo, *Introduction to Machine Learning with Python*. p35

[9] C. J. Matheus, P. K. Chan, G. Piatetsky-Shapiro. *System for Knowledge Discovery in Databases*. Journal *IEEE Transactions on Knowledge and Data Engineering Volume 5 Issue 6* 1998 p903-913.

[10] F. Gullo. *From Patterns in Data to Knowledge Discovery: What Data Mining Can Do*. Physics Procedia 62: 18-22 10.1016/j.phpro.2015.02.005 2015.

[11] Salvador Garcia, Julian Luengo, Francisco Herrera. *Data Preprocessing in Data Mining (1^{st}ed. Springer)* 2015. p2

[12] Garcia, Luengo, Herrera. *Data Preprocessing in Data Mining*. p8-16

[13] Charu C. Aggarwal. *Data Classification: Algorithms and Applications (CRC Press Taylor & Francis Group)* 2014. p87-93

[14] S. A. Panimalar. *Data Mining Techniques in Medical Sector.* International Research Journal of Engineering and Technology (IRJET) Volume 5 Issue 4 2018.

[15] S. Shalev-Schwartz, S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms (Cambridge University Press)* 2014. p258-259

[16] K. P. Murphy, *Machine Learning: A Probabilistic Perspective (MIT Press)* 2012. p16-18

[17] Schwartz, David. *Understanding Machine Learning: From Theory to Algorithms.* p126-128

[18] Aggarwal. *Data Classification: Algorithms and Applications.* p69-71

[19] G. Bonaccorso. *Machine Learning Algorithms: Reference guide for popular algorithms for data science and machine learning (Packt Publishing)* 2017. p133

[20] T. Hastie, *The Element of Statistical Learning: Data Mining, Inference, and Prediction.* p417-421

[21] Schwartz. *Understanding Machine Learning: From Theory to Algorithms.* p268

[22] S. Marsland. *Machine Learning: An Algorithmic Perspective (*2^{nd}*ed. CRC Press)* 2015. p39-46, 71-78

[23] Mueller, Giudo, *Introduction to Machine Learning with Python*. p266

[24] T. Fawcett. *An introduction to ROC analysis.* Pattern Recognition Letters 27(8): 861-874 10.1016/j.patrec.2005.10.010 2006.

[25] T. Hastie, R. Tibshirani, J. Friedman. *The Element of Statistical Learning: Data Mining, Inference, and Prediction (*2^{nd}*ed. Springer)* 2001. p241

[26] J. Grus. *Data Science from Scratch: First Principles with Python.* *(*1^{st}*ed. O'Reilly)* 2015.

[27] W. McKinney. *Python for Data Analysis: Agile Tools for Real-World Data.* (*1^{st}*ed. O'Reilly)* 2012.

[28] G. Hackeling. *Mastering Machine Learning with scikit-learn: Apply effective learning algorithms to real-world problems using scikit-learn. (*1^{st}*ed. Packt Publishing)* 2014.

[29] J. VanderPlas. *Python Data Science Handbook: Essential Tools for Working with Data.* (*1^{st}*ed. O'Reilly)* 2016.

[30] A Vijayvargia. *Machine Learning with Python: An approach to Applied Machine Learning.* (*1^{st}*ed. BPB Publications)* 2018.

[31] H. Brink, J. W. Richards, M. Fetherolf. *Real-World Machine Learning.* (*2^{nd}*ed. Manning)* 2018.

[32] F. Kane. *Hands-On Data Science and Python Machine Learning: Perform data mining and machine learning efficiently using Python and Spark.* *(1sted. Packt Publishing)* 2017.

[33] M. Bowles. *Machine Learning in Python: Essential Techniques for Predictive Analysis. (John Wiley & Sons Inc.)* 2015.

[34] T. Hauck. *Scikit-learn Cookbook: Over 50 recipes to incorporate scikit-learn into every step of data science pipeline, from feature extraction to model building and model evaluation (1sted. Packt Publishing)* 2017.