POLITECNICO DI TORINO

Master's Degree in COMPUTER ENGINEERING

Master's Degree Thesis

Deep Learning for Assessing Risk of Prostate Cancer

Supervisors

Prof. SANTA DI CATALDO

Prof. MASSIMILIANO RUOCCO

Prof. FRANCESCO PONZIO

Candidate

ELISA TEDDE

DECEMBER 2022

Summary

Cancer detection is one of the leading research topics in medical science. Whether it is breast, lung, brain, or prostate cancer, progress is being made to improve the accuracy and timing of detection. Prostate cancer is the second most common cancer in men and the sixth leading cause of cancer death among men in the world. Many prostate cancers are indolent and do not result in cancer mortality, even without treatment. However, a significant percentage of prostate cancer patients have aggressive cancers that rapidly progress to metastatic disease and are often dangerous. The current diagnostic pathway is based on serum Prostate-Specific Antigen (PSA) levels. Although PSA screening reduces the spread and death from cancer, it overdiagnoses some low-risk cancers that may not have caused damage, leading to unnecessary invasive examinations, such as biopsies.
Several methods have been proposed in the past, such as studying the evolution of prostate antigen over time using different velocity formulas, which have often led to inconsistent results. Furthermore, the available datasets were small as they contained at most a few thousand patients.
In this thesis, some Deep Learning methods have been applied to time series to predict prostate cancer risk. The main objectives are early diagnosis and reduction of the number of unnecessary biopsies. The dataset I worked on is unique because it contains information on more than one million patients who underwent PSA testing in Norwegian clinics. Several approaches were proposed to deal with the irregularity of the time series, one of the most recurrent problems in clinical data. The first technique was based on regularizing the time series, while the second added new features, such as the time distance between visits. The results were compared using different metrics such as Specificity, Sensitivity and F1score and finally, the best model was selected. The performance obtained suggests that the proposed methods are promising and can be a helpful tool to support clinical decision-making.

Acknowledgements

First, I would like to thank my parents for always supporting me and making me realize how important constant effort is to achieve great results.
Thanks to my brother because even though he is five years younger than me, he always has a solution for everything.
I would like to express my deep and sincere gratitude to Professor Massimiliano Ruocco and his team for allowing me to do this research, but above all, for always supporting me.
Thanks to my university because the path was hard, I gave up many things to be able to spend an extra hour on books, but the satisfaction was unquenchable.
Thanks to Paolo because he has been so patient to put up with me and support me. Together we have faced many challenges and have always come out on top.
I thank myself because without these achievements, I would never have been able to participate in one of the most beautiful experiences of my life, the Erasmus in Norway. So thank you to Norway, to all the trips to its wonderful lands, to the endless hours of light in summer and darkness in winter and to apartment 4. Thanks to all of you guys, Giulia, Andrea, Francesco, Lukas, Marco, Sander, Leornardo, Michele, Fabio, Ola, Ottar, Ludovica, Miriam and Gabriele.

Table of Contents

List of Tables VII

List of Figures IX

Acronyms XIII

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Problem definition . . . 3
- 1.3 Contribution . . . 3

2 Background theory 6
- 2.1 Deep learning for time series classification . . . 6
  - 2.1.1 Definitions . . . 6
  - 2.1.2 Deep learning models . . . 6

3 State of the art 15
- 3.1 Irregular time series . . . 15
  - 3.1.1 Imputation and data generation models . . . 16
- 3.2 Classification for time series . . . 20
  - 3.2.1 Time series transformations approaches . . . 20
  - 3.2.2 Deep learning approaches . . . 21
  - 3.2.3 More scalable methods . . . 22
- 3.3 Application on PSA . . . 24
  - 3.3.1 PSA velocity techniques . . . 24
  - 3.3.2 Machine learning models . . . 25

4 Dataset and Data Exploration 28
- 4.1 Introduction . . . 28
- 4.2 Data exploration . . . 30
- 4.3 Type of problem . . . 35

5 Methods 37
5.1 Baseline models . . . 37
5.1.1 Preprocessing . . . 37
5.1.2 Models . . . 38
5.2 From Baselines to Deep Learning approaches . . . 39
5.2.1 Regularized time series . . . 40
5.2.2 Irregular time series . . . 43

6 Experimental setting 47
6.1 Hyperparameter tuning . . . 47
6.1.1 Hyperparameter optimization method . . . 48
6.2 Loss Function . . . 49
6.3 Evaluation metrics . . . 49
6.4 Services for sensitive data . . . 51
6.4.1 GPU for Deep Learning . . . 52
6.5 Libraries . . . 52
6.6 Configuration . . . 53

7 Results and Discussion 55
7.1 Baseline models . . . 55
7.2 Regularized time series . . . 59
7.3 Irregular time series with the addition of new features . . . 61
7.4 The best model . . . 64
7.5 The ability of AI to support medical decision . . . 66

8 Conclusion and Future works 69
8.1 Conclusion . . . 69
8.2 Further Work . . . 71

Bibliography 74

List of Tables

5.1 *Initial input: example of patient's time series.* 40
5.2 *Insertion of minimum and/or maximum in the time series.* 40
5.3 *The discretization technique is applied to the time series.* 41
5.4 *Zero imputation is applied to the time series.* 41
5.5 *Linear inteporpolation is applied to the time series.* 41
5.6 *Missing value indicators are added to the input.* 41
5.7 *Example of a patient who has undergone ten examinations for two years.* 42
5.8 *The time series after discretization and interpolation.* 42
5.9 *The four different types of input.* 42
5.10 *The resulting input.* 44
5.11 *Categorical features.* 45
5.12 *The four different types of input.* 45

7.1 *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. Acc, Sen, Spe are respectively the Accuracy, Sensitivity and Specificity.* . 56
7.2 *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. Acc, Sen, Spe are respectively the Accuracy, Sensitivity and Specificity.* . 59
7.3 *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. The following techniques are applied: interpolation and zero-filling.* . . . 61
7.4 *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics.* 62
7.5 *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics.* 62
7.6 *Bold represents the best performing configuration.* 64
7.7 64
7.8 *The performance according to the risk categories. Bold represents the best performing configuration.* 66

7.9 *The performance according to the time series length. Bold represents the best performing configuration.* 67
7.10 *The performance according to the age. Bold represents the best performing configuration.* 68

List of Figures

1.1 *Inferior view of the structures in the male reproductive system [2].* . 1
1.2 *A schematic representation of a deep-learning healthcare system is shown [10].* 4

2.1 *Multilayer perceptron for time series classification [17].* 8
2.2 *Fully Convolutional Neural Network architecture [17].* 9
2.3 *Left: CAM on class 1, right: CAM on class 2. The trends in each graph show the results of each CAM time series. The color indicates the time segment's contribution to the class (if predicted as class 1 or 2) [17].* 10
2.4 *(Left) A typical Recurrent Neural Network and (Right) an RNN unfolded in time [31].* 11
2.5 *Forget gate [37].* 12
2.6 *Input gate [37].* 12
2.7 *Cell state [37].* 13
2.8 *Output gate [37].* 14
2.9 *BiLSTM cells: the output layer receives information from past and future states simultaneously [40].* 14

3.1 *The figure illustrates a taxonomy of methods based on the five high-level modeling primitives including discretization, interpolation, recurrence, attention and structural invariance [43].* 17
3.2 *The figure illustrates the discretization approach [43].* 17
3.3 *(top left) zero-filling and no indicators, (bottom left) forward-filling and no indicators, (top right) indicators and zero-filling, (bottom right) indicators and forward-filling. Time flows from left to right [46].* 19
3.4 *The figure illustrates the Interpolation-Prediction Network [47].* . . . 20
3.5 *The diagram shows the statistical comparison of nine classifiers on the UCR/UEA univariate time series classification archive [17].* . . 22
3.6 *(Left) Accuracy and (Right) Training Time versus Training Dataset Size [12].* 23

3.7 *Statistics extracted from the dataset used by Nitta [68]* 26
3.8 *ROC curves for prediction of prostate cancer [68]* 26
3.9 *Flow chart illustrating the process to detect the presence of prostate cancer and its clinical significance [69].* 27

4.1 *PSA evolution* 29
4.2 *Risk categories distribution for patients with cancer.* 31
4.3 *The age distribution at first PSA test.* 32
4.4 *PSA distribution for patients without a biopsy or with negative biopsy.* 33
4.5 *PSA distribution for patients with cancer.* 33
4.6 *Time series length for patients without cancer.* 34
4.7 *Time series length for patients with cancer.* 34
4.8 Δt *between visits for patients without cancer.* 35
4.9 Δt *between visits for patients with cancer.* 35

5.1 *Example of time series for a patient with cancer: the yellow point at time* t_d *represents the biopsy.* 38
5.2 *Deep Neural Network* 39
5.3 *The decision table is used directly by doctors in Norway to classify patients with cancer.* 44
5.4 *Multi-Kernel Learning approach: example of a possible configuration.* 46

6.1 *Confusion matrix for binary classification problem.* 50
6.2 *ROC curve for binary classification problem. The diagonal shows the performance of a random classifier. Three classifiers (blue, orange, green) are shown [92]* 51
6.3 *Training, Validation and Test set.* 53

7.1 *Correlation matrix. The features are: psa, age and velocity.* 56
7.2 *The permutation of feature importance. Feature0, feature1 and feature2 are respectively age, PSA and velocity.* 57
7.3 *The correlation matrix for* ΔT *features. The features are: mean, median, standard deviation and quantile.* 58
7.4 *Input1: interpolation* 60
7.5 *Input2: interpolation + binary missing indicators* 60
7.6 *Input3: discretization* 60
7.7 *Input4: discretization + binary missing indicators* 60
7.8 *LSTM model* 63
7.9 *CNN model* 63
7.10 *MKL model* 63
7.11 *MLP model* 63
7.12 *The confusion matrix.* 65

7.13 *The ROC curve.* 65

8.1 *Highlighting with the class activation map of the contribution of time series region for the two classes when using the FCN and ResNet classifiers. Red corresponds to a high contribution and blue to almost no contribution to correct class identification [17].* 72

Acronyms

AI
artificial intelligence

DL
deep learning

ML
machine learning

TS
time series

TSC
time series classification

MTS
multivariate time series

UTS
univariate time series

NN
neural network

DNN
deep neural network

MLP
multi layer perceptron

CNN

convolutional neural network

LSTM

long short-term memory

GRU

gated recurrent unit

RNN

recurrent neural network

CCE

categorical cross entropy

PCA

prostate cancer

NaN

missing values

MKL

multi kernel learning

SOTA

state-of-the-art

Chapter 1

Introduction

1.1 Motivation

The prostate is a part of the male reproductive system. It is located on the pelvic floor and surrounds the bladder's neck and the urethra as shown in Figure 1.1. The prostate gland's main function is to store part of the seminal fluid and facilitate ejaculation during sexual activity [1]. One of the main constituents of prostate secretion is prostate-specific antigen (PSA), together with citrate (18.7 mg/ml), zinc (488 µg/ml), spermine (243 mg/ml) and cholesterol (78 mg/ml). PSA is a glycoprotein produced by the acinar cells of the prostate and it is specific for the prostate gland. The function of PSA is to dissolve the seminal clot after ejaculation to facilitate the transport of spermatozoa along the female reproductive tract [1].

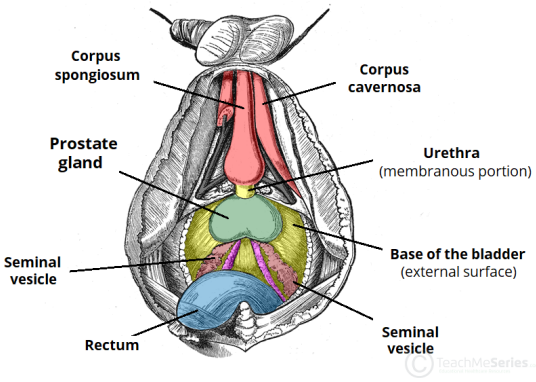


Figure 1.1: *Inferior view of the structures in the male reproductive system [2].*

Prostate cancer is one of the most frequent cancers in the world, with approximately 1.414.000 new cases and 375.304 deaths in 2020. The risk of the disease varies depending on several factors such as race, family medical records and diet [3].

However, the family medical record is still very important because if a relative has prostate cancer, it doubles his future chances of being affected. Prostate cancer incidence increases with age, and the median age of prostate cancer diagnosis in Norway is 74 years. Many men die with prostate cancer, and not from prostate cancer.
Prostate cancers are usually located in the back-part of the prostate, often slow-growing and symptoms may not appear for many years [1]. Compression of the urethra may appear, but frequently the tumor has reached an advanced stage before any symptoms. End-stage prostate cancer is often characterized by tumor spreading to bone marrow or other organs, resulting in pain and a reduced quality of life.
Although prostate cancer is responsible for many deaths every year, it does not have a high mortality rate, especially if it is detected early [4]. Therefore, early detection could reduce cancer mortality and ensure less complicated treatment. In contrast, cancers diagnosed at an advanced stage significantly impact quality of life and treatment costs are very high [1]. In addition, about one-third of prostate cancers are estimated to grow aggressively and benefit from early diagnosis. On the other hand, the others grow more slowly and, in many cases, do not affect a man's life [3].
Prostate cancer diagnostics is usually performed by the general practitioner as a combination of a digital rectal examination and a PSA measurement [5]. It may be part of a screening program, or outside a screening program and/or as a part of a visit for other conditions (sometimes named opportunistic screening). Results are usually reported as nanograms of PSA per milliliter (ng/mL) of blood plasma. The blood level of PSA is elevated in people with prostate cancer, but it does not necessarily mean that a man has the disease [3]. Therefore, it does not guarantee the presence of cancer, as even benign prostate enlargement can be responsible for an increased PSA [5]. The second screening technique is digital rectal exploration (DRE), which consists of inserting a finger into the rectum to directly sense abnormalities in the prostate [6]. A good screening test should be sensitive, safe, inexpensive, and used for diseases where early diagnosis improves prognosis. However, these tests are controversial among the scientific community; for example, PSA has a high sensitivity but poorly discriminates between low and high-risk tumors, and thus overdiagnosis and overtreatment. DRE has a low sensitivity, and depends on the experience of the investigator. Currently, a percentage of men are at risk of developing infections, such as loss of sexual function and blood in the urine [3].
Since screening is not accurate enough, these methods are often used as a procedure to decide whether or not to perform a biopsy. High PSA values require a biopsy, which consists of directly taking prostate samples with a needle. The process carries certain risks for the patient due to the procedure's invasiveness, such as

urosepsis [3]. Therefore, since the mid-1980s, biopsies have been taken using a radiological guidance such as ultrasound or MRI, to minimize side effects, and increase the chance of the needle hitting the tumor. However, they cannot correctly detect prostate cancer because they have poor resolution compared to biopsies. It is therefore essential to have additional strategies, such as further testing after screening and risk stratification, to determine whether it may be helpful to continue screening at an older age [7]. For example, Heijnsdijk et al. [8] demonstrated that stopping screening for men at the age of 60 with a PSA level < 1ng/ml significantly reduce the amount of screening compared to continuing to offer the test to all men with a similar number of detected prostate diseases.

1.2 Problem definition

These concerns have accentuated the need for doctors to rely on data experts and machine-learning algorithms. Indeed, the acceleration of Big Data and the exponential growth of computer power has changed the nature of medical care. Thus, there is a growing development of the application of ML in the medical sector, which can gather information from numerous sources and aid the decision-making process of highly skilled professionals [9]. Figure 1.2 shows in detail the process of interaction between data, machine learning and clinical decision-making. A machine learning model can thus learn the patterns of health trajectories of a large number of patients. This tool can help clinicians predict future events by drawing on information that goes far beyond the practical experience of the individual doctor [10]. However, the algorithm cannot replace the medical practitioner, but provides useful suggestions in scenarios where high-risk misdiagnoses are common (e.g. childbirth) or when doctors are uncertain. Early research on prostate cancer diagnosis involved statistical approaches, such as establishing formulas for PSA velocity [11]. However, statistics deals with simpler things and it would not be easy to make predictions when the context is more complex. The implementation of ML can therefore help to predict things more accurately and achieve better results. Some of these tasks have led to challenges and competitions to motivate research on these topics [10].

1.3 Contribution

In this context, the thesis is developed to reduce overtreatment and provide early diagnosis for high-risk patients using Deep Learning techniques. In the past, researchers have been unable to obtain satisfactory and reliable mathematical or machine-learning models because (1) the existing datasets were too small, from

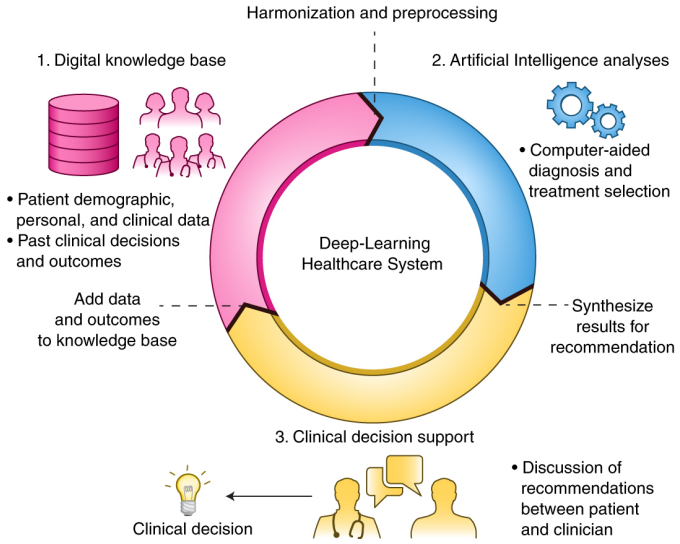


Figure 1.2: *A schematic representation of a deep-learning healthcare system is shown [10].*

hundreds to thousands of patients, and (2) they tried to extract velocity functions in a 'handcrafted' way that led to very inconsistent results.
In this thesis, the main task is classifying irregular time series for prostate cancer risk prediction. This constitutes a fundamental challenge for classical machine learning models mainly due to non-uniform intervals between observations.

Thus, the main contributions are:

- **The f-function**: the use of deep learning algorithms allows the automatic approximation of the best function representing the evolution of the PSA. This makes it possible to overcome the limitations of velocity formulas defined in the past.
- **Handling data irregularity**: irregular data are pervasive in clinical time series, where the time intervals between visits vary, leading to difficulties in modeling the entire time series. However, these irregular intervals may contain valuable hidden information. For example, short time intervals may imply more frequent examinations, indicating a worsening patient's condition. Deep learning can capture this effect; thus, two different approaches have been proposed. The first is the regularization of time series, while the second is the introduction of new features, such as delta time δT between two visits.
- **The unique PSA dataset**: the Norwegian health system has provided a real-world dataset containing information on more than one million men. ML

models can therefore obtain more promising results than in the past because the algorithm has a larger sample of patients.

- **Scalable algorithm**: neural networks are highly performing, but training times are often high because the number of hyper-parameters to be defined is large. Therefore, we compared the performance of Rocket, a simple and scalable model [12], with that of deep learning models. The significant advantage is that Rocket can obtain a prediction in just a few minutes.

In conclusion, this work is structured as follows: Chapter 2 explores the theory of time series and deep learning algorithms, while Chapter 3 analyses in detail the state-of-the-art of time series irregularity and classification. The dataset and related statistics are further explored in Chapter 4. Three approaches for cancer risk prediction are proposed and discussed in detail in Chapter 5. The parameters of the experiments, the libraries, and the development environment are described in Chapter 6. Finally, the results obtained and future work are discussed in Chapters 7 and 8.

Chapter 2

Background theory

2.1 Time series classification

The time series classification (TSC) problem has been studied in many real-world applications ranging from electronic health records [13] and human activity recognition [14] to acoustic scene classification [15] and cyber-security [16]. This section introduces the necessary background for this type of problem.

2.1.1 Definitions

Before presenting the different types of neural networks, we go through some formal definitions for TSC.

Definition 1. (TS) A time series X is a series of time-ordered values, X = $[x^{(1)}, x^{(2)}, ..., x^{(T)}]$, where $x^t \in \mathbb{R}^d$, T is the length of time series and d is the dimension of the feature vector that describes each point [17].
Definition 2. (UTS) A univariate time series X = $[x_1, x_2, ..., x_T]$ is an ordered set of real values. The length of X is equal to the number of T values [17].
Definition 3. (MTS) A M-dimensional time series, X = $[X^1, X^2, ..., X^M]$ consists of M different univariate time series with $X^i \in \mathbb{R}^T$ [17].
Definition 4. (TSC) A dataset D = $\{(X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N)\}$ is a collection of pairs (X_i, Y_i) where X_i could be a univariate or multivariate time series with Y_i as its corresponding label [17].

2.1.2 Deep learning models

A neural network is a graph of computing units, called *neurons*, structured in layers. Each neuron receives signals from other neurons and outputs a single value [18]. In

general, a neuron applies a nonlinearity function on a linear combination of input values, and the output of a specific neuron can be defined as:

$$output = f(\omega^\tau x + b) \tag{2.1}$$

where $x = (x_1, x_2, ..., x_d)$ is the d-dimensional input vector of the neuron, $\omega = (\omega_1, \omega_2, ..., \omega_d)$ the neuron weights, b the bias, $(*)^\tau$ the transpose operation, and f the non linear activation function. The number of neurons and the number of layers could be considered hyperparameters. By combining neurons with non-linearities, the neural networks have a great capacity to form complex functions and their architecture depends on how the combination is made. Note that the vector of weights in equation 2.1 must be learned automatically through an optimisation algorithm that minimises a cost function [19]. The cost function computes the distance between the current and expected output using the weights. The usual loss function is the multiclass categorical cross entropy (CCE), defined in the following equation:

$$L(X) = -\sum_{i=1}^{K} y_i \log(\hat{y}_i) \tag{2.2}$$

where L represents the cost when classifying the input time series X and $\hat{y}_i$ the probability of the input having the class y equal to class i out of K classes [19]. Thus, the average loss of the entire dataset is defined as follows:

$$J = \frac{\sum_{j=1}^{N} L(X_j)}{N} \tag{2.3}$$

where N represents the number of samples.
The loss function is then minimized to learn the weights in Ω (it denotes the set of weights to be learned by the network) using a gradient descent method which is defined as:

$$\omega = \omega - \alpha \frac{\partial J}{\partial \omega} | \forall \omega \in \Omega \tag{2.4}$$

where α is the learning rate of the optimization algorithm [17].

Although there exist many types of DNNs, in this section, we focus on three main neural network architectures used for the TSC task: multilayer perceptron, convolutional neural networks, and recurrent neural networks. These architectures were chosen because they are adopted for deep learning models for the classification of time series.

Multilayer perceptron

A multilayer perceptron (MLP) is one of the most traditional architectures for deep learning models. It consists of at least three layers (an input, an output layer, and one or more hidden layers) of nonlinearly-activating nodes. Since MLPs are fully connected, each node in one layer connects with a certain weight ω_{ij} to every node in the following layer [20]. In addition, most deep learning approaches employ a softmax layer corresponding to a FC layer with softmax as an activation function and a number of neurons equal to the number of classes in the dataset. The softmax function calculates the probability that the input time series belongs to the class j:

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} i = 1, ..., J \tag{2.5}$$

where all the x_i values are the elements of the input vector and J represents the number of classes.
In recent years, researchers have decided to use these neural networks for the analysis of time series as shown in Figure 2.1. However, the MLP is unsuitable for time series data because it has no spatial invariance: each date and time has its weight, and temporal information is lost [17]. Another obstacle of the MLP is that the length of the first layer of the network is fixed, so it is unsuitable for time series with inputs of different lengths.

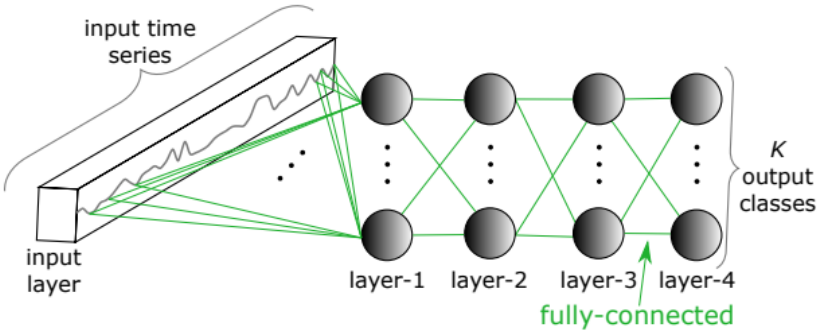


Figure 2.1: *Multilayer perceptron for time series classification [17].*

Convolutional Neural Networks

MLPs with too many layers can become impractical to train. By contrast, convolutional neural networks (CNN) can successfully capture an image's spatial and temporal dependencies by applying relevant filters [21]: they reduce the number of parameters to be learnt by using fewer connections of the hidden layer [22].

They can have five types of layers, which are:

- **The input layer**: it is the input of the entire CNN. For example, in an image-processing neural network, the pixel matrix represents the input.
- **The convolutional layer**: it contains a set of filters (or kernels) where the size of each filter is usually smaller than the image. Each filter convolves with the image through the convolution operation: it is a linear calculation of the scalar product between the weights and the region connected to the input volume. As the filter is applied multiple times to the input array, the result is a two-dimensional array of output values representing filtering of the input, called *feature map* [23].
- **The nonlinearity activation layer**: each value of the feature map passes through a nonlinearity function.
- **The pooling layer**: it simply performs downsampling along the spatial dimensionality of the given input, further summarizing the number of parameters detected in the input [23].
- **The fully-connected layer**: it produces class scores after the activation function.

In general, CNN outperforms other neural networks in various applications, such as image recognition problems [24], natural language processing [25], or the input of audio signals [26]. Researchers have also started to use them for time series analysis in recent years, motivated by the success of these architectures [17].

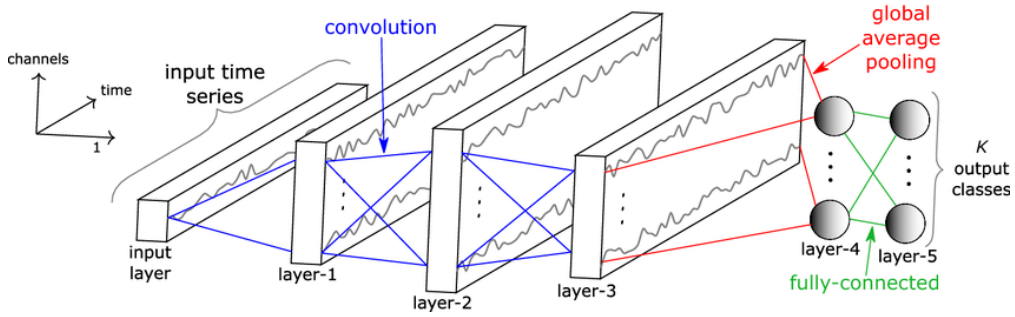
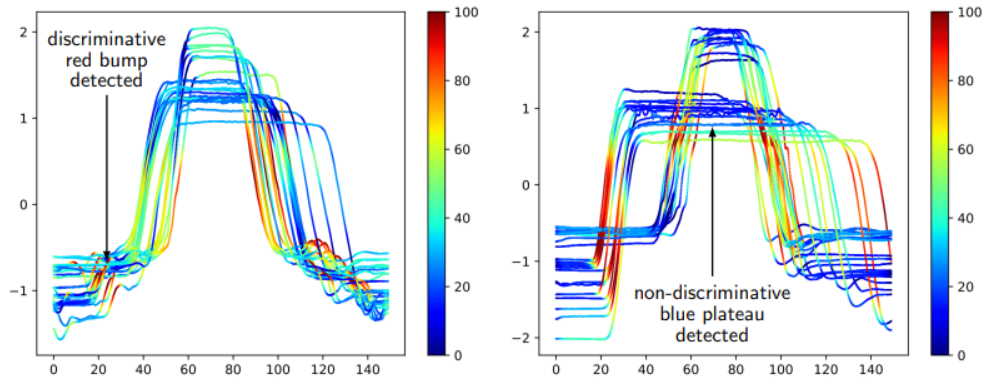


Figure 2.2: *Fully Convolutional Neural Network architecture [17].*

The convolution operation explained above is used slightly differently in time series tasks as shown in Figure 2.2: instead of extracting spatial information,

one-dimensional convolution is applied to learn patterns within the time window [17]. The result of a filter on a univariate time series X is another univariate time series; whereas applying several filters to the same time series X results in a multivariate time series whose size is equal to the number of filters used.

It is essential to clarify that the convolutional layer applied to a multivariate time series no longer has only one dimension (time) but also another equal to the number of input features. In addition, local pooling can be applied to reduce the time series length, e.g., the global pooling operation aggregates the time series over the entire time dimension resulting in a single value. The final layer could be fully connected, but it can also be replaced by global mean pooling, which allows the Class Activation Map (CAM) to be used to explain model decisions [27]. In other words, CAM makes it possible to highlight which parts of the time series input time series that contributed most to class prediction, producing 'visual explanations' of how a CNN model based its classification as shown in Figure 2.3 [17].



(a) FCN on GunPoint: Class-1 (b) FCN on GunPoint: Class-2

Figure 2.3: *Left: CAM on class 1, right: CAM on class 2. The trends in each graph show the results of each CAM time series. The color indicates the time segment's contribution to the class (if predicted as class 1 or 2) [17].*

Recurrent Neural Networks

Recurrent neural networks (RNNs) model sequential data of varying lengths, achieving state-of-the-art results for time series and data problems involving sequences such as natural language processing [28], image captioning [29] and handwriting recognition [30].

The RNN is a neural network that repeats itself over time because it maintains information about past knowledge using a particular type of ring architecture. These ring networks are called recurrent because they perform the same operations and calculations for each element of an input data sequence as shown in Figure

2.4. In addition, RNNs have the concept of 'memory' that helps them store the states or information of previous inputs to generate the next sequence output, thus capturing longer dependencies [31]. Some sequence models, such as Markov models, conditional random fields, and Kalman filters, can also handle sequential data but cannot learn long-range dependencies. In contrast, neural networks learn representations and can automatically discover unexpected structures.

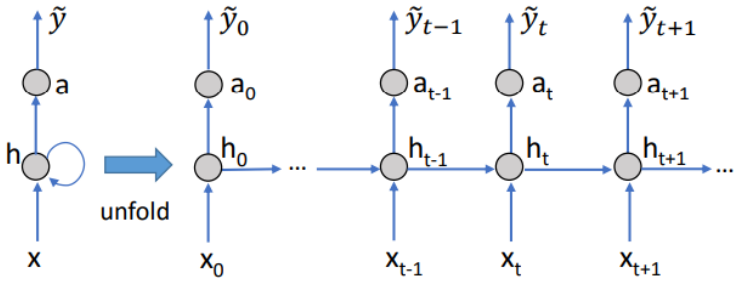


Figure 2.4: *(Left) A typical Recurrent Neural Network and (Right) an RNN unfolded in time [31].*

Like feedforward and convolutional neural networks, recurrent neural networks use training data to learn. However, many experiments have shown that gradient-based learning algorithms encounter difficulties when training RNNs: the gradient of some of the weights becomes too small or too large if the network is deployed for too many time steps [32]. This is the vanishing gradient problem: long-term dependencies in long input sequences lead to gradients disappearing or exploding. Many approaches have been proposed to solve this problem, such as lossy units, non-linear autoregressive models with eXogenous (NARX), and short-term memory (LSTM). However, LSTM has proven to be the most effective for handling long sequences [33].

LSTM Layer

An LSTM functions similarly to an RNN: it processes data by transmitting information as it propagates forward. The difference between these two neural networks is the operations within the cell [34]. These operations allow the LSTM to learn what is to be remembered, what is to be forgotten, and what is to be used immediately. The presence of four fundamental components enables this functionality [35]:

- **Forget gate**: the forgetting gate decides which information is to be thrown away or kept. Information from the current input $X(t)$ and the hidden state

$h(t-1)$ passes through the sigmoid function as shown in Figure 2.5. The sigmoid generates values between 0 and 1; the closer to 0 means forget, and the closer to 1 means keep. The cell will later use this value of $f(t)$ for point-by-point multiplication [36].

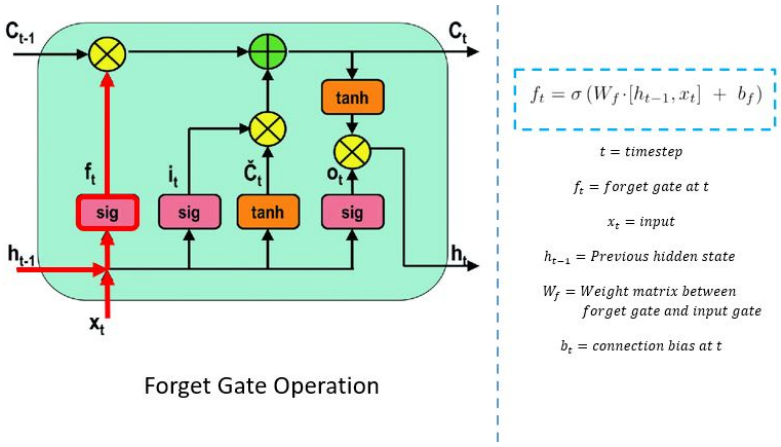


Figure 2.5: *Forget gate [37].*

- **Input gate**: the current state $X(t)$ and the previously hidden state $h(t-1)$ are passed into the second sigmoid function. That decides which values will be updated by transforming the values between 0 (unimportant) and 1 (important). Then the hidden state and current input pass into the tanh function as shown in Figure 2.6. To adjust the network, the tanh operator generates a vector $(C\ (t))$ with all possible values between -1 and 1. The output values generated by the activation functions are then used for point-by-point multiplication.

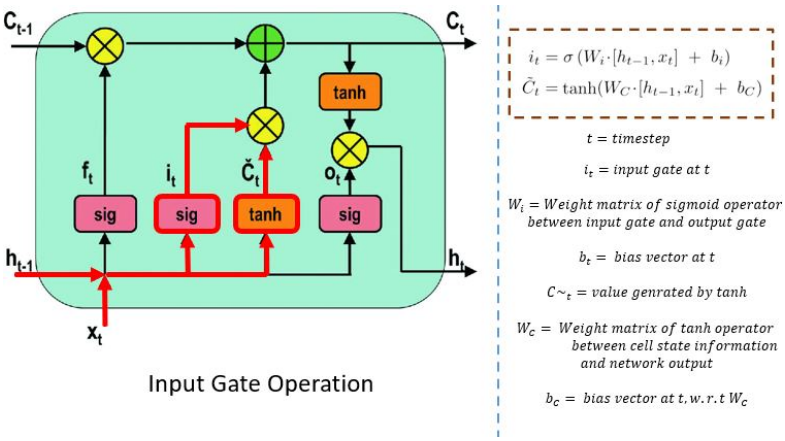


Figure 2.6: *Input gate [37].*

- **Cell state**: now the network has sufficient information from the forgetting and input gates. The next step is to decide and store the new state information in the cell state as shown in Figure 2.7. First, the cell state $C(t-1)$ gets pointwise multiplied by the forget vector $f(t)$: it has a possibility of dropping values in the cell state if it gets multiplied by values close to 0. Next, the network takes the output value of the input vector $i(t)$ and performs a point-by-point addition, which updates the cell state with the new values considered important by the neural network. That gives us the new cell state $C(t)$.

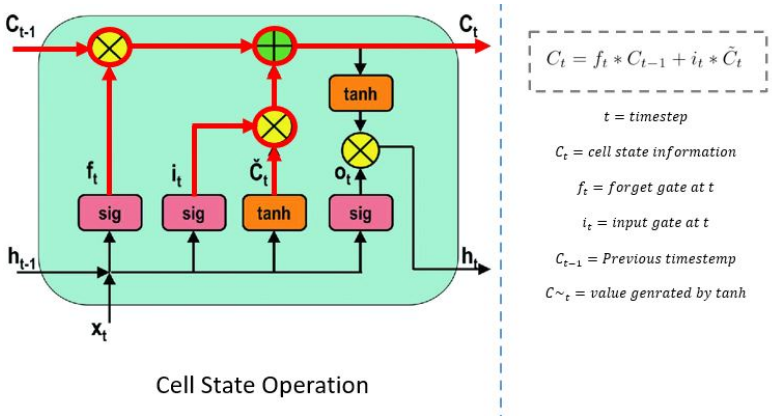


Figure 2.7: *Cell state [37].*

- **Output gate**: the output gate decides the next hidden state; this state contains information about the previous inputs. First, the values of the current state and the previous hidden state are passed into a sigmoid function. Then, the modified cell state passes into the tanh function. Finally, both outputs are multiplied point by point. The network decides which information the hidden state should contain based on the final value. The new cell state and the new hidden are then carried over to the next step as shown in Figure 2.8.

GRU Layer

GRU is the new generation of RNNs that implement a gating mechanism: the main differences between them are a smaller number of parameters and the lack of cell status [38]. Unlike LSTM, GRU has only two gates, an update gate and a reset gate. The first gate decides which information to throw away and which to add, and the second gate decides how much past information is to be forgotten. Training should be faster because there are fewer tensor operations. Generally, GRUs should work better with shorter sequences, whereas LSTMs with longer sequences due to the presence of the cell state.

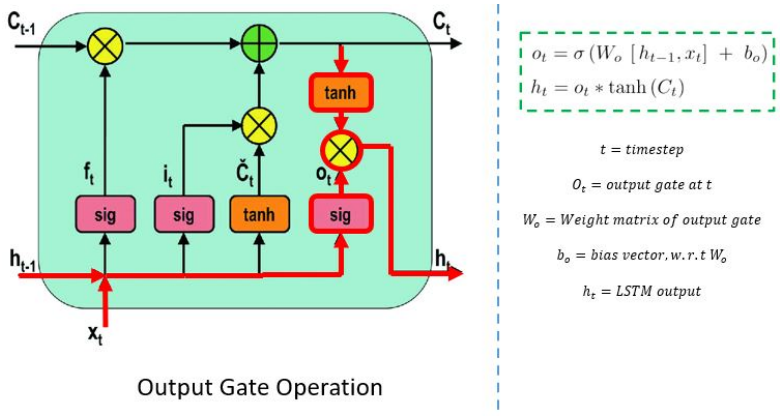


Figure 2.8: *Output gate [37].*

Bidirectional Layer

Unlike the standard LSTM, BiLSTM adds an LSTM layer that allows the input to flow in both directions, thus utilizing information from both sides as shown in Figure 2.9. In other words, this bidirectional architecture aims to divide neurons into two groups, one communicating positively and the other negatively [39]; it can produce a more meaningful output by combining LSTM layers from both directions.

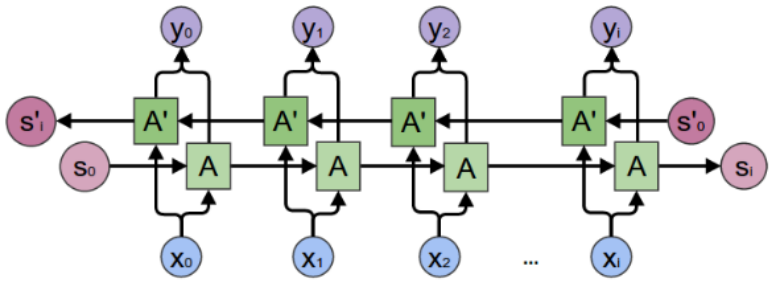


Figure 2.9: *BiLSTM cells: the output layer receives information from past and future states simultaneously [40].*

Chapter 3

State of the art

3.1 Irregular time series

Despite the obvious evolution of time series modelling from simple linear models to deep learning networks, most of these models are only concerned with regular time series data. Due to the many types of sensing devices or recording practices that generate the data, it is rare for raw data to be provided with all input variables sampled with constant timestamps [41]. Irregularly sampled time series are characterised by non-uniform time intervals between successive measurements and they are often characterised by sparsity, i.e. the percentage of missing data from a dataset. In time series there is a large difference between the degree of irregularity of data belonging to different domains. For example, data collected from intensive care units (ICUs) can often contain 80% missing data in the multivariate feature space [41].
Such data are present in many scientific and industrial domains and they are also an important feature of certain types of data in healthcare. For example, in clinical data, the health status of an individual patient can only be recorded at irregular intervals: vital signs and other measurements are recorded irregularly because they depend on the health status of the patient and the availability of clinical staff [42]. Irregular data and missing values strongly influence and limit the ability to analyse and model data for classification and prediction tasks. Often standard methods used to handle time series data introduce distortions and make strong assumptions about the underlying data generation process, which can lead to low-performing model predictions. Nowadays, irregularly sampled time series data represent a key challenge for leading machine learning models. The main challenges in modelling such data are [43]:

- **Irregular intervals**: the distance between two-time points varies by days, months, or years.

- **The different number of observations**: the total number of observations between dimensions can be different: for example, given two variables, A and B, there may be 3 measurements for variable A and 10 for variable B.

- **Lack of alignment**: multiple dimensions of a multivariate time series may be observed across different time points. The collection of observation times between dimensions may also differ for different data cases.

Thus, these features of irregularly sampled time series invalidate the assumption of a coherent fixed-dimensional feature space; most supervised and unsupervised learning models, including support vector machines, K nearest neighbors, logistic regression and others are based on this assumption [43]. It is therefore evident that classical machine learning methods are compromised by these characteristics and fail to model temporal irregularity.
However, in the last decade, clear progress has been made in developing models capable of handling irregular time series; they can produce predictive models close to those handling regular data. These methods belong into several categories, including approaches based on temporal discretization, similarity, interpolation, attention, recurrence, and structural invariance, as shown in Figure 3.1 [43]. Some of them can be applied to multiple tasks, while others cannot: it is therefore essential to understand to which type of problems a given approach can be applied. For simplicity, the mentioned approaches can be grouped into two macro areas [41]:

- The use of imputation strategies to produce a resulting time series without missing values.

- The development of a predictive model capable of handling irregular time series with minimal data preprocessing.

3.1.1 Imputation and data generation models

Many approaches have focused on a discretization and data generation phase to construct a regular time series. This method allows the series to be included in any regression or classification task.
Time discretization is the first step used to convert irregularly sampled time series data into regular time series, as shown in Figure 3.2.

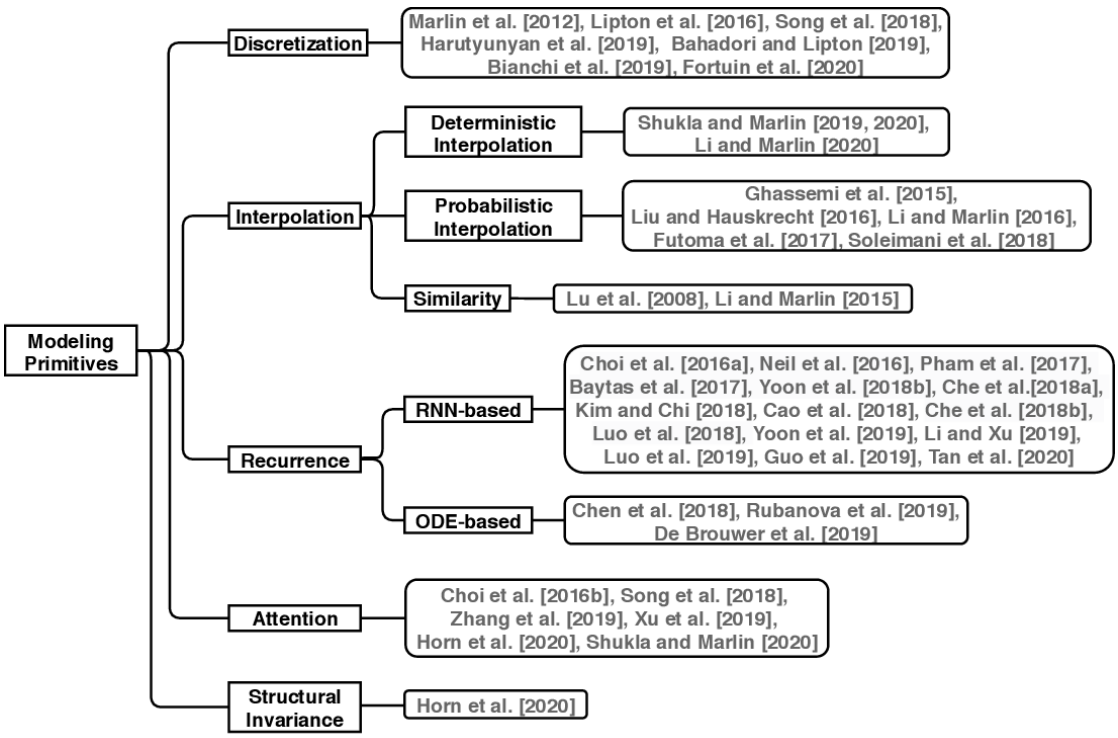


Figure 3.1: *The figure illustrates a taxonomy of methods based on the five high-level modeling primitives including discretization, interpolation, recurrence, attention and structural invariance [43].*

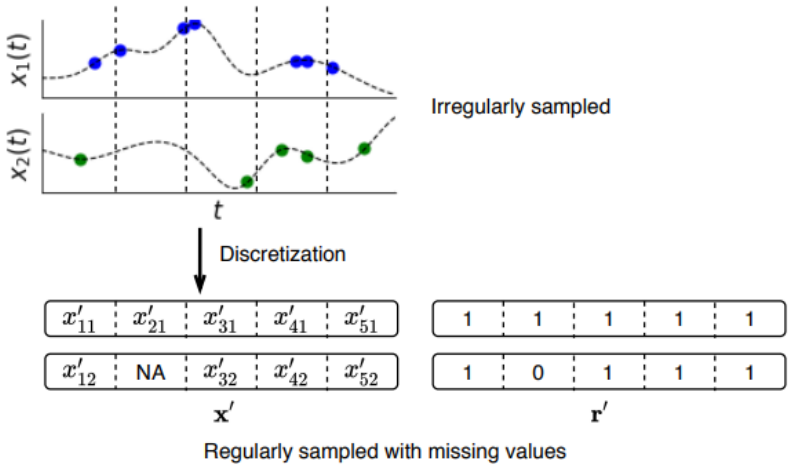


Figure 3.2: *The figure illustrates the discretization approach [43].*

The period represents the discretization window, and then time is discretized into fixed intervals. The value of a variable for each interval is the average of all measurements taken during the chosen interval. This is a standard procedure for dimensionality reduction, described as Piecewise Aggregate Approximation (PAA) by Keogh et al. [44]. The discretization method solves the three challenges mentioned above using fixed intervals, the same number of observations, and alignment between the different features. However, discretization introduces two further challenges: firstly, if the time series becomes too short due to a large discretization window, some essential information may be lost; thus, the sampling period is a hyperparameter chosen carefully [11]. Moreover, now regularly sampled time series may contain missing data: several recent approaches have addressed this problem using probabilistic models and imputation.

Probabilistic models

The probabilistic model is a possible approach to deal with missing values, particularly to mitigate the effects of temporal sparsity in electronic medical record data. Marlin et al. hypothesise that the missing data obtained after discretization are missing at random and apply probabilistic models to deal with missing data due to this assumption [45]. However, this condition is not a real property of the data but helps define the models that can be used. It is therefore essential to clarify that these values are usually not missing at random but reflect decisions made by clinicians: for example, if a doctor prescribes visits with a high frequency, he might suspect a disease. Given this condition, well-known clustering methods, such as K-means and hierarchical clustering, cannot be chosen because distance metrics cannot handle missing measures in data vectors. In contrast, probabilistic clustering models, also called Mixture Models, can efficiently handle missing data under certain conditions. These models use an empirical prior to mitigate the inherent sparsity of physiological data extracted from real data. The prior is constructed using a similarity kernel that encourages the average parameters of each cluster to be smooth with respect to time.
Researchers have shown that clusters can discover distinct physiological patterns with prognostic meaning [45].

Imputation

The basic imputation approaches are forward-filling and zero imputation. Missing values are replaced with zeros in the zero-imputation strategy as shown in Figure 3.3 - Top Left. By contrast, in the forward-filling strategy, x_i^t is imputed as follows (Figure 3.3 - Bottom Left):

- $x_i^t := x_i^{t'}$ if there is a previously recorded measure of the variable i at a time

$t' < t$.

- The median over all measurements in the dataset is calculated if no previously recorded measurement exists.

The RNN should learn to distinguish between imputed and real values. For example, the RNN could learn to recognize exact repeats with forward-filling. For the zero-filling technique, the RNN should understand that zero values correspond to missing values.
Lipton et al. apply the imputation strategies by adding a missing data indicator: this method consists of augmenting the inputs with binary variables, as shown in Figure 3.3 (Right). They include as input both the observed and imputed values of the time series and the values of the response indicator vector [46]. Recurrent neural networks should learn arbitrary functions of past observations and missing patterns using these indicators. However, given the same data, linear models can only learn strict substitution rules. For this reason, Lipton et al. add hand-engineered features derived from the indicator vector, such as the mean and the standard deviation of the indicator variables [46].

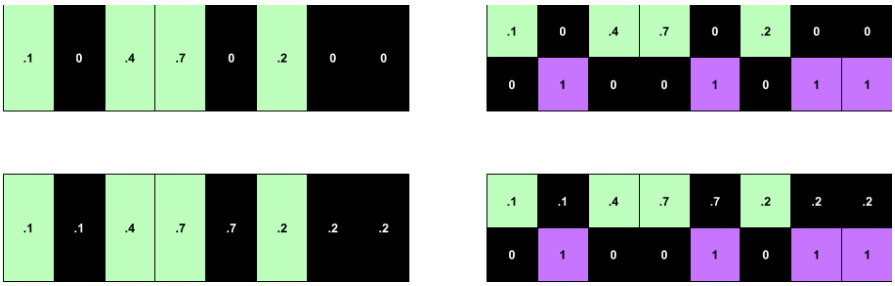


Figure 3.3: *(top left) zero-filling and no indicators, (bottom left) forward-filling and no indicators, (top right) indicators and zero-filling, (bottom right) indicators and forward-filling. Time flows from left to right [46].*

Shukla and Marlin proposed an alternative framework called Interpolation-Prediction Network (IPNet) [47]. The network is based on the use of several semi-parametric interpolation layers to produce multiple interpolants given as input to an irregularly sampled multivariate time series, as shown in Figure 3.4. The first layer of the network takes as input the dimensions of the time series and transforms them separately, creating several intermediate interpolants. The second interpolation layer merges the information on all dimensions at each time point, considering the correlations learned on the dimensions. Finally, the prediction network takes the output of the interpolation network as input and produces a prediction [47]. Any standard neural network, such as GRU, can be used in the prediction network and this approach can be applied to both regression and

classification problems. The parameters of the interpolation and prediction networks are learnt through an objective function consisting of supervised and unsupervised components.

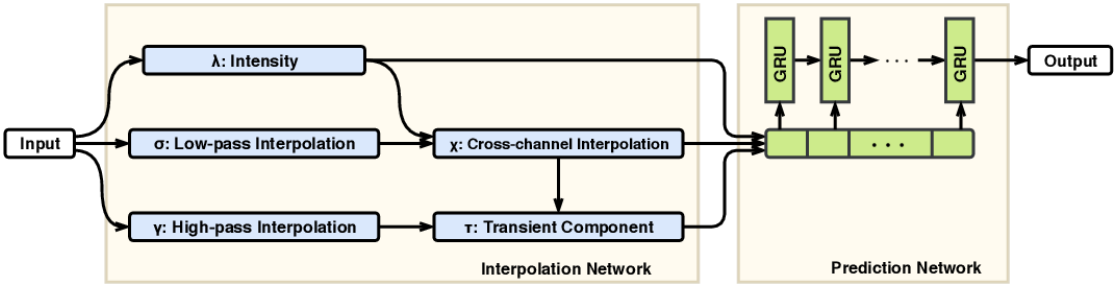


Figure 3.4: *The figure illustrates the Interpolation-Prediction Network [47].*

3.2 Classification for time series

The classification of time series (TSC) is an important task and has attracted great interest due to the large amount of data, from meteorological data, financial data, physiological signals to industrial observations. However, this is a challenging task due to the nature of the data: high dimensionality, large data size, and continuous updating [48].

3.2.1 Time series transformations approaches

One of the most traditional TSC approaches is the k-nearest neighbors classifier coupled with the dynamic time warping (DTW), that measures the similarity between two-time sequences [49]. On the other hand, some approaches rely on feature extraction to generate local or global patterns from time series. These methods transform the starting time series into a composition of transformed time series and then apply a conventional classification algorithm. It is important to note that the series can be transformed into other series, such as the Fourier transform, or into primitive values (mean, standard deviation, and more). However, the performance of these algorithms is highly dependent on the type of features, as the feature extraction mechanism is varied and complex. For example, TSBF is a framework that classifies time series based on a bag-of-features representation [50]. Sub-sequences of different lengths are randomly selected from the time series and then divided into shorter intervals to capture patterns. The extracted patterns are inserted into a feature vector, and each time series represents a bag. The vectors are then used as input to train a random forest classifier. On the other hand, BOSS is a dictionary-based classifier: a sliding window is applied to the series by

transforming the numerical series relative to that window into a word [51]. The classification is then based on the distribution of the extracted words.
Ensemble-based approaches instead combine several classifiers to achieve greater accuracy; the most common are Shapelet [52] and COTE [53]. For example, COTE combines 35 different classifiers into ensemble structures. HIVE-COTE has instead outperformed COTE using a new hierarchical structure with probabilistic voting [54]; this model includes two additional ensemble classifiers built into existing feature spaces and additional modules to represent two new transformation domains. To reach its accuracy, the model becomes computationally intensive and impractical to run on a large dataset; in some cases, the approach is unfeasible to train because it requires the training of 37 classifiers and the cross-validation of each hyperparameter.

3.2.2 Deep learning approaches

After describing the current state of the art of nondeep models, we examine the success of some deep learning approaches that have been adopted to solve classification tasks. Different approaches can be defined, but generally they can be divided into two subgroups: (1) deep learning models with feature engineering and (2) end-to-end deep learning models [17]. One of the most common feature extraction technique is inspired by computer vision problems: the transformation of time series into images [55]. However, unlike feature engineering, end-to-end learning aims to incorporate the feature learning approach while fine-tuning the classifier.
For instance, Zhicheng Cui adopted Multi-Scale Convolutional Neural Networks (MCNN) to classify univariate time series [56]. This approach led to state-of-the-art results, but the data preprocessing (e.g., downsampling and skip sampling) and the large number of selected hyperparameters make the model poorly scalable. Instead, Wang and his team tried to solve the problem of preprocessing and feature engineering by using directly the raw data [48]. The performance analyzed by the team using MLP, CNN, and ResNet have similar or better results than MCNN and COTE. Furthermore, the average pooling in the convolutional model allows the use of the class activation map (CAM) to identify the region that contributes most to the prediction. ResNet, on the other hand, is a state-of-the-art deep-structure neural network because each residual block has a shortcut connection that allows the gradient flow to pass directly through the hidden layers. Instead, Zhao et al. proposed a novel network, Time-CNN, for univariate and multivariate time series [57]. This is a CNN with a few differences: the use of the mean square error (MSE) instead of the standard cross-entropy categorical loss function, the use of a local average pooling operation instead of the local maximum, and the final layer is a FC layer with a sigmoid activation function; therefore, for Time-CNN, the sum of

the probabilities of the output classes is not guaranteed to be equal to one. They showed that this model has comparable classification performance to the networks described above.
In general, several new models have been developed in recent years, such as Time Le-Net [58], Multi Channel Deep Convolutional Neural Network (MCDCNN) [59], and Time Warping Invariant Echo State Network (TWIESN) [60]; the performance of all these networks has been compared using the UCR Time Series Archive as shown in Figure 3.5.

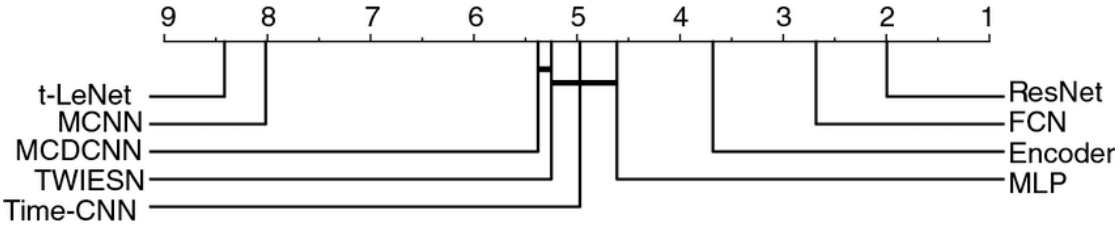


Figure 3.5: *The diagram shows the statistical comparison of nine classifiers on the UCR/UEA univariate time series classification archive [17].*

One of the possible limitations of neural networks is the risk of overfitting because it uses a large number of parameters; however, many generalization techniques, such as dropout, batch normalization, data augmentation and average pooling can significantly reduce the number of parameters [48]. Therefore, it is reasonable to conclude that deep learning models can perform well in time series classification tasks.

3.2.3 More scalable methods

The models mentioned above are often characterized by high computational complexity and therefore require significant training time despite the use of the GPU. Therefore, researchers have developed more scalable methods such as Proximity Forest [61], TS-CHIEF [62], and InceptionTime [63].
The success of convolutional neural networks in time series, such as ResNet and InceptionTime, demonstrates that convolutional kernels are a powerful tool to capture basic patterns or shapes. Additionally, the combination of multiple kernels can capture more complex patterns. The feature maps produced by convolution reflect the measure to which the pattern represented by the kernel is present in the time series. Dempster et al. developed a new algorithm, ROCKET (for RandOm Convolutional KErnel Transform), which achieves state-of-the-art performance [12]. Rocket uses convolutional kernels to transform time series and the obtained features represent the input for a linear classifier. In other words, the algorithm

transforms time series using a large number of random convolutional kernels, i.e. kernels with random dilation, length, bias, weights and padding. The obtained features are then used as input to train a linear classifier (Ridge Regression or Logistic Regression classifier). The four main elements that distinguish Rocket from classical Convolutional Networks are:

- Rocket uses many kernels because there is only a single "layer" of kernels, and kernel weights do not have to be learned. Hence, the computational cost of computing convolutions is low.
- Each kernel has length, padding, dilation, bias, and random weights.
- Rocket extracts two real values from each feature map: the maximum value (equivalent to the global maximum pooling) and the proportion of positive values.
- The only hyperparameter is the kernel number, k. The k value should be a trade-off between classification accuracy and computational time. However, even with an extremely large number of kernels (default 10,000), Rocket is fast.

To summarize, each kernel is applied to each input time series, producing a feature map. For example, if k = 10,000 kernels, Rocket produces 20,000 features for each time series.
The Figure 3.6 shows the Accuracy and the Time vs Dataset Size for Rocket, Proximity Forest, and TS-CHIEF. With 10,000 kernels, Rocket achieves similar classification accuracy as the other two algorithms. However, with 100 kernels, Rocket takes less than a minute to learn from more than 1 million time series while having lower accuracy. In addition, Rocket is an order of magnitude faster than Proximity Forest, even with a considerable number of kernels, 10,000.

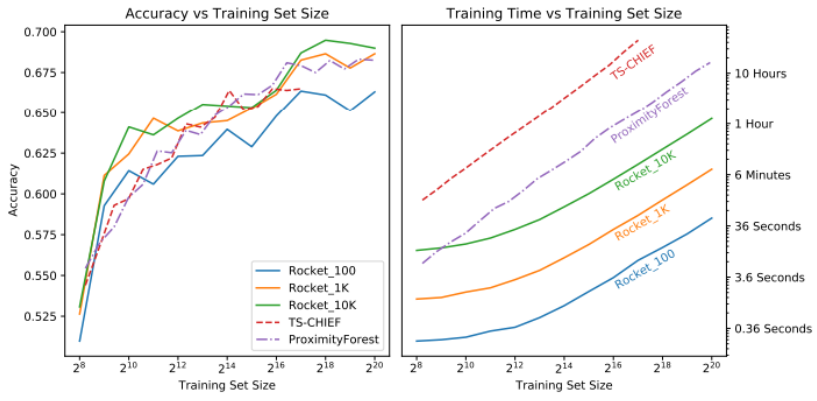


Figure 3.6: *(Left) Accuracy and (Right) Training Time versus Training Dataset Size [12].*

In conclusion, ROCKET is able to achieve approximately the same level of accuracy in a fraction of the time as competing SOTA algorithms.

3.3 Application on PSA

Since prostate cancer is a slowly evolving process, attention should be paid to the evolution of the PSA level over time rather than its absolute value [11]. The variation of PSA over time is represented by the concepts of PSA velocity and PSA doubling time, which are taken together as PSA kinetics. PSA velocity is expressed in ng/ml/year and can be considered a prediction: for example, a patient with a PSA of 2 ng/ml and a PSA velocity of 1.0 ng/ml/year should have a PSA of 3.0 ng/ml after one year. Instead, the PSA doubling time is the number of months it takes for the PSA to increase by a factor of two [11].

3.3.1 PSA velocity techniques

It is not uncommon for researchers to look at 5 or 10 distinct PSA kinetics criteria in a research study [64] [65]. For example, in [66] the authors found no fewer than 22 alternative PSA velocity and doubling time definitions. The method frequently employed in clinical practice simply subtracts the initial value from the final value ignoring the middle values, then divides by time. Consider for example two patients with annual PSA values of 1, 1.9, 2.0, and 2.1 vs. 1, 1.3, 1.7, and 2.1. This would result in a PSA velocity for both patients of 0.37 ng/ml/year, which does not appear to represent their divergent PSA histories. A good alternative would be to conduct a line of best fit, also known as Ordinary Least Squares Regression. As a result, the PSA velocities for the above example are 0.34 and 0.37 ng/ml/year, which seems to reflect a clinical intuition that the second man is more likely to have a higher PSA value in a year. However, remember that a regression requires a demanding mathematical calculation, making it difficult to perform in daily practice.
It should be clear that it is possible to create a variety of definitions of PSA kinetics, each with a different method of calculation and eligible PSAs (how many, over what period of time, with what minimum period of time between measurements) [11]. It would seem useful to have a criterion for the period of time over which PSA is measured, excluding PSA measures taken many years ago. One possibility is that PSAs measured closely together create statistical "noise", and as a result, a specific minimum interval between PSA measurements is necessary. Consider a patient with just two PSA values—1.8 and 2.1—taken six months apart. Claiming the patient's PSA velocity is 0.6 ng/ml/year could be illogical. Therefore, another essential condition might be that at least three PSAs are required over a minimum total length of time.
Over the last decade, several 'hand-crafted' methods have been developed taking

these parameters into account: for example, the MSKCC approach calculates a regression slope including all PSA values; the Thompson definition log transforms PSA before calculating the slope and it only includes PSAs within 3 years [67]; the Sengupta method is based on raw PSAs within 2 years, but also specifies that PSA measures have to be at least 3 months apart; the Smith approach applies log transformation and involves the specification that at least three PSAs have to be taken at least 4 months apart [66]. However, O'Brien emphasized how PSA kinetics values can change significantly between methods: for example, one patient has PSA velocities of 0.27, 0.76, 1.47, 2.64 and 32.0 ng/ml/year for each of five different calculation techniques [66].
Furthermore, Vickers and Brewster show another drawback of these approaches: PSA kinetics play no role in the diagnosis of prostate cancer in men before treatment because the proposed techniques are better at understanding prognosis in advanced or relapsed prostate cancer [11]. Consider for example a situation where doctors have a PSA level of an untreated man: should they make a decision based on the absolute PSA value or consider past PSAs and then extract a velocity? Vickers' team determined the added value of PSA velocity using data from the Malmo cohort. First, they found that PSA predicts the long-term risk of prostate cancer with a concordance index of 0.771. They then created a new model that also included PSA velocity. Although PSA velocity is a statistically significant predictor of prostate cancer risk, it did not add anything to the predictivity of PSA; indeed, the concordance index of the second model was identical [11].
It is therefore clear that these techniques cannot bring the doctor a unique and satisfactory conclusion because, on the one hand, they are not effective for untreated men and, on the other hand, the different calculation methods lead to different results.

3.3.2 Machine learning models

To overcome the previous obstacles, constructing an adequate model using several predictive variables is a promising approach. To this end, machine learning techniques have been widely used in clinical medicine, especially for constructing predictive models. Nitta and the team decided to use as input not only PSA-based parameters but also data on continuous changes in the PSA level over the past 2 years [68]. Thus, the set of input variables includes patients' age, PSA level (maximum, minimum, median, mean, and level of variance), prostate volumes, and white blood cell count in urinalysis, while the biopsy result is used for prostate cancer prediction. It is challenging to have all this information for a large number of people, so the final sample size is relatively small and unbalanced as shown in the Table 3.7.

Characteristics	Patients diagnosed with PCA N = 193	Patients with negative biopsy N = 319	*P* value
Age (year)			0.52
50–59	7	30	
60-69	82	149	
≧70	104	140	
Mean PSA level (ng/ml)[a)]	8.6	4.5	0.08
PSA (ng/ml)[a)]			0.67
<4	2	9	
4 < PSA<10	149	260	
10 ≦ PSA<20	38	47	
≧20	4	3	
Mean prostate volume (cc)	55.6	44.8	<0.05
Mean PSA density (ng/ml/cc)	0.20	0.19	0.87
Mean PSA velocity (ng/ml/year)	0.96	0.71	0.08

PCA, prostate cancer; PSA, prostate-specific antigen.
[a)] Average of two serial PSA testing

Figure 3.7: *Statistics extracted from the dataset used by Nitta [68]*

According to 3.7, there are some statistics on PSA values, and it is clear that patients with cancer have about twice as high a PSA value as healthy patients. Three different types of supervised machine learning algorithms (artificial neural networks, random forest, and support vector machine) were compared with the PSA-based parameters methods: PSA level (cutoff of 4 ng/ml), PSA density (cutoff of 0.20 ng/ml/cc) and PSA velocity (cutoff of 0.75 ng/ml/year) [68]. As shown in Figure 3.8, machine learning models could predict a diagnosis of PCA with significantly better AUCs than PSA density, PSA velocity and PSA level; thus, these algorithms, despite being more complex, lead to a marked improvement in performance.

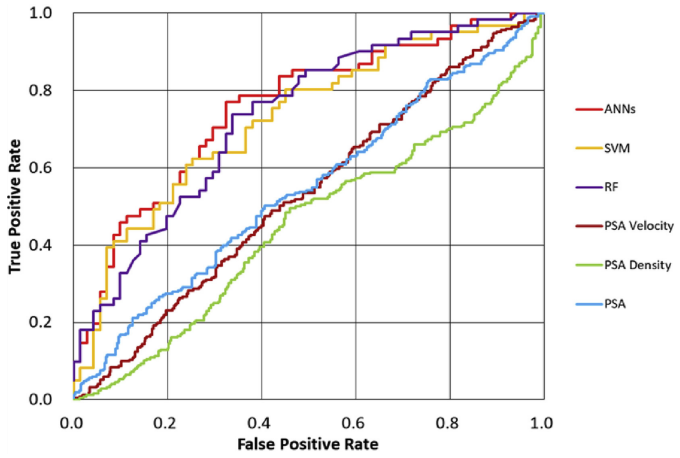


Figure 3.8: *ROC curves for prediction of prostate cancer [68]*

Instead, Cosma and his team decided to use statistical models with a feature set that combines the phenotypic profile of peripheral blood with PSA levels [69]. They showed the value of computational intelligence-based approaches

for interrogating immunophenotyping datasets, improving accuracy compared to the model using only the PSA value. They were the first to successfully devise a combinatorial feature selection method to identify a unique phenotypic profile of peripheral blood immune cells, consisting of five phenotypic features of T and B cells [70]. The extracted features were: CD8+CD45RA-CD27-CD28-, CD4+CD45RA-CD27-CD28-, CD4+CD45RA+CD27-CD28-, CD3-CD19+ (B cells) and CD3+CD56+CD8+CD4+ (NKT cells). Furthermore, they showed that there are no strong positive or negative correlations between the different features that will be used to build an interpretable machine-learning model.
This study demonstrates that machine learning approaches based on peripheral blood phenotyping profiles can distinguish between benign prostate disease and prostate cancer; it can also predict the clinical risk in asymptomatic men with elevated PSA levels using two biLSTMs. The LSTM model is more complex than the techniques described by Nitta [68] because it learns long-term bidirectional dependencies in the sequence data. A biLSTM model takes as input immunophenotypic characteristics and clinical data and it is trained to detect the presence of cancer. The second model takes as input a set of biomarkers, including immunophenotypic features and clinical data (PSA level and age), and it is trained to predict the clinical risk when cancer has been identified. The Figure 3.9 shows how prediction models to detect the presence of prostate cancer and the associated clinical risk can be used to aid clinical diagnosis, especially when dealing with high-risk cancers.

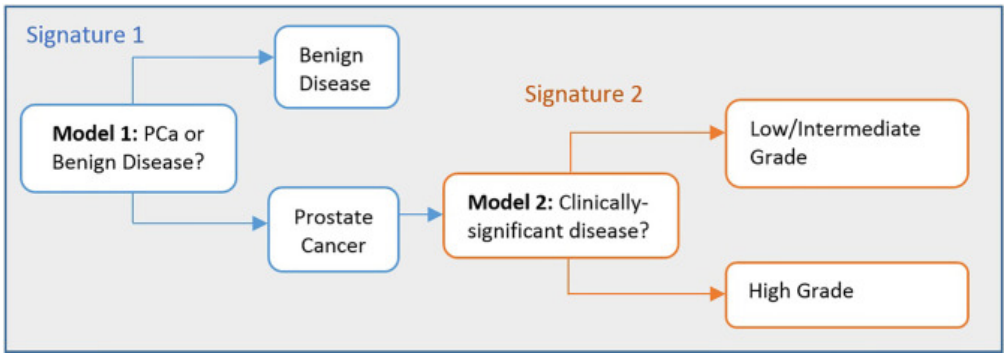


Figure 3.9: *Flow chart illustrating the process to detect the presence of prostate cancer and its clinical significance [69].*

The experimental results described above found that age is a feature that provides greater predictive accuracy when combined with immunophenotypic features than when used alone (86.79% vs. 66% on the test set).
In conclusion, machine learning techniques overcome previous drawbacks by also adding new features such as immunophenotypic or clinical data. As a result, they are promising models to predict the prostate cancer risk and to reduce overdiagnosis.

Chapter 4

Dataset and Data Exploration

4.1 Introduction

The Norwegian Prostate Cancer Consortium (NPCC) has collected an extensive database that includes Cancer Registry data, Population Registry records and causes-of-death, together with PSA test results obtained from Norwegian laboratory information systems from 1990 to 2022. We have been granted access to anonymized data on the tsd security platform, run by the University of Oslo. The main purpose of the NPCC is to use data records to make suggestions to improve the diagnostics and treatment of prostate cancer.
THE DATASET. The data set has restricted access and contains approximately 8,785,147 laboratory results on 1,306,003 unique patients, of which 105317 have a prostate cancer diagnosis.
It is a relational database and we have mainly used three tables:

- psa results: the numerical PSA value, the day of the visit and the anonymized project patient identifier.
- cancer registry: it contains data from the Cancer Registry, complete for all men diagnosed with prostate cancer. The features used here are the anonymized project patient identifier, the date of diagnosis and NPCC prostate cancer classification (risk factors).
- ss numbers: it contains information about the the anonymized project patient identifier and date of birth (anonymized to the 15th of the month).

The Cancer Registry does only have rights to store information from subjects having cancer, thus a positive biopsy indicating cancer is included in data, but a

negative biopsy is not stored. Thus, men not in the cancer registry did not undergo biopsy or had a negative outcome. There are several reasons why patients did not undergo biopsy:

- The doctor does not consider it necessary.
- The patient is waiting for the biopsy result; therefore, the table has not yet been updated.
- The man does not want to do a biopsy.
- Due to age and/or comorbid diseases, the risks of a biopsy outweighs the benefits.

More generally, with over 1 million patients, patients with undetected cancer should be a minority.

Figure 4.1 shows some examples of how the data for patients with cancer can appear. On the x-axis, there is the time and on the vertical axis, the PSA value. In each time series, after n visits, the outcome of the biopsy is shown (the colored dot in each time series). Therefore, it is essential to define the degree of cancer to understand how to treat the clinical case. The cancer grade is calculated by statisticians using various parameters such as the Gleason score, PSA value, and cTNM (clinical examination of cancer).

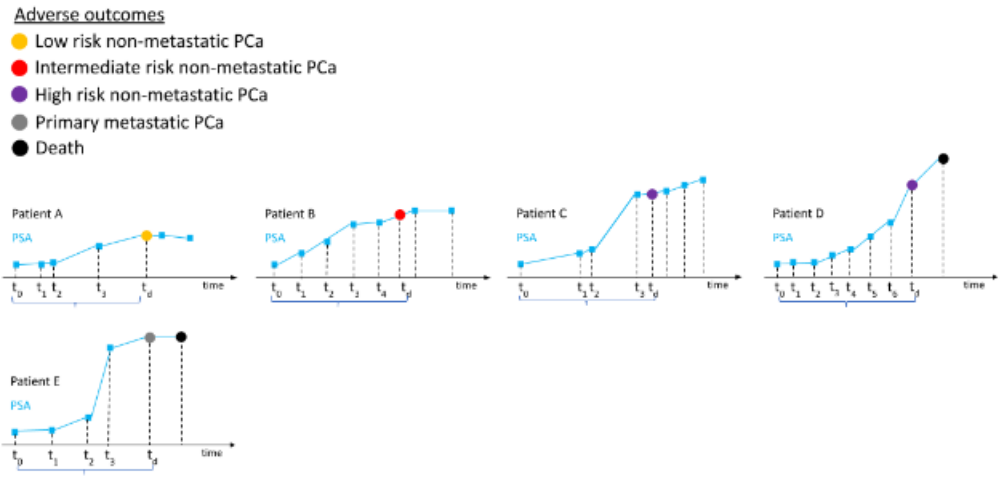


Figure 4.1: *PSA evolution*

The main risk categories are:

- **Low risk**: the cancer grows very slowly or not at all. Therefore, a therapeutic approach known as 'active surveillance' can be considered as a viable alternative to radiotherapy or surgery. In this case, the cancer is monitored regularly and treated with radiotherapy or surgery only if it grows [71].
- **Intermediate risk**: it consists of a heterogeneous patient group, so it is impossible to provide uniform treatment recommendations for men in this group: treatment options may be active surveillance, partial gland ablation, radical prostatectomy, and others [72].
- **High risk**: it is the potential to progress to a lethal phenotype that can be fatal, in marked contrast to low-risk cancer deemed suitable for active surveillance [73].
- **Metastatic risk**: in some cases, cancer cells spread to nearby lymph nodes, often the first destination for cancer spread. Men with metastatic prostate cancer often do not undergo treatment such as surgery or radiotherapy but start hormone therapy [72].
- **Missing risk**: it was not possible to define the risk category due to the lack of certain parameters.

In addition, all data after the biopsy were removed because it is difficult to classify the cancer after that date: in some cases, the cancer has been treated and therefore the risk may decrease, while in others, different decisions have been made, but there is no updated risk category.

4.2 Data exploration

In order to be able to extract the most important features from the dataset and subsequently create a prediction model, it is essential to have a good understanding of the data:

- **Data size**: the amount of data is important in deciding which machine learning model can be efficient.
- **Types of data**: it is essential to know the type of data you have because most machine learning models receive numerical data as input, so in the case of categorical variables (race, gender, age group, and level of education), data preprocessing is required.
- **Missing data**: they are handled by evaluating the most appropriate technique (mean, median, elimination, interpolation, and others).

- **Handling of outliers**: anomalous observations that distort the distribution of data must be dealt with by removing or changing their value.

Therefore, the data will be analyzed in statistical terms using count, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum.
Before starting an in-depth analysis, it is essential to emphasize that for each patient with cancer, only one time series is associated with a risk category. If a patient has undergone multiple biopsies, the outcomes of the biopsies following the first one are not recorded. It is also important to remember that all data in cancer patients do not consider time series points after biopsy (for the reasons explained above). Instead, for the remaining men, the entire time series is considered.
First of all, it is interesting to understand the distribution of the time series by risk categories because it is expected that they are pretty unbalanced. However, as the Graph 4.2 shows, more than 50% of people with cancer fall into the high and intermediate risk categories; this is an unexpected result as I supposed the category with the most patients to be 'low risk.' As the doctors pointed out, it is also good to realize that it is essential to recognize medium and high-risk patients because immediate action must be taken. On the other hand, low-risk patients need to be continuously monitored, but they are not a concern for clinicians. On the other hand, patients with metastatic risk represent a small percentage, which is good because they are the men most at risk.

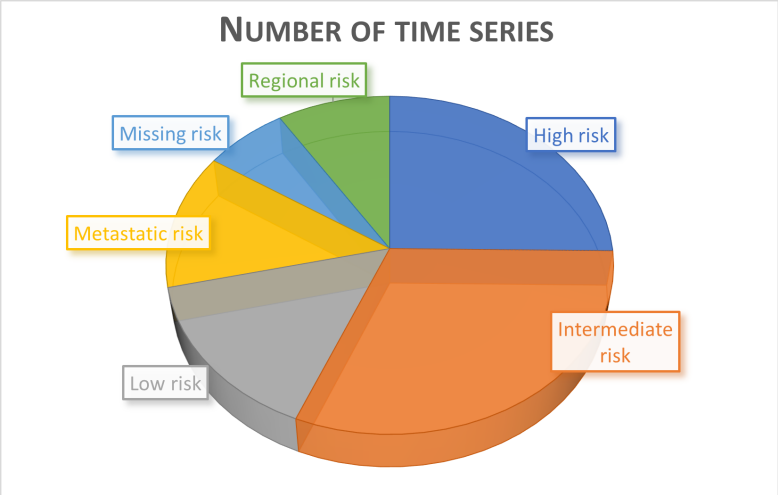


Figure 4.2: *Risk categories distribution for patients with cancer.*

Another interesting aspect is understanding the age distribution at the time of the first PSA test. The health institutes recommend starting PSA testing from 50 unless there is a family history where the man is considered to be at risk. It is also

recommended to have one test per year to monitor the PSA value, as the risk of prostate cancer increases with age. The Graph 4.3 represents the number of patients who underwent the PSA test for the first time divided by age group. It can be seen that most men started undergoing the test between the ages of 50 and 60, but a large number of patients started at a younger age. In the latter case, it is likely that the patient had a family history of prostate cancer or had suspicious symptoms.

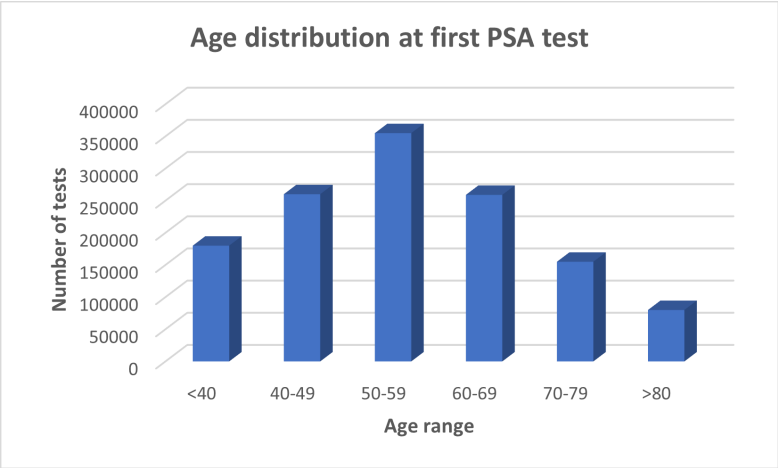


Figure 4.3: *The age distribution at first PSA test.*

Furthermore, to build a good model, it is necessary to understand the permissible PSA values and how the distribution of values varies between patients with cancer and patients without a biopsy or with a negative biopsy. Since the PSA value is the first alarm bell for detecting prostate cancer, one would expect a significant difference in PSA distributions. In fact, the two Graphs 4.4 and 4.5 show clear differences: in patients without a biopsy or with a negative biopsy most PSA values are below 4; there are also some very high values (>10), and in these cases, it may be that the test result is wrong, that the patient has inflammation or that the result is unexplained. On the other hand, the distribution is completely different in cancer patients because most men have a PSA value between 4 and 10. This range is the parameter taken into account by doctors in order to proceed with further examinations. It should also be noted that the number of examinations with a PSA < 4 is high, and this is because some cancer patients started undergoing examinations when cancer had probably not yet manifested itself.

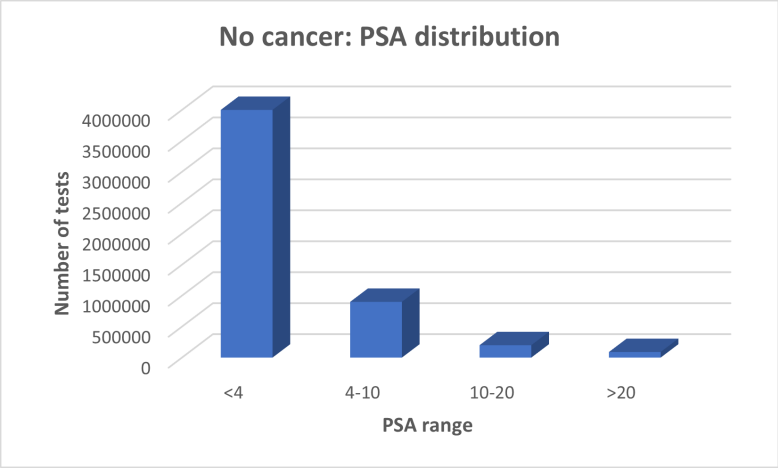


Figure 4.4: *PSA distribution for patients without a biopsy or with negative biopsy.*

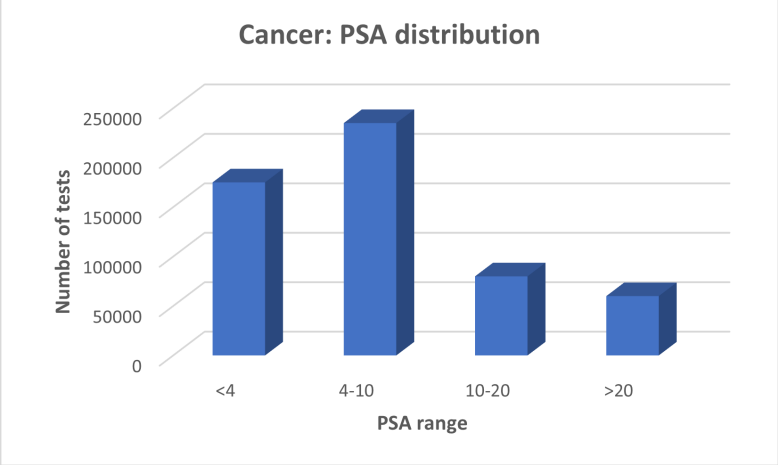


Figure 4.5: *PSA distribution for patients with cancer.*

Another crucial aspect is to understand the length of the time series. If each patient underwent only one or two visits, it would be challenging to consider the input as a time sequence. However, as the Tables 4.6 and 4.7 show, the average length for patients without cancer is 5, while for cancer patients, it is 7. These statistics make sense because patients diagnosed with cancer have more visits since the doctor suspected a problem. The maximum numbers of visits are 153 and 76, respectively; these values are excessively high, but it could mean that some men start having PSA tests at a young age and finish at an old age.

No cancer: time series length (number of points)	
count	1200686
mean	5.37
std	6.11
min%	1.0
25%	1.0
50%	3.0
75%	7.0
max	153

Figure 4.6: *Time series length for patients without cancer.*

Cancer: time series length (number of points)	
count	82827
mean	6.89
std	6.47
min%	1.0
25%	2.0
50%	5.0
75%	9.0
max	76.0

Figure 4.7: *Time series length for patients with cancer.*

Finally, as mentioned in the previous Chapters, one of the main objectives of this thesis is handling time series irregularity. The aim is to assess how much time δt can elapse between two successive visits. As shown in the Tables 4.8 and 4.9, the statistics show that δt is extremely variable. It can therefore be days or many years. This factor cannot be overlooked because if the doctor decides to prescribe frequent visits, it means that the clinician must constantly monitor the situation. For cancer patients, the average frequency is about one year, while for other men, it is almost two years; this result is consistent because if PSA values are normal, it is unnecessary to undergo new tests too frequently.

No cancer: ΔT (months)	
mean	17
std	21.8
min	0
25%	5
50%	11
75%	20
max	356

Figure 4.8: *Δt between visits for patients without cancer.*

Cancer: ΔT (months)	
mean	11.88
std	18.7
min	0
25%	3
50%	6
75%	13
max	321

Figure 4.9: *Δt between visits for patients with cancer.*

4.3 Type of problem

After thoroughly analyzing and understanding the data, the following considerations were made.

FEATURES. The best set of features must be selected to build a good machine-learning algorithm, and this is not always an easy task. Indeed, adding redundant variables compromises the model's generalization ability and final accuracy. Therefore, the features considered most important for the study of PSA evolution are:

- AGE: age of the patient at the time of the visit.
- DATE: date when the patient did the visit.
- PSA value: numeric value of the PSA.
- RISK FACTOR: type of cancer risk.

DATA CLEANING. As a first step, patients older than 100 years were dropped as they are unlikely to undergo biopsy, being dangerous for elderly patients. Then patients aged between 30 and 100 years were selected. Next, time series points considered "duplicate" were eliminated: a doctor can prescribe two PSA visits within a few days because the clinician is unsure of the first result. So, if two visits were less than nine days apart in the time series, the least recent one was eliminated. Finally, time series with a minimum number of points equal to 5 were considered: a machine learning model needs a sufficient number of points to learn distributions from the input data, and, based on statistics, the selected minimum length may be a good choice.

SUPERVISED LEARNING. Supervised learning uses a set to train models to produce the expected output. The training data includes the inputs and targets, which allow the model to learn over time. Finally, the algorithm estimates its accuracy through the loss function, adjusting until the error has been well minimized. Since this is the first time the dataset has been analyzed with machine learning techniques, a binary classification problem was chosen to test whether a model can easily identify people with cancer. Therefore, the target label 0 means patient without a biopsy or with a negative biopsy, while 1 means patient with cancer.

BALANCING TECHNIQUE. The dataset has an uneven distribution of observations because about 90% of patients did not undergo a biopsy or had negative biopsy results. Therefore, one possible way to proceed is to balance the data: there are several techniques, such as resampling (Oversampling and Undersampling). In this case, the best method is undersampling, where rows are randomly removed from the majority class to match the minority one. However, this technique was applied only to the training set, while the test set must be able to represent the actual clinical situation.

Chapter 5

Methods

5.1 Baseline models

As explained in Chapter 3, in the past, researchers have tried to extract velocity formulas using the PSA value or they have tried to combine other features such as age [68]. Currently, doctors decide whether to prescribe a biopsy based on certain thresholds: e.g., if the patient is young (∼40 years old) and has a very high PSA (»4 ng/ml), a biopsy is highly recommended; on the other hand, if the patient is old and he has a high PSA, the doctor has to weigh up the risks and benefits of the biopsy; thus, the way clinicians make a decision can be compared to a simple decision tree model. It is, therefore, clear that machine learning can be used to create more elaborate decision tree models by varying the parameters of the architecture, e.g., the maximum height of the tree. Thus, the first approach used is extracting and creating certain features and using specific machine learning models that come close to the way doctors reason.

5.1.1 Preprocessing

As a first step, we tried to understand the performance by using only the PSA value and the associated age as input and then adding another information: the velocity. For the first type of input, we decided to use the penultimate value in the time of PSA and age because the last value would have been too indicative (it is important to remember that the values taken into account are prior to the biopsy, if there is). Instead, for the second input, the velocity formula is extracted using the penultimate value over time and its previous. The Figure 5.1 shows in details a time series for a patient with cancer: age and PSA values are extracted from time t_2 and the velocity formula is calculated as follows:

$$velocity = \frac{PSA_2 - PSA_1}{t_2 - t_1} \tag{5.1}$$

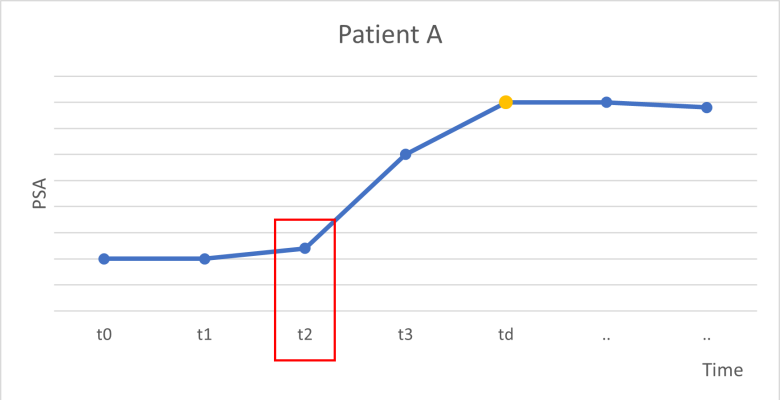


Figure 5.1: *Example of time series for a patient with cancer: the yellow point at time t_d represents the biopsy.*

Next, we tried to understand how much the temporal distance between two successive visits have informative power: it is essential to emphasize that in contrast to the probabilistic model proposed by Marlin [45], missing values cannot be said to be randomly missing. Thus, the aim is to create a model where the input consists only of features relating to the time between two consecutive visits; the following ΔT statistics were extracted from the dataset: mean, median, mode, and quantile. The objective is to understand whether using only temporal features as input is sufficient to achieve a non-random prediction; in that case, it means that the time between two visits can be used as a valid indicator of the patient's state of health: ΔT is, therefore, a feature to be taken into account.

5.1.2 Models

As mentioned earlier, we decided to use simple models to better approximate the medical reasoning, so the algorithms chosen are:

- **Decision Tree** is a supervised learning method used for classification. The goal is to create a model that predicts the value of a target variable by learning decision rules inferred from the dataset [74].

- **Random Forest** is a meta-estimator that fits certain decision trees on various subsamples of the dataset and uses the average to improve predictive accuracy and manage overfitting [75].

- **Ada Boost** is a distributed gradient boosting library designed to be efficient and portable [76].

It should be noted that Support Vector Machine is considered a good model for this type of task, but the computational cost is too high, so it was decided not to use it.

5.2 From Baselines to Deep Learning approaches

Now we want to create more sophisticated models as the results obtained in the past were not particularly satisfactory: on the one hand, researchers did not have sufficiently large datasets and, on the other hand, they tried to extract velocity formulas in a 'handcrafted' way that gave very different results. Thus, by using a deep learning model, it is possible to automatically derive an f-function that better approximates the evolution of PSA over time. As shown in the Figure 5.2, the idea is to have as input a set of raw features and as output the prediction; in the middle, there is a black box, i.e., a set of layers with a given number of neurons that, through the data learns the best weights to derive the PSA evolution formula. In this scenario, doctors may be more sceptical because it is hard to control what happens inside a black box and to understand why a prediction is made.

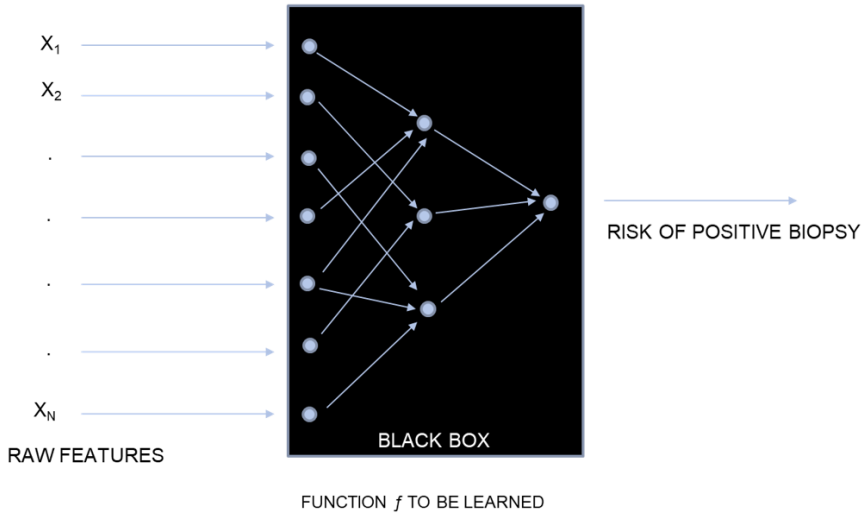


Figure 5.2: *Deep Neural Network*

Since it is crucial to deal with the problem of time series irregularity, there are two possible approaches:

- regularization of the time series.
- irregular time series with the addition of certain features.

5.2.1 Regularized time series

Time series regularization is one of the most frequently used approaches when there are time series with many missing values [42][47]. Therefore, it was decided to use the techniques of zero imputation and interpolation combined with missing value indicators proposed by Lipton [46].

Preprocessing

In the preprocessing phase, PSA values and corresponding age were extracted for each patient as shown in the Table 5.1.

AGE	30	32	32	33	34	59	70
PSA	5	6	7	7.1	7.2	21	24

Table 5.1: *Initial input: example of patient's time series.*

The input was processed according to the following steps:

- **Equal length** The time series are irregular both in the number of points and in the distance between points; thus, the objective is to have time series with the same length and a PSA value every few months/years.
  In order to proceed, a common minimum and maximum point is set for all patients: the selected patients are aged between 30 and 100 years, so the minimum and maximum points will be 30 and 100, respectively for each man. As shown in 5.2, the minimum age is already present while the maximum must be added with the corresponding PSA value (it is a missing value).

AGE	30	32	32	33	34	59	70	**90**
PSA	5	6	7	7.1	7.2	21	24	**NaN**

Table 5.2: *Insertion of minimum and/or maximum in the time series.*

- **Discretization** The period represents the discretization window and then age is discretized into fixed intervals. The value of a variable for each interval is the average of all measurements taken during the chosen interval. For example, the Table 5.3 shows the new input after this operation with a ten-year period. The period is an essential hyperparameter: if the time series becomes too short due to too large a discretization window, some essential information may be lost; on the other hand, if the time series becomes too long, there are too many missing values.

AGE	30	40	50	60	70	80	90
PSA	6.5	**NaN**	21	**NaN**	24	**NaN**	**NaN**

Table 5.3: *The discretization technique is applied to the time series.*

- **Handling of missing values** There are multiple possibilities for dealing with missing values; two different techniques were chosen in this context: zero-imputation and interpolation. Zero imputation means replacing the missing value with 0, but in this case, zero is an admissible value, so the value -1 will be used, as shown in the Table 5.4.

AGE	30	40	50	60	70	80	90
PSA	6.5	**-1**	21	**-1**	24	**-1**	**-1**

Table 5.4: *Zero imputation is applied to the time series.*

Instead, the Pandas library's interpolate() technique allows NaN values to be replaced by an interpolation method; in this context, the linear method has been chosen. It is also essential to select the direction of interpolation between 'forward,' 'backward,' or 'both': in this case, 'both' seems to be the most appropriate in order to be able to consider both forward and backward values as shown in the Table 5.5.

AGE	30	40	50	60	70	80	90
PSA	6.5	**13.75**	21	**22.5**	24	**24**	**24**

Table 5.5: *Linear inteporpolation is applied to the time series.*

- **Missing value indicators** Now a binary vector can be added to the resulting input: for each point, it indicates whether the value is real or has been handled with the missing value technique. The example 5.6 shows the PSA values with the zero-filling technique and the corresponding binary vector where 0 means 'real 'value and 1 'generated' value.

PSA	6.5	**-1**	21	**-1**	24	**-1**	**-1**
VECTOR	0	**1**	0	**1**	0	**1**	**1**

Table 5.6: *Missing value indicators are added to the input.*

This method proposed by Lipton [46] can further help the neural network to distinguish real from generated values.

In conclusion, the time series regularization technique can help to handle the problem of missing values when there are not too many values to be generated. However, there is a non-negligible limitation: let us consider, for example, a patient who has undergone ten examinations for two years as shown in the Table 5.7.

AGE	41	41	41	41	42	42	42	42	42	42
PSA	5	5	5	6	6	6.2	7	6.9	7	8

Table 5.7: *Example of a patient who has undergone ten examinations for two years.*

Therefore, if we consider the discretization window to be ten years, after discretization, we will have only one point (corresponding to the average of the 10 points) and the remaining ones are missing. Then the series has a constant trend after applying interpolation, as shown in the Table 5.8.

AGE	30	40	50	60	70	80	90
PSA	**6.2**	6.2	**6.2**	**6.2**	**6.2**	**6.2**	**6.2**

Table 5.8: *The time series after discretization and interpolation.*

This case is a limiting example, but it shows how some time series after preprocessing can begin with a constant trend because the patient started to undergo visits at a late age. On the other side, it is also possible that the time series ends with a constant trend in the case that the man stopped undergoing visits at a young age. So, the neural network can make false assumptions by learning from time series with constant trends.
Therefore, four different inputs (shown in the Table 5.9) are analyzed to compare them and see whether it is better to use the zero-filling technique rather than interpolation and whether the missing value indicators methods can improve the performance.

Input	Type of preprocessing
Input 1	Zero-Filling
Input 2	Interpolation
Input 3	Zero-Filling + Binary Vector
Input 4	Interpolation + Binary Vector

Table 5.9: *The four different types of input.*

Models

One of the most advanced models for classifying time series is the Recurrent Neural Networks. The RNN cell adds long-term memory because it maintains the memory of inputs; as a result, the RNNs provide good classification performance and allow the user to efficiently assess the degree of reliability of classification results. Three different RNN types are analyzed:

- vanilla RNN [33]
- long short-term memory (LSTM) [34]
- gated recurrent units (GRU) [77]

5.2.2 Irregular time series

In this approach, it was decided to avoid the regularization of time series and to switch directly to the use of irregular series, but with some additional features. As mentioned in the article [11], using the PSA value alone as input may sometimes not be sufficient; therefore, some researchers used other characteristics such as family history, age, ethnicity, or blood phenotypes. For example, age has been found to be strongly correlated with PSA evolution, which is considered one of the most important features to use with the PSA value. Moreover, doctors believe that it is not the absolute value of PSA that needs to be assessed directly but how it varies over time. For this reason, researchers have always tried to derive some velocity formulas; unfortunately, they have extracted very different velocity formulas, which have often led to inconsistent results.
In this context, we want to include information on how PSA varies over time; the formula that best describes the evolution of PSA will be learned directly from the neural network. This point represents a significant step forward compared to 'hand-crafted' velocity formulas.

Preprocessing

Let us now consider the features that will be considered for this approach, starting with an example of a time series (for simplicity's sake, in this case, the date consists of year and month only)

DATE	01-2000	05-2001	01-2002	12-2003
AGE	30	31	32	33
PSA	5	6	7	7.1

The resulting input will consist of the following features:

- The PSA value
- Age
- ΔT: the difference calculated in months between the date and the previous date.
- ΔPSA: the difference between the PSA measured at visit n and the PSA at visit n-1.

The processed input is shown in the Table 5.10 and the first values at time t_0 are removed as they contain null fields.

time	t_0	t_1	t_2	t_3
age	**30**	31	32	33
PSA	**5**	6	7	7.1
δT	**NaN**	16	8	23
δPSA	**NaN**	1	1	0.1

Table 5.10: *The resulting input.*

In addition, the doctors have defined PSA ranges with the corresponding risk category to define the patient's medical history as shown in the Figure 5.3.

Risk category	PSA value
Low risk	<10
Intermediate risk	10-20
High risk	20-50
Regional risk	50-100
Metastatic risk	>100

Figure 5.3: *The decision table is used directly by doctors in Norway to classify patients with cancer.*

However, it is crucial to emphasize that a higher-than-expected PSA for people of different ages does not mean the same thing: in the case of a young person, it is necessary to do thorough checks, whereas, in the case of an older man, it may be expected as the PSA increases with age. So risk categories must be associated with age categories. These types of information can be added to the initial input: the continuous values of age and PSA are thus transformed into categorical values as shown in the Table 5.11.

Age range	PSA range
30-40 years	< 4 ng/ml
40-50 years	4-10 ng/ml
50-60 years	>10 ng/ml
60-70 years	-
70-80 years	-
80-100 years	-

Table 5.11: *Categorical features.*

The neural network should then have extra support in its classification task thanks to some knowledge injection.
Therefore, four different inputs were analysed to compare them and to see whether it is better to use categorical features or not as shown in the Figure 5.12.

Input	Feature extraction
Input 1	PSA + age
Input 2	PSA + age + δt + δPSA
Input 3	Categorical(PSA, age)
Input 4	Categorical(PSA, age) + δt + δPSA

Table 5.12: *The four different types of input.*

Models

There are more extracted features in this approach than in the time series regularization method. Furthermore, in the case of input with categorical age and PSA, these are transformed using the One-Hot Encoding technique, so the input size increases considerably. For these reasons, it is not only essential to capture the long-term dependencies using, e.g., RNNs but also other neural networks such as MLP and CNN can be applied. For example, CNN can capture temporal dependencies, while MLP loses the temporal information because the input is not considered as an ordered sequence. The other models are Rocket (explained in Chapter 3), which is a very scalable algorithm for classifying time series, and the Multi-Kernel Learning approach (MKL). The MKL technique can increase the discrimination power and improve the performance of classifiers [78]. The idea is to optimally combine kernel matrices computed from the feature types with some kernel functions as shown in Figure 5.4. Therefore, it is expected to perform better than CNN.

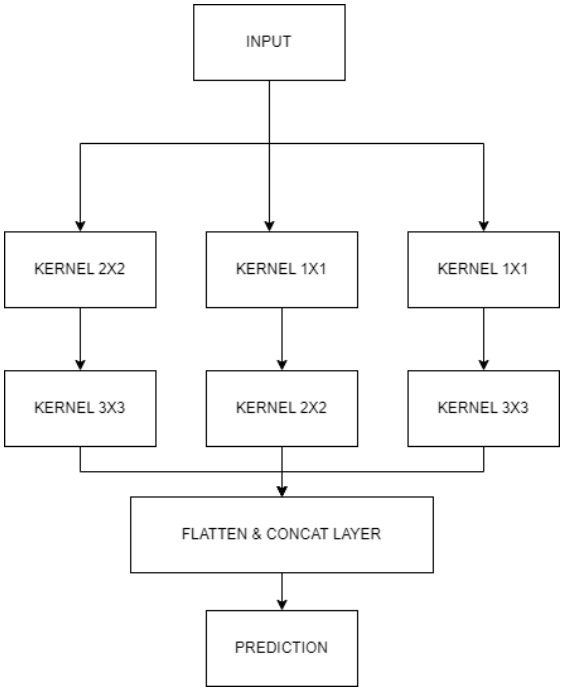


Figure 5.4: *Multi-Kernel Learning approach: example of a possible configuration.*

Chapter 6

Experimental setting

6.1 Hyperparameter tuning

In machine learning, there is a distinction between model parameters, configuration variables internal to the model and whose values can be estimated from the data provided, and hyperparameters, which are external to the model because the model cannot change its values during learning/training [79]. In other words, they are used by the algorithm during learning, but they are not part of the resulting model. The choice of hyper-parameters is crucial for obtaining a good model, but choosing the best combination is not a trivial procedure. The process is called hyperparameters tuning and in this thesis, it has been performed on different input and models (LSTM, CNN, and more). The most critical hyperparameters in a deep learning model are:

- **Batch size**: the training set is divided into several subsets, called batches, to improve the speed of the learning process [79].
- **Learning rate**: it controls how much the model must change in response to the estimated error each time the model weights are updated. Selecting the optimized learning rate is challenging because if the learning rate is too low, it could slow down the training process. In contrast, if the learning rate is too high, the model may not be optimal [80].
- **Number of Layers**: a neural network consists of vertically arranged components called layers; the layers are the input, output, and hidden layers that can range from 1 to n [79].
- **Number of Hidden Neurons**: the number of neurons required for the hidden layers depends on the amount and complexity of the data, the amount of outliers and the type of activation functions selected. In general, the number

of hidden neurons can be between the size of the input layer and the output layer [79].

- **Activation function**: it decides whether a neuron should be activated: the activation function decides whether or not the neuron's input is important in the prediction process using simple mathematical operations [81].

- **Dropout**: dropout ignores some randomly selected neurons during the training phase; therefore, these units are not considered during a particular step forward or backward. This technique is often used to avoid overfitting [82].

- **Number of Epochs**: an epoch can be defined as the complete training cycle of the machine learning model. Usually, the number of epochs is greater than 1, and the validation error is used to determine the correct number of epochs. Therefore, the number of epochs is increased until there is a reduction in the validation error. If there is no improvement in the reduction error for consecutive epochs, the number of epochs stops rising [83].

6.1.1 Hyperparameter optimization method

The traditional algorithms for optimizing hyperparameters are Grid Search (exhaustive search in a predefined subset) [84] and Random Search (randomly selected from a predefined subset) [85]. However, as the amount of data and the space of hyperparameters increase, these algorithms are not efficient because they take too long to find the minimum or fail to reach it.
Optuna software allows users to adopt state-of-the-art algorithms for sampling hyperparameters and eliminating unpromising trials [86]. Optuna can be implemented with several state-of-the-art optimization methods (Sampling and Pruning strategies) to perform hyperparameter optimization quickly. Optuna implements a Bayesian optimization algorithm (Tree-structured Parzen Estimator - TPE) by default. In each trial, for each parameter, TPE fits a Gaussian mixture model (GMM) l(x) to the set of parameter values associated with the best target values and another GMM g(x) to the remaining parameter values. It chooses the parameter value x that maximises the ratio l(x)/g(x) [87].
In this thesis, it was therefore decided to use Optuna for Deep Learning models by setting optuna.pruners.MedianPruner() as the pruner (it uses the median stop rule) and optuna.samplers.TPESampler() as the sampler.

6.2 Loss Function

A fundamental neural network component is the loss function, which compares predicted and target output values to measure how well the neural network models the training data. During training, the goal is to minimize the loss between the predicted and target output values [88]. In machine learning, the performance of a classification algorithm in terms of accuracy depends on the choice of a loss function. In a binary classification problem, the typical loss function is binary cross entropy where there are only two possible output values: 0,1. The real value is compared with the probability that the input aligns to that category as shown in 6.1:

$$CELoss = -\frac{\sum_{i=1}^{N} \hat{y}_i \log(p_i) + (1 - \hat{y}_i) \log(1 - (p_i))}{N} \tag{6.1}$$

where $\hat{y}_i$ is the real target, p_i is the probability that the class is 1, 1 - p_i is the probability that the category is 0 and N is the number of points.

6.3 Evaluation metrics

The most crucial task in building any machine learning model is to evaluate its performance. Therefore, the main point is to measure the success of a machine learning model. Cancer events lead to an unbalanced dataset in which few people have cancer, so we have to choose the right evaluation metrics to account for this property. Consider, for example, the case of an unbalanced dataset with a class imbalance of 1:100, where the evaluation metric is the classification accuracy. In this problem, each example of the minority class (class 1 - abnormal class) will have 100 corresponding examples for the majority class (class 0 - regular class). Considering a user preference towards the examples of the minority class (positive), the accuracy is not adequate because the impact of the less represented but more important examples is reduced compared to that of the majority class. In this problem, a model that predicts class 0 (majority class) for all examples in the test set will have a classification accuracy of 99%, reflecting the distribution of major and minor examples predicted on average in the test set [89]. It is clear that accuracy fails in unbalanced classification because it is easy to achieve high accuracy without making valuable predictions. Therefore, accuracy as an evaluation metric only makes sense if the class labels are evenly distributed.
The metrics that best fit problems with unbalanced classes are: specificity, sensitivity, precision, recall, and f1 score. The range of these metrics is between 0 and 1, where 1 represents the ideal case. The confusion matrix is a popular measure for classification problems, which best describes the metrics used for this thesis. For binary classification, the confusion matrix scheme is described in Figure 6.1.

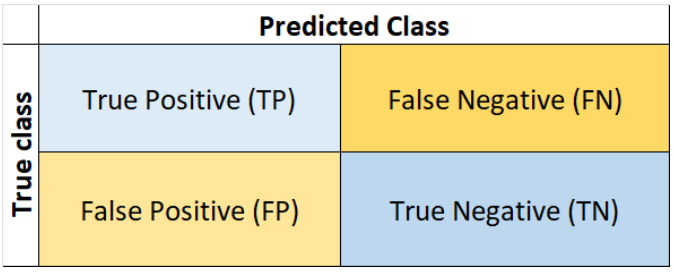


Figure 6.1: *Confusion matrix for binary classification problem.*

It represents TP values correctly classified, FP values in the relevant class while they should be in another class, FN values in another class while they should be in the relevant class, and TN values correctly classified in the other class [90]. Based on this table, precision, recall, specificity and sensitivity can be described as follows:

- **Precision**: it explains how many correctly predicted cases turned out to be positive. Precision is helpful in cases where false positives are more worrying than false negatives.

$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$

- **Recall or Sensitivity**: it explains how many actual positive cases we could predict correctly with our model. It is very useful when false negatives are more worrying than positives. It is therefore essential in this task, where it does not matter if a false alarm is generated, but positive cases must not go unnoticed.

$$Recall = Sensitivity = \frac{TP}{TP + FN} \tag{6.3}$$

- **F1 Score**: Precision and Recall are combined through an harmonic mean.

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6.4}$$

- **Specificity**: it measures the proportion of actual negatives correctly identified as such, i.e., the extent to which actual negatives are classified as such (i.e., there are few false positives).

$$Specificity = \frac{TN}{TN + FP} \tag{6.5}$$

In this scenario, the most essential metrics are sensitivity and specificity since:

- Sensitivity is the percentage of sick people correctly identified as having the disease: it is important to avoid undiagnosing sick people.
- Specificity is the percentage of healthy people correctly identified as not having the disease: it is therefore necessary to avoid a redundant number of biopsies that harm patients.

In the previous metrics, the two classes are distinguished using the threshold of 0.5: if the probability is less than 0.5, the class is assumed to be 0; otherwise, 1. In contrast, the ROC curve is a graph that illustrates the diagnostic capability of a system as the threshold varies [91]. For example, does it belong to the positive class if it is greater than 0.5 or 0.8? The curve plots the false positive rate (x-axis) against the true positive rate (y-axis) for different candidate threshold values between 0.0 and 1.0, as shown in Figure 6.2. A threshold can then be chosen that gives a balance between false positives and false negatives.

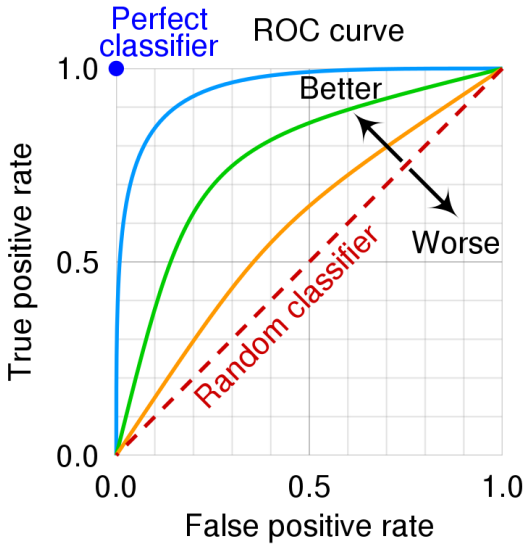


Figure 6.2: *ROC curve for binary classification problem. The diagonal shows the performance of a random classifier. Three classifiers (blue, orange, green) are shown [92]*

6.4 Services for sensitive data

As anticipated in Chapter 4, the dataset contains highly sensitive and private data; therefore, it is necessary to work in a specific platform for this type of data. The

project members had access through MinID to TSD, a service for sensitive data. The TSD is an IT platform operated by the University of Oslo for the storage and analysis of private data in compliance with privacy legislation [93]. It includes primary and general services and the development of new and tailored solutions to meet special needs in the user community. In the modern age of digital data, it is no longer possible to conduct research on sensitive data in local facilities, with limited capacity and no possibility of sharing results with colleagues. Research environments in universities and teaching hospitals have expressed the need for ample storage space for sensitive research data to be managed, such as clinical data, high-resolution MRI images, and patient video recordings.
In addition, an external service, Colossus, is available to use GPU nodes within TSD [94]. It is a cluster designed to handle multiple concurrent jobs and it is offered in collaboration with the University of Oslo. The cluster profile is suitable for parallel applications with high memory demands: it consists of 57 compute nodes, 4 extended memory nodes, and 6 GPU-accelerated nodes.

6.4.1 GPU for Deep Learning

For any neural network, the training phase of the deep learning model is the most resource-intensive activity because it processes input through hidden layers by updating the weights each time. Furthermore, these operations consist of multiplications between matrices: a computer can support these operations when you have thousands of parameters; if the neural network, on the other hand, has more than 10 billion parameters, a processor would take years to train these kinds of systems using the traditional approach. Thus, in order to solve this problem, GPUs were adopted. Graphics processing units (GPUs) are processing cores to speed up computational processes. These cores were initially designed to process images and visual data. Nowadays, GPUs have been adopted to improve computational processes because they can be used in parallel for huge distributed computational processes. Deep learning models, therefore, can be trained faster by performing all operations simultaneously instead of one after another.

6.5 Libraries

The main libraries used to perform the analysis are:

- Pandas: it is an open-source data analysis tool built on top of the Python programming language [95].
- NumPy: it offers comprehensive mathematical functions, random number generators and linear algebra routines [96].

- Matplotlib: it is a library for creating static or interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible [97].

- Optuna: it is an automatic hyperparameter optimization software framework, particularly designed for machine learning [87].

- Pytorch Lightning: it is a high-level python framework built on top of Pytorch. It was created for researchers, specifically for running deep learning models which involved research scaling, multi-GPU training and TPU [98].

6.6 Configuration

For each approach described, the dataset was randomly divided into two parts: one for model definition and the other for testing part. The two proportions are, respectively, 80%-20% of the total. The first set is then subdivided into training and validation sets with a proportion of 80:20 as shown in Figure 6.3. The validation set is the dataset that provides an unbiased evaluation of the model fit on the training set when modifying the model hyperparameters [99]. So naturally, the evaluation becomes more biased when the skill on the validation dataset is incorporated into the model setup.

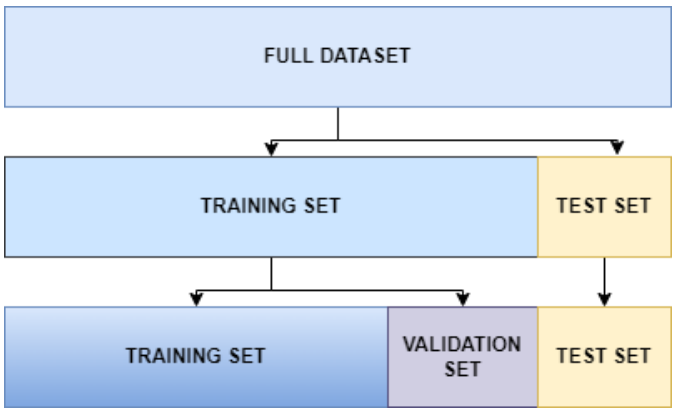


Figure 6.3: *Training, Validation and Test set.*

For each type of input and machine learning model described in Chapter 5, the hyperparameters are defined using the training and validation set. The model is then created using the same data set by setting the following rules:

- **The number of epochs**: it is set to 100, but the epochs are stopped before if there is no improvement.

- **Early Stopping**: it is a callback that can be used to monitor the validation loss [100]. The patience parameter counting the number of validation checks without any improvement is set to 15.
- **Learning Rate Monitor**: it monitors and logs the learning rate for learning rate schedulers during the training phase [101].
- **Optimizer**: the Pytorch Lightning library automatically selects it [102].
- **GPU support**: PyTorch's CUDA library is a parallel processing platform and programming model developed by Nvidia that focuses on general GPU processing [103]. It keeps track of which GPU is being used and all tensors are automatically allocated to that device.

Finally, the best-performing model is evaluated against the test division in terms of accuracy, precision, recall, specificity, and F1-score.

Chapter 7

Results and Discussion

This section shows the results obtained from the input and the respective models described in Chapter 5.
First of all, the final dataset consists of 85.000 patients after data cleaning operations, where:

- The training set has 67.998 time series and it is fully balanced between the two classes.

- The test set has 17.002 time series and it is unbalanced for the reasons explained in the Section 4.3.

It is also important to note that in the test set, the percentage of patients with cancer is no longer 10% of the total, but about 36%. This is due to data cleaning operations: the time series of people without biopsy or with negative biopsy have an average length of 5. They often have 'duplicate' values that were removed: thus, many time series did not meet the minimum length and were discarded.

7.1 Baseline models

The first approach uses the most simple machine learning models, as the goal is to use select algorithm that can reflect the medical way of thinking.
As a first step, we want to compare the model with PSA and age as input and then the same input with the addition of velocity. First, we analyze the correlation of the extracted features with risk. The matrix 7.1 shows that the features age, PSA and velocity are not correlated with risk; a higher correlation was expected but probably considering only absolute PSA values and age at the second-to-last visit does not create a linear association with the risk.

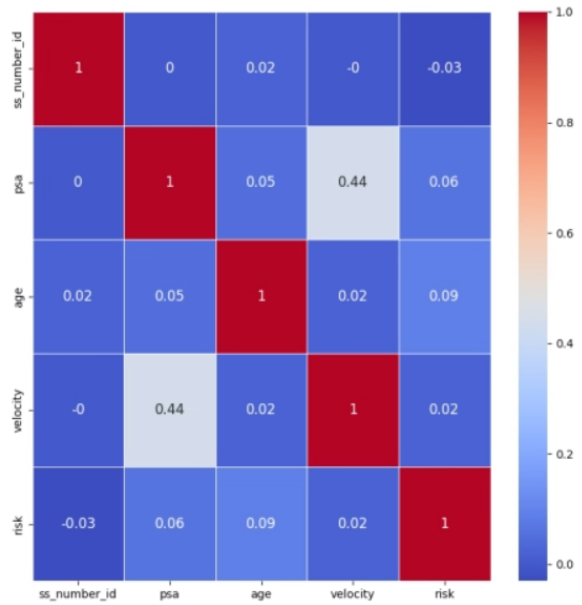


Figure 7.1: *Correlation matrix. The features are: psa, age and velocity.*

The Table 7.1 then shows the results using three different models, Decision Tree, Random Forest and Ada Boost, with the two input types.

Baselines									
Different	Decision Tree			Random Forest			Ada Boost		
input types	Acc	Sen	Spe	Acc	Sen	Spe	Acc	Sen	Spe
psa+age	0.824	**0.894**	0.785	**0.847**	0.870	**0.835**	0.824	0.892	0.786
psa+age+veloc	0.820	0.892	0.779	0.834	0.885	0.806	0.822	0.890	0.783

Table 7.1: *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. Acc, Sen, Spe are respectively the Accuracy, Sensitivity and Specificity.*

Let us now consider the performance with the first input (age and PSA): the random forest algorithm has better accuracy, probably because the other two algorithms are more prone to overfitting. On the one hand, decision trees are inclined to overfitting, especially when the tree is deep. On the other hand, Ada Boost, while providing more accurate predictions, is more susceptible to overfitting than Random Forest. The results obtained are generally better than expected as the preprocessing is very simple. Decision Tree, however, has a very high Sensitivity value, about 90%, so if we consider a set of 100 sick patients, only 10 are not detected. This is a good result, because we used a straightforward machine-learning

model. In contrast, the best Specificity is achieved by Random Forest. In general, the best model with this type of preprocessing is Random Forest.
Let us now consider the second input, where the velocity parameter is added to the previous input. In this case, better performance is expected because the variation in PSA is considered. Instead, this model adds no information, thus, no better results are obtained; probably, the extracted velocity formula is too trivial and it is not enough to consider only two PSA values over time. Perhaps other complex calculation methods, such as Ordinary Least Squares Regression, should be considered.
Another way to assess the relevance of features in the model is by using the permutation of feature importance. This procedure breaks the relationship between the feature and the target, so the decrease in accuracy indicates the model's dependence on the feature. Thus, a feature is 'important' if mixing its values increases the model's error because in this case, the model relied on the feature for prediction. This technique has the advantage of being model agnostic and it is applied to the Test Set as shown in Figure 7.2.

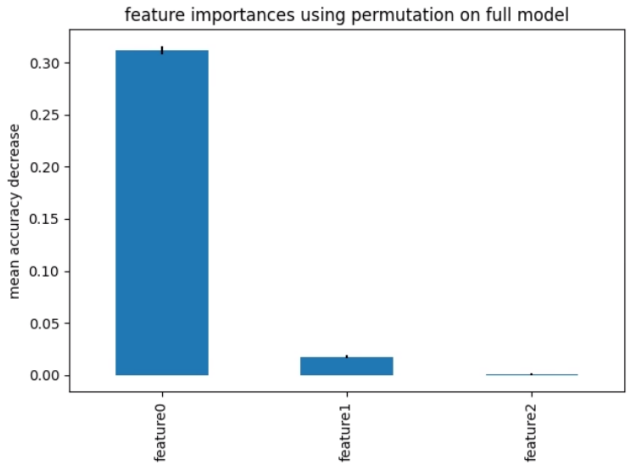


Figure 7.2: *The permutation of feature importance. Feature0, feature1 and feature2 are respectively age, PSA and velocity.*

The graph, therefore, shows that the most important features are:

- feature0 that is the age and the accuracy decreases by 30% when this feature is removed.
- feature1 that is the PSA value.

Surprisingly, age seems to be more important than PSA: in general, we realised that PSA and age are two relevant features in the prediction task. Furthermore,

the graph shows that when feature2, the velocity, is removed, the accuracy does not change, as expected.
Finally, we tried to understand the predictive power of features based only on δtime. The main reason is that strong irregularities in time characterize the treated time series; this is due to the doctor that decides the date of the next visit based on the patient's state of health. As the Matrix 7.3 shows, there is a correlation, although low, between the extracted features and the risk; if the time between visits increases, the risk decreases, and this assumption agrees with the medical opinion. It is therefore expected that the algorithm can learn patterns using only time features such as mean, median and mode of the distance between two visits.

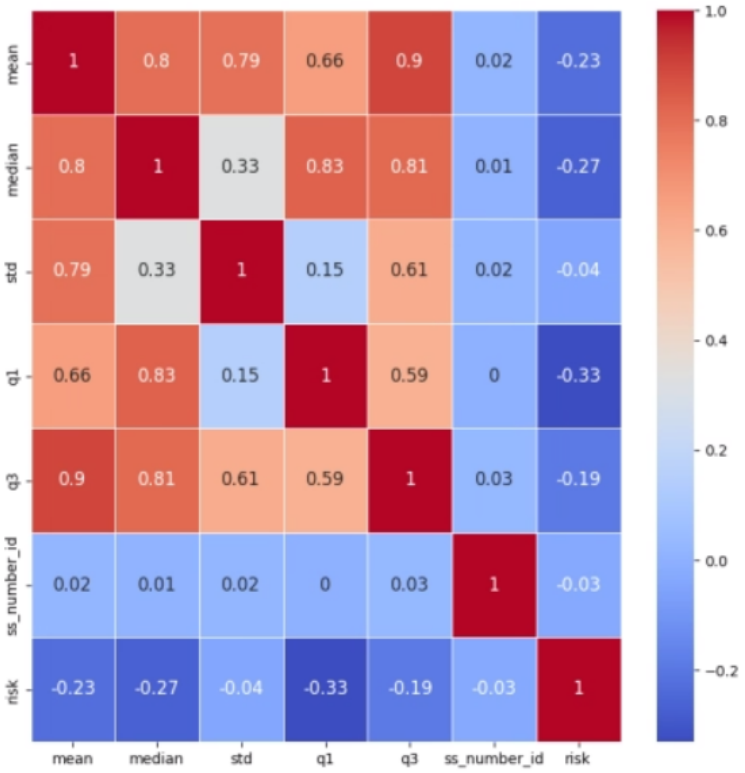


Figure 7.3: *The correlation matrix for ΔT features. The features are: mean, median, standard deviation and quantile.*

Model with Time Features

	Decision Tree			Random Forest			Ada Boost		
	Acc	Sen	Spe	Acc	Sen	Spe	Acc	Sen	Spe
ΔT	0.681	**0.712**	0.664	0.681	0.699	**0.678**	**0.686**	0.711	0.664

Table 7.2: *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. Acc, Sen, Spe are respectively the Accuracy, Sensitivity and Specificity.*

The Table 7.2 shows the results using the machine learning methods; the classification is not random, as accuracy of almost 70% is achieved. Thus, the irregularity of the time series cannot be ignored and especially unlike Marlin's assumptions [45], missing values are not missing at random.

7.2 Regularized time series

After evaluating the baselines, the analysis continued with implementing Deep Learning methods for time series.
In this case, it was decided to take into account the irregularity of the time series by working on the input; the series were therefore transformed from irregular to regular, as explained in Section 5.2.1. As shown in the Table 7.3, there are four different types of input where either the interpolation or zero-filling method was applied. In addition, BinVec represents the missing value indicators technique. First, during the hyperparameter tuning process, an essential parameter was selected: the discretization window. The permissible values were one year, two years, three years, and five years; it was impossible to select a smaller window such as six months, otherwise, the number of values in the time series would have become 142 and the training phase would have taken too long. In general, it is important to have a window so that, on the one hand, no information is lost and on the other hand not too many missing or interpolated values are inserted. Analyzing the statistics, on average, each man undergoes one PSA examination per year and in fact, the best window was found to be one year. This is probably due to fewer 'fake' values. The following Boxplots show the performance (evaluation metric = F1 Score) for each type of input using four different windows size; WND1Y, WND2Y, WND3Y and WND5Y mean respectively one years, two years, three years and five years. Each input confirms that the best model uses a one-year-window-size.

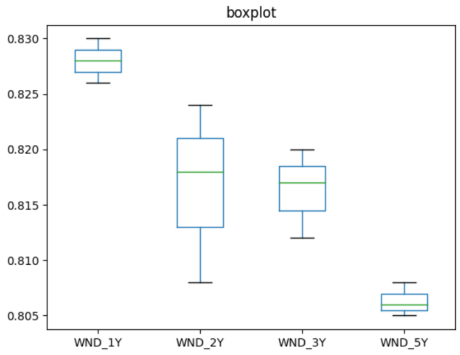


Figure 7.4: *Input1: interpolation*

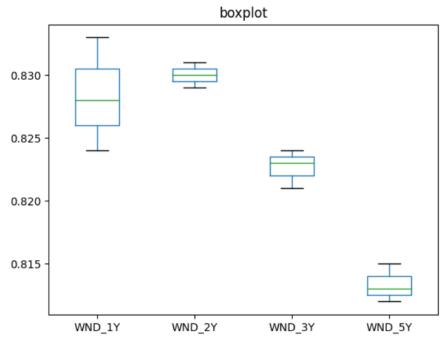


Figure 7.5: *Input2: interpolation + binary missing indicators*

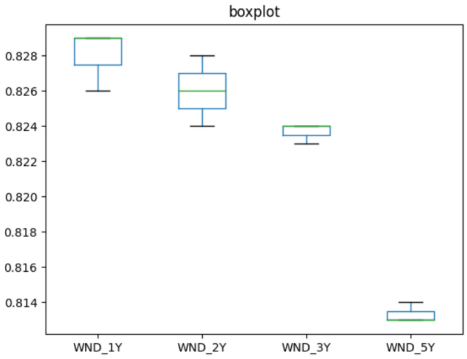


Figure 7.6: *Input3: discretization*

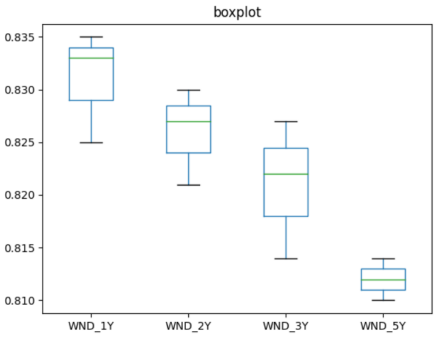


Figure 7.7: *Input4: discretization + binary missing indicators*

So let us now analyze the results using one year as the discretization window. The main difference to Baselines is that in this approach, the input is not just a few features such as age and PSA but rather a time series. In this case, the deep learning models are Recurrent Neural Networks; a higher performance is therefore expected as the network should have a memory of past states and the ability to learn from long-time sequences. Furthermore, adding the binary vector of missing values should help the network distinguish between real and generated values with a consequent increase in accuracy.

As the Table 7.3 shows, the accuracy, approximated to two decimal places, is about the same across all four input types; however, it is not a reliable metric for unbalanced datasets. So let us now analyze the other two metrics: the linear interpolation technique appears to be better than zero-filling in both Specificity and Sensitivity measures; moreover, the addition of the binary vector makes the model perform slightly better in Sensitivity calculations. In general, however, the missing value indicators did not add any information for a marked performance

Approach 1: Regularization

Different input types	RNNs Accuracy	Sensitivity	Specificity
Interpolation	**0.871**	0.862	**0.877**
Interpolation + BinVec	0.866	**0.879**	0.854
Zero-filling	0.868	0.872	0.863
Zero-filling + BinVec	**0.871**	0.871	0.870

Table 7.3: *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics. The following techniques are applied: interpolation and zero-filling.*

improvement. In addition, no significant differences in performance can be found between the interpolation technique and zero-filling, so this means that (1) -1 is recognized by the network as a missing value and (2) the linear interpolation technique did not create too many "outliers" in the handling of null values.
In conclusion, this approach is significantly better than the previous one, as it goes from accuracy of 0.847 to 0.871. However, the limitations of time series regularization mentioned in Section 5.2.1, must also be taken into account: the generation of time series with constant false trends could reduce significantly the performance.

7.3 Irregular time series with the addition of new features

Finally, a different approach was proposed after analyzing the possible limitations of the time series regularization technique. In this case, the problem of time series irregularity is treated without modifying the input; instead, new features can help the neural network estimate the best PSA evolution formula. As described in Section 5.2.2, the four selected input types are compared with the following Deep Learning models: RNN, LSTM, CNN, MKL and ROCKET. It is expected that the model that performs best will be different depending on the input type. In general, this approach uses raw features and thus should not generate fake assumptions in the Preprocessing Phase.
As shown in the Tables 7.4 and 7.5, we can see that there are input-model combinations that give promising results, around 90% of accuracy. Furthermore, in the previous models, Specificity never reached results higher than 0.90, whereas in this case, we have a value of about 0.92; this allows us to say that out of 100 healthy patients, only 8 will undergo unnecessary biopsies. One of this thesis' goals is to reduce overtreatment, which can often damage the patient's quality of life. Thus,

although Sensitivity is considered more important in the medical field, in this case, Specificity is also relevant; the result obtained is therefore satisfactory.

Approach 2: No Regularization

Different input types	LSTM			CNN			MKL		
	Acc	Sen	Spe	Acc	Sen	Spe	Acc	Sen	Spe
age+psa	0.887	0.860	0.895	0.437	0.999	0.507	0.817	**0.899**	0.767
age+psa+others*	**0.889**	0.878	0.900	0.867	0.887	0.864	0.863	0.886	0.848
categ(age+psa)*	0.863	0.849	0.881	0.866	0.876	0.866	0.865	0.884	0.859
categ(age+psa)+others*	**0.889**	0.852	**0.915**	0.884	0.864	0.889	0.887	0.878	0.901

Table 7.4: *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics.*
* age+psa+others means that the features are age, psa, δT and δPSA while categ(age+psa) represents the categorical features of PSA and age.

Approach 2: No Regularization

Different input types	MLP			ROCKET		
	Acc	Sen	Spec	Acc	Sen	Spec
age+psa	0.861	0.889	0.849	0.522	0.467	0.552
age+psa+others*	0.875	0.863	0.876	0.524	0.469	0.554
categ(age+psa)*	0.858	0.887	0.844	0.844	0.829	0.852
categ(age+psa)+others*	0.867	0.846	0.878	0.867	0.845	0.879

Table 7.5: *The results are the average of 6 iterations. Bold represents the best performing configuration using different evaluation metrics.*
* age+psa+others means that the features are age, psa, δT and δPSA while categ(age+psa) represents the categorical features of PSA and age.

In general, the mentioned results show that adding new features is better than introducing new values to regularize the time series. However, it must be noted that given a certain input, some models do not perform well. Thus, it is necessary to understand which input is best for each model. In order to compare the results more easily, it was decided to use a single metric, the F1 score.

LSTM is the model expected to perform best because, compared to other neural networks, it can remember historical states; in prostate cancer prediction, it is essential to consider past PSA values because the most important factor is how PSA varies over time. The Boxplots below show that LSTM performs well and the standard deviation is low even when the input includes only PSA and age. In contrast, with categorical input, there is a significant reduction in performance; the

two possible explanations are (1) the categories for PSA and age were not precisely defined or (2) LSTM fails to efficiently extract patterns from categorical variables. On the other hand, CNN shows a good performance in all cases except the input with only PSA and age; moreover, the standard deviation is very high. This means that the network, through an initial input corresponding to a 2 x time series length matrix fails to efficiently extract temporal and spatial information.
In contrast, the MLK model shows improvements because it processes the same input through different kernel sequences, thus, it can extract more information. But, again, performance is lower for case 1. Instead, MLP shows that the performance remains always low: the time series is no longer treated as a sequence but rather as a set of features, so any temporal relationship is lost.

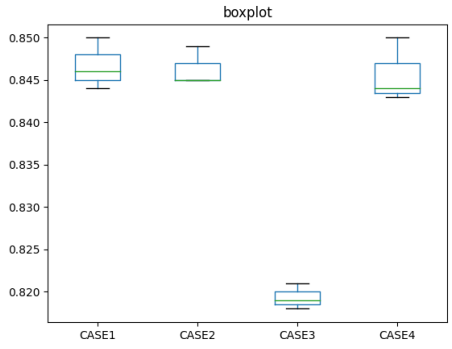


Figure 7.8: *LSTM model*

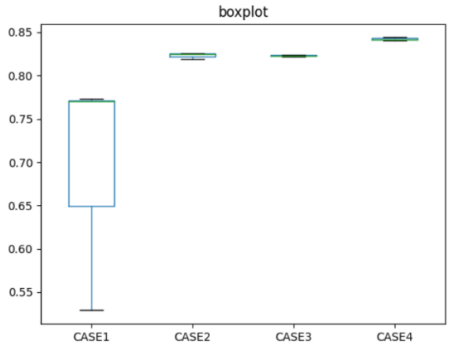


Figure 7.9: *CNN model*

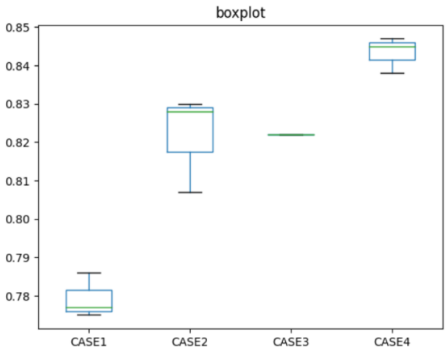


Figure 7.10: *MKL model*

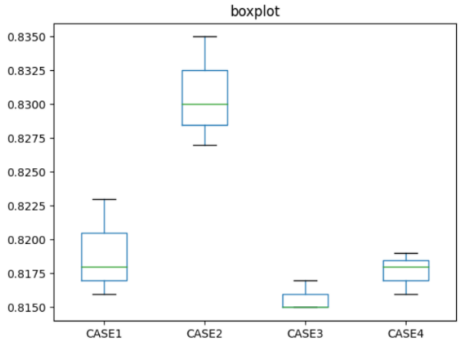


Figure 7.11: *MLP model*

Let us now analyze ROCKET [12]. It was decided to implement this algorithm because the time for the training phase and the selection of hyperparameters is often too long in neural networks. Instead, this algorithm has only one hyperparameter (the number of kernels) and allows a prediction to be made in a few minutes. First, ROCKET was developed as a model for univariate time series but it was

later adapted for multivariate series. In fact, in this thesis, it was applied to multivariate time series. Secondly, the performance obtained should be similar to state-of-the-art algorithms for time series classification, such as LSTM [12]. In reality, as the Table 7.6 shows, acceptable performance is obtained when PSA and age are used as categorical variables. One possible explanation is that ROCKET works on thousands of random kernels and it extracts the most essential features from these; however, if only 2 x time series size matrices are provided as input, the algorithm cannot extract enough spatial and temporal information. In general, it is still an algorithm that is easy to understand, scalable and with acceptable performance when several features are selected as input.

Rocket: the performance

INPUT	F1 SCORE
Case1	0.41
Case2	0.41
Case3	0.79
Case4	**0.82**

Table 7.6: *Bold represents the best performing configuration.*

7.4 The best model

In conclusion, after analyzing all the proposed approaches, we have to choose the best model, considering both Specificity and Sensitivity values. Thus, the best model is the "unregularized time series" approach with features addition; the input selected is age, PSA, ΔT and ΔPSA. The neural network is the RNN.
The main hyperparameters 7.7 are:

Hyperparameters

batch size	512
rnn type	GRU
bidirectional	True
hidden size	1024
number of layers	2
learning rate	0.002

Table 7.7

The confusion matrix 7.12, therefore, shows the results of the model where label 0 indicates patients without biopsy or negative biopsy and 1 indicates patients with cancer. Sensitivity and Specificity are respectively 0.89 and 0.88.

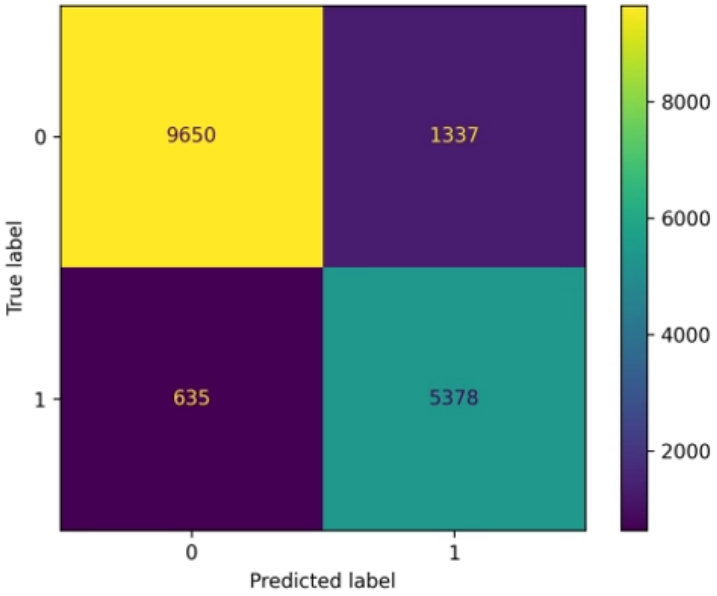


Figure 7.12: *The confusion matrix.*

Finally, the Figure 7.13 shows the ROC curve where on the x-axis the FPs are shown and on the y-axis the TPs.

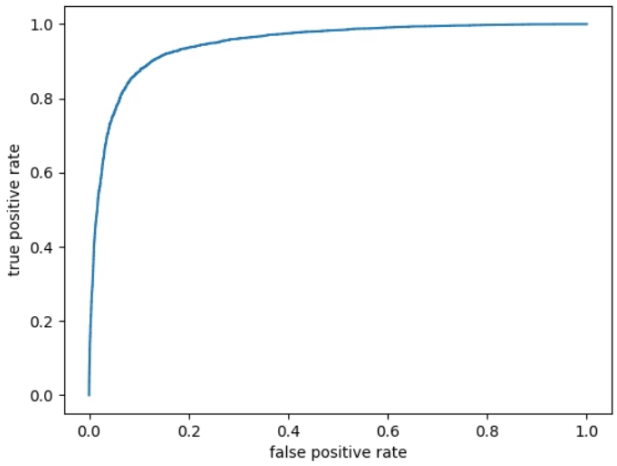


Figure 7.13: *The ROC curve.*

7.5 The ability of AI to support medical decision

The use of AI to support doctors in patient care is auspicious [104]. From the rapid identification of new drug candidates to aiding in the diagnosis of severe illnesses to helping manage resources in hospitals, AI has proven its value. Artificial intelligence is therefore intended to help doctors work more efficiently. However, current AI-based software is limited to performing specific tasks to support the doctor's decision-making process. It cannot replace the doctor's figure and, therefore, it cannot handle difficult tasks such as making clinical decisions [104].
Thus, some information has been extracted from the best model to support medical decisions. For example, a doctor might need to understand the number of visits for each patient to have a good prediction; or the doctor might be interested in understanding the age range for which the machine learning model can provide a prediction with a certain accuracy. Finally, the doctors would like to detect the most aggressive cancers because these are the ones that need early intervention. Therefore, interesting information has been extracted for each type of cancer risk.

The Table 7.8 shows the predictive accuracy for each risk category in patients with cancer. It is evident that the algorithm learns patterns more easily in patients with advanced cancer; for example, PSA values may be particularly elevated, PSA velocity is high and the frequency between visits is also high. The model, on the other hand, learns with greater difficulty from the data of low-risk patients: in this case, it is possible that patients have a PSA slightly higher than the cut-off value and the doctor has decided to take a biopsy to better delineate the situation. However, a slightly above-average PSA value is not an indicator of prostate cancer, as healthy men can have PSAs higher than 4 ng/ml. The accuracy for the 'unknown risk' category is instead particularly low, and in order to understand why, it would be necessary to learn more about which patients were included in this category. In any case, these results show that the model can be sufficiently reliable for patients with cancer that need to be treated immediately.

Cancer risk categories	Accuracy
Low risk	0.780
Intermediate risk	0.922
High risk	**0.954**
Metastatic risk	0.938
Regional risk	0.951
Unknown risk	0.607

Table 7.8: *The performance according to the risk categories. Bold represents the best performing configuration.*

The Table 7.9 shows the model's performance according to the time series length. Contrary to expectation, the prediction has better accuracy if there are fewer points in the time series. LSTM is generally suitable for learning from long sequences by forgetting non-essential information and remembering important information. However, in this case, "too old" PSA values could cause noise; this is only an assumption and it would be necessary to implement interpretability models to understand why it happens. It should also be noted that this is a favorable point from a medical point of view: the doctor doesn't need to have too many values to have good prediction accuracy. So if a patient has collected 6/7 values in three or four years, the algorithm can predict the risk accurately.

Number of visits	Accuracy	Sensitivity	Specificity	F1 score
Visits < 10	**0.898**	0.889	**0.902**	**0.890**
$10 \leq$ Visits < 20	0.877	**0.902**	0.861	0.871
$20 \leq$ Visits < 30	0.828	0.893	0.787	0.825
$30 \leq$ Visits < 40	0.821	0.880	0.797	0.804
Visits ≥ 40	0.848	0.875	0.824	0.828

Table 7.9: *The performance according to the time series length. Bold represents the best performing configuration.*

The doctors also emphasized that it is essential to discover cancer in young patients because it can be treated more effectively. When prostate cancer is diagnosed in elderly patients, the doctor may not necessarily decide to treat the cancer; clinicians must weigh the risks and benefits of possible treatments/interventions on patients of advanced age. Thus, when the patient is too old and suffering from other diseases, doctors may decide not to intervene to compromise the patient's quality of life. The table 7.10 shows that patients who start PSA tests between the ages of 30 and 45 and record their last visit between the ages of 55 and 65 have an exceptionally high predictive accuracy, around 93%. This is a promising result from a medical point of view, as the algorithm performs very well on non-elderly patients. Furthermore, it can be seen that the performance decreases by keeping the minimum age range of the first test fixed but increasing the age at which the last visit was recorded. It should also be noted that by starting PSA tests between the ages of 55 and 65, the predictive accuracy continues to be satisfactory, at around 90%; this is an important result as most patients start checking their PSA value around the age of 50.

Age range	Mean Length	Accuracy	Sensitivity	Specificity	F1 score
30≤min<45 55≤max< 65	11	**0.931**	0.899	**0.937**	0.855
30≤min<45 65≤max<75	13	0.866	1.0	0.846	0.791
30≤min<45 75≤max<85	-	-	-	-	
45≤min<55 55≤max<65	9	0.895	0.868	0.905	0.875
45≤min<55 65≤max<75	13	0.885	0.829	0.904	0.854
45≤min<55 75≤max< 85	19	0.878	0.833	0.891	0.834
55≤min<65 65≤max<75	11	0.894	**0.915**	0.878	**0.891**
55≤min<65 75≤max<85	15	0.879	0.823	0.899	0.849
65≤min<75 75≤max<85	11	0.853	0.887	0.827	0.852

Table 7.10: *The performance according to the age. Bold represents the best performing configuration.*

In conclusion, the algorithm seems to perform best on young patients with an average of about ten values; moreover, if a patient has cancer, the model detects advanced stages of the disease more easily. These conditions represent doctors' demands: diagnose aggressive cancers in young patients with a reasonable number of PSA tests.

Chapter 8

Conclusion and Future works

8.1 Conclusion

This research dealt with predicting prostate cancer risk based on PSA variation over time. An attempt was made to overcome the limitations of "handcrafted" velocity formulas using machine-learning algorithms. Therefore, this was a challenge that required an in-depth analysis of the problem, preprocessing techniques, models and hyperparameters. Overall, each approach was conducted with in-depth performance analysis and the results were discussed. In addition, the time series irregularity problem was handled by proposing two different approaches, regularization of the time series and extraction of new features, such as the temporal distance between visits. The problem of time series irregularity is a fundamental challenge in the clinical field that cannot be ignored.
The dataset provided by the Norwegian health system is unique and was the result of the effort of the doctors who collected and recorded all the data over the years. A large number of recorded men made it possible to obtain robust models, mainly avoiding the problem of overfitting or having a sample of patients with cancer limited to a few hundred, as is often the case.
The proposed approaches and the baseline models were compared using different metrics such as Sensitivity and Specificity. It was emphasized that, although Sensitivity is fundamental from a medical point of view, Specificity cannot be ignored because one of the objectives of this thesis is to reduce the number of biopsies, considered an extremely invasive procedure for the patient. Therefore, the best model was selected, emphasizing how new features such as temporal distance and PSA variation between visits lead to better performance. Furthermore, the RNN proved to be the best neural network as it is able to capture temporal dependencies.

The performance of the selected model shows satisfactory results with an accuracy of 89%, Specificity of 90%, and Sensitivity of 88%.
In addition, a scalable algorithm, ROCKET [12], was proposed to reduce training time exponentially. However, contrary to expectation, ROCKET's performance was highly input-dependent, and in some cases, its prediction was random with 50% of accuracy.
The results produced by this thesis suggest that the proposed methods and the use of Deep Learning for prostate cancer risk prediction are promising and can be an excellent tool to support clinical decision-making.

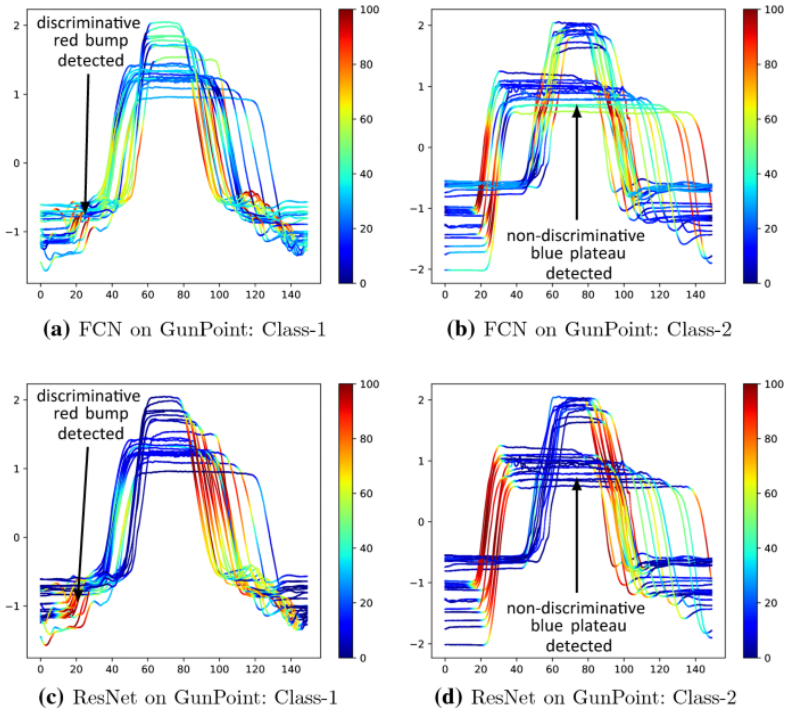
However, it is now necessary to analyze how the models developed by machine learning experts can be used by doctors and the possible limits of AI. In recent years, it has been shown that deep learning is suitable for medical data as it can identify patterns in sparse and noisy data with minimal feature engineering [105]. Current successes have shown that ML models outperform experienced doctors; many health conditions present heterogeneously, making it difficult to establish an accurate diagnosis over time [104]. A deep-learning healthcare system would allow all doctors to work with the same level of expertise as a group of top doctors. Indeed, a fundamental difference between human and machine learning is that humans can learn to make general associations from small amounts of data. In contrast, a machine learning model can be trained using tens of millions of medical records containing billions of data points without any lapse of attention. It would be challenging for a doctor to see more than a few tens of thousands of patient data points in an entire career [105].
Another key aspect is that deep learning models can be shared between hospitals through the transmission of patients data; thus, a new system of precision medicine could be created by learning from the decisions and outcomes of different doctors treating different patients [105]. Contrary to what many think, introducing artificial intelligence into medicine will not sideline doctors but enhance their strengths. Doctors will be able to focus on uniquely human elements, such as asking accurate questions to the patient to uncover more nuanced symptoms and building trust through personal relationships to guide the implementation of computerized diagnoses and treatment plans [104].
Moreover, doctors cannot constantly interact individually with all patients who may need treatment. In the future, however, machine learning may extend the reach of clinicians to provide expert-level medical assessments without personal involvement. For example, patients with new rashes could get a diagnosis by sending a photo taken with their smartphone, thus avoiding unnecessary emergency visits. The model could then identify doctors with the most relevant expertise and availability. At the same time, comfort would increase and costs would be reduced [104].
Although there are large volumes of clinical data from which meaningful information

could be extracted, it is still not used to improve medical practice. Today, many industries use previous actions and results from data to enable smarter choices. For example, Amazon recommends products to a user based on their previous research. However, the obstacles to bringing medicine into the era of Big Data are operational and cultural. In addition, there are still strong doubts about how reliable AI models can be and public privacy concerns [104]. The problem is that artificial intelligence could base its recommendations on false assumptions in the data, leading to misdiagnoses. It is therefore necessary to understand which AI application is reliable to improve the treatment of patients. In general, the people and doctors have a certain mistrust of machine learning [106]. This is because its inner functionalities are difficult to interpret and there is also a fear of how personal data will be used. Indeed, a key issue is compliance with privacy requirements, as clinical data must only be used in a secure and patient-friendly manner. People fear that their data could be used to discriminate against them. For this reason, certain types of information, such as genetic conditions, require anonymization procedures. Public concern for privacy influences people's willingness to share data, which can affect the accuracy of AI recommendations, as the model would use a dataset that does not truly represent the population [106]. However, by being transparent and demonstrating the steps taken to verify the reliability of the AI, researchers and developers can help give confidence to people who want to provide their data.
In conclusion, it has been shown that using AI with its ML models to support clinicians in patient care is very promising [106]. Problems only arise if the data quality is not adequately verified and the reliability of the AI has not been tested. In general, there is fear among people that robots will take people's jobs away, fear of data privacy, and fear of who is ultimately responsible if an AI decision turns out to be wrong. But rather than throwing away innovative tools that can improve people's life expectancy, there is a need for more excellent knowledge and awareness among people of this unknown science, AI.

8.2 Further Work

As explained before, AI suffers from people's scepticism because an important decision, such as predicting cancer, is entrusted to a machine [107]. Unfortunately, recent deep learning models have millions of parameters and why they make a particular decision have become obscure. Therefore, people can hardly trust the model until they understand how the model works; people tend to trust a white box rather than a black one. To interpret means 'to explain or present in understandable terms' and interpretability techniques make certain decisions made by machines understandable by humans [108]. Furthermore, the interpretability of the model offers other advantages, including finding biases in the dataset and debugging the

model. Since biases can be easily detected in interpretable models, interpretable models are more ethical and comply with the requirements of the General Data Protection Regulation (GDPR) [109]. Finally, interpretability could help find causality, such as 'smoking is one of the causes of lung cancer'. From the point of view of interpretability, confidence does mean not only the 'performance' of a model but also the robustness the model has for real scenarios [107]. Without interpretability, it is difficult to understand whether the learned characteristics make sense; therefore, the model cannot be trusted. This definition of interpretability corresponds to the one commonly used for Explainable Artificial Intelligence (XAI), which is the research domain that provides insights into the behavior of complex models learned by various machine learning algorithms [108].
For example, CAM and Grad-CAM are two methods used to explain the results of time series classification. The idea is to highlight the discriminative regions of the input time series to show feature importance using a heatmap [17]. The Figure 8.1 illustrates a CAM-based explanation for the ResNet and FCN classifiers: it can be seen that the parts of the time series that contributed most to the prediction are shown in red.



(a) FCN on GunPoint: Class-1 **(b)** FCN on GunPoint: Class-2

(c) ResNet on GunPoint: Class-1 **(d)** ResNet on GunPoint: Class-2

Figure 8.1: *Highlighting with the class activation map of the contribution of time series region for the two classes when using the FCN and ResNet classifiers. Red corresponds to a high contribution and blue to almost no contribution to correct class identification [17].*

Thus, in the context of the thesis, these methods could be used to understand which period was most influential in cancer prediction. For example, it is expected that for a patient with cancer, the period in which the velocity of change in PSA increases rapidly is the most influential. Furthermore, one of the results where no explanation could be given is: why do shorter time series perform better than long time series? RNNs can handle long-term time dependencies, so it may be that PSA values further apart in time only create noise. Thus, the only way to understand these results is through interpretation methods. The thesis has therefore laid a reasonable basis and raised good questions for further work with XAI methods.

Bibliography

[1] Rawla and Prashanth. «Epidemiology of prostate cancer». In: *World journal of oncology 10.2* (2019) (cit. on pp. 1, 2).

[2] URL: `https://teachmeanatomy.info/pelvis/the-male-reproductive-system/prostate-gland/` (cit. on p. 1).

[3] «Improving cancer screening in the European Union». In: (2022) (cit. on pp. 1–3).

[4] Georgina Cosma, Ste´phanie E. McArdle, Gemma A. Foulds, Simon P. Hood, Stephen Reeder, Catherine Johnson, Masood A. Khan, and A. Graham Pockley. «Prostate Cancer: Early Detection and Assessing Clinical Risk Using Deep Machine Learning of High Dimensional Peripheral Blood Flow Cytometric Phenotyping Data». In: (Dec. 2021) (cit. on p. 2).

[5] Ilic and Dragan. «Screening for prostate cancer». In: *Cochrane database of systematic reviews 1* (2013) (cit. on p. 2).

[6] Thompson and Ian. «Assessing prostate cancer risk: results from the Prostate Cancer Prevention Trial». In: *Journal of the National Cancer Institute 98.8* (2006) (cit. on p. 2).

[7] Gulati et al. «Reconciling the Effects of Screening on Prostate Cancer Mortality in the ERSPC and PLCO Trials». In: (Oct. 2017) (cit. on p. 3).

[8] Heijnsdijk et al. «Lifetime Benefits and Harms of Prostate-Specific Antigen–Based Risk-Stratified Screening for Prostate Cancer». In: (May 2020) (cit. on p. 3).

[9] Samuel and Arthur L. «Some studies in machine learning using the game of checkers. II—recent progress». In: *Computer Games I* (1988) (cit. on p. 3).

[10] Norgeot, Beau, Benjamin S. Glicksberg, and Atul J. Butte. «A call for deep-learning healthcare». In: *Nature medicine 25.1* (2019) (cit. on pp. 3, 4).

[11] Vickers AJ and Brewster SF. «PSA Velocity and Doubling Time in Diagnosis and Prognosis of Prostate Cancer». In: *Br J Med Surg Urol* (2012) (cit. on pp. 3, 18, 24, 25, 43).

[12] Angus Dempster, François Petitjean, and Geoffrey I. Webb. «ROCKET: Exceptionally fast and accurate time series classification using random convolutional kernels». In: *arXiv* (2019) (cit. on pp. 5, 22, 23, 63, 64, 70).

[13] Rajkomar et al. «Scalable and accurate deep learning with electronic health records». In: (2018) (cit. on p. 6).

[14] Nweke et al. «Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges». In: (2018) (cit. on p. 6).

[15] Nweke et al. «Convolutional neural network with multi-task learning scheme for acoustic scene classification». In: (2017) (cit. on p. 6).

[16] Susto GA, Cenedese A, and Terzi M. «Time-series classification methods: Review and applications to power systems data». In: (2018) (cit. on p. 6).

[17] Fawaz, Hassan Ismail, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. «Deep Learning for Time Series Classification: A Review». In: *Data Mining and Knowledge Discovery 33.4* (2019) (cit. on pp. 6–10, 21, 22, 72).

[18] McCulloch, Warren S., and Walter Pitts. «A logical calculus of the ideas immanent in nervous activity». In: *The Bulletin of Mathematical Biophysics 5.4* (1943) (cit. on p. 6).

[19] Bottou and Léon. «Stochastic gradient learning in neural networks». In: *Proceedings of Neuro-Nımes 91.8* (1991) (cit. on p. 7).

[20] Suter and Bruce. «The multilayer perceptron as an approximation to a Bayes optimal discriminant function». In: *IEEE transactions on neural networks 1.4* (1990) (cit. on p. 8).

[21] Jiuxiang Gu and Zhenhua Wang. «Recent advances in convolutional neural networks». In: *Pattern Recognition* (2018) (cit. on p. 8).

[22] Cristian Borges Gamboa et al. «Deep Learning for Time-Series Analysis». In: *ArXiv* (2017) (cit. on p. 8).

[23] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. «Understanding of a convolutional neural network». In: *International Conference on Engineering and Technology (ICET)* (2017) (cit. on p. 9).

[24] R. Chauhan, K. K. Ghanshala, and R. C. Joshi. «Convolutional Neural Network (CNN) for Image Detection and Recognition». In: *First International Conference on Secure Cyber Computing and Communication (ICSCCC)* (2018) (cit. on p. 9).

[25] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. «Comparative Study of CNN and RNN for Natural Language Processing». In: *arXiv* (2017) (cit. on p. 9).

[26] et al. McLaren Mitchell. «Application of convolutional neural networks to speaker recognition in noisy conditions.» In: *Fifteenth Annual Conference of the International Speech Communication Association* (2014) (cit. on p. 9).

[27] Kevin Fauvel, Tao Lin, Veronique Masson, Elisa Fromont, and Alexandre Termier. «XCM: An Explainable Convolutional Neural Network for Multivariate Time Series Classification». In: *arXiv* (2020) (cit. on p. 10).

[28] Yin and Wenpeng. «Comparative study of CNN and RNN for natural language processing». In: *arXiv* (2017) (cit. on p. 10).

[29] Mao and Junhua. «Deep captioning with multimodal recurrent neural networks (m-rnn)». In: *arXiv* (2014) (cit. on p. 10).

[30] Koutnik and Jan. «A clockwork rnn». In: *International Conference on Machine Learning. PMLR* (2014) (cit. on p. 10).

[31] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. «DeepCare: A Deep Dynamic Memory Model for Predictive Medicine». In: *Lecture Notes in Computer Science(), vol 9652. Springer* (2017) (cit. on p. 11).

[32] Johannes Kepler. «The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions». In: *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems* (1998) (cit. on p. 11).

[33] Alex Sherstinsky. «Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network». In: *arXiv* (2021) (cit. on pp. 11, 43).

[34] Sepp Hochreiter. «Long Short-term Memory». In: *PubMed* (1997) (cit. on pp. 11, 43).

[35] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. «A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures». In: *Neural Comput* (2019) (cit. on p. 11).

[36] Jos van der Westhuizen and Joan Lasenby. «The unreasonable effectiveness of the forget gate». In: *arXiv* (2018) (cit. on p. 12).

[37] K. Smagulova and A.P. James. «Overview of Long Short-Term Memory Neural Networks». In: *Deep Learning Classifiers with Memristive Networks. Modeling and Optimization in Science and Technologies* (2020) (cit. on pp. 12–14).

[38] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. «Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling». In: *arXiv* (2014) (cit. on p. 13).

[39] M. Schuster and K. K. Paliwal. «Bidirectional recurrent neural networks». In: *IEEE Transactions on Signal Processing* (1997) (cit. on p. 14).

[40] URL: http://colah.github.io/posts/2015-09-NN-Types-FP/ (cit. on p. 14).

[41] P.B. Weerakody, K.W. Wong, G. Wang, and W. Ela. «A review of irregular time series data handling with gated recurrent neural networks.» In: *Neurocomputing* (2021) (cit. on pp. 15, 16).

[42] Steven Cheng-Xian Li and Benjamin M. Marlin. «Learning from Irregularly-Sampled Time Series: A Missing Data Perspective». In: (2020) (cit. on pp. 15, 40).

[43] Satya Narayan Shukla and Benjamin M. Marlin. «A Survey on Principles, Models and Methods for Learning from Irregularly Sampled Time Series». In: (2020) (cit. on pp. 15–17).

[44]] E. J. Keogh, K. Chakrabarti, M. J. Pazzani, and S. Mehrotra. «Dimensionality reduction for fast similarity search in large time series databases». In: *Knowledge and Information Systems* (2001) (cit. on p. 18).

[45] Marlin et al. «Unsupervised Pattern Discovery in Electronic Health Care Data Using Probabilistic Clustering Models». In: (2012) (cit. on pp. 18, 38, 59).

[46] Zachary C. Lipton, David C. Kale, and Randall Wetzel. «Modeling Missing Data in Clinical Time Series with RNNs». In: *ArXiv* (2016) (cit. on pp. 19, 40, 41).

[47] Satya Narayan Shukla and Benjamin M Marlin. «Interpolation-Prediction Networks for Irregularly Sampled Time Series». In: *ArXiv* (2019) (cit. on pp. 19, 20, 40).

[48] Wang, Yan, and Oates. «Time series classification from scratch with deep neural networks: A strong baseline». In: *International Joint Conference on Neural Networks (IJCNN)* (2017) (cit. on pp. 20–22).

[49] Müller and Meinard. «Dynamic time warping. Information retrieval for music and motion». In: *Information retrieval for music and motion* (2007) (cit. on p. 20).

[50] Baydogan, Mustafa Gokce, George Runger, and Eugene Tuv. «A bag-of-features framework to classify time series». In: *IEEE transactions on pattern analysis and machine intelligence* (2013) (cit. on p. 20).

[51] Schafer and Patrick. «The BOSS is concerned with time series classification in the presence of noise». In: *Data Mining and Knowledge Discovery 29.6* (2015) (cit. on p. 21).

[52] Lines and Jason. «A shapelet transform for time series classification». In: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (2012) (cit. on p. 21).

[53] Bagnall and Anthony. «Time-series classification with COTE: the collective of transformation-based ensembles». In: *IEEE Transactions on Knowledge and Data Engineering* (2015) (cit. on p. 21).

[54] Lines, Jason, Sarah Taylor, and Anthony Bagnall. «Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles». In: *ACM Transactions on Knowledge Discovery from Data 12.5* (2018) (cit. on p. 21).

[55] Nweke and Henry Friday. «Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges». In: *Expert Systems with Applications 105* (2018) (cit. on p. 21).

[56] Cui, Zhicheng, Wenlin Chen, and Yixin Chen. «Multi-scale convolutional neural networks for time series classification». In: *arXiv* (2016) (cit. on p. 21).

[57] Zhao and Bendong. «Convolutional neural networks for time series classification». In: *Journal of Systems Engineering and Electronics 28.1* (2017) (cit. on p. 21).

[58] Le Guennec, Arthur, Simon Malinowski, and Romain Tavenard. «Data augmentation for time series classification using convolutional neural networks». In: *ECML/PKDD workshop on advanced analytics and learning on temporal data* (2016) (cit. on p. 22).

[59] Zheng and Yi. «Time series classification using multi-channels deep convolutional neural networks». In: *International conference on web-age information management* (2014) (cit. on p. 22).

[60] Tanisaro, Pattreeya, and Gunther Heidemann. «Time series classification using time warping invariant echo state networks». In: *15th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2016) (cit. on p. 22).

[61] Lucas and Benjamin. «Proximity forest: an effective and scalable distance-based classifier for time series». In: *Data Mining and Knowledge Discovery 33.3* (2019) (cit. on p. 22).

[62] Shifaz and Ahmed. «TS-CHIEF: a scalable and accurate forest algorithm for time series classification». In: *Data Mining and Knowledge Discovery 34.3* (2020) (cit. on p. 22).

[63] Ismail Fawaz and Hassan. «Inceptiontime: Finding alexnet for time series classification». In: *Data Mining and Knowledge Discovery 34.6* (2020) (cit. on p. 22).

[64] Thompson IM, Ankerst DP, Chi C, Goodman PJ, Tangen CM, Lucia MS, and et al. «Assessing prostate cancer risk: results from the Prostate Cancer Prevention Trial». In: *J Natl Cancer Instl* (2006) (cit. on p. 24).

[65] Vickers AJ, Till C, Tangen CM, Lilja H, and Thompson IM. «An empirical evaluation of guidelines on prostate-specific antigen velocity in prostate cancer detection». In: *J Natl Cancer Instl* (2011) (cit. on p. 24).

[66] O'Brien MF, Cronin AM, Fearn PA, Smith B, Stasi J, and Guillonneau B et al. «Pretreatment prostate-specific antigen (PSA) velocity and doubling time are associated with outcome but neither improves prediction of outcome beyond pretreatment PSA alone in patients treated with radical prostatectomy». In: *Journal of Clinical Oncology* (2009) (cit. on pp. 24, 25).

[67] Mihaylova and Borislava. «Review of statistical methods for analysing healthcare resources and costs». In: *Health economics 20.8* (2011) (cit. on p. 25).

[68] Nitta S, Tsutsumi M, Sakka S, Endo T, Hashimoto K, Hasegawa M, Hayashi T, Kawai K, and Nishiyama H. «Machine learning methods can more efficiently predict prostate cancer compared with prostate-specific antigen density and prostate-specific antigen velocity». In: *Prostate Int.* (2019) (cit. on pp. 25–27, 37).

[69] Cosma G, McArdle SE, Foulds GA, Hood SP, Reeder S, Johnson C, Khan MA, and Pockley AG. «Prostate Cancer: Early Detection and Assessing Clinical Risk Using Deep Machine Learning of High Dimensional Peripheral Blood Flow Cytometric Phenotyping Data». In: *Front. Immunol.* (2021) (cit. on pp. 26, 27).

[70] Cosma G, Mcardle SE, Reeder S, Foulds GA, Hood S, and Khan M. «Identifying the Presence of Prostate Cancer in Individuals With Psa Levels <20 Ng.Ml–1 Using Computational Data Extraction Analysis of High Dimensional Peripheral Blood Flow Cytometric Phenotyping Data». In: *Front. Immunol.* (2017) (cit. on p. 27).

[71] URL: `https://www.ncbi.nlm.nih.gov/books/NBK487255/` (cit. on p. 30).

[72] URL: `https://www.cancer.org/cancer/prostate-cancer/detection-diagnosis-staging/risk-groups.html` (cit. on p. 30).

[73] URL: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4508854/` (cit. on p. 30).

[74] URL: `https://scikit-learn.org/stable/modules/tree.html` (cit. on p. 38).

[75] URL: `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html` (cit. on p. 38).

[76] URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html (cit. on p. 38).

[77] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. «On the Properties of Neural Machine Translation: Encoder-Decoder Approaches». In: *arXiv* (2014) (cit. on p. 43).

[78] Althloothi and Salah. «Human activity recognition using multi-features and multiple kernel learning». In: *Pattern recognition 47.5* (2014) (cit. on p. 45).

[79] Smith and Leslie. «A disciplined approach to neural network hyper-parameters¦ Part 1–learning rate, batch size, momentum, and weight decay». In: *arXiv* (2018) (cit. on pp. 47, 48).

[80] Gyoung S.Na. «Efficient learning rate adaptation based on hierarchical optimization approach». In: *ELSEVIER* (2022) (cit. on p. 47).

[81] Johannes Lederer. «Activation Functions in Artificial Neural Networks: A Systematic Overview». In: *arXiv* (2021) (cit. on p. 48).

[82] Gangi Siva, Nandini, and Siva Kumar Chidananda K. «Dropout technique for image classification based on extreme learning machine». In: *Global Transitions Proceedings* (2021) (cit. on p. 48).

[83] Saahil Afaq and Dr. Smitha Rao. «Significance Of Epochs On Training A Neural Network». In: *INTERNATIONAL JOURNAL OF SCIENTIFIC TECHNOLOGY RESEARCH VOLUME* (2020) (cit. on p. 48).

[84] URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (cit. on p. 48).

[85] URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html (cit. on p. 48).

[86] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. «Optuna: A Next-generation Hyperparameter Optimization Framework». In: *arXiv* (2019) (cit. on p. 48).

[87] URL: https://optuna.org/ (cit. on pp. 48, 53).

[88] Tyler Sypherd, Mario Diaz, Lalitha Sankar, and Peter Kairouz. «A Tunable Loss Function for Binary Classification». In: *arXiv* (2019) (cit. on p. 49).

[89] Paula Branco, Luis Torgo, and Rita Ribeiro. «A Survey of Predictive Modelling under Imbalanced Distributions». In: *arXiv* (2015) (cit. on p. 49).

[90] Cyril Goutte and Eric Gaussier. «A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation». In: *Advances in Information Retrieval* (2005) (cit. on p. 50).

[91] Andrew P.Bradley. «The use of the area under the ROC curve in the evaluation of machine learning algorithms». In: *Pattern Recognition* (1997) (cit. on p. 51).

[92] McClish and Donna Katzman. «Analyzing a portion of the ROC curve». In: *Medical decision making 9.3* (1989) (cit. on p. 51).

[93] URL: `https://www.uio.no/english/services/it/research/sensitive-data/` (cit. on p. 52).

[94] URL: `https://www.uio.no/english/services/it/research/sensitive-data/help/hpc/` (cit. on p. 52).

[95] URL: `https://pandas.pydata.org/` (cit. on p. 52).

[96] URL: `https://numpy.org/` (cit. on p. 52).

[97] URL: `https://matplotlib.org/` (cit. on p. 53).

[98] URL: `https://www.pytorchlightning.ai/` (cit. on p. 53).

[99] Xu, Yun, and Royston Goodacre. «On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning». In: *Journal of analysis and testing 2.3* (2018) (cit. on p. 53).

[100] URL: `https://pytorch-lightning.readthedocs.io/en/stable/common/early_stopping.html` (cit. on p. 54).

[101] URL: `https://pytorch-lightning.readthedocs.io/en/stable/api/pytorch_lightning.callbacks.LearningRateMonitor.html` (cit. on p. 54).

[102] URL: `https://pytorch-lightning.readthedocs.io/en/stable/common/optimization.html` (cit. on p. 54).

[103] URL: `https://pytorch.org/docs/stable/notes/cuda.html` (cit. on p. 54).

[104] Yu, Kun-Hsing, Andrew L. Beam, and Isaac S. Kohane. «Artificial intelligence in healthcare». In: *Nature biomedical engineering 2.10* (2018) (cit. on pp. 66, 70, 71).

[105] Alvin Rajkomar, Jeffrey Dean, and Isaac Kohane. «Machine Learning in Medicine». In: *The new england journal of medicine* (2019) (cit. on p. 70).

[106] URL: `https://senseaboutscience.org/activities/guides/` (cit. on p. 71).

[107] Zhang, Quan-shi, and Song-Chun Zhu. «Visual interpretability for deep learning: a survey». In: *Frontiers of Information Technology Electronic Engineering 19.1* (2018) (cit. on pp. 71, 72).

[108] Arrieta and Alejandro Barredo. «Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI». In: *Information fusion 58* (2020) (cit. on pp. 71, 72).

[109] Peloquin and David. «Disruptive and avoidable: GDPR challenges to secondary research uses of data». In: *European Journal of Human Genetics 28.6* (2020) (cit. on p. 72).