Universitatea Transilvania din Brașov

Universitatea Transilvania din Brașov
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

**INTERDISCIPLINARY DOCTORAL SCHOOL**

**Faculty of Mathematics and Computer Science**

**Árpád KERESTÉLY**

# Machine Learning Models in Cancer Prediction

SUMMARY

Scientific supervisor

Prof. Dr. Marius-Sabin TĂBÎRCĂ

BRAȘOV, 2022

## Topic of the thesis and the fields in which it integrates

The thesis presents the research and result of an interdisciplinary topic, the usage of computer science in healthcare, namely the topic of cancer prediction using Machine Learning, thus it touches fields from both. From the computer science perspective, which is the main focus of the thesis, the field of Machine Learning is the centerpiece, which brings with itself many of the closely connected fields such as Data Science, Data Mining, High Performance Computing or Big Data. From a healthcare perspective, the centerpiece is the field of cancer prediction, with a focus on molar pregnancy and breast cancer, but the parent field of healthcare is also reviewed in the process. Thus, the topic of the thesis integrates well in fields from both computer science and healthcare.

Zooming in on the treated problems and solutions from a computer science perspective, it is worth mentioning that the used datasets were tabular and problems like classification and timeseries computation were addressed, thus algorithms like Logistic Regression Classifier, Random Forest, Artificial Neural Networks, Recurrent Neural Networks, or the Least Squares Method were the main focus points in finding solutions. Figure 1 shows an example of measurement in the context of molar pregnancy, for which the problem is in finding a regression curve that best matches the data using only a subsample of the first data points.

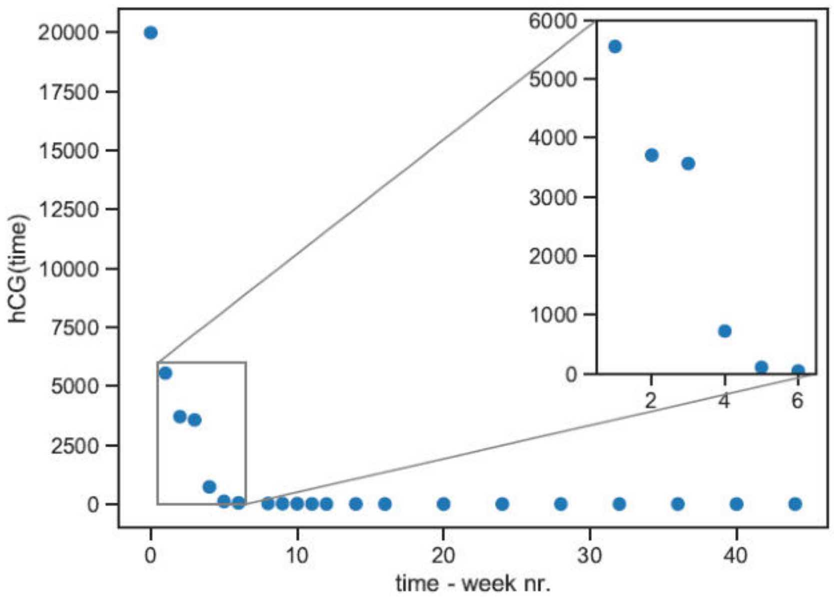


Figure 1: A sample of hCG measurement in molar pregnancy.

## Purpose of the thesis

The research in the context of this thesis was started by the need of a precise and automated solution for the problem of determining the devolution of cancer in patients diagnosed with and treated of molar pregnancy. An example of such a problem and possible solution can be seen in Figure 2, where the first five weeks' hCG measurements (blue data points) of patient were used to forecast the devolution of the cancer (represented by the blue curve), and as it can be seen, the curve matches the real data (orange data points) very closely. Thus, the main objective of the thesis was to find solutions to the aforementioned problem. As secondary objectives, the thesis aimed to find and cover niches in the computer science literature that were not covered yet, as well as to gather a set of tools and processes that can be used by researchers when working with the prediction of cancer but also with other types of problems that involve classification or time series computation using tabular data.

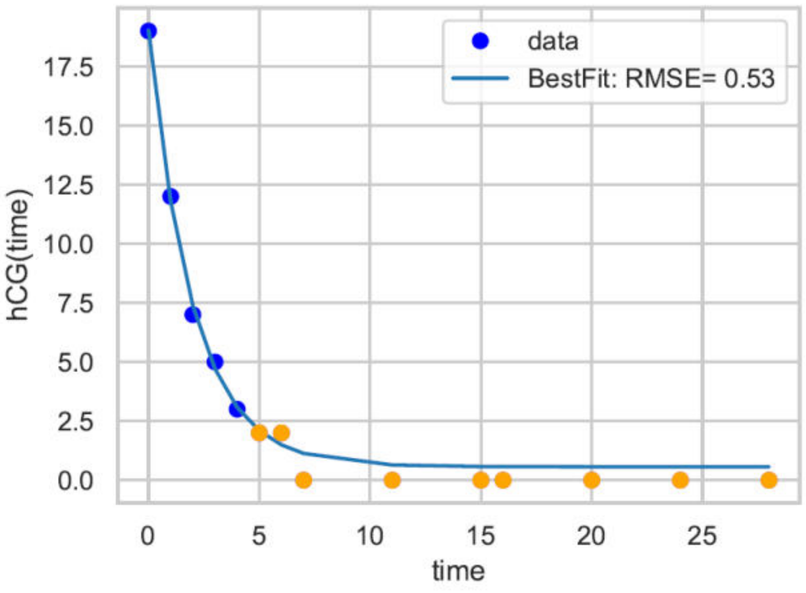


Figure 2: Forecasting the devolution of cancer using the first five data points.

## Structure of the thesis

The thesis consists of six chapters.

The first chapter is the "Introduction" which contains a detailed description of: the motivations behind starting this thesis, the objectives, the research outcome together with the papers published in the scope of the research, and the structure of the thesis.

The second chapter is entitled "Machine Learning in Healthcare and Cancer Prediction" and it reviews state-of-the-art machine learning models and techniques used in various problems from the healthcare and two specific cancer prediction domains, namely the breast cancer and the molar pregnancy domains.

The third chapter, called "Data, Data Preprocessing and Feature Handling", presents the difficulties in gaining access to medical data, but also presents different sources where public cancer data can be found, what are the typical data exploration workflows, what methods are used to treat or compute missing data, tricks for enhancing a dataset with additional features as well as removing unneeded features to reduce data complexity, all in the context of first the breast cancer and then in the molar pregnancy prediction. To support the theories, experimental results are also presented on real-world data considering both types of cancer.

The main contribution of this research is in "Time Series and Forecasts in Cancer Prediction", the fourth chapter of the thesis, where a substantial number of results are presented on the molar pregnancy dataset. A new method is presented for fitting a curve to data points which decrease in an exponential rate, it is compared and evaluated against other existing methods, and applications of the mentioned curve fitting method are presented, like forecasting or anomaly detection. In the second part of this chapter, Recurrent Neural Networks (RNNs) are used to predict a patient's disease devolution from other patients' data together with a set of tricks that can help RNNs perform better when data is scarce.

Chapter five, "High Performance Computing in context of Big Data and Machine Learning", reviews the existing HPC methods available for ML and Big Data, while in the second part, two popular ML frameworks are compared against each other, and their performances assessed. This chapter aims to forecast the time and resource complexity required for running a classification on a bigger breast cancer dataset, but since at the writing of this thesis breast cancer data was not available in big sizes, a fallback solution was adopted, in which the algorithms were run on a dataset with similar format but bigger size, from the domain of bearing fault detection.

The last chapter of the thesis, "Conclusions", sums up the contributions that this thesis made to the research community, and presents the future research directions.

## Research methodology

Existing research for the main topic of the thesis, namely machine learning in predicting the disease devolution in patients treated of molar pregnancy, is quite scarce, so the research for the thesis started with first reviewing the state-of-the-art regarding the broader healthcare domain, and a well-known type of cancer for women, breast cancer. The idea was to find problems and solutions that are similar in nature to the one regarding molar pregnancy. Chapter 2 describes all the findings from all three domains and collects a handful of tools and processes that can be usually applied in the cancer prediction domain.

Chapter 3 continues with the study of both breast cancer and molar pregnancy, but this time, by applying the previously gathered knowledge on concrete datasets. First, new contributions are made to a publicly available breast cancer dataset, by doing an in-depth exploratory analysis, applying manual and automatic feature selections, and providing classification scores as a result of running the Logistic Regression and Random Forest classifiers. Next, contributions are made to a private dataset regarding molar pregnancy. As in the previous case, an in-depth exploratory analysis was done, as well as manual feature selection. In addition, methods for treating missing values were discussed, and needed transformations were proposed in order to turn the dataset into one that is ready for sequence learning. These experiments enabled focusing exclusively on the machine learning part in the coming chapters.

Chapter 4 is a continuation of Chapter 3's section regarding molar pregnancy. In this chapter the main focus is finding solutions for predicting the devolution of cancer in patients treated for molar pregnancy, thus the main topic of the thesis. First, this is done by using the domain knowledge, that the hCG measurements in patients treated of molar pregnancy decreases based on a vertically shifted exponential curve. This curve is unique for each patient, so the solutions involved using the first three measurements of a patient in determining the parameters for the curve. The resulting curve allowed forecasting the devolution of the disease. Next, a Recurrent Neural Network approach was investigated, where the aim was to learn the decreasing nature of the measurements using data from all patients. This allowed avoiding the incorporation of domain knowledge in the solution, thus enabling the learning and prediction of cases where the disease recurred.

Chapter 5 is a perpetuation of Chapter 3's section regarding breast cancer. The idea behind this chapter was that at some point, massive amounts of data will be available for research, and especially in the breast cancer domain, as it is one of the most studied ones. Preparing for this, it was obvious that the framework used in Chapter 3 will not be able to handle the large volume of data, so a change was required to a more scalable, high performance computing solution. With this idea in mind, a review was performed of the existing high performance computing solutions that involve machine learning. Spark was chosen as the solution, as it can handle tabular data by design, and it's API is very similar to scikit-learn's API, the framework used in Chapter 3 for the study of the breast cancer dataset. Thus, the second part of the chapter deals with the comparative analysis of these two frameworks, in order to first assess the performance of Spark in relation to the performance of a known framework, and second, to prepare the ground for when breast cancer data will be available in huge quantities. To be able to perform the comparison, a big, tabular dataset was used from the bearing fault detection as a replacement, which is very similar in structure to the breast cancer dataset and is labeled for classification problems as well.

# Original results, conclusions, contribution to the field and relevance

## *Chapter 2: Machine Learning in Healthcare and Cancer Prediction*

### 2.1 Introduction

This chapter aims to present, analyze, and discuss some of the latest advancements in machine learning from the healthcare and cancer prediction point of view. Important aspects that this chapter covers are recently used machine learning algorithms, data available for research purposes and the fields that healthcare and cancer prediction extend to. A conclusion based on these aspects is drawn, whether there is a need and possibility for potential further development of machine learning algorithms in healthcare and cancer prediction.

The methodology applied in making the review is comprised of well-known methods. First, search strings were composed with the following keywords: healthcare, cancer prevention and prediction, breast cancer, molar pregnancy, gestational trophoblastic disease, machine learning, data mining, time series, forecast. Next, the search string was used in the following databases to find relevant studies: Google Scholar, Elsevier Scopus, Elsevier Science Direct and Web of Science. These databases were chosen due to the number of disciplines in which they are indexed. The list of research studies gathered this way, was filtered, in order to get the most relevant ones from a data science and machine learning point of view. First these were filtered by title. If the title seemed good enough or if it was at least ambiguous, the abstract of that research was inspected. The papers that were found to be relevant after checking the abstract were taken through another round of filtering based on introduction and conclusions. Finally, those research papers that still seemed relevant, were read from beginning to end. Those that presented relevant, interesting, innovative, and diverse ideas, were reported in the following review.

## 2.2 Healthcare

It is interesting to see that researchers from all around the world are interested in the topic of machine learning related to healthcare. The studied papers are written in different parts of the world, yet they share the same goal, of bringing improvements to healthcare. For example: [82] was done in Brazil, [26] in California, [23] in China. The aforementioned papers also implemented their research on the data available locally in their region (it's important to note that some diseases can manifest in different ways depending also on where they were observed [23]).

For most of the machine learning algorithms it is a necessity to have training data to work on. Typically, the larger the data, the more accurate the solution will be, as stated by many research papers including those from the cancer prediction domain [64]. The data can come from various sources: mobile data, medical records, social network, internet usage, genomic data or environmental data.

It is certain that more and better results and solutions in healthcare are needed. Data amount has increased recently and it will probably increase even further in the future. Machine learning algorithms have been researched to solve some specific problems, but there is still plenty of room for improvement, as very few of them address problems with high accuracy and even fewer in a generic way. None of the machine learning algorithms used are suitable for more than a few problems, furthermore as seen in the mentioned studies, none of them appeared more than twice, which is a sign that probably better algorithms are still undiscovered. Considering these facts, the future of healthcare looks promising with the aid of machine learning.

## 2.3 Breast cancer

Breast cancer is one of the world's most notorious cancer types for women (see Figure 3, Figure 2.1c from thesis), and together with the rest of the cancer types it forms the world's second most lethal disease for humans [89]. Much research effort has been put into predicting and even preventing it. While the field of cancer prediction greatly benefited from machine learning in the past decades as many research papers addressed the problem of cancer prediction, there is still space for improvement. Because this thesis focuses on cancer prediction, the study of breast cancer can provide valuable insight and starting points for developing solutions for the problem mentioned in the introduction. This section aims to review some of the existent research from the field of breast cancer, gather a set of useful methods and processes, and apply them in Chapter 3 on a publicly available breast cancer dataset, with a focus on understanding how different features, or their absence, can influence the outcome of a prediction.

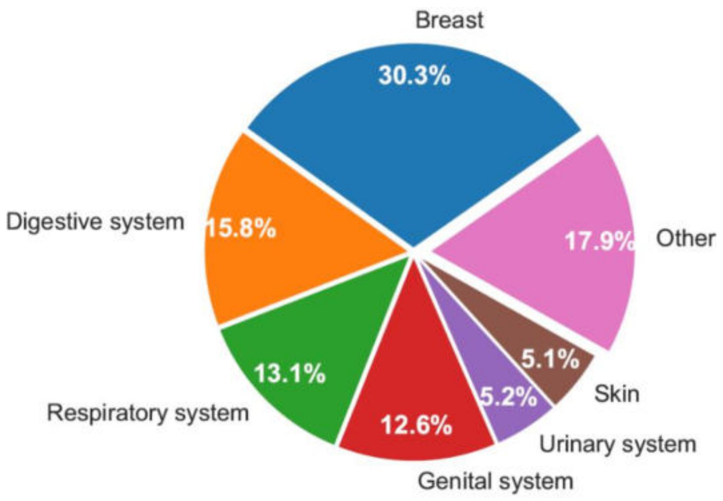


Figure 3: Estimated new cancer cases for women in USA by type, based on data from [89]

Breast cancer is one of the most invasive and dangerous cancers among women as it can be seen in [89, 47], thus it draws the attention of many researchers. In the past few decades, it has seen improvements because of the newer machine learning algorithms. Still there is room for improving even more. As machine learning requires data to learn from, the speed at which this improvement progresses is strongly dependent on the amount of publicly available datasets. Unluckily, and mostly because of the recent data privacy concerns, not much data is publicly available, and even less of it is new, so a researcher either uses what is publicly available, or teams up with a medical institute. In the case of this thesis, the focus will be on research that is done on publicly available data, so that the reported results can be verified and, in a case-by-case scenario, improved.

A recent study of breast cancer by the authors of [4] reports that on the Wisconsin Diagnostic Breast Cancer (WDBC) they achieved nearly perfect accuracy with the K-Nearest Neighbors, Random Forest, and Multilayer Perceptron models. Besides accuracy, they also measured precision, recall, F-measure, and ROC Area. While not mentioned by the authors, after trying to reproduce the results, it was discovered that in their experiments, the validation of the results was most probably done on the same dataset as the training, hence the very high accuracy.

In [3] the author studied multiple ML algorithms: a combination of Gated Recurrent Unit and Support Vector Machine, Linear Regression Classifier, Multilayer Perceptron, K-Nearest Neighbors search, Softmax Regression, and Support Vector Machine on the same WDBC dataset. Prior to running the ML algorithms, a preprocessing step to standardize the dataset, namely the StandardScaler from scikit-learn [79] was applied. The initial dataset was partitioned in 70% training and 30% testing sets, and every tested algorithm performed with over 90% accuracy on the test set, with MLP reaching an accuracy of 99.04%.

The authors of [72] took a somewhat different approach than the already mentioned papers. They used Topological Data Analysis on the Netherlands Cancer Institute (NKI) Breast Cancer dataset, to gather information from more than 1500 gene expression features. Their research provided interesting insights about the data by visually inspecting the resulting graphs. They reported finding some of the most important features that could determine the survival chances of a patient, one of which they reported to be the ESR1 gene.

Considering the reviewed research papers, the following aspects should be considered when engaging in a cancer prediction problem. Often, we want to know the malign and benign cases, or recurrence and no recurrence, so generally speaking, most of the time we will have a binary classification. The accuracy should only be used as a metric if the dataset is balanced. Furthermore, the accuracy can mean two things: the patient is either detected as having cancer so a treatment can be administered as soon as possible, or the patient is free of cancer, meaning that no further investigations or administration of treatment is necessary. If the dataset is unbalanced as it is usually in cancer prediction datasets, the F1 score, precision and recall should be used instead. Most often it is best to tune the algorithm to fully avoid false negatives, because if not done so, it can lead the patient to death. On the other hand, false positives should be minimized in order to reduce the costs of medical investigations and treatment. In cancer prediction data is often noisy, because of records that were badly input so these need to be detected and sometimes even manually removed. Finally, algorithms which cannot be easily interpreted or explained to a doctor, could be rejected, so, if possible, simpler, white-boxed algorithms should be chosen.

## 2.4 Molar Pregnancy

The main problem addressed in this thesis is from the domain of molar pregnancy, more precisely the recurrence of this type of cancer (abnormal cell growth) in treated patients, the forecasting of a patient's recovery and the detection of abnormalities in a patient's healing period. This section will describe molar pregnancy in general, present the related studies that use machine learning as their solution and describe the particular problems that this thesis seeks solutions for.

Gestational trophoblastic disease (GTD) is a term used to describe a range of rare illnesses in which abnormal trophoblast cells grow inside the uterus after conception [90, 5, 96]. The most common type of GTD is molar pregnancy, also known as hydatidiform mole [69]. GTD causes the hormone human chorionic gonadotropin (hCG) to be very high, as the trophoblastic cells produce hCG. Multiple elevated hCG measurements strongly suggested the presence of a complete hydatidiform mole [17].

Several research studies use machine learning for detecting and categorizing the hydatidiform mole. The authors of [76] use classical image processing and segmentation with the guidance and knowledge of expert pathologists to analyze images of hydatidiform moles. Later, the same authors use a multineural network to analyze the images and recognize the patterns of either the partial hydatidiform moles or those of the complete hydatidiform moles in [77]. They claim the performance of their approach is better than those of most human experts.

With the removal of the hydatidiform mole it has been shown that the hCG hormone levels drop exponentially in women diagnosed with GTD [81, 86, 97, 105]. In most cases the hCG levels will drop to normal with no further treatment but it has been shown that in the U.K. 15% of women with a complete hydatidiform mole will require chemotherapy [88].

After the removal of the hydatidiform mole, there are risks that a patient will redevelop GTD. Thus, continued surveillance is mandatory although the risks of developing GTD are slightly less, 5%, for patients who previously had partial hydatidiform mole compared to 20% to 25% for those who had complete hydatidiform mole [69]. In summary a patient is generally monitored for one year although redevelopment of GTD usually occurs in the first six months, out of which the hCG levels may take up to 24 weeks to become undetectable.

The existing research on GTD and molar cancer is scarce. Although it's true that it is a rare form of cancer, it's one that affects many patients. The literature's focus is mainly on detecting mole types from image data and very few studies deal with the post treatment phase of the disease, meaning that contributions are needed and welcome.

## 2.5 Conclusions

This chapter gradually covered the fields of healthcare and cancer prediction where machine learning was used as a response to certain issues. Research was gathered regarding two types of cancer: breast cancer and molar pregnancy (hydatidiform moles). With both being cancers specific to woman, the first being one of the most notorious ones with lots of research behind it and the second one being a less frequent, less researched type of cancer, the literature review was focused on finding general information that could aid in solving problems regarding the second one, namely the molar cancer.

It was found that generally machine learning algorithms are chosen based on:

- data type i.e., structured (tabular) and unstructured (texts, images, videos)
- problem type i.e., classification, regression, clustering, forecasting or even defensive mechanism for adversarial attacks

Moreover, research showed that although based on the trend, more and more data should be available for researchers, this is not always the case, and for those who deal with small datasets, tricks like k-fold cross-validation, leave-one-out or data enhancing exist.

Research also showed that for historical data, like in the problem of molar pregnancy recurrence, Recurrent Neural Networks can be used, as in certain cases they can outperform even medical experts at forecasting. Thus, proving to be a reliable ally of the medical staff, in predicting and preventing future diseases.

In support of this chapter the following research articles were published:

- *Machine Learning in Healthcare: An Overview*, by Árpád Kerestély, Lucian Mircea Sasu and Marius-Sabin Tăbîrcă, in Bulletin of the Transilvania University of Brasov, 2018. This article is the basis for Section 2.2, and it covers recently used machine learning algorithms in healthcare, data available for research purpose and the fields that healthcare extends to.
- *Feature Inspection and Elimination in the Context of Breast Cancer Prediction*, by Árpád Kerestély, in Proceedings of the 36th International Business Information Management Association (IBIMA), 2020. This paper aims to review some of the existent research from the field of breast cancer, thus being the basis for Section 2.3, gather a set of useful methods and processes, and finally apply them on a publicly available breast cancer dataset, with a focus on understanding how different features, or their absence, can influence the outcome of a prediction.

# *Chapter 3: Data, Data Preprocessing and Feature Handling*

Data is an important part of every machine learning research [58]. One has a problem in the absence of data but has more problems in the presence of it. It's hard to do research on an unknown dataset, or on one that was not fully understood. Therefore, it's important to know a few data preprocessing techniques, which will aid in the exploratory analysis of the data thus reducing the number of potential future problems. This chapter aims to show exploratory analysis, data visualization, feature selection and engineering, as well as some frequent issues that a researcher might encounter, on two concrete datasets: on the publicly available NKI Breast Cancer Data and on a private Molar Pregnancy Data.

## 3.1 Challenges in gathering clinical data

Access to clinical data, which is the main subject of every medical research, has been greatly reduced by the recent EU's General Data Protection Regulation (GDPR), which entered into force on 25 May 2018. Although the EU had a previous legal framework which dates to 1995, this new regulation, while retaining the overall regulatory approach of its predecessor, also introduces a number of new compliance obligations and together with that, higher sanctions. This encourages medical institutions to be less open to sharing clinical data with researchers, not to mention that it entirely restricts them from publishing data to the public space, thus it's an area of significant concern for those engaged in clinical research.

EU-GDPR clearly states that processing of personal (i.e., identifiable, not anonymized) genetic, biometric or health data is prohibited [30]. Yet, Article 89 of the EU-GDPR states that the processing is allowed for archiving purposes in the public interest, scientific or historical research purposes, or statistical purposes if appropriate safeguards are taken, like data minimization, pseudonymization, and anonymization where possible. This exemption allows member states the freedom to legislate at national level in certain areas, one of these being the processing of personal data for scientific and research purposes.

With these regulations in place, although data is generated in enormous quantities by each medical institution, it's becoming a precious resource for the researchers. Looking at the whole picture, a researcher can have access either to anonymized public data possibly collected and published before GDPR, or by applying for an ethical approval which is a long and time-consuming process that often ends in rejection. The rest of the chapter will present a dataset for the study of breast cancer, that can be obtained from public sources, and a dataset for the study of molar pregnancy, that was obtained from a medical institution located in Ireland before the GDPR came into force. Both datasets are relatively small, reinforcing the scarcity and preciousness of clinical data. In the case of the molar pregnancy dataset, initiatives were made to gain access to more data, but because of the regulation and the need for ethical approval, they failed.

## 3.2 Breast Cancer

This section aims to experiment on and discover interesting knowledge in a publicly available breast cancer dataset, by using the knowledge gathered from Chapter 2. This knowledge mainly consists of exploratory analysis, data visualization, clustering, preprocessing, feature selection, partitioning, classification, evaluation of learning models and cross validation.

Guided by the studied papers and from the careful searching on various public dataset repositories [20, 35, 75, 93] it was discovered that there are few publicly available datasets about breast cancer. Among these, the Wisconsin Diagnostic Breast Cancer (WDBC) dataset is the most commonly used, providing 32 features of 569 patients. It's a reasonably small dataset published in 1995, that has been the subject of so many research studies that it's hard to bring any new contributions to it. Unfortunately, some of the other datasets are not too large either and some also have a large number of missing values, which is why they got down on the priority list.

One particular dataset, compiled by the Netherlands Cancer Institute (NKI), while containing only 272 breast cancer patient records, has an impressive amount of 1570 features. This dataset is relatively new in its current form, although traces suggest that it is more or less the same dataset that was studied by [98]. The dataset was not fully explored by the literature, so it's a good candidate for making new contributions. In addition, the fact it has a great number of features, makes it a good candidate for all the data preprocessing and feature selection algorithms.

The Netherlands Cancer Institute (NKI) Breast Cancer Data was downloaded from [84]. It contains demographic and clinical data about the patients and gene expression (genomic) data from breast tumors, that add up to 1570 features. The "eventdeath" feature gives information on which cancer cases were fatal. The class distribution is unbalanced, 77 (28.3%) "deceased" cases and 195 (71.7%) "survived" cases (Figure 3.1 from the thesis). As seen from Chapter 2 the class distribution is an important aspect, as it will influence the choice of the scoring metric later on.

As a starting point, the features were manually inspected and it turned out that the first 16 represent clinical data, while the rest of the 1554 features represent genomic data. Out of these 16, three features were manually removed, namely "Patient", "ID" and "barcode", which clearly represented irrelevant information for a learning algorithm. The "eventdeath" feature was also extracted as the dependent variable. The dataset's features were numerically analyzed, and a histogram of the 12 remaining clinical features was done (Figure 3.2 from the thesis). No missing values were found.

On a dataset containing the remaining 13 clinical data features, a check was done to see how the 12 independent variables correlate to each other and to the 1 dependent variable. The Pearson correlation score was computed between pairs of features, and the results were color coded to be able to better visualize which two features correlate the most, as it can be seen in Figure 4 (Figure 3.3 from thesis). It was discovered that "eventdeath" is strongly correlated to the "survival" and "timerecurrence" features, as well as that "survival" and "timerecurrence" strongly correlate to each other.

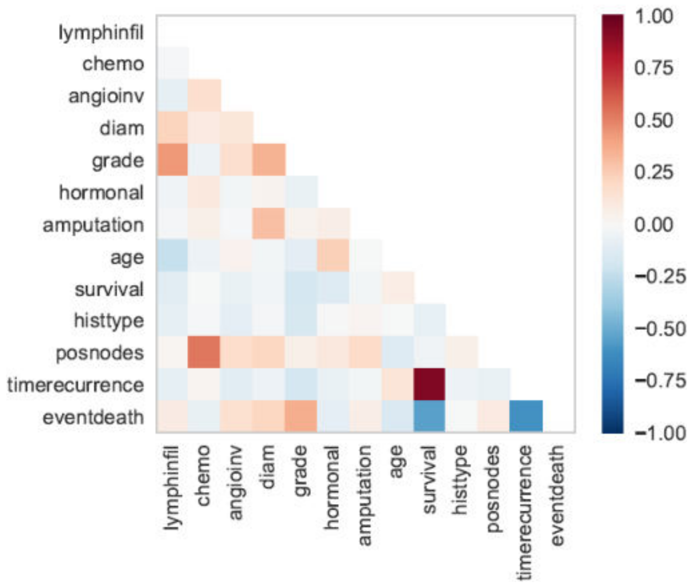


Figure 4: Pearson correlation between 13 features including the dependent variable.

The class separability was inspected using the 12 independent clinical variables, with the RadViz visualization method. Results can be seen in Figure 3.4 from the thesis, from which it could be concluded, that while there is a certain separation between the two classes, it is not a very clear one.

Preliminary classifications done on the dataset containing at first just the 12 clinical features then containing all the 1566 features showed that the classification score is strongly influenced by the selected features, as seen in the first two rows of Table 1 (Table 3.1 from the thesis). After a close review of the results, the conclusion was that an automatic feature selection should be run on the dataset. Running the Recursive Feature Elimination with Cross-Validation (RFECV) using the Logistic Regression classifier showed that the best scores can be achieved using only 32 features when using accuracy as a scorer, and 71 features when using the F1 score as a scorer. The time required for fitting the classifier was also improved. The concreate features can be found in Table 3.2 and 3.3 from the thesis. The evolution of the scores, removing one feature at a time, can be seen in Figure 5 (Figure 3.7b from the thesis).

Table 1: Summary of running the Logistic Regression classifier with different features.

| | Accuracy (%) | | | F1 score | | |
|---|---|---|---|---|---|---|
| | score | fit_time* | score_time* | score | fit_time* | score_time* |
| Clinical features (12) | $88.20 \pm 4.35$ | $0.181 \pm 0.036$ | $0.008 \pm 0.002$ | $0.78 \pm 0.10$ | $0.183 \pm 0.030$ | $0.011 \pm 0.002$ |
| All features (1566) | $84.15 \pm 7.46$ | $1.424 \pm 0.419$ | $0.075 \pm 0.030$ | $0.68 \pm 0.18$ | $1.530 \pm 0.457$ | $0.070 \pm 0.020$ |
| Best features (32/71) | $95.60 \pm 4.30$ | $0.053 \pm 0.017$ | $0.010 \pm 0.005$ | $0.96 \pm 0.03$ | $0.057 \pm 0.014$ | $0.015 \pm 0.009$ |

(*) Tested on a laptop having an Intel i7-4720HQ CPU, with 12 GB RAM

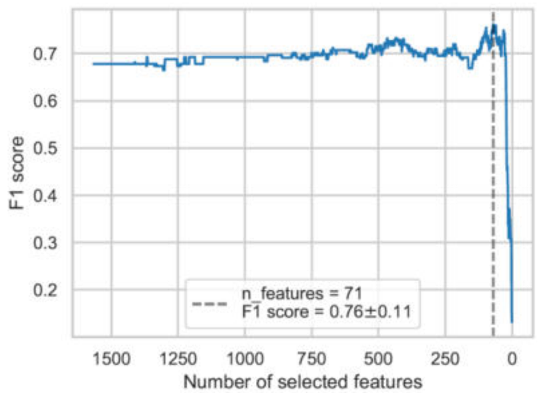


Figure 5: Logistic Regression classifier F1 scores with regards to the number of features.

To reinforce the usefulness of the automatic feature selection on this dataset, the tests were repeated using a different classifier, the Random Forest classifier. As seen in Figure 6 (Figure 3.8b from the thesis), the evolution of the scores is even more visible for the Random Forest classifier. The comparison of results from before the feature selection and after the feature selection can be seen in Table 3.4 from the thesis, and the selected features can be seen in Table 3.5 from the thesis. This time 23 features were selected when using the accuracy as a scorer and 9 features were selected when using F1 score as a scorer. The mean accuracy improved from 79.03% to 91.19%, while the mean F1 score from 0.48 to 0.84.

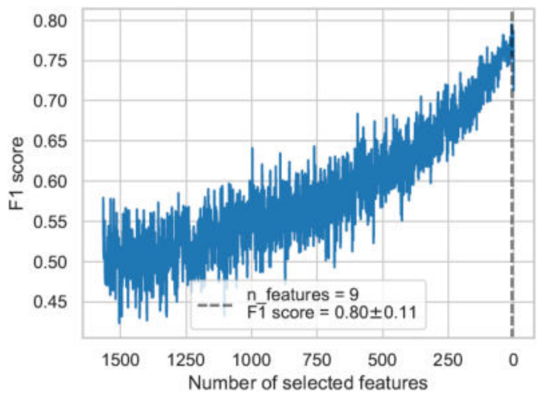


Figure 6: Random Forest classifier F1 scores with regards to the number of features.

### 3.3 Molar Pregnancy

This section covers the dataset used for the main problem addressed by this thesis, that of molar pregnancy. Knowledge is extracted from the features that exist in the dataset, as well as the relations between them. Feature engineering and missing data handling techniques are applied to prepare the dataset for future learning algorithms. Throughout the section, data visualization is used as often as possible, to aid in perceiving the dataset in depth, as this knowledge is essential for understanding the algorithms presented in Chapter 4.

The dataset studied in this section is a private one, from Cork University Maternity Hospital, where Professor John Coulter collected data between 2008 and 2013 about patients who had molar pregnancy (hydatidiform mole). The dataset contains post treatment measurements, retained in a table like format. After treatment, the patients who had molar pregnancy undergo a period of surveillance throughout which their hCG levels are measured periodically. These measurements are recorded in the aforementioned table, along with other information like the age of the patients or the type of hydatidiform moles the patients developed.

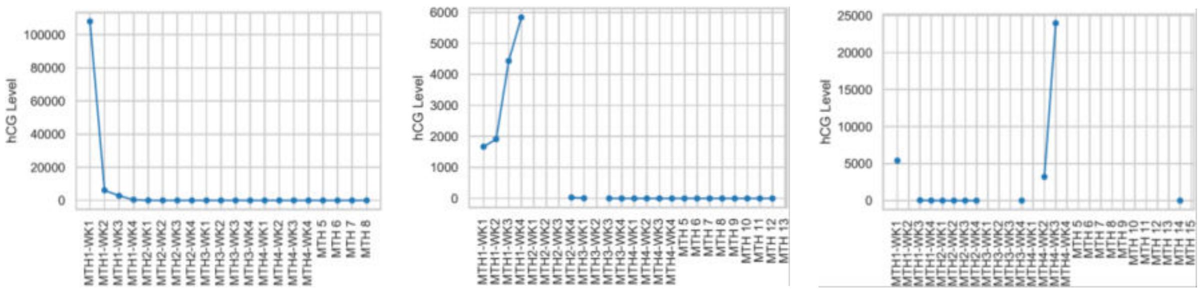
The dataset has the following features: MRN - medical record number, I - patient identifier, DX - diagnostic, which can be used to determine the mole type, AGE - age of the patient at the time of diagnosis, DDX DATE - date on which the diagnosis was recorded. The next set of features are the hCG measurement, one feature per week for a total of 4 months (4 weeks per month), then one feature per month until a total of 2 years (24 months). In total 41 features, out of which 5 represent clinical data, while the rest of 36 features represent hCG measurements, out of which 16 weekly and 20 monthly. It's interesting to see that the hCG sampling in this dataset correlates strongly with the sampling suggested by the literature regarding post molar cancer surveillance. The other dimension of the dataset is 57, meaning that it contains data about 57 patients.

Following the analysis of the dataset, several remarks can be made:

- From the diagnostic dates feature, it can be determined that this dataset contains patient data from 2008 to 2013, with most patients being diagnosed with molar cancer in 2009 and 2010.
- Looking at the age statistics it can be concluded that molar pregnancy can happen at any age, yet in this dataset most patients were: 39, 31 or 32 years old when diagnosed with molar pregnancy. See details in Figure 3.10 in the thesis.
- From the hydatidiform mole type point of view, the DX feature, it can be said that the dataset is mostly balanced. It could also be said that the chances of developing either a partial mole (PM) or a complete mole (CM) are nearly the same, 54.4% and 45.6% respectively. See details in Figure 3.11 in thesis.
- Using the age feature, after filtering the patients by the type of moles they developed, it can be concluded that both mole types occur at mostly any age. See details in Figure 3.12 in thesis.
- The hCG levels are sampled either weekly or monthly, which certain learning algorithms won't be able to handle in their current form.
- The hCG measurements have many missing values, probably because either the patient did not show up for the test, or the doctor decided that the patient can skip the test at a certain time.
- On average, each patient has 13 measurements, yet the distribution is not normal, to be more precise, 13 patients have less than 5 measurements, while 4 patients have more than or equal to 27 measurements. Making a split in the middle, it can be said

that 24 patients have more than 15 measurements while the rest of 33 patients have less than 15. The complete view can be seen from Figure 3.13 in the thesis.

- All patients took the first test, at month 1 - week 1, although this test can easily represent the first test after the mole was removed. Next, a steady drop in test can be observed, in Figure 3.14 from the thesis, until months 5-6, which is probably a key test to determine if the hCG measurements are as they should be, nearly undetectable, thus more patients show up for the test. Then another steady drop in test attendance can be observed until month 17, after which very few patients are tested again until, and including, month 24.
- Figure 7 (Figure 3.15 from the thesis) shows some of the hCG measurements of the patients, and by having a closer look, it can be seen that the disease can manifest in different ways for different patients. In Figure 7a a normal disease devolution can be observed, that follows an exponentially decreasing trend. Next, in Figure 7b, the measurements suggest that the patient had complications after the extraction of the mole, but after therapy, the hCG levels returned to the normally expected, exponentially decreasing values. The third case, which can be seen in Figure 7c, shows an example where the disease probably recurred after a longer period of time, but after the intervention of the doctors, the levels returned to normal, as it can be seen from the measurements from 10 months later.



(a) Normal disease devolution (b) Early anomaly (c) Possible disease recurrence

Figure 7: Samples of different hCG measurements.

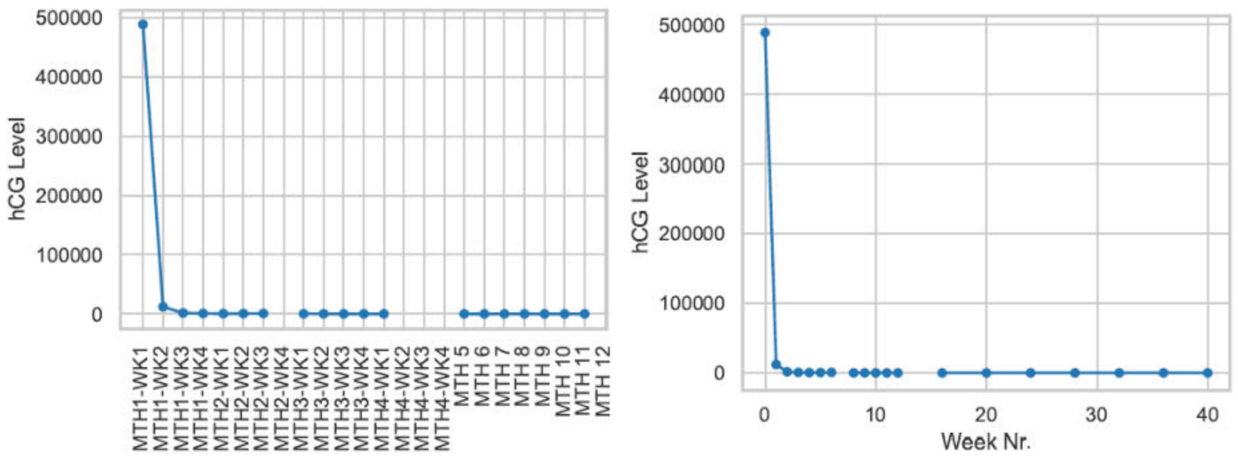
Having done a proper and in-depth exploratory analysis of the dataset makes the manual feature selection a much easier job. Looking again at the features of the dataset, but this time having a feeling of what knowledge they actually retain, it's easy to determine which of them could be useful in the future. For starters, the medical record number (MRN) and the patient identifier (I) will surely have to be removed, as they don't contain important information from the perspective of a learning algorithm. Next, the diagnostic date, although it helped to grasp the period in which the measurements were done, no information from the literature suggests that it could be correlated to the development of the molar pregnancy, so it's safe to assume it's irrelevant, thus removed from the dataset.

The age and the diagnostic (DX) features require more thought before making a decision about their fate. Literature suggests that the age of a patient could be an important factor in the development of the moles, yet the current dataset does not entirely reflect this. Considering that a learning algorithm only sees the data, it's easy to imagine that it could develop possibly wrong assumptions. On the other hand, a possible correlation could exist between the age of the patient and the devolution or recurrence of the disease, i.e., between the age and the hCG measurements features. The story is more or less the same for the diagnostic feature. Thus, the choice of keeping both or any of the age and diagnostic (DX) features will be deferred to the moment the learning algorithm is chosen.

As seen in the exploratory analysis phase, the bulk of the dataset are the hCG measurements. In their current state though, they are to some extent unusable because they hide important information about the rate at which the sampling was done. This is why the features need to be transformed so that they reflect the correct time span while retaining the current hCG measurements.

The aforementioned transformation consists of relabeling and mapping each original feature to an integer that represents the number of weeks that passed since the patient started the surveillance period. Thus "MTH1-WK1" becomes 0, ..., "MTH3-WK3" becomes 10, and so forth. Having the measurement feature names as numbers enables their correct representation on a time axis, or the computation of delta times if and when the need arises.

Figure 8 shows how the hCG levels of a patient look before and after the transformation. It can easily be seen in Figure 8a, the distance between two points in the beginning, when the sampling was weekly, is the same as between two points at the end when the sampling was monthly. In Figure 8b, this aspect is improved, and a change can be observed even in the shapes of the curves. Before the transformation, the curve was less steep than after the transformation.



(a) Before transformation (b) After transformation

Figure 8: The hCG levels of a patient.

As mentioned before, the dataset contains quite a lot of missing values especially in the features that represent the hCG measurements. Section 3.3.4 from the thesis enumerates and discusses some of the possible solutions, but a decision is deferred until the Chapter 4 where it will be taken based on the chosen algorithm. Among the proposed missing value handling methods, there are methods that help in keeping the problem univariate, while other methods are more suited for algorithms like Recurrent Neural Networks.

In its current state, it's difficult to apply any type of machine learning algorithms on the dataset, that could extract knowledge and make predictions about future or missing hCG levels, thus the dataset needs to undergo another set of transformations to be fit for sequence learning algorithms. Section 3.3.5 of the thesis together with Algorithm 1 from the thesis show how the dataset is transformed so that algorithms in Chapter 4 can use it for both univariate and multivariate sequence learning, prediction and forecasting.

In support of this chapter the following research articles were published:

- *Feature Inspection and Elimination in the Context of Breast Cancer Prediction*, by Árpád Kerestély, in Proceedings of the 36th International Business Information Management Association (IBIMA), 2020. This paper aims to review some of the existent research from the field of breast cancer, gather a set of useful methods and processes, and finally apply them on a publicly available breast cancer dataset, with a focus on understanding how different features, or their absence, can influence the outcome of a prediction, thus being the basis for Section 3.2.
- *Vertically Shifted Exponential Best-Fit*, by Árpád Kerestély, Catherine Costigan and Marius-Sabin Tăbîrcă, in Proceedings of the 35th International Business Information Management Association (IBIMA), 2020. This paper introduces a new method for fitting data to an exponentially decreasing curve and represent a basis for Section 3.3 as it carries out the experiments on the presented molar pregnancy dataset.

# *Chapter 4: Time Series and Forecasts in Cancer Prediction*

This chapter aims to introduce new methods [61, 63] for forecasting the evolution of the hCG levels in patients diagnosed with GTD as this could reduce the number of weekly blood tests that a patient would require throughout the post evacuation surveillance period. The hCG levels are modeled as a vertically shifted exponential curve, and this chapter proposes and demonstrates a mathematical solution to finding the best parameters for this model, considering each patient individually. In the second part of the chapter, RNNs will be used to model the behavior of this exponential decay, using data from a group of patients. The methods will be validated using synthetic data as well as real-world data.

## 4.1 Introduction to exponentially decaying time series

Exponentially decreasing data examples can be found all over the world. While most of these decrease to zero, there are some examples that decrease with a vertical shift. Data in this form can be difficult to fit to a mathematical model. Iterative algorithms, such as Levenberg-Marquardt for nonlinear least squares curve-fitting, exist to fit the data to a curve, but have some constraints that need to be considered, some hyperparameters that need to be tuned and convergence can be slow.

Exponential decay is common in the natural sciences. It occurs when a quantity decreases at a rate proportional to its current value. Some examples of this are the decay in time of a pendulum swinging in the air [51] and the growth in time of an initially small bacterial colony [107]. Data that decays exponentially in the form $y(t) = Ae^{-\alpha t}$ is very simple to fit a model to by simply taking the natural log of each data point and using simple linear regression to fit the transformed data to a line. When the data does not decay to zero, however, it is not as simple. Data in the form

$$y(t) = Ae^{-\alpha t} + B \tag{4.1}$$

also occurs naturally. Data in this form is not as simple to fit a model to, especially if only a small number of data points are available.

One set of data in this form is the human chorionic gonadotropin (hCG) measurements in women with Gestational Trophoblastic Disease (GTD) [90, 97]. It has been shown in [103, 104, 41] that hCG levels in these women decrease exponentially according to $(4.1)(4.1)$.

Figure 9 (Figure 4.1 from the thesis) shows a sample of hCG measurement. It can be observed that the hCG measurements follow an exponential decay curve, that they can be noisy, and that there are time intervals where there is no hCG measurement, which significantly increase the difficulty of using certain algorithms. If a forecast is desired from the first few measurements, then a slight vertical shift will have to be considered.

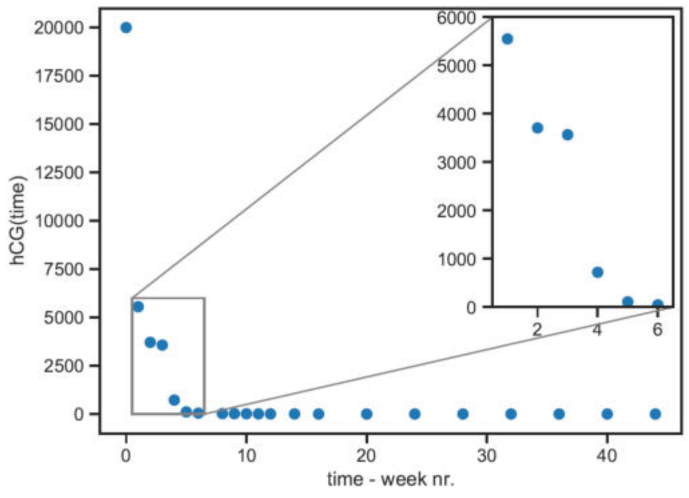


Figure 9: A sample of hCG measurement.

The problem of mathematically modeling hCG in Women with Gestational Trophoblastic Disease was done in [31] using logarithmic transformations. Later in this chapter, this method will be referred to as the "PseLogLin" method. A summary of this method can be found in Section 4.1.1 of the thesis, and the implementation of the method in Algorithm 3 from the thesis.

Another solution to fit a sample of hCG measurements to a curve is the "The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems" [44]. A summary of this method can be found in the thesis at Section 4.1.2, and later in this chapter, this method will be referred to as the "Iterative" method.

### 4.2 Direct computation methods with the advantages of using domain knowledge

A new method will be presented in this section that is competitive against the above-mentioned algorithms. While maintaining their accuracy, it has no hyperparameters and the only constraint it has is that the data, which can be noisy, needs to *roughly* follow an exponential decay curve: $Ae^{-\alpha x} + B$ with $A, \alpha$ and $B \geq 0$. The method was tested on both simulated and real data and performed equally well in both cases.

**Definition 1.** *Given the points* $(x_i, y_i)_{i=0}^{n-1}$ *and the model* $f(x)$ *that has been fit to the data points, the residuals (errors) are the values*

$$r_i = y_i - f(x_i) \tag{4.3}$$

*for each value of* $i \in \{0, \dots, n-1\}$.

The method of least squares is used, to minimizes the sum of the squares of the residuals in order to find the best parameters to fit the model to the data. When using the least squares method to fit a model, say $\mathrm{f(x; a_1, \dots, a_k)}$, where $\mathrm{a_1, a_2, \dots, a_k}$ are the parameters to be estimated, to a set of data $\{\mathrm{x_I, y_i}\}_{\mathrm{i=0}}^{\mathrm{n-1}}$, the aim is to minimize the sum of the squares of the residuals

$$\phi(a_1, a_2, \dots, a_k) = \sum_{i=0}^{n-1} (f(x_i; a_1, \dots, a_k,) - y_i)^2 . \tag{4.4}$$

Usually, when using this method $\phi$ is minimized by solving the system of equations

$$\begin{aligned} \frac{\partial \phi}{\partial a_1}(a_1, \dots, a_k) &= 0 \\ \frac{\partial \phi}{\partial a_2}(a_1, \dots, a_k) &= 0 \\ &\vdots \\ \frac{\partial \phi}{\partial a_k}(a_1, \dots, a_k) &= 0 \end{aligned} \tag{2.5}$$

for $\mathrm{a_1, a_2, \dots, a_k}$.

In the case of model (4.1), the function ϕ is defined to be

$$\phi(A,B,\alpha) = \sum_{i=0}^{n-1}(Ae^{-\alpha x_i} + B - y_i)^2 \tag{4.6}$$

Then

$$\begin{aligned}\frac{\partial\phi}{\partial A}(A,B,\alpha) &= \sum_{i=0}^{n-1} 2(Ae^{-\alpha x_i} + B - y_i)e^{-\alpha x_i}\\ &= 2A\sum_{i=0}^{n-1} e^{-2\alpha x_i} + 2B\sum_{i=0}^{n-1} e^{-\alpha x_i} - 2\sum_{i=0}^{n-1} y_i\, e^{-\alpha x_i}\end{aligned} \tag{4.7}$$

$$\begin{aligned}\frac{\partial\phi}{\partial B}(A,B,\alpha) &= \sum_{i=0}^{n-1} 2(Ae^{-\alpha x_i} + B - y_i)\\ &= 2A\sum_{i=0}^{n-1} e^{-\alpha x_i} + 2B\sum_{i=0}^{n-1} 1 - 2\sum_{i=0}^{n-1} y_i\end{aligned} \tag{4.8}$$

$$\begin{aligned}\frac{\partial\phi}{\partial \alpha}(A,B,\alpha) &= \sum_{i=0}^{n-1} -2(Ae^{-\alpha x_i} + B - y_i)Ax_i e^{-\alpha x_i}\\ &= -2A^2\sum_{i=0}^{n-1} x_i\, e^{-2\alpha x_i} - 2AB\sum_{i=0}^{n-1} x_i\, e^{-\alpha x_i} + 2A\sum_{i=0}^{n-1} x_i\, y_i e^{-\alpha x_i}\end{aligned} \tag{4.9}$$

The following functions will be defined $f_k$, $g_k$, $h_k$and $l_k$

$$\begin{aligned}f_k(\alpha) &= \sum_{i=0}^{n-1} e^{-k\alpha x_i}\\ g_k(\alpha) &= \sum_{i=0}^{n-1} y_i\, e^{-k\alpha x_i}\\ h_k(\alpha) &= \sum_{i=0}^{n-1} x_i\, e^{-k\alpha x_i}\\ l_k(\alpha) &= \sum_{i=0}^{n-1} x_i\, y_i e^{-k\alpha x_i}\end{aligned} \tag{4.10}$$

in order to have a simpler form of the above equations

$$\frac{\partial\phi}{\partial A}(A,B,\alpha) = 2Af_2(\alpha) + 2Bf_1(\alpha) - 2g_1(\alpha) \tag{4.11}$$

$$\frac{\partial\phi}{\partial B}(A,B,\alpha) = 2Af_1(\alpha) + 2Bf_0(\alpha) - 2g_0(\alpha) \tag{4.12}$$

$$\frac{\partial\phi}{\partial \alpha}(A,B,\alpha) = -2A^2h_2(\alpha) - 2ABh_1(\alpha) + 2Al_1(\alpha). \tag{4.13}$$

In order to minimize $\phi(A,B,\alpha)$, the following equations will need to be solved

$$\begin{aligned}\frac{\partial \phi}{\partial A}(A,B,\alpha) &= 0\\ \frac{\partial \phi}{\partial B}(A,B,\alpha) &= 0\\ \frac{\partial \phi}{\partial \alpha}(A,B,\alpha) &= 0.\end{aligned} \tag{4.14}$$

which after the substitution look like this

$$\begin{aligned}2Af_2(\alpha)+2Bf_1(\alpha) &= 2g_1(\alpha)\\ 2Af_1(\alpha)+2Bf_0(\alpha) &= 2g_0(\alpha)\\ 2A^2h_2(\alpha)+2ABh_1(\alpha) &= 2Al_1(\alpha).\end{aligned} \tag{4.15}$$

Using Cramer's Rule on the first two equations

$$\begin{aligned}\Delta &= \begin{vmatrix} f_2(\alpha) & f_1(\alpha)\\ f_1(\alpha) & f_0(\alpha)\end{vmatrix} = f_0(\alpha)f_2(\alpha)-f_1^2(\alpha),\\ \Delta_1 &= \begin{vmatrix} g_1(\alpha) & f_1(\alpha)\\ g_0(\alpha) & f_0(\alpha)\end{vmatrix} = g_1(\alpha)f_0(\alpha)-g_0(\alpha)f_1(\alpha),\\ \Delta_2 &= \begin{vmatrix} f_2(\alpha) & g_1(\alpha)\\ f_1(\alpha) & g_0(\alpha)\end{vmatrix} = g_0(\alpha)f_2(\alpha)-g_1(\alpha)f_1(\alpha),\end{aligned} \tag{4.16}$$

results in

$$A=\frac{\Delta_1}{\Delta}=\frac{g_1(\alpha)f_0(\alpha)-g_0(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)} \tag{4.17}$$

$$B=\frac{\Delta_2}{\Delta}=\frac{g_0(\alpha)f_2(\alpha)-g_1(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)} \tag{4.18}$$

The value for α can be found by finding one of the roots of the third equation

$$Ah_2(\alpha)+Bh_1(\alpha)=l_1(\alpha) \tag{4.19}$$

which after replacing $A$ and $B$, becomes

$$\frac{g_1(\alpha)f_0(\alpha)-g_0(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)}h_2(\alpha)+\frac{g_0(\alpha)f_2(\alpha)-g_1(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)}h_1(\alpha)=l_1(\alpha). \tag{4.20}$$

**Proposition 1.** *Given a set of data* $\{x_i, y_i\}_{i=0}^{n-1}$ *and a model*

$$y(t)=Ae^{-\alpha t}+B \tag{4.21}$$

*the best value for* $\alpha$ *can be determined by finding a root to the equation*

$$\frac{\mathrm{g}_1(\alpha)\mathrm{f}_0(\alpha)-\mathrm{g}_0(\alpha)\mathrm{f}_1(\alpha)}{\mathrm{f}_0(\alpha)\mathrm{f}_2(\alpha)-\mathrm{f}_1^2(\alpha)}\mathrm{h}_2(\alpha)+\frac{\mathrm{g}_0(\alpha)\mathrm{f}_2(\alpha)-\mathrm{g}_1(\alpha)\mathrm{f}_1(\alpha)}{\mathrm{f}_0(\alpha)\mathrm{f}_2(\alpha)-\mathrm{f}_1^2(\alpha)}\mathrm{h}_1(\alpha)=\mathrm{l}_1(\alpha). \tag{4.22}$$

*and then using this value of* α, *the best values of* $\mathrm{A}$ *and* $\mathrm{B}$ *can be found using the equations*

$$A=\frac{g_1(\alpha)f_0(\alpha)-g_0(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)} \tag{4.23}$$

$$B=\frac{g_0(\alpha)f_2(\alpha)-g_1(\alpha)f_1(\alpha)}{f_0(\alpha)f_2(\alpha)-f_1^2(\alpha)}. \tag{4.24}$$

The algorithm to find the best parameters for equation (4.1) is straightforward and can be seen in Algorithm 2 from the thesis.

Equation (4.22) does not have a single solution and with mathematical calculus it's not easy to find any of its roots. Thus, the following changes are introduced.

Equation (4.22) will be rewritten as a polynomial, see Section 4.2.5 from the thesis for the proof, by making the following substitution

$$e^{-\alpha} = t. \tag{4.26}$$

This substitution reduces the functions from (4.10) to a simpler form

$$\begin{aligned} f_k(t) &= \sum_{i=0}^{n-1} t^{kx_i} \\ g_k(t) &= \sum_{i=0}^{n-1} y_i\, t^{kx_i} \\ h_k(t) &= \sum_{i=0}^{n-1} x_i\, t^{kx_i} \\ l_k(t) &= \sum_{i=0}^{n-1} x_i\, y_i t^{kx_i}. \end{aligned} \tag{4.27}$$

and results in equation (4.22) taking the following form

$$\frac{g_1(t)f_0(t) - g_0(t)f_1(t)}{f_0(t)f_2(t) - f_1^2(t)} h_2(t) + \frac{g_0(t)f_2(t) - g_1(t)f_1(t)}{f_0(t)f_2(t) - f_1^2(t)} h_1(t) - l_1(t) = 0. \tag{4.28}$$

One of the useful properties of this equation is that it has a root in the interval $(0,1)$, again, see Section 4.2.5 from the thesis for proof, which makes finding $t$ guaranteed and independent of the data points, with the Bisection method (see Section 4.2.3 from the thesis). Finding $t$ helps in finding $\alpha$ using the inverse transformation

$$\alpha = -ln(t) \tag{4.29}$$

In summary, finding $\alpha$ using equation (4.22) and the Bisection method is difficult, because it has multiple roots and the interval in which to search for at least one of them is missing, but using equation (4.28), t can be found in the interval $(0,1)$, independent of the data points and independent of what should be the value of $\alpha$. Finally, using equation (4.29), the best value of $\alpha$ can be found from the value of t. See Algorithm 4 from the thesis for details on the concrete implementation. The full source code for this section can also be accessed on the publicly accessible GitHub repository: https://github.com/akerestely/nonlinearBestFit.

In order to test the method for estimating A, B and $\alpha$, synthetic data in the form $\{x_i, y_i\}_{i=0}^{n-1}$ was created using Algorithm 5 presented in the thesis. Several tests were performed using the synthetic data.

First, a model was fit to the data to see if the method for estimating the parameters could find the original parameters ($A, B, \alpha$ and noise having the values $1000, 3, 1$ and $0$ respectively), results can be seen in Figure 10 (Figure 4.3 in the thesis). To validate the accuracy of the found model numerically also, not only visually, the *root mean square error* (RMSE) method was used.

Next step was to test the robustness of the method. Noise was added to the generated data and different combinations of parameters were tested. Figure 4.4 from the thesis or Appendix A.1 show the results and demonstrate that the method is capable of working with noisy data.

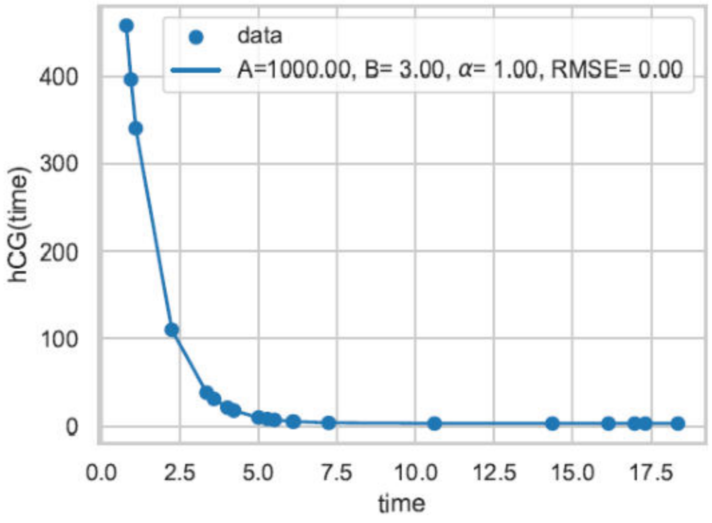


Figure 10: The curve generated by the found parameters matches the data perfectly.

To truly grasp the effectiveness of this new method, a comparison was made with two other existing methods from the literature, *curve_ fit* from the SciPy Python library, which internally uses Levenberg-Marquardt, an iterative curve fitting method, further labeled as 'Iterative', and the pseudo logarithmic curve fitting approach presented in [31], further labeled as 'PseLogLin'. The new method is labeled 'BestFit' for quick reference. **Error! Reference source not found.** (Figure 4.5 from the thesis) shows the results of running all three algorithms on different generated datasets. Note, that 'Iterative' and 'BestFit' completely overlap, denoted also by the same RMSE values, while the 'PseLogLin' underperforms in all three cases. The thesis also presents cases where the 'Iterative' method does not converge, while 'BestFit' and 'PseLogLin' do. See Figure 4.6 from the thesis.

**Remarks:**

'BestFit' method compared to the 'Iterative' method does not need any hyperparameters to work properly, as long as the data is on an exponential decay curve. A second advantage of the 'BestFit' method is that it also converges faster than the 'Iterative' method, almost in all cases. The variation in the convergence time of the 'BestFit' is given only from the bisection method, which iteratively searches for the root of equation (4.28).

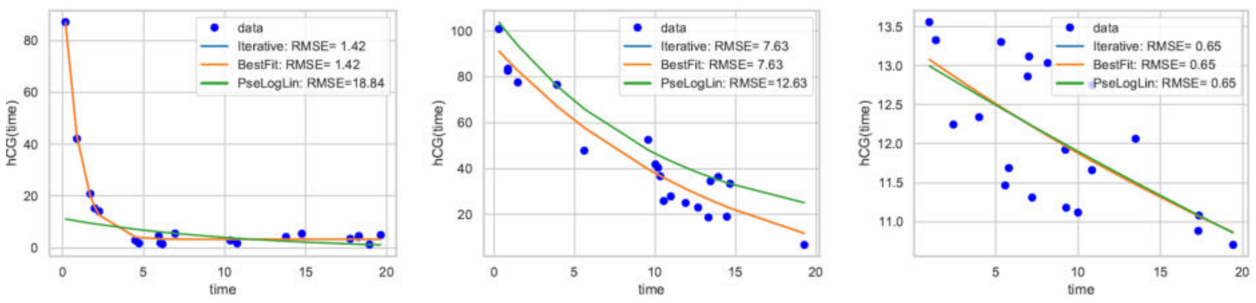


Figure 11: Comparison of three different curve fitting algorithms on synthetic data.

Testing the new method on the real molar pregnancy data presented in Chapter 3, required first eliminating the samples where the number of hCG measurements were less than three (see Section 4.2.8 from the thesis for more details), then a treating the missing hCG measurements, by eliminating the corresponding $\{x, y\}$ pairs from the dataset.

Again, the results of all three curve fitting methods from above were compared against each other. Figure 4.7 from the thesis shows the findings and reinforce that 'BestFit' and 'Iterative' are equally good, while 'PseLogLin' underperformes.

Next, a few interesting applications of the new method were tested. Like accurately predicting the trendline of the hCG measurements, by using only the first few values, see Figure 12 (Figure 4.8 from the thesis).

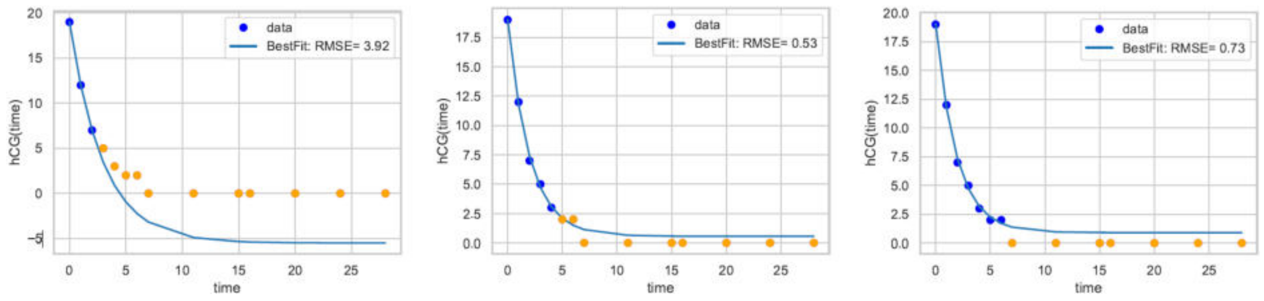
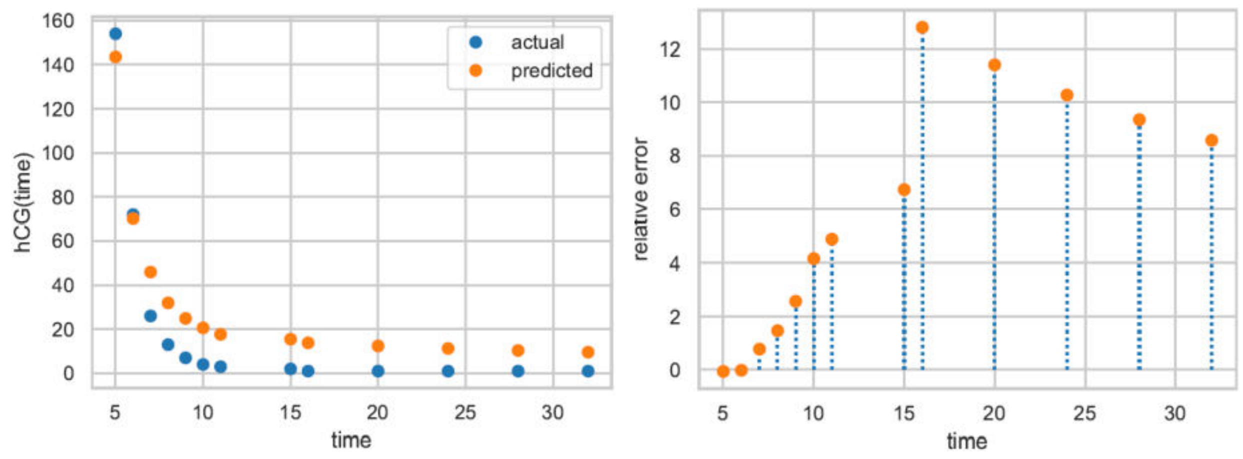


Figure 12: Fitting the curve on the same sample with different number of starting data points (blue dots).

Another interesting but mainly more accurate use-case of the algorithm presented in this section is one, in which all the available measurements, until some point in time, are used for fitting the curve, which then yields the value of the next measurement in the sequence. Figure 13 (Figure 4.9 from the thesis) presents a case where this approach was used. The first 5 measurements (weeks 0 to 4) were used to fit the initial curve and predict the value of week 5. Then the first 6 measurements were used to fit the curve and predict the measurement for week 6, and so on. The figure also shows the relative errors to better reflect the error rate of each prediction.



(a) Predictions (b) Relative errors

Figure 13: Next measurement predicted from all previous data.

The model generated by the algorithm can be used as AI assistant to validate new measurements thus aiding the judgment of a medical expert. Having a forecast, a doctor can easily tell if a new measurement is in acceptable thresholds and decide if actions should be taken. The model can also be updated with new data points, making it generate a new forecast. If at any time the algorithm fails to converge to a result, that means that the measurements are too noisy, or that the disease is not devolving. In both cases it should be a warning to the doctor. Figure 14 (Figure 4.10 from the thesis) clearly shows these cases. When using the first

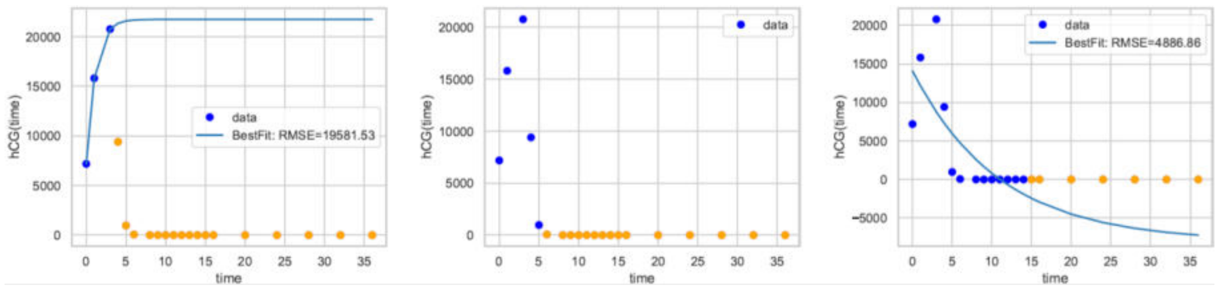


Figure 14: Three behaviors of the curve fitting algorithm, depending on which moment in time the prediction was run.

3 data points, the algorithm finds that the measurements increase instead of decreasing. The doctor probably notices the same thing and takes action. When using the first 5 data points, the algorithm cannot determine an exponentially decreasing curve, as there is big uncertainty from the algorithm's point of view. Yet a doctor can see that this uncertainty is merely because the patient started recovering only from week 3, in which case the doctor can adjust the window that is used for computing the trendline, making the algorithm use the data points only from week 3 onward, in which case the results start to look like in Figure 15 (Figure 4.11 from the thesis). If no change is done to the input, then only after the 13th data point the algorithm starts to guess that the measurements decrease exponentially, converging slowly to the form of the final curve.

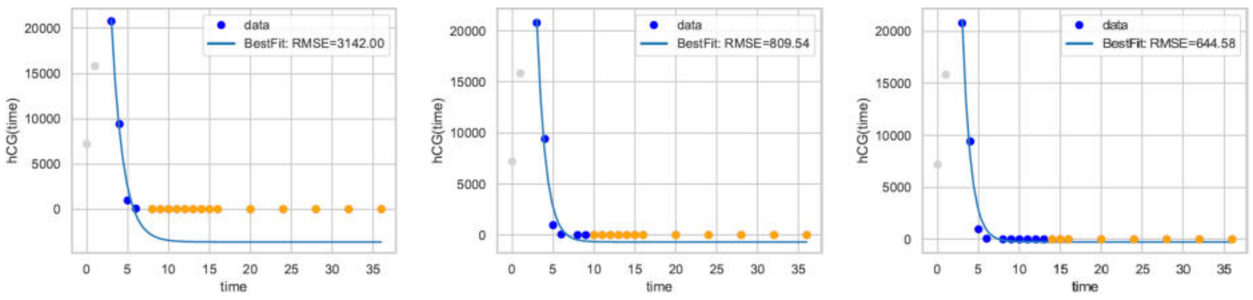


Figure 15: Skipping the first two points in this sample (gray dots), helps the algorithm in getting back on track and performing better overall.

In summary, this section presented a new method to fit a model to data points that follow an exponential decay curve in the form $\mathrm{Ae}^{-\alpha x} + \mathrm{B}$ with $\mathrm{A}, \alpha$ and $\mathrm{B} \geq 0$. Data of this type appears in the hCG measurements of women diagnosed and treated of molar pregnancy. This new method uses mathematical calculus to determine the best model that matches the data points. It outperforms other existing methods either in accuracy, convergence time or ease of usage, by not having hyperparameters. Ideas were also presented in this section as to how the curve fitting could be used in real life scenarios. Extrapolation, forecasting, interpolation, or validation of data points are some of these use cases.

## 4.3 Recurrent Neural Networks

The methods discussed in the previous section are really good at learning data that decreases with an exponential rate. It could be said that they are the perfect choice when thinking of normal cases of the molar pregnancy, as they incorporate, and make use of the domain knowledge by design. But what happens when exceptions happen and the hCG measurements stop decreasing as expected, like in the cases when the disease reoccurs, or when the patient's hCG levels have an initial increase for a few weeks? The answer is simple, the curve fitting methods stop being accurate and totally reliable. This section aims to study methods that don't have domain knowledge incorporated in their design, but actually learn the rate at which a sequence increases or decreases by looking at a number of example samples.

There are quite a few algorithms in the literature that can learn from sequence data and make predictions based on what was learned. Yet the dataset available for this study is quite small, thus based on the existing literature, the RNNs look like the ones that could learn even from a low number of samples.

Experiments were run, first, on a series of synthetically generated, exponentially decreasing data, which will become progressively irregular, thus assessing the performance of the RNNs, one irregularity at a time. Methods were presented to overcome some of the limitations. Next, with this knowledge, attempts were made at learning and making predictions on real data regarding the molar pregnancy.

The synthetic dataset tries to replicate data that closely resembles the hCG measurements from the molar pregnancy dataset. From an ideal case, where the data perfectly decreases exponentially, meaning that the patients have all their measurements done and recover normally without complications or recurrences, to cases where measurements are missing or noisy, much like in the real-world data. The dataset is composed of a number of virtual patients. For each patient, the hCG measurements are generated using a modified version of Algorithm 5 from the thesis, where it's guaranteed that the time values are positive natural numbers, unique and increasing. Each patient has its hCG measurements generated by a different combination of the $\mathrm{A}, \mathrm{B}$ and $\alpha$, where $10 < \mathrm{A} < 10^5$, $0 < \mathrm{B} < 50$ and $0.3 < \alpha < 4$, see Table 4.1 from the thesis for more details.

An additional data preprocessing step is required before attempting to run RNNs on the aforementioned dataset. More precisely, the data needs to be transformed into inputs and outputs. To achieve this, an imaginary window will be placed over the samples of the dataset, which will yield an input and out pair for the RNNs. Next, the window will be shifted, thus providing a new input-output pair. The inputs represent a subset of the dataset's features, and the outputs represent the feature or features that are desired to be predicted, in the current experiments, the hCG measurement. The size of this window is dependent on the number of steps that the RNN will process at once. If for example, using the measurements of the first four weeks, the measurement from the fifth week is desired to be predicted, then the window will have a size of five, and will split the data so that the measurements from the first four weeks will represent the input, and the fifth measurement will represent the output when training an RNN.

The architecture of the RNN used for the next experiments is as follows:

- an input layer with at least one neuron for the hCG measurements, and where required, additional neurons for other features like the week number
- a hidden layer with 50 LSTM cells using the ReLU activation function, which was empirically determined to be both sufficient and small enough so that the learnable parameters can converge to optimal values even in the case of a few input samples, for all following experiments
- fully connected output layer, with one neuron using linear activation function, for the predicted value

Additionally, the model was configured to use the Adam optimization function and to compute the mean squared error (MSE).

Multiple experiments were carried on the synthetic dataset. The first set of experiments took three consecutive hCG measurements and tried to predict the fourth one, to closely resemble the functionality of the method presented in the previous section. Additionally, these first experiments only used the hCG measurements both as inputs and outputs, basically the univariate case. The results of training a RNN network for 1000 epochs, on all samples except one, that was used for testing, can be seen in Figure 16 (Figure 4.13 from the thesis).

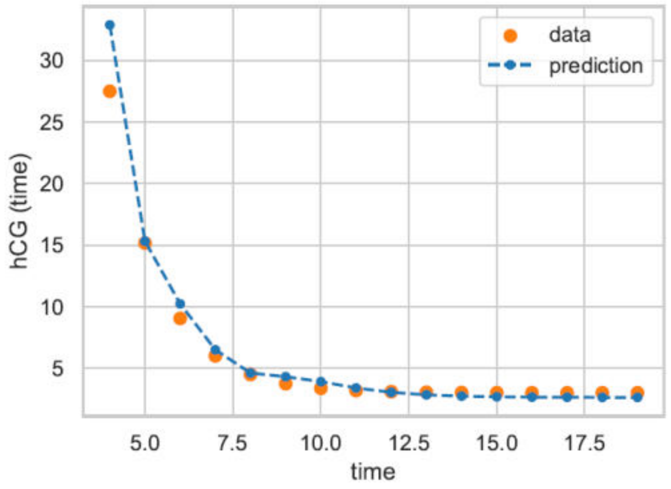


Figure 16: Prediction of the RNN on the generated dataset using only the hCG feature.

The performance of the RNN was assessed also using cross-validation (CV), but because the dataset is small, a special case of the CV was used, the leave-one-out cross-validation (LOO CV). Results can be seen in Figure 4.14 in the thesis.

Next, to enhance the performance of the RNN model, the *week_nr* feature was added to the inputs, making the problem a multivariate one. The accuracy of the model slightly improved compared to the univariate case. Results can be seen in Figure 4.15 in the thesis.

Next, the RNN model was tested against datasets containing either noisy or missing hCG values. In both cases, the performance degraded, slightly more in the univariate case, yet the model was able to make acceptable predictions. See Figures 4.16 and 4.17 from the thesis for more details. In an attempt to help the RNN in making better predictions, a new feature was introduced in place of the *week_nr* feature, called *delta_time*. This new feature was computed from the *week_nr* feature and represented the number of weeks that have passed since the last measurement was done. Figure 4.18 from the thesis shows that with this feature engineering, the model's prediction was improved in the case of a dataset containing missing values.

RNNs trained in the aforementioned ways, can also be used to forecast more than just one measurement. To better showcase this behavior, the generated dataset was reverted to the version which contained no noise and missing values. As previously, it was trained on all, except one patient's data, having only the hCG measurements as input. In the testing phase, the first 3 hCG measurements of the remaining patient were fed to the RNN model. The only prediction resulting this way, was appended to the 3 initial values, and a new prediction was made with the last 3 elements of the resulting sequence. This procedure can be recursively repeated for an unlimited number of times, yielding forecasts even for a distant future, although caution is advised when doing it for many iterations, as the predictions can accumulate errors, making the forecast deviate from a normal course. **Error! Reference source not found.** (Figure 4.19 from the thesis) shows the predictions made with this procedure as well as the original measurements for comparison.

Sequence-to-sequence models, or sequences with arbitrary or long inputs were left out of these set of experiments, as they are not suitable for the current context. More details can be seen in Section 4.3.1 of the thesis.

Instead, another interesting solution is proposed, in which the RNN learns to make predictions on a fixed sequence length but is still able to provide good predictions from shorter or longer sequences. A solution which could potentially achieve this behavior is one in which all the hidden states are returned from the network and are trained using the target (output) variables. This solution requires two adjustments to the current workflow: modifying the window that generates the input-output pairs and modifying the RNN setup. Having these

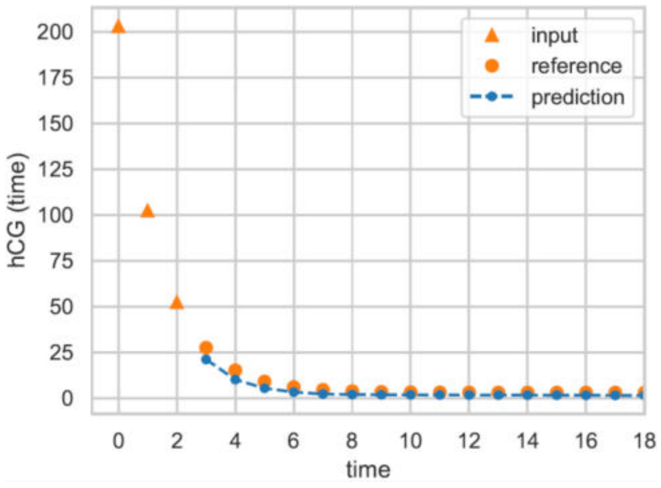


Figure 17: RNN forecast, by reusing predictions.

two adjustments, the RNN can learn not only from the error of one output, but from the accumulated errors of all the outputs, thus effectively providing better predictions even from shorter sequences. It needs to be noted, that even if the network returns multiple outputs, the one that provides the actual one step ahead prediction is the last one, while all the rest are used only for training. Figure 18 (Figure 4.20a from the thesis) shows a scenario, where a RNN model was trained to predict from a sequence of length 7. At testing, the same model was used to predict from sequences of length $1, 2, \dots, 18$. It can be seen that although the predictions are not accurate when the sequence lengths are on the extremes, they surely improve as soon as the sequence lengths start reaching the vicinity of 7. Figure 4.20b from the thesis shows the MSEs resulting from training and testing a RNN model, in the same manner, although this time reported for each fold from a LOO CV.

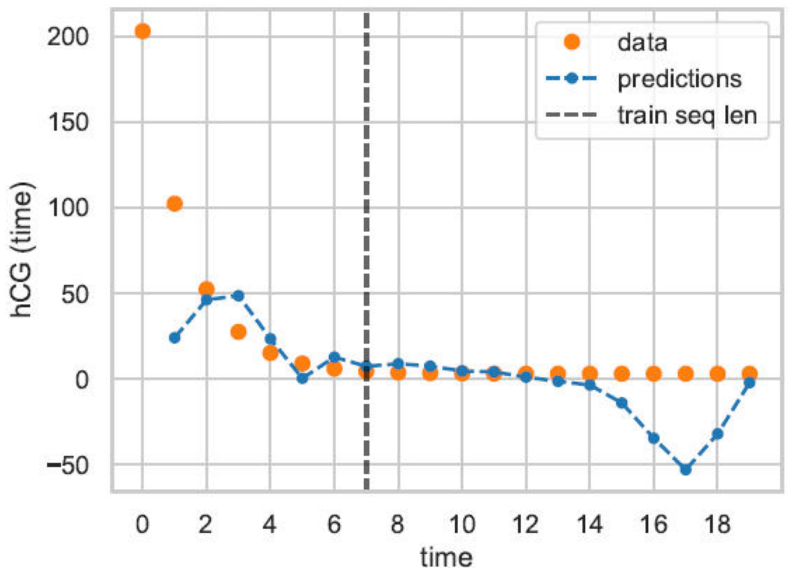


Figure 18: RNN training on fixed sequence length while testing on arbitrary sequence lengths.

It's interesting to see how well the RNNs can perform on real data, considering that they had good results on generated data, that was either noisy, or had missing values, or that different sequence lengths were used for training and testing.

As it can be recalled from Chapter 3, the real dataset contains data from patients diagnosed with molar pregnancy. The dataset's bulk are the hCG measurements, but a few clinical records are also available about the patients. In most cases the devolution of the disease was normal, thus following an exponentially decaying curve, nevertheless there were a few cases where the healing occurred after a few weeks delay or cases where the disease reoccurred. The initial dataset is transformed similarly as in the case of the generated data, thus all the hCG measurements are in one column, which also enables the usage of the windowing procedure from the previous subsection, to prepare the input-output pairs for the RNN.

Furthermore, to treat the problem of missing values, for reasons presented in Section 4.3.2 regarding the real dataset, the missing values were linearly interpolated and a new Boolean feature was introduced in the inputs, named *imputed*, which was set to true for each interpolated hCG measurements and otherwise it was set to false.

The prediction results of training a RNN with the above mentioned transformations can be seen in Figure 19 (Figure 4.21 from the thesis). The network was trained to make a prediction based on 3 previous inputs. The figure shows a normal disease devolution and one where a recurrence can be observed. In both cases the RNN model managed to provide fair predictions, proving that RNNs can learn to predict an individual patient's disease evolution based on what was learned from other patients' data.

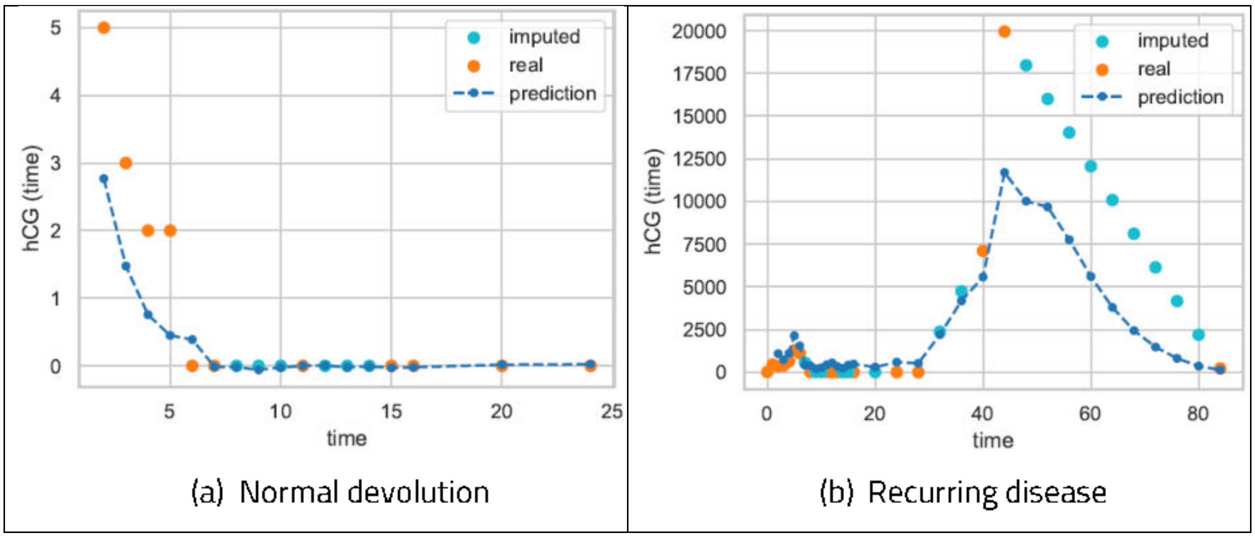


Figure 19: RNN predictions on real data.

In summary, Recurrent Neural Networks showed good potential at learning and predicting exponentially decreasing sequences. Although they can do better when the circumstances are more ideal, it proved that decent predictions are made even in noisy environments or when the sequences are incomplete. The different use-cases in which the RNNs can be applied, namely predicting from fixed length sequences, predicting from varying length sequences, or even forecasting by feeding back the predictions, makes them handy in a variety of situations. Overall, the performance of the presented RNN solutions could also be improved by having a larger dataset, with more examples of high hCG levels or recurring disease. Although more real-world samples would be the best option to increase the dataset, expanding it with synthetic data could also be a good option especially for the underrepresented cases. Having more data would certainly also open the way towards exploring other algorithms that handle time series and sequential data.

In support of this chapter the following research articles were published:

- Vertically Shifted Exponential Best-Fit, by Árpád Kerestély, Catherine Costigan and Marius-Sabin Tăbîrcă, in Proceedings of the 35th International Business Information Management Association (IBIMA), 2020. This paper aims to introduce a new method for forecasting the decrease of the hCG levels as this could reduce the number of weekly blood test that a patient would require throughout the one year of monitoring, thus being the basis for the experiments conducted in the section regarding the new method for determining the devolution curve in the study of molar pregnancy.
- Theoretical Study of Exponential Best-Fit: Modeling hCG for Gestational Trophoblastic Disease, by Árpád Kerestély, Catherine Costigan, Finbarr Holland and Marius-Sabin Tăbîrcă, in Proceedings of the 14th International Conference on Knowledge Science, Engineering and Management (KSEM), 2021. The hCG levels are modeled as a vertically shifted exponential curve, and this paper proposes and demonstrates a mathematical

solution to finding the best parameters for this model, thus being the basis for the proof behind the new method proposed in the section regarding the molar pregnancy.

# *Chapter 5: High Performance Computing in context of Big Data and Machine Learning*

Efficient High Performance Computing for Machine Learning has become a necessity in the past few years. Data is growing exponentially in domains like healthcare, government, economics and with the development of IoT, smartphones and gadgets [59]. This big volume of data needs a storage space which no traditional computing system can offer, and needs to be fed to Machine Learning algorithms, so useful information can be extracted out of it. The larger the dataset that is fed to a Machine Learning algorithm, the more precise the results will usually be, but also the time to compute those results will increase [60]. Thus, the need for Efficient High Performance computing in the aid of faster and better Machine Learning algorithms. This chapter aims to unveil how one benefits from another, what research has achieved so far and where it is heading.

## 5.1 Introduction

High Performance Computing, Big Data and Machine Learning started as different topics. They developed analogous of each other driven by different necessities. Still, they joined at some point and now it is hard to think of one, without the others lurking in the background. Before diving into details, it's important to look into some basic concepts regarding each of them.

"High Performance Computing (HPC) most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business." [100] Aggregating computing power can be done to result in a single supercomputer or a cluster of computers.

The most commonly used are the clusters of computers, as a cluster can easily be extended with additional nodes (computers) to quickly gain more computing power or storage space. The nodes are usually connected and communicate through Ethernet connection to act as one computer.

A strong bond exists between Machine Learning algorithms and High Performance Computing [55], even if at first glance it is not that obvious. Most Machine Learning algorithms need to "train" (we can call this fit, find the best parameters, or compute some distance) before being able to predict (or generalize) the results of unseen input. Training takes time. One fact is proven: the more the training data (or training iterations), the slower the train session will be. Another fact is certain too: the more training iterations and data an algorithm gets, the more chance it has to give better results.

The term "Big Data" [66] refers to a collection of large data sets that cannot be processed using traditional database administration tools. This generated numerous scientific challenges that are related to the provision of support for storage, and also concerning the data processing and retrieval.

The related literature suggests that the life cycle of the big data analysis process consists of three consecutive stages: data acquisition, data preprocessing and storage, and data analytics. This chapter asserts that there are also some variations of this taxonomy. Nevertheless, a separation of the general life cycle of the big data analytics process is done into the above three stages, considering that the mentioned taxonomy can accurately capture the key features of big data analytics.

The results of the performance assessment stage of the research that is reported in this chapter, and also the relevant contributions that are reported in the existing literature demonstrate that Spark, a high performance computing framework for running machine learning algorithms, exhibits a great potential to scale. Thus, the design decision to consider Spark for the processing core of the data analytics system is completely justified.

The research that is reported in this chapter focuses on Spark and its behavior in the context of big data, high performance computing and machine learning. Furthermore, it tries to assess whether Spark is a better tool for processing and running classification algorithms on big datasets. Moreover, the solutions are analyzed with respect to the following set of questions:

- How much is big data?
- Is Spark good enough for processing big data?
- Is Spark a scalable machine learning tool?
- Can the memory restrictions be solved by using the Spark framework?
- Does big data processing require a different mindset?
- Why and when should one use the Spark framework for machine learning?
- Could smaller datasets benefit from the power of Spark or is it proper just for big datasets?

### 5.2 Literature Overview

In the digitalized era we live in, large amounts of data are generated by the healthcare [62], government and economic systems. Some of the notable sources of data are "record keeping, compliance and patient related data, ..., national health register data, ..." [14]. These data help by providing "patient centric services, detecting spreading diseases earlier, monitoring the hospital's quality and improving the treatment methods" [14].

"Large-Scale Machine Learning based on Functional Networks for Biomedical Big Data with High Performance Computing Platforms" [40] outlines that Map-Reduce built on top of HDFS has some drawbacks, which are crucial when working with machine learning algorithms. One of the major drawbacks is that Map-Reduce architecture is designed to reload data from disk at each Map-Reduce processing function. Thus, it relies heavily on the disk I/O speed which is usually very slow compared to the speed of in-memory operations. Machine learning algorithms can't be efficient using Map-Reduce in its native form. "Spark" [11] overcomes this limitation by reducing the disk I/O operations and offering an in-memory solution, while keeping the fault tolerant behavior of Map-Reduce. The speed gain is claimed to be 100x the speed of Map-Reduce.

Recent work in the field of machine learning focused heavily on two types of algorithms: Convolutional Neural Networks (with all their flavors) and Deep Neural Networks. Because of the heavy processing and high memory footprint, these algorithms can't benefit from a framework like Spark, as much of the time would be wasted exchanging information between the computing nodes. To achieve faster communication speed between the memory and the computing unit, these algorithms have been moved to the Graphical Processing Unit (GPU). The result was an increase in speed from 10 to nearly 60 percent, compared to a run on CPU (using Single Instruction Multiple Data (SIMD)) [24].

Using Hadoop to connect multiple computers to act as one brings the advantage of having access to a variety of smaller components with which to extend the functionality of a big data oriented, machine learning application. Using the GPU's power to tame some of the most challenging machine learning algorithms can be a solution, but maybe connecting more GPUs or even having a cluster of machines with each having a couple of GPUs can be a solution too. It is important also to know the data that needs to be processed, because structured and unstructured data are stored and processed differently.

The fact that High Performance Computing and Machine Learning coexist is certain, from the high number of papers that are available on this topic. More important is that Machine Learning is fueled by High Performance Computing, thus reaching new heights every now and then.

## 5.3 Comparative Analysis of Spark and scikit-learn

This section presents an integrated data analytics system, which considers an assessment model that aims to optimize the big data analytics processes. Although the real-world data is not from the field of healthcare, but that of the automotive industry, the results are valid for any classification problem that uses tabular data, like the one presented in Chapter 3 for breast cancer. The motivation behind this data switch is that healthcare data is hard to come by in big quantities in the open, especially since GDPR kicked in, and while acquiring data from medical institutions is possible, getting the approvals takes a lot of effort and time, which was not available for this research.

Swapping the breast cancer data from Chapter 3 with a dataset concerning bearings (described in detail later in this section), was possible because they share many characteristics. First of all, both are tabular data, that have as columns the features of the data, and as rows the samples i.e., individual measurements. The features of the datasets are represented with numerical values in both cases. And finally, yet most importantly, both treat classification problems. There are also disadvantages regarding this approach, namely that the number of features don't match, and that the bearing dataset does not contain time series, so it cannot be a straightforward replacement for the dataset treating the molar pregnancy.

The experimental evaluation that is reported in this paper considers a dataset concerning the bearings' fault detection. Vibrations and acoustic signals were measured on an electric engine mounted on bearings, with four different bearing conditions: healthy, inner and outer race fault, and ball defect. The bearing condition marks the class of each entry, enabling the utilization of supervised learning. Concerning the classification process, the problem can be considered a multi-class classification problem, having as classes the four health conditions of a bearing. The dataset was generated using a machinery fault simulator from SpectraQuest.

The bearing fault detection dataset originally came split into 336 MATLAB files, with a total size of 19.69 GB. Considering the purposes of efficient storage and easier data processing, the 336 MATLAB files were converted into the same number of Apache Parquet files, with a total size of 9.75 GB. While there is no general agreement concerning the threshold for a dataset to be categorized as big, by the suggestions from [38], this dataset falls into the medium category, as its size belongs to the range 10 GB-1 TB, and it can be stored on a machine's storage drive, rather than in its memory.

The dataset contains 14 float64 and an int32 columns (features), and about 262 million rows (entries / samples). The features are as follows: BL_[X, Y, Z] (Left bearing - axis X, Y and Z), BR_[X, Y, Z] (Right bearing - axis X, Y and Z), MR_[X, Y, Z] (Motor - axis X, Y and Z), [BL, BR]_AE (Left bearing and Right bearing Acoustic Emission), [BL, BR]_Mic (Left bearing and Right bearing Microphone), defect_type. The last column represents the class and can take four different values: Healthy, Ball, Inner, Outer. These are stored as integer numbers, which take the values 0, 1, 2, 3, respectively. The speed feature is also interesting, as it depicts the motors' rotation per minute (rpm) parameter, and it has its values clustered in the vicinity of 300, 420, 540, ..., 2580, 2700, with an increment of 120 rpm.

In an attempt to overcome the data not fitting into memory or the computation of the results taking too long, four smaller datasets were generated. First, $10^2$ lines were randomly sampled from each of the 336 files, each of them originally containing 780800 lines, thus resulting in a dataset of uncompressed data with the size of 3.71 MB, which represents approximately 0.01% of the entire dataset. For further reference, the name of this dataset will be *data100*. Moreover, $10^3$, $10^4$ and $10^5$ lines were extracted in the same manner, which produced three datasets with the sizes of 37.17 MB (≈ 0.12%), 371.7 MB (≈ 1.28%), and 3,62 GB (≈ 12.8%). These datasets will be referred to as *data1k*, *data10k* and *data100k*. These subsets of the initial dataset enable the quick assessment of some initial setups, but they also enable the progressive comparison of the results of the two frameworks on larger datasets.

Scikit-learn is a framework that provides a high-level application programming interface (API) in order to easily specify and run standard machine learning models, such as Logistic Regression, SVM, and so on. It is usually used for running classification, regression and clustering algorithms on datasets that fit into the system's memory.

Considering the algorithms that are evaluated by the authors of [16] on the same dataset, Artificial Neural Networks, also known as Multilayer Perceptrons, were chosen in order to be tested on the scenarios required for this section.

Before outlining the results of the tests from the current study, there are some aspects to consider regarding how the scikit-learn version of the Multilayer Perceptron (MLPClassifier) was used. The considered network is composed of a 14-neuron input layer, three hidden layers with 50, 100 and 50 neurons respectively, and a four-neuron output layer. Scikit-learn required only the hidden layer setup, as the input and output were inferred from the training dataset. The maximum number of iterations was empirically fine-tuned at 500. The activation function was relu, the solver was adam, the L2 regularization term was 1e-4, the learning rate was constant throughout the training and had a value of 1e-3.

It is relevant to note that the scikit-learn algorithms and the results they generate in comparison to the Spark results, suggest that they may take advantage of the parallel computing power of multicore machines. Some of the scikit-learn algorithms are naturally linear. Therefore, they cannot be calculated in parallel, but in case of the MLPClassifier, one can talk about parallelization, as this benefits from the optimized BLAS implementation, which ensures the occurrence of multithreaded calls for different linear algebra routines, such as matrix multiplications. In Figure 5.1 from the thesis, it can be observed that there are 4 runs on 4 different *data100* datasets. The median accuracy is 48%, while the maximum accuracy is around 53%. Furthermore, the mean time is 1 minute and 33 seconds, and the time varies between runs, but it is almost always directly proportional to the values of the accuracy. The last result is somewhat odd relative to the other results. It's probably a consequence of splitting the dataset into training and test sets. In Figure 20, two of the best time and accuracy measurements can be observed on a logarithmic scale, for the datasets *data100*, *data1k*, and *data10k*. The accuracy and training time increases for larger datasets, which is somehow obvious. Considering an empirical point of view, the accuracy increases logarithmically, while the time increases exponentially. The figure shows the actual measurements with dots. Because of the memory constraints, the measurements for *data100k* could not be conducted, but the forecasted values, marked with an "x", and the trendline, marked with a dashed line, suggest that the accuracy could have reached a value around 71%, while the training time could have ranged between 24 and 27 hours.

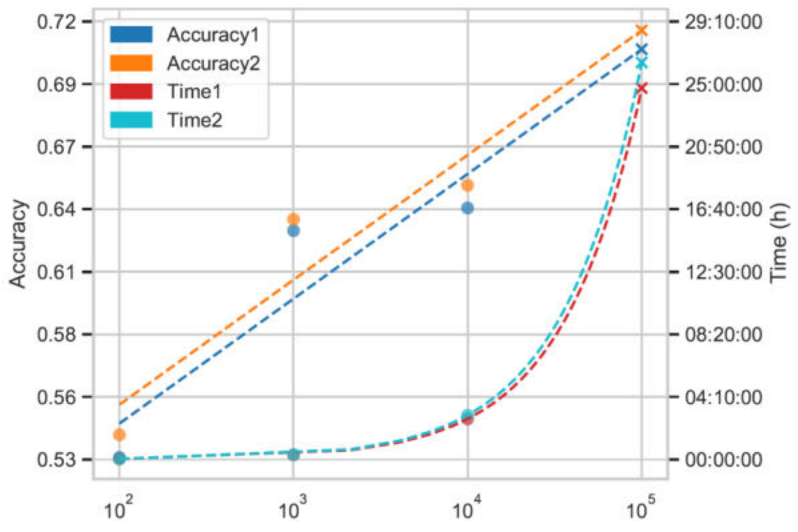


Figure 20: Assessment and forecasts for accuracy and time on larger datasets.

The scikit-learn experiments provided a solid base of comparison with Spark, although they also brought to surface a notable limitation in the form of the memory constraints. Consequently, this approach is not scalable. Considering a project that requires scaling, it can be asserted that a cloud solution could be a better choice. Therefore, the switch to Spark appears to be an optimal research direction.

Apache Spark is an open-source distributed general-purpose cluster-computing framework, which is mostly known for its capabilities to run data analytics on large-scale data. Furthermore, in the course of its development, the ability to run machine learning algorithms was also added to its code base.

Before outlining the results of the tests from the current study, there are some aspects to consider regarding how the Spark version of the Multilayer Perceptron (MultilayerPerceptronClassifier) was used. The considered network has the same architecture as the one from the scikit-learn tests. Spark requires specifying also the number of neurons of the input and output layer, besides the hidden layer. The number of input neurons is equal to the number of features in the training set minus one, as obviously, the defect_type column is not used as input. The number of output neurons is four, as there are four possible defect types (outcome classes). The maximum number of iterations is again 500, as for the scikit-learn tests. The solver was l-bfgs, the learning rate, called stepSize, was constant throughout the training and had a value of 0.03, finally, the tolerance value was 1e-6.

In the remaining part of this section, several scenarios are described, analyzed, discussed and compared with the scikit-learn baseline from a performance perspective.

### Scenario 1

Spark was configured to run one executor, with one core, and with 30 GB of associated memory. The input was one parquet file for each of the datasets *data100*, *data1k*, *data10k*, and *data100k*. The import of the data from a parquet file into a Spark DataFrame does not imply that all the data is loaded into memory, but only the subsets that are needed considering an on-demand model.

This scenario tried to replicate the scikit-learn setup as closely as possible, assuming that the scikit-learn algorithms ran on a single thread. Considering the two selected samples, as they are presented in Table 5.1 from the thesis, it can be observed that there is a big discrepancy in runtime between this scenario and the one that pertains to the scikit-learn implementation, which led to some investigations. It turned out that scikit-learn's implementation of MLP uses multithreading for most of its computation so, in order to be aligned, in the next scenario, multiple cores will be allocated to the Spark executor.

### Scenario 2

Spark was configured to run with a similar configuration as in Scenario 1, with the only difference that instead of one core, the executor was allowed to use all the twelve cores available on the machine.

The expectation from this scenario was to obtain similar results as compared to the scikit-learn implementation. Nevertheless, the results that are displayed in Table 5.2 from the thesis suggest an odd behavior. Thus, the results are nearly the same as in the case of Scenario 1. Moreover, the inspection of the processor usage while the tests are performed confirms that only one core was actively running. Consequently, further research and investigation efforts defined the third scenario, which aims to find a proper solution that effectively makes use of the available multiple cores.

### Scenario 3

Spark was configured to run with the same configuration as in Scenario 2, but instead of one parquet file, multiple parquet files were used as input. This resulted in Spark running jobs on multiple cores. While it may sound strange, initially it was just a trial-and-error subject, but later it turned out that Spark cannot run jobs in parallel on a monolithic parquet file. The author of [36] also specifies that this behavior only happens when the data is on the local system and not on a HDFS (Hadoop Distributed File System). Due to this shortcoming, some exploration was done on splitting data, in the following ways:

- based on the defect_type feature, which resulted in 4 splits, which in turn allowed only 4 parallel jobs at a time, while the available cores were 12
- by grouping the speed in discrete values, and split by speed, with the disadvantage of losing information (regarding speed)

- introducing a new column which would contain the remainder of dividing the index by 12, then split based on this new column, with the disadvantage of having an extra column
- using the maxRecordsPerFile parameter like in Algorithm 6 from the thesis, available from Spark 2.2+

Considering also the cases where more tasks would access the same file, the last solution was used for splitting the data into not 12, but 20 chunks to avoid potential concurrent file access.

The results are outlined in Table 2, and depict that Spark now indeed runs multi-cored, outperforming the speed of Spark Scenario 1 five times, and also matching the speed of scikit-learn. The noticeable difference in accuracy between scikit-learn and Spark is due to the different solvers used by the two, but it is the closest comparison that can be achieved by the default implementation of Spark.

Table 2: Comparison of the Spark Scenario 2, Spark Scenario 3 and scikit-learn.

| | *data100* | | *data1k* | | *data10k* | | *data100k* | |
|---|---|---|---|---|---|---|---|---|
| | **Acc.** | **Time** | **Acc.** | **Time** | **Acc.** | **Time** | **Acc.** | **Time** |
| **Spark S2** | 0.36 | 0:11:00 | 0.33 | 1:34:02 | 0.35 | 17:03:22 | 0.35 | 176:29:48 |
| **Spark S3** | 0.354 | 0:02:36 | 0.354 | 0:18:37 | 0.308 | 3:43:26 | 0.352 | 36:43:53 |
| **Scikit-learn** | 0.53 | 0:02:41 | 0.63 | 0:18:18 | 0.64 | 2:42:00 | N/A | N/A |

### Scaling with Spark

As demonstrated in the previous subsection, Spark can run as fast as scikit-learn on a single machine, utilizing the full power of the CPU. The next step is to try to get a new speed record in terms of training time. Along with this, another target is to break the barrier of how much data can be processed. Both of these objectives could theoretically be achieved by increasing the processing power of Spark, with the addition of new computers (nodes) to the cluster.

The cluster for the following tests was deployed considering seven workstations with the same hardware configuration. Thus, they use hard disk drives for the local storage, 16 GB of RAM memory, and eight CPU cores. The workstations are connected physically to a switch with a 100 Mpbs link speed. One computer was configured to host the manager and driver, and the rest as worker machines. Each worker was configured to spawn two executors with 3 GB of RAM memory, and 4 CPU cores. The results from Table 3 show that, considering this new setup, Spark proves to be slower for small datasets than the setup from Scenario 3, but things get progressively better as the size of the data grows, considering a logarithmic curve. The table also shows that the training on the whole dataset is possible using the integrated data analytics system on multiple machines, and it takes approximately 45 hours.

Table 3: Comparison of a seven node Spark cluster with Scenario 3.

| | *data100* | | *data1k* | | *data10k* | | *data100k* | | *all* | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Acc.** | **Time** | **Acc.** | **Time** | **Acc.** | **Time** | **Acc.** | **Time** | **Acc.** | **Time** |
| **Spark S3** | 0.354 | 0:02:36 | 0.354 | 0:18:37 | 0.308 | 3:43:26 | 0.352 | 36:43:53 | N/A | N/A |
| **Spark cluster** | 0.35 | 0:09:17 | 0.35 | 0:18:14 | 0.3 | 1:17:45 | 0.346 | 9:49:37 | 0.352 | 44:29:34 |

## 5.4 Conclusions and Open Problems

The contribution that is presented in this chapter is significant in several respects. It describes an integrated data analytics system, which is capable of fully processing large tabular datasets using a multinode cluster setup. This achievement is significant considering that many of the existing similar approaches are able to process only subsets of similarly large datasets.

The current version of the system is designed in order to optimize the training phase of the data analytics process. Furthermore, it is important to note that the architecture of the data analytics system allows for relevant data processing routines to be re-engineered. This allows for the future optimized iterations of the system to be implemented in an efficient way, so that useful improvements, such as a stronger consideration of accuracy, will become available. The software system that was considered in order to conduct the research is available at this address: https://github.com/akerestely/hpc-hadoop-spark.

Although many of the questions raised in the introduction were at least partially answered throughout the chapter, a summary is presented here as well, for quick referencing.

- *How much is big data?* - Depending on the used processing framework, data that exceeds a typical end-user RAM memory (≈ 32 GB) should already be considered as being big data.
- *Is Spark good enough for processing big data?* - Spark turned out to be good for processing the tabular data available for this research with Artificial Neural Networks, the system responded well, when adding more and more data, so it's safe to assume that it could possibly handle any amount of data. Yet for other data types or other machine learning algorithms this could change.
- *Is Spark a scalable machine learning tool?* - By design, Spark splits a problem into multiple smaller problems, thus can handle growing data, of course at the cost of slower computation, as the results from those small problems need to be merged at some point, which in turn also requires computing power. On the other hand, the system can be extended by additional worker nodes, which provides more computing power, thus resulting in faster processing. All these statements were proved by the experiments from this chapter.
- *Can the memory restrictions be solved by using the Spark framework?* - Because Spark is designed to split a problem into multiple smaller problems, it can use as much

memory as it's available on the particular worker, thus the memory restrictions available in other frameworks can be solved by using Spark.

- *Does big data processing require a different mindset?* - Processing big data indeed requires a different mindset, one in which the focus is on parallelization. Although most of the parallelization is already done and available when using the existing machine learning algorithms, the data that is fed to the system still needs to be split beforehand, to enable the parallel processing.
- *Why and when should one use the Spark framework for machine learning?* - First of all, if a Spark system is up and available, experiments showed that data exceeding 10 MB can already be handled faster with Spark than scikit-learn. On the other hand, if a Spark system is not available out of the box, setting up one is not straightforward, so the recommendation of this research is to only start setting up and using Spark if other methods failed to yield results.
- *Could smaller datasets benefit from the power of Spark or is it proper just for big datasets?* - In the case of the dataset type, machine learning algorithm, and cluster setup used in this research, datasets exceeding 10 MB can already benefit from the power of Spark, so even smaller datasets can benefit from the power of Spark.

In support of this chapter the following research articles were published:

- *High Performance Computing for Machine Learning*, by Árpád Kerestély, in Bulletin of the Transilvania University of Brasov, 2020. This paper aims to unveil how Machine Learning and High Performance Computing are benefitting from each other, thus being the basis for the review part of the chapter.
- *A Research Study on Running Machine Learning Algorithms on Big Data with Spark*, by Árpád Kerestély, Alexandra Băicoianu and Răzvan Bocu, in Proceedings of the 14th International Conference on Knowledge Science, Engineering and Management (KSEM), 2021. This paper describes an integrated machine learning-based data analytics system, which processes large amounts of data, thus being the basis for the experimental part and comparative analysis of the chapter.

## Bibliography

[1] Martín Abadi et al. "Tensorflow: A system for large-scale machine learning". 12th USENIX symposium on operating systems design and implementation (OSDI 16). 2016, pp. 265–283.

[2] Pedro Henriques Abreu et al. "Predicting Breast Cancer Recurrence Using Machine Learning Techniques: A Systematic Review". In: (2016).

[3] Abien Fred M Agarap. "On breast cancer detection: an application of machine learning algorithms on the wisconsin diagnostic dataset". In: Proceedings of the 2nd International Conference on Machine Learning and Soft Computing. 2018, pp. 5–9.

[4] Bassam Al-Shargabi and Fida'a Al-Shami. "An experimental study for breast cancer prediction algorithms". In:Proceedings of the Second International Conference on Data Science, E-Learning and Information Systems. 2019, pp. 1–6.

[5] R Almufti et al. "A critical review of the analytical approaches for circulating tumor biomarker kinetics during treatment". In: Annals of oncology 25.1 (2014), pp. 41–56.

[6] G Ammar, W Dayawansa, and C Martin. "Exponential interpolation: theory and numerical algorithms". In: Applied Mathematics and Computation 41.3 (1991), pp. 189–232.

[7] Apache Flume. URL: https://flume.apache.org/ (visited on 06/17/2021).

[8] Apache Hadoop. URL: http://hadoop.apache.org/ (visited on 06/17/2021).

[9] Apache HBase. URL: https://hbase.apache.org/ (visited on 06/17/2021).

[10] Apache Hive. URL: https://hive.apache.org/ (visited on 06/17/2021).

[11] Apache Spark. URL: https://spark.apache.org/ (visited on 06/17/2021).

[12] Apache Sqoop. URL: https://sqoop.apache.org/ (visited on 06/17/2021).

[13] Apache Storm. URL: https://hortonworks.com/apache/storm/ (visited on 06/17/2021).

[14] J Archenaa and EA Mary Anita. "A survey of big data analytics in healthcare and government". In: Procedia Computer Science 50 (2015), pp. 408–413.

[15] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling". In: arXiv preprint arXiv:1803.01271 (2018).

[16] Alexandra Baicoianu and Andreea Mathe. "Diagnose Bearing Failures With Machine Learning Models". In: 2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA). IEEE. 2021, pp. 1–6.

[17] Ross S Berkowitz and Donald P Goldstein. "Molar pregnancy". In:New England Journal of Medicine 360.16 (2009), pp. 1639–1645.

[18] Declan Butler. "When Google got flu wrong". In: Nature News 494.7436 (2013), p. 155.

[19] Antonio Cachuan. A gentle introduction to Apache Arrow with Apache Spark and Pandas. 2019. (Visited on 06/01/2021).

[20] Cancer databases. URL: https://public.opendatasoft.com/explore/dataset/cancer-databases/table/.

[21] Enrique Castillo and Ali S Hadi. "Functional networks". In: Wiley StatsRef: Statistics Reference Online (2014).

[22] Fay Chang et al. "Bigtable: A distributed storage system for structured data". In: ACM Transactions on Computer Systems (TOCS) 26.2 (2008), pp. 1–26.

[23] Min Chen et al. "Disease prediction by machine learning over big data from healthcare communities". In: Ieee Access 5 (2017), pp. 8869–8879.

[24] Yunji Chen et al. "Dadiannao: A machine-learning supercomputer". In:2014 47th Annual IEEE/ACM International Symposium on Microarchitecture. IEEE. 2014, pp. 609–622.

[25] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: arXiv preprint arXiv:1406.1078 (2014).

[26] Edward Choi et al. "Doctor ai: Predicting clinical events via recurrent neural networks". In: Machine learning for healthcare conference. PMLR. 2016, pp. 301–318.

[27] François Chollet et al. Keras. https://keras.io. 2015.

[28] Niamh Clarke et al. "GDPR: an impediment to research?" In: Irish Journal of Medical Science (1971-) 188.4 (2019), pp. 1129–1135.

[29] Adam Coates et al. "Deep learning with COTS HPC systems". In:International conference on machine learning. PMLR. 2013, pp. 1337–1345.

[30] European Commission.Regulation EU 2016/679 of the European Parliament and of the Council. 2016. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679 (visited on 07/2021).

[31] Catherine Costigan, Sabin Tabirca, and John Coulter. "Mathematically Modelling hCG in Women with Gestational Trophoblastic Disease Using Logarithmic Transformations". In: 2016 UKSim-AMSS 18th International Conference on Computer Modelling and Simulation (UKSim). IEEE. 2016, pp. 55–59.

[32] Karen Daniels et al. "Properties of normalized radial visualizations". In:Information Visualization 11.4 (2012), pp. 273–300.

[33] Saptarshi Das and Koushik Maharatna. "Machine learning techniques for remote healthcare". In: Systems Design for Remote Healthcare. Springer, 2014, pp. 129–172.

[34] Databricks. Parquet files. 2020. URL: https://docs.databricks.com/data/data-sources/readparquet.html (visited on 02/2020).

[35] data.world. URL: https://data.world/datasets/cancer.

[36] Colin Davis. Big Data on a Laptop: Tools and Strategies - Part 3. Ed. by POP Tech. 2018. URL: https://tech.popdata.org/big-data-on-a-laptop-tools-and-strategies-part-3/ (visited on 06/01/2021).

[37] Jeffrey Dean et al. "Large Scale Distributed Deep Networks". In: NIPS. 2012.

[38] Michael Driscoll. Winning with Big Data: Secrets of the Successful Data Scientist. Ed. by Inc. O'Reilly Media. 2010. URL: https://conferences.oreilly.com/datascience/public/schedule/detail/15316.

[39] JC Ehiwario and SO Aghamie. "Comparative study of bisection, Newton-Raphson and secant methods of root-finding problems". In: IOSR Journal of Engineering 4.04 (2014), pp. 01–07.

[40] Emad Elsebakhi et al. "Large-scale machine learning based on functional networks for biomedical big data with high performance computing platforms". In: Journal of Computational Science 11 (2015), pp. 69–81.

[41] Heiko Enderling and Mark AJ Chaplain. "Mathematical Modeling of Tumor Growth and Treatment". In: Current pharmaceutical design 20.30 (2014), pp. 4934–4940.

[42] George Elmer Forsythe. "Computer methods for mathematical computations." In: Prentice-Hall series in automatic computation 259 (1977).

[43] C Freitas et al. "Comparison of vibration and acoustic measurements for detection of bearing defects". In:International Conference on Noise and Vibration Engineering 2016 and International Conference on Uncertainty in Structural Dynamics 2016. Vol. 1. 2016.

[44] Henri Gavin. “The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems”. In: Department of Civil and Environmental Engineering, Duke University 28 (2011), pp. 1–5.

[45] Marzyeh Ghassemi, Leo Anthony Celi, and David J Stone. ”State of the art review: the data revolution in critical care“. In: Critical Care 19.1 (2015), pp. 1–9.

[46] Dan Gillick, Arlo Faria, and John DeNero. ”Mapreduce: Distributed computing for machine learning”. In: Berkley, Dec 18 (2006).

[47] GLOBOCAN (Global Cancer Observatory). Cancer Statistics – Romania. May 2020. URL: https://gco.iarc.fr/today/data/factsheets/populations/642-romania-fact-sheets.pdf.

[48] Isabelle Guyon et al. “Gene selection for cancer classification using support vector machines”. In: Machine learning 46.1 (2002), pp. 389–422.

[49] Sepp Hochreiter and Jürgen Schmidhuber. ”Long short-term memory”. In: Neural computation 9.8 (1997), pp. 1735–1780.

[50] Patrick Hoffman et al. “DNA visual and analytic data mining”. In: Proceedings. Visualization’97 (Cat. No. 97CB36155). IEEE. 1997, pp. 437–441.

[51] Shiuh-Jer Huang and Chien-Lo Huang. “Control of an inverted pendulum using grey prediction model“. In: Industry Applications, IEEE Transactions on 36.2 (2000), pp. 452–458.

[52] Introduction to High-Performance Machine learning @SURFsara. 2018. URL: https://events.prace-ri.eu/event/693/attachments/626/ (visited on 06/17/2021).

[53] Tado Juric. “Google Trends as a method to predict new COVID-19 cases“. In: medRxiv (2021).

[54] Marcel Adam Just et al. “Machine learning of neural representations of suicide and emotion concepts identifies suicidal youth“. In: Nature human behaviour 1.12 (2017), pp. 911–919.

[55] Kadupitiya Kadupitige. Intersection of HPC and Machine Learning. Indiana University Bloomington, 2017. URL: http://dsc.soic.indiana.edu/publications/ENGR-E%20687%20_%20IND%20STUDY%20INTEL%20SYS%20_%20Intersection%20of%20HPC%20and%20machine%20learning.pdf (visited on 6/17/2021).

[56] Holden Karau and Rachel Warren. High performance Spark: best practices for scaling and optimizing Apache Spark. “O’Reilly Media, Inc.”, 2017.

[57] Freund Karl. What’s Hot At SC17: The Synthesis Of Machine Learning & HPC. 2017. URL: https://www.forbes.com/sites/moorinsights/2017/11/14/whats-hot-at-sc17-the-synthesis-ofmachine-learning-hpc/#2ef32b2759a7 (visited on 06/17/2021).

[58] Arpad Kerestely. “Feature Inspection and Elimination in the Context of Breast Cancer Prediction“. In: Proceedings of the 36th International Business Information Management Association (IBIMA). Granada, Spain, 2020, pp. 13487–13493. ISBN: 978-0-9998551-5-7.

[59] Arpad Kerestely. ”High Performance Computing for Machine Learning“. In: Bulletin of the Transilvania University of Brasov. Mathematics, Informatics, Physics. Series III 13.2 (2020), pp. 705–714.

[60] Arpad Kerestely, Alexandra Baicoianu, and Razvan Bocu. ”A Research Study on Running Machine Learning Algorithms on Big Data with Spark“. In: Proceedings of the 14th International Conference on Knowledge Science, Engineering and Management (KSEM 2021). Tokyo, Japan: Springer, 2021, pp. 307–318.

[61] Arpad Kerestely, Catherine Costigan, and Sabin Tabirca. "Vertically Shifted Exponential Best-Fit". In: Proceedings of the 35th International Business Information Management Association (IBIMA). Seville, Spain, 2020, pp. 13855–13868. ISBN: 978-0-9998551-4-0.

[62] Arpad Kerestely, Lucian Mircea Sasu, and Marius Sabin Tabirca. "Machine Learning in Healthcare: An Overview". In: Bulletin of the Transilvania University of Brasov. Mathematics, Informatics, Physics. Series III 11.2 (2018), pp. 273–278.

[63] Arpad Kerestely et al. "Theoretical Study of Exponential Best-Fit: Modeling hCG for Gestational Trophoblastic Disease". In: Proceedings of the 14th International Conference on Knowledge Science, Engineering and Management (KSEM 2021). Tokyo, Japan: Springer, 2021, pp. 426–438.

[64] Konstantina Kourou et al. "Machine learning applications in cancer prognosis and prediction". In: Computational and structural biotechnology journal 13 (2015), pp. 8–17.

[65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: Advances in neural information processing systems 25 (2012), pp. 1097–1105.

[66] Andrew Kusiak. "Smart manufacturing". In: International Journal of Production Research 56.1-2 (2018), pp. 508–517.

[67] Prasanth Lade, Rumi Ghosh, and Soundar Srinivasan. "Manufacturing analytics and industrial internet of things". In: IEEE Intelligent Systems 32.3 (2017), pp. 74–79.

[68] Colin Lea et al. "Temporal convolutional networks: A unified approach to action segmentation". In: European Conference on Computer Vision. Springer. 2016, pp. 47–54.

[69] Megan Lenhart. "Diagnosis and treatment of molar pregnancy". In: Topics in Obstetrics & Gynecology 27.17 (2007), pp. 1–4.

[70] John Levesque and Aaron Vose. Programming for Hybrid Multi/Manycore MPP Systems. CRC Press, 2017.

[71] Zachary C Lipton, David Kale, and Randall Wetzel. "Directly modeling missing data in sequences with rnns: Improved classification of clinical time series". In:Machine learning for healthcare conference. PMLR. 2016, pp. 253–270.

[72] Pek Y Lum et al. "Extracting insights from the shape of complex data using topology". In:Scientific reports 3 (2013), p. 1236.

[73] Mehran Mozaffari-Kermani et al. "Systematic poisoning attacks on and defenses for machine learning in healthcare". In:IEEE journal of biomedical and health informatics 19.6 (2014), pp. 1893–1905.

[74] Abhinav Nagpal and Goldie Gabrani. "Python for data analytics, scientific and technical applications". In: 2019 Amity international conference on artificial intelligence (AICAI). IEEE. 2019, pp. 140–145.

[75] Open Datasets and Machine Learning Projects. URL: https://www.kaggle.com/datasets.

[76] Patison Palee et al. "Image analysis of histological features in molar pregnancies". In: Expert systems with applications 40.17 (2013), pp. 7151–7158.

[77] Patison Palee et al. "Heuristic neural network approach in histological sections detection of hydatidiform mole". In: Journal of Medical Imaging 6.4 (2019), p. 044501.

[78] Ramprasad Pedapatnam. Understanding Resource Allocation configurations for a Spark application. Ed. by Clairvoyant. 2016. URL: http://site.clairvoyantsoft.com/understanding-resourceallocation-configurations-spark-application/ (visited on 06/01/2021).

[79] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: Journal of Machine Learning Research 12 (2011), pp. 2825–2830.

[80] George M Phillips. Interpolation and approximation by polynomials. Vol. 14. Springer Science & Business Media, 2003.

[81] Narendra Pisal, John Tidy, and Barry Hancock. "Gestational trophoblastic disease: is intensive follow up essential in all women?" In:BJOG: An International Journal of Obstetrics & Gynaecology 111.12 (2004), pp. 1449–1451.

[82] Juliana T Pollettini et al. "Using machine learning classifiers to assist healthcare-related decisions: classification of electronic patient records". In: Journal of medical systems 36.6 (2012), pp. 3861–3874.

[83] William H Press et al. Numerical recipes 3rd edition: The art of scientific computing. Cambridge university press, 2007.

[84] Devi Ramanan. NKI Breast Cancer Data. 2017. URL: https://data.world/deviramanan2016/nkibreast-cancer-data.

[85] Gesine Richter et al. "Patient views on research use of clinical data without consent: Legal, but also acceptable?" In: European Journal of Human Genetics 27.6 (2019), pp. 841–847.

[86] Mark R Schoeberl. "A model for the behavior of β-hCG after evacuation of hydatidiform moles". In: Gynecologic oncology 105.3 (2007), pp. 776–779.

[87] scipy.optimize.curve_fit. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html (visited on 08/2019).

[88] Michael J Seckl, Neil J Sebire, and Ross S Berkowitz. "Gestational trophoblastic disease". In: The Lancet 376.9742 (2010), pp. 717–729.

[89] Rebecca L. Siegel et al. "Cancer statistics, 2021". In:CA: A Cancer Journal for Clinicians 71.1 (2021), pp. 7–33. URL: https://acsjournals.onlinelibrary.wiley.com/doi/10.3322/caac.21654.

[90] John T Soper. "Gestational trophoblastic disease". In: Obstetrics & Gynecology 108.1 (2006), pp. 176–187.

[91] Apache Spark. PySpark Usage Guide for Pandas with Apache Arrow. 2020. URL: https://spark.apache.org/docs (visited on 02/2020).

[92] Ferenc Szidarovszky and Sidney J Yakowitz. Principles and procedures of numerical analysis. Vol. 14. Springer, 2013.

[93] UCI Machine Learning Repository: Data Sets. URL: https://archive.ics.uci.edu/ml/datasets.php.

[94] National Health Service UK. Molar pregnancy. 2020. URL: https://www.nhs.uk/conditions/molar-pregnancy/.

[95] Case Western Reserve University. The Case Western Reserve University Bearing Data Center Website. 2020. URL: https://csegroups.case.edu/bearingdatacenter (visited on 06/2020).

[96] National Cancer Institute USA. Gestational Trophoblastic Disease Treatment. 2020. URL: https://www.cancer.gov/types/gestational-trophoblastic/patient/gtd-treatment-pdq.

[97] NE Van Trommel et al. "Early identification of persistent trophoblastic disease with serum hCG concentration ratios". In:International Journal of Gynecological Cancer 18.2 (2008), pp. 318–323.

[98] Laura J Van't Veer et al. "Gene expression profiling predicts clinical outcome of breast cancer". In: nature 415.6871 (2002), pp. 530–536.

[99] Ashish Vaswani et al. "Attention is all you need". In: arXiv preprint arXiv:1706.03762 (2017).

[100] What is high performance computing? URL: https://insidehpc.com/hpc-basic-training/whatis-hpc/ (visited on 06/17/2021).

[101] William Wolberg, W. Street, and Olvi Mangasarian. Breast Cancer Wisconsin (Diagnostic) Data Set. UCI Machine Learning Repository. 1995. URL: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic).

[102] Xiao Xiao et al. "On the use of log-transformation vs. nonlinear regression for analyzing biological power laws". In: Ecology 92.10 (2011), pp. 1887–1894.

[103] B You et al. "Predictive values of hCG clearance for risk of methotrexate resistance in low-risk gestational trophoblastic neoplasias". In: Annals of oncology 21.8 (2010), pp. 1643–1650.

[104] B You et al. "Early prediction of treatment resistance in low-risk gestational trophoblastic neoplasia using population kinetic modelling of hCG measurements". In: British journal of cancer 108.9 (2013), pp. 1810–1816.

[105] Tracey Young et al. "Predicting gestational trophoblastic neoplasia (GTN): is urine hCG the answer?" In: Gynecologic oncology 122.3 (2011), pp. 595–599.

[106] S. Zhang et al. "Machine learning and deep learning algorithms for bearing fault diagnostics-a comprehensive review". In: arXiv preprints arXiv:1901.08247 (2019).

[107] MH Zwietering et al. "Modeling of the bacterial growth curve". In: Applied and environmental microbiology 56.6 (1990), pp. 1875–1881.

[108] Matjaz Zwitter and Milan Soklic.Breast Cancer Data Set. UCI Machine Learning Repository. 1988. URL: https://archive.ics.uci.edu/ml/datasets/breast+cancer.