

UNIVERSITY OF KENT

ACCESS TO A MASTER'S DEGREE OR POSTGRADUATE DIPLOMA DISSERTATION

In accordance with the Regulations, I hereby confirm that I shall permit general access to my dissertation at the discretion of the University Librarian. I agree that copies of my dissertation may be made for Libraries and research workers on the understanding that no publication in any form is made of the contents without my permission.

Notes for Candidates

- 1 Where the examiners consider the dissertation to be of distinction standard, one copy will be deposited in the University Library.
- 2 The copy sent to the Library becomes the property of the University Library. the copyright in its contents remains with the candidate. A duplicated sheet is pasted into the front of every thesis or dissertation deposited in the Library. The wording on the sheet is:

"I undertake not to use any matter contained in this thesis for publication in any form without the prior knowledge of the author."

Every reader of the dissertation must sign and date this sheet.

- 3 The University has the right to publish the title of the dissertation and the abstract and to authorise others to do so.

.....
SIGNATURE



DATE: 12/09/2021

.....
FULL NAMES: Provost Simon

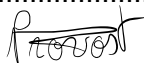
(Please print, underlining surname, in order to assist the cataloguing of theses deposited in the Library.)

CERTIFICATE ON SUBMISSION OF DISSERTATION

I certify that:

- 1 I have read the University Degree Regulations under which this submission is made;
- 2 In so far as the dissertation involves any collaborative research, the extent of this collaboration has been clearly indicated; and that any material which has been previously presented and accepted for the award of an academic qualification at this University or elsewhere has also been clearly identified in the dissertation.

.....
SIGNATURE



DATE: 12/09/2021

This form should be completed and submitted with your dissertation to the Course Administration Office Room S132 School of Computing

X:\proj\cas\MSc - General\Information sheets for MSc results\Access to a Masters or PG Diploma.doc

Machine Learning Applied to Medical data

Simon Provost - sgp28@kent.ac.uk
Supervised by: Prof. Alex A. Freitas.

University of Kent - School of Computing
MSc Advanced Computer Science

Word count: 10385

2020/2021

Declaration

I hereby declare that this thesis is my original work, completed after registering for the MSc computer science degree at the University of Kent's school of computing, and has not previously been included in a thesis or dissertation submitted to this or any other institution for a degree, diploma, or other qualification. Additionally, all sentences or passages from other people's work that appear in this report have been explicitly acknowledged through clear cross-referencing to the author's work. Any illustrations that are not the author's own work have been used with the author's explicit permission and are specifically acknowledged. I understand that failing to do so constitutes plagiarism and will be grounds for dismissal from this project and the degree examination in general.

Provost Simon.

Acknowledgments

Without the help of numerous individuals, this project would not have been possible. My heartfelt appreciation to my research adviser, Prof Alex. A. Freitas, who read my numerous report and helped me make sense of the confusion, guided and challenged me throughout this endeavour, and was extremely prompt from start to finish. I would like to express my gratitude to Dr. David TIGHE, the research supervisor's NHS contact, for his assistance with medical knowledge and for providing real-world data that was instrumental in testing our solution.

I am grateful to the University of Kent for providing me with the opportunity to work on such an interesting project, which has resulted in me being prepared to continue working on this subject next year, and hopefully let me candidate for a doctoral program, most likely in a subject related to the one covered in this thesis. Many thanks to the University of Kent's CEMS support for complying with my request regarding the covid-19, which resulted in me being granted a small extension to complete my thesis on time. Finally, I would like to express my gratitude to my university research fellow friends, family, and close individuals who supported and were provided love throughout this lengthy process.

Abstract

Machine learning (ML) has long enthralled and intrigued a large number of domain in the history of mankind. The basic motivation for machine learning is to ascertain the foundations of the future. Recently, approaches for automated machine learning (Auto-ML) have been presented to address the tedious chore of manually creating a machine learning model. Auto-ML tools automatically select the appropriate classification workflow for a particular dataset. If you are not an expert in machine learning but need its assistance for a project in order to save time and ensure accuracy, Auto-ML is the ideal answer for you because it does not require a senior level of competence in the field to do the analysis. Clinical settings lack machine learning knowledge; nonetheless, auto-ML is one solution to this problem; however, with the enormous number of frameworks available on the market, which one should be used and how is it superior to another? Auto-Sklearn and Auto-WEKA are two prominent and powerful frameworks for Automated Machine Learning; testing medical datasets on them would allow us to determine if Sklearn or Weka is more suitable for clinical use. Additionally, this study used not only online data but also real-world data which are provided by David Tighe, a physician and facial surgeon at EAST KENT HOSPITALS UNIVERSITY NHS FOUNDATION TRUST in the United Kingdom.

Contents

1	Introduction	7
1.1	Objectives	7
1.2	Overview of the contents	8
2	Review of the Literature	10
2.1	Introduction to Machine Learning	10
2.1.1	Define	10
2.1.2	Machine Learning's Advantages and Disadvantages	11
2.2	Auto Machine learning	11
2.2.1	Define	12
2.2.2	Combined Algorithm and hyper-parameter Selection (CASH) and Hyper-parameter optimisation(HPO)	13
2.2.3	Sequential Model-based Algorithm Configuration (SMAC)	13
2.2.4	Neural Architecture Search (NAS) and Architecture Optimisation (AO)	14
2.2.5	Advanced Techniques	15
2.2.6	Auto-Machine Learning's Advantages and Disadvantages	15
2.3	Auto-WEKA (Framework of Auto-ML)	16
2.3.1	Awarded prizes	16
2.3.2	Pre-Configured Algorithms	17
2.4	Auto-Sklearn (Framework of Auto-ML)	17
2.4.1	Awarded prizes	17
2.4.2	Pre-Configured Algorithms	17
3	Methodology and Datasets	19
3.1	Proposed methodology for frameworks comparison	19
3.1.1	Orientation	20
3.1.2	Frameworks chosen	20
3.1.3	Selected computational characteristics (Cloud server)	20
3.2	Collections of data	21
3.2.1	Online datasets acquired for experiments	21
3.2.2	Real-world data acquired for experiments	23

4	Computational operations & Framework experiments	26
4.1	Implementation of a computational program	26
4.1.1	Orientation	26
4.1.2	Wrapper for Auto-Sklearn/Sci-kit learn using Python	27
4.1.3	10-fold cross validation program in Python	27
4.1.4	Utilisation of the Bash script technology to centralise the analysis of a dataset	32
4.2	10-fold cross validation experiment	32
4.2.1	Auto-Sklearn Selected version and its characteristics	32
4.2.2	Auto-WEKA Selected version and its characteristics	34
4.2.3	Auto-Weka vs. Auto-Sklearn on online-dataset	34
4.2.4	Auto-Weka vs. Auto-Sklearn on real-world dataset	37
5	Conclusions	39
5.1	Discussion and evaluation of the usage of Auto Machine-Learning frameworks using medical data	39
5.2	Conclusion	42
5.3	Future Work	42

List of Figures

2.1	Machine learning published papers between HPO and NAS	14
5.1	Pipeline profiler - Plugin to see the output of Auto-Sklearn and the process of the black-box	41

List of Tables

3.1	Google Cloud virtual machine instance characteristics	21
3.2	Online datasets used for the experiments	23
3.3	Real-world datasets used for the experiments	25
4.1	Auto-Weka's pipeline parameters (10-fold cross validation experiment)	33
4.2	Auto-Weka's pipeline parameters (10-fold cross validation experiment)	34
4.3	Auto-Sklearn versus Auto-Weka on the average of the 10 fold-cross validation process (online-dataset)	35
4.4	Auto-Sklearn versus Auto-Weka on the average of the 10 fold-cross validation process (real-world dataset)	37

Chapter 1

Introduction

The world of machine learning (ML) has always captivated and intrigued a lot of scientists as well as people from any domain, any background. Artificial Intelligence has exploded in popularity over the last decade. Technology has had an effect on nearly every aspect of our lives. The primary motivation for machine learning is to discover what the future will be built upon; however, the primary challenge is determining where to begin. Given that, ‘The most accurate method of forecasting the future is to examine the past, or prognosticate.’ [1]. Machine learning can easily be defined such as ‘Machine learning is the science (and art) of programming computers so they can learn from data’ [2]. However, with the significant growing number of pre-processing and classification algorithms, manually selecting the optimal algorithm and hyper-parameter settings ¹ is an unsettling task. Recently, methods for automating machine learning (Auto-ML) have been proposed to address this issue. Auto-ML tools are designed to choose the optimal classification workflow for a given dataset automatically. Additionally, a quick advantage against Machine learning only is that with ML, scientists do not exhaust all possibilities offered to them, while, auto-machine learning exhausts all available possibilities for a given dataset and outputs the most accurate model based on a chosen metric.

1.1 Objectives

First and foremost, medical data is extremely valuable and with it, we have the potential to save lives. However, physicians already face a great deal on a daily basis and frequently lack expertise in machine learning to handle their data faster. Nonetheless, the world of machine learning opened up a plethora of possibilities for the clinical world. Unfortunately, one of the most time-consuming tasks for those physicians is using the numerous machine learning libraries available today on the market. Given that, due to the ease with which Auto-ML works, it may be a fantastic idea for them to use it on their data. Nonetheless, given the growing number of

¹Given a dataset, the most accurate combination of parameters for a given algorithm.

libraries in the field, how do we decide which one to choose and based on what arguments?

Thus, the following research is divided into two major sections. To begin, the objective is to conduct a thorough analysis/comparison of one of the two most popular Auto-ML frameworks available today, and then use the results to determine which framework performs the best on medical data. The results of this first section will already inform medical physicians about the framework they should use to classify their data and take advantage of all the benefits that Machine Learning offers in general. In terms of framework analysis, to assure fair comparison we have to objectively use the same computer characteristics for both frameworks, such as a Google Cloud server. The data used in both frameworks would be identical, and would come from a secure location where the data would be well anonymize and clearly presented, and would not come from the real world at this point of the analysis. Additionally, the data must come from the healthcare world, preferably from a variety of subfields within the healthcare field, in order to cover a broad range of medical subfields.

The second part of the research would involve working with real-world data from an NHS hospital in England, demonstrating that the framework chosen as the most accurate in comparison to others for medical data would easily aid in classifying such data from the real world, but could be slightly more complicated in terms of formatting the data in order to be well-prepared for machine learning. The findings from this section of the research would easily demonstrate that the framework is worth employing.

Auto-ML is new and seems to be a powerful tool to use, the ultimate goal of this research is to inform physicians about which frameworks, out of two popular ones, are the most important to use, and would assure them a high degree of accuracy which is very needed in clinical settings. Finally, with regards to the technical aspects of the objectives, the use of Python, Sci-kit Learn, Auto-Sklearn, Weka, and Auto-WEKA will be required in order to compare, analyse, and prove statements as well as desired outcomes in order to broaden the field of machine learning in clinical settings.

1.2 Overview of the contents

The following paragraphs will discuss the entire process we used to arrive at the framework analysis's conclusion, as well as the pipeline, from the data acquisition to the preparation and analysis. The thesis is divided into five distinct chapters that include (1) an introduction and explanation of our objectives, (2) a review of the literature in the domain we are researching, (3) the methodology proposed as well as detailed descriptions of the datasets we used throughout the research, (4) the computational aspects of our research, and finally, (5) a discussion and recommendations for future work. Each chapter includes some subsequent sections that delve deeper into why we chose this option over others, how we arrived at this stage of the analysis,

and how we concluded to this type of recommendation. Finally, some programming examples of what we have created will be included in several of the subsequent sections in the penultimate chapter. Which leads to a conclusion with an assessment of the use of Auto-ML with medical data and future endeavours that could be undertaken as a result of this thesis.

Chapter 2

Review of the Literature

2.1 Introduction to Machine Learning

The field of machine learning emerged in the 1940's/50's [3, 4, 5, 6, 7] respectively. Machine learning (ML) is a subset of Artificial Intelligence (AI) that enables software applications to improve their predictive accuracy without being explicitly programmed to do so. The idea behind machine learning algorithms is to use previously known data to train a machine (i.e., a model) to predict the output in relation to the data provided, allowing this model to be deployed in production, where it will be fed with unknown data and should output predicted values in a timely manner based on the previously known data training. Humans acquire knowledge through exposure to something, either through personal experience or through hearing about it from someone else. Machines acquire knowledge through shared experience from the past.

In 2021, machine learning seems to have appeared in many guises such as with the global pandemic COVID-19: a model for determining whether a subject has COVID-19 based on the subject's symptoms, according to [8]. In 2021, machine learning will be beneficial for image classification problems involving COVID-19 using chest x-ray or CT images, which can be used to determine whether a subject has COVID-19 or something worse, such as lung cancer symptoms, which can be a risk depending on metabolism [9]. However, the issue is that the majority of applications are created and experienced by hand, whereas the power of a computer could be beneficial in analysing all possibilities.

2.1.1 Define

While the world of machine learning is dense with terminology, there are two of them that are particularly useful and necessary to be defined in order to get into the field. From the book Efficient Learning Machines [10], 'Supervised Learning, or extracting correlations between independent attributes and a specific dependent attribute using machine learning techniques. Through the consumption of incoming data and output values, supervised learning develops

a prediction model using a training dataset. Given that, the model can be used to forecast the output values for a fresh new dataset. Model performance is dependent on the size and variance of the training dataset in order to obtain higher generalisation and prediction power for fresh datasets. The majority of induction algorithms are classified as supervised learning algorithms. Unsupervised learning, or techniques for grouping instances that lack a predefined dependant attribute. Generally, this strategy entails discovering structured patterns in data by filtering out pure unstructured noise. Typically, techniques for clustering and dimensionality reduction are unsupervised¹.

2.1.2 Machine Learning's Advantages and Disadvantages

While machine learning is being heralded as the world of the future, it does have significant flaws that must be understood by the general public. To begin, machine learning is extremely powerful at identifying trends or patterns within a given dataset in a very short period of time, which is extremely useful in industry and medicine for instance. While the human brain is kept busy with more important tasks than data analysis, the computer will be able to traverse gigabytes of data quickly using intelligent methods that save time, which is something that people are looking for in particular. Another advantage of machine learning is that it is global in scope rather than local; the world is full of events that occur every day, and sometimes what occurs in our lives occurred three years ago in another region of the world, particularly in the world of healthcare; based on what other subjects had, you could predict outbreaks before they occur, which can save lives. Nonetheless, machine learning has flaws and is not magical; it is possible that machine learning is underdeveloped or that a mistake was made while coding the model; a biased or undersampled dataset can also cost a lot for an ML model, lowering its accuracy and usefulness in the real world. Given this, the requirements for human setup from beginning to end remain necessary. The final disadvantage of the world of machine learning is that it is not yet capable of being applied to two distinct fields concurrently; two distinct machine learning models must be thought of/developed; some similarities will be obvious, but not highly accurate models can be developed to be powerful as a charm in two distinct fields of our daily lives. A common example is the relationship between healthcare and manufacturing.

2.2 Auto Machine learning

The following are the fundamental processes that a machine learning expert follows when designing a machine learning classification algorithm: Pre-processing ¹, feature extraction ², and

¹To make databases usable for data mining, they must undergo preprocessing, which includes data cleansing, data reconstruction, data transformation, and many other methods otherwise machine would not easily pin-point pattern through the data. Also called Feature engineering.

²The dimension reduction technique that reduces an initial collection of raw data to more manageable groupings for better processing and easier way of pin-point patterns through the data.

feature selection methods³ may be necessary to alter data to conform to the machine learning algorithm's criteria. Following that, both method selection and hyperparameter adjustment are essential to ensure that their model achieves the best prediction performance feasible. As a result, non-experts or those unfamiliar with the discipline may find these processes time consuming, providing substantial hurdles to machine learning adoption. On the other hand, Auto Machine-Learning (AutoML) arises as a solution to this time-consuming and inefficient technique when confronted with an unknown situation. AutoML dates all the way back to 1995, when a private technology published by Unica Software [11] enabled automatic hyper parameter tuning for neural networks and, later, automated model selection across four different types of predictive models along a Pattern recognition software. AutoML has been in development for two decades and has been a hot topic since the release of the AUTO-WEKA tool [12], which is a fork of the well-known WEKA Machine Learning program [13]. The publication of the AUTO-WEKA tool and its extensive research on the subject using a large number of datasets demonstrated the efficacy of this technique.

2.2.1 Define

The concept of Auto Machine-Learning has been disputed across the literature, however, the following definition is sufficiently clear in terms of its description and long-term objective for both new and experienced researchers in the field:

From the book Machine learning [14], as described by E, T, and P, seeks to increase its performance on task T as evaluated by P given training data E. Historically, machine learning has placed a higher premium on creating and analysing learning tools than on their usability. AutoML's goal is to create high-level control methods for underlying learning technologies, allowing for the discovery of suitable configurations without human involvement. Thus, AutoML focuses on the simplicity of use of learning tools. Not only does AutoML seek excellent learning performance, but it also demands that performance be accomplished without human intervention. (Note: Adapted from [15]).

As a result of this definition, the area of automation that AutoML aims to address is data preparation, feature engineering (selection; extraction; meta learning; transfer learning; skew data detection), model selection⁴, hyperparameter optimization⁵, time and memory constraints

³To identify the optimal collection of characteristics that enables the construction of effective models of investigated events, sometimes it require to remove some feature from a dataset due to low predictive power of them.

⁴When we have a variety of models of different complexity (e.g., linear or logistic regression models with different degree polynomials, or KNN classifiers with different values of K), how should we pick the right one? [16], The model selection process depicts which algorithm will be chosen as the most accurate for a given training dataset given a training set.

⁵hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm.' [17].

⁶, evaluation metric selection ⁷, analysis of obtained results, and creation of a user interface for visualisation ⁸. AutoML intends to widen the applicability of machine learning by making it accessible to non-experts while overcoming severe obstacles imposed on such profiles by regular machine learning.

2.2.2 Combined Algorithm and hyper-parameter Selection (CASH) and Hyper-parameter optimisation(HPO)

The following is about classification algorithms in Machine Learning.

The CASH problem [12] is concerned with automatically and concurrently selecting a learning algorithm and its settings, whereas the HPO problem is concerned with providing the best feasible model instance from a vector of chosen algorithms. It is straightforward to discover the most accurate hyper-parameter combinations for a particular algorithm by iterating through all potential combinations and calculating their rate of error. When the HPO technique is combined with the CASH problem, all conceivable learning algorithms, as well as their possible hyper-parameter combinations, are encoded as hyper-parameters. Due to the CASH problem, the search space can become extremely wide, generating a significant number of solutions that would not have been achievable with human hands. However, it is computationally costly, and the research indicates that the computational budget can be constrained to maintain a fair level of performance.

The cash and HPO problem can be summarised as a huge number of hypotheses which are evaluated and their feasibility is determined, and the most interesting one is produced as the best predictive model for the training set provided. To illustrate a brief real-world example with vast possibilities, we will use the machine learning algorithm Random Forest, which has no fewer than ten factors to consider. Each parameter can take on average ten different values, which means that investigating the configuration space created by the CASH & HPO problems will require 1010 different combinations to give it a try. Now, tuning n algorithms with j hyper-parameters is expensive.

2.2.3 Sequential Model-based Algorithm Configuration (SMAC)

Sequential Model-Based algorithm configuration or SMAC [18, 19, 20], is a versatile tool for HPO whose primary goal is to free algorithm designers from the tedious chore of optimising hyper-parameters of an algorithm. The fundamental premise of this promising technique, state-of-the-art of configuration procedures[21], is that it constructs a promising configuration that may

⁶Finding the best algorithm, along with its hyper parameters, could take weeks. This issue is addressed using time and memory constraints, some quotas must be set.

⁷Because HPO tends to reduce the error rate of a metric chosen, the metric evaluation that will be used to assess a model's capacity and performance must be chosen (e.g, F1-score, Accuracy/Error rate, Area Under the Receiver Operating Characteristics score, etc.).

⁸A dashboard that allows the user to run or view the results of a specific model.

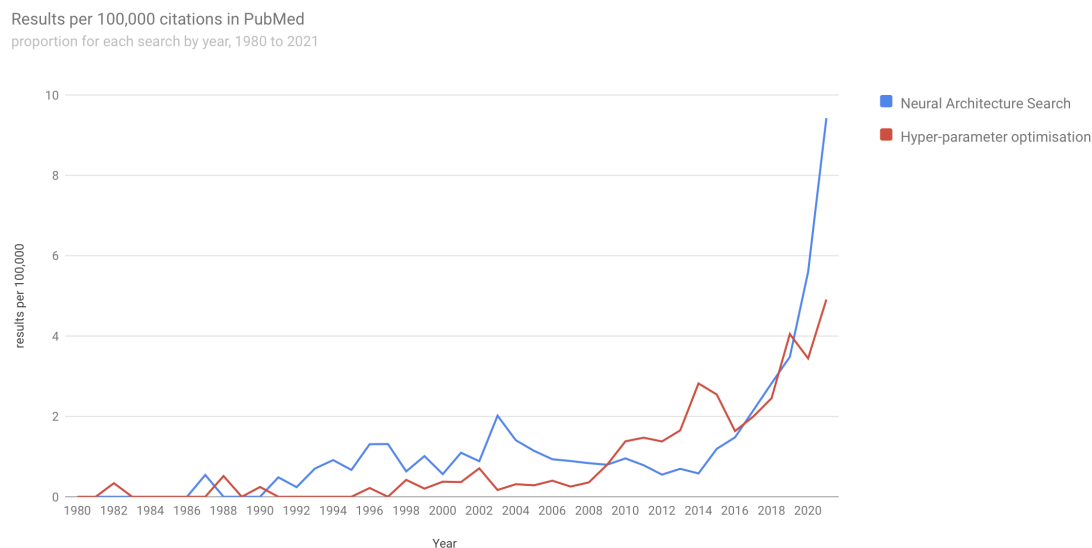
be the most efficient for a particular algorithm using tree/local search, benchmarks all possible configurations with the use of a Random Online Adaptive Racing (ROAR) method to compare, and releases the most accurate combination found for the algorithm and its dataset chosen.

2.2.4 Neural Architecture Search (NAS) and Architecture Optimisation (AO)

The following is about neural network algorithm in Machine Learning.

The NAS problem's objective is to find a robust and high-performance neural architecture by selecting and combining various basic operations from a predefined search space [22, 23, 24, 25]. As stated previously (Section: 2.2.2), the objective of a classification problem is to choose the optimal algorithm and hyperparameters, whereas the objective of a NAS problem is to find the optimal architecture for a given dataset. AO first appeared in an application by Barret Zoph and Quoc Le, as stated in the survey [26], and the approach used reinforcement learning to train a recurrent neural network to automatically search for the best-performing architecture. NAS employs AO to determine the optimal architecture, and AO is composed primarily of a few popular methods [26], including reinforcement learning; evolution-based algorithms; gradient descent; surrogate model-based optimization; and hybrid AO methods, all of which have the primary objective of convergent to the optimal solution (i.e: most accurate architecture). Furthermore, the accompanying graph indicates that NAS is beginning to receive significant research attention this year, surpassing Hyper parameter optimization in terms of published papers, implying that HPO is now likely over-optimized and NAS is not yet. (Fig: 2.1).

Figure 2.1: Machine learning published papers between HPO and NAS



2.2.5 Advanced Techniques

Meta-Learning

The meta-learning [27] phase's primary concept is to minimise the search space by learning from models that performed well on similar datasets; meta-learning can be thought of as a senior researcher using his previous experience to a new problem. More in general, meta-learning, is a technique for finding a good combination of hyperparameters for a particular dataset from what has been working well on similar dataset in the past. Given that, it enables an analysis to begin on a solid foundation rather than having to start from scratch. Following this method, we can see how beneficial this step would be for a junior machine learning researcher who does not yet know everything in the field, such as what works well on a breast cancer dataset, versus a senior who would likely have a rough idea of which model would perform well on the classification of images containing or not containing breast cancer tissues. Meta-learning analyses the dataset and exports pre-configuration that must be optimised using other techniques[28].

Transfer-Learning

Transfer learning aims to transfer knowledge gained by learning about a configuration space to another environment (e.g., the use of different hardware). While there are some similarities between Transfer and Meta learning, they are distinct in that Transfer-Learning (i.e., well-used in Deep Learning and Computer Vision, see [29, 30]) would most likely employ knowledge from previously learned models to develop another via current one. It takes a model that already has information and gently retrains it for a different usecase but in a comparable field.

2.2.6 Auto-Machine Learning's Advantages and Disadvantages

While auto machine learning is being lauded as a weapon capable of displacing data scientists or ushering in the era of machine learning 4.0, it does have fundamental faults that the general public must understand. To begin, one of the most compelling advantages of Auto-ML is the ability to obtain results faster than manually developing a model. After a few hours of searching in the spatial dimension of possibilities, the results can be rather detailed and precise, but a human brain can take up to a week to get the same results. However, manually manipulating the data enables you to understand why you are getting bad results and whether you should go back and acquire additional data; it also provides you with a better understanding of what you are doing and what you can report from the machine learning analysis of the data provided; where Auto-ML does it automatically, you are a little bit dodged by the computer and do not fully comprehend each step and which. With the assistance of Auto-ML, you reduce the likelihood of being out of date, as you can re-generate the best model even if the environment has changed, whereas manually, it would cost additional time and probably lack the necessary expertise. Conducting analysis on a large spatial dimension of possible outcomes also requires

time and computing capacities, which may be prohibitively expensive or time-consuming. One of the most significant disadvantages of Auto-ML is that it may produce a wrong model in comparison to a carefully constructed one. Auto-ML is a generalisation in which a talented designer can create an innovative and unknown method that results in a very performant model, but where Auto-ML will fail due to the generalisation it performs. Finally, the last advantages of Auto-ML is its simplicity; developing a solution using machine learning requires setting up several tools for each component in the pipeline, whereas Auto-ML already does it by itself, resulting in a decrease of the machine learning files in the project you are working on, and an increase in the project's ease of usage

2.3 Auto-WEKA (Framework of Auto-ML)

Auto-WEKA comes from Freiburg University and the University of British Colombia [12, 31], originally forked from the Waikato WEKA tool [13]. It is a black box tool for automating the selection of methods and hyperparameters of WEKA's algorithms. From the Auto-WEKA github's repository 'Auto-WEKA examines the challenge of picking a learning algorithm and configuring its hyperparameters at the same time, moving beyond prior techniques that dealt with these concerns separately. Auto-WEKA does this using a completely automated method that takes use of recent advances in Bayesian optimization. Auto-WEKA's goal is to make it easier for non-expert users to find machine learning algorithms and hyperparameter settings that suit their applications, and therefore enhance their performance.' [32]. Furthermore Auto-WEKA came with two version, the 1.0 [12] and the 2.0 [31]. The initial version of auto-WEKA included three search and eight evaluator methods, as well as the classification algorithm already implemented in the parent software WEKA. Thus, the framework encompasses two ensemble methods and ten meta-methods, in addition to 29 base classifiers and hyperparameter settings for each classifier. The first version's limitation is the absence of the possibility of an overfitted model, while the second limitation was about improving the current method by allowing to share between classifiers. A second version of the library was released in 2019 with improvements to the library's integration with the rest of the WEKA software and a few other features.[31]. The library has been shown with pretty reasonable results, shown throughout the Thorton's study. However, the most recent stable version is no longer maintained; only bugs will be fixed, and no further improvements will be made.

2.3.1 Awarded prizes

Auto-Weka has yet not won any awards, although it has inspired several other libraries, such as Auto-Sklearn.

2.3.2 Pre-Configured Algorithms

The following is the list of pre-configured algorithms found in Auto-WEKA:

```
J48; DecisionTable; GaussianProcesses; M5P; KStar; LMT; PART;  
SMO; BayesNet; NaiveBayes; JRip, SimpleLogistic;  
LinearRegression; VotedPerceptron; Logistic; OneR;  
MultiLayerPerceptron; REPTree; IBK; M5Rules; RandomForest;  
RandomTree; SMOreg; Vote; Stacking; Bagging; RandomSubSpace;  
AttributeSelectedClassifier; RandomCommittee.
```

This list shows a total of 29 different algorithms with hyperparameter settings to explore by Auto-WEKA.

2.4 Auto-Sklearn (Framework of Auto-ML)

By its initial release in 2015 by the Freiburg university lab, the auto-Sklearn library aimed to improve Bayesian Optimization using meta-learning (section: 2.2.5). With an overall range of 132 hyper-parameters, the framework uses 15 classifiers, 14 methods for feature processing, and 3 methods for data preprocessing. It is a framework in two version, the 1.0 [33] and the 2.0 [28]. The first version's main improvement to Auto-ML was its CASH and HPO performance and the second version's main improvement was the integration of a simpler and powerful approach for meta-learning.

2.4.1 Awarded prizes

The first award for Auto-Sklearn came during the first international AutoML Challenge, where they outperformed other frameworks but not on all sub-challenges; however, they still placed first in the competition held between 2015 and 2016 [34, 35, 36, 37]. They had a few months to improve their framework so that they could return the following year and still perform well, and they did it again, winning the second international AutoML challenge in 2017-2018 [38, 39, 40].

2.4.2 Pre-Configured Algorithms

The following is the list of pre-configured algorithms found in Auto-Sklearn:

```
AdaBoost ; Bernoulli naive Bayes ; decision tree ; extrem1.  
rand. trees ; Gaussian naive Bayes ; gradient boosting ; k-  
Nearest Neighbour ; LDA ; linear SVM ; kernel SVM ;  
multinomial naive Bayes ; passive aggressive ; QDA ; random  
forest ; Linear Class.
```

This list shows a total of 15 different algorithms with hyperparameter settings to explore by Auto-WEKA.

Chapter 3

Methodology and Datasets

3.1 Proposed methodology for frameworks comparison

Walking through a comparison of two entities implies that we have considered some rules. The proposed methodology is divided into three major sections, each of which contains a subset of work; the following is a detailed description of each section of the proposed methodology:

To begin, it is necessary to perform a classification analysis on each dataset; knowing the most accurate model over a training set of the entire dataset enables us to determine whether the framework works on such data and, more importantly, whether the framework is performant or chaotic at first glance. Nonetheless, this step's primary purpose is to determine whether the frameworks selected are compatible and do not crash when used with the acquired dataset. The auto-ML analysis must be limited to one hour per dataset and framework in order to give us a sense of the framework's capabilities without wasting too much time waiting for the framework's processes to complete.

Secondly, after conducting an initial analysis across the entire dataset, we enhanced the framework's limit analysis by creating a k-fold cross validation Python program that would train 'k' models across different groups of folds and tested them using a machine learning framework on unseen data (i.e, data that has not been seen during the training of the model). This section of the methodology is used if the first phase's results are not significantly different, it would allow us to determine which frameworks would react most intelligently to unseen data. The k-fold cross validation value (i.e, k) must be set to ten to avoid excessively long wait times for the pipeline to provide results, assuming one hour per fold is specified. The number ten for the k is also a good balance because it would ensure that each fold contains a reasonable amount of data regardless of the size of the dataset used, and would avoid any errors such as having a fold with less than five sample support. The auto-ML analysis must be configured to run for one hour per fold, resulting in ten hours of search time per dataset.

Thirdly, once we have determined which framework performs the best on medical data collected during phases one and two of the analysis, it is time to demonstrate that the winner framework is also effective on real-world data obtained recently from an NHS hospital in England. As a first step, the analysis is runs on the entire dataset and with the previously implemented k-fold programme to see how it reacts to unseen data when the number of real-world samples is sufficient. A small amount of pre-processing will certainly be required to ensure that the data is in the correct format for Auto-Sklearn or Auto-Weka.

After completing the three primary phases, the analysis can be reported and compared in order to demonstrate the utility of the chosen framework using the output from the second phase and the real-world application via the third phase.

3.1.1 Orientation

The proposed methodology has now been stated. However, in practise, it is slightly different and requires a few additional steps to reach the desired results. To begin, those analyses require identical hardware and the same amount of time for the pipeline search. Numerous hours of running on our personal laptop may deplete our memory and prevent us from preparing for the next phase; in this case, the use of a virtual machine is the solution within the proposed methodology.

3.1.2 Frameworks chosen

The frameworks chosen for comparison are Auto-Sklearn, a fork of Sci-kit-learn (2.4), and Auto-WEKA, a fork of the WEKA tool (2.3). These frameworks are well-known in the data science world, having garnered numerous awards over the course of their existence. However, they lack clear information based on medical data, and because they are both extremely powerful, a comparison of those two distinct frameworks was chosen.

3.1.3 Selected computational characteristics (Cloud server)

The needs to run the analysis on a virtual machine could have been done on many different cloud server such as AWS, but Google Cloud been chosen for the following reasons:

- Budget-friendly free to use.
- Google's cyber security team pays close attention to the security of all Cloud images built; given that we are dealing with medical data, this is critical.
- It is extremely simple to use and has a very friendly web/mobile interface to monitor everything.

The following characteristics are the chosen ones to conduct our analysis (Table: 3.1). The instance’s operating system was chosen to be Ubuntu stable latest version, which ensures that we can run all of the required Python packages on it.

Table 3.1: Google Cloud virtual machine instance characteristics

Param name	Param value
Machine type	e2-standard-2
CPU	2 vCPU
Virtual Machine Memory	8 Go
Image size	10 Go

3.2 Collections of data

The data collection process needs to be meticulous. Additionally, the need for diverse fields of healthcare was necessary to ensure we covered a broad range of medical knowledge, as well as obtaining them from a secure location where no subjects would be jeopardised. The following subsections detail each dataset we used, as well as the real-world data we obtained from a United Kingdom NHS hospital.

3.2.1 Online datasets acquired for experiments

The datasets used in the subsequent experiments are detailed in the table Table 3.2. Nonetheless, each dataset originates from the Machine Learning repository, which typically provides the dataset in csv format along with a description of each feature and the task at hand. These datasets did not require any human intervention other than changing the type of the columns to allow Auto-Sklearn and Auto-Weka to process them properly; otherwise, no human intervention was required on the dataset. The datasets have already been anonymize by the UCI Machine Learning Repository, which ensures that we can process them securely without risking putting someone in jeopardy.

Thoracic Surgery dataset [41]

Between 2007 and 2011, data on patients who underwent major lung resections for primary lung cancer were collected retrospectively at the Wroclaw Thoracic Surgery Centre. The dataset demonstrates numerous characteristics, with the predict class variable indicating whether the subject x died a year after surgery or survived which in the case would means that the surgery was a success. Knowing this information in advance after the surgery would enable physicians to determine the life expectancy of a new subject and treat them earlier, avoiding the subject dying after a year following thoracic surgery. Nonetheless, the data demonstrate a high degree

of disparity across subjects; Consider the following table, which summarises the characteristics of each dataset: Table 3.2. [41].

Diabetic Retinopathy Debrecen dataset [42]

This dataset includes features extracted from the [Messidor image](#) set that can be used to determine whether or not an image contains signs of diabetic retinopathy. All features are either detected lesions, anatomical structure descriptors, or image-level descriptors. The predictor class variable was the presence or absence of Diabetic Retinopathy tissue in subject x . Knowing this information enables physicians to initiate treatment early in the event of a problem detected. Consider the following table, which summarises the characteristics of each dataset: Table 3.2. [42].

Estimation of obesity levels based on eating habits and physical condition [43]

This dataset contains estimates of the obesity prevalence in individuals in Mexico, Peru, and Colombia based on their eating habits and physical condition. The predict class variable contained classifications for Insufficient Weight, Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Obesity Type II, and Obesity Type III. Nota bene: 77% of the data were generated synthetically using the Weka tool and the SMOTE filter, while 23% came directly from users via a web platform. Consider the following table, which summarises the characteristics of each dataset: Table 3.2. [43].

Breast Cancer Coimbra Data Set [44]

This dataset contains information about whether a patient has been diagnosed with breast cancer or is in good health. Knowing this information would prompt an early intervention to save the genital area of a woman's body. The predict class variable was a categorization based on one or two, for healthy controls (1) and patients who needed to be considered (2). Consider the following table, which summarises the characteristics of each dataset: Table 3.2. [44].

Heart failure clinical records Data Set [45]

This dataset contains the medical records of 299 heart failure patients collected between April and December 2015 at the Faisalabad Institute of Cardiology and the Allied Hospital in Faisalabad. It illustrates a variety of features that would inform physicians, via a machine learning model, whether a new subject x would suffer a heart attack as a result of a collection of features that aid in understanding how a heart attack could occur. Consider the following table, which summarises the characteristics of each dataset: Table 3.2. [45].

Table 3.2: Online datasets used for the experiments

dataset name	number of samples	number of features	number of classes	class_label_1(%)	class_label_2(%)	class_label_3(%)	class_label_4(%)	class_label_5(%)	class_label_6(%)	class_label_7(%)
Breast-cancer-Coimbra	116	9	2	44.827586	55.172414					
diabetic-retinopathy-Debrecen	1151	19	2	46.915725	53.084275					
Heart-failure-clinical-records dataset	299	12	2	67.892977	32.107023					
ObesityDataSet_raw_and_data_synthetic	2111	16	7	12.884889	13.595452	16.627191	14.069162	15.348176	13.737565	13.737565
Thoracic-Surgery-binary-survival	470	16	2	85.106383	14.893617					

(1) The dataset name indicates the dataset's name; (2) The number of samples indicates the total number of samples available for this dataset; (3) The number of features indicates the number of features available without the class labels column; and (4) The number of classes indicates the number of distinct class labels associated with the appropriate dataset, which also indicates whether it is a binary/multi class classification problem. Finally, (5) the column class_label x indicates the proportion of samples for that class in the entire dataset.

3.2.2 Real-world data acquired for experiments

Thanks to Doctor David Tighe, my research supervisor's NHS contact (Prof Alex, A. Freitas) at EAST KENT HOSPITALS UNIVERSITY NHS FOUNDATION TRUST, for providing us with real-world data from facial surgery. Doctor David Tighe is a facial surgeon who treats a large number of persons with facial skin cancer. The following are the two types of skin cancer that are included in the dataset following the goal that has to be attempted using Auto-Machine Learning:

- The most frequent type of skin cancer, as well as the most common type of cancer overall, is basal cell carcinoma (BCC). BCC accounts for 75% of all skin cancer in the United Kingdom[46]; furthermore, that the malignant cell grows slowly and can be effectively treated if found early. Regardless, according to BCC, it can return after treatment in the same area of skin, which is known as a local recurrence. Early detection of such information would allow clinicians to begin another treatment to address the problem.
- Squamous cell carcinoma (SCC) is a cancer of the keratinocyte cells in the outer layer of the skin. It is the second most common type of skin cancer in the United Kingdom[46], this is a cell that grows as well slowly but can be dangerous if not treated in timely manner as well as after a surgery.

The purpose of using machine learning to interpret those datasets is to learn if a certain subject x would recurrence of skin cancer in the same place where he was treated. Based on previous data, how could we pin-point how to determine if a subject x's skin cancer would recur following surgery?

Data Pre-Processing / Human-intervention

The dataset provided, however, required some human intervention in terms of pre-processing in order to construct a small number of datasets for a small number of analyses. The following will explain what was done and why it was done that way:

First and foremost, the dataset provided was not updated with the most recent information gathered by the Doctor; therefore, it was necessary to open two separate CSV files and intelligently merge the updated one to the master one to ensure that we had the most recent dataset

updated, with the most recent information, from the surgery the Doctor performed during his work at the NHS's hospital. **Secondly**, some column values were in a format that no machine learning framework could understand or sort in the event that it was necessary, and some values were given some symbol such as angle brackets (i.e., for range or showing a sort of limit), so it was necessary to decide with the Doctor and the research supervisor what we could do in that specific case. We grouped some values to make range of value such as between 1 to 3 and 4 to 6 for instance. Furthermore, some data was incorrectly typed, such as '?Clark 3' for 'Clark 3', all of these errors were detected and corrected such that no additional value was created that had the same significance in terms of medical knowledge. **Thirdly**, there were two different class variables that needed to be predicted, either *Peripheral Margin (Deep)* or *Peripheral Margin (Raw)*, and the goal was to extract them into two separate datasets, one with the Deep column and the other with the Raw column, in order to perform two different analyses. Furthermore, because the values in those columns ranged from 0 to infinity, the step was also to centralise them in binary variable. With the help of Doctor David Tighe and his coworkers, the cut off was to put all values above or equal to 0.51 to 1, and lower or equal to 0.50 to 0. In our example, a value of 0 indicates a positive margin, which is a cancer cell located at the sample's edge, indicating that the disease has not been completely eradicated, whereas, a value of 1 indicates a clear margin, which means that tissue around the tumor is healthy and cancer free. **Fourthly**, we needed to ensure that the prediction column did not have any missing values and, if it did, to delete them from the dataset so that we could avoid training a machine learning model with missing values in the class variable column. Additionally, some columns were reassessed because if the proportion of missing values in a specific column exceeded 90, the predictive power of those column on the entire dataset was too low and not even practical to evaluate, and therefore those columns were removed from the dataset. **Fifthly**, the datasets were segmented to allow for a more in-depth examination of the data's behaviour. The datasets were divided into BCC and SCC cases; one contained only BCC cases, another contained just SCC cases, and the third contained both BCC and SCC cases. Each of the third datasets consisted of two subsets: Peripheral margin (Deep) and Peripheral margin (Raw). As a result of that division, greater sense has been able to be made regarding the datasets output from this stage. **Finally**, the columns in each dataset that were composed of nominal values were rethought in order to avoid discrimination during the Auto-ML pipeline's pre-processing of that data. To accomplish this, we performed a one-hot encoding process on all columns that contained more than a binary unique value (i.e, for example a column with three distinct string possible). Additionally, the one-hot method outcome was analysed and post-processed because if a column included less than ten (i.e., a threshold we established) positive values (i.e., 1 and not 0), it would be simple to overfit the model, and therefore those columns were eliminated from the dataset. To finish, the final step was to ensure that there were no columns with 90 missing values in the divided dataset otherwise removing them.

After rigorous pre-processing and evaluation by the research supervisor and the doctor with whom we were collaborating, we finished and the datasets were ready for analysis. The primary

purpose of this data pre-processing it is to provide a process of that data in order to allow David Tighe and his work collaborators, with a model capable of predicting if a new subject has a likelihood of having his skin cancer recur.

The following table summarises the properties of the datasets we processed; however, only the first two will be analysed and reported on in this section 4.2.4:

Table 3.3: Real-world datasets used for the experiments

dataset name	number of samples	number of features	number of classes	predictive_feature_0.0(%)	predictive_feature_1.0(%)
BCC_SCC_peripheral_margin_deep_prediction	1390	29	2	17.266187	82.733813
BCC_SCC_peripheral_margin_raw_prediction	1439	30	2	9.312022	90.687978
BCC_peripheral_margin_deep_prediction	970	26	2	14.020619	85.979381
BCC_peripheral_margin_raw_prediction	999	26	2	10.31031	89.68969
SCC_peripheral_margin_deep_prediction	384	27	2	25.0	75.0
SCC_peripheral_margin_raw_prediction	403	27	2	6.451613	93.548387

(1) The dataset name indicates the dataset's name; (2) The number of samples indicates the total number of samples available for this dataset; (3) The number of features indicates the number of features available without the class labels column; and (4) The number of classes indicates the number of distinct class labels associated with the appropriate dataset, which also indicates whether it is a binary/multi class classification problem. Finally, (5) the column class_label x indicates the proportion of samples for that class in the entire dataset.

Request the Data

If you wish to obtain both the original and pre-processed datasets, please write us an email stating what you intend to do with the data and whether you are willing to mention both the EAST KENT HOSPITALS UNIVERSITY NHS FOUNDATION TRUST, the University, and as well as the authors of this study. The request should be sent to one of the authors through email. The procedure will be as follows: both the university and the research supervisor must consent to the disclosure of such material, and the Doctor and his department must also agree to the request prior to sending any datasets. *Note: Please include a cc to all individuals engaged in this article to ensure that your request is addressed by one of us.*

Chapter 4

Computational operations & Framework experiments

The chapter on computational operations and framework experiments will be separated into two big sections. All technical details will be presented, the program's design and implementation will be demonstrated, as well as the parameters for the Auto-ML framework that were chosen. The chapter will conclude with the compilation of all technical materials into a solution for conducting analyses on the previously introduced dataset [3.2](#), as well as the reporting of the results and a thorough examination of them. Note: All results shown below will only be on unseen data and not the results of the training of the model.

4.1 Implementation of a computational program

Those experiments required getting out of the comfort zone of tools already existent, and then developed what I would refer to as a solution capable of performing the analyses desired with both frameworks, obtaining the results in the most smart format possible, and reporting them as quickly as feasible.

4.1.1 Orientation

Instead of going over a simple PROOF-OF-CONCEPT with a notebook, the conception of a solution requires to be considered in advance carefully. At the start of the project, I had access to only three datasets; however, the research supervisor told me that within a few weeks, we would upgrade to at least five. Given this information, I reasoned that concocting a solution with sufficient genericity to swiftly pass over a new dataset, a new set of parameters, and so on would be the most prudent technological choice we could make; it costs time but results in a better outcome in the long run.

4.1.2 Wrapper for Auto-Sklearn/Sci-kit learn using Python

Given that I desired to create a solution capable of rapidly processing a single new dataset or new set of parameters, I needed to pin-point the way an analysis is conducted or the methods available in Auto-Sklearn or Sci-kit learn. Thus, the development of the two most critical Python classes of the wrapper has been significant and here is a detail of them:

The first class is the solution's Pre-Process class; it wraps several Sci-kit Learn methods in order to let us apply one technique to our datasets in a single line, such as 'GenericFeatureScaling'¹ or the 'One-Hot rescaling'². The class was mostly a wrapper for the methods of Sci-kit learn.

The second class in the solution was the Process class; it mostly wrapped Auto-Sklearn in order to allow us to perform some analysis on our dataset with a single line. The wrapper consists in obtaining a generic parameter file that enables you to use a different Auto-ML's parameter file for each of your analyses, specified as parameters, without modifying your solution's py files. Additionally, the process class wraps all evaluation metrics available across Auto-Sklearn and Sci-kit Learn, allowing us to log the output of the analysis with a level of precision via an internal logger class (e.g, the level 1 will only display the classification report of sci-kit learn on the data, whereas the level 4, will get into the balance error across classes.). Following the final and most critical method in this class, the display latex report, which creates a latex table as we were looking for for each analysis and exports it so that we can simply report on them.

Thanks to the wrapper, we were able to add a new Python file for a new analysis (i.e., a new dataset) and run everything in less than five slight minutes, whereas it would have taken us several hours to ensure everything was setup properly if we had to repeat all the processes following the addition of a new dataset each time. The wrapper significantly reduces the time required to construct the analysis for a new dataset or modifying one already existent. Given this step, we were able to run analysis using one Auto-ML framework on the entire dataset easily.

4.1.3 10-fold cross validation program in Python

As the solution was sufficient to conduct an analysis on the entire dataset provided, we decided with the research supervisor to better investigate the limit of the Auto-ML frameworks we were using. To do so, a good practise is to run a 10-fold cross validation³ program on the dataset, which results in a more thorough investigation of the framework's limit. The programme was also capable of checking beyond the Auto-ML framework we were using, as we believed it was

¹Takes a list of values as input and transform those values in the column selected with what you want in the list that is use for conversion

²returns a vector containing the length of the categories in the data collection. When the ith category is assigned to a data point, all components of this vector are set to zero except the ith component, which is set to one. This enables the categories to be classified quantitatively meaningfully.' [47].

³This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k minus 1 folds.' [48].

worthwhile to determine whether a model trained by a framework reacted effectively to unknown data that was not included during the training, so, using a machine-learning framework instead of an Automated machine learning one.

All code snippets shown below have been written in Python 3.8.x, or for some, the most recent stable version of the Bash script technology.

Auto-Sklearn System design

The following are the program's design step for the Auto-Sklearn experiment: (1) The program randomly divides the dataset into tenfolds; (2) it saves one fold for later use; (3) it feeds the Auto-ML pipeline with the remaining folds to find the best model using these training samples; and (4) once Auto-Sklearn outputs the best fitted model, the fold saved previously becomes useful because Sci-kit-learn will use it to determine the best fitted model's viability on this dataset. Finally, the algorithm would generate ten models with ten results using previously unseen data for a single dataset. This is a normal k-fold process validation; it is very similar to other implementations; the only difference is in how it is performed.

Auto-Sklearn Implementation

The algorithm 1 is the program's main process, from which everything begins. When necessary, it executes the appropriate method. The algorithm 2 is used to divide a dataset into k-folds. The final algorithm 3, is the complete internal process of the k-fold cross validation program, which its design is described in detail in the section 4.1.3. *Nota bene: In the first algorithm, the final line is not shown since it simply a collection of simple Sci-kit learn methods that have been gathered into a single function using a log system; no innovation is made and no information is relevant to provide here.*

Algorithm 1 Three main steps of the 10-fold cross-validation process, *Note: See snippets belows for more details..*

```
1: k_folds_split(k_folds  $\leftarrow$  10)
2: cross_validation_process(all_10_folds)
3: show_latex_cross_validation_process()
```

Algorithm 2 Function that split the dataset into 10 random folds.

```
1: function K_FOLDS_SPLIT(entireDataset, k_folds)
2:   data  $\leftarrow$  entireDataset.copy()
3:   data  $\leftarrow$  data.sample(frac  $\leftarrow$  1)
4:   all_10_folds  $\leftarrow$  array_split(data, k_folds)
5: end function
```

Algorithm 3 The function that performs the cross-validation over the ten random folds. *Note: The reSampling if condition indicates whether a dataset should be re-sampled owing to data discrepancy.*

```
1: function CROSS_VALIDATION_PROCESS(all_10_folds, reSampling  $\leftarrow$  False)
2:
3:   for idx, fold  $\in$  enumerate(all_10_folds) do
4:     training_set  $\leftarrow$  all_10_folds.copy()
5:     training_set.pop(idx)
6:     test_set  $\leftarrow$  fold
7:
8:     input, output  $\leftarrow$  __get_inputs_outputs_from_folds(training_set)
9:     if reSampling then
10:       input, output  $\leftarrow$  reSamplingSMOTE(training_set, input, output)
11:     end if
12:     setup(input, output)
13:     fit_predict()
14:
15:     save_model(path  $\leftarrow$  filepath)
16:     loaded_classifier  $\leftarrow$  load_model(path  $\leftarrow$  filepath)
17:     all_best_models.append(loaded_classifier)
18:
19:     crossValUnseenDataOutput  $\leftarrow$  __predict_unseen_data_cross_validation_process(test_set,
        loaded_classifier)
20:     ensemble_results.append(crossValUnseenDataOutput[0])
21:   end for
22: end function
```

Auto-Weka System design

Concerning Auto-weka's experiment, the Python code implementation revealed in section 4.1.3 had to be refactored. The steps involved in developing the new program's design are as follows: (1) The programme divides the dataset randomly into ten folds; (2) [while looping over the ten-fold each time:] save one fold as a ".arff" (i.e., WEKA supported file) file for later use (i.e., test set); (3) [while looping over the tenfold each time:] save the remaining folds (i.e., training set) as a ".arff" file; (4) After saving all the test set and training set files, the Auto-Weka implementation is complete, and the remaining files are processed by the bash script technology, which performs the following: it runs an Auto-Weka analysis over the ten previously saved training sets using a java command line, and additionally performs a test over the best fitted model using a test set provided as parameter.

Auto-Weka Implementation

The Auto-Weka's implementation require that the algorithm 1 has been executed before executing the algorithm 5.

Algorithm 4 Saves ten random folds into the training/test set for use with Auto-Weka.

```
1: function CROSS_VALIDATION_PROCESS(all_10_folds, reSampling  $\leftarrow$  False)
2:
3:   for idx, fold  $\in$  enumerate(all_10_folds) do
4:     training_set  $\leftarrow$  all_10_folds.copy()
5:     training_set.pop(idx)
6:     test_set  $\leftarrow$  fold
7:
8:     input, output  $\leftarrow$  __get_inputs_outputs_from_folds(training_set)
9:     if reSampling then
10:       input, output  $\leftarrow$  reSamplingSMOTE(training_set, input, output)
11:     end if
12:     inputArffDf  $\leftarrow$  input.copy()
13:     output  $\leftarrow$  output.apply(str).astype("category")
14:     inputArffDf['class']  $\leftarrow$  output
15:     trainingAttributes = self.save_set_arff(inputArffDf, "training_set_fold", str(idx), True)
16:     self.save_set_arff(test_set, "test_set_fold", str(idx), False, trainingAttributes)
17:   end for
18: end function
```

Algorithm 5 The Bash function that run auto-weka for each of the 10-fold (i.e, test_set and training_set) available following the python program execution (algorithm 4).

```
1: function WEKACV
2:   for i ∈ 0..9 do
3:     FILE_TRAIN ← "./training_set_old_i.arff"
4:     FILE_TEST ← "./test_fold_i.arff"
5:     if FILE_TRAIN or FILE_TEST do not exist then
6:       continue
7:     end if
8:     java -cp autoweka.jar weka.classifiers.meta.AutoWEKAClassifier -t FILE_TRAIN -T
       FILE_TEST -timeLimit 60 -seed 85 -memLimit 5000 -nBestConfigs 50
9:   end for
10: end function
```

4.1.4 Utilisation of the Bash script technology to centralise the analysis of a dataset

The Bash script technology was one of the most elegant technical choices we made because it was configured to execute each dataset we had with the right Auto-ML parameter file and or framework too, redirect the output to an appropriate location, and repeat for each dataset in the project. Additionally, this technique was fully executable on the Google Cloud server (see section: [3.1.3](#)) that we used for all of our trials. Additionally, the usage of such technology opens the door to any contributors who wishes to contribute datasets that adhere to the same conventions as those already included in the script. As bash scripts operate as basic command lines, it was fulfilled of some basics

```
Python3 [...] > outputAutoML/[...]
```

commands, so it is not relevant to export it here.

4.2 10-fold cross validation experiment

The experiments detailed below will give the results and characteristics of each dataset for both frameworks with the 10-fold cross validation process. To prevent becoming overwhelmed by the number of tables, a table that contains the average results of Auto-Sklearn and Auto-Weka experiments has been created. By the average, it means that the results shows the average of the ten folds for a given dataset. Nevertheless, if you wish to view more detail about each dataset's fold, the experiment will not display every table generated by the solution due to space constraints, but the appendix does.

4.2.1 Auto-Sklearn Selected version and its characteristics

The version of Auto-Sklearn chosen was the most recent, issued in August 2021. (i.e, 0.13.0). This choice was taken because the authors implemented a change that was necessary to ensure that our analysis functioned properly without crashing. (see issue: [1203](#)). Furthermore, the following table depicts the Auto-Sklearn's parameters (i.e, Auto-ML's pipeline parameters) used for the experiments (refer to the [Auto-Sklearn API for more details](#)):

Table 4.1: Auto-Weka's pipeline parameters (10-fold cross validation experiment)

Param name	Param value
time_left_for_this_task	3600
per_run_time_limit	360
n_jobs	4 (i.e.: maximum 4, if only 2 available 2 will be taken)
memory_limit	5000
seed	2/11/21/42/85
resampling_strategy	holdout
ensemble_size	50

4.2.2 Auto-WEKA Selected version and its characteristics

The version of Auto-Weka selected was the most recent, released on June 7, 2021. (i.e, 2.6.3). This choice was made because the authors said that no more functionality would be released rather only patches would be issued. Thus, the most recent stable version is unquestionably the most recent. Furthermore, the following table depicts the Auto-Weka's parameters (i.e, Auto-ML's pipeline parameters) used for the experiments (refer to the [Auto-Weka API for more details](#)):

Table 4.2: Auto-Weka's pipeline parameters (10-fold cross validation experiment)

Param name	Param value
-t <name of training file>	training_set of the fold (i)
-T <name of test file>	test_set of the fold (i)
-timeLimit <limit> (approximately in minutes)	60 (3600 seconds)
-seed <seed>	85
-memLimit <limit> (in MiB)	5000
-nBestConfigs <limit>	50

4.2.3 Auto-Weka vs. Auto-Sklearn on online-dataset

To summarise, Auto-WEKA vs. Auto-Sklearn's experiment over online datasets, involves the analysis of five distinct datasets (see section 3.2) , the execution of tenfold cross validation program (see section 4.1.3), and the average report of the tenfold cross validation results for each dataset. The outcome is depicted below (Table : 4.3), along with an explanatory of the results. Finally, the analytic setup takes one hour every fold(see section 4.2.1 and 4.2.2), resulting in a total of ten hours per dataset, or fifty hours for all datasets combined then one hundred hours of run for both frameworks combined.

Table 4.3: Auto-Sklearn versus Auto-Weka on the average of the 10 fold-cross validation process (online-dataset)

	Auto-Sklearn Classifier	Auto-Weka Classifier	Auto-Sklearn Accuracy	Auto-Weka Accuracy	Auto-Sklearn Precision	Auto-Weka Precision	Auto-Sklearn Recall	Auto-Weka Recall	Auto-Sklearn AUROC	Auto-Weka AUROC
Diabetic-retinopathy-Debrecen	mlp	LWL	0.750	0.7131	0.752	0.72	0.750	0.71	0.750	0.77
Obesity Dataset raw and data syntheti	gradient_boosting	AdaBoostM1	0.967	0.9208	0.965	0.92	0.965	0.92	0.998	0.99
Thoracic Surgery Binary Survival	gradient_boosting	All different	0.72	0.8532	0.51	0.31	0.52	0.85	0.52	0.49
Breast cancer Coimbra	mlp	Bagging	0.72	0.6977	0.70	0.72	0.69	0.70	0.69	0.68
Heart failure clinical records	random_forest	All different	0.86	0.8405	0.83	0.84	0.82	0.84	0.82	0.82

Each row column has a collection of distinct metrics: (1) The dataset's name; (2) and (3) the classifier chosen by the AutoML pipeline, in terms of which one was chosen the majority of the time; (4 and 5) the accuracy for the given framework, which is the error rate of the chosen model; (6 and 7) the precision for the given framework, which is the quality of the chosen model along the positive predicted values, in terms of how many times they were actually positive. (8 and 9) the recall for the supplied framework, which is the quantity of the chosen model coupled with the number of times the positive values were genuinely positive; (10 and 11) The Area Under the Receiver Operating Characteristics (AUROC) score for a specific framework indicates the model's discriminatory strength, or its ability to distinguish between instances (positive examples) and non-instances (negative examples).

NB: 'All Different' means that the Auto-ML pipeline used a different algorithm for each fold. The bolded value indicates that it is superior to its adjacent.

Examination

To begin, along the results, which are the averages for each ten-fold metric, see the appendix for additional detail. The ‘**classifier**’ column indicates which algorithm was most frequently used across the tenfold results. Gradient boosting appears to be a powerful algorithm most of the time with Auto-Sklearn; however, this algorithm is not pre-configured in Auto-WEKA; Nonetheless, it appears to be significant across medical datasets (i.e, at least for Auto-Sklearn). Auto-weka appears to have struggled to find a single algorithm that works for each fold of a given dataset, as demonstrated by the fact that for the Thoracic Surgery Binary Survival and Heart Failure clinical records, the Auto-WEKA Auto-ML pipeline outputs a different algorithm for each fold of the analysis. The interesting difference between Auto-Sklearn and Auto-Weka in terms of the classifier column is that both mlp (multi-layer perceptron) and random forest algorithms are included in both frameworks’ pre-configured algorithms but were never used by Auto-WEKA, whereas Auto-Sklearn did.

Concerning the metric ‘**accuracy**’, Auto-Sklearn appears to be the winner on almost every dataset except the Thoracic Surgery Binary Survival (i.e., Auto-Sklearn won four times on five, while Auto-Weka won once on five), which means that Auto-Sklearn optimises the error rate or accuracy better on balanced datasets, but Auto-WEKA appears to optimise it better than Auto-Sklearn on imbalanced datasets at least (See section 3.2 for more details about which dataset is balanced and imbalanced).

Regarding the ‘**precision**’ of the model chosen to actually predict a positive value on all positive values predicted, Auto-Sklearn appears to be the most performant as well (i.e., Auto-Sklearn won three times out of five, while Auto-Weka won twice out of five); the results are similar, but Auto-Sklearn was slightly more detailed and produced more accurate results of the value of 0.001, but on the long run it could be easier to see a difference between frameworks. According to the assessment metric ‘**recall**’, Auto-WEKA appears to be more accurate in detecting positive values as positive values (i.e, Auto-Sklearn won two times over five, and Auto-WEKA won three times over five).

Finally, the ‘**AUROC**’ score appears to have been optimised better by the Auto-Sklearn framework, indicating that Auto-Sklearn is better at discriminating between positive and negative classes than Auto-Weka (i.e., Auto-Sklearn won three times over five, while Auto-WEKA won once over five); however, for the final dataset, the Heart failure clinical records, the results were equal between the two frameworks.

At first glance, the outcomes appear to be better with Auto-Sklearn; however, the difference is always extremely minor. In the long term, the difference would undoubtedly be large, even though it is able to pin-point the details at the moment. As a result, Auto-Sklearn remains the clear winner in the framework comparison over online-dataset. Auto-WEKA appears to be more effective at coping with imbalanced datasets than Auto-Sklearn, however Auto-Sklearn is more performant given all the other metrics available in our analysis. Additionally, Auto-Sklearn appears to be more consistent in selecting the method for its model, whereas Auto-Sklearn does

not, because sometimes each fold was composed of a different algorithm. Finally, the difference between the Frameworks is minimal, even though it should be easier to see in the long run. In my opinion, the difference that we can see over an hour of search in the spatial dimension is due to the fact that Auto-Sklearn utilises a Meta-Learning component in order to cope with a previously successful combination on a similar dataset.

4.2.4 Auto-Weka vs. Auto-Sklearn on real-world dataset

To summarise, Auto-WEKA vs. Auto-Sklearn's experiment, involve the analysis of the real-world dataset obtained via NHS's hospital (see section 3.2.2), the execution of tenfold cross validation programm (see section 4.1.3), and the average report of the tenfold cross validation results for each dataset. The outcome is depicted below (Table : 4.4), along with an explanatory of the results. Finally, the analytic setup takes one hour per fold(see section 4.2.1 and 4.2.2), resulting in a total of ten hours per dataset, or twenty hours for all datasets combined then 40 hours for both framework combined. *Nota bene: Only the dataset that has both BCC and SCC cases will be analyse and report here with both class variables in a separate dataset, namely one for Peripheral margin Raw and another for Peripheral margin Deep, both of which will contain BCC and SCC samples.*

Table 4.4: Auto-Sklearn versus Auto-Weka on the average of the 10 fold-cross validation process (real-world dataset)

	Auto-Sklearn Classifier	Auto-Weka Classifier	Auto-Sklearn Accuracy	Auto-Weka Accuracy	Auto-Sklearn Precision	Auto-Weka Precision	Auto-Sklearn Recall	Auto-Weka Recall	Auto-Sklearn AUROC	Auto-Weka AUROC
Peripheral Margin Deep	lda	Bagging	0.834	0.8395683	0.717	0.817	0.562	0.840	0.562	0.673
Peripheral Margin Raw	mlp	REPTree	0.87	0.9068764	0.56	0	0.51	0.91	0.51	0.52

Each row column has a collection of distinct metrics: (1) The dataset's name; (2) and (3) the classifier chosen by the AutoML pipeline, in terms of which one was chosen the majority of the time; (4 and 5) the accuracy for the given framework, which is the error rate of the chosen model; (6 and 7) the precision for the given framework, which is the quality of the chosen model along the positive predicted values, in terms of how many times they were actually positive. (8 and 9) the recall for the supplied framework, which is the quantity of the chosen model coupled with the number of times the positive values were genuinely positive; (10 and 11) The Area Under the Receiver Operating Characteristics (AUROC) score for a specific framework indicates the model's discriminatory strength, or its ability to distinguish between instances (positive examples) and non-instances (negative examples).

NB: 'All Different' means that the Auto-ML pipeline used a different algorithm for each fold. The bolded value indicates that it is superior to its adjacent.

Examination

The datasets used to feed both Frameworks were already highly skewed, as demonstrated in section 3.2.2. The results indicated that Auto-WEKA performed better on both frameworks for the 'Accuracy', 'Recall', and 'AUROC' evaluation metrics. It confirmed what was previously stated: Auto-WEKA performs well on imbalanced datasets, whereas Auto-Sklearn struggled in this trial. Except for the 'recall' column, all other results were comparable between the two frameworks. The results from the real-world dataset were not poor, but they were not ready for production in clinical settings. Additionally, additional research of the dataset, such as requesting additional sample for the minority class, would be a good place to start in order to develop a model capable of accurately forecasting unknown data. Finally, while Auto-Sklearn did not perform as well as

Auto-WEKA on this imbalanced real-world dataset, he was still performing well or nearly as well as Auto-WEKA; there was not a significant difference in performance between the frameworks, but Auto-WEKA performed better due to its ability to cope with imbalanced datasets.

Chapter 5

Conclusions

5.1 Discussion and evaluation of the usage of Auto Machine-Learning frameworks using medical data

Medically speaking, Auto-Sklearn has demonstrated that it is extremely powerful on balanced datasets, but struggles with imbalanced datasets, whereas Auto-WEKA copes wonderfully with imbalanced datasets than Auto-Sklearn and performs well but not as well as Auto-Sklearn on the online datasets acquired for this study in general on both experiments. Both frameworks are extremely powerful; however, Auto-Sklearn appears to be more powerful over a long period of time, which would be relevant in real-world applications requiring more than 5 hours of run time per fold, or even more if necessary considering the dataset is not too imbalanced. Finally, once the data is balanced and there are few missing values, we can be confident that both frameworks will produce excellent results. Additionally, we still need to improve the imbalanced data issues associated with Auto-Sklearn.

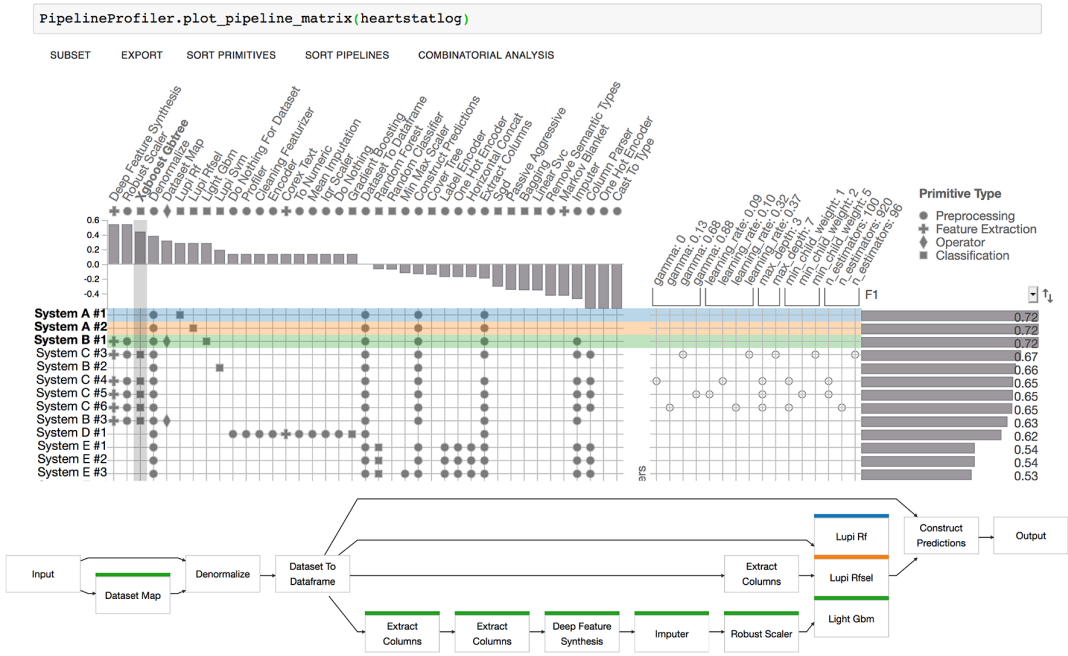
In terms of usage, while Auto-WEKA is composed of the WEKA GUI interface, which is more intuitive for non-experts in machine learning, it does have limitations that Auto-Sklearn does not. For example, Auto-WEKA provides control over various Auto-ML pipeline parameters but not a list of algorithms to search, it is required to execute the analysis on all pre-configured algorithms, which may be unnecessary in some circumstances. Now, while performing an analysis with specific algorithms and receiving a callback from internal processing (e.g., the SMAC result), Auto-Sklearn becomes more personalised and suitable.

Auto-Sklearn has several options for customising the research you wish to perform; additionally, it allows us to access the black box via callbacks in Python, for example, which allows us to delve into the black box's explanation, but Auto-Weka does not. Additionally, such access enables other researchers to produce a profiler of the Auto-WEKA process, therefore elucidating the black box. Auto-Sklearn has an excellent external plugin interface that allows you to examine the process by which the model constructed itself for example [49] (See Fig: 5.1). Nonetheless, Auto-WEKA is a powerful program that is still in use [50, 51], however, such a tool is somewhat

self-contains and prevents others from contributing to innovation and growth in the realm of machine learning, and year after year it will render such a tool obsolete. Due to the fact that Auto-Sklearn is extremely open and allows for complete control from start to finish, it easily facilitates contributions. Additionally, because Auto-Sklearn incorporates a Meta-Learning component, it adds another Auto-ML pipeline parameter to tune in addition to Auto-WEKA, which might be quite useful and promising in some ways. Finally, it would be fantastic to begin developing a web interface to control the fundamentals of Auto-Sklearn and an integrated web Python interface for more detailed processes, which would enable physicians in clinical settings to run such software more easily and with greater assurance than how I proceeded as a Computer Scientists for this thesis project, which required writing everything in lines of code and using a command line interpreter. Finally, and clearly in favour of Auto-Sklearn, the community is very active and fresh work is now underway on a daily basis, whereas Auto-WEKA has been suspended and no new features will be released except fix of crash/bug.

Furthermore, the authors of Auto-Sklearn are extremely responsive via Github issues, which allows us to be confident that if we wanted to construct a solution to analyse our data, they would be here to help us resolve any difficulties we might encounter as quickly as possible. However, as previously stated, one limitation of Auto-Sklearn is its inability to deal with imbalanced datasets even when pre-processing algorithms are pre-configured, something that Auto-Weka seems to excel at.

Figure 5.1: Pipeline profiler - Plugin to see the output of Auto-Sklearn and the process of the black-box



5.2 Conclusion

In terms of which framework to use, both would perform well on medical data; however, for a serious solution that requires time for data pre-processing and post-processing (i.e., understanding the black-box results), Auto-Sklearn would be the most worthwhile framework to use, and especially in the coming years, given the community's growth, the continued release of external plugins, and the authors' commitment to the framework. Additionally, as ML-ops (i.e., Machine learning with DevOps) become more sophisticated nowadays, Auto-Sklearn fits well with on-premise or cloud-based execution, whereas Auto-WEKA would not have a graphical user interface on a cloud server, only a command-line interpreter. Nonetheless, when discussing a Proof-of-Concept or delving into the analysis of an imbalanced dataset, Auto-WEKA is the best choice to take.

This thesis demonstrated a variety of computational materials that will be made open-source on my Github Repository at the following address: [Auto-Sklearn vs. Auto-WEKA](#). The project is thoroughly explained via the main readme, including how to install, use, and contribute to it.

To conclude this thesis, clinical settings are not the easiest domains to work with in machine learning due to the high number of needs. However, there are ways to assist clinical physicians, and one of them is through automated machine learning. Significant collaborations between computer scientists and physicians should continue in order to produce powerful models capable of detecting earlier diseases, saving lives, and assisting physicians in their daily tasks.

5.3 Future Work

The future work could be divided into few separate studies, such as modifying the meta learning hyperparameters of the Auto-ML's pipeline of Auto-Sklearn in order to try to obtain more accurate results for imbalanced datasets. Additionally, it may be configuring an analysis using only a few of the algorithms provided (i.e., a list of the same algorithms between both) for both frameworks to determine whether they always perform well under these limits. Furthermore, increasing the number of hours of analysis on the Google cloud server, as well as the quantity of datasets, would be more relevant and provide a more in-depth sense, even if it already does. However, we would see a greater gap between the two frameworks. Nonetheless, upgrading Auto-classifier Sklearn's to version 2.0 as soon as it becomes more reliable than version 1.0 would be an interesting method to investigate and compare Auto-Sklearn 1.0/2.0 and Auto-WEKA, but the current version 2.0 is not as stable as version 1.0. Finally, a good last investigation would be to know why Auto-WEKA cope well-better imbalanced dataset than Auto-Sklearn, and how Auto-Sklearn could be improved in that way.

Bibliography

- [1] R. Kiyosaki, “The best way to predict the future is to study the past, or prognosticate. richdadrobert kiyosaki,” Jan 2017. [Online]. Available: <https://twitter.com/theRealKiyosaki/status/826208120510636032>
- [2] G. Aurélien, “Hands-on machine learning with scikit-learn & tensorflow,” *Geron Aurelien*, 2017.
- [3] A. M. TURING, “I.—COMPUTING MACHINERY AND INTELLIGENCE,” *Mind*, vol. LIX, no. 236, pp. 433–460, Oct. 1950. [Online]. Available: <https://doi.org/10.1093/mind/lix.236.433>
- [4] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. [Online]. Available: <https://doi.org/10.1007/bf02478259>
- [5] B. Farley and W. Clark, “Simulation of self-organizing systems by digital computer,” *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 76–84, Sep. 1954. [Online]. Available: <https://doi.org/10.1109/tit.1954.1057468>
- [6] N. Rochester, J. Holland, L. Haibt, and W. Duda, “Tests on a cell assembly theory of the action of the brain, using a large digital computer,” *IEEE Transactions on Information Theory*, vol. 2, no. 3, pp. 80–93, Sep. 1956. [Online]. Available: <https://doi.org/10.1109/tit.1956.1056810>
- [7] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. [Online]. Available: <https://doi.org/10.1037/h0042519>
- [8] Y. Zoabi, S. Deri-Rozov, and N. Shomron, “Machine learning-based prediction of COVID-19 diagnosis based on symptoms,” *npj Digital Medicine*, vol. 4, no. 1, Jan. 2021. [Online]. Available: <https://doi.org/10.1038/s41746-020-00372-6>
- [9] A. Abbas, M. M. Abdelsamea, and M. M. Gaber, “Classification of COVID-19 in chest x-ray images using DeTraC deep convolutional neural network,” *Applied Intelligence*, vol. 51,

- no. 2, pp. 854–864, Sep. 2020. [Online]. Available: <https://doi.org/10.1007/s10489-020-01829-7>
- [10] M. Awad and R. Khanna, *Efficient Learning Machines*. Apress, 2015. [Online]. Available: <https://doi.org/10.1007/978-1-4302-5990-9>
 - [11] T. W. Dinsmore, *Disruptive Analytics: Charting your strategy for next-generation business analytics*. Springer, 2021.
 - [12] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Auto-weka: Combined selection and hyperparameter optimization of classification algorithms,” in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 847–855.
 - [13] U. of Waikato, “Weka,” <https://github.com/Waikato/weka-3.8>, 1993.
 - [14] T. M. Mitchell *et al.*, “Machine learning,” 1997.
 - [15] Q. Yao, M. Wang, Y. Chen, W. Dai, Y.-F. Li, W.-W. Tu, Q. Yang, and Y. Yu, “Taking human out of learning applications: A survey on automated machine learning,” *arXiv preprint arXiv:1810.13306*, 2018.
 - [16] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.
 - [17] “Hyperparameter optimization,” Aug 2021. [Online]. Available: https://en.wikipedia.org/wiki/Hyperparameter_optimization
 - [18] M. Lindauer and F. Hutter, “Warmstarting of model-based algorithm configuration,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
 - [19] F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Sequential model-based optimization for general algorithm configuration,” in *International conference on learning and intelligent optimization*. Springer, 2011, pp. 507–523.
 - [20] F. Hutter, H. H. Hoos, K. Leyton-Brown, and K. Murphy, “Time-bounded sequential parameter optimization,” in *International Conference on Learning and Intelligent Optimization*. Springer, 2010, pp. 281–298.
 - [21] I. Guyon, I. Chaabane, H. J. Escalante, S. Escalera, D. Jajetic, J. R. Lloyd, N. Macià, B. Ray, L. Romaszko, M. Sebag *et al.*, “A brief review of the chlearn automl challenge: any-time any-dataset learning without human intervention,” in *Workshop on Automatic Machine Learning*. PMLR, 2016, pp. 21–30.
 - [22] M. F. Tenorio and W.-T. Lee, “Self organizing neural networks for the identification problem,” 1988, pp. 57–64. [Online]. Available: <https://papers.nips.cc/paper/149-self-organizing-neural-networks-for-the-identification-problem>

- [23] H. Kitano, "Designing neural networks using genetic algorithms with graph generation system," vol. 4, no. 4, 1990. [Online]. Available: http://www.complex-systems.com/abstracts/v04_i04_a06/
- [24] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," vol. 5, no. 1, pp. 54–65, 1994. [Online]. Available: <https://ieeexplore.ieee.org/document/265960/>
- [25] S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," 2015, pp. 4:1–4:5. [Online]. Available: <https://dl.acm.org/citation.cfm?id=2834896>
- [26] X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, p. 106622, Jan. 2021. [Online]. Available: <https://doi.org/10.1016/j.knosys.2020.106622>
- [27] D. B. Maudsley, "A theory of meta-learning and principles of facilitation: An organismic perspective." 1980.
- [28] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter, "Auto-sklearn 2.0: The next generation," *arXiv preprint arXiv:2007.04074*, 2020.
- [29] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Artificial Neural Networks and Machine Learning – ICANN 2018*. Springer International Publishing, 2018, pp. 270–279. [Online]. Available: https://doi.org/10.1007/978-3-030-01424-7_27
- [30] X. Li, Y. Grandvalet, F. Davoine, J. Cheng, Y. Cui, H. Zhang, S. Belongie, Y.-H. Tsai, and M.-H. Yang, "Transfer learning in computer vision tasks: Remember where you come from," *Image and Vision Computing*, vol. 93, p. 103853, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0262885619304469>
- [31] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-weka: Automatic model selection and hyperparameter optimization in weka," in *Automated Machine Learning*. Springer, Cham, 2019, pp. 81–95.
- [32] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Weka," <https://github.com/automl/autoweika>, 2015.
- [33] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015. [Online]. Available: <https://proceedings.neurips.cc/paper/2015/file/11d0e6287202fcd83f79975ec59a3a6-Paper.pdf>

- [34] I. Guyon, L. Sun-Hosoya, M. Boullé, H. J. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, M. Sebag, A. Statnikov, W. Tu, and E. Viegas, “Analysis of the automl challenge series 2015-2018,” in *AutoML*, ser. Springer series on Challenges in Machine Learning, 2019. [Online]. Available: <https://www.automl.org/wp-content/uploads/2018/09/chapter10-challenge.pdf>
- [35] I. Guyon, K. Bennett, G. Cawley, H. J. Escalante, S. Escalera, T. K. Ho, N. Macià, B. Ray, M. Saeed, A. Statnikov, and E. Viegas, “Design of the 2015 ChaLearn AutoML challenge,” in *Proc. of IJCNN*, 2015. [Online]. Available: http://www.causality.inf.ethz.ch/AutoML/automl_ijcnn15.pdf
- [36] —, “AutoML challenge 2015: Design and first results,” in *Proc. of AutoML 2015@ICML*, 2015. [Online]. Available: <https://drive.google.com/file/d/0BzRGLkqgrI-qWkpzcGw4bFpBMUK/view>
- [37] I. Guyon, I. Chaabane, H. J. Escalante, S. Escalera, D. Jajetic, J. R. Lloyd, N. Macià, B. Ray, L. Romaszko, M. Sebag, A. Statnikov, S. Treguer, and E. Viegas, “A brief review of the chalearn automl challenge,” in *Proc. of AutoML 2016@ICML*, 2016. [Online]. Available: <https://docs.google.com/a/chalearn.org/viewer?a=v&pid=sites&srcid=Y2hhbGVhcm4ub3JnfGF1dG9tbHxneDoyYThjZjhhNzRjMzI3MTg4>
- [38] “Automl 2018 challenge :: Pakdd2018,” March 2020. [Online]. Available: <https://competitions.codalab.org/competitions/17767>
- [39] I. Guyon, L. Sun-Hosoya, M. Boullé, H. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, M. Sebag *et al.*, *Analysis of the AutoML Challenge series 2015-2018*, 2017. [Online]. Available: <https://www.automl.org/book/>
- [40] W. Tu and E. Viegas, “Analysis of the automl challenge series 2015-2018.”
- [41] M. Zikeba, J. M. Tomczak, M. Lubicz, and J. 'Swiatek, “Boosted svm for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients,” *Applied Soft Computing*, 2013.
- [42] B. Antal and A. Hajdu, “An ensemble-based system for automatic screening of diabetic retinopathy,” *Knowledge-based systems*, vol. 60, pp. 20–27, 2014.
- [43] F. M. Palechor and A. de la Hoz Manotas, “Dataset for estimation of obesity levels based on eating habits and physical condition in individuals from colombia, peru and mexico,” *Data in brief*, vol. 25, p. 104344, 2019.
- [44] M. Patrício, J. Pereira, J. Crisóstomo, P. Matafome, M. Gomes, R. Seíça, and F. Caramelo, “Using resistin, glucose, age and bmi to predict the presence of breast cancer,” *BMC cancer*, vol. 18, no. 1, pp. 1–8, 2018.

- [45] D. Chicco and G. Jurman, "Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone," *BMC medical informatics and decision making*, vol. 20, no. 1, pp. 1–16, 2020.
- [46] NHS, "Skin cancer," Oct 2020. [Online]. Available: <https://www.christie.nhs.uk/patients-and-visitors/your-treatment-and-care/types-of-cancer/skin-cancer>
- [47] DeepAI, "One hot encoding," May 2019. [Online]. Available: <https://deepai.org/machine-learning-glossary-and-terms/one-hot-encoding>
- [48] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.
- [49] J. P. Ono, S. Castelo, R. Lopez, E. Bertini, J. Freire, and C. Silva, "Pipelineprofiler: A visual analytics tool for the exploration of automl pipelines," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 390–400, 2020.
- [50] D. Singh, P. K. Pant, H. Pant, and D. C. Dobhal, "Robust automated machine learning (automl) system for early stage hepatic disease detection," in *Intelligent Data Communication Technologies and Internet of Things: Proceedings of ICICI 2020*. Springer Singapore, 2021, pp. 65–76.
- [51] T.-D. Nguyen, K. Musial, and B. Gabrys, "Autoweka4mcps-avatar: Accelerating automated machine learning pipeline composition and optimisation," *Expert Systems with Applications*, vol. 185, p. 115643, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0957417421010368>