# Generating Representative Artificial Datasets for Algorithm Evaluation

Dan-Florin Berbec
Supervisor: Peter Bloodsworth

Kellogg College

University of Oxford


Project Proposal
MSc in Software Engineering

**Abstract**

Machine Learning algorithms become the main assets for many IT companies and their correctness is fundamental in many fields such as healthcare and finance, or simply when using an app that provides directions to a user or makes suggestions based on personal information. Computational power increases exponentially and the complexity and performance of algorithms follows the same pattern, while explainability becomes more challenging. For regulators, investors or companies that have an interest in buying or investing in such a company, the task of validating the IP becomes difficult, especially when all that is accessible is a black box with a limited amount of test data. More importantly, AI decisions can significantly impact an end user's life. We aim to assess the tools and best practices currently used to reliably and independently evaluate and explain an algorithm decision, which in turn, will provide confidence to all stakeholders that it works indeed as specified and is trustworthy for real world use cases.

# Contents


1 Motivation . . . . . 2
2 Introduction . . . . . 4
2.1 Explainability - Problem Outline . . . . . 4
2.2 Proposed work . . . . . 5
3 Background Literature . . . . . 7
3.1 Overview . . . . . 7
3.2 What is explainability? . . . . . 7
3.3 Current approaches . . . . . 7
3.4 LIME . . . . . 8
3.5 SHAP . . . . . 9
3.6 Selection of methods - reflection on why these are best to be compared 9
3.7 Artificial Neural Networks . . . . . 9
4 Exploratory work . . . . . 11
4.1 Building an ANN . . . . . 11
4.2 Choosing a Dataset . . . . . 11
4.3 Exploratory Data Analysis . . . . . 12
4.4 Categorical and Missing Data . . . . . 12
4.5 Architecture of the ANN with Keras . . . . . 14

# 1 Motivation

Algorithms are present in almost every electronic device we buy. From a simple light switch to running an OS and applications on a smartphone, their influence reaches beyond our daily decisions. For example, algorithms in a smartphone may direct a taxi driver on an alternative route based on congestion indicators set by a real-time data feed. Similarly, the prices of products in a supermarket may be influenced by low latency foreign exchange algorithms trading in an investment bank.

Another example shows the US justice system relying much more heavily on algorithms - these are used to provide a likelihood score of a convict to re-offend and is one of the main factors for a judge when deciding a sentencing[2]. The algorithm used is trained on historical data and seems to have biases correlated to a convict financial status - it may transform these biases into causations. Importantly, the underlying algorithm is private and not made available to the public for assessment.

Provided that secrecy is of utmost importance for machine learning companies across all industries, many questions arise from stakeholders: how can regulators verify the fairness/lack of bias of a financial algorithms? Will an algorithmic robot trained on worst case scenario data from the 2008 financial crises perform as well during a new financial crises with different root causes? How can a judge have confidence that no bias is part of the score given to a convict if the underlying algorithm is only accessible to a handful of people? How can investors evaluate that the product of an ML company does indeed what is says it does?

More importantly, given the coronavirus outbreak that started in December 2019, can a doctor trust ad-hoc built AI systems that claim to diagnose the virus or help find a cure? Many companies are racing to produce products that have eye-catching accuracy and performance, but more discussion is needed on how these algorithms perform on real test data.

For example, Darwin AI [25] claims to work on a novel Covid-19 test that uses a classification system based on CT scans to detect the spread of the virus in the lungs vs common pneumonia or healthy patients[33]. The authors do warn that their algorithm should not be used in production systems as it is only trained on a limited number of samples for positive patients - in an online presentation the authors describe how they initially produces a model with perfect predictions for all classes, but it was using the colour of the bed the patient was lying on while the CT scan was performed. Such a perfect model would have disastrous consequences if performance was the only important metric. However, similar products are already used in hospitals[21] and within the scientific community many such approaches are discussed with a focus on performance.[29].

Another example within healthcare of a neural network that performs well but picks biases is described by Rich Caruana, a principal Microsoft researcher. He built a network which classifies severe cases of pneumonia vs mild ones[24]. On close inspections he notices that having asthma lowers the risk of having severe pneumonia (among several other biases). This correlation is related to the extra care taken for such patients - they are more likely to see a doctor immediately if they don't feel well. If his network would be deployed to decide on real cases, it would implicitly classify them as a lower risk and could have catastrophic consequences - such patients would likely not be admitted to hospital. Importantly, he stresses out how hard it is to detect such biases and why at least in healthcare it is crucial that a doctor understands why a classification has been made.

Furthermore, machine learning classifiers are becoming increasingly complex: specifically, deep learning neural networks may outperform other classifiers such as SVMs or decision trees but the results are hardly explainable due to the hidden layers that aim to mimic real brain activity. For example, an architecture for convolutional neural networks such as ResNet[27] used for image classification can use 152 layers. It may output a superior performance compared to other architectures, but how can we detect biases and explain how it learned them, thus helping us to improve the algorithm?

The new general data protection regulations (GDPR) which came into effect in the European Union, further stress the importance of being able to explain automated decisions to individuals. This allows individuals to challenge decisions [32] and requests explanations.

Therefore, more discussion is needed on why an algorithm outputs a specific classification and how it will generalize on real data outside it's training set. This need constitutes the core motivation for this project. The aim is to survey available tools that help explain machine learning algorithms. A complete survey of such tools for every machine learning algorithm would be outside the scope. Deep Learning is a relatively new field and that neural networks become ubiquitous, a more narrowed scope is to mainly assess tools used to explain artificial and convolutional neural networks. However, most tools and techniques treat AI models as a blackbox and do not require an understanding of the underlying architecture. As such, these tools are applicable to a wide range of models.

Although some techniques for explainability are fairly recently developed, the interest within the scientific community in this topic is very high and many pros and cons have been discussed. The aim of this project will be two-fold: one one side we aim to discuss and compare the various techniques and on the other to survey the actual applicability in the real world, the latter not being researched as much.

For this purpose, the project will start with some exploratory work, with implementation of neural networks and discussing approaches for explaining their outputs. The next part of the project is to work on a couple of real case studies and answer questions such as: How do commercial companies use these tools? What are the best practices used to develop trustworthy models? What features would data scientists like to see in these tools in the future to help them better understand and improve their models? Can their models be assessed independently?

This approach differs slightly form the one described in the original project proposal. Instead of evaluating the performance and determine the real value of commercial algorithms, we focus instead on explaining their decisions. This new approach is motivated after extensive reading and discussions with researchers who work with commercial companies. Aiming to challenge their IP and requesting their help in providing their data, algorithms, benchmark tools and most importantly their time, would be unrealistic and provide little value. Likely companies would dismiss such requests and furthermore it would be unlikely to outperform real employees who have a vast amount of experience in machine learning.

On the other hand, understanding the practices used commercially to explain models, could benefit them in return by providing recommendations of how such practices can be further improved. In addition, only access to a blackbox algorithm with a few test samples would be enough to perform an attainability evaluation as opposed to entire datasets required to build the algorithm completely. It is reasonable to expect a higher willingness from private companies to collaborate under such circumstances of mutual benefit.

# 2 Introduction

## 2.1 Explainability - Problem Outline

While simple algorithms have been around for many years aiding our decisions, complex ones now become ubiquitous in our decision-making process, many times with a significant outcome. Almost all industries including finance and healthcare or governmental institutions use algorithms to process large amounts of data to optimise spending resources. For example, a review[3] from the New Zealand government shows how crucial algorithms are in ensuring the wellbeing of the population. Ministries of justice, education and healthcare, customs and police, all use algorithms and data to improve their efficiency. Algorithms allow for data to be gathered and analysed, identifying key areas where young people need support or helping to aggregate data and address risks at border control. Passport and visa applications can also be automated to reduce waiting times and allow staff to manually process the ones that reach a risk threshold. However, the review clearly states that significant decisions are still currently made by humans.

Finance also relies heavily on automation and algorithmic trading is rising very fast. Robot advisors are slowly replacing or at least aiding financial advisors to reduce the commissions charged as well as improve the performance of a fund[6]. For most financial institutions, especially large investment banks, regulators require backtesting to be executed and results sent for analysis[5]. This aims to ensure that trading strategies performing live today can also handle data in extreme environments such as the financial crisis in 2008.

While most of the algorithms used in decision making are currently crunching data to produce a deterministic output, newer types of algorithms within Machine Learning (ML) and Artificial Intelligence (AI) aim to train on historical data to find patterns that are applicable on new test data. From systems that recommend similar items to online shoppers to a virtual Go player that can beat a world champion[4], ML algorithms become increasingly trusted to take decisions that may significantly influence people's life. Huge investments are poured into self driving cars and a large number of big tech companies as well as many start ups[20] claim a share of this funding. These smart algorithms powering the cars may be expected to take the right decisions in noisy environments that have not been part of the training data, or ethical ones with no clear solutions, such as the trolley problem[18]. As they are designed by humans, biases are also likely to influence the algorithm design. For such critical systems, especially where different decisions can be regarded as optimal, explainability becomes crucial.

But the main assets of ML companies lie in developing such algorithms and preserving their secrecy. For big tech giants, such as Google, Uber or Waymo, theft of trade secrets is a serious concern and lawsuits are commonplace[7, 19]. Algorithmic trade secrets, such as trading software, are also crucial in the financial industry, and any attempt for industrial espionage is thoroughly chased[17].

Therefore, most of the time, a blackbox is provided to an external audit or potential investor with limited test data. Furthermore, this data could easily be proprietary as well and cleaned to fit the claims of the product. Currently, it seems that stakeholders do not have any means to independently verify that an algorithm does indeed what it says it does.

With the increasing rise of tech startups, we may assume that in the near future, multi billion dollar companies may have only a handful of employees with a codebase living in the cloud and their main asset being IP. Their algorithms may have significant decisions for the public, yet

understanding why a decision has been taken may be challenging to explain. In particular, end users should always be able to understand exactly what factors mattered when an automated decision is taken (e.g. Why has my loan been denied? Why was I not admitted to the hospital when ill? Where did my pension's trading strategy go wrong?) - importantly such explanations should also be easily understandable. From a human computer interaction perspective this will help users accept decision more easily: a good example is Netflix, which provides clear explanations to users for new shows it recommends based on previous viewings.

### 2.2 Proposed work

The aim of this project is to investigate the available tools used by researchers and other stakeholders in understanding how their models behave when making classifications. Ideally, the main focus will be to compare tools and processes that can fit scenarios where the model is treated as a blackbox with a limited amount of test data. The next stage is to assess the suitability of these tools within real case studies and eventually make realistic proposals for improvements. The focus will be on explaining artificial and convolutional neural networks, but the general guidelines derived should be applicable to any organisation independent of the size, technology stack or industry.

Deep learning mainly provides two types of algorithms: supervised and unsupervised learning. Supervised learning is used for classification purposes while unsupervised learning aims to find patterns within a data sets that are not mapped to specific labels.

Artificial neural networks (ANN) are trained on test data sets and make predictions on new inputs. Their efficiency can usually be measured on how accurate their predictions are. For example, given a dataset of customers leaving a bank, a ANN can make future predictions based on independent variables on whether a customer is likely or not to leave the bank as well. An ANN could also decide based on income and age, whether a customer should be granted a loan or not. Another example could be a ANN making stock price predictions. While the algorithm is not public, it would be crucial for customers to have the ability to understand its behaviour before placing any money of their savings.

The aim is also to test the behaviour of the algorithm under worst case scenarios or perform other modifications to the dataset to identify potential poor performance that impacts the validity of testing and in particular pick any unhealthy biases. The project will evaluate a number of open source ANNs used in a variety of fields such as healthcare, finance, or law. The algorithms may be packaged as runnable code with training and test data or as a black box with test data only.

A first step is to review the historical and current approaches used to train and explain model predictions. By reviewing the current literature, the most suitable tools and techniques will be selected. The next stage will be to briefly discuss the architecture or neural networks and provide an implementation for an artificial and convolutional network. The solutions identified in the background literature can be applied and their performance can be assessed. Are these solutions indeed able to explain how the models make predictions and identify biases? What are the computational requirements and how feasible are they if they were used in real world scenarios?

The second step will discuss specific use cases. Of particular interest are models that perform within healthcare, especially during the coronavirus pandemic where AI is heavily relied on to diagnose or find treatments. Many solutions are rapidly deployed and their correctness can

be crucial in reducing the number of deaths caused by COVID-19. Informal discussions with software engineers designing these models will be conducted aiming to find out what tools and best practices are used to ensure correctness of deployed models. Where possible, the selection of models discussed in the background literature will be applied on such models to make further inferences. This will of course be done with the express consent of the companies or where such models are open source.

The expected outcome will present the reader with a set of best practices and tools currently used in academia and in the real world for explaining AI models. A comparison of the tools and use cases they cover best and what is needed in the future will be addressed. These examples do not aim to provide a silver bullet for implementing foolproof explainable models, but rather a sample of what might be useful in the development and testing process of new models to ensure that correct outputs are produced for the right reasons. This is subject to further background literature and the rapid growth of new ideas being published in the field.

Further notes: novelty of exaplinability -whhy and what that means. models are designed and trained to perform well and reduce the loss function not to be explainable to humans in an intitive way

The ‘black box’ problem that plagues AI — our inability to peek inside exotic neural networks and understand how they work — represents one of the most urgent moral and

impaortant areas such as healthcare will avoid using where possible models that cannot explain descisions even when the performance of a deep learning is by far superior to other models.

# 3 Background Literature

## 3.1 Overview

## 3.2 What is explainability?

Explainability is a new filed in AI and the focus is mainly to explain what features in the input significantly influence the model to provide a higher probability to a class than to others. But what can we classify as a well explained decision? What metrics can be used?

Christoper Molnar [30]

## 3.3 Current approaches

Ideally this process should be incorporated by design and starts at the very beginning when exploratory data analysis is performed.

Khaleghi [9] explains the importance of identifying shortcoming of the training data such as class imbalance issues or dealing with missing data. This can be achieved by using different types of charts to plot the data and using statistical analysis to fill in missing values. Importantly, features should also be explainable and meaningful to the end user who should also intuitively asses their importance. When the number of features in a dataset is too large, dimensionality reduction techniques such as principal component analysis can be applied.

At a model implementation stage, many new architectures emerge that incorporate explainability by design, in particular for neural networks. For example, Alvarez-Melis et al (2018)[23] propose Self-Explaining Neural Networks which work by *"progressively generalizing linear classifiers to complex yet architecturally explicit models"*. Such designs aim to make explanation understandable, faithful and consistent. Similarly, Al-Shedivat et al(2017)[22] propose Contextual Explanation Networks: these *"generate parameters for intermediate graphical models which are further used for prediction and play the role of explanations."* These models not not offer a clear understandable interpretation and more research is needed to develop them further.

However, explainability is mainly assessed after the model has been created. In turn, this approach does allow for improved models to be generated after assessment in an iterative process and are not different than other methods used to improve performance such as hyper parameter tuning.

In a broad overview, Sheldon Fernandez [26] describes four general approaches currently used to interpret and explain models, which grew from theoretical scenarios into commercial use-cases. The first approach is to create **perturbations** of a particular input set and asses the impact of the predicted decisions. **Backwards propagation** estimates the weight of neurons in each layer to derive an importance score for each feature. Another approach is to use a simple but more explainable model that is trained to imitate the complex one (e.g. a decision tree imitating a deep neural network). The last approach, **Activation Optimization** searches for input patterns to maximise/minimise the classification probability for an input.

Each of these techniques has benefits and limitations. In the article mentioned, Fernandez further details some limitations: for example, using a proxy to simplify a model will rather provide hints on how the underlying algorithms performs. While these may be useful, they may not be accurate and should not be relied upon in mission-critical systems.

Other techniques will perturb the input to assess the importance of its features. Perturbing such features even slightly can produce opposing results and furthermore, they can only be applied to a number of limited inputs and would not account for global behaviour. Perturbing data with a large number of features would further also increase the computational cost exponentially.

In all cases, the scores attributed to different features will need to be interpreted by the user and models improved accordingly. Inherently, such decisions would be debatable by different users. In healthcare for example, two specialists could have different domain views on weather some features are more important than others.

For our purposes, we need to identify solutions that are readily available to developers, easy to implement on current models and understandable by all stakeholders, ideally applicable to any type of model and not exclusive to neural networks.

Model-specific or model-agnostic focus on explaining after the network is build. Local or global? -area I am going to consider and why and area I will not consider and ww - mostly interested in black box approaches as this allows independent evaluation -

- high level overview of different approaches with pros and cons Perturbation: - treats models as a blackbox, and probes it. the explanation is not a clear verifiable one. they fail to generalise for all classes. only analyses part of networks and are not globally applicable to all classes. not verifaiable

-most of these produce heatmaps and lets the reader decide on the correctness - but what if perople interpret these resutls differently?

Backwards Propagation Proxy: Activation Optimization:

split into categories of approaches , not only what the stuff is and educate the reader.

- developers require an deep understanfing of how the models arhitecure learns so they can improve it, whil those using it need to understand how it works as a whole when a decision is taken - fro example a written text on what features are important

input features are not the only thing - is the dataset imbalanced? what optimization functions were used?

Influence Functions -

For our purposes, we need to find solutions that are available to developers, easy to implement on current models and understandable by all stakeholders.

The perturbation mechanism methods have two main advantages: they are typically easy to implement, and they are not limited to a specific model architecture but compuytationally expensive if the models are high dimentaional.

Gradient-weighted Class Activation Mapping (Grad-CAM)

### 3.4 LIME

what is useful, what I learn, what areas are weak, what can be improved like a critic, this area can be better, waht to take forward

(LIME) approach provides explanation for an instance prediction of a model,

### 3.5 SHAP

### 3.6 Selection of methods - reflection on why these are best to be compared

We do not aim to implement techniques from scratch based on papers

### 3.7 Artificial Neural Networks

Artificial Neural Networks have become very popular in recent years for creating Artificial Intelligence algorithms. The aim is to teach a machine how to extract and learn features of a data set to make new predictions by replicating processes happening in the biological brain. Using a simplified neural representation: the dendrites, nucleus and axon of a neuron are respectively abstracted for a theoretical algorithm as the input, processing node, and output. This idea dates back from 1957 when Frank Rosenblatt demonstrated the first perception: "the first machine which is capable of having an original idea" [15], but only recently has computational power increased to develop the ability to make predictions.

The ANN is built using visible input nodes (data you can see - ex. age, address, education, job status) that feed into several hidden layers of neural nodes further feeding into output nodes, the visible binary or categorical output, for example a loan application being granted or declined. A node in any layer can be connected to one or many nodes in the next layer. Due to the complexity of the connections in the middle layers and the fact that the data is not easily readable, the middle layers are called hidden layers (fig 1).

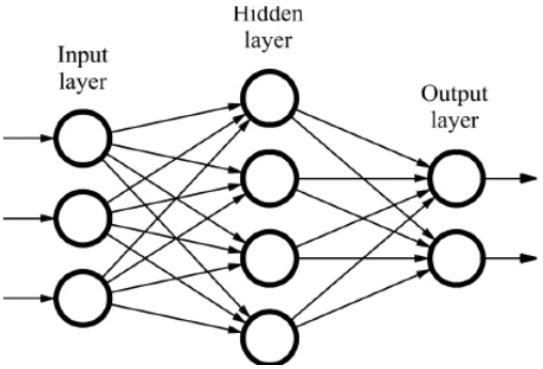
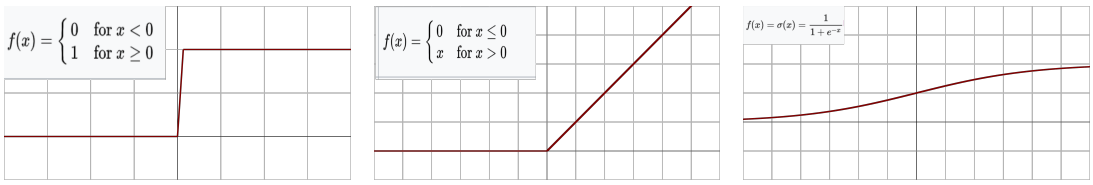



Figure 1: Simple neural network with one hidden layer [13]

Each neuron in the network receives input from other nodes and computes this into a single output that feeds into other neurons or output nodes. A simple approach for a neuron's prediction could be to multiply an adjustable weight to each incoming input. To avoid producing zero values for zero inputs a bias can be added further as shown in eq 1

$$\sum_{i=1}^{n} x_i w_i + b_i \tag{1}$$

When the input $x_i w_i$ is significantly less than the bias $b_i$ this implies that input has little effect. But over the training period the weight will be adjusted accordingly to overcome the effect of the bias. The final sum is not bounded and thus can be largely different than the output of other nodes. Activation functions are used to contain this value between a specific range, most commonly between -1 and 1. For example, a simple activation function, binary step, can solve this issue. But this cutoff function however will drop many small details and may only be used




(a) Binary Step (b) ReLu (c) Sigmoid

Figure 2: Activation functions [1]

in binary classification problems. While there are many activation functions available, the most commonly used alternatives are the *sigmoid* function and *rectified linear unit* (ReLu) which preserve and propagate more information as shown in figure 2.

In multi-class classification problems, the output layer will have one neuron per class. The categories may or may not be mutually exclusive and a common way to pick the best prediction is to choose the one with the highest probability when using for example a sigmoid function. As neurons in the last hidden layer are not interconnected these probabilities are independent and need to be normalized such that their sum equals to 1. A simply function such as *softmax* divides each probability by the total sum. When categories are non exclusive, a probability threshold for each output neuron such as 0.5 can be used to pick the final outcomes.

The ANN can be trained using an existing dataset of examples to make future predictions and the weights of each neuron are adjusted depending on the performance of the prediction vs the actual value. Equations such as the quadratic cost function (see eq 2) or cross entropy are commonly used to make this measurement for both binary or multi-class classification. The equation provides a positive measurement of the distance from the real value. By defining $a^L$ as the activation function applied on $x_i w_i$, at the last layer, the cost function takes into account a large amount of information as vectors: the weights, biases, activation function and true outcome across a number of predictions called the batch size.

$$C = \frac{1}{2n} \sum_x \|y(x) - a^L(x)\|^2 \tag{2}$$

To find the correct values for the weights and biases, the cost function needs to be minimal. By plotting a simple version of this exponential cost function against the weight of one neuron (see fig 3), one could theoretically simply calculate the derivative for 0, but in reality the complexity is directly proportional to the number of neurons and will have many local minimum values making such computations infeasible.

Instead, the slope can be calculated and moved downwards to find the minimum value that optimises $w$, known as gradient descent. After running a fraction or the entire dataset also known as the batch size, weights can be adjusted based on minimising the slope. Large step sizes may overshoot the minimum while small ones will take a long time to converge. Thus, a good approach is to adapt this learning rate and Adam[28] is one of the best performing algorithms.

When running on the entire batch size however, the updates happen less frequently and a local minimum is more likely to be chosen instead of the global one. An alternative is to optimize weights after each input in the training data, known as stochastic gradient descent. Due to the

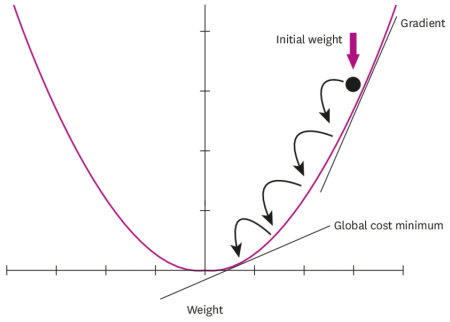



Figure 3: Simple cost function graph[8]

increased update frequency, this is more likely to find a global minimum, but is dependent on the order of the rows in the training set and is not deterministic when data is picked randomly. Finding the global minimum has many other challenges depending on the dataset, such as very big slopes or many small ones. Trusk 2015 [14] provides possible solution to these issues.

Backpropagation uses the cost function to apply changes on the weights and biases of each neuron in every layer. Taking a multi-class classification problem, such as labeling dogs, cats, and fish, the aim is to make adjustments starting from the output layer and propagating these back to the first hidden layer. If the training example has a true value of a cat and the output of the activation function has a high probability for dog, then the aim is to decrease $a^L(x)$ for the dog label and increase the the weight for cat label using the cost function. The in depth mathematical details of the algorithm are beyond the scope of this project but the main equations including formal proofs are discussed by Nielsen 2018 [31] in chapter 2.

# 4 Exploratory work

## 4.1 Building an ANN

## 4.2 Choosing a Dataset

Kaggle[11] in a popular online platform for data scientists to share knowledge: this includes uploading real datasets and making them available for public use. Each dataset is open to a discussion thread and possible sample solution are discussed for different types of classification, including deep learning algorithms. Many top machine learning experts who work in the industry or academia will take part in competitions and their solutions are made public. This makes such datasets an excellent choice for the purpose of this project as a simple ANN implementation can be assessed against other solutions which arguably would end up powering real consumer products.

The Stroke dataset[16] contains lifestyle and medical data about patients, some of which suffered a stroke. This is an anonymous dataset the was first published be McKinsey as part of an online Hackathon on Healthcare although the source of this information cannot be verified online. The dataset has no personal identifiable information. The 14 columns contain attributes such as gender, age, hypertension, heart disease, smoking status, or the work type of a patient. More importantly, it also specifies if a patient had a stroke which allows for classifier to be trained and make future predictions.

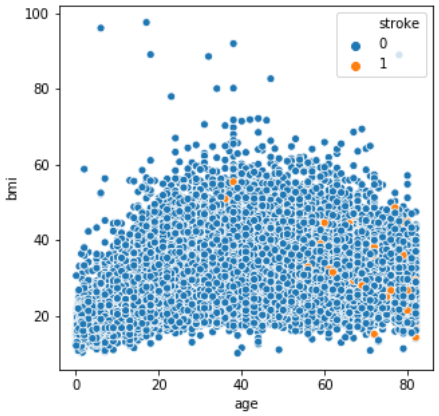



Figure 4: Age vs body mass index

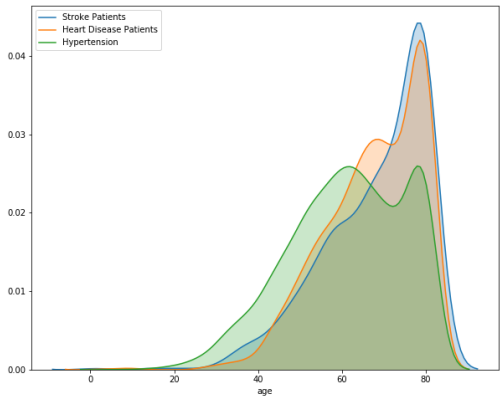



Figure 5: Age distribution of patients who suffered a stroke

### 4.3 Exploratory Data Analysis

When building predictive models, a crucial part is understanding the data before building a predictive model. For example, what columns are significant in predicting a stroke? Can we derive insights based on correlations? Can some information be discarded or can missing data be filled?

The stroke dataset contains 43400 entries of which only 783 (1.8%) are patients who suffered a stroke. Intuitively, we can assume that strokes will occur more often in adults and elderly (see fig 5). A heat-map using a balanced subset (equal numbers of stroke and non-stroke patient) of the available data shows correlations among all the data columns ( fig 6). Missed or less intuitive correlations for a non medically trained viewer become clearer.

For example, it is expected that age is highly correlated with a person's lifestyle choices and health such as being married, work type, having a heart disease or hypertension. It may not be immediately obvious that heart diseases and strokes follow an almost identical distribution with a higher likelihood at older ages (fig 5)). Hypertension starts at earlier ages and may be a key indicator for later strokes, both having a high correlation. Another less clear correlation is the age vs the body mass index (bmi): fig 4 shows how the bmi is increasing from childhood through to adulthood and decreasing again for elderly people. Interestingly, the bmi is also correlated with the work type - this could likely be explained by the fact that those who work are also more active.

Surprisingly, a strong correlation is seen among smoking status and gender (fig 7) than among smoking vs strokes or hypertension. There are more males who smoke or formerly smoked than females. The number of females who never smoked is also significantly higher. Only 1.7% of those who never smoked suffered a stroke, only slightly better than 2% of those who regularly smoke. Unexpectedly, the highest rate of strokes is for former smokers at 3%.

### 4.4 Categorical and Missing Data

The correlations and insights derived from the previous section allow us to fill in missing data in a manner that is consistent with the dataset's features. Two columns, the *bmi* and *smoking status* are missing information.

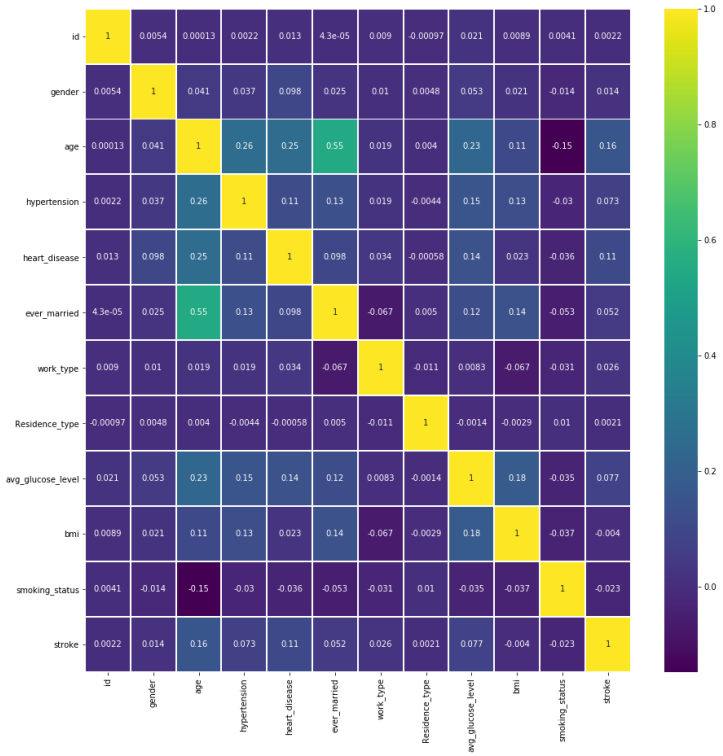



Figure 6: Correlations among all data columns using a balanced subset - smoking status is mapped to binary values

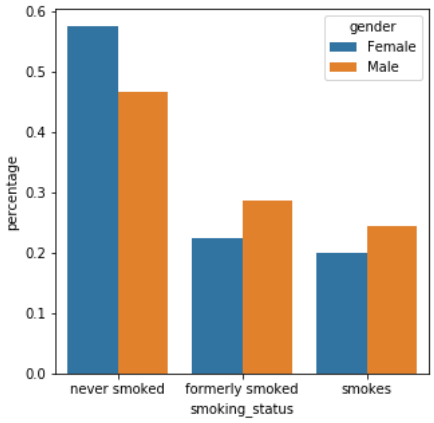



Figure 7: Percentage of male/female smokers

There are 1462 (3.3%) rows missing the bmi value. To ensure the dataset has no *nan* values, we could simply delete these rows or replace the missing bmi with an average over the entire dataset. However, the strong correlations with age and work type allow us to be more precise: replacing the bmi with the average bmi for that age group would produce a larger distribution, but the bmi has a significantly stronger correlation (0.42) with the work type, which in turn will distribute bmi average values over five types of work. Averaging the bmi of each work type to

replace missing bmi is the better choice.

The smoking column is missing in 30% of the data. Surprisingly, this is not strongly correlated with other attributes, and has a particularly low correlation with the stroke column. A reasonable approach could be to delete the column entirely, but in a realistic scenario, stakeholders would loose confidence if they learned that smoking information has been dropped in a stroke prediction product. As such, we can fill this information using an empirical approach: if the work type is children or age is less than 18, we set this values to *never smoked*. Otherwise, we can make use of the fact that males are significantly heavier smokers and thus decide the status based on gender.

The least significant column in the dataset is the id of patients. This is a numerical value that replace personal identifiable data. This has no correlations with any attributes and should be dropped.

Categorical data needs to be mapped to numerical values for an ANN to work. One hot encoding is used to avoid wrongly increasing the importance of a category based on it numerical mapping. For example, the column residence type will translate to columns *urban* and *rural* with values of 0 or 1 only. As this is duplicating the data, only one column is need and the other is dropped. For attributes with more categories, columns are created for each value with mappings of 0 or 1. This approach ensures that no categorical value has a higher rank than others.

By creating these mapping to numerical values, the ANN will need more neurons and more processing resources. Such mappings are usually evenly distributed, but in the case of gender which has three categories (Male, Female, Other) , only 11 entries are labelled as Other out of 43400 rows. One hot encoding would add unnecessary overhead on the dataset and a good approach would be to merge such values in one of the other categories. In this case however, the eleven rows can be dropped. Finally, the number of columns after the data translation increased from 11 to 15 columns.

### 4.5 Architecture of the ANN with Keras

Keras[12] is the official api library for implementing high-level ANNs in Python using tensorflow (2.0). Hidden and output layers can be easily added with parameters such as the number of neurons or activation function to be used. A model can be trained using a large number of parameters that abstract away from the granular implementation details that would otherwise require an in depth understanding of complex mathematical formulas.

The first step is to split the dataset into a training and test set. We use 80% to train and 20% to test. The validation data will only be used to measure the performance and not for any training purposes during backpropagation. The data is then sclaed and normalized using the *std* and *min/max* values of the dataset.

The number of epochs is set to 200 with an early stopping callback to avoid over fitting after the valuation loss stops improving. Given this is an imbalanced dataset, the batch size is set to 2048 to ensure the model trains on enough rows from the minority class. The loss function is binary cross entropy with Adam as the optimizer.

On the first training attempt of the model, this achieved a 98% accuracy. However, this metric is misleading due to the fact that the dataset is highly imbalanced. The precision for class 0 (no stroke) is nearly 99% while 0 for the minority class 1 (suffered a stroke). The algorithm learns

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.994689 | 0.732786 | 0.843884 | 12780.000000 |
| 1 | 0.052705 | 0.791667 | 0.098830 | 240.000000 |
| accuracy | 0.733871 | 0.733871 | 0.733871 | 0.733871 |
| macro avg | 0.523697 | 0.762226 | 0.471357 | 13020.000000 |
| weighted avg | 0.977326 | 0.733871 | 0.830150 | 13020.000000 |

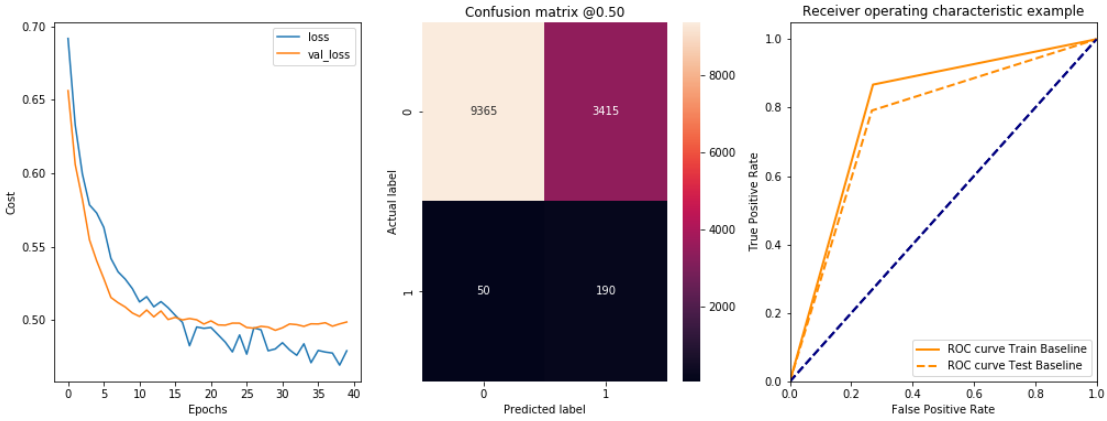



Figure 8: Weighted network

to always predict 0 regardless of the input. This is a common problem for Deep Learning. There are several approaches worth discussing to overcome this.

One solution is to attach weights to each class[10][34]. Thus, the ANN becomes sensitive to the importance of the class: the cost function is punishing the network proportionally higher when it miscalssfies a row from the under-represented class.

The networks significantly improves when the weights are set proportionally to the class distributions {class 0:0.51 and class 1:27.71}. Training is stopped early after 32 epochs (fig:**??**). The classification report in figure 9 show how label 1 now has an 83% recall score (TP/(TP +FN)) but only 4% precision (TP/(TP + FP)). If this was a real medical test, it would correctly identify most of those patients who have a stroke, but would also wrongly misdiagnose 1/3 of healthy patients.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.994689 | 0.732786 | 0.843884 | 12780.000000 |
| 1 | 0.052705 | 0.791667 | 0.098830 | 240.000000 |
| accuracy | 0.733871 | 0.733871 | 0.733871 | 0.733871 |
| macro avg | 0.523697 | 0.762226 | 0.471357 | 13020.000000 |
| weighted avg | 0.977326 | 0.733871 | 0.830150 | 13020.000000 |

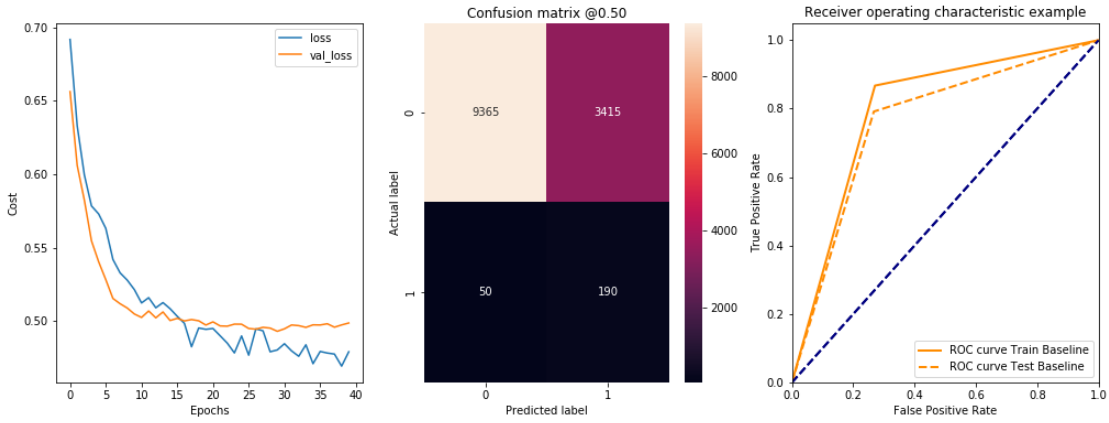



Figure 9: Weighted network

# Bibliography


[1] Activation function - wikipedia. https://en.wikipedia.org/wiki/Activation_function.

[2] Ai is sending people to jail—and getting it wrong - mit. https://www.technologyreview.com/s/612775/algorithms-criminal-justice-ai/.

[3] Algorithm assessment report. https://data.govt.nz/use-data/analyse-data/government-algorithm-transparency-and-accountability/algorithm-assessment-report/.

[4] Alphago — deepmind. https://deepmind.com/research/alphago/.

[5] Backtesting - investopedia. https://www.investopedia.com/terms/b/backtesting.asp.

[6] Coding your own algo-trading robot - investopedia. https://www.investopedia.com/articles/active-trading/081315/how-code-your-own-algo-trading-robot.asp.

[7] Did uber steal google's intellectual property? — the new yorker. https://www.newyorker.com/magazine/2018/10/22/did-uber-steal-googles-intellectual-property.

[8] A graph of a cost function - researchgate. https://www.researchgate.net/figure/A-graph-of-a-cost-function-modified-from_fig1_329920042.

[9] The how of explainable ai: Pre-modelling explainability. https://towardsdatascience.com/the-how-of-explainable-ai-pre-modelling-explainability-699150495fe4.

[10] How to develop a cost-sensitive neural network. https://machinelearningmastery.com/cost-sensitive-neural-network-for-imbalanced-classification.

[11] Kaggle. https://www.kaggle.com//.

[12] Keras. https://keras.io/.

[13] Neural network - databricks. https://databricks.com/glossary/neural-network.

[14] A neural network in 13 lines of python (part 2). https://iamtrask.github.io/2015/07/27/python-network-part2/.

[15] Professor's perceptron paved the way for ai – 60 years too soon. https://news.cornell.edu/stories/2019/09/professors-perceptron-paved-way-ai-60-years-too-soon/.

[16] Stroke dataset. https://www.kaggle.com/asaumya/healthcare-dataset-stroke-data/.

[17] The triple jeopardy of ke xu, a chinese hedge fund quant .... https://www.bloomberg.com/news/features/2018-11-19/the-triple-jeopardy-of-ke-xu-a-chinese-hedge-fund-quant.

[18] Trolley problem - wikipedia. https://en.wikipedia.org/wiki/Trolley_problem.

[19] Uber and waymo settle autonomous driving tech lawsuit. https://www.wired.com/story/uber-waymo-lawsuit-settlement/.

[20] Well-funded autonomous vehicle startups — crunchbase. https://www.crunchbase.com/lists/well-funded-autonomous-vehicle-startups/f2214864-27b0-47cf-b596-257602ab8145/organization.companies.

[21] AI aids doctors assess Covid-19 CT scans. https://www.imveurope.com/news/ai-aids-doctors-assess-covid-19-ct-scans.

[22] Maruan Al-Shedivat, Avinava Dubey, and Eric P. Xing. Contextual explanation networks, 2017.

[23] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks, 2018.

[24] Rich Caruana. Intelligible machine learning models for healthcare. https://www.youtube.com/watch?v=ezSG9GORF54.

[25] DarwinAI. https://www.darwinai.com/index.html.

[26] Sheldon Fernandez. Dark ai and the promise of explainability (part i). https://medium.com/@sheldon.fernandez/dark-ai-and-the-promise-of-explainability-part-i-a6b35009a88c.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. pages 770–778, June 2016.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[29] Lin Li, Lixin Qin, Zeguo Xu, Youbing Yin, Xin Wang, Bin Kong, Junjie Bai, Yi Lu, Zhenghan Fang, Qi Song, Kunlin Cao, Daliang Liu, Guisheng Wang, Qizhong Xu, Xisheng Fang, Shiqin Zhang, Juan Xia, and Jun Xia. Artificial intelligence distinguishes covid-19 from community acquired pneumonia on chest ct. *Radiology*, page 200905, 03 2020.

[30] Christoph Molnar. *Interpretable Machine Learning*. 2019. https://christophm.github.io/interpretable-ml-book/.

[31] Michael A. Nielsen. Neural networks and deep learning, 2018.

[32] Information Comissioner's Office. Rights related to automated decision making including profiling. https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/individual-rights/rights-related-to-automated-decision-making-including-profiling/.

[33] Linda Wang. COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest Radiography Images. *Not Published*, 2009.

[34] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P.J. Kennedy. Training deep neural networks on imbalanced data sets. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2016-, pages 4368–4374. Institute of Electrical and Electronics Engineers Inc., 2016.