NATIONAL TECHNICAL UNIVERSITY OF ATHENS

School of Rural, Surveying and Geoinformatics Engineering

MSc Geoinformatics

Remote Sensing Lab

DIPLOMA THESIS

LEVERAGING ATTENTION MECHANISM IN CLINICAL RISK ASSESSMENT: A FRAMEWORK FOR HETEROGENEOUS DATA

Supervisor: K. Karantzalos

CHRISTOPOULOS DIONYSIS

ATHENS, JULY 2024

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Σχολή Αγρονόμων και Τοπογράφων Μηχανικών - Μηχανικών Γεωπληροφορικής

ΔΠΜΣ Γεωπληροφορικής

Εργαστήριο Τηλεπισκόπησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΕΚΤΙΜΗΣΗ ΡΙΣΚΟΥ ΣΕ ΚΛΙΝΙΚΑ ΠΟΛΥΤΡΟΠΙΚΑ ΔΕΔΟΜΕΝΑ ΜΕ ΤΗ ΧΡΗΣΗ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΚΑΙ ΜΗΧΑΝΙΣΜΟΥ ΠΡΟΣΟΧΗΣ

Επιβλέπων: Κ. Καράντζαλος

ΧΡΙΣΤΟΠΟΥΛΟΣ ΔΙΟΝΥΣΗΣ

ΑΘΗΝΑ, ΙΟΥΛΙΟΣ 2024

RSLab

Remote Sensing Laboratory

National Technical University of Athens

✓ Sensing ✓ Analytics ✓ Monitoring

Acknowledgements

I would like to express my gratitude to the supervisor of this Thesis, Professor Konstantinos Karantzalos, for his guidance, support, and invaluable trust throughout this research. I am also deeply grateful to my co-supervisor, Makis Douskos, whose assistance and advice were crucial at every stage of this work. The collaboration developed within the Remote Sensing Lab was exceptional, with tactical meetings in order to discuss and review the progress of this study. Lastly, I would like to thank my family for their unwavering support and understanding, which provided me with the strength and motivation to complete this journey. Their constant encouragement was a source of inspiration and comfort.

RSLab
Remote Sensing Laboratory
National Technical University of Athens
✓ Sensing ✓ Analytics ✓ Monitoring

Abstract

In this study, we present two novel frameworks for clinical risk assessment utilizing heterogeneous tabular data, including numerical, categorical, and checkbox-type features. The architectures developed are a non-negative multi-layer perceptron (MLP), which enforces non-negativity constraints on all weights during training and a network that deploys the self-attention mechanism for enhanced feature interaction. This work involves training and evaluating the models on two distinct datasets, one focused on melanoma classification and the other on a highly imbalanced survey dataset from Behavioral Risk Factor Surveillance System (BRFSS), preprocessed for heart attack/disease classification. Additionally, we integrate a privacy-preserving pipeline, including anonymization and minimization techniques, to ensure personal data protection. The performance metrics indicate that both the non-negative MLP and the attentive network significantly outperform Logistic Regression, which is a widely used method in clinical risk assessment task. The attentive network, in particular, while effectively handling missing values, mitigates overfitting and demonstrates superior robustness. Furthermore, the attention weights generated, provide easily interpretable insights, enhancing the model's transparency during decision-making process.

Keywords: melanoma, heart attack/disease, BRFSS, non-negative, self-attention, binary classification, class imbalance

Εκτενής Ελληνική Περίληψη

Στη παρούσα εργασία, πραγματοποιείται η ανάπτυξη και αξιολόγηση δύο καινοτόμων αρχιτεκτονικών βαθιάς μάθησης, για την πρόωρη εκτίμηση ρίσκου σε κλινικά δεδομένα, τα οποία περιλαμβάνουν αριθμητικά, κατηγοριοποιημένα και δεδομένα τύπου πολλαπλής επιλογής (“checkbox”). Η πρώτη και απλούστερη αρχιτεκτονική που υλοποιήθηκε, είναι ένα πλήρως συνδεδεμένο δίκτυο, δύο επιπέδων, μη αρνητικών βαρών (non-negative MLP) (Σχήμα 4.1), το οποίο βελτιώνει την υπάρχουσα βιβλιογραφία όσον αφορά τα δίκτυα μη αρνητικών βαρών [25] με τις εξής συμβολές:

- αύξηση του βάθους και κατ’ επέκταση της πολυπλοκότητας του μοντέλου.
- τα βάρη στο επίπεδο εξόδου συνεισφέρουν στην εκπαίδευση του δικτύου, χωρίς να είναι παγωμένα, παραμένοντας βέβαια περιορισμένα σε μη αρνητικές τιμές, ώστε κάθε χαρακτηριστικό να συμβάλλει μόνο θετικά στην αύξηση του ρίσκου διάγνωσης της εκάστοτε ασθένειας.
- κατάλληλη επιλογή και παραμετροποίηση της συνάρτησης κόστους (loss function) και των αλγορίθμων βελτιστοποίησης με στόχο τη βελτίωση της βαθμονόμησης του ρίσκου που εκτιμάται από τα μοντέλα.

Η δεύτερη και πιο σύνθετη αρχιτεκτονική, παρουσιάζεται στο Σχήμα 4.2 και αξιοποιεί τον μηχανισμό προσοχής (self-attention). Ο πυρήνας της προτεινόμενης μεθόδου είναι ένας Transformer Encoder (κωδικοποιητής) με ορισμένες ιδιαιτερότητες βάσει των απαιτήσεων και προκλήσεων που εισάγουν τα κλινικά δεδομένα:

- διαχειρίζεται με επιτυχία όλα τα πιθανά είδη κλινικών δεδομένων. Τα συνεχή αριθμητικά δεδομένα περνούν από ένα πλήρως συνδεδεμένο δίκτυο δύο επιπέδων, ενώ τα κατηγοριοποιημένα και πολλαπλής επιλογής δεδομένα περνούν από προσαρμοσμένα επίπεδα ενσωματώσεων με αποτέλεσμα, προτού τροφοδοτηθούν στον Transformer Encoder, να βρίσκονται σε έναν κοινό χώρο χαρακτηριστικών.
- διαθέτει την ικανότητα να χειριστεί με επιτυχία την έλλειψη συγκεκριμένων χαρακτηριστικών, σε οποιονδήποτε τύπο δεδομένων, χωρίς να απαιτεί την αφαίρεση τους από το σετ εκπαίδευσης.

Περισσότερες λεπτομέρειες και τεχνικά χαρακτηριστικά των παραπάνω αρχιτεκτονικών παρουσιάζονται στο Κεφάλαιο 4. Τα μοντέλα αυτά συγκρίνονται με το πρότυπο μοντέλο λογιστικής παλινδρόμησης (Logistic Regression), μια μέθοδο που χρησιμοποιείται ευρέως για την εκτίμηση κλινικού κινδύνου.

Η παρούσα μελέτη περιλαμβάνει την εκπαίδευση και αξιολόγηση των μοντέλων σε δύο διαφορετικά σετ δεδομένων (Κεφάλαιο 5.1):

1. Το πρώτο είναι επικεντρωμένο στη ταξινόμηση και την πρόωρη εκτίμηση κινδύνου εμφάνισης μελανώματος, με μόλις 415 εγγραφές και 29 διαθέσιμα χαρακτηριστικά μετά από προ-επεξεργασία των αρχικών δεδομένων.

2. Το δεύτερο είναι βασισμένο στα πιο πρόσφατα διαθέσιμα δεδομένα που προέρχονται από την ετήσια τηλεφωνική έρευνα BRFSS για το έτος 2022, με πάνω από 400.000 συμμετέχοντες και 48 χαρακτηριστικά, επιλεγμένα για την ταξινόμηση καρδιακής προσβολής ή νόσου. Η μεγαλύτερη πρόκληση που εισάγει το παρών σετ δεδομένων είναι η υψηλή ανισορροπία των κλάσεων. Η αναλογία είναι 90%-10% των αρνητικών διαγνώσεων έναντι των θετικών, γεγονός πολύ συχνό στη περίπτωση των κλινικών δεδομένων, το οποίο ευθύνεται για την δημιουργία μεροληπτικών μοντέλων που δεν διαθέτουν την ικανότητα σωστής και ακριβής αναγνώρισης θετικών διαγνώσεων. Αμφότερες οι δύο προτεινόμενες μεθοδολογίες ξεπερνούν το ζήτημα αυτό με την υλοποίηση του Focal loss ως συνάρτηση κόστους.

Επιπλέον σεβόμενοι τους κανονισμούς και περιορισμούς περί ασφάλειας των προσωπικών δεδομένων, ειδικότερα στον τομέα της υγείας, ενσωματώνονται τεχνικές ανωνυμοποίησης [24] και ελαχιστοποίησης [22], που διαδέχονται την εκπαίδευση των μοντέλων.

Τα αποτελέσματα, καταγράφουν τις μετρικές αξιολόγησης, συγκρίνοντας όλα τα προαναφερθέντα μοντέλα μεταξύ τους, σε αμφότερα τα σετ δεδομένων, αποδεικνύοντας πως οι πιο σύνθετες αρχιτεκτονικές που προτείνονται σε αυτή την εργασία ξεπερνούν σε όλους τους τομείς την απόδοση της λογιστικής παλινδρόμησης, με ιδιαίτερα χαμηλό υπολογιστικό κόστος. Συγκεκριμένα, το δίκτυο που αξιοποιεί τον μηχανισμό προσοχής, παρουσιάζει ισχυρότερες ιδιότητες γενίκευσης σε κάθε πιθανό σενάριο, ακόμα και με μη-ισορροπημένα σετ δεδομένων, προσαρμόζει με επιτυχία τυχόν ελλιπή δεδομένα διατηρώντας υψηλά επίπεδα ακρίβειας στις προβλέψεις του. Τέλος, παρέχει αποτελέσματα, εύκολα ερμηνεύσιμα μέσω των βαρών που εξάγονται από τον μηχανισμό προσοχής, ενισχύοντας τη χρηστικότητα του μοντέλου ως βοηθητικό εργαλείο στη διαδικασία λήψης αποφάσεων ενός κλινικού ιατρού ή οποιουδήποτε ενδιαφερόμενου, χωρίς να χρειάζεται εξειδικευμένες γνώσεις στον τομέα της τεχνητής νοημοσύνης.

Contents

CHAPTER 1

Problem Definition, Motivation and Challenges

In the realm of healthcare, taking into account the rapidly evolving landscape of deep learning, the need of accurate and robust risk assessment tools is significant. In domains such as melanoma, heart disease and stroke early detection, the tools proposed in this work, aim to provide a reliable risk score prediction, based on the available clinical data. In terms of neural networks, the problem could be defined as a binary classification task, on tabular data, aiming to predict the likelihood, interpreted as the probability of the target positive label, associated with the corresponding critical health conditions.

This work is driven by several key motivations. First and foremost, the significance behind the nature of the task itself. The capability to evaluate a risk score for various diseases with a single global framework, is a really crucial matter, as it enables early intervention and medical treatment in a personalized manner. Regarding the technical domain, we extend the typical machine learning models and take neural network architectures like Multi-Layer Perceptron (see 2.1.3) a step further, proposing two different architectures. The first and simpler one, was inspired by the interpretability of non-negative networks, that aligns well with the need to understand causality in clinical data. The second architecture, took inspiration by the potential of attention mechanism, used in transformer-based models, leveraging its explainable property, in order to finally explore the underlying relationships within healthcare conditions. A more detailed analysis about the proposed models is provided in Chapter 4. The two options mentioned above, have the ability to deliver robust results with high accuracy, outperforming all machine learning models and standard MLP baselines. Additionally they possess the property of explainability, enabling any interested party, who are not experts in computer vision and AI domain, to make use of and interpret the results. Besides, the ultimate purpose of this task is to support the decision-making process of the corresponding clinician.

However, this task is not without its challenges, especially when dealing with healthcare data. The main limitation is the access to high-quality, meaningful and with sufficient samples, datasets due to privacy concerns and data regulations. Additionally, clinical datasets, even if accessed, tend to be very imbalanced, with a disproportionately small number of positive instances rela-

tive to negative ones (a typical ratio is 10%-90% respectively). Handling class imbalance, while maintaining high accuracy and generalizability of the models, is a very crucial aspect in deep learning. Finally, healthcare data, most of the times exhibit a heterogeneous nature comprising a combination of numerical (i.e age, height, weight etc.) and categorical features (i.e eyes, hair, clinical check frequency etc.) or even "checkboxes", where multiple fields within the same category can exist simultaneously (i.e ancestry, doctors usually visited etc.). Ways to mitigate each one of those key challenges, are described extensively during Chapters 4 and 5.

CHAPTER 2

Introduction

2.1 Basic Theory - Notations

In this section, we will cover some of the most important concepts, both simple and complex, that we will encounter throughout this work. These notations will aid the reader in fully understanding the content of this study and the logic behind each decision made in the architecture (Chapter 4), how the training framework is defined (Section 5.2) and the meaning of the ablation study (Section 5.4).

Machine Learning: The science that explores the design and construction of mathematical models, which, through the use of suitable algorithms, approximate ("learn") an unknown function/distribution directly from the provided data and make decisions, on new and unseen data, based on the assigned goal. In their application, the human factor is limited solely to supplying the data. The goal of machine learning is therefore to understand the structure of the data and adapt theoretical functions/distributions to it. For this purpose, iterative approaches are usually used, feeding data into the model, helping it adjust its parameters, based on the observed errors between predicted and actual outcomes, until the algorithm "learns" a strong pattern that adapts to the data.

Deep Learning: Deep Learning is considered a subset of Machine Learning, that consists of more complex and deeper models, known as "Neural Networks", which comprise multiple layers. Their purpose is to learn high-level representations on input data, leading to a greater number of trainable parameters compared to traditional machine learning algorithms. Consequently, they require higher computational resources while training and inference times are typically longer. It is worth mentioning that the methodology proposed in this study falls under the supervised Deep Learning algorithms exploiting the MLP architecture which will be explained in this section and the self-attention mechanism (see Section 2.2).

Machine learning can be broadly divided into four major categories depending on the nature

and structure of the training data, or the model's supervision during the aforementioned iterative learning process:

- **Supervised Learning:** The most straight-forward category, in which the algorithm receives labeled training data, where each input is associated with a corresponding target outcome. Its goal is to learn the function (mapping), that connects the input data with the known target labels. Typical applications of supervised learning include classification, regression, and prediction of future states.
- **Unsupervised Learning:** In this scenario, the machine receives unlabeled training data, meaning that it does not have explicit guidance on the desired output. Its goal is to autonomously learn the underlying structure and possible patterns or relationships within the input data.
- **Semi-supervised Learning:** This is a combination between the two previous approaches. Commonly, semi-supervised learning receives relatively small amount of labeled data together with a much larger amount of unlabeled data. This method is utilized when the cost of feeding the training algorithm with a full set of labeled data is very high or time consuming. It can be used for the same applications as supervised learning.
- **Reinforcement Learning:** The algorithm interacts with a dynamic environment where a specific goal must be achieved. Through trial and error and with a reward-penalty system, it learns which process yields the most optimal outcome, and adjust its behavior in order to maximize the cumulative reward over time. The main domains where reinforcement learning is mainly utilized, are robotics, gaming, and navigation.

In order to solve the task at hand, some standard steps and decisions should be considered:

a. **Data Pre-processing and Splitting:** The raw data collected, may need a step of pre-processing in order to meet the requirements of the model, which will be deployed. Such steps are normalization of values, handling missing values etc. After being processed, the dataset is splitted into subsets for training, validation and testing. A common split ratio is 80% for training and 20% for validation/testing, although this is not absolute and can be modified based on the nature and the size of the dataset.

b. **Model's architecture and design:** the model's architecture should be selected based on the task and the unique nature of input data. This step involves choosing the correct general architecture along with its interior parameters like the number of layers, controlling how deep the model will eventually be. Also, during this step, regularization techniques can be integrated into the model, like Dropout or L1/L2 regularization, to avoid overfitting.

c. **Loss function and Optimizer:** In this step the most suitable loss function and optimization algorithm are defined. The first, quantifies the difference between the model's predic-

tions and the ground truth (target) label, while the latter updates the model's parameters, in order to minimize the loss value.

d. **Iterative Training Loop:** Utilizing only the training data, the model is trained for a number of iterations (epochs). During every epoch, the model make predictions (forward pass) on the input training data, calculates the loss value, and consequently updates its learnable parameters (backpropagation).

e. **Evaluation:** Commonly, throughout the training process the model's performance is evaluated on the validation set, which contains samples, unseen during the training loop. The evaluation protocol is defined differently for each task with the appropriate metrics (accuracy, F1-score, PSNR, MSE, etc.). This step is extremely crucial to detect overfitting or underfitting problems.

f. **Inference:** Finally, after the model is fully and properly trained, it utilizes the test dataset, which contains new and unlabeled data, to obtain the desired prediction outputs.

In the subsequent paragraphs of this section we will briefly mention some oh the most popular ML/DL architectures, activation functions, loss functions and optimizers, while explaining their differences and use cases.

2.1.1 Supervised Machine Learning Algorithms

One of the simplest machine learning algorithm is **k-Nearest Neighbor** classifier, which memorizes input training data and during inference predicts the label of a sample, by finding the label, with majority vote, of its k closest/similar train samples, using either L1 (Manhattan) or L2 (Euclidean) distance. This algorithm is slow during inference since it needs to take all the training data into account. Specifically for N examples it has a cost of $\mathcal{O}(N)$. **Support Vector Machines** (SVMs) are classifiers effective for both linear and non-linear tasks, that identify the optimal hyperplane to distinguishably separate classes in high-dimensional space. **Decision Trees** are a non-parametric supervised method used for classification and regression tasks. The purpose is to create a model, that learns the underlying rules that characterize the input training data features. Desicion Trees are useful, since they are very simple to implement, are self-explained and handles both numerical and categorical data. However, due to its non-parametric nature, they tend to be affected by outliers and sparse input data, resulting in a not generalizable model. Finally, **Logistic Regression** is a widely used classifier for binary tasks, where the goal is to predict the probability (or risk score) of an observation to belong in a particular class. It utilizes sigmoid function (Figure 2.2 [left]) to map the values between 0 and 1, representing the probability of the positive class.

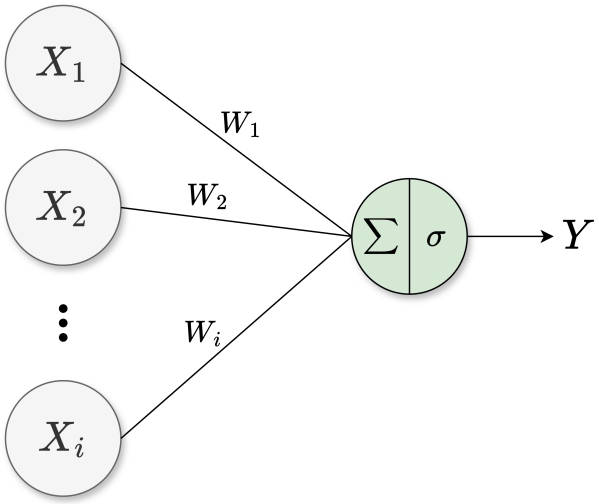


Figure 2.1: Single Layer Perceptron where i the number of inputs and σ the non-linear activation function.

2.1.2 Perceptron and MLPs

Perceptron is the simplest form of a neural network, working as a linear binary classifier in supervised deep learning tasks. It receives external input data, multiplying them with corresponding weights and adding a bias term, so they are combined linearly. Typically these outputs pass through an activation function, which introduces non-linearity to the network (see Figure 2.1). Extending the concept of a perceptron, Multi-layer Perceptrons (MLPs) combine multiple layers of neurons, also known as "Fully Connected" or "Dense", because every neuron in a layer interacts with all neurons of the subsequent layer. An MLP consists of an input layer, a number of hidden layers, each with possibly varying number of neurons and finally an output layer. MLPs can solve multiclass supervised tasks, where the number of neurons on the output layer equals the number of classes on the task.

$$\hat{y}_j = \sigma\left(\sum_{i=1}^{n} W_{ij}x_i + b_j\right) \tag{2.1}$$

where $\hat{y}_j$ is the output of j-th neuron, σ the selected activation function, x_i the i-th input to the neuron, W_{ij} the weight matrix between input i and neuron j and b_j the bias term of the neuron.

2.1.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks consist a class of deep learning models, used widely in computer vision tasks such as image classification, segmentation and object detection. Their main block is convolutional layers that use learnable filters (kernels) to apply convolutions to the input data, typically images, in order to export meaningful representations and underlying patterns. These

kernels are moving windows that slide over the input image performing convolutions in each step. For each convolutional layer, a set of hyperparameters need to be defined:

- the filter's size (1×1, 3×3, 5×5 etc.).
- the number of filters in each layer.
- stride, which defines the step by which the filter (kernel) is moved across the input image during the convolution operation. It is typically set to 1, although for bigger stride values, the smaller the spatial dimension of the output feature map becomes.
- padding, which refers to the amount of pixels added to the borders of the input image, before applying convolutions. It is used to control the spatial dimension of the output feature map and if no padding is applied then the output will be smaller than the input image.

Typically after each convolutional operation, a non linear activation function and a pooling layer are applied to reduce the dimenstionality of the resulting feature map, while maintaining important information. Max pooling and average pooling are the two most commonly used pooling layers. After the final convolutional layer of the network, an adaptive pooling (either max or average) is applied to control the spatial resolution of the final representation. Finally the output is flattened into a vector, that contains high level information, and subsequently pass through a set of Fully Connected layers, since the dimensionality is greatly reduced in comparison with the input image. All of the above describe CNNs in general, but more specifically, some of the most popular architectures include AlexNet [5], GoogleNet [8], VGGNet [7], ResNet [9] and U-Net [6].

2.1.4 Non-linear Activation Functions

Non-linear Activation Functions are significant components of all neural networks, as they are responsible for introducing non-linearity into the model's architecture, allowing it to learn complex patterns and relationships on the input data. They decide, whether a neuron should be activated or not, based on a threshold set by each specific function. Without them, neural networks would always be representing linear transformations of the input data, restricting the model's power and ability to generalize effectively.

- **Sigmoid:** It squashes the input values to range $[0, 1]$ making it a popular choice in tasks like binary classification, where the output needs to be interpreted as a probability. Mathematically it is defined as:

$$\sigma(x) = \frac{1}{(1 + e^{-x})} \tag{2.2}$$

  The limitations that arise, is that sigmoid suffers from the "vanishing gradient" problem, meaning that for very large or very small input values, the gradient becomes almost zero,

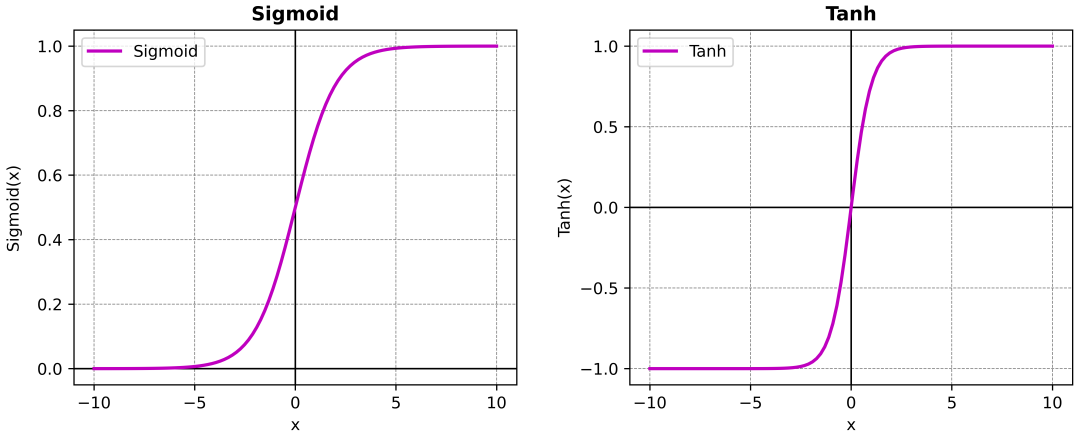


Figure 2.2: Sigmoid and Hyperbolic Tangent activation functions.

deactivating the corresponding neuron and slowing the learning process. Additionally, its outputs are not zero centered, which could result in a convergence issue in some optimization algorithms.

- **Hyperbolic Tangent:** Commonly denoted as **tanh**, is very similar to the Sigmoid function, with the main difference that its output range is $[-1, +1]$, acquiring a useful property where the outputs are zero centered. However, tanh does not overcome the "vanishing gradient" problem. Mathematically it is defined as:

$$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})} \tag{2.3}$$

- **Softmax:** It is commonly used in multiclass classification tasks. Similar to the sigmoid activation function, Softmax returns the probability of each class, as a ditribution of probabilities $\sigma \in [0, 1]$ where all sum up to 1.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \tag{2.4}$$

where z_i represents the input vector, z_j the output vector and N the total number of possible classes defined in the task.

- **ReLU (Rectified Linear Unit):** One of the most widely used activation functions due to its simplicity and efficiency. ReLU sets every negative input value to zero, while maintaining the positive values unchanged. Thus, it follows the equation $f(x) = max(0, x)$ and the range the values it could possibly produce as outputs is $[0, +\infty]$. ReLU, also, converges much faster than sigmoid and tanh, which is a significant advantage in training. The main drawback of ReLU, known as "dying ReLU" problem, is the possibility that, since all outputs can be zero, then all neurons will become inactive, which essentially stops the learning process.

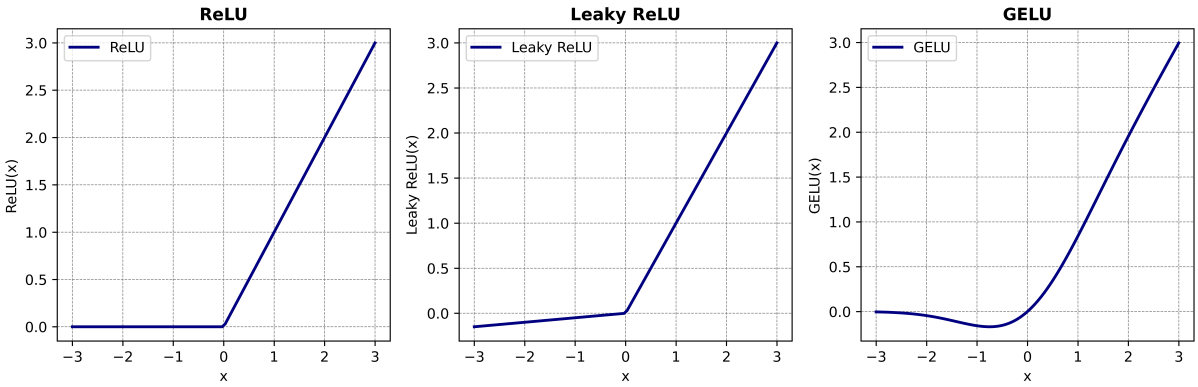


Figure 2.3: ReLU, Leaky ReLU and GELU activation functions respectively.

- **Leaky ReLU:** A variation of ReLU function, that mitigates the "dying ReLU" issue, by employing a small, positive gradient for the negative inputs. It follows the equation $f(x) = max(ax, x)$ where typically $a = 0.01$, with range of values $[-\infty, +\infty]$. This non-zero slope in the negative inputs prevents the neurons from being completely inactive, even if all the inputs are negative.

- **GELU (Gaussian Error Linear Unit):** An activation function that has gained popularity, mainly because of its smoothness across the full range of input values, also mitigating the vanishing gradient problem [27]:

$$GELU(x) = x \times \Phi(x) \tag{2.5}$$

  where $\Phi(x)$ is the Cumulative Distribution Function (CDF) for Gaussian Distribution. GELU is a good choice in deeper and more complex deep learning tasks such as natural language processing (NLP), speech recognition and in most Transformer architectures. GPT-3 [19] and BERT [15] are some of the most popular models utilizing GELU activation function.

2.1.5 Loss Functions

As mentioned before, "loss functions" or "cost functions", provide a method to quantify the distance between the model's predicted output with the actual corresponding value, in order to evaluate how well the model performs on the data. The main goal of the training process is to minimize the loss function's output while maintaining increasing accuracy on unseen data. The choice of an appropriate loss function depends mainly on each specific task and the nature of the available data.

- **Mean Squared Error (MSE):** Also known as L2 loss, MSE is a widely used loss function in deep learning tasks. It computes the average of the squared differences between the predicted and true values. Due to the squaring operation MSE loss results in higher

penalties, as deviations from the actual values grow bigger, but it is prone to outliers on the data. Mathematically it is defined by Equation 2.6 and it is a popular choice in regression tasks.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.6}$$

where n is the number of samples, y_i the target value for the i-th sample and $\hat{y}_i$ the model's predicted value for the ith sample.

- **(Binary) Cross Entropy:** Binary Cross Entropy loss (BCE) and Categorical Cross Entropy loss (CE) are typically used in most binary and multiclass classification problems respectively. BCE loss (or else Log Loss) compares each predicted probability from the model, with the actual target value, which can either be 0 or 1:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{2.7}$$

Similarly Categorical Cross Entropy is utilized in multiclass classification problems where the target labels are one-hot encoded (in case they are not, Sparse Categorical Cross Entropy could be used):

$$CE = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}\left[y_{ij}\log(\hat{y}_{ij})\right] \tag{2.8}$$

where n, m the number of samples and classes respectively, y_i the target value for the ith sample and $\hat{y}_i$ the model's predicted value for the i-th sample. In the context of CE y_{ij} is the ground truth, one-hot encoded, probability of sample i for class j (could be either 0 or 1) and $\hat{y}_{ij}$ the model's predicted probability that sample i belongs to class j.

- **Focal Loss:** Focal loss is specifically deployed in this work as it is known to handle class imbalance better than Binary Cross Entropy, while resulting in more calibrated models. A more detailed explanation of Focal loss and its properties is demonstrated on Section 5.2.

2.1.6 Optimizers

Optimizers are algorithms used to adjust the trainable parameters of a neural network, by updating the model's weights in the direction that minimizes the loss function (Figure 2.4). A key hyperparameter of optimization is learning rate "lr", which determines the size of each step taken during the parameter updating. Finding the optimal learning rate could be challenging and depends on several factors such as the optimizer and loss function, the model architecture and the nature of training data. Too low learning rate values lead to slow convergence and

too high values lead to instability with the danger that the model will never achieve the global minima and it won't converge. Some of the most commonly used optimization algorithms are:

- **Stochastic Gradient Descent (SGD):** computes the gradient using B number of mini-batches, with common mini-batch sizes $\in (32, 64, 128, 256, 512)$ and a fixed learning rate. However SGD can produce high variations in parameter updates, leading to slow convergences, while many variations like "Momentum SGD" have been proposed to mitigate these issues.
- **Root Mean Square Propagation (RMSProp):** it typically used in training deep and complex neural networks. RMSProp mitigates the issue of "vanishing gradients" by adapting the learning rate for each parameter. It specifically utilizes a weighted moving average of the squared gradients for each parameter, meaning that when it encounters high variance gradients it reduces the learning rate, in contrary with encountering low variance gradients where it allows bigger steps by increasing learning rate. Equation 2.9, shows the moving average calculation, while the parameters update follows Equation 2.10.

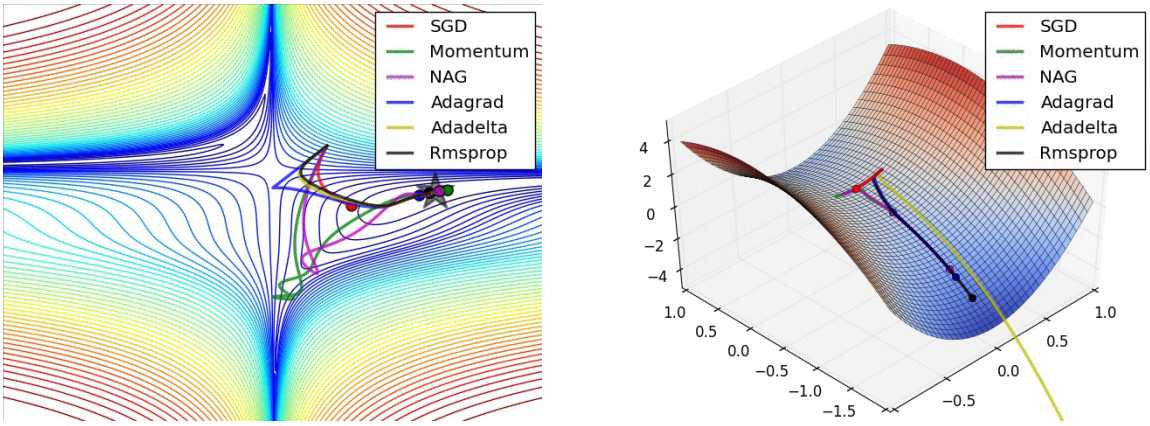
$$E[g^2]_t = \beta E[g^2]_t + (1-\beta) g_t^2 \tag{2.9}$$

$$w_t = w_{t-1} - \frac{\alpha}{\sqrt{E[g^2]_t + \epsilon}} g_t \tag{2.10}$$

  where $E[g^2]_t$ the moving average of the squared gradients during time step t, g_t the gradients at time step t with respect to the weights w, α the initialized learning rate, β the decay rate of the moving average value which typically set to 0.9 and ϵ a small constant for numerical stability.
- **Adam [10]:** a combination of momentum SGD and RMSProp, that achieves robust and adaptive learning rates. Utilizing two moving averages of gradients, it adapts a learning rate for each parameter, which leads to faster and numerically stable convergence.

2.1.7 Overfitting

In previous paragraphs we referred to a term called "overfitting" - but what exactly does that mean? The answer is simple, overfitting is a common challenge in ML/DL models and it occurs when a model learns to capture outliers or random trends in the training data, rather than the underlying pattern. This can happen when the model becomes overly complex or it is initially deeper than needed, fitting to the training data extremely well, but resulting in poor generalization abilities when presented with new, unseen data (during validation step). To mitigate this issue, techniques such us regularization, cross validation and early stopping of the training process, can be utilized.



(a) Optimizers on 2D loss surface contours.

(b) Optimizers on 3D saddle point case.

Figure 2.4: **(a)** Illustrates how different optimization algorithms converge over time on a 2D representation of a loss surface. **(b)** Represents the behavior of the same optimization algorithms in a saddle point case, meaning that the gradient is zero in that point but it's neither a minimum or maximum. By tracking the footprints it is observed that SGD, Momentum and NAG have difficulties or they even don't converge at all when they reach a saddle point, while most recent techniques like Adagrad, Adadelta and RMSProp quickly converge to the negative slope.

2.1.8 Regularization & Imbalanced Dataset Handling

Dropout is simple regularization technique commonly used in neural networks to prevent overfitting. With dropout, a random proportion of the neurons in the training are "killed", meaning that their outputs are set to zero. For each training epoch, a new random set of neurons are selected to be deactivated, forcing the model to learn more robust representations of training data and not rely solely on a single neuron. Of course, the convergence of the model becomes slower, but training time per epoch becomes faster due to the smaller amount of activated neurons. Dropout is a method used only in training and it is usually deactivated while making new predictions during inference.

Batch Normalization is, also, a very commonly used technique, in neural networks, that is typically placed after the activation function outputs and improves both training stability and speed. This is achieved, by normalizing the activations of each layer in a mini-batch, so that they have a mean of zero and standard deviation of one. This normalization, is applied to each feature dimension independently. Subsequently, the normalized activations, are scaled and shifted, using a set of learnable parameters γ and β respectively. The mathematic equations 2.11 - 2.14 describe this process in detail.

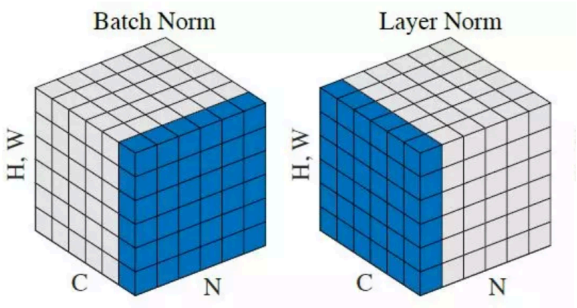


Figure 2.5: Batch Normalization - Layer Normalization

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.11}$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2 \tag{2.12}$$

$$x_{i,norm} = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{2.13}$$

$$BN(x_i) = \gamma \cdot x_{i,norm} + \beta \tag{2.14}$$

where n represents the mini-batch size, μ, σ the mean and standard deviation of the mini-batch elements, ϵ a small fixed constant variable added for numerical stability and γ, β the layer's learnable parameters.

Layer Normalization is another technique used to normalize the activations of each layer in a neural network. Batch normalization, normalize the values across the mini-batch dimension, on the contrary with layer normalization which operates along the feature dimension. This means that layer normalization computes the mean and standard deviation of the activations along the feature dimension for each example in the mini-batch separately. Finally, its equations are identical to batch normalization, with the difference that they are computed in another axis as shown in Figure 2.5. Layer normalization is widely used for NLP tasks in architectures like Transformers (see Section 2.2), where the input data length may vary.

Cross-validation for imbalanced datasets: In Machine Learning, a common technique used to train a robust and generalizable model, is cross-validation. It divides the dataset into multiple subsets, where typically one of them is the validation set and the remainder consist the training

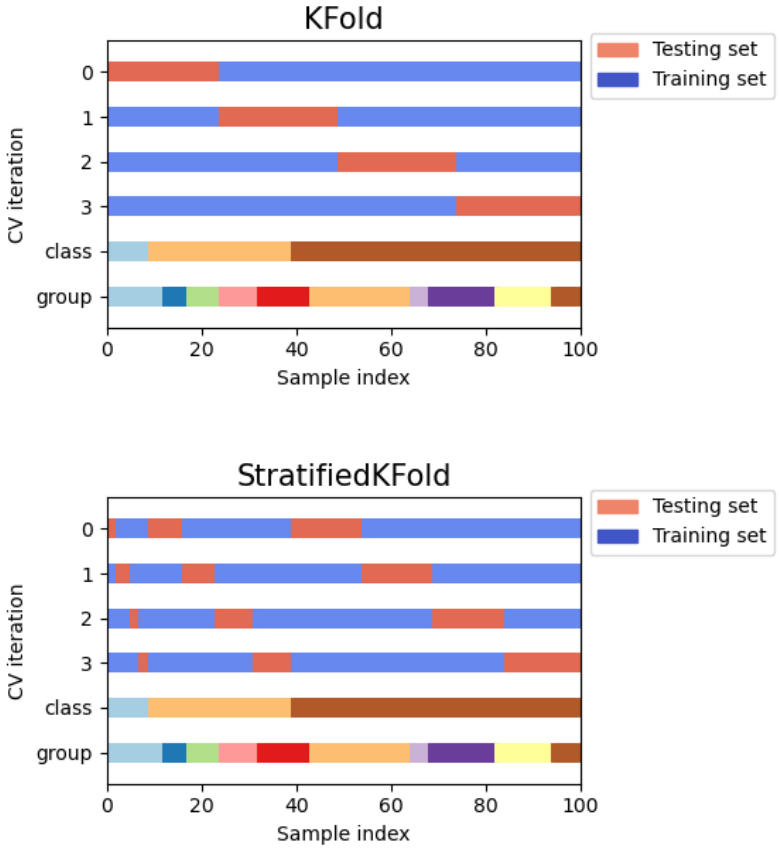


Figure 2.6: K-Fold vs Stratified K-Fold cross-validation. Source: scikit-learn

set. This process is repeated multiple times, with different combinations of subsets to ensure a more comprehensive evaluation of the model's performance.

Between the different methods of cross-validation, Stratified K-Fold is an enhancement of the traditional K-Fold, able to handle imbalanced datasets. In short, Stratified K-Fold ensures that each fold preserves the same class distribution as the original dataset. In standard K-Fold cross-validation, random sampling is used to split the subsets. Supposedly the original dataset is imbalanced, there is a great risk that the output subsets won't retain the class distribution and certain classes will be significantly less represented than others. This problem is averted with the stratified sampling, with the aforementioned property of preserving the class distribution in each fold, reducing the risk of biased model evaluation.

2.2 Transformers

Building upon our previous discussions of foundational algorithms and architectures in the field of machine learning and deep learning, such as linear regression, desicion trees, multilayer perceptrons (MLPs) and convolutional neural networks (CNNs), this section shifts our focus to self-attention mechanism and the Transformer architecture. These methodologies hold particular significance for understanding and effectively modeling sequential data within the broader context of machine learning and can be extended in various computer vision tasks such as classification through the use of Vision Transformers (VITs), language modeling and structured data analysis.

We begin by exploring in depth the self-attention mechanism (2.2.1), a key innovation in modern sequence modeling. Self-attention allows the model to dynamically focus on different parts of the input sequence, capturing complex relationships and long-range dependencies within the data, more effectively than traditional recurrent models.

Next, we delve into the Transformer architecture (2.2.2). By incorporating self-attention layers in conjuction with feed forward neural networks, Transformers offer an efficient and flexible approach to understanding and processing sequential data across various domains, particularly in language modeling tasks, where it excels at understanding and generating natural language text.

Additionally, Tansformers have been extended to computer vision through the use of Vision Transformers (ViTs). By representing images as sequences of patches and applying self-attention, ViTs effectively capture both local and global patterns in images. This architecture has proven successful in a range of supervised vision tasks, including image classification, image segmentation and object detection as well as self-supervised learning, where models learn representations from unlabeled data. These are just some examples of the diverse applications where Transformer-based models can be utilized.

Through the lens of sequence modeling and Vision Transformers, this chapter ties these innovative methods back to the domain of tabular data, illustrating the potential for Transformer-based models to enhance tasks such as prediction and classification in structured data.

2.2.1 Self-Attention mechanism

The self-attention mechanism operates by taking an input sequence of n elements and returns an output sequence of the same length n. This process allows the inputs to interact with each other and determine which parts of the input information to focus on more and which to focus on less, based on the context relevance. This is achieved by computing attention scores that represent the relevance of each input element with respect to every other element in the sequence. The final output for each input is a weighted sum of the values, where the weights are determined by the attention scores. Let's take a closer look at how this mechanism operates through several simple steps:

Scaled Dot-Product Attention

Step 1. Input Data and Weight Initialization: Assume we have an input sequence of n elements, each of dimension d, forming a matrix $X \in \mathbb{R}^{n \times d}$. Consequently we initialize the weight matrices $W_Q \in \mathbb{R}^{d \times d_q}$, $W_K \in \mathbb{R}^{d \times d_k}$ and $W_V \in \mathbb{R}^{d \times d_v}$.

Step 2. Q, K, V computation: During this step, the inputs are multiplied with the corresponding learnable weight matrices as illustrated in the equations 2.15. The resulting three sets of vectors represent the queries (Q), keys (K) and values (V).

$$Q = X \cdot W_Q \qquad K = X \cdot W_K \qquad V = X \cdot W_V \tag{2.15}$$

where $Q \in \mathbb{R}^{n \times d_q}$, $K \in \mathbb{R}^{n \times d_k}$, $V \in \mathbb{R}^{n \times d_v}$ and $d_q = d_k$. The dimension d_v may differ from the other two and always equals with the dimension of self-attention output.

Step 3. Attention Scores computation: During this step, vector q, corresponding to a single input element, is multiplied by the vector K, which corresponds to every element of the input, including itself. This process is known as dot product attention.

Step 4. Normalizing Attention Scores: This intermediate step, (proposed by [13]), involves dividing the attention scores from the previous step, by the square root of the dimension d_k. This process is referred to as scaled dot product attention (see Figure 2.7). Utilizing the ouputs of Step 3, without applying this normalization, can lead to large attention scores, resulting in very small values after applying the Softmax function during the next step. This effect is mitigated by considering the factor $(\frac{1}{\sqrt{d_k}})$.

Step 5. Applying Softmax: The Softmax function is applied on the results from the previous step.

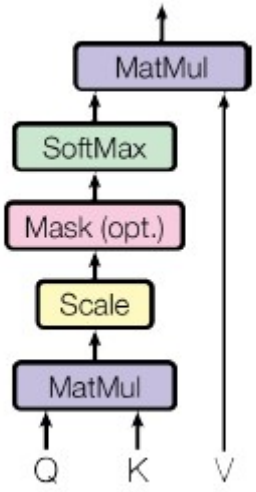


Figure 2.7: Scaled Dot-Product Attention.

Step 6. The outputs of the Softmax function are multiplied by the value matrix V, resulting the weighted values.

Step 7. Aggregated Results: This final step, involves summing up the weighted values, element-wise, in order to obtain the final output of the attention mechanism, corresponding to the desired input. The process from Step 3 onwards is repeated for each input element separately.

The above pipeline is summarized in Equation 2.16:

$$Attention(Q, K, V) = softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{2.16}$$

Multihead Attention

Extending the Scaled Dot Product mechanism, the multi-head self-attention method (see Figure 2.8) has been proposed by using multiple sets of weight matrices to produce different sets of queries, keys and values. Essentially, this approach is quite similar to the previous one, except that now multiple parallel and independent self-attention layers are utilized. The number of heads h employed in parallel each time, is not predetermined and is considered a hyperparameter of the model. Let's summarize the multi-head operation:

Step 1. We should first determine the number of heads h to be used. Then each head has its own weight matrices, initialized as previously described.

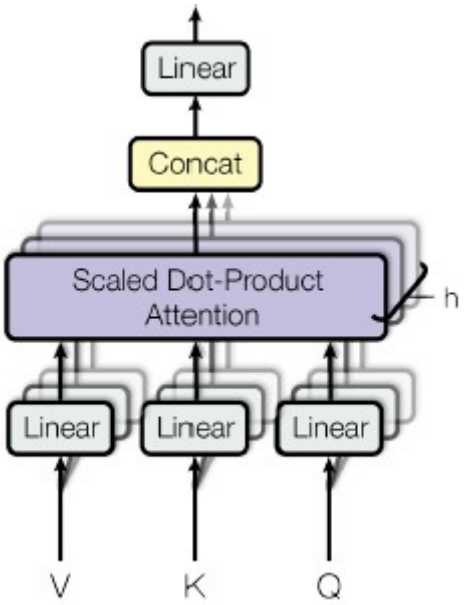


Figure 2.8: Multihead Attention.

Step 2. Extract vectors Q, K and V by multiplying the inputs with the weight matrices for each head separately. The self-attention mechanism is then applied to these vectors independently.

$$Q_i = X \cdot W_i^Q \qquad K_i = X \cdot W_i^K \qquad V_i = X \cdot W_i^V \tag{2.17}$$

$$head_i = Attention(Q_i, K_i, V_i) \tag{2.18}$$

for $i \in \{1, ..., h\}$ representing the index of the specific attention head.

Step 3. Finally, during the last step, we concatenate all the intermediate outputs from each attention head, into a single output, which is consequently multiplied by the weight matrix W^O, in order to obtain the desired final vector as shown in Equation 2.19.

$$MultiHead(Q, K, V) = concat(head_1, head_2, ..., head_h) \cdot W^O \tag{2.19}$$

The multi-head self-attention method enhances the model's ability to focus on different elements and positions in the input data, and it is a part of the Tansformer architecture, as well as of the framework proposed in this study.

2.2.2 Transformer Architecture

Both self-attention mechanism and the overall Transformer architecture were initially introduced in 2017 [13] as a novel approach in sequence-to-sequence tasks. Originally proposed for auto-

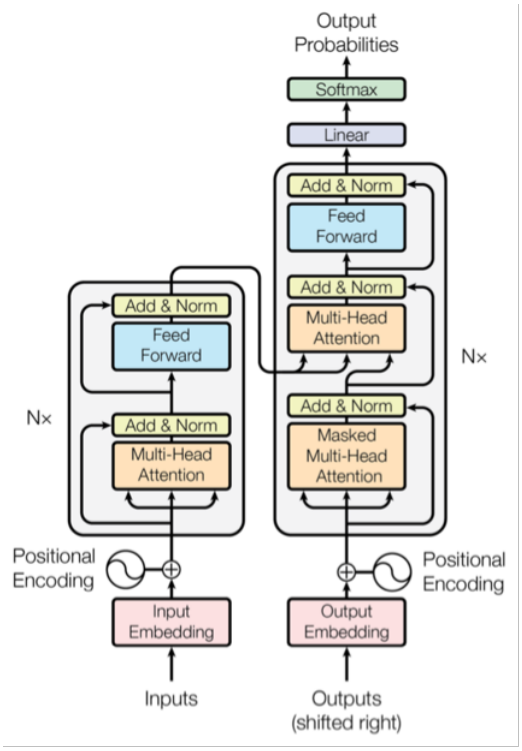


Figure 2.9: Transformer Architecture. Source: [13]

matic text translation, the Transformer has since been adapted for a wide range of applications due to its flexibility and powerful performance.

In this section we will briefly review the Transformer architecture. As shown in Figure 2.9, it employs an Encoder-Decoder framework, where both modules operate in parallel, consisting of N identical layers.

Encoder

Each one of the N Encoder layers comprises two main components:

- **Multi-Head Attention:** As we described in detail on 2.2.1, the multi-head attention mechanism, allows the encoder to focus on different parts of the input sequence and by capturing long-range dependencies and varying attention across the sequence, it forms a comprehensive understanding of the input.
- **Feed-Forward Neural Network:** A fully connected network (as described in 2.1) with two linear layers and a non-linear activation function (commonly ReLU) in between them.

Additionally, it incorporates residual connections around each component, which are then followed by layer normalization. This technique helps stabilize training and mitigate the vanishing gradient problem, allowing the utilization of deeper models. Each component has the form of:

$LayerNorm(x+Sublayer(x))$ where $Sublayer$ could either be the multi-head attention block or the feed forward network and x their corresponding input. Notably, unlike RNNs, the feed forward network accepts as inputs one attention vector at a time, and since they are independent of one another, that makes the whole Encoder block parallelized. Finally, the Encoder's output, full of contextual understanding on the input sequence is now passed as input in the Decoder module.

Decoder

The decoder module, comprises N identical layers, similarly to the encoder module but with some variations:

- **Masked Multi-Head Attention:** This block is similar to the encoder's multi-head self-attention block, except here it is masked in order to prevent the decoder from attending on future (subsequent) positions in the sequence. This preserves causal dependencies and ensures that a prediction on position i depends only on the positions $< i$. The masking is typically applied by assigning the value $-\infty$ in the attention scores of the subsequent tokens, before they are passed through the Softmax activation function.
- **Encoder-Decoder Attention:** A second multi-head self-attention block, in which the queries Q come from the decoder's masked attention block, while keys K and values V originate from the output of the final encoder layer. This is called **cross attention** and allows the decoder to gain access into relevant parts of the encoded input sequence, acquiring global receptive field, when generating the desired output.
- **Feed-Forward Neural Network:** A fully connected network exactly similar with the one consisting the encoder and applied independently to each positional element of the sequence.

Following the same style as the encoder, all the three decoder components (sub-layers) include a residual connection and each one is followed by a layer normalization step. Finally the Nth Decoder layer output is passed into another linear layer, with size equal to the number of classes that exist in each specific task (e.g the length of vocabulary used in language modeling for the task of translation). This linearly transformed vector is finally passed through a softmax activation function in order to generate probabilities over the available classes.

It is important to note that the computational complexity of self-attention mechanism is typically $\mathcal{O}(n^2)$, where n denotes the input sequence length, meaning that it depends quadratically on the sequence elements and this can lead to prohibitively large model sizes. In order to address this issue, several studies propose more efficient transformer variations like Axial Transformer

[16] which uses a pair of row and column attention blocks, maintaining the global receptive field and providing $\mathcal{O}(\sqrt{n})$ savings. A detailed survey on such techniques can be found in [26].

Embeddings

Transformers use learned embeddings in order to convert the input and output tokens into vectors of dimension d_{model}. These embeddings map the discrete token values (words, symbols, pixels, categorical features etc.) into numerical matrices, creating a representation of the semantic meaning of tokens in a continuous space.

Positional Encoding

Self-attention mechanism itself does not account the order of tokens while working with sequential data, since it has no recurrence or convolutions. To this end, Transformers make use of positional encodings applied to the input and output embeddings before the encoder and decoder respectively, adding the information about each token's position in the input sequence. Specifically the positional encodings have the same dimension d_{model} as the embeddings and could be learnable or fixed. In [13] sine and cosine functions of different frequencies are used to encode positional information:

$$PE_{(pos,2i)} = sin\left(\frac{pos}{10000^{2i/d}}\right)$$

$$PE_{(pos,2i+1)} = cos\left(\frac{pos}{10000^{2i/d}}\right)$$

where i is the dimension of the embedding and pos the corresponding position.

2.3 Privacy in AI

As Artificial Intelligence (AI) continues to evolve rapidly each year, modern machine learning and deep learning frameworks often require access to and processing of vast amounts of personal data. Many large datasets have been developed to meet these requirements, and they often include sensitive personal information like financial data and medical history, possibly accompanied with corresponding images or videos. Therefore, ensuring privacy and data protection is crucial when working with AI models to prevent data misuse and protect each individual's rights.

In this context the European Union's (EU) General Data Protection Regulation (GDPR) has emerged as a key legal framework defining strict limitations on the collection and processing of personal data. Similar laws and regulations also apply in many countries worldwide.

Recent studies, reveal that a trained machine learning model may expose private, personal details about the individuals whose data was used for its training, even if the actual training dataset is not directly accessible. This exact vulnerability underscores the significance of safeguarding not just data, but trained models themselves.

Some fundamental principles, outlined by GDPR and should be considered while implementing a machine/deep learning model:

- **Lawfulness and Transparrency:** Data processing behind each machine learning algorithm, should follow a legal basis on personal data and must fully inform individuals about how the data will be used and for which purpose.
- **Consent and Accountability:** Every individual who have given consent regarding their personal data, should as well have the right to withdraw their consent at any time. At the same time, organizations must take responsibility for compliance with GDPR principles.
- **Data Minimization:** Only the required amount of data, based on the purpose of each specific AI task, should be collected and processed.
- **Data Anonymization:** Anonymization is an essential technique, which enhance the model's privacy by ensuring that, once personal information has been anonymized, the identity of an individual included in the training set cannot be re-identified.
- **Accuracy and storage limitation:** Every personal data must be updated and maintained accurate, but not retained longer than its compeletely necessary.

The following two subsections provide an more detailed analysis of the data anonymization [24] and data minimization [22] pipelines integrated in this study.

2.3.1 Data Anonymization

As aforementioned, machine learning (ML) models can inadvertently expose personal information, so anonymized data, that comply with EU General Data Protection Regulation (GDPR) and California Consumer Protection Act (CCPA) regulations, provide security against these attacks. In order to enhance the level of safety, [24] propose a complete anonymization process to create an anonymized model itself (as shown in Figure 2.10).

However, learning on anonymized data often results in a loss of model accuracy. The method utilized in this study, improves model accuracy by leveraging knowledge from the original trained model and guides the anonymization process to minimize its impact on performance. This accuracy-guided anonymization method, outperforms traditional k-anonymity methods.

K-anonymity proposed by [2], is a baseline privacy-preserving technique designed to protect individuals' identities in datasets. It involves generalizing or removing attributes until each record is indistinguishable from at least $k-1$ others, reducing the risk of re-identification when the data is linked with external sources. The approach targets quasi-identifiers (QI), which could lead to re-identification if combined with other data sources. While k-anonymity minimizes identity disclosure to $\frac{1}{k}$, it may not fully prevent attribute disclosure if the records within a group share similar sensitive information.

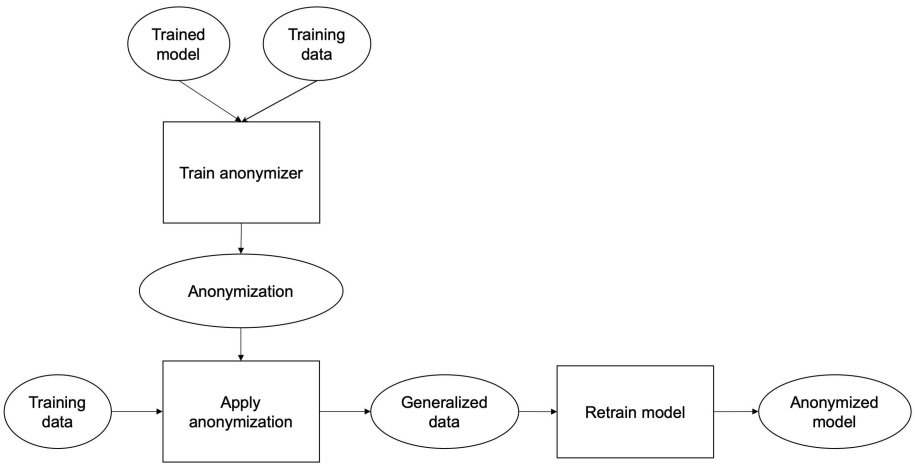


Figure 2.10: Anonymization process.

Inspired by k-anonymity, the method outlined in Figure 2.10, begins by using the model trained on the raw data, which accuracy should be preserved, and run the evaluation process on the training data itself. These outputs are considered as labels for training the "anonymizer" model. The anonymizer, a decision tree model with k minimum samples per leaf node, is designed to

learn the original model's "decision boundaries". Each leaf in the resulting mapping contains a group of records generalized to the same representative value.

Subsequently, the raw training data pass through the trained anonymizer, in order to get the final mapping for the data points in each leaf to a representative value. The chosen approach uses an actual data point from the cluster that falls closest to the median, based on the majority label within the cluster. This preserves higher prediction accuracy, while satisfying the k-anonymity requirement. Finally, the model is re-trained using the generalized data, resulting in the desired anonymized model.

2.3.2 Data Minimization

As mentioned earlier in this subsection, one of the key principles of the GDPR is data minimization, meaning that data collection and processing should be limited to only what is necessary for the task at hand. This is particularly challenging in complex machine learning (ML) and deep learning (DL) models, since many of them are considered "black-boxes".

A study proposed by [22] introduces a novel approach to determine the minimum amount of personal data required for developing and training a ML/DL model, by either removing or generalizing certain input features. By leveraging the inherent knowledge of the pretrained model, the proposed method produces generalizations without significantly affecting the initial accuracy.

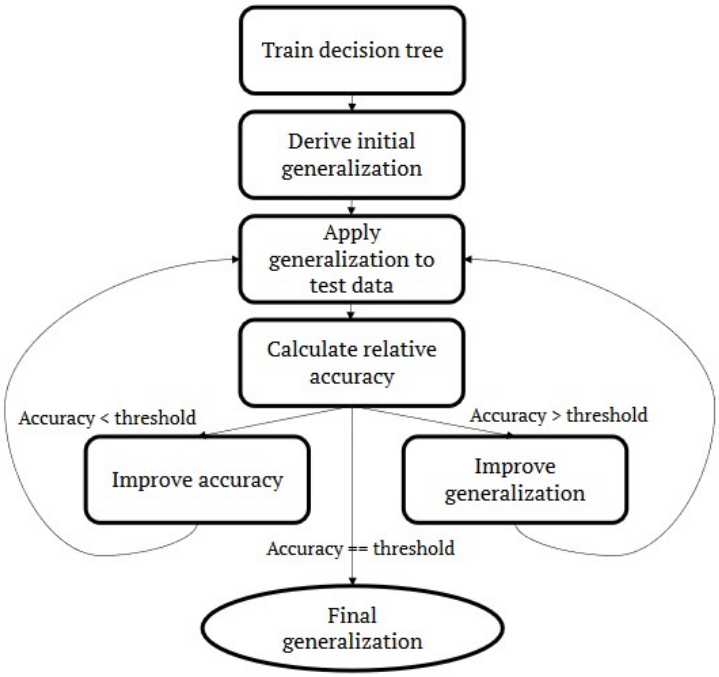


Figure 2.11: Minimization process.

The minimization process (as shown in Figure 2.11) begins with an already trained ML/DL model paired with a dataset of samples along with the model's outputs for them, which serve as their corresponding labels. The minimization strategy, similarly to the anonymization process, uses the model's predictions to cluster similar records, guiding the generalization process. This approach aligns closely with the model's "decision boundaries", allowing for a targeted generalization that maintains accuracy, while achieving data minimization. The desired accuracy is a hyperparameter and could either match the original model's accuracy, meaning no degradation is permitted, or it could be set as a deviation percentage from the initial accuracy.

This specific approach is particularly adaptable for existing models, since the original model remains unchanged and does not require retraining. The outcome of the process is the transformation of input features, some of which may be removed entirely, while others may be generalized.

Firstly, a univariate decision tree is trained and this will be the "generalizer model". The decision tree is created with leaves containing inputs that yield the same prediction in the original model. This guides the initial set of generalizations by merging feature split values from the tree's internal nodes. After applying these generalizations to the test data, the model's relative accuracy, representing the proportion of original predictions retained, determines whether to continue refining the process based on the threshold set:

- In case that the achieved accuracy is greater than the threshold, then the subsequent step is to "improve the generalization", by iteratively merging lower level ranges into a single range. This is applied from lower to higher nodes of the tree, reducing feature splits, until either the root node or the desired accuracy threshold is met.
- In case that the achieved accuracy is less than the threshold, then the subsequent step is to "improve the accuracy". In this scenario, specific features are removed from generalization and will be left unchanged.

Ultimately, the process results in a minimal dataset tailored to the model's required accuracy. This includes specific generalized feature ranges and representatives, also valid for future data collection.

CHAPTER 3

Related Work

3.1 Non-negative networks

An undisputed issue of neural networks, despite their powerful capabilities, is their lack of interpretability and for that reason they have been characterized as “black-boxes” in terms of how and why each specific result has been emerged. Explainability is also a crucial matter, especially in the development of health-related deep learning models for tasks such as image classification, object detection and risk assessment from tabular clinical data. “Non-negative” neural networks have emerged as a promising solution to this challenge. In this context, recent works like [25] and [28] have utilized the properties of “non-negative” networks, demonstrating their effectiveness on exploring various combinations of potential causes on health outcomes, based on clinical data, or increase the interpretability on medical image analysis respectively.

In 2022, [25] proposed the “Causes of Outcome Learning” (CoOL) approach, aiming to explore and identify combinations of exposures that increase the risk of a specific health outcome, since most epidemiological studies focus solely on a single exposure. The CoOL roadmap, illustrated in Figure 3.1, is divided in three different steps, the pre-computational phase, the computational phase and the post-computational phase. During pre-computational phase a causal model, using a directed acyclic graph (DAG) [4], is deployed, in order to identify the final set of exposures that will potentially be included in the analysis. Computational phase, consists of the non-negative model initialization and training, the decompose of risk contributions and the subsequent clustering of individuals based on the computation of Manhattan distances and Ward’s method [12] of the latter. Finally, the post-computational phase defines the final hypotheses and validate the model’s results.

The key aspects and properties regarding the overall non-negative network architecture and training pipeline deployed during computational phase:

- **Model Simplicity and Interpretability:** The non-negative network proposed in this work (Figure 3.1e) consists of a single hidden layer, resembling a linear regression model, but

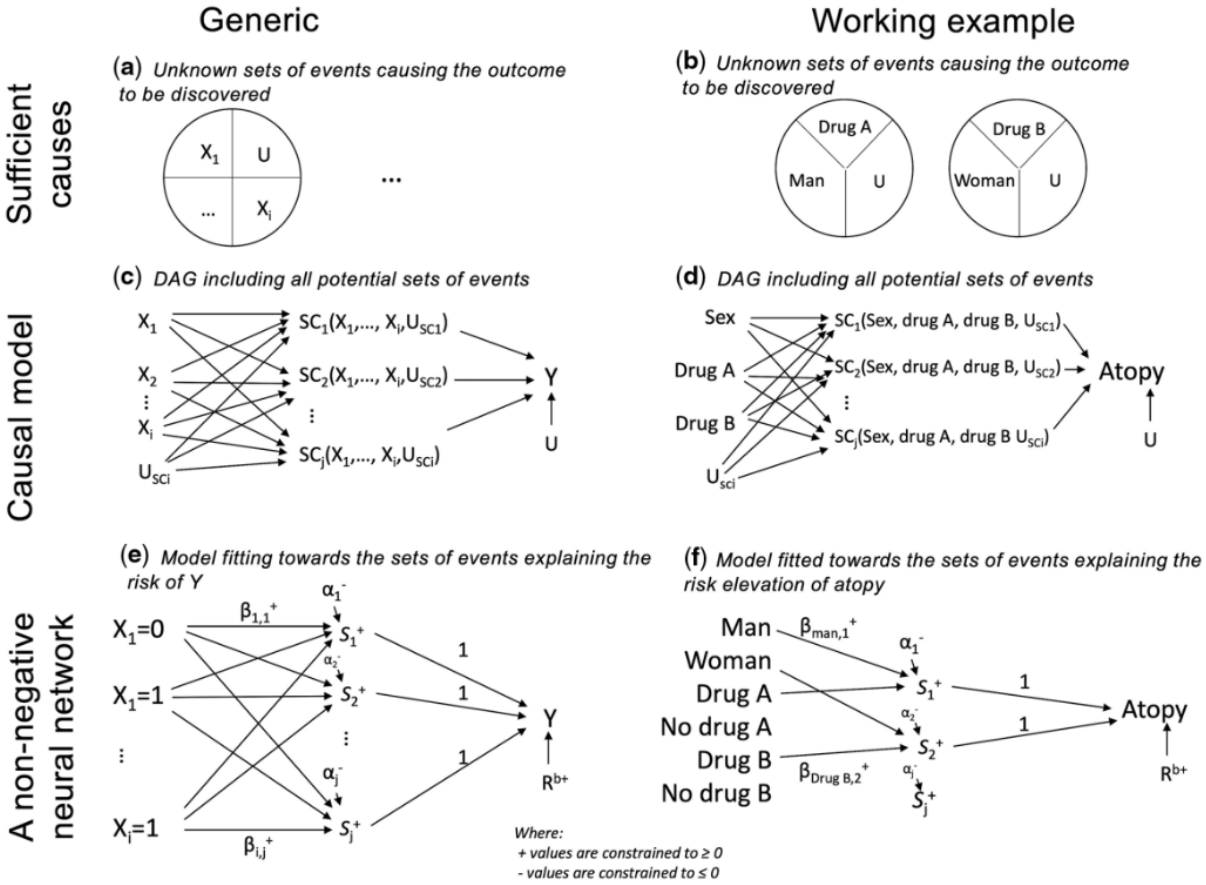


Figure 3.1: Non-negative network as proposed in CoOL.

since it follows a fully connected architecture, each node is able to interact with combinations of various exposures. Activation functions (S^+) apply a non-linearity to the linearly transformed inputs of each node, to eventually be interpreted as the final risk score on an additive scale. Diving into more technical details, Mean Square Error (MSE) is employed as the loss function, while Stochastic Gradient Descent (SGD) is utilized as the optimization algorithm to minimize this error. Both weights and biases were initialized using values drawn from a gamma distribution.

- **Input Data Types:** This network receives as inputs continuous, binary and one-hot encoded data with 1 representing the existence and 0 the absence of each variable.

- **Restricting Weights to Non-Negative Values:** During training all the learnable weights (connection parameters - β^+) are constrained to non-negative values (≥ 0), in order to ensure that the existence of an exposure can only increase the risk of the final outcome.

- **Restricting Biases to Negative Values:** During training, in contrast with the weights, biases (intercepts - α^-) are constrained to negative values, acting as a threshold, that only allows large weights to pass the activation function and subsequently affect (positively) the

final outcome. If a person ends up having no risk contribution on any of the exposures, meaning that the outputs across every node was 0, then it is assumed that this person has a risk equal to a predefined baseline risk (R^{b+})

Mathematically, the network's architecture can be described by the following equation:

$$P(Y=1|X) = \sum_j \left(S^+ \left(\sum_i \left(X_i \cdot \beta^+_{i,j} \right) + \alpha^-_j \right) \right) \tag{3.1}$$

where i, j the exposure and node indices respectively, X_i the i-th exposure value (either numerical or 0 for absence and 1 for existence), $\beta^+_{i,j}$ the positive weight that connects the i-th exposure with j-th node, α^-_j the negative bias of the j-th node and S^+ the non-linear activation function.

An even more recent work of 2024 [28] propose the utilization of non-negative neural networks in medical image analysis (classification and segmentation), in order to increase their explainability and interpretability. Briefly, the proposed model (as shown in Figure 3.2) consists of an Encoder E which outputs features f with the same spatial dimension as the input image X. Sequentially these features pass through a monotonic network M, with constrained non-negative weights, to finally acquire the output logits (y) of the binary classification task. Due to the monotonic property there exists a positive value α, that classifies the monotonic outputs $M(f-\alpha)$ as healthy, leading also to a segmentation result.

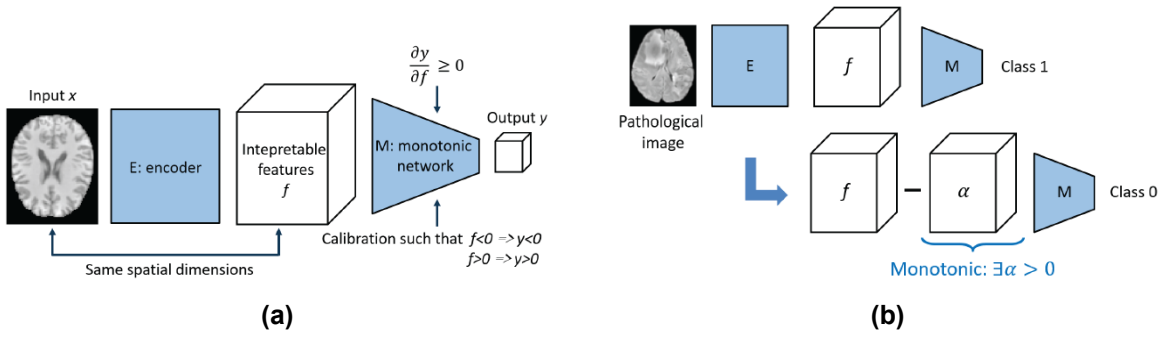


Figure 3.2: **(a)** Non-negative network architecture as proposed by [28] **(b)** Result interpretation strategy for classification and segmentation tasks.

3.2 Risk Assessment models on Healthcare

Through the years, numerous risk assessment models have been developed for binary classification tasks in healthcare community, subsequently providing risk scores for health condition diagnoses, based on personal clinical data. Logistic regression is a widely used method for

tasks such as primary melanoma classification, as demonstrated by [3] and [21]. Australia's Melanoma Institute provide a tool for first primary melanoma risk calculation[1] following the model proposed in [21]. While many studies rely on multiple individual factors, the method proposed in this study focuses on combinations of both clinically and self-assessed risk factors to develop a model for first primary melanoma prediction. The method employed for this purpose was an unconditional logistic regression model, trained on the Australian Melanoma Family Study and further validated by Leeds Melanoma Case-Control Study.

Another study [17], aims to develop and compare machine learning models for predicting type 2 diabetes risk and identifying associated risk factors using data from the 2014 Behavioral Risk Factor Surveillance System (BRFSS) [29]. Specifically, it employs various machine learning classifiers like Decision Trees, Polynomial/Rbf/Linear SVM, Naive Bayes, Random Forest and Logistic Regression, as well as a neural network model. After several preprocessing steps, the dataset comprised 138.146 participants, 20.467 of whom were diagnosed with type 2 diabetes. Regarding the predictive accuracy, the neural network model, as expected, showcased the best results, suggesting that deep learning models can potentially develop critical decision-making abilities in medical tasks, to enhance prevention on various healthcare chronic conditions. Similarly [14] employs a single hidden layer MLP network for cardiovascular disease risk prediction, although stating its weak interpretability.

Handling imbalanced datasets is a critical challenge in machine learning, particularly in the context of clinical data. Several methods have been developed to address this issue, like SMOTE [1], a widely used technique for oversampling the minority class with new synthetic examples. More specifically, SMOTE identifies examples that are close to each other in the feature space, draws a line between them and then selects a new sample at a point along that line. Typically, before oversampling the minority class, a common practice is to undersample the majority class to an extent and then apply SMOTE. A downside of this method is that generated entry points, does not take into account the majority class distribution, resulting in many ambiguous and overlapping samples among different classes. A more recent work, Conditional Tabular GAN (CTGAN) [18] propose a generative adversarial network, which is designed specifically for tabular data generation. It effectively handles class imbalance using a conditional generation mechanism and an innovative training-by-sampling method. CTGAN models tabular data distribution and samples rows from it, ensuring that every category, even the minority ones, are well represented in the synthetic generated samples. This conditional approach, combined with the mode-specific normalization, in order to handle numerical, categorical data and complex distributions helps to balance class distribution during both training and generation phases. As a result, CTGAN can produce high-quality synthetic data that accurately reflects the true dis-

[1] First Primary Melanoma Risk Calculator available here

tribution of the original dataset, including the minority classes.

3.3 Tabular Data Handling

In recent years, many works shifted towards leveraging Transformer architectures, originally designed for Natural Language Processing tasks, to handle tabular data. This shift is motivated through the self-attention mechanism, which provides the Transformer model with the ability to capture complex feature interactions and dependencies. Firstly, Tab-Transformer [20] developed a framework that is composed of several key components as shown in Figure 3.3:

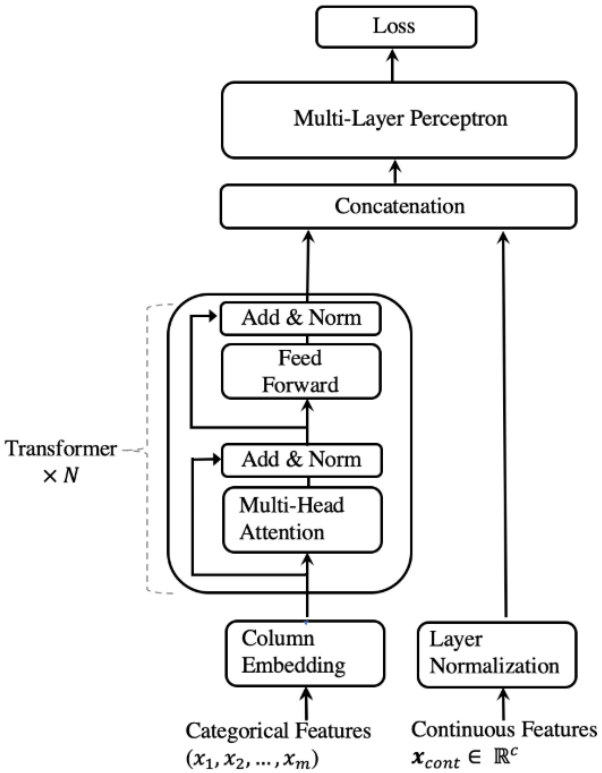


Figure 3.3: Tab-Transformer architecture.

- Column Embedding layer: Each categorical feature is transformed into a set of learnable embedding vectors.
- Transformer Encoder layer: The column embedding layer's outputs are then passed into a Transformer Encoder layer, applying multi-head self-attention and allowing each feature attend to itself and every other input feature.
- MLP Head: Finally, continuous features, are combined with the contextual embeddings from the Transformer output and subsequently passed through a multi-layer perceptron,

in order to perform classification.

FT-Transformer [23] improves on Tab-Transformer, as they propose a Feature Tokenizer, converting both categorical and numerical features into learnable embeddings. This ensures a uniform representation that the Transformer layers can process effectively. Categorical features are typically mapped to distinct embedding vectors, while numerical features are scaled and then embedded. However there are limitation on these methods such as the missing values handling. In such cases, Tab-Transformer uses the average of the learned embeddings of all classes within the corresponding column with the missing entries. This means that during training, the model calculates the average embedding for each feature, and this average is used to fill in the gaps where data is missing during prediction. This method, although, could lead to misleading results, especially in healthcare domain, where available datasets usually contain relatively small amount and sparse entry points.

CHAPTER 4

Proposed Framework

4.1 Non-negative Fully Connected Network

Inspired by [25], we propose a non-negative neural network, designed for risk assessment on healthcare clinical data. Our proposed architecture, illustrated in Figure 4.1, features a 3-layer MLP that consists of an input layer, two hidden layers and one output layer, all fully connected. This design ensures that, during training, the weights are constrained to be non-negative, enhancing the interpretability and robustness in the domain of healthcare data analysis. In this context, exposures could either contribute positively to the risk outcome (when $w > 0$) or have no effect (when $w = 0$). Following the baseline, the biases of the two hidden layers are also constrained to be negative, allowing only sufficiently large weights to pass through the activation function, thereby positively affecting the final outcome.

The non-negativity constraint on network's weights, lead to a non-linear, though non-decreasing relationship between the input features and the predicted outputs. This property can be beneficial in many clinical applications where certain continuous risk factors are known to have a monotonic relationship with the outcome. A monotonic relationship is defined between two variables where one of them either consistently increases or consistently decreases as the other variable changes in always the same direction without switching. Potential features like age and genomics risk ratios could be usefully maintained as continuous values, because typically they are linearly affect the outcome. However, this monotonic non-decreasing relationship can be problematic for certain features like weight, height and number of naevi. In example, an underweight or an overweight patient may be associated with higher risk compared to normal weight, although this would not be captured by the model due to monotonicity. Therefore, it may be more appropriate for these kind of features to use bins, assigning them into distinct categories, instead of treating them as continuous variables, in order to allow the model capture their relationship with the outcome more effectively.

Our contributions and improvements over the baseline non-negative network proposed in [25],

are as follows:

- **Increased Depth and Complexity:** We utilize two hidden layers instead of one, with an increased number of nodes per layer compared to the baseline. This grants the model the ability to explore deeper and more complex relationships within the input clinical data.
- **Unconstrained Output Bias:** The bias term b^3 in the output layer is left unconstrained, providing additional flexibility in the model. Given the non-negativity of network's weights and the utilization of Sigmoid as the output layer's activation function, the unconstrained bias provides a baseline risk calculation. Considering the case where all input features are zero, the baseline risk is computed by the sigmoid of last bias. In this case if b^3 equals zero the baseline risk is 0.5, while if b^3 is either positive or negative, the baseline risk will be greater or less than 0.5, respectively.
- **Learnable Output Weights:** The weight parameters connecting the second hidden layer and the output layer $W^{3(+)}$, are now learnable. Consequently, the output logit vector Y is characterized as a weighted sum of the second hidden layer's activation outputs, rather than simply aggregating them.
- **Advanced Optimization Techniques:** We employ focal loss and the RMSProp optimizer, in contrast to the Mean Square Error (MSE) loss and SGD optimizer used in the baseline model. More technical details, regarding the training pipeline, are provided in Section 5.2.

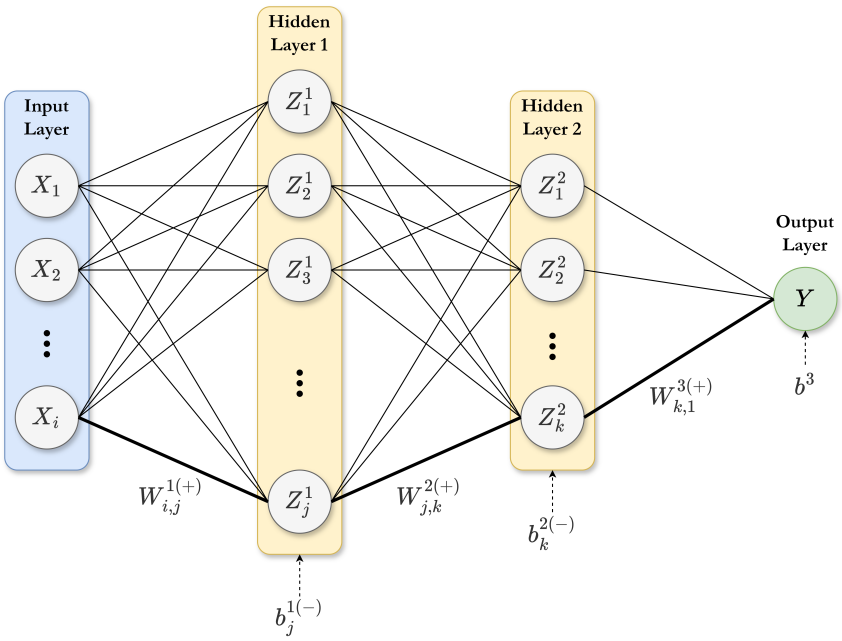


Figure 4.1: Non-negative fully connected network (MLP) architecture.

Equations 4.1 - 4.3 describe the mathematical operations behind the propose architecture. It is worth mentioning that $ReLU$ is employed as the non-linear activation function on both hidden

layers. Additionally, the output logits Y, can be interpreted as probabilities of the positive class outcome, with values in range $[0, 1]$, through the application of the $Sigmoid$ activation function.

$$Z_j^1 = ReLU\left(\sum_i \left(W_{i,j}^{1(+)} \cdot X_i\right) + b_j^{1(-)}\right) \tag{4.1}$$

$$Z_k^2 = ReLU\left(\sum_j \left(W_{j,k}^{2(+)} \cdot Z_j^1\right) + b_k^{2(-)}\right) \tag{4.2}$$

$$Y = \sum_k \left(W_{k,1}^{3(+)} \cdot Z_k^2\right) + b^3 \tag{4.3}$$

where i, j, k the indices of the nodes in the input layer, the first hidden layer, and the second hidden layer respectively, $X_{\{1...i\}}$ the input data features, $Z_{\{1...j\}}^1, Z_{\{1...k\}}^2$ the activation outputs of each hidden layer, W, b the learnable weight parameters and biases connecting the layers, and superscripts $^{(+)}$,$^{(-)}$ indicate the value constraints on the corresponding matrices or vectors, if applicable.

4.2 Attentive Network

In this section, we introduce a novel architecture for risk assessment on clinical structured data (Figure 4.2), as we propose a network that leverages the strengths of a Transformer Encoder model. The choice of a Transformer Encoder as the main component, lies on its capability to provide insights about the decision-making process of the model, thereby enhancing the explainability of our method, while maintaining accuracy and robustness. Through attention weights, it becomes straight-forward to identify which features contribute the most, towards a higher risk score and consequently to a diagnosis of a health outcome. This inherent property of the attention mechanism is particularly crucial in healthcare domain, where transparency and justification of each decision are essential.

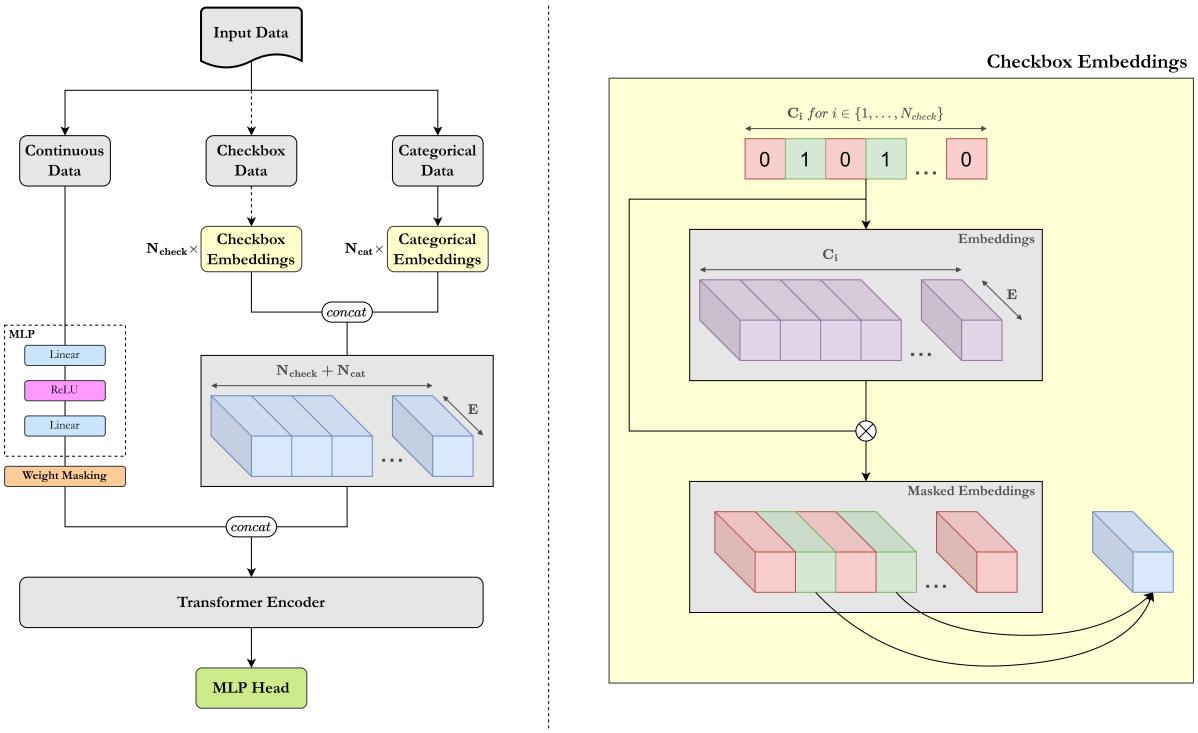


Figure 4.2: Attentive model architecture proposed.

The proposed architecture adeptly handles various data types encountered in clinical datasets, namely: numerical (continuous), categorical and checkbox data. We define a global embedding size E applying for every data type, in order to finally bring all the available features in the same feature space. For **continuous data**, we employ a Multi-Layer Perceptron (MLP), consisting of two linear layers and a ReLU activation function between them, to introduce non-linearity. The dimensions of the MLP output are (B, N_{num}, E) where B is the batch size and N_{num} the number of numerical features available on the dataset. Regarding **categorical data**,

we deploy standard categorical embeddings for each feature, ensuring their discrete nature is maintained, while creating the embedded matrix (B, N_{cat}, E) where N_{cat} represents the number of categorical features available in the dataset. Additionally, we define **checkbox data**, as vectors receiving values 0 or 1, with ones representing the existence of the corresponding category and zeros its absence, but with the possibility of multiple positive categories on a single feature, thereby making it more complex to handle than standard one-hot encoded data. This type of data are processed with a custom embedding layer, where each checkbox feature $C_i \ \ for \ i \in \{1, ..., N_{check}\}$ is represented as a binary vector of size c_i, where c is the total number of possible categories within the feature. In example, if a checkbox feature C_i can have 5 possible categories, the vector might look like $[1, 0, 1, 0, 0]$, indicating the presence of the first and third categories. Each category, treated as a binary feature, passes through a categorical embedding, resulting in a matrix of size (B, c_i, E). Next, an element-wise multiplication is applied between the embedded matrix and the initial binary vector C_i, serving as a mask, zeroing out all the embedded vectors of non-active categories. Subsequently, the active embedded vectors are aggregated to allow all valid categories interact with each other, creating a single vector of size E, representing the combined information within the checkbox feature. This process is repeated N_{check} times, resulting in the final embedding matrix (B, N_{check}, E).

The concatenated embedded matrix of shape $(B, N_{num} + N_{check} + N_{cat}, E)$ integrates all these diverse data representations into a unified feature space, serving as the input to the Transformer Encoder module. This module utilizes the multi-head self-attention mechanism, allowing the model to capture complex relationships and interactions across the entirety of the features. Finally, the MLP head is a linear transformation of the encoded features, responsible for the risk score assessment, which subsequently can be interpreted into a binary classification output by passing it through a Sigmoid function. The proposed model, provides enhanced interpretability, through the attention weights of the Transformer Encoder, which can reveal insights about the features that have the greatest influence on the decision-making process.

Another contribution of our proposed network is that, it effectively handles missing values. For continuous data, we apply an element-wise weight masking on the MLP outputs, masking out the weight vectors that correspond to missing numerical values in the input data. For categorical and checkbox data, we define a padding index (equal to zero), corresponding to the missing values, on the embeddings, effectively ignoring these entries during the embedding process. Thus, for every missing registration, identified by a zero index within the pre-processed dataset, the embedding layer's output will be a "zero vector".

The input data for the attentive network are formatted to allow the model handle various data types efficiently. Continuous data remain numerical as the network process them directly with

the multi-layer perceptron. Regarding categorical data, each feature is represented by a vector, where each category is assigned a distinct integer value, with zero defining missing entries. Therefore, one-hot encoded data need to be transformed in the aforementioned format, in order to appropriately apply the categorical embeddings. For instance, if a feature contains four categories, the vector values would range from *0*, for missing data, to *1*, *2*, *3*, *4*, each representing the corresponding category. Checkbox data can have multiple valid categories simultaneously which makes it unfeasible to merge each feature into a single vector with distinct indices. Instead every possible category of a checkbox feature is represented as a standalone binary feature, indicating either the presence or absence of that category. Subsequently, as described above, these vectors pass through custom embeddings and interact with each other. For instance, if a checkbox feature consists of four possible categories, there would be four separate binary vectors representing the existence of each category.

CHAPTER 5

Experiments and Results

5.1 Dataset Analysis and Preprocessing

5.1.1 Melanoma Clinical Dataset

The first dataset considered in this work is dedicated for melanoma binary classification, through the use of clinical data. It comprises a total of 415 patients, with a ratio of 68.7% who have been diagnosed with melanoma at least once, to 31.3% who have never had melanoma. This ratio is further broken down by sex in Figure 5.1. It is also significant to examine the age distribution of the patients within the dataset, as well as the age stratified by the target label variable (melanoma existence), as illustrated in Figure 5.2. An in-depth statistical analysis on this dataset is available on Appendix A.1.

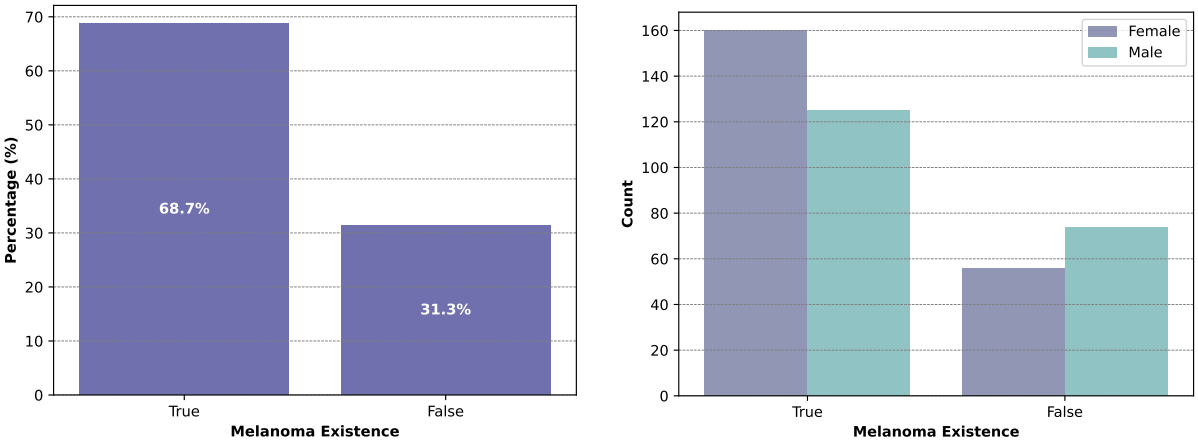


Figure 5.1: Comparison of target label (melanoma existence) distribution across the dataset **(left)** and by sex **(right)**.

To utilize the dataset for both of our proposed architectures, it had to undergo several preprocessing steps, in order to meet each model's requirements, while maintaining its validity with respect to the patients personal privacy. The initial raw data, in *CSV* format, consists of 86 total features (columns). Some of them contain unknown (NaN) values for the majority of pa-

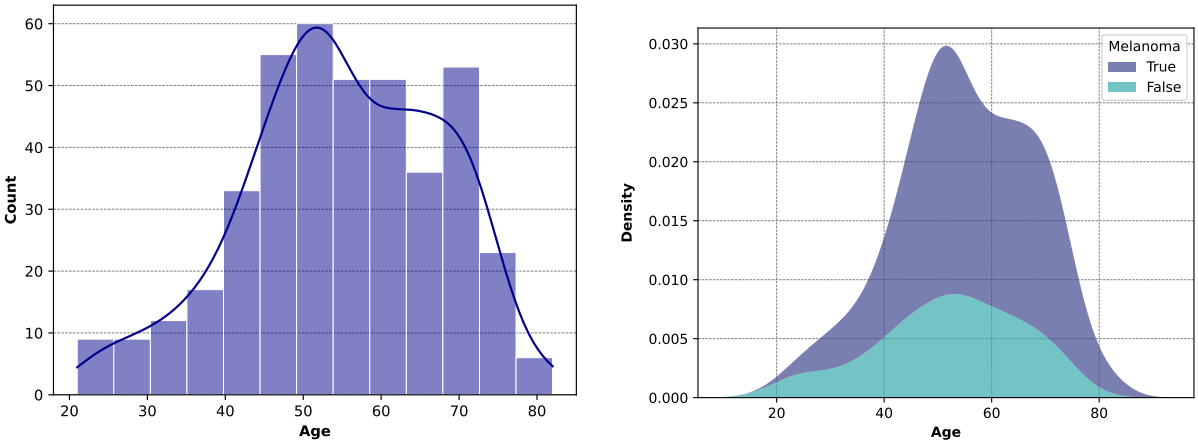


Figure 5.2: Age distribution across the dataset **(left)** and by melanoma existence variable **(right)**.

tients, while others are not relevant for the melanoma classification task. These features are excluded from the final dataset. Additionally, the preprocessing pipeline, analyzed in detail on Figure 5.4, includes an optional step that converts numerical features into categorical ones, with a fixed number of bins (default is 5), mostly used for ablation analysis. Due to high sparsity and privacy measures, the feature indicating each patient's birthplace is removed. As depicted in Figure 5.3, a significant proportion of patients, comprising 53% and 45.6% were residents of Australia and Spain, respectively, during the data collection period. Conversely, only 1.4% resided in other countries, so in order to mitigate data sparsity, these specific instances were excluded too. Finally, rather than keeping each first and second degree relative's history separate, the corresponding features are truncated into two binary columns, on whether the patient has a first/second degree relative ever diagnosed with melanoma, or not.

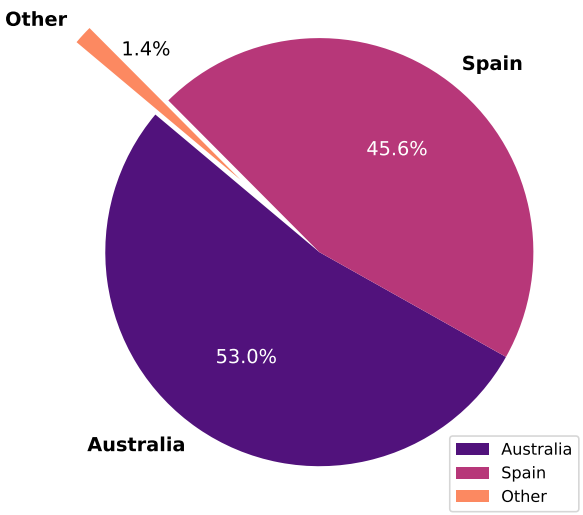


Figure 5.3: Current residency distribution across the raw data (before preprocessing).

Since the task of this study is supervised classification, the target label is derived from the melanoma history feature, which numerically indicates the number of positive melanoma diagnoses each subject has received. This feature is subsequently converted into a binary format: a value of “1” is assigned if the patient has been diagnosed with one or more melanomas, and a value of “0” if the patient has never been diagnosed with melanoma. Similarly, in the case of multi-label classification tasks, a second target label is created by aggregating information across three different numerical features, squamous cell carcinoma (*“scc”*), basal cell carcinoma (*“bcc”*), and other skin cancer history. The complete list of features included in the final dataset is presented in Table 5.1.

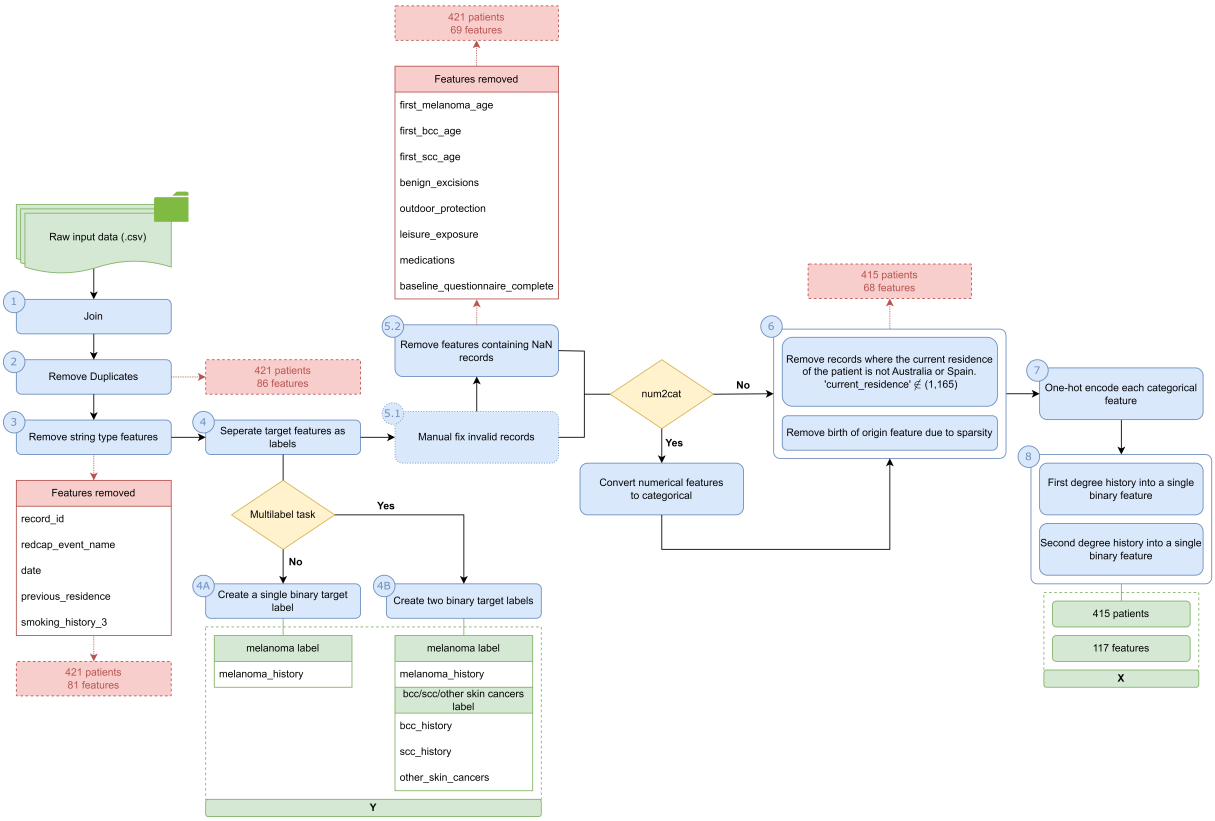


Figure 5.4: Age distribution across the dataset **(left)** and by melanoma existence variable **(right)**.

Regarding the structure of the features considered, depending on the chosen network and the specifications of each experiment, categorical features can either be interpreted as one-hot encoded vectors or remain in their original form, where each unique category is assigned a distinct integer value. During one-hot encoding process, each category is represented by a binary vector, where value “1” indicates the presence of that category, while “0” indicates its absence. This transformation is typically applied on the input data of the non-negative MLP model architecture proposed in this study. Usually, on questionnaire-based datasets, features could be obtained by “checkbox” fields, that allows multiple options of the same feature to be

selected. In contrast with categorical data, where each feature has one valid category, checkbox features could be interpreted like one-hot encoded data, but each vector is allowed to have many positive values ("1s"). These features require special attention, because they cannot be directly converted into categorical data for embedding application. Therefore, they need a distinct methodology tailored specifically to handle them as described in Chapter 4. Finally, as mentioned before, numerical data can either remain unchanged, or converted into categorical values with a selected number of bins (categories).

Feature Name	Description	Data Type
age at baseline	Age at survey time	continuous
years current residence	Years living in current country at survey time	continuous
height	Height in cm	continuous
weight	Weight in kg	continuous
ancestry (1 to 22)	Ancestry	checkbox
who do you usually see for (1 to 5)	Which doctor do you usually see for skin checks?	checkbox
smoking history	Ever been regular smoker? (smoked daily for at least 6 months)	categorical
smoking history 2	Are you a regular smoker now?	categorical
sex	Sex (as defined at birth)	categorical
current residence	Country of residence at survey time	categorical
marital status	Marital status	categorical
highest qualification	Educational level	categorical
employment	Occupational status at survey time	categorical
occupational exposure	Have your occupations been mainly indoors/outdoors/both?	categorical
clinical skin check	Clinical skin check frequency	categorical
child	How many times were you sunburned badly as a child (under 18 years old)?	categorical
adult	How many times were you sunburned badly as an adult (under 18 years old)?	categorical
sunbed use	How many times have you used sunbeds/tanning beds?	categorical
child sunscreen	Frequency of sunscreen used during summer during childhood (up to 10 years old)	categorical
adolescent sunscreen	Frequency of sunscreen used during summer during adolescence (11-18 years old)	categorical
adult sunscreen	Frequency of sunscreen used during summer during adulthood (over 18 years old)	categorical
hair	Natural hair color at age 21	categorical
eye	Eye color at age 21	categorical
burn	Skin response to sun exposure at noon for 30 minutes without sunscreen/clothing protection	categorical
tan	Does your skin tan after prolonged and repeated sun exposure without sunscreen or clothing?	categorical
freckless	Amount of freckless	categorical
naevi	Naevi in childhood/adolescence (up to 18 years old)	categorical
family history 1st	Have any of your first-degree relatives ever been diagnosed with melanoma?	categorical
family history 2nd	Have any of your second-degree relatives ever been diagnosed with melanoma?	categorical

Table 5.1: Complete list of the 29 features considered within the **melanoma dataset**, after preprocessing/cleaning steps, along with their corresponding descriptions and data types. (some data types may differ depending on the network architecture chosen)

5.1.2 BRFSS 2022 Survey Dataset

The BRFSS - **Behavioral Risk Factor Surveillance System** stands as the foremost platform for conducting health-related telephone surveys, across the United States of America, regarding health-related risk behaviors, cronic conditions and preventive service utilization among U.S. residents on state-level analysis. Originating in 1984 with participation from 15 states, the BRFSS has since expanded its reach to encompass 50 states, the District of Columbia, Guam,

Puerto Rico and the US Virgin Islands. Each year, the BRFSS conducts over 400.000 interviews with adults (≥ 18 years old), solidifying its status as the largest ongoing health survey initiative worldwide.

This study utilizes the most recent, 2022 BRFSS data [29], that includes 445.132 participants with a total of 328 features (columns) in the raw dataset. Many possibilities arise from such a large amount of features, as the dataset can be exploited in various classification scenarios, depending on the target label selection. This work focuses on predicting a risk score with heart attack or heart disease as the binary target feature. Of course many features were not related to that task so the final training dataset consists of 48 selected features as listed in Table 5.2.

Feature Name	Description	Data Type
State	State FIPS Code	categorical
Sex	Sex of Respondent	categorical
GeneralHealth	Personal evalution of General Health	categorical
PhysicalHealthDays	For how many days during the past 30 days was your physical health not good?	continuous
MentalHealthDays	For how many days during the past 30 days was your mental health not good?	continuous
MedicalCost	Was there a time in the past 12 months when you needed to see a doctor but could not because you could not afford it?	categorical
LastCheckupTime	About how long has it been since you last visited a doctor for a routine checkup?	categorical
PhysicalActivities	During the past month did you participate in any physical activities such as running, calisthenics, golf, gardening, or walking for exercise?	categorical
SleepHours	On average, how many hours of sleep do you get in a 24-hour period?	continuous
RemovedTeeth	How many of your permanent teeth have been removed because of tooth decay or gum disease?	categorical
HadStroke	(Ever told) (you had) a stroke.	categorical
HadAsthma	(Ever told) (you had) asthma?	categorical
StillHaveAsthma	Do you still have asthma?	categorical
HadSkinCancer	(Ever told) (you had) skin cancer that is not melanoma?	categorical
HadMelanoma	(Ever told) (you had) melanoma or any other types of cancer?	categorical
HadCOPD	(Ever told) (you had) C.O.P.D. (chronic obstructive pulmonary disease), emphysema or chronic bronchitis?	categorical
HadDepressiveDisorder	(Ever told) (you had) a depressive disorder (including depression, major depression, dysthymia, or minor depression)?	categorical
HadKidneyDisease	Not including kidney stones, bladder infection or incontinence, were you ever told you had kidney disease?	categorical
HadArthritis	(Ever told) (you had) some form of arthritis, rheumatoid arthritis, gout, lupus, or fibromyalgia?	categorical
HadDiabetes	(Ever told) (you had) diabetes?	categorical
Marital	Marital status	categorical
Education	Level of education completed	categorical
Employment	Employment status	categorical
Income	Income categories	categorical
DeafOrHardOfHearing	Are you deaf or do you have serious difficulty hearing?	categorical
BlindOrVisionDifficulty	Are you blind or do you have serious difficulty seeing, even when wearing glasses?	categorical
DifficultyConcentrating	Because of a physical, mental, or emotional condition, do you have serious difficulty concentrating, remembering, or making decisions?	categorical
DifficultyWalking	Do you have serious difficulty walking or climbing stairs?	categorical

Continued on next page

Feature Name	Description	Data Type
DifficultyDressingBathing	Do you have difficulty dressing or bathing?	categorical
DifficultyErrands	Because of a physical, mental, or emotional condition, do you have difficulty doing errands alone such as visiting a doctor´s office or shopping?	categorical
SmokerStatus	Four-level smoker status	categorical
ECigaretteUsage	Four-level e-cigarette usage status	categorical
ChestScan	Have you ever had a CT or CAT scan of your chest area?	categorical
RaceEthnicityCategory	Five-level race/ethnicity category	categorical
AgeCategory	Fourteen-level age category	categorical
HeightInMeters	Reported height in meters	continuous
WeightInKilograms	Reported weight in kilograms	continuous
BMI	Body Mass Index (BMI)	continuous
AlcoholDrinkers	Adults who reported having had at least one drink of alcohol in the past 30 days.	categorical
HIVTesting	Adults who have ever been tested for HIV	categorical
FluVaxLast12	During the past 12 months, have you had either flu vaccine that was sprayed in your nose or flu shot injected into your arm?	categorical
PneumoVaxEver	Have you ever had a pneumonia shot also known as a pneumococcal vaccine?	categorical
TetanusLast10Tdap	Have you received a tetanus shot in the past 10 years? Was this Tdap, the tetanus shot that also has pertussis or whooping cough vaccine?	categorical
HighRiskLastYear	HIV high risk for the past 12 months	categorical
HadCovid	Has a doctor, nurse, or other health professional ever told you that you tested positive for COVID 19?	categorical
CovidSymptoms	Did you have any symptoms lasting 3 months or longer that you did not have prior to having coronavirus or COVID-19?	categorical
PrimaryCovidSymptom	Which was the primary COVID-19 symptom that you experienced?	categorical
HeavyDrinkers	Heavy drinkers (adult men having more than 14 drinks per week and adult women having more than 7 drinks per week)	categorical

Table 5.2: Complete list of the 48 features considered within the **2022 BRFSS dataset**, after preprocessing, along with their corresponding descriptions and data types.

The main challenge behind this dataset is the significant imbalance in the binary target label. The ratio of patients who have been diagnosed with heart attack/disease at least once, to those who have never been diagnosed is approximately 90% to 10%, as shown in Figure 5.5. The target label imbalance pose a very common issue on healthcare datasets, as the negative samples typically outnumber the positive samples, and if not handled correctly, ML/DL models usually become biased towards predicting the majority (negative) class, thereby reducing the model's ability to generalize and be able to correctly identify the minority (positive) class. More details about how this study manages to tackle this problem can be found in the next section, where the training framework is explained.

The right plot on Figure 5.5 depicts that, on both diagnostic results, the amount of males and females are balanced, preventing the models to grow biased towards a specific gender. Figures 5.6, 5.7 illustrate Body Mass Index (BMI) and smoking habits distributions, analyzed in relation to the presence or absence of heart attack/disease. As expected we can observe that 3.5% of the former smokers have been diagnosed with a heart attack or disease, a percentage equal to those who never smoked, despite the fact that the total number of individuals in the latter group

is more than twice that of the first group. A more detailed statistical analysis on this version of the dataset, can be found in Appendix A.2.

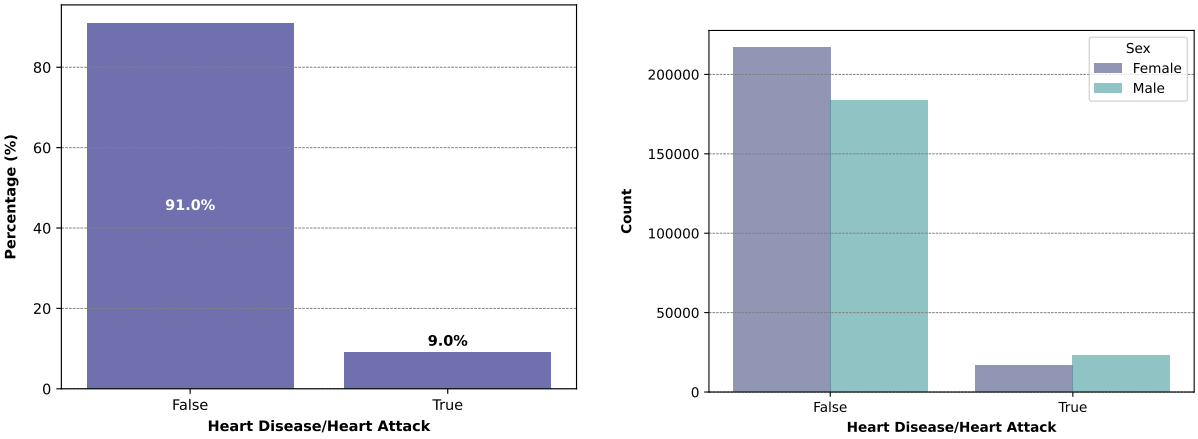


Figure 5.5: Binary target label distribution (Heart disease/attack history) across the dataset **(left)** and sex **(right)**.

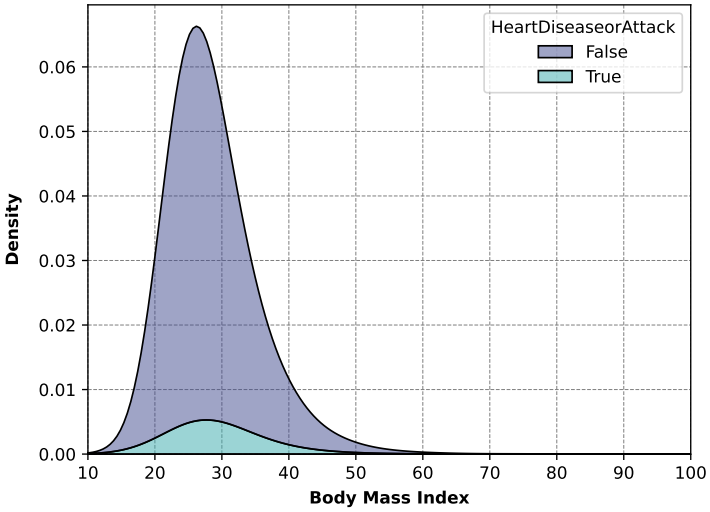


Figure 5.6: Body Mass Index (BMI) distribution in relation to the heart disease/attack diagnosis.

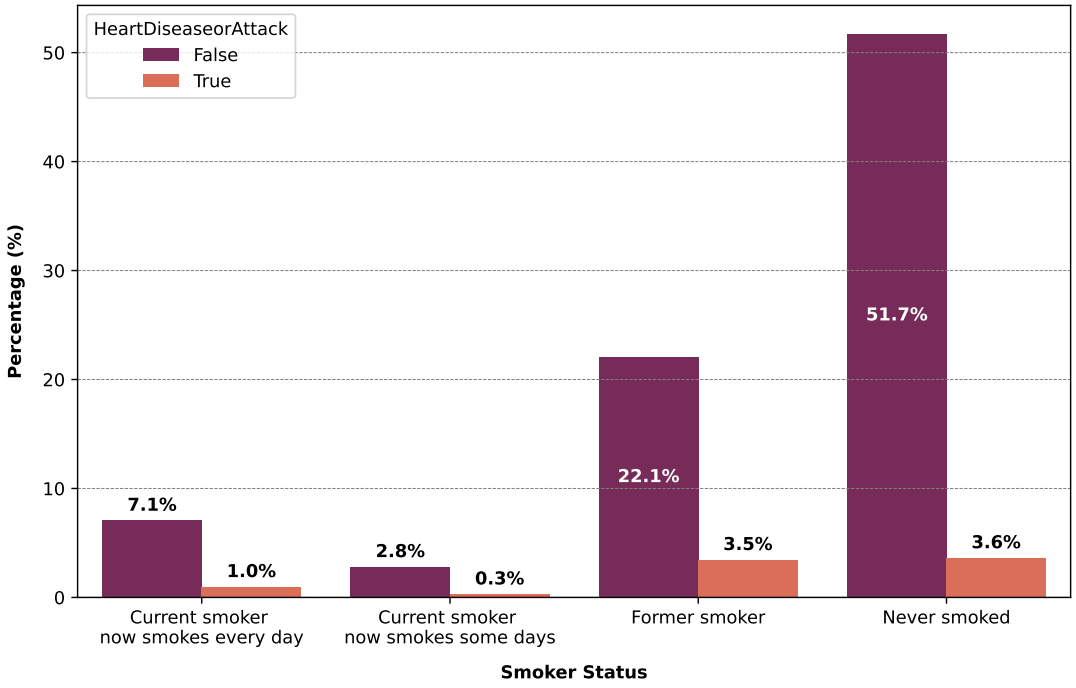


Figure 5.7: Smoking status in relation to the heart disease/attack diagnosis.

5.2 Training Framework and Implementation Details

During training of the non-negative MLP network, we employ Focal loss, which was initially proposed by [11] in 2017. It is typically deployed in object detection tasks, instead of Cross Entropy loss, to handle the class imbalance between the background and foreground objects. However, it can be particularly effective for healthcare clinical datasets, where the positive class (disease presence) is significantly outnumbered by the negative class (disease absence). Additionally, it emphasizes on hard examples, which in our context, are the False Negatives (instances where the model failed to detect a disease). These examples are usually harder to identify, during clinical risk assessment, compared to False Positives (instances where the model incorrectly identified a disease) due to class imbalance. Finally, Focal Loss provides more calibrated probability estimates, meaning that the output risk scores are distributed smoother across the range $[0, 1]$. This is a crucial aspect, in order for the model to produce reliable and confident predictions during decision-making processes.

Mathematically, Focal Loss is defined by Equation 5.1, incorporating the modulating factor $(1 - p_t)^\gamma$ on top of the cross entropy loss criterion. This factor forces the model to instantly focus on hard examples, during training. For $\gamma = 0$ Focal loss is equivalent to Cross Entropy, but for $\gamma > 0$ it raises the confidence and subsequently down-weights the contribution of easy examples. In practice, we deploy α as a weighting factor to balance the contribution from both

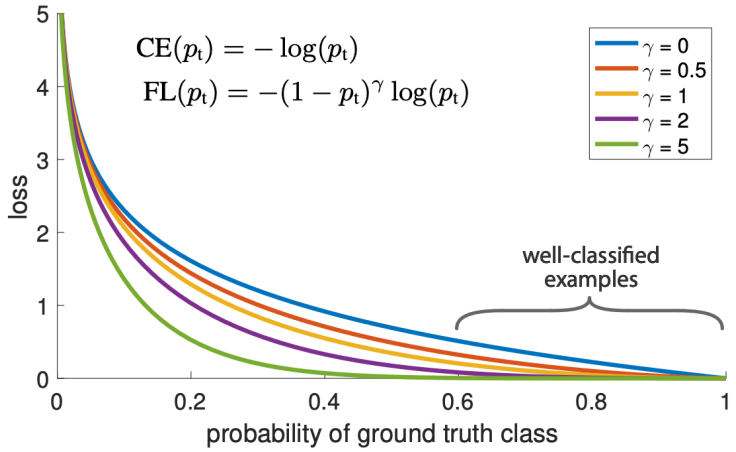


Figure 5.8: Focal Loss behavior for different γ values. By the authors of [11] the default value is $\gamma = 2$.

classes, where $0 < \alpha < 1$. Setting α near 0 increases the influence of the negative class, while setting it near 1 the positive class will contribute more to the final result, despite being the minority. The latter scenario aligns perfectly with our objective on classsifying rarely observed instances.

$$FocalLoss(p_t) = -a_t \cdot (1-p_t)^{\gamma} \cdot log(p_t) \tag{5.1}$$

where p_t represents the probability corresponding to the true class (disease existence).

For the proposed non-negative MLP network, we employ a 5-fold stratified k-fold cross validation approach, to ensure that each split maintains the class distribution of the dataset. Regarding the proposed attentive network, we consider a custom sampling pipeline, to address the significant class imbalance. More specifically, we ensure that, during training, each batch preserves a 90%-10% ratio of negative to positive classes respectively, to mitigate the large contribution of the majority class that leads to overfitting. Each experiment for both architectures, follows an 80%-20% training-validation split on the melanoma dataset and a 90%-10% ratio on the 2022 BRFSS dataset. Additionally, unless specified otherwise, categorical data are transformed into one-hot

Model size	64
Loss function	$Focal\ Loss(\gamma = 2,\ \alpha = 0.5)$
Optimizer	$RMSProp$
Momentum	0.9
Weight decay	$1 \cdot 10^{-3}$
Learning Rate	$2 \cdot 10^{-4}$

Table 5.3: Default parameters for the proposed non-negative MLP model during training.

Model size	128
# Transformer encoder layers	1
# Attention heads	2
Transformer MLP ratio	4
Final representation	GAP
Loss function	$Focal\ Loss(\gamma = 2,\ \alpha = 0.5)$[1]
Optimizer	$Adam$
Learning Rate	$2 \cdot 10^{-4}$

Table 5.4: Default parameters for the proposed attentive model during training.

encoded vectors, to meet the input requirements of the non-negative MLP network. In contrast, for the attentive network, categorical data remain in their original form, with distinct indices assigned on each category and missing values represented by 0 across all features. Finally, during validation and inference steps, we convert the model's output logits to probabilities using the sigmoid function. This transformation yields the probabilities of the positive class for each instance. We then apply a threshold of 0.5 to perform binary classification, determining whether each instance belongs to the positive or negative class. Further technical details regarding the default parameters used during the training of both our models, are summarized in Tables 5.3 and 5.4.

5.3 Results & Metrics

5.3.1 Metrics

In this study, we utilize three key metrics to evaluate the performance of our trained models: accuracy, F1-score and the confusion matrix. *Accuracy* is a straightforward metric, defined as the ratio of correctly predicted examples, both true negatives and true positives, to the total number of instances evaluated:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where *TP* (True Positives) and *TN* (True Negatives) are the instances correctly identified as positives and negatives respectively, while *FP* (False Positives) and *FN* (False Negatives) the instances incorrectly predicted as positives and negatives respectively. *F1-score* is the harmonic mean of the Precision and Recall metrics of the positive class. Precision calculates the

[1] α value might change depending on the dataset and the class imbalance ratio.

model's accuracy on its positive predictions, while Recall measures the model's ability to identify every positive instance. Thus, F1-score combines both of these metrics, taking into account both FP and FN instances, which is particularly useful for clinical risk assessment models. In cases of imbalanced datasets such as the 2022 BRFSS dataset, we deploy the "weighted" variant of the F1-score metric, that performs separate metric calculation for each label and then measures their weighted average. Equation 5.4 is also known in medical tasks as *Sensitivity*. Additionally, we track the model's *Specificity*, by computing the recall of the negative class $(TN/(TN+FP))$.

$$F1score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} = 2 \cdot \left(\frac{Precision \cdot Recall}{Precision + Recall} \right) \tag{5.2}$$

$$Precision = \frac{TP}{TP + FP} \tag{5.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.4}$$

Confusion matrix (Figure 5.9) is a visualization that represents the distribution of TP, FP, TN and FN instances during evaluation. It can provide insights about the performance of our model and helps us understand the challenges and the nature of the errors occurred during training.

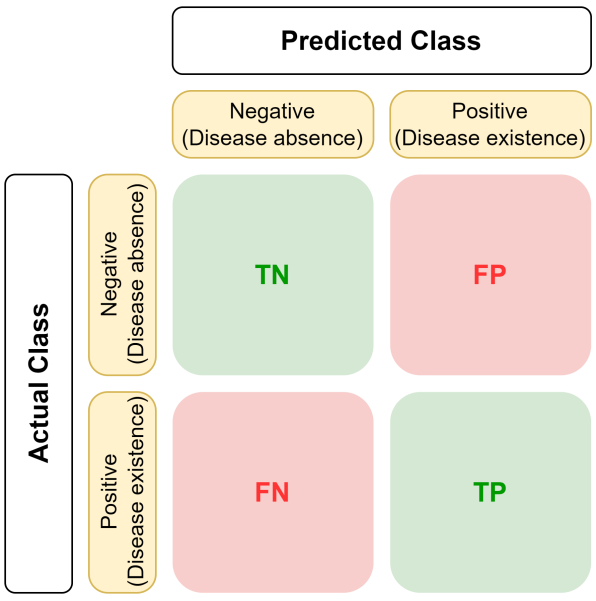


Figure 5.9: Confusion matrix showing the counts of true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN).

5.3.2 Results

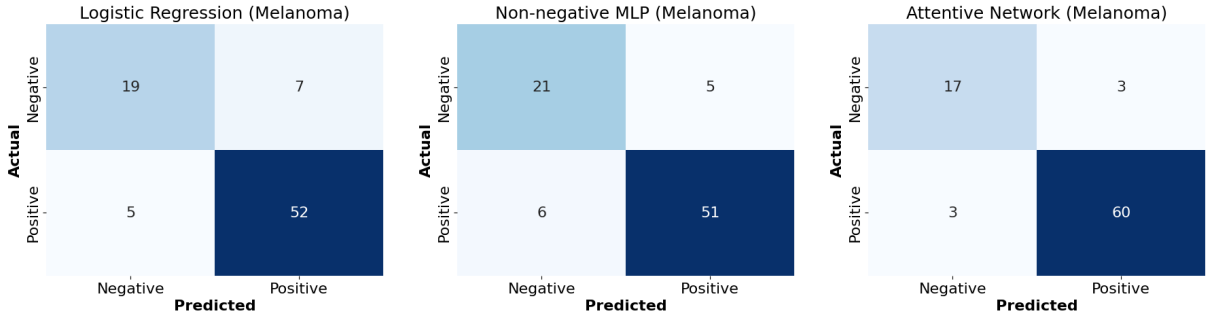
Table 5.5 provides a performance comparison between the three different models deployed for the melanoma classification task, evaluating their effectiveness based on the aforementioned metrics. The models compared include Logistic Regression, which is the most popular and widely used technique for clinical risk score assessment, a Non-negative MLP and an Attentive Network. The latter two frameworks are designed and analyzed in detail in the context of this study. Figure 5.10 further illustrates the performance of these models through their corresponding confusion matrices on the validation set.

More specifically, the Logistic Regression model achieved the lowest accuracy (85.5%) and F1-score (0.897). Its confusion matrix (Figure 5.10a) indicates that while the model performs reasonably well, particularly in detecting positive instances, it shows slightly lower performance in accurately predicting negative cases, with 7 False Positives (cases wrongly classified as melanoma). The non-negative MLP model outperforms the Logistic Regression model, achieving an accuracy of 86.8% and an F1-score of 0.903. The confusion matrix for the Non-negative MLP (Figure 5.10b) suggests that this model is more balanced and slightly better at identifying both positive and negative instances compared to Logistic Regression. Additionally, the Sensitivity and Specificity metrics (Recall of the positive and negative classes, respectively) confirm the balance of the resulting MLP model, in contrast with the Logistic Regression, where the Specificity is significantly lower than the Sensitivity. The Attentive Network model, shows the highest performance among the three, further improving accuracy (92.8%) and F1-score (0.953). Its corresponding confusion matrix (Figure 5.10c) shows 60 true positives, 17 true negatives and only 3 false negatives and 3 false positives instances. This model demonstrates superior capability in both precision and recall metrics across the two classes, indicating a strong performance in correctly identifying between melanoma-diagnosed cases and non-melanoma cases.

Model	Accuracy	F1-score	Precision (pos/neg)	Recall (pos/neg)
Logistic Regression	85.5%	0.897	0.881/0.792	0.912/0.731
Non-negative MLP	86.8%	0.903	0.911/0.778	0.895/0.808
Attentive network	**92.8%**	**0.953**	**0.954/0.852**	**0.954/0.850**

Table 5.5: Performance comparison of different models on the melanoma classification task. Accuracy, F1-score and Precision-Recall for both positive and negative classes are reported for Logistic Regression, Non-negative MLP, and Attentive Network models.

Table 5.6 showcases the resulting metrics after applying the anonymization [24] and minimiza-



(a) **(b)** **(c)**

Figure 5.10: Confusion Matrix for **(a)** Logistic Regression, **(b)** Non-negative MLP Model and **(c)** Attentive Model on Melanoma Classification Task.

tion [22] toolkits on the pretrained Logistic Regression and non-negative MLP models. Both privacy pipelines were applied across three different scenarios, with varying features selected as Quasi Identifiers (QIs):

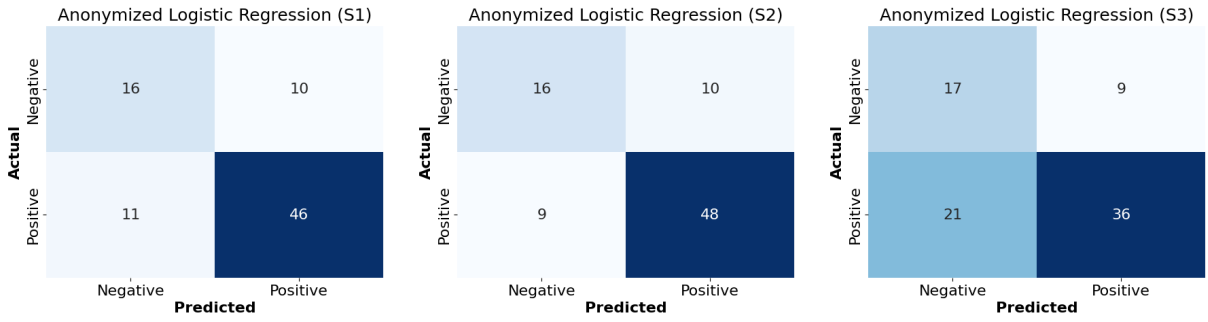
- **Scenario 1:** age, sex, current residence, years current residence, height, weight, marital status, highest qualification, ancestry, employment, smoking history (past), smoking history (now), family histories (first and second degree relationship each summed in a single binary feature)
- **Scenario 2:** age, sex, current residence, years current residence, height, weight, marital status
- **Scenario 3:** all features listed in Table 5.1

During *Scenario 1*, the anonymized Logistic Regression model achieves accuracy of 74.7%, indicating more than 10% of drop rate, for the same validation subset, compared to the non-anonymized model presented in Table 5.5. This considerable drop in accuracy is also depicted in Figure 5.11a with 10 False Positive and 11 False Negative instances. On the contrary, the anonymized non-negative MLP model, achieved significantly higher accuracy of 84.3%, reflecting an approximate 2.5% drop and restricting the False Positives and False Negatives to 7 and 6 respectively, as shown in Figure 5.13a. During minimization, a *minimizer* instance is trained, based on the already anonymized model, identifying the optimal generalizations and subsequently transforming a subset of the validation set accordingly. The anonymized model is then evaluated again on the generalized subset, achieving an accuracy of 82.4% on both Logistic Regression and non-negative MLP networks.

For *Scenario 2*, we utilize a subset of quasi-identifiers from Scenario 1, adopting a more lenient approach. Consequently, the accuracy achieved for the anonymized Logistic Regression model is 77.1% with seemingly improved Specificity indicated by fewer False Negatives as shown in

	Anonymization		Minimization	
Model	**Accuracy**	**Test Samples**	**Accuracy**	**Test Samples**
Scenario 1				
Logistic Regression	74.7%	83	82.4%	17
Non-negative MLP	83.1%	83	76.5%	17
Scenario 2				
Logistic Regression	77.1%	83	76.5%	17
Non-negative MLP	84.3%	83	82.4%	17
Scenario 3				
Logistic Regression	63.9%	83	70.6%	17
Non-negative MLP	78.3%	83	76.5%	17

Table 5.6: Performance comparison of Anonymization/Minimization modules for Logistic Regression and Non-negative MLP models, across three scenarios with varying Quasi-identifiers (QIs).



(a) **(b)** **(c)**

Figure 5.11: Confusion matrices for the Logistic Regression pretrained model, after applying the anonymization module, across three scenarios with different Quasi Identifiers (QIs).

Figure 5.11b. However, after the minimization procedure, the accuracy drops to 76.5%, which is slightly lower than the minimization accuracy in Scenario 1. Regarding the anonymized non-negative MLP network, maintains consistent accuracy levels, identical to those in Scenario 1, demonstrating a more stable network architecture compared to Logistic Regression.

Scenario 3, being the “strictest” possible scenario,posed the greatest challenge for both models, as it considers all input features listed in Table 5.1, as quasi-identifiers. Both anonymized models showed a significant decrease in accuracy, with Figures 5.11c and 5.13c highlighting their considerable misclassification issues. The anonymized Logisitc Regression model, suffers from many False Negative cases (predicted as non-melanoma but actually diagnosed as melanoma), conversely with the anonymized non-negative MLP, which tends to overfit to the positive class, failing to correctly predict most of the negative cases.

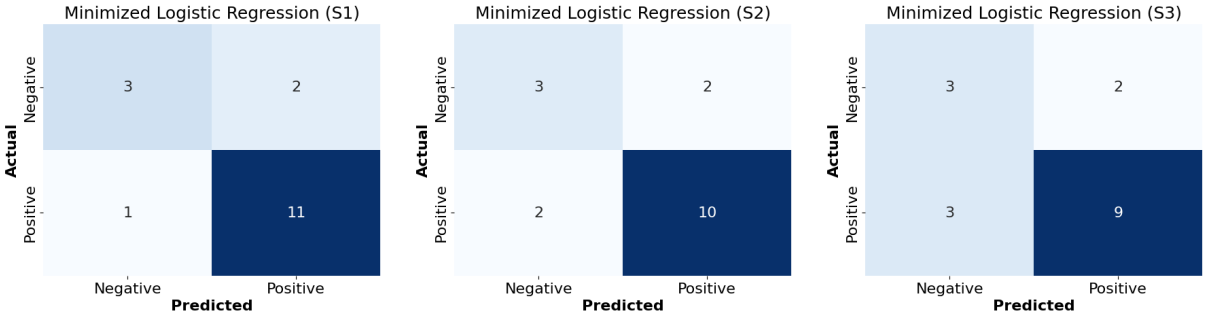


Figure 5.12: Confusion matrices for theLogistic Regression anonymized model, after applying the minimization module, across three scenarios with different Quasi Identifiers (QIs).

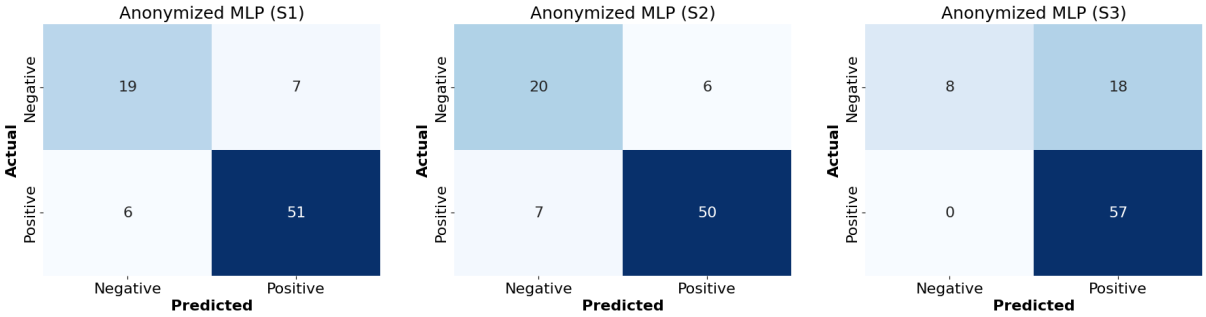


Figure 5.13: Confusion matrices for the non-negative MLP pretrained model, after applying the anonymization module, across three scenarios with different Quasi Identifiers (QIs).

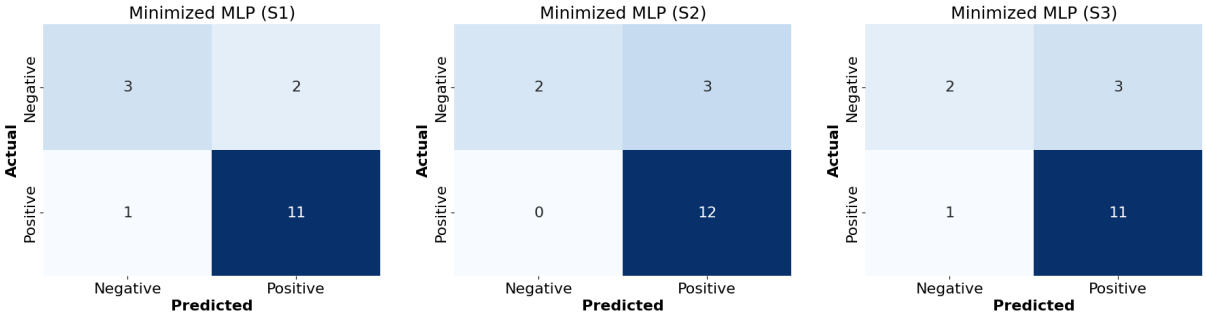


Figure 5.14: Confusion matrices for the non-negative MLP anonymized model, after applying the minimization module, across three scenarios with different Quasi Identifiers (QIs).

Table 5.7, presents the performance metrics for the three models on a binary classification task utilizing a publicly available clinical dataset, described in depth at Section 5.1.2. This task aims to provide a risk score assessment for a heart attack/disease, challenging the robustness of

Model	Accuracy	F1-score	Precision	Recall
Logistic Regression	56.1%	0.228/0.461	0.937/0.533	0.130/0.991
Non-negative MLP	76.9%	0.782/0.769	0.740/0.806	0.830/0.709
Attentive network	**78.5%**	**0.794/0.784**	**0.759/0.815**	**0.834/0.736**

Table 5.7: Performance comparison of different models on the heart disease/attack classification task, with imbalanced target instances. Accuracy, F1-score, Precision and Recall for both positive and negative classes are reported for Logistic Regression, Non-negative MLP, and Attentive Network models.

each architecture, when faced with an imbalanced dataset, which exhibits a 90%-10% ratio of negative over positive target labels. This is a very common and realistic scenario, especially within clinical datasets, where positive instances tend to typically be the vast minority. This imbalance, causes models to overfit on the negative class and evidently appears to happen in the Logistic Regression model, which performs poorly on this task, with accuracy of 56.1% and a Sensitivity (recall of the positive class) of just 0.13. The binary variant of F1-score, that does not take class imbalance into account, achieves a score of 0.23, indicating the model's overfitting issue. These results were highly expected from a standard Logistic Regression model, although we observe that more complex architectures, such as the non-negative MLP and the attentive network, mitigate overfitting and perform significantly better, achieving accuracies of 76.9% and 78.5% respectively. Similarly to the melanoma classification task, the attentive network outperforms the non-negative MLP model, while both of them utilize Focal Loss with $\alpha = 0.9$ to handle the class imbalance effectively.

Since the Attentive network deploys the self-attention mechanism, it offers great insights and interpretability into the decision-making process and results, as visualized by the attention maps in Figures 5.15 - 5.18. Specifically, Figure 5.15 demonstrates a weight distribution heatmap across queries (rows) and keys (columns). Each cell represents how much focus (attention) to place on each key feature when processing a single query feature, revealing underlying relationships between pairs of features. Diagonal elements, typically represent self-attention and high diagonal values indicate that the model is giving significant importance to individual features. In this case we observe that crucial features, such as the type of doctor a patient usually visits for skin checks, the clinical skin check frequency and the skin's response to sun exposure without sunscreen have high attention weights across the majority of the queries, meaning that they are significantly accounted for during the decision making process. Some of the most important relationships identified include:

- The *type of doctor a patient visits* with respect to the participant's current smoking status, place of residency, marital status, ancestry, use of sunbed and family history of melanoma

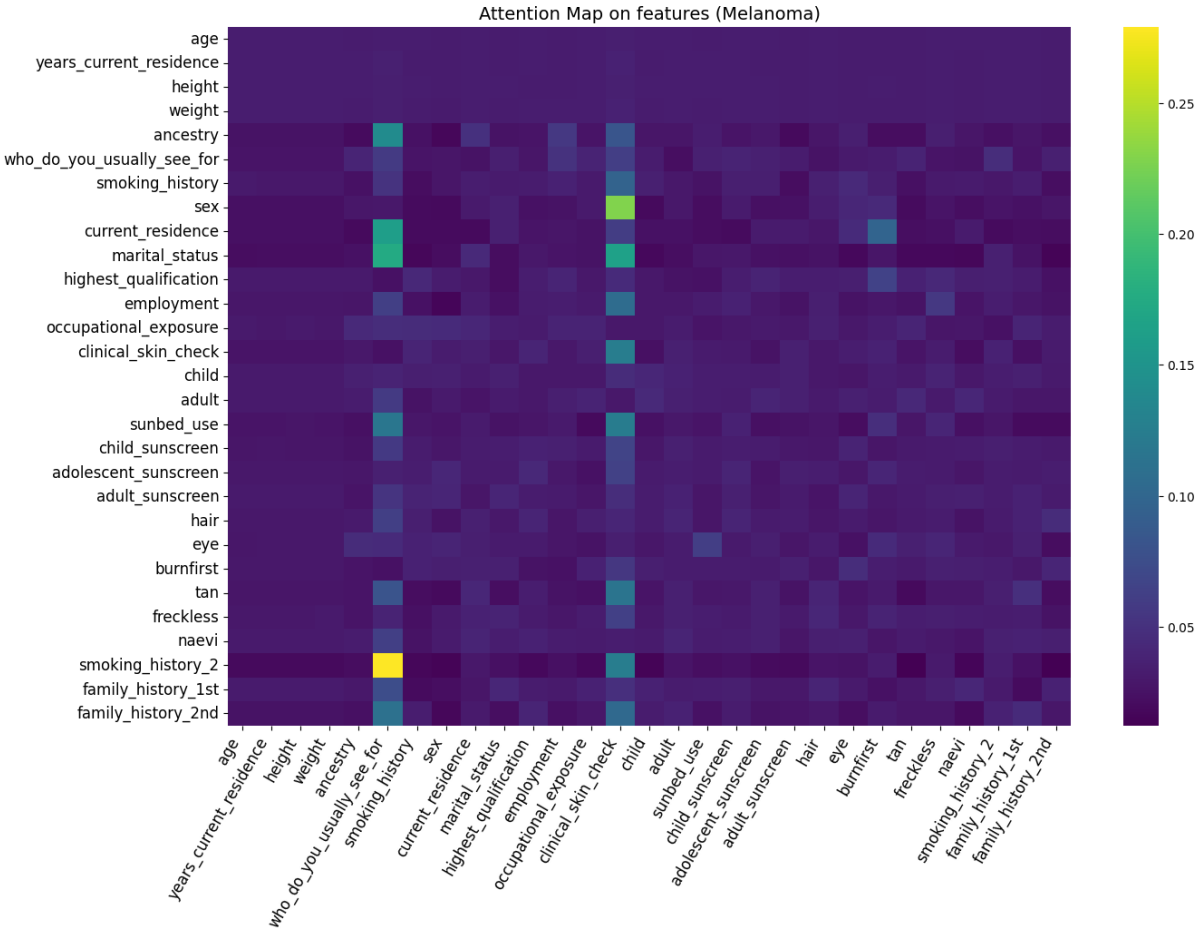


Figure 5.15: Attention map visualization generated by the attentive network for the melanoma dataset, depicting the relationship between queries (rows) and keys (columns).

diagnosis (either first or second degree).

- The *clinical skin check frequency* with respect to sex, marital status, current smoking status, skin tanning after sun exposure without sunscreen and use of sunbed.
- The *skin's response to sun exposure* with respect to the place of current residency.

Similarly, Figure 5.16 illustrates the attention scores that emerged during the heart attack/disease classification task. Some significant relationships underlined by high attention weights include:

- The existence of *kidney disease* (either currently or in the past), in relation to sex, appears to be one of the most indicative relationships, exhibiting a very high attention score. Additionally, its association with the individual's self-evaluation of general health seems significant as well, with a slightly lower attention score.
- Another crucial relationship, is between *stroke occurence* and the existence of kidney disease. If a participant has experienced at least one stroke, it strongly influences most of the features during the decision-making process.

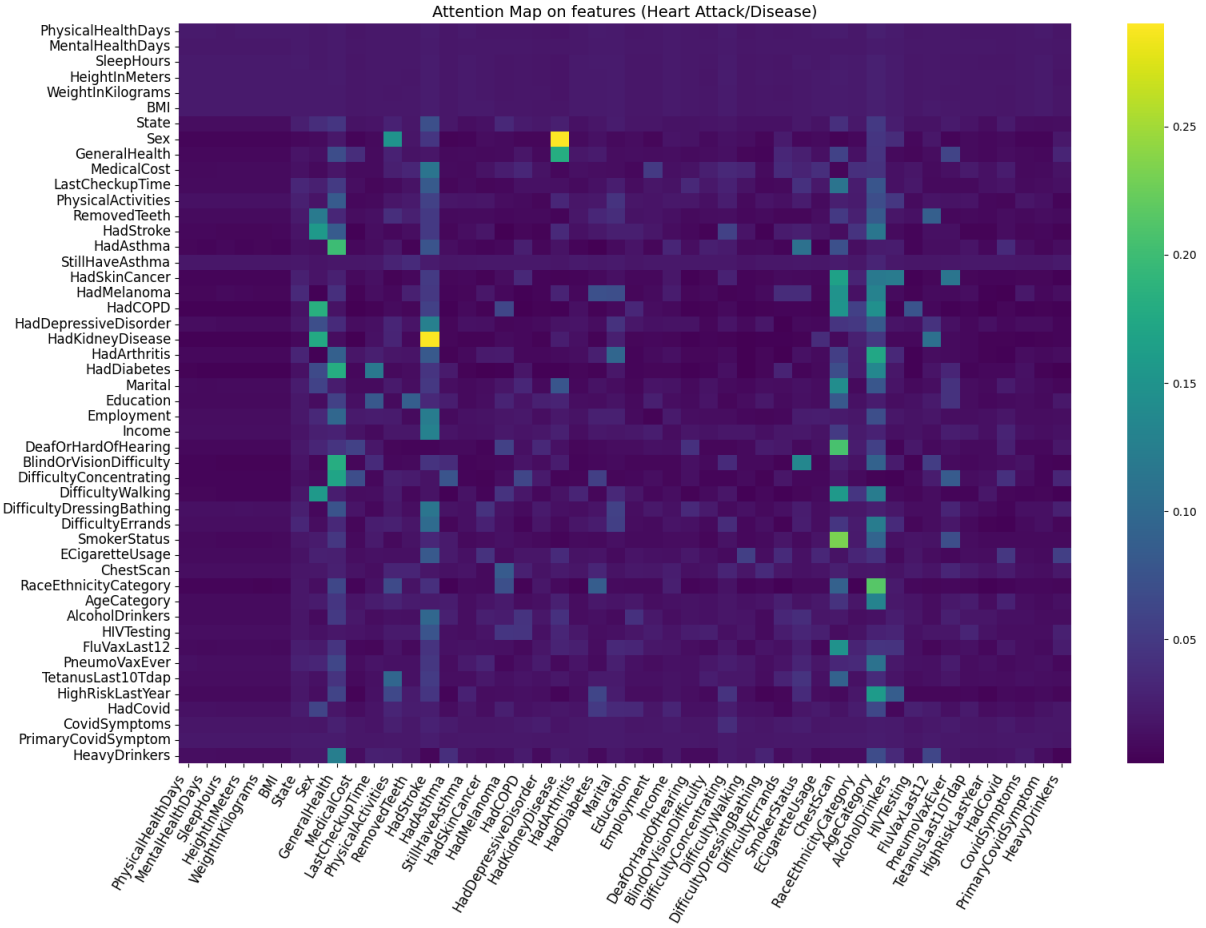


Figure 5.16: Attention map visualization generated by the attentive network for the BRFSS 2022 dataset, depicting the relationship between queries (rows) and keys (columns).

- Similarly features such as *general health self-evaluation*, *sex*, *chest scan examination* and *age* provide high attention weights across the majority of input features, with age/race and chest scan/smoking status being particularly considerable.
- On the contrary, features like *state*, *Covid diagnosis and its symptoms* and *skin cancer*, consistently provide low attention weights across all input features.

Figure 5.17 represents the True Positive instances from the validation set, during the melanoma classification task. This figure includes only participants who have been positively diagnosed with melanoma and assessed with a high risk score. Its aim is to inspect the features that predominantly influence the final positive diagnosis. More specifically, rows represent the participants mentioned above and columns the key features considered during training. Each cell is calculated by averaging attention weight of each key feature across all input queries. The features *"who do you usually see for"* and *"clinical skin check"* consistently provide the highest attention weights across almost every participant diagnosed correctly with melanoma. The frequency of skin checks and the type of doctor visited (e.g. a dermatologist or a general practi-

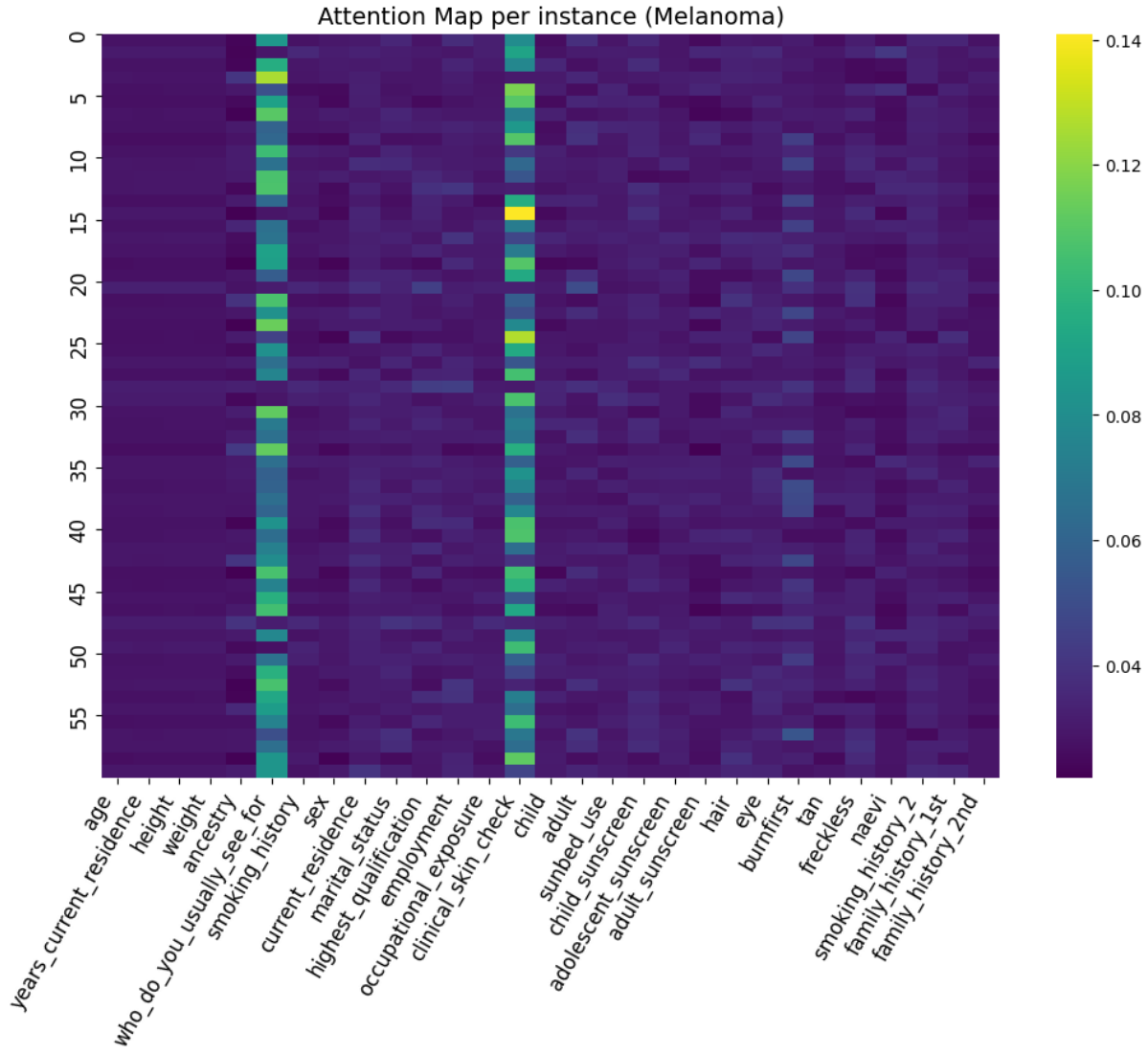


Figure 5.17: Attention map visualization for the melanoma dataset, demonstrating the important features among the True Positive instances from the validation set.

tioner) are indeed crucial and indicative factors, in determining whether a participant's individual risk of being diagnosed with melanoma is high or not. Following these, the features *"burnfirst"*, representing the skin response to sun exposure without sunscreen, *"adult"*, representing the number of sunburns experienced as an adult, *"freckless"* and *"ancestry"*, also receive relatively high attention scores, although not consistently across all samples.

In the same manner, but for the heart attack/disease risk assessment task, Figure 5.18 demonstrates 100 randomly selected samples from the validation set. The figure clearly reveals a pattern where *stroke* occurence presents the highest attention scores. With similar consistency and with relatively high weights, *general health* self-evaluation, *sex*, *chest scan* examination, *age* and *kidney disease* significantly influence the risk score assessment for the majority of

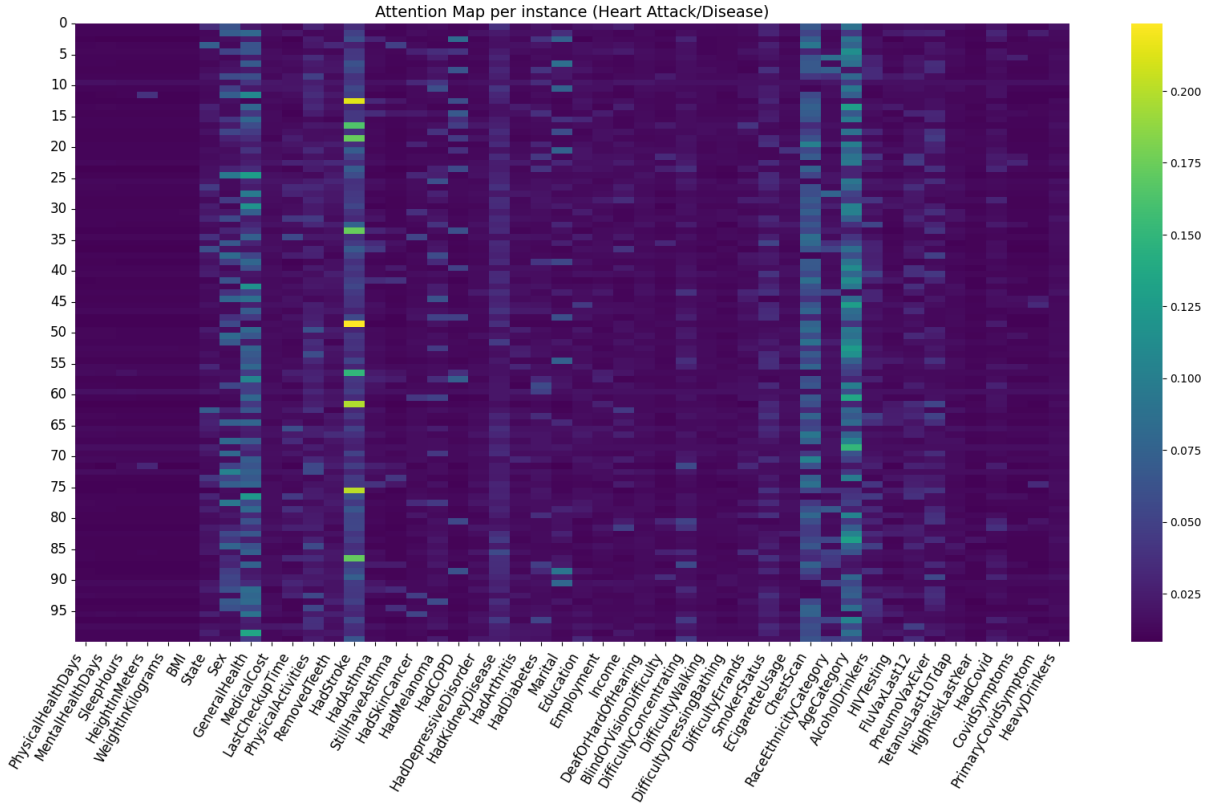


Figure 5.18: Attention map visualization for the BRFSS 2022 dataset, demonstrating the important features among 100 randomly selected True Positive instances from the validation set.

individuals. Conversely, features such as *marital status*, *diabetes*, *melanoma* and *C.O.P.D.* occasionally produce high attention scores, which can be attributed to varying underlying relationships between them and the different query features.

5.4 Ablation Analysis

In this section, we conduct an ablation study in order to evaluate the impact of various model hyperparameters and configurations on the performance during the melanoma classification task. The following experiments aim to provide insights into the optimal settings for achieving high accuracy, robustness, generalization and highlight the trade-offs between various training strategies. Through this study we investigate different model sizes, loss functions, optimizers as well as the model's calibration plots to justify the choice of cost functions.

Table 5.8 presents the results of the ablation study conducted with various model sizes for both the non-negative MLP and the attentive network architectures, on the melanoma risk assessment task. This table shows the model's performance metrics (accuracy & f1-score) in relation

Model	Model Size	Params	Accuracy	F1-score
Non-negative MLP	**64**	**9.7K**	**86.8%**	**0.903**
	128	23.4K	85.5%	0.895
	256	63.2K	86.8%	0.903
	512	192.0K	85.5%	0.895
Attentive network	64	76.3K	89.2%	0.930
	128	**283.4K**	**92.8%**	**0.953**
	256	1.09M	89.2%	0.930
	512	4.28M	88.0%	0.923

Table 5.8: Ablation on non-negative mlp and attentive model sizes based on results for the melanoma dataset.

to the hidden size and the number of learnable parameters. For the non-negative MLP, four different model sizes were evaluated, 64, 128, 256 and 512. It appears that the smaller model size, with the fewer learnable parameters, is sufficient in order to achieve the maximum accuracy of 86.8% and f1-score of 0.903. Further increasing the model size to 128 or even 512, results in a decrease of performance, which is expected due to the very small size of the dataset with just 415 samples available, indicating potential overfitting. For the attentive network, the same four model sizes were tested, with the size 128 being the optimal configuration, achieving an accuracy of 92.8%. It is observed that increasing the model size to 256 and 512 provides decreased results, likely due to overfitting. However a model size of 128 outperforms the size of 64, indicating a more robust architecture with superior generalization ability. This suggest that, in contrast with the non-negative MLP, the attentive network, benefits from an increased complexity up to a certain point.

Table 5.9 presents the ablation study conducted to evaluate the impact of different loss functions and optimizer combinations, on the performance of the non-negative MLP model for both the melanoma and BRFSS 2022 datasets. The loss functions examined include the Binary Cross Entropy (BCE) with and without positional weight for the positive class, as well as Focal Loss with varying α values. The optimizers tested were Adam and RMSProp, which were experimentally selected from preliminary experiments as the ones with a superior performance, as opposed to other optimizers like SGD that totally failed to converge. For the BRFSS 2022 dataset, using a standard Binary Cross Entropy loss function with Adam resulted in a very low accuracy of 59.9% and an F1-score of 0.359. Similar results, with accuracy of 60.7% and F1-score of 0.385, arise by utilizing Focal Loss with $\alpha = 0.5$. These results were greatly affected by the significant imbalance within this dataset. Introducing positional weight *(pos.weight=9)* to the positive class in BCE improved the performance significantly, achieving an accuracy of 76.9% and an F1-

Dataset	Loss Function	Optimizer	Accuracy	F1-score
BRFSS 2022	BCE	Adam	59.9%	0.359/0.533
	BCE w\ pos.weight	Adam	76.9%	0.782/0.769
	Focal w\ $\alpha = 0.5$	Adam	60.7%	0.385/0.548
	Focal w\ $\alpha = 0.7$	Adam	70.9%	0.641/0.698
	Focal w\ $\alpha = 0.9$	Adam	76.4%	0.780/0.763
Melanoma	BCE	Adam	79.5%	0.866/0.772
	BCE	RMSProp	85.5%	0.895/0.855
	Focal w\ $\alpha = 0.5$	Adam	80.7%	0.871/0.792
	Focal w\ $\alpha = 0.5$	RMSProp	86.8%	0.903/0.868

Table 5.9: Ablation on different loss functions and optimizers for both datasets.

score of 0.782. Focal Loss with alpha values of 0.7 and 0.9 also showed improved results, with $\alpha = 0.9$ yielding, among them, the best performance, achieving an accuracy of 76.4%. Greater alpha values and positional weight, help mitigate class imbalance by letting the model to proportionally focus more on the positive class which is the great minority in the dataset. Regarding the melanoma dataset, which does not have the same class imbalance issue, Adam combined with BCE and Focal loss ($\alpha = 0.5$) achieved accuracy of 79.5% and 80.7% respectively. Switching to RMSProp, the results with both loss functions consistently improved, showcasing better convergence. Focal loss outperformed BCE with a resulting accuracy of 86.8%. The aforementioned results highlight the importance of the optimal loss function and optimizer selection, in order to enhance the model's performance, depending on the dataset and its unique properties.

Figures 5.19 - 5.21 provide the calibration plots for logistic regression model and the non-negative MLP with both Binary Cross Entropy and Focal loss utilized, on the melanoma classification task. These plots are useful in order to assess the distribution of the output probabilities and how well they fit in realistic scenarios. The calibration plot consists of the predicted probabilities, within the range $[0, 1]$, calculating the mean value for each bin *(X-axis)* and the actual fraction of melanoma-diagnosed cases within each corresponding bin of predicted probabilities *(Y-axis)*. The dashed diagonal line indicate a perfectly calibrated model, while the calibration curve (magenta line) represent the actual calibration of the model. Additionally, the histogram plot demonstrates the distribution of the predicted probabilities in a more straight-forward way. Figure 5.19 corresponds to the Logistic Regression model, indicating a weakly calibrated model, as the calibration curve deviates from the diagonal line in many cases and with extreme output values in the fraction of positives (0s and 1s). The model tends to overestimate (squared points under the perfectly calibrated line) and underestimate (squared points above the perfectly calibrated line) for many of the average predicted probabilities bins considered. The histogram plot

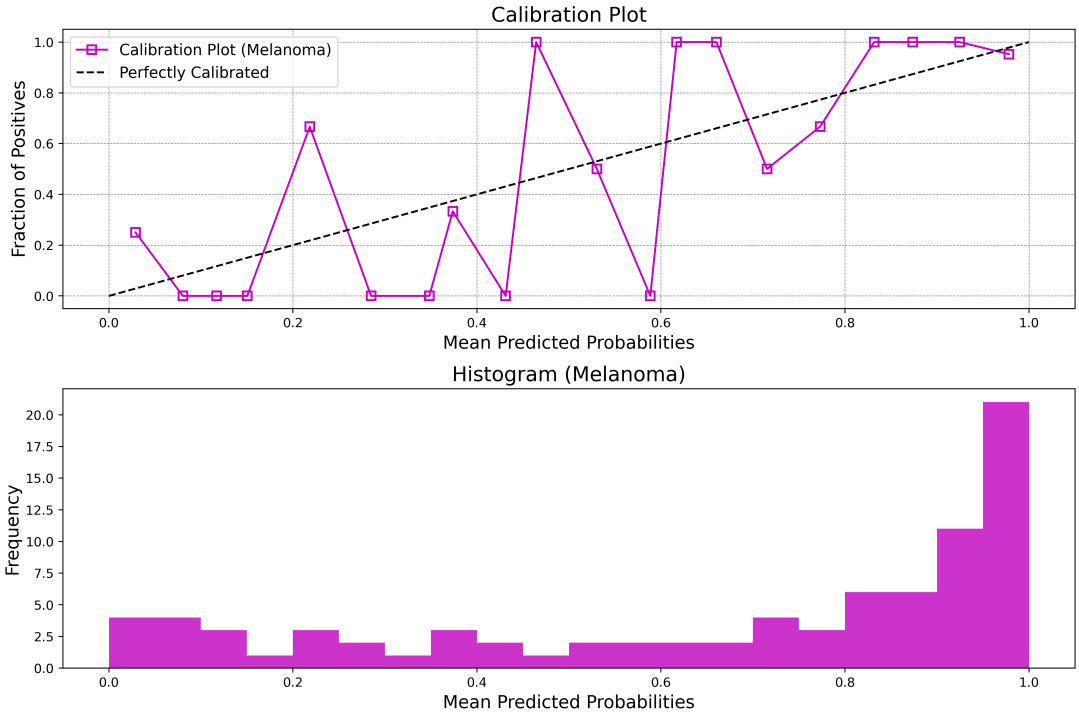


Figure 5.19: Calibration and Histogram Plots for Logistic Regression model on melanoma classification task.

illustrates the model's confidence in its predictions, with a notably high frequency of predictions exceeding 0.8 (80% risk score) being the prime example. Although this confidence might appear robust at first glance, it does not align with realistic scenarios, since in clinical tasks like melanoma and heart attack/disease risk assessment it is uncommon to provide such definitive diagnoses without either a supplementary imaging from a medical examination or an expert's evaluation.

Figures 5.20 and 5.21 provide the calibration and histogram plots for the non-negative MLP models, with the utilization of Binary Cross Entropy and Focal loss (with $\alpha = 0.5$) respectively. It is notable that the non-negative architecture provides a baseline risk around 0.4 (40%) suggesting a realistic uncertainty for every participant and reflecting to real clinical scenarios. We observe that Focal loss provide more calibrated results as shown from the calibration curve around 0.4-0.5 and 0.8 mean probabilities. Finally, BCE demonstrates two significant peaks on the predicted probabilities, at the baseline risk and within a cluster at higher values (0.9-1.0). Conversely, the histogram of the model utilizing Focal loss, provide a more evenly distributed probabilities, avoiding extremely confident predictions (over 90%) while maintaining high accuracy.

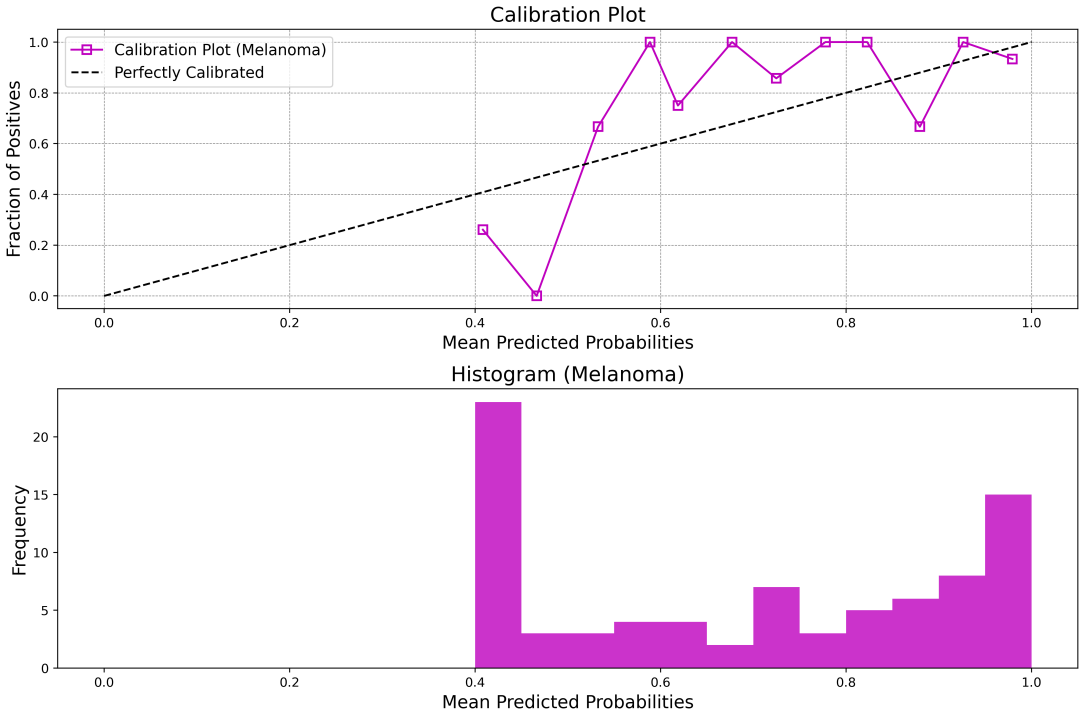


Figure 5.20: Calibration and Histogram Plots for Non-negative MLP model, trained with Binary Cross Entropy (BCE) loss, on melanoma classification task.

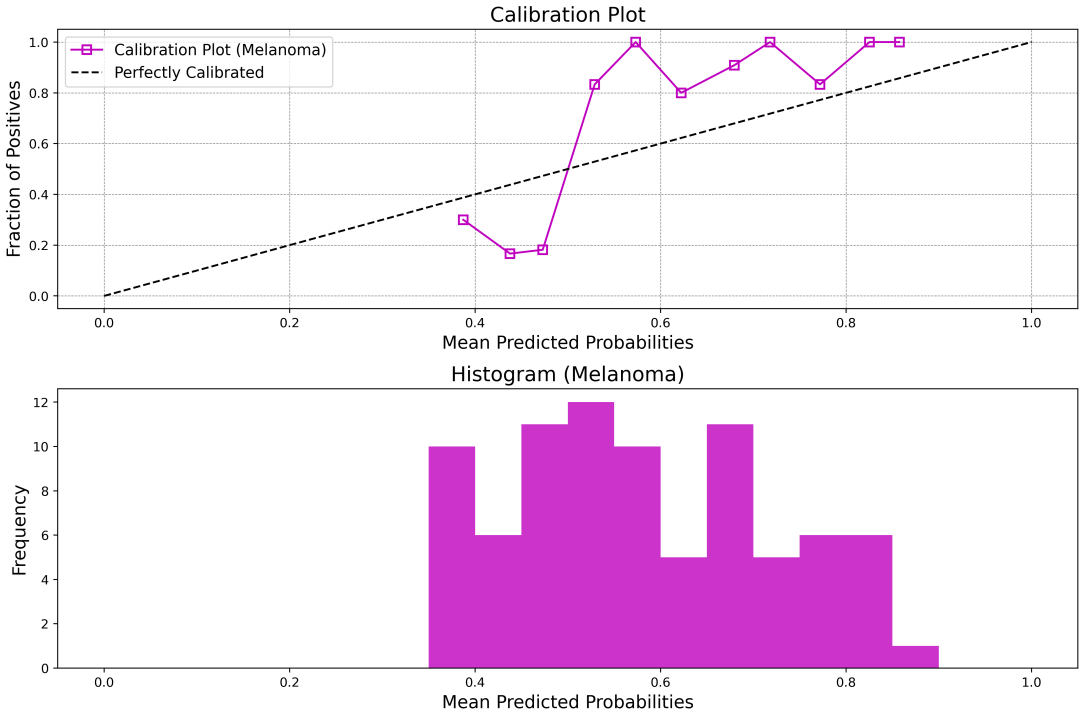


Figure 5.21: Calibration and Histogram Plots for Non-negative MLP model, trained with Focal loss $(\alpha = 0.5)$, on melanoma classification task.

CHAPTER 6

Conclusions

6.1 General Conclusions

In this work, we developed and evaluated two different frameworks for clinical risk assessment, handling tabular data with heterogeneous data types, including numerical, categorical and "checkbox"[1] data. The first architecture is a non-negative multi-layer perceptron (MLP), which constrains all weights to be non-negative (≥ 0), ensuring positive contributions from each feature and the second architecture, an attentive network, that leverages the self-attention mechanism to capture complex feature interactions and provide greater explainability properties. We compare these two approaches with a Logistic Regression model, that stands as the standard method for such tasks in bibliography. All models were trained and evaluated for two different datasets, one for melanoma classification task and a highly imbalanced dataset, from the annual Behavioral Risk Factor Surveillance System (BRFSS) telephone survey for the year 2022, that was preprocessed and utilized for heart attack/disease classification. The results demonstrate that both the non-negative MLP and the attentive network outperform Logistic Regression in terms of quantitative metrics like accuracy, f1-score, specificity, sensitivity etc. Among the two, the attentive network proved to be the most robust in every configuration, effectively handling class imbalance, heterogeneous data and successfully mitigating overfitting. These improved results highlight the potential of deeper and more complex neural network architectures in clinical risk score assessment, providing not only superior predictive performance, by interpreting the task as a binary classification problem, but also useful insights into the underlying data relationships. The proposed architectures, particularly the attentive network, aim to provide a useful supportive tool for clinicians, during the decision-making process for diagnosing various diseases. Due to the interpretability of the results, these models could also be utilized by any interested individual who is not an expert in the field of Artificial Intelligence, making them useful and accessible to a wide audience.

[1] for "checkbox" data definition see Section 5.1.1

6.2 Technical Discussion

Both models considered in this study, have been implemented using the PyTorch framework, and have relatively low computational costs, due to their small number of trainable parameters. Although the attentive network has a slightly increased number of parameters compared to the non-negative MLP, it is still very efficient within the context of tabular data. Additionally, leveraging GPU resources, speeds up the training of each model from scratch, even with very large datasets. For instance, training the attentive network on the BRFSS 2022 dataset, with over 400.000 training samples, took approximately an hour with an NVIDIA RTX A5000 GPU with 24GB VRAM. Evidently, one of the biggest challenges in developing a robust framework for clinical risk assessment, is the requirement for large amount of data points, in order to achieve optimal levels of generalization. The difficulty of data acquisition, for various diseases, primarily due to strict privacy measures in the field of healthcare, pose a significant concern while implementing complex deep learning models. Moreover, most of the publicly available clinical datasets, are characterized by a significant class imbalance. Typically, around 90% of the samples represent negative cases, while only about 10% represent positive instances (participants diagnosed with the specific disease). If not handled correctly, this imbalance will possibly lead to substantial overfitting, undermining the model's ability to accurately predict the positive cases. Finally, another technical challenge is the handling of missing values, which can be present in both numerical and categorical/checkbox features, especially if the dataset originates from a questionnaire survey like BRFSS. In this work we address this issue within the attentive framework for both cases. Regarding the numerical input features, after passed through a two-layer MLP, their weights are correspondingly masked based on data availability. For categorical and "checkbox" feature data types, we apply a padding index to their embeddings before concatenating them with the masked numerical features and passing them through the Transformer Encoder. This approach ensures that missing values are appropriately handled, without the need of dropping the majority of samples and maximize the data utilization.

6.3 Future Work

In the field of clinical risk assessment, data privacy is crucial when training new models. Currently, only Logistic Regression and the proposed non-negative MLP models support anonymization and minimization techniques, ensuring that the resulted models, if required to be shared, they will not leak any personal information. Implementing a uniform pipeline for anonymization/minimization, including the attentive network, would be beneficial. Another important matter is to evaluate how each model performs across more datasets, for various diseases, in order to note possible deviations in accuracy and determine how to make the models more

robust depending on the properties of each dataset. Additionally, more sophisticated methods could be explored to address class imbalance, such as generative techniques and synthetic data creation. Finally, although the proposed methods produce standalone models, capable of assessing an individual's risk score for a disease based solely on tabular data, they could be expanded to integrate multi-modal data. This includes incorporating every possible healthcare data type like text, images and genomics providing a holistic approach for early detection of diseases.

References

1. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* **16,** 321–357. ISSN: 1076-9757. http://dx.doi.org/10.1613/jair.953 (June 2002).
2. Sweeney, L. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **10,** 557–570. ISSN: 0218-4885. https://doi.org/10.1142/S0218488502001648 (Oct. 2002).
3. Cho, E., Rosner, B. A., Feskanich, D. & Colditz, G. A. Risk factors and individual probabilities of melanoma for whites. *J. Clin. Oncol.* **23,** 2669–2675 (Apr. 2005).
4. VanderWeele, T. J. & Robins, J. M. Directed Acyclic Graphs, Sufficient Causes, and the Properties of Conditioning on a Common Effect. *American Journal of Epidemiology* **166,** 1096–1104. ISSN: 0002-9262. https://doi.org/10.1093/aje/kwm179 (Aug. 2007).
5. Krizhevsky, A., Sutskever, I. & Hinton, G. E. *ImageNet Classification with Deep Convolutional Neural Networks* in *Advances in Neural Information Processing Systems* (eds Pereira, F., Burges, C., Bottou, L. & Weinberger, K.) **25** (Curran Associates, Inc., 2012). https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.
6. Ronneberger, O., Fischer, P. & Brox, T. *U-Net: Convolutional Networks for Biomedical Image Segmentation* in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (eds Navab, N., Hornegger, J., Wells, W. M. & Frangi, A. F.) (Springer International Publishing, Cham, 2015), 234–241. ISBN: 978-3-319-24574-4.
7. Simonyan, K. & Zisserman, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition* 2015. arXiv: 1409.1556 [cs.CV].
8. Szegedy, C. *et al. Going Deeper With Convolutions* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015).
9. He, K., Zhang, X., Ren, S. & Sun, J. *Deep Residual Learning for Image Recognition* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016).
10. Kingma, D. P. & Ba, J. *Adam: A Method for Stochastic Optimization* 2017. arXiv: 1412.6980 [cs.LG].

11. Lin, T.-Y., Goyal, P., Girshick, R., He, K. & Dollar, P. *Focal Loss for Dense Object Detection* in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (Oct. 2017).

12. Strauss, T. & von Maltitz, M. J. Generalising Ward's Method for Use with Manhattan Distances. *PLOS ONE* **12,** 1–21. https://doi.org/10.1371/journal.pone.0168288 (Jan. 2017).

13. Vaswani, A. *et al. Attention is all you need* in *Advances in neural information processing systems* (2017), 5998–6008.

14. Weng, S. F., Reps, J., Kai, J., Garibaldi, J. M. & Qureshi, N. Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PLOS ONE* **12,** 1–14. https://doi.org/10.1371/journal.pone.0174944 (Apr. 2017).

15. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (eds Burstein, J., Doran, C. & Solorio, T.) (Association for Computational Linguistics, Minneapolis, Minnesota, June 2019), 4171–4186. https://aclanthology.org/N19-1423.

16. Ho, J., Kalchbrenner, N., Weissenborn, D. & Salimans, T. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180* (2019).

17. Xie, Z., Nikolayeva, O., Luo, J. & Li, D. Building risk prediction models for type 2 diabetes using machine learning techniques. *Prev. Chronic Dis.* **16,** E130 (Sept. 2019).

18. Xu, L., Skoularidou, M., Cuesta-Infante, A. & Veeramachaneni, K. *Modeling Tabular data using Conditional GAN* in *Advances in Neural Information Processing Systems* (2019).

19. Brown, T. *et al. Language Models are Few-Shot Learners* in *Advances in Neural Information Processing Systems* (eds Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. & Lin, H.) **33** (Curran Associates, Inc., 2020), 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

20. Huang, X., Khetan, A., Cvitkovic, M. & Karnin, Z. *TabTransformer: Tabular Data Modeling Using Contextual Embeddings* 2020. arXiv: 2012.06678 [cs.LG].

21. Vuong, K. *et al.* Development and external validation study of a melanoma risk prediction model incorporating clinically assessed naevi and solar lentigines. *Br. J. Dermatol.* **182,** 1262–1268 (May 2020).

22. Goldsteen, A., Ezov, G., Shmelkin, R., Moffie, M. & Farkash, A. Data minimization for GDPR compliance in machine learning models. *AI and Ethics* **2,** 477–491. ISSN: 2730-5961. http://dx.doi.org/10.1007/s43681-021-00095-8 (Sept. 2021).

23. Gorishniy, Y. V., Rubachev, I., Khrulkov, V. & Babenko, A. *Revisiting Deep Learning Models for Tabular Data* 2021. arXiv: 2106.11959 [cs.LG].

24. Goldsteen, A., Ezov, G., Shmelkin, R., Moffie, M. & Farkash, A. in *Data Privacy Management, Cryptocurrencies and Blockchain Technology* 121–136 (Springer International Publishing, 2022). ISBN: 9783030939441. http://dx.doi.org/10.1007/978-3-030-93944-1_8.

25. Rieckmann, A. *et al.* Causes of Outcome Learning: a causal inference-inspired machine learning approach to disentangling common combinations of potential causes of a health outcome. *Int. J. Epidemiol.* **51,** 1622–1636 (Oct. 2022).

26. Tay, Y., Dehghani, M., Bahri, D. & Metzler, D. Efficient Transformers: A Survey. *ACM Comput. Surv.* **55.** ISSN: 0360-0300. https://doi.org/10.1145/3530811 (Dec. 2022).

27. Hendrycks, D. & Gimpel, K. *Gaussian Error Linear Units (GELUs)* 2023. arXiv: 1606.08415 [cs.LG].

28. Dauchelle, V. W., Grenier, T., Durand-Dubief, F., Cotton, F. & Sdika, M. *Constrained non-negative networks for a more explainable and interpretable classification* in *Medical Imaging with Deep Learning* (2024).

29. Centers for Disease Control and Prevention (CDC). *Behavioral Risk Factor Surveillance System Survey Data* Atlanta, Georgia: U.S. Department of Health and Human Services, Centers for Disease Control and Prevention. 2014-2022.

List of Tables

List of Figures

Appendix

A. Dataset Statistical Analysis

A.1 Melanoma Detection Clinical Dataset

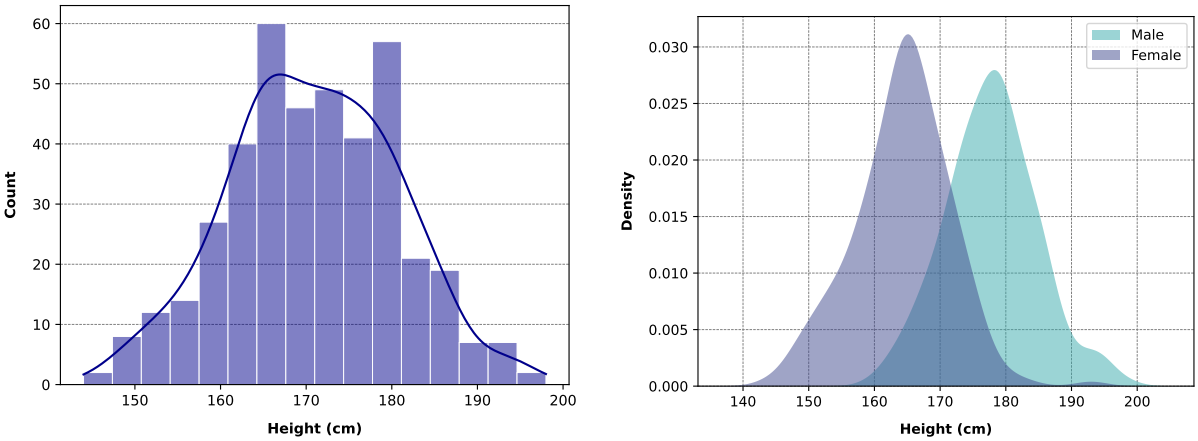


Figure A.1: Comparison of height distribution across the dataset **(left)** and by sex **(right)**.

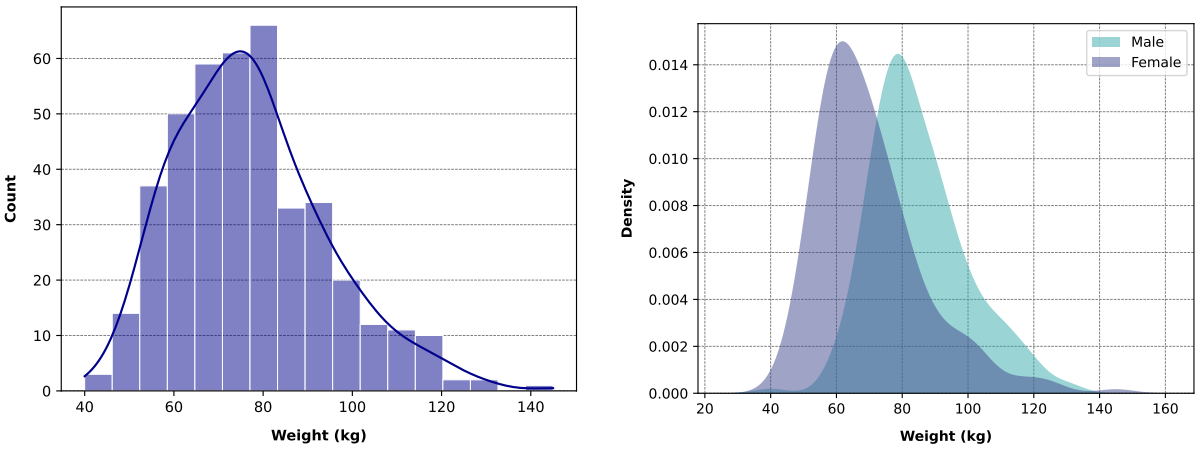


Figure A.2: Comparison of weight distribution across the dataset **(left)** and by sex **(right)**.

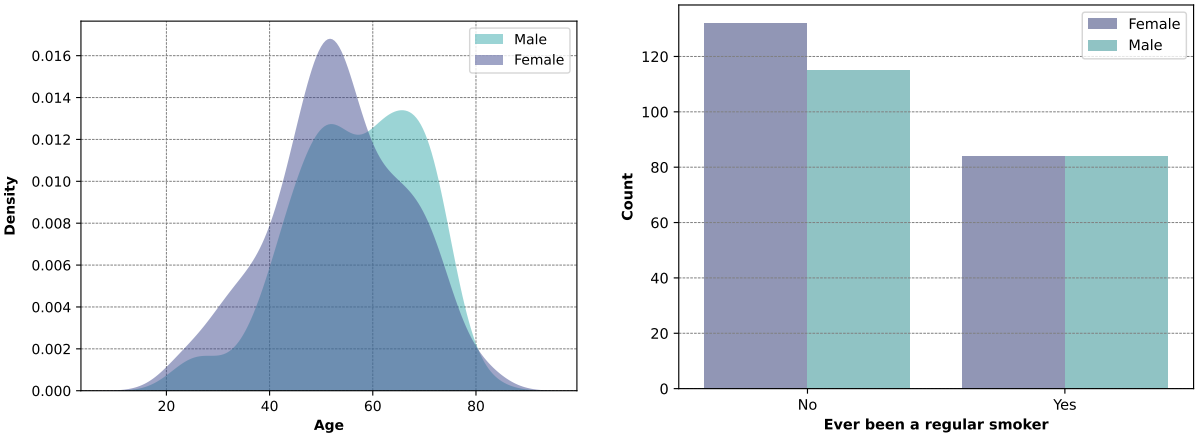


Figure A.3: Age distribution **(left)** and regular smokers count **(right)** by sex.

A.2 2022 BRFSS Survey Data (Heart Attack version)

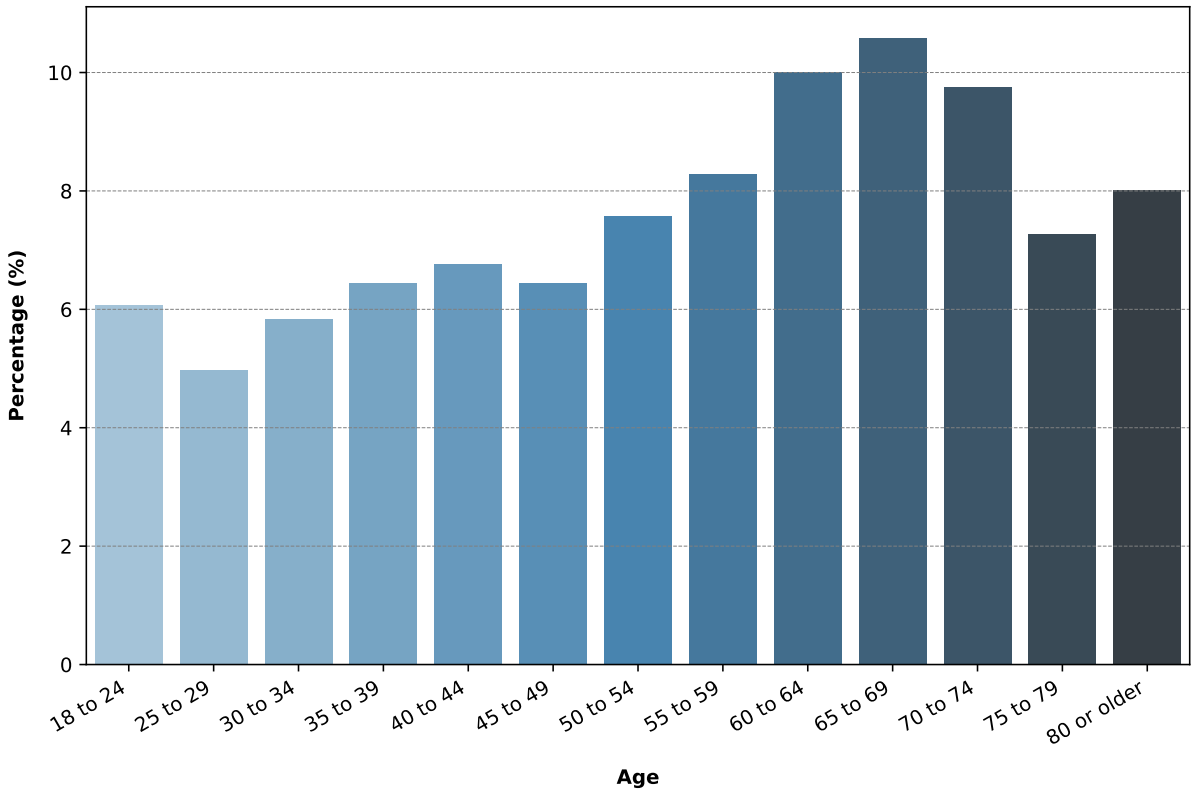


Figure A.4: Fourteen-level age category distribution across the dataset.

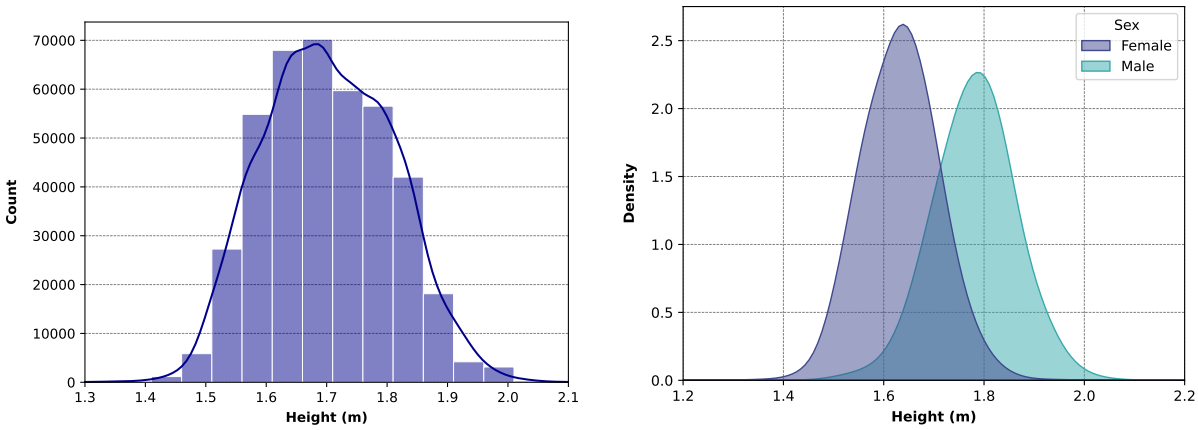


Figure A.5: Comparison of height distribution across the dataset **(left)** and by sex **(right)**.

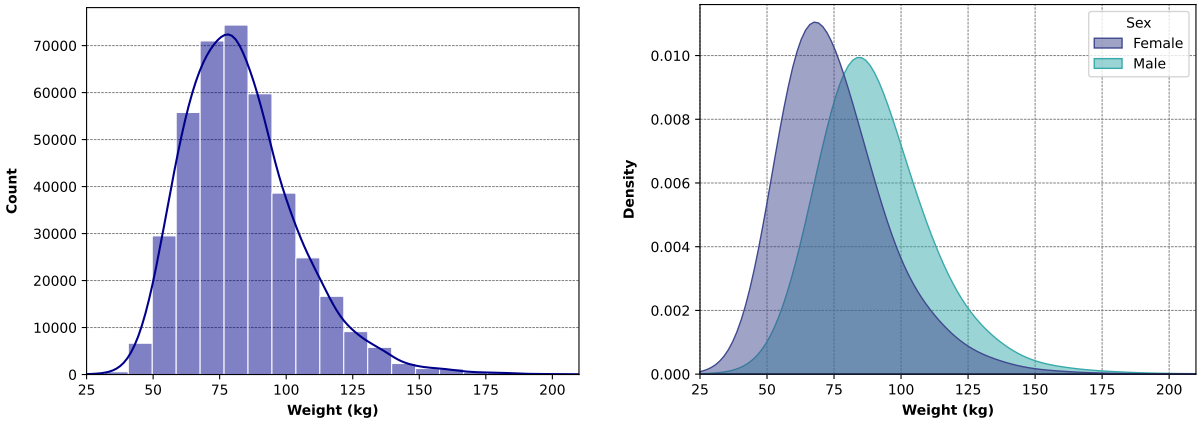


Figure A.6: Comparison of weight distribution across the dataset **(left)** and by sex **(right)**.

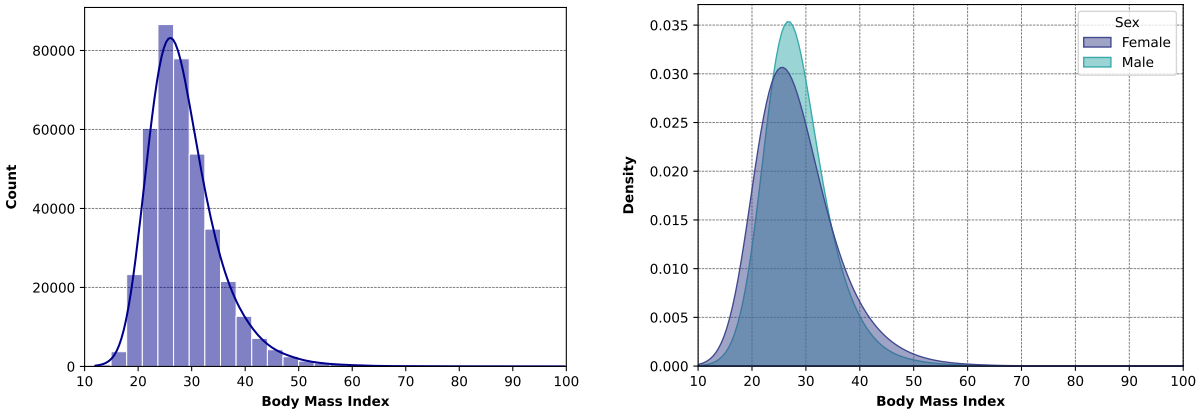


Figure A.7: Comparison of Body Mass Index (BMI) distribution across the dataset **(left)** and by sex **(right)**.

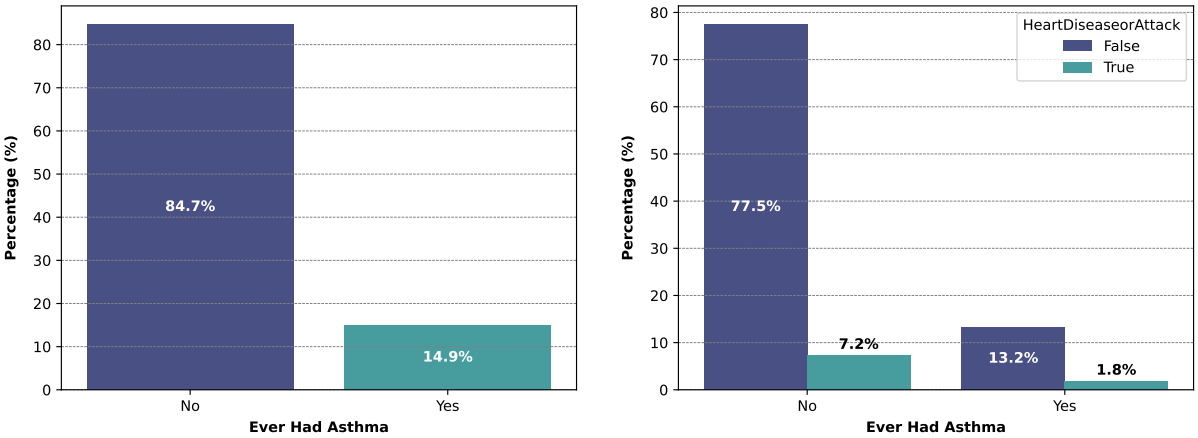


Figure A.8: Patients with asthma history across the dataset **(left)** and by Heart Disease positive/negative history **(right)**.

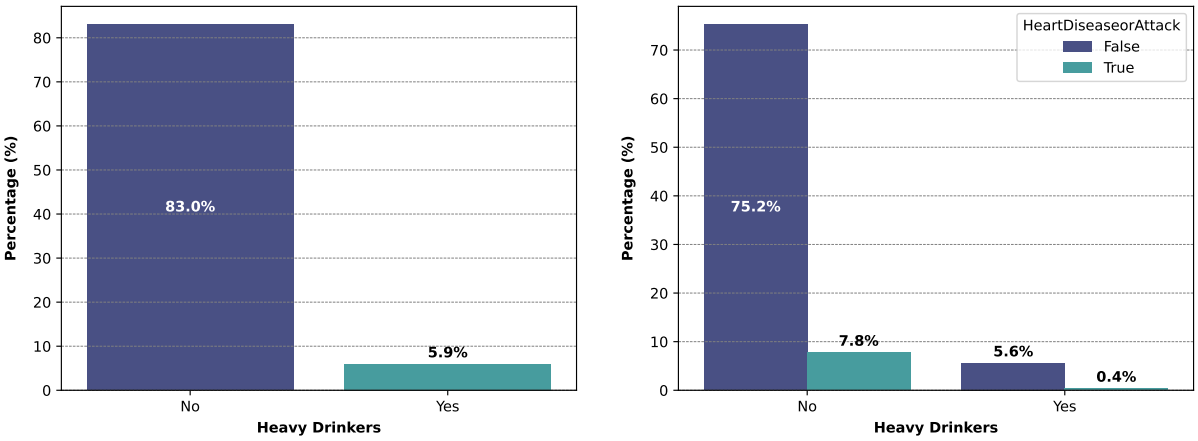


Figure A.9: Patients with heavy drinking history across the dataset **(left)** and by Heart Disease positive/negative history **(right)**.

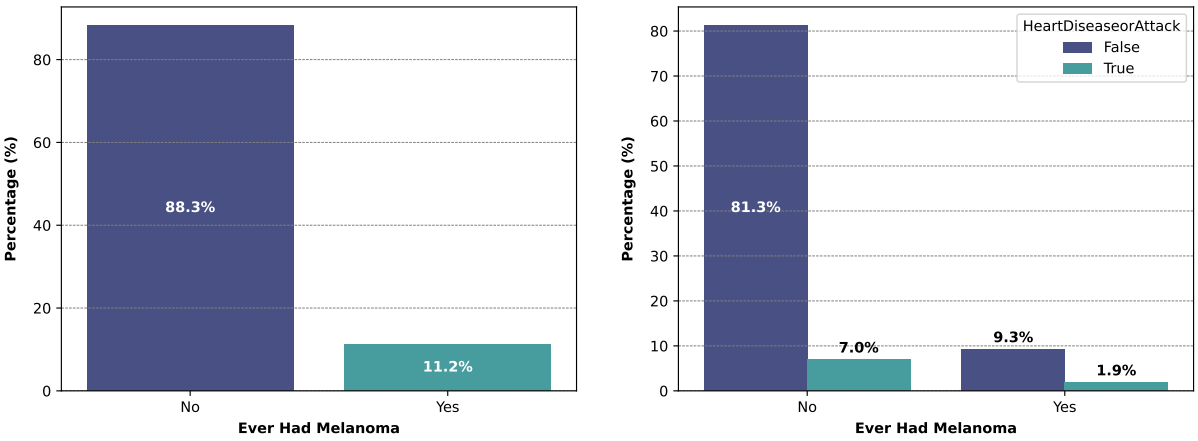


Figure A.10: Patients with melanoma history across the dataset **(left)** and by Heart Disease positive/negative history **(right)**.

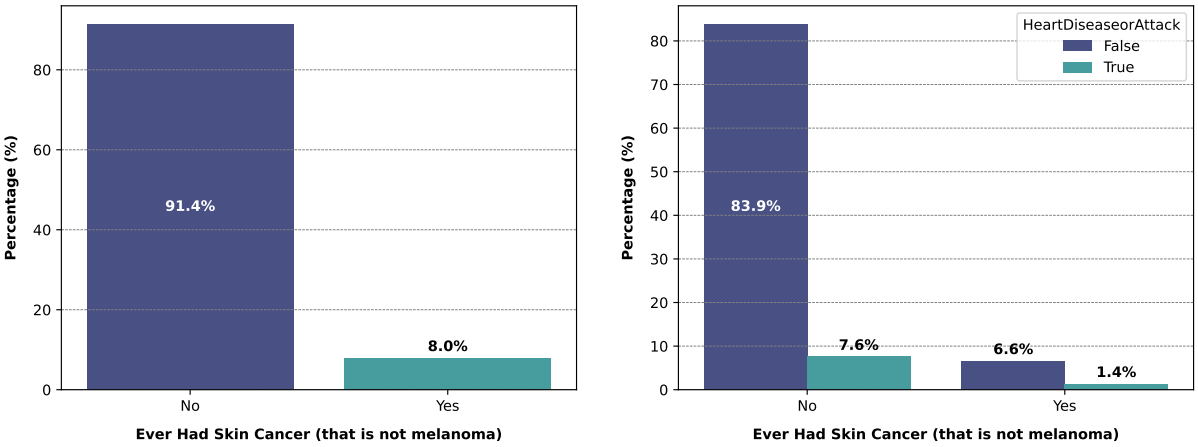


Figure A.11: Patients with skin cancer history across the dataset **(left)** and by Heart Disease positive/negative history **(right)**.

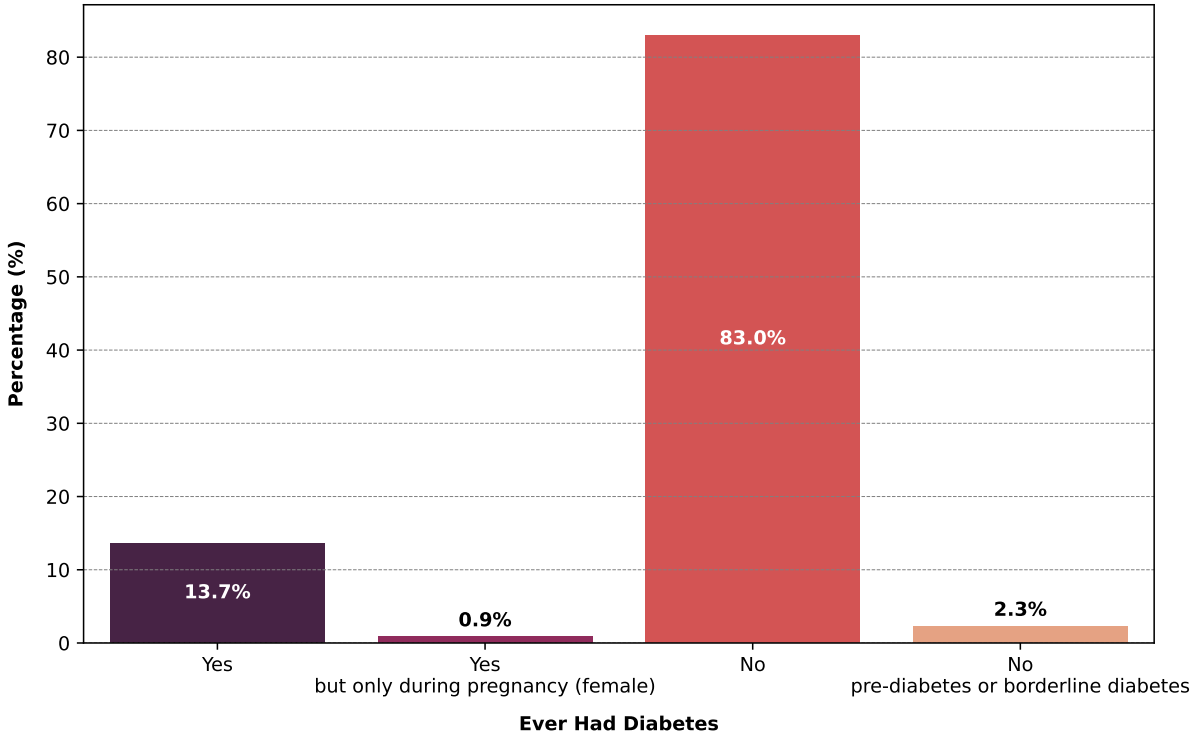


Figure A.12: Patients ever been diagnosed with diabetes across the dataset.

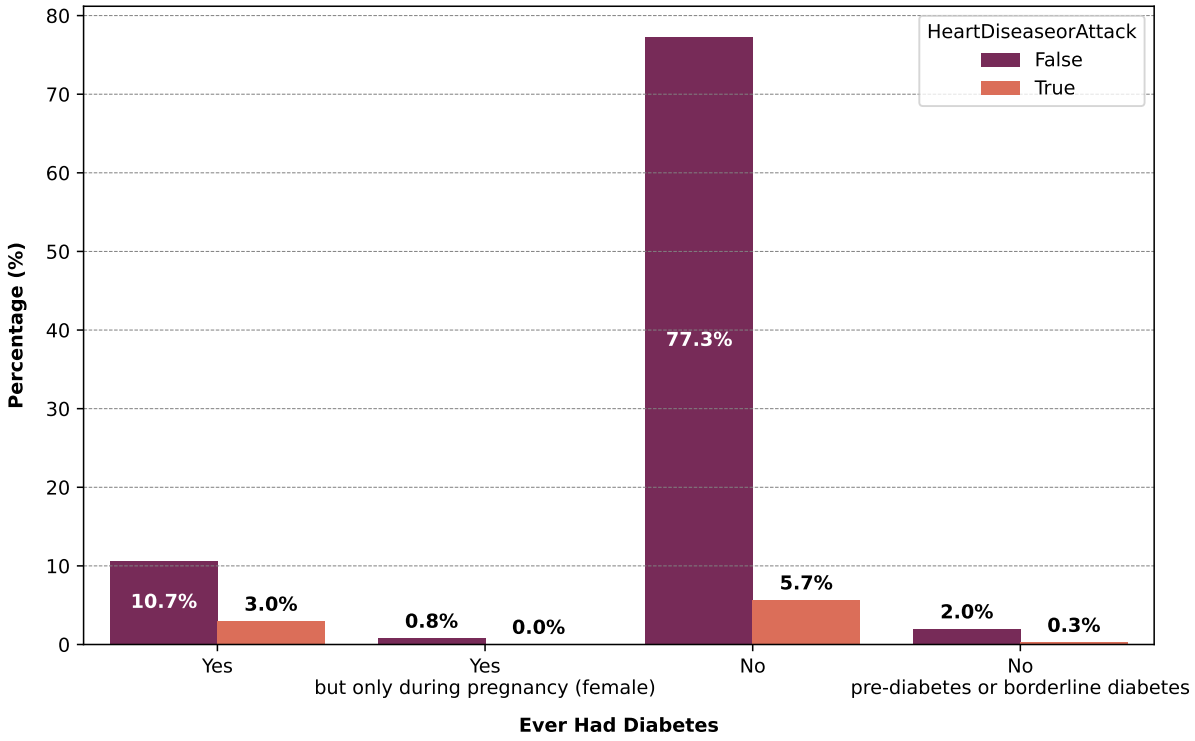


Figure A.13: Patients ever been diagnosed with diabetes in relation to heart disease/attack diagnosis result.