

UPPSALA
UNIVERSITET

UPTEC STS 25037

Degree project 30 credits

June, 2025

Application of neural network techniques for risk stratification of multiple myeloma patients

Daniel Ceoca



Civilingenjörsprogrammet i system i teknik och samhälle



UPPSALA
UNIVERSITET

Application of neural network techniques for risk stratification of multiple myeloma patients

Daniel Ceoca

Abstract

Multiple myeloma(MM) is a challenging and incurable type of blood cancer. The global impact of MM was estimated in 2018, with 160,000 recorded deaths worldwide. Accurate risk stratification remains critical for personalized treatment, yet current models such as the International Staging System (ISS) lack sufficient accuracy and the goal is to build models that can stratify the risk of patients better than ISS.

There remains a lack of accurate and reliable risk stratification models for MM. In this thesis, we investigated neural network (NN)-based predictive models to stratify high-risk MM patients from low-risk ones.

We implemented four NN architectures including 1D-CNN, CNN-MLP hybrid, Resnet and Tabnet. The effect of using input features without any selection was compared to two feature selection methods, variance threshold and NetRank. Furthermore, we investigated the effect of representing input data with and without sorting by chromosomal location of genes. The dataset consisted of 241 patients, with 97 categorised as high-risk patients (death within 3 years of diagnosis) and 144 as as low-risk (survived longer than 3 years).

Our results show that Tabnet combined with feature selection via variance thresholding outperformed all other methods. It achieved an accuracy of 0.72, F1-score of 0.76, recall of 0.92, precision of 0.60 and area under curve (AUC) of 0.58. The performance on accuracy was better than the current clinically used international staging system (ISS), which achieves accuracy of 0.65.

Faculty of Science and Technology

Uppsala University, Place of publication Uppsala

Supervisor: Muhammad Kashif Subject reader: Sven-Erik Ekström

Examiner: Elísabet Andrésdóttir

Sammanfattning

Multipelt myelom (MM) är en hematologisk malign sjukdom som uppkommer i det lymfatiska B-cellssystemet, där myelomcellen utgörs av en malignt omvandlad plasmacell som infiltrerar benmärgen. I Sverige rapporteras lite under 800 nya fall av Multipelt Myelom varje år. Tiden från diagnos till död, varierar kraftigt från några veckor upp till 10 år eller mer. Denna stora variation i överlevnad utgör en betydande utmaning för kliniker, eftersom bristen på tillförlitliga riskstratifieringsmöjligheter försvårar valet av optimal behandling.

För att kunna välja den mest lämpliga behandlingen och utforma personifierade behandlingsstrategier är det avgörande att förstå en patients riskprofil.

I detta arbete användes 241 patienters RNA-data, klassificerade som antingen hög- eller lågrisk patienter beroende på om det levde längre än tre år eller inte för att testa huruvida neurala nätverkstekniker kunde vara användbara i att säga vare sig en patient givet patientens RNA bör klassificeras som hög- eller lågrisk. De fyra neurala nätverksmodellerna testade var ett endimensionell konvultionellt neuralt nätverk, hybrid flerlager perceptron konvolutionellt neural nätverk, Resnet och Tabnet. De bästa resultaten uppnåddes med Tabnet som nådde fram till noggrannhet på 0.72, F1-poäng 0.76, känslighet 0.92 och Precision 0.60. Noggrannheten är bättre än moderna tekniker så som ISS som uppnår cirka 0.65 noggrannhet, men sämre än andra moderna tekniker så som AlgoOS som uppnår en noggrannhet på 0.79.

Acknowledgement I want to express my utmost gratitude to my supervisor during this project, Muhammad Kashif. Not only for providing valuable insights and displaying great domain knowledge, but also for being a very humble and kind mentor. I could not have asked for a better supervisor. Furthermore I would like to thank Karolinska Institutet for providing me with stimulating setting for academic growth. Lastly I would like to thank my reviewer Sven-Erik Ekström for his accurate and incisive help during this project.

Contents

1 Introduction **6**
- 1.1 Multiple myeloma . . . 6
  - 1.1.1 Challenges . . . 6
  - 1.1.2 Current methods of diagnosis . . . 6
- 1.2 Aim . . . 7

2 Background **7**
- 2.1 Neural networks and cancer research . . . 7
- 2.2 Network analysis methods . . . 9

3 Method **9**
- 3.1 Data description . . . 9
  - 3.1.1 Defining OS . . . 9
- 3.2 Dimensionality reduction techniques . . . 9
  - 3.2.1 Variance thresholding . . . 10
  - 3.2.2 NetRank . . . 10
- 3.3 Artificial Image Objects . . . 11
  - 3.3.1 Augmentation . . . 11
- 3.4 Neural network architecture . . . 11
  - 3.4.1 1D CNN . . . 12
  - 3.4.2 CNN-MLP Hybrid . . . 13
  - 3.4.3 Resnet . . . 14
  - 3.4.4 Tabnet . . . 15
- 3.5 Model training and optimization . . . 15
  - 3.5.1 Input size . . . 15
  - 3.5.2 Evaluation metrics . . . 16
  - 3.5.3 Dataset splitting and nested cross-validation . . . 16
  - 3.5.4 Baseline comparisons . . . 16
  - 3.5.5 Workflow . . . 16

4 Results **19**

5 Discussion **20**

6 Conclusion **22**

7 Future Work **22**

1 Introduction

The introduction provides background on multiple myeloma (MM), outlines current clinical challenges, summarizes diagnostic approaches and presents the objectives of the project.

1.1 Multiple myeloma

MM stems from the uncontrolled proliferation of malignant plasma cells within the bone marrow, resulting in the disease phenotype [1]. It accounts for 0.9% of total cancer-related mortalities and 10% of haematological malignancies [2]. The time from diagonsis to death, named overall survival (OS) comes with big variations, ranging from a few weeks to 10 years or more [3]. The substantial variation in OS among patients poses significant challenges for clinicians, as the absence of reliable risk stratification models hampers optimal treatment selection.

Understanding a patient's risk profile is essential for selecting the most appropriate treatment and personalized treatment strategies.

1.1.1 Challenges

MM is a biologically and clinically heterogeneous disease, posing significant challenges for treatment and long-term disease management [4]. The tumor's genetic landscape varies considerably between patients, contributing to differential therapeutic responses and complicating the development of universal treatment protocols. Resistance to therapy — both intrinsic and acquired is a major obstacle, often resulting in relapse and disease progression [4]. Mechanisms of resistance may include clonal evolution, mutations in key oncogenic pathways, and alterations in the bone marrow microenvironment. While some patients benefit from sustained disease control, others experience aggressive disease courses that are refractory to multiple lines of therapy. These challenges underscore the need for personalized treatment strategies and ongoing monitoring to adapt interventions over time [4].

1.1.2 Current methods of diagnosis

Multiple Myeloma (MM) is diagnosed using criteria from the International Myeloma Working Group (IMWG), incorporating laboratory values, imaging, and cytogenetics. The IMWG defines MM as clonal bone marrow plasma cells $\geq 10\%$ plus at least one of the following: CRAB features (hypercalcemia, renal failure, anemia, bone lesions), clonal plasma cells $\geq 60\%$, involved/uninvolved serum free light chain ratio ≥ 100, or ≥ 1 focal lesion on MRI (≥ 5 mm) [5].

Risk stratification is performed using the International Staging System (ISS), based on serum albumin and β_2-microglobulin (s-B2M), and the Revised ISS (R-ISS), which adds lactate dehydrogenase (LDH) and high-risk cytogenetics (e.g., del(17p), t(4;14), t(14;16)) [6].

These criteria support both diagnosis and prognostication, and they guide treatment decisions.

1.2 Aim

In scenarios where no prior knowledge about the problem exists, the primary objective of a machine learning model is to perform better than random guessing. A common metric to quantify this is the Area Under the Receiver Operating Characteristic Curve (AUC), where a score above 0.5 indicates performance superior to chance.

However, for the present task – risk stratification in multiple myeloma (MM) patients, complete ignorance cannot be assumed. Established clinical staging systems such as the International Staging System (ISS) and the Revised ISS (R-ISS) [6] already offer baseline performance, achieving approximately 65% accuracy and recall around 90%.

Therefore, the aim of this project is twofold: first, to develop a model that outperforms random guessing (i.e., AUC > 0.5), and second, to investigate whether performance can surpass that of the ISS. These two baselines will serve as primary benchmarks for evaluating model performance.

In summarizing the objectives, they can be written as:

1. Development of NN based architectures for risk stratification of MM patients.
2. Studying the effect of input data representation and feature selection on model performance.
3. Comparison of model performance against the current clinical used ISS model of risk stratification.

2 Background

Using NN to draw insights from RNA-seq to predict different outcomes associated with cancer has been done over the recent 5-10 years since the emergence of NN in different machine learning applications [7]. Other than cancer, NN techniques have been used to make genomic predictions in live stock and crop breeding, network assessments and alignment classification. [8]–[10].

2.1 Neural networks and cancer research

Little research [7] leveraging NN in MM risk stratification research has been done, although other fields of cancer has had NN applied to them [8], [11], [12]. Many of those papers use the data from The Cancer Genome Atlas (TCGA) [13] which is a dataset covering all 33 cancer types with RNA seq data. CNNs were particularly useful in one paper [14] where RNA-seq samples were reshaped, in no particular way, into what will be denoted artificial image objects (AIO). The

reason this is done is because the RNA-seq data is presented in vector form in the form of $samples \times columns$ where the data and depending on the neural network architecture may require to be passed in as 2d matrices. The AIOs allow CNNs to have the potential of extracting spatial patterns even though the reshaping is arbitrary and may not reflect meaningful interactions in the AIOs gene image.

In another paper CNNs were also used, but instead for classifying the biomarker discovery of triple negative breast cancer, using various databases where TCGA and FUSFCC databases were used [15]. In many articles the genes are reduced down to some number, lets denote it as P amount of genes where $P < features$ that is then transformed into an AIO, but in some papers the genes are few enough that to make an entire image (130×130 size, 169000 pixels) they padded the remaining pixels that couldn't be padded with zero values [16]. To resemble an image, the transcriptomic data was scaled to values in the range of $(0, 255)$ to resemble image data all of the papers.

A notable difference is that in [15] positions of the chromosomes were used for sorting to create a spatial relationship in the AIOs, unlike other articles [14] where just simple reshaping was used. There are other examples of sorting the genes according to their function in cluster and dividing the image into certain areas [17] with a method called $TreeMap$. Other attempts [11], [12] have utilized KEGG-based gene sorting, which involves classifying genes into functional categories using the Kyoto Encyclopedia of Genes and Genomes (KEGG) [18] database. KEGG is a manually curated resource that integrates systems, genomic, chemical, and health information across eighteen databases. Through KEGG mapping tools and the concept of functional orthologs, genes can be organized into molecular pathways and network systems, enabling predictions about cellular functions, disease mechanisms, and virus-host interactions. This form of sorting helps interpret large-scale molecular data in a biologically meaningful way.

The way NN have been used in previous work incorporating NN for risk stratification of MM is by selecting some features, sometimes using genetic data [19], and sometimes without [20]. Prior work has typically followed a workflow of (i) selecting prognostic features, (ii) fitting survival curves using those features, and (iii) optimizing the survival function using neural networks. Notably, these methods employ neural networks primarily for regression or curve-fitting, without explicitly modelling the gene expression data itself. MM presents unique challenges due to its high clonal heterogeneity, complex genetic landscape, and the lack of consistent driver mutations, making it particularly difficult to stratify risk accurately.

As previously stated, it is hard to stratify the risk profile of a patient. One of the best results today stems from Miami University [19] where they developed the *Individualized Prognosis in Multiple Myeloma* (IRMMa) model where they reached a c-index for OS of 0.726 compared to $revised - ISS(0.572)$ and $R2 - ISS(0.625)$.

2.2 Network analysis methods

By integrating the Netrank algorithm [21] and an algorithm called AlgoOS [22] that was designed to stratify the risk newly diagnosed MM patients. AlgoOS uses the feature reduction of transcriptomic data via integrating the NetRank method and then combining clinical data with the selected features with ML techniques such as random forests and SVM to form a biosignature for OS. In this thesis, the feature reduction techniques will be used with the integration of neural networks as the ML component of AlgoOS.

3 Method

The core components of this project consisted of data preprocessing and analysis of the preprocessed data. Both the preprocessing and the analysis could be conducted using multiple approaches. The complete compact workflow of the project is illustrated in Figure 2. The machine learning architectures used and reasoning behind the workflow was made possible via conducting a literature review and builds upon the work referenced in section 2.

A considerable amount of additional work was also carried out, including the construction of artificial image objects, testing of alternative dimensionality reduction techniques, and exploration of other neural network architectures inspired by prior research, although these aspects are not discussed further in this report.

3.1 Data description

The dataset used consisted of 241 patients with a total of 24001 features, all of the features being transcriptomic.

3.1.1 Defining OS

To define the patients as high and low risk for the binary classification problem there was a need to select some threshold marking a patient as high risk or low risk. Dummy variables 0 and 1 were selected for defining low and high risk respectively. Building upon earlier work the threshold was set to < 3 years (1096 days).

Meaning, high risk patients were defined as those who lived for less than 1096 days and vice versa for the low risk patients.

3.2 Dimensionality reduction techniques

As stated in Section3.1, the dataset contains a larger number of categories as opposed to samples, which potentially creates a problem that suffers from curse of dimensionality and features that contributes more to noise than actually being able to represent any meaningful relationship between OS and gene expressions

of the patients. Hence, trying to reduce the dimensionality of the data was a crucial task in this project.

While dimensionality reduction is often justifiable in theory, its practical utility must be evaluated against a baseline model to determine whether it actually enhances predictive performance. In the the project this was controlled for and tested for by comparing against random selection of features and no dimensionality reduction.

3.2.1 Variance thresholding

One of the dimensionality reduction techniques used was using the pre-built selector $VarianceThreshhold$ using *scikit-learns* [23] package. Citing from the source code itself: *"This feature selection algorithm looks only at the features (X), not the desired outputs (y)..."*. Meaning, this follows the principle of checking the variance of the features across the samples and selecting for some set threshold, depending on which threshold that is set, then some number of features is selected.

As the project progressed and different numbers of columns were selected for the testing of models with different input sizes M then numpy features were sorted according to their variance depending on the number of features $M \in [100, 3000]$, for example.

3.2.2 NetRank

Citing from the the original NetRank paper [21]:

> *"in order to measure the relative importance of a document within the set of all web documents, PageRank ranks a document according to the number of highly ranked documents that point to it. Similarly, NetRank assigns a score to a gene which is influenced by the scores of genes linked to it. .."*

Hence, the genes has to be defined according to some network, for the techniques used this time they were defined from the gene co-expression networks [24]. The ranking of the genes was made according to the formula seen in equation (1).

$$r_j^n = (1-d)c_j + d\sum_{i=1}^{N} \frac{w_{ij} r_i^{n-1}}{\deg_i} \qquad 1 \leq j \leq N \tag{1}$$

Where r_j^n denotes the rank of gene (or page) j after n iterations. The symmetric adjacency matrix $W \in \mathbb{R}^{N \times N}$ represents the gene network, where $w_{ij} = w_{ji} = 1$ if genes i and j are connected, and $w_{ij} = w_{ji} = 0$ otherwise. The vector $\mathbf{c}$ contains the absolute Pearson correlation coefficients between gene expression levels and patient survival time.

The damping factor $d \in (0, 1)$ determines the influence of the network versus gene expression data. Specifically:

- $d = 0$: ranking is based entirely on gene expression data.
- $d = 1$: ranking is based solely on network connectivity.

The normalization by $\deg_i$ (degree of gene i) ensures equal voting weight across genes. Each gene receives a base rank of $(1-d)c_j$ and an additional rank based on votes from connected genes scaled by d. In previous applications of AlgoOS d was set to $d = 0.5$, hence this value was used in application and set as a fixed value where the number of genes to be selected were determined by the input length that was to be sent to the neural network.

3.3 Artificial Image Objects

To be able to make use of state-of-the-art neural network technologies, some of them required the input in the form of either a 1D-vector or 2D-matrix. Since the samples of each patient with the transcriptomics features are represented as vectors, there was a need to create Artificial Image Objects (AIO) of the samples so the samples could be represented as AIOs instead of just 1D vectors to be able to use the neural network technologies that needed 2D inputs.

The simplest way of turning the transcriptomic samples into images was by simply reducing the dimensionality to some selected number where the size of the image determined the amount of features needed. For each image, dimensionality reduction using variance thresholding was performed by selecting the M genes with the highest variance. Selecting M features corresponded to an image size of $N \times N$, where $M = N^2$. The other option to create an AIO was by sorting the genes in the order of the location of the chromosomes and then selecting only the genes with highest variance.

To simplify the problem and limit the number of tunable parameters, the size of the AIO matrix was fixed to an $N \times N$ format. A choice was made not to experiment with varying the size of rows and columns of the images but instead to represent the AIOs as $N \times N$ one channel greyscale images. Another limit that was imposed on AIO creation was that all AIOs would be represented as greyscale images, 3 channel images were not tested for.

3.3.1 Augmentation

Data augmentation was applied to increase the diversity of the training set and reduce over-fitting. The process generated additional samples that contained flipped AIOs, vertically and horizontally and zoomed inputs of the AIOs. Implementing this was done using the python package keras built in features that already handles data augmentation.

3.4 Neural network architecture

This section will provide a description of the models used, their structure and hyper-parameters. In Table (1) each model tested can be seen with it's given inputs and it's ability to handle 1D or 2D data.

Table 1: Overview of data categories used in model evaluation

Input	**Neural network**
1D	1D CNN
2D	CNN-MLP hybrid
1D	Resnet
1D	Tabnet

3.4.1 1D CNN

The 1D CNN architecture can be seen in Table 2 and tested hyper-parameters in Table 3.

Table 2: Architecture summary of the 1D CNN model used for binary classification.

Layer (Type)	**Output Shape**
Input (InputLayer)	(None, *input_length*, 1)
Conv1D	(None, *input_length*, *conv1_filters*)
BatchNormalization*	(None, *input_length*, *conv1_filters*)
MaxPooling1D	(None, *reduced_length*, *conv1_filters*)
Conv1D	(None, *reduced_length*, *conv2_filters*)
BatchNormalization*	(None, *reduced_length*, *conv2_filters*)
GlobalMaxPooling1D	(None, *conv2_filters*)
Dense	(None, *dense_units*)
Dropout	(None, *dense_units*)
Dense (sigmoid)	(None, 1)

Batch normalization was included as a tunable option. Final output shape depends on hyperparameter values.

Table 3: Search space for hyperparameter optimization using Optuna.

Hyperparameter	**Search Space**
`conv1_filters`	{64, 128, 256}
`conv1_kernel`	{3, 5, 7}
`pool1_size`	{2, 3}
`conv2_filters`	{32, 64, 128}
`conv2_kernel`	{3, 5}
`dense_units`	{32, 64, 128}
`activation`	{relu, elu, leaky_relu}
`dropout_rate`	Uniform[0.2, 0.6]
`learning_rate`	LogUniform[1e-4, 1e-2]
`batch_size`	{32, 64, 128, 256}
`use_batch_norm`	{True, False}
`epochs`	Integer[10, 30]

3.4.2 CNN-MLP Hybrid

The architecture for the CNN-MLP hybrid can be seen in Table 4. The hyperparameters tested can be seen in Table 5.

Table 4: Advanced Hybrid CNN-MLP Model Architecture

Layer Name	**Type**	**Output Shape**
Input	Input Layer	(img_size, img_size, 1)
v_conv	Conv2D (`vertical_kernels`)	(img_size, img_size, filters)
v_act	Activation (`activation`)	(img_size, img_size, filters)
v_pool	MaxPooling2D (2x2)	(img_size/2, img_size/2, filters)
v_flat	Flatten	(None, N1)
h_conv	Conv2D (`horizontal_kernels`)	(img_size, img_size, filters)
h_act	Activation (`activation`)	(img_size, img_size, filters)
h_pool	MaxPooling2D (2x2)	(img_size/2, img_size/2, filters)
h_flat	Flatten	(None, N2)
g_conv	Conv2D (`global_kernels`)	(img_size, img_size, filters)
g_act	Activation (`activation`)	(img_size, img_size, filters)
g_pool	GlobalAvgPooling2D	(None, filters)
concat_all	Concatenate	(None, N1 + N2 + filters)
dense_1	Dense (`units`)	(None, units)
act_1	Activation (`activation`)	(None, units)
dropout_1	Dropout (`dropout`)	(None, units)
dense_2	Dense (`units`)	(None, units)
act_2	Activation (`activation`)	(None, units)
dropout_2	Dropout (`dropout`)	(None, units)
add	Add (residual)	(None, units)
output	Dense (1, sigmoid)	(None, 1)

Table 5: Hyperparameter Search Space for Advanced Hybrid CNN

Hyperparameter	**Range / Options**	**Type**
`filters`	8, 16, 32	Categorical
`units`	32, 64, 128	Categorical
`dropout`	0.1 to 0.5 (step 0.1)	Continuous (Uniform)
`activation`	relu, elu, leaky_relu	Categorical
`learning_rate`	1×10^{-5} to 1×10^{-3}	Log-Uniform
`batch_size`	32, 64, 128	Categorical

3.4.3 Resnet

The architecture for the Resnet model can be seen in Table 6. The hyperparameters tested can be seen in Table 7.

Table 6: ResNet Model Architecture, example showcases for when the residual blocks are two.

Layer (Type)	**Output Shape**
InputLayer	(None, input_dim)
Dense	(None, 128)
BatchNormalization	(None, 128)
Activation (ReLU)	(None, 128)
Dropout (0.3)	(None, 128)
Residual Block 1	
Dense	(None, 128)
BatchNormalization	(None, 128)
Activation (ReLU)	(None, 128)
Dropout (0.3)	(None, 128)
Add	(None, 128)
Residual Block 2	
Dense	(None, 128)
BatchNormalization	(None, 128)
Activation (ReLU)	(None, 128)
Dropout (0.3)	(None, 128)
Add	(None, 128)
Output Dense	(None, 1)

Hyperparameter	Values
num_blocks	Integer ∈ [1, 3]
units	{32, 64, 128, 256}
activation	{relu, elu, leaky_relu}
dropout_rate	Float ∈ [0.2, 0.6]
learning_rate	Log-uniform ∈ [1e-4, 1e-2]
batch_size	{32, 64, 128, 256}
use_batch_norm	Boolean {True, False}
epochs	Integer ∈ [10, 30]

Table 7: ResNet Hyperparameter Search Space

3.4.4 Tabnet

The Tabnet model used the prebuilt tabnet binary classifier using *pytorch − tabnet* [25], which is a Python package released and maintained by an AI company called DreamQuark that builds on the TabNet [26] model proposed by Google in 2019. The hyperparameters tested for the network can be seen in Table 8.

Table 8: Search space for TabNet hyperparameter optimization using Optuna.

Hyperparameter	Search Space
n_d	{8, 16, 32, 64}
n_a	{8, 16, 32, 64}
n_steps	{3, 4, 5, 6, 7}
gamma	Uniform[1.0, 2.0]
lambda_sparse	LogUniform[1e-5, 1e-2]
optimizer_lr	LogUniform[1e-4, 1e-2]
batch_size	{64, 128, 256}
epochs	Integer[10, 30]

3.5 Model training and optimization

Each model, except for Tabnet which can make use of both categorical and continuous data has to be fed in the transcriptomic data only. That leaves the following for optimisation; inputs, sorted or no sorted and hyper-parameters. For each input and whether the genes were sorted or not there will be a hyper-parameter nested cross-validation loop to fit models where the ones with the best hyper-parameters will be in the final benchmarks.

3.5.1 Input size

Different input sizes for the AIOs and 1D inputs were tested. Values tested for each model can be seen in Table 9. No particular thought process was behind the selection of these inputs, they were selected from the premise of rough testing

of where adding more inputs just seemed to increase noise and not provide any more information through manual testing.

Table 9: Tested input sizes for each model, excluding using all of the features

Model	**Tested Input Sizes**
1D CNN	100, 200, 400, 100, 2000, 4000
CNN-MLP Hybrid	512, 768, 1024
ResNet	100, 200, 400, 1000, 2000
TabNet	50, 80, 100, 200, 400, 1000, 2000, 400

3.5.2 Evaluation metrics

The objective function of optimisation was optimising for $F1 - score$. The severity of the class imbalance in this problem depends on both the dataset used and the operating threshold selected, it can range from somewhat to heavily imbalanced depending on the chosen threshold. $F1 - score$ was selected because accurately identifying true cases is critical in medical contexts.

3.5.3 Dataset splitting and nested cross-validation

To test and evaluate the model and fit the best hyper-parameters nested cross-validation was used. Nested cross validation in this case involved two cross validation loops, both having 5 folds. One outer loop where 20% of the data was held out for validation and the inner loop that was split in an 80/20 split again for training of the hyper-parameters.

Inside the inner loop for each training fold a Bayesian optimisation loop using optuna was ran for 20 trials to find and select the best hyper-parameters. The best hyper-parameters meant the ones occurring most frequently. The best hyper-parameters found were then used on the validation data to evaluate performance of the model. See Figure (1) for visualisation of the nested CV workflow.

3.5.4 Baseline comparisons

As mentioned in Section 1.2, the relevant baseline comparisons that were to be made are against, first and foremost, the random guessing baseline and secondly to compare how these methods compare against ISS and R-ISS. Actual results were displayed using the outer validation data averaged between the 5-fold cross validation used.

3.5.5 Workflow

This section shows pictures of described workflows referenced in the method to give an overview of what was done and how the results were produced in a clean and easy manner.

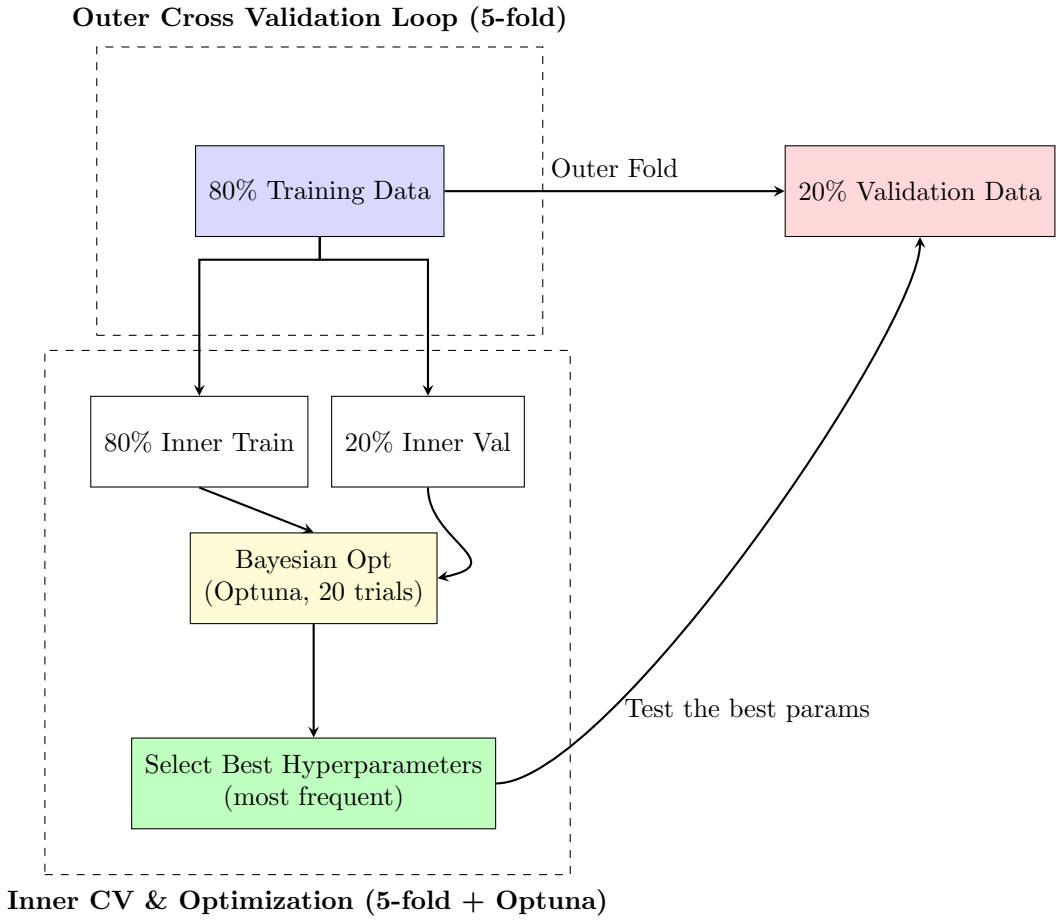


Figure 1: Nested cross-validation with Bayesian optimization inside inner loop using Optuna (20 trials).

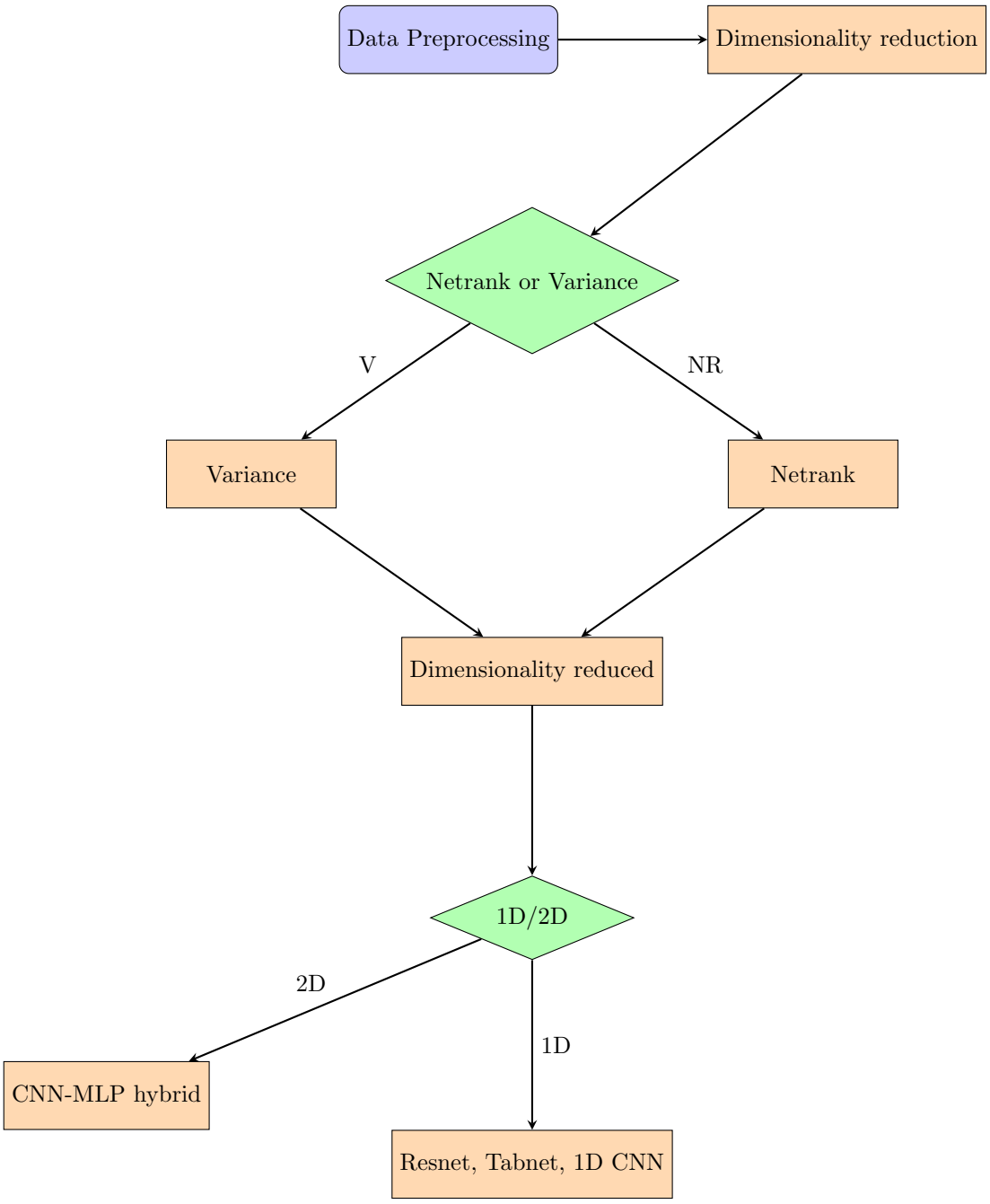


Figure 2: Flowchart with Two Decision Nodes: Feature Reduction and Model Selection

4 Results

For each tested input size, a nested cross validation loop was ran to select the best hyper-parameters which were then used the cross-validation to display the final results. Optimisation for the hyper-parameters were done and the results can be seen in Table 10 displaying the found hyper-parameters for each model for the tested hyper-parameters. The hyper-parameters shown are the ones for the models that performed the best.

Table 10: Selected Hyperparameters for Each Model After Optimization

Model	**Selected Hyperparameters**
1D CNN	`conv1_filters`: 128, `conv1_kernel`: 5, `pool1_size`: 2, `conv2_filters`: 64, `conv2_kernel`: 3, `dense_units`: 64, `activation`: relu, `dropout_rate`: 0.5, `learning_rate`: 0.0016, `batch_size`: 64, `use_batch_norm`: True, `epochs`: 20
CNN-MLP Hybrid	`units`: 16, `dropout_rate`: 0.3, `learning_rate`: 0.0002, `batch_size`: 32
ResNet	`num_blocks`: 3, `units`: 128, `activation`: relu, `dropout_rate`: 0.40, `learning_rate`: 0.0017, `batch_size`: 32, `use_batch_norm`: False, `epochs`: 20
TabNet	`n_d`: 48, `n_a`: 48, `n_steps`: 20, `gamma`: 1.62, `lambda_sparse`: 6.6e-5, `optimizer_lr`: 0.0003, `batch_size`: 64, `epochs`: 15

Table 11: Performance metrics, no dim reduction, no sorting

Model	**AUC**	**Accuracy**	**F1 Score**	**Recall**	**Precision**
1D CNN	0.42	0.59	0.71	0.90	0.60
CNN-MLP Hybrid	0.64	0.58	0.60	0.77	0.49
ResNet	0.70	0.64	0.69	0.68	0.76
TabNet	0.57	0.60	0.70	0.82	0.65

Table 12: Performance metrics for each model, using variance thresholding with no chromosome location sorting

Model	**Input Size**	**AUC**	**Accuracy**	**F1 Score**	**Recall**	**Precision**
1D CNN	400	0.46	0.54	0.66	0.76	0.60
CNN-MLP Hybrid	256	0.53	0.60	0.75	0.97	0.60
ResNet	400	0.61	0.59	0.66	0.7	0.64
TabNet	50	0.58	0.72	0.76	0.92	0.60

Table 13: Performance metrics for each model, using variance thresholding and chromosome location sorting

Model	Input Size	AUC	Accuracy	F1 Score	Recall	Precision
1D CNN	400	0.50	0.55	0.64	0.72	0.61
CNN-MLP Hybrid	1024	0.63	0.61	0.73	0.89	0.63
ResNet	400	0.60	0.63	0.71	0.76	0.66
TabNet	100	0.54	0.56	0.70	0.87	0.59

Table 14: Performance metrics for each model, using the NetRank algorithm for dimensionality reduction. No chromosome sorting applied

Model	Input Size	AUC	Accuracy	F1 Score	Recall	Precision
1D CNN	200	0.49	0.54	0.65	0.72	0.60
CNN-MLP Hybrid	1024	0.51	0.60	0.72	0.62	0.86
ResNet	400	0.68	0.65	0.70	0.71	0.70
TabNet	100	0.48	0.54	0.34	0.25	0.53

Table 15: Performance metrics for each model using random gene selection

Model	Input Size	AUC	Accuracy	F1 Score	Recall	Precision
1D CNN	400	0.47	0.51	0.53	0.63	0.53
CNN-MLP Hybrid	1024	0.54	0.58	0.71	0.9	0.60
ResNet	400	0.64	0.62	0.68	0.7	0.7
TabNet	100	0.52	0.54	0.53	0.56	0.52

5 Discussion

The models that achieved an accuracy beating ISS, meaning accuracy > 0.65 were Tabnet using variance thresholding with no chromosome location sorting, see Table 12 with an accuracy of 0.72. The model that came the closest after Tabnet to matching the accuracy of ISS, but not beating it was Resnet using the NetRank algorithm for dimensionality reduction, see Table 14 and also using no dimensionality reduction and no sorting, see Table 11.

The respective accuracies for the two resnet models were 0.65 and 0.64, indicating what could be denoted as equal performance on accuracy. Meaning, Resnet and Tabnet were the two best performing models. Comparing the F1, Recall and Precision scores in Table 12 for Tabnet and Tables 14, 11 for Resnet, using no dimensionality reduction and no sorting achieved a slightly better F1 score than the Resnet model in Table 12 indicating the best results. Comparing this to Tabnets F1 in Table 12 the comparison of the two best models are:

Tabnet with $F1 = 0.76$, Recall = 0.92, Precision = 0.60 as opposed to Resnet with $F1 = 0.69$, Recall = 0.68, Precision = 0.76. While Resnet outperformed Tabnet only in precision of the metrics mentioned, one could argue that Tabnet is the preferable model due to its significant improvement in recall, a critical metric in medical risk stratification, where correctly identifying positive cases is paramount. Although comparing the AUC metric between the two of $AUC = 0.58$ for Tabnet and $AUC = 0.70$ for Resnet the argument could be made that the threshold for Resnet is performing more than just random guessing which the AUC of Tabnet suggests.

The model that barely beat random guessing with an averague accuracy of 0.54 and had an $AUC < 0.5$ in all of Tables 12 14, 11, 13. The model was the best performing when it used all input features with no sorting, but even then it had an accuracy of 0.59, but at least a $F1 = 0.71$ with a high recall. The results for the other tested inputs were similar with a slightly lower F1-score due to the decrease in Recall. For example using variance thresholding with no sorting of chromosomes with an input size of 400 achieved a Recall of 0.76 versus the Recall of 0.90 using all features with no sorting.

The CNN-MLP hybrid model achieved close to the same F1-score ranging between 0.72 to 0.75 with $F1 = 0.75$ being achieved in Table 12 using variance thresholding and no chromosome location sorting. The accuracy was 0.60 with a bad $AUC = 0.53$, one takeaway from this model is that it is the model that achieved the highest Recall of 0.97 but that came with the issue of having a Precision of 0.60, hence probably why the AUC of the model was poor, indicating that it's a poor cut-off and why the accuracy didn't beat ISS.

It is important to acknowledge several methodological limitations. For example, the use of Binary Cross-Entropy (BCE) as the loss function may have impacted model performance, particularly given the class imbalance in the dataset. Alternative approaches, such as focal loss or weighted cross-entropy, could potentially improve results by better handling under-represented classes.

In a more charitable reading, while the methodological limitations and suboptimal results are valid criticisms, they also reflect the inherent difficulty of the problem at hand. The task—predicting binary outcomes from high-dimensional, class-imbalanced, and noisy clinical data—remains a challenging endeavor even with more sophisticated approaches. This is not to suggest that the method is flawless or the results particularly impressive—on the contrary, there are undoubtedly many additional improvements beyond those already discussed. Rather, the point is to acknowledge that the underlying problem is inherently difficult. A realistic outcome of this project was always the possibility that the proposed approach would struggle to outperform established baselines such as ISS—or even a random classifier—which, to some extent, turned out to be the case. While some models did surpass these baselines, they fell short compared to the performance of more modern methods like AlgoOS [22] that achieved accuracy of 79%, $f1 - score = 85\%$ with $AUC = 76\%$ and $AUC = 92\%$ in two different cohorts.

Lastly, looking at Table 11 and 15 to compare with the dimensionality reduction techniques used in Tables 12, 13 and 14 it looks as if the dimensionality

reduction techniques barely helped for the 1D CNN, CNN-MLP hybrid and only gave a slight boost to Resnet using Netrank for dimensionality reduction. The only model that performed significantly worse using randomly selected genes was Tabnet. Similar results were found for Tabnet using the complete dataset seen in Table 11 as opposed to the dimensionaltiy reduced dataset seen in Table 12 that gave a slight boost in performance using variance thresholding for dimensionality reduction.

6 Conclusion

While TabNet surpassed the ISS baseline (accuracy ¿ 0.65), the other methods did not and in many settings barely performed better than random guessing. Tabnet achieved superior recall (0.92)—critical for minimizing false negatives, despite its lower AUC (0.58), whereas Resnet offered higher precision (0.76) and AUC (0.70). The CNN-MLP hybrid, though achieving the highest recall (0.97), suffered from poor precision and AUC, underscoring the difficulty of balancing metrics in class-imbalanced data. Dimensionality reduction provided marginal gains, with Tabnet being the only model sensitive to feature selection showing significant benefits when using dimensionality reduced data instead of randomly selected.

While the models (TabNet, ResNet) surpassed or came close to the ISS accuracy baseline (accuracy > 0.65), they underperformed compared to modern methods like AlgoOS [22] (accuracy 79%, F1 85%, AUC $76 - 92\%$).

7 Future Work

Although this project did not achieve the predictive performance it initially aimed for, the results do not preclude the potential of neural network-based methods in similar contexts. One clear limitation was the size of the dataset. Neural networks, particularly deep architectures, typically require large volumes of data to effectively learn generalizable patterns. With more samples, especially in the high-dimensional setting of RNA-seq data, models may be better positioned to distinguish between high-risk and low-risk profiles.

A fundamental challenge lies in the nature of the data and how neural networks operate. These models are highly effective when patterns are consistent and plentiful: when the principle of "if it quacks like a duck, walks like a duck, and looks like a duck, it's probably a duck" can be applied. However, in this case, the data may not contain enough clear or repeated instances for the model to reliably learn what characterizes a "high-risk" versus "low-risk" RNA profile. The abstract, non-visual structure of RNA-seq data may also limit the effectiveness of neural networks originally designed for tasks like image or audio recognition.

One further exploration could be to look into AIOs further and see if creating AIOs in other ways could yield better results since it gives the 2D input neural

networks a more meaningful image to learn.

References

[1] D. E. Joshua, C. Bryant, C. Dix, J. Gibson, and J. Ho, "Biology and therapy of multiple myeloma", *Medical Journal of Australia*, vol. 210, no. 8, pp. 375–380, 2019.

[2] S. A. Padala, A. Barsouk, A. Barsouk, *et al.*, "Epidemiology, staging, and management of multiple myeloma", *Medical Sciences*, vol. 9, no. 1, p. 3, 2021.

[3] S. V. Rajkumar, "Multiple myeloma: 2022 update on diagnosis, risk stratification, and management", *American journal of hematology*, vol. 97, no. 8, pp. 1086–1107, 2022.

[4] S. V. Rajkumar, "Treatment of multiple myeloma", *Nature reviews Clinical oncology*, vol. 8, no. 8, pp. 479–491, 2011.

[5] S. V. Rajkumar, M. A. Dimopoulos, A. Palumbo, *et al.*, "International myeloma working group updated criteria for the diagnosis of multiple myeloma", *The lancet oncology*, vol. 15, no. 12, e538–e548, 2014.

[6] A. Palumbo, H. Avet-Loiseau, S. Oliva, *et al.*, "Revised international staging system for multiple myeloma: A report from international myeloma working group", *Journal of Clinical Oncology*, vol. 33, no. 26, pp. 2863–2869, 2015, PMID: 26240224. DOI: 10.1200/JCO.2015.61.2267. eprint: https://ascopubs.org/doi/pdf/10.1200/JCO.2015.61.2267. [Online]. Available: https://ascopubs.org/doi/abs/10.1200/JCO.2015.61.2267.

[7] M. Romero, A. Mosquera Orgueira, and M. Mejía Saldarriaga, "How artificial intelligence revolutionizes the world of multiple myeloma", *Frontiers in Hematology*, vol. 3, p. 1 331 109, 2024.

[8] T. Pook, J. Freudenthal, A. Korte, and H. Simianer, "Using local convolutional neural networks for genomic prediction", *Frontiers in Genetics*, vol. 11, Nov. 2020, ISSN: 1664-8021. DOI: 10.3389/fgene.2020.561497. [Online]. Available: http://dx.doi.org/10.3389/fgene.2020.561497.

[9] A. P. Camargo, V. Sourkov, G. A. G. Pereira, and M. F. Carazzolle, "Rnasamba: Neural network-based assessment of the protein-coding potential of rna sequences", *NAR genomics and bioinformatics*, vol. 2, no. 1, lqz024, 2020.

[10] G. Aoki and Y. Sakakibara, "Convolutional neural networks for classification of alignments of non-coding rna sequences", *Bioinformatics*, vol. 34, no. 13, pp. i237–i244, 2018.

[11] M. Kanehisa, M. Furumichi, Y. Sato, M. Ishiguro-Watanabe, and M. Tanabe, "Kegg: Integrating viruses and cellular organisms", *Nucleic Acids Research*, vol. 49, no. D1, pp. D545–D551, Oct. 2020, ISSN: 0305-1048. DOI: 10.1093/nar/gkaa970. eprint: https://academic.oup.com/nar/article-pdf/49/D1/D545/35363888/gkaa970.pdf. [Online]. Available: https://doi.org/10.1093/nar/gkaa970.

[12] G. López-García, J. M. Jerez, L. Franco, and F. J. Veredas, "Transfer learning with convolutional neural networks for cancer survival prediction using gene-expression data", *PLOS ONE*, vol. 15, no. 3, pp. 1–24, Mar. 2020. DOI: 10.1371/journal.pone.0230536. [Online]. Available: https://doi.org/10.1371/journal.pone.0230536.

[13] J. N. Weinstein, E. A. Collisson, G. B. Mills, *et al.*, "The cancer genome atlas pan-cancer analysis project", *Nature Genetics*, vol. 45, no. 10, pp. 1113–1120, Sep. 2013, ISSN: 1546-1718. DOI: 10.1038/ng.2764. [Online]. Available: http://dx.doi.org/10.1038/ng.2764.

[14] M. Mostavi, Y.-C. Chiu, Y. Huang, and Y. Chen, "Convolutional neural network models for cancer type prediction based on gene expression", *BMC Medical Genomics*, vol. 13, no. S5, Apr. 2020, ISSN: 1755-8794. DOI: 10.1186/s12920-020-0677-2. [Online]. Available: http://dx.doi.org/10.1186/s12920-020-0677-2.

[15] X. Chen, J. M. Balko, F. Ling, *et al.*, "Convolutional neural network for biomarker discovery for triple negative breast cancer with rna sequencing data", *Heliyon*, vol. 9, no. 4, e14819, Apr. 2023, ISSN: 2405-8440. DOI: 10.1016/j.heliyon.2023.e14819. [Online]. Available: http://dx.doi.org/10.1016/j.heliyon.2023.e14819.

[16] L. Rukhsar, W. H. Bangyal, M. S. Ali Khan, A. A. Ag Ibrahim, K. Nisar, and D. B. Rawat, "Analyzing rna-seq gene expression data using deep learning approaches for cancer classification", *Applied Sciences*, vol. 12, no. 4, p. 1850, Feb. 2022, ISSN: 2076-3417. DOI: 10.3390/app12041850. [Online]. Available: http://dx.doi.org/10.3390/app12041850.

[17] A. Abu-Doleh and A. Al Fahoum, "Xgcpred: Cell type classification using xgboost-cnn integration and exploiting gene expression imaging in single-cell rnaseq data", *Computers in Biology and Medicine*, vol. 181, p. 109 066, Oct. 2024, ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2024.109066. [Online]. Available: http://dx.doi.org/10.1016/j.compbiomed.2024.109066.

[18] M. Kanehisa, "Kegg: Kyoto encyclopedia of genes and genomes", *Nucleic Acids Research*, vol. 28, no. 1, pp. 27–30, Jan. 2000, ISSN: 1362-4962. DOI: 10.1093/nar/28.1.27. [Online]. Available: http://dx.doi.org/10.1093/nar/28.1.27.

[19] F. Maura, A. R. Rajanna, B. Ziccheddu, *et al.*, "Genomic classification and individualized prognosis in multiple myeloma", *Journal of Clinical Oncology*, vol. 42, no. 11, pp. 1229–1240, 2024.

[20] L. Bao, Y.-t. Wang, J.-l. Zhuang, *et al.*, "Machine learning–based overall survival prediction of elderly patients with multiple myeloma from multicentre real-life data", *Frontiers in Oncology*, vol. 12, Jun. 2022, ISSN: 2234-943X. DOI: 10.3389/fonc.2022.922039. [Online]. Available: http://dx.doi.org/10.3389/fonc.2022.922039.

[21] C. Winter, G. Kristiansen, S. Kersting, *et al.*, "Google goes cancer: Improving outcome prediction for cancer patients by network-based ranking of marker genes", *PLoS Computational Biology*, vol. 8, no. 5, D. K. Slonim, Ed., e1002511, May 2012, ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1002511. [Online]. Available: http://dx.doi.org/10.1371/journal.pcbi.1002511.

[22] M. Kashif, Y. Zhou, V. Luong, *et al.*, "A novel biosignature for potential stratification and elucidation of newly diagnosed multiple myeloma patients at-risk", *Blood*, vol. 142, no. Supplement 1, pp. 1960–1960, Nov. 2023, ISSN: 1528-0020. DOI: 10.1182/blood-2023-179999. [Online]. Available: http://dx.doi.org/10.1182/blood-2023-179999.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[24] T. Obayashi, S. Hayashi, M. Shibaoka, M. Saeki, H. Ohta, and K. Kinoshita, "Coxpresdb: A database of coexpressed gene networks in mammals", *Nucleic Acids Research*, vol. 36, no. Database, pp. D77–D82, Dec. 2007, ISSN: 1362-4962. DOI: 10.1093/nar/gkm840. [Online]. Available: http://dx.doi.org/10.1093/nar/gkm840.

[25] DreamQuark, *Pytorch tabnet: Attentive interpretable tabular learning*, https://github.com/dreamquark-ai/tabnet, Accessed: 2025-05-08, 2020.

[26] S. O. Arik and T. Pfister, *Tabnet: Attentive interpretable tabular learning*, 2019. DOI: 10.48550/ARXIV.1908.07442. [Online]. Available: https://arxiv.org/abs/1908.07442.