Université Paris Dauphine-PSL
Département de Mathématiques

Project Statistical Learning Subject 3 :

Early Breast-Cancer Detection Using Blood Biomarkers: A Comparative Modelling Study

Teacher :
Mr. Gabriel TURINICI
Ms. Laetitia COMMINGES

Author :
Erwan Ouabdesselam
Antonin Durousseau
Moritz von Siemens
Arthur Danjou
Thaïs Forest

Année académique 2024-2025

Contents

1 Introduction

Breast cancer remains the leading cause of cancer-related death in women: in 2020 the *World Health Organization* recorded 2.3 million new cases and 685 000 deaths. The prognosis depends largely on early detection; yet the reference imaging techniques (mammography and ultrasound) require costly, often overstretched equipment whose sensitivity varies with age and breast density. Blood-borne biomarkers are easy to assay and minimally invasive, therefore, offer a promising trigger for earlier targeted imaging.

All the graphics and numerical results aswell as the corresponding code are to be found at the following github depository: https://github.com/ErwanR123/breast-cancer-detection

1.1 Dataset and Variables

The study draws on the *Breast Cancer Coimbra* dataset, containing the clinical records of 116 women patients followed at Coimbra University Hospital in Portugal. Nine biomarkers were measured for each patient; their clinical rationale is summarized in Table 1.

Table 1: Blood biomarkers and clinical rationale

Biomarker	Rationale
Age	Incidence rises almost exponentially with age owing to accumulated mutations and diminished immune surveillance.
BMI	In post-menopausal women, overweight status is linked to higher tumour risk and poorer prognosis.
Glucose	Chronic hyperglycaemia correlates with greater tumour aggressiveness.
Insulin	Elevated fasting insulin (insulin resistance) is associated with increased breast-cancer risk.
HOMA	Captures metabolic imbalance more finely than insulin alone; high values accompany obesity and heightened risk.
Leptin	Serum leptin is typically raised in breast-cancer patients and may mark disease progression.
Adiponectin	Low levels are often an independent risk factor and may exert a protective effect.
Resistin	High levels correlate with aggressiveness and metastasis, indicating systemic inflammation.
MCP.1	Raised serum MCP-1 links to macrophage tumour infiltration and poorer prognosis.

These variables are the features The target variable `Classification`= 1 denotes histologically confirmed cancer; 0 otherwise.

1.1.1 Type of machine learning

Here the model will train on labeled data to learn mappings from inputs to outputs. Futhermore here the dataset D is represented as :

$$D = \{((X_i, Y_i)\}_{i=1}^{N}$$

where

$$X_i \in \mathbb{R}^9, Y_i \in \{0, 1\}$$

Hence, we face a supervised binary classification Our task is to find

$$\hat{f}\colon \mathbb{R}^9 \longrightarrow \{0,1\}$$

that maximizes positive-case detection.

1.2 Project Goals

1. Detect the vast majority of truly affected patients.
2. Identify the most discriminative biomarkers to inform clinical practice and guide subsequent biological work.
3. Systematically compare methods (bias, variance, interpretability, data requirements) and recommend a model suitable for hospital deployment.

1.3 Modelling Methods

Why test several model families? The *Breast Cancer Coimbra* data set is small ($N = 116$ patients, $d = 9$ biomarkers). The underlying biological mechanisms may be *simple* (quasi-linear relationships) or *complex* (non-linear interactions, threshold effects). Evaluating a range of algorithms therefore serves three purposes:

- **Probe different assumptions** about the mapping $\mathbb{R}^9 \to \{0,1\}$, from strict linearity to highly non-linear decision surfaces;
- **Compare bias, variance and interpretability**, gauging how each model balances overfitting and explanatory power;
- **Select the most stable and clinically useful solution** once performance is assessed on an independent test set.

To span interpretable linear to flexible non-linear approaches, we tested:

K-Nearest Neighbours. Non-parametric; captures complex decision boundaries. Sensitive to scaling and dimensionality. Optimal k chosen via cross-validation.

Penalised logistic regression. Readable coefficients; supports class weighting and variable selection. Assumes a log-linear link between standardised biomarkers and log-odds.

Gaussian Naïve Bayes. Very low variance; effective on small samples. Assumes conditional independence and normality.

Shallow neural network (MLP). It explores the potential non-linearity among biomarkers, while the constrained architecture and early stopping help curb overfitting on this small dataset.

Method comparison at a glance

Together, these four approaches cover a spectrum from a highly interpretable linear model (penalised logistic regression) to a non-parametric learner (k-NN), via a generative classifier (Naive Bayes) and a more flexible multilayer perceptron. Their comparative performance on the held-out test set (accuracy, F1-score and AUC-ROC) allows us to select the solution that offers the best trade-off between **sensitivity**, **interpretability** and **robustness** for the intended clinical application.

2 Modeling methods

2.1 Data processing and analysis

Before proceeding with any modeling phase, it is essential to preprocess the dataset to make it readable and suitable for use with various predictive methods. This preprocessing step allows us to: identify the variables, detect any missing values, understand their behavior and distribution, and assess their influence on the target variable. These insights will help guide the selection of the most appropriate predictive models to apply.

Not all of these steps are required for every model, and some may vary slightly to ensure compatibility between the dataset and each specific algorithm. Any such differences will be detailed at the beginning of each modeling approach.

2.1.1 Data import and preliminary exploration

Dataset overview

The first 5 rows of the dataset are :

	Age	BMI	Glucose	Insulin	HOMA	Leptin	Adiponectin	Resistin	MCP.1	Classification
0	48	23.500000	70	2.707	0.467409	8.8071	9.702400	7.99585	417.114	1
1	83	20.690495	92	3.115	0.706897	8.8438	5.429285	4.06405	468.786	1
2	82	23.124670	91	4.498	1.009651	17.9393	22.432040	9.27715	554.697	1
3	68	21.367521	77	3.226	0.612725	9.8827	7.169560	12.76600	928.220	1
4	86	21.111111	92	3.549	0.805386	6.6994	4.819240	10.57635	773.920	1

The dataset contains a total of 116 observations and 10 variables, including:

- **9 continuous explanatory variables:** *Age*, *BMI*, *Glucose*, *Insulin*, *HOMA*, *Leptin*, *Adiponectin*, *Resistin*, and *MCP.1*.
- **1 target variable:** *Classification*, which takes the value 2 if the patient has cancer (positive class) and 1 otherwise (negative class).

There are no missing values in the dataset, allowing us to proceed directly with the exploratory analysis.

Model	**Bias**	**Variance**	**Interpretability**	**Main interest**
k-NN	low	moderate (↑ with small k)	medium (nearest neighbours)	detect complex fronts
Naive Bayes	high (strong assumptions)	very low	good (means / σ)	ultrafast probabilistic baseline
Logistic Regression	medium	low–moderate	excellent (coefficients)	clinically standard, tunable threshold
MLP	low	higher	low	probes non-linear gain

Table 2: Complementary strengths of the four modelling families.

Variable distribution

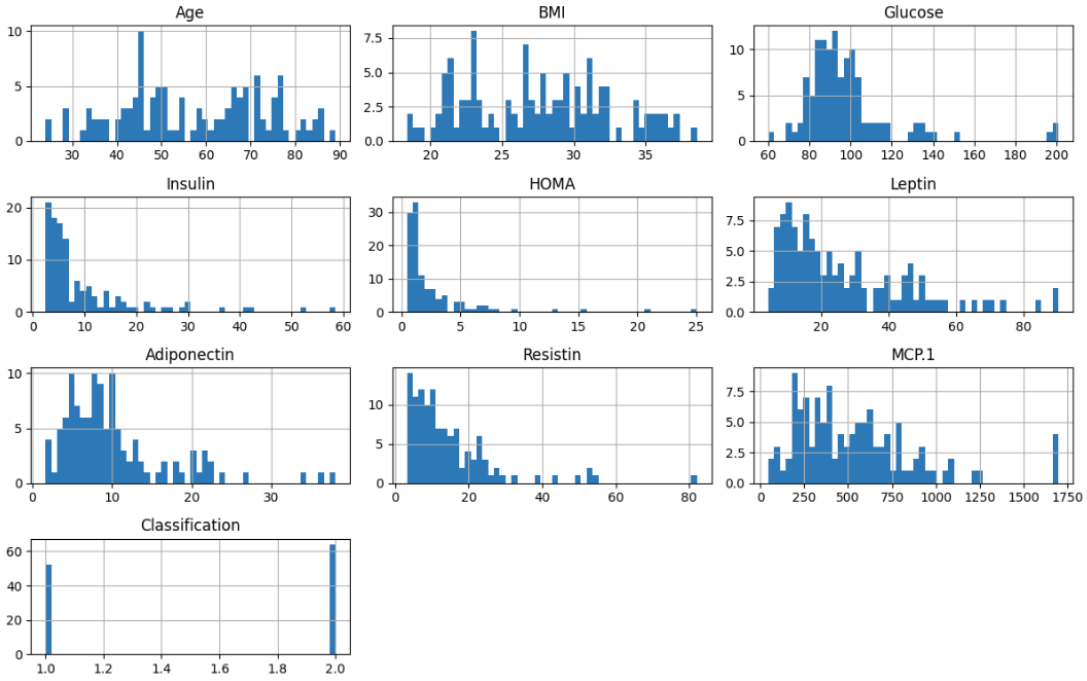


Figure 1: Histograms of variables

The *Classification* variable is not balanced:

- Class 2 (cancer): 55.2%
- Class 1 (healthy): 44.8%

The distribution of the variables was examined using histograms (cf 1). Several variables, including *Insulin*, *HOMA*, *Leptin*, *Adiponectin*, *Resistin*, and *MCP.1*, show a strong asymmetry with extreme values. This may justify applying data transformations (logarithmic scaling or normalization) prior to model training, in order to enhance model performance and stability.

In addition, the *Classification* variable is imbalanced, with 55% of the patients classified as sick (value 2). This imbalance is important to consider, as it directly affects the choice of evaluation metrics. Relying solely on accuracy would not be sufficient to assess model quality in this context.

We will now prepare the data for training by separating the explanatory variables (features) from the target, and performing a stratified split of the dataset into training and test sets.

2.1.2 Stratified train-test split

```
Train: (92, 9) (92,)
Test : (24, 9) (24,)

Train class distribution: Classification
1    0.554348
0    0.445652
Name: proportion, dtype: float64

Test class distribution : Classification
1    0.541667
0    0.458333
Name: proportion, dtype: float64
```

We began by separating the features from the target variable (*Classification*), recoding the latter to facilitate result interpretation:

- 1: patient with disease (positive class),
- 0: healthy patient (negative class).

A **stratified train-test split** was performed using the `train_test_split` function from `scikit-learn`, ensuring that the original class distribution was preserved in both the training and test sets. This is particularly important in the present case, given the observed class imbalance.

The resulting dataset sizes are as follows:

- **Training set**: 92 observations
- **Test set**: 24 observations

This corresponds to a standard 80% / 20% split. This ratio split are not the same for all the modeling methods.

2.1.3 Visualising feature distributions by class in the training dataset

To better understand the differences between patients with the disease (Class 1) and healthy patients (Class 0), we visualised the distributions of explanatory variables segmented by class. Three complementary types of plots were produced: histograms, boxplots, and jittered scatter plots.

Histograms by Class

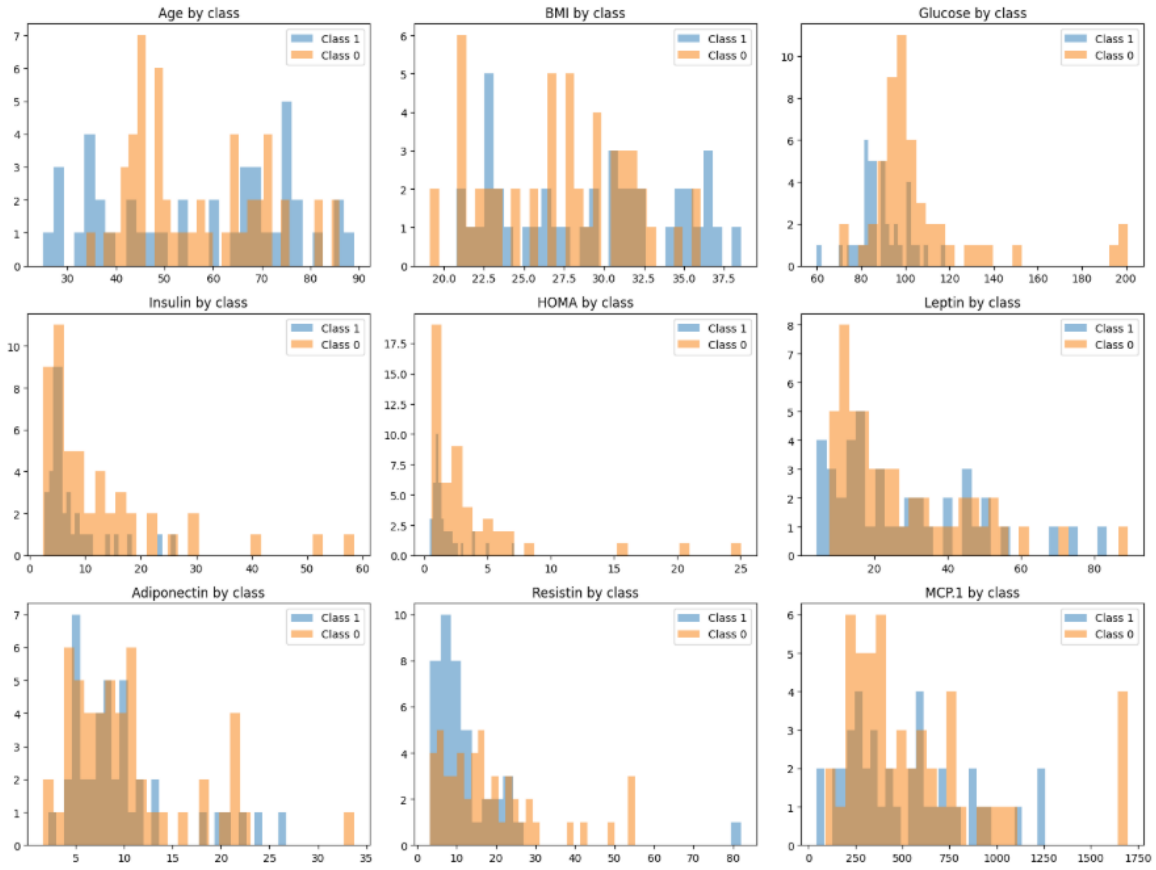


Figure 2: Histograms by class (before transformation)

Histograms (cf 2) allow us to observe the distribution of values within each class. Several variables exhibit noticeable shifts between the two groups, in particular:

- **Age**: patients with the disease tend to be slightly older on average.
- **Resistin**, **MCP.1**, and **Adiponectin** show density shifts between classes, suggesting potential discriminative power.
- In contrast, variables such as **BMI** and **Leptin** exhibit strong overlap across classes.

These qualitative observations indicate that some variables may hold predictive value, but none alone offers a clear class separation.

Boxplots by Class

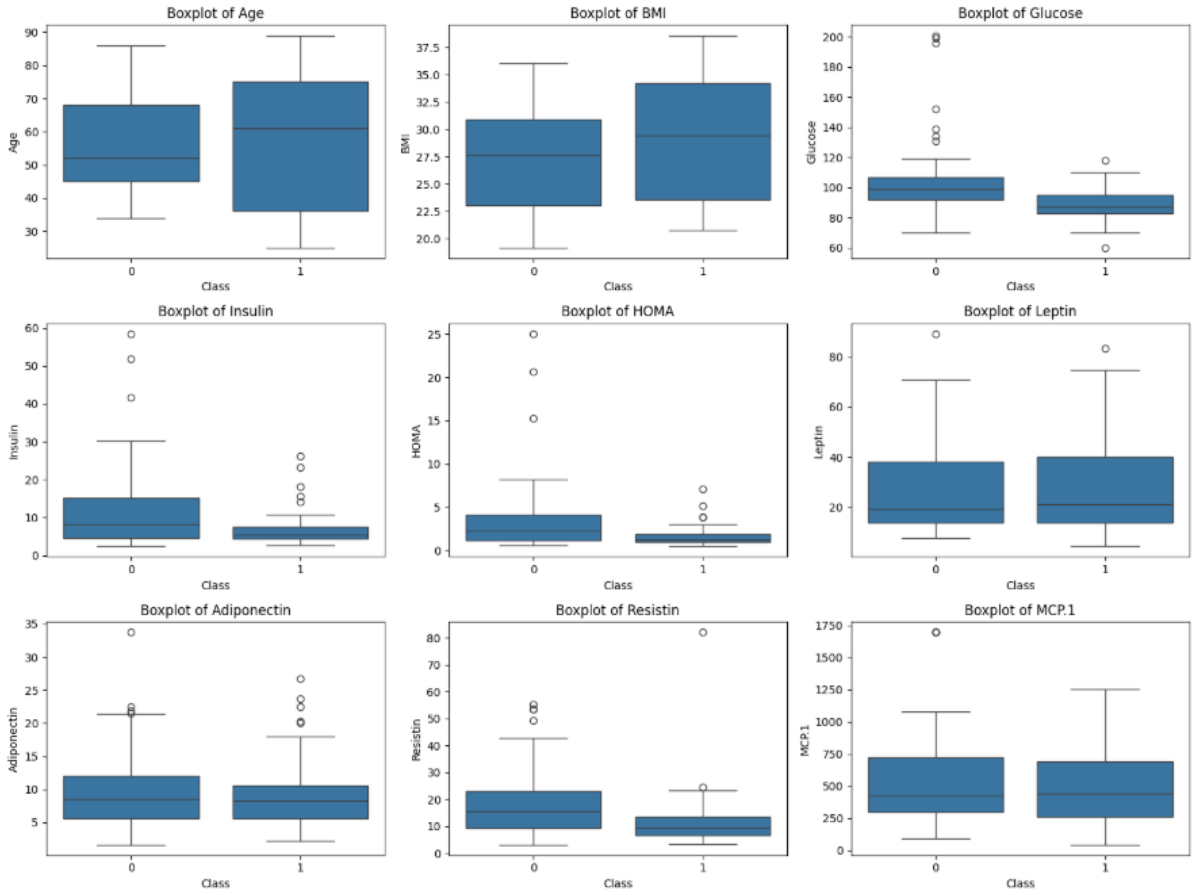


Figure 3: Boxplots by class (before transformation)

Boxplots (cf 3) provide a more precise visual assessment of distributional differences:

- Most variables display high dispersion and numerous outliers in both classes, notably **Insulin**, **Leptin**, and **MCP.1**.
- While medians are generally close between groups, some trends are visible (higher **Resistin** levels in Class 1).

These patterns suggest significant intra-class variability, which may impact the performance of models that are sensitive to non-standardised feature scales.

Jittered Scatter Plots

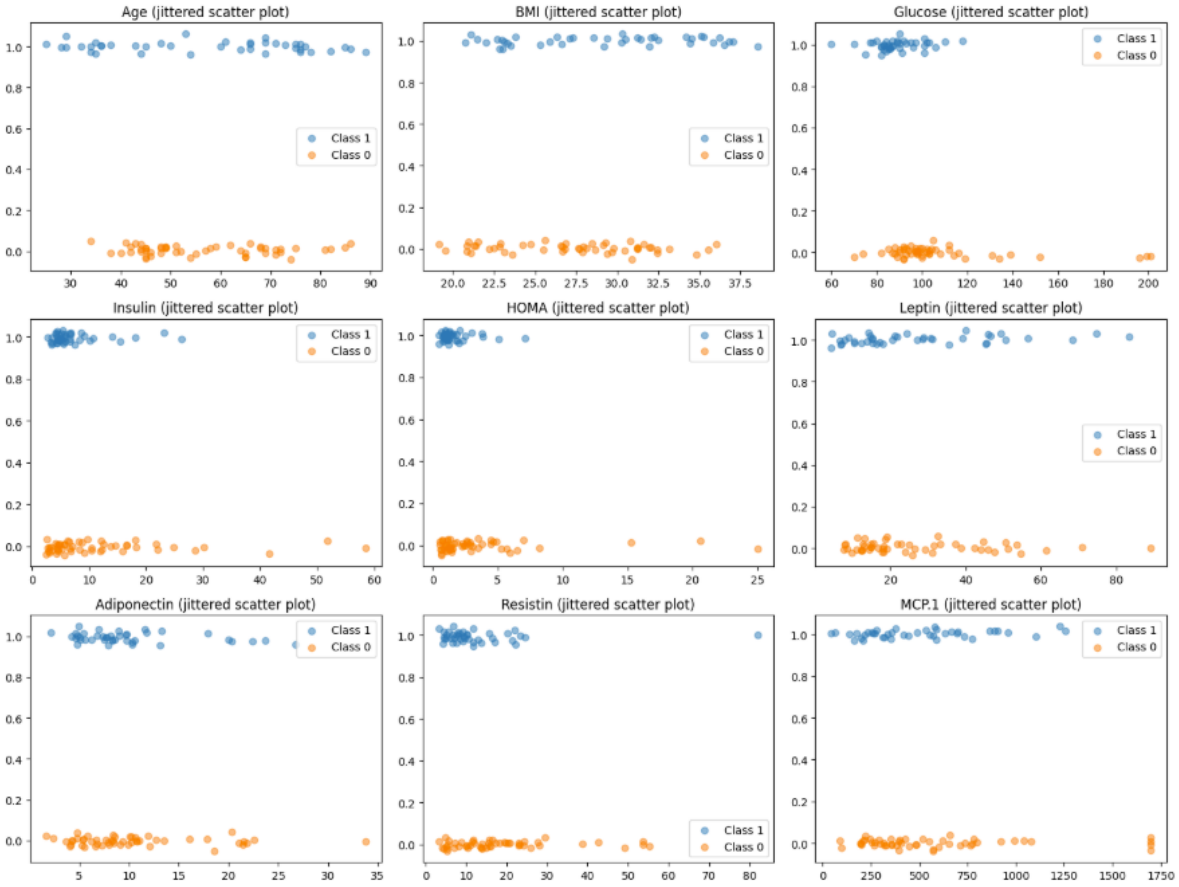


Figure 4: Jittered scatter plots by class (before transformation)

These plots (cf 4) make the vertical separation between classes (0 or 1) explicit:

- No single variable is sufficient to clearly separate the classes.
- Some patterns emerge (low values of **Insulin**, **HOMA**, and **Adiponectin** are associated with Class 0), although significant overlap remains.

Overall, these visualisations support the idea that class separation may only be achieved through multivariate modelling rather than univariate thresholds.

Descriptive statistics by class

To complement the visual inspection of variable distributions, we computed descriptive statistics separately for each class (*healthy* vs *cancer*) to numerically assess the differences in feature distributions.

Healthy patients (class 0):

- Mean Glucose: 87.8, Insulin: 7.3, HOMA: 1.63, Resistin: 11.6
- Standard deviations are relatively high for most variables (e.g., Insulin: 5.35, Leptin: 20.24), indicating substantial within-group variability.

Cancer patients (class 1):

- Mean Glucose: 106.7, Insulin: 11.8, HOMA: 3.36, Resistin: 16.6

- The distributions tend to be more dispersed (e.g., std for Glucose: 25.5 vs 9.8 in class 0) and show higher mean values across most metabolic variables.

These descriptive differences confirm and reinforce the trends observed in the histograms: class 1 patients tend to have higher values for glucose metabolism markers (Glucose, Insulin, HOMA) and inflammatory indicators (e.g., Resistin). The combination of elevated mean values and strong variability highlights the importance of careful feature scaling and transformation prior to modeling.

Summary of the Exploratory Analysis

The exploratory analysis suggests that several variables hold partial discriminative power, but their individual effect is limited. Combining them within a multivariate model will likely be necessary to capture relevant interactions.

We observed substantial skewness and outliers in variables such as *Insulin*, *HOMA*, *Leptin*, and *MCP.1*, particularly within the cancer group. These distortions justify:

- Applying logarithmic transformations to reduce skewness and stabilize variance,
- Standardising all features to account for differences in scale before model training.

Descriptive statistics by class also confirmed the trends seen in visualizations: patients with cancer tend to have higher values of metabolic markers (e.g., *Glucose*, *Insulin*, *Resistin*) and more dispersion, suggesting a more heterogeneous profile.

Finally, the strong overlap between classes in most variables indicates that linear separation may be insufficient, motivating the consideration of non-linear models in subsequent analyses.

These findings provide a clear rationale for the preprocessing steps and modeling choices adopted in the next sections.

2.1.4 Correlation matrix and scatter plot matrix (training dataset only)

To detect potential redundancies and anticipate multicollinearity issues, we analysed the linear relationships between explanatory variables using two complementary tools: the correlation matrix and the scatter plot matrix. All analyses in this section are based solely on the training dataset to avoid any form of data leakage.

Correlation Matrix

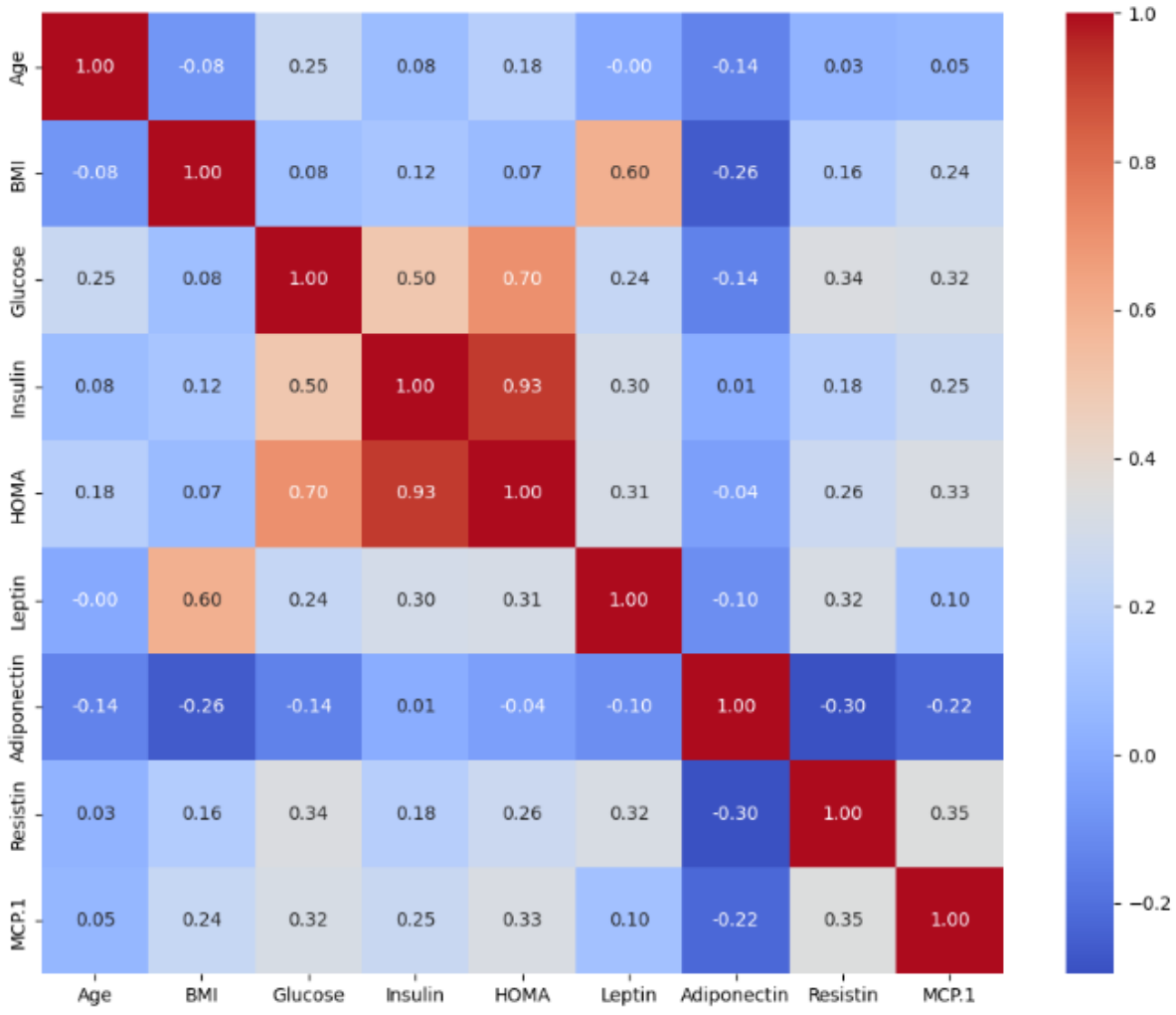


Figure 5: Correlation matrix (X_train)

The Pearson correlation matrix (cf 5) helps identify linear dependencies between continuous variables. Key findings include:

- A very strong correlation between *Insulin* and *HOMA* ($\rho \approx 0.93$), which is expected since HOMA is a deterministic function of insulin and glucose:

$$\text{HOMA} = \frac{\text{Glucose} \times \text{Insulin}}{405}$$

- Moderate correlations observed between:
  - *Glucose* and *HOMA* ($\rho \approx 0.70$),

  - *BMI* and *Leptin* ($\rho \approx 0.60$).
- Most other variable pairs exhibit relatively low correlations ($\rho < 0.4$), suggesting that the features are largely non-redundant and provide complementary information.

These observations justify the potential use of dimensionality reduction techniques (e.g., PCA) or regularization methods when using models that are sensitive to collinearity, such as logistic regression.

Coloured Scatter Matrix

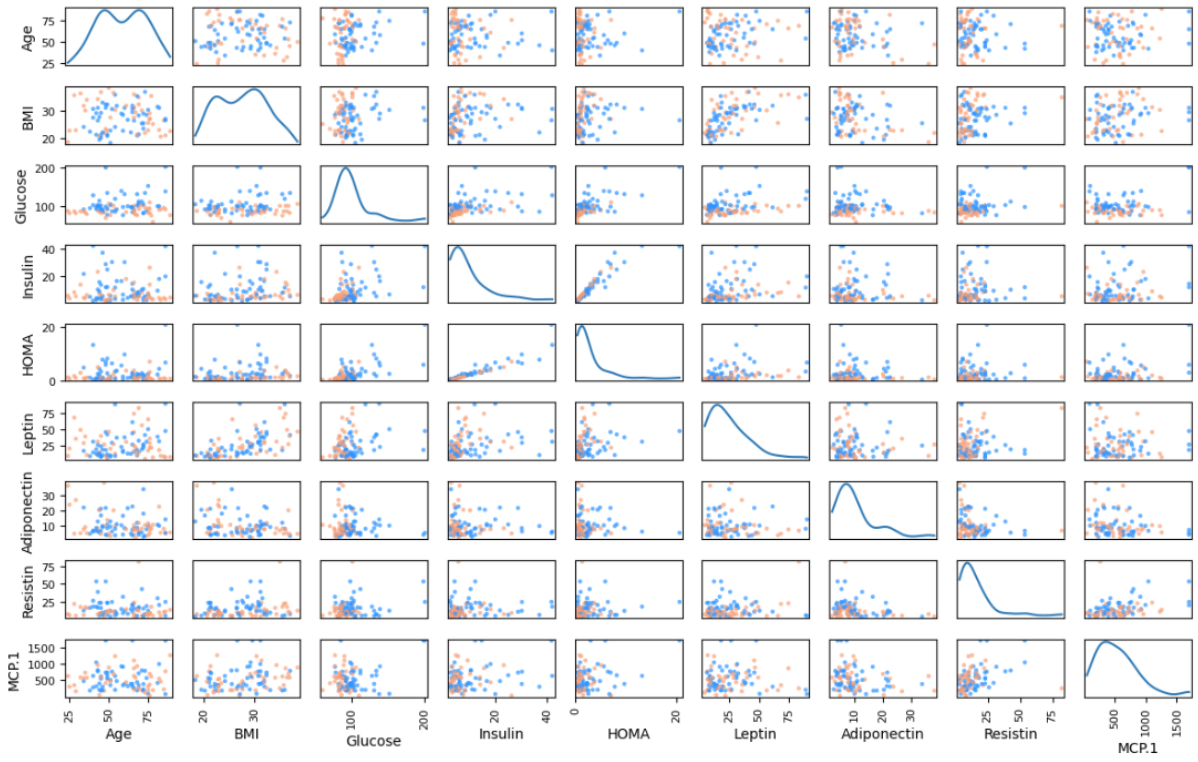


Figure 6: Colored scatter matrix (X_train)

The scatter plot matrix (cf 6), coloured by class (*Classification*), provides a global view of pairwise variable interactions and visual insight into class-specific groupings.

- Blue points represent patients with cancer (class 1), and red points represent healthy individuals (class 0).
- Significant overlap is observed between the two classes in most 2D projections.
- A few variable pairs such as (*Resistin*, *MCP.1*) or (*Adiponectin*, *HOMA*), suggest partial separation, though not sufficient for clear discrimination.

Conclusion of this step

The analysis reveals limited linear separability between the two classes based on individual or pairwise variables. While some features are correlated, most offer distinct contributions to the dataset.

These findings motivate the use of multivariate models capable of capturing complex or non-linear relationships. Moreover, given the heterogeneous scales and presence of strong correlations (e.g., *HOMA* and *Insulin*), standardization will be essential prior to modeling.

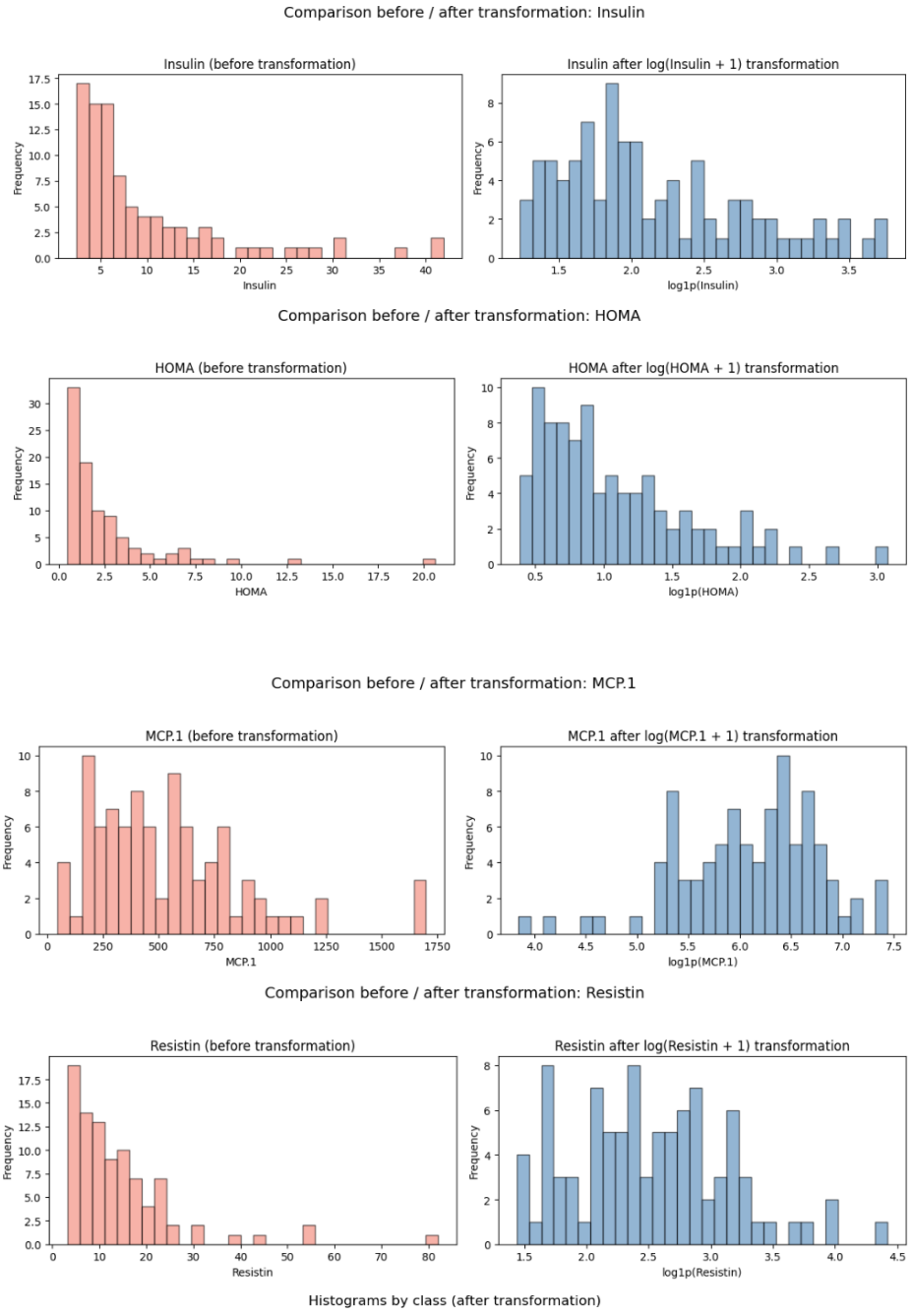
We now proceed to the data preprocessing stage, starting with standardization, before implementing and evaluating various supervised learning methods.

2.1.5 Logarithmic transformation

The initial exploratory analysis revealed strong right-skewed distributions for several variables, including *Insulin*, *HOMA*, *MCP.1*, and *Resistin*. These variables exhibited extreme values that could negatively affect the performance of certain machine learning models (logistic regression, k-NN), which are known to be sensitive to magnitude discrepancies and deviations from normality.

To make these distributions more symmetric and reduce the impact of extreme values, we applied a logarithmic transformation of the form $\log(x + 1)$ using the `np.log1p` function.

Before and After Transformation Comparison



For each transformed variable (cf 2.1.5), we compared the distribution before and after transformation using histograms.

- The long right tails were significantly reduced.
- The resulting distributions became more compact, less skewed, and in many cases closer to a Gaussian shape.
- This improves the homogeneity of the data set and facilitates subsequent standardization steps.

These visual comparisons clearly illustrate the benefit of the logarithmic transformation.

Dataset Update

The transformed variables (*Insulin_log*, *HOMA_log*, *MCP.1_log*, and *Resistin_log*) were added to the dataset, and the original versions were removed to prevent redundancy.

Effect of transformation on class-separated distributions

To evaluate the impact of the transformation on class-wise distributions (healthy / diseased), we reproduced several visualizations using the transformed features:

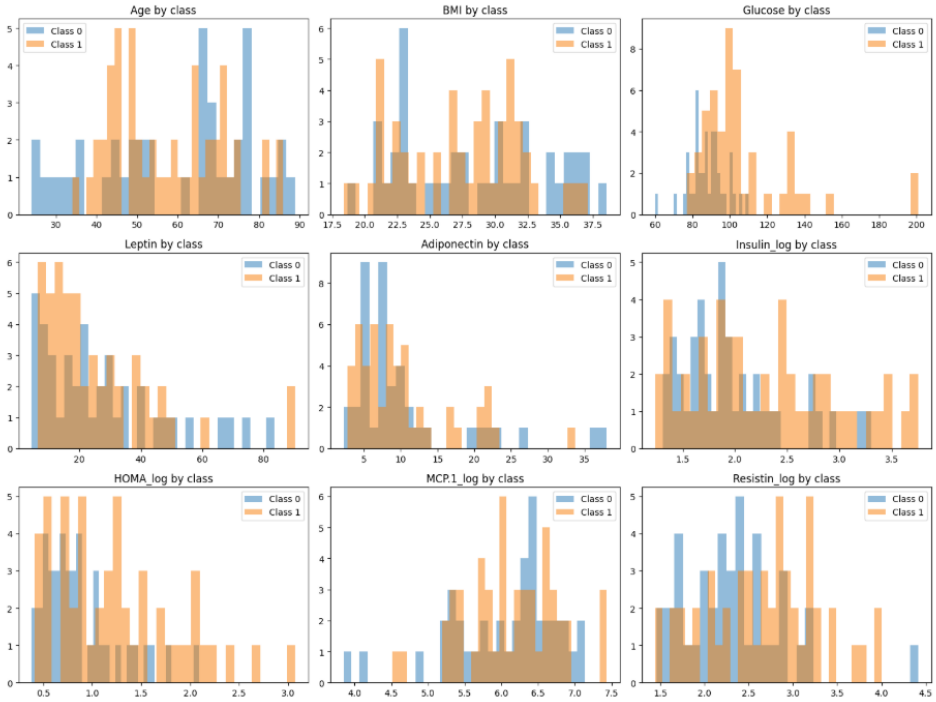


Figure 7: Histograms by class (after transformation)

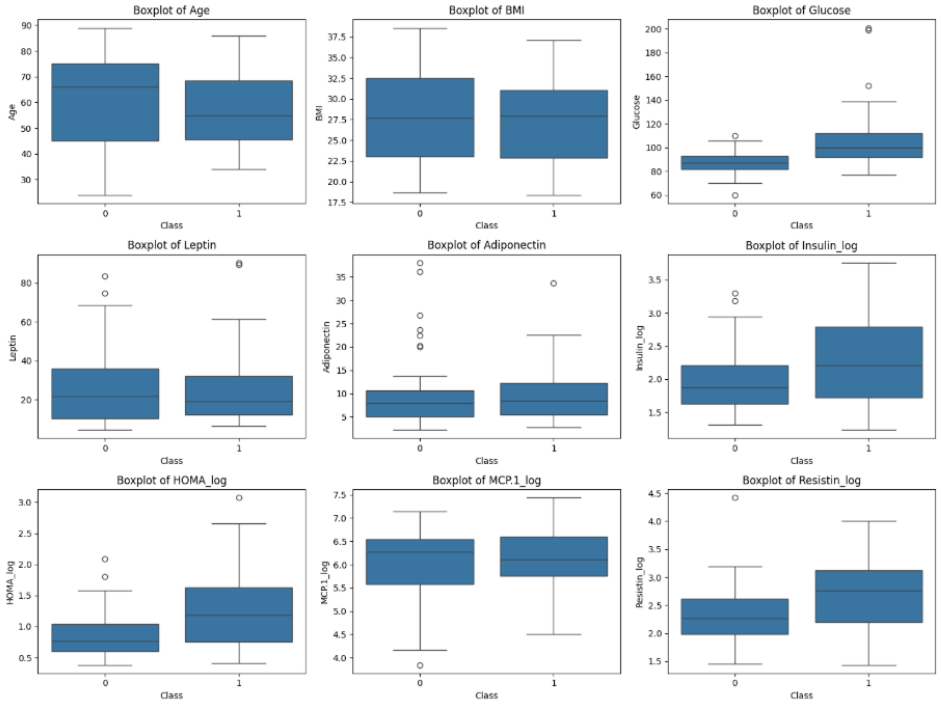


Figure 8: Boxplots by class (after transformation)

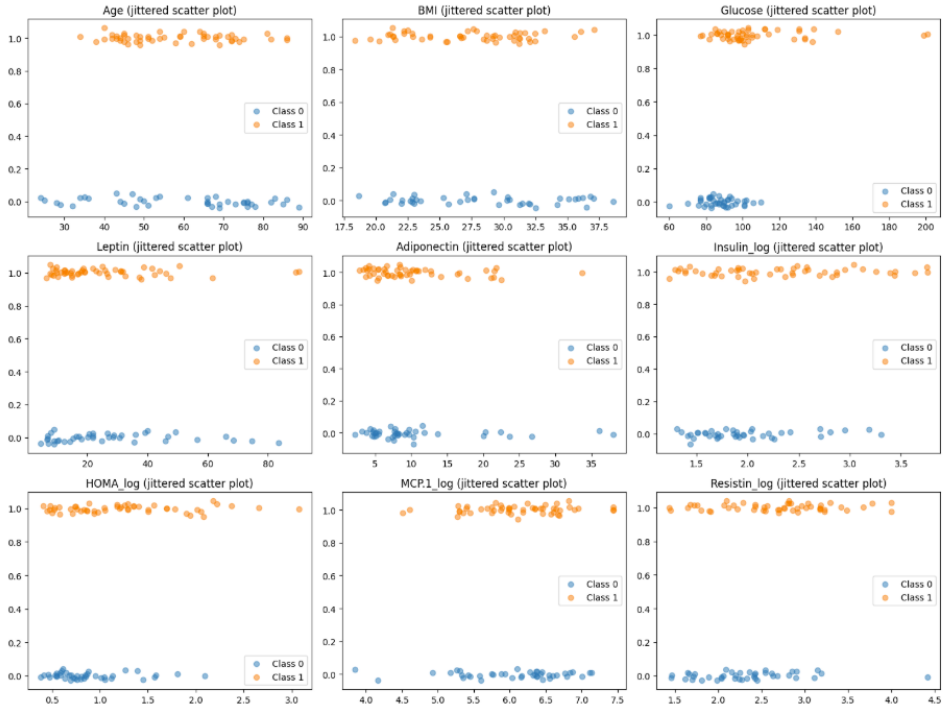


Figure 9: Histograms by class (after transformation)

• **Histograms by Class** These histograms (cf 7) display the relative density within each class after transformation. Variables such as *Insulin_log* and *Resistin_log* still show class-specific differences in density, but their distributions are much more regular.

• **Boxplots by Class** (cf 8) The logarithmic transformation clearly reduces the impact of outliers visible in the boxplots. Medians become more representative, and interquartile ranges are tighter and more stable.

• **Jittered Scatter Plots** These visualizations (cf 9) confirm that vertical class separation (0 / 1) is preserved, while the horizontal distribution becomes more controlled (an advantage for distance) based models.

Conclusion

The application of a logarithmic transformation has:

- Reduced the influence of extreme values,
- Stabilised variance across variables,
- Improved the geometric structure of the dataset for standardisation and learning purposes.

These transformations represent a key step in the preprocessing pipeline prior to model construction.

2.1.6 Standardisation (StandardScaler)

After applying a logarithmic transformation to the skewed variables, we performed standardization on all explanatory variables in the training dataset. The goal of this step is to bring all features onto the same scale by centering them around zero and scaling them to unit variance.

Standardization is particularly important for models that are sensitive to differences in scale or distance, including:

- logistic regression (for more stable convergence),
- k-nearest neighbors (k-NN),
- regularization methods (Ridge, Lasso).

Method

The transformation used is the classic Z-score standardization, defined as:

$$X^{(i)}_{\text{scaled}} = \frac{X^{(i)} - \mu^{(i)}}{\sigma^{(i)}}$$

where $\mu^{(i)}$ and $\sigma^{(i)}$ denote the mean and standard deviation of variable $X^{(i)}$ computed from the training set.

We used `StandardScaler` from `scikit-learn`, applying it only to the training data using `fit_transform`, in order to avoid data leakage.

Out[59]:

	mean	std
Age	-2.003229e-16	1.005479
BMI	-6.389816e-16	1.005479
Glucose	8.326673e-17	1.005479
Leptin	-5.068409e-17	1.005479
Adiponectin	8.839547e-17	1.005479
Insulin_log	-1.194697e-16	1.005479
HOMA_log	-2.256649e-16	1.005479
MCP.1_log	1.128324e-15	1.005479
Resistin_log	6.552729e-16	1.005479

2.1.7 Preparation of `X_test`

The transformations previously applied to the training set (logarithmic transformation + standardization) must be reproduced identically on the test set, without re-estimating any parameters.

Steps performed:

- Logarithmic transformation using $\log(x + 1)$ on the same variables: *Insulin*, *HOMA*, *MCP.1*, and *Resistin*;
- Removal of the original (raw) variables to avoid redundancy;
- Standardization using the same parameters (μ, σ) learned from `X_train`, applied via `scaler.transform()`.

We also verified that the columns in `X_test_scaled` exactly match those in `X_train_scaled` (both in names and order) to ensure compatibility with downstream learning models.

This step completes the preprocessing pipeline and allows us to move forward with supervised modeling.

2.2 Modeling with Logistic Regression

For this first modeling step, we train a simple logistic regression model **without explicit regularization**, on the preprocessed dataset. The preprocessing pipeline included a logarithmic transformation of skewed variables and a standardization of all features to ensure comparability in scale.

This method assumes a linear relationship between the log-odds of the target and the explanatory variables. Since the dataset contains no missing values, and a moderate number of observations ($n = 116$), logistic regression is a natural starting point.

Why this method?

- **Explainable linear model:** coefficients are interpretable and quantify the effect of each biomarker.
- **Well–suited to small datasets:** the model estimates only $d + 1$ parameters.
- **Compatible with regularization:** L1 (sparse selection) and L2 (shrinkage) will be explored later.

Results on the Test Set

After training the model on the standardized training set, we evaluated it on the test set ($n = 24$ observations). The results are as follows:

- **Recall:** 0.692
- **F1-score:** 0.750

Analysis

The results indicate a reasonable trade-off between **recall** and **precision**, which is particularly important in a medical context. The model correctly identifies both healthy and sick patients, but still **misses a few cancer cases** (false negatives), which is a critical concern.

This is reflected in the **F1-score** of 0.75, which captures this balance. Its use is especially appropriate given the **moderate class imbalance** (55.2% vs 44.8%), as relying solely on accuracy would be misleading in this context.

As this is a **non-regularized model**, it does not yet account for multicollinearity or perform automatic feature selection. These aspects will be addressed in the next step through **regularization and hyperparameter tuning**.

2.2.1 Cross-validation of the logistic regression model

To evaluate the **stability** of the logistic regression model and its **generalization capacity**, we perform a k-fold cross-validation (here, $k = 5$) on the training dataset. This procedure provides a more robust estimate of model performance by reducing sensitivity to the specific train-test split.

Evaluation protocol We monitor two key performance metrics:

- **F1-score**, which balances precision and recall, particularly suitable for imbalanced datasets;
- **Recall**, which measures the true positive rate, crucial in medical diagnosis where missing a positive case can be critical.

Cross-validation is performed using the `cross_val_score` function from `scikit-learn`, applied to the standardized and log-transformed training set `(X_train_scaled, y_train)`.

Cross-validation results

- **Mean F1-score:** 0.808 ± 0.071
- **Mean Recall:** 0.805 ± 0.084

These values provide a global estimate of the model's baseline performance prior to any regularization or tuning.

Analysis The observed standard deviations (≈ 0.07–0.08) reflect a **moderate variability** in performance between folds. This is expected due to:

- the **limited training sample size** ($n = 92$),
- the **imbalance in class distribution**,
- the **absence of regularization** in this baseline model.

These findings motivate the exploration of:

- Regularization techniques (L1 or L2),
- More complex models (e.g., nonlinear classifiers).

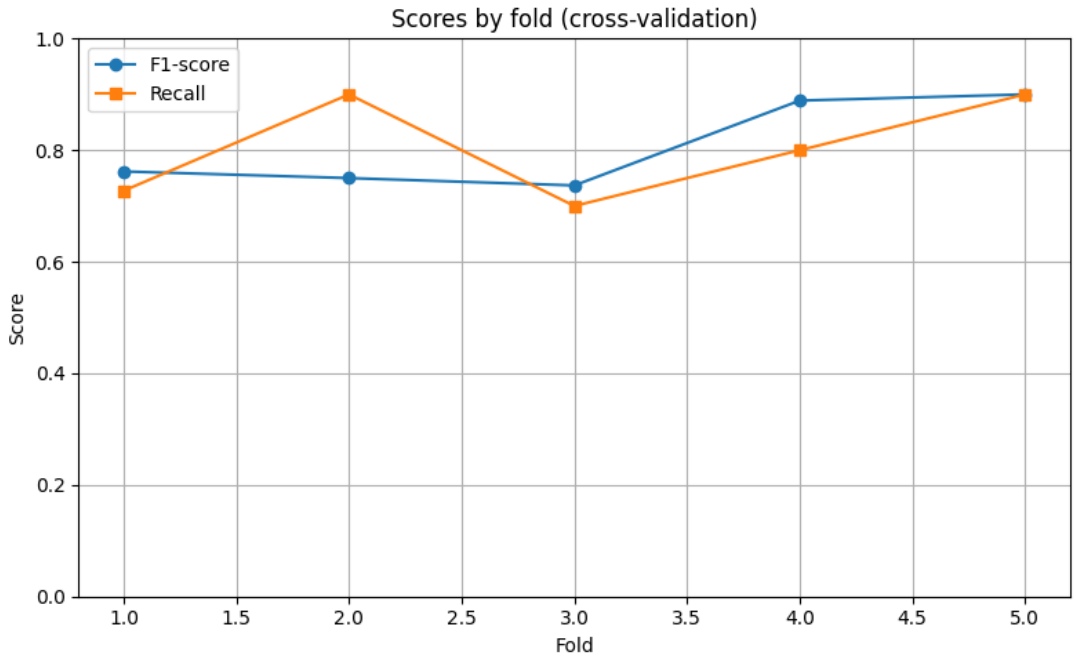


Figure 10: Scores by fold (F1-score and Recall) from 5-fold cross-validation

This figure displays the F1-score and Recall obtained on each fold. It confirms that:

- There is **no catastrophic failure** on any fold,
- Both metrics remain in acceptable ranges ($\approx$ 0.7–0.9), showing **reasonably consistent performance**.

This step establishes a reliable **baseline** for model performance before exploring improvements via **hyperparameter optimization** or **regularized variants**.

2.2.2 Hyperparameter Optimization (GridSearchCV)

To improve the generalization performance of the logistic regression model, we perform a grid search using `GridSearchCV` to identify the best combination of hyperparameters. The objective

is to find the optimal regularization strength (C) and penalty type (`l1` or `l2`) that balance underfitting and overfitting, while enhancing model interpretability and robustness.

Hyperparameter grid tested The following configurations were evaluated via 5-fold cross-validation on the training set:

- **Penalty type (`penalty`):**
  - `'l1'`: Lasso regularization, encouraging sparsity,
  - `'l2'`: Ridge regularization, penalizing large coefficients.
- **Regularization strength (C):** 10 values sampled on a logarithmic scale from 10^{-2} to 10^{4}. Smaller C implies stronger regularization; larger C reduces penalty.
- **Solver:** `'liblinear'`, which is compatible with both `'l1'` and `'l2'` penalties in `scikit-learn`.

The **F1-score** was used as the selection criterion, as it balances precision and recall, which is especially relevant in the presence of class imbalance.

Grid search results

- **Best hyperparameters:** `{'C': 4.64, 'penalty': 'l2', 'solver': 'liblinear'}`
- **Best mean F1-score (5-fold CV):** 0.814

Interpretation The selected model uses a **moderate L2 regularization** ($C \approx 4.64$), which limits overfitting while retaining all features. This results in a **dense model** that preserves full interpretability and avoids zeroing out coefficients. Importantly, the fact that the optimal C lies well within the search range (and not on the boundaries) confirms that the model **benefits from controlled regularization**.

This tuning step provides a meaningful improvement over the baseline model trained without hyperparameter optimization.

Expected impact The optimized model will next be evaluated on the test set to determine whether the observed gain in cross-validation translates into a **real generalization improvement**. If confirmed, this would validate the value of **hyperparameter tuning**, even for interpretable linear models such as logistic regression.

2.2.3 Evaluation of the Optimized Logistic Regression Model

After selecting the best logistic regression model through cross-validation, we now assess its generalization performance on the **independent test set**, which remained untouched during training and hyperparameter tuning.

Objective

The goal is to evaluate whether the optimized model, after regularization (L1 or L2), improves over the baseline logistic regression in terms of:

- **Recall** : sensitivity to detecting cancer patients,
- **F1-score** : balance between recall and precision,
- **Discriminative ability** : assessed via the ROC curve and AUC.

Performance on the Test Set

- **Recall:** 0.692

- **F1-score:** 0.750

These values confirm the model's ability to balance sensitivity and precision, despite a moderate class imbalance. No significant gain is observed compared to the baseline model. The overall test results remain **comparable to those of the baseline model**, suggesting that regularization had **limited impact** in this context, possibly due to the small dataset size or the already good calibration of the initial model.

Confusion Matrix

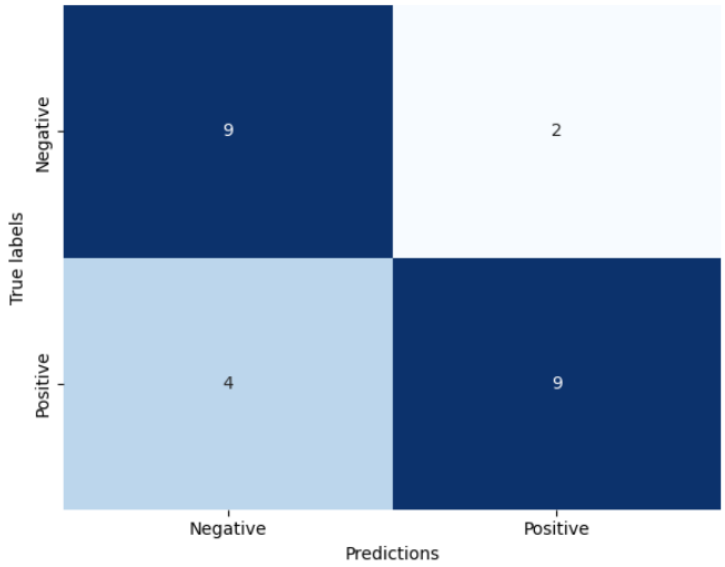


Figure 11: Confusion matrix of the optimized model on the test set

- 9 true positives and 9 true negatives were correctly identified,
- 4 cancer cases were missed (false negatives), which is a **critical risk in medical diagnosis**.

ROC Curve and AUC

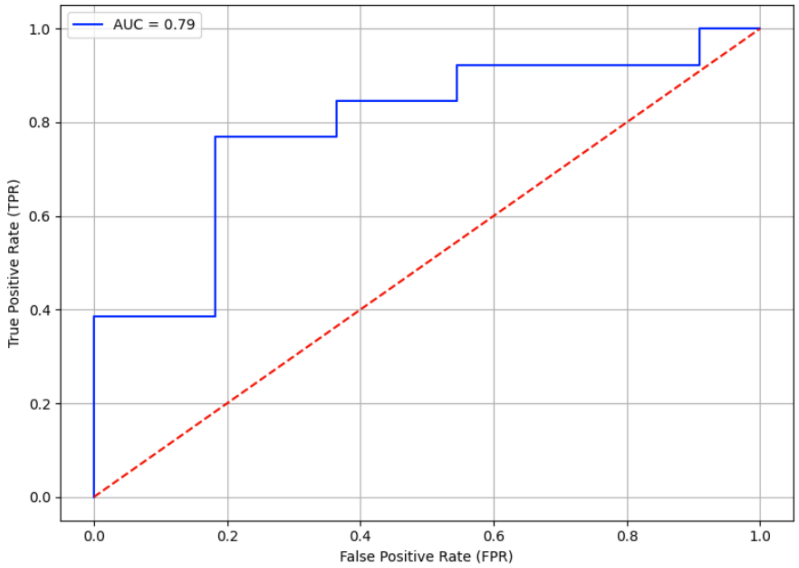


Figure 12: ROC curve – Optimized logistic regression

The model achieves an **AUC** (Area Under the Curve) of **0.79**, indicating good discriminative ability. On average, the model assigns a higher probability to patients with cancer than to healthy patients in 79% of the cases. The ROC curve lies significantly above the diagonal, confirming the model's utility for clinical decision-making.

Conclusion

The optimized model demonstrates a **robust trade-off between performance and interpretability**:

- **L2 regularization** reduces overfitting by penalizing large coefficients,
- **Good test performance**: F1-score = 0.75, AUC = 0.79,
- **Balanced recall and precision**,
- **Interpretable coefficients** valuable in medical applications.

However, performance remains comparable to the unregularized model. This suggests that **regularization had limited marginal benefit** in this setting, likely due to the **small sample size**. More complex or non-linear models (e.g., SVMs or tree-based methods) may be required to improve predictive accuracy.

2.2.4 Comparison of Logistic Regression Variants

We now compare the two logistic regression models developed previously:

- A **simple model**, trained without hyperparameter tuning (default `penalty='l2'`, `C=1.0`);
- An **optimized model**, trained via cross-validation using `L2` regularization and `C = 4.64`.

The comparison includes: test set performance, ROC curves, learned coefficients, predicted probability distributions, and individual-level error analysis.

Overall Performance on the Test Set

Model	Recall	F1-score	AUC
Simple LogReg	0.692	0.750	0.783
Optimized LogReg (L2, C=4.64)	0.692	0.750	0.790

Both models achieve **identical recall and F1-score**, confirming a similar balance between sensitivity and precision. The optimized model exhibits a **slightly higher AUC** (+0.007), indicating a marginal gain in discriminative ability.

ROC Curve Comparison

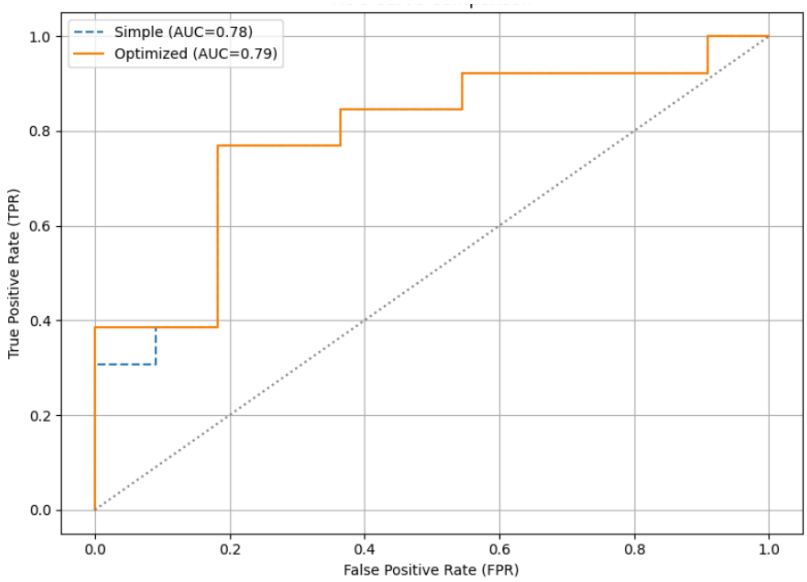


Figure 13: ROC Curve comparison: simple vs optimized model

As shown in Figure 13, the ROC curves are nearly identical. The optimized model reaches an AUC of 0.790 compared to 0.783 for the baseline model, confirming a minor but consistent improvement.

Coefficient Comparison

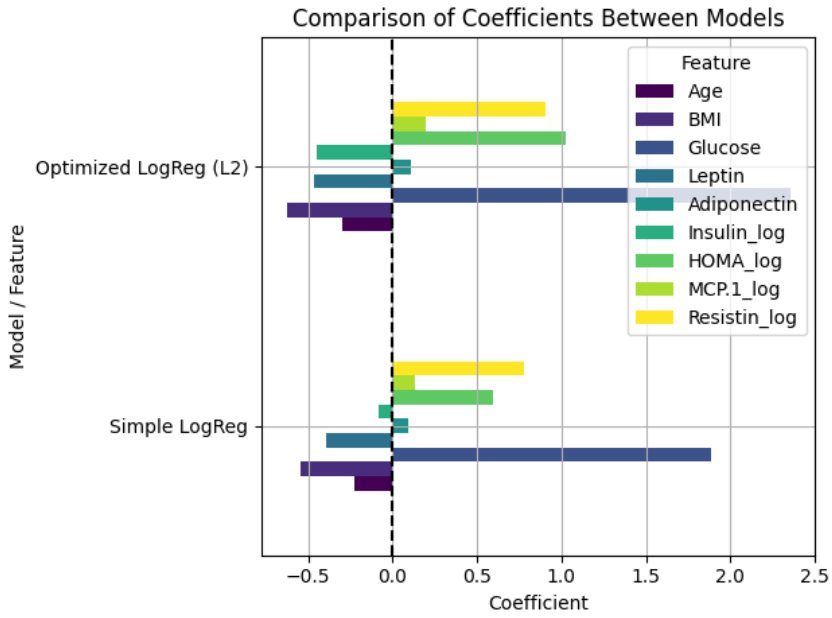


Figure 14: Comparison of learned coefficients

Figure 14 shows that L2 regularization **shrinks** the coefficients without enforcing sparsity:

- `Insulin_log` and `HOMA_log` are slightly reduced but not eliminated;
- `Glucose` and `BMI` remain strong predictors in both models.

This reflects the role of L2 regularization in promoting **model stability** without discarding features.

Predicted Probability Distributions

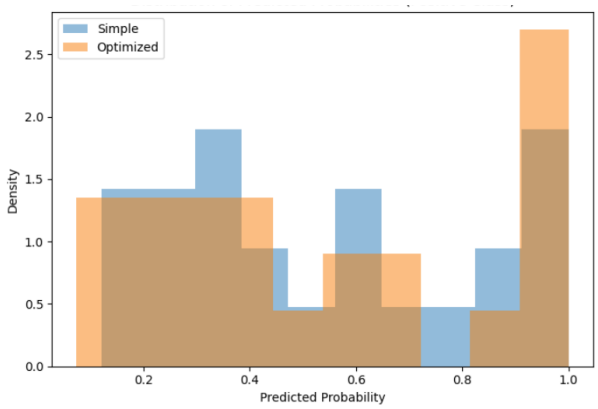


Figure 15: Distribution of predicted probabilities (class 1)

As illustrated in Figure 15, the distributions are broadly similar. However, the optimized model yields **more polarized predictions** (closer to 0 or 1), suggesting increased confidence in classification decisions.

Error Analysis by Case

Index	True Label	Simple LogReg	Optimized LogReg
111	1	0	0
104	1	0	0
95	1	0	0
38	0	1	1
7	0	1	1
58	1	0	0

Table 3: Examples consistently misclassified by both models

Most predictions are consistent across both models. The examples above are misclassified identically by both classifiers. This suggests that the remaining errors are not due to model instability or overfitting, but likely reflect ambiguous cases or class overlap in the feature space. Such consistency strengthens confidence in the stability of both models.

Final Remarks

Model	Key Strength
Simple LogReg	Simpler, slightly faster to train
Optimized LogReg (L2)	Greater robustness, improved generalization

- Both models yield **equivalent predictive accuracy** on the test set (same recall and F1-score).
- The **L2-regularized model** is preferable when stability, generalization, or coefficient shrinkage is desired, especially in the presence of multicollinearity.
- Although the AUC gain is modest (+0.007), the optimized model confirms the benefit of **regularization and hyperparameter tuning**.
- The limited gain observed here may stem from the **small dataset size** or the already strong calibration of the baseline model.

2.3 Modeling with Gaussian Naive Bayes

Why this method?

- **Very low variance**: the estimators μ_j, σ_j are stable even on a small sample, so the model hardly overfits.
- **Instant training & reference quality**: provides a quick yard-stick score.
- **Clinically understandable**: class-wise means and variances have direct biomedical meaning.

The data of the features is continuous and non-binary. As such, we use a GaussianNB, the performance will nevertheless suffer as the features aren't all Normal distributed and the dimension is rather small, we cannot suppose normal distribution through size. As with previous models, the data is first transformed using a logarithmic function, followed by standardization, to ensure consistency in preprocessing. But here we do a standard 70/30 split on the data.

Model evaluation

Because of its simplicity and without a real fit in law, the GaussianNB is unlikely to produce good results. In order to maximise the performance, we want to know if a logarithmic transformation is useful.

2.3.1 Model Evaluation with logarithmic transformation

A stratified split was used to preserve the original proportion of healthy and sick patients in both the training and test sets. This ensures that the model is not trained with a dominant class, which could bias the learning process.

The resulting dataset sizes are:

- **Training set:** 81 observations
- **Test set:** 35 observations

This corresponds to a standard 70/30 split.

Confusion Matrix with logarithmic transformation

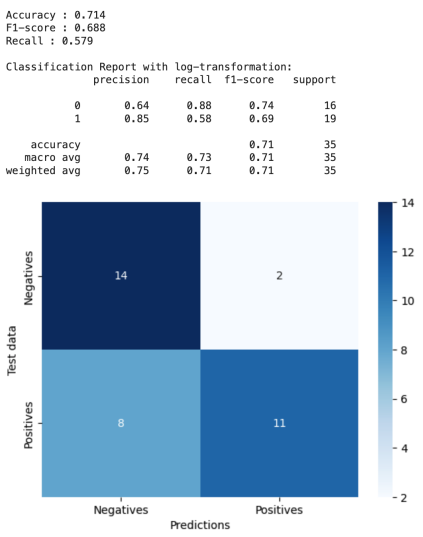


Figure 16: Confusion Matrix with transformation

2.3.2 Model Evaluation without logarithmic transformation

Confusion Matrix without logarithmic transformation

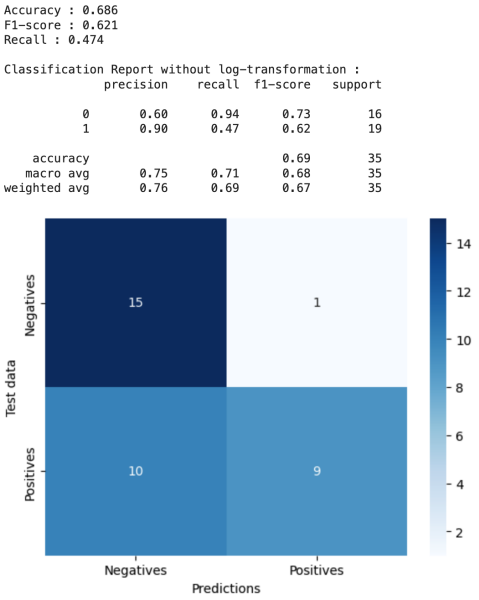


Figure 17: Confusion Matrix without transformation

We want to maximise the recall, even if it might compromise the accuracy: it is more important to find every sick patient than every healthy one. In our case, the model doesn't mark enough people as sick: only 11 out of the 19 sick patients are identified as such (58%). (cf 17). However, without the logarithmic transformation, the model performance is downright horrible: the recall is worse than tossing a coin (47.4%). This illustrates the impact the logarithmic transformation of the data can have on model performance.

This gives us the following ROC-curve:

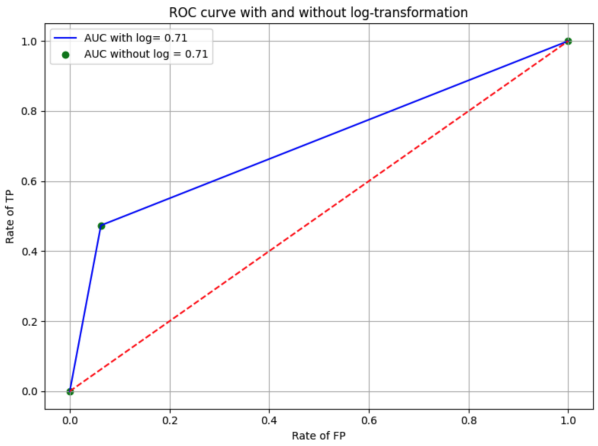


Figure 18: ROC curve

2.4 Modeling with k-Nearest Neighbours

Why this method?

- **Non-parametric**: makes no functional assumption; if the biomarkers naturally form "clouds" of healthy vs. cancer cases, k-NN will pick them up.
- **Small data set friendly**: with $N = 116$ and $d = 9$, storing all points and computing distances is inexpensive.
- **Interpretable**: the clinician can be shown *which* similar patients motivated the prediction.
- **Baseline**: if a sophisticated model cannot beat k-NN, preprocessing or feature engineering must be revisited.

Here there is no assumptions about the underlying data distribution so or the Data processing and analysis we just check the missing values and change the label on the target variable. We also do a train/test split 70%/30% and a stadardisation: Feature scaling is important for k-NN, as it is a distance-based algorithm and is sensitive to the scale of the features. This means that if some features have significantly larger ranges than others, they can dominate the distance computation and negatively affect the quality of the predictions. Therefore, scaling ensures that all features contribute equally to the distance metric.

No curse of dimension

- Number of samples: 116
- Number of features: $d = 9$
- Number of classes: 2

Therefore, the number of features d is small enough to ensure that we are not affected by the curse of dimensionality.

2.4.1 Model construction

Cross-Validation to find the optimize K nearest neighbor

In k-NN classification, to achieve the best prediction performance, we need to find the optimal number of neighbors k that maximizes the evaluation score of our models. Here, we use the `f1_score` from `sklearn.metrics`, as it provides a good balance between precision (e.g., correctly predicting a sick patient as sick, or a healthy patient as healthy) and recall (e.g., correctly identifying sick patients among all those predicted as sick).

To determine this hyperparameter, we apply a 5-fold cross-validation strategy. We chose 5 folds instead of 10 due to the limited amount of data, as this provides a better balance between the sizes of the training and validation sets.

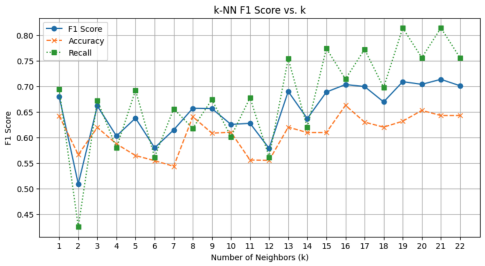


Figure 19: Cross validation

After cross-validation, we find that the optimal number of neighbors is $k = 23$. (cf 19)

2.4.2 Model Evaluation

Confusion Matrix

```
Accuracy: 0.7916666666666666
F1 Score: 0.7883597883597883
Recall: 0.9166666666666666
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.67      0.76        12
           1       0.73      0.92      0.81        12

    accuracy                           0.79        24
   macro avg       0.81      0.79      0.79        24
weighted avg       0.81      0.79      0.79        24
```

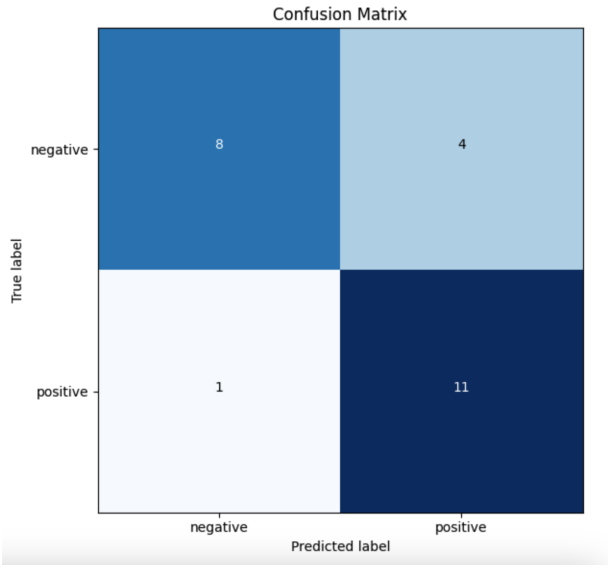


Figure 20: Confusion Matrix

Receiver Operating Characteristic

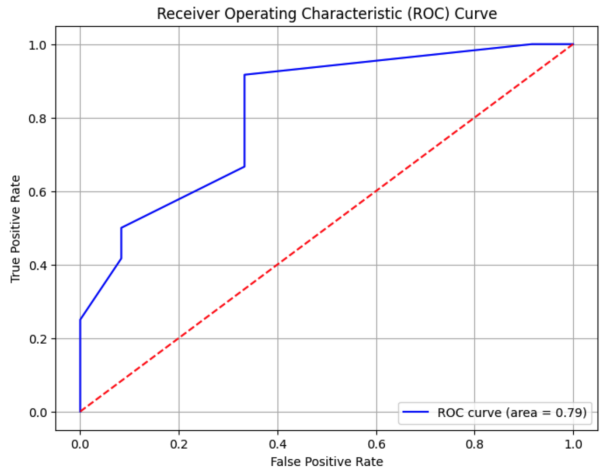


Figure 21: ROC Curve

In this optimized k-NN classification, we aim to maximize recall while maintaining good accuracy, in order to minimize the number of misclassifications, particularly cases where a sick patient is incorrectly predicted as healthy (cf 20). This objective is crucial in medical diagnosis, where false negatives can have serious consequences.

We achieve this goal with a recall of 91.6% and an accuracy of 79.1% (cf 21).

2.5 Modeling with Shallow Neural Network (MLP)

Why this method ?

- **Captures non-linearities**: can learn interactions such as *Insulin* × *BMI* or threshold effects.
- **Compact architecture**: 16-8-1 (< 500 weights) is trainable without over-parametrising this small data set.
- **Performance ceiling check**: if the MLP fails to improve on logistic regression, a linear model is sufficient.

For the data processing and analysis phase, we assume that all features are statistically independent of one another. Although this assumption is rarely strictly true in practice, it simplifies the modeling process and is commonly made in many machine learning pipelines.

We begin by loading the dataset and performing an exploratory data analysis (EDA) to examine the distribution of each variable, detect any missing values, and understand the data types involved. This step helps assess data quality and informs the choice of preprocessing techniques.

Next, we preprocess the target variable by converting its categorical labels into binary numerical values, which is necessary for compatibility with classification algorithms.

Finally, we apply feature scaling through standardization to ensure that all variables contribute equally to distance-based algorithms (e.g., k-NN) and to facilitate faster and more stable training of neural networks. Standardization transforms the features to have zero mean and unit variance.

Stratified K Fold

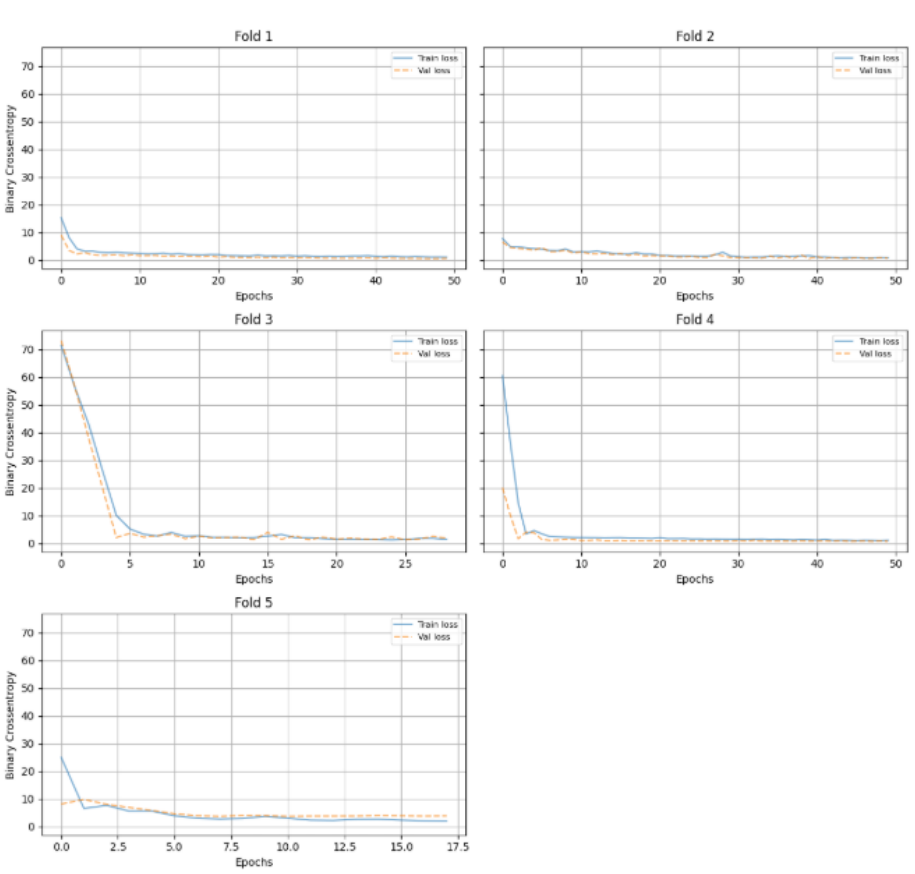


Figure 22: Loss evolution across each fold

In the context of binary classification using the Breast Cancer Coimbra dataset, preserving the class distribution when splitting the data is a crucial condition for ensuring the validity of the experimental results.

Use of `stratify=y` in `train_test_split`. When splitting the dataset into training and test sets, we use the `train_test_split` function from the scikit-learn library. To ensure that the proportions of the target classes are preserved in both subsets, the `stratify=y` argument is specified.

This precaution is particularly important in cases of imbalanced datasets, as is the case here, where the two classes of the target variable ("Classification") are not equally represented. Simple random sampling could lead to a significant imbalance in the test set, making performance metrics unreliable and potentially biasing the model toward the majority class. The `stratify=y` option therefore ensures the statistical representativeness of both classes in each subset.

Use of `StratifiedKFold` for cross-validation. Similarly, during model evaluation by cross-validation, we use the `StratifiedKFold` method. Unlike standard `KFold`, this method maintains the class distribution within each of the k folds.

The goal is to obtain a more robust and stable estimate of model performance, particularly in the presence of class imbalance. Preserving the dataset's original structure in each fold reduces the risk of overfitting or underfitting due to folds dominated by a single class.

Statistical Justification. Maintaining class distribution during sampling procedures is a standard requirement in statistics, grounded in the principle of sample representativeness. In supervised classification, the systematic use of stratified methods enhances the external validity of results (i.e., the model's generalization capability) while reducing the variance of performance estimates obtained through cross-validation.

2.5.1 Model Construction

	precision	recall	f1-score	support
Healthy	0.50	0.09	0.15	11
Cancer	0.55	0.92	0.69	13
accuracy			0.54	24
macro avg	0.52	0.51	0.42	24
weighted avg	0.52	0.54	0.44	24

Neural Network Architecture

We chose to build a shallow neural network composed of two hidden layers. This architecture provides sufficient capacity to model moderately complex relationships in the data, while limiting the risk of overfitting associated with deeper networks.

The `ReLU` (Rectified Linear Unit) activation function is applied to the hidden layers to introduce non-linearity and enable the network to learn complex patterns. For the output layer, we use the `sigmoid` activation function, which is suitable for binary classification tasks as it produces outputs in the range $[0, 1]$, interpretable as class probabilities.

To reduce the risk of overfitting, we apply L2 regularization (also known as weight decay) to the dense layers. This technique penalizes large weights during training, encouraging the model to learn simpler and more generalizable solutions.

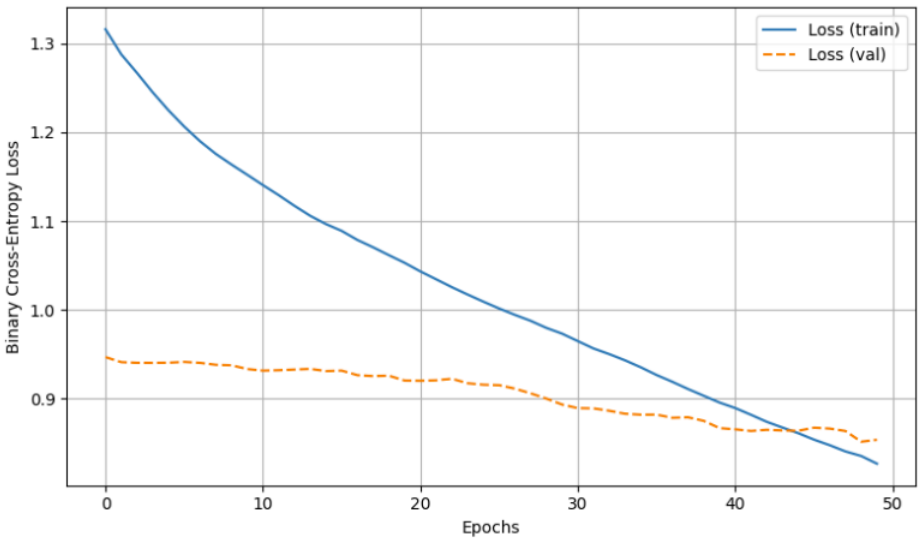


Figure 23: Learning curve

2.5.2 Model Evaluation

Model Performance and Training Behavior

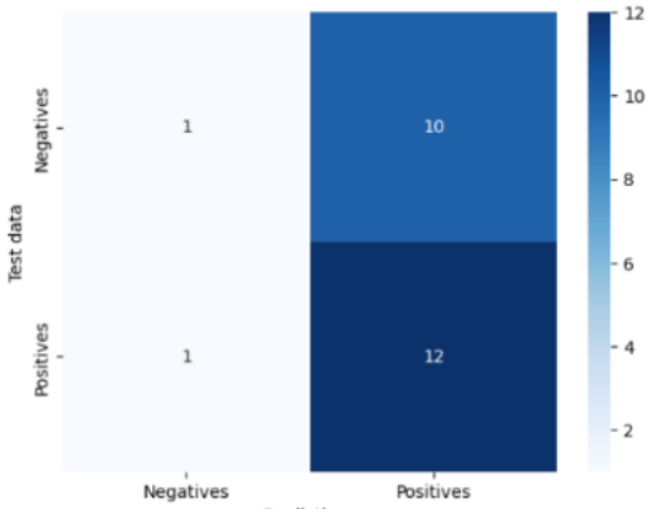


Figure 24: Confusion Matrix

When training the neural network on the Breast Cancer Coimbra dataset, the model achieved an F1-score of 0.69, and more importantly a recall of 0.92. However, we recognise that the model has a clear tendency to mark patients as sick, thus a precision of merely 0.55. A notable behavior observed during training is that the validation loss (`val_loss`) mostly remains lower than the training loss (`train_loss`) (cf 23). This phenomenon is primarily due to the use of L2 regularization, which penalizes the model's weights during training but not during validation, leading to artificially higher training loss values. Additionally, the small size of the dataset, the application of class weighting to address class imbalance, and the use of early stopping may contribute to this gap.

Such behavior is not problematic as long as validation performance remains stable and satisfactory, which is the case here. The F1-score confirms that the model prioritizes recall—an essential metric in medical diagnosis, where minimizing false negatives is critical.

3 Conclusion

3.1 Comparison Criterion: Mathematical Formula

For a rigorous comparison of the classification methods, we rely on the following formulas for the positive class:

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics are particularly well-suited for a medical context where:

- **Recall** (sensitivity) must be maximized to avoid missing actual diseased patients.
- **Precision** must remain reasonably high to limit unnecessary follow-up tests on healthy individuals. However, in this context, false positives are acceptable, because it's better to identify a potentially ill patient even if the medical test ultimately comes back negative.
- The **F1-score** balances both, giving a single performance measure.

3.2 Results

The table below summarizes Precision, Recall, and F1-Score (positive class) for each method on the same test set (24 observations):

- **Logistic Regression (L2)**
  - Precision: 0.818 (9 TP, 2 FP)
  - Recall: 0.692 (9 TP, 4 FN)
  - F1-Score: 0.750
- **Gaussian Naïve Bayes**
  - Precision: 0.846 (11 TP, 2 FP)
  - Recall: 0.579 (11 TP, 8 FN)
  - F1-Score: 0.68
- **k-Nearest Neighbors (k = 23)**
  - Precision: 0.73 (11 TP, 4 FP)
  - Recall: 0.916 (11 TP, 1 FN)
  - F1-Score: 0.788
- **Shallow Neural Network (MLP)**
  - Precision: 0.55 (12 TP, 10 FP)
  - Recall: 0.92 (12 TP, 1, FN)
  - F1-Score: 0.69

3.3 Comparison of the Four Methods

Considering only recall and F1-score:

- **k-NN (k = 23)** achieves the highest recall (0.920) and the highest F1-score (0.788). It misses only one diseased patient (1 false negative out of 13), making it ideal for maximizing detection of positive cases.
- **Gaussian Naïve Bayes** is the worst of the tested models and doesn't perform well, neither when it comes to recall nor when it comes to F1-score. The precision is good but less relevant medically than the recall.
- **Logistic Regression (L2)** has an F1-score of 0.750, yet its recall (approximately 0.69) is substantially lower. It correctly identifies 9 diseased patients out of 13, placing him behind k-NN and MLP in terms of recall.
- **MLP** has an F1-score of 0.69,and an impressive 0.92. It correctly identifies all the diseased patients except one, but but it labels lots of healthy patients as sick patients, placing him behind Naïve Bayes, Logistic regression and k-NN as last place in terms of precision.

Overall, if one accepts a high number of false positives and wants primarily a **high recall** with a **solid F1-score**, **k-NN (k = 23)** is the most appropriate model on this dataset.

3.4 Conclusion for the Clinic

Under the assumption that false positives are acceptable in favor of maximizing recall and F1-score:

- **k-NN (k = 23):**
  - Recall of 0.92, only one diseased patient missed out of 12.
  - F1-score of 0.788, the best recall-F1 balance.

  This model should be chosen if capturing nearly all positive cases is the primary goal, even at the cost of many healthy patients being falsely flagged.
- **MLP:**
  - Recall around 0.92, 1 diseased patient missed.
  - F1-score of 0.69, the model has a tendency to mark every subject as positive. Not a good compromise from a performance level.
- **Logistic Regression (L2):**
  - Recall around 0.69, the model misses 4 diseased patients out of 13.
  - F1-score of 0.750, acceptable but lower than k-NN.
- **Gaussian Naïve Bayes:**
  - Recall of 0.579, eight diseased patients missed.
  - F1-score of 0.68, inferior to k-NN in terms of F1.

  Logistic regression and NB are less suitable when the main objective is to maximize detection, since their recall is too low compared to k-NN and MLP.

In conclusion, under the scenario where false positives are negligible compared to the need to minimize missed diagnoses, **k-NN (k = 23)** stands out as the optimal choice, ensuring a recall of 0.920 and an F1-score of 0.788.“‘

3.5 Improvement

Even though k-NN appears optimal under our “recall and F1-priority” scenario, several avenues can further enhance sensitivity and F1-score: one could explore Advanced Non-Parametric Methods, such as for example SVM (Support Vector Machine)