Evaluating Model Accuracy and Interpreting Variable Selection in Predicting Malignant Breast Tumors

Stephanie Daniella Hernandez Prado

December 2025

1 Introduction

Early detection of breast cancer is crucial to improve clinical outcomes and reduce mortality. In this study, we analyzed a dataset composed of morphological measurements obtained from digitized images of breast cells, with the aim of predicting whether a sample corresponds to a malignant or benign tumor. Due to the high dimensionality of the dataset and the strong correlations between many of the variables, it is necessary to carefully evaluate different modeling methods to determine which offer the best predictive performance and greatest stability.

The main objective of this work is to compare the performance of different classification models: Logistic Regression, Stepwise Selection, Backward Selection, Forward Selection, LASSO, and Ridge, with the intention of comparing them to a Random Forest machine learning model. To make the comparison as stable and comparable as possible, we will use a 5-fold Cross-Validation procedure, performing it 50 times with the same 5 folds for each model in each iteration.

Performance evaluation is performed by comparing the averages of the cross-validation Accuracy, which measures the percentage of correct classifications, and the Brier Score, which quantifies the confidence with which a model makes a prediction. We will also explore why models like LASSO choose certain variables to synthesize information and achieve simpler models in the presence of multicollinearity.

Finally, we will seek to identify the most accurate, stable and appropriate model for each situation to predict tumor diagnosis.

2 Dataset

The used dataset is the Breast Cancer Wisconsin (Diagnostic) Dataset [1], which is a widely used machine learning dataset containing 569 samples from fine needle aspirate (FNA) images of breast masses, collected at the University of Wisconsin Hospitals by Dr. William H. Wolberg. The dataset includes 30 numerical features derived from digitized cell nuclei images, where each of 10 cell characteristics (radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension) is measured three ways: mean, standard error, and ”worst” (mean of three largest values). Each sample is labeled as either malignant (212 cases, 37.3%) or benign (357 cases, 62.7%), making it a binary classification problem.

3 Methodology

3.1 Models

In this study, several algorithms were used to compare their predictive capacity and stability, measuring their performance under the same cross-validation conditions repeated multiple times. Each model was trained using the same folds during the same iteration, ensuring a fair comparison.
The models used are listed below along with a brief description:

3.1.1 Logistic Regression

Logistic regression is a type of model used to predict categorical variables based on a combination of predictor variables. This model uses the logit function as its link function, allowing us to estimate the probabilities of each possible outcome modeled as a function of the explanatory variables using the logistic function. This model is easy to understand, effective, and widely known, so we will use it as our base model.

Logistic regression looks like this:

$$logit(p_i) = ln(\frac{p_i}{1-p_i}) = \beta_0 + \sum_{j=1}^{k} \beta_j x_{j,i}$$

where x_i is the probability of success of the i-th observation, while $X_{j,i}$ is the j-th predictor variable of the i-th observation.

3.1.2 Stepwise

This model is a version of the previous model where we remove variables, add variables, or both actions based on their significance with the goal of identifying a set of predictors that improve the fit without overfitting.

The Forward method starts with a model containing only the intercept and adds variables with the smallest p-value, until no variables can be added due to having too large a p-value. The Backward method starts with the entire model and removes variables using the same criteria. The Both method combines both approaches.

3.1.3 LASSO

This is an extension of logistic regression, where an L1 constraint is incorporated into the size of the coefficients. This constraint favors simpler models and eliminates irrelevant predictors. The regularization parameter λ chosen was the one that performed best during cross-validation, giving us a balanced penalty parameter.

The parameters of the model applying LASSO penalty are found as follows:

$$\hat{\beta}_{LASSO} = \text{arg min}_\beta \{-logLikelihood(\beta) + \lambda \textstyle\sum_{j=i}^{k} |\beta_j|\}$$

where λ is the regularization parameter, which controls how much the coefficients are penalized.

3.1.4 Ridge

A model similar to Lasso, but using an L2 penalty that reduces the magnitude of the coefficients without making them exactly zero. It tends to work better when many predictors are correlated.

The parameters of the model applying Ridge penalty are found as follows:

$$\hat{\beta}_{Ridge} = \text{arg min}_\beta \{-logLikelihood(\beta) + \lambda \textstyle\sum_{j=i}^{k} \beta_j^2\}$$

3.1.5 Random Forest

Decision trees work by recursively splitting the dataset based on the values of the predictor variables. At each node, the variable and split point that best separate the classes are selected. The process continues until a stopping criterion is reached, either because the maximum depth of the tree has been reached or because the observations within a node are sufficiently homogeneous. At the end, each leaf of the tree assigns a predicted class based on the observations it contains.

A Random Forest involves training many decision trees, each constructed from a subset of observations obtained through bootstrap sampling and using only a random subset of variables at each split. Each tree produces its own prediction, and the final model response is obtained through majority voting or averaging. This approach reduces variance and improves the model's generalizability compared to a single tree.

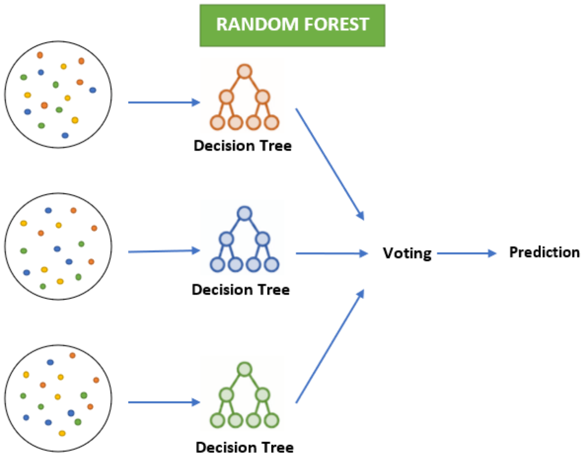


Figure 1: Diagram of the 5-Fold Cross-Validation process (Illustration of the Random Forest algorithm (Obtained from:[2]).

3.2 Metrics

The performance of the models was evaluated using the following metrics:

3.2.1 Accuracy

Accuracy is the proportion of correct predictions out of the total number of observations.

$$Accurracy = \frac{\text{Correct predictions}}{\text{Total predictions}}$$

3.2.2 Brier Score

Brier Score is the mean squared error of the forecast. It tells us how confidently the model predicts the response variable in each observation.

$$\text{Brier Score} = \frac{1}{N}\sum_{i=1}^{N}(p_i = y_i)^2$$

Where:
p_i: Predicted probability of the positive class.
y_i: Actual value encoded as 0 or 1.

3.3 K-Fold Repeated Cross Validation.

To obtain a robust evaluation, each model was trained and evaluated using a process that we will call K-Fold Repeated Cross Validation

In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples, often referred to as ”folds”. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k - 1 subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data.

In our case, we used 5 folds and took the average performance (Accuracy and Brier Score) of these 5 folds for each model, then repeated this process 50 times. This allows for obtaining performance distributions, comparing variability, and avoiding conclusions based on a single partition of the data set.

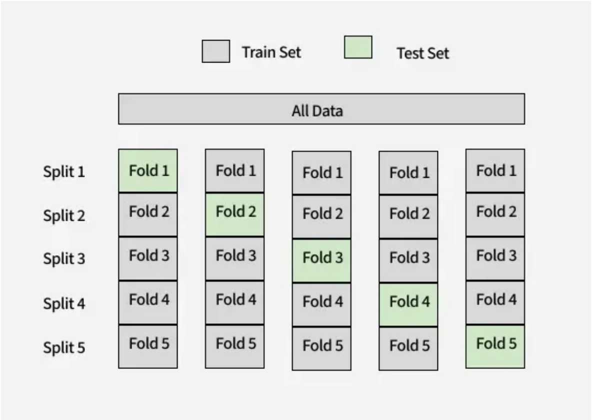


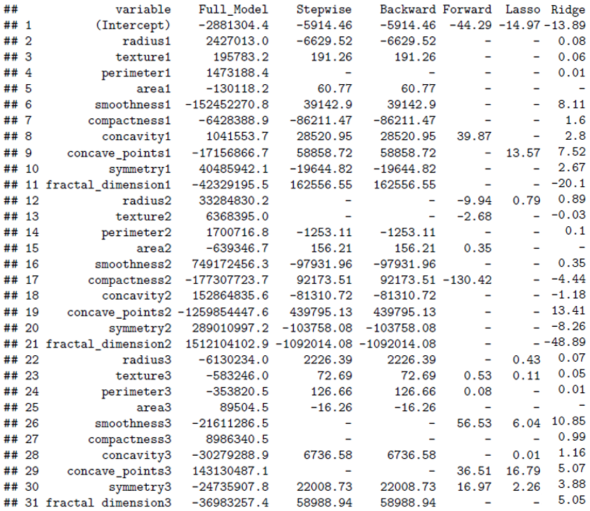
Figure 2: Ilustration of the 5-Fold Cross-Validation process (obtained from [3]).

4 Results

4.1 Results of the first iteration.

4.1.1 Parameters

After the first iteration of parameter adjustment for all models, we obtain the following estimated parameters:



```
##              variable    Full_Model    Stepwise    Backward Forward  Lasso  Ridge
## 1         (Intercept)    -2881304.4    -5914.46    -5914.46  -44.29 -14.97 -13.89
## 2             radius1     2427013.0    -6629.52    -6629.52       -      -   0.08
## 3            texture1      195783.2      191.26      191.26       -      -   0.06
## 4          perimeter1     1473188.4           -           -       -      -   0.01
## 5               area1     -130118.2       60.77       60.77       -      -      -
## 6         smoothness1  -152452270.8     39142.9     39142.9       -      -   8.11
## 7        compactness1    -6428388.9   -86211.47   -86211.47       -      -    1.6
## 8          concavity1     1041553.7    28520.95    28520.95   39.87      -    2.8
## 9     concave_points1   -17156866.7    58858.72    58858.72       -  13.57   7.52
## 10          symmetry1    40485942.1   -19644.82   -19644.82       -      -   2.67
## 11 fractal_dimension1   -42329195.5   162556.55   162556.55       -      -  -20.1
## 12            radius2    33284830.2           -           -   -9.94   0.79   0.89
## 13           texture2     6368395.0           -           -   -2.68      -  -0.03
## 14         perimeter2     1700716.8    -1253.11    -1253.11       -      -    0.1
## 15              area2     -639346.7      156.21      156.21    0.35      -      -
## 16        smoothness2   749172456.3   -97931.96   -97931.96       -      -   0.35
## 17       compactness2  -177307723.7    92173.51    92173.51 -130.42      -  -4.44
## 18         concavity2   152864835.6   -81310.72   -81310.72       -      -  -1.18
## 19    concave_points2 -1259854447.6   439795.13   439795.13       -      -  13.41
## 20          symmetry2   289010997.2  -103758.08  -103758.08       -      -  -8.26
## 21 fractal_dimension2  1512104102.9 -1092014.08 -1092014.08       -      - -48.89
## 22            radius3    -6130234.0     2226.39     2226.39       -   0.43   0.07
## 23           texture3     -583246.0       72.69       72.69    0.53   0.11   0.05
## 24         perimeter3     -353820.5      126.66      126.66    0.08      -   0.01
## 25              area3       89504.5      -16.26      -16.26       -      -      -
## 26        smoothness3   -21611286.5           -           -   56.53   6.04  10.85
## 27       compactness3     8986340.5           -           -       -      -   0.99
## 28         concavity3   -30279288.9     6736.58     6736.58       -   0.01   1.16
## 29    concave_points3   143130487.1           -           -   36.51  16.79   5.07
## 30          symmetry3   -24735907.8    22008.73    22008.73   16.97   2.26   3.88
## 31 fractal dimension3   -36983257.4    58988.94    58988.94       -      -   5.05
```

Figure 3: Parameters adjusted for each parametric model.

We can see that the Lasso and Forward models are the simplest, with much smaller coefficient values. The Ridge model is slightly more complex, but its coefficient values are also very small. The Backward, Stepwise, and especially the Full model are more complex, and their coefficients are much larger.

4.1.2 Accuracy results of the first repetitions

By obtaining the Accuracy of the 5 folds of each model, we can analyze its dispersion.

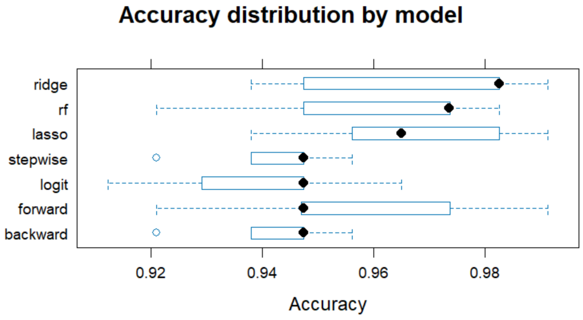


Figure 4: Parameters adjusted for each parametric model.

This chart shows a statistical summary of the accuracy of the 5 folds used.

The box shows the first and third quartiles; the black dot is the median. Points outside the whiskers are outliers.

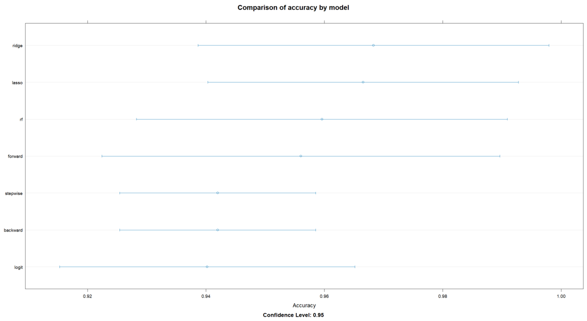


Figure 5: Parameters adjusted for each parametric model.

In this graph, the dot shows the average accuracy across the 5 folds for each model. The line around it shows the dispersion in accuracy. Shorter lines indicate a more consistent metric, while longer lines indicate less consistency and greater variability.

4.1.3 Training times for the first repetition

We timed how long each model took to adjust and the results are as follows.

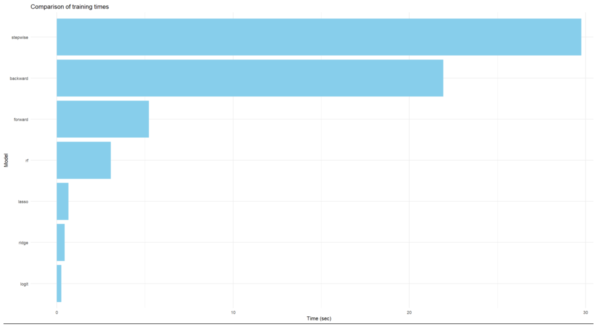


Figure 6: Time to adjust the different models in the first repetition.

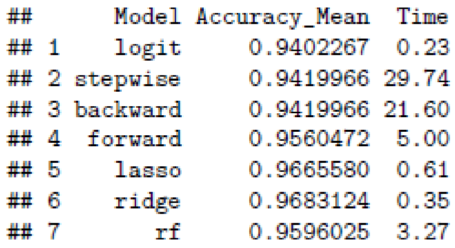
Note that the most accurate models are Ridge, followed by Lasso, and then Random Forest. Ridge's training time is slightly faster than Lasso, but Random Forest takes almost six times longer to be adjusted. Recall that in terms of complexity, the Lasso model was the simplest, but Ridge, despite being more complex, wasn not as complex as the other models.

The time required for stepwise, backward, and forward models is significantly longer due to the iterative process of removing or adding variables and checking their significance for the next step.

Similarly, the random forest model takes slightly longer due to its nature; the data subsetting process is time-consuming and is performed multiple times for different trees.

4.1.4 Analysis of overall performance after a single repetition.

Below is a table summarizing the results of the first repetition.



```
##       Model Accuracy_Mean  Time
## 1    logit     0.9402267  0.23
## 2 stepwise     0.9419966 29.74
## 3 backward     0.9419966 21.60
## 4  forward     0.9560472  5.00
## 5    lasso     0.9665580  0.61
## 6    ridge     0.9683124  0.35
## 7       rf     0.9596025  3.27
```

Figure 7: Summary of metrics for the first repetition.

Note that the most accurate models are Ridge, followed by Lasso, and then Random Forest. Ridge's training time is slightly faster than Lasso, but Random Forest takes almost six times longer to be adjusted. Recall that in terms of complexity, the Lasso model was the simplest, but Ridge, despite being more complex, was not as complex as the other models.

We can conclude that the Lasso model is the best option, as it offers simplicity, good accuracy, and a short adjustment time. If we are willing to sacrifice simplicity for a shorter training time and greater accuracy, Ridge is the best choice. Random Forest, being a machine learning model and therefore much less interpretable, has similar accuracy to the other models, which, while not bad, are inferior to Lasso and Ridge. However, it is faster and more accurate than the other models. The other models are not worth it in this situation; they have a significantly longer training time, their accuracy is worse, and they are much more complex.

4.2 Results after 50 repetitions

To obtain a more reliable assessment of model performance, we repeated the entire 5-fold cross-validation procedure 50 times. A single cross-validation split can introduce variability because the specific assignment of observations to folds may favor or disadvantage certain models. By repeating the process with different random fold partitions and averaging the results, we reduce the influence of any one particular split and obtain a more stable estimate of both accuracy and Brier score. This repetition also allows us to quantify the variability of each model's performance on different data partitions, providing a more robust comparison between methods.

Due to the time it takes for them to adjust, we will not be using the Backward and Stepwise models. These models did not demonstrate outstanding performance, so we decided to forget them at this stage.

4.2.1 Accuracy results after 50 repetitions

The box plot for the 50 repetitions shows that Ridge achieves the highest accuracy overall, followed by LASSO. Random Forest also performs well but is comparable to the Forward model and falls behind Ridge and LASSO.

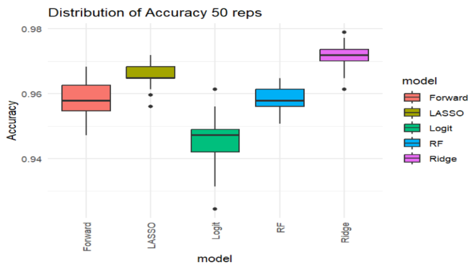


Figure 8: Accuracy summary after 50 repetitions.

4.2.2 Brier Score results after 50 repetitions

The Brier Score box plot indicates that all models achieve low error values, with Ridge and LASSO showing the lowest scores. Their box plots are also narrower, suggesting more consistent performance compared with the other models.

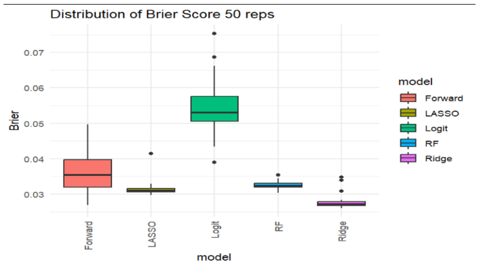


Figure 9: Brier Score summary after 50 repetitions.

5 Discussion

5.1 Interpretation and comparison of models

The overall results across the 50 repetitions show a consistent pattern in model performance. Ridge regression achieves the highest average accuracy, closely followed by LASSO, while Random Forest performs well but shows a larger spread in its predictive performance. This trend is also reflected in the Brier Score, where Ridge and LASSO obtain the smallest values and the tightest boxplots, indicating not only strong predictive accuracy but also high consistency across resampling. The stochastic nature of Random Forest, combined with the variability in bootstrap samples, likely explains why its performance exhibits greater dispersion.

5.2 LASSO Variable Selection Analysis

The drastic reduction in the choice of parameters by LASSO and the very different scale between the different estimated parameters of the different models lead us to analyze the correlation in the database.

A heat map showing the correlation of the explanatory variables will be displayed below.

There are too many variables, but we can easily notice that most of the squares are painted red, which indicates a strong correlation between the different variables in our data set.

This highly correlated data creates a multicollinearity problem in the logistic model that uses all the variables. This makes the model unstable and requires large coefficients to balance the repeated effects explained by other variables.

It is surprising that the base model has such high accuracy given its clear multicollinearity issues. But how does Lasso achieve slightly better precision without using so many variables and with much smaller coefficients?

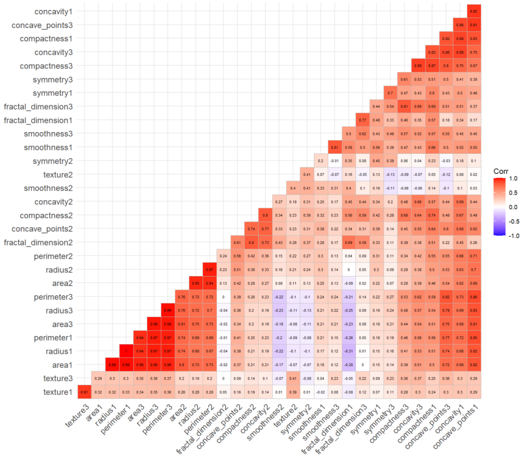


Figure 10: Heat map of the correlation in predictor variables.

By averaging the correlation of each variable with the others, we obtain the 10 variables with the highest average correlation and the 10 with the lowest average correlation. Comparing variables with those used by LASSO, we can note that 4 of the 10 most correlated variables are used by Lasso: concave_points1, radius3, concavity3 and concave_points3. Lasso, in turn, 3 of the variables with the least average correlation in absolute value: texture3, smoothness3 and symmetry3.

And finally there is the variable radius 2, which remains in the variables with intermediate average absolute correlation, specifically it is the 14th ordered from highest to lowest.

We can see that Lasso uses a combination of highly correlated variables, with poorly correlated variables and an intermediate variable.

The heat map of the variables used by Lasso is shown below.

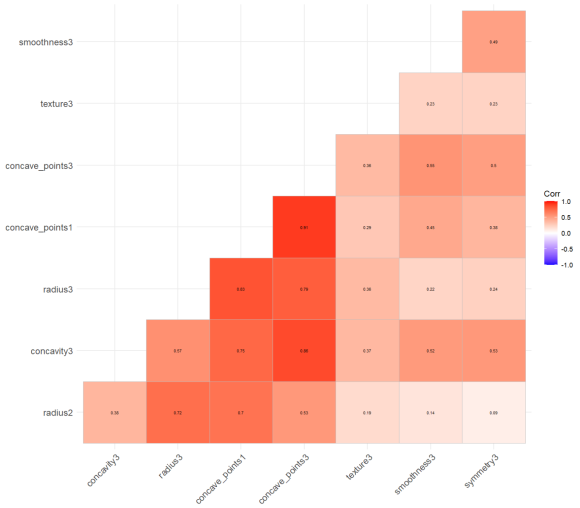


Figure 11: Heat map of the correlation in predictor variables used by LASSO.

We can see that variables with low correlation are in more orange or white tones, while variables with higher correlation tend to be more red.

This choice of variables is not accidental; in fact, it is quite logical. This behavior aligns with expectations for penalized regression: highly correlated variables tend to carry redundant information, and LASSO absorbs that shared signal by selecting only a few representatives. On the other hand, including some weakly correlated predictors may help the model capture additional, more specific patterns in the data that strong correlations do not explain. This suggests that the selected subset efficiently balances redundancy reduction with information specificity.

5.3 Variability and consistency of models

Repeating the full model fitting process 50 times offers insight into the stability of the algorithms. While a single cross-validation run can give a reasonable estimate of performance, it may be influenced by how the folds were partitioned. By varying the fold assignments across many repetitions, we gain a clearer view of the distribution of metrics and reduce the risk of over-interpreting results that may be tied to a particular random split. The narrow distributions of Ridge and LASSO demonstrate that their performance is robust to changes in resampling, whereas models like Logistic Regression and Forward selection exhibit more sensitivity.

6 Conclusion

The repeated evaluation of models across 50 iterations provides a robust assessment of predictive performance and variability. Ridge regression consistently achieves the highest accuracy and lowest Brier scores, indicating not only precise predictions but also stable performance across different data splits. LASSO performs similarly well, slightly below Ridge in accuracy, but with the added benefit of automatic variable selection, reducing the number of predictors while capturing the most informative variables. This behavior illustrates how penalization allows LASSO to summarize correlated information into fewer features while retaining predictive power. Forward selection and Random Forest also show strong predictive performance; however, they exhibit slightly higher variability across iterations, with Random Forest variability stemming from its stochastic nature due to bootstrap sampling and random feature selection at each split. Logistic regression without selection shows good average performance but lacks the refinement in variable selection and consistency offered by Ridge and LASSO. Overall, these results emphasize the importance of considering both predictive accuracy and model interpretability, suggesting that Ridge is ideal when stability and precision are the priority, while LASSO provides a practical trade-off between accuracy and feature reduction.

Future work could include validating these models on external datasets to assess generalizability, exploring Elastic Net to combine the strengths of LASSO and Ridge, and performing more extensive hyperparameter tuning for tree-based models like Random Forest. Such approaches could enhance model reliability, improve prediction stability, and provide deeper insights into the variable relationships driving tumor classification.

Appendix

The Rmd document containing the code ready to be executed in R, as well as its compiled PDF version and the database used, can be found in the following GitHub repository: https://github.com/Stephanie-Daniella/Breast-Tumors

References

[1] Wolberg W, Mangasarian OL, Street N, Street W. Breast Cancer Wisconsin (Diagnostic) [Database]. UCI Machine Learning Repository; 1993. DOI: 10.24432/C5DW2B. Available from: https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data

[2] An illustration of Random Forest [Internet]. ResearchGate; [cited 2025 Dec 10]. Available from: https://www.researchgate.net/publication/372809468/figure/fig3/

[3] GeeksforGeeks. K-Fold Cross-Validation in Machine Learning [Internet]. GeeksforGeeks; [cited 2025 Dec 10]. Available from: https://www.geeksforgeeks.org/machine-learning/k-fold-cross-validation-in-machine-learning/