A Comparative Evaluation of Homogeneous and Heterogeneous Ensemble Methods for Breast Cancer Classification

Jean-Pierre du Preez
Department of Computer Science
University of Stellenbosch
Stellenbosch, South Africa
26356163@sun.ac.za

***Abstract*—This report compares homogeneous and heterogeneous ensemble classifiers for breast-cancer diagnosis using Data/breastCancer.csv (diagnosis B→0, M→1). A single stratified 80/20 train/test split is created. For each seed (42, 1337, 2020, 7, 99), hyperparameters are selected by GridSearch with stratified 5-fold cross-validation on the training split only; the best pipeline is then refit on the full training portion and evaluated once on the held-out test set. Results are aggregated across seeds to stabilize estimates.**

Models include a homogeneous Random Forest and a heterogeneous set—Logistic Regression, SVM (RBF), and Decision Tree—combined via soft voting (averaged calibrated probabilities) and stacking (Logistic Regression meta-learner trained on out-of-fold base predictions). Evaluation uses ROC-AUC (primary), Average Precision, Macro-F1, Accuracy, Brier score, and Expected Calibration Error, with supporting ROC/PR and calibration plots.

Across seeds, all methods deliver strong discrimination on the held-out test set. Stacking provides small, consistent gains and slightly better probability calibration, while paired fold-wise tests show no statistically significant advantage over alternatives. Overall, ensemble approaches yield robust, well-calibrated predictions, with stacking offering a marginal edge without materially increasing deployment complexity.

***Index Terms*—Ensemble Learning, Breast Cancer Classification, Random Forest, Logistic Regression, SVM, Decision Tree, Soft Voting, Stacking**

I. Introduction

Accurate and reliable prediction of breast cancer from clinical features remains a central objective in medical decision support. Ensemble learning—combining multiple classifiers to reduce generalization error—offers a principled route to improved discrimination and calibration. Within this paradigm, two design families are commonly contrasted: homogeneous ensembles, which aggregate models of the same type, and heterogeneous ensembles, which blend different model classes to leverage complementary inductive biases. Determining which family yields superior performance for this classification task is the primary objective of this study.

This report conducts a controlled comparison on the Breast Cancer dataset, contrasting a homogeneous Random Forest with heterogeneous ensembles built from Logistic Regression, SVM (RBF), and Decision Tree base learners. Two combination strategies are examined for the heterogeneous case: soft voting, which averages calibrated class probabilities, and stacking, which trains a Logistic Regression meta-learner on out-of-fold base predictions. To ensure fair model selection and robust estimates, hyperparameters ("control parameters") are tuned using GridSearch with stratified 5-fold cross-validation within the training split, repeated across multiple random seeds; performance is then assessed on a held-out test set per seed. Evaluation emphasizes ROC-AUC as the primary metric, complemented by Average Precision, Macro-F1, Accuracy, and probability-calibration measures (Brier score, Expected Calibration Error). Statistical comparisons employ paired Wilcoxon tests on fold-wise scores.

The investigation addresses three questions: (i) whether a homogeneous Random Forest or a heterogeneous ensemble yields better discrimination and calibration on this dataset; (ii) whether a learned combiner (stacking) offers advantages over probability averaging (soft voting); and (iii) how sensitive conclusions are to hyperparameter choices and data resampling. Results indicate uniformly strong performance across all ensembles; stacking provides small, consistent gains and slightly improved calibration, while fold-wise tests do not show statistically significant superiority.

The remainder of the report provides background on ensemble learning concepts, implementation, empirical design, results and Conclusion.

II. Background

A. Binary medical diagnosis

Breast cancer diagnosis from tabular clinical features is a supervised binary classification task. Typical constraints include limited sample sizes, mild class imbalance, nonlinear feature interactions, and the need for robust generalization to unseen patients. In such contexts, methods that curb variance and stabilize predictions are preferred [2].

B. Ensembles for tabular clinical data

Ensemble learning combines multiple models to exploit diversity in inductive biases. When individual learners make partially uncorrelated errors, aggregation reduces variance and improves robustness to idiosyncrasies of a given sample or split. Standard overviews emphasize bagging, boosting, and stacking as complementary strategies for balancing bias and variance [2], [4].

C. Homogeneous ensembles (Random Forest)

Homogeneous ensembles repeat a single learner family while injecting randomness to induce diversity. Random Forests exemplify this approach by training many decision trees on bootstrap resamples and considering random feature subsets at each split, which decorrelates trees and lowers variance relative to a single deep tree [1]. Motivations in this domain:

- Trees capture nonlinearities and interactions common in biomedical features;
- Random Forests are comparatively insensitive to feature scaling and tolerate moderate outliers;
- Simple feature-use summaries provide coarse interpretability for clinical communication.

Data quality notes: consistent handling of missing values and duplicates remains important; light class imbalance may call for class weighting or stratified resampling [2].

D. Heterogeneous ensembles (model diversity via voting and stacking)

Heterogeneous ensembles blend different learner families—for example, linear, margin-based kernel, and tree models—to leverage complementary strengths [2], [3]. Two aggregation schemes are common: *soft voting*, the average of calibrated class probabilities from the base models, which offers simplicity, low variance, and straightforward deployment when base probabilities are well-behaved; and *stacking*, where a second-level (meta) model learns how to combine base predictions using out-of-fold estimates as inputs. A simple meta-learner (often logistic regression) is typically preferred to limit variance while enabling data-driven weighting of base learners [2], [3].

E. Base learner roles and complementary strengths

Logistic Regression provides a linear, well-regularized baseline whose coefficients support transparent effect summaries; it benefits from standardized inputs and handles multicollinearity with appropriate regularization [2]. Support Vector Machines with RBF kernels offer flexible nonlinear boundaries and can be effective in high-dimensional settings when margin softness and kernel width are properly controlled; scaled inputs are required. Decision Trees capture threshold rules and interactions and are highly interpretable locally, yet exhibit high variance as single models, motivating their frequent use inside ensembles. Combining LR (linear bias), SVM-RBF (nonlinear margins), and trees (rule-based splits) aims to maximize error diversity, a key ingredient for ensemble gains [2], [4].

F. Data quality considerations

Clinical tabular data commonly present missing values (including placeholder tokens), duplicate records, outliers, and heterogeneous feature scales. Linear and kernel methods generally require standardized inputs; tree-based approaches do not, but remain sensitive to noisy or duplicated observations. Mild class imbalance is typical and can be mitigated through class weighting or stratified resampling [2].

G. Control parameters

Generalization depends on a small set of control parameters across the chosen models and combiners: for Random Forests, the number of trees, maximum depth, and feature-subsampling level govern the variance–bias balance [1]; for Logistic Regression, the regularization strength and penalty manage variance and multicollinearity; for SVM-RBF, the margin parameter and kernel scale set boundary complexity; and for voting/stacking, the use of calibrated probabilities, weighting strategy, and meta-learner simplicity shape the stability of the combined model [2], [4].

III. Implementation

This section presents the system architecture, components, and algorithms independent of any particular experiment. The solution is an end-to-end, *leakage-safe* pipeline that transforms raw tabular rows into calibrated class probabilities via base learners and ensemble combiners. All modeling logic is encapsulated in `scikit-learn` pipelines and meta-estimators to ensure consistent preprocessing, selection, and inference.

A. Architecture and data flow

The workflow proceeds from data ingestion to calibrated prediction in a single, composable graph. Data are loaded, sanitized, and typed; features and labels are separated; numeric and categorical columns are enumerated. Two preprocessing branches are constructed to match model families:

- **Linear branch (LR/SVM):** median imputation, low-variance filtering, robust outlier clipping, and standardization; categorical features one-hot encoded.
- **Tree branch (DT/RF):** median imputation and low-variance filtering; categorical features one-hot encoded; scaling omitted.

Base pipelines pair each learner with the appropriate branch. Logistic Regression, SVM-RBF, and Decision Tree include probability calibration inside the pipeline; Random Forest relies on native probabilities. Ensemble combiners operate atop these pipelines: *soft voting* averages calibrated probabilities, and *stacking* trains a Logistic Regression meta-learner on out-of-fold (OOF) base probabilities. At inference, raw features traverse the assigned

preprocessor, produce base probabilities, and are combined into a final calibrated probability.

B. Leakage-safe design

Definition. Data leakage occurs when information from validation or test partitions influences training or preprocessing, inflating performance. **Safeguards.** All transforms (imputation, filtering, clipping, scaling, encoding) and probability calibration are embedded *inside* estimator pipelines, so fitting occurs only on the training portion of each fold; validation/test partitions receive transform/predict only. Stacking uses OOF base probabilities to train the meta-learner. The held-out test split is never used for selection or calibration. A single CV splitter is reused across models within a run, and no target-derived preprocessing is performed. These guardrails make the implementation leakage-safe by construction.

C. Algorithms and design rationale

The model set targets complementary inductive biases with post-hoc calibration ensuring reliable probabilities:

- **Logistic Regression (LR):** linear, regularized baseline; stable and interpretable with standardized inputs.
- **SVM-RBF:** margin-based nonlinear classifier capturing smooth boundaries; benefits from standardized features.
- **Decision Tree (DT):** rule-based thresholds and interactions; high variance in isolation, informative in ensembles.
- **Random Forest (RF):** bagged, feature-subsampled trees reducing variance; strong baseline with minimal preprocessing.

Calibration for LR, SVM-RBF, and DT is implemented via `CalibratedClassifierCV`. Soft voting provides a low-variance average of calibrated probabilities. Stacking employs a regularized LR meta-learner trained on OOF base probabilities to learn data-driven weights while constraining variance at the second level. The combined set—LR (linear), SVM-RBF (smooth nonlinear), and tree-based learners (rule-based)—maximizes error diversity.

D. Training logic (model-agnostic)

Training is fold-aware and pipeline-centric. Each base specification selects its corresponding preprocessor (linear for LR/SVM, tree for DT/RF). Where applicable, a calibrator is wrapped around the classifier inside the pipeline. A fold-aware model-selection routine explores control parameters and retains the best estimator, refitted within the routine on the training partition. For stacking, OOF probabilities are produced by training bases on each fold's training subset and predicting on its validation subset; these columns form the meta-feature matrix for the LR meta-learner. At deployment, bases are refit on all available training data; inference concatenates base probabilities and applies the trained meta-learner to obtain the final probability. Solver choices favor determinism and robust convergence for LR; tree/RF growth and bagging are deterministic under fixed random states; SVM margins are combined with calibrator-based probabilities.

E. Control parameters

The components expose a small set of tunable parameters that regulate capacity, variance, and probability quality. The table lists the primary knobs for each module; conceptual roles are described in §III and exact grids/values are specified in the empirical procedure. Parameters are adjusted only within cross-validation to prevent leakage, and selected settings are later refit on the full training partition for deployment.

Table I. Control parameters and key tunable settings for each component.

Component	Key Parameters
Random Forest	`n_estimators`, `max_depth`, `max_features`, `min_samples_split/leaf`
Logistic Regression (base/meta)	`C`, penalty/solver, `max_iter`
SVM-RBF	`C`, `gamma`
Decision Tree	`max_depth`, `min_samples_split/leaf`, `ccp_alpha`
Calibration	method (sigmoid/isotonic), internal `cv`
Stacking meta-learner	LR with `C` (L2)
Soft voting	fixed/equal weights (if used)

F. Implementation specifics (condensed)

Preprocessing, calibration, and ensemble composition are encapsulated within pipelines to guarantee reproducibility and enforce the leakage-safe protocol in §III. Data ingestion normalizes placeholder missings, removes duplicates, coerces numeric-like strings, enumerates categorical columns explicitly, and excludes obvious identifiers, yielding a stable feature matrix across models and folds. Each base learner—calibrated where applicable—is exposed as a first-class pipeline; soft voting averages base probabilities, and stacking trains an LR meta-learner on OOF probabilities and, at inference, consumes refit base probabilities. The software stack comprises Python with `numpy`, `pandas`, `scikit-learn`, YAML/dataclasses configuration, and `joblib` for artifact persistence. Random-state wiring is consistent across learners and cross-validation objects to enable deterministic reruns. Protocol-specific details (splits, seeds, grids, metrics, step order) are defined in the empirical procedure.

IV. Empirical Procedure

This section specifies the reproducible protocol used in the study. It first describes cross-validation and leakage controls, then the hyperparameter grids, ensemble assembly, the number of runs and reproducibility choices, and

finally the evaluation metrics and statistical tests with brief motivations. All preprocessing and calibration are fitted strictly within training folds/partitions (see §III for the leakage-safe design).

A. Cross-validation, data splits, and leakage control

Dataset and target. The experiments use BreastCancer.csv. The prediction target is diagnosis, with labels mapped as B→0, M→1. If a gender column is present, it is treated as categorical and encoded with one-hot encoding.

Splitting strategy. A single *stratified 80/20* train/test split is created. Model selection within the training portion uses *stratified 5-fold cross-validation (CV)* with shuffling. Within a run (seed), the same folds are used for every model to ensure paired comparisons.

Independent runs. Five independent runs are executed with seeds $[42, 1337, 2020, 7, 99]$. Multiple seeds stabilize estimates, reduce dependence on any single split, and enable fold-wise paired testing.

B. Hyperparameter optimisation (GridSearchCV)

Searcher and selection rule. Model selection uses `GridSearchCV` (`refit=True`) on the training split, sharing the same stratified 5-fold CV across all models within each seed. The scoring metric is ROC–AUC. For Logistic Regression, SVC (RBF), and Decision Tree, probability calibration is included inside each pipeline via `CalibratedClassifierCV(method='sigmoid', cv=5)`.

Table II. Parameter grids used by `GridSearchCV`.

Model	Param	Grid
Random Forest	`n_estimators`	{300}
	`max_depth`	{None, 8, 12}
	`max_features`	{"sqrt", 0.5}
	`min_samples_split`	{2, 5}
Logistic Regression (cal.)	`C`	{0.1, 1.0, 10.0}
SVC (RBF, cal.)	`C`	{1.0, 10.0}
	`gamma`	{0.01, 0.1}
Decision Tree (cal.)	`max_depth`	{None, 5, 8, 12}
	`min_samples_split`	{2, 5, 10}
	`min_samples_leaf`	{1, 2, 5}
	`ccp_alpha`	{0.0, 0.001}
Calibration (bases)	`method`, `cv`	{`sigmoid`}, {5}

C. Ensemble assembly (concise)

Soft voting. Average the *calibrated* class-1 probabilities from Logistic Regression, SVC (RBF), and Decision Tree with equal weights.

Stacking. Generate *5-fold OOF* base probabilities (same CV as above), then train a *Logistic Regression* meta-learner (L2, C=1.0) on the OOF matrix. For inference, refit each base on the full training split and feed their probabilities to the trained meta-learner.

D. Evaluation framework and statistical tests

Table III lists the metrics reported in this study. ROC–AUC is used as the primary discrimination metric, complemented by Average Precision (PR–AUC) for positive-class retrieval, Macro-F1 and Accuracy for thresholded performance, and two calibration measures—Brier Score and ECE—to assess the reliability of predicted probabilities.

Table III. Evaluation metrics and concise motivations.

Metric	Rationale
ROC–AUC	Threshold-free ranking quality; robust under mild imbalance.
Average Precision (PR–AUC)	Emphasizes positive-class retrieval;
Macro-F1	Balances precision and recall across classes; class-size agnostic.
Accuracy	Overall correctness; included for completeness.
Brier Score	Proper scoring rule;
ECE (Expected Calibration Error)	Summarizes the gap between predicted and empirical frequencies.

For statistical comparison, fold-wise *paired Wilcoxon signed-rank tests* are applied to ROC–AUC when comparing Random Forest with Soft Voting and with Stacking providing a robust check for systematic improvements.

V. Results

This section presents the results of the empirical procedure. The optimial hyperparameters, Cross Validation performance and statistical tests are presented.

A. Optimal hyperparameters

Table IV summarizes the hyperparameters selected most often across seeds. Random Forest typically prefers deeper (unconstrained) trees with `max_features="sqrt"` and moderate regularisation via `min_samples_split`. Logistic Regression favours stronger L2 regularisation (`C=0.1`). The calibrated SVC (RBF) commonly selects `C=10.0` with a smoother kernel (`gamma=0.01`). The calibrated Decision Tree tends toward a shallow, pruned configuration. For LR, SVC, and DT, probabilities are calibrated with `sigmoid` using 5-fold internal CV.

Table IV. Selected hyperparameters.

Model	Selected Hyperparameters
Random Forest	`n_estimators=300`, `max_depth=None`, `max_features="sqrt"`, `min_samples_split=5`
Logistic Regression (cal.)	`C=0.1`; calibration: `sigmoid`, `cv=5`
SVC (RBF, cal.)	`C=10.0`, `gamma=0.01`; calibration: `sigmoid`, `cv=5`
Decision Tree (cal.)	`ccp_alpha=0.001`, `max_depth=5`, `min_samples_leaf=5`, `min_samples_split=2`; calibration: `sigmoid`, `cv=5`

Table V. Aggregated cross-validation results (mean ± std over 25 fold×seed evaluations).

Model	ROC-AUC	PR-AUC	Macro-F1	Accuracy	Brier ↓
Random Forest	0.984 ± 0.0138	**0.981** ± 0.0147	0.947 ± 0.0268	0.951 ± 0.0246	0.0393 ± 0.0142
LR (cal.)	0.985 ± 0.0151	0.976 ± 0.0240	0.952 ± 0.0197	0.956 ± 0.0177	0.0402 ± 0.0116
SVC (RBF, cal.)	0.986 ± 0.0144	0.975 ± 0.0274	**0.960** ± 0.0203	**0.963** ± 0.0188	0.0361 ± 0.0122
DT (cal.)	0.974 ± 0.0208	0.962 ± 0.0296	0.928 ± 0.0283	0.933 ± 0.0258	0.0554 ± 0.0170
Soft Voting	0.986 ± 0.0142	0.979 ± 0.0224	0.957 ± 0.0248	0.960 ± 0.0224	0.0393 ± 0.0124
Stacking	**0.986** ± 0.0143	0.979 ± 0.0226	0.958 ± 0.0251	0.961 ± 0.0233	**0.0335** ± 0.0134

Table VI. Held-out test performance aggregated over five seeds (mean ± std).

Model	ROC-AUC	PR-AUC	Macro-F1	Accuracy	Brier ↓	ECE ↓
Random Forest	0.988 ± 0.0066	0.984 ± 0.0060	0.949 ± 0.0140	0.953 ± 0.0133	0.0416 ± 0.0098	0.0605 ± 0.0119
LR (cal.)	0.989 ± 0.0046	0.981 ± 0.0065	0.952 ± 0.0068	0.956 ± 0.0062	0.0412 ± 0.0051	0.0753 ± 0.0096
SVC (RBF, cal.)	0.991 ± 0.0046	**0.987** ± 0.0053	0.962 ± 0.0095	**0.965** ± 0.0088	0.0330 ± 0.0041	0.0698 ± 0.0112
DT (cal.)	0.974 ± 0.0069	0.960 ± 0.0128	0.922 ± 0.0375	0.926 ± 0.0365	0.0544 ± 0.0154	0.0946 ± 0.0133
Soft Voting	0.990 ± 0.0036	0.985 ± 0.0062	0.953 ± 0.0132	0.956 ± 0.0124	0.0379 ± 0.0061	0.0795 ± 0.0122
Stacking	**0.991** ± 0.0038	0.986 ± 0.0052	**0.963** ± 0.0172	**0.965** ± 0.0164	**0.0316** ± 0.0089	**0.0602** ± 0.0152

B. Cross-Validation Performance (Aggregated)

Table V shows the aggregated cross-validation results. Results are aggregated across seeds by first averaging scores over the 5 folds within each seed, then averaging those per-seed means across all seeds. ROC-AUC is the primary metric; complementary metrics include PR-AUC, Macro-F1, Accuracy, and Brier (↓ indicates lower is better). Bold indicates the best value per column. Parentheses show across-seed standard deviation.

Both heterogeneous ensembles (Soft Voting, Stacking) slightly outperform the Random Forest on the primary metric (ROC-AUC) by ∼0.16–0.18 percentage points, with Stacking narrowly leading overall. While Random Forest attains the highest PR-AUC, Stacking provides the strongest balanced classification (best Macro-F1 and Accuracy) and the most reliable probability estimates (lowest Brier). In practice, Soft Voting offers near-best ROC-AUC with lower implementation complexity, whereas Stacking is the strongest all-round choice if a modestly more complex pipeline is acceptable.

C. Statistical Comparison of Models (Cross-Validation)

Test design. Each ensemble was compared to the Random Forest (RF) baseline using a paired Wilcoxon signed-rank test on fold-wise ROC-AUC values, pairing the same fold within the same seed. Two planned comparisons (RF vs Soft Voting, RF vs Stacking) were evaluated with a Holm correction for multiple testing ($\alpha = 0.05$, two-sided). Reported Δ values are mean differences in ROC-AUC expressed in percentage points (pp), computed from the aggregated CV means.

Interpretation. Both ensembles show small positive mean gains over RF (approximately +0.16–0.18 pp ROC-AUC), but the paired tests do not reject the null hypothesis of equal performance after multiplicity adjustment. Under this CV protocol, the ensembles perform on par with RF; any advantage is modest and not statistically reliable at $\alpha = 0.05$.

Table VII. Paired Wilcoxon tests vs RF on fold-wise ROC-AUC (pp = percentage points).

Comparison	ΔROC-AUC vs RF (pp)	Wilcoxon W	Wilcoxon p-value	Holm-adj p	Significant (α=0.05)
RF vs Soft	+0.157	125.5	0.4929	0.4929	No
RF vs Stacking	+0.184	116.0	0.3383	0.6767	No

D. Final Test Set Performance (Held-Out)

Overview. Table VI shows the aggregated held-out test performance. Evaluation on the held-out test split aggregates results across five seeds (mean ± standard deviation). Primary metric is ROC-AUC; complementary metrics include PR-AUC (average precision), Macro-F1, Accuracy, Brier score (lower is better), and Expected Calibration Error (ECE; lower is better). Bold indicates the best value per column. Now do the statistical part for the held-out asset.

Interpretation. Stacking achieves the strongest results across all reported metrics on the held-out test split, with small but consistent gains in ROC-AUC and PR-AUC relative to Random Forest and Soft Voting, and the lowest Brier and ECE, indicating better-calibrated probabilities. Soft Voting improves discrimination over Random Forest and reduces Brier, but exhibits higher ECE, suggesting overconfident probabilities. Overall, Stacking offers the best balance of discrimination and calibration on the final test set.

*E. **Conclusion***

Across cross-validation and the held-out test split, all models achieved high discrimination on the breast-cancer task. Heterogeneous ensembles (soft voting and stacking)

delivered small, consistent lifts over the Random Forest baseline in aggregated cross-validation ROC–AUC; however, paired Wilcoxon tests on fold-wise scores did not indicate statistical significance at $\alpha = 0.05$. On the held-out test set, stacking provided the strongest overall profile, combining near-best ROC–AUC and PR–AUC with the lowest Brier score and ECE, indicating improved probability calibration. A calibrated RBF-SVC occasionally matched or exceeded ensemble discrimination, underscoring that a well-tuned, well-calibrated base learner remains competitive when model diversity is limited.

Overall, the evidence supports three conclusions: (i) ensemble methods yield robust predictions with modest average advantages over a strong Random Forest baseline; (ii) stacking offers the best balance of discrimination and calibration under the current protocol; and (iii) observed gains over Random Forest are not statistically decisive in cross-validation. For deployment, stacking is recommended when probability quality and incremental ROC–AUC improvements justify added complexity, while soft voting serves as a simpler alternative with near-equivalent discrimination. Further improvements are most likely from increasing base-model diversity, learning non-uniform ensemble weights with strict out-of-fold training, and refining per-base calibration (e.g., isotonic or Platt) to enhance probability reliability without inflating variance.

References

[1] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] J. D. Kelleher, B. Mac Namee, and A. D'Arcy, *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*, 2nd ed. Cambridge, MA: MIT Press, 2020.

[3] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[4] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, ser. Machine Learning & Pattern Recognition. Boca Raton, FL: Chapman and Hall/CRC, 2012.

Implementation details can be found in the following repository: *GitHub repository: https://github.com/JP-DuPreez/ML_Assignment3_RNN*