Contents

S1 Optuna hyperparameter tuning - Tuning history **2**

S2 Optuna hyperparameter tuning - Training time **4**

S3 Optuna hyperparameter tuning - Hyperparameter importance **6**

S4 Nested cross-validation - Model generalisation **12**

S5 Verification results for all models tested - Overall scores **14**

S6 Verification results for all models tested - Breakdown scores (ROC curves) **16**

S7 Verification results for all models tested - Breakdown scores (Precision-Recall curves) **18**

S8 Verification results for all models tested - Breakdown scores (Reliability diagrams) **20**

S1 Optuna hyperparameter tuning - Tuning history

The hyperparameter optimisation history plot (Figure S1) shows the overall good performance of the Optuna library in identifying quickly (within the first 10 trials, orange lines) and with relatively small variations (grey lines) the sets of hyperparameters that maximise the chosen evaluation metrics (AUC-ROC and AUC-PR). Performance remains fairly consistent between different outer folds (lines in shades of grey and orange). The optimisation histories for AUC-ROC (Figure S1a-l) and AUC-PR (Figure S1m-x) are very similar, with plots for loss functions specific for imbalanced datasets (Figure S1g-l and s-x) showing more variability, but in general better overall performance, than those for more general loss functions (Figure S1a-f and m-r).

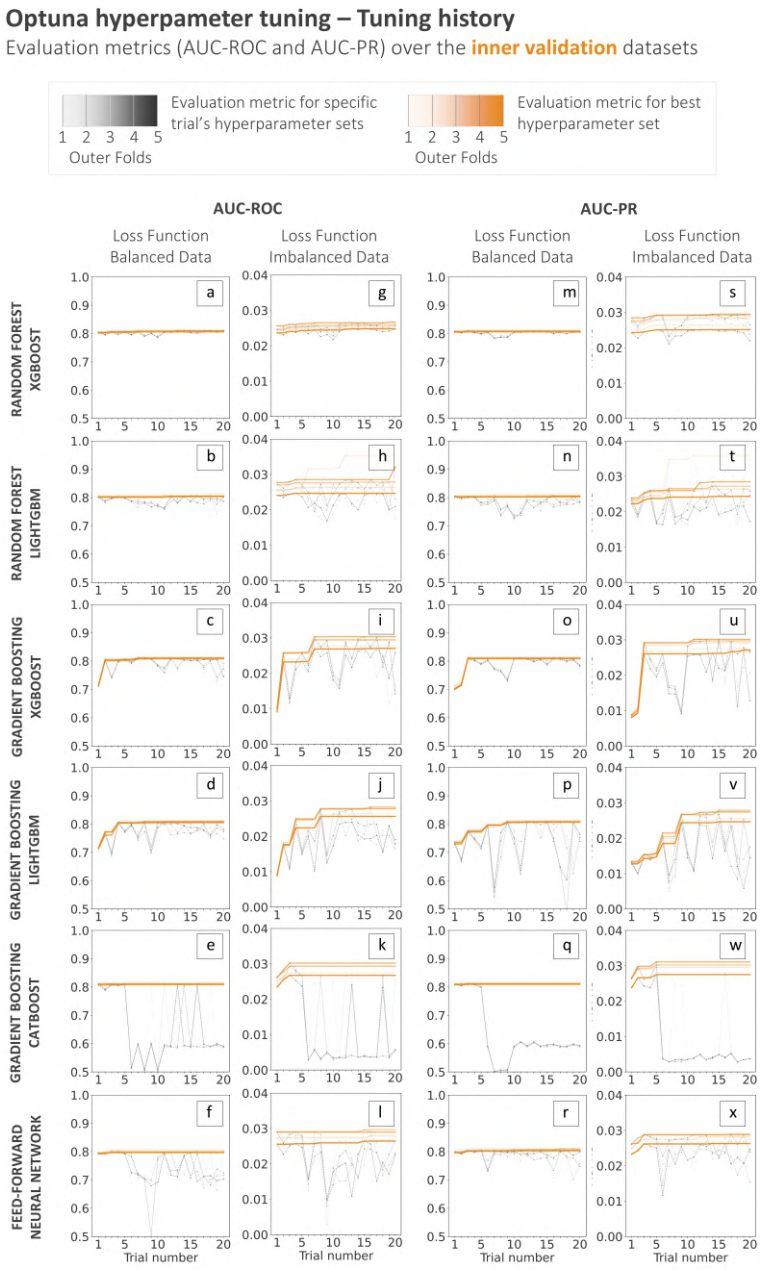

Figure S1. Optuna's hyperparameter optimisation history. Evolution of the two evaluation metrics (AUC-ROC, panels (a) to (l) - and AUC-PR, panels (m) to (x)) maximised during the 20 trials run over the **inner validation folds** to tune the hyperparameters of six data-driven models (from top to bottom): random forest XGBoost, random forest LightGBM, gradient boosting XGBoost, gradient boosting LightGBM, gradient boosting CatBoost, and feed-forward neural network. The lines in shades of grey indicate individual trial performances, whilst lines in shades of orange highlight the best-performing hyperparameter set, identified by Optuna's Bayesian optimisation process. The shades of grey and orange represent the values of the evaluation metrics for each outer fold (lightest shade for the first outer fold - 1 - and darkest for the last - 5). Panels (a) to (f) and (m) to (r) represents the results obtained using the standard binary cross-entropy loss functions - mostly used for balanced datasets - whilst panels (g) to (l) and (s) to (x) present the outcomes obtained with the weighted loss functions (specifically configured for imbalanced data).

S2 Optuna hyperparameter tuning - Training time

The training times show substantial computational differences across model architectures (Figure **??**). Decision-tree-based implementations (except for CatBoost) demonstrate more efficient training times. On average, the training time per outer fold remains between 75 and 100 seconds, with peaks that do not exceed 500 seconds. CatBoost's training times are around 500 seconds per outer fold, with peaks reaching 2000 seconds. The feed-forward neural network exhibits the longest training times among all models, with an average training time of around 2000 seconds and peaks ranging from 4000 to 6000 seconds. These times indicate that it requires 20 minutes per outer fold to optimise hyperparameters and train the decision-tree-based models (except for CatBoost), compared to 2.5 and 8 hours per outer fold, respectively, for CatBoost and the feed-forward neural network. The choice of loss function and evaluation metric shows minimal impact on training duration across all architectures.

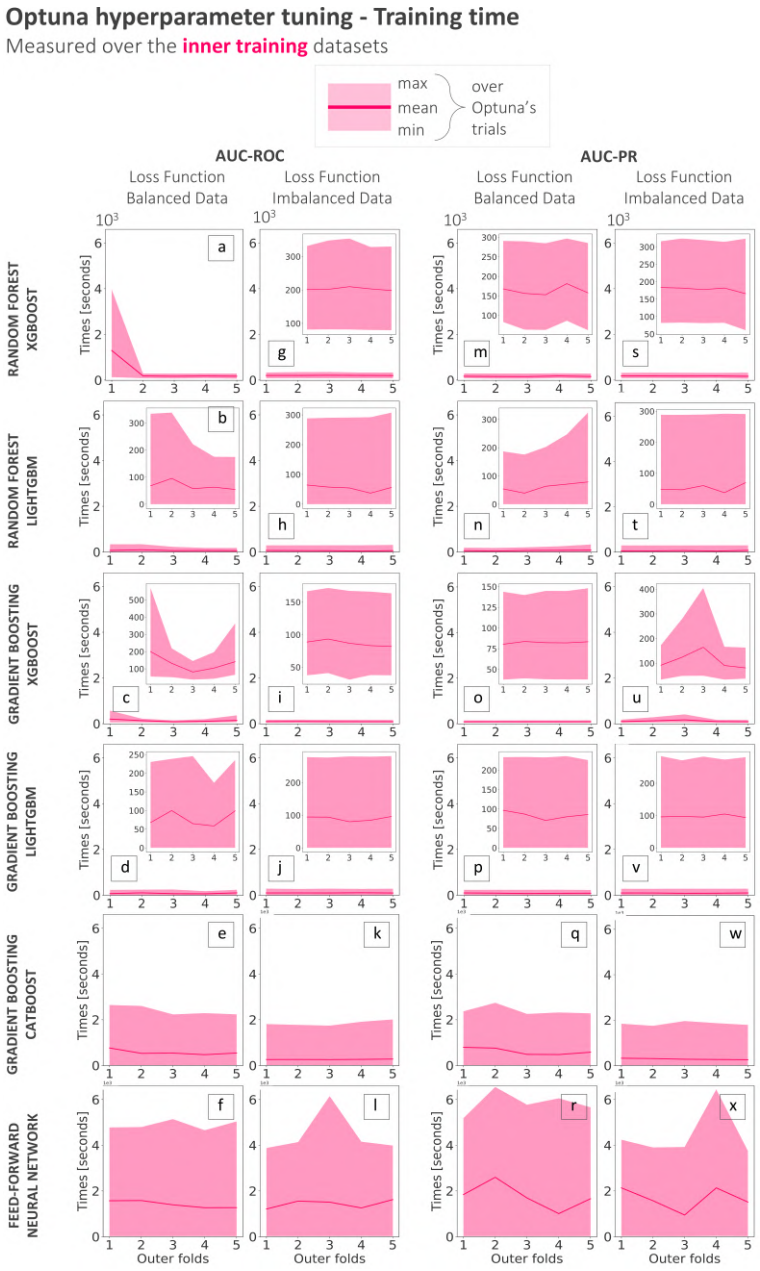

Figure S2. Optuna's training time. Evolution of training times (in seconds) for each k_outer=5 outer folds across the n_trials = 20 trials run over the *inner training folds* (the solid line represents the mean while the shaded area represents the minimum and maximum values). The training times for six data-driven models (from top to bottom) are shown: random forest XGBoost, random forest LightGBM, gradient boosting XGBoost, gradient boosting LightGBM, gradient boosting CatBoost, and feed-forward neural network. Training times are shown for both evaluation metrics (AUC-ROC - panels (a) to (l) - and AUC-PR - panels (m) to (x)) and loss function configurations (balanced and imbalanced datasets). Inset plots provide magnified views where appropriate.

S3 Optuna hyperparameter tuning - Hyperparameter importance

The hyperparameter importance analysis for XGBoost and LightGBM gradient boosting implementations reveals that maximum depth and learning rate consistently emerge as the most influential parameters. This aspect suggests that optimal performance depends fundamentally on striking a balance between model complexity and convergence dynamics. Maximum depth controls the model's capacity to capture complex hydro-meteorological interactions, whilst learning rate determines the magnitude of iterative corrections, requiring careful calibration to preserve gradient signals from rare positive events. LightGBM demonstrates additional sensitivity to the number of estimators due to its leaf-wise tree construction, producing more informative individual trees that necessitate precise ensemble size optimisation. In contrast, XGBoost's level-wise approach generates simpler trees that plateau predictably, reducing the criticality of this parameter. Notably, the positive class weight parameter exhibits a generally lower influence across both implementations, suggesting that structural parameters governing tree complexity and learning dynamics exert a greater impact on minority class detection than explicit re-weighting strategies. Thus, this emphasises the primacy of architectural optimisation over class balancing approaches. CatBoost exhibits considerable variability in the hyperparameters that most strongly influence model performance, with this variability dependent on both the chosen evaluation metric and the loss function configuration. When optimising for AUC-ROC, a set of parameters proves critical (e.g. depth and learning rate), yet these same parameters have minimal impact when considering AUC-PR. Furthermore, the implementation of weighted loss functions fundamentally alters the previously seen importance rankings, creating distinct optimisation priorities for balanced versus imbalanced scenarios. Unlike XGBoost and LightGBM, where maximum depth and learning rate consistently dominate, CatBoost lacks such universal governing parameters. The absence of consistent parameter hierarchies suggests that CatBoost's distinctive algorithmic approach generates a more complex hyperparameter space.

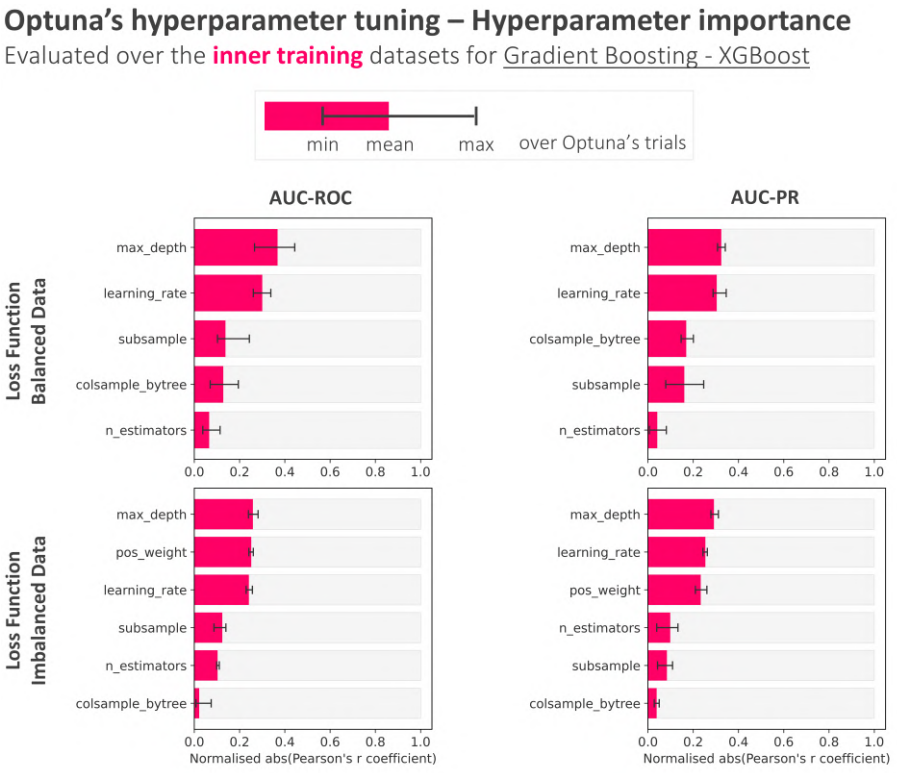

Figure S3. Optuna's hyperparameter importance for the XGBoost implementation of gradient boosting. Normalised absolute Pearson's correlation coefficients obtained for the n_trials = 20 trials run over the *inner training folds* (bars represent mean values, whilst error bars show the minimum and maximum values across the Optuna trials). Feature importance is shown for models trained with loss functions for balanced datasets and specific for imbalanced datasets, and for both evaluation metrics (AUC-ROC and AUC-PR).

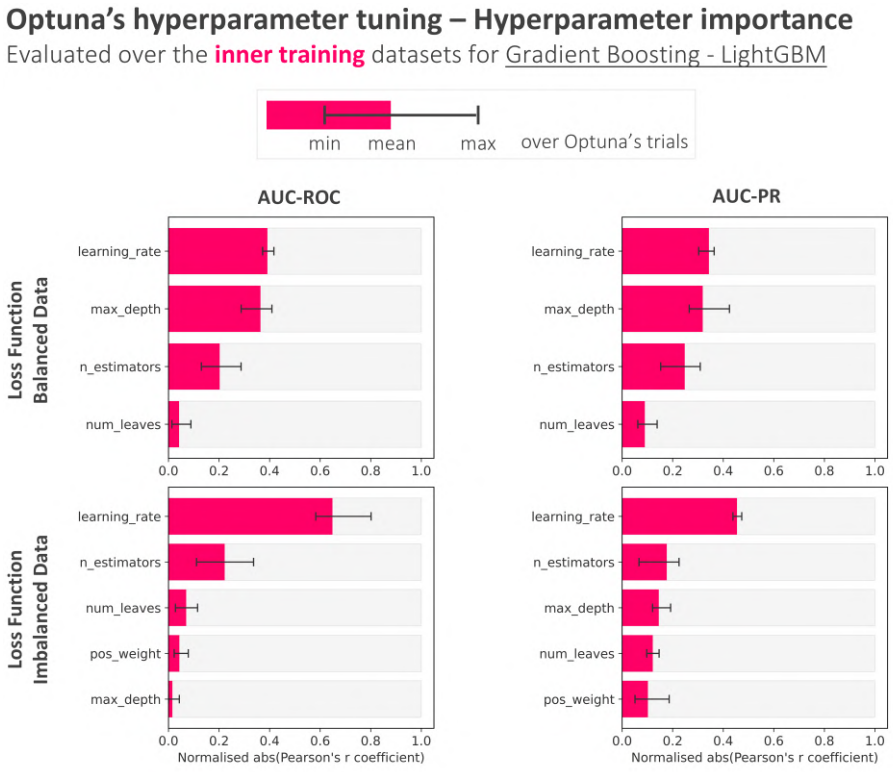

Figure S4. Optuna's hyperparameter importance for the LightGBM implementation of gradient boosting. Similar to Figure S3

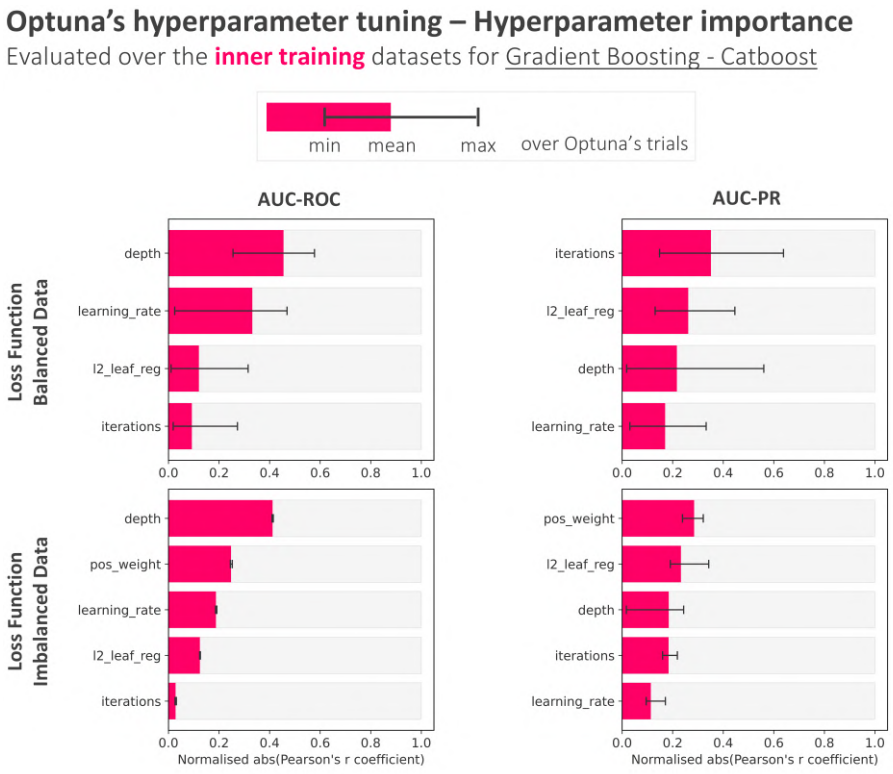

Figure S5. Optuna's hyperparameter importance for the CatBoost implementation of gradient boosting. Similar to Figure S3

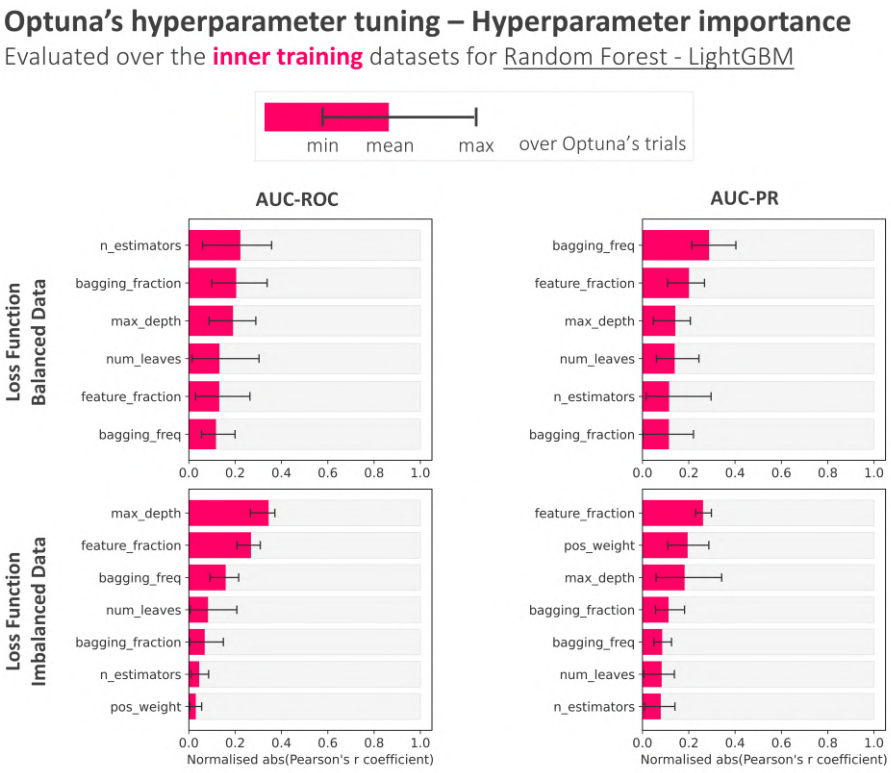

Figure S6. Optuna's hyperparameter importance for the LightGBM implementation of random forest. Similar to Figure S3

In contrast to gradient boosting implementations, random forest models exhibit inconsistent hyperparameter importance rankings across different loss functions and evaluation metrics, with parameters such as maximum depth, feature sampling ratios, and the number of estimators alternating in their relative influence depending on whether AUC-ROC or AUC-PR is optimised. This variability suggests that random forest hyperparameter spaces possess multiple viable configurations that achieve similar performance through different mechanisms (some configurations may excel through deeper individual trees, whilst others compensate with more aggressive feature sampling or a larger ensemble size), making the optimisation landscape more flexible but potentially more challenging to navigate systematically.

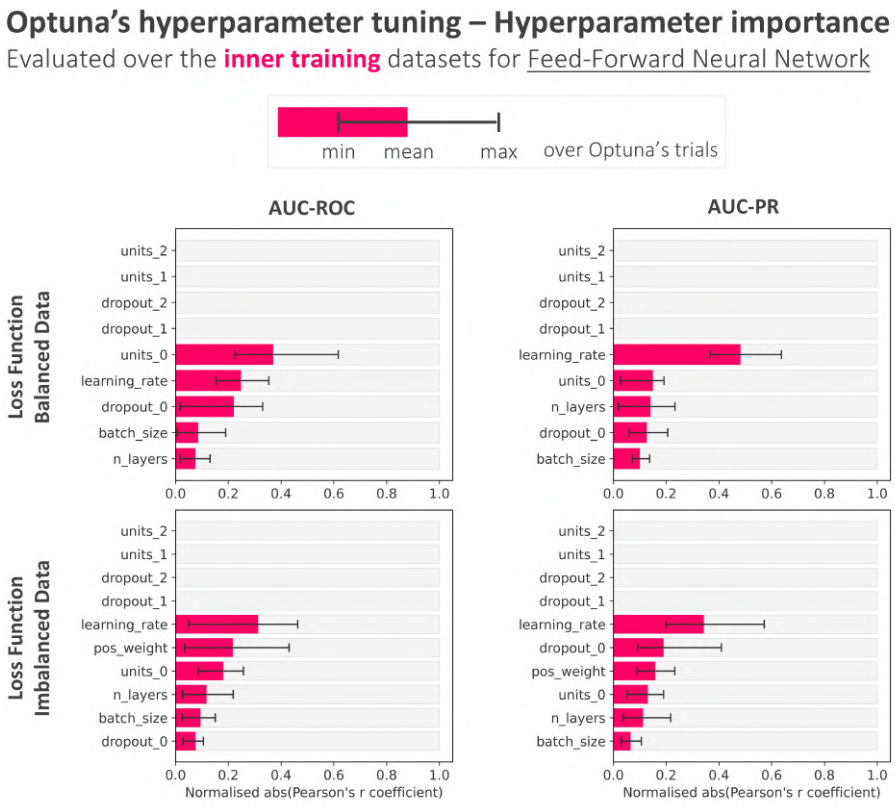

Figure S7. Optuna's hyperparameter importance for feed-forward neural network. Similar to Figure S3

The dominance of units_0, dropout_0, and learning rate for neural networks suggests that performance is primarily determined by the configuration of the first hidden layer, rather than the overall network depth. The parameter units_0 controls the initial representational capacity, determining how effectively raw hydro-meteorological features are transformed into meaningful intermediate representations for detecting rare flash flood patterns. This layer must balance sufficient complexity to capture non-linear relationships whilst avoiding overfitting to the sparse positive examples. The high importance of dropout_0 demonstrates that regularisation at this critical layer is essential for generalisation under extreme class imbalance. By randomly deactivating neurons during training, dropout forces the development of robust, redundant representations that prevent the model from memorising noise in the limited positive examples. The prominence of the learning rate parameter reflects the challenge of navigating an imbalanced dataset, where the gradient signal from rare positive examples can be easily overwhelmed by the abundance of negative cases. The diminished importance of deeper layer parameters suggests that shallow, well-regularised architectures may be the best choice for this application.

S4 Nested cross-validation - Model generalisation

The close values for both evaluation metrics (AUC-ROC and AUC-PR) estimated over the *inner validation folds* and the *outer test folds* show the robustness of the nested cross-validation framework in mitigating overfitting during hyperparameter optimisation (Figure S8). The fact that outer test performance remains generally bounded or close to the performance ranges estimated over the inner validation folds demonstrates that Optuna's Bayesian optimisation successfully identified hyperparameter configurations with robust generalisation capabilities to previously unseen data. The comparative analysis reveals minimal divergence between models trained with standard binary cross-entropy and those employing weighted loss functions (formulated specifically for class-imbalanced datasets). This observation holds for both evaluation metrics. Across the evaluated models, performance metrics demonstrate remarkable consistency, with the notable exception of CatBoost. Specifically, mean AUC-ROC values cluster between 0.7 and 0.8, whilst CatBoost exhibits inferior performance with values between 0.6 and 0.7. Similarly, mean AUC-PR values range from 0.02 to 0.03 across most models, with CatBoost again demonstrating diminished performance with values between 0.01 and 0.02. For both metrics, random forest implementations exhibit the narrowest bands, followed by gradient boosting implementations (except for CatBoost, which displays the widest bands among all models) and the feed-forward neural network. The wider bands for CatBoost exhibit greater sensitivity to hyperparameter choices and require more careful tuning to achieve optimal results. Random forest implementations exhibit the most constrained performance bands, indicating robust hyperparameter spaces that yield consistent results across outer folds. Gradient boosting implementations (except for CatBoost) and the feed-forward neural network demonstrate intermediate variability. CatBoost exhibits the broadest performance bands amongst all evaluated models, suggesting that CatBoost's hyperparameter space possesses heightened sensitivity, necessitating more meticulous optimisation procedures to achieve competitive performance levels.

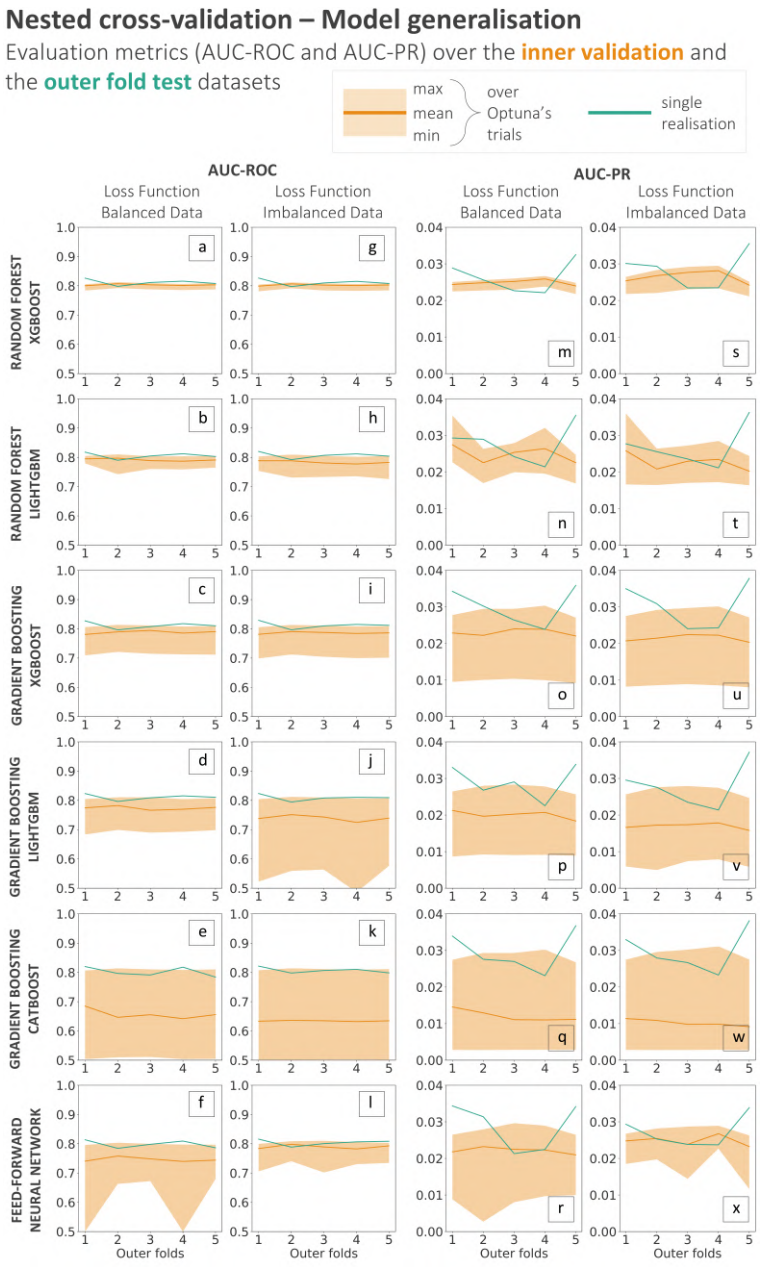

Figure S8. Model generalisation from nested cross-validation Values of the two evaluation metrics (AUC-ROC, panels (a) to (l) - and AUC-PR, panels (m) to (x)) across the 20 trials run over the *inner validation folds* (the solid line represents the mean while the shaded area represents the minimum and maximum values) and the *outer test fold* (the solid line correspond to the single realisation per outer fold). Panels (a) to (f) and (m) to (r) represents the results obtained using the standard binary cross-entropy loss functions - mostly used for balanced datasets - whilst panels (g) to (l) and (s) to (x) present the outcomes obtained with the weighted loss functions (specifically configured for imbalanced data).

S5 Verification results for all models tested - Overall scores

All data-driven models exhibit relatively stable AUC-ROC values around 0.8 for the verification dataset with minimal decrease from the AUC-ROC values obtained for the training dataset (Figure S9, first column). This behaviour is constant across both evaluation metrics (AUC-ROC and AUC-PR) and both types of loss function (general for balanced datasets and specific for imbalanced datasets). Hence, all data-driven models show an overall discrimination ability that generalises well from training to verification datasets. The AUC-PR values demonstrate more variability across data-driven models (Figure S9, second column), with CatBoost showing the highest estimates overall, especially when hyperparameters are tuned maximising the AUC-ROC evaluation metric (for both types of loss function). For all models, however, values of AUC-PR remain relatively small (Close to 0). Even though all models show a tendency to slightly overestimate the frequency of areas at risk of flash flood (FB just above 1, Figure S9, third column), the FB values also show high variability across the data-driven models. When considering the generic loss function, the XGBoost and LightGBM implementations for random forest and gradient boosting obtain the closest values to 1, but increase of 20% when considering the loss function specific for imbalanced datasets, showing even greater FB than the other tested models (which maintain similar values than those obtained when considering the generic loss function).

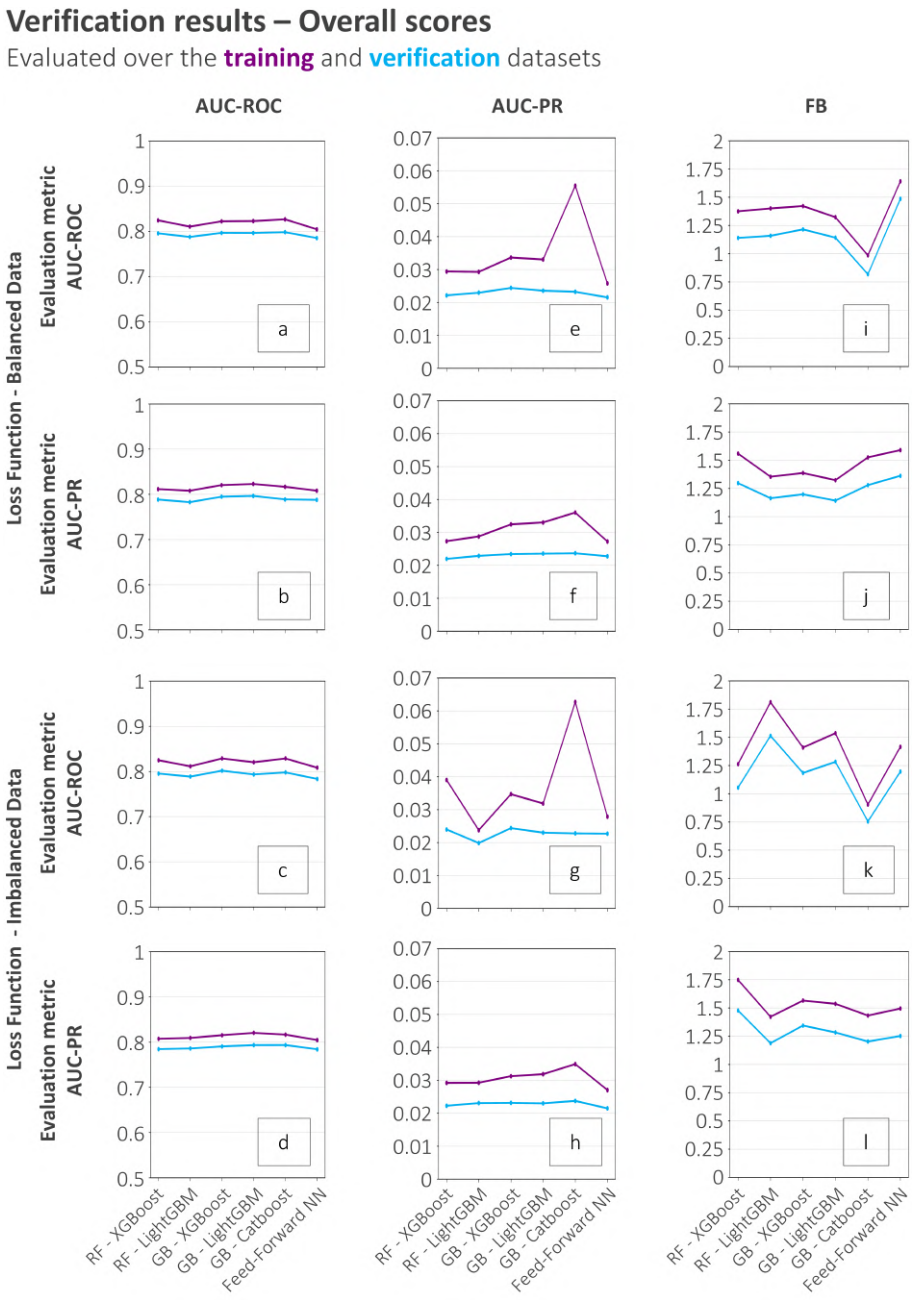

Figure S9. Verification results: overall scores The first, second, and third columns show, respectively, the estimates for the area under the ROC curve (AUC-ROC), the area under the precision-recall curve (AUC-PR), and the frequency bias (FB) for the six considered data-driven models. The estimates of the overall verification scores are shown for the training dataset and the verification dataset. Results are presented for the models trained considering both types of loss functions and evaluation metrics.

S6 Verification results for all models tested - Breakdown scores (ROC curves)

Overall, ROC curves exhibit minimal degradation between training datasets and verification datasets (Figure S10). The ROC curve, however, reveals distinct patterns between models trained with balanced S10a-f and m-r) versus imbalanced loss functions S10g-l and s-x). The ROC curves computed for models employing balanced loss functions maintain a remarkably consistent shape across all data-driven models and evaluation metrics, as well as a consistent relative difference between ROC curves computed using the 1% discretisation (solid lines) and the 0.01% (dashed lines). The former ROC curves yield lower AUC-ROC values than the latter due to the "truncation effect" caused by stopping the ROC curve at the 1% probability threshold. The same does not hold when considering models trained with weighted loss functions, except for CatBoost. These ROC curves achieve higher hit rates, e.g., 0.9 for the XGBoost and LightGBM implementations of gradient boosting (Figures S10i-j) and 0.6 for the feed-forward neural network (Figures S10l) compared to 0.4 for their balanced counterparts (Figures S10c-d and f), but at the cost of much higher false alarm rates, e.g., 0.5 for XGBoost and LightGBM an 0.2 for the feed-forward neural network compared to 0.025 in their balanced counterparts. Whilst XGBoost and LightGBM implementations of gradient boosting and the feed-forward neural network show consistent patterns across evaluation metrics, the random forest implementations display metric-dependent responses. Specifically, LightGBM Random Forest shows the altered behaviour when optimised for AUC-ROC, whereas XGBoost Random Forest responds differently under AUC-PR.

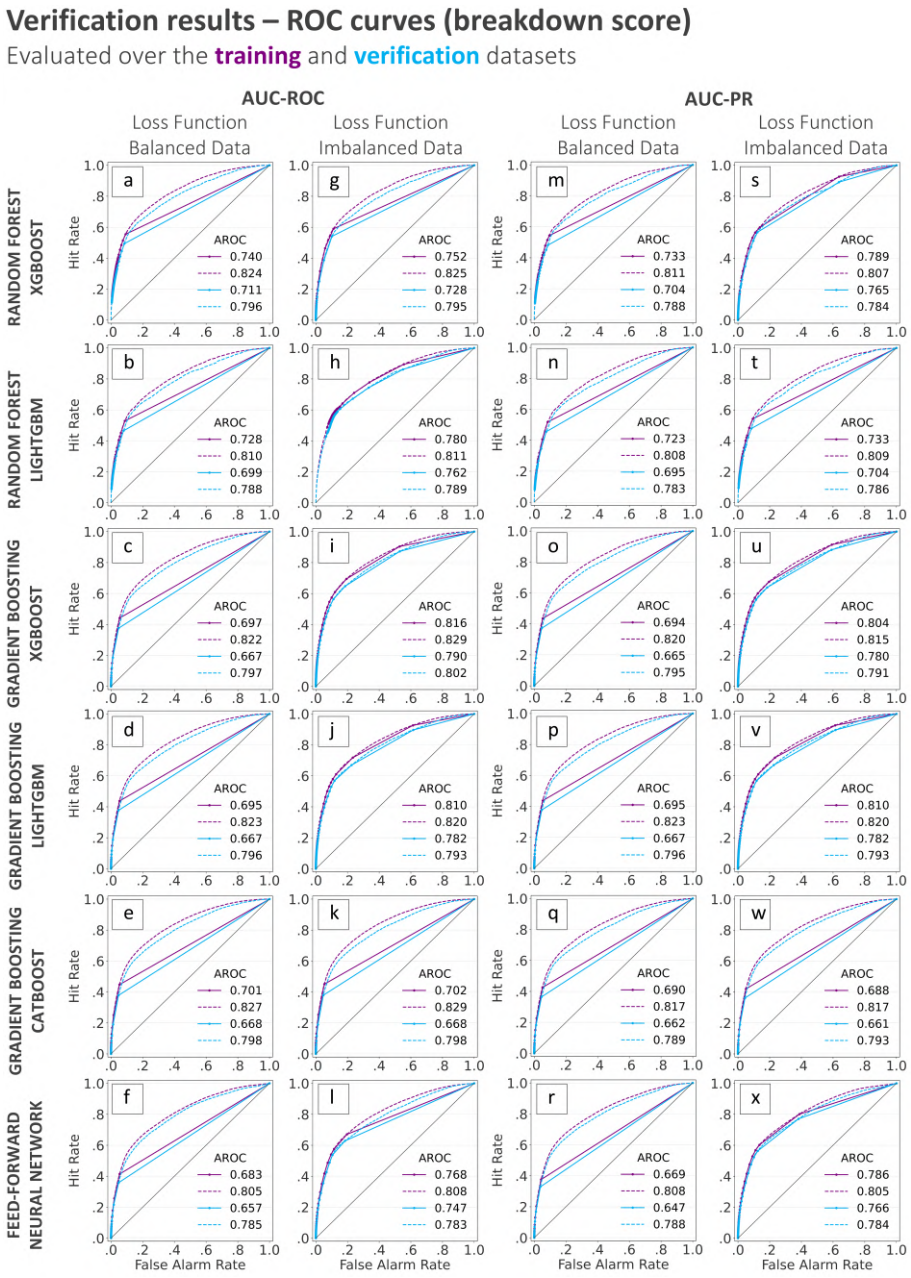

Figure S10. Verification results: breakdown scores (ROC curves) ROC curves computed for the training dataset and the verification dataset, for six data-driven models (from top to bottom): random forest XGBoost, random forest LightGBM, gradient boosting XGBoost, gradient boosting LightGBM, gradient boosting CatBoost, and feed-forward neural network. The solid lines represent ROC curves computed considering forecast probabilities discretised every 1%, whilst the dashed lines represent ROC curves computed using a finer discretisation of probabilities (0.01%). ROC curves are shown for the two evaluation metrics (AUC-ROC - panels (a) to (l) - and AUC-PR - panels (m) to (x)) and types of loss function configurations (balanced and imbalanced datasets) considered during the training of the data-driven models.

S7 Verification results for all models tested - Breakdown scores (Precision-Recall curves)

Overall, the precision-recall curves exhibit minimal degradation between training datasets (in solid lines) and verification datasets (in dashed lines), with the major differences concentrated mainly over recall values smaller than 0.2 (Figure S11). All precision-recall curves, while obtaining fairly small values of AUC-PR (see Figure S9, second column), remain away from their corresponding lines of no skill (grey solid lines for the training dataset and grey dashed line for the test dataset, which mostly overlap). Whilst there are some differences between the precision-recall curves attributable to the differences in model training, this time (as opposed to what was seen for the ROC curves), the most considerable differences are observed between the models themselves. In all implementations of gradient boosting (Figures S11c-e, i-k, o-q, and u-v), the training datasets achieve high precision (between 0.6 and 1) over very small values of recall, while the verification datasets achieve a precision value between 0.2 and 0.6. Only CatBoost, trained with the weighted loss function, maintains the value of 1. In the feed-forward neural network (S11f, l, r, and x), the precision-recall curve remains virtually identical between the training datasets and the verification datasets, including at very low values of recall. Finally, both XGBoost and LightGBM implementations of random forest show very different precision-recall curves. LightGBM (S11b, h, n, and t) shows a precision-recall curve that begins at recall values between 0.2 and 0.4, and precision values that do not exceed 0.1. XGBoost shows a similar behaviour when the model is trained using the generic loss function (S11a and m), whilst the curves for the models trained with the weighted loss function (S11g and s) shows a shape that is similar to that for the corresponding XGBoost implementation of gradient boosting (S11i and u).

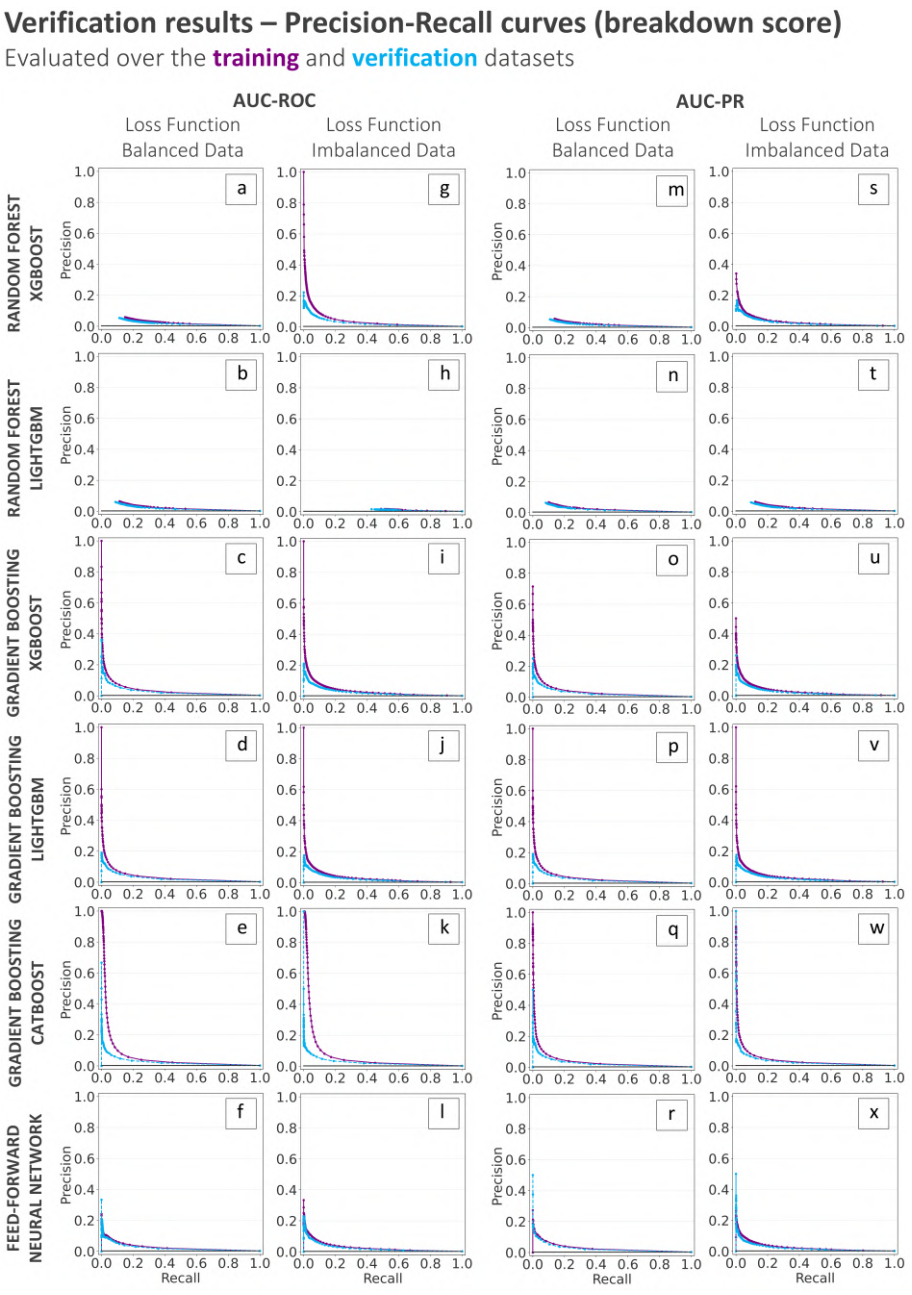

Figure S11. Verification results: breakdown scores (Precision-Recall curves) Similar to Figure S10 but for the Precision-Recall curve

S8 Verification results for all models tested - Breakdown scores (Reliability diagrams)

Overall, the reliability diagrams exhibit minimal degradation between training datasets and verification datasets, but it increases with increasing forecast probabilities (Figure S12). Models trained with balanced loss functions (Figures S12a-f and m-r) demonstrate different calibration characteristics compared to those using weighted loss functions S12g-l and s-x). For models trained with balanced loss functions, the gradient boosting implementations and the feed-forward neural network yield reliable forecast probabilities under 10% (20% for XGBoost). At higher probability ranges, where predictions become less frequent and diagrams noisier, distinct patterns emerge. LightGBM and XGBoost show mostly reliable probabilities across all probability ranges (Figures S12c-d and o-p). CatBoost systematically underestimates forecast probabilities (Figures S12e and q). The feed-forward neural network shows contrasting behaviour depending on the evaluation metric used during hyperparameter tuning: overestimation occurs when tuned for AUC-ROC (Figure S12f), while forecast probabilities remain reliable when tuned considering AUC-PR (Figure S12r). Models trained with weighted loss functions (Figures S12g-l and s-x) display systematic overestimation across the entire probability spectrum, except for CatBoost, which maintains similar calibration patterns to its balanced function counterpart. Random forest implementations show systematic overestimations independent of the considered loss function S12a-b, g-h, m-n, and s-t).

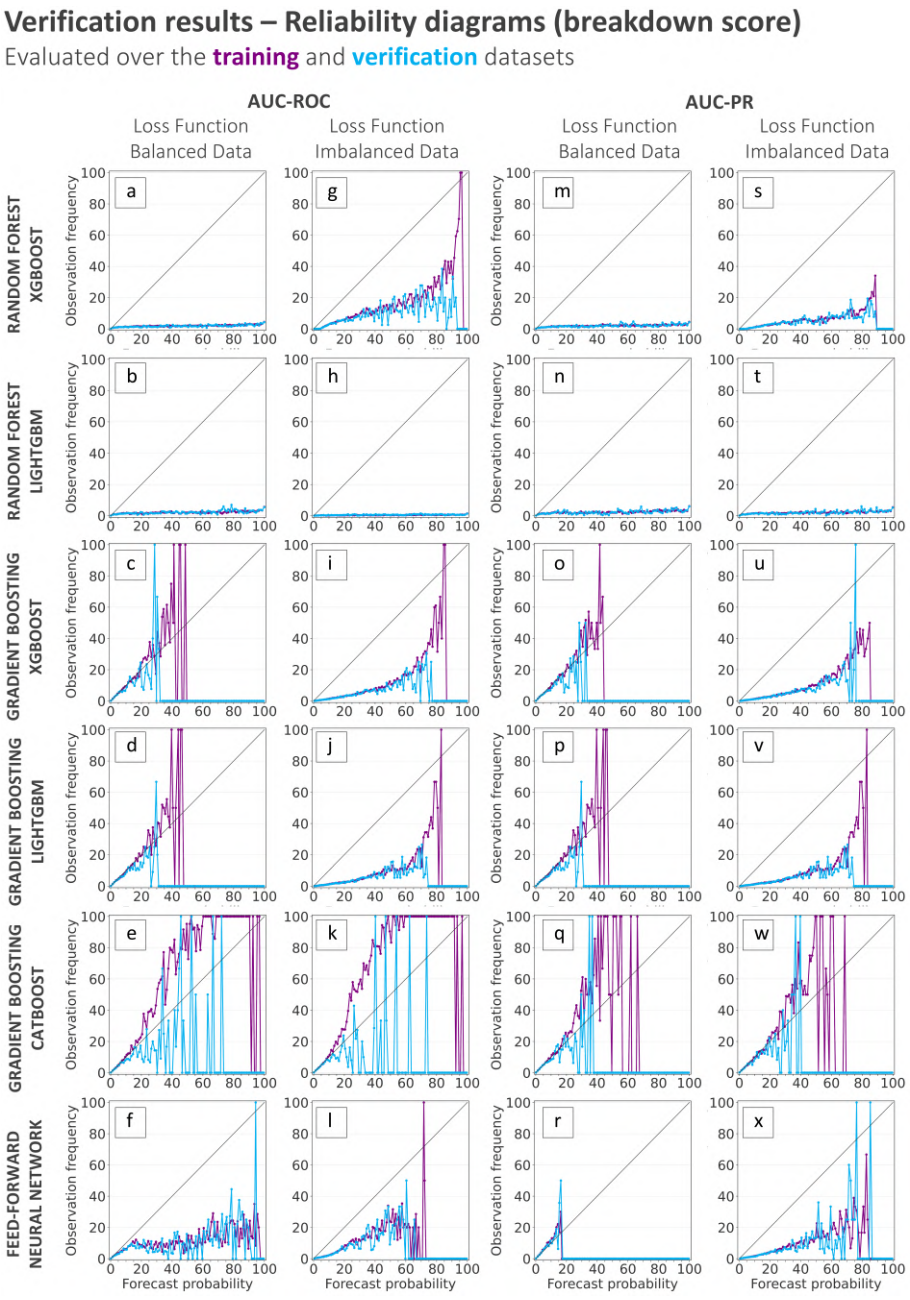

Figure S12. Verification results: breakdown scores (Reliability diagrams). Similar to Figure S10 but for reliability diagrams.

References