Statistical Learning for Classification: Multiclass Hyperspectral Analysis and High-Dimensional Feature Selection

Authors:

Prithika Narayanan Anastasiia Prokhorova Samuel Inman-Altass Fatma Kassim

Affiliation:
London School of Economics and Political Science

Abstract

We apply statistical learning methods to two classification problems. **First**, we address multiclass classification on hyperspectral satellite imagery (215,604 pixels, 218 spectral bands, 420–2450 nm) from an alpine region in Tyrol, Austria. We evaluate seven classifiers with and without PCA (10 components) using macro-F1 due to class imbalance (4.7%–23% per class). Linear SVM achieves best performance (F1 = 0.995, BalAcc = 0.997), though Logistic Regression delivers nearly identical results (F1 = 0.994, BalAcc = 0.996) with 250× faster inference (0.14s vs 36s). For glacier-ice detection (binary task, 5% positive class), we optimize F_2 score prioritizing recall, achieving near-perfect separation (F_2 = 0.998, recall = 1.000) with Logistic Regression demonstrating that glacier ice has a highly distinctive spectral signature. **Second**, we study feature selection in a high-dimensional neural decoding task where the dataset exhibits extreme $p \gg n$ imbalance ($p/n \approx 16$) and class imbalance (Left: 30.5%, Right: 69.5%), making stable generalisation difficult. We compare six feature selection paradigms: Forward Stepwise Selection (FSS), mRMR, Lasso, Elastic Net, Random Forest VI, and Gradient Boosting VI within a unified evaluation pipeline that ranks features, evaluates performance across subset sizes K, and assesses generalisation at optimal K^*. FSS yields the best overall trade-off, achieving strong performance (BalAcc = 0.774) with compact subsets ($K^* = 30$). Gradient Boosting VI performs similarly (BalAcc = 0.729) with slightly larger subset ($K^* = 35$), while Elastic Net achieves the highest BalAcc (0.805) at the cost of dense, less transferable solutions ($K^* = 2168$). We discuss why these methods behave differently and outline limitations affecting reliability of our findings.

1 Hyperspectral Land-Cover and Glacier-Ice Classification

1.1 Introduction

Hyperspectral land-cover classification poses two central challenges: high-dimensional correlated spectral features and substantial class imbalance. In this context, our goal is to identify which supervised learning method achieves the best overall performance on the eight-class task while also remaining computationally efficient and scalable. To answer this, we benchmark fourteen models (seven classifier families, each with and without PCA) and evaluate them using macro-F1, balanced accuracy, and prediction time. This formulates our research question:

Which classifier provides the best balance between predictive quality and computational cost for this dataset?

This trade-off is relevant, as models with near-identical accuracy can differ by orders of magnitude in inference speed. Having formulated the problem, we begin with an exploratory analysis of the spectral data and class structure.

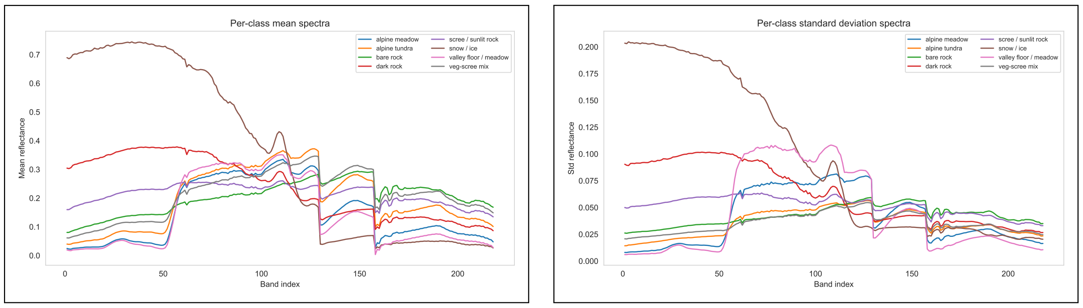
1.2 Exploratory Data Analysis

The dataset comprises hyperspectral satellite imagery from an alpine region in Tyrol, Austria (Guanter et al., 2015), with 215,604 pixels, each with 218 spectral bands and one of 8 land-type classes. The data is already clean: no missing, no infinite values. Class distribution is strongly imbalanced: alpine meadow $\approx 23\%$, alpine tundra and valley floor / meadow $\approx 17\%$, while the smallest class has only $\sim 4.7\%$ (Appendix Figure 6). Because of this imbalance, simple accuracy is not informative — even a naïve classifier predicting the majority class will show a high score. So we are going to use macro-F1, balanced accuracy, and stratified splitting.

From the spectral distributions (Appendix Figure 7 and 8) we see that many bands have shifted centers for different classes: this means that the features actually contain useful class information, and linear boundaries have a chance to work. This motivates trying LDA, logistic regression and linear SVM. At the same time, some classes show strong overlap in their distributions, so for these cases more flexible models (e.g., SVM with RBF) may work better.

The per-class mean spectra, standard deviation spectra and 2 components PCA (Figure 1 and Appendix Figure 11) clearly show that different land types have different shapes of spectral curves, which explains why classification should be possible. Classes with small internal variance (e.g., snow/ice) will be easier for models, and classes with a wide spread will be harder. Since amplitudes across bands differ a lot, `StandardScaler` is required, otherwise models will confuse scale differences with class differences.

Correlation analysis (Appendix Figure 9) shows very strong multicollinearity between neighbouring spectral bands, which supports the use of PCA techniques to avoid overfitting. Also, LDA's equal-covariance assumption is not really satisfied, so we do not expect LDA to be the best model, but it is still reasonable as a baseline.



(a) Mean spectra **(b)** Standard deviation spectra

Figure 1: Per-class mean and std reflectance spectra for the eight land types. Each curve represents the mean and std of a random sample (up to 5000 pixels) per class, showing distinct spectral signatures and strong inter-band correlation.

1.3 Training and Evaluation of Multiclass Classifiers

For the modelling part we only use the spectral bands Band_1 ... Band_218 as features. Pixel coordinates p_x, p_y are excluded: otherwise the model would mostly learn where on this particular scene each class is located, and such a model will not generalize to another image or geometry. However spatial distribution of land types can still be visualised and we can observe a kind of a real map snapshot. (Appendix Figure 10)

Because fitting all models with full hyper-parameter search on 215k pixels is computationally heavy, we first ran a sample-size study to determine the optimal training subset size (see Appendix, Figure 12). Based on this analysis, we fixed $n = 10{,}000$ as the working sample for hyper-parameter tuning: this keeps training time reasonable while already giving almost the final performance. For each model we then refit the best configuration on 70% of the full dataset and evaluate on the remaining 30%.

For every classifier, StandardScaler and PCA are placed inside the pipeline to avoid data leakage. Hyper-parameters are selected with StratifiedKFold cross-validation on the 10k tuning sample, using GridSearchCV. Because of strong class imbalance, we optimise macro-F1 rather than accuracy.

1.3.1 Hyperparameter reasoning and Models analysis

The choice of hyperparameter grids is guided by the main bias–variance trade-offs while keeping grid sizes manageable. For each classifier family we tune only the most influential parameters: regularisation strength, kernel choice, ensemble size, tree depth, or distance metric depending on the model. The detailed grids and justifications for each classifier are provided in Appendix A.2.

The final evaluation (Table 1) shows that the best overall macro-F1 and balanced accuracy are achieved by the linear SVM without PCA: F1 ≈ 0.995 and balanced accuracy ≈ 0.997. Multinomial logistic regression is extremely close (F1 ≈ 0.994), making it a practical alternative given its simpler interpretation and lower computational cost. Tree ensembles and k-NN also perform strongly (F1 around 0.978–0.985), while LDA underperforms (F1 $\approx$ 0.86) due to violated equal-covariance assumptions.

Adding PCA with 10 components slightly reduces F1 for most models (except QDA, where it stabilises covariance estimation), but strongly reduces prediction time for heavier models like SVM and k-NN.

In summary, we highlight SVM without PCA as the top performer and logistic regression as a practically equivalent, faster alternative.

1.4 Multiclass Performance Summary and Final Classifier

In total we trained 14 models (7 classifiers, each in two versions: raw features and PCA with 10 components). For each model family we first selected the best-performing configuration based on macro-F1 on the test set. For example, for the Logistic family the winner was the multinomial logistic regression without PCA, and for the SVM family the winner was the linear SVM without PCA. This gave us 7 "family-best" models, one per classifier type, which we then compared directly.

Table 1: Test performance of the best model from each classifier family for the 8-class land-type task (T1.2). Bold indicates the best value for each metric (ties all bold). Values taken from the notebook's t13_table summary (rounded to 3 d.p.).

Model	Accuracy	Bal. Acc	AUC	F1	Pred. time (s)
LDA	0.864	0.920	0.991	0.863	**0.06**
QDA	0.961	0.977	0.998	0.959	0.18
k-NN	0.979	0.987	**1.000**	0.978	165.67
GBDT	0.985	0.991	**1.000**	0.985	5.65
RF	0.985	0.991	**1.000**	0.984	0.78
Logistic	0.993	0.996	**1.000**	0.994	0.14
SVM	**0.995**	**0.997**	**1.000**	**0.995**	36.15

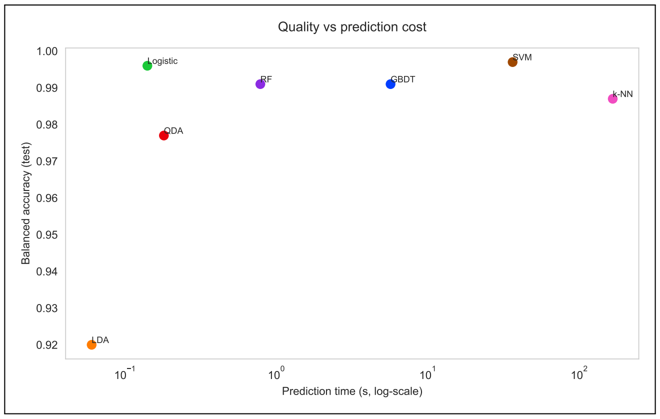


Figure 2: Balanced accuracy versus prediction time (log-scale) for the best model from each classifier family. Logistic regression lies close to SVM in accuracy while being substantially faster at inference.

Appendix figure 14 shows that almost all family-best models perform very well. The only clear outlier is LDA, which is consistent with the EDA results: equal-covariance assumptions are not satisfied, so the model systematically underperforms.

Among the remaining models, the strongest two are SVM and Logistic regression. Their test macro-F1 and balanced accuracy are almost identical (SVM: F1 $= 0.995$, BalAcc $= 0.997$; Logistic: F1 $= 0.994$, BalAcc $= 0.996$). Both models produce almost perfect confusion matrices with only small symmetric misclassifications between the classes that are known to overlap spectrally (Appendix Figure 5). To choose between these two top models, we also consider prediction time and computational cost. In figure 2, SVM sits at the very top-right: the highest quality, but also noticeably slower prediction ($\sim$ 36 s). Logistic regression, however, gives almost the same quality while being much faster ($\sim$ 0.14 s) and simpler to deploy. Because of this trade-off, we treat SVM as the "best pure-performance model" and Logistic regression as the "best efficiency–performance compromise". Since in many applications computational efficiency and scalability are important, **logistic regression without PCA is selected as the final model.**

1.5 Glacier-Ice Detection

In this part we transform the original 8-class problem into a binary detection task where glacier ice is the positive class. We create a new label `is_glacier` which is equal to 1 for pixels with land type "snow / ice" and 0 for all other classes. The proportion of glacier pixels is about 5%, so the data is strongly imbalanced and the usual accuracy is not a good objective. From initial trials, missing glacier pixels (false negatives) proved much more prevalent than occasionally labelling non-glacier pixels as ice. Therefore, we choose the F_2 score with glacier as the positive class as the main metric: compared with F_1, F_2 puts four times more weight on recall than on precision and directly penalises false negatives more heavily, which better matches the scientific objective.

The data is split into training and test sets using a 70/30 stratified split. As in Task 1.2, we first select a stratified subsample of 10,000 observations from the training set and use it only for hyper-parameter tuning. (Appendix Figure 13) This is a compromise between computational cost and stability of cross-validated estimates: on this subsample it is feasible to run 5-fold `GridSearchCV` with reasonably rich grids, while the resulting F_2 values are already very close to what we get on the full training set. After tuning, each best model is refitted on the entire 70% training data and evaluated once on the 30% test set.

We focus on three classifiers that represent different modelling ideas: Bagging, AdaBoost and Logistic Regression. For Bagging we use an ensemble of decision trees with depth 5 as base learners. The grid for `n_estimators` is $\{50, 100, 200\}$, which interpolates between relatively weak ensembles and more stable ones; going beyond 200 trees for this dataset gives almost no gain but increases computation. The parameter `max_samples` is varied in $\{0.5, 0.7, 1.0\}$ to see the effect of stronger bootstrap randomness: with 0.5 each tree sees only half of the training points, which can reduce variance and help generalisation, while 1.0 corresponds to classical bagging with full-size bootstrap samples. For AdaBoost we use very shallow trees (max depth 2) as weak learners and tune `n_estimators` $\in \{100, 200, 300\}$ together with the learning rate $\in \{0.05, 0.1, 0.2\}$. These ranges cover the usual trade-off between the number of boosting steps and the step size: very small learning rates or extremely large ensembles tend to be unnecessarily slow here without improving F_2.

Logistic Regression is used with StandardScaler inside a pipeline and class_weight="balanced", which up-weights glacier pixels in the loss. We tune only the main regularisation parameters: $C \in \{0.1, 1, 10, 100\}$ controls the strength of the penalty and spans from quite strong to very weak regularisation on this scale, and we allow several penalty–solver combinations (L1, L2, elastic net and no penalty together with `lbfgs`, `liblinear` and `saga`). The idea is not to exhaust all possible solver options, but to see whether sparsity-inducing penalties bring any advantage over simple L2 when optimising F_2.

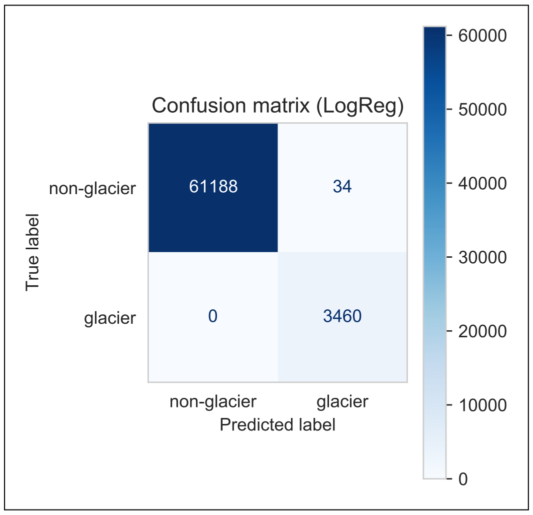
Table 2: Test performance of the glacier detection models (T1.4). The positive class is glacier ice. Bold indicates the best value for each metric (ties all bold; values from t14_summary).

Model	F_2	F_1	Rec	Prec	Bal. Acc	Acc	ROC-AUC	AveragePrecision
Bagging	0.984	0.985	0.983	0.987	0.991	0.998	**1.000**	0.999
AdaBoost	0.991	0.991	0.991	**0.991**	0.995	**0.999**	**1.000**	**1.000**
Logistic	**0.998**	**0.995**	**1.000**	0.990	**1.000**	**0.999**	**1.000**	**1.000**

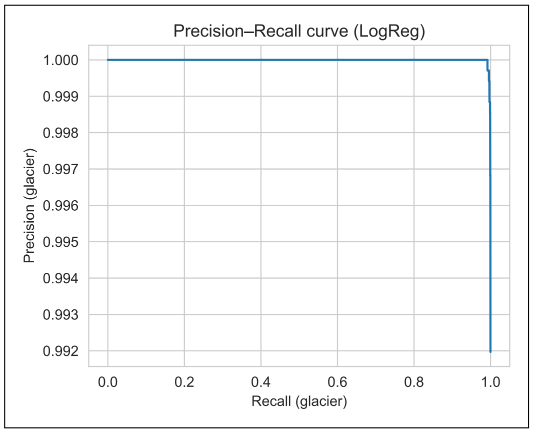
Hyper-parameters in all three models are selected with 5-fold Stratified `GridSearchCV` using F_2 as scoring function. The final test results are summarised in Table 2. All three methods achieve strong performance: Bagging reaches $F_2 = 0.984$, AdaBoost improves this to $F_2 = 0.991$, and Logistic

Regression achieves the highest $F_2 = 0.998$ with recall for glacier = 1.000 and precision = 0.990. Balanced accuracy is between 0.991 and 1.000 for all models, and ROC-AUC and average precision = 1.0. This confirms glacier ice has a very distinctive spectral signature and that even relatively simple models can separate it from other land types almost perfectly.

At the same time the differences between the three classifiers are informative. Bagging and AdaBoost are both strong, but they sometimes sacrifice a small amount of recall to gain precision, which slightly lowers F_2. Logistic Regression with class-balanced loss pushes recall to the maximum while still keeping precision high, which explains why it becomes the best model under the F_2 criterion. However, such nearly perfect scores must be interpreted with care: they are obtained on one specific scene with clean glacier spectra and may overestimate performance in more challenging real-world conditions (other regions, sensors or illumination). In practice, validation on different regions would be needed before using these models operationally.



(a) Confusion matrix



(b) Precision–recall curve

Figure 3: Confusion matrix and precision–recall curve for the final glacier detector (logistic regression). The model attains perfect recall for glacier pixels with only a small number of false positives and a near-perfect precision–recall curve.

1.6 Conclusion

The analysis showed that hyperspectral land-cover classification is highly accurate when models account for class imbalance, strong inter-band correlations and high dimensionality. Among fourteen tuned models, linear SVM achieves the best pure macro-F1 and balanced accuracy, but multinomial logistic regression without PCA delivers almost identical performance while being much faster and simpler to deploy. Thus, in terms of the overall quality–cost trade-off posed in our research question, logistic regression without PCA is selected as the final classifier. Despite near-perfect performance (F1 > 0.99), results are based on a single scene with clean spectral signatures and may not generalize to other regions, sensors, or challenging conditions. Validation on diverse geographic areas is needed before operational deployment.

2 Feature Selection for High-Dimensional Neural Decoding: A Comparative Evaluation of Sparsity, Accuracy, and Runtime

2.1 Introduction

High-dimensional neural decoding is challenging because behavioural information is often weakly distributed across thousands of noisy, correlated neural features, producing extremely low marginal signal-to-noise. Classical assumptions of sparsity or dominant predictors rarely hold, and spike-count features introduce additional non-Gaussian noise and instability (Paninski, 2004). This motivates our central question:

Which feature selection method is most effective when information is globally distributed and each feature exhibits low signal-to-noise?

We study six feature selection methods spanning three families:

- **Wrapper**: Forward Stepwise Selection (FSS), Minimum Redundancy Maximum Relevance (mRMR)
- **Shrinkage-based**: Lasso, Elastic Net
- **Tree-based**: Random Forest Variable Importance, Gradient Boosting Variable Importance

These approaches capture different methods: linearity and sparsity (shrinkage), greedy joint modeling (wrapper), and non-linear interaction learning (trees). For feature selection methods hyperparameter grids, see Appendix section B.1

2.2 Problem Setup and Notation

The dataset contains spike-count recordings from mouse frontal cortex during a binary decision-making task (IBL et al., 2023), with $p = 11{,}190$ neuron–timebin features across $n = 683$ trials. We denote each trial by index $i = 1, \ldots, n$, with feature vector $x_i \in \mathbb{R}^p$ and binary behavioral label $y_i \in \{0, 1\}$. Let $X \in \mathbb{R}^{n \times p}$ denote the feature matrix and $y \in \{0, 1\}^n$ the vector of labels.

We evaluate model performance using **balanced accuracy**, which weights both classes symmetrically to account for class imbalance (30.5% / 69.5%).

We then construct **feature subsets** S_K using two approaches:

- *Ranking-based methods* (FSS, mRMR, RF-VI, GB-VI) produce an ordered ranking $r(1), \ldots, r(p)$, where $r(k)$ denotes the k-th most informative feature. For any $K \leq p$, we define $S_K = \{r(1), \ldots, r(K)\}$ and $X_{S_K} = X(:, S_K)$.
- *Shrinkage-based methods* (LASSO, Elastic Net) fit penalised logistic regression with parameters λ (LASSO) or (λ, α) (Elastic Net). For each regularisation parameter setting, the fitted model selects a subset S_K of features with non-zero coefficients. We vary penalty strength to obtain subsets of different sizes $K = |S_K|$.

We measure **runtime** as total time for feature selection plus evaluating BalAcc across all subset sizes K on the test set.

2.3 Evaluation Pipeline

Step 1: Optimal subset selection: For ranking-based methods, we evaluate subsets $K = 5, 10, 15, \ldots, 200$ using method-specific classifiers: logistic regression for FSS and mRMR, Random Forest for RF-VI, and Gradient Boosting for GB-VI. We define optimal subset size as

$$K^* = \arg\max_K \mathrm{BA}(S_K).$$

For LASSO and Elastic Net, we perform 5-fold cross-validation across the regularisation path. For each parameter setting, we obtain a subset S_K of non-zero features and define optimal subset size as

$$K^* = \arg\max_{S_K} \text{CV-BA}(S_K),$$

where CV-BA(S_K) is the cross-validated balanced accuracy for the subset of non-zero features S_K. We then fit the model with the K^* non-zero features on the training set and evaluate on the held-out test set for final balanced accuracy.

Step 2: Method ranking: We rank methods using a composite score:

$$\text{Score}(m) = w_K \times \text{K\_score}(m) + w_{\text{BalAcc}} \times \text{BalAcc\_score}(m) + w_{\text{RT}} \times \text{Runtime\_score}(m),$$

where $w_K = 0.45$, $w_{\text{BalAcc}} = 0.45$, $w_{\text{RT}} = 0.10$. Scores are normalised to $[0, 100]$. Weights reflect priorities: predictive accuracy and sparsity dominate, runtime is secondary. See Appendix B.2 for complete scoring procedure.

Step 3: Cross-classifier comparison: At each method's K^*, we evaluate four classifiers:

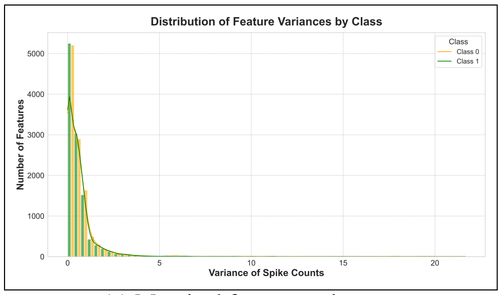
- **Logistic Regression**: `solver=liblinear`, `class_weight=balanced`
- **Linear SVM**: `kernel=linear`, `class_weight=balanced`
- **kNN**: $k = 10$
- **Random Forest**: `n_estimators=100`, `max_depth=5`, `class_weight=balanced`

We compute average balanced accuracy across classifiers for each feature selection method.

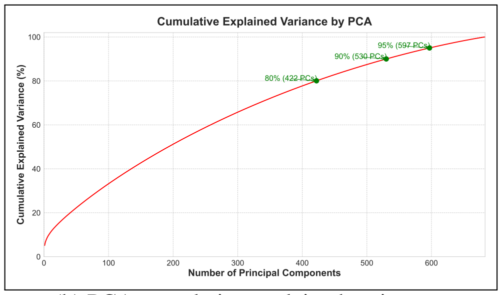
2.4 Exploratory Data Analysis

We extract the following insights from EDA:

- **High $p \gg n$ setting:** With $p/n \approx 16$, standard classifiers overfit without dimensionality reduction or regularisation. Feature selection is therefore essential for variance control and computational tractability.
- **Class imbalance:** The behavioural labels are skewed (Left: 30.5%, Right: 69.5%; Appendix Fig. 15). Since standard accuracy is misleading, balanced accuracy is used throughout.
- **Low signal-to-noise ratio:** Multiple indicators show that predictive signal is weak but distributed across many features:
  - **Sparse signals:** Many neuron–timebin features take near-zero values (Appendix Fig. 16, 17).
  - **Low-variance features:** 536 features have zero variance; $\sim$1,100 have variance ≤ 0.01; nearly 6,000 have variance ≤ 0.5 (Figure 4a).
  - **Distributed information:** PCA shows no clear elbow; 80%, 90%, and 95% variance require 422, 530, and 597 components (Figure 4b).
  - **Low inter-feature correlation:** Average $|\rho| < 0.05$ (Appendix Fig. 18), indicating weak but complementary signals across features.



(a) Marginal feature variance



(b) PCA cumulative explained variance

Figure 4: Exploratory analysis of variance structure: (a) marginal spike-count feature variance and (b) PCA cumulative explained variance.

Together, these properties imply that no trivially identifiable feature subset exists. Removing too many predictors risks discarding real signal, while retaining too many harms generalisation. We therefore expect the optimal subset size K^* for each method to lie in a mid-range rather than being extremely small or extremely large.

2.5 Results

Table 3 summarises the results from our analysis. FSS achieves competitive accuracy (0.774) with a highly compact subset of features ($K^* = 30$), while GB-VI produces a slightly larger subset ($K^* = 35$) with strong performance (0.770). Elastic Net achieves the highest BalAcc (0.805) but requires a substantially larger feature subset ($K^* = 2168$), which explains its lower overall ranking despite superior predictive performance. Notably, RF-VI, mRMR, and Lasso select $K^* = 70, 185, 30$ respectively and yield lower BalAccs (0.753, 0.634, 0.729); we examine possible explanations in section 2.6: Discussion. Appendix Figure 26 shows the trajectory each method follows to reach its optimal BalAcc across different values of K. Tree-based methods (RF-VI, GB-VI) complete feature selection and evaluation in <10 minutes, while wrapper methods (FSS, mRMR) incur runtimes >4.5 hours. Based on the composite scoring framework that balances accuracy, sparsity, and runtime, FSS achieves the highest overall score (84.2), followed by GB-VI (74.2).

Method	K^*	**BalAcc**	**RT (h)**	K^* **Score**	**BalAcc Score**	**RT Score**	**Total Score**
FSS	**30**	0.774	5.383	45.0	37.5	1.7	**84.2**
GB-VI	35	0.770	0.145	37.5	30.0	6.7	74.2
RF-VI	70	0.753	0.039	30.0	22.5	8.3	60.8
ENet	2168	**0.805**	1.617	7.5	45.0	5.0	57.5
Lasso	350	0.729	**0.022**	15.0	15.0	10.0	40.0
mRMR	185	0.634	4.534	22.5	7.5	3.3	33.3

Table 3: Comparison of feature selection methods. Weighted scores are computed by ranking K^* and runtime in ascending order, BalAcc in descending order, converting ranks to scores out of 100, then applying weights ($w_K = 0.45$, $w_{\text{BalAcc}} = 0.45$, $w_{\text{RT}} = 0.10$). See Appendix B.2 for complete scoring procedure

Using the optimal K^* from each method, we fitted four classifiers on the selected feature subsets. The results are summarised in Table 4. **GB-VI and FSS achieve the highest mean BalAcc** (0.737 and 0.727 respectively) across all four classifiers due to the compact, highly informative feature subsets these methods select ($K^* = 35$ and $K^* = 30$), which reduce model variance and enable stable generalisation across diverse classifier types.

Elastic Net exhibits strong performance on linear models but poor generalisation to non-linear classifiers. With $K^* = 2168$, it achieves the highest individual BalAcc (0.809) on LogReg and performs well on linear SVM (0.729), but degrades substantially on kNN (0.560) and Random Forest (0.617), yielding a mean BalAcc of only 0.679.

mRMR and Lasso achieve the lowest mean BalAccs (0.592 and 0.677). Both methods struggle in this low-correlation regime (average $|\rho| < 0.05$): with minimal feature redundancy, redundancy minimisation criterion (mRMR) and ℓ_1 penalty cannot exploit correlated groups to stabilise coefficient paths, resulting in suboptimal feature selection.

Method	K^*	LogReg	$\text{SVM}_{\text{linear}}$	kNN	RF	Avg BalAcc
ENet	2168	**0.809**	0.729	0.560	0.617	0.679
FSS	30	0.774	**0.751**	0.675	0.708	0.727
GB-VI	35	0.770	0.743	**0.697**	0.737	**0.737**
Lasso	350	0.699	0.707	0.627	0.676	0.677
RF-VI	70	0.699	0.703	0.641	**0.753**	0.699
mRMR	185	0.634	0.621	0.552	0.559	0.592

Table 4: Balanced accuracy of four classifiers trained on each method's optimal feature subset.

2.6 Discussion

Forward Stepwise Selection (FSS) achieves the highest total score in our ranking (84.2) with the smallest feature subset ($K^* = 30$), high BalAcc on the test set (0.774), and strong generalisation across classifiers (mean BalAcc = 0.727). It's strong predictive performance stems from its greedy AIC-driven search: at each step, it adds the feature that minimizes AIC, which approximates expected test risk under log-loss. This joint optimization of prediction and feature selection explains why FSS discovers the "first layer" of neural features with strongest marginal contributions; AIC decreases sharply then plateaus around $K = 30$ (Appendix Fig. 19), effectively isolating the most signal-dense features. *However, FSS has limitations:* it becomes computationally prohibitive (runtime >5 hrs), and the plateau indicates that once the strongest marginal features are selected, the greedy search becomes unstable and fails to discover additional weak but collectively informative features. This makes FSS ideal for finding a minimal feature subset for interpretability but limited when higher BalAcc is the priority.

Gradient Boosting Variable Importance (GB-VI) achieves the second-best total score (74.2): compact subset ($K^* = 35$), high test BalAcc (0.770), and highest avg BalAcc across classifiers (0.737). Unlike RF, which builds deep trees in parallel and averages predictions, GB builds shallow trees sequentially, fitting each to the residuals of the previous ensemble. This concentrates importance on genuinely informative features: GB achieves 80% cumulative importance with only 143 features (Appendix Fig. 24), compared to 796 for RF (Appendix Fig. 23). This accumulation of weak, local improvements aligns well with distributed weak-signal datasets where no single feature is strongly predictive but many small increments collectively matter. Runtime is highly efficient (<10 minutes), making GB-VI a strong competitor to FSS when both sparsity and balanced accuracy are priorities.

Random Forest Variable Importance (RF-VI) achieves modest performance ($K^* = 70$, BalAcc = 0.753) but low mean BalAcc across classifiers (0.699). Two issues limit this method. First, impurity-based splits are biased toward high-variance features, which can be noisy rather than informative. In our dataset, where most features have low variance ($\sim$6,000 features with variance ≤ 0.5), this bias causes RF-VI to favour noisy, high-variance features over weakly informative predictors. The cumulative feature importance curve confirms this: RF requires 796 features to reach 80% cumulative importance (Appendix Fig. 23). Second, in our $p \gg n$ setting, each tree overfits severely with terminal nodes containing very few observations. Although ensemble averaging reduces variance, aggregated importance rankings are unreliable as individual trees are fundamentally unstable.

Elastic Net achieves the highest single-classifier BalAcc (0.805 on LogReg) by retaining a large feature subset ($K^* = 2168$). CV selects $\alpha = 0.2$ and $C = 0.45$, (ℓ_2-dominant). It also performs well on linear SVM (0.729), but fails to generalise to other classifiers. kNN (BalAcc = 0.560) suffers from the curse of dimensionality, and Random Forest (BalAcc = 0.617) struggles with variance instability. As a result, avg BalAcc across classifiers (0.679) is modest despite superior linear model performance. Runtime (1.6 hrs) is less than FSS and mRMR due to a coarse grid search; finer tuning would increase runtime, yet even with coarse search Elastic Net yields the highest BalAcc.

LASSO ranks second-last with a large feature subset ($K^* = 350$) and BalAcc = 0.729. While it performs reasonably on logistic regression, it generalises poorly to other classifiers (mean BalAcc = 0.677). In typical high-correlation settings, LASSO struggles because the ℓ_1 penalty arbitrarily selects one feature from correlated groups. However, with low inter-feature correlation (average $|\rho| < 0.05$), LASSO can retain more features without redundancy penalties. Despite this, LASSO imposes hard sparsity which discards weakly informative features that collectively contribute to prediction. In a distributed weak-signal regime, this aggressive shrinkage eliminates signal prematurely, explaining why LASSO underperforms relative to Elastic Net.

Minimum Redundancy Maximum Relevance (mRMR) achieves the lowest total score (33.3) and ranks last: $K^* = 185$ features that prove least informative (BalAcc = 0.634, lowest avg BalAcc = 0.592). This poor performance stems from two structural mismatches with our dataset. First, mRMR relies on mutual information (MI) estimation between features and labels. With $p \gg n$, MI estimates are highly unstable; the top 200 ranked features show shallow decay with nearly indistinguishable MI values (Appendix Fig. 20), leading to noisy feature rankings. Second, mRMR's core principle of minimising redundancy offers no advantage where features exhibit low mutual correlation. When features are largely independent, redundancy minimisation becomes irrelevant, and mRMR degenerates into a purely relevance-based ranking. Additionally, the computational runtime exceeds 4.5 hours, making mRMR entirely unsuitable for our dataset.

2.6.1 Classifier Performance Across Feature Subsets

Feature selection method matters more than classifier choice. Compact, informative subsets enable stable generalization, while large subsets restrict performance to regularized linear models. Methods fall into three groups:

Compact and highly informative (FSS: $K^* = 30$, GB-VI: $K^* = 35$): Methods effectively distinguish signal from noise, producing small subsets that generalize well across all classifiers (avg BalAcc ≥ 0.72).

Moderately compact, mixed informativeness (RF-VI: $K^* = 70$, mRMR: $K^* = 185$): RF-VI achieves modest avg BalAcc (0.70) with relatively weak but compact features, while mRMR's larger, uninformative subset yields the lowest mean BA (0.59).

Large subsets with variable informativeness (LASSO: $K^* = 350$, Elastic Net: $K^* = 2168$): LASSO's features prove weakly informative (avg BalAcc = 0.68), while Elastic Net's dense but informative subset excels on linear models (LogReg: 0.81, SVM: 0.73) but degrades on kNN (0.56) and RF (0.62) due to curse of dimensionality, yielding avg BalAcc = 0.68.

2.7 Conclusion

We sought to identify the best feature selection method for high-dimensional neural decoding with weakly correlated features and sparse signals. Performance depends critically on how each method identifies important features in this setting. FSS and GB-VI succeed by identifying compact, highly predictive subsets that generalize well (FSS via greedy AIC minimization, GB-VI via sequential residual fitting). RF-VI underperforms due to bias toward high-variance features and tree overfitting. mRMR fails entirely as redundancy minimization offers no benefit when features are weakly correlated. Among shrinkage methods, LASSO's hard sparsity discards weak signals prematurely, while Elastic Net retains a large informative subset achieving highest BA on linear models (0.805) but failing on distance-based classifiers. Depending on priorities—interpretability (FSS), generalization (GB-VI), or peak linear performance (Elastic Net)—practitioners can select accordingly.

However, our analysis relies on a single train-test split, coarse grid search, and basic class imbalance handling (`class_weight=balanced`), yielding a performance ceiling of BalAcc = 0.805. Further analysis can incorporate repeated cross-validation for stable performance estimates, finer hyperparameter tuning, advanced resampling techniques (SMOTE, ensemble methods), feature stability analysis via bootstrapping, and domain-aware feature engineering (temporal smoothing, neuron interactions) or non-linear models (kernel SVM, neural networks) to push beyond the current ceiling.

References

Breiman L. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

Friedman J. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

Guanter L, Kaufmann H, Segl K, Foerster S, Rogass C, Chabrillat S, et al. The EnMAP Spaceborne Imaging Spectroscopy Mission for Earth Observation. *Remote Sensing*, 7:8830–8857, 2015.

International Brain Laboratory, Carandini M, Harris KD, Steinmetz NA, Coen P. A brain-wide map of neural activity during decision-making. *Nature*, 619:712–719, 2023.

Paninski L. Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15(4):243–262, 2004.

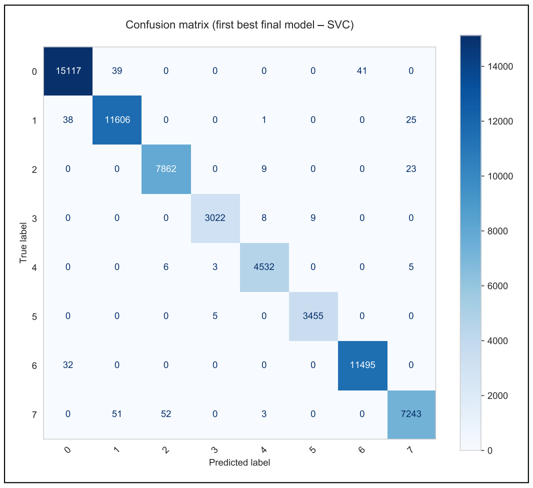
Peng H, Long F, Ding C. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

Tibshirani R. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

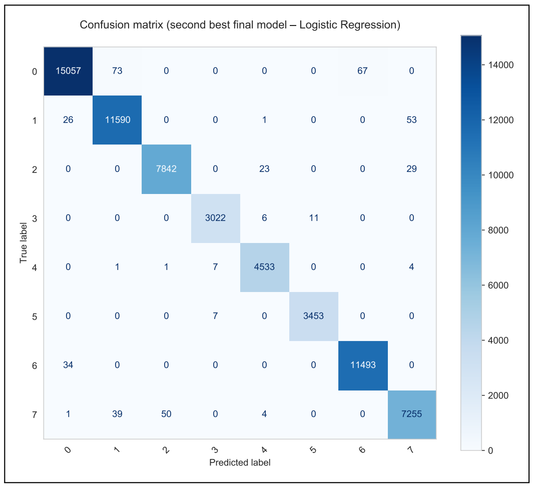
Zou H, Hastie T. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320, 2005.

A Appendix A: Hyperspectral Land-Cover and Glacier-Ice Classification

A.1 Confusion Matrices and Results



(a) SVM



(b) Logistic regression

Figure 5: Confusion matrices for the SVM and logistic regression classifiers on the 8-class land-type task. Both achieve near-perfect classification, with residual errors mainly between similar vegetation classes.

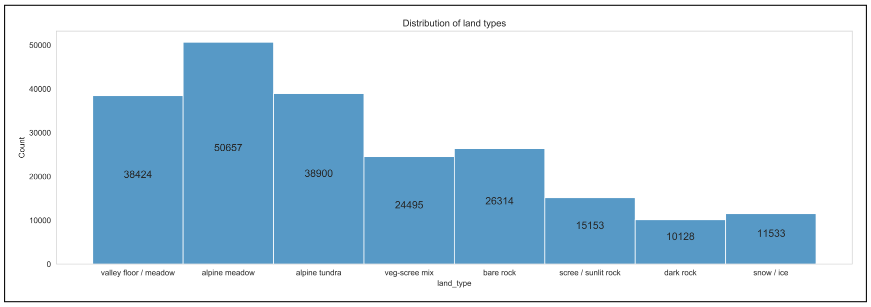


Figure 6: Class frequencies in the dataset, showing a strong imbalance: alpine meadow and alpine tundra dominate, while dark rock and snow/ice are much smaller classes. This imbalance motivates the use of macro-F1 and balanced accuracy instead of plain accuracy.

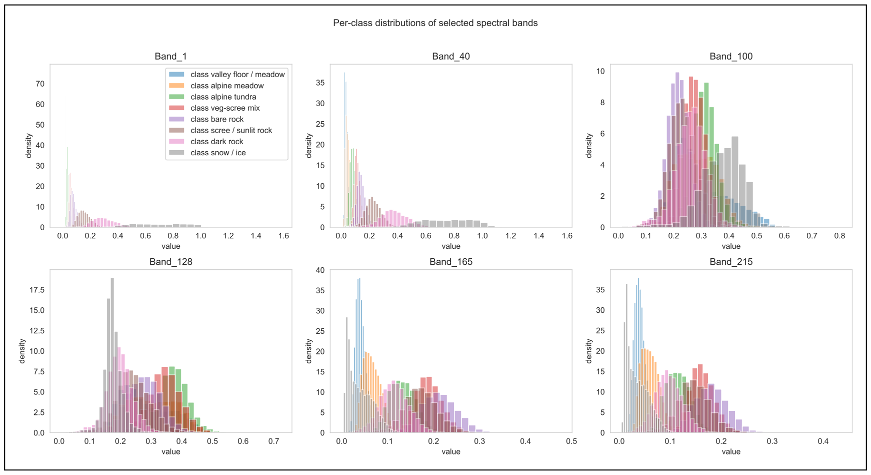


Figure 7: Distributions of several spectral bands for each land type. Some bands show clear shifts between classes, while others overlap, suggesting that the task is partly but not perfectly separable.

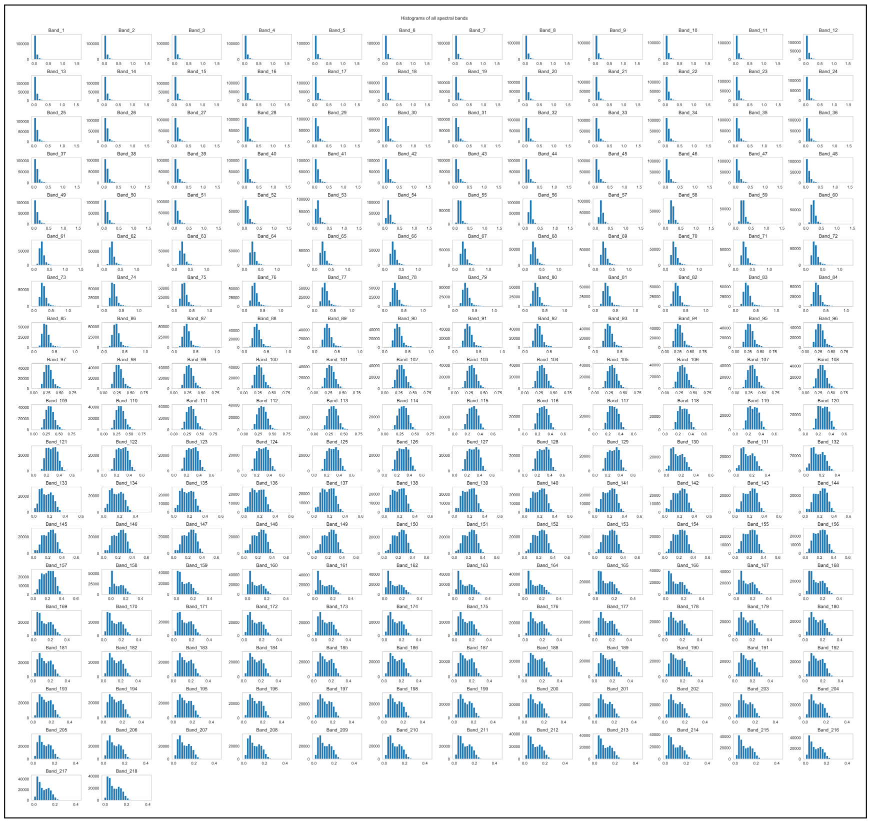


Figure 8: Histograms of all 218 spectral bands. Many bands have very similar distribution shapes, reflecting redundancy across neighbouring wavelengths; this is consistent with the strong inter-band correlations observed in the full EDA.

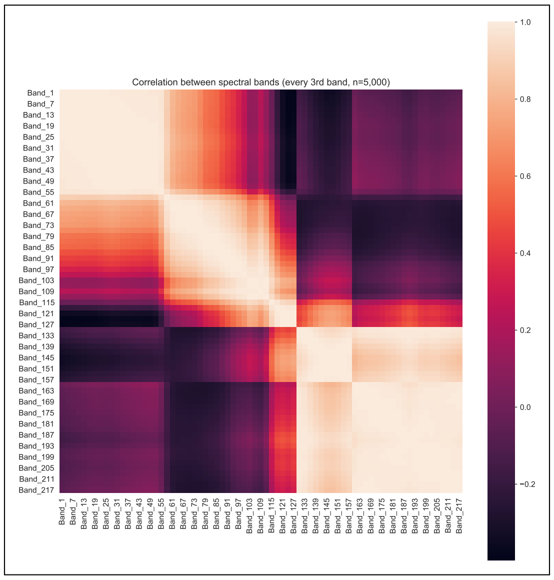


Figure 9: Correlation heatmap for every third spectral band (n = 5,000 pixels). Neighbouring bands are highly correlated, confirming strong multicollinearity and motivating the use of PCA or regularised classifiers.

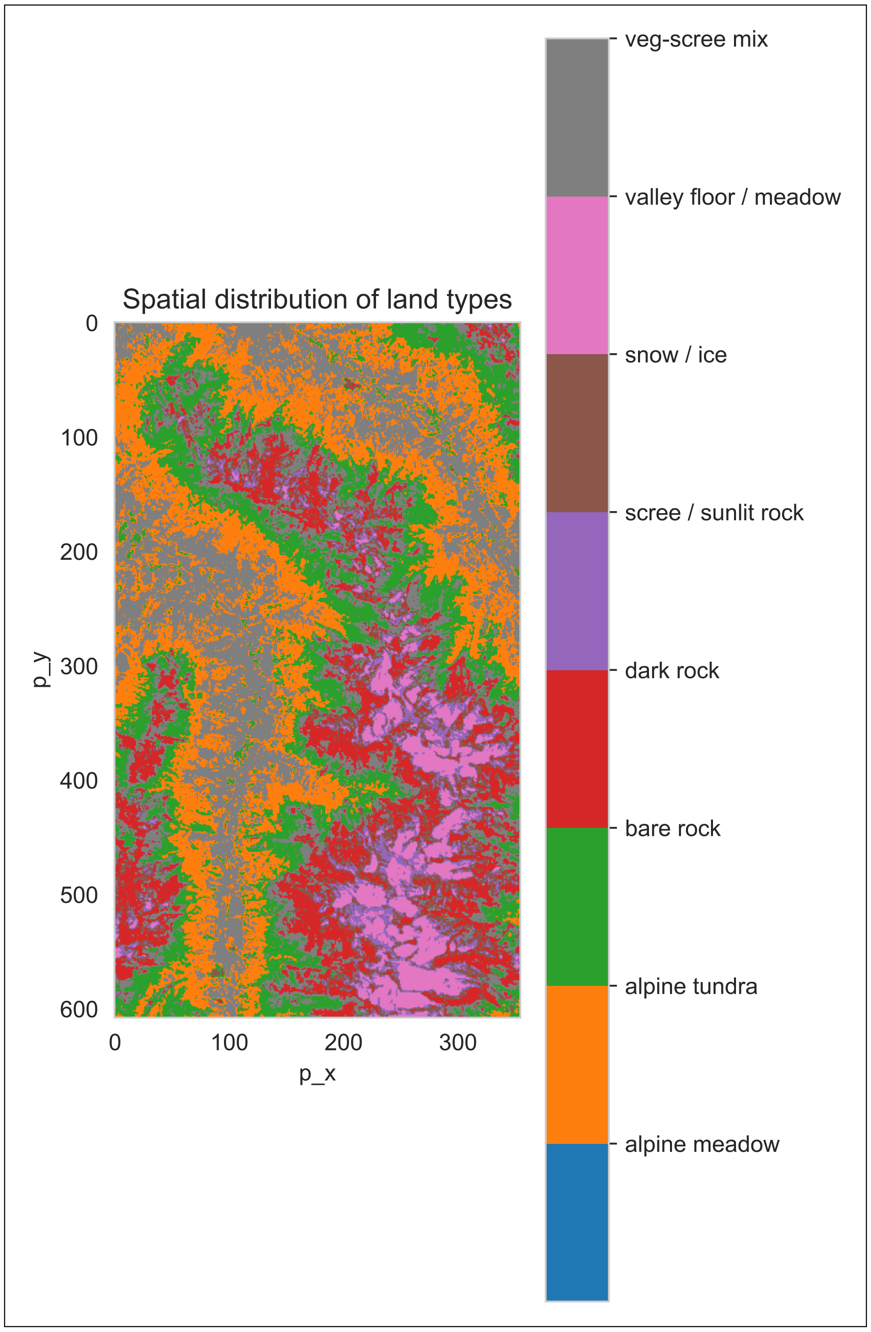


Figure 10: Spatial distribution of land types in the scene. The strong spatial clustering shows why pixel coordinates cannot be used as features—models would overfit to this specific geometry rather than learn spectral differences.

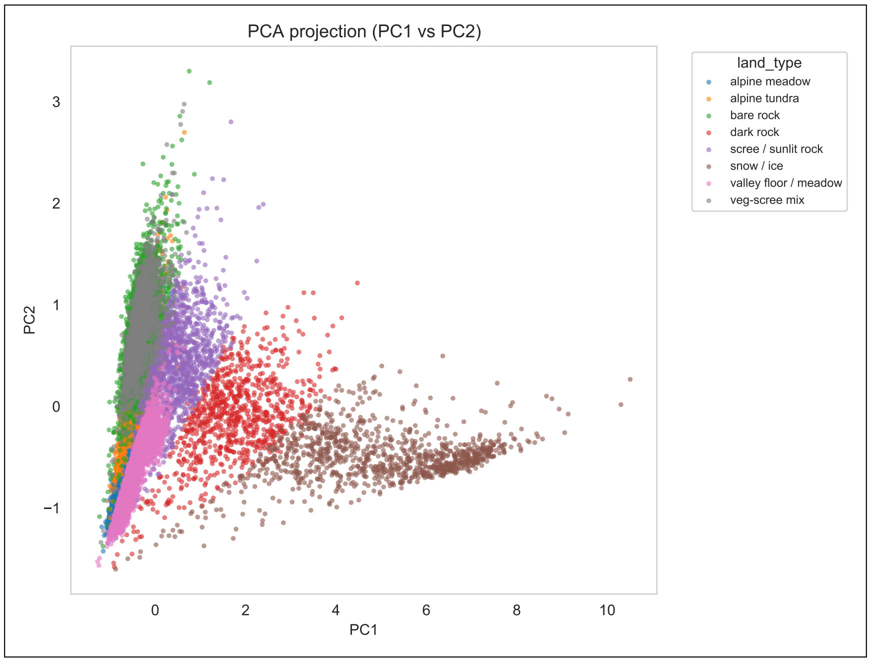


Figure 11: PCA projection onto the first two components. Several land types form partially separated clusters, confirming that the spectral features contain useful discriminative structure despite some overlap.

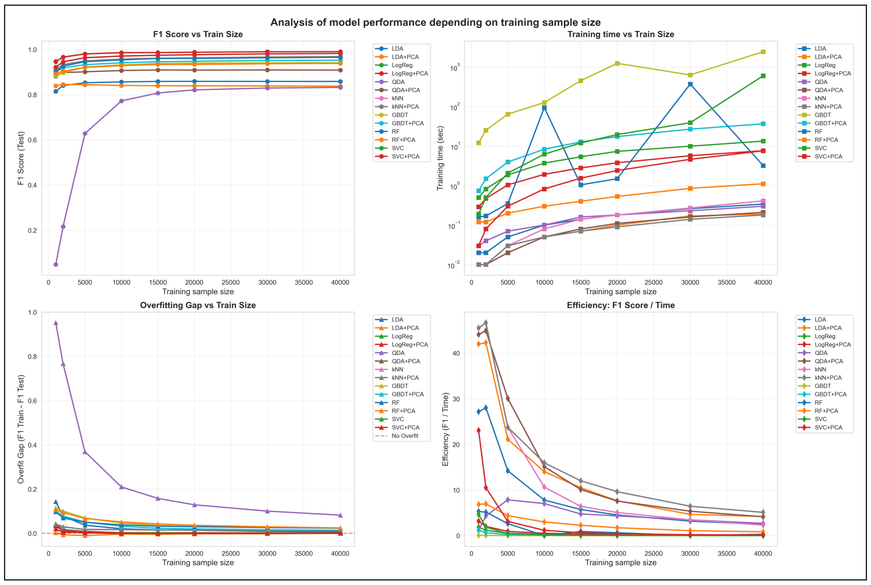


Figure 12: Effect of training sample size on model performance. The plot shows test macro-F1, training time, and overfitting gap as functions of training size. Performance curves flatten after approximately 10,000 observations, justifying our choice of $n = 10{,}000$ for hyper-parameter tuning.

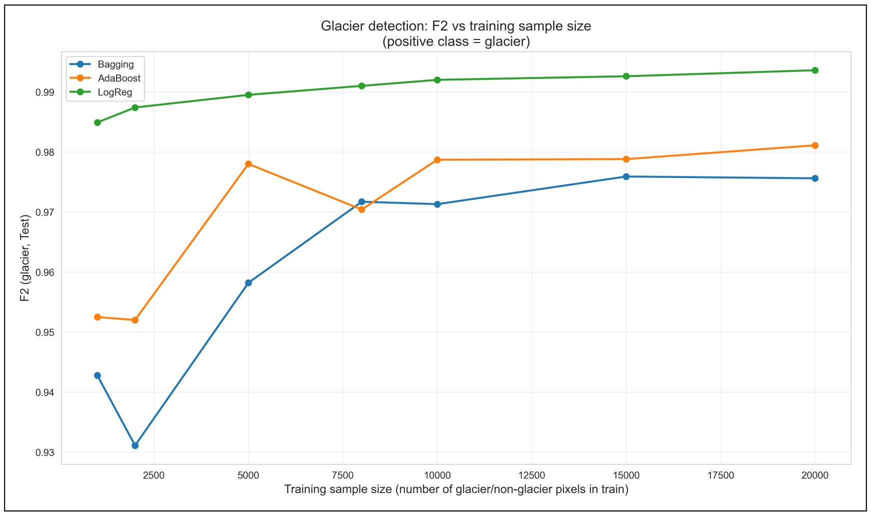
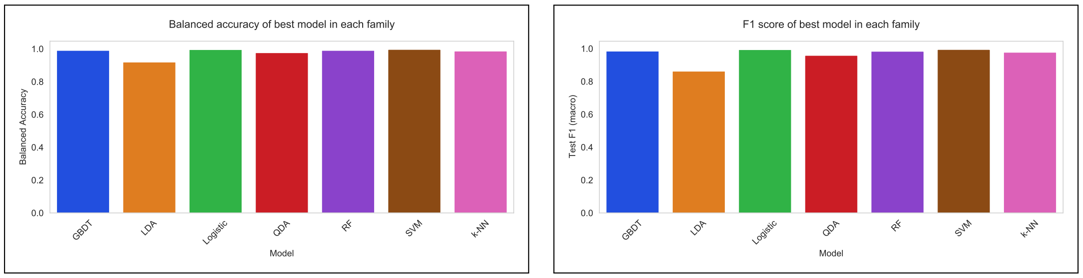


Figure 13: The plot shows test F_2 as a function of training size. The curves flatten around $n \approx$ 10,000, which supports using $n =$ 10,000 for hyper-parameter tuning.



(a) balanced accuracy

(b) F1-score

Figure 14: Balanced accuracy and macro-F1 of the best model in each classifier family. SVM and Logistic Regression achieve the highest scores, while LDA clearly underperforms, consistent with EDA and model assumptions.

A.2 Hyperparameter Grid Details

LDA / LDA + PCA(10): High dimension combined with multicollinearity motivates using `solver='lsqr'` with `shrinkage='auto'` to stabilise covariance estimates.

Logistic Regression: We use the `lbfgs` solver with L2 penalty, tuning only C and `class_weight`. Tuning the solver or elastic-net `l1_ratio` would increase grid size without improving statistical properties.

QDA: We tune only `reg_param` and compare empirical versus uniform priors, reflecting the tension between class imbalance and macro-F1. PCA(10) makes QDA much more stable.

k-NN: We restrict $k \in \{3, 5, 7, 9, 11\}$ (odd integers to avoid ties), compare Euclidean and Manhattan distances, and two weighting schemes (`uniform`, `distance`).

Gradient Boosting: We tune `n_estimators` $\in \{150, 300\}$, `learning_rate` $\in \{0.05, 0.1\}$, `max_depth` $\in \{3, 5\}$, and `subsample` $\in \{1.0, 0.7\}$.

Random Forest: We vary the number of trees, maximum depth, feature subsampling (`sqrt` or `log2`), and minimum leaf size in a small range appropriate for 10k observations.

SVM: We compare linear and RBF kernels, tuning C and γ (for RBF). The best models use a linear kernel with moderate C, consistent with classes being largely linearly separable after scaling.

B Appendix B: Feature Selection for High-Dimensional Neural Decoding

B.1 Feature Selection Methods: Implementation Details

We evaluate six feature selection methods: wrapper (FSS, mRMR), shrinkage-based (LASSO and Elastic Net), and tree-based embedded approaches (Random Forest and Gradient Boosting Variable Importances). Below we provide the mathematical formulations for each method.

Forward Stepwise Selection is a greedy wrapper method that iteratively adds the feature that yields the largest decrease in Akaike Information Criterion (AIC). At step k, we choose

$$j_k = \arg\min_{j \notin S_{k-1}} \mathrm{AIC}(S_{k-1} \cup \{j\}),$$

where $\mathrm{AIC}(S)$ is computed from a logistic regression fitted on subset S. FSS does not require hyperparameter tuning. We evaluate up to $K_{\text{max}} = 200$ features.

Minimum Redundancy Maximum Relevance (mRMR) ranks features by maximising mutual-information relevance while minimising redundancy:

$$j_k = \arg\max_{j \notin S_{k-1}} \big[I(x_j; y) - \tfrac{1}{|S_{k-1}|} \sum_{\ell \in S_{k-1}} I(x_j; x_\ell) \big].$$

mRMR is deterministic and ranks up to $K_{\text{max}} = 1500$ features.

LASSO performs feature selection via ℓ_1-penalised logistic regression,

$$\hat{\beta} = \arg\min_{\beta} \big(-\ell(\beta; X, y) + \lambda \|\beta\|_1 \big).$$

We tune $C = 1/\lambda$ via 5-fold cross-validation over 50 logarithmic values in $[10^{-3}, 10^{1}]$.

Elastic Net combines ℓ_1 and ℓ_2 penalties:

$$\hat{\beta} = \arg\min_{\beta} \big(-\ell(\beta; X, y) + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 \big).$$

We tune $\alpha \in \{0.2, 0.8\}$ and $C \in \{0.05, 0.10, \ldots, 0.50\}$ via 5-fold cross-validation.

Random Forest Variable Importance (RF-VI) ranks features using impurity reduction across trees:

$$\mathrm{VI}_{\mathrm{RF}}(j) = \sum_{t=1}^{T} \sum_{\text{splits } s \text{ on } j} \Delta\mathrm{Gini}(s).$$

We tune via 5-fold CV: `n_estimators` $\in \{100, 200, 300, 400\}$, `max_depth` $\in \{5, 10, 20, \text{None}\}$, `min_samples_leaf` $\in \{1, 2, 5\}$, `max_features` $\in \{\text{sqrt}, \text{log2}\}$.

Gradient Boosting Variable Importance (GB-VI) computes importance from the total loss reduction contributed by feature j:

$$\mathrm{VI}_{\mathrm{GB}}(j) = \sum_{t=1}^{M} \sum_{\text{splits } s \text{ on } j} \Delta\mathcal{L}_t(s).$$

We tune via 5-fold CV: `n_estimators` $\in \{100, 200\}$, `learning_rate` $\in \{0.05, 0.1\}$, `max_depth` $\in \{3, 4\}$, `subsample` $\in \{0.8, 1.0\}$.

B.2 Scoring Framework

To evaluate feature selection methods, we combine three criteria: optimal subset size (K^*), balanced accuracy (BalAcc), and runtime. The scoring procedure consists of three steps:

Step 1: Ranking. For each criterion, we rank all $n = 6$ methods:

- K^* and runtime: ranked in ascending order (lower is better)
- Balanced accuracy: ranked in descending order (higher is better)

Ranks are assigned using the minimum method (ties receive the same rank).

For example, FSS with $K^* = 30$ receives rank 1 in sparsity (most compact), BalAcc = 0.774 receives rank 2 (second-highest), and runtime = 5.38 hrs receives rank 6 (slowest).

Step 2: Score conversion. Each rank is converted to a score out of 100:

$$\text{Score} = \frac{n + 1 - \text{Rank}}{n} \times 100$$

For FSS: K^* rank 1 $\rightarrow$ score = $\frac{6+1-1}{6} \times 100 = 100$; BalAcc rank 2 $\rightarrow$ score = $\frac{6+1-2}{6} \times 100 = 83.33$; runtime rank 5 $\rightarrow$ score = $\frac{6+1-6}{6} \times 100 = 16.67$.

Step 3: Weighted combination. Individual scores are weighted and summed:

$$\text{Total Score}(m) = w_K \cdot \text{K\_score}(m) + w_{\text{BA}} \cdot \text{BA\_score}(m) + w_{\text{RT}} \cdot \text{RT\_score}(m)$$

with weights $w_K = 0.45$, $w_{\text{BA}} = 0.45$, $w_{\text{RT}} = 0.10$.

For FSS: Total Score = $0.45(100) + 0.45(83.33) + 0.10(16.67) = 45 + 37.5 + 1.67 = 84.2$. The final total score determines overall method ranking.

B.3 Exploratory Data Analysis

B.3.1 Class Distribution

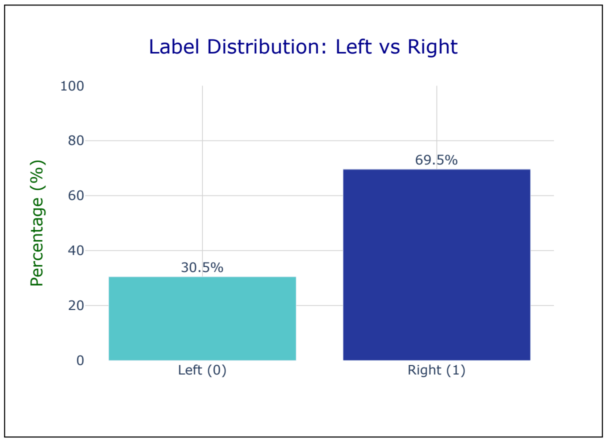


Figure 15: Class distribution showing 30.5% Left (0) and 69.5% Right (1). This imbalance motivates the use of balanced accuracy as the primary evaluation metric.

B.3.2 Feature Distribution: First 25 Features

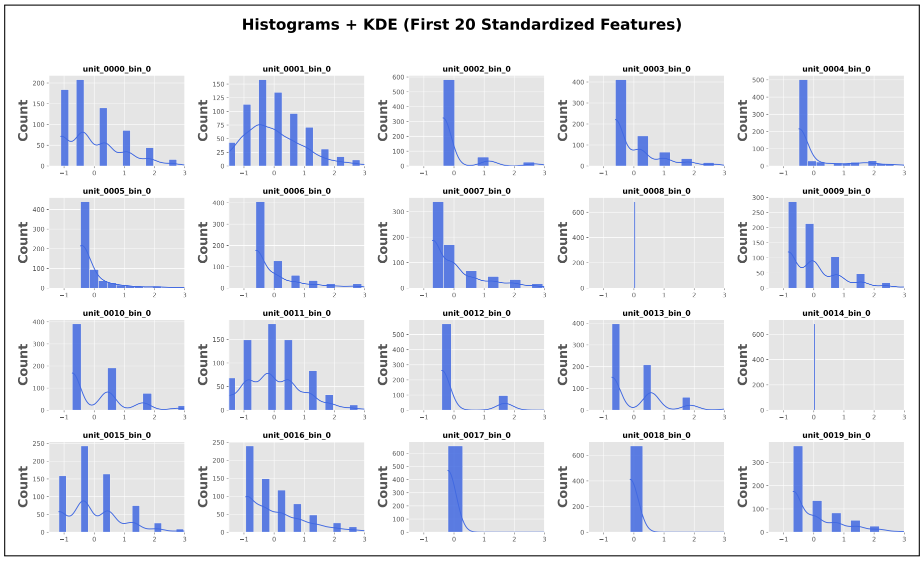


Figure 16: Histograms with KDE overlays for the first 25 standardised features. Most features exhibit right-skewed distributions with a dominant mode near zero, reflecting sparse neural firing patterns typical of spike-count data.

B.3.3 Feature Distribution: Last 25 Features

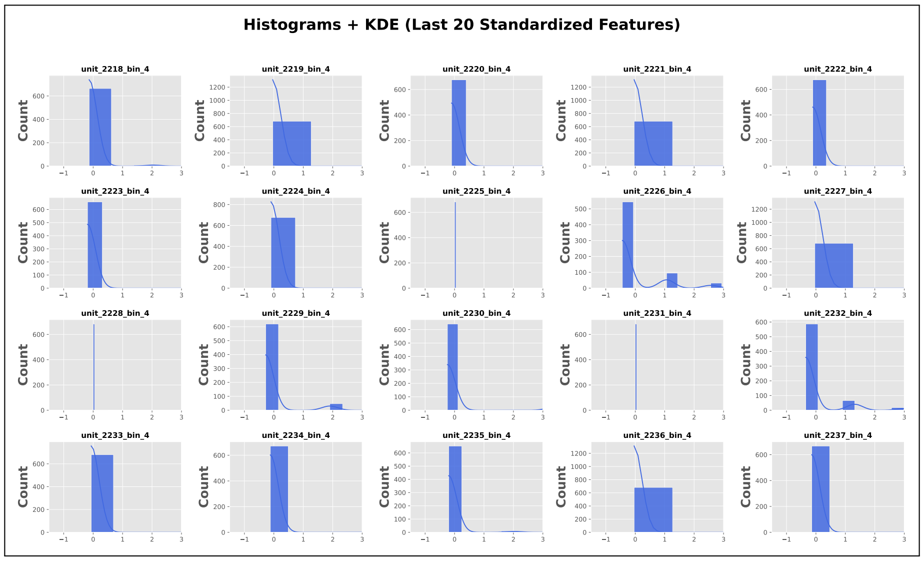


Figure 17: Histograms with KDE overlays for the last 25 standardised features. These later time-bin features show even stronger zero-inflation and reduced variance compared to earlier bins, suggesting weaker signal in later temporal windows.

B.3.4 Feature Correlation Heatmap

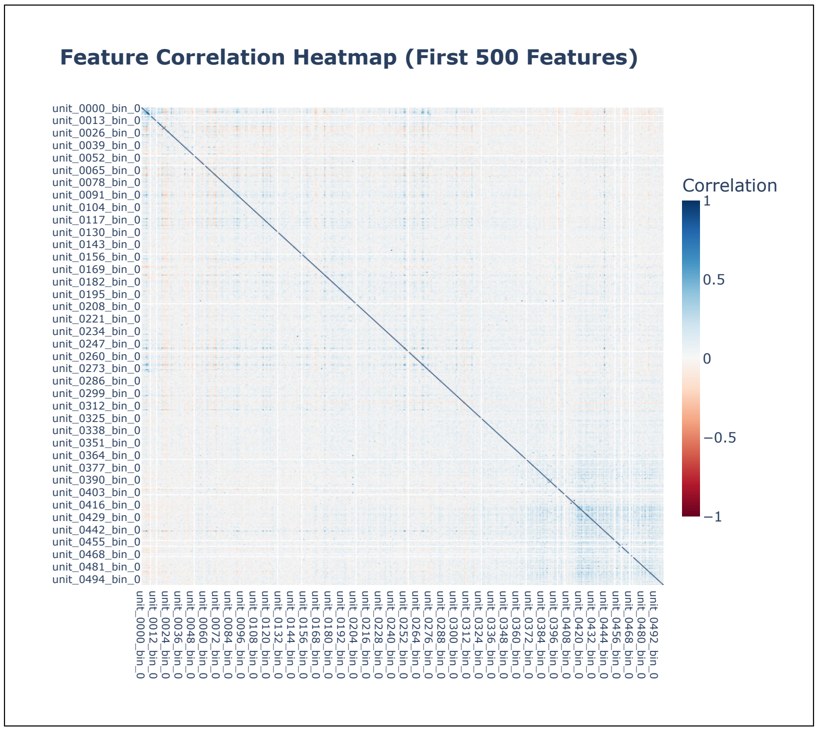


Figure 18: Correlation matrix of the first 500 features. Off-diagonal values are uniformly low (average $|\rho| = 0.046$), indicating weak inter-feature dependence. This low correlation structure explains why LASSO retains more features ($K^* = 212$) than typically expected and why mRMR's redundancy minimisation offers little benefit.

B.4 Feature Selection Performance by Method

B.4.1 Forward Stepwise Selection: AIC vs K

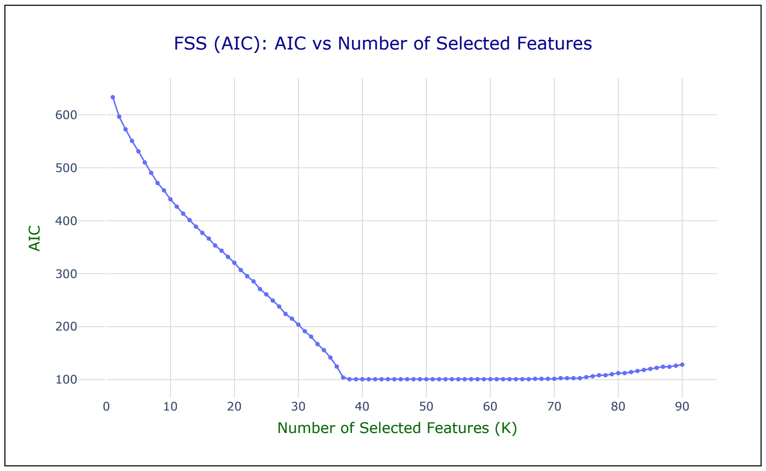


Figure 19: AIC versus number of selected features K for Forward Stepwise Selection. AIC decreases sharply in early steps, then plateaus around $K = 30$, indicating diminishing returns from adding additional features.

B.4.2 mRMR: Top 200 Feature Importances

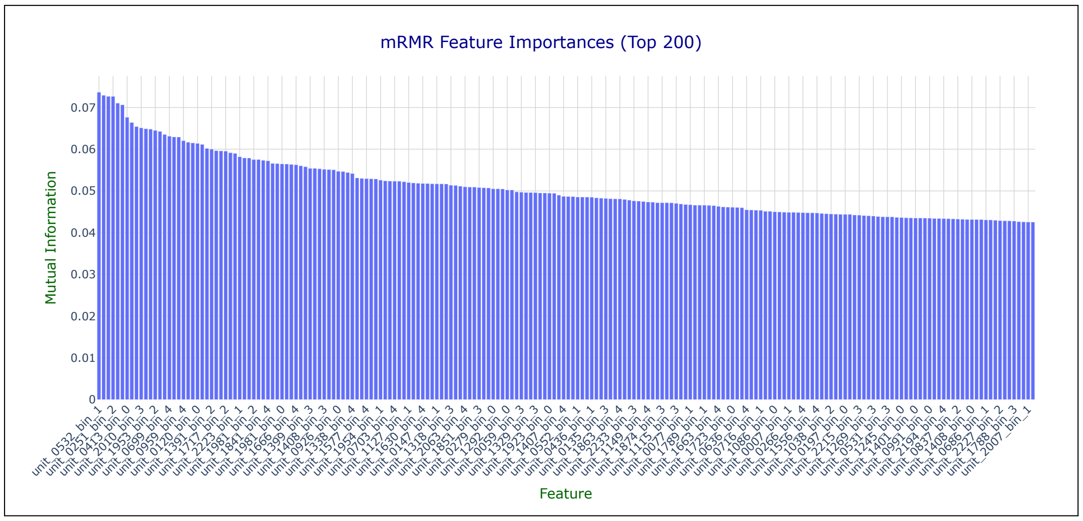
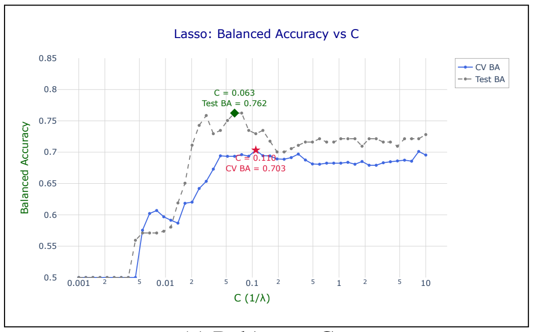
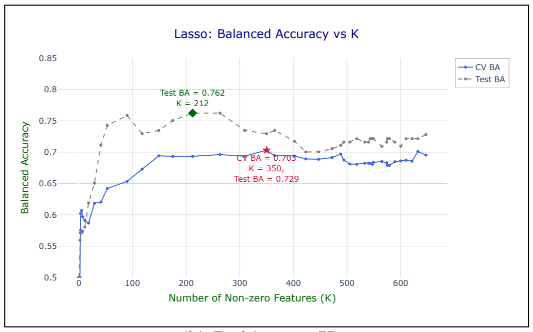


Figure 20: Mutual-information–based importance scores for the top 200 mRMR-ranked features. The shallow decay and nearly indistinguishable MI values reflect unstable estimation in the $p \gg n$ regime (11,190 features, 683 samples), leading to poor feature rankings and the lowest balanced accuracy (0.69) among all methods.

B.4.3 LASSO: Regularisation Path



(a) BalAcc vs C



(b) BalAcc vs K

Figure 21: LASSO regularisation path showing (a) balanced accuracy vs regularisation parameter C and (b) balanced accuracy vs number of non-zero features K. CV-optimal configuration selects $K^* = 350$ features, achieving BalAcc = 0.729 on the test set. Note that test BA peaks earlier at $K = 212$ (BalAcc = 0.74), but we avoid this to prevent data leakage.

B.4.4 Elastic Net: Regularisation Path

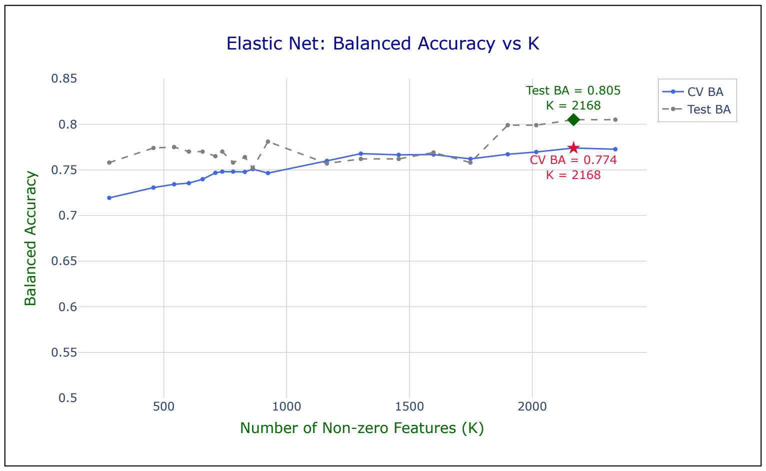


Figure 22: Elastic Net balanced accuracy vs K for different values of mixing parameter α. CV-optimal configuration ($\alpha = 0.2$, $C = 0.45$) selects $K^* = 2168$ features where both CV BA and test BA peak (test BalAcc = 0.806). Limited hyperparameter tuning contributes to this alignment.

B.4.5 Random Forest: Cumulative Feature Importance

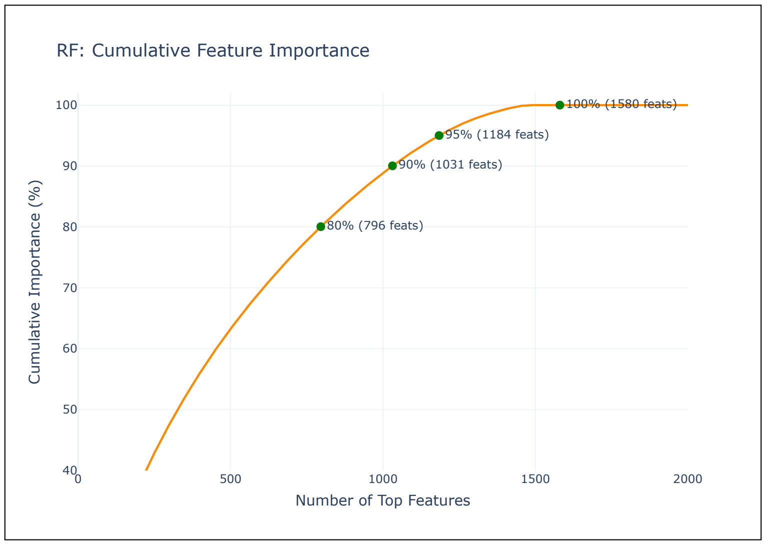


Figure 23: Cumulative feature importance for Random Forest. To reach 80% cumulative importance, RF requires 796 features, indicating diffuse importance distribution across many noisy features due to impurity-based splitting bias.

B.4.6 Gradient Boosting: Cumulative Feature Importance

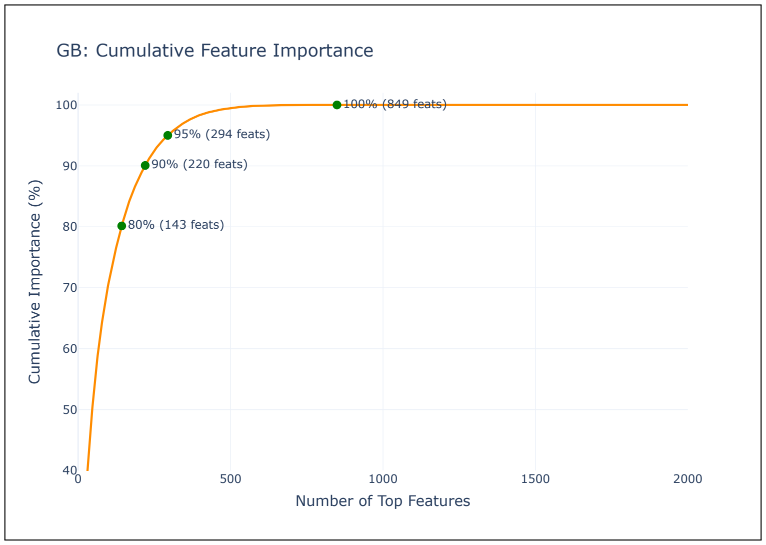
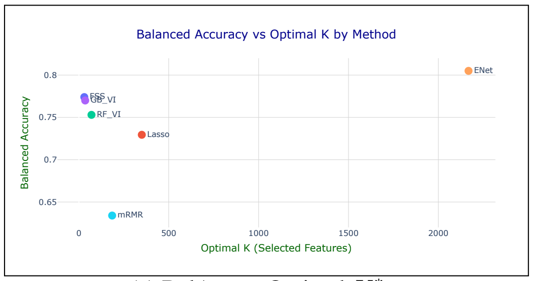


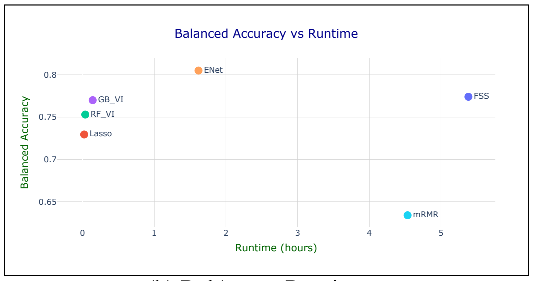
Figure 24: Cumulative feature importance for Gradient Boosting. GB achieves 80% cumulative importance with only 143 features, demonstrating that sequential residual-fitting concentrates importance on genuinely informative features. This explains GB's superior performance with compact subsets compared to RF

B.5 Feature Selection Evaluation

B.5.1 Balanced Accuracy vs Optimal K^* and Runtime



(a) BalAcc vs Optimal K^*



(b) BalAcc vs Runtime

Figure 25: (a) Balanced accuracy versus optimal subset size K^* for all methods. FSS and GB-VI achieve high BalAcc with compact subsets ($K^* < 40$), while Elastic Net requires over 2000 features to attain the highest BalAcc. (b) Balanced accuracy versus runtime (hrs). Tree-based methods (RF-VI, GB-VI) and LASSO complete in under 10 minutes, while wrapper methods (FSS, mRMR) require over 4 hours due to iterative model refitting.

B.5.2 Balanced Accuracy vs K by Selection Paradigm

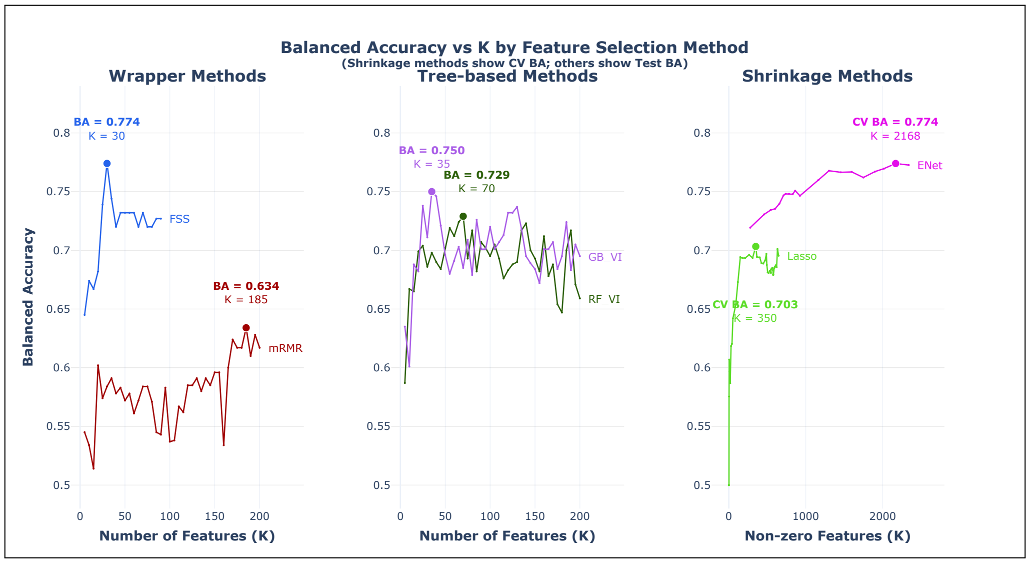


Figure 26: Balanced accuracy versus K grouped by selection paradigm: (left) wrapper methods, (center) tree-based methods, (right) shrinkage methods. FSS peaks early then plateaus; tree-based methods (RF-VI, GB-VI) show similar early peaks with more variance; shrinkage methods (LASSO, Elastic Net) require larger K to reach optimal BalAcc, with Elastic Net continuing to improve up to $K^* = 2168$. Shrinkage methods display CV balanced accuracy; all others show test balanced accuracy.

B.5.3 Model Performance: Weighted Scoring Framework

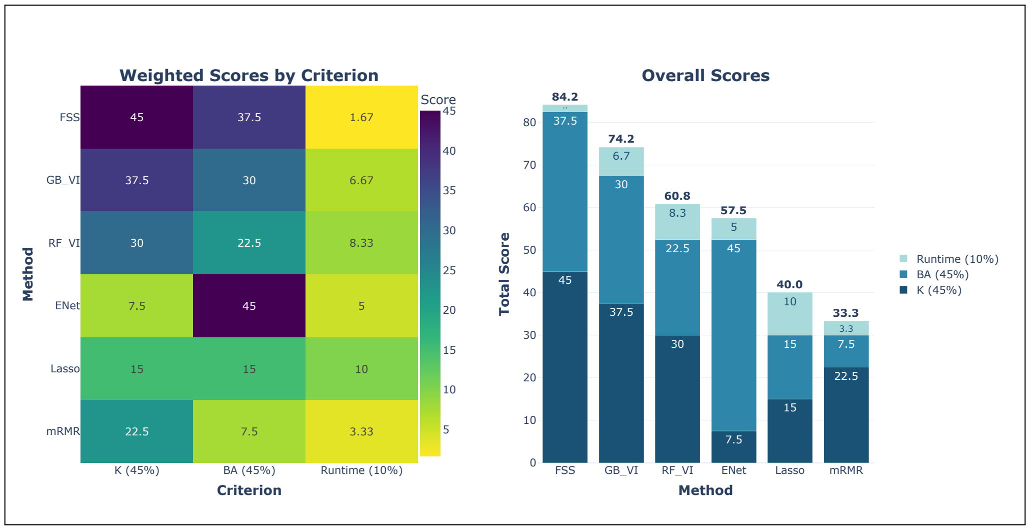


Figure 27: (Left) Weighted scores by criterion: balanced accuracy (45%), sparsity/K* (45%), and runtime (10%). (Right) Overall composite scores showing FSS ranks highest (84.2) due to excellent sparsity ($K^* = 30$) and strong balanced accuracy (0.774), while mRMR ranks lowest (33.3) due to poor balanced accuracy (0.634) despite moderate sparsity. The composite framework reflects the priorities of interpretability and prediction quality.

B.5.4 Balanced Accuracy by Method and Classifier

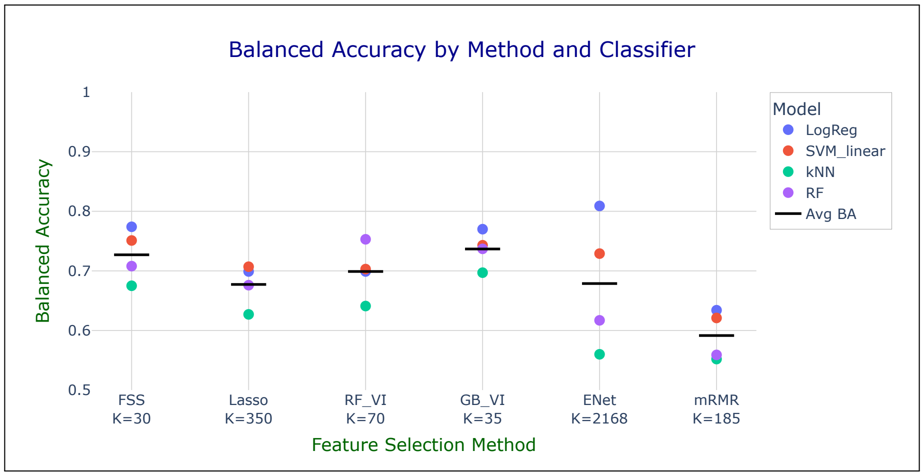


Figure 28: Balanced accuracy for four classifiers (Logistic Regression, SVM, kNN, Random Forest) trained on optimal feature subsets from each method. FSS and GB-VI achieve highest mean BalAcc (0.727 and 0.737) across classifiers due to compact, informative feature subsets. Elastic Net performs well only on linear models (LogReg: 0.805, SVM: 0.729) but poorly on kNN (0.560) and RF (0.617) due to curse of dimensionality with $K^* = 2168$ features.

B.6 Top-200 selected feature overlap across methods

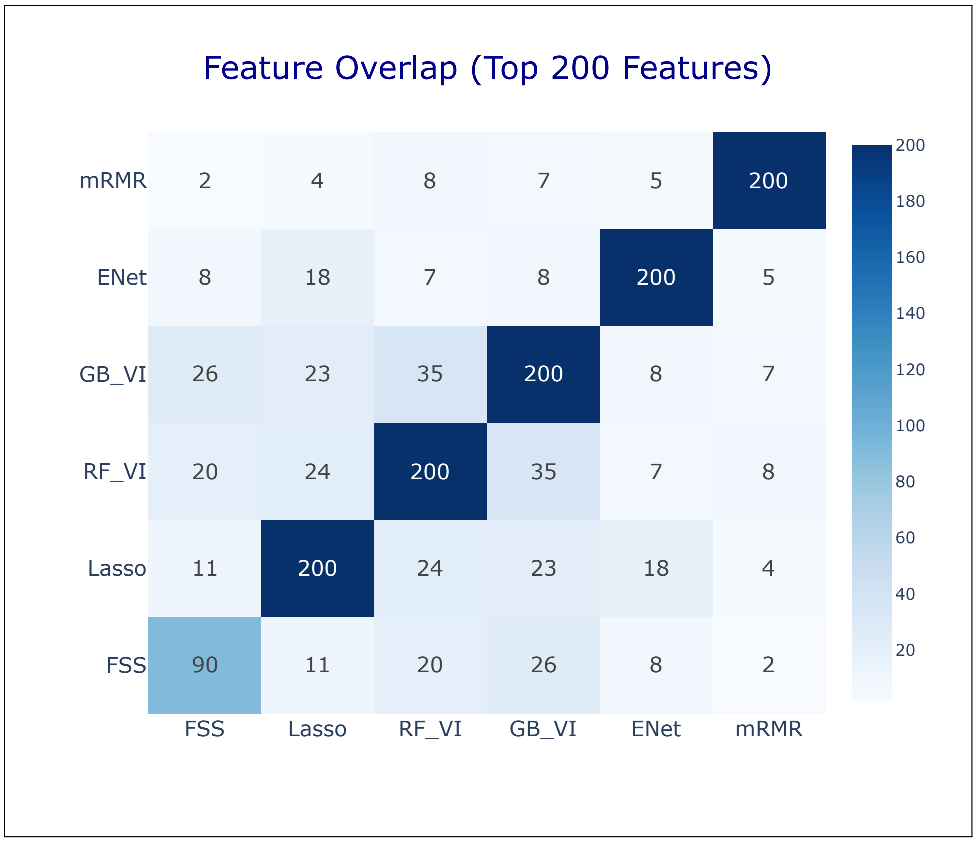


Figure 29: Overlap of the top 200 selected features across all six methods. Low pairwise agreement (maximum 35 shared features between any two methods) reflects the distributed neural encoding structure with no dominant predictors. Different methods capture complementary aspects of the weak signal: FSS identifies strong marginal predictors, GB-VI captures residual structure, and Elastic Net retains correlated feature groups.