Mini-Project (ML for Time Series) - MVA 2024/2025

Jonas Amar jonas.amar@etu.minesparis.psl.eu
Cécile Liu cecile.liu@eleves.enpc.fr

December 17, 2024

1 Introduction and contributions

The discovery of biomarkers from high-dimensional omic data has become a critical task in clinical and biomedical research. However, translating these findings into reliable clinical applications remains challenging. The STABL method, introduced by Julien Hedou and al Hedou et al. (2024), addresses this issue by proposing a machine learning framework that identifies a sparse, reliable set of biomarkers. This method utilizes noise injection and a data-driven signal-to-noise threshold in multivariate predictive modeling.

The STABL algorithm extends traditional sparse regression methods (SRM), such as Lasso, Elastic Net (EN), Adaptive Lasso (AL), and Sparse Group Lasso (SGL), by integrating artificial features into the feature selection process. The goal is to identify sparse and reliable omic biomarkers while controlling for false discovery rate (FDR).

However, because the STABL algorithm relies on base models (mentioned above) which are only fit to detect linear relationships, we propose to adapt the method so that it can detect features which are not necessarily linearly related to the outcome. To do that, we decided to replace the base model by either a random forest or an XGBoost model and select features based on how much improvement a split using one particular feature brings to the objective function. In practice, 90% of the source code for the algorithm was recycled (from `Stabl`) to our purpose and the code that generates the synthetic test datasets relied on the `synthetic-data` library. Our proposed improvement was tested on one real Covid 19 dataset mentioned in the article and on 4 synthetic datasets (1 for which the output is linearly correlated to the informative features, and 3 are non-linear). The coding part was mainly handled by Jonas and the report was mainly handled by Cécile.

2 Method

2.1 STABL Method Overview

(a) Base Sparse Regression Method (SRM) Selection: The first step is to choose a SRM, such as Lasso or EN, to serve as the base estimator. This procedure works on the original dataset $X \in \mathbb{R}^{n \times p}$, where n is the number of samples and p is the number of covariates (features). The model assumes a linear relationship between the outcome vector $Y \in \mathbb{R}^n$ and the feature matrix X, parameterized by the vector of coefficients $\beta \in \mathbb{R}^p$, in the form:

$$Y = X\beta + \epsilon$$

where ϵ is the error term. The set of informative features $S \subset \{1, \dots, p\}$ is defined as those features that are related to the outcome (i.e., $\beta_i \neq 0$).

(b) Generation of Artificial Features: To mitigate overfitting and control FDR, artificial features $\tilde{X} \in \mathbb{R}^{n \times p}$ are generated and appended to the original dataset. These features are constructed through methods like MX knockoffs Candès et al. (2018) or random permutations of the original features. This results in an augmented matrix $\mathcal{X} = [X|\tilde{X}] \in \mathbb{R}^{n \times 2p}$, where half of the features are informative (original) and the other half are uninformative (artificial).

(c) Subsampling and Model Fitting: Next, the STABL algorithm performs B subsampling iterations. For each iteration $k \in \{1, \ldots, B\}$, a random subsample of size $\lfloor n/2 \rfloor$ is drawn from the original dataset $(Y, \mathcal{X})$, and the chosen SRM (such as Lasso) is applied to this subsample with varying regularization parameters $\lambda \in \Lambda \subset \mathbb{R}_{\geq 0}$. The resulting estimate of the coefficients is equal to: $\hat{\beta}^{(k)}(\lambda) = \arg\min_{b \in \mathbb{R}^{2p}} \left(\|Y - \mathcal{X}b\|_2^2 + \lambda \|b\|_1 \right)$

(d) Feature Selection Frequency and Stability Path: After performing the B subsampling iterations, the frequency with which each feature is selected is computed. For each feature i, the selection frequency $f_i(\lambda)$ is the proportion of subsamples where the feature is included in the model:

$$f_i(\lambda) = \frac{1}{B} \sum_{k=1}^{B} \mathbb{1}[\hat{\beta}_i^{(k)}(\lambda) \neq 0]$$

(e) Stability Threshold and Selection: A feature is considered reliable if its maximum selection frequency $\max_\lambda f_i(\lambda)$ exceeds a given threshold $t \in [0, 1]$. The set of stable features is defined as: $\hat{S}(t) = \{i \in [p] : \max_\lambda f_i(\lambda) \geq t\}$

(f) False Discovery Rate (FDR) Control and Reliability Threshold: The final step of STABL involves controlling for false discoveries by optimizing the false discovery proportion (FDP). The augmented FDP at a threshold t is defined as:

$$\text{FDP}_+(t) = \frac{1 + \sum_{j \in A} \mathbb{1}[f_j \geq t]}{\sum_{j \in O} \mathbb{1}[f_j \geq t] \vee 1}$$

where A denotes the set of artificial features and O denotes the set of original features. The reliability threshold θ is selected as the value of t that minimizes the augmented FDP_+. At this threshold θ, the final set of selected features $\hat{S}(\theta)$ is used to construct the final predictive model.

(g) Final Model Construction: The final model is then built using the selected features $\hat{S}(\theta)$.

2.2 Proposed Improvement: Replacing Linear Models with Tree-Based Models

We modify the STABL framework by replacing the linear regression component with a tree-based model. Taking XGBoost as an example, we make slight modifications to the following steps:

(c) New Model Fitting: The SRM in step (c) is replaced with XGBoost, a tree-based model. In this case, the hyperparameter space is defined by a combination of parameters specific to XGBoost, defined later in table 3.

For each subsample iteration $k \in \{1, \ldots, B\}$, XGBoost is fitted on a random subsample of the augmented dataset $(Y, \mathcal{X})$ with varying hyperparameter configurations. Instead of selecting features based on the coefficient estimates ($\hat{\beta}^{(k)}(\lambda)$) as in Lasso, we utilize the normalized feature importance of each feature.

The normalized feature importance for feature i in iteration k is given by: $\widetilde{\texttt{importance}}_i^{(k)} = \frac{\texttt{importance}_i^{(k)}}{\sum_{j=1}^{2p} \texttt{importance}_j^{(k)}}$ where $\texttt{importance}_i^{(k)}$ quantifies the contribution of feature i to the reduction of the model's loss function during tree splitting.

- In **XGBoost**, $\texttt{importance}_i^{(k)}$ corresponds to the "gain," representing the average improvement in the objective function when feature i is used for a split.
- In **Random Forest**, $\texttt{importance}_i^{(k)}$ measures the average decrease in impurity (e.g., Gini or variance) attributed to feature i across all trees.

Normalization ensures that feature importance scores remain comparable across iterations and hyperparameter settings.

(d) New Feature Selection Frequency: For each feature i, the selection frequency f_i is defined as the average normalized gain across all subsamples. This is computed as: $f_i = \frac{1}{B}\sum_{k=1}^{B} \widetilde{\texttt{importance}}_i^{(k)}$. It replaces the original frequency computation used in Lasso-based STABL. Instead of counting the number of times a feature's coefficient is non-zero, XGBoost-STABL and Random Forest-STABL use the aggregated importance scores as a proxy for feature selection frequency. Features with higher aggregated normalized importance scores are considered more stable and informative. **The other steps will stay identical as in the paper.**

3 Experiment Design

The goal of this study was to assess the performance of our proposed improvement to the STABL algorithm compared to the original version in both linear and non-linear settings. Specifically, we aimed to determine whether the improved STABL maintains its performance in linear scenarios while outperforming the original in non-linear cases.

3.1 Datasets

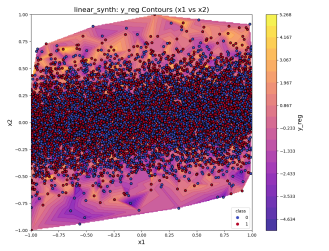
To achieve this, we created four synthetic datasets and utilized one real-world dataset:

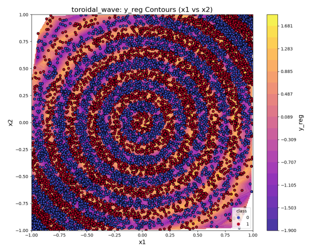
1. **Linear Synthetic Dataset:** Contains 15 informative variables and 85 noisy variables.
2. **Rippling Hyper-Shell Dataset:** Similar in structure to the linear dataset but with a non-linear relationship between features and outcomes.
3. **Toroidal Wave Dataset:** A non-linear dataset where 3 out of 20 features are informative.
4. **Polynomial**: The output was produced as the polynomial of 25 out of 100 features
5. **Real Dataset (COVID-19):** From the original STABL paper, used as a benchmark to evaluate model performance on real-world data.

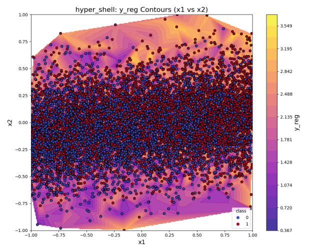
The synthetic datasets are represented in Figure 2, where we see the nice Toroidal Wave in Figure 2 (b), a mixed cloud of points for both the Linear and Rippling Hyper-Shell datasets due to the higher number of informative featuresa and a clear alternation of regions with labels 0 and 1 in Figure 2 (d)

3.2 Results

The STABL framework includes hyperparameter tuning (presented in table 3) for each base model to ensure optimal performance. For STABL-based models, hyperparameter grids were applied within the STABL pipeline, which includes bootstrap sampling and feature selection thresholds. All STABL models used 100 bootstraps, an artificial knockoff noise proportion of 1.0, and a false discovery rate threshold range of $\{0.1, \ldots, 1.0\}$. All models were validated using 5-fold cross-validation repeated 3 times to ensure


(a) Projection of the Linear synthetic dataset on (x_1, x_2)


(b) Projection of the Toroidal Wave dataset on (x_1, x_2)


(c) Projection of the Rippling Hyper-Shell dataset on (x_1, x_2)

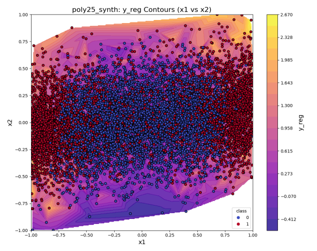

(d) Projection of polynomial dataset on (x_1, x_2)

Figure 1: Projection of the four synthetic datasets on x_1 and x_2, which are two informative variables.

robustness of the results. The detailed results are presented in tables 1 and 2.
In table 1, we notice that the sparsity obtained with STABL is stronger on all datasets than the base models alone. Although STABL with the linear base models outperforms the tree based models on the linear dataset, the latter achieve better or similar IoU while selecting **fewer** features on the Rippling Hyper-Shell and polynomial datasets. What is even more interesting is that STABL with tree-based models lead to equal or better performances (measured in table 2 by ROC AUC for classification tasks and R2 for regression tasks) with **sparser** models than STABL with linear base models. This is true in particular for the Covid-19 dataset. However, none of the proposed models yield satisfying sparsity or performance results on the torroïdal wave dataset. That may be due to the fact that this dataset only had a small number of features (in addition of a really complex structure as can be seen in figure 1b) and STABL was designed to select features in datasets containing several hundreds of attributes.

Table 1: Mean IoU[1] over cross-validation (n_folds = 5, n_repeat = 3) folds (and mean number of selected features).

Task type	Dataset	True ratio[2]	Lasso	EN	RF	XGB	STABL Lasso	STABL EN	STABL RF	STABL XGB
Binary	Linear	15/100	0.30 (45/100)	0.31 (39/100)	0.15 (100/100)	0.17 (88/100)	0.66 (10/100)	0.59 (10/100)	0.29 (4/100)	0.35 (7/100)
	Rippling Hyper-Shell	15/100	0.12 (7/100)	0.15 (5/100)	0.15 (100/100)	0.18 (50/100)	0.12 (4/100)	0.12 (4/100)	0.16 (2/100)	0.26 (5/100)
	Toroidal Wave	3/20	0.10 (0/20)	0.10 (0/20)	0.15 (20/20)	0.15 (20/20)	0.17 (10/20)	0.07 (0/20)	0.15 (20/20)	0.16 (14/20)
	Polynomial	25/100	0.10 (8/100)	0.12 (12/100)	0.25 (100/100)	0.25 (98/100)	0.14 (11/100)	0.15 (12/100)	0.15 (4/100)	0.17 (5/100)
	COVID-19	NA	(16/1420)	(298/1420)	(208/1420)	(29/1420)	(8/1420)	(20/1420)	(5/1420)	(2/1420)
Regression	Linear	15/100	1.00 (15/100)	1.00 (15/100)	0.15 (100/100)	0.15 (98/100)	1.00 (15/100)	1.00 (15/100)	0.33 (5/100)	0.33 (5/100)
	Rippling Hyper-Shell	15/100	0.19 (10/100)	0.20 (8/100)	0.15 (100/100)	0.15 (98/100)	0.16 (5/100)	0.17 (5/100)	0.22 (3/100)	0.13 (2/100)
	Toroidal Wave	3/20	0.17 (1/20)	0.18 (1/20)	0.15 (20/20)	0.15 (20/20)	0.25 (6/20)	0.25 (6/20)	0.15 (20/20)	0.08 (0/20)
	Polynomial	25/100	0.10 (8/100)	0.12 (12/100)	0.25 (100/100)	0.25 (98/100)	0.14 (11/100)	0.15 (12/100)	0.15 (4/100)	0.17 (5/100)

[1]**Intersection over Union metric:** it can only be computed if you know the true informative features, which is why it does not appear for the Covid dataset.
[2]**True ratio:** the number of true informative features over the total number of features.

Table 2: Performance of the models after feature selection (ROC AUC for binary tasks and R2 for regression tasks, both averaged over the cross-validation folds)

Task type	Dataset	Lasso	EN	RF	XGB	STABL Lasso	STABL EN	STABL RF	STABL XGB
Binary	Linear	0.998	0.998	0.931	0.972	0.995	0.996	0.925	0.982
	Rippling Hyper-Shell	0.977	0.976	0.971	0.974	0.977	0.976	0.979	0.978
	Toroidal Wave	0.482	0.475	0.481	0.492	0.479	0.495	0.482	0.481
	Polynomial	0.503	0.503	0.954	0.965	0.539	0.517	0.485	0.526
	COVID-19	0.852	0.861	0.860	0.876	0.851	0.869	0.871	0.890
Regression	Linear	0.999	1.000	0.688	0.943	1.000	1.000	0.824	0.911
	Rippling Hyper-Shell	0.851	0.950	0.780	0.931	0.851	0.852	0.851	0.852
	Toroidal Wave	-0.002	-0.002	0.000	-0.074	-0.008	-0.009	-0.018	-0.03
	Polynomial	0.020	0.024	0.663	0.920	0.014	0.019	0.021	0.023

4 Conclusion

Our study demonstrates the versatility and effectiveness of the STABL framework when applied to both linear and non-linear datasets. By integrating tree-based models like Random Forest and XGBoost into the STABL pipeline, we showed that it is possible to achieve remarkable sparsity in feature selection without significant performance degradation. In certain cases, such as the regression task on the Linear dataset and the binary task on the COVID-19 dataset, the STABL framework with tree-based models even outperformed traditional methods like Lasso and ElasticNet in terms of predictive accuracy.

However, challenges remain, particularly with datasets containing few features or highly intricate relationships, such as the Toroidal Wave dataset. These limitations underscore the need for further exploration of more diverse dataset structures, extended hyperparameter grids, and advanced feature importance metrics like Shapley Additive Explanations (SHAP) values (Lundberg and Lee (2017)). SHAP values quantify each feature's contribution to the model's predictions in a way that considers interactions between features, offering a more nuanced understanding of variable importance.

In conclusion, the STABL framework with tree-based models offers a promising avenue for sparse and reliable biomarker discovery, especially in settings where non-linear interactions are prevalent. Future work could build on these findings to optimize and extend the framework for even broader applicability and impact.

References

Emmanuel Candès, Yingying Fan, Lucas Janson, and Jinchi Lv. 2018. Panning for gold: 'model-X' knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 80, 3 (2018), 551–577. https://doi.org/10.1111/rssb.12265

Julien Hedou, Ivana Maric, Grégoire Bellan, Jakob Einhaus, Dyani Gaudilliere, et al. 2024. STABL: Sparse and Reliable Biomarker Discovery in Predictive Modeling of High-Dimensional Omic Data. *Nature Biotechnology* 42, 1 (2024), 22–31. https://doi.org/10.1038/s41587-023-02033-x

Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems* 30 (2017), 4765–4774. https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

5 Appendix

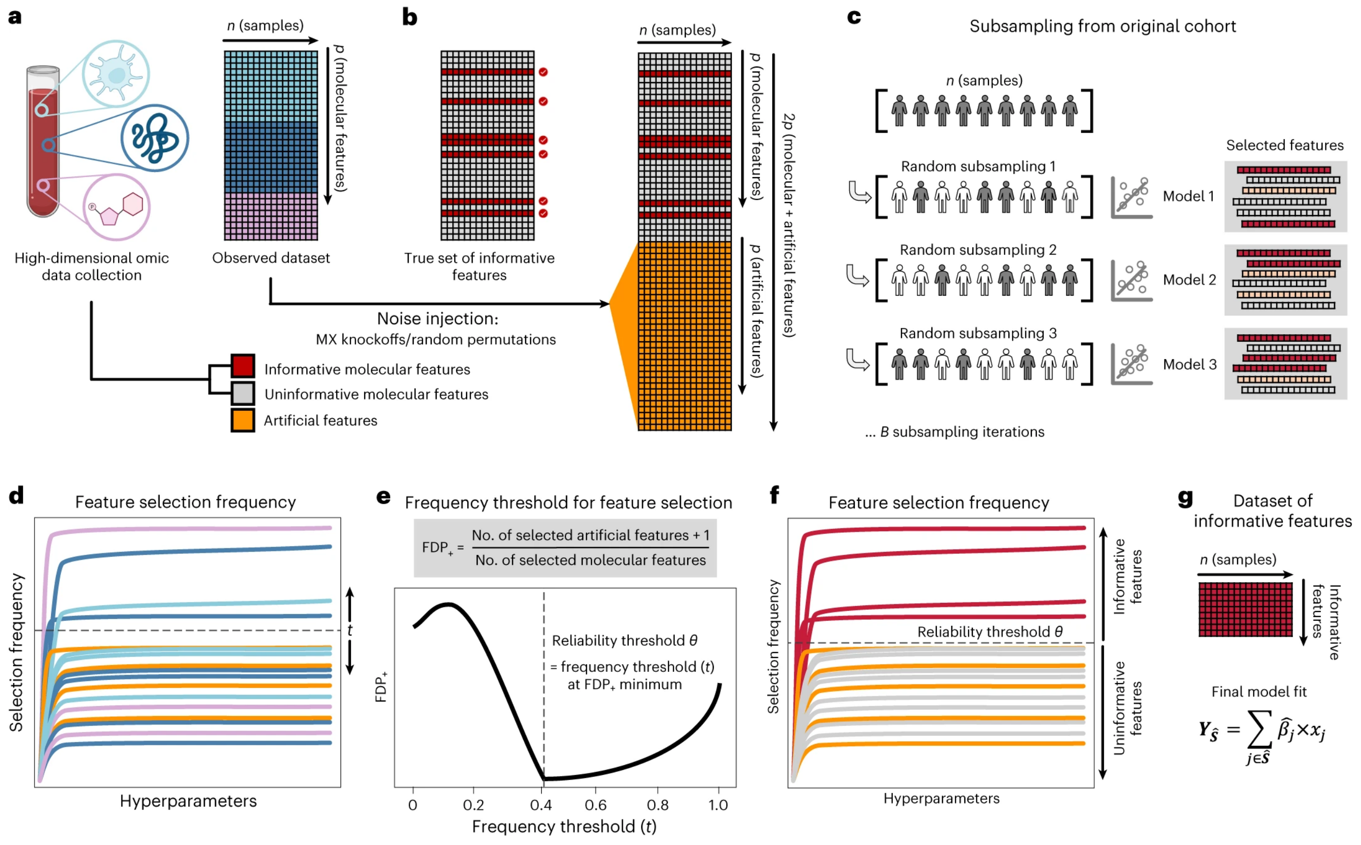

a
High-dimensional omic data collection
n (samples)
p (molecular features)
Observed dataset
b
True set of informative features
n (samples)
p (molecular features)
2p (molecular + artificial features)
p (artificial features)
Noise injection:
MX knockoffs/random permutations
Informative molecular features
Uninformative molecular features
Artificial features
c
Subsampling from original cohort
n (samples)
Selected features
Random subsampling 1
Model 1
Random subsampling 2
Model 2
Random subsampling 3
Model 3
... B subsampling iterations
d
Feature selection frequency
Selection frequency
Hyperparameters
t
e
Frequency threshold for feature selection
FDP+ = No. of selected artificial features + 1 / No. of selected molecular features
FDP+
Reliability threshold θ
= frequency threshold (t) at FDP+ minimum
0
0.2
0.4
0.6
0.8
1.0
Frequency threshold (t)
f
Feature selection frequency
Selection frequency
Reliability threshold θ
Informative features
Uninformative features
Hyperparameters
g
Dataset of informative features
n (samples)
Informative features
Final model fit
$Y_{\hat{S}} = \sum_{j \in \hat{S}} \hat{\beta}_j \times x_j$

Figure 2: The STABL framework: Main figure illustrating the pipeline for feature selection and stability-based model building.

Model	**Hyperparameters**
Lasso	Alpha values: $\{10^{-2}, 10^{-1}, 1, 10, 10^2\}$
ElasticNet	Alpha values: $\{10^{-2}, 10^{-1}, 1, 10, 10^2\}$; L1 ratio: $\{0.5, 0.7, 0.9\}$
Random Forest	Maximum tree depths: $\{3, 5, 7, 9, 11\}$
XGBoost	Max depth: $\{3, 6, 9\}$; Alpha: $\{0, 1, 2, 5\}$

Table 3: Hyperparameter grids explored for different models.