Random subsampling techniques for sea bass mortality prediction

Giovanni Gaio, Simone Moretti

July 20, 2025

1 Introduction

The study of the impact of Single Nucleotide Polymorphisms (SNPs) on the effects of diseases has been largely celebrated. However, actually finding such links is usually not easy, and an automatic data-driven technique to predict disease outcome would be very useful. Given the size of genomes it is not always easy to understand the genes that have an phenotype related to the disease effects. In most cases the SNPs have little to no effect on the mortality of a population to a given disease, and only a few portion of them exerts any difference. Moreover, the size of the datasets hinders the efficacy of standard machine-learning techniques, as few example genomes are available but each is composed of hundreds of thousands of bases, if not many millions.

We are dealing with a sea-bass population suffering from viral nervous necrosis (VNN) [2], which is a highly spread disease among sealife. The broad goal of this work is to asses the impact that SNPs in the sea-bass genome have on the mortality of fishes after the contraption of VNN. SNPs might increment or decrement che chance of a fatal outcome. We would be satisfied with a model that given the SNPs of a sea-bass which has got VNN, correctly predicts if the fish will die or not with high probability. Commonly used models for this task come from machine learning: we focus on the XGBoost classifier [1].

However, training a model on the whole genome might make the resulting model suffer from overfitting. Specifically, in this work we experiment with some simple random subsampling techniques. The idea is to keep only a subset of the genome bases while keeping all the individual genomes sequenced. The hope is that this will allow a standard machine-learning algorithm to better understand the structure and importance of the mutations with respect to the disease outcome.

2 Methodology

For each of our experiments we followed a common pipeline, shown in Figure 1.

In the following part we'll describe the details.

2.1 Dataset

The main dataset we used was a table characterising SNPs of 990 individual sea basses, each sea bass having 6072853 nucleotides included. We refer to it as the **features** dataset.

Each row of the features dataset corresponds to a sea bass, and each column corresponds to a specific nucleotide in the genome. The values in the table represent SNPs. Each value can be a number from 0 to 2, representing the number of alleles mutated for that particular sea bass in that specific nucleotide position:

- 0: no mutation (homozygous reference),

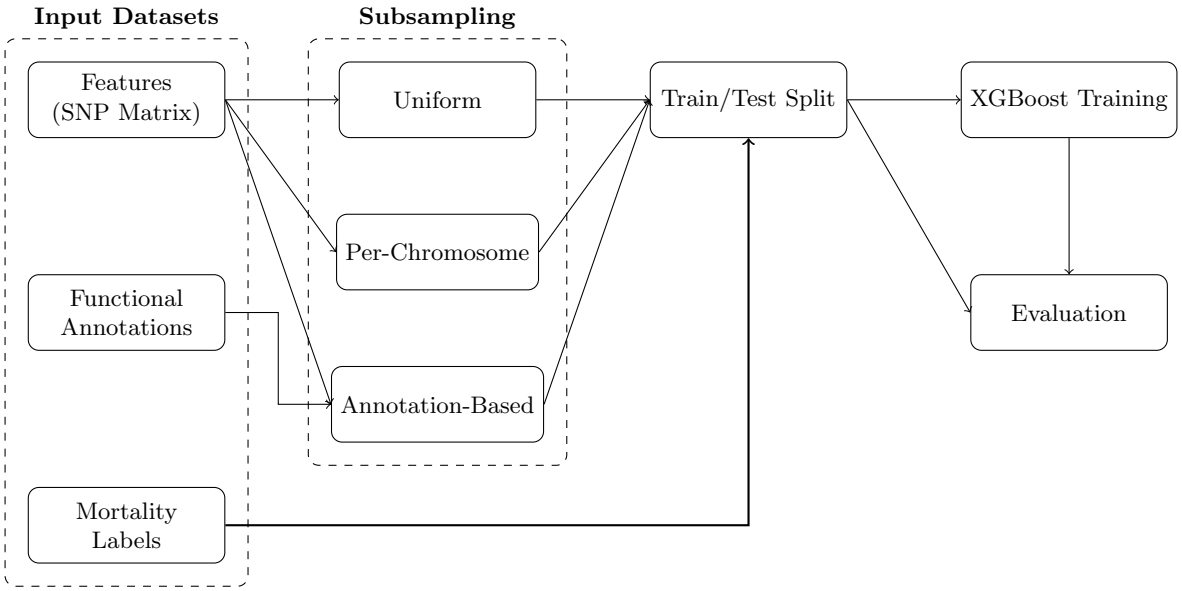


Figure 1: Pipeline of the model training after subsampling of the data

- 1: one allele mutated (heterozygous),
- 2: both alleles mutated (homozygous alternate).

Table 1: Example rows from the SNP features dataset. Each cell denotes a genotype value at a specific nucleotide.

features	CAJNNU010000001.1:299	CAJNNU010000001.1:903	CAJNNU010000001.1:986	...
PL04-A06	0	1	2	...
PL04-A07	1	0	1	...
PL04-A08	2	2	0	...
⋮	⋮	⋮	⋮	⋱

Of each nucleotide we know the chromosome that it belongs to.

Paired with the features dataset we have a **mortality** dataset. It stores for each sea bass if it died or not after the contraption of VNN.

In addition to these two datasets, we had a set of **annotated** nucleotides. These are special nucleotides for which further information is known on their role in the sea bass genome. This information comes in the form of a function (either `Open_chromatin`, `Enhancer` or `Promoter`) and a tissue number (between 0 and 25).

2.2 Subsampling

The subsampling is done over the nucleotides, meaning that each individual remains in the dataset but we consider only a random subset of its nucleotides.

We introduce the **subsampling ratio** p parameter, which just specifies the ratio of nucleotides to be randomly selected over the available ones.

We then find three main subsampling techniques to apply to our features dataset.

Uniform subsampling. The first and simplest thing we tried was to randomly and uniformly subsample the nucleotides on the entire genome. Given p, we find the number of nucleotides to be subsampled by multiplying p by the number of nucleotides, rounding down. Then we uniformly subsample that number of nucleotides.

Uniform subsampling on chromosomes. One observation that could be made with the previous method is that it could select a really high number of nucleotides present in a chromosome while next to no one from another. So the importance that chromosomes might have in the information given to the learning model is not captured. We can constrain the subsampling to take from each chromosome the same number of nucleotides uniformly at random, getting a subsampling tecnique that balances each chromosome contribution in the training set.

Subsampling using annotations. We could also make use of the further informations that we have on the annotated nucleotides. Specifically, we restrict the pool of SNPs to those with certain annotations (e.g., only `Enhancer` regions) or within a desired tissue number range. From this filtered pool, we then apply uniform subsampling as before.

2.3 Training and testing

After the subsampling we need to train and test our model on the resulting data. We split the population in two sets (training and testing), then we train xgboost on the training set and find out if the resulting model has predictive power on the test set.

The individuals used for training and testing are fixed before subsampling the nucleotides. This is made in order to reduce the statistical perturbation that might come from another random procedure. We also fixed the seed of XGBoost, although we don't expect this to make any change on the variance, as the inputs given to the model change with every subsampling.

Since we are dealing with random processes, for each subsampling method and ratio we ran and evaluated multiple instances, taking mean an variance of the scores.

3 Results

In this section we'll look at the results we got from each of the different techniques outlined in section 2.

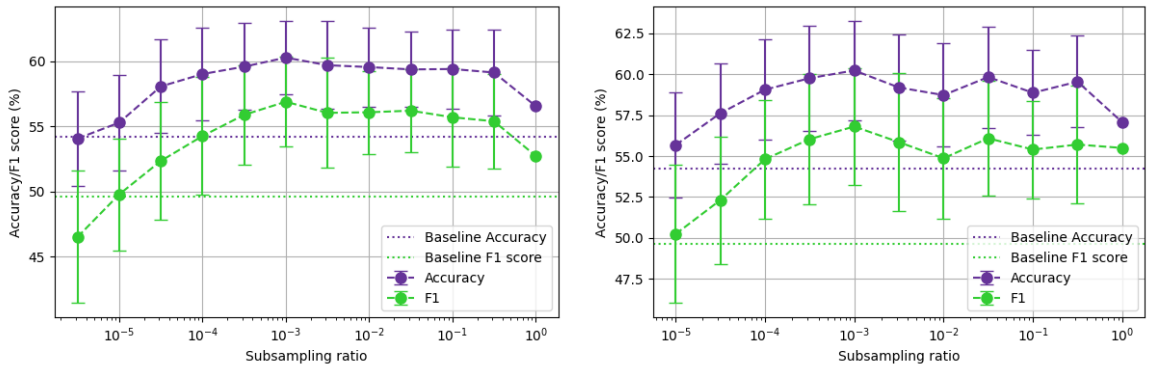
Execution notes. All the results were obtained by running our program with about 16 GB of working memory and 1 core per instance. The system running our program was equipped with an Intel Xeon [1]. The runtime for a sweep on the selected subsample rates is on the order of a few hours. Though note that for different subsampling ratios memory requirements and runtime changed by over an order of magnitude in the extreme cases.

Baseline results. To understand the improvement given by our proposed techniques, in every plot we compared our scores with some "baseline scores". These baseline scores are the score of a "trivial" classifier, that always reports the most common class. Our data samples were in 54.32% of cases of one class, thus the baseline accuracy is this value.

3.1 Subsampling on the whole genome

The first simple tests were done selecting a random subset of genes among all the SNPs that we had. The results for each subsampling rate are shown in Figure 2. These plots show a mostly flat

[1] The cluster we worked on, is more thoroughly described here: `https://docs.dei.unipd.it/en/CLUSTER/Overview`



(a) Plot of the scores when subsampling uniformly on the whole genome.

(b) Plot of the scores when subsampling uniformly on each chromosome.

Figure 2: Plots of accuracy and F1 scores for different subsampling rates on the whole genome.

picture: both for the accuracy and F1 score there isn't a definite trend or variation as a function of the subsampling rate.

The only exception to this is Figure 2a, that shows a diminishing trend with the smallest subsampling rates. Nevertheless both of the last two data points are compatible with the middle points.

Remark that our experimental setup is designed to have no random components out of the subsampling step, thus no subsampling implies no randomization. For this reason the values for subsampling rate $1 = 10^0$ (no subsampling) are the same in the two plots of Figure 2 and have no error bars. This single values could be slightly far from than the true expected value because of random variation in the results. Thus the slight decrease between the "middle" subsampling rates and this last data point is probably not meaningful.

3.2 Annotated genes subsampling

The second set of tests was done using only the annotated genes. The results on each set of similarly annotated genes are shown in Figure 3: Figure 3a shows the results using function annotations, and Figure 3b the results when selecting genes with tissue number in different intervals.

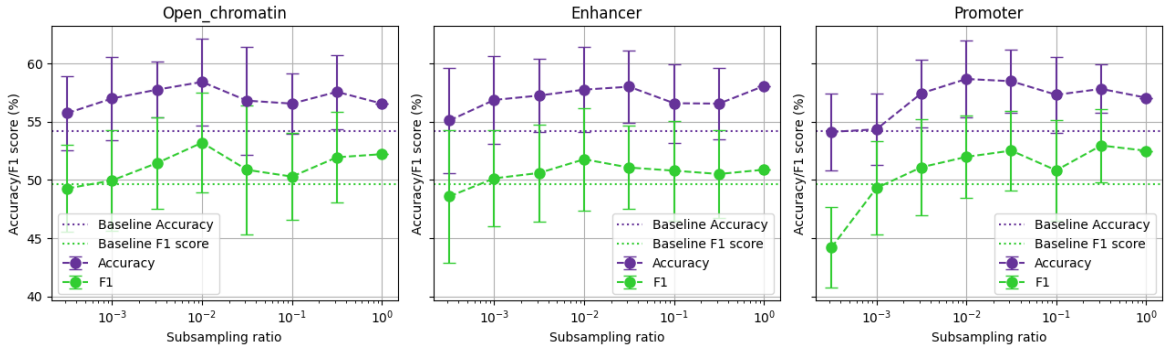
As before, these plots show a mostly flat picture, both for the accuracy and F1 score.

3.3 Considerations on the results

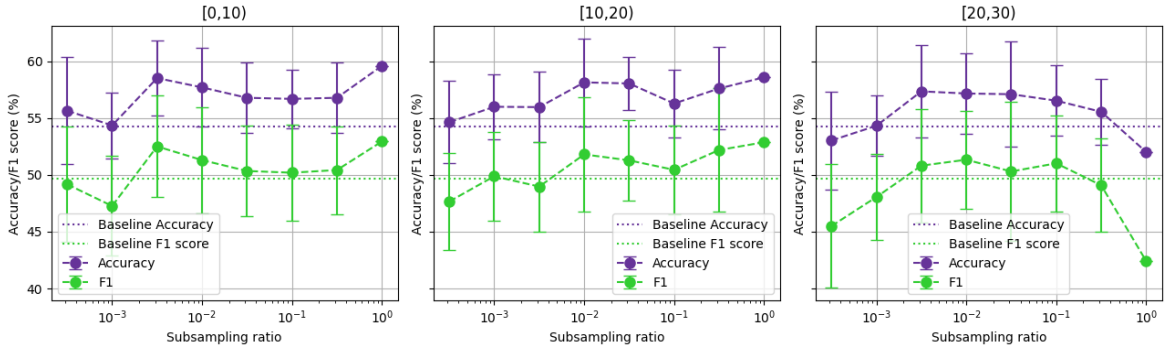
Overall the results do not show any meaningful trend. The most likely pattern to be inferred from our data is a flat trend, meaning that the scores and subsampling rates are nearly independent. This trend seems to hold well outside the extreme ends of our sampled interval of subsampling rates.

In fact, observing the lower extreme, a simple observation can be drawn from our results: keeping only a very small subset of genes (in the order of tens or less) has a negative effect on the scores. This can be seen especially in the "Promoter" plot of Figure 3a or in Figure 2a, but is probably the cause of the lower scores for lower subsampling rates in other plots. More extreme values of the subsampling rates have been tried and support this hypothesis, though the results are not included in this work.

The flat trend could be ultimately a good thing: this could allow future experiments of better models, hyperparameters or techniques to use only a subsample of the data without losing quality in the results. This would enable the use more complex models and faster testing.



(a) Plots of the scores while subsampling genes annotated in different categories.



(b) Plots of the scores while using only genes with tissue number in a given range.

Figure 3: Plots of the accuracy and F1 scores for the different subsampling rates on the annotated genes. Each single plot is done using only the genes annotated with the indicated values.

4 Conclusions

We successfully experimented with a set of different techniques that subsample the dataset of SNPs and disease outcomes, maintaining each sampled individual while discarding some of the SNPs. In some cases we also used annotations given to SNPs, grouping by their function.

We got interesting results, though they are not deeply unexpected: subsampling does not impact the accuracy or F_1 score of the disease outcome prediction; this is true if the subsampling rate is not extreme, when only very few genes are actually used thus inevitably limiting the actual quality of the predictions.

Using a standard machine learning model as a black-box we didn't observe a statistically significant change in the prediction accuracy or F_1 score. If further supported, this fact can allow future experimentation of machine learning techniques on this kind of problem to be done on a subsampled version of the data, without a significant degradation of the quality of the results, but at a much increased speed.

References

[1] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, 2016, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

[2] Giovanni Faldani et al. "Machine Learning Methods for Phenotype Prediction from High-Dimensional, Low Population Aquaculture Data". In: *BIOSTEC* (2025), pp. 638–646.