Random subsampling techniques for sea bass mortality prediction

Giovanni Gaio, Simone Moretti

July 18, 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), 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 come from machine learning, we focus on the XGBoost classifier.

However, training a model on the whole genome might be overkill as the resulting model might 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:

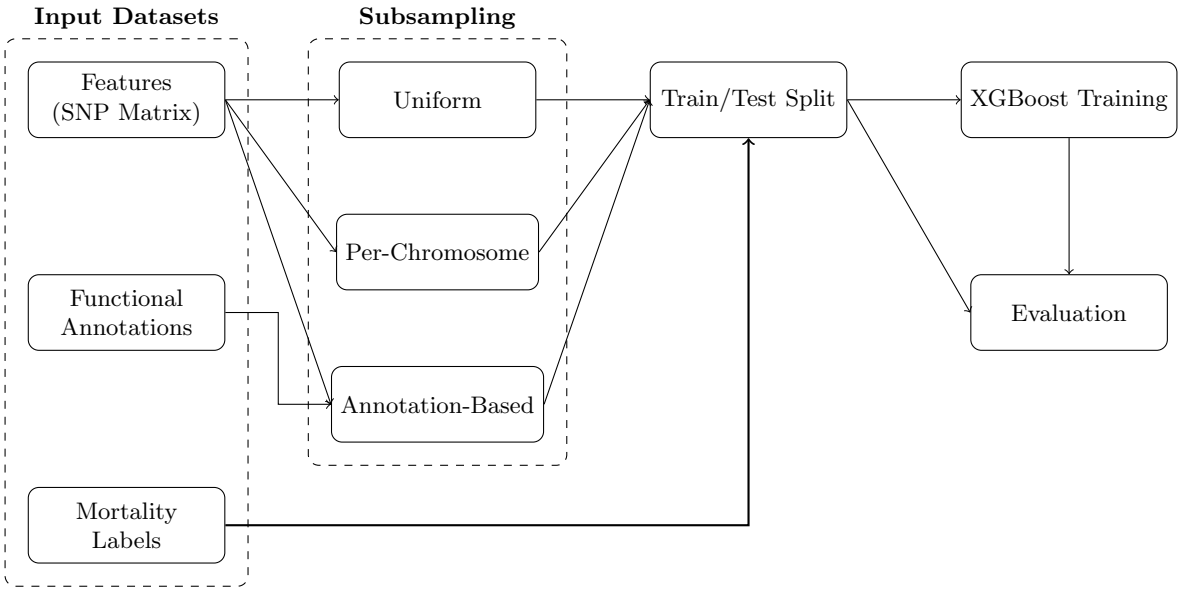


Figure 1: Pipeline of the model training after subsampling of the data

Here we describe the details.

2.1 Dataset

The main dataset we used was a table characterising SNPs of 990 individual sea basses. We call it the **features** dataset.

Each row of the features dataset represents a sea bass, whilst each column represents a nucleotide in the genome. In each cell of this grid we save the SNP information of the corresponding individual on the corresponding nucleotide: 0 if there are no mutations, 1 if only one of the nucleotides of the chromosome pair is mutated and 2 if both the nucleotides are mutated.

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. 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 could only select nucleotides that have a specific function or such that the number of tissues that influences falls into some range. Then we could uniformly subsample among these selected nucleotides as before.

2.3 Training and testing

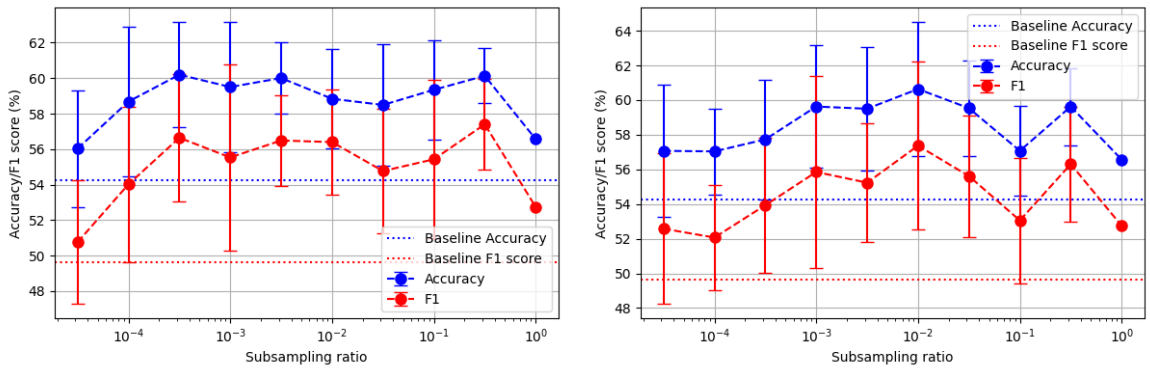
After the subsampling we need to train and test our model on the resulting data. We split the population in two sets (training and testing), then we train xgboost on the training set and find out if the resulting model has predictive power on the test set.

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.

3.1 Subsampling on the whole genome



(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.

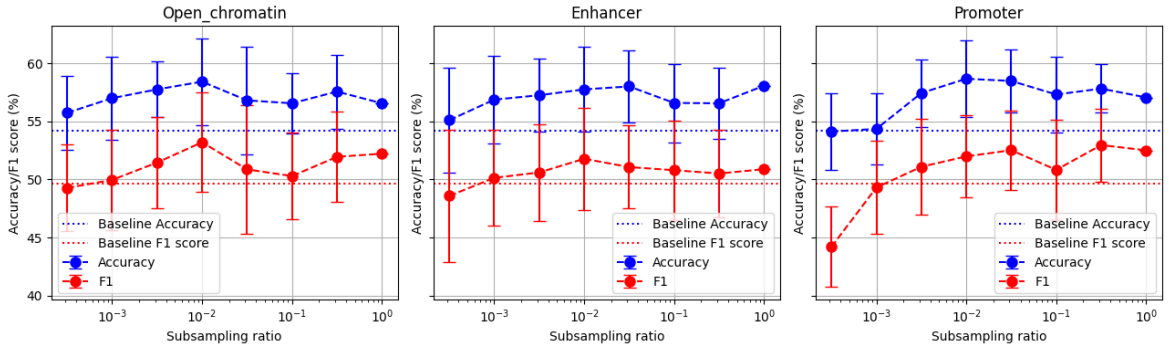
The first simple tests were done selecting a random subset of genes among all the SNPs that we had. The results for each subsampling rate are shown in Figure 2. These plots show a mostly flat picture: both for the accuracy and F1 score there isn't a definite trend or variation as a function of the subsampling rate.

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

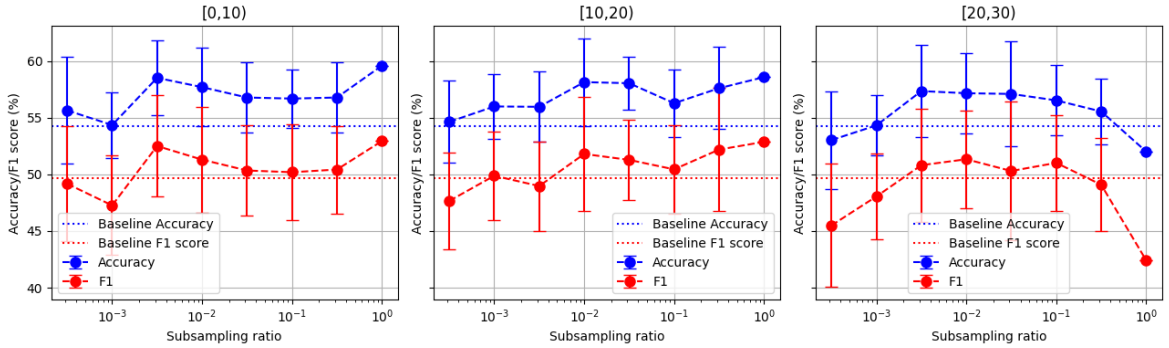
The only exception to this is Figure 2a, that shows a diminishing trend with the smallest subsampling rates. Nevertheless both of the last two data points are compatible with the middle points.

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



(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.

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 seems to hold outside the extreme ends of our sampled interval of subsampling rates.

Another simple observation that can be drawn from our results is that 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, and could be a cause of the upwards trend on the left of Figure 2a.

4 Conclusions