Machine Learning in Computational Biology
Assignment 1
Bonus Questions

Konstantinos Konstantinidis
Student number: 7115152400017

April 6, 2025

Repo: The repository for this assignment can be found here:
`https://github.com/KonsKons26/Assignment-1`

Contents

1 First bonus task: Using Optuna for hyperparameter tuning 2
1.1 Migration to Optuna 2
1.2 Hyperparameter tuning 2
1.3 Results 3

2 Second bonus task: Converting the problem to a binary classification task 4

1 First bonus task: Using Optuna for hyperparameter tuning

1.1 Migration to Optuna

Since the `Regressor` object I had created to perform all regression tasks was highly modularized, with specific methods for each task, moving from `sklearn`'s `GridSearchCV` to `Optuna` was quite simple. I created a new class named `RegressorOptuna` which inherited from `Regressor`, to keep it's basic functionalities the same. Then I modified the private methods handling each model's hyperparameter optimization to allow me to pass the appropriate grid spaces –and in the appropriate format– for `Optuna` to tune the models.

The objective function **minimizes** the RMSE of the test set. The sampler I chose is the `TPESampler`, a tree-structured Parzen estimator, which fits one Gaussian Mixture Model (GMM) (lx) to the set of parameters associated with the best objective values and another GMM $(g(x))$ to the remaining parameter values. It chooses the parameter that minimizes the ration $l(x)/g(x)$.

All files are in the appropriate directories, `src/` and `notebooks/` and the models were saved in `models/bonus1_optuna/`. Since I used the features I selected in the previous task, to keep everything organized, the process includes copying the features files in the `models/bonus1_optuna` directory.

1.2 Hyperparameter tuning

I set the number of trials for each model tuning to 1000, which seemed appropriate, as it was not such a demanding number for my machine, and since the hyperparameter space has so many dimension, I believe that a small number of trials might not be enough to search the whole space. The complete hyperparameter space, along with the values chosen by `Optuna` are shown in Table 1.

Model	Parameter	Values	Picked value
ElasticNet	α	(0.01, 1.0)	0.11850
	`l1 ratio`	(0.0, 1.0)	0.77838
	`tolerance`	[1e-3, 1e-4, 1e-5, 1e-6, 1e-7]	0.00100
SVR	`kernel`	['rbf', 'linear', 'poly', 'sigmoid']	'rbf'
	`degree`	(2, 5)	4
	γ	['scale', 'auto']	'scale'
	`coef_0`	(0.0, 1)	0.93453
	`tolerance`	[1e-3, 1e-4, 1e-5, 1e-6, 1e-7]	1e-7
	`C`	(0.1, 10)	1.63421
	ϵ	(0.0, 10.0)	0.92788
BayesianRidge	`tolerance`	[1e-3, 1e-4, 1e-5, 1e-6, 1e-7]	1e-7
	α_1	(1e-9, 1e-3)	0.0
	α_2	(1e-9, 1e-3)	0.00065
	λ_1	(1e-3, 1e-9)	0.0009999
	λ_2	(1e-3, 1e-9)	0.0
	`compute_score`	[True, False]	True

Table 1: Hyperparameter spaces for each model.

1.3 Results

After tuning, the best models were saved (along with their scalers, like in the previous task) and then the validation set was used to measure their performance. For testing, the same approach with resampling was followed, for 1000 repeats. The resulting boxplots are shown in Figure 1.

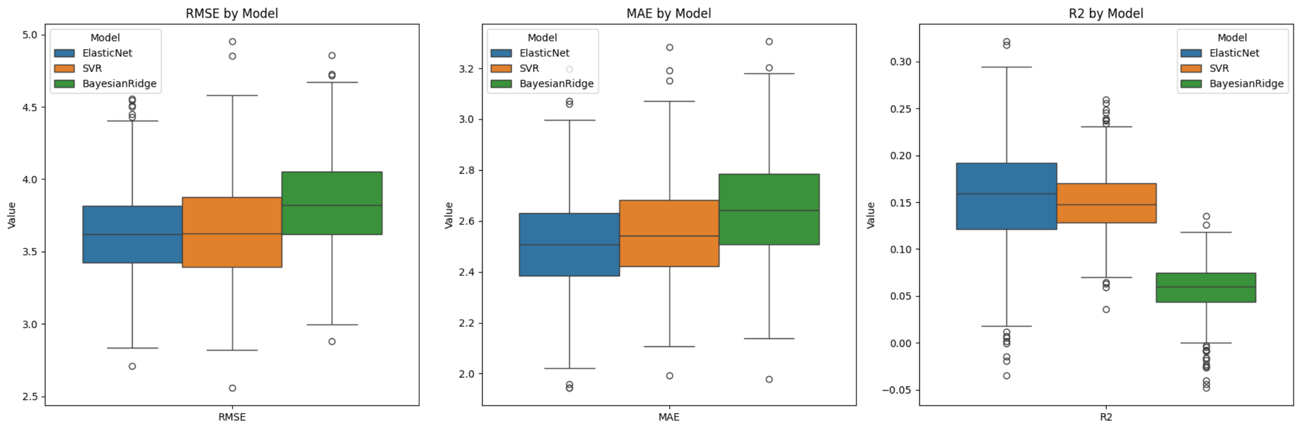

Figure 1: Test metrics after tuning with `Optuna`. ElasticNet in blue, SVR in orange, and Bayesian Ridge in green.

Similar to what was shown in the previous task, Elastic Net is the best performing model, but its RMSE and MAE still fail to drop below 3, while r^2 fails to overcome 0.2. These findings further increase my confidence that this dataset can not be used for regression tasks, at least no with the models that we analyzed.

2 Second bonus task: Converting the problem to a binary classification task