Exercise 07: Learning Rules

In this exercise we will use and compare the results yielded by three different learning rules: the **pseudoinverse rule**, the **adaline** and the **Bayes**. We consider a student perceptron with $N = 20$ binary inputs and one binary output.

Step 1, 2 and 3: Apply the pseudoinverse and find $\alpha(\epsilon)$ by trial

The pseudoinverse rule consists of the prescription that stabilities Δ^μ are all equal, and they are equal to κ where κ is as large as possible when $\vec{J}^2$ is fixed. Hence, the $\vec{J}^{\text{pi}}$ is the vector that is equally inclined with respect to all $\vec{\xi}^\mu \sigma_T^\mu$. Notice that this can only happen when $\alpha = P/N < 1$. From the general form

$$\vec{J} = \frac{1}{\sqrt{N}} \sum_\mu x^\mu \vec{x} i^\mu \sigma_T^\mu$$

where x^μ's are *embedding strengths*, we can devise a way to find the pseudoinverse without explicitly defining them. In fact we define the *correlation matrix*

$$C_{\mu\nu} = \frac{1}{N} \vec{\xi}^\mu \vec{\xi}^\nu \sigma_T^\mu \sigma_T^\nu$$

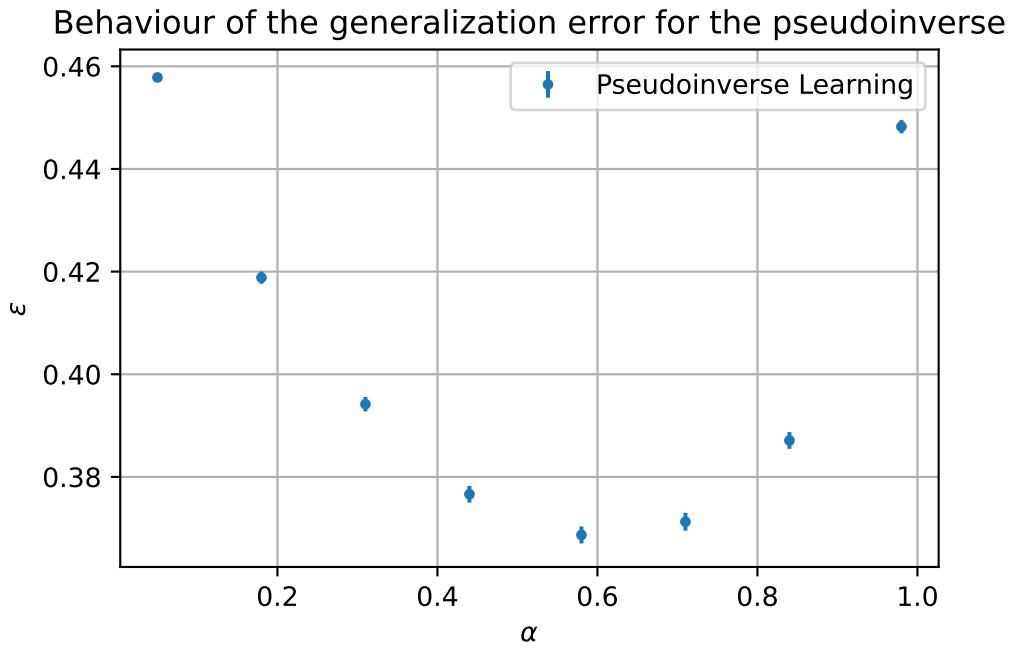
and know that if it is such that it solves the system of equations

$$\sum_\nu C_{\mu\nu} x^\nu = 1$$

and $\xi^\mu \sigma_T^\mu$ are linearly independent, then $C_{\mu\nu}$ has a unique solution. Then $\vec{J}$ can be written as

$$\vec{J}^{\text{pi}} = \frac{1}{\sqrt{N}} \sum_{\mu\nu} (C^{-1})_{\mu\nu} \vec{x} i^\mu \sigma_T^\mu.$$

If we apply this procedure in 1000 training-test loops, with $\alpha \in (0, 1)$ and $P_{\text{test}} = 1000$, we get the following behaviour of the generalization error.



As α approaches one, the generalization error fails to decrease as we would expect from a good learning rule.

This is due to the fact that the prescription imposed by the pseudoinverse rule *overcomes* the goal of the learning, placing the students further away from the teacher as examples get bigger and bigger.

Step 4, 5 and 6: Apply the adaline and find $\alpha(\epsilon)$ by trial

We now focus on another learning rule: it is a generalization of the pseudoinverse rule over the entire $\alpha \in (0, +\infty)$ domain, so we expect that they coincide over the common interval of definition of the α's.

The adaline rule consists of a *gradient descent* over the energies, defined as

$$E(\vec{J}) = \frac{1}{2} \sum_{\mu} (\vec{J}\vec{x}i^{\mu} - \sigma_T^{\mu})^2$$

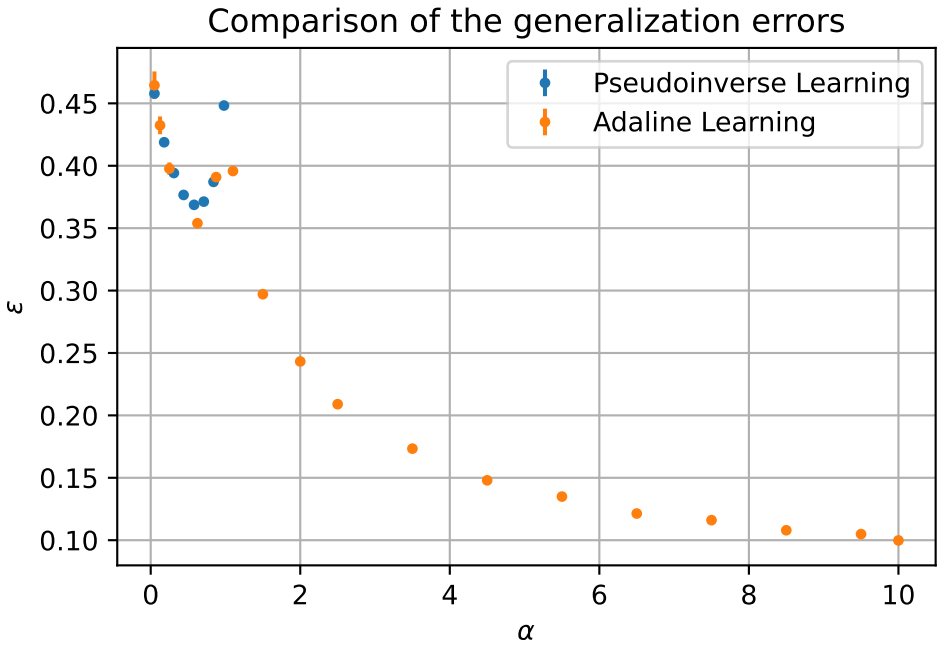
hence the learning steps are given by a multiple of its derivative, namely

$$\delta J = -\gamma \frac{\partial E}{\partial \vec{J}} = \gamma \sum_{\mu} (\vec{J}\vec{\xi}^{\mu} - \sigma_T^{\mu})$$

which in some sense amounts to finding a $\vec{J}$ that yields a perceptron result that mostly resembles σ_T^{μ}. One must pay particular attention to the $\vec{J}_0$: intialization of the algorithm must be

performed over a vector that lie in the span of the example vectors. One can, for example, initialize the vector to be the properly normalized sum of all the examples.

We can then perform the procedure over 1000 train-test loops, with $\alpha \in (0, +\infty)$ and $P_{\text{test}} = 1000$, and we get the following behavior for the generalization error.



We notice that over the $\alpha \in (0, 1)$ interval the two results have a compatible behavior. The rest of the domain is covered only by the adaline, and the generalization error decays to zero as α's get bigger and bigger.

Step 7, 8 and 9: Use Bayes Rule and superimpose analytic predictions

A third rule that can be further adopted is the so-called **Bayes rule**. Let's say that for Bayes theorem we know that

$$P(\vec{T}|\{\vec{\xi}^{\mu}\}, \sigma_T^{\mu}) = \frac{P(\{\vec{\xi}^{\mu}\}, \sigma_T^{\mu}|\vec{T})P(\vec{T})}{P(\{\vec{\xi}^{\mu}\}, \sigma_T^{\mu})}$$

so the *posterior* probability (the probability of having a teacher given a dataset observed) is given by the *likelihood* (the probability of observing a dataset given a teacher) times the *prior* (the probability that a dataset occurs) divided by the *marginal likelihood* (the probability that a teacher occurs).

Since we know that the teacher is uniform on the sphere, and since the likelihood measures the probability of correct classifications, the posterior as well will be uniform on the sphere. This being given, we want to get the optimal student $\vec{J}^b$. How do we do that?

Optimal in our case means that we are looking for the student with the lowest generalization error. This means that we are looking for the vector on the sphere which will yield the largest overlap with the teacher, on an average over the distribution of the teachers. This average overlap will be then given by

$$\langle R \rangle_{\vec{T}} = \frac{1}{N} \frac{\int_{VS} d\mu(\vec{T}) \vec{J}\vec{T}}{\int_{VS} d\mu(\vec{T})} = \frac{\vec{J}}{N} \frac{\int_{VS} d\mu(\vec{T}) \vec{T}}{\int_{VS} d\mu(\vec{T})}$$

where the fraction on the right in the last term is exactly the *center of mass* of the teachers. The vector such that the center of mass is maximised is given by

$$\vec{J}^b = \int_{VS} d\mu(\vec{T}) \vec{T}$$

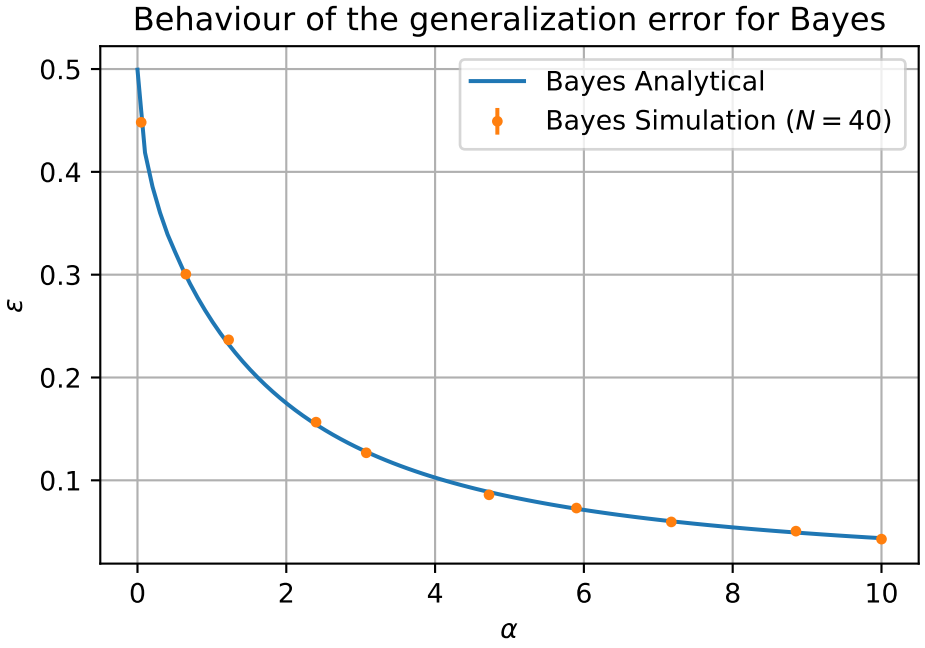
as far as possible from the boundaries of the version space.

This is implemented algorithmically by initializing M random perceptron students, and by taking their average (normalized to $\sqrt{N}$).

$$\vec{J}^b = \lim_{M \to \infty} \sqrt{N} \frac{\sum_{m=0}^{M} \vec{J}^m}{|| \sum_{m=0}^{M} \vec{J}^m ||}$$

We can compare this behavior to the analytical predictions given by the expression

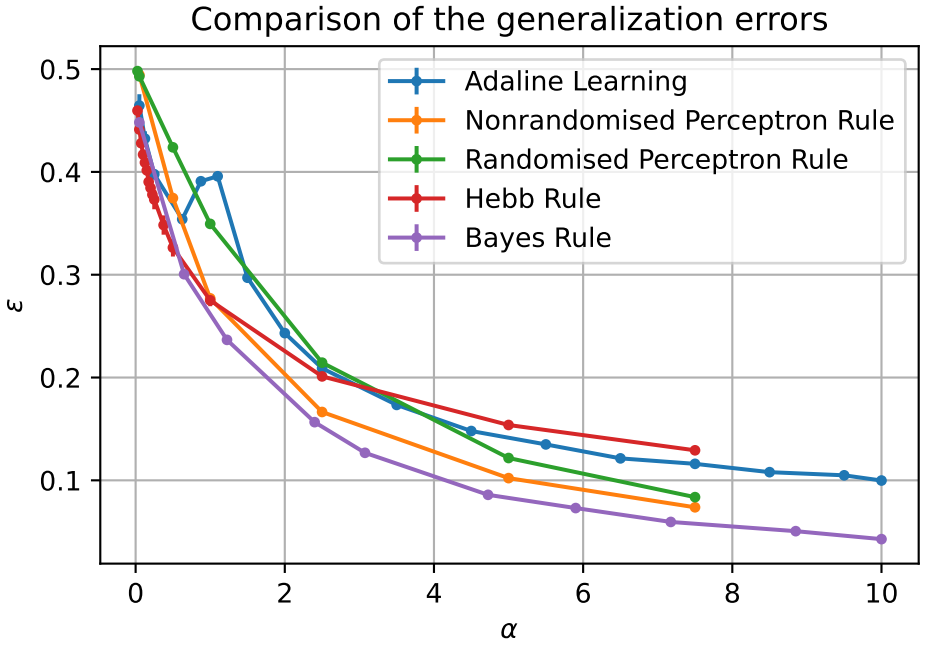
$$\frac{R_B^2}{\sqrt{1 - R_B^2}} = \frac{\alpha}{\pi} \int Dt \, \frac{\exp -\frac{1}{2} R_B^2 t^2}{H(-R_B t)}$$

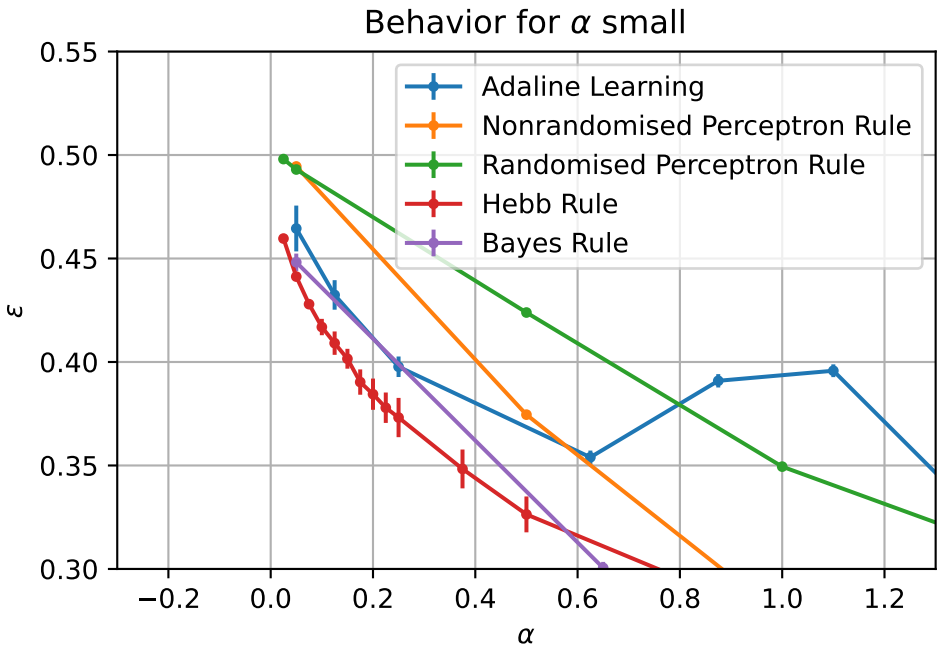


Step 10: Compare with all the other rules

We can notice from the picture below that:

- the **randomised perceptron** rule is the worst performing learning rule for small α values;
- the **nonrandomised perceptron** performs better with respect to the randomised version over the whole domain, because of the fact that the randomised one takes up more time to explore the phase space;
- the **Hebb rule** performs well for small α's, though the learning becomes slower and slower as the size of the training set increases, due to the nonzero training error that keeps the generalization error lower-bounded;
- the **Bayes rule** performs the best (notice that the *crossing* between Bayes and Hebb happens only because of the choice of spacing between data points) having similar performances to those of Hebb at lower values of α, but maintaining optimal performance over the entire domain. Remember that the Bayes rule is *defined* in such a way to get optimal learning, so this is what we expected.




Behavior for α small
Adaline Learning
Nonrandomised Perceptron Rule
Randomised Perceptron Rule
Hebb Rule
Bayes Rule
0.55
0.50
0.45
0.40
0.35
0.30
ε
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
α