Assignment I

Andrei Secuiu (s4260732), Roberto Schinina' (S4612299)

Introduction

The Rosenblatt perceptron is one of the first examples of Machine Learning architectures. Its design is simple, but not less worthy of a dedicated study. From a mathematical point of view, the simplicity is a strength, e.g. because results such as guaranteed convergence in a finite time given linear separability (l.s.) can be given. Furthermore, the problem of linear separability leads naturally to the notion of the *storage capacity* of the perceptron and to $P_{l.s.}$. We recall that $P_{l.s.}(\alpha)$ describes the fraction of l.s. data sets with $P = \alpha N$ *fixed* vectors $\xi^\mu \in \mathbf{R}^N$ in general position and *variable* labels $S^\mu \in \{\pm 1\}$. For this assignment, our aim is to:

1. devise and implement a mechanism for empirically validating $P_{l.s.}(\alpha)$
2. **bonus**: provide a mathematical proof for the asymptotic behaviour of $P_{l.s.}$ when $N \to \infty$ and α is fixed. We further validate the limit empirically

Method

This section introduces the relevant theory and algorithms necessary for determining the fraction of l.s. data sets $P_{l.s.}$ empirically, let us denote it by $Q_{l.s.}$. We begin with a brief discussion of the data generation mechanism, followed by Rosenblatt's algorithm and its role within the assignment. The section continues with a short mathematical derivation that helps with correctly implementing and interpreting $P_{l.s.}$. The section concludes with the experimental setup, including Python implementation tricks.

Data generation and verification of l.s.

The generation mechanism. The data is generated according to the instructions in the assignment, with the help of the `numpy.random` library. In particular, $P = \alpha N$ (rounded to the nearest integer) points are sampled independently from

$$S^\mu = 2U - 1 \quad \text{with } U \sim \text{Ber}(0.5) \qquad \xi^\mu \sim \text{N}(\mathbf{0}, I_N)$$

where Ber(0.5) is a Bernoulli distribution with parameter $p = 0.5$, and $\text{N}(\mathbf{0}, I_N)$ is the multivariate normal distribution with mean vector $\mathbf{0}$ and covariance matrix the identity matrix. When sampling ξ^μ as vectors from $\text{N}(\mathbf{0}, I_N)$, it is automatically ensured that the components ξ_i^μ are independent and each is individually distributed $\text{N}(0, 1)$.

When ξ^μ are sampled from $\text{N}(\mathbf{0}, I_N)$ independently, then we know from the lecture that the points will be in general position with probability 1. Hence $P_{l.s.}$'s formula is valid and $Q_{l.s.}$ will not have a bias when estimating it, at least with regards to ξ. Sampling the labels independently, both from ξ and

among each other, also has the purpose of making $Q_{l.s.}$ an unbiased (or a less-biased) estimator.

Rosenblatt's perceptron algorithm. In the context of this report, Rosenblatt's algorithm plays a pivotal role. As far as it was seen during the lectures, there are no simple necessary and sufficient conditions which can be used to determine if a given data set is l.s. Rosenblatt's algorithm is our best tool available because if a data set is l.s. (then the algorithm converges in a finite number of steps). If a data set is not l.s., then the algorithm cannot converge (there is no solution by definition). Thus, the convergence of Rosenblatt's perceptron algorithm can be used as an equivalent condition for linear separability[1]. Algorithm 1 shows the pseudo-code used for our Python implementation of Rosenblatt's perceptron algorithm.

Algorithm 1 Rosenblatt's perceptron algorithm. The logic is adapted from lecture D1. Heaviside's function has been imported from the `numpy` library

Require: ξ^μ, S_R^μ, $c \geq 0$, $n_{\max}$ (maximum number of epochs)
  unpack P, N from the data
  initialize: $n \leftarrow 1$, $E^\mu \leftarrow 0$, $\mathbf{w} \leftarrow \mathbf{0}$, *convergence_reached* $\leftarrow$ FALSE
  while $n \leq n_{\max}$ and *convergence_reached* = FALSE **do** ▷ Iterate over epochs
    for $i \leftarrow 1$; $i \leq P$; $i \leftarrow i + 1$ **do** ▷ Present all data sequentially per epoch
      compute local potential: $E^i \leftarrow \xi^i \cdot \mathbf{w}\ S_R^i$
      Hebbian learning on $\mathbf{w}$: $\mathbf{w} \leftarrow \mathbf{w} + \frac{1}{N}\xi^i S_R^i \Theta\left(c - E^i\right)$
    end for
    if $E^i \geq c \quad \forall\ i$ **then** ▷ Check if convergence is reached
      convergence_reached $\leftarrow$ TRUE
    end if
    $n \leftarrow n + 1$
  end while
    return $\mathbf{w}$, *convergence_reached*

Efficient computation of $P_{l.s.}$

Computing $P_{l.s.}$ using a computer can be challenging if its formula is implemented naively. The root of the problem lies in overflow/underflow errors, i.e. when a computer cannot accurately store a very large/small number because the number of digits exceeds the assigned memory. The problem can be overcome by simply assigning more memory to critical variables, but there is another elegant method that works in particular for $P_{l.s.}$. To be precise, we prove that $P_{l.s.}$ can be interpreted as the Cumulative Distribution Function (CDF) of a known distribution; we then rely on the `scipy.stats.binom.cdf()` function to compute $P_{l.s.}$.

When $P > N$, we recall the formula for $P_{l.s.}(P, N)$. Let us also rewrite it in an advantageous form:

$$P_{l.s.}(P,N) = 2^{1-P} \sum_{i=0}^{N-1} \binom{P-1}{i} = \sum_{i=0}^{N-1} \binom{P-1}{i} \left(\frac{1}{2}\right)^i \left(\frac{1}{2}\right)^{P-1-i}$$

Inside the summation, we recognize the probability mass function of the binomial distribution. Indeed, for a binomial distribution with $P-1$ trials, each having a success probability of $1/2$, the probability that there are precisely i successes is $\binom{P-1}{i}\left(\frac{1}{2}\right)^i\left(\frac{1}{2}\right)^{P-1-i}$. When i is summed for all values in between 0 and $N-1$, we obtain the probability that there are at most $N-1$ successes $\Rightarrow$ recover the formula for the CDF. In mathematical terms:

[1] This condition introduces a bias towards *underestimating* the fraction of l.s. data sets. If a data set is not l.s., then a non-convergence is the expected outcome. However, if a data set is l.s. and the number of epochs necessary for convergence is larger than what is set in the simulation, the data set is misclassified as non-l.s.

$$P_{l.s.}(P, N) = F(N-1|P-1, 0.5)$$

where $F(.|P-1, 0.5)$ is the CDF of the binomial distribution with $P-1$ trials and probability of success 0.5. This formulation generalizes immediately for $P \leq N$ because the CDF is well-defined on the entire real line; when $P \leq N$, the CDF equals 1 (exactly the intended behaviour for $P_{l.s.}$).

Experimental setup

In our study, we have set the following:

- the dimensions of the vectors: $N \in \{20, 40, 100\}$
- the number of data points, given through $\alpha = P/N$: $\alpha \in \{0.75, 1.0, 1.25, \ldots, 3.0\}$
- the number of data sets generated for each fixed N and α: $n_D = 100$
- the maximum number epochs to be used for Rosenblatt's algorithm: $n_{\max} = 5000$

Our objective is to obtain the empirical fraction of l.s. data sets $Q_{l.s.}$. The logic by which it is obtained is shown schematically in Algorithm 2.

Algorithm 2 Calculation of $Q_{l.s.}$ and $P_{l.s.}$ for all parameters

Require: *N_list*, *alpha_list*, n_D, $n_{\max}$
  for $N \in$ *N_list* **do** ▷ Iterate via dimensions
    compute *P_list* $\leftarrow N \cdot$ alpha_list (rounded to nearest integer)
    for $P \in$ *P_list* **do** ▷ Iterate via data set sizes
      compute $P_{l.s.}(\alpha)$ for N using the CDF
      initialize seed and $Q_{l.s.}(\alpha) \leftarrow 0$
      for $i \leftarrow 1$; $i \leq n_D$; $i \leftarrow i+1$ **do** ▷ Iterate via n_D data sets
        generate data set with N, P
        run Rosenblatt's algorithm with $n_{\max}$ ▷ Verify data set is l.s.
        if all labels correctly classified **then**
          $Q_{l.s.}(\alpha) \leftarrow Q_{l.s.}(\alpha) + 1/n_D$
        end if
      end for
    end for
    store $P_{l.s.}$, $Q_{l.s.}$ for N
  end for
    return vectors $Q_{l.s.}$ and $P_{l.s.}$, for each N

Implementation and tricks. Our code has been written in a Python Notebook, which was run on the cloud platform Google Colab. The following libraries have been imported: `numpy`, `matplotlib.pyplot` and `scipy.stats`.

Python, and in particular the `numpy` library, allow for compact and efficient computations that effectively reduce amount of code compared to what would have been suggested by our pseudo-code. For example, the computation of $Q_{l.s.}$ given N is done compactly through list comprehension:

```
np.array([get_Qls(P, N, seed=42) for P in P_list])
```

where `get_Qls()` is the function which summarizes the loop in Algorithm 2 that iterates through the data sets n_D. Another compactification is for obtaining the potentials E^μ given a data set and a perceptron $\mathbf{w}$, achieved via `numpy` matrix multiplications:

```
E = (data[0] @ w) * data[1]
```

where `data[0]` is a (`numpy` array) matrix where each row μ is ξ^μ and `data[1]` is a vector containing the labels S^μ in order. The multiplication between the resulting vectors is element-wise. Verifying that all points are correctly classified is as simple as running:

```
np.sum(E <= c) == 0
```

The expression $E \leq c$ generates a `numpy` array of Booleans: at the μ-th position, it is true if $E^\mu \leq c$ and false otherwise. The summation returns the number of true elements in the array (i.e. the number of misclassifications), which is compared to 0.

Results

Figure 1 summarizes the results of our main investigation by displaying both the theoretical value for $P_{l.s.}$, as well as its empirical estimate $Q_{l.s.}$.

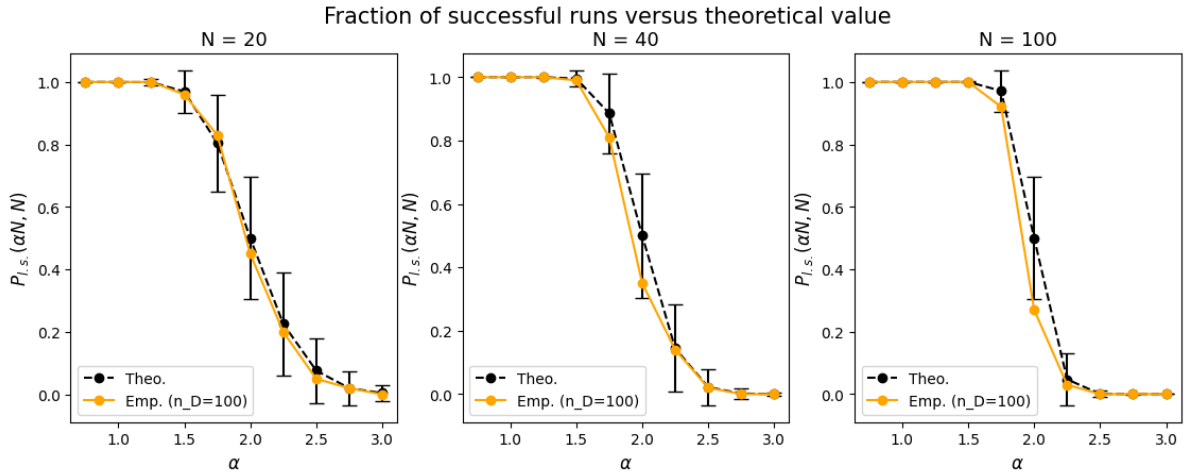

Figure 1: The fraction $P_{l.s.}(\alpha)$, both with the theoretical values (black) and verified empirically by $Q_{l.s.}(\alpha)$ (colour). The error-bars highlight the acceptance region of an asymptotic statistical test level 5% (more details in the Appendix), with the null hypothesis that $Q_{l.s.}$ is an unbiased estimate for $P_{l.s.}$

Discussion

Two main facts may be observed in Figure 1:

1. In each plot, the two curves tend to differ the most for $\alpha \in [1.75, 2.25]$ (with peak error at $\alpha = 2$). Not surprisingly, this is also where the acceptance region is the largest.

2. As N increases, the two curves differ more and more in the region $\alpha \in [1.75, 2.25]$ until the case $(N, \alpha) = (100, 2)$ where the empirical result is outside the acceptance region.

Generally, the disagreement between the two curves is impacted by the maximum number of epochs. In fact, while the Rosenblatt perceptron algorithm converges in a finite number of iterations, determining a sufficient number of training epochs *a priori* to training, for a given data set, is very difficult. This implies that, if the algorithm does not converge for a given dataset, we cannot conclude whether the dataset is non-linearly separable or simply that we did not use enough iterations [1, p.42] (which we mentioned in an earlier footnote).

However, it may not be evident why the error between the curves is so evident for αs around 2. We hypothesize the following

- For $\alpha \leq 1.5$, the number of data points is not large with respect to the dimension of the space, and therefore the data is likely going to be linearly separable and correctly classified as the dataset is not very complex.
- When $\alpha \geq 2.5$, even if a linearly separable dataset is generated, it will likely be misclassified. However, generation of linearly separable data sets for large α is unlikely, as implied by $P_{l.s.}(\alpha)$ and thus misclassifications have a negligible impact on $Q_{l.s.}$.
- Lastly, for the remaining values of α, the chance of misclassification increases, while it is still likely that the randomly generated dataset is linearly separable, which explains the larger discrepancy.

Role of n_D. The role of n_D in our investigation can be well understood by studying its impact on $Q_{l.s.}$. We recall that n_D is the number of simulated data sets for each (N, α) pair. The fraction $Q_{l.s.}$ is defined as

$$Q_{l.s.} = \frac{1}{n_D} \sum_{i=1}^{n_D} 1\,(i\text{-th data set is l.s.})$$

where the argument inside the sum is an indicator function. $Q_{l.s.}$ is an empirical approximator to $P_{l.s.}$, computed using n_D samples. Provided that $P_{l.s.}$ is the true fraction of linearly separable data sets, the Law of Large Numbers guarantees that $Q_{l.s.}$ converges almost surely to $P_{l.s.}$ as $n_D \to \infty$. In other words, the larger n_D is, the better the approximation given by $Q_{l.s.}$ becomes.

There is a deeper layer to n_D's role. Because of the inherent randomness in the simulated samples, the probability that $Q_{l.s.}$ equals $P_{l.s.}$ when $n_D < \infty$ is vanishingly small. However, it is important to judge how large fluctuations from the expected behaviour can become before we start to question the theory. This line of reasoning leads directly to *statistical hypothesis testing*, which we further expand on in Appendix A. In that context, n_D affects the *power* of the statistical test: if n_D increases, then the probability of making type I or type II errors decreases. Put differently, a large n_D helps to construct tight acceptance regions (i.e. error bars in Figure 1) that allow one to efficiently identify departures from the expected outcome.

Conclusion

In this paper, we have implemented the Rosenblatt perceptron algorithm and applied it to randomised data. In particular, the fraction of successfully classified datasets and the fraction of linearly separable functions are compared. The main observation we found was that for values of $\alpha \in [1.75, 2.25]$ the number of misclassified datasets increases while the number of linearly separable functions is still relatively high and thus we observe a significant discrepancy (particularly for large values of N) between the two quantities. However, that deviation can be plausibly explained by an insufficient number of training epochs, leading to a misclassification of linearly separable data sets as non linearly separable. Hence, through the statistical hypothesis test, we find no strong evidence against $Q_{l.s.}$ being an unbiased estimator for $P_{l.s.}$.

Individual contributions to the assignment

The division of tasks was fluid, as we have helped each other with ideas and verifications everywhere. Main contributions:

- Andrei Secuiu: Introduction, Method.
- Roberto Schinina: Discussion, Conclusion.
- Both: The coding was done in parallel and the results were compared

For the next project we are going to swap the sections.

References

[1] Michael Biehl. *The Shallow and the Deep*. University of Groningen Press, 2023.

Bonus

Mathematical proof for its limiting behaviour. Re-interpreting the fraction $P_{l.s.}$ as a CDF unlocks a neat proof for the storage capacity of the perceptron using the Central Limit Theorem. Let $X \sim \text{Bin}(P-1, 0.5)$ be a random variable (RV) distributed s.t. its CDF is $P_{l.s.}$. Because X is binomially distributed, it can be written as the sum of $P-1$ i.i.d. Bernoulli RVs with probability 0.5. The Central Limit Theorem written for the $P-1$ Bernoulli RVs allows us to obtain

$$\mathbb{P}\left(\sqrt{P-1}\frac{\frac{X}{P-1}-0.5}{\sqrt{0.5(1-0.5)}} \leq t\right) = \mathbb{P}\left(\sqrt{P-1}\frac{\frac{X}{P-1}-0.5}{0.5} \leq t\right) \to \mathbb{P}\left(Z \leq t\right) = \Phi(t) \quad \text{as } P \to \infty$$

where $Z \sim \text{N}(0,1)$ is a standard normal variable, and $\Phi(.)$ is the CDF of the standard normal distribution. The fraction $P_{l.s.}(P,N) = P_{l.s.}(\alpha N, N) = P_{l.s.}(\alpha)$ becomes

$$\begin{aligned} P_{l.s.}(\alpha) = F(N-1|P-1,0.5) &= \mathbb{P}\left(X \leq P/\alpha - 1\right) = \mathbb{P}\left(\sqrt{P-1}\frac{\frac{X}{P-1}-0.5}{0.5} \leq \sqrt{P-1}\frac{\frac{P/\alpha-1}{P-1}-0.5}{0.5}\right) \\ &= \Phi\left(\sqrt{P-1}\frac{\frac{P/\alpha-1}{P-1}-0.5}{0.5}\right) + o(1) \end{aligned}$$

where in the latest equality we use the Central Limit Theorem. It remains to evaluate the behaviour of the argument of $\Phi(.)$ when $P \to \infty$, from which we recover the asymptotics of $P_{l.s.}$.

$$\begin{aligned} \sqrt{P-1}\frac{\frac{P/\alpha-1}{P-1}-0.5}{0.5} = \sqrt{P-1}\frac{2}{\alpha}\left(\frac{P-\alpha}{P-1} - \frac{\alpha}{2}\right) = \sqrt{P-1}\frac{2}{\alpha}\frac{(2-\alpha)P-\alpha}{2(P-1)} &= \frac{(2-\alpha)P-\alpha}{\alpha\sqrt{P-1}} \\ \to \begin{cases} +\infty & \alpha < 2 \\ -\infty & \alpha > 2 \\ 0 & \alpha = 2 \end{cases} & \quad \text{as } P \to \infty \end{aligned}$$

Based on the above, we recover immediately

$$P_{l.s.}(\alpha) \to \begin{cases} \Phi(+\infty) + 0 & \alpha < 2 \\ \Phi(-\infty) + 0 & \alpha > 2 \\ \Phi(0) + 0 & \alpha = 2 \end{cases} = \begin{cases} 1 & \alpha < 2 \\ 0 & \alpha > 2 \\ \frac{1}{2} & \alpha = 2 \end{cases} \quad \text{as } P \to \infty$$

Empirical validation. The empirical estimation of $P_{l.s.}$ via $Q_{l.s.}$ has been achieved in a manner identical to that of Figure 1. However, there were several changes. These were done because of time constraints; keeping $n_{\max}$ to the previous value would have required a run time of at least 83 hours.

- $\alpha \in \{1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5\}$.
- $n_{\max} = 100$.
- $n_D = 100$.
- $N \in \{100, 500, 1000\}$.

The results are shown in Figure 2. Observations:

- $N = 100$ has been verified in both Figures 1 and 2. In the latter, the agreement between theory and empirical validation is significantly worse. As the key difference between experiments was the decrease of $n_{\max}$, our hypothesis on why $P_{l.s.}$ and $Q_{l.s.}$ diverge is strengthened

- $Q_{l.s.}$ is much more different than $P_{l.s.}$. It can be explained by the fact that $n_{\max}$ is much lower than in the main report, leading to more misclassifications of l.s. data sets as non-l.s. Increasing N indirectly increases the number of data points $P = \alpha N$, making the data set more complex and harder to learn. That exacerbates the problem that causes the bias discussed in the main report

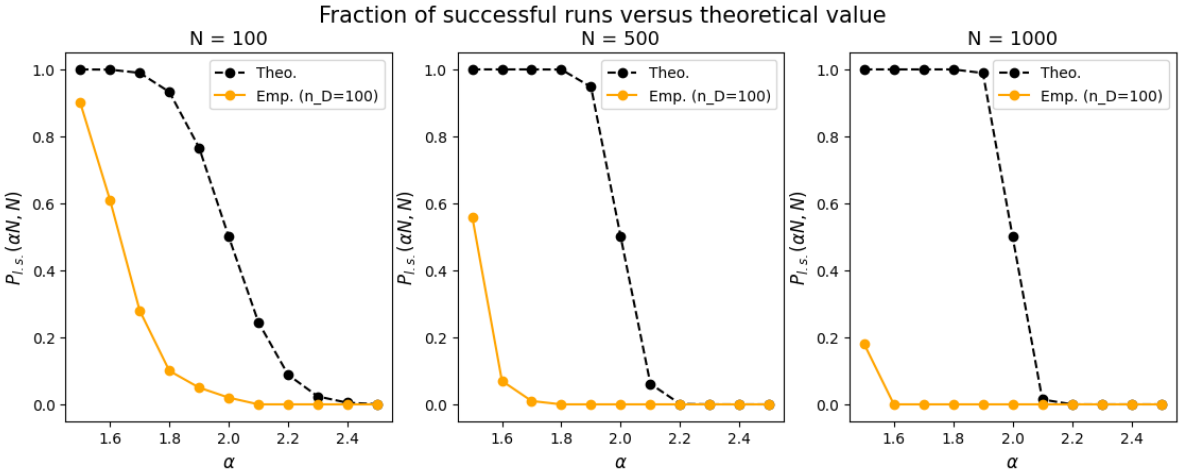

Figure 2: An equivalent plot to that of Figure 1, but for different large values of N and a finer, shorter grid of α values

Conclusion. While our mathematical proof for the limiting behaviour of $P_{l.s.}$ matches the known results from the lecture, the empirical results do not support it. Nonetheless, we have identified a plausible cause for the divergence between the two results. Thus, the theory cannot be rejected nor confirmed based on the values in Figure 2.

A Statistical hypothesis testing for $Q_{l.s.}$

A key part of the assignment is to estimate the fraction $Q_{l.s.}$ empirically and compare it to the known formula $P_{l.s.}$. Due to the inherent stochasticity of the simulation, it is extremely unlikely that $Q_{l.s.}$ equals precisely $P_{l.s.}$ from only a finite sample. Thus, we would like to quantify how large the deviation between $Q_{l.s.}$ and $P_{l.s.}$ can become, and still plausibly claim that $Q_{l.s.}$ is an estimator for $P_{l.s.}$. To answer this query, we turn to statistical hypothesis testing.

Let us fix an α and let us denote $P_{l.s.}(\alpha) = p$. The *null hypothesis* is that the generation mechanism in the assignment produces a l.s. data set with probability p. From n_D generated data sets, $Q_{l.s.}$ can be written with the help of indicator functions:

$$Q_{l.s.} = \frac{1}{n_D} \sum_{i=1}^{n_D} 1\left(i\text{-th data set is l.s.}\right)$$

Each indicator function is a Bernoulli RV with parameter p. These Bernoulli RVs are independent due to how the data has been generated, so the Central Limit Theorem is applicable. Knowing the expectation and variance of each Bernoulli RV, we obtain:

$$\lim_{n_D \to \infty} \mathbb{P}\left(\sqrt{n_D} \frac{Q_{l.s.} - p}{\sqrt{p(1-p)}} \leq t \right) = \Phi(t)$$

where $\Phi(.)$ is the CDF of the standard normal distribution. The above allows us to derive an *asymptotic statistical test* level β for the given null hypothesis. The decision rule is

$$\begin{cases} \text{Accept } H_0 & \text{if} \quad Q_{l.s.} \in \left[p - \sqrt{\frac{p(1-p)}{n_D}} z_{1-\beta/2},\ p + \sqrt{\frac{p(1-p)}{n_D}} z_{1-\beta/2} \right] \\ \text{Reject } H_0 & \text{otherwise} \end{cases}$$

where $\Phi(z_{1-\beta/2}) = 1 - \beta/2$. The level of the test is asymptotically β:

$$\begin{aligned} \mathbb{P}\left(\text{Reject } H_0 | H_0 \text{ true}\right) &= 1 - \mathbb{P}\left(Q_{l.s.} \in \left[p - \sqrt{\frac{p(1-p)}{n_D}} z_{1-\beta/2},\ p + \sqrt{\frac{p(1-p)}{n_D}} z_{1-\beta/2} \right] | H_0 \text{ true} \right) \\ &= 1 - \mathbb{P}\left(\sqrt{n_D} \frac{Q_{l.s.} - p}{\sqrt{p(1-p)}} \in \left[-z_{1-\beta/2},\ z_{1-\beta/2} \right] | H_0 \text{ true} \right) \approx 1 - \Phi\left(z_{1-\beta/2}\right) + \Phi\left(-z_{1-\beta/2}\right) \\ &= 1 - 1 + \frac{\beta}{2} + \frac{\beta}{2} = \beta \end{aligned}$$

The higher n_D is, the better the approximation becomes. As a rule of thumb, $n_D \approx 30$ is already an adequate threshold for many statistical applications. In the results, we highlight both the theoretical limit, as well as the interval which allows us to accept the null hypothesis based on a finite sample size n_D with significance level $\beta = 5\%$.