LLN, meaning, proof and simulation.

Giacomo Babudri

December 20, 2023

1 Introduction

The law of large numbers (LLN) is a fundamental theorem in probability theory and statistics that describes the behavior of sample averages as the sample size increases. In particular it gives informations about the result of performing the same experiment a large number of times. According to the law, the average of the results obtained froma a large numbers of trials, should be close to the expected value, and tends to become closer to the expected value, as more trials are performed.

This law it's really important, because it gives crucial details, about the long-term results, for the average of some random events.

2 The Law of Large Numbers in detail

It is often divided into two main versions: the Weak Law of Large Numbers (WLLN) and the Strong Law of Large Numbers (SLLN).

1. ***Strong Law of Large Numbers*** The strong law of large numbers, states that the sample average converges almost surely to the expected value.
   P(lim n→ Xn=E[X])=1
   In other words as we explained above, with probability one, the sample mean(Xn) converges almost surely to the population mean(E[X]) as the sample size tends to infinity(n→).

2. ***Weak Law of Large Numbers*** The Weak Law of Large Numbers provides a probabilistic convergence result (states that the sample average converges in probability towards the expected value).
   lim n→P(Xn E[X] ¿)=0, ¿0
   In simpler terms, as the sample size (n) approaches infinity, the probability that the sample mean (Xn) deviates significantly from the population mean (E[X]) tends to zero.

The difference beetween the two laws, is purely theorical, stating just that the strong law is more precise than the weak one.

3 Proof

Introducing this division in the LLN, is really important for the execution of the proof. Infact, even if the difference is small, based on the kind of LLN we are treating, the demonstration is built differently.

3.1 *Demonstration of the Weak Law of Large Numbers:*

Premise Let X1,X2,...,Xn, be independent and identically distributed random variables with an expected value E(X) and finite variance Var(X).

Proof
Define the sample mean as $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$.
Apply Chebyshev's Inequality to the sample mean:
P$(|Xn - E(X)| \geq \epsilon) \leq \frac{Var(X)}{n\epsilon^2}$

In conclusion As n approaches infinity, the right-hand side of the inequality goes to zero, showing that Xn converges in probability to E(X). All the passages of the Chebyshev's Inequality, weren't shown, because i decided to show only the key passages. These were infact enough to proof the Weak Law of Large Number

3.2 *Demonstration of the Strong Law of Large Numbers:*

Premise Let X1,X2,...,Xn, be independent and identically distributed random variables with an expected value E(X) (assuming finite) and a finite mean).

Proof
Define the sample mean as $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$.
Using the Borel-Cantelli Lemma it's easy to show that the probability of the sample mean deviating from the expected value infinitely often is zero.
P$\left(\limsup_{n\to\infty} |\bar{X}_n - E(X)| > \epsilon\right) = 0$

Conclude that Xn converges almost surely to E(X), meaning that the set of sample points where convergence doesn't occur has a probability measure of zero.

Also in this case was taken the decision to treat only the key parts of the demonstrations, focusing less on the long and detailed mathematical operations, leadeing to this key points.

4 Examples

Casino in the casino a random event as a spin in a slot machine, can bring to a win, but in long term the earnings will tend to a preditcable percentage, over a large number of spins.

Coin If a fair coin (one with probability of heads equal to 1/2) is flipped a large number of times, the proportion of heads will tend to get closer to 1/2 as the number of tosses increases.

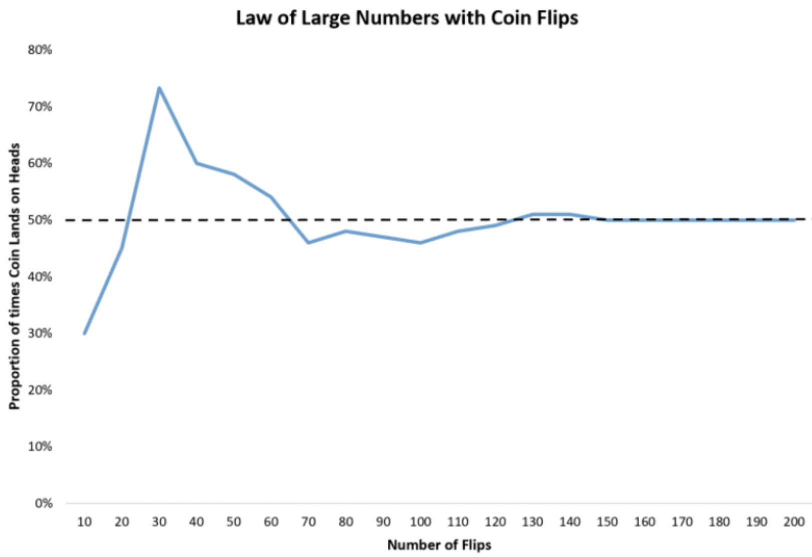

Figure 1: Representation of the flip of a coin over a large number of trials

Dice If we roll the dice only three times, the average of the obtained results may be far from the expected value. Let's say you rolled the dice three times and the outcomes were 6, 6, 3. The average of the results is 5. According to the law of the large numbers, if we roll the dice a large number of times, the average result will be closer to the expected value of 3.5.

5 Simulation

During the course were given as tasks, to execute many differents kind of simulation. In particular in one simulation was asked to: Simulate the score "trajectories" for M systems that are subject to a series of N attacks. On the x-axis, we indicate the attacks and on the Y-axis we simulate the accumulation of a "security score" (-1, 1), where the score is -1 if the system is penetrated and 1 if the system was successfully "shielded" or protected. assuming, for simplicity, a constant penetration probability p at each attack.

It's interesting to study the output of this simulation in two differents cases first with a large N and then with a low N, to see how the representation will change. P will be fixed as 0.5.

First case:
N = 500, M = 10, P = 0.5: In this case it's evident how most of the representations of the systems, with an increasing number of attacks, go towards the expected value. Infact we would expect a probability of 0.5, it's easy to calculate that on 500 attacks, most of the systems succeeds around 225 and 275 times

(p1 = 0.45 and p2 = 0.55). It's pretty close to what we would expect.

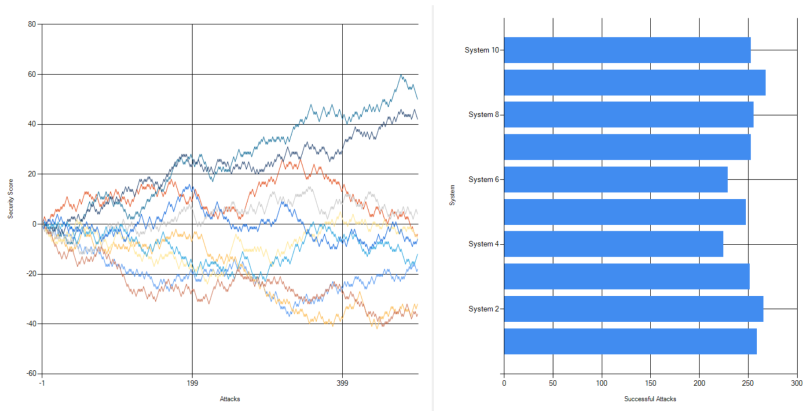

Figure 2: Representation of a positive case

Second case: N = 10, M = 10, P = 0.5: In this case the representations it's much more impredictable. Over five systems infact, in average a system succeeds beetween 4 and 9 attacks, with an average of circa seven successes. In probability, this it's equal to an average success rate of 0.7, very different to the value expected at the start(0.5)

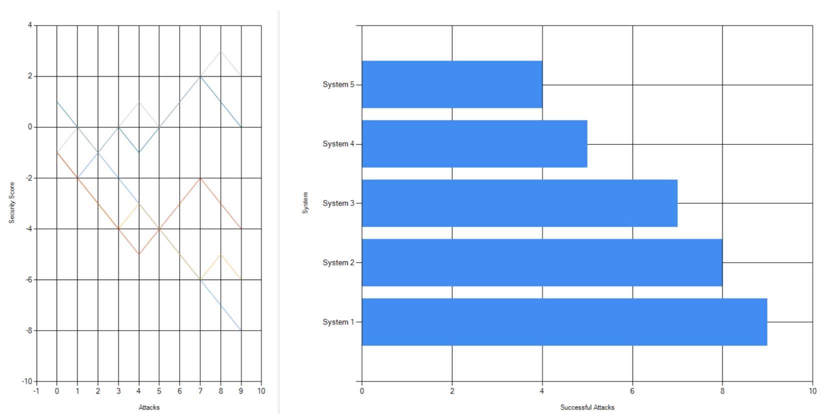

Figure 3: Enter Caption

The simulation was executed based off this simple c# code, that takes in input the number of systems, attacks and the probability, shown below. It was in the end represented through a linechart and a barchart:

```
1 riferimento
private int[][] AttacksLanded(int M, int N, double p)
{

    Random random = new Random();
    int[][] Systems = new int[M][];
    for (int i = 0; i < M; i++)
    {
        int[] attacks = new int[N];
        for (int j = 0; j < N; j++)
        {

            double att = random.NextDouble();
            if (att < p)
            {
                attacks[j] = 1;
            }
            else
            {
                attacks[j] = -1;
            }
        }
        Systems[i] = attacks;

    }
    return Systems;

}
```

Figure 4: Code of the simulation