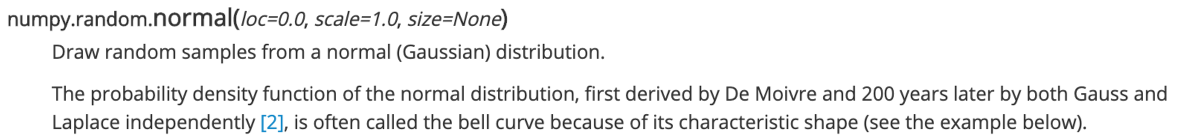
Algo-Project-Part1

Part 1 : The SnowPlow Problem

To solve this problem, we first use a dummy recursive algorithm that check every possibility to get the best result. The issue of this algorithm is that the complexity is exponential and we just can't resolve it quickly.

In second time, we tried to start with all positives homes or all negatives homes, according to the biggest number of home. But now, we always lose against the dummy algorithm.

In a third time, we thought that we should begin with the biggest density of homes in the array, to clean a maximum of homes quickly. At the begin we couldn't build a good algorithm that works and always beat the dummy algorithm. We tried a lot of stuff until we found out that the houses were created with a normal Gaussian distribution.



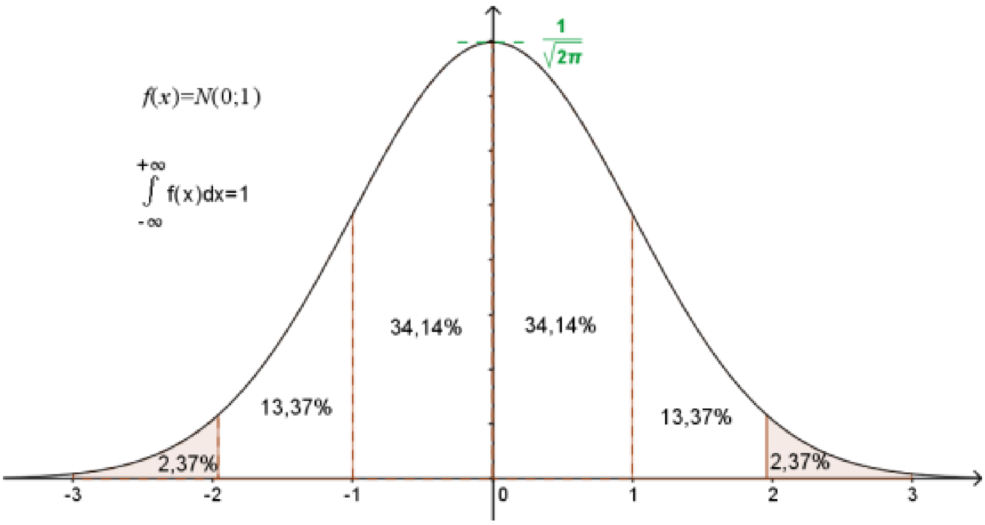
numpy.random.normal(*loc=0.0*, *scale=1.0*, *size=None*)

Draw random samples from a normal (Gaussian) distribution.

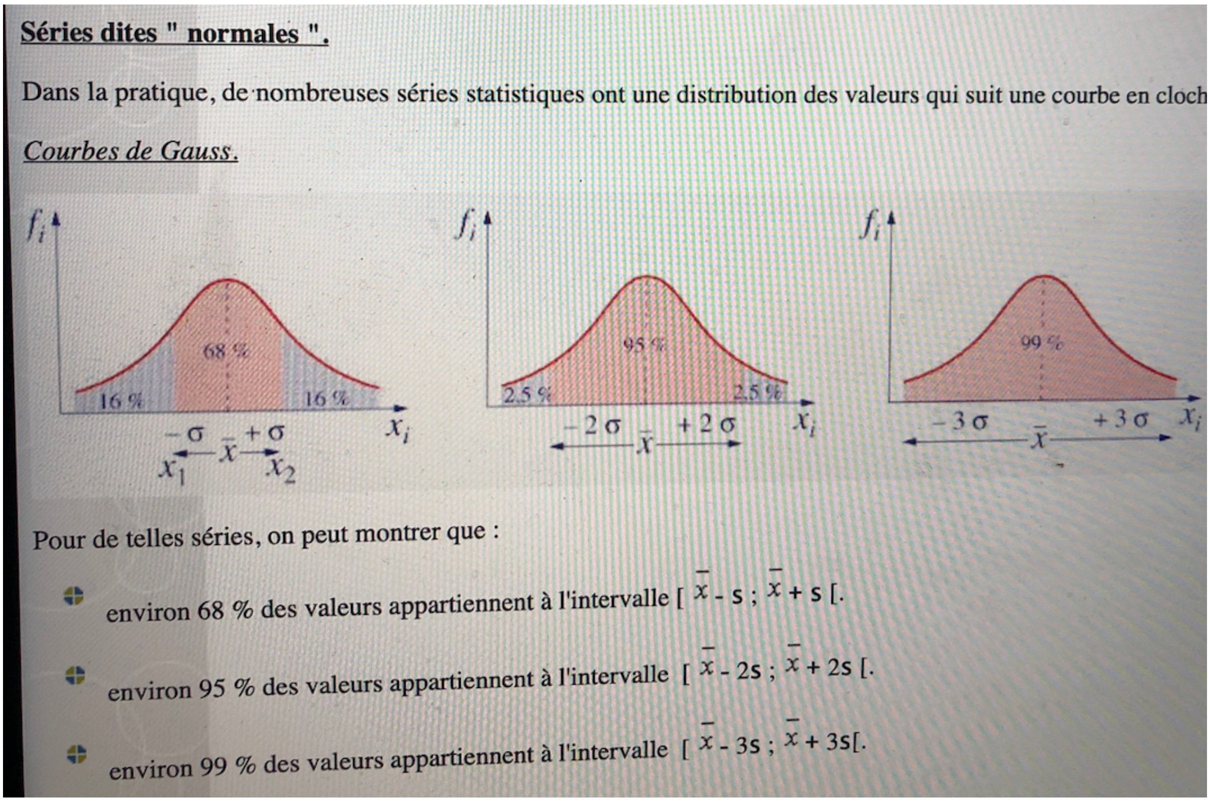
The probability density function of the normal distribution, first derived by De Moivre and 200 years later by both Gauss and Laplace independently [2], is often called the bell curve because of its characteristic shape (see the example below).

> https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.random.normal.html

Here is the curve of a normal Gaussian distribution :



> http://unt-ori2.crihan.fr/unspf/2010_Limoges_Vignoles_StatsDescriptives/co/04-3-3 loi normale reduite.html



Séries dites " normales ".

Dans la pratique, de nombreuses séries statistiques ont une distribution des valeurs qui suit une courbe en cloch

Courbes de Gauss.

Pour de telles séries, on peut montrer que :

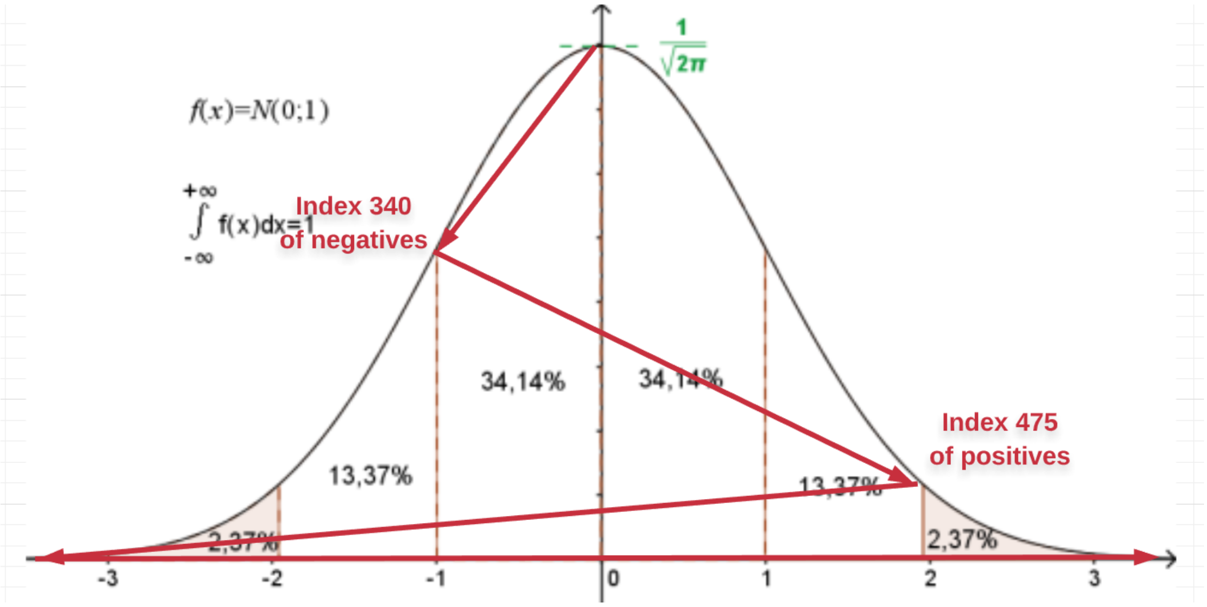
- environ 68 % des valeurs appartiennent à l'intervalle [$\bar{x}$ - s ; $\bar{x}$ + s [.
- environ 95 % des valeurs appartiennent à l'intervalle [$\bar{x}$ - 2s ; $\bar{x}$ + 2s [.
- environ 99 % des valeurs appartiennent à l'intervalle [$\bar{x}$ - 3s ; $\bar{x}$ + 3s[.

So with this distribution, we know that about 68% of homes are in a certain interval between 0. It's really great for us, because they are close to our snowplow, so we can go over a large number of homes in a small distance.

We could use a formula to get the real percentage but we are missing of time so we just used approximative indexes.

Here is our goal :

- Sort and divide our array with two arrays, one of positive homes and one of negative homes.
- Start with the array that have the biggest length.
- Go to the first 340 homes (first 68% interval : 1000 * 0.68 = 680 ; 680 / 2 = 340)
- Next, go to the first 475 homes of the other array (second 95% interval, same operation)
- After that, we clean all homes of the first array
- And to finish, we clean all homes of the second array.



Our algorithm is n log(n), because the method sorting of python is n log(n) complexity, and the rest of our algorithm is just to browse tables (n complexity).