Practice 13: Law of large numbers.

Mayra Cristina Berrones Reyes 6291

December 1, 2020

1 Introduction

The Law of Large Numbers (LLN) is a theorem of probability theory, which describes the results of performing the same experiment a large number of times. In broad terms, this law states that the average of the results by a large number of trials should be close to the expected value, and as more trials are performed, this result should appear closer and closer [4].

Explained in a more mathematical approach, we have Theorem 1.1 for the LLN [5].

Theorem 1.1 (**Law of Large Numbers**). *Let $X_1, X_2, ..., X_n$ be an independent trial process, with finite expected value $\mu = E(X_j)$ and finite variance $\sigma^2 = X(X_j)$. Let $S_n = X_1 + X_2 + ... + X_n$. Then for any $\epsilon > 0$,*

$$P\left(\left|\frac{S_n}{n} - \mu\right| \geq \epsilon\right) \to 0$$

as $n \to \infty$. Equivalently,

$$P\left(\left|\frac{S_n}{n} - \mu\right| < \epsilon\right) \to 1$$

as $n \to \infty$.

■

1.1 Example: Die roll

To better understand the LLN we also have an example 1 from the same book [5], replacing some things from the Theorem 1.1 we have,

Example 1. *Consider n rolls of a die. Let X_j be the outcome of the jth roll. Then $S_n = X_1 + X_2 + ... + X_n$ is the sum of the first n rolls. This is an independent trials process with $E(X_j) = 7/2$. Thus, by the Law of Large Numbers, for any $\epsilon > 0$*

$$P\left(\left|\frac{S_n}{n} - \frac{7}{2}\right| \geq \epsilon\right) \to 0$$

as $n \to \infty$. An equivalent way to state this is that, for any $\epsilon > 0$,

$$P\left(\left|\frac{S_n}{n} - \frac{7}{2}\right| < \epsilon\right) \to 1$$

as $n \to \infty$.

■

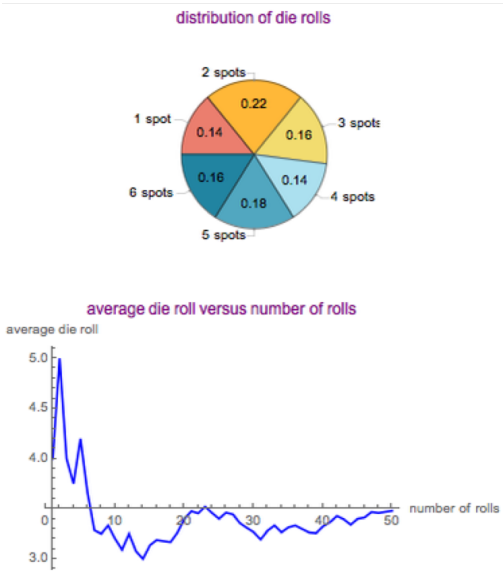
This example tells us that, according to the LLN if a large number of six sided fair dice are rolled, then the average of their values is likely to be close tho 3.5, with an increasing precision the more dice are rolled. Thanks to a tool from Wolfram we are able to visualize the different stages of this process in Figure 1.

As we can see in Figure 1 the mean is set to be 3.5, and as the iterations keep building, the line representing the average sum gets closer to converge to the mean line.

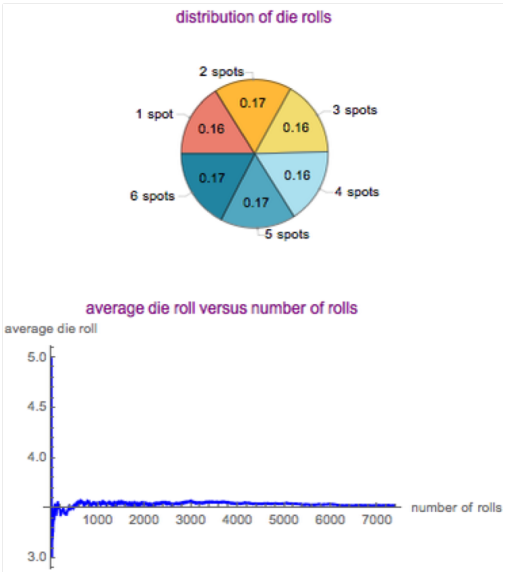
We can also develop a similar experiment in `R`, with simple functions such as `sample`. In this case, we take the average sum of two rolled dice. In Figure 2 we represent with a lollipop plot the behavior of the different sizes of iterations of the experiment.

```r
two.dice <- function(){
  dice <- sample(1:6, size = 2, replace = TRUE)
  return(sum(dice))
}

two.dice()
replicate(n = 20, expr = two.dice())

sims <- replicate(100, two.dice())
table(sims)
df = as.data.frame(table(sims)/length(sims))
barplot(table(sims)/length(sims), xlab = 'Sum', ylab = 'Relative Frequency', main
    = '100 Rolls of 2 Fair Dice')

# Libraries
library(ggplot2)

# Create data
data <- data.frame(x=df$sims,y=df$Freq)

# Plot
png("Ej13_dice1.png", width = 1000, height = 1300, res = 300)
ggplot(data, aes(x=x, y=y)) +
  geom_segment( aes(x=x, xend=x, y=0, yend=y), color="grey") +
  geom_point( color="orange", size=4) +
  theme_light() +
  theme(panel.grid.major.x = element_blank(),
    panel.border = element_blank(),
    axis.ticks.x = element_blank()) +
  xlab("Sum") +
  ylab("Relative Frequency")
dev.off()
```

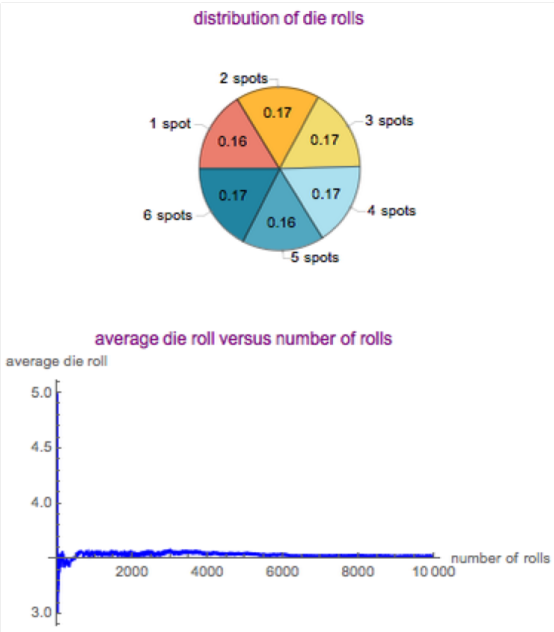
Listing 1: R extract of the code used to perform the dice experiment



(a) 50 rolls of the dice

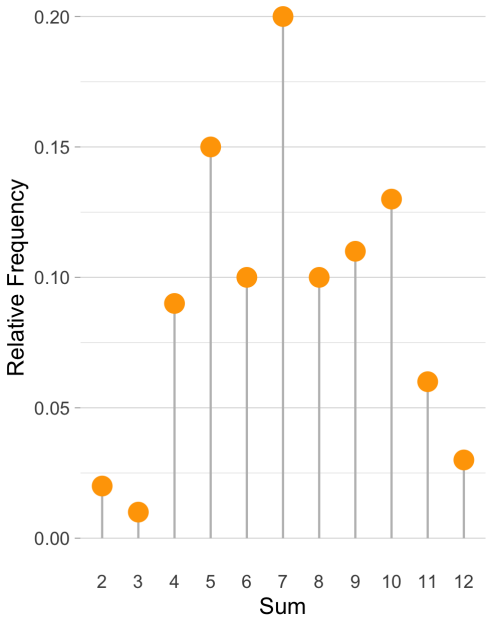


(b) 7,500 rolls of the dice

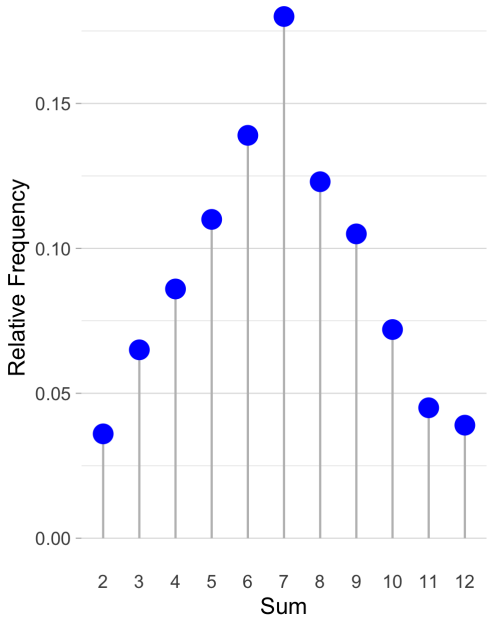


(c) 10,000 rolls of the dice.

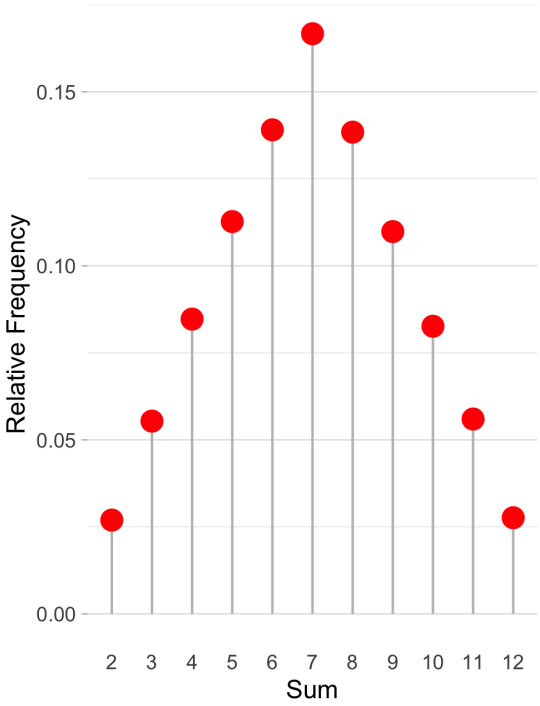
Figure 1: Different parameters for the dice experiment in the Wolfram page.



(a) 100 rolls of the dice



(b) 1,000 rolls of the dice



(c) 10,000 rolls of the dice.

Figure 2: Different parameters for the dice experiment performed in R.

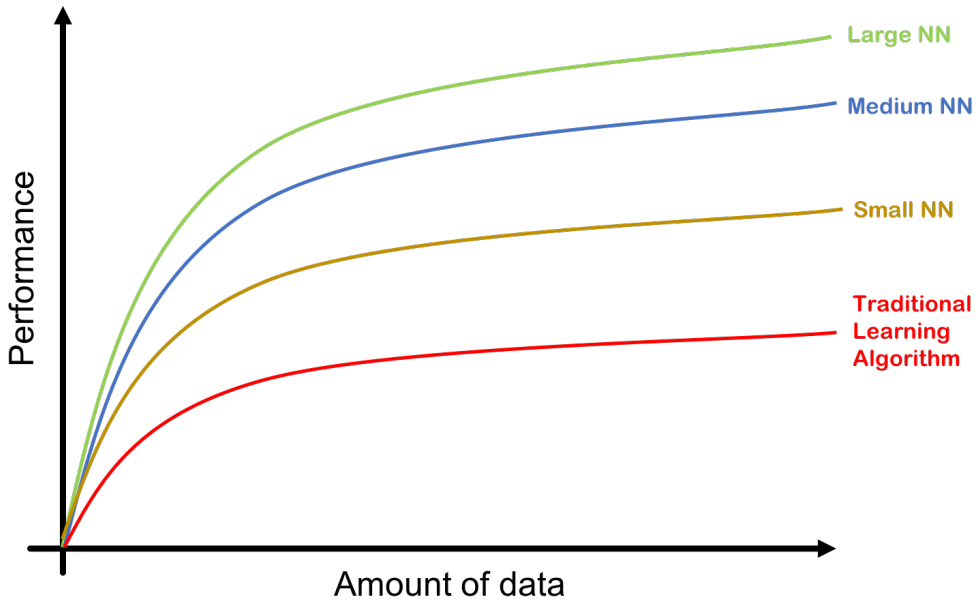


Figure 3: Representation of the learning curve with more amount of data.

2 Applications of Law of Large Numbers in Machine Learning

Now that the concept of the law of large number is presented, we searched for a topic related to the work of our thesis. In this case, there is a question that has always been present when working with Machine Learning tools such as Neural Networks (NN), and in this case is, how much data do I need to properly train my NN.

It all comes down to accuracy. When a experiment is performed, an accurate and precise result is desired. Accuracy of an experiment is defined as to whether or not the result of a measurement conforms to the correct value or expected value. Precision refers to the degree to which these values agree and can have a repeatable outcome if we performed the experiment again. The basic notion of this comes from the Law of Large Numbers (LLN), one of the basic principles of experimental physics and statistics. The law states that the average of an experiment performed many times converges to its true or expectation value.

In ML applications, this can translate as to training our NN with a defined set of data, and then testing this model with some more sets of data. The performance of the model we create depends upon the amount of data we use. When the data increases, we force the algorithm to fit the data, and with that, we minimize the error [1].

In Figure 3 we see the representation often used in articles to show the same behavior represented in the example of the dice 1 we saw in the introduction, where we can see how the learning curve slowly flattens the more data we add to our model.

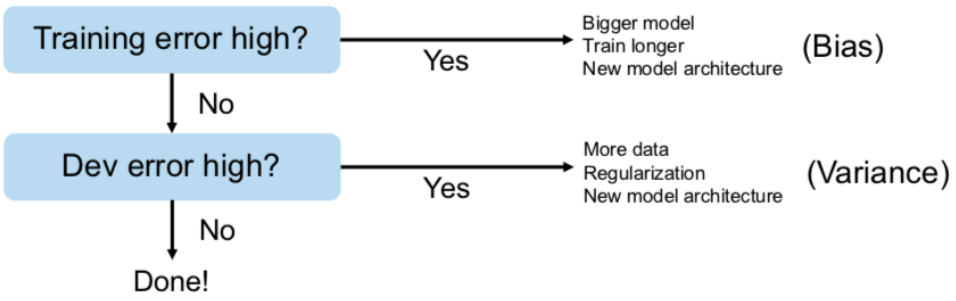


Figure 4: Basic recipe for machine learning [6]

Since our main goal is that our model gives us the best accuracy possible, we also have to consider the distribution of our data. Many cases have shown that the expected result of our test set will not match the accuracy performance of our training and developing sets if we use different distribution of data [6]. This also comes around to having a great amount of data to use for our model. Figure 4 shows the "basic recipe" for machine learning.

As one Google Translate engineer put it, "when you go from 10,000 training examples to 10 billion training examples, it all starts to work. Data trumps everything".

In our case, at the beginning of our Masters thesis [3], we started to develop toy CNN with the available data we had, which was not much. The maximum accuracy we could achieve for our image detection model, was of 65% to 70% in the test set. Later, when we were able to get more data sets, our accuracy improved to a 89% accuracy.

As a final note, some of the things we learned along the way, is that is not always possible to have a large data set available to train our models, and in this case, there has been several application of transfer learning, that basically uses data form a different distribution (in our case, normal images, not medical ones) to pre train the wights of the NN, and improve the results of our training and testing. We are currenty looking for more examples of this in the medical field, since the ones we already found are lacking in accuracy performance.

3 Conclusion

The theory of Large Numbers applied in Machine Learning was something we heard of in a conference of the ENOAN in Zacatecas that we were able to attend. In this case, it was a really interesting topic for us, because in our mentioned masters thesis, we had a little idea when it came to one of the optimizers we used to perform our experiments in the architecture of the classification model for our CNN.

The SGD or Stochastic Gradient Decent optimizer, was one of the more robust algorithms used, since in all of the models we trained, it consistently kept upgrading the accuracy in the model, ending up in a steady 85% to 95%, and the difference in percentage from the training and

tests sets where not so far apart. Even the optimizer we ended up using for the final experiment had some models in which the accuracy went bellow 40%.

In the conference we mentioned, Dr. Hugo Estrada Esquivel with his lecture titled "Ciencia de datos en la era del internet de las cosas" [2] commented about the experiments he had developed trying to stabilize the SGD optimizer. In his work, he found out, as we did, that the SGD optimizer is the most steady one, but also that it need significantly more time to converge in a optimal solution. Similar to the experiment in Figure 1 it gets close to the desired output, but eventually swerves again.

We did not follow up with this investigation, because it was very costly (computational wise) to perform this type of experiments, but it would be very interesting to dig up again some information about it, to see if it has made some progress.

References

[1] The law of large numbers in ai and big data. `https://discover.bot/bot-talk/law-of-large-numbers-ai/`. Accessed: 2020-11-30.

[2] Ciencia de datos en la era del internet de las cosas. `http://smcca.org.mx/enoan2019/plenariaHugoEstrada.html`. Accessed: 2020-11-30.

[3] Clasificación de mamografías mediante redes neuronales convolucionales. `http://eprints.uanl.mx/17656/`. Accessed: 2020-11-30.

[4] Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester. *A Modern Introduction to Probability and Statistics: Understanding why and how.* Springer Science & Business Media, 2005.

[5] Charles Miller Grinstead and James Laurie Snell. *Introduction to probability.* American Mathematical Soc., 2012.

[6] Andrew Ng. Machine learning yearning. *URL: http://www. mlyearning. org/(96)*, 2017.