I have been contacted by many people asking me to predict the outcome of some events that in theory are random. For example, they want me to predict lottery games like Keno, Lotto, Casino Roulette numbers and so on so forth. My answer is that you cannot predict something which is supposed to be random. No model can give you a better estimate than what you already know, for example, in roulette th probability to get the number 0 is 1/37 no matter what were the previous numbers. This implies that befor start building advanced Machine Learning and Artificial Intelligence models to predict the outcome of the next draw, try to check if these numbers are actually random. If there are actual random then it means tha there is no pattern and you should not waste your time with ML and AI.

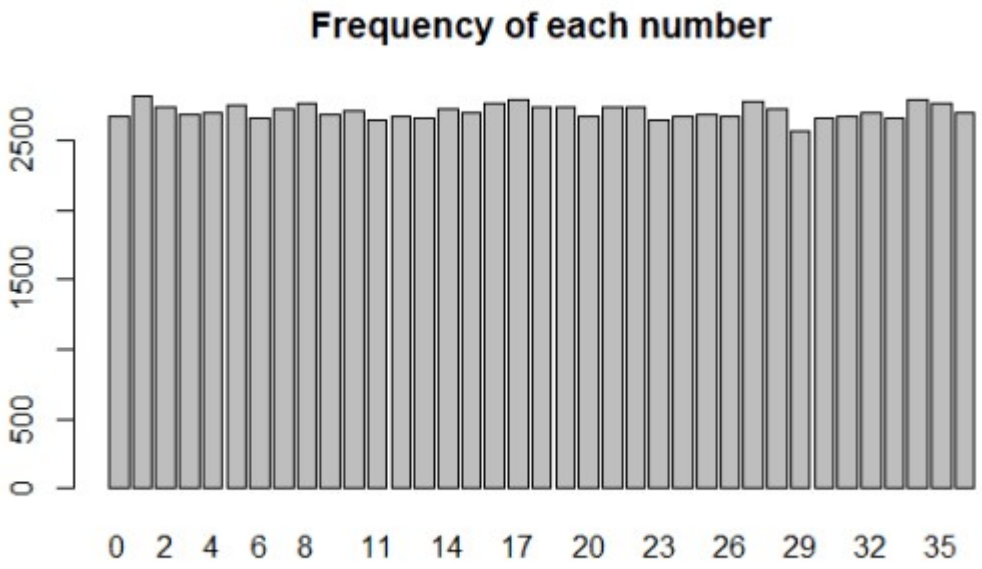
For demonstration purposes, we will assume that we are dealing with numbers obtained from an **unbiased casino roulette** with numbers from **0 to 36**. Let's create our sample in R.

```
set.seed(5)

# Generate 10K random numbers from 0 to 36
casino<-sample(c(0:36), 100000, replace = TRUE)
```

Chi-Square Test for the Frequency of the Numbers

In the beginning, we can test if the frequency of the drawn numbers is random. A barplot of the frequency of each number will help us to get a better idea.

```
barplot(table(casino), main="Frequency of each number")
```



Let's now run the Chi-Square test:

```
chisq.test(table(casino))
```

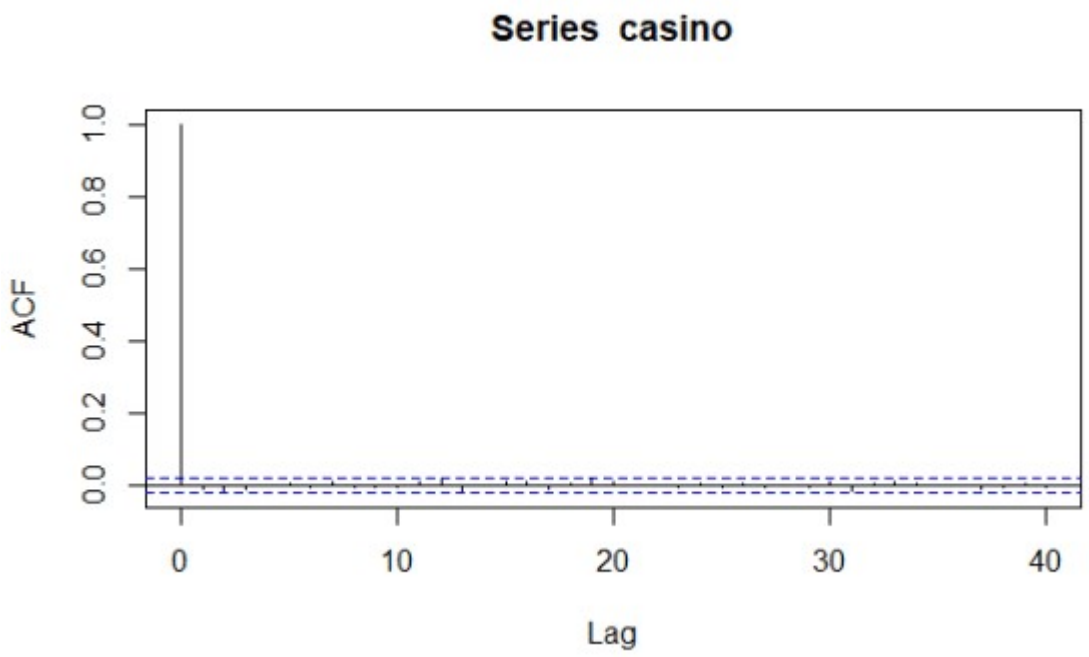
```
> chisq.test(table(casino))

        Chi-squared test for given probabilities

data:  table(casino)
X-squared = 44.471, df = 36, p-value = 0.1571
```

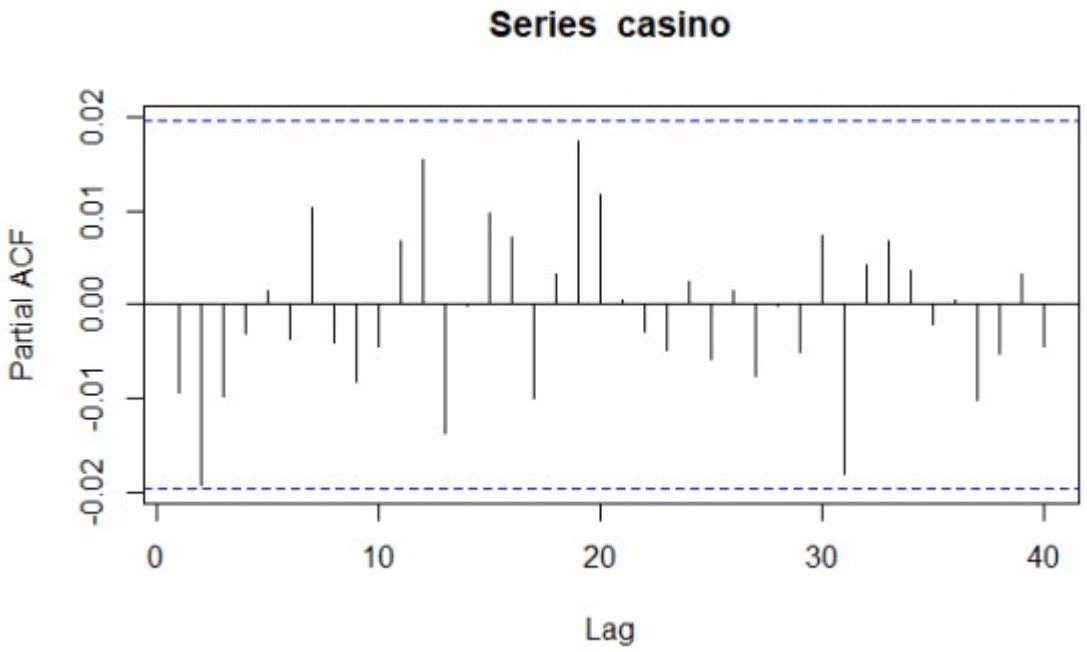
As we can see the p-value is greater than 5% which means that we **do not reject** the null hypothesis which was that the distribution of the digit was independent. Of course, this test does not check if there is a pattern in the way that the numbers are served.

Autocorrelation and Partial Autocorrelation

A quick way to see if there is a pattern in the way that the numbers are served is to plot the acf and pacf.

```
acf(casino)
pacf(casino)
```





If you see get similar plots as the above ones it means that there is no correlation between the current drawn number with the previous (lag) ones.

Wald-Wolfowitz Runs Test

Regarding the sequence of the numbers, we can apply the Wald-Wolfowitz Runs Test that is a non-parametric statistical test that checks a randomness hypothesis for a two-valued data sequence. More precisely, it can be used to test the hypothesis that the elements of the sequence are mutually independent. Notice that this was suggested for binary cases but the runs tests can be used to test the randomness of a distribution, by taking the data in the given order and marking with + the data greater than the median, and with – the data less than the median (numbers equalling the median are omitted.)

We can run this test using the `randtests` package in R. Let's do it for our sample data:

```
library(randtests)
runs.test(casino)
```

```
> runs.test(casino)

        Runs Test

data:  casino
statistic = 0.31562, runs = 4889, n1 = 4926, n2 = 4820, n = 9746, p-value = 0.7523
alternative hypothesis: nonrandomness
```

Again, we accept the null hypothesis that was that the sequence of the numbers is random.

Bartels Test for Randomness

Another test that you can apply is the Bartels Test for Randomness which is the rank version of von Neumann's Ratio Test for Randomness. Let's run it in r using the `randests` package.

```
runs.test(casino)
```

```
> bartels.rank.test(casino)

        Bartels Ratio Test

data:  casino
statistic = 0.91296, n = 10000, p-value = 0.3613
alternative hypothesis: nonrandomness
```

Again, we can claim that the numbers are random.

Cox Stuart Test

The proposed method is based on the *binomial distribution*. We can easily run this test in R using again the `randests` package.

```
cox.stuart.test(casino)
```

```
> cox.stuart.test(casino)

        Cox Stuart test

data:  casino
statistic = 2485, n = 4851, p-value = 0.09022
alternative hypothesis: non randomness
```

Again, we accept the null hypothesis at 5% level of significance.

Difference Sign Test

Another test that we can run, is the non-parametric “Difference Sign Test”:

```
difference.sign.test(casino)
```

```
> difference.sign.test(casino)

        Difference Sign Test

data:  casino
statistic = 0.54325, n = 9768, p-value = 0.587
alternative hypothesis: nonrandomness
```

Again, we accept the null hypothesis.

Conclusion

We provided some different approaches to how you can test if a sequence of numbers is actually random

or not. You can apply these tests if you want to make ensure the randomness of a sequence of numbers.