EXPLORING TIME SERIES MODELING

Fundamental Statistics

Time Series Modeling With Python Code

Jiahui Wang Apr 5 · 6 min read ★

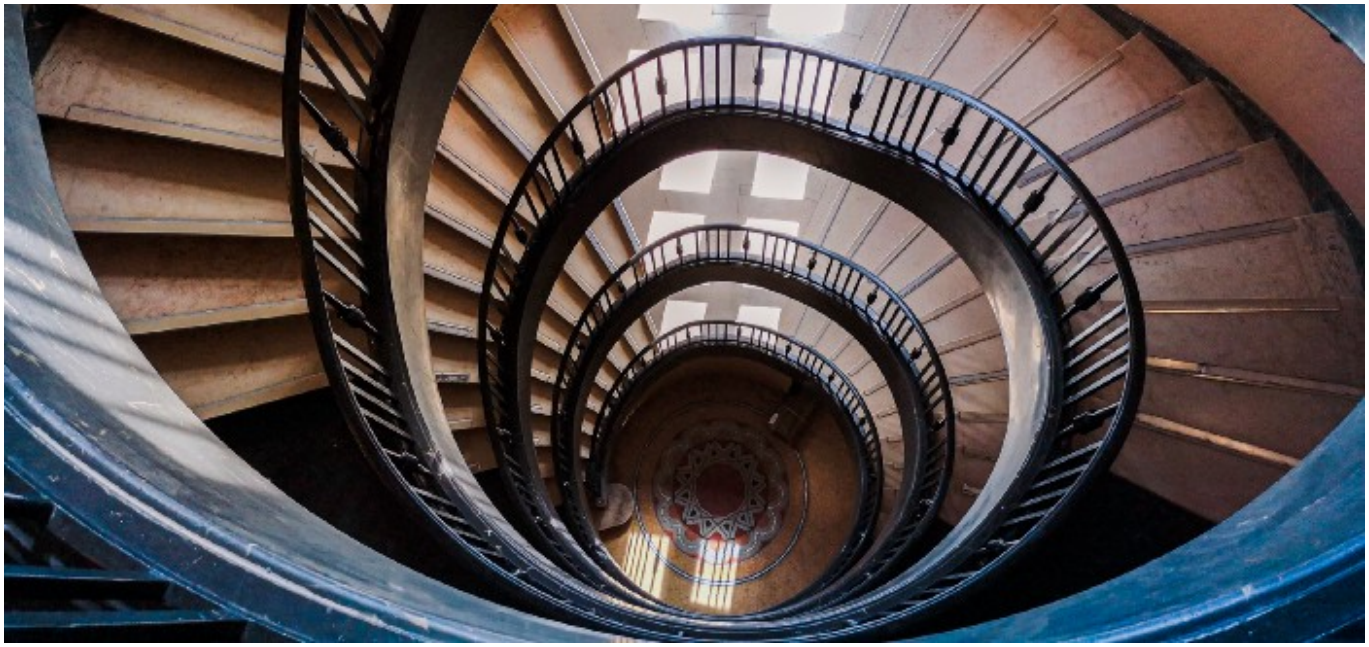


Photo by tangi bertin on Unsplash

Time series data is everywhere around us, ranging from the stock market price to the daily temperature of your city. As its name suggests, the x axis of time series data is time. We are always standing at the current time point. Towards the left-hand side of the x axis, we are looking into the past. If we are lucky, at a simple glimpse, we may be able to find some periodic patterns in the past. Or, if we put in more effort and pull some other variables, we may be able to get some 'correlations' to explain our past data. But, can we evaluate how well these other variables models the data? Will these 'correlations' hold in the future? Can we count on these past 'correlations' and make predictions?

If you also have these questions, let's explore analysing and modeling time series data together! I will code using Python to conduct some experiments and statistical tests. A good thing about learning statistics using Python is that we can use established libraries and plot nice graphs to better understand the complex statistical concepts.

In the future, I also plan to have the following posts:

Time Series Modeling With Python Code: How To Analyse A Single Time Series Variable.

Time Series Modeling With Python Code: How To Analyse Multiple Time Series Variable.

Time Series Modeling With Python Code: How To Model Time Series Data With Linear Regression.

Since I will start from the very basic statistical concepts to more complex analysis and modeling, if you are a beginner to time series data, I would strongly recommend you start from this post.

1. Population, Sample, and Estimators

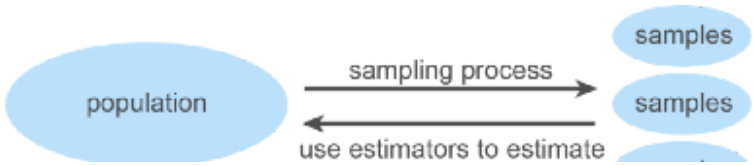
Population has an underlying distribution process, which we usually are not able to know exactly. What we can do is to sample from the population and use the samples to estimate the population. But how to choose the proper estimators? There are three properties to define a good estimator: unbiased, consistent, and efficient.

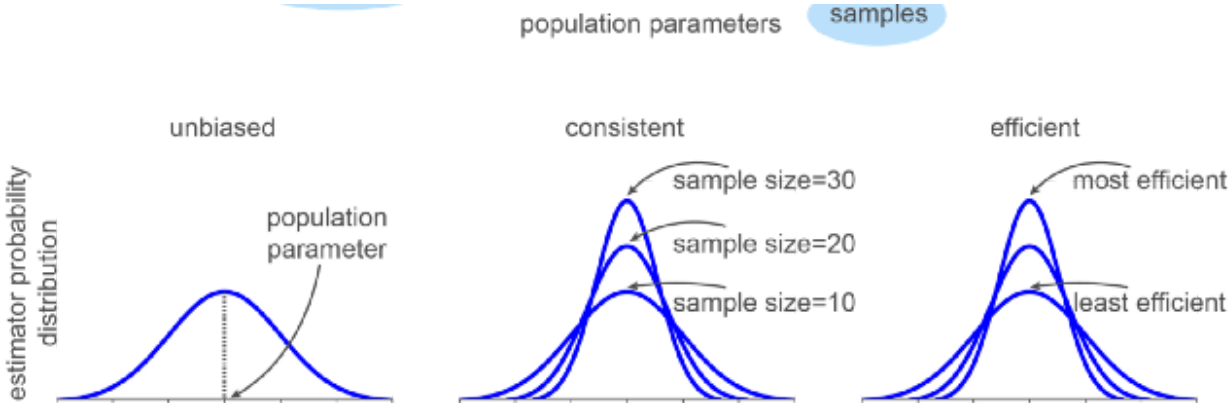
The estimator is unbiased, when the expected value of the sample parameter is equal to the population parameter:

$$E[parameter_{sample}] = parameter_{population}$$

If the variance of the sample parameter decreases with the increasing sample size, the estimator is consistent.

With the same sample size, the estimator with lower variance is more efficient.

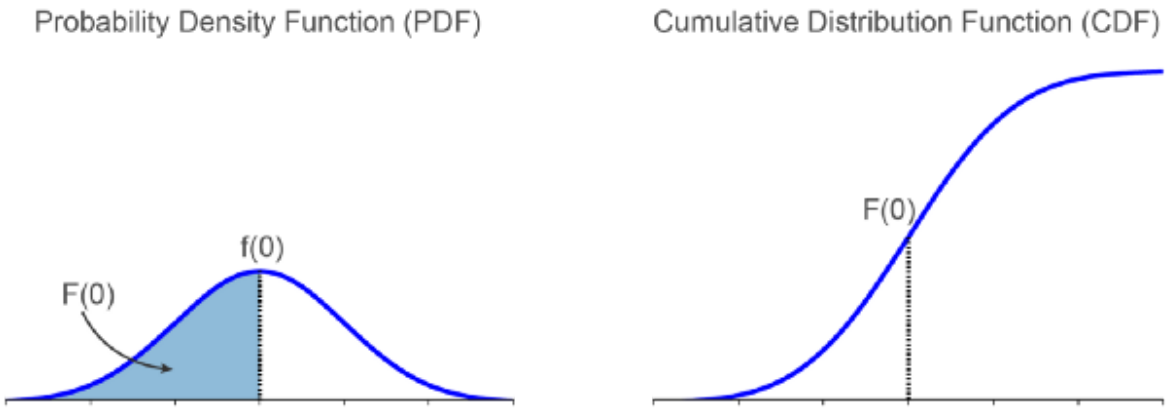




A good estimator is unbiased, consistent, and efficient.

2. Probability Density Distribution (PDF)

The probability density distribution (PDF) is used to specify the probability of the random variable falling within a particular range of values. The probability density at a certain x is denoted as f(x). By applying integral function to f(x) over a range of (x1,x2), the probability of x falling in (x1,x2) can be calculated.



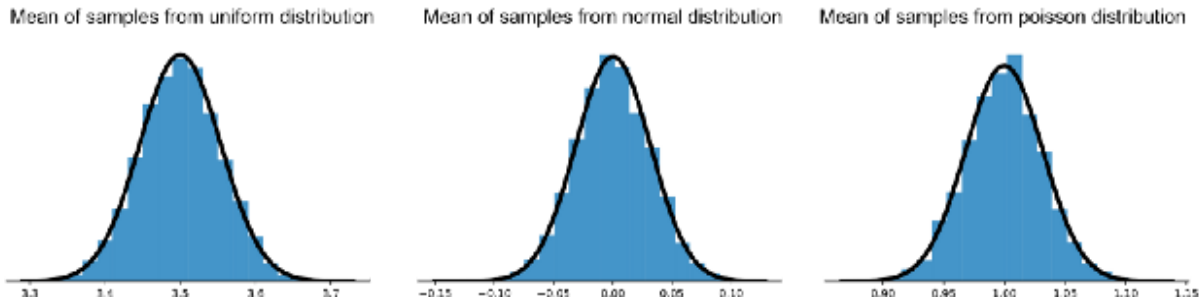
PDF and CDF of normal distribution

3. Central Limit Theorem and Law of Large Numbers

Central Limit Theorem states that when the sample size is large, the sample mean of the independent random variable follows normal distribution. Typically, when sample size is larger than 30, the requirement of large sample size is considered fulfilled. The independent random variables can follow any distribution, while the sample mean of these independent random variables follows normal distribution.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm


meanList = []


num_trials = 10000
num_observations = 1000


for i in range(num_trials):
    # sample from uniform distribution
    numList = np.random.randint(1,7,num_observations)
    # sample from normal distribution
    #numList = np.random.normal(loc=0,scale=1,size=num_observations)
```

```
    # sample from poisson distribution
    #numList = np.random.poisson(lam=1,size=num_observations)
    meanList.append(np.mean(numList))


mu, std = norm.fit(meanList)
fig, ax = plt.subplots()
ax.hist(meanList, bins=20, density=True, alpha=1, color='#4495c9')


xmin, xmax = ax.get_xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
ax.plot(x, p, 'k', linewidth=4)


ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['left'].set_visible(False)
ax.set_yticks([])
plt.show()
```

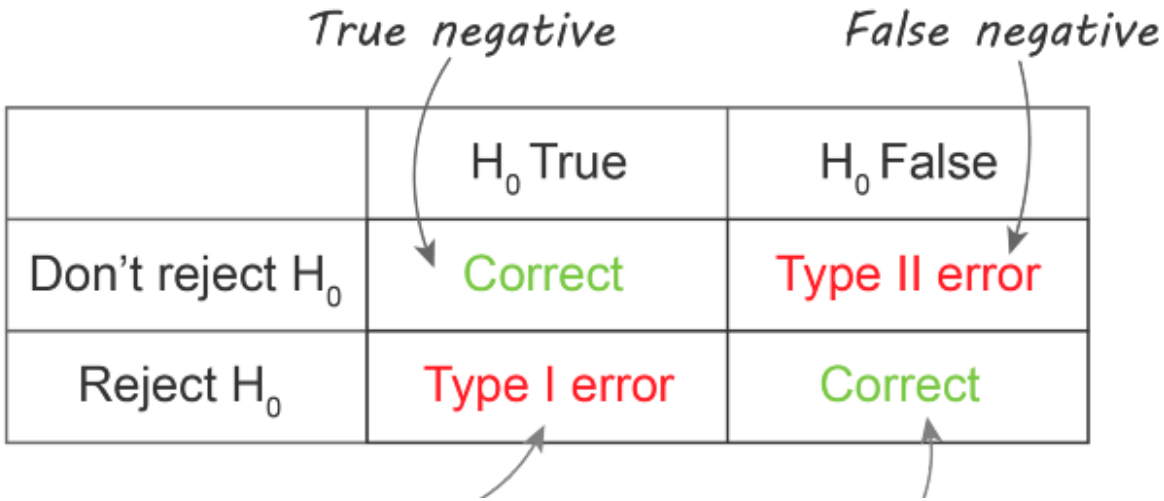


Law of large numbers states that given a large number of trials, the average of the estimator gets closer to the theoretical value. For the above experiment, if we only repeat the trials for 10 times, the distribution will be

very different from the plots. If you are interested, you can quickly test out the results to visualize how law of large numbers makes a difference.

4. Hypothesis Testing

Given that only sample parameters can be calculated, we need to make inference on the population parameters using hypothesis testing. In hypothesis testing, a set of complementary hypotheses is proposed, which consists of a null hypothesis and an alternative hypothesis. When conducting the hypothesis testing, we choose to believe that the null hypothesis holds true. If the observed value is likely to occur under the condition that the null hypothesis is true, then we do not reject the null hypothesis. However, if the observed value is unlikely to occur, then we reject the null hypothesis and accept the alternative hypothesis.



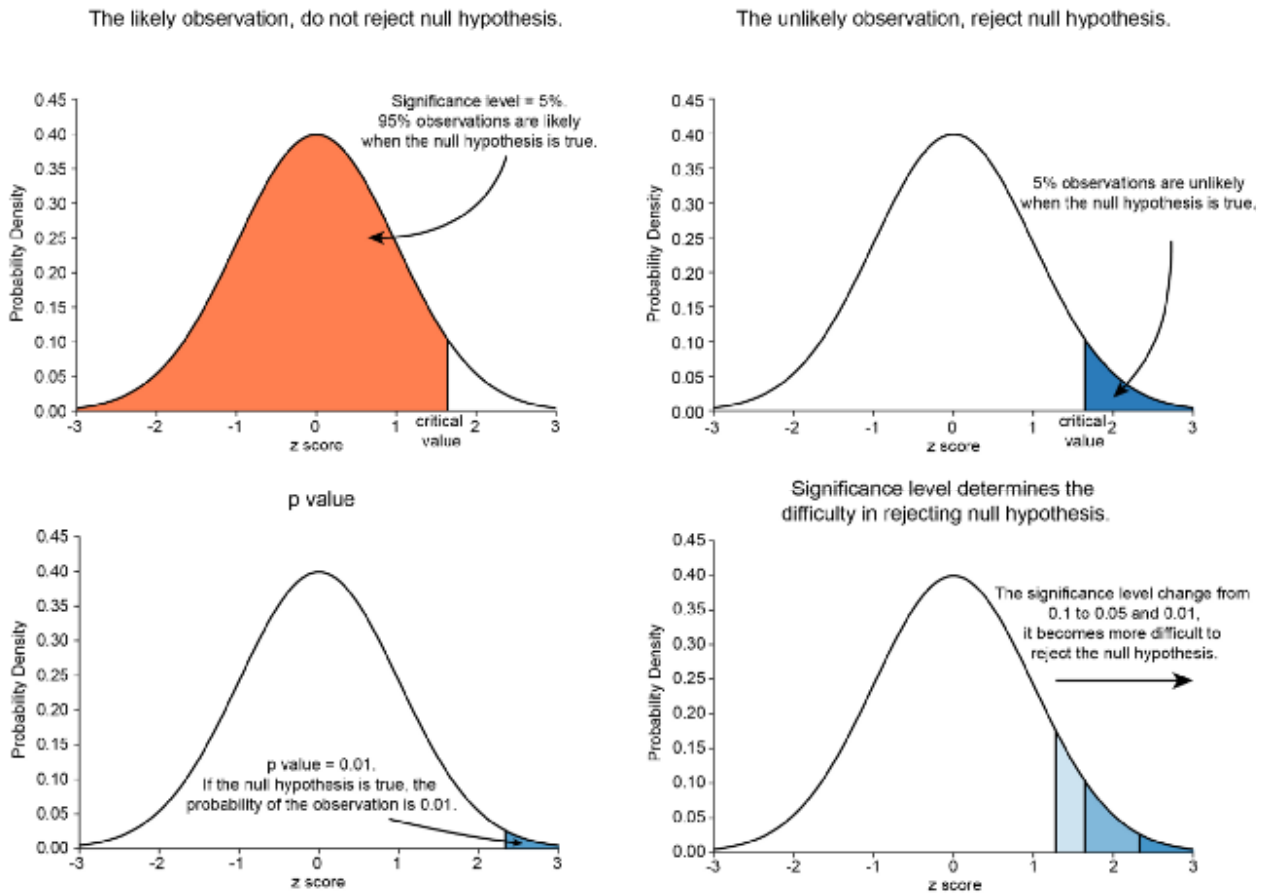
Hypothesis matrix

5. Significance Level and P Value

Before conducting hypothesis testing, we need to define a significance level first. Significance level determines the level that we want to believe in the null hypothesis. If we set the significance level as 0.05, then as long as the probability of the observation is higher than 5%, we do not reject the null hypothesis. However, if the probability of the observation falls below 5%, we reject the null hypothesis and accept the alternative hypothesis. There is a tradeoff between the Type I and Type II error. Basically, a higher significance level makes it easier to reject the null hypothesis. Although in this way, a higher significance level reduces the Type II error, it also results in a higher Type I error at the same time. The only way to reduce both Type I and Type II error is by increasing the sample size.

The probability of the observed value is called p value. A low p value means that the observation is unlikely to occur under the condition that the null hypothesis holds true. When the p value is lower than significance level, then we reject the null hypothesis. However, one thing to note is that p

value should be interpreted as binary: it is only larger or smaller than the significance level.



How to interpret significance level and p value

Summary

In this post, I highlighted some fundamental statistical concepts which are important to understand the future posts on analysing and modeling time series data. Hope you are confident to make a further step on our journal to explore time series data! Stay tuned for the following post on how to analyse a single time series variable!

Time Series Modeling Data Science Machine Learning Programming Python