ITESO, Universidad Jesuita de Guadalajara

Simulación montecarlo aplicado a inversiones

Pablo Alejandro Rivera Sánchez

Claudio Rodriguez Orozco

9 de noviembre de 2019

Simulación matemática

Carlos Arellano Muro

Objetivos

El objetivo principal de este proyecto es encontrar la manera de simular una inversión a largo plazo de manera más realista, utilizando un interés variable o aleatorio (controlado). Dicho objetivo se logrará utilizando los temas vistos en clases anteriores, así como las tareas que hemos realizado a lo largo del periodo, de igual manera, este proyecto también tiene como objetivo aplicar todos los temas que hemos visto en clase a una situación real propuesta por nosotros, además de poder practicar los mismos temas de clase. Se analizará las formas tradicionales en las que se calcula una inversión y posteriormente se explicará cómo llegamos a nuestra propuesta/solución.

Problema

Empezando con $10,000 e invirtiendo otros $10,000 adicionales cada año, ¿cuál es la probabilidad de que tengas al menos $1,000,000 después de 30 años de invertir en el S&P 500?.

Usualmente cuando queremos calcular o resolver este problema, asumimos que obtendremos un rendimiento anual, si no es que fijo, muy cercano a lo que esperamos, es decir, no esperamos que este varíe mucho año con año. Sin embargo, este no es el caso, el rendimiento en una inversión de este tipo no es fija y puede variar en grandes cantidades, ya sea subir o bajar. Esta forma tradicional arrojaría un resultado igual al siguiente:

```
# Manera tradicional (con un interés fijo).

cap=10000 # Inversión o capital inicial.
años=30 # Número de años.
i=.07 # Interés.
acap=10000 # Capital aportado cada año.

cy=[]
tabla=pd.DataFrame(index=list(range(años)),columns=['Cantidad'])
for a in range(años):
    cf=cap*(1+i)+acap
    tabla.loc[a,'Cantidad']=locale.currency(cf,grouping=True)
    cap=cf

tabla
```

	Cantidad
0	$20,700.00
1	$32,149.00
2	$44,399.43
3	$57,507.39
4	$71,532.91
5	$86,540.21
6	$102,598.03
7	$119,779.89
8	$138,164.48
9	$157,835.99
10	$178,884.51
11	$201,406.43
12	$225,504.88
13	$251,290.22
14	$278,880.54
15	$308,402.17
16	$339,990.33
17	$373,789.65
18	$409,954.92
19	$448,651.77
20	$490,057.39
21	$534,361.41
22	$581,766.71
23	$632,490.38
24	$686,764.70
25	$744,838.23
26	$806,976.91
27	$873,465.29
28	$944,607.86
29	$1,020,730.41

En dónde empezamos con la cantidad de $10,000, invertimos es misma cantidad por 30 años con un interés del 7%. Cada año realizamos la siguiente operación:

Capital x (1 + interés) + Capital aportado

En este caso si cumplimos con tener $1,000,000 al final de los 30 años, pero como ya se mencionó anteriormente, esto no es muy realista debido a que el interés puede variar.

Solución

El problema a resolver, en pocas palabras, era encontrar la manera de hacer que el interés de la inversión cambiara cada año. Esto lo pudimos resolver gracias a una función random (np.random.normal), en la cual introducimos el interés para que nos devolviera un valor aleatorio del mismo. Además, creamos una nueva variable, volatilidad, la cual también introducimos a la función random, de manera que la función nos devolvía un valor aleatorio de interés cada vez que la corríamos. Todo lo anterior lo acomodamos en un for (con n=30) para generar un posible futuro usando la misma fórmula interés:

Capital x (1 + interés) + Capital aportado

Lo único que cambia que el interés está dado por la función random, es decir, cada año (cada vez que el for se repite) el interés es diferente ya que se genera aleatoriamente. Esto lo podemos visualizar mejor en la siguiente tabla, dónde “ir” es el interés obtenido ese año y “vf” es la cantidad de dinero al finalizar el año.

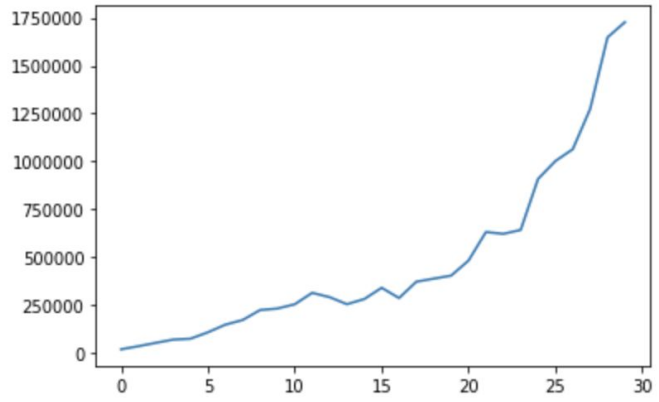
	ir	vf
0	-0.240452	$17,595.48
1	-0.0202169	$27,239.75
2	0.168225	$41,822.16
3	0.218379	$60,955.26
4	-0.0607709	$67,250.95
5	0.0060418	$77,657.26
6	0.0116107	$88,558.92
7	0.179883	$114,489.12
8	0.296532	$158,438.83
9	-0.00319575	$167,932.50
10	0.171872	$206,795.44
11	0.0228842	$221,527.80
12	-0.196477	$188,002.63
13	0.033863	$204,368.97
14	0.184878	$252,152.23
15	0.107553	$289,272.01
16	0.049914	$313,710.72
17	0.192375	$384,060.88
18	0.205743	$473,078.77
19	0.155372	$556,582.11
20	0.125011	$636,161.25
21	-0.132809	$561,673.32
22	-0.02895	$555,412.87
23	0.348108	$758,756.33
24	0.135368	$871,468.01
25	0.159136	$1,020,150.08
26	-0.011275	$1,018,647.92
27	0.100634	$1,131,158.26
28	-0.227852	$883,421.27
29	0.224175	$1,091,461.82

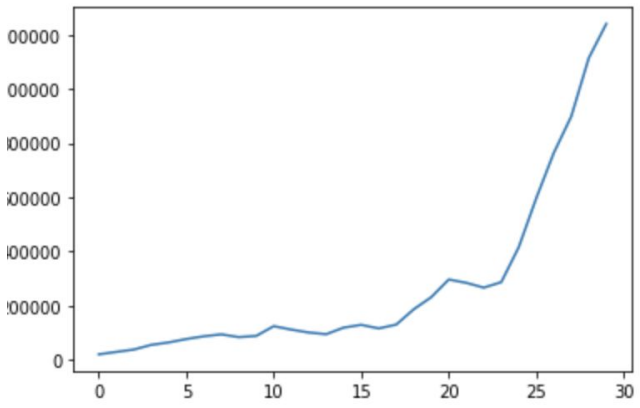
```
capii=10000 # Inversión inicial.
ie=.09 # Interés esperado.
v=.18 # Volatilidad, variación del 9%.
años=30 # Años de la inversión.
acap=10000 # Aporte al capital.
```

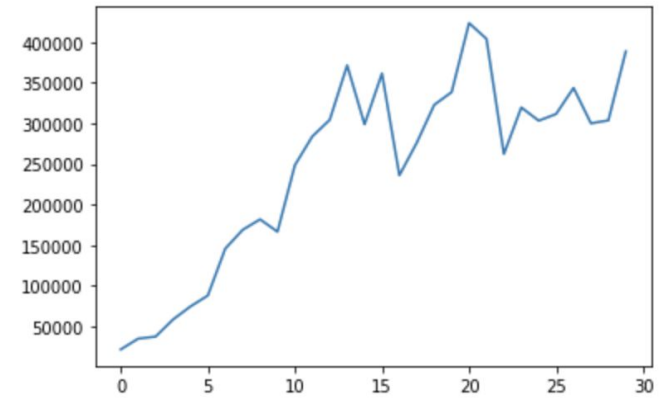
```
def inv(capii,ie,v,años,acap):
    tabla=pd.DataFrame(index=list(range(años)),columns=['ir','vf'])    # 1,años+1
    for a in range(años):
        ir=np.random.normal(ie,v) # Rendimiento aleatorio por año, basado en el interés esperado y volatilidad.
        vf=capii*(1+ir)+acap # Valor final.
        tabla.loc[a,'ir'],tabla.loc[a,'vf']=ir,locale.currency(vf,grouping=True)
        capii=vf
    return tabla
```

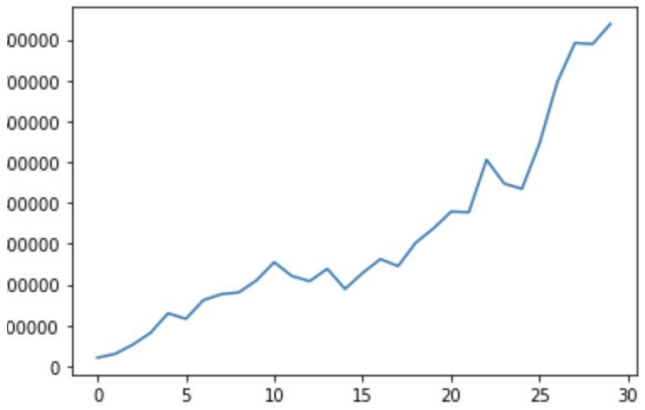
En este caso, también logramos nuestro objetivo de tener más de $1,000,000 después de 30 años, hasta logramos llegar a dicha cantidad en menos años (25). A pesar de que también se cumplió el objetivo, habrá simulaciones u futuros en los que no se cumpla o que se sobrepase por mucho como en las siguientes.

(Eje x = años; Eje y = cantidad de dinero)



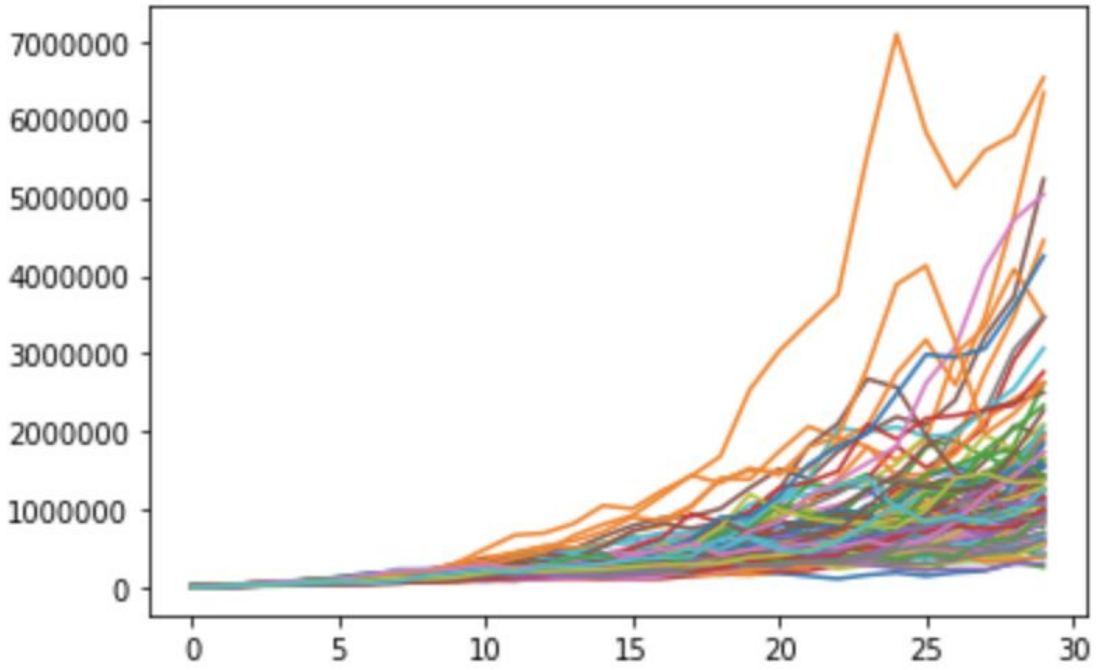






Todo este proceso lo podemos formular o plantear como una simulación montecarlo, en la que simulará cierta cantidad de veces la simulación de una inversión a 30 años, ya tomando en cuenta el interés aleatorio para así poder obtener datos estadísticos como el promedio, entre otras cosas. A continuación se muestra una gráfica de 100 escenarios diferentes, usando el método montecarlo.

```
N=100 # Número de inversiones.
for i in range(N):
    plt.plot(invv(10000,.09,.18,30,10000))
```



Al repetir el mismo proceso, pero en esta ocasión simular 10,000 escenarios, obtenemos una media regular que siempre se acerca a $1,500,000 en todos los casos. Sin embargo, este puede variar de acuerdo al número de escenarios seleccionados.

```
def invvv(capii,ie,v,años,acap):
    for a in range(años):
        ir=np.random.normal(ie,v)
        vf=capii*(1+ir)+acap
        capii=vf
    return round(capii,4)
```

```
# Simulación montecarlo.
# Simulación de varios (10000) capitales finales para poder analizar.

N=10000
x=[]

for w in range(N):
    x.append(invvv(10000,.09,.18,30,10000))
x
```

Resultados (promedio):

1525533.5879913901 1492152.2171211198

1506179.9199585698 1498124.0420894802

Conclusiones

Al realizar los proyectos y los trabajos hechos en clase, pudimos observar que la simulación montecarlo tiene un gran poder ya que puede simular una gran cantidad de procesos aleatorios, haciéndolos de cierta manera no tan aleatorios, ya que podemos obtener varios datos de las mismas simulaciones que nos ayudarán a reducir esta aleatoriedad. Anteriormente no conocíamos absolutamente nada acerca de este tema y al principio se veía un poco extraño, sin embargo, ya después de entenderlo y ponerlo en práctica nos dimos cuenta que es una gran herramienta de análisis para cualquier persona y puede ser muy útil casi para cualquier carrera debido a que puedes simular casi cualquier proceso. Además del concepto de simulación montecarlo, ya tenemos una idea bastante buena de como hacer dicho proceso con herramientas de programación.

Referencias

- Alpha Bench. (2017). Monte Carlo Simulation with Python. 08/11/19, de Alpha Bench Sitio web: https://alphabench.com/data/monte-carlo-simulation-python.html