190Class9.1_MonteCarloSimulation

March 10, 2023

Monte Carlo Simulation: a model used to predict the probability of a variety of outcomes when the potential for random variables(risk) is present. Monte Carlo Simulation is one method for representing a complex financial system in which a number of random variables interact Repeating the event again and again (think Groundhog Day, same day multiple times) The simulation provides an empirical probability distribution of possible outcomes. Only provides statistical estimates, not exact results. We can ask questions such as: What's the probability of profit being above x%? Historical data might not be representative of future(like all empirical statistics)

Finance Analysts will use a model to find data, then use other models to try and check the model's results. A stock split will theoretically keep a company's total evaluation the same, just a stock worth $100 split into 2 stocks would be 2 $50 stocks. However, since people might see the stock as mroe affordable the stock price might increase temporarily due to investor perception. Insider trading makes stocks more volatile, higher increases, sharper drops.

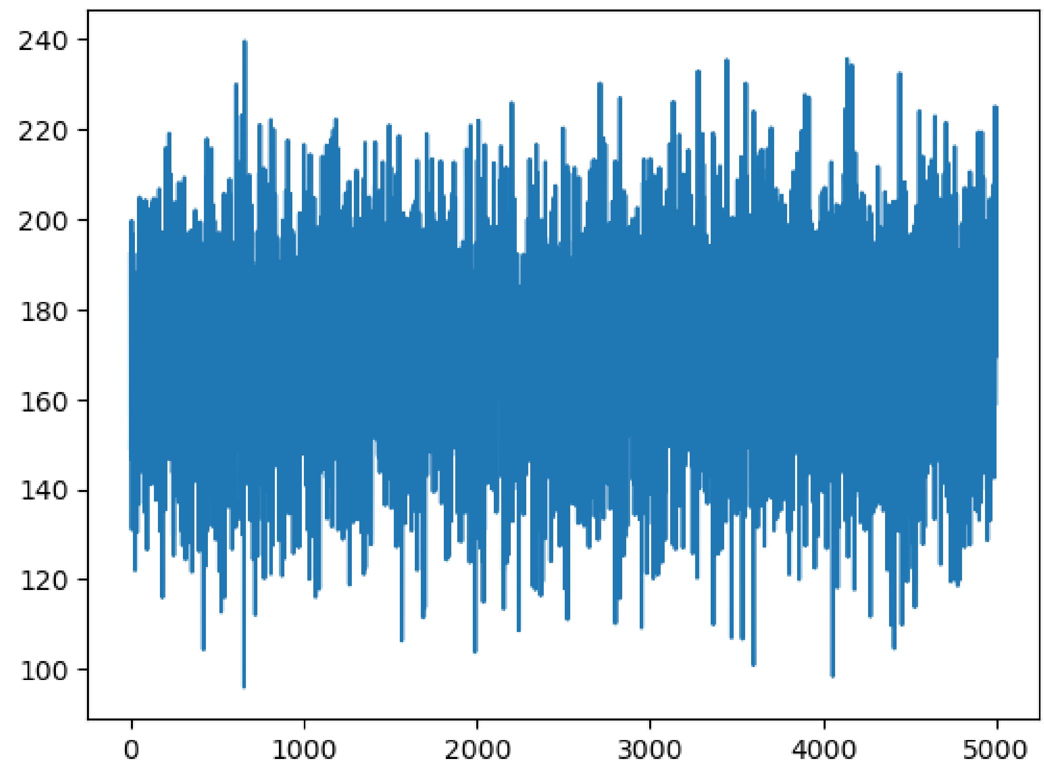
[1]:
```
#Monte Carlo Simulation
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sst
```

[2]:
```
#Make assumptions about revenue and _ procedure.
#Simulation outputs are only as good as assumptions (mean & std)
rev_m=170
rev_std=20
iterations=5000
```

[3]:
```
#Drawing values from a random distribution
#Assuming revenue has a normal distribution
rev=np.random.normal(rev_m,rev_std,iterations)
rev
```

[3]:
```
array([192.29550371, 148.71011735, 153.35142831, …, 200.31733916,
       172.84957   , 224.64904963])
```

[4]:
```
#Plot revenue
plt.plot(rev)
plt.show()
```



[5]:
```
sst.describe(rev)
```

[5]:
```
DescribeResult(nobs=5000, minmax=(96.05479545318671, 239.52910304228269),
mean=170.10358793637772, variance=420.57418602393864,
skewness=-0.07486957212143266, kurtosis=0.020748890866650078)
```

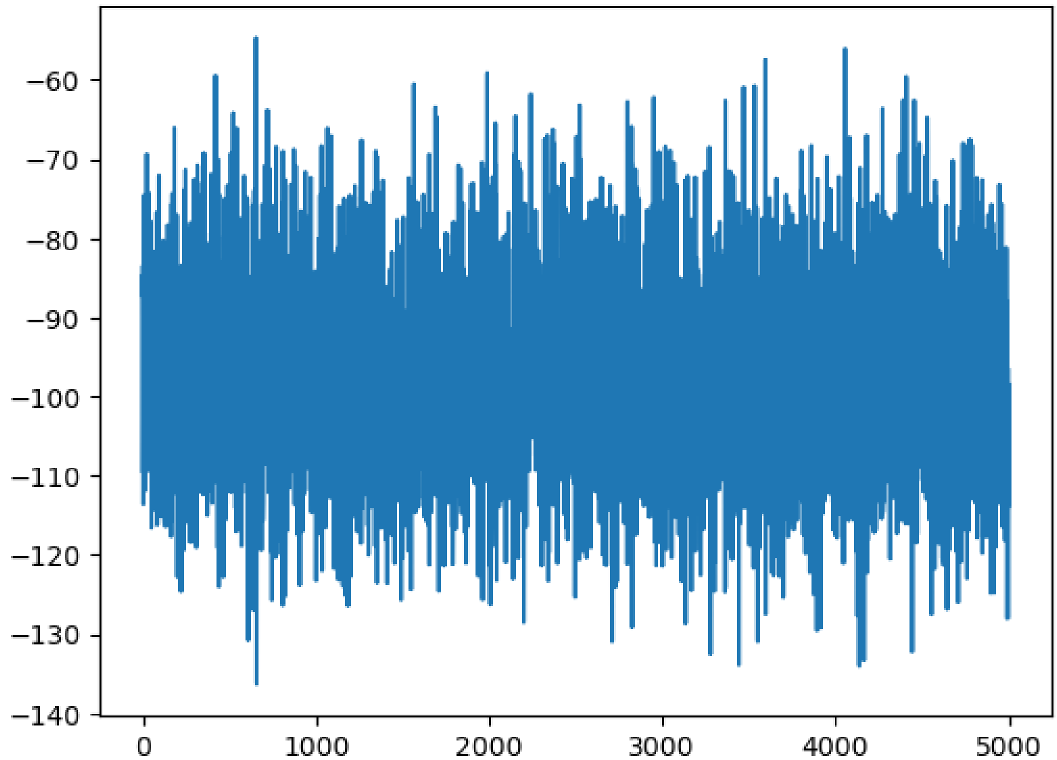
[14]:
```
#We have 2 random variables, REV & 60%.
#We interact the two random variables and see all possible outcomes
#Make COGS negative since it is a negative cash flow
COGS=-(rev*np.random.normal(0.6,0.1))
COGS
```

[14]:
```
array([-109.42584198,  -84.62355847,  -87.26469854, …, -113.99067102,
        -98.36012475, -127.83664169])
```

[15]:
```
sst.describe(COGS)
```

[15]:
```
DescribeResult(nobs=5000, minmax=(-136.30414270869665, -54.66002410986632),
mean=-96.79752243404215, variance=136.1896638297048,
skewness=0.07486957212142868, kurtosis=0.020748890866650078)
```

[16]:
```
plt.plot(COGS)
plt.show()
```



[17]:
```
Gross_Profit=rev+COGS
Gross_Profit
```

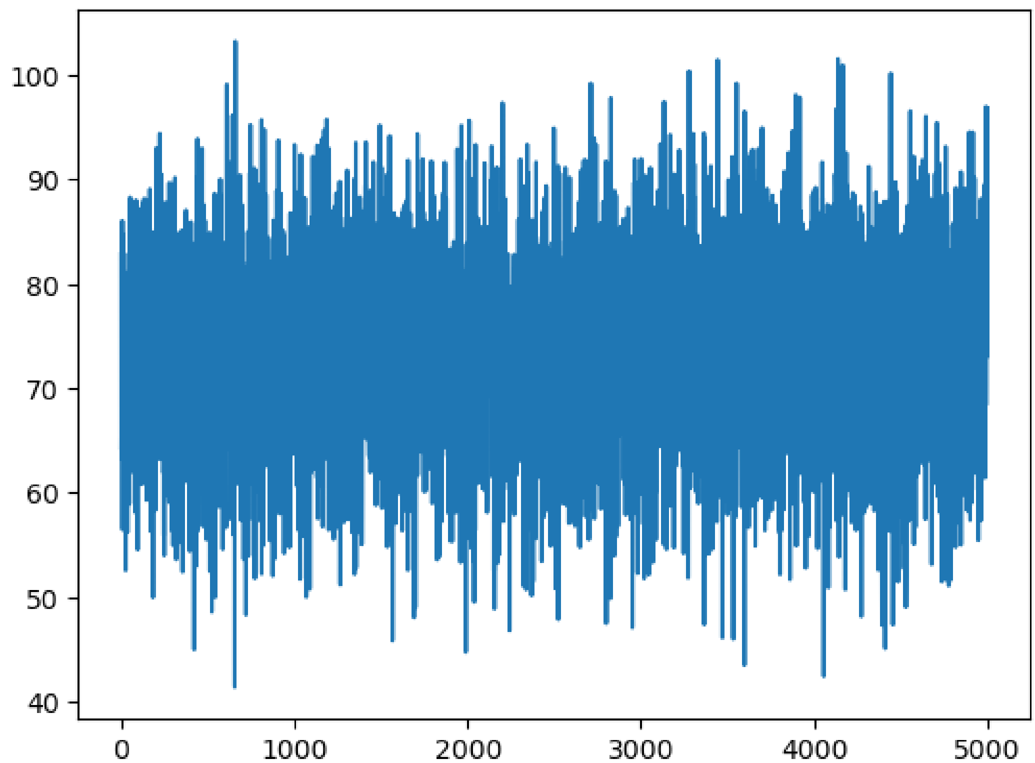
[17]:
```
array([82.86966173, 64.08655888, 66.08672977, …, 86.32666814,
       74.48944525, 96.81240794])
```

[18]:
```
sst.describe(Gross_Profit)
```

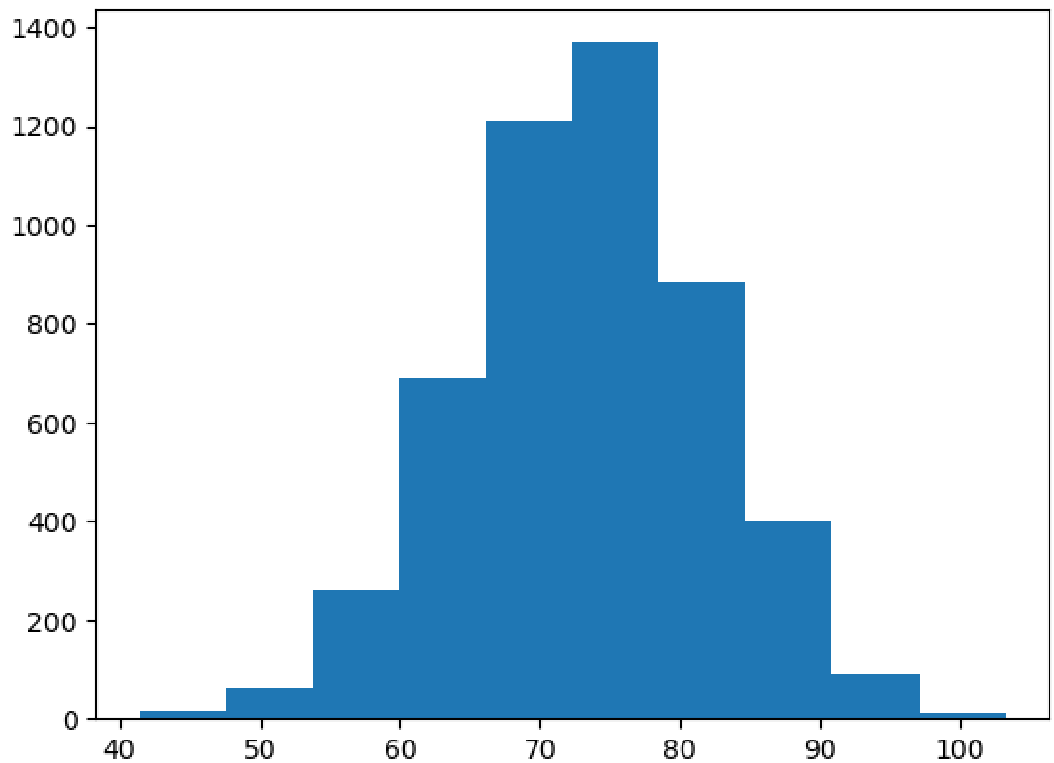
[18]:
```
DescribeResult(nobs=5000, minmax=(41.39477134332039, 103.22496033358604),
mean=73.30606550233554, variance=78.10799431627748,
skewness=-0.07486957212142831, kurtosis=0.020748890866499003)
```

[19]:
```
plt.plot(Gross_Profit)
```

[19]:
```
[<matplotlib.lines.Line2D at 0x7fcf11dcd5e0>]
```



[20]:
```
plt.hist(Gross_Profit)
plt.show()
```



[23]:
```
#What is the probability of getting a gross profit < 60?
hist=np.histogram(Gross_Profit)
hist_dist=sst.rv_histogram(hist)
```

[23]:
```
<scipy.stats._continuous_distns.rv_histogram at 0x7fcf10fae670>
```

[24]:
```
#cdf gives probability of outcome
#This probability can vary among tests, increase iterations
#for more trials and a more accurate percentage
hist_dist.cdf(60)
```

[24]:
```
0.06905371287899637
```

[26]:
```
#What is the probability of gross profit more than 90?
1-hist_dist.cdf(90)
```

[26]:
```
0.03239665934950531
```

[27]:
```
#What is the probability of gross profit from 60 < x < 90?
hist_dist.cdf(90)-hist_dist.cdf(60)
```

[27]:
```
0.8985496277714984
```