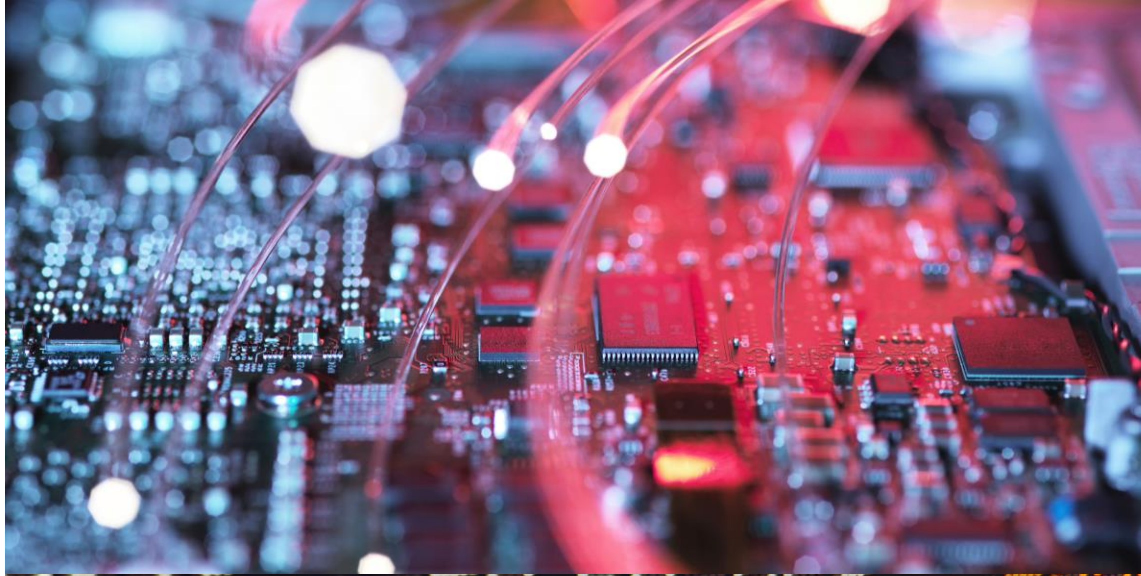
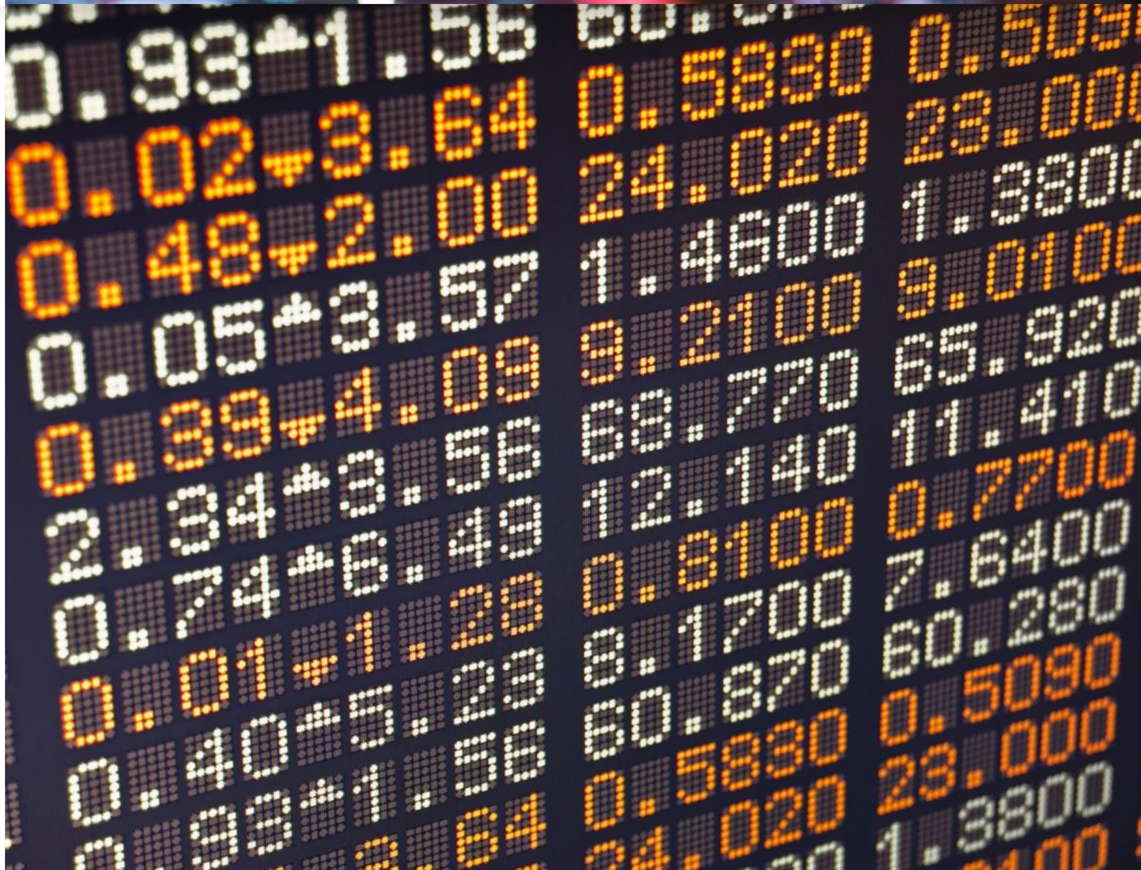
Monte Carlo Simulation

NVDA & ASML Stock Price – The backbone of GPUs

Market Research?





- The global GPU market is expected to grow at a compound annual growth rate (CAGR) of 33.5% from 2022 to 2030. This growth is due to rising demand for scientific simulations, data analytics, and artificial intelligence.
- The GPU market was worth more than $40 billion in 2022 and is expected to reach $40,093.24 million by 2028.
- The rapid penetration of big data technology is expected to complement the industry expansion.
- The GPU shortage has been caused by cryptocurrency mining, which reached critical levels in 2020. Disruptions to manufacturing, such as the coronavirus disrupting the manufacturing process, also contributed to the shortage.
- Nvidia is seeing a huge surge in demand for its chips due to the explosion in artificial intelligence (AI) tools like ChatGPT. AMD is aiming to catch up to Nvidia in the AI GPU market.

Why Monte Carlo?

- Monte Carlo simulations are like playing out every possible scenario in a game of chance to see what could happen. Imagine you're trying to predict the weather. You can't say for sure if it will rain tomorrow, but you can guess the likelihood based on various factors like humidity, wind speed, and so on. Named after a famous casino, this method helps us deal with the unknown and make educated guesses about future events.

Proposed Research Questions

- I. What is the expected stock price trajectory for NVDA and ASML over a specified time horizon?
- II. How do the risk profiles of NVDA and ASML compare, as measured by metrics such as Value at Risk (VaR) and Conditional Value at Risk (CVaR)?
- III. Can Monte Carlo simulations provide actionable insights for portfolio diversification involving NVDA and ASML stocks?

Mathematical Framework & Motivation

i. Simulate Sample Paths:

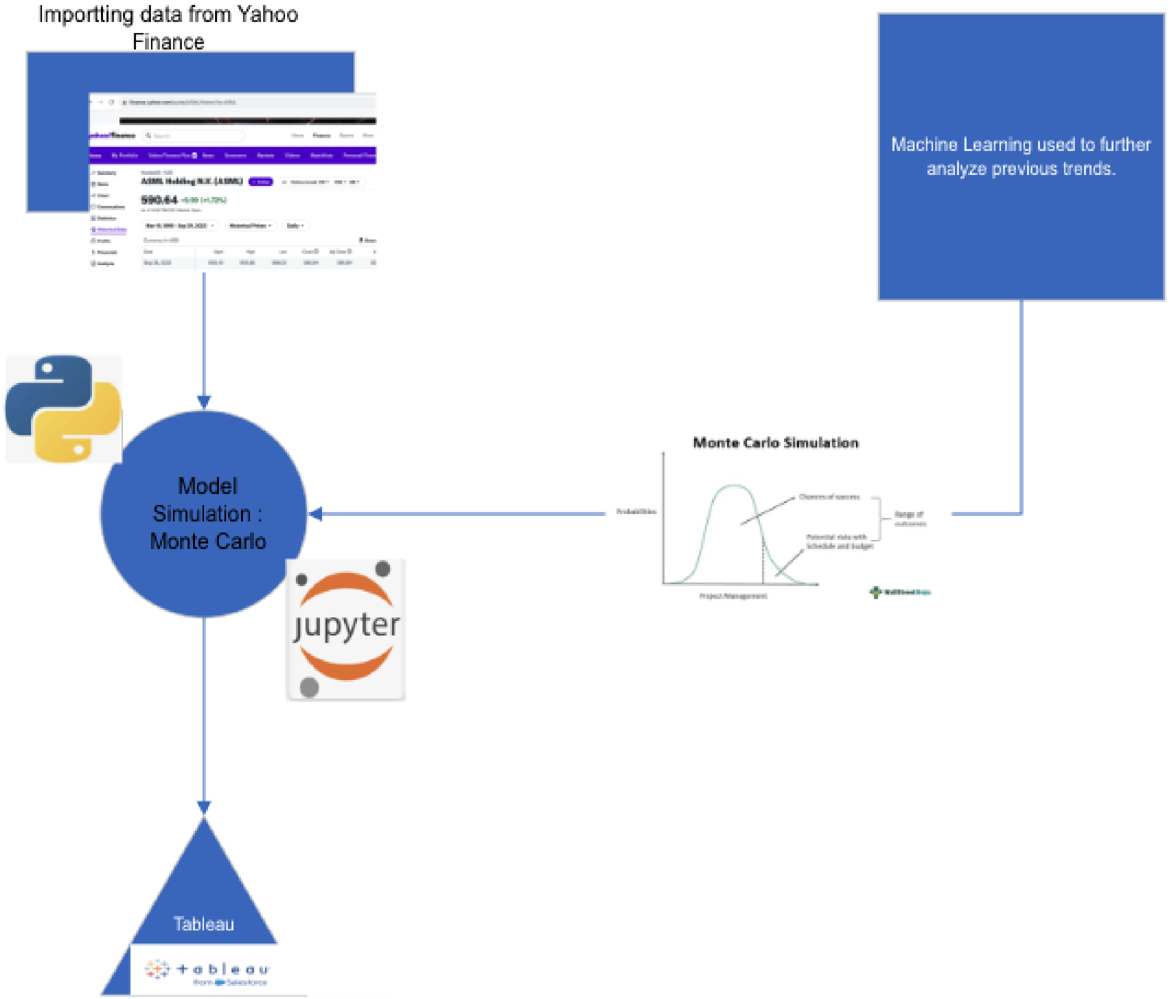
- Imagine you're watching the prices of certain items (like houses or stocks) and how they change over time. We create a pretend or "simulated" version of this scenario to see how these prices might change in the future, based on certain assumptions.

ii. Evaluate Discounted Cash Flows:

- Now, think about the money you could make or spend on these items in the future. The value of money changes over time due to things like inflation or interest rates. So, we adjust the future money amounts to reflect what they are worth today. We do this for each simulated scenario from step 1.

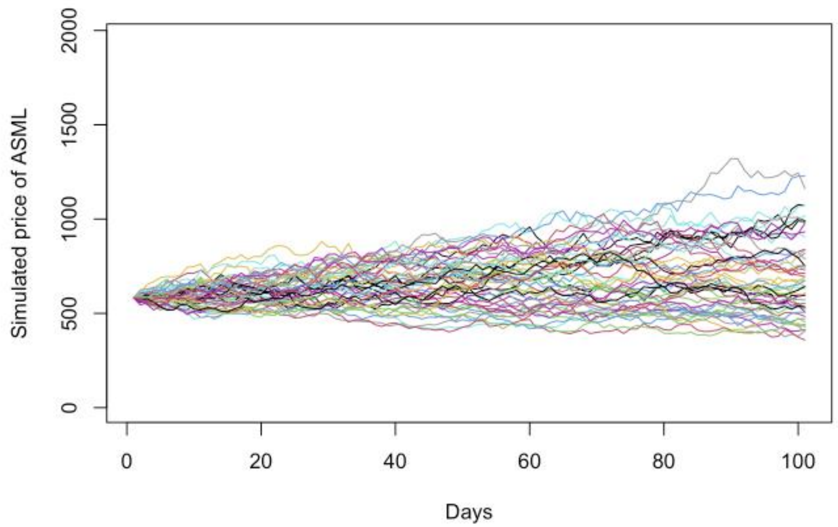
iii. Take the Sample Average:

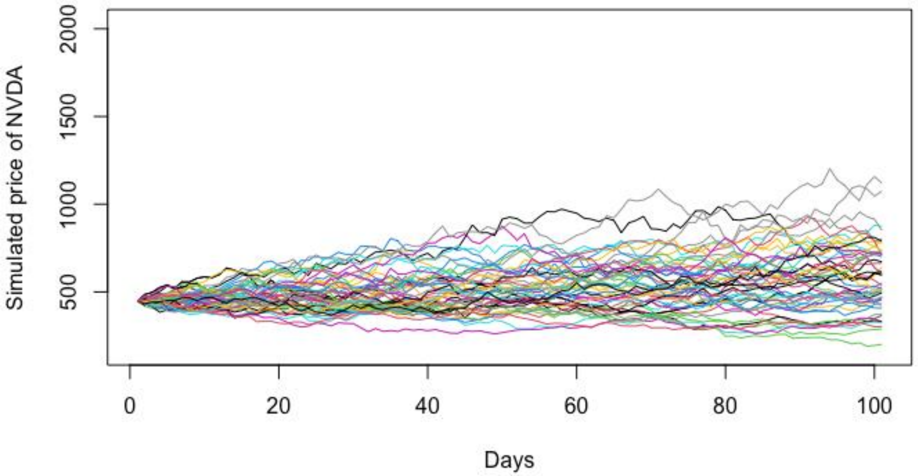
- Finally, we look at all the adjusted money amounts from all the simulated scenarios and find the average. This gives us an idea of what the money related to these items could be worth on average, in today's terms, across many possible future scenarios.



Data Architecture

Results – Monte Carlo Simulation





- Parameters Used :
  - Simulations: 4000 to 9000
  - Number of days: 100 days.

- Starting Price: Last Day Close Price
- Mean,S.D. : Daily returns Max used

R code - ASML

ASML stock price

```
# Get stock data for NVDA
getSymbols("ASML", from = "1995-03-14", to = "2023-10-03", auto.assign = TRUE, warnings= FALSE)
```

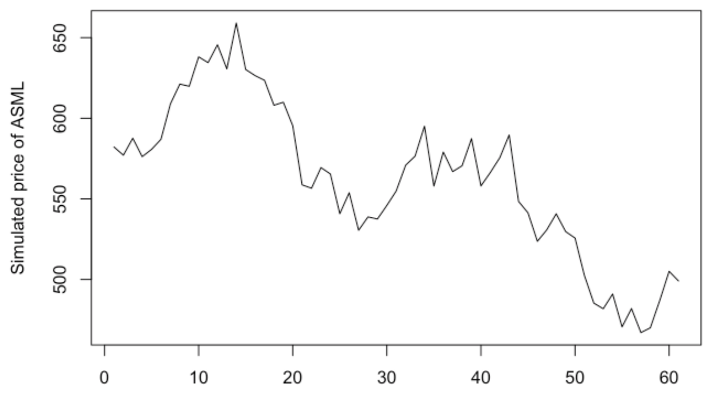
```
## [1] "ASML"
```

```
daily_mean1 <- mean(dailyReturn(ASML)) #mean
daily_std_dev1 <- sd(dailyReturn(ASML)) #standard deviation
```

```
no_of_days1 <- 60 # Set variable to 60 days
starting_price1 <- last(ASML$ASML.Close)[[1]] #Closing price from 434.95 - last item in our frame
```

```
set.seed(101) #Set seed for reproducibility of the random numbers
returns <- 1+rnorm(no_of_days1, mean=daily_mean1, sd=daily_std_dev1) #Generate random variables
prices <- cumprod(c(starting_price1, returns)) #Calculate cumulative product

plot(prices, type='l', ylab="Simulated price of ASML", xlab="Days")
```



Monte Carlo 100 Simulations - ASML

```
# Existing setup code
no_of_sims1 <- 9000 # times to run the loop used in the for loop
no_of_days1 <- 100  # assuming you have this variable somewhere
starting_price1 <- last(ASML$ASML.Close)[[1]] #Closing price
daily_mean1 <- mean(dailyReturn(ASML)) #mean
daily_std_dev1 <- sd(dailyReturn(ASML)) #standard deviation

returns_list <- matrix(0, nrow = no_of_sims1, ncol = no_of_days1) #define matrices
prices_list <- matrix(0, nrow = no_of_sims1, ncol = no_of_days1+1)

for(i in 1:no_of_sims) {
  returns_list[i,] <- rnorm(no_of_days1, mean=daily_mean1, sd=daily_std_dev1)
  prices_list[i,] <- cumprod(c(starting_price1, 1+returns_list[i,]))
}

# Calculate dynamic y-limits
y_min <- min(prices_list)
y_max <- max(prices_list)

# Plotting
plot(prices_list[1,], type='l', ylab="Simulated price of ASML", xlab="Days", ylim=c(y_min, y_max))
for(i in 1:50) {
  lines(prices_list[i, ], type = 'l', col=i)
}
```

R Code - NVDA

NVDA stock price

```r
# Get stock data for NVDA
getSymbols("NVDA", from = "1999-01-22", to = "2023-10-03", auto.assign = TRUE, warnings= FALSE)
```

```
## [1] "NVDA"
```

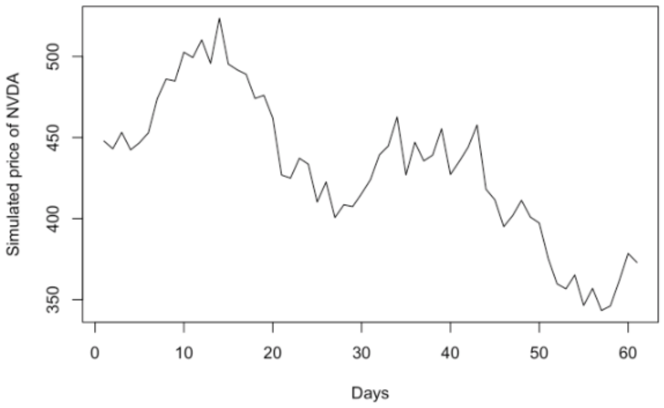
```r
#getSymbols("SPY", from = '2011-09-08', to = "2021-09-08", auto.assign = TRUE, warnings = FALSE)
```

```r
daily_mean <- mean(dailyReturn(NVDA)) #mean
daily_std_dev <- sd(dailyReturn(NVDA)) #standard deviation
```

```r
no_of_days <- 60 # Set variable to 60 days
starting_price <- last(NVDA$NVDA.Close)[[1]] #Closing price from 434.95 - last item in our frame
```

```r
set.seed(101) #Set seed for reproducibility of the random numbers
returns <- 1+rnorm(no_of_days, mean=daily_mean, sd=daily_std_dev) #Generate random variables
prices <- cumprod(c(starting_price, returns)) #Calculate cumulative product

plot(prices, type='l', ylab="Simulated price of NVDA", xlab="Days")
```



Monte Carlo 100 Simulations - NVDA

```r
# Existing setup code
no_of_sims <- 4000 # times to run the loop used in the for loop
no_of_days <- 100  # assuming you have this variable somewhere
starting_price <- last(NVDA$NVDA.Close)[[1]] #Closing price from 434.95 - last item in our frame
daily_mean <- mean(dailyReturn(NVDA)) #mean
daily_std_dev <- sd(dailyReturn(NVDA)) #standard deviation

returns_list <- matrix(0, nrow = no_of_sims, ncol = no_of_days) #define matrices
prices_list <- matrix(0, nrow = no_of_sims, ncol = no_of_days+1)

for(i in 1:no_of_sims) {
  returns_list[i,] <- rnorm(no_of_days, mean=daily_mean, sd=daily_std_dev)
  prices_list[i,] <- cumprod(c(starting_price, 1+returns_list[i,]))
}

# Calculate dynamic y-limits
y_min <- min(prices_list)
y_max <- max(prices_list)

# Plotting
plot(prices_list[1,], type='l', ylab="Simulated price of NVDA", xlab="Days", ylim=c(y_min, y_max))
for(i in 1:50) {
  lines(prices_list[i, ], type = 'l', col=i)
}
```

References

- Mathematical ideas - https://www.math.hkust.edu.hk/~maykwok/courses/MAFS5250/lecture%20notes/MAFS5250_Topic_5.pdf
- Financial background - https://www.investopedia.com/terms/m/montecarlosimulation.asp
- Definitions - https://en.wikipedia.org/wiki/Monte_Carlo_method
- R codes - https://www.countbayesie.com/blog/2015/3/3/6-amazing-trick-with-monte-carlo-simulations

- https://bstaton1.github.io/au-r-workshop/ch4.html
- https://rpubs.com/minnasan/monte_carlo_simulation_of_stock_market_returns