Pricing Options Using Neural Networks

Silèye Ba

Adjunct Lecturer, Neoma Business School, Paris
Machine Learning Scientist, L'Oreal Research & Innovation, Paris

11-12.03.2021

Tutorial outline

Neural network option pricing: motivations

- Many finance practisioners think machine learning/AI is the futur.
- AI models will be combined to or replace stochastic based models

Machine learning based pricing

- Replicate Monte Carlo based pricer with higher computational speed
- Machine learning models allow interpolating from sparse samples
- Allow learning very complex models without strong assumptions

Financial options

Definition

A financial option give its holder the right to buy or sell an asset as she determine the situation is favourable for her to do so

Remarks

- Assets:
  - Interest rates
  - Bonds
  - Equity indices
  - Foreign exchanges
  - Stocks
- Option cost money: need for pricing
- Option price derived from underlying asset

Some options types: vanilla call option

Call options gives the holder the right, but no the obligation to buy a particular asset S, for an agreed strike price K, at a specified expiry time T in the future

Call options pay off

- $S(t), t = 0, ..., T$: underlying asset prices
- $D(0, T)$: discount factor
- Option pay off at expiry time:

$$C(S(T), K) = \max(S(T) - K, 0)$$

- Option value at initial time:

$$C(0, S(0), K) = D(0, T)\mathbb{E}[C(S(T), K)]$$

Vanilla call option pricing analytical solution

Assumptions

Asset price S dynamic is driven by a geometric Brownian motion

$$\frac{S_{t+\Delta t} - S_t}{S_t} = r\Delta t + \sigma\sqrt{\Delta t}\xi_t \text{ with } \xi_t \sim N(0,1)$$

Analytical solution

$$C(0, S(0), K) = D(0,T)(F\mathcal{N}(d_1) - K\mathcal{N}(d_2))$$

$$\begin{aligned}
F &= S(0)e^{rT} \\
d_1 &= \frac{\log\frac{F}{K} - \frac{\sigma^2}{2}T}{\sigma\sqrt{T}} \\
d_2 &= d_1 - \sigma\sqrt{T} \\
\mathcal{N}(d_1) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{d} e^{-\frac{1}{2}x^2}dx
\end{aligned}$$

Some options types: vanilla put option

Put options give the holder the right, but not the obligation to sell a particular asset S, for an agreed strike price K at a specified expiry time T in the future

Call options pay off

- $S(t), t = 0, ..., T$: underlying asset prices
- $D(0, T)$: discount factor
- Put option pay off:

$$P(S(T), K) = \max(K - S(T), 0)$$

- Option value at initial time:

$$P(0, S(0), K) = D(0, T)\mathbb{E}[P(S(T), K)]$$

Some options types: vanilla put option

Analytical solution

$$P(0, S(0), K) = D(0, T)(K\mathcal{N}(-d_2) - F\mathcal{N}(-d_1))$$

$$\begin{aligned} F &= S(0)e^{rT} \\ d_1 &= \frac{\log \frac{F}{K} - \frac{\sigma^2}{2}T}{\sigma\sqrt{T}} \\ d_2 &= d_1 - \sigma\sqrt{T} \\ \mathcal{N}(d_1) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{d} e^{-\frac{1}{2}x^2} dx \end{aligned}$$

Some options types: an exotic option

Path dependant options

- Option value depends on path taken by the option underlying S up to expiry date T
- Barrier options: value depend on price $S(t)$ hitting or not a barrier price H before expiry date T

Barrier up in options

Up in options pay off only if barrier is reached before expiry date

- Pay off at expiry time:

$$C^{UI}(S(T)) = \mathbb{I}(\max_{0<t\leq T} S(t) > H) \max(S(T) - K, 0)$$

- Value at initial time:

$$C^{UI}(0, S(0), K, H) = D(0, T)\mathbb{E}[C^{UI}(S(T), K)]$$

Barrier up and in option pricing: analytical solution

Analytical solution

$$
\begin{aligned}
C^{UI}(0, S(0), K, H) &= \left(\frac{H}{S(0)}\right)^{\frac{2\nu}{\sigma^2}} \\
&\times \left(P\left(0, \frac{H^2}{S(0)}, K\right) - P\left(0, \frac{H^2}{S(0)}, H\right) + (H-K)D(0,T)\mathcal{N}(-d(H, S(0)))\right) \\
&+ C(0, S(0), H) + (H-K)D(0,T)\mathcal{N}(d(S(0), H))
\end{aligned}
$$

where:

$$
\begin{aligned}
\nu &= r - \frac{\sigma^2}{2} \\
d(x, y) &= \frac{\log \frac{x}{y} + \nu T}{\sigma\sqrt{T}}
\end{aligned}
$$

Monte Carlo option price estimation

Fundamental equation of product valuation

- $S(t)$: asset price driven by geometric Brownian motion (GBM)
- $D(0, T)$: discount factor
- $V(T) = f(S(T))$: payoff at expiry time
- Value at initial time

$$V(0) = \mathbb{E}[D(0, T)f(S(T))]$$

Monte Carlo estimation principle

- Sample $n = 1, ..., N$ trajectories $S_n(0), ..., S_n(T)$ using GBM model.
- Estimate product price as:

$$\hat{V}(0) = D(0, T)\frac{1}{N}\sum_{n=1}^{N} f(S_n(T))$$

Geometric Brownian motion price trajectory sampling

Given $S(t)$ we can deduce $S(t + \Delta t)$ from GBM model:

$$S_{t+\Delta t} = S_t \left(1 + r\Delta t + \sigma\sqrt{\Delta t}\xi_t\right)$$

where :

$$\xi_t \sim N(0, 1)$$

Simulating scenarii for price process $(S(t), t = 1, ..., T)$

1. Choose a time step Δt and number of simulation N
2. Construct a time partition $t_i = i\Delta t, i = 0, ..., M$ with $M = \frac{T}{\Delta t}$
3. For $n = 1, ..., N$:
   a. Initialize $S_{n,0} = S(0)$
   b. For $i = 0, ..., M$:
      i. Simulate $\xi \sim N(0, 1)$
      ii. Compute $S_{n,i} = S_{n,i-1}\left(1 + r\Delta t + \sigma\sqrt{\Delta t}\xi\right)$
   c. For each scenario price at expiry time is $S_n(T) = S_{n,M}$
4. Use $S_1(T)$, ..., $S_N(T)$ to compute Monte Carlo approximations for product price

Vanilla call option pricing with Monte Carlo solution

- Pay off at expiry time:

$$C(S(T)) = \max(S(T) - K, 0)$$

- Value at initial time:

$$C(0, S(0), K) = D(0, T)\mathbb{E}[C(S(T))]$$

Monte Carlo solution

If for sampled scenarios, for $n = 1, ..., N$, price at expiry time is $S_n(T)$ then Monte Carlo estimation is:

$$C(0, S(0), K) \approx D(0, T)\frac{1}{N}\sum_{n=1}^{N} C(S_n(T))$$

Vanilla put option pricing with Monte Carlo

- Pay off at expiry time:

$$P(S(T)) = \max(K - S(T), 0)$$

- Value at initial time:

$$P(0, S(0), K) = D(0, T)\mathbb{E}[P(S(T))]$$

Monte Carlo solution

If for sampled scenarios, for $n = 1, ..., N$, price at expiry time is $S_n(T)$ then Monte Carlo estimation is:

$$P(0, S(0), K) \approx D(0, T)\frac{1}{N}\sum_{n=1}^{N} P(S_n(T))$$

Black Scholes option pricing with neural networks

Data

- Inputs:
  - r: interest rates
  - σ: volatilities
  - $S(0)$: spot prices
  - K: strike prices
  - T: expiry dates
- Output:
  - $V(r, \sigma, S(0), K, T)$: initial values

Problem:

1. Generate dataset using analytical Black Scholes call option pricer
2. Build a Monte Carlo Black Scholes call option pricer
3. Generate dataset using Monte Carlo Black Scholes call option pricer
4. Build a 3 layer neural network to map inputs onto output

Heston option pricing with neural networks

Heston price model

$$
\begin{aligned}
\frac{S_{t+\Delta t} - S_t}{S_t} &= \mu \Delta t + \sqrt{v_t \Delta t}\,\xi_t^S \\
v_{t+\Delta t} - v_t &= \kappa(\theta - v_t)\Delta t + \nu\sqrt{v_t \Delta t}\,\xi_t^v \text{ with } 2\kappa\theta > \nu^2
\end{aligned}
$$

- μ: asset rate of return
- θ: square volatility reversal value
- κ: rate at which v_t reverts to θ
- ν: volatility standard deviation

Problem

1. Build a Monte Carlo Heston call option pricer
2. Build a dataset using the pricer: (inputs, ouput)
3. Build a 3 layer neural network to map inputs onto output