Gated Neural Networks for Option Pricing

Enforcing Sanity in a Black Box Model

Y. Yang, Y. Zheng and **T. Hospedales**

Updated: 2016/09/21

Queen Mary, University of London

Overview

ML for Option Pricing. With implications for:

- How do we know if/when a model is (sufficiently) sane?
- When is a horse more than just overfitting?
- How can one make ML robust to horses?

Option Pricing

Problem Definition

What is an *option*?

Option: Formal Definition

A (call) option is a contract that gives the buyer the *right*, but not the *obligation*, to acquire the underlying asset (e.g., stock) at a specified price (called strike price) on a certain future date (called maturity date).

Problem Definition (Cont.)

Option: An Example

At time $t = 0$ (today), a company's stock is worth \$100, and Bob buys a call option with strike price \$110 and maturity date $T = 5$ (five days later). After five days,

- If the company's stock price is \$120, Bob will exercise the option to get the stock by paying the strike price \$110. If he sells the stock immediately at \$120, he will profit \$10.
- If the company's stock price is below \$110, Bob will not exercise the option and he has only lost the price paid for the option.

Problem Definition (Cont.)

Option Pricing

Option Pricing is to answer *how much should the mentioned call option contract be sold for at* $t = 0$*?*

Econometrics v.s. Machine Learning

Different Approaches

Econometrics (Let's talk about Levy Process)

- I observed the price (dependent variable) and the relevant factors (independent variables) in the market.
- I made assumptions on these variables (which kind of stochastic process can describe it?)
- I figured out a formula ☺

Different Approaches (Cont.)

Machine Learning (It is a regression problem)

- I observed the price (dependent variable) and the relevant factors (independent variables) in the market
- I feed them into a neural network (kernel machine, random forest)
- I got a model ☻

Different Approaches (Cont.)

Claims by Machine Learning Researchers

- The trained model does the same job as an economist's formula – produces estimated price given the same inputs, e.g., strike price, asset price, time-to-maturity, etc.
- It performs better in terms of smaller difference when the estimated price is compared to real market's price.

Different Approaches (Cont.)

Questions

- Do the machine learning methods *really* solve the problem? or they *actually* built a Potemkin village?
  - Sadly, No
- Should I trust the outputs of *block-box* ML methods?
  - Sadly, No
- Generally, How can I *examine* a machine learning method apart from reading its MSE?
  - We'll give some examples.
- How to de-horse my model?
  - We'll give some examples.

An econometrically valid option pricing model

Notations correspond to the example at beginning

At time $t = 0$, a company's stock is worth \$100*, and Bob buys a call option with strike price* \$110 *and maturity date $T = 5$ (five days later).*

	Value	Meaning
K	110	Strike Price
S_t	100	Underlying Asset Price at Time t
m	1.1	Moneyness
T	5	Future Date (when the option is ready to exercise)
t	0	Current Data (when to price the option)
τ	5	Time to Maturity
S_T	?	Underlying Asset Price at Time T (unknown at time t)

Option pricing model that obeys no-arbitrage principle

Aiming for $\hat{c}(K, S_t, \tau) \approx c$.

$\hat{c}$ is the estimated price, c will be the true market price.

General arbitrage free formula

$$\hat{c}(K, S_t, \tau) = \mathrm{e}^{-r\tau} \int_0^{\infty} \max(0, S_T - K) f(S_T | S_t, \tau) \mathrm{d}S_T.$$

Option pricing model

General arbitrage free formula

$$\hat{c}(K, S_t, \tau) = \mathrm{e}^{-r\tau} \int_0^{\infty} \max(0, S_T - K) f(S_T | S_t, \tau) \mathrm{d}S_T$$

$\int_0^{\infty} \max(0, S_T - K) f(S_T | S_t, \tau) \mathrm{d}S_T$ is the expected value of option at time T, so it has to be discounted at risk free rate r.

$\mathrm{e}^{-r\tau}$ is a discount factor.

Option pricing model – Core Part

Core part

$$\tilde{c}(K, S_t, \tau) = \int_0^{\infty} \max(0, S_T - K) f(S_T|S_t, \tau) \mathrm{d}S_T$$

$\max(0, S_T - K)$ is the potential revenue of having this option at time T

$f(S_T|S_t, \tau)$ is the probability density of that revenue

=> Integral term is in fact the expected revenue at time T.

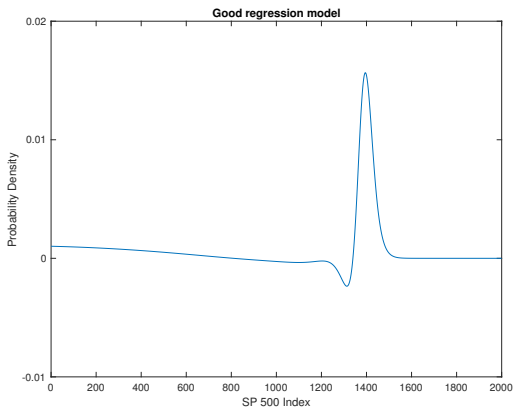
Why do we need rational option pricing?

- Simple ML solution $w = \min \|\hat{c}_w(K, S_t, \tau) - c\|_2^2$
  - Use your favourite kernel method / deep learner / etc.
- What does a *sane* model mean, and why do we need it?
  - We want to *explain* the behaviour of the market, not just predict it.
  - Option pricing is interrelated with other financial concepts and prediction tasks.
  - (Some trading strategies depend on 'rational' prediction strategies.)

Why do we need rational option pricing?

A low-MSE option pricing model can be invalid / meaningless.

- $\tilde{c}(K, S_t, \tau) = \int_0^\infty \max(0, S_T - K) f(S_T|S_t, \tau) \mathrm{d}S_T$
  - E.g., $\frac{\partial^2 \tilde{c}}{\partial K^2} = p(S_T|S_t, \tau)$.
- Model second derivative: (risk neutral) **distribution over future asset price.**
  - Second derivative must be non-negative (its a distribution). Doesn't hold for arbitrary regression models $\hat{c}(K, S_t, \tau)$.



  - E.g., This implied asset price distribution $p(S_T|S_t, \tau)$ negative, and doesn't sum to 1.

Option pricing model – Rationality Checks

$\tilde{c}$ HAS TO PASS A NUMBER OF SANITY CHECKS

$\tilde{c}$ can not be an arbitrary function, it has to meet some conditions to be meaningful.
ML models implicitly assume that model learns all these from data *automatically* – unfortunately this hope is *not* true.

Option pricing model – Condition 1

English: "*Asset Price CDF Constraint*"

$$\frac{\partial \tilde{c}}{\partial K} \leq 0 \qquad \text{(C1)}$$

Why?

$\frac{\partial \tilde{c}}{\partial K} = \int_0^K f(S_T|S_t, \tau)\mathrm{d}S_T - 1$ and $\int_0^K f(S_T|S_t, \tau)\mathrm{d}S_T$ is a cumulative distribution function $\mathbb{P}(S_T \leq K)$ thus its value can not be larger than one.

Option pricing model – Condition 2

English: "*Asset Price (Risk Neutral) Distribution Constraint*"

$$\frac{\partial^2 \tilde{c}}{\partial K^2} \geq 0 \qquad \text{(C2)}$$

Why?

$\frac{\partial^2 \tilde{c}}{\partial K^2} = f(S_T|S_t, \tau)$ is a probability density function so its value can not be smaller than zero.

Option pricing model – Condition 3

English: "*More costly to hedge far future*"

$$\frac{\partial \tilde{c}}{\partial \tau} \geq 0 \qquad (C_3)$$

Why?

This is intuitive: the longer you wait (larger τ), the higher chance that the underlying asset price will eventually be greater than the strike price. Thus the price should be non-decreasing with time to maturity.

Option pricing model – Condition 4

English: "*Useless option has no value*"

$$\lim_{K \to \infty} \tilde{c}(K, S_t, \tau) = \tilde{c}(\infty, S_t, \tau) = 0 \qquad (C4)$$

Why?

If the strike price is infinity, the option price should be zero because the underlying asset price is always smaller than the strike price. There is no point in trading the option.

Option pricing model – Condition 5

English: "*Market price for a trivial option*"

$$\tilde{c}(K, S_t, 0) = \max(0, S_t - K) \quad \text{when} \quad \tau = 0 \qquad \text{(C5)}$$

Why?

When $\tau = 0$, the option is ready to execute immediately, so its price should be exactly $\max(0, S_t - K)$ since $S_t = S_T$.

Option pricing model – Condition 6

English: "*Boundary conditions to disallow arbitrage*"

$$\max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t \qquad \text{(C6)}$$

Why?

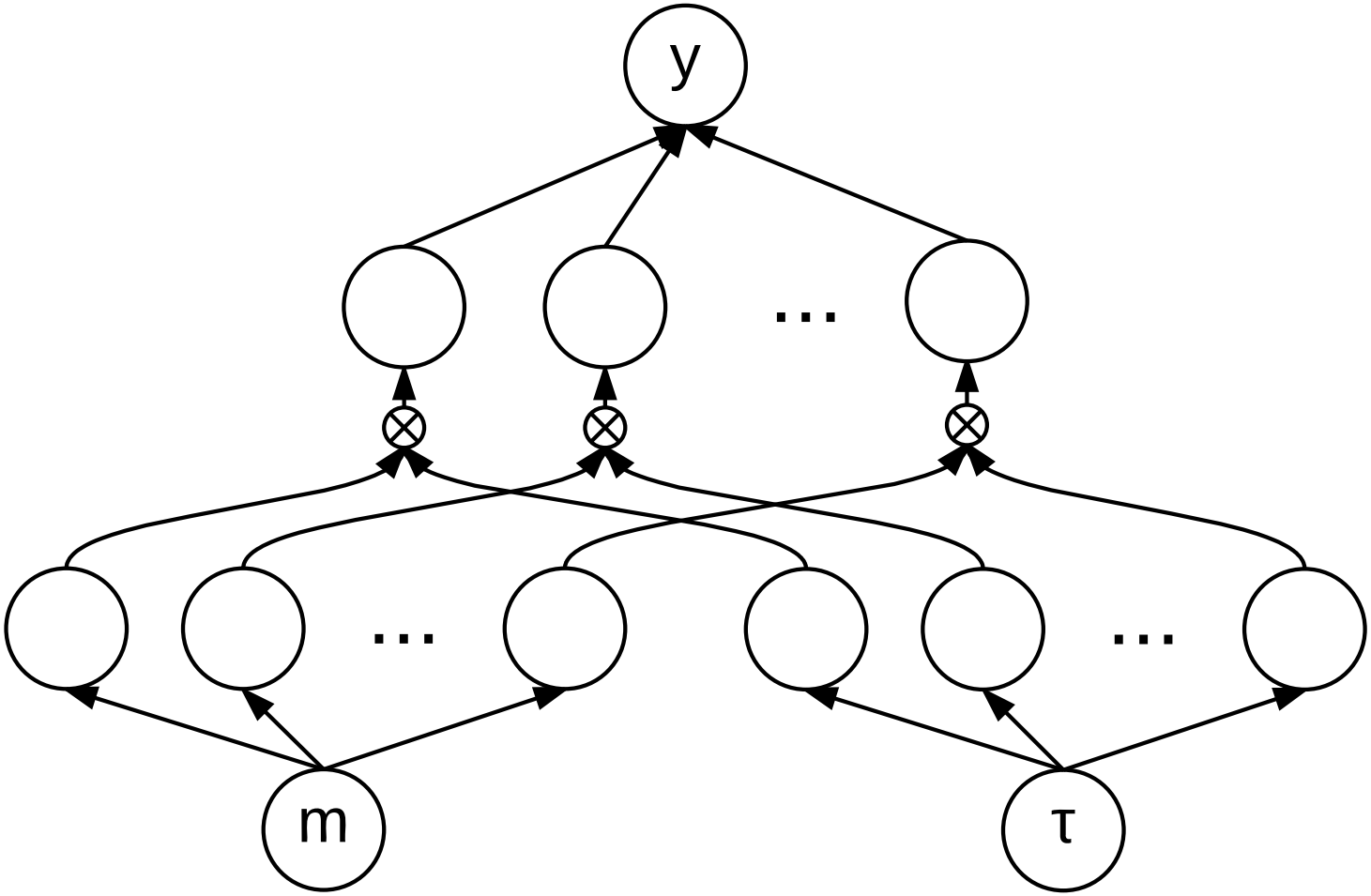
This boundary can be easily derived from put-call parity and payoff of the call option. Call option price can not exceed the underlying price, otherwise an investor can arbitrage by buying the stock and selling the option at same time and closing all positions when the option is expired.

Can we meet these predictions?

- How can we make a ML model that meets these conditions for rational option pricing?
  - $\frac{\partial \tilde{c}}{\partial K} \leq 0$
  - $\frac{\partial^2 \tilde{c}}{\partial K^2} \geq 0$
  - $\frac{\partial \tilde{c}}{\partial \tau} \geq 0$
  - $\lim_{K \to \infty} \tilde{c}(K, S_t, \tau) = \tilde{c}(\infty, S_t, \tau) = 0$
  - $\tilde{c}(K, S_t, 0) = \max(0, S_t - K) \quad \text{when} \quad \tau = 0$
  - $\max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t$
- Surely not that those horrible black box neural nets?!
  - We claim a carefully designed NN can indeed do it.

Our proposed models

Single model - Graph



Single model

We assume the pricing model is rescalable w.r.t. S_t, i.e., $\tilde{c}(\frac{K}{S_t}, 1, \tau) := \frac{\tilde{c}(K, S_t, \tau)}{S_t}$ and build a neural network model $y(m, \tau)$ for $\tilde{c}(\frac{K}{S_t}, 1, \tau)$. As $m = \frac{K}{S_t}$, we have $y(m, \tau) \equiv \tilde{c}(m, 1, \tau)$

Formula (Single Model)

$$y(m,\tau) = \sum_{j=1}^{J} \sigma_1(\tilde{b}_j - m e^{\tilde{w}_j}) \sigma_2(\bar{b}_j + \tau e^{\bar{w}_j}) e^{\hat{w}_j}.$$

$\sigma_1(x) = \log(1 + e^x)$ (softplus function) and $\sigma_2(x) = \frac{1}{1+e^{-x}}$ (sigmoid function). J is the number of hidden neurons.

Single model - Sanity Check

We can verify that this model meets conditions C1-C3 since

$$\frac{\partial y}{\partial m} = \sum_{j=1}^{J} -e^{\tilde{w}_j}\sigma_2(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \leq 0$$

$$\frac{\partial^2 y}{\partial m^2} = \sum_{j=1}^{J} e^{2\tilde{w}_j}\sigma_2'(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \geq 0$$

$$\frac{\partial y}{\partial \tau} = \sum_{j=1}^{J} e^{\bar{w}_j}\sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2'(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \geq 0$$

Single model - Sanity Check (Cont.)

and the conditions C1, C2 and C3 can be rewritten as,

$$\frac{\partial \tilde{c}}{\partial K} = \frac{\partial S_t y}{\partial K} = S_t \frac{\partial y}{\partial m}\frac{\partial m}{\partial K} = S_t \frac{\partial y}{\partial m}\frac{1}{S_t} = \frac{\partial y}{\partial m} \leq 0$$

$$\frac{\partial^2 \tilde{c}}{\partial K^2} = \frac{\partial^2 y}{\partial m \partial K} = \frac{\partial^2 y}{\partial m^2}\frac{\partial m}{\partial K} = \frac{1}{S_t}\frac{\partial^2 y}{\partial m^2} \geq 0$$

$$\frac{\partial \tilde{c}}{\partial \tau} = \frac{\partial S_t y}{\partial \tau} = S_t \frac{\partial y}{\partial \tau} \geq 0$$

Condition C4 can be verified as $m \to \infty$ when $K \to \infty$, and $\sigma_1(\tilde{b}_j - m e^{\tilde{w}_j}) = 0$ when $m \to \infty$. Therefore $y = 0$ and then $\tilde{c} = S_t y = 0$. This explains why there is no top layer bias term.

=> Take Home: Can enforce rationality (CDF, PDF, time & useless option constraints) through careful choice of activations & weight constraints.

Single model - Sanity Check (Cont.)

Condition C5 ("*Trivial Option*" constraint).

$\tilde{c}(K, S_t, 0) = \max(0, S_t - K)$ when $\tau = 0$

Hard to achieve by network architecture design.

meet them by synthesising virtual option contracts with $\tau = 0$.

τ	$e^{-r\tau}$	S_t	K	$\hat{c}$ and c	$\tilde{c}$	y (expected)
0	1	1000	10	990	990	0.9900
0	1	1000	20	980	980	0.9800
0	1	1000	990	10	10	0.0100
..	..	..	..	..	..	..
0	1	1100	10	1090	1090	0.9909
..	..	..	..	..	..	..

Table: Virtual Options for Condition C5

Single model - Sanity Check (Cont.)

Condition C6 ("*No Arbitrage Boundary Conditions*")

$\max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t$

Hard to achieve by network architecture design.

Meet them by synthesising virtual option contracts with $K = 0$.

τ	$e^{-r\tau}$	S_t	K	$\hat{c}$ and c	$\tilde{c}$	y (expected)
7	0.98	1000	0	1000	1020	1.0200
14	0.95	1100	0	1100	1158	1.0526
..	..	..	..	..	..	..

Table: Virtual Options for Condition C6

Multi model

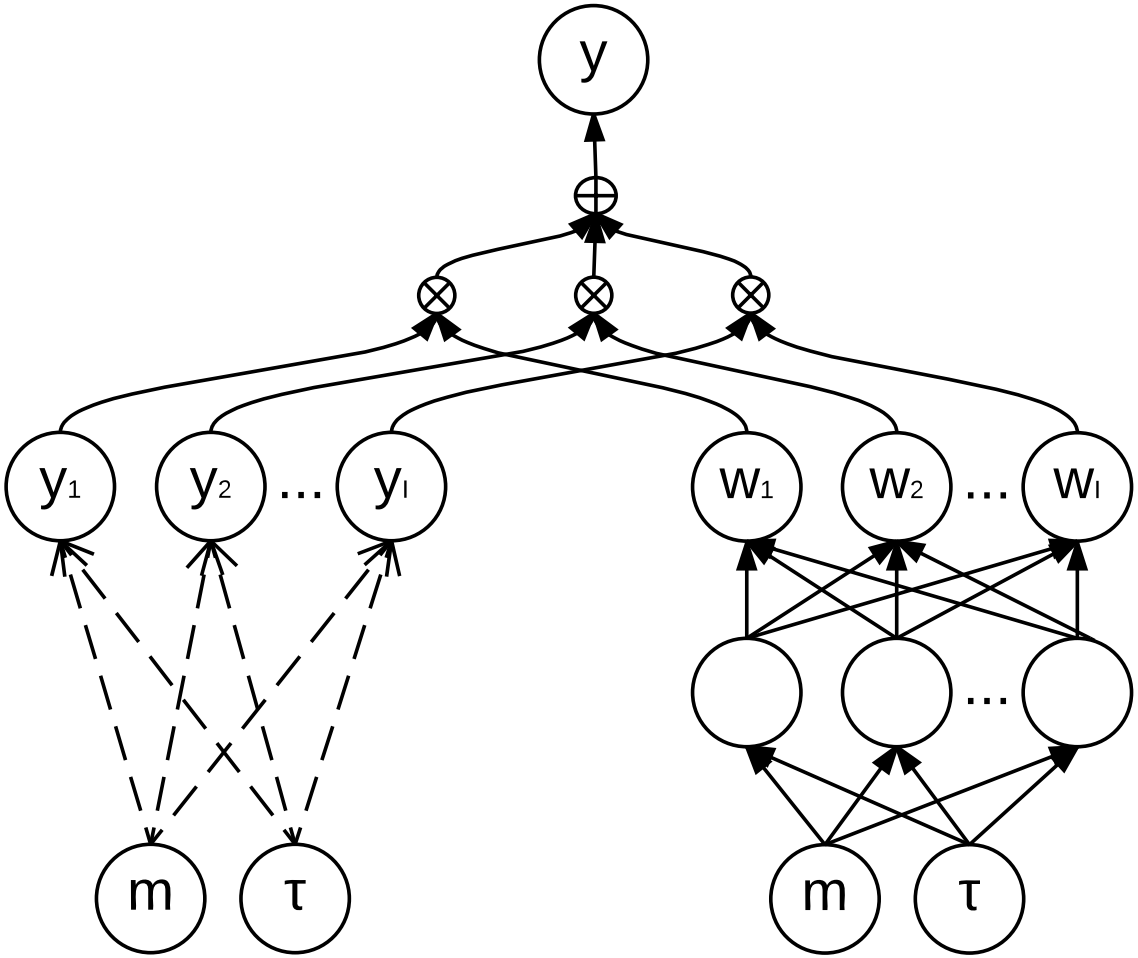
From Single to Multi-model

Single model meets constraints. But weak performance (MSE): It is insufficiently expressive (single model for all options).

=> Doesn't deal with the phase shifts, e.g., "*deep out of money*".

We propose a multi model. It jointly trains a number of *single* pricing models, as well as a weighting model to softly switch them (cf mixture of experts).

Multi model - Graph



Multi model (Cont.)

The single model's performance (MSE) is unsatisfactory as it is insufficiently expressive, so we propose a multi model. It jointly trains a number of *single* pricing models, as well as a weighting model to softly switch them (cf mixture of experts).

The multi model's left-hand side has $i = 1 \ldots I$ single pricing models:

Left-hand side

$$y_i(m, \tau) = \sum_{j=1}^{J} \sigma_1(\tilde{b}_j^{(i)} - m\mathrm{e}^{\tilde{w}_j^{(i)}})\sigma_2(\bar{b}_j^{(i)} + \tau\mathrm{e}^{\bar{w}_j^{(i)}})\mathrm{e}^{\hat{w}_j^{(i)}}$$

Multi model (Cont.)

Its right-hand branch is a network with one K unit hidden layer, and the top layer has an I-way softmax activation function that provides a model selector for the left branch.

Right-hand side

$$w_i(m,\tau) = \frac{e^{\sum_{k=1}^{K} \sigma_2(m\dot{W}_{1,k}+\tau\dot{W}_{2,k}+\dot{b}_k)\ddot{W}_{k,i}+\ddot{b}_i}}{\sum_{i=1}^{I} e^{\sum_{k=1}^{K} \sigma_2(m\dot{W}_{1,k}+\tau\dot{W}_{2,k}+\dot{b}_k)\ddot{W}_{k,i}+\ddot{b}_i}}$$

Multi model (Cont.)

Finally, the overall output y is the softmax weighted average of the I local option pricing models' outputs. Due to the softmax activation, the sum of weights (w_i's) is one.

Final Prediction

$$y(m,\tau) = \sum_{i=1}^{I} y_i(m,\tau) w_i(m,\tau).$$

Multi model - Sanity Check

The multi-network can still meet Conditions C1, C3-C6.

Outstanding issue: multi-model breaks Condition C2 ("Asset Price Distribution Constraint")

To alleviate this, we use the *learning from hints* trick.

Enforcing $\frac{\partial^2 \tilde{c}}{\partial K^2} \geq 0$: a new loss term

$$\mathcal{L} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \max(0, \frac{\partial y}{\partial m}(m_{p,q}, \tau_q) - \frac{\partial y}{\partial m}(m_{p,q} + \Delta, \tau_q))$$

Multi model - Sanity Check (Cont.)

Enforcing $\frac{\partial^2 \tilde{c}}{\partial K^2} \geq 0$: a new loss term

$$\mathcal{L} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \max(0, \frac{\partial y}{\partial m}(m_{p,q}, \tau_q) - \frac{\partial y}{\partial m}(m_{p,q} + \Delta, \tau_q))$$

Δ: a small number, e.g., $\Delta = 0.001$. Q: a number of unique time-to-maturity.

P: number of pseudo data generated.

=> New loss pushes $\frac{\partial y}{\partial m}(m, \tau)$ to a monotonically increasing function w.r.t. m, thus $\frac{\partial^2 y}{\partial m^2}$ tends to be larger than zero.

Benchmark

We do something big – **70** times larger dataset than previous studies.

Our dataset

- 5,139 trading days
- 3,029,327 option contracts
- all actively traded call options
- call options converted from all actively traded put options
- carefully pre-processed

Model Performance

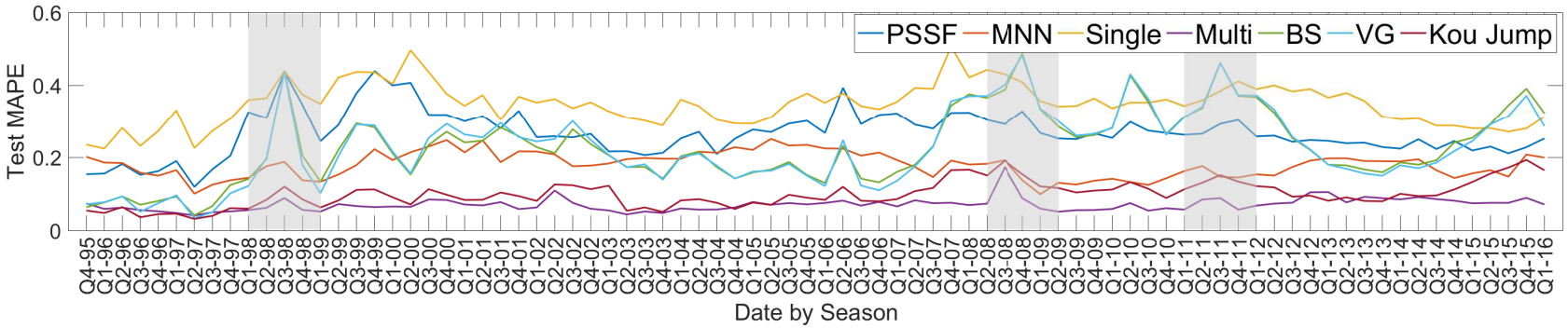


Figure: Test MAPE by Seasons: The shadowed parts correspond to the following events: Dot-com bubble (1998), global financial crisis (2008), and European debt crisis (2011).

Model Performance (Cont.)

	Train		Test	
	MSE	MAPE (%)	MSE	MAPE (%)
PSSF	267.48	25.77	269.56	26.25
MNN	50.08	16.89	63.16	18.22
Single	579.74	34.74	580.47	34.99
Multi	**9.91**	**5.75**	**12.11**	**6.84**
BS	63.73	21.64	64.71	22.42
VG	55.40	18.42	61.57	22.64
Kou Jump	18.37	8.69	20.13	9.90

Table: Quantitative comparison of pricing on **3 million** contracts.

Model Examination Beyond MSE/MAPE

Beyond Mean Squared Error

- MSE/MAPE: Solved. But we care about more than this:
- We check $f(x|S_t, \tau)$. PDF for the asset price at time T given its price at time t (i.e., S_t) and time difference (i.e., $\tau = T - t$).
  - $f(x|S_t, \tau)$ has to be a valid probability density function (i) non-negative and (ii) integrate to one
  - $f(x|S_t, \tau)$ should also be economically reasonable, e.g. asset price close to zero after $\tau = 7$ days should be unlikey.

Model Examination Beyond MSE/MAPE (Cont.)

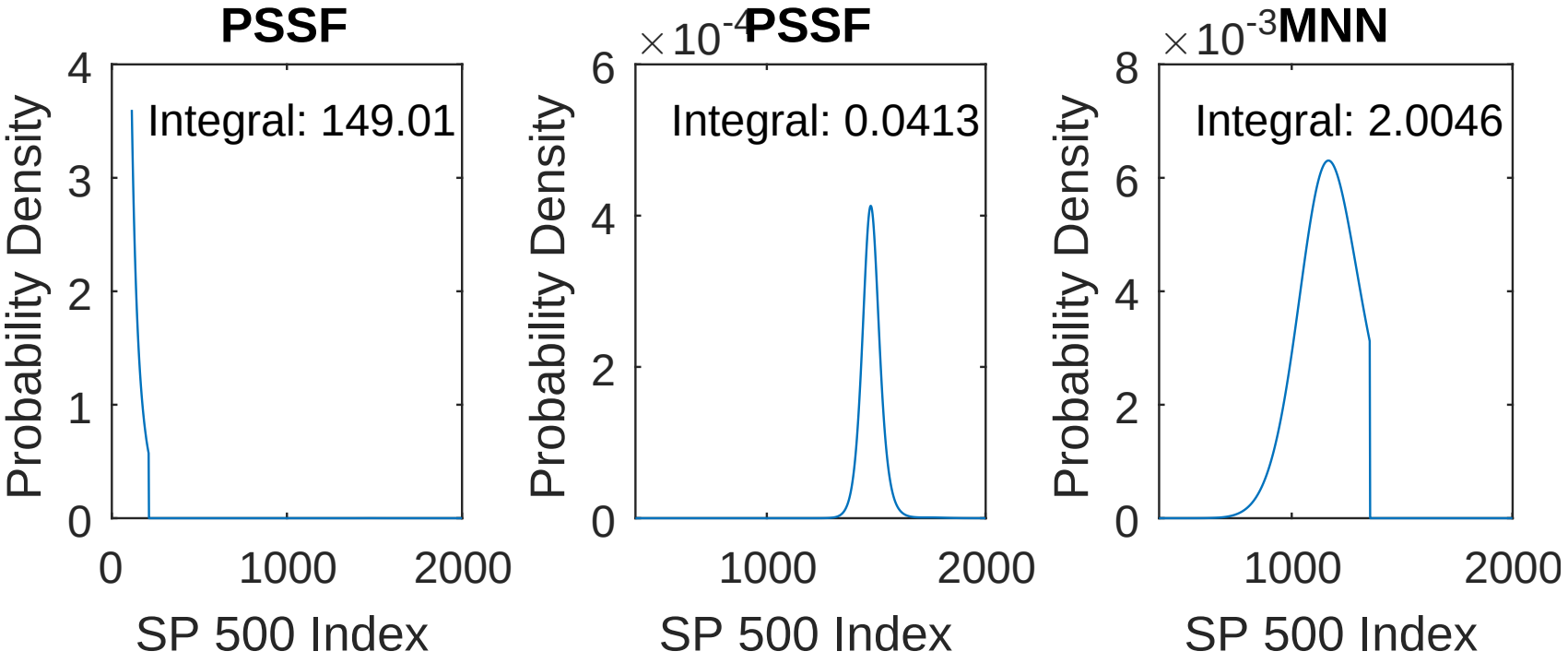


Figure: Neither PSSF nor MNN produces a valid distribution. Left: PSSF risk neural density for X-axis range $[0, 2000]$. Middle: PSSF risk neural density for X-axis range $[400, 2000]$ (note the difference on Y-axis scale). Right: Risk neural density of MNN.

Model Examination Beyond MSE/MAPE (Cont.)

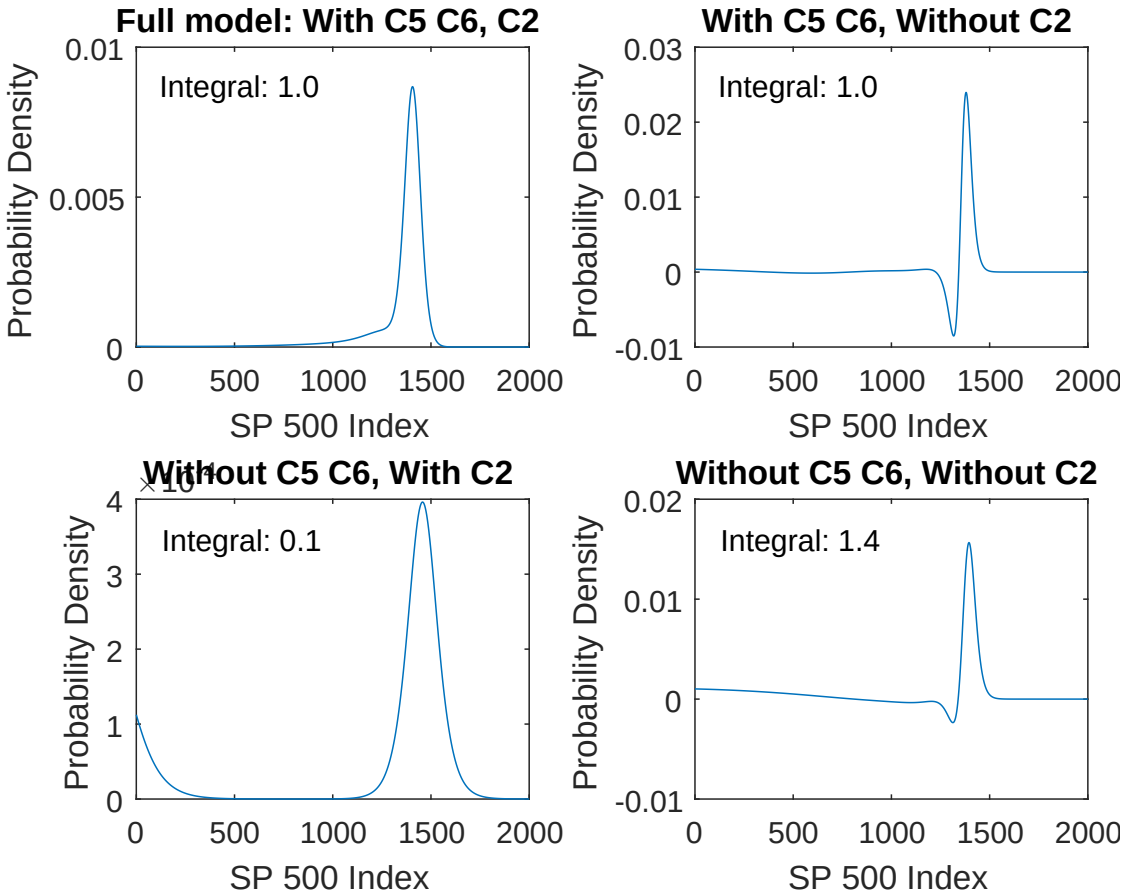


Figure: Implied distribution over future asset price. Top Left: Our multi model. Top Right: Without second derivative constraint (C2), we observe invalid negative values. Bottom Left: Without virtual options (conditions C5 and C6): we see density around zero which is senseless. Bottom Right: No derivative constraint or virtual options gives invalid and meaningless density.

Discussion

Summary

- We can build high-accuracy neural networks that....
  - Draw their conclusions in the *way* that we want.
  - ...Via meaningful internal representations.
  - Make predictions with certain sanity guarantees.
- Notes:
  - This was NOT an overfitting issue. Test-MSE good before we started.
  - This was an 'intrinsic' horse infestation, not a biased dataset.

Discussion

Useful tips

- Neural network model does not have to be a black-box.
  - With attention, can be 'grey'.
- Pay attention to the choice on activation functions
- Think about controlling the weight/bias terms
- Can softly enforce constraints through synthesising pseudo data.

Types of Constraints

- Pin the model's behaviour at corner cases:
  - $\lim_{K \to \infty} \tilde{c}(K, S_t, \tau) = \tilde{c}(\infty, S_t, \tau) = 0$
  - $\tilde{c}(K, S_t, 0) = \max(0, S_t - K) \quad \text{when} \quad \tau = 0$
  - $\max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t$
- Constrain 'how' the model uses inputs with semantics.
  - $\frac{\partial \tilde{c}}{\partial K} \leq 0$
  - $\frac{\partial^2 \tilde{c}}{\partial K^2} \geq 0$
  - $\frac{\partial \tilde{c}}{\partial \tau} \geq 0$
- Collection of constraints can have qualitative impact. E.g., internally encode PDFs.

Discussion

- How do we know if/when a model is (sufficiently) sane?
  - Domain knowledge may give you other requirements.
  - But necessary ≠ sufficient. Not insane ≠ completely sane.
- When is a horse not just overfitting?
  - Our constraints are inductive bias to improve generalisation, but also much more...
- When is it important to avoid horses?
  - When your model/domain has other requirements. E.g., implying distributions.
  - When you want to understand rather than 'just' predict.
    - (Not just academic. E.g., Certain specific trading strategies depend on knowing asset price distribution variance.)
- How can one make ML robust to horses?
  - Insight to understand requirements and semantics.
  - Careful engineering to meet them.

Thank you.