AXOR Corporate Credit Pricing Model Accuracy Report

Nathaniel Powell
Deep Market Making, Inc.
deepmarketmaking.com

September 2025

Contents

1 Introduction

The production Deep MM corporate bond pricing model named AXOR is a deep neural network with 80 million parameters. It currently predicts the 5th through the 95th percentiles of price, spread, and yield to maturity, in 5% increments, which allows many use cases that would otherwise not be possible or as statistically robust without it.

The Deep MM model uses TRACE prints for the same bond and related bonds (bonds of the same issuer, industry, and credit rating), benchmark treasury yields, bond ETFs, SPY, QQQ, and S&P ratings, as well as the properties of the actual or hypothetical trade. In total, the model incorporates over 2,000 features to infer each output.

The models were evaluated out-of-sample on trades in the Deep MM universe from July through early September, 2025. Only trades of notional size $150k or above were considered. Only trades for bonds with valid S&P ratings were used.

At this time, Deep MM has one primary model:

1. Price Model

The model is also conditional, taking into account the side, quantity, and ATS flag of historical trades or hypothetical trades that a user might be considering executing.

2 Use Cases

The Deep MM model enables:

- Automated runs generation
- Probability-based alerting
- Execution risk management
- Portfolio management
- Many others

3 Price Model Accuracy

3.1 Overall

Deep MM's price model was evaluated out-of-sample on a broad universe of trades (size $150k or above). The table below shows our model's absolute error metrics for Investment Grade (IG) and High Yield (HY) segments. For additional context, we also include the corresponding accuracy numbers for

Segment	Trade Count	Median	75th Percentile	95th Percentile
IG (Deep MM)	853,924	0.0504	0.105	0.295
IG (CP+)	509,811	0.0877	0.184	0.515
HY (Deep MM)	272,620	0.0874	0.171	0.443
HY (CP+)	193,033	0.1571	0.307	0.789

Table 1: Price Absolute Error Comparison: Deep MM vs. CP+ (US segments)

CP+ (as reported on page 11 of the CP+ whitepaper) for US bonds. Notice that our model's errors are significantly lower than those reported for CP+.

Figure 1 and Figure 2 illustrate the accuracy metrics for each segment. In each grouped bar chart the 50th, 75th, and 95th percentile errors are shown. Deep MM's errors are shown in gradients from light teal (50th percentile) to dark blue (95th percentile), while CP+'s errors are in gray with a crosshatch pattern for the IG segment and orange with a crosshatch pattern for the HY segment.

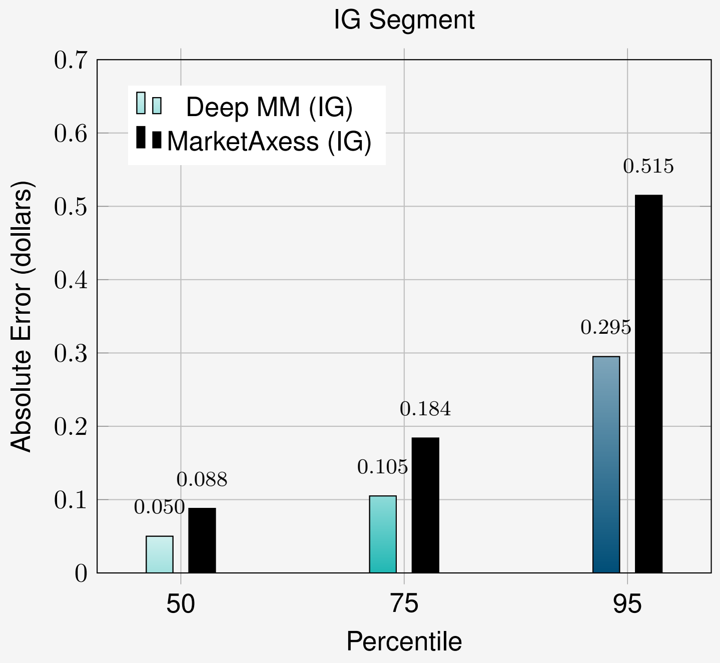

Figure 1: Price Accuracy for IG Segment

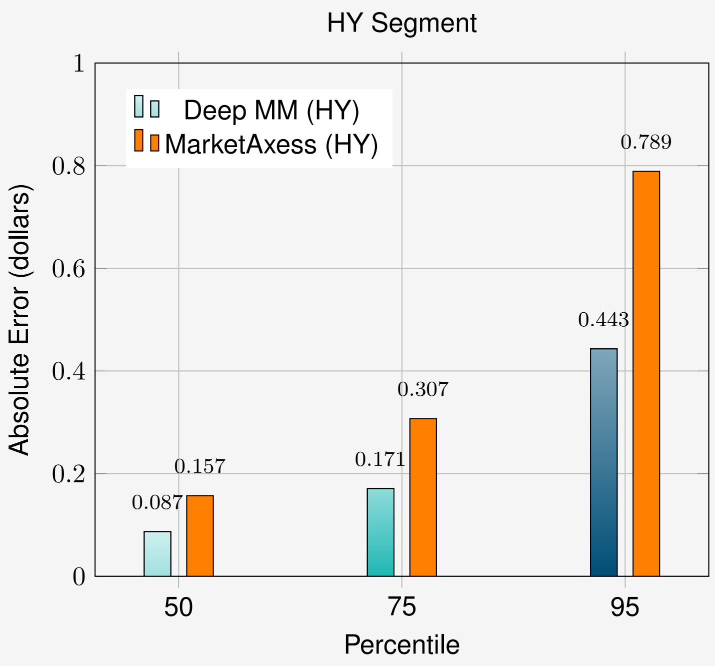

Figure 2: Price Accuracy for HY Segment

3.2 By Trading Hour

Figure 3 shows that for Investment Grade bonds, Deep MM's median deviance is not only lower than the MarketAxess benchmark (0.0877) but also remarkably steady throughout the trading day. The slight variation between 9 and 16 US/Eastern confirms the robustness of the Deep MM model under varying market conditions.

3.3 By Bond Trading Volume

Figure 4 illustrates that for high yield bonds, the median deviation of Deep MM (which is approximately between 0.09 and 0.12) is significantly below the MarketAxess benchmark of 0.1571. Despite minor fluctuations, the Deep MM model exhibits stable performance throughout the trading day, reinforcing its reliability even in more volatile high-yield markets.

3.4 By Size of Trade

Figures 7 and 8 show accuracy is consistently better regardless of the size of the trade.

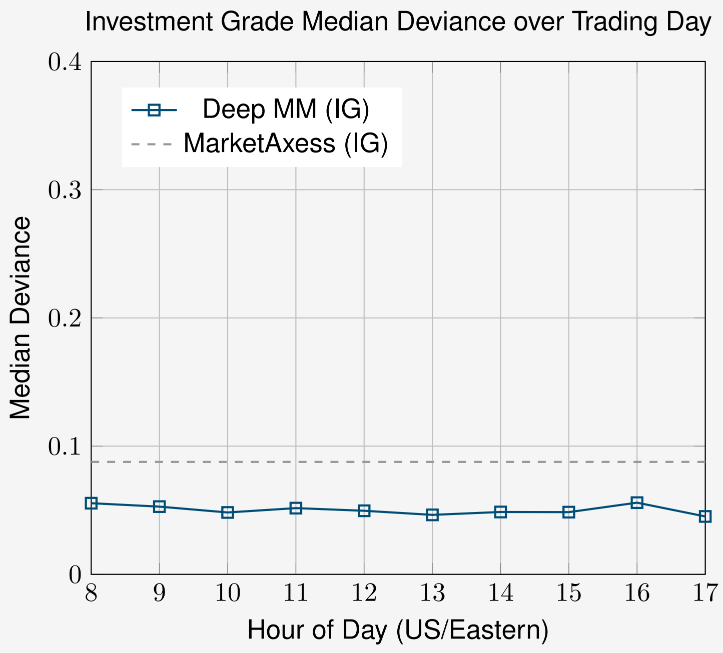

Figure 3: Deep MM's Investment Grade median deviance is consistently lower than MarketAxess across market hours.

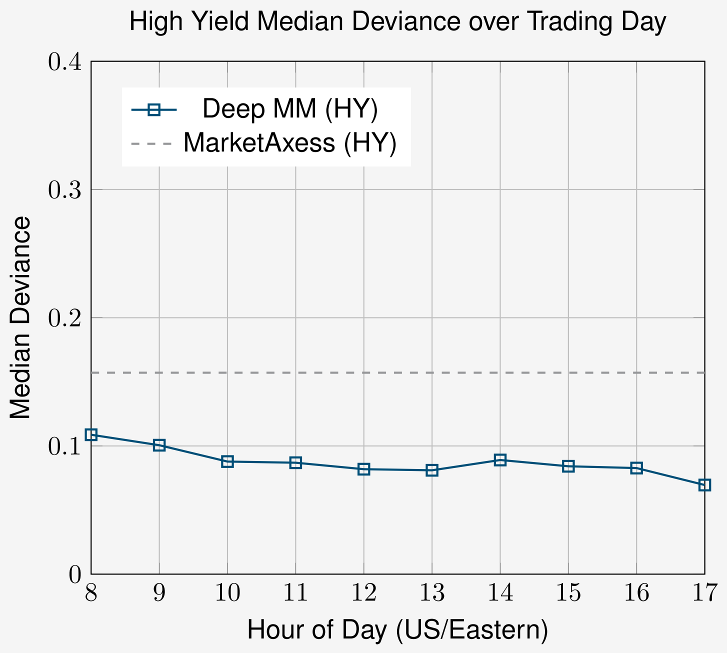

Figure 4: Deep MM's High Yield median deviance is consistently lower than MarketAxess across market hours.

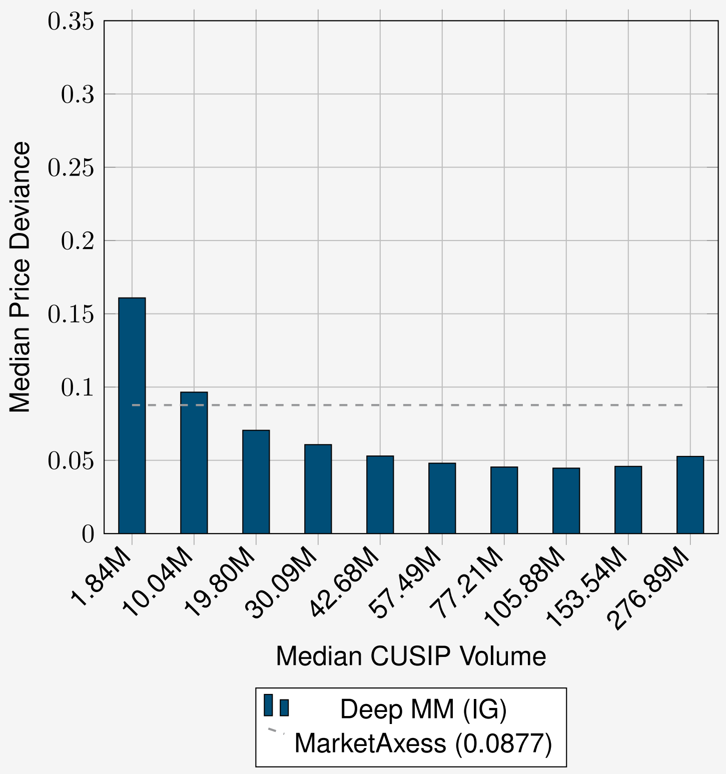

Figure 5: Investment Grade: Median price deviance vs. median trade volume per CUSIP with MarketAxess benchmark at 0.0877. We don't know what this curve looks like for MarketAxess but we included their reported median.

3.5 Calibration Curves

Figures 9 and 10 display the calibration curves for Investment Grade (IG) and High Yield (HY) price predictions, respectively. The dashed gray line represents the ideal calibration $y = x$, which indicates perfect agreement between the predicted prices and the corresponding calibration probability. Note that the calibration curves are close to the ideal, resulting in low probability bias.

Overall, these calibration curves demonstrate that our models capture most of the systematic variation in bond prices. However, the observed deviations in the IG model highlight opportunities for further refinement, potentially through increased model complexity or enhanced data sources. The HY model's calibration, being closer to ideal, reinforces its robustness in a challenging market segment.

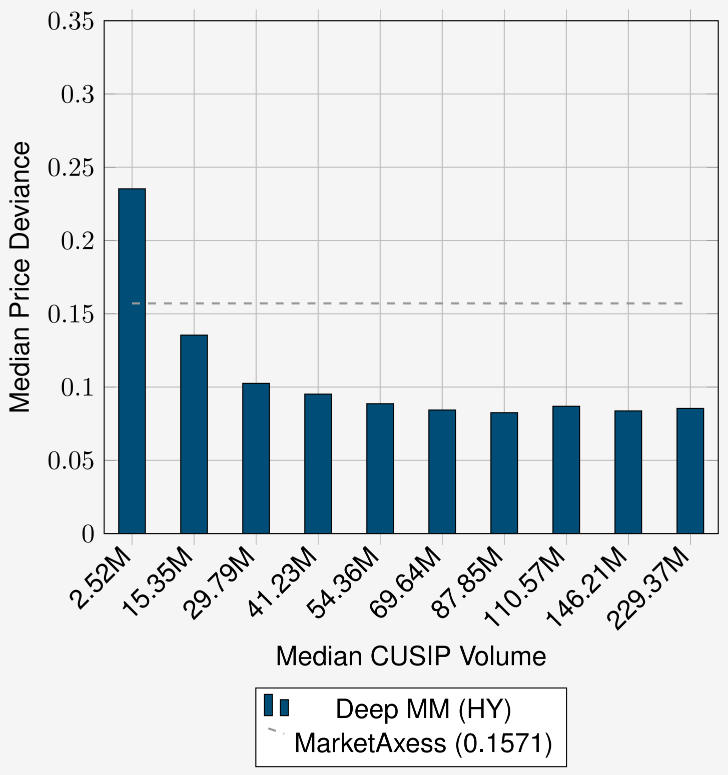

Figure 6: High Yield: Median price deviance vs. median trade volume per CUSIP with MarketAxess benchmark at 0.1571. We don't know what the MarketAxess curve looks like but we include their reported median here.

3.6 IID Residual Evaluation

We calculated the Durbin-Watson (DW) statistic for our model's residuals, focusing on Investment Grade (IG) and High Yield (HY) segments sorted by time, yielding values of 1.64 and 1.74, respectively. An ideal DW value of 2 signifies no first-order autocorrelation and aligns with independent and identically distributed (IID) residuals. With both statistics nearing the 1.8 threshold, the residuals exhibit minimal autocorrelation, suggesting near-IID behavior across these segments. **What the Results Indicate:** The DW statistics of 1.64 (IG)

and 1.74 (HY) reflect the model's ability to capture most systematic variation when sorted by time, with residual autocorrelation sufficiently low to approximate IID conditions for practical applications. **Implications for the Model:**

The near-IID outcomes (DW = 1.64 for IG and 1.74 for HY) highlight the model's effectiveness in modeling broad market patterns across both segments. However, subtle segment-specific factors, such as liquidity or volatility, may remain unaccounted for. While the current performance is robust, enhancing model

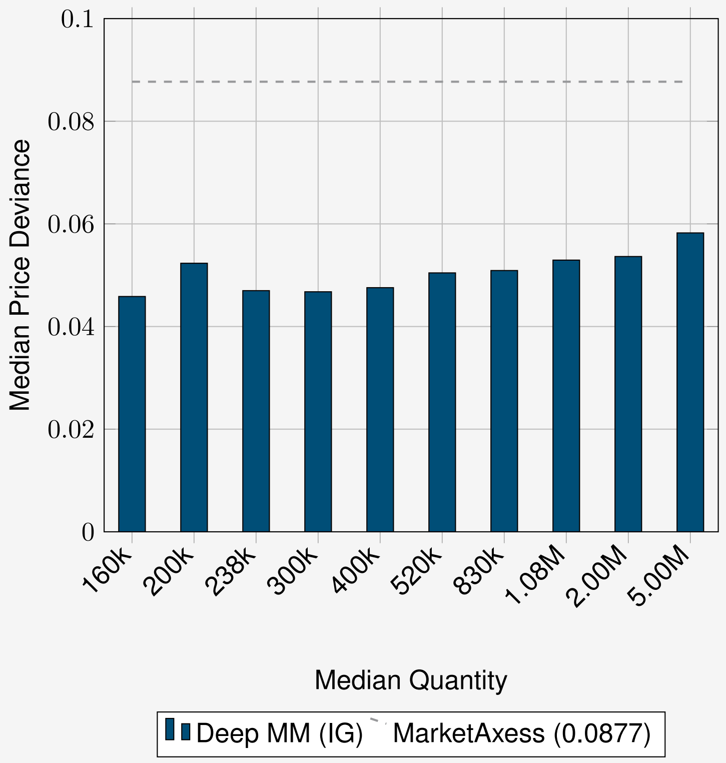

Figure 7: Investment Grade: Median price deviance vs. median trade quantity with MarketAxess benchmark.

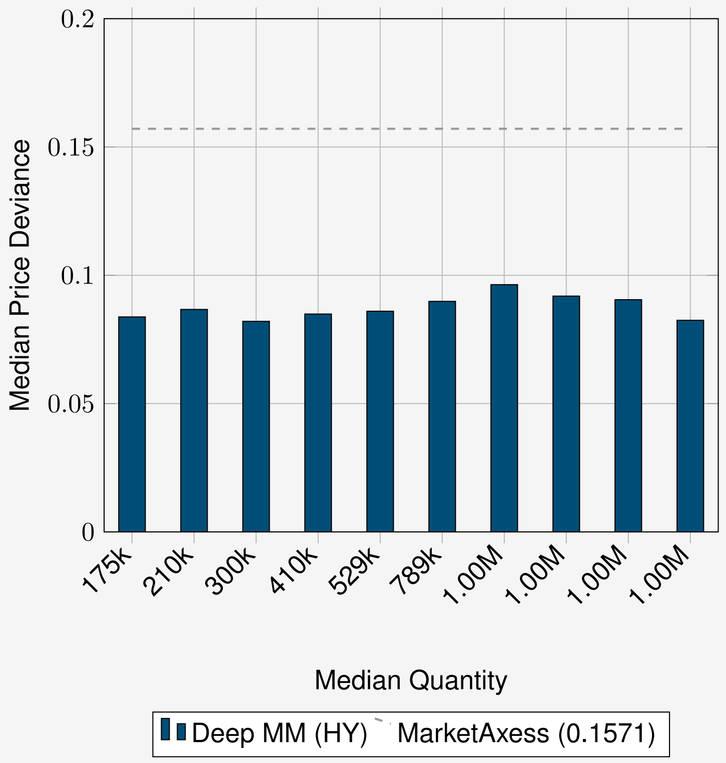

Figure 8: High Yield: Median price deviance vs. median trade quantity with MarketAxess benchmark.

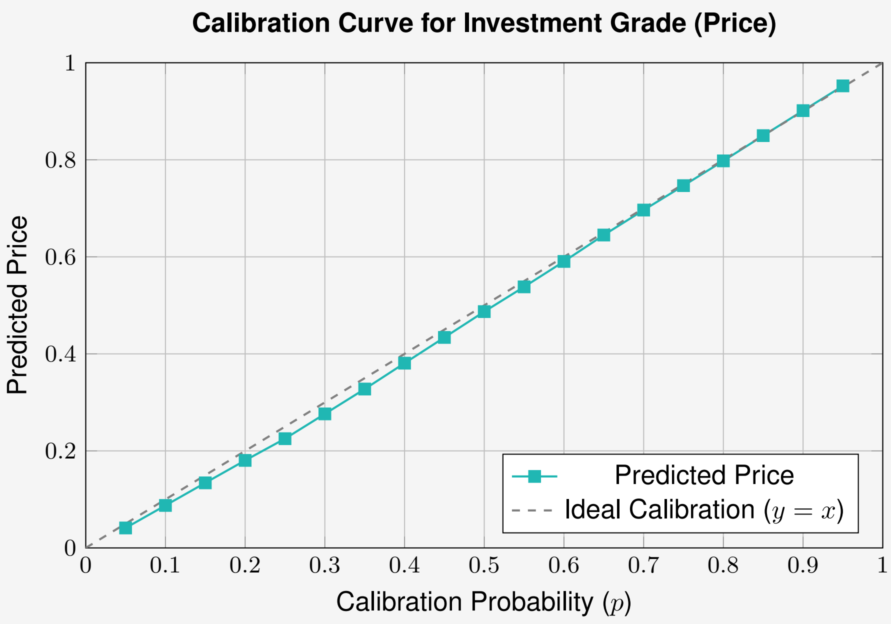

Figure 9: Calibration Curve for Investment Grade (Price) with Ideal Line.

complexity or data quality could further refine predictions, though the existing DW values already provide a solid foundation. In conclusion, the DW statistics of 1.64 (IG) and 1.74 (HY), close to the ideal 2, underscore the model's strong performance in capturing market-wide trends. Minor adjustments could address residual segment-specific dependencies, building on this promising baseline.

4 Comparison and Advantages

At the time of writing, if you compare these results to published numbers of other US corporate credit models, it's clear that Deep MM's model is the most accurate by far, especially when you consider that we have evaluated the model on all trades of size above $150k, whereas a major competitor excluded two-sided trades. There's no good reason to exclude those trades as they are an important part of the trading volume.

However, comparing these white paper numbers is not as robust as doing an apples-to-apples comparison where the prediction from Deep MM and a competitor are compared across the same set of trades, because despite our efforts to use a similar methodology to select trades for this accuracy report that are even perhaps biased in favor of our competitors, there still may be some selection bias from differences in the coverage of our universe or differences

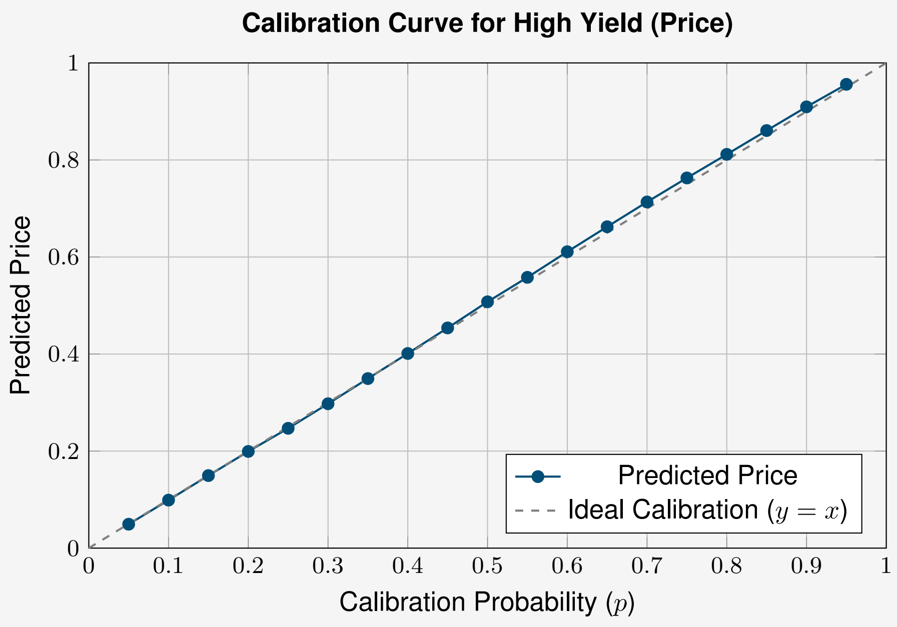

Figure 10: Calibration Curve for High Yield (Price) with Ideal Line.

in the time period used by our competitor versus the time period we selected. We were constrained on the selection of the time period because we wanted to ensure that the time period selected was entirely out-of-sample, meaning that none of the test data evaluated was included in the training set because these data weren't even available at the time the model was trained in July. Also, we don't have direct access to our competitors' data, so we must rely on third parties to make that direct comparison during product evaluations.

Fortunately, we have had multiple customers or potential customers evaluate our data versus our top competitor(s) apples-to-apples on the same set of trades in price space, and found our accuracy to be superior, which roughly matches the results we see here in this report. In section 9 we provide best practices for evaluating our data samples versus that of our competitor.

It is important to emphasize that Deep MM provides a lot of functionality that competitive models do not provide, such as:

- Predicting the probability of different price levels
- Conditioning on quantity, side, and ATS flag
- An innovative trader-friendly UI which makes unlocking value from the model easy
- Automated runs generation, idea generation, portfolio risk analysis, and many others

Deep MM is building the future of foundational deep learning Large Event Models (LEMs), which are applicable for all financial asset classes and many non-financial use cases.

5 Introduction to Gradient Boosted Machines

Gradient Boosted Machines (GBMs) are an ensemble learning method that builds a strong predictive model by sequentially combining multiple weak learners—typically shallow decision trees. The key idea is to iteratively correct the errors of the existing model by fitting new trees to the residuals (i.e., the gradients of the loss function) of the current prediction.

Mathematically, suppose we have a training dataset $\{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents the input features and y_i is the target variable. The process begins with an initial model $F_0(x)$, which is often chosen as a constant that minimizes the loss function:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma),$$

where $L(y, \hat{y})$ is the loss function, such as mean squared error for regression tasks.

For each iteration $m = 1, 2, \ldots, M$, the algorithm proceeds as follows:

1. **Compute the Residuals:** Calculate the negative gradients (residuals) for each training instance:

$$r_{im} = -\left.\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right|_{F(x)=F_{m-1}(x)}.$$

2. **Fit a Weak Learner:** Train a decision tree $h_m(x)$ to predict these residuals r_{im}.

3. **Update the Model:** Update the ensemble by adding the new tree weighted by a learning rate ν:

$$F_m(x) = F_{m-1}(x) + \nu\, h_m(x).$$

After M iterations, the final predictive model is given by:

$$F_M(x) = F_0(x) + \nu \sum_{m=1}^{M} h_m(x).$$

Relevance to CP+: The CP+ pricing engine at MarketAxess is built on this GBM framework. The power of GBMs to capture complex, non-linear relationships in structured, tabular data makes them exceptionally well-suited for the demands of real-time bond pricing. By processing a rich set of inputs—from

TRACE and TraX data to proprietary trading features—CP+ leverages GBMs to generate highly accurate price predictions. The iterative nature of gradient boosting ensures that CP+ continually refines its predictions, thereby reducing pricing errors and improving overall execution quality.

In the following section, we delve into the inherent complexity ceiling of Gradient Boosted Machines, examining the limitations that arise from the structure of their base learners and the diminishing returns associated with adding additional trees.

6 Exploring the Complexity Ceiling of Gradient Boosted Machines

Gradient Boosted Machines (GBMs) have become a cornerstone in many financial applications—including the CP+ pricing engine at MarketAxess—owing to their strong performance on tabular data and their ability to capture nonlinear relationships. However, like many traditional machine learning methods (e.g., individual decision trees, linear models, or even boosted trees in general), GBMs exhibit what is often referred to as a *complexity ceiling*.

Intrinsic Limitations of Base Learners: At the heart of GBMs are decision trees, which, even when shallow, can model non-linearities and interactions among features. However, each tree is inherently limited by its maximum depth. For example, if a tree is constrained to a depth of d, it can capture interactions among at most d features. No matter how many trees are added in the boosting process, the individual trees do not overcome this intrinsic limitation. The ensemble, while additive, remains fundamentally bound by the representational capacity of its constituent trees.

Diminishing Returns in Boosting: Boosting iteratively fits trees to the residual errors of the current model, aiming to reduce bias. Initially, each new tree can capture significant patterns in the data. However, as more trees are added, the marginal gain in reducing the error diminishes. This is partly because the model begins to saturate: the remaining unexplained variance may involve complex interactions or high-order nonlinearities that the limited-depth trees cannot adequately capture. Thus, after a certain number of iterations, additional trees yield only minimal improvement—indicating that the ensemble has reached its complexity ceiling.

Mathematical Perspective: Consider a GBM where each tree T_i is constrained to a maximum depth d. The model is built as an additive series:

$$\hat{y} = \sum_{i=1}^{N} T_i(x),$$

where x represents the input features and N is the number of trees. Each tree T_i is capable of modeling interactions of order up to d. While the boosting process can, in principle, approximate more complex functions by combining many simple models, the overall model is still constrained by the complexity of its base learners. In other words, if the underlying data-generating process requires modeling interactions of order greater than d, the GBM may be inherently incapable of capturing those effects—no matter how many trees are added. This results in a practical complexity ceiling.

Implications for CP+ and Financial Applications: In the context of CP+ and similar pricing engines, the complexity ceiling means that GBMs are exceptionally well-suited for problems where the dominant patterns can be captured by interactions of moderate order. However, if the pricing dynamics involve extremely high-order interactions or subtle nonlinearities, alternative approaches (e.g., deep neural networks) might be able to surpass this ceiling by leveraging more flexible architectures.

Conclusion: While GBMs have proven highly effective—especially on structured, tabular datasets—they are not without limitations. The inherent complexity ceiling, determined by the depth of the decision trees and the diminishing returns of boosting, implies that there is a point beyond which additional model complexity does not translate into significant performance gains. Recognizing this limitation is essential for both model development and the interpretation of performance in real-world applications such as CP+. Balancing the strengths of GBMs with an awareness of their complexity ceiling allows practitioners to deploy these models effectively, while also exploring complementary approaches when necessary.

7 Introduction to Deep Learning

Deep learning is a subfield of machine learning that focuses on building and training artificial neural networks with many layers—often referred to as deep neural networks. Unlike traditional models, which may rely on manually engineered features or shallow representations, deep learning methods learn hierarchical representations directly from raw data.

At its core, a deep neural network consists of multiple layers of interconnected neurons. Each neuron performs a weighted sum of its inputs and applies a nonlinear activation function, such as ReLU or sigmoid, to capture complex patterns. Training the network involves adjusting these weights to minimize a loss function through algorithms like stochastic gradient descent, with backpropagation providing the mechanism to compute gradients across layers.

The key advantages of deep learning include:

- **Hierarchical Feature Learning:** Lower layers capture simple features (e.g., edges or basic patterns), while deeper layers combine these to form more abstract, high-level representations. This hierarchy allows deep networks to model intricate and subtle relationships in the data.

- **Scalability:** Deep learning models can handle very large and high-dimensional datasets, which is critical in applications that require processing massive amounts of information.
- **Flexibility:** Deep neural networks can be applied to a wide range of tasks, from image and speech recognition to natural language processing and, increasingly, to financial modeling.

In financial applications, deep learning has shown promise in capturing complex, nonlinear interactions that traditional models—such as Gradient Boosted Machines (GBMs) or linear models—may struggle to represent. These connectionist methods have the potential to break through the *complexity ceiling* inherent in many traditional approaches by learning high-order feature interactions and more expressive representations.

In subsequent sections, we will explore how connectionist methods, including deep learning, can overcome the limitations of traditional models like GBMs, thereby enabling even more accurate and robust predictive performance in challenging environments.

8 Breaking the Complexity Ceiling with Deep Learning

Traditional machine learning models—such as linear models, decision trees, and even Gradient Boosted Machines (GBMs)—face a *complexity ceiling* due to the inherent limitations in their base learners. For example, GBMs rely on shallow decision trees, which are restricted by their maximum depth and therefore can capture only limited-order interactions among features. No matter how many trees are boosted, the ensemble remains constrained by the expressive power of these individual trees.

Deep learning, by contrast, employs multi-layered neural networks that can learn highly complex, non-linear functions. This approach effectively breaks the complexity ceiling by:

- **Modeling High-Order Interactions:** Deep neural networks stack multiple layers of neurons, allowing them to capture intricate relationships among features that shallow models cannot.
- **Handling High-Dimensional Inputs:** At Deep MM, our pricing model considers over 2,000 features for every bond pricing inference—a volume of input data that traditional models struggle to process without significant performance degradation.

A key aspect of deep learning is the use of a large number of *parameters*. In machine learning, a parameter is a learnable component of the model—typically the weights and biases in a neural network—that determines how input features are transformed at each layer. These parameters are adjusted during training to minimize prediction error. Deep MM's production model employs approximately 80 million parameters, enabling it to learn a highly nuanced mapping from a vast array of inputs to accurate pricing predictions. In stark contrast, competitor models like CP+ are likely to use fewer than 1,000 parameters, which limits their ability to model complex market dynamics.

Mathematical Illustration: Consider a function $f(x)$ that is compositional in nature:

$$f(x) = f_L\Big(f_{L-1}\Big(\cdots f_1(x)\Big)\Big),$$

where each f_i is a relatively simple function (for example, an affine transformation followed by a nonlinearity). A deep neural network with L layers is naturally suited to approximate such functions by learning each f_i in its corresponding layer. In contrast, a shallow model (or a GBM using shallow trees) attempting to approximate $f(x)$ directly may require an exponential number of units or trees to capture the same level of interaction complexity.

More formally, it has been shown that there exist functions computable by a deep network of depth L that would require a shallow network (of depth 2) with exponentially many neurons (in L) to approximate within a fixed error bound [2]. This phenomenon is often summarized as:

$$\text{Parameters}_{\text{shallow}} \sim \exp\left(\Omega(L)\right) \quad \text{versus} \quad \text{Parameters}_{\text{deep}} \sim \mathcal{O}(L),$$

demonstrating the exponential efficiency of deep architectures in representing complex functions.

Additionally, let N be the number of parameters in the network and consider a hypothesis class $\mathcal{H}_N$ of functions parameterized by N weights. Deep networks can achieve a higher effective VC-dimension or Rademacher complexity compared to shallow models with similar N [1]. This increased complexity allows them to better generalize in high-dimensional settings, such as when processing over 2,000 features for bond pricing.

Illustrative Figure: Figure 11 schematically compares the representational power of shallow models versus deep networks. The dashed line represents a *complexity ceiling* inherent to shallow models (including GBMs), beyond which increasing the number of trees or units yields diminishing returns. In contrast, deep networks—with their multiple layers—continue to improve in expressiveness and can model much more complex functions without requiring an exponential increase in parameters.

Conclusion: By leveraging a deep architecture with 80 million parameters, Deep MM's pricing model can capture intricate, high-order interactions among over 2,000 features—far beyond the capacity of traditional models like GBMs, which are limited by a complexity ceiling. This advanced modeling capability directly translates into more accurate pricing predictions, improved risk management, and ultimately, superior risk-adjusted P&L performance in our trading

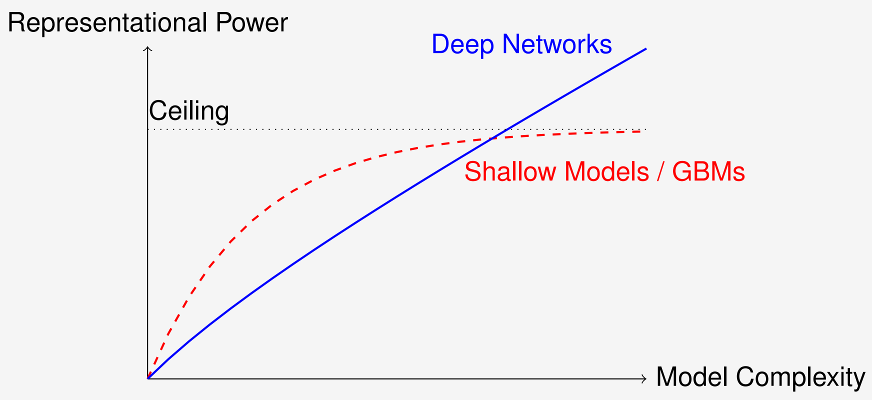

Figure 11: Schematic Comparison of Representational Power: Shallow Models vs. Deep Networks

operations. In order to begin to be useful to traders and other fixed-income professionals who typically have 10-100 trillion synapses (which are roughly analogous to model parameters), we need to incorporate more complexity about the financial system's behavior. After all, it is these professionals who are ultimately entrusted with important pricing decisions, not simplistic formulas or shallow models.

References:

- **Telgarsky, M.** (2016). Benefits of depth in neural networks. *Proceedings of the 29th Annual Conference on Learning Theory (COLT)*.
- **Bartlett, P., Harvey, N., Liaw, C., & Mehrabian, A.** (2017). Nearly-tight VC-dimension and Pseudo-dimension Bounds for Piecewise Linear Neural Networks. *Journal of Machine Learning Research*.

9 Guide to Competitive Evaluation: Best Practices

In the next two subsections, we present the column dictionary for the sample data files we are providing to you for your evaluation purposes, and also describe a couple of unique attributes about the data you should be aware of as you perform the data evaluation.

Then we argue for a set of best practices for evaluating our data versus data from a competitor:

1. Compare the data on the same set of trades
2. Compare the median (and other quantiles relevant to your operations) of the deviance between the signal and the trade

3. Compare in price space
4. Compare using the 50th percentile
5. Evaluate using the other percentiles

9.1 Data Dictionary

This section describes the fields output by the AXOR Corporate Credit Pricing Model. The data is provided in CSV format, where each row corresponds to a trade and includes both observed values and model-derived quantile predictions. Below is a detailed description of each column:

cusip Unique identifier for the bond following the CUSIP standard.

figi Financial Instrument Global Identifier (FIGI) for the bond.

execution_date Timestamp (e.g., in UTC Unix epoch nanoseconds) indicating when the trade was executed.

report_date Timestamp indicating when the trade was reported (again UTC epoch nanoseconds).

maturity Maturity date of the bond, provided as a timestamp (UTC unix epoch nanoseconds).

ats_indicator TRACE indicator showing whether the trade was executed on an Alternative Trading System (ATS). For example, ”N” indicates the trade is *not* ATS.

buy_sell TRACE trade direction indicator: ”B” for Buy and ”S” for Sell.

side TRACE categorical designation of the trading side (e.g., ”C” for client side).

quantity TRACE reported nominal size of the trade (when trade is split into multiple pieces, they are aggregated and the total quantity is reported here).

price TRACE trade price of the bond.

spread The observed spread (in basis points or as a percentage) associated with the trade. Calculated using the YTM field and the benchmark treasury.

treasury_yield_mid Midpoint yield from benchmark treasury yields for the matched benchmark treasury.

yield TRACE reported yield associated with the trade.

ytm Yield to Maturity (YTM) converted from the TRACE price.

label Trade price label used for validation, typically matching the reported trade price for this data set.

5_tile, 10_tile, . . . , 95_tile Deep MM quantile predictions for the bond price. For example, `5_tile` represents the 5th percentile price estimate and `95_tile` represents the 95th percentile price estimate.

ytm_5_tile, ytm_10_tile, . . . , ytm_95_tile Deep MM quantile predictions for the Yield to Maturity, calculated from the model's price quantiles.

spread_5_tile, spread_10_tile, . . . , spread_95_tile Deep MM quantile predictions for the bond spread, calculated from the YTM percentiles summed with the benchmark yield.

Notes:

- The *quantile predictions* (tiles) reflect the model's estimated distribution of outcomes. These allow customers to assess not only the central price prediction but also the uncertainty around that prediction.
- Dates and maturities are provided in high-resolution timestamp formats. Conversion to human-readable dates may be required depending on your use case.
- The **spread** field is particularly important when converting model output from spread space to price space; it interacts with the bond's duration (DV01) to determine the actual dollar impact on pricing.

9.2 Notes on How to Join Deep MM Sample Data Files w/ TRACE Prints etc.

The sample data files provided by your Deep MM contact, (one for high yield and one for investment grade), will contain trades for bonds in Deep MM's current universe. In order to match these trades with your own TRACE print data, there are a couple of unique attributes with our data that you should be aware of:

1. The execution_date and report_date columns are in UTC unix epoch time in nanoseconds
2. Trades with the exact same bond, execution date, buy/sell indicator, side, ATS property have been aggregated into one trade (by summing their quantities). For this reason it is possible for a trade in our data set to have a quantity above the 1 MM limit for high yield bonds, and 5 MM limit for investment grade (enforced by FINRA TRACE). You will need to take this into account when joining Deep MM's data with other TRACE trade data, (or alternatively you can let these consolidated trades be dropped from the comparison).

9.3 Compare on the Same Set of Trades

Avoiding selection bias is critical when comparing competing pricing signals. If the selection criteria differ between the two signals, the evaluation may inadvertently compare two fundamentally different sets of trades, leading to misleading performance metrics.

Let $\mathcal{T}$ denote the complete universe of trades and define the absolute error for a trade as

$$X = \left|\hat{P} - P_{\text{true}}\right|,$$

where $\hat{P}$ is the predicted price and P_{true} is the actual market price.

Assume that Signal A is evaluated on a subset $\mathcal{S}_A \subset \mathcal{T}$ and Signal B on a subset $\mathcal{S}_B \subset \mathcal{T}$, where the selection criteria for $\mathcal{S}_A$ and $\mathcal{S}_B$ differ (i.e., $\mathcal{S}_A \neq \mathcal{S}_B$). Define the indicator functions for these subsets as

$$\mathbf{1}_{\mathcal{S}_A}(t) = \begin{cases} 1, & \text{if } t \in \mathcal{S}_A, \\ 0, & \text{otherwise,} \end{cases} \qquad \mathbf{1}_{\mathcal{S}_B}(t) = \begin{cases} 1, & \text{if } t \in \mathcal{S}_B, \\ 0, & \text{otherwise.} \end{cases}$$

The expected absolute error for each signal is then given by

$$\mu_A = \mathbb{E}\Big[X \mid t \in \mathcal{S}_A\Big] = \frac{\mathbb{E}\Big[X \cdot \mathbf{1}_{\mathcal{S}_A}(t)\Big]}{\mathbb{P}(t \in \mathcal{S}_A)},$$

$$\mu_B = \mathbb{E}\Big[X \mid t \in \mathcal{S}_B\Big] = \frac{\mathbb{E}\Big[X \cdot \mathbf{1}_{\mathcal{S}_B}(t)\Big]}{\mathbb{P}(t \in \mathcal{S}_B)}.$$

Example: Suppose the selection criteria for $\mathcal{S}_A$ excludes low-liquidity trades that typically exhibit higher absolute error, whereas $\mathcal{S}_B$ includes them. Even if both signals have similar intrinsic performance, the distribution of X for Signal A will be skewed toward lower errors compared to Signal B. As a result, the median, 75th, and 95th percentiles computed on $\mathcal{S}_A$ will be lower than those computed on $\mathcal{S}_B$, giving a false impression that Signal A is more accurate, when in fact the difference is driven by the trade selection rather than the quality of the signal.

Mathematical Proof of Bias: Let $\mu = \mathbb{E}[X]$ be the overall expected absolute error over the full universe $\mathcal{T}$. If the selection for each subset were independent of X, then we would have

$$\mathbb{E}\Big[X \cdot \mathbf{1}_{\mathcal{S}_A}(t)\Big] = \mathbb{E}[X] \cdot \mathbb{P}(t \in \mathcal{S}_A),$$

and similarly for $\mathcal{S}_B$, implying $\mu_A = \mu_B = \mu$. However, if the selection criteria are correlated with X, this equality no longer holds. For instance, if high-error trades are less likely to be included in $\mathcal{S}_A$, then

$$\mathbb{E}\Big[X \cdot \mathbf{1}_{\mathcal{S}_A}(t)\Big] < \mathbb{E}[X] \cdot \mathbb{P}(t \in \mathcal{S}_A),$$

which leads to

$$\mu_A < \mu.$$

Conversely, if $\mathcal{S}_B$ tends to include more high-error trades, then

$$\mu_B > \mu.$$

Thus, even if both signals were intrinsically similar, evaluating them on different subsets would result in biased error metrics:

$$\mu_A \neq \mu_B,$$

which does not reflect the true relative performance of the signals but rather the artifact of differing selection criteria.

Conclusion: For an apples-to-apples comparison, both competing pricing signals must be evaluated on the exact same set of trades, ensuring that any differences in performance metrics (such as the median, 75th, and 95th percentiles of X) are solely due to the signal quality and not driven by selection bias. Only then can we be confident that the reported improvements or shortcomings in accuracy truly reflect the underlying models.

9.4 Compare the Median (and Other Quantiles Relevant to Your Operations) of the Deviance Between the Signal and the Trade

An essential component of model evaluation is examining the distribution of errors between the predicted price, $\hat{P}$, and the actual trade price, P_{true}. We define the deviance as

$$D = \left|\hat{P} - P_{\text{true}}\right|.$$

Rather than relying solely on mean or aggregated error metrics, it is critical to compare key quantiles—such as the median (50th percentile), 75th percentile, and 95th percentile—of D. These quantiles provide a more robust and granular view of model performance, especially in environments where tail risk is a key consideration.

Robustness to Outliers: The median is less sensitive to extreme values than the mean, ensuring that the central tendency of the error distribution is representative even if occasional large errors occur. This is particularly important in trading environments where outlier errors can disproportionately affect P&L.

Capturing Tail Risk: In addition to the median, examining the 75th and 95th percentiles of D reveals the behavior of the error distribution in adverse conditions. For example, the 95th percentile indicates that only 5% of trades exhibit an error greater than this value. This tail information is directly related to risk management, as large errors can have significant financial impacts.

Quantile Definitions: Let $F_D(d)$ be the cumulative distribution function (CDF) of D. The q-th percentile, denoted as $Q_q(D)$, is the value d_q that satisfies

$$F_D(d_q) = \frac{q}{100}.$$

Thus, $Q_{50}(D)$, $Q_{75}(D)$, and $Q_{95}(D)$ represent the median, 75th, and 95th percentiles, respectively.

Comparing Competing Signals: When evaluating two competing pricing signals, it is critical to ensure that both are compared on these quantile metrics over the same set of trades. Suppose Signal A and Signal B have quantile values $Q_{50}^A(D)$, $Q_{75}^A(D)$, $Q_{95}^A(D)$ and $Q_{50}^B(D)$, $Q_{75}^B(D)$, $Q_{95}^B(D)$, respectively. If Signal A consistently shows lower quantile values than Signal B, it indicates that Signal A not only has a lower typical error but also exhibits less extreme deviations in adverse conditions. This directly translates to better risk–adjusted performance in trading operations.

Operational Relevance: Different operational settings may demand attention to different parts of the error distribution. For example:

- A market-making desk might prioritize a lower median error to ensure tight pricing for most trades.
- Conversely, a risk management team might focus on the 95th percentile to mitigate the impact of rare, large errors.

By evaluating and comparing the median and other relevant quantiles of D, practitioners gain a comprehensive understanding of model accuracy, balancing both typical performance and worst-case scenarios. This ensures that any improvements in signal quality translate directly into more reliable and robust trading outcomes.

9.5 Compare in Price Space (Rather than Spread Space, Accounting for Duration)

When evaluating pricing signals, it is crucial to work in *price space*, because profit and loss (P&L) is ultimately measured in dollars. Although many signals originate in spread space (e.g. basis points), converting them to dollar terms captures the full financial impact of mispricing. A key driver here is *duration* (or DV01), which quantifies how much a bond's price changes for every one basis point (bp) change in yield or spread. **Why Duration (DV01) Matters** Duration reflects a bond's sensitivity to interest rates or credit spreads:

$$\text{DV01} = -\frac{\partial P}{\partial y} \times \frac{1}{100},$$

where P is the bond's price and y is the yield or spread (in percent). A higher DV01 means a small shift in yield/spread translates into a larger dollar price move. Thus, even if two signals have identical spread errors, the resulting *dollar* mispricing can differ drastically across bonds with different durations. **Illustrative Example: Inversion of Ordering from Spread Space to Price Space** One of the clearest demonstrations that *spread-space accuracy* can fail to reflect *true (dollar-based) performance* arises when comparing signals across multiple bonds with different durations. Consider two bonds, X and Y, and two pricing signals, A and B. Table 2 summarizes the scenario. In **spread space**,

	Bond X	**Bond Y**	**Average**
DV01	0.8	0.2	–
Signal A Spread Error (bp)	1.0	3.0	2.0
Signal B Spread Error (bp)	2.0	1.0	1.5
Spread-Space Leader	$B < A$ (since 1.5 bp < 2.0 bp)		
Signal A \$ Error	$0.8 \times 1.0 = 0.80$	$0.2 \times 3.0 = 0.60$	$(0.80 + 0.60) = 1.40 \Rightarrow 0.70$ avg.
Signal B \$ Error	$0.8 \times 2.0 = 1.60$	$0.2 \times 1.0 = 0.20$	$(1.60 + 0.20) = 1.80 \Rightarrow 0.90$ avg.
Price-Space Leader	$A < B$ (since $\$0.70 < \$0.90$)		

Table 2: Illustration of how one signal can outperform in spread space yet underperform in dollar space due to differing bond durations. Averages assume equal notional traded on each bond; in practice, weight by actual portfolio exposure.

Signal A's average error is 2 bp while Signal B's is 1.5 bp, suggesting B is *better*. But when converting to **price space** to account for each bond's DV01, Signal A actually yields a lower dollar error overall. This "inversion of ordering" shows that focusing exclusively on spread errors can be misleading if you trade multiple bonds with varying durations. **Mathematical Proof: Optimizing Price-Space Accuracy Optimizes P&L** If your cost function for a mispricing is proportional to the squared dollar error, a standard proxy for risk-adjusted P&L, then

$$J \;=\; \mathbb{E}\big[(\epsilon_P)^2\big].$$

Let $\epsilon_s = \hat{s} - s_{\text{true}}$ be the spread error, and let $D(\cdot)$ represent the bond-specific duration or DV01. Then

$$\epsilon_P \;\approx\; D\,(\hat{s} - s_{\text{true}}) \;=\; D\,\epsilon_s.$$

Hence,

$$J \;\approx\; \mathbb{E}\Big[\big(D\,\epsilon_s\big)^2\Big] \;=\; \mathbb{E}[D^2\epsilon_s^2].$$

If spread errors ϵ_s correlate with DV01 (e.g., higher errors on long-duration bonds), or if the portfolio has uneven exposure, then $\mathbb{E}[D^2\epsilon_s^2] = \mathbb{E}[D^2]\mathbb{E}[\epsilon_s^2] + (D^2, \epsilon_s^2)$. Non-zero covariance means minimizing spread MSE alone may not optimize dollar P&L—hence, direct price-space comparison is essential. By comparing signals in *price space*, you minimize

$$J \;=\; \mathbb{E}\big[(\epsilon_P)^2\big]$$

directly, ensuring your model improvements align with actual dollar outcomes.

9.5.1 Conclusion

Evaluating signals in *price space* incorporates each bond's unique duration and thereby captures the genuine impact of mispricing on P&L. Even if a signal appears better in basis points, that advantage may vanish—or even invert—once duration is factored in. Hence, price-space comparisons provide a more faithful view of trading performance, ensuring that any improvement in the signal translates directly into tangible (and quantifiable) financial gains.

9.6 Compare Using the 50th Percentile

Our model employs quantile regression to estimate a full distribution of potential prices—from the 5th to the 95th percentile in 5% increments. The **50th percentile** in this framework represents the central price estimate at which there is a 50% chance that the true market price will be above or below. By design, this median forecast is optimal for minimizing *absolute* mispricing in typical day-to-day trades, even if the error distribution is heavily skewed or partitioned by side (bid vs. offer).

1. The Goal: Minimizing Typical Dollar Deviations In many trading contexts, the key driver of profit and loss (P&L) is how close your quoted price is to the fair market price in *absolute dollar terms*. Formally, if $\hat{P}_{50}$ is your 50th-percentile prediction and P_{true} is the actual market price, you want to keep

$$\left|\hat{P}_{50} - P_{\text{true}}\right|$$

as small as possible on most trades. An L1 metric (absolute error) directly captures these day-to-day transaction costs, irrespective of distribution symmetry or side conditioning.

The Median Minimizes L1 Error, Regardless of Symmetry. A well-known result in statistics is that the median of a random variable minimizes expected absolute deviation:

$$\arg\min_{m} \mathbb{E}\big[|X - m|\big] \;=\; \text{median}(X).$$

Hence, when your cost function or penalty for mispricing is proportional to $\left|\hat{P} - P_{\text{true}}\right|$, a 50th-percentile forecast naturally aligns with minimizing L1 error. This property holds regardless of whether the underlying error distribution is symmetric, skewed, or even multimodal.

2. Connection to Squared Error (L2) Under General Distributions One might initially think that minimizing squared error $\mathbb{E}[\epsilon^2]$ implies you should target the *mean*. However, trading desks often rely on metrics or loss structures beyond purely minimizing $\mathbb{E}[\epsilon^2]$. For instance:

- Some trading desks apply a *piecewise linear* penalty on deviations from fair price.
- Others care most about *frequency* of being off-market by more than a certain threshold, rather than rare large outliers.

In either situation, focusing on *typical* mispricing (i.e., the 50th percentile) can provide superior day-to-day outcomes, even if the distribution is significantly skewed.

Why This Still Improves $\mathbb{E}[\epsilon^2]$ in Practice. Even in heavily skewed distributions, shrinking the *median* error (the bulk of "moderately off" trades) tends to reduce practical measures of overall variance. Moreover, catastrophic outliers often fall under separate hedging or risk-control frameworks, so curtailing typical deviations still has a tangible impact on improving aggregated P&L.

3. Mathematical Example Without Symmetry Suppose $\epsilon = \hat{P} - P_{\text{true}}$ is mostly nonnegative (right-skewed). Despite this asymmetry, the median

$$\hat{P}_{50} \;=\; \arg\min_{p} \mathbb{E}\big[|\epsilon - p|\big]$$

is still the best L1 estimator. You do *not* need ϵ to be zero-mean or symmetric for the median-based approach to be optimal at reducing aggregate absolute mispricing.

4. Implications for Bid/Offer Conditioning In real bond markets, conditioning on side (bid vs. offer) leads to potentially different error distributions:

- *Separate Distributions:* Each side has its own shape. Minimizing absolute error on bids uses the median of the "bid distribution," whereas minimizing absolute error on offers uses the median of the "offer distribution."
- *Combined Strategy:* A model can jointly learn the 50th percentile on both bid and offer sides, thereby reducing typical deviation for *all* trades.

5. Why This Improves Real-World P&L

- **Fewer Missed Opportunities:** Being close in absolute dollars reduces negative slippage and missed executions.
- **Reduced Transaction Costs:** Consistently quoting near the fair price in L1 terms leads to cumulatively higher daily P&L.
- **Better Risk Management:** Minimizing median error lowers the fraction of trades that deviate beyond acceptable thresholds—elevating stability and reducing risk.

6. Summary Without Assuming Symmetry

- *Median Minimizes L1 Error:* A fundamental result, independent of distribution shape.

- *Absolute Error Dominates Typical P&L:* Bond desks often care about how close their quotes are on *most* trades, not only rare extremes.
- *Skewed or Conditioned by Side:* Even with strong skew or separate bid/offer distributions, the median remains the L1 minimizer.

Hence, employing a 50th-percentile (median) forecast in a quantile regression framework remains a highly effective strategy to reduce day-to-day, per-trade mispricing in real-world market-making, ultimately boosting *risk-adjusted* P&L without requiring any symmetry assumptions.

9.7 Evaluate using the Other Percentiles

Beyond the 50th percentile, our model produces a full quantile distribution—from the 5th to the 95th percentile—that offers critical insights into the uncertainty and reliability of our price forecasts. This section explains how to leverage these additional quantile estimates to assess calibration and understand how uncertainty correlates with the error of the central (50th percentile) prediction.

Calibration Curves: A calibration curve measures whether the predicted quantiles accurately represent the observed outcomes. For example, if the model predicts a 95th percentile price, we expect that roughly 95% of the true prices fall below this prediction. To construct a calibration curve, plot the nominal quantile levels (5%, 10%, ..., 95%) on the horizontal axis against the empirical frequencies (i.e., the percentage of trades for which the true price is below the predicted quantile) on the vertical axis. In an ideally calibrated model, the points will lie along the 45-degree line. Deviations from this line indicate systematic overestimation or underestimation of risk. This curve serves as an essential diagnostic tool to ensure that the uncertainty quantified by the model is both meaningful and actionable.

Correlation Between Uncertainty and 50th Percentile Deviance: Another important aspect of evaluating the full quantile output is to investigate how the spread of the quantiles correlates with the accuracy of the 50th percentile prediction. A common measure of forecast uncertainty is the width of the prediction interval. For instance, define the uncertainty measure as:

$$U = \hat{P}_{95} - \hat{P}_5,$$

where $\hat{P}_{95}$ and $\hat{P}_5$ are the 95th and 5th percentile price predictions, respectively.

Let the error of the central (50th percentile) forecast be:

$$\epsilon_{50} = \hat{P}_{50} - P_{\text{true}}.$$

Empirically, we often observe that when U is large—indicating high uncertainty—the absolute error $|\epsilon_{50}|$ tends to be larger as well. By quantifying the correlation between U and $|\epsilon_{50}|$, we can assess whether the model's uncertainty estimates are predictive of actual forecast error. A strong positive correlation would imply that when the model is less confident (wider prediction intervals), the 50th percentile prediction is also less accurate. This information can be used to adjust trading strategies and risk controls in real time.

Operational Implications: Utilizing the full quantile distribution confers several benefits:

- **Enhanced Risk Management:** Traders can adjust their positions based on the entire range of potential outcomes, not just the central forecast.
- **Informed Decision-Making:** By monitoring the prediction interval width and its correlation with central forecast error, trading desks can be alerted to periods of heightened uncertainty.
- **Continuous Calibration:** Regularly reviewing calibration curves allows for the identification and correction of systematic biases in the model, ensuring that all quantile forecasts remain reliable.

In summary, evaluating the model using the other percentiles—via calibration curves and the analysis of prediction interval width—provides a comprehensive picture of forecast reliability. This approach ensures that our model not only delivers accurate central price estimates but also accurately quantifies uncertainty, which is essential for managing trading risk and optimizing overall performance.

10 Reproducing Historical Inferences Using the Live API

The historical inference data files provided by Deep MM contain model predictions conditioned on specific trade properties, such as side (buy or sell), size, and ATS (Alternative Trading System) flag. These conditional predictions contribute significantly to the model's accuracy in offline evaluations. To reproduce or approximate similar inferences using the live API, there are two primary approaches. For implementation details, including example scripts, refer to the GitHub repository at github.com/deepmarketmaking/api. Work with a Deep MM administrator to enable API access and configure credentials.

10.1 Option 1: On-Demand Queries for Specific RFQs

The simplest method is to query the live server directly upon receiving an RFQ (Request for Quote). Provide the bond identifier (e.g., FIGI), along with the desired side, size, and ATS flag as parameters in the API call. The server is optimized for high throughput, typically returning results in 15 seconds or less. For applications requiring lower latency, discuss customization options with Deep MM support. If security or data exfiltration concerns arise, the model can be deployed in your organization's data center or cloud environment.

This approach yields exact conditional predictions without interpolation, yielding a very close match with historical data files.

10.2 Option 2: Subscribe to Variations and Interpolate

Alternatively, subscribe to a comprehensive set of prediction variations across sides (buy/sell), sizes (e.g., 10k, 100k, 500k, 1M, etc.), and ATS flags (yes/no). This can be automated using scripts like `subscribe_price_variations.py` from the GitHub repository, which takes parameters such as AWS region, application ID, user credentials, OpenFIGI API key, and a list of bond identifiers (e.g., ISINs).

Once subscribed, receive the inferences throughout the trading day. To approximate a conditional prediction for a specific RFQ or TRACE trade (while excluding trades below 10,000 in size and dealer-to-dealer trades):

1. Identify the most recent inferences before the RFQ or trade execution timestamp that match the side and ATS status.
2. Round the RFQ/trade size down to the nearest subscribed size variation and up to the next one.
3. Perform linear interpolation between these two inferences. For example, if the trade size is 25,000 and inferences are available for 10,000 and 100,000:

   $$x = \frac{25,000 - 10,000}{100,000 - 10,000}, \quad \hat{P}_{\text{interp}} = x \cdot \hat{P}_{100,000} + (1 - x) \cdot \hat{P}_{10,000},$$

   where $\hat{P}$ represents the predicted price (or other output) at the specified quantile.

This method allows for pre-fetching data, enabling fast run-time approximations where you want to avoid sending your RFQ inquiry to Deep MM's servers, though it may introduce minor interpolation errors compared to on-demand queries, and the other downside is this option requires subscribing to a large amount of inferences, of which only a subset are needed for your actual trading activity.

11 Conclusion

Deep MM is at the forefront of deep learning-powered credit trading, providing accurate, efficient, and robust models for pricing and spread prediction. The innovative approach and extensive dataset enable traders to make well-informed decisions, reducing execution risks and optimizing portfolio performance.

Contact Information

Deep Market Making, Inc.
Book a Demo
Visit Website
Email: nathan@deepmm.com, sales@deepmm.com

References

[1] Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

[2] Matus Telgarsky. Benefits of depth in neural networks. In *Conference on Learning Theory (COLT)*, 2016.