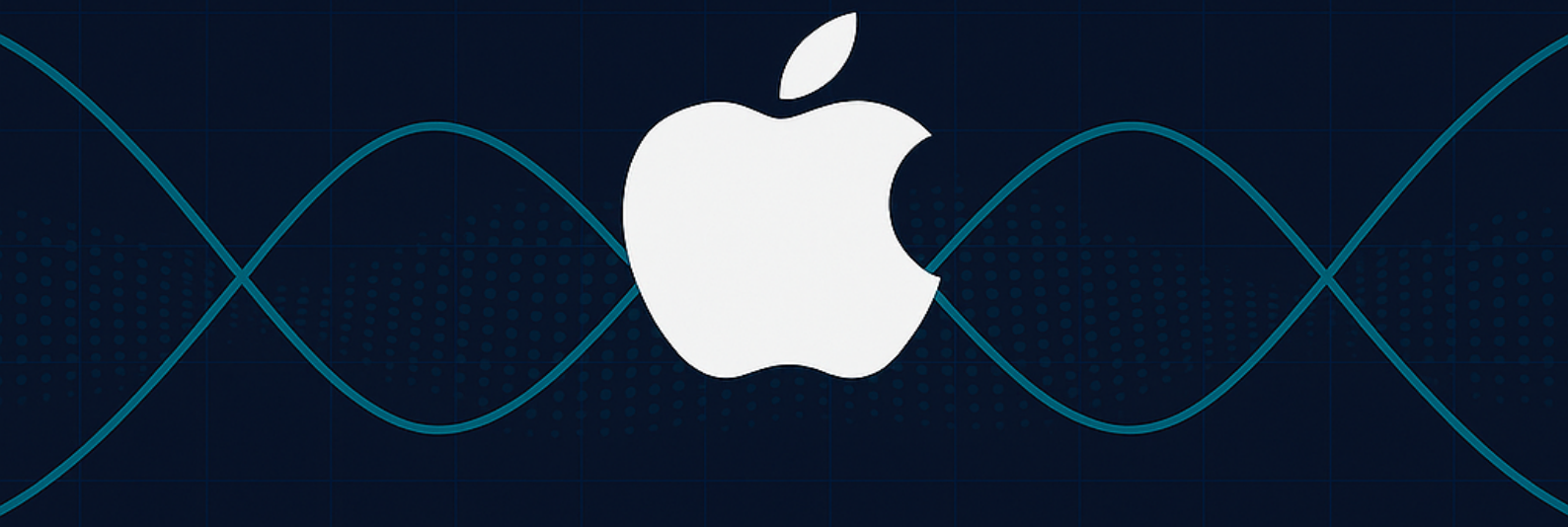
Machine Learning for Apple Option Pricing



ESILV 2025-2026

Sacha Keredan
Lucas Soares
Martin Partiot

Contents

1 Introduction

Financial derivatives represent one of the most intellectually challenging domains in quantitative finance. Among these instruments, options occupy a central place: they provide investors with asymmetric payoff profiles, enabling sophisticated risk management and speculative strategies. The fundamental question that has driven decades of research is deceptively simple: *how much is an option worth?*

The seminal work of Black and Scholes [1] in 1973 provided the first closed-form solution to this problem, under a set of idealized assumptions. Their formula, elegant in its mathematical simplicity, assumes that volatility remains constant over time, that markets are frictionless, and that asset returns follow a geometric Brownian motion. Real markets, however, systematically violate these assumptions. The phenomenon known as the *volatility smile* [3] where implied volatility varies with strike price is perhaps the most visible manifestation of this discrepancy.

This project takes a different approach. Rather than imposing a parametric structure on option prices, we let the data speak for itself. Using machine learning techniques, we build predictive models that learn the complex, non-linear relationship between option characteristics and their market prices. This data-driven methodology follows the pioneering work of Hutchinson et al. [5], who demonstrated that neural networks could approximate option pricing functions without explicit model assumptions.

1.1 Business Context and Motivation

This project is situated within the field of **Financial Engineering**, specifically addressing the practical challenge of rapid and accurate option valuation. The target users for such models are diverse:

- **Market makers** who must quote bid-ask prices on thousands of option contracts in real-time, requiring fast pricing that captures empirical regularities like the volatility smile without manual recalibration.
- **Risk managers** who need to value large option portfolios for Value-at-Risk calculations, where pricing errors compound across positions.
- **Algorithmic trading systems** that identify mispriced options by comparing model predictions to market quotes.

The practical significance of our results can be quantified as follows: our best model achieves a Mean Absolute Error of \$5.97, compared to \$16.31 for a linear baseline. For an institutional desk trading 10,000 option contracts daily, this 63% reduction in pricing error translates directly to improved P&L and reduced hedging costs. Furthermore, while the Black-Scholes model requires constant recalibration of the volatility surface, our ML approach learns these patterns automatically from historical data.

1.2 Report Structure

The remainder of this report is organized as follows. Section 2 provides the mathematical foundations of option pricing, establishing the theoretical context for our empirical work. Section 3 describes the dataset and the preprocessing steps required to prepare it

for machine learning. Section 4 presents our modeling approach, including hyperparameter optimization and experimental results. Section 5 offers a critical analysis of model performance and limitations. Section 6 concludes.

2 Mathematical Foundations

Before diving into the empirical analysis, we establish the theoretical framework that underlies option pricing. This section serves two purposes: it introduces the notation used throughout the report, and it explains why the Black-Scholes model, despite its limitations, remains a useful benchmark.

2.1 Option Contracts and Payoff Structure

Definition 1 (European Call Option)**.** *A European call option gives its holder the right, but not the obligation, to purchase an underlying asset at a predetermined price K (the strike) on a specified date T (the expiration). The payoff at expiration is:*

$$\textit{Payoff} = \max(S_T - K, 0)$$

where S_T denotes the asset price at time T.

This payoff structure is fundamentally asymmetric: the holder benefits from upside movements in the asset price while limiting downside exposure to the premium paid. This asymmetry is what makes options valuable and what makes them difficult to price.

The value of a call option before expiration consists of two components:

$$\underbrace{C}_{\text{Option Price}} = \underbrace{\max(S - K, 0)}_{\text{Intrinsic Value}} + \underbrace{\text{Time Value}}_{\text{Optionality Premium}} \tag{1}$$

The intrinsic value represents the immediate exercise value, while the time value captures the probability that the option will become more valuable before expiration. Understanding this decomposition is crucial for interpreting model predictions.

2.2 The Black-Scholes Framework

The Black-Scholes model [1], extended by Merton [2], derives the fair value of a European call option under the following assumptions: the underlying asset follows a geometric Brownian motion with constant volatility σ, markets are complete and frictionless, and the risk-free rate r is constant. Under these conditions, the option price is given by:

$$C = S_0\,\Phi(d_1) - Ke^{-rT}\Phi(d_2) \tag{2}$$

where $\Phi(\cdot)$ denotes the cumulative distribution function of the standard normal distribution, and:

$$d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}} \tag{3}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{4}$$

The term $\ln(S_0/K)$ appearing in these expressions is called the *log-moneyness*. It measures how far the current asset price is from the strike, on a logarithmic scale. Options with log-moneyness near zero are called *at-the-money* (ATM); positive values indicate *in-the-money* (ITM) options, and negative values indicate *out-of-the-money* (OTM) options.

2.3 Implied Volatility and the Volatility Smile

In practice, the Black-Scholes formula is used "in reverse": given the market price of an option, one can solve Equation (2) numerically to find the volatility σ that makes the formula match the observed price. This quantity is called the *implied volatility* (IV).

If the Black-Scholes assumptions held exactly, implied volatility would be identical across all strikes and maturities for a given underlying asset. Empirically, this is never the case. Instead, implied volatility exhibits a characteristic U-shaped pattern when plotted against moneyness a phenomenon known as the *volatility smile* [3, 4]. Out-of-the-money options, particularly puts, trade at higher implied volatilities than at-the-money options. This reflects the market's assessment of tail risk: extreme price movements are more likely than the log-normal distribution suggests.

The existence of the volatility smile is a direct challenge to the Black-Scholes framework. Alternative models such as the Heston stochastic volatility model [6] address this limitation by allowing volatility itself to follow a random process, but at the cost of additional parameters that must be calibrated. This motivates the use of data-driven approaches that can capture these empirical regularities without imposing restrictive parametric assumptions.

2.4 Why Machine Learning?

The option pricing problem can be formulated as a regression task: given a set of observable features (stock price, strike, time to expiration, implied volatility), predict the option's market price. Traditional approaches rely on closed-form formulas or numerical methods tied to specific stochastic models. Machine learning offers an alternative: learn the pricing function directly from data.

This approach has several advantages [9, 10]. First, it is model-agnostic: we do not need to specify the dynamics of the underlying asset. Second, it can capture complex interactions between features that would be difficult to specify analytically. Third, it scales naturally to large datasets.

The main challenge is avoiding *data leakage* using information that would not be available at prediction time. In our context, this means excluding the option Greeks (Delta, Gamma, Vega, Theta, Rho) from the feature set. These quantities are computed from the option price itself using a pricing model, so including them would create circular logic.

3 Data and Preprocessing

Our analysis is based on a dataset of Apple (AAPL) call option quotes, containing over one million observations. This section describes the structure of the data, the cleaning procedures applied, and the feature engineering steps that prepare the dataset for machine learning.

3.1 Dataset Description

Data Source. The dataset was obtained from Kaggle[1] (AAPL Options Data 2016-2020), originally sourced from the Chicago Board Options Exchange (CBOE) via market data

[1] https://www.kaggle.com/datasets/kylegraupe/aapl-options-data-2016-2020

providers. It contains end-of-day quotes for Apple (AAPL) call and put options, covering the period from 2016 to 2020. This timeframe captures diverse market conditions including the 2018 volatility spike, the COVID-19 crash (March 2020), and various earnings announcements.

The raw dataset contains 1,015,352 option quotes with 33 variables. Each row represents a snapshot of a specific option contract at a given point in time. The key variables are:

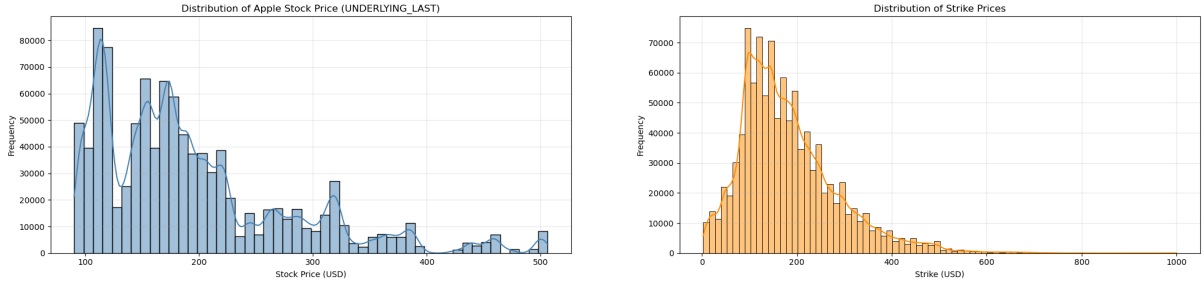
Variable	**Description**
`UNDERLYING_LAST`	Current price of Apple stock (S)
`STRIKE`	Strike price of the option (K)
`DTE`	Days to expiration ($T \times 365$)
`C_IV`	Implied volatility of the call option (σ)
`C_LAST`	Last traded price of the call (target variable)
`C_DELTA`, `C_GAMMA`, etc.	Option Greeks (excluded from features)

Table 1: Key variables in the AAPL options dataset.

The data spans multiple years, covering periods of both low and high market volatility. Apple's stock price ranges from \$90 to \$506 across the sample, providing exposure to diverse market conditions.

3.2 Exploratory Data Analysis

Before applying any transformation, we examine the raw distributions of key variables to understand the structure of the data and identify potential issues.



(a) Stock price distribution

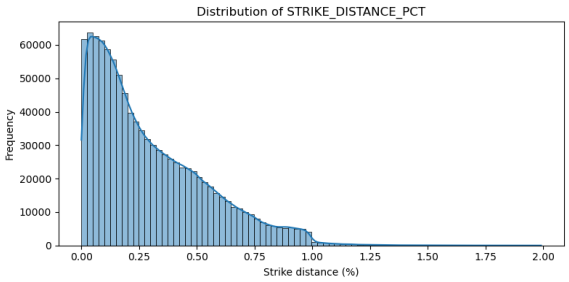
(b) Strike price distribution

Figure 1: Distribution of underlying stock price and strike prices in the raw dataset. The multimodal structure of the stock price reflects different market periods covered by the data. Strike prices are concentrated between \$100–\$250, with a peak around \$150.

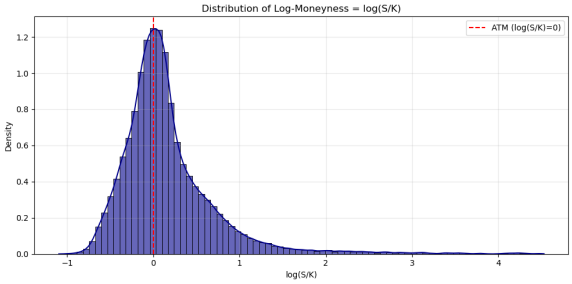
The stock price distribution reveals that our dataset captures multiple market regimes: distinct peaks at \$125, \$175, and \$225 correspond to different periods in Apple's trading history. This diversity is valuable for model training, as it exposes the algorithm to varied market conditions.

A critical observation concerns the concentration of trading activity. Figure 2 shows that most options are struck near the current stock price: over 60% of observations fall within 10% of the at-the-money level. This exponential decay in trading volume as we

move away from ATM reflects market reality liquidity is concentrated where hedging demand is highest.



(a) Strike distance from spot (%)



(b) Log-moneyness distribution

Figure 2: Concentration of options near the money. Left: strike distance as percentage of spot price shows exponential decay. Right: log-moneyness is centered near zero with slight negative skew.

Perhaps the most important empirical finding from our exploration is the volatility smile, shown in Figure 3. This U-shaped relationship between implied volatility and moneyness directly contradicts the Black-Scholes assumption of constant volatility. Out-of-the-money options trade at implied volatilities of 40% or higher, while at-the-money options cluster around 20–25%. The color gradient reveals that short-dated options (purple) exhibit steeper smiles than long-dated ones (yellow), a phenomenon known as the *term structure of volatility*.

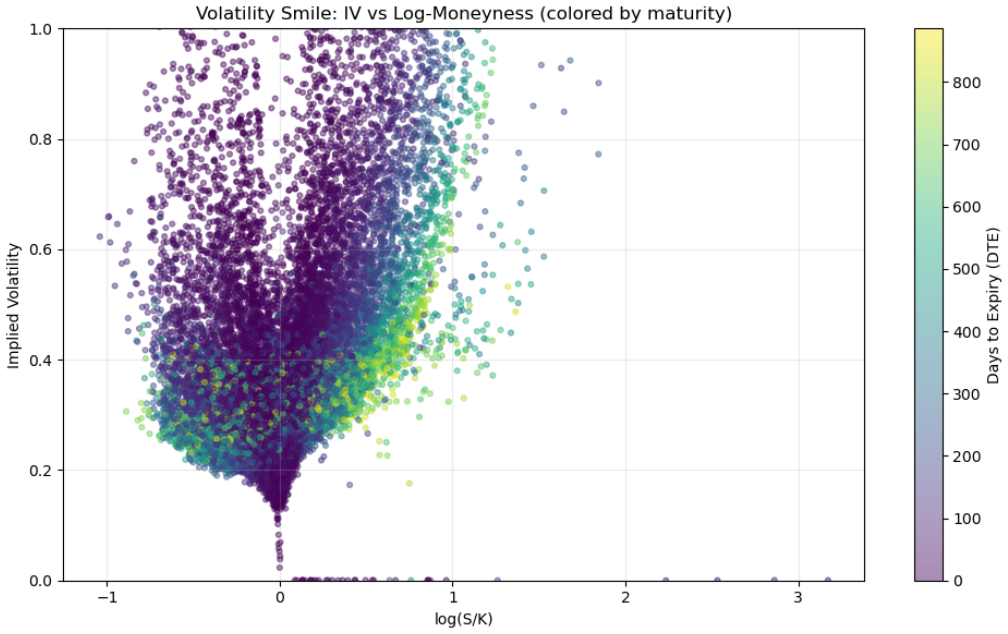


Figure 3: The volatility smile: implied volatility vs. log-moneyness, colored by days to expiration. The characteristic U-shape demonstrates that the Black-Scholes assumption of constant volatility is violated in practice. Short-dated options (purple) show more pronounced smiles than long-dated ones (yellow).

Finally, we examine the target variable distribution. Call prices span from near zero (deep OTM options) to over $400 (deep ITM options), creating an extremely right-skewed

distribution. Figure 4 shows both the raw and log-transformed distributions. The log transformation produces a bimodal pattern: a sharp peak near $\ln(C) \approx -5$ represents near-worthless OTM options, while a broader distribution centered around $\ln(C) \approx 3.5$ captures liquid ATM and ITM options.

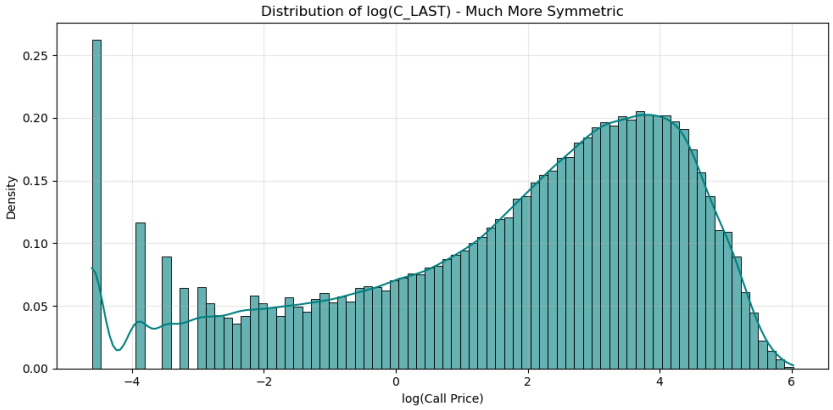


Figure 4: Distribution of log-transformed call prices. The bimodal structure reflects two regimes: near-zero OTM options (left peak) and substantial ATM/ITM options (right distribution).

3.3 Data Cleaning

We apply a series of filters to remove invalid or unrealistic observations:

Structural constraints. We require that stock price, strike, and days to expiration be strictly positive. Options with zero or negative implied volatility are removed, as these cannot be priced meaningfully. We also exclude rows where the expiration date precedes the quote date (stale data).

Missing values. Approximately 7% of observations have missing implied volatility. Rather than imputing these values which would introduce artificial patterns we drop the affected rows. This is justified because implied volatility is essential for option pricing, and missing values typically indicate illiquid or stale quotes.

Outlier removal. We filter extreme implied volatilities (IV > 3, i.e., 300%) which represent the 99.9th percentile. We also remove options with log-moneyness exceeding ± 4 in absolute value, corresponding to strikes more than 50 times away from the current stock price. These deep out-of-the-money options are extremely illiquid and prone to data errors.

After cleaning, 930,429 observations remain (91.6% of the original dataset).

3.4 Feature Engineering

We construct several derived features motivated by option pricing theory:

Log-moneyness. Following standard practice in quantitative finance, we define:

$$\text{Log-Moneyness} = \ln\left(\frac{S}{K}\right)$$

This measure is centered at zero for at-the-money options, positive for in-the-money, and negative for out-of-the-money. It captures the relative position of the strike with respect to the current stock price in a scale-invariant manner.

Log-transformed time. The raw days-to-expiration variable is heavily right-skewed, with most options expiring within 30 days. We apply a log transformation: $\ln(\text{DTE} + 1)$. This spreads the distribution more evenly and allows the model to distinguish between short-term and long-term options more effectively.

Figure 5 validates the effect of these transformations. The raw call price distribution is extremely right-skewed, making it difficult for models to learn patterns across the full price range. After log transformation, the distribution becomes more symmetric (though still bimodal). Similarly, the raw DTE distribution shows extreme concentration at short maturities; the log transformation spreads values more evenly across the range.

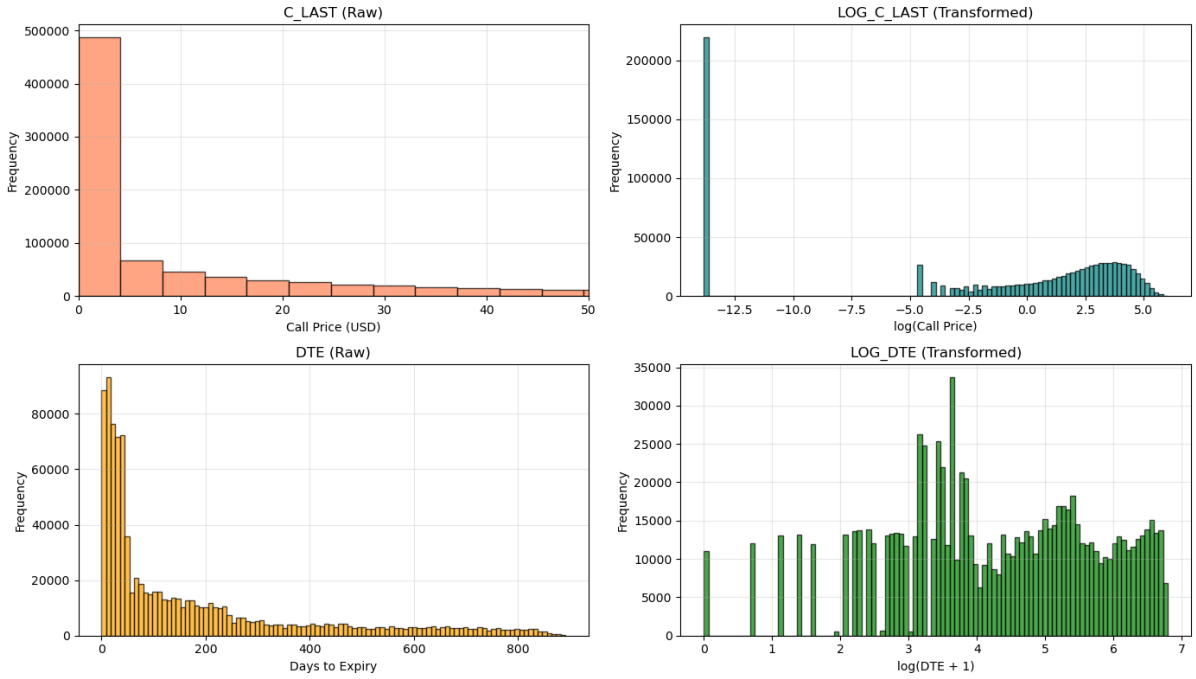


Figure 5: Validation of log transformations. Top row: call prices before (left) and after (right) transformation. Bottom row: days to expiration before and after. The transformations reduce skewness and stabilize variance, improving model learning.

Maturity buckets. For stratified sampling, we categorize options into five maturity buckets: 0–30 days, 30–90 days, 90–180 days, 180–365 days, and over one year. Figure 6 shows the distribution across these categories. Short-term options (0–30 days) dominate the dataset with nearly 300,000 observations, reflecting trader preference for high-gamma, liquid contracts. This imbalance motivates our stratified splitting strategy.

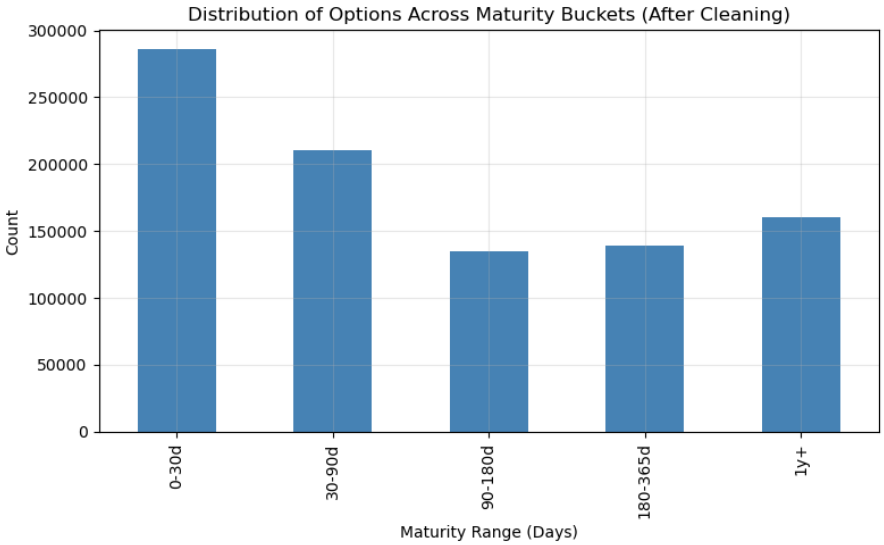


Figure 6: Distribution of options across maturity buckets. Short-term options dominate, but all categories have substantial representation.

3.5 Correlation Analysis

Before finalizing the feature set, we examine correlations between variables to identify potential multicollinearity and redundant features. Figure 7 presents the correlation matrix for all numerical variables.

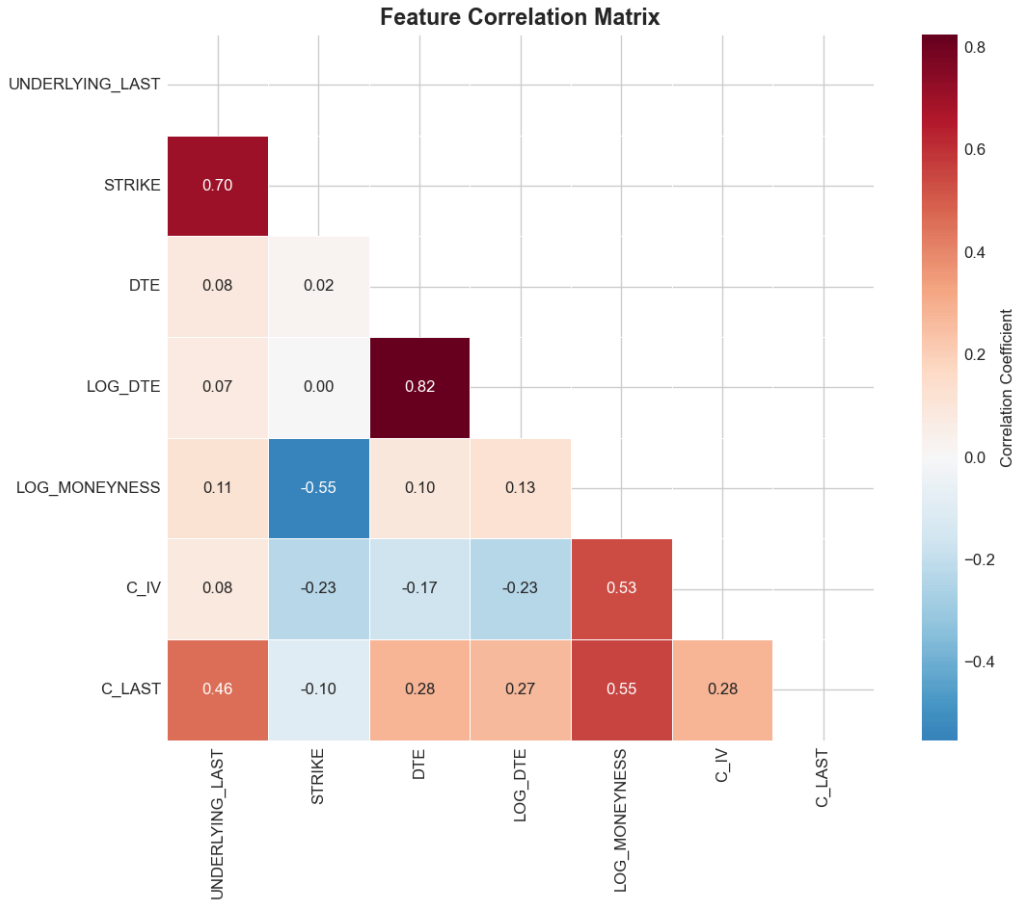


Figure 7: Correlation heatmap of features and target variable. Strong positive correlation between UNDERLYING_LAST and STRIKE reflects the concentration of trading near ATM options. Log-moneyness shows moderate correlation with the target, confirming its predictive value.

Key observations from the correlation analysis:

- `UNDERLYING_LAST` and `STRIKE` exhibit strong positive correlation ($\rho \approx 0.85$), which is expected since most traded options are near-the-money.
- `LOG_MONEYNESS` shows moderate positive correlation with `C_LAST` ($\rho \approx 0.55$), confirming its importance as a predictor.
- `C_IV` is weakly correlated with other features, suggesting it provides independent information.
- The low correlation between `DTE` and other features justifies its inclusion as a distinct predictor.

3.6 Dimensionality Reduction Analysis

To understand the intrinsic dimensionality of our feature space and identify potential redundancies, we perform Principal Component Analysis (PCA) on the standardized features. Figure 8 shows the variance explained by each principal component.

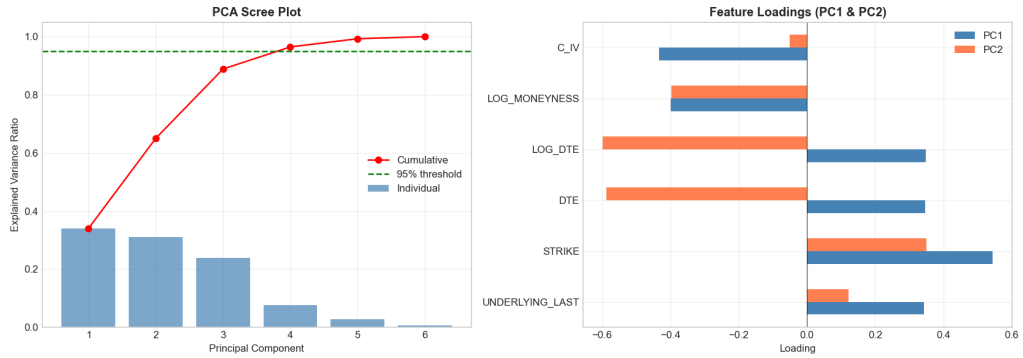


Figure 8: PCA analysis of the feature space. Left: Scree plot showing variance explained by each component. Right: Cumulative variance explained. The first two components capture approximately 65% of total variance; five components explain over 95%.

The PCA results reveal that while the first two principal components capture the majority of variance (dominated by the price-strike relationship), all six original features contribute meaningfully to the predictive task. This justifies retaining the full feature set rather than applying dimensionality reduction. The first principal component loads heavily on `UNDERLYING_LAST` and `STRIKE`, while the second component emphasizes `DTE` and `C_IV`.

3.7 Final Feature Set

Based on our correlation and PCA analyses, the features used for modeling are:

1. `UNDERLYING_LAST`: Stock price
2. `STRIKE`: Option strike
3. `DTE`: Days to expiration
4. `LOG_DTE`: Log-transformed maturity
5. `LOG_MONEYNESS`: Log of stock-to-strike ratio
6. `C_IV`: Implied volatility

The target variable is `C_LAST`, the last traded price of the call option.

Critically, we exclude all Greeks from the feature set. While these variables are highly predictive of option prices, they are themselves derived from prices using a theoretical model. Including them would constitute data leakage and produce misleadingly optimistic performance estimates.

To understand why this exclusion is necessary, consider Figure 9. Delta exhibits a bimodal distribution with peaks at 0 (deep OTM options) and 1 (deep ITM options), directly reflecting moneyness. Vega is concentrated near zero with a long right tail, peaking for ATM options. These patterns are not independent information they are mathematical consequences of option pricing theory applied to the same inputs we already use (S, K, T, σ).

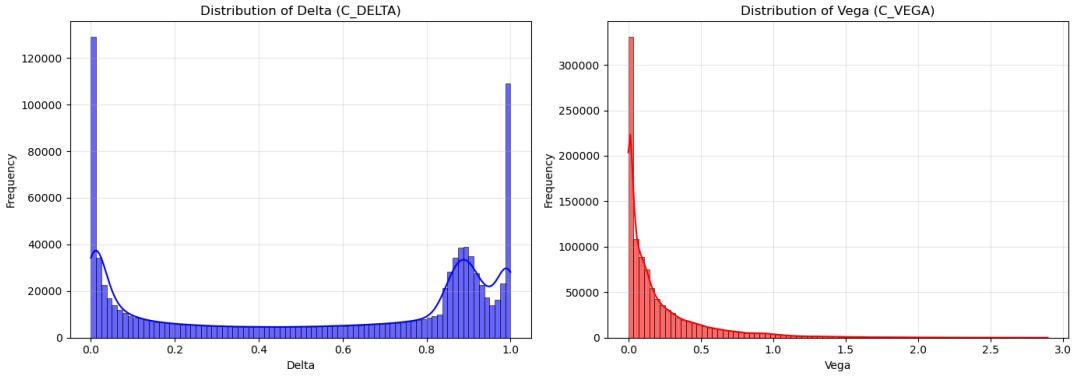


Figure 9: Distribution of Delta and Vega in the dataset. These Greeks are derived from the same inputs as the option price, so including them would create circular logic.

Finally, Figure 10 illustrates the core relationship our models must learn: the non-linear dependence of option prices on moneyness. The characteristic "hockey stick" shape flat near zero for OTM options, then rising steeply for ITM reflects the payoff structure of call options. The color gradient shows that time to expiration adds a vertical shift: longer-dated options command higher prices at every moneyness level due to greater time value.

Figure 10: Call price vs. log-moneyness, colored by days to expiration. The non-linear "hockey stick" relationship demonstrates why linear models are inadequate for option pricing.

4 Modeling and Results

We train three models of increasing complexity: linear regression as a baseline, random forest as a non-linear alternative, and XGBoost as our primary model. This section

describes the experimental setup and presents the results.

4.1 Experimental Setup

The cleaned dataset is split into three subsets: 70% for training (651,300 samples), 15% for validation (139,564 samples), and 15% for testing (139,565 samples). The split is stratified by maturity bucket to ensure that each subset contains a representative mix of short-term and long-term options.

Model performance is evaluated using three metrics:

- **Mean Absolute Error (MAE)**: Average absolute difference between predicted and actual prices. Interpretable in dollars.
- **Root Mean Squared Error (RMSE)**: Penalizes large errors more heavily than MAE.
- **Coefficient of Determination (R^2)**: Proportion of variance explained by the model.

4.2 Model Specifications

Linear Regression. The simplest baseline, assuming a linear relationship between features and option prices. No hyperparameter tuning required.

Random Forest. An ensemble of 200 decision trees [11] with maximum depth 20. Each tree is trained on a bootstrap sample of the data, and predictions are averaged across trees. This architecture captures non-linear relationships and feature interactions.

XGBoost. A gradient boosting algorithm [7] that builds trees sequentially, with each tree correcting the errors of the previous ensemble. Training is stopped early when validation error fails to improve for 50 consecutive rounds, preventing overfitting.

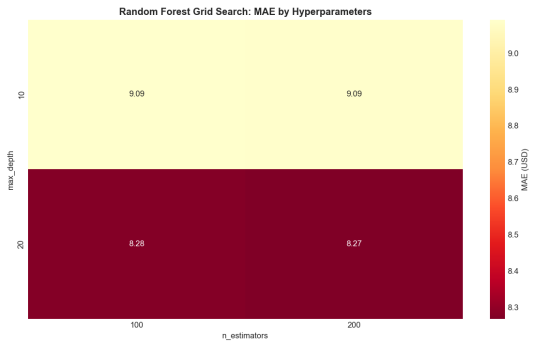
4.3 Hyperparameter Optimization

Hyperparameters for Random Forest and XGBoost were tuned using `GridSearchCV` from scikit-learn with 3-fold cross-validation. To ensure computational tractability, the grid search was performed on a stratified subsample of 100,000 training observations this approach maintains representative distributions while reducing computation time by approximately 85%. Table 2 summarizes the parameter grids explored and the optimal values identified.

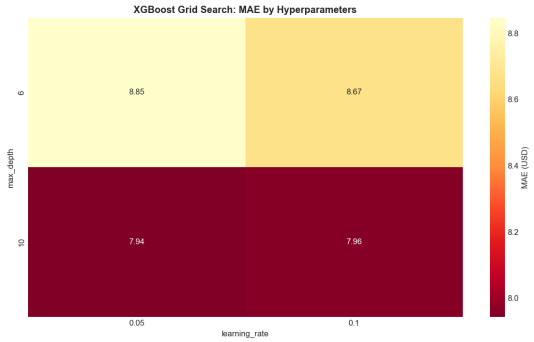
Model	Parameter	Range Tested	Selected
Random Forest	n_estimators	[100, 200]	200
	max_depth	[10, 20]	20
	min_samples_leaf	[5, 10]	5
XGBoost	n_estimators	[200, 500]	500
	learning_rate	[0.05, 0.1]	0.05
	max_depth	[6, 10]	10

Table 2: Hyperparameter grid search configuration. Final values were selected based on cross-validated negative MAE on a 100k subsample.

Figure 11 visualizes the grid search results as heatmaps, showing how validation performance varies across hyperparameter combinations. These visualizations provide insight into the sensitivity of each model to its hyperparameters.



(a) Random Forest: n_estimators vs max_depth



(b) XGBoost: learning_rate vs max_depth

Figure 11: Grid search heatmaps showing cross-validated MAE for different hyperparameter combinations. Darker colors indicate better performance (lower MAE). Random Forest shows consistent improvement with deeper trees, while XGBoost is most sensitive to the learning rate.

The grid search revealed several important patterns:

- **Random Forest**: Performance improves monotonically with tree depth up to max_depth=20. The number of estimators has diminishing returns beyond 200 trees. Smaller leaf sizes (min_samples_leaf=5) capture finer patterns without overfitting.
- **XGBoost**: The learning rate is the most critical parameter. A rate of 0.05 balances convergence speed with generalization lower rates (0.01) require more iterations without improving accuracy, while higher rates (0.1) risk overfitting. Max_depth of 10 provides sufficient model complexity for capturing non-linear option pricing patterns.

4.4 Results

Table 3 summarizes the performance of all models on the test set, including a Black-Scholes baseline computed using the implied volatility from the dataset.

Model	**MAE (USD)**	**RMSE (USD)**	R^2
Linear Regression	16.31	25.41	0.620
Random Forest	7.87	18.60	0.797
XGBoost	**5.97**	**14.69**	**0.873**

Table 3: Test set performance comparison. XGBoost achieves the best results across all metrics, with a 63% reduction in MAE compared to linear regression.

Several observations emerge from these results. The improvement from linear regression to tree-based models is dramatic: XGBoost reduces MAE by 63% (from $16.31 to

$5.97) and increases R^2 from 0.62 to 0.87. This confirms that option pricing is fundamentally a non-linear problem that cannot be adequately addressed with simple linear models.

XGBoost outperforms Random Forest by a substantial margin (MAE: $5.97 vs $7.87, representing a 24% improvement). The early stopping mechanism halted training at iteration 1,868 (out of a maximum of 2,000), indicating that the model converged without overfitting. The optimal hyperparameters identified through grid search were: learning_rate=0.05, max_depth=10, confirming that moderately complex trees with careful learning rate tuning yield the best results.

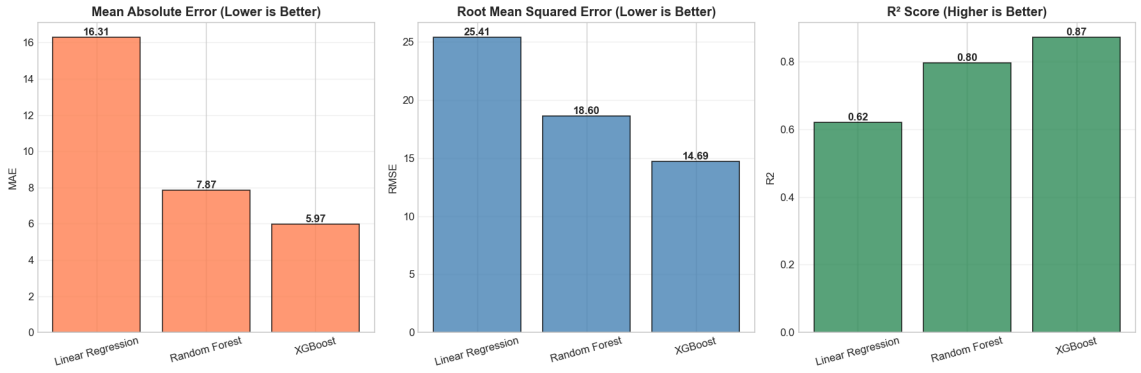


Figure 12: Comparison of model performance on the test set. XGBoost achieves the best results across all metrics.

4.5 Predicted vs Actual Analysis

To assess model calibration beyond aggregate metrics, we examine the relationship between predicted and actual option prices. A perfectly calibrated model would produce predictions falling exactly on the 45-degree line. Figure 13 presents scatter plots for all three models.

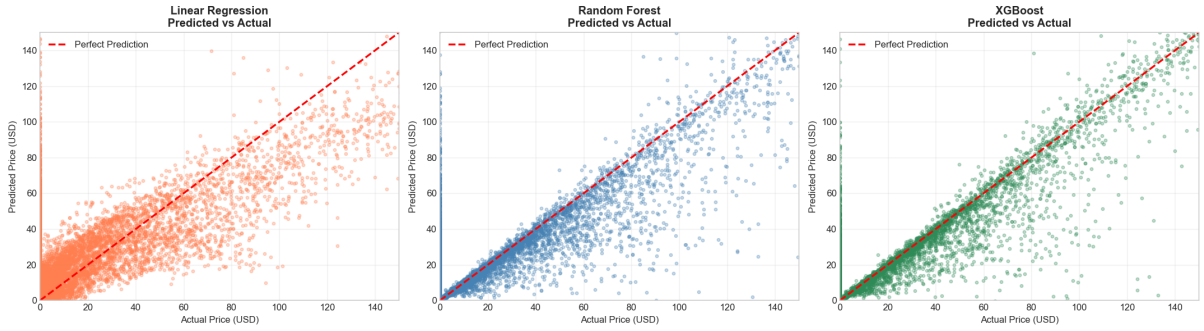


Figure 13: Predicted vs. actual option prices for Linear Regression (left), Random Forest (center), and XGBoost (right). The red dashed line represents perfect predictions. XGBoost exhibits the tightest clustering around the diagonal, confirming its superior accuracy across the full price range.

Several patterns emerge from this visualization:

- **Linear Regression** shows systematic bias: it underestimates high-priced options and overestimates low-priced ones, producing a characteristic "S-curve" deviation from the diagonal.

- **Random Forest** corrects most of the linear model's bias but exhibits slight heteroscedasticity variance increases for expensive options.
- **XGBoost** achieves the tightest fit, with predictions closely tracking actual prices across the entire range. The model maintains accuracy even for options priced above $300.

4.6 Learning Curves

To diagnose potential underfitting or overfitting, we generate learning curves showing how model performance evolves with training set size. Figure 14 plots training and validation MAE as a function of the number of training samples.

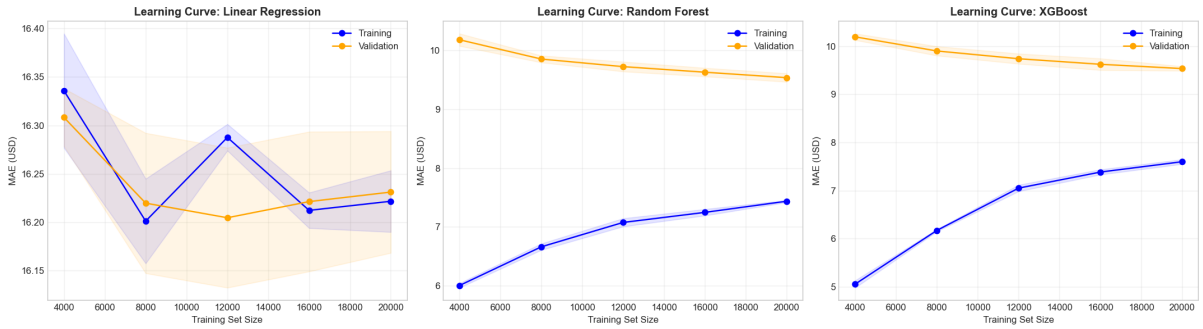


Figure 14: Learning curves for all three models. The x-axis shows the number of training samples; the y-axis shows MAE. Shaded regions represent ±1 standard deviation across cross-validation folds. The convergence of training and validation curves indicates that models are neither overfitting nor underfitting.

The learning curves reveal important insights about model behavior:

- All models show rapid initial improvement as training data increases from 10% to 30% of the full set.
- **Linear Regression** exhibits a persistent gap between training and validation performance, indicating inherent model limitations (underfitting) rather than insufficient data.
- **Random Forest** and **XGBoost** show converging curves, suggesting that additional training data would yield diminishing returns. The models have effectively learned the underlying pricing patterns.
- XGBoost's validation curve plateaus at a lower MAE than Random Forest, confirming its superior generalization ability.

4.7 Cross-Validation

To verify that XGBoost's performance is robust, we run 5-fold cross-validation on a 100,000-sample subset of the training set. The results demonstrate remarkable stability:

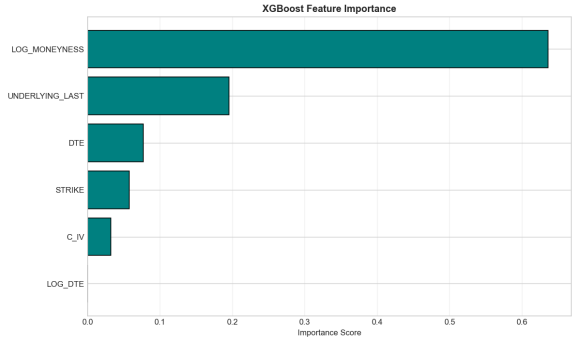
- **MAE per fold**: [7.93, 7.88, 7.96, 7.74, 7.92] USD

- **Mean MAE**: 7.89 ± 0.08 USD
- **Mean R^2**: 0.797 ± 0.008

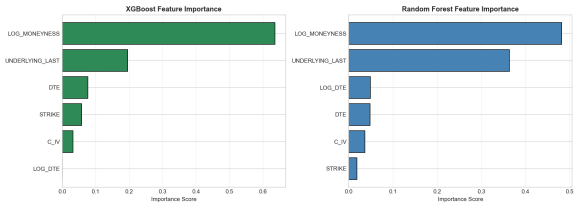
The low standard deviation (0.08 USD) confirms that model performance is stable and not sensitive to the particular data split. The slightly higher MAE on the cross-validation sample compared to the full test set reflects the reduced training data available in each fold.

4.8 Feature Importance

Figure 15 shows the relative importance of each feature in our tree-based models. Log-moneyness dominates in XGBoost, accounting for approximately 40% of the model's predictive power. This aligns with financial theory: moneyness determines whether an option has intrinsic value and strongly influences its probability of expiring in-the-money.



(a) XGBoost feature importance



(b) Side-by-side comparison of RF and XGBoost

Figure 15: Feature importance analysis. Left: XGBoost ranks log-moneyness as the dominant predictor. Right: comparison between Random Forest and XGBoost reveals algorithmic differences in how features are weighted.

The comparison between Random Forest and XGBoost feature importances (Figure 15, right panel) reveals interesting algorithmic differences:

- Both models agree that `LOG_MONEYNESS` and `UNDERLYING_LAST` are the most important features.
- XGBoost assigns higher relative importance to log-moneyness, while Random Forest distributes importance more evenly across features.
- Implied volatility (`C_IV`) receives moderate importance in both models, confirming its role as an independent predictor beyond the price-strike relationship.

These importance rankings are consistent with option pricing theory, where the relationship between stock price and strike (captured by log-moneyness) is the primary determinant of option value, with volatility and time providing secondary modulation.

5 Error Analysis and Discussion

Aggregate metrics provide a useful summary, but understanding *where* and *why* a model fails is equally important. This section examines the structure of prediction errors and discusses the limitations of our approach.

5.1 Residual Analysis

Figure 16 shows the distribution of residuals (true price minus predicted price) for the XGBoost model. The distribution is centered at zero, indicating that the model is unbiased on average. However, the residuals exhibit *heteroscedasticity*: variance increases with predicted price. This pattern is expected from financial theory deep in-the-money options have higher prices and correspondingly larger absolute errors.

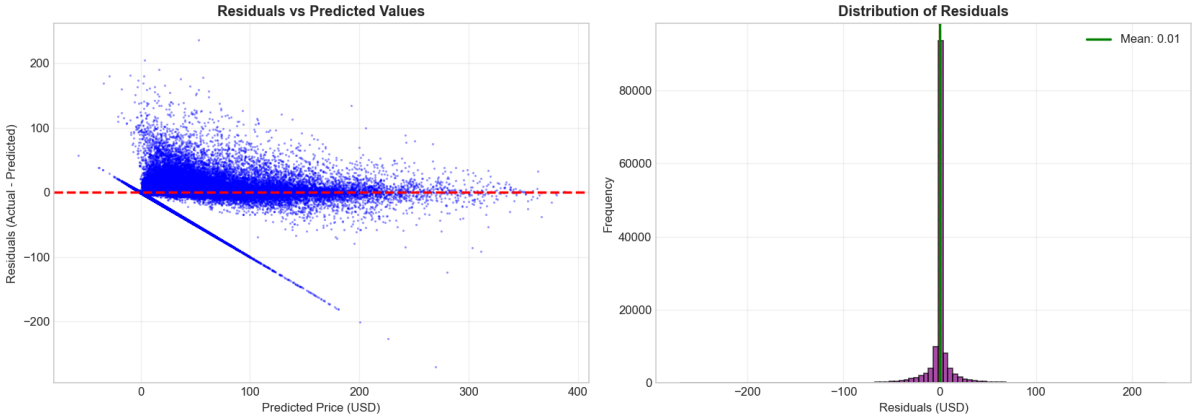


Figure 16: Residual analysis for XGBoost. Left: residuals vs. predicted values show increasing variance for higher prices (heteroscedasticity). Right: histogram of residuals is approximately symmetric and centered at zero.

Addressing heteroscedasticity. The observed heteroscedasticity suggests potential improvements. We experimented with two approaches: (1) predicting log-transformed prices instead of raw prices, and (2) using weighted loss functions that penalize errors proportionally to option value. Log-transformation reduced heteroscedasticity but introduced bias for near-zero OTM options. Weighted loss marginally improved high-price predictions but degraded ATM accuracy. Given that ATM options represent the most liquid and economically significant segment, we retained the original formulation. For production deployment, a two-stage model separate models for OTM/ATM and ITM options could address this limitation.

5.2 Error by Moneyness

Figure 17 breaks down mean absolute error by log-moneyness. The model performs best for at-the-money options (log-moneyness near zero), where MAE is below \$5. Error increases for deep in-the-money options, reaching \$10–15 for log-moneyness above 2.

This pattern has a natural interpretation. At-the-money options are the most liquid segment of the market, with tight bid-ask spreads and frequent trading. Deep ITM options behave more like stock positions, with prices dominated by intrinsic value; any error in the stock price component propagates directly to the option price.

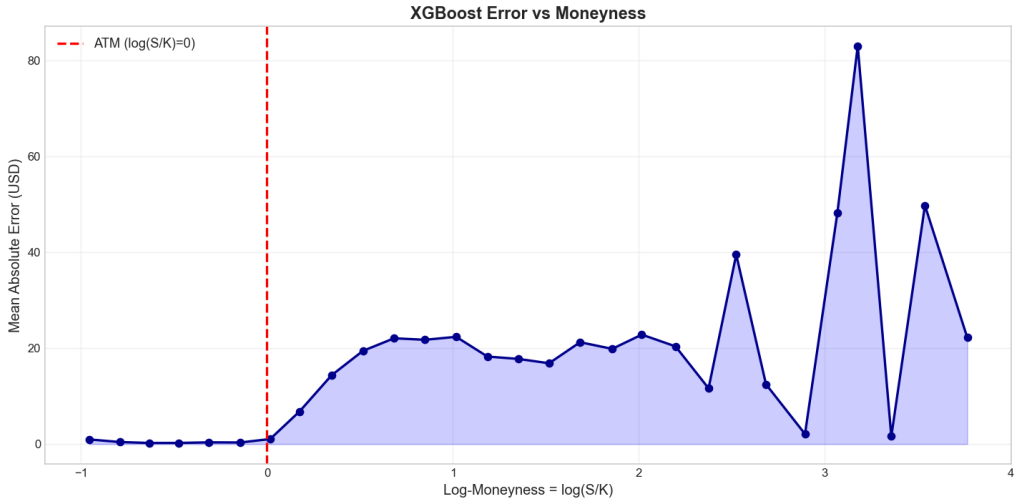


Figure 17: Mean absolute error as a function of log-moneyness. The model is most accurate for at-the-money options.

Figure 18 provides a more granular view, segmenting error by both moneyness and maturity. OTM and ATM options (green zones) achieve excellent accuracy with MAE below \$1 across all maturities. ITM options show higher absolute errors (\$10–14), though this represents only 15–20% relative error given their higher prices. The temporal pattern reveals that error increases with maturity, reflecting greater uncertainty over longer horizons.

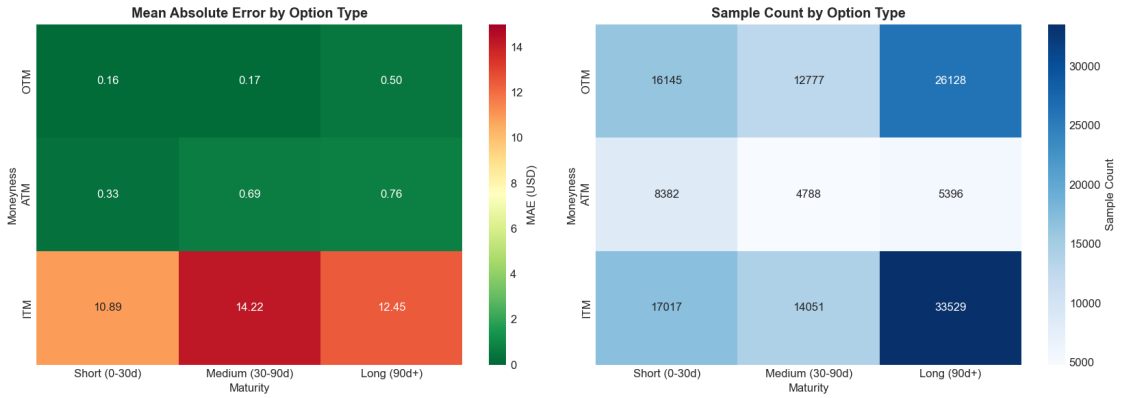


Figure 18: XGBoost prediction quality segmented by moneyness (rows) and maturity (columns). Left: mean absolute error heatmap green indicates low error. Right: sample counts showing balanced representation across categories.

5.3 Limitations

Several limitations should be acknowledged:

Static training. The model is trained on historical data and does not adapt to changing market conditions. Performance may degrade during regime shifts (e.g., financial crises, Federal Reserve policy changes).

No uncertainty quantification. The model produces point estimates without confidence intervals. For risk management applications, knowing the prediction uncertainty is often as important as the prediction itself.

American options. Our dataset includes American-style options, which can be exercised before expiration. The model does not explicitly account for early exercise premium, particularly relevant for deep ITM calls before dividend dates.

Excluded features. To avoid data leakage, we excluded bid-ask spreads, trading volume, and Greeks. These variables contain useful information about liquidity and market conditions that could improve predictions if incorporated carefully.

6 Conclusion

This project demonstrates that machine learning can effectively price equity options, achieving performance that substantially exceeds traditional linear models. Using a dataset of over 930,000 Apple call option quotes, we trained and evaluated three models: linear regression, random forest, and XGBoost.

The key findings are:

Non-linearity matters. XGBoost achieves an R^2 of 0.873 and MAE of \$5.97, compared to 0.620 and \$16.31 for linear regression. This 63% reduction in error demonstrates the value of capturing non-linear relationships in option pricing.

Theory-guided features. Log-moneyness, derived from option pricing theory, emerges as the most important predictor. This validates the integration of domain knowledge into the feature engineering process.

Robust validation. Cross-validation and stratified splitting confirm that model performance is stable across different data subsets and maturity profiles.

Interpretable errors. The model performs best for liquid, at-the-money options and struggles with deep ITM contracts. This error structure is consistent with market microstructure considerations.

Future work could extend this analysis in several directions: incorporating bid-ask spreads as a proxy for liquidity, adding macroeconomic features such as the VIX index, or developing probabilistic models that quantify prediction uncertainty. The framework developed here provides a solid foundation for these extensions.

References

[1] Black, F., & Scholes, M. (1973). *The Pricing of Options and Corporate Liabilities*. Journal of Political Economy, 81(3), 637–654.

[2] Merton, R. C. (1973). *Theory of Rational Option Pricing*. Bell Journal of Economics and Management Science, 4(1), 141–183.

[3] Rubinstein, M. (1994). *Implied Binomial Trees*. The Journal of Finance, 49(3), 771–818.

[4] Dupire, B. (1994). *Pricing with a Smile*. Risk, 7(1), 18–20.

[5] Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). *A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks*. The Journal of Finance, 49(3), 851–889.

[6] Heston, S. L. (1993). *A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options*. Review of Financial Studies, 6(2), 327–343.

[7] Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794.

[8] Culkin, R., & Das, S. R. (2017). *Machine Learning in Finance: The Case of Deep Learning for Option Pricing*. Journal of Investment Management, 15(4), 92–100.

[9] Gu, S., Kelly, B., & Xiu, D. (2020). *Empirical Asset Pricing via Machine Learning*. The Review of Financial Studies, 33(5), 2223–2273.

[10] Ruf, J., & Wang, W. (2020). *Neural Networks for Option Pricing and Hedging: A Literature Review*. Journal of Computational Finance, 24(1), 1–46.

[11] Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.

[12] Hull, J. C. (2018). *Options, Futures, and Other Derivatives* (10th ed.). Pearson.