USC Marshall
Master of Science in Business Analytics

DSO 530 | Group 13

Application of Machine Learning Techniques in European S&P 500 Call Option Pricing

Presented by:

Tom Hultsch (****787248) thultsch@usc.edu
Pedram Bazargani (****329414)
Abed Kassem (****385995)
Nico Santoso (****798204)
Nelly Tian (****110835)

Executive Summary

The Black-Scholes model, a classical method for valuing European call options on the S&P 500, has longstandingly been the industry standard. However, its reliance on strict assumptions and inability to capture the complexities of real-world market dynamics has prompted a shift towards exploring alternative valuation techniques. In this study, we apply Machine Learning models to undertake the same valuation task, and extend this with a secondary classification task to predict whether the Black-Scholes model would underestimate or overestimate the current option value.

Our models are trained and fitted on a dataset comprising 5,000 European call options, and predictions are made on a test set consisting of 500 European call options. Our models were constructed using four essential parameters: Current Asset Value (S), Strike Price (K), Annual Risk-Free Interest Rate (r), and Time to Maturity in Years (τ).

Through a rigorous process of systematic hyperparameter tuning, leveraging standard Machine Learning techniques as well as 10-fold Cross-Validation, we identified the Support Vector Machine (SVM) Regression model as the most accurate for option valuation, achieving the highest mean R-squared value. Concurrently, the XGBoost Classification model emerged as the top performer for the classification task, boasting the lowest Mean Classification Error.

Options trading is a highly dynamic and fast-paced domain, characterized by high-frequency transactions and non-linear market movements. We posit that the ability of Machine Learning techniques to adapt to these complexities and capture intricate patterns can significantly enhance the predictive accuracy compared to rigid, assumption-based pricing formulas like Black-Scholes. Our findings could have profound implications for options traders, risk managers, and the broader financial industry, potentially revolutionizing the way options are valued and traded.

Problem Statement & Data

European Call Options are incredibly dynamic securities, providing investors with an opportunity to purchase underlying stock at a pre-specified price at maturity of the option. Historically, the Black-Scholes model has been used as a standard approach for valuing such options. We look to explore different Machine Learning approaches to predict the current option value and classify the over-/underestimation of the Black-Scholes prediction, using options written on the S&P 500 Index.

Exploratory Data Analysis

To begin, we undertake usual data exploration and preprocessing best practices, including ensuring no null values, removing unnecessary columns, and extracting summary statistics.

Feature Selection

The four predictor variables included within our dataset are Current Asset Value (S), Strike Price (K), Annual Risk-Free Interest Rate (r), and Time to Maturity in Years (τ). The issue of multicollinearity between these predictors needs to be scrutinized, especially as the correlation matrix (*see Graph 1*) shows a strong negative relationship between Current Asset Value (S) and Annual Risk-Free Interest Rate (r).

In order to assess whether feature selection is necessary, we perform Best Subset Selection. This allows us to confirm whether model performance is superior when removing one or more predictors. Results show that our full model, without eliminating any of the predictors, performs better than in any other scenario. Despite high correlation between Current Asset Value (S) and Annual Risk-Free Interest Rate (r), from a business perspective, inclusion of both these predictors is justified. In fact, while multicollinearity can bring unstable coefficient estimates and high standard errors for the collinear variables, our output shows that it does not affect the model's overall predictive power, nor its ability to make accurate predictions.

To preface, when Annual Risk-Free Interest Rate (r) increases, riskier asset classes, such as stocks, become less attractive relative to other investment vehicles such as fixed-income investments (ie. treasury securities). Thus, stocks' values are expected to decrease. If we were to remove a predictor in our case, it would be that of the Annual Risk-Free Interest Rate (r). However, our objective is not to estimate the individual effects of each explanatory variable, nor is it to study the structure of multicollinearity and provide an understanding of the regression relationships. Rather, we want our model to provide the most comprehensive and accurate view of the factors that play a role in options pricing, and to achieve this objective, the Annual Risk-Free Interest Rate (r) must be included, even if it is correlated with another predictor.

Omitting any of these four variables, which are critical inputs to call option pricing models, would render suboptimal predictions. Generally, Current Asset Value (S) and Strike Price (k) are the most important determinants of intrinsic value, while Annual Risk-Free Interest Rate (r) and Time to Maturity in Years (τ) impact the time value component of the option premium.

Outliers

The boxplots on each of our variables *(see Graph 2)* reveal skewness and the presence of outliers on the Value (C) and Time to Maturity in Years (τ) variables. However, we retain all data points in our analysis, given that the distribution of data points on these features is continuous, rather than displaying "random" outliers that spur at extreme values. This tells us that the outliers are part of the natural distribution of these variables.

Removal of such outliers could lead to loss of important information and potentially biased option pricing models. This is further justified by widely-adopted option pricing models such as the Black-Scholes model, which in fact, assumes a lognormal distribution of the underlying asset price (which in itself, is inherently skewed). Thus, the presence of skewness in the data is not a concern in this specific case.

Balance in the Dataset

For our classification task, it is crucial to assess the balance of our dataset between the "Under" and "Over" classes. Since we observe that "Under" occurs three times more frequently than "Over" in the training data, this indicates an imbalanced dataset *(see Graph 3)*.

To address this imbalance, we will employ stratified k-fold cross-validation. This technique maintains the same proportion of minority ("Over") and majority ("Under") classes within each fold. By preserving class distribution, we prevent the models from becoming biased towards the majority class. This allows us to assess model performance more reliably.

Approach

Regression Models (see Table 1)

In order to arrive at the best model for predicting the Current Option Value (C), we utilize the Mean R-squared obtained from 10-Fold Cross-Validation, to understand the model's ability to generalize to previously unseen data. Additionally, where appropriate, we hypertune the model parameters to improve model performance.

We use Linear Regression as a baseline model and implement Lasso and Ridge Regression to test potential shrinkage, and to evaluate if a smaller model may perform better due to addressing the potential multicollinearity issues outlined above. However, compared to KNN Regression, it is evident that the relationship between the predictors and Current Option Value (C) is non-linear, as shown by a Mean R-Squared that is over 2.5% higher after hypertuning the number of nearest neighbors. Therefore we turn to tree-based models to further uncover this non-linear relationship. We tune the number of leaves in the Decision Tree algorithm and obtain a Mean R-Squared greater than 99%.

Summary of Final Approach for Prediction

Still, acknowledging that an ensemble method may yield more accurate results than decision tree algorithms, we employ a Random Forest Regression model, tuning the number of trees in the forest and the maximum number of splits for each tree. This further improves our model performance to a Mean R-Squared of 99.73%. However, we understand that this may point towards overfitting to the training data, since we continue to split each tree until the leaves are pure without any pruning. To address this concern and to better model the continuous and non-linear distribution of the variables, we look to Support Vector Machine (SVM) Regression using a Radial Kernel and hypertune the following parameters using cross-validation: the regularization parameter (C), and epsilon (specifying the epsilon-tube within which no penalty is associated). When epsilon is large in value, this permits greater flexibility, reduces model complexity, and lessens the likelihood of overfitting. Conversely, a larger regularization parameter (C) minimizes the L2 norm of the coefficient vector, thereby still promoting accurate predictions. Our resulting parameters signify this relationship, as both C and epsilon are large.

In comparison to Random Forest regression, while the mean R-Squared is similar, the mix of regularization and added leeway for error demonstrates the benefits of a SVM regression approach for generalization, especially when considering the trade-offs between model interpretability and prediction accuracy. As explained, SVM regression displays a slightly higher mean R-Squared, while also being significantly more interpretable than an ensemble method like Random Forest regression, where the contributions of each predictor are not as easily identifiable. Therefore, we implement SVM regression as our best model to make predictions on our test data.

Classification Models (see Table 2)

Similar to our approach for prediction, we are interested in understanding our classification models' ability to generalize when classifying the option value as Over or Under the prediction made by the Black-Scholes formula. To address the class imbalances in the training data, we conduct Stratified 10-fold Cross-Validation to preserve the original proportions in each fold, and subsequently compare models based on the resulting mean classification error. Following our regression exercise, we understand the nonlinearity present in our data; however, we test this assumption by establishing linear baseline classifiers using Logistic Regression and Linear Discriminant Analysis.

Again, comparing the baseline to KNN Classification, it is evident that non-linear models outperform. Therefore, following the same rationale as for regression, we construct a Decision Tree classifier. While the Cross-Validation Score obtained beats the benchmark set for the assignment, we posit that ensemble methods can further improve the accuracy of classification.

However, in this case, SVM approaches would not render improvement to ensemble methods as seen previously in Regression, since epsilon is not a tunable parameter in Support Vector Classification, which thereby focuses on feature combinations.

Summary of Final Approach for Classification

Therefore, we build a Random Forest classifier tuning the number of trees in the forest, the maximum depth of each tree, the minimum number of samples required to split an internal node, and the minimum number of samples required to be at a leaf node. While the resulting Mean Classification Error decreases, we further explore approaches using boosting, which creates decision trees sequentially to correct errors made by existing models.

Gradient Boosting and XGBoost both continue to decrease the Mean Classification Error. XGBoost, being an enhanced version of Gradient Boosting, is our best performing classification model, given it uses more accurate approximations, second-order gradients, advanced regularization, and sparse data. This provides better performance and generalization.

Beyond the parameters tuned for Random Forest, we also tune our boosting algorithms on the 'subsample' and 'colsample_bytree' parameters. For each tree, the 'subsample' parameter specifies the fraction of data points to be used to make decisions, while 'colsample_bytree' specifies the fraction of features to be used to make decisions. By finding the optimal values for these parameters, we control for model complexity, introduce randomness and diversity in the tree, and thereby reduce potential overfitting and improve generalization. Therefore, we implement XGBoost as our best classification model.

Conclusion

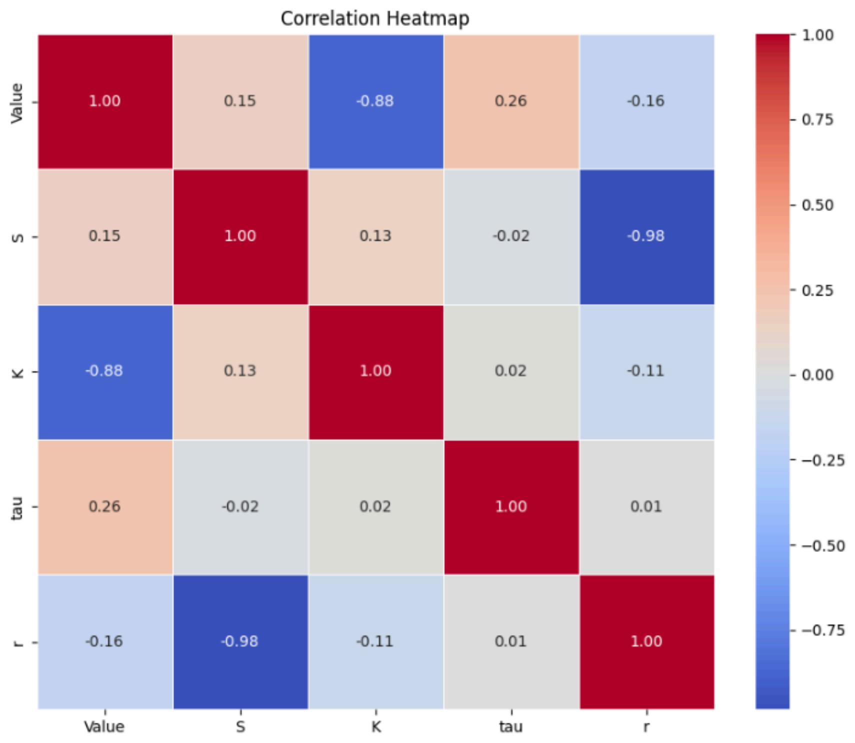
When predicting option values, the performance advantage of Machine Learning (ML) models over the Black-Scholes (BS) model is justified by ML models' flexibility and ability to learn directly from the data. ML provides a new programming paradigm to traditional programming (i.e., rigid, assumption-based, mathematically-heavy models such as the BS model), where the inputs are rules and data. In contrast, ML models are trained on data rather than explicitly programmed with rules, with the machine learning from input data and corresponding outputs to then figure out what rules should be. As such, ML models can learn useful representations of the input data at hand, including complex, non-linear relationships between predictor variables and the desired response variable. In other words, ML models can capture more intricate dynamics, update predictions as options markets evolve, and provide more accurate predictions, while the BS model struggles with these changing dynamics.

However, without additional validation, the top-performing ML models presented in this report could not be used to predict option values for individual stocks, such as Tesla stocks. Since these models were trained on an S&P 500 Options dataset, the training data may only partially capture the unique dynamics and the risks associated with singular stocks. For instance, Tesla's volatility, price movements, and time horizons may not follow closely to those of the S&P 500 Index, especially given the possibility of company-specific events, news, or other external factors. Additionally, there could be different levels of liquidity inherent in Tesla stocks, comparatively to the S&P 500 Index.

Prior to deploying the trained models to predict option values for Tesla stocks, an investor would need to first test the performance of these trained models on a holdout dataset specific to Tesla Options Pricing. Only then, upon checking the generalizability of the trained model, and undertaking required adjustments to ensure accurate and reliable predictions, would it make sense to deploy the models for Tesla stocks.

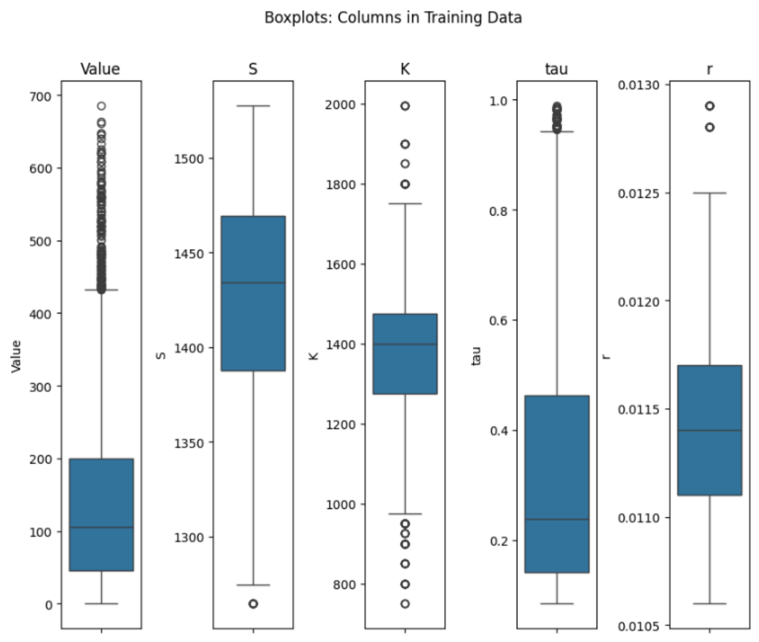
Appendix

Graph 1: Correlation Matrix



	Value	S	K	tau	r
Value	1.00	0.15	-0.88	0.26	-0.16
S	0.15	1.00	0.13	-0.02	-0.98
K	-0.88	0.13	1.00	0.02	-0.11
tau	0.26	-0.02	0.02	1.00	0.01
r	-0.16	-0.98	-0.11	0.01	1.00

Graph 2: Boxplots of Continuous Variables



Graph 3: Bar Chart of Categorical Training Labels for Classification

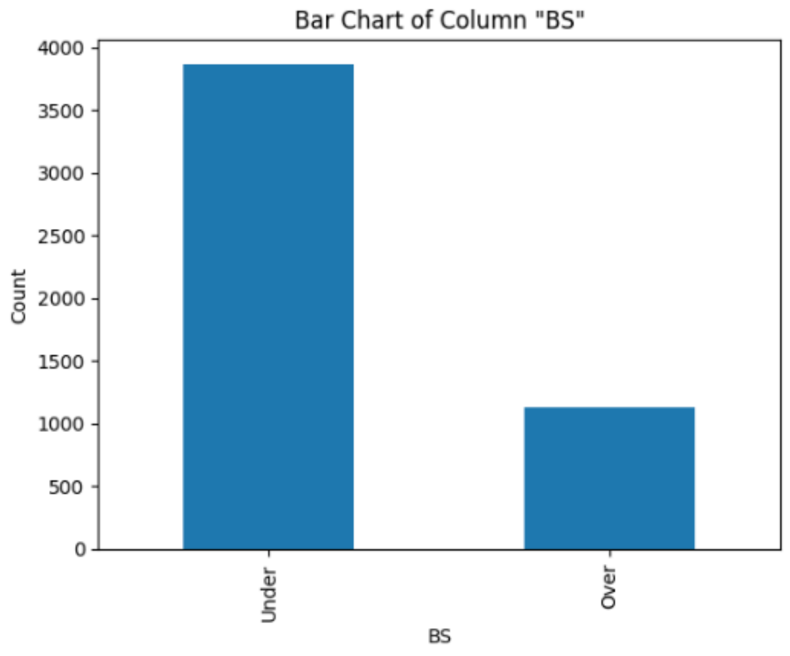


Table 1: Regression Model & Cross-Validation Scores

Regression Model	**Mean R-Squared of 10-Fold Cross-Validation**
Linear	0.9248207
Lasso	0.9248248
Ridge	0.9248221
KNN	0.9522437
Decision Tree	0.9934173
Random Forest	0.9968478
Support Vector Machine	0.9973791

Table 2: Classification Models and Cross-Validation Scores

Classification Model	**Mean Classification Error of 10-Fold Cross-Validation**
Logistic Regression	0.127599
Linear Discriminant Analysis	0.130800
KNN	0.112599
Decision Tree	0.080999
Random Forest	0.062399
Gradient Boosting	0.058799
XGBoost	0.058400