Enhanced Prediction Model for Options Pricing Using Machine Learning and Deep Learning Approaches

Group 36:

Mackenzie Chen	3544870056
David Liu	8491459656
Joyce Chen	5175022775
Jessie Yang	1695351583
Shih-Ling Hsu	1189609076

Email: yangjess@usc.edu

Date: May 2nd, 2024

Executive Summary

Our approach began with a thorough data preparation and inspection phase, where we confirmed the absence of arbitrage opportunities and ensured data integrity for further analysis. Initial feature engineering utilized domain knowledge to enhance the dataset with additional relevant features. A rigorous feature selection process followed, initially using a random forest for determining feature importance but eventually settling on forward selection methods due to issues with feature similarity and correlation. This method, combined with a stringent correlation threshold, refined our feature set to the most relevant predictors. Then, we trained five models each, with proper hyperparameters tuned and k-fold cross-validation method, for the regression and classification model to find those with outstanding performance.

After fitting 5 models for each question and to leverage the strengths of different models and stabilize predictions, ensemble methods were then employed: we decided to use the meta-model approach for regression and Weighted Soft Voting approach for classification because of the better prediction accuracy. The results of these two ensemble learning models both surpassed the best single prediction model.

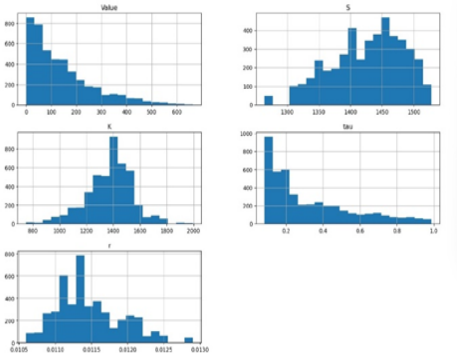
In conclusion, the comprehensive modeling effort culminated in highly accurate predictions, achieving an R^2 of 99% for value predictions and 95.2% accuracy for BS predictions. This high level of performance underscores the robustness of the prediction models and their potential applicability in real-world financial settings. However, the complexity of the models poses challenges in terms of interpretability, and their effectiveness may vary for alpha-driven stocks. Continuous updates and refinements with new data are recommended to maintain the model's relevance and accuracy in a changing financial landscape.

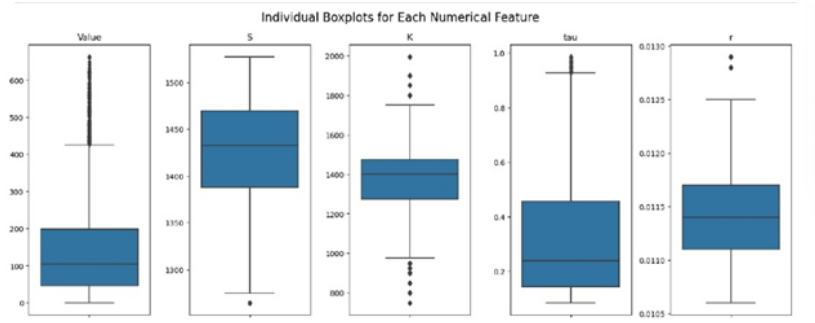
Data Preparation and Inspection

In the first step, we checked for arbitrage opportunities in our dataset by comparing 'S' with 'Value' using the condition $C > Max[S\text{-}PV(X), 0] \geq Max[S\text{-}X, 0]$. This condition sought instances where the adjusted market price exceeded the model's valuation. An empty result from this filter confirmed the absence of arbitrage opportunities.

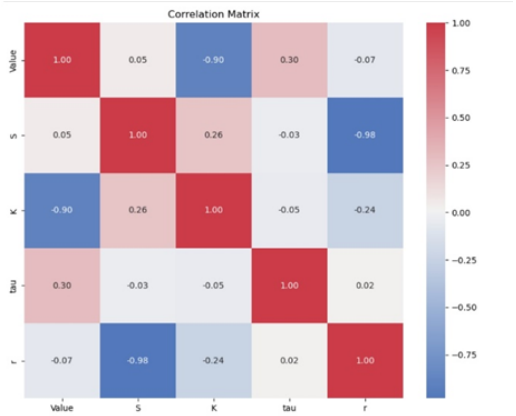
Then, to ensure the generalizability of the final prediction model, we started with splitting the original data into two parts: the first 4500 rows were used as the training and validation set, and the rest were used as the testing set.

In exploratory data analysis, we confirmed no negative numbers in 'Value' and 'S' columns, and that 'tau' values were positive. Histograms showed options with longer expiries had higher prices, suggesting higher profitability. We also generated Boxplots to identify outliers, which, though present, were justified (e.g., higher values correlate with lower strike prices and longer expiries). Since these did not contradict financial norms, we retained all outliers.





Our correlation analysis revealed a near-perfect negative correlation between 'S' and 'r' (-0.98), showing they typically moved in opposite directions. 'Value' and 'K' also exhibited a strong negative correlation (-0.90). A moderate positive correlation existed between 'Value' and 'Tau' (0.30), with minimal correlation observed among other variable pairs.



Data Manipulation: Feature Engineering

Using two-way design and domain knowledge, we created 9 additional features from existing data. For instance, 'S/K' represents the asset price to strike price ratio, and 'r**tau' adjusts 'r' for the time value of money by raising it to the power of 'tau'.

In addition, we standardized all columns except 'BS' and 'Value', converting 'BS' to numerical format for simpler model fitting.

Feature Selection

Next, we would like to conduct feature selection on both the features we initially had and the ones we generated from the feature engineering process. At the beginning, we tried the feature importance method. We used random forest to find out the relative importance of the features. However, we discovered that since we don't have a lot of unique features initially (only 4 distinct ones), several of our features are quite similar — which could lead to highly correlation.

Therefore, we turned to using the ensembled forward selection method. We ran 3 different models: logistic regression, decision tree, and SVM, each model 5 times to do so. In other words, we ran 15 times. We set random seeds to generate train test splits, then ran forward selection for all 3 models. Finally, we counted the times each feature was selected. We ultimately decided to use features that occurred more than two times in the total 15 runs.

The benefit of using forward selection is that it prevents us from choosing highly correlated features. To demonstrate, if the second feature we initially want to choose is highly correlated with the one we have previously chosen, the increase in R-squared would be very low. Therefore, we will not choose this feature.

Furthermore, to prevent selecting highly correlated features, we used correlation matrix to examine the correlation. We set an optimal threshold of 0.97, which prevented us from removing too many features at the same time being able to keep the necessary ones. After identifying highly correlated pairs, we then removed the one who had the lower importance in the previous forward selection. In this stage, we took out two features.

After these two processes, we received our final features to run models for the next step. Our 5 selected features are: 'K', 'r**tau', 'S*tau', 'S*r', 's-k*tau'.

Training Using Single Model(s)

Moving on to the modeling design section, we choose to adopt **k-fold cross validation** method for all the applicable models to better conclude the level of accuracy (R^2 for regression models and accuracy for classification models). The choice of 3-fold cross validation method for regression models, in our opinion, works well in terms of balancing processing time and prediction accuracy as it enables the system to save time for the **grid search** method that runs hundreds of times per applicable model. More details about the prediction models executed in our analysis are as follows.

For the regression models, we include 5 models in our analysis: they are **linear regression, k-nearest neighbors (KNN), random forest, XGBoost, and neural network models**. Among these five models, we apply the grid search cross-validation technique only to KNN, random forest, and XGBoost for hyperparameter optimization. Such a method improves the accuracy of corresponding models, enabling us to generalize an enhanced prediction model based on tuned models. Results are as follows. For the classification models, we process with similar approaches but with column BS as the new outcome variables and 5-fold cross-validation.

Using 3-fold CV method, XGBoost model achieves the highest R^2 value

Model	K-folds	Hyperparameter Tuning	CV Score
Linear	3	/	0.8657
KNN	3	'n_neighbors': 7, 'p': 1, 'weights': 'distance'	**0.9803**
Random Forest	3	'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200	**0.9913**
XGBoost	3	'colsample_bytree': 1, 'learning_rate': 0.2, 'max_depth': 5, 'n_estimators': 200, 'subsample': 0.8	**0.9953**
Neural Network	5	/	0.9253

Using 5-fold CV method, XGBoost model achieves the highest level of accuracy

Model	K-folds	Hyperparameter Tuning	CV Score
Logistic	5	'C': 100, 'penalty': 'l1'	0.8903
KNN	5	'metric': 'euclidean', 'n_neighbors': 15, 'weights': 'distance'	**0.9283**
Random Forest	5	'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200	**0.9290**
XGBoost	5	'learning_rate': 0.1, 'max_depth': 5, 'min_child_weight': 1, 'n_estimators': 300, 'subsample': 0.9	**0.9333**
Neural Network	5	/	0.9151

Voting Ensembles

After training the models, we employed ensemble techniques to enhance our predictions. Each model type has its unique strengths and weaknesses, so using a voting method could potentially perform better than any single model.

Classification:
Our classification performance is not really satisfactory, so we made further improvements through the following steps:

1. We trained each of the five models on the training set. Then, we used them to predict the probability that Y equals 1 on the validation set.

2. For the weighted soft voting method, based on cross-validation results, we excluded the logistic regression model and assigned weights to other models to compute the final weighted probability predictions, which were then converted into binary outcomes (0 or 1).

3. For the meta-model method, we excluded the logistic regression model, using probability predictions from other models as features for training. In other words, it's a model of the models. We tested both logistic regression and a gradient boosting classifier for ensembling. Due to its comparable performance and greater stability, which reduces overfitting, we chose logistic regression as our meta-model.

4. We evaluated our models using the testing set to determine our optimal prediction method and estimate performance on the private dataset. The weighted soft voting has the highest accuracy (0.952), so we use this approach as our classification voting method.

Regression:
Using the same process as classification except predicting r-squared instead of probabilities, the meta-model method got the highest r-squared (0.99), so we used this approach as our regression prediction method.

Result

In the final stage, we retrained our models on the entire dataset of 5000 options. This step was crucial in refining our models' accuracy and generalizability by fully utilizing all available data. We then used this comprehensive model to make our final predictions. The following table summarizes the result of each method we tried to make predictions. We mainly focused on the performance on the hidden, untouched testing dataset since it would be more representative of the actual result.

		Validation R^2 / Accuracy using CV	"Actual" R^2 / Accuracy on Hidden Testing Dataset
Regression: Value Prediction	Best single model	XGBoost: 0.9951	Random Forest: 0.9782
	Weighted soft voting	0.9945	0.9792
	Meta-model	0.9954	**0.9900**
Classification: BS Prediction	Best single model	XGBoost: 0.9244	KNN & Random Forest: 0.948
	Weighted soft voting	0.9230	**0.952**
	Meta-model	0.937	0.928

Therefore, we decided to use meta-model to predict values, and weighted soft voting to predict BS, as they performed the best.

Conclusion

We successfully designed an integrated enhanced prediction model that achieved R2 of 99% for value predictions and 95.2% accuracy for BS predictions on the unseen test dataset. The high level of performance demonstrates the model's robustness and its potential to apply in real-world financial markets. While our model provides substantial predictive accuracy, it does have some limitations in terms of interpretability due to the complexity. Additionally, the model might not be as effective when predicting option prices for alpha-driven stocks, which are those with unique competitive advantages like patents or market entry barriers. Further validation and potential model re-training with specialized datasets are recommended. In conclusion, while our model showed powerful performance, it is important to keep the model updated with new data to maintain its effectiveness and adaptability in the dynamic financial market.