Options Pricing Project

University of Southern California

DSO 530: Applied Modern Statistical Learning Methods

May 2, 2024

Group 49
Jessica Bratahani - 7041404049
Pin Hsuan Chang - 3949731079
Suhan Ho - 9648292115
Sheena Huang - 7622640822
Yunchi Lee - 1283919193
Contact Person Email: jessica.bratahani.2024@marshall.usc.edu

Executive Summary

In this project, we explore the application of supervised machine learning methods to predict the pricing of European call options on the S&P 500. The ability to predict option prices accurately is important for investors managing risks and seeking profitable opportunities.

Our team applied various predictive models including Linear Regression, Logistic Regression, K-Nearest Neighbors, Decision Trees, and Random Forests to datasets provided. The focus was on two main tasks: regression to predict option values and classification to predict compliance with the Black-Scholes (BS) model, whether the options value is over or underestimated.

Based on our experiments, the Random Forest model is selected for both tasks as it achieved a mean R-squared value of 0.996 in 5-fold cross-validation for regression. For the classification task, the model demonstrated an impressive accuracy rate of 93.66% from 10-fold cross validation. These results show the potential of machine learning models to enhance the traditional option pricing methods like the Black-Scholes formula.

The implications of our findings suggest that machine learning can significantly enhance financial analytics, providing more dynamic and robust tools for price prediction. Future work could explore further enhancements in model accuracy and adaptability under various market conditions.

However, when applying these machine learning results to investments with unknown future characteristics, it is crucial to exercise caution. Our team does not suggest solely relying on ML predictions for decision-making. Instead, we recommend integrating these insights with robust risk management strategies, market dynamics, expert opinion, and quantitative financial Analysis. This balanced approach ensures that while leveraging the predictive power of AI, investors also safeguard against potential uncertainties in financial markets.

In this report, we will introduce the machine learning methods we used, discuss our selection process, and cover key considerations for applying these predictions in business scenarios.

Methodology

I. Regression

Review of Approaches

In order to predict Current Option Value (Value) from the parameters Current Asset Value (S), Strike Price (K), Time to Maturity (tau) and Interest Rate (R), a variety of supervised learning methods for regression were explored. The methods our group explored includes: Linear Regression, Lasso and Ridge Regression, K-Nearest Neighbors (KNN) regression, Decision Tree and Random Forest for regression.

In the case of linear regression, we also explored best subset selection to find the impact of the number of parameters used in the model to model performance. Since there are only 4 parameters in this case, best subset selection is still computationally feasible and was our selected choice compared to forward or backward stepwise selection. However, our findings (as shown in Figure 1 in the appendix) suggest that using 3 vs. 4 predictors did not impact the in-sample R-square significantly as both models resulted in having an R-squared value of 0.925. In the next section, we will review how we selected our final model and discuss further actions that could be taken to improve our predictions.

Selection and Summary of Final Approach

To find the best model, our team used k-fold cross validation, with 5 and 10 folds, with the training data, across the models and compared them based on mean r-squared and mean-squared error (MSE) metrics. K-fold cross validation is a non-parametric evaluation method, thus allowing us to assess each model's generalization capability. For our comparison, the training data parameters was standardized using Scikit-learn's StandardScaler().
Table 1 below illustrates the comparison of the models.

Model	KFolds	Mean R Squared	Mean MSE
Lasso	5	0.924447	1182.78
LinearRegression	5	0.924657	1179.48
Ridge	5	0.924658	1179.47
KNeighborsRegressor	5	0.990365	150.12
DecisionTreeRegressor	5	0.992236	120.97
RandomForestRegressor	5	0.996515	54.40
Lasso	10	0.924599	1183.26

LinearRegression	10	0.924821	1180.05
Ridge	10	0.924821	1180.04
KNeighborsRegressor	10	0.991001	140.81
DecisionTreeRegressor	10	0.993242	105.15
RandomForestRegressor	10	0.996776	50.43

Table 1: K-Fold Cross Validation on Regression Models

Random Forest Regressor is the model with the highest mean R-squared(0.996) and lowest MSE(54.40) compared to other regression models for both 5 and 10 fold cross validation.

With Random Forest as the optimal regression model, our team decided to perform hyperparameter tuning on the random forest regressor, starting with RandomForestRegressor(random_state=0, oob_score=True) as the base model. Utilizing GridSearchCV(), allowed us to evaluate a range of parameter values and select the configuration that yielded the best performance: RandomForestRegressor(max_depth=30, max_features=3, min_samples_leaf=2, n_estimators=1000, oob_score=True, random_state=0). However, when implementing 5-fold cross validation on the two models, the base model still resulted in higher mean R-squared and lower MSE, therefore we decided to stay with the base random forest model as our final model for regression.

For our final prediction on Value from the test data, we used StandardScaler transform() on the test data and made the prediction on our final regression model fitted on the whole training dataset (with scaled X values and y training data).

Further Steps

To enhance our model further, we can broaden our scope by incorporating additional parameter values during the tuning process. Additionally, we could expand our exploration by incorporating other regression models, such as Neural Networks, to leverage their potential for capturing complex relationships within the data.

II. Classification

Review of Approaches

Classification methods were used in predicting Black Sholes' performance (whether the options value is over or underestimated) from the same 4 variables used to predict Value in regression. The models we tried for classification include Logistic Regression, KNN for classification, Decision Tree, Gradient Boosting, Random Forest and Support Vector Classifier.

In order to assess the accuracy of our BS predictions based on those parameters, we investigated the impact of feature scaling on our results. To evaluate the different models, we split the training data into training and test data sets to fit the best model and compare the accuracy scores of different models. In addition, we also employed K-Fold cross validation in the similar manner we did to compare regression models.

Selection and Summary of Final Approach

To find the best model, our team used k-fold cross validation, with 5 and 10 folds, with the training data, across the models and compared them based on prediction accuracy of classification. Table 2 below illustrates the comparison of the classification models.

Model	KFolds	Accuracy
LogisticRegression	5	0.8784
KNeighborsClassifier	5	0.8552
DecisionTree	5	0.9124
RandomForestRegressor	5	0.9334
GradientBoosting	5	0.9264
SupportVectorClassifier	5	0.8856
LogisticRegression	10	0.8732
KNeighborsClassifier	10	0.8594
DecisionTree	10	0.9128
RandomForestRegressor	10	0.9366
GradientBoosting	10	0.9268
SupportVectorClassifier	10	N/A

Table 2: K-Fold Cross Validation on Classification Models

With Random Forest as the optimal classification model (mean accuracy of 93.66% in 10 fold cross validation), our team decided to perform hyperparameter tuning on the random forest classifier, starting with RandomForestClassifier(random_state=1, n_estimators=200) as the base model. We then used GridSearchCV() again to evaluate a range of parameter values and select the configuration that yielded the best performance: RandomForestClassifier(max_depth=6, max_leaf_nodes=9, n_estimators=200, random_state=1).

However, when implementing 10-fold cross validation on the two models, the base model still resulted in higher prediction accuracy (93.66% vs. 89.38%), therefore we decided to stay with the base random forest model as our final model for classification. With our best model, we

compared the 10-fold cross validation score when training using scaled versus non-scaled parameters, and found that the prediction accuracy improved by 0.02% from 93.66% to 93.68%. Therefore, for our final prediction on BS from the test data, we used StandardScaler transform() on the test data and made the prediction on our final classification model fitted on the whole training dataset (with scaled X values and y training data).

Further Steps

To improve our models in the future, we can explore other boosting models that can help us have a broader understanding of the dataset, such as XGBoost, LightGBM and CatBoost, and apply Neural Networks to the dataset as well. Additionally, we can carefully observe the distribution of our raw data and remove the outliers before training models.

Conclusion

Based on our findings, machine learning models can significantly enhance financial analytics, providing more dynamic and robust tools for price prediction. In the future, further enhancements in model accuracy and adaptability under various market conditions could be explored. In this project, our team has explored the application of multiple supervised machine learning models to predict the pricing of European call options on the S&P 500 and found that the Random Forest Models were the best models for both regression and classification tasks.

Machine learning models excel over traditional methods like Black-Scholes due to their adeptness at managing non-linear complexities and adapting to changing market conditions, continuously learning from new data and integrating diverse inputs to capture the complexities of financial markets. However, when applying such models to specific contexts like predicting option values for high volatility stocks, challenges arise due to limited historical data, the complexity of market dynamics, and variations in regional and industrial characteristics. These factors underscore the importance of cautious consideration and adaptation when leveraging machine learning in real-world financial scenarios.

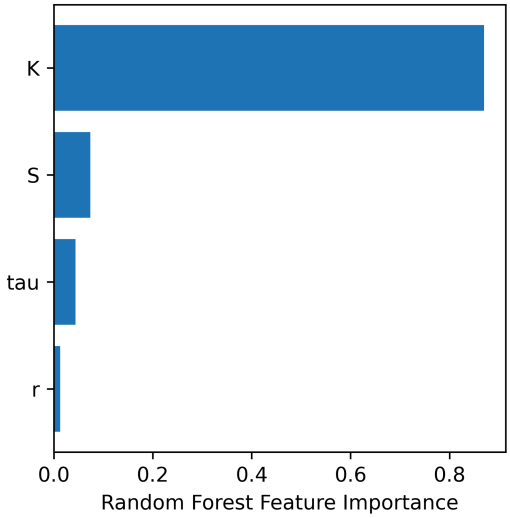
Thus, our team does not suggest solely relying on ML predictions for decision-making. Instead, we recommend integrating these insights with robust risk management strategies, market dynamics, expert opinion, and quantitative financial Analysis. This balanced approach ensures that while leveraging the predictive power of AI, investors also safeguard against potential uncertainties in financial markets.

References

- Python Code:
  DSO530 Python Tutorial 8
  DSO530 Python Tutorial 9
  DSO530 Python Tutorial 10
  DSO530 Python Tutorial 11

Appendix

1. Figure 1: Best Subset Selection on OLS Regression
   When performing best subset selection on OLS regression, the model with 3 parameters is selected to be the best subset in terms of adjusted R-squared, AIC and BIC, however it is very similar to the values from the model with 4 parameters.
2. Figure 2: Feature Importances on Random Forest Regression



The following pages part of the appendix are the PDFs of our .ipynb code. A brief overview of the codes and content in the pdfs appended are below:

3. Final Prediction Code
   a. DSO530Project_Final_Prediction.ipynb code in PDF
      - Code for final predictions of 'Value' and 'BS' using our best models on the given test data
4. Regression Code
   a. CV_Regression_Models.ipynb code in PDF
      - K-fold cross validation across different regression models
      - Hyperparameter tuning on Random Forest Regressor
      - Feature importances of random forest regressor

   b. Linear_Regression.ipynb code in PDF
      - Comparison of 3 vs. 4 parameter linear regression models
      - Best Subset Selection for linear regression model
5. Classification Code
   a. Model without scaled features
      - K-fold cross validation different classification models
      - Hyperparameter tuning on Random Forest Classifier
      - Tuning the hyperparameter for Support Vector Classifier on 10-fold cross validation cannot run on the computer
   b. Model with scaled features on 10-fold cross validation
      - Pick the models that have better performance from the unscaled version
      - Compare the cross validation across different classification models
   c. Train_Test_Split on 10-fold cross validation
      - Utilized the train_test_split on the training data to find the best model
      - Tuning the hyperparameter for Support Vector Classifier on 10-fold cross validation cannot run on the computer