Applied Modern Statistical Learning Method on Predicting Options Pricing

DSO530 group 25
May 5, 2022

En-Ning Chiang | 4522028870
Shih-Ting Liu | 7923289754
Sih-Yu Huang | 6123726133
Chin-Kai Huang | 6552002967

Executive Summary

Suppose that we would like to invest in a European call option and buy this option 6 months later at a strike price, how can we completely promise that the price would fluctuate as regularly as to how we expect in the future? A half year later, the market price would fall into two possibilities. One is that the market price would be higher than the strike price. That is, we could own the option with less cost. On the other hand, the possibility is that the market price would be lower than the strike price. Although we could choose not to buy the option at that time, we still would have already paid the deposit, which is the current option value.

For now, we have another question. How do we choose the amount of the current option value? Although there is Black-Scholes(BS) formula we could utilize to analyze, in this project we will try to rebuild a machine learning model that could perform the tasks using data from S&P 500. Besides, we would compare the predictions with the formula as well.

To achieve this goal, we first examine the dataset we possess and explore distributions of different fields. In the dataset, we have four dependent variables and two independent variables. For dependent variables, S is the current asset value, K stands for the strike price of the option, r represents the annual interest rate, and tau is time to maturity. For independent variables, value means current option value and BS is that, if the value predicted by the BS formula is less than the current option value, then it is "Under", otherwise, it is "Over". To go further in our analysis, we clean some empty data and remove outliers as well to ensure the data quality is beyond the standard. Then, apart from renaming the BS label in our training dataset, to enhance our accuracy and make the most of our current dataset, we also split our dataset into two parts, in that the first part is the training part, and the second one is the testing part.

After setting up all the predecessor tasks, we begin to leverage machine learning algorithms to build different kinds of supervised regression and classification models, and we also tune parameters to conclude an optimized real-time model. To make the outcome more ideal, we also endeavor to try different kinds of sets for the data split and random state.

Our 8 different supervised regression models explored and compared demonstrate that the XGboost model shows the most stable performance with over 99% accuracy. For our 10 classification models, the Gradient Boosting model demonstrates the most stable performance with a 6.08% classification error rate.

As a result, through the implementation of our model, we look forward to making the current option more predictable and understandable. We could also adjust the threshold to cater to different investors based on the extent of risk aversion. Moreover, we hope that we could apply the model to other options that have similar nature to S&P 500. With the further application, we believe that the model and concept could also be utilized in other financial products.

Data Exploration and Data Preprocessing

Before diving into the dataset directly, we start our analysis by checking the quality of data and process data cleaning. There are 1,680 observations with 6 fields in our training dataset, with 2 records that have missing values. We decide to drop these two rows directly since they only account for less than 1% of the entire dataset. Besides, we found 2 outliers within the variable tau and 1outlier within variable S, and we remove them as well. After cleaning our data, we have 1,675 rows total. Moreover, we rename the BS labels, which 1 represents 'Over' and 0 means 'Under', in our training dataset. Finally, before building our model, we utilize standardscaler to standardize our data.

Regression analysis

To value the European call option, we use the parameters S, K, r, and tau in our dataset to run the regression model and predict the value of the option. We first run the baseline model and then move forward to non-linear ones.

To make good use of our dataset, we use the train-test split and cross-validation method to make sure our models are sufficiently trained.

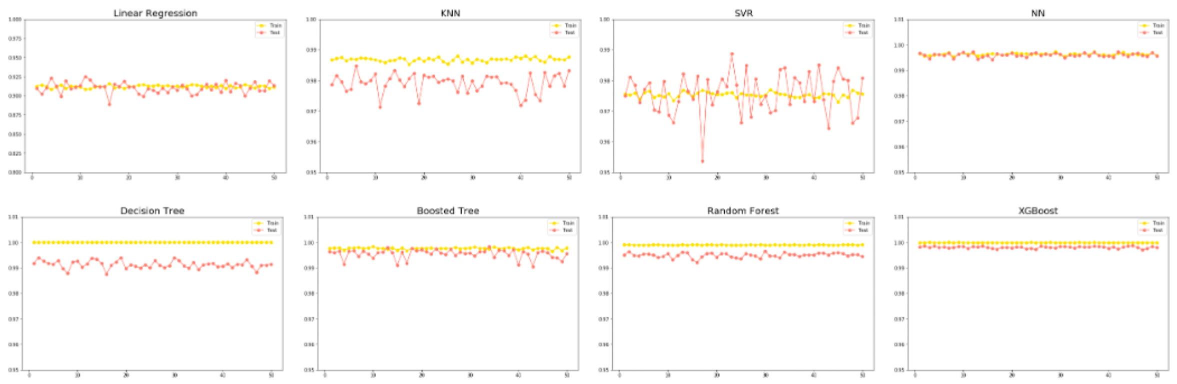
- Train-test split: Set the test size as 20% of the training dataset
- Cross-validation: Split 10 times with shuffle

From Table 1 below, we can tell that all tree-based models and Neural Network one perform better than others. All of them have over 99% R-squared. To better understand if the model is overfitting, we plot out the R-squared for the training and testing set with different random seeds to compare and see if there are large differences in between. Regarding Figure 1, XGBoost is the most stable model with a high level of variation explanation power. After the trade-off between bias and variance, **we decide to choose XGBoost as our final regression model to predict option value.**

[Table 1] Model performance comparison (R-squared)

Regression Model	Train-test split	Cross-validation
Linear Regression	91.7%	91.0%
Decision Tree	99.1%	99.3%
Boosted Tree	99.5%	99.6%
XGBoost	**99.5%**	**99.5%**
Random Forest	99.8%	99.8%
KNN	98.3%	98.0%
Support Vector	98.1%	97.7%
Neural Network	99.7%	99.7%

[Figure 1] R-square for training and testing set



Classification analysis

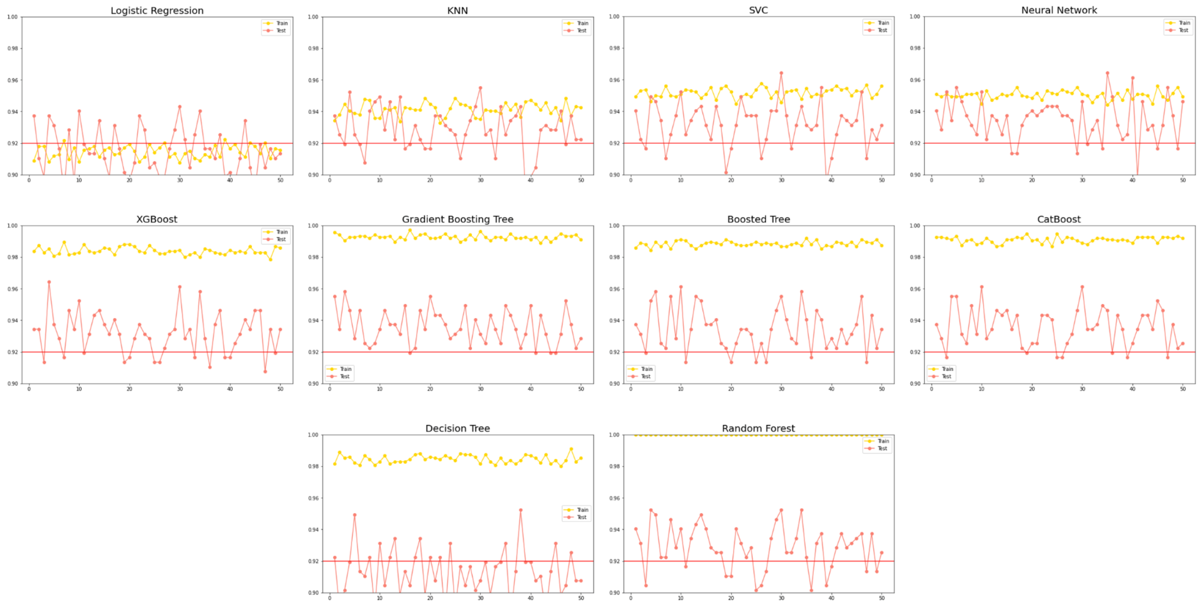
To predict the BS label, we use the parameters S, K, r, and tau in our dataset to run the classification model to classify whether the option is over or under the value predicted by the BS model.

We use the train-test split and cross-validation method as we do in the regression part. From Table 2 below, we find that the tree-based models perform better than others. Then, we tune the hyperparameters using Grid Search to optimize those models. The XGBoost model has the lowest classification error rate, which is 5.91%, after optimizing the model. However, after plotting out the classification score for the training and testing set with different random seeds, we find that the Gradient Boosting Tree is the most stable model. Therefore, after the trade-off between bias and variance, **we decide to choose Gradient Boosting Tree as our final model to predict BS value.**

[Table 2] Classification model performance comparison (classification error)

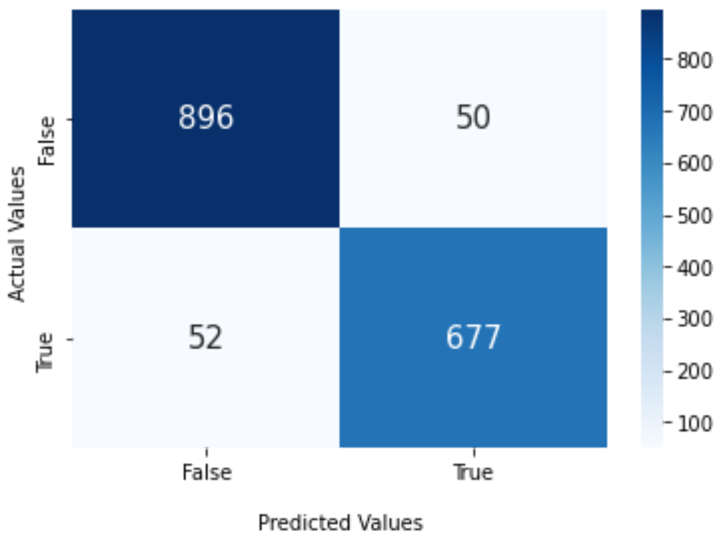
Classification Model	Train-test split	Cross-validation	Grid Search
Logistic Regression	6.87%	8.54%	N/A
Decision Tree	9.55%	8.72%	8.48%
Boosted Tree	5.67%	6.33%	6.27%
Gradient Boosting Tree	**5.07%**	**6.68%**	**6.08%**
XGBoost	6.27%	6.09%	5.91%
CatBoost	5.67%	6.09%	5.97%
Random Forest	6.27%	6.87%	6.27%
KNN	7.76%	7.82%	7.34%
SVM	5.67%	7.16%	6.45%
Neural Network	6.57%	7.16%	6.69%

[Figure 2] Classification accuracy for training and testing set



After choosing Gradient Boosting Tree as our final model, we plot out the confusion matrix to check type I error and type II error. Type I error means predicting “Under” as “Over”, so we would lose money because of overvaluing the asset. On the other hand, type II error means predicting “Over” as “Under”, so we would lose the opportunity to earn profit because of undervaluing the asset. In our model, the type I and type II error are quite balanced. We can change the decision threshold to rebalance the trade-off between type I and type II errors depending on the extent of risk aversion.

[Figure 3] Confusion matrix for Gradient Boosting Tree



Conclusion

In this report, we compare 8 competing option pricing models for regression analysis and 10 models for classification to determine which model has the best performance. Since we would like to implement models to accurately predict whether we should purchase the options, we value prediction better than interpretation. For regression analysis, we choose the XGBoost model as our final model and Gradient Boosting Tree model as our classification model after the trade-off between bias and variance.

To dive deep into the performance of machine learning models compared to the BS model, which is widely used to determine the value of an option, machine learning models can be constantly trained and adapted to the new data and market fluctuations. However, the BS model is a financial formula in which all parameters are fixed. In addition, there are multiple algorithms such as Tree-based, KNN, and Neural Network that can be applied to approach the problem with different dimensions. **Therefore, our machine learning models could outperform the BS model.**

In our machine learning model, we include four variables: strike price of an option(K), current stock price(S), time to expiration(tau), and an annual interest rate(r). The indicators provide us with a concrete foundation to value the call option. The strike price of the option evaluates the reward payoff and the key determinant of option value. The current asset value measures the price volatility and is one means of gauging the attractiveness of a stock. For time to maturity, it is used for the evaluation of time value and to check if the option contract expires within the desired period. And to assess ROI, we include the annual interest rate. With these variables, it is theoretically possible for option sellers to set rational prices for the options that they are selling. **Due to the nature of the four pillars in financial contracts, including these four indicators in our model help explain the variation of the call option pricing and level up our predictive power.**

Our models are used to predict S&P 500 stock options prices, and the data that we input is not included in Tesla's stock options data. Due to the limitation of features in our model and the data amount we have, we cannot predict Tesla's option value. In addition to that, because of Tesla's unique position in the industry and its CEO Elon Musk who is influential to Tesla's option value, it is difficult to apply our model to predict Tesla's option value. **Overall, our model is untransferable to predict Tesla's option values for stock prices.**

For future research, we would like to seek advice from domain experts for financial engineering and solidify our domain knowledge before building our model. Besides, we consider gaining more data to train our model and increase the prediction precision and upgrade the computational power simultaneously to deal with the big data we could possess.