USC Marshall School of Business

Predicting European Call Option Pricing using Machine Learning Models

Dan (Emily) Luo

Agenda

1. Introduction
2. Data Exploration & Cleaning
3. Model Exploration – Classification
4. Model Exploration – Regression
5. Conclusion

1/6

/5

Introduction

Introduction

- Training models to examine European call option pricing data on the S&P 500
- Build several machine learning models to perform our task:
  - Multiple Linear Regression, Logistic Regression, LDA, KNN, Decision Trees, Random Forest, Boosting
  - Regression and Classification
- Black-Scholes formula
  - An important tool for valuing European options
  - Takes into account several key variables, including the current price of the underlying asset, the option's strike price, the time until expiration, the risk-free interest rate, and the expected volatility of the underlying asset

1 Business Understanding

Interpretation is more important than prediction accuracy

- According to guest speaker YiYue
- Most of the models are not that accurate for predicting the stock market
- we use models to analyse and learn the history
- In this case, our interpretation is more important than prediction accuracy

Machine Learning Models might outperform Black-Scholes

Black-Scholes Formula Assumptions:

- Lognormal distribution of prices
- Constant volatility
- Continuous price movement

VS.

Machine Learning Models:

- Non-linearity
- Incorporating additional factors
- Adaptability

2/6

Data Exploration & Cleaning

2 Data Exploration & Cleaning

Quick Look in option_train dataset

- Value (C): Current option value
- S: Current asset value
- K: Strike price of option
- r: Annual interest rate
- tau: Time to maturity (in years)
- BS: Use the Black-Scholes formula to get C_pred.
  - Over: C_pred - C > 0 → Overestimated
  - Under: C_pred - C < 0 → Underestimated

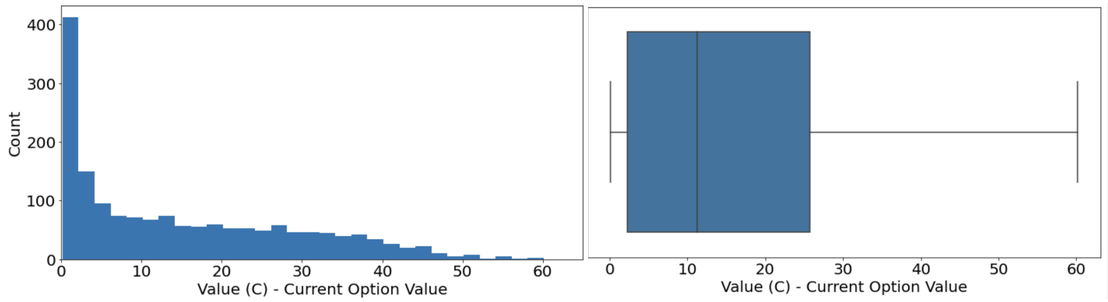
	Value	S	K	tau	r	BS
count	1678.000000	1679.000000	1678.000000	1679.000000	1680.000000	1680
unique	NaN	NaN	NaN	NaN	NaN	2
top	NaN	NaN	NaN	NaN	NaN	Under
freq	NaN	NaN	NaN	NaN	NaN	946
mean	15.068709	464.402535	438.241955	0.437519	0.030235	NaN
std	14.040023	973.652179	23.408989	7.057555	0.000557	NaN
min	0.125000	0.000000	375.000000	0.003968	0.029510	NaN
25%	2.255001	433.863864	420.000000	0.119048	0.029820	NaN
50%	11.190967	442.634081	440.000000	0.202381	0.030130	NaN
75%	25.747434	447.320414	455.000000	0.285714	0.030540	NaN
max	60.149367	40333.000000	500.000000	250.000000	0.031880	NaN

Original Train Data:1680 rows × 6 columns

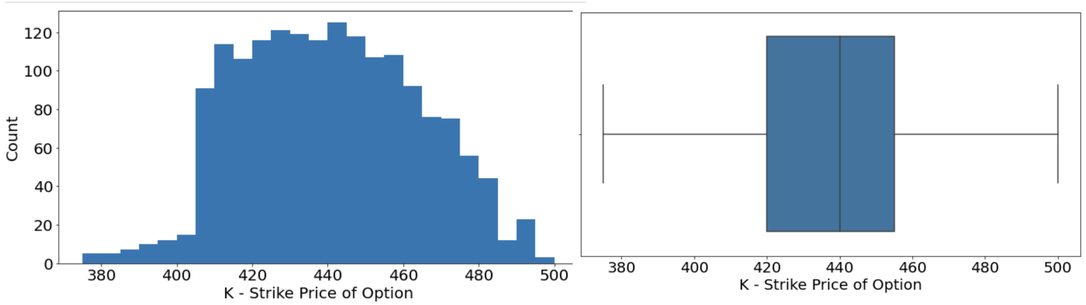
USC Marshall School of Business

Checking for missings and erroneous

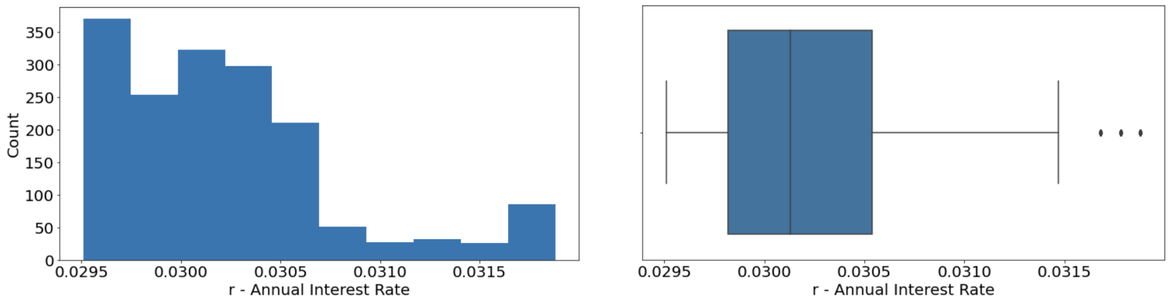
- Value (C)
  - No erroneous data
  - 2 missing values
  - Use sklearn.impute.SimpleImputer - Mean Imputation



- K
  - No erroneous data
  - 2 missing values
  - Use sklearn.impute.SimpleImputer - Mean Imputation



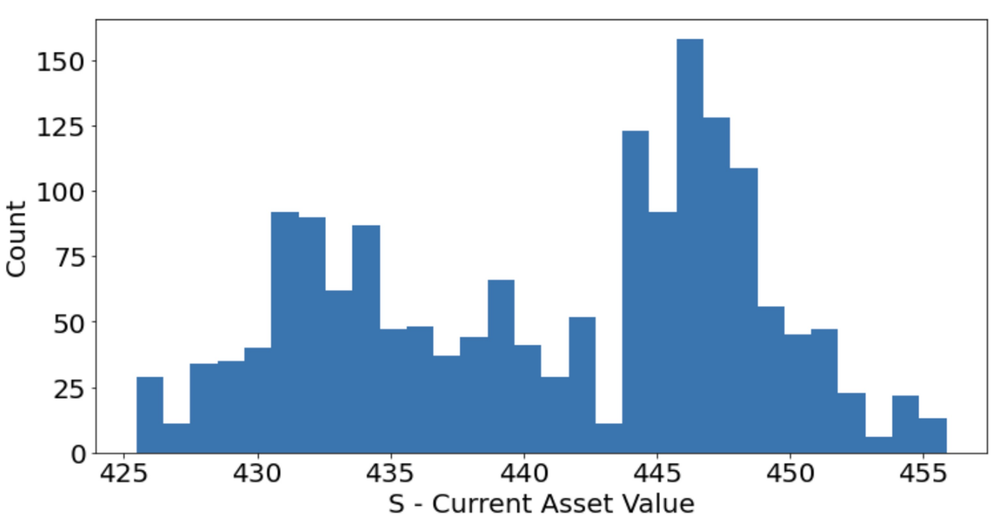
- r
  - No erroneous data
  - No missing values

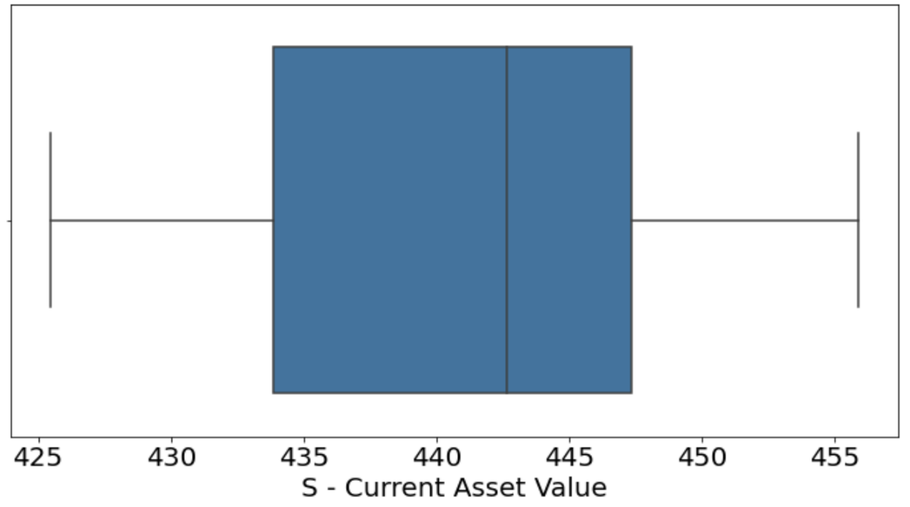


USC Marshall School of Business

Checking for missings and erroneous

- S
  - Max: 40,333 Min: 0 → Two Erroneous values
  - 1 missing values
  - Exclude two erroneous values
  - Use sklearn.impute.SimpleImputer - Mean Imputation



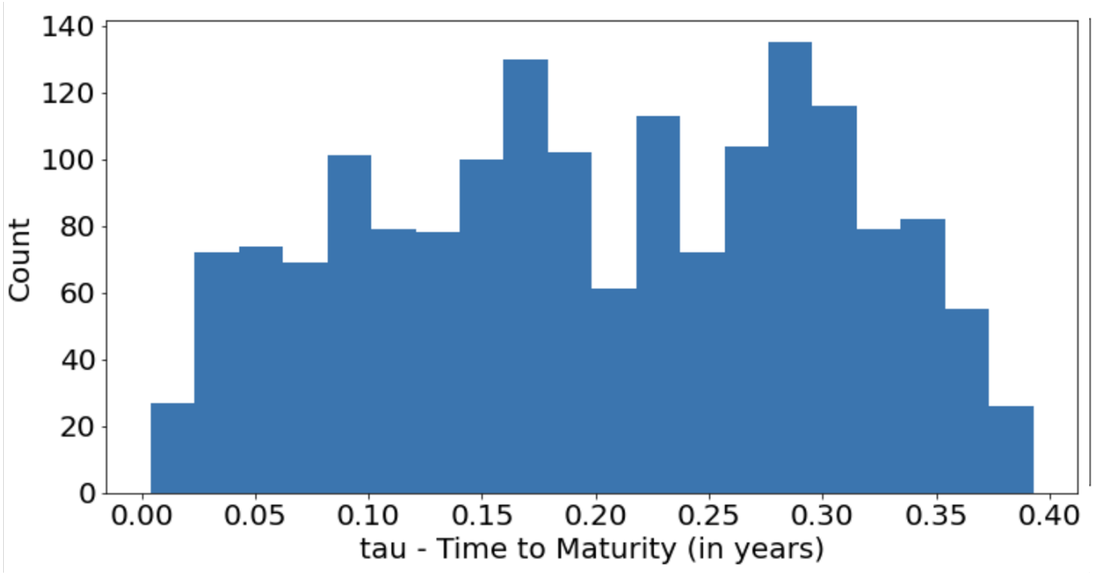


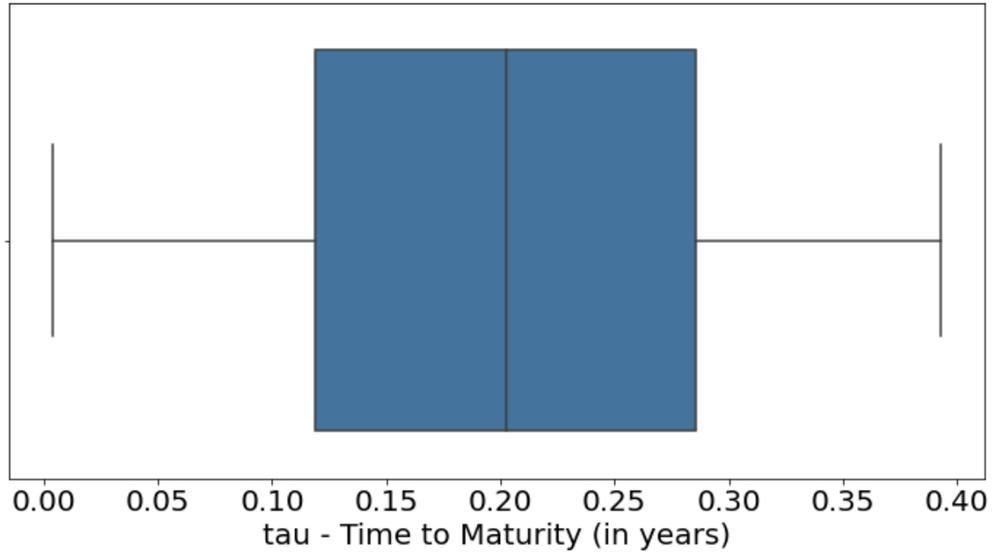
Data Exploration & Cleaning

Checking for missings and erroneous

- tau
  - Sort values → Two Erroneous value
  - 1 missing values
  - Exclude values greater than 1 (year)
  - Use sklearn.impute.SimpleImputer - Mean Imputation

```
12      250.000000
33      146.000000
1387      0.392857
904       0.392857
1538      0.392857
             ...
1457      0.007937
1016      0.003968
1337      0.003968
1267      0.003968
292            NaN
Name: tau, Length: 1678, dtype: float64
```

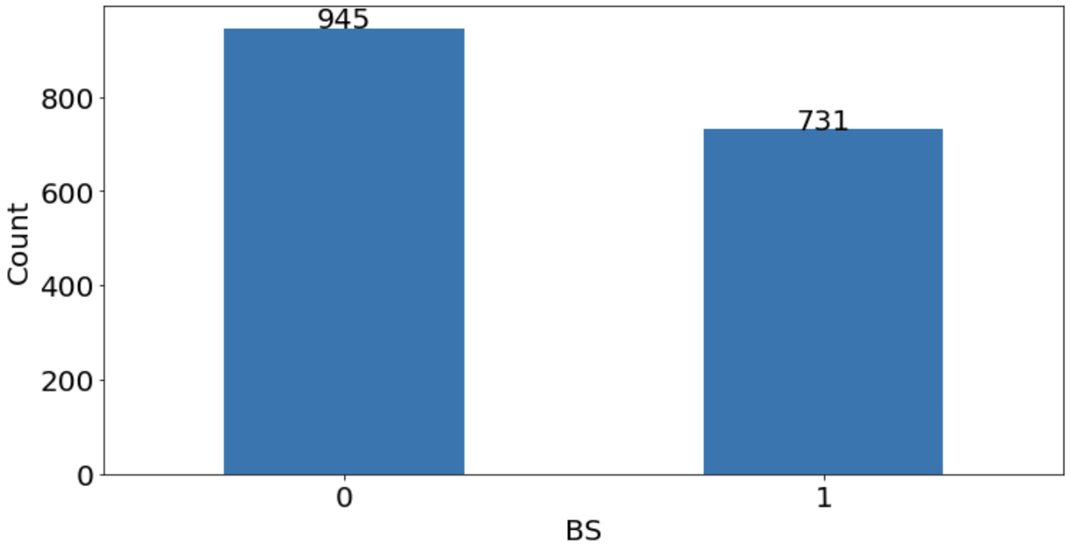




2 Data Exploration & Cleaning

Checking for missings and erroneous

- BS
  - No erroneous data
  - No missing values
  - Change “Over” to 1 and “Under” to 0.
  - There are 945 records show “Under” and 731 records show “Over”



2 Data Exploration & Cleaning

Option_train dataset after exclusion and imputation

	Value	S	K	tau	r	BS
count	1676.000000	1676.000000	1676.000000	1676.000000	1676.000000	1676.000000
mean	15.092467	440.898587	438.210907	0.202018	0.030234	0.436158
std	14.038778	7.527344	23.400683	0.099730	0.000557	0.496055
min	0.125000	425.472331	375.000000	0.003968	0.029510	0.000000
25%	2.246251	433.863864	420.000000	0.119048	0.029820	0.000000
50%	11.250000	442.525366	438.239857	0.202381	0.030130	0.000000
75%	25.766604	447.320414	455.000000	0.285714	0.030540	1.000000
max	60.149367	455.880619	500.000000	0.392857	0.031880	1.000000

- There are 1,676 records. All 4 predictors should be included in the prediction.
- Use the cleaned option_train dataset to build statistical/ML models with
  - Value as the response variable (Regression problem)
  - BS as the response Variable (Classification Problem)

3/6

Model Exploration – Regression

4 Model Exploration – Regression

Linear Regression - MLR

- k-Fold Cross-validation
  - n_splits=10, random_state=2, shuffle=True; scoring = ‘r2’
- R-square = 0.91

KNN

- Standardized the features before model training
- n_neighbors = 5 has a higher R-Square than n_neighbors = 10
- R-square = 0.976 < 0.981

Decision Tree

- Use ccp_alphas to find the optimal alpha value (hyperparameter) for this model
- R-square = 0.992

*Note: All models use 10-fold CV and random_state=2

4 Model Exploration – Regression

XGBoost

- Use for loop to check training and test score for different combinations of hyperparameters
- Hyperparameter Tuning
  - Objective: Define optimization objective
  - Learning_rate: Control the speed at which the models learn
  - max_depth: Details to define a tree
  - alpha: L1 regularization term

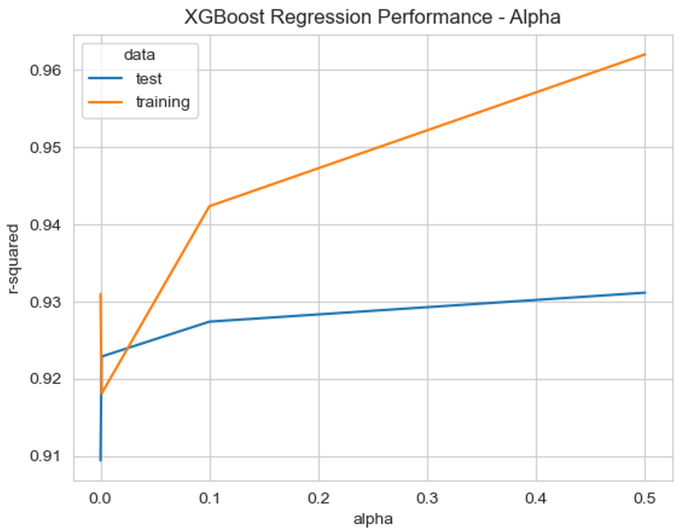
```
# Initiate values
alphas_list = [0,0.001,0.1,0.5]
max_depths_list = [1,2,3,4,5]
learning_list = [0.05,0.1,0.3]
```

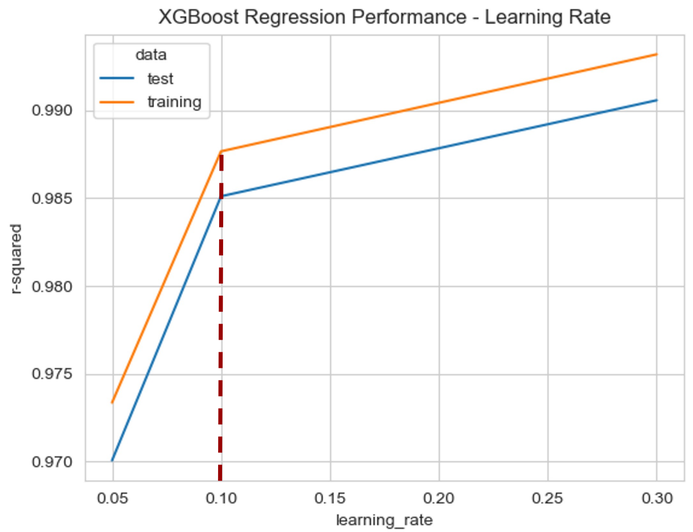

*Note: All models use 10-fold CV and random_state=2

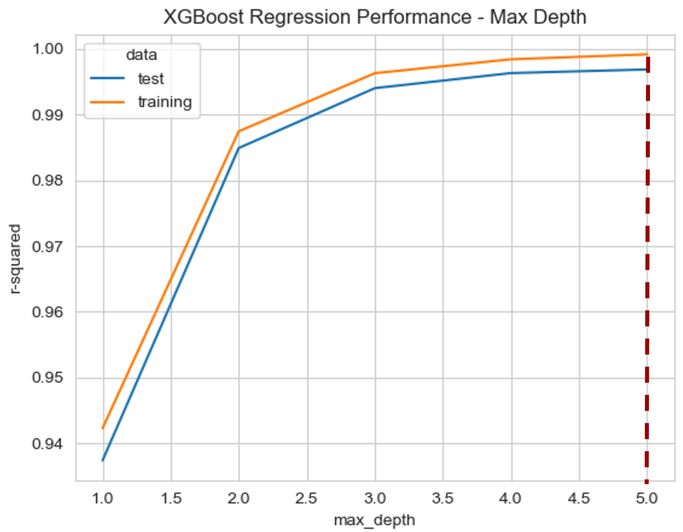
Model Exploration – Regression

XGBoost

- Overfitting or Underfitting? - No.
- The optimal parameters {max_depth=5, reg_alpha=0, learning_rate=0.1}
- R-square = 0.997







*Note: All models use 10-fold CV and random_state=2

Model Exploration – Regression

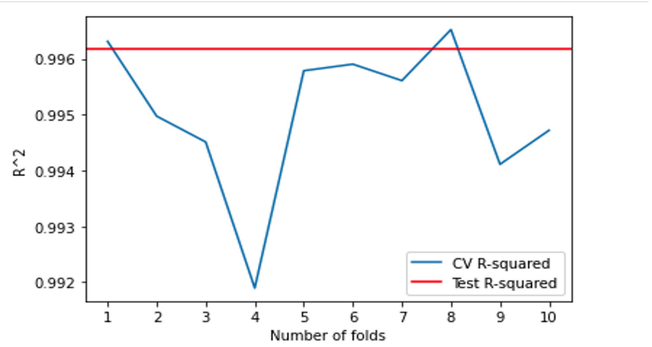
Random Forest

- Use GridSearchCV for hyperparameters tuning to find the optimal set of hyperparameters for the model

```
# Define the hyperparameters to search through
param_grid_reg = {
    'n_estimators': [50, 100, 200, 300],
    'max_depth': [None, 5, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}

param_grid_cla = {
    'n_estimators': [50, 100, 200, 300],
    'max_depth': [None, 5, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}
```

- Hyperparameter {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 300}
- R-square = 0.996



*Note: All models use 10-fold CV and random_state=2

4/5

Model Exploration – Classification

3 Model Exploration – Classification

Logistic Regression

- Classification Error = 0.092

Linear Discriminant Analysis (LDA)

- Classification Error = 0.088

K-Nearest Neighbors (KNN)

- Classification Error = 0.070
- n_neighbors = 3

Decision Tree

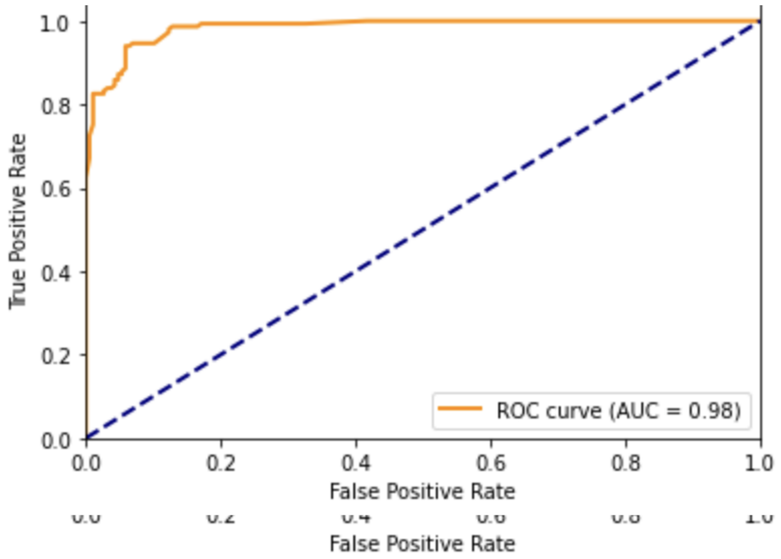
- Classification Error = 0.077

*Note: All models use 10-fold CV and random_state=2

Model Exploration – Classification

Random Forest

- Use GridSearchCV to find the optimal set of hyperparameters for the model
- Hyperparameter {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}
- Classification Error = 0.068

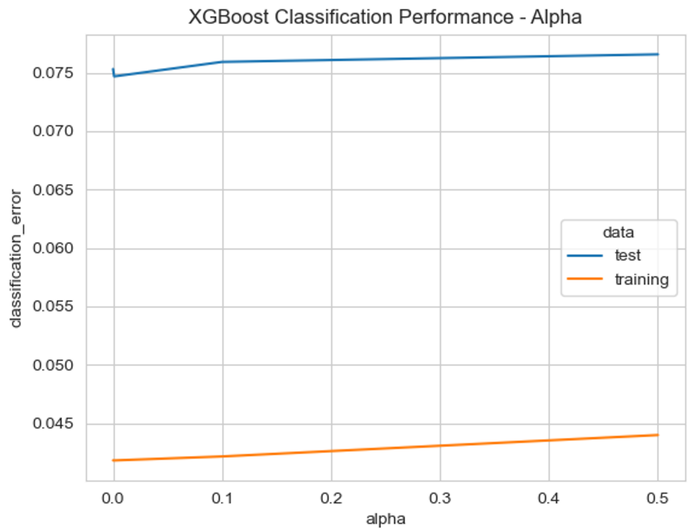


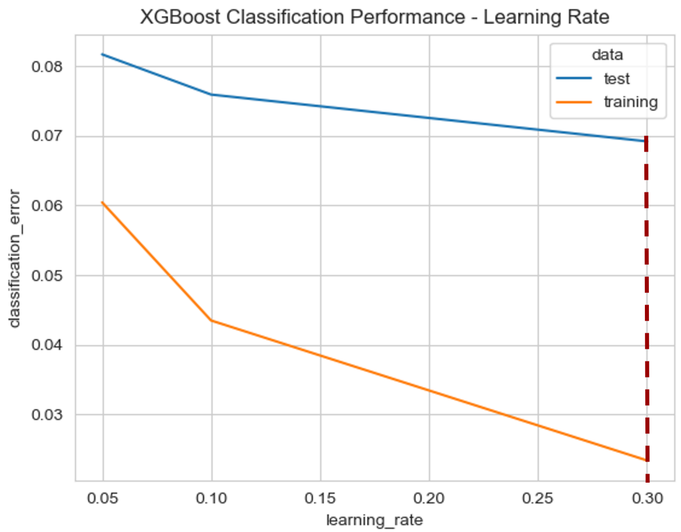
*Note: All models use 10-fold CV and random_state=2

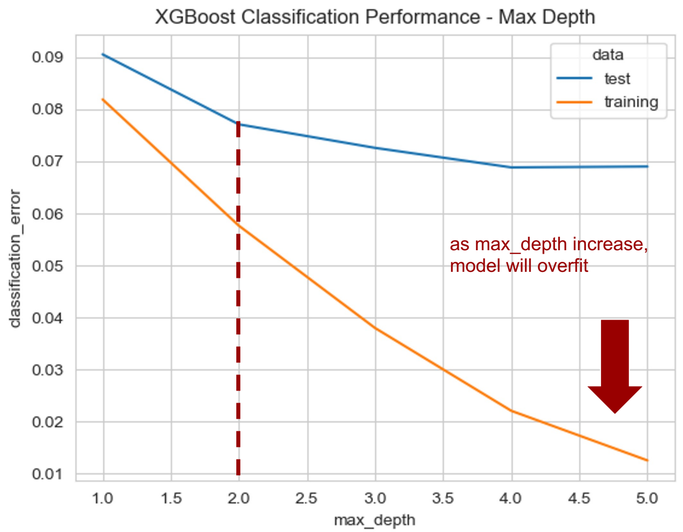
3 Model Exploration – Classification

XGBoost

- Overfitting or Underfitting? No.
- The optimal hyperparameters {max_depth=2, reg_alpha=0.001, learning_rate=0.3}
- Classification error = 0.072

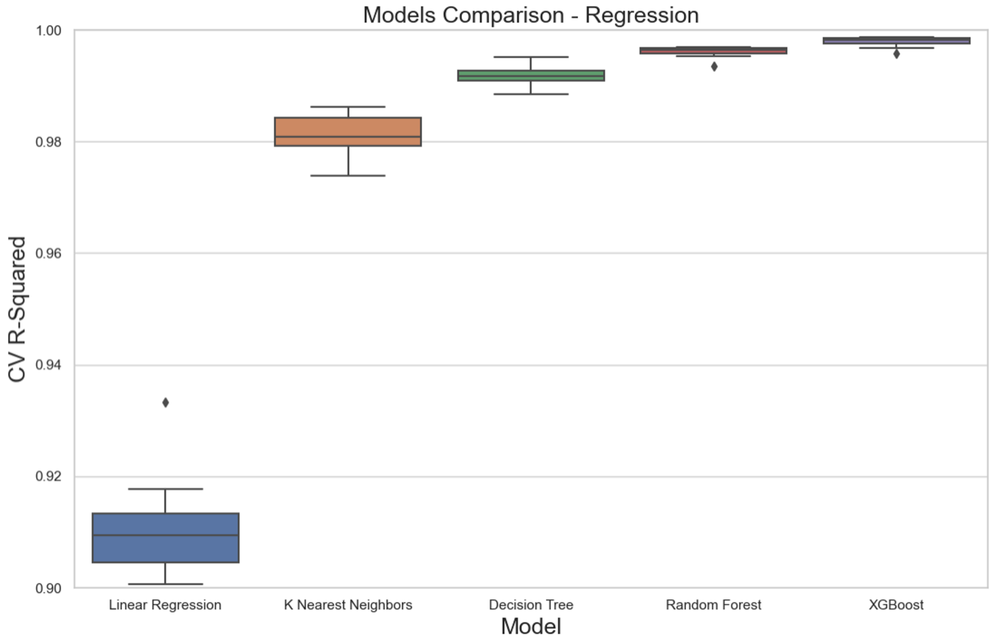


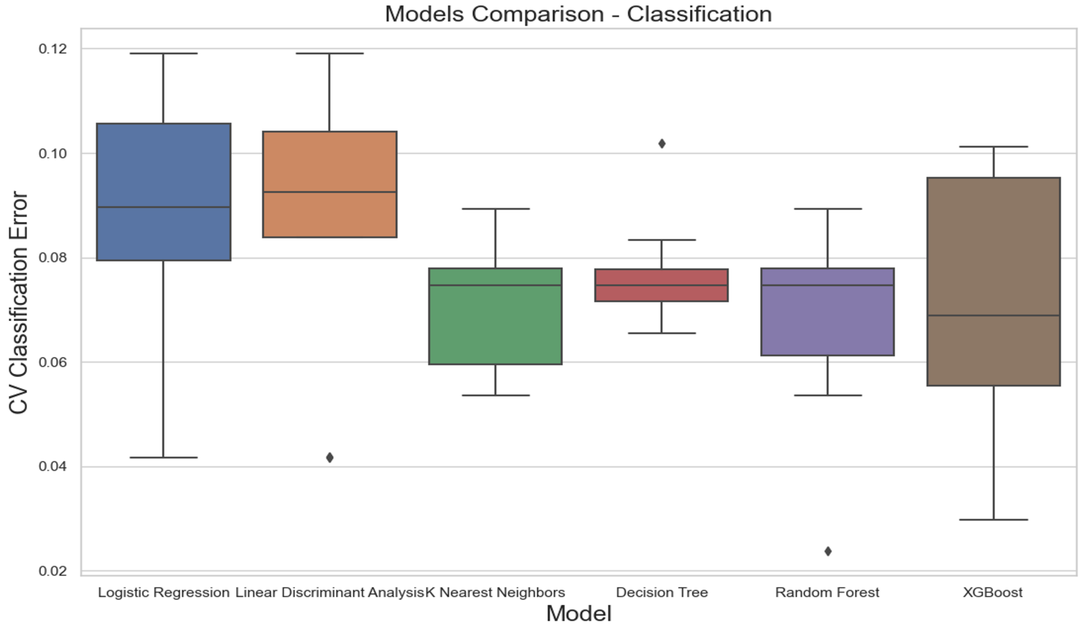




*Note: All models use 10-fold CV and random_state=2

3 Model Comparison – Regression & Classification





- As you can see the chart show above, Random Forest model has a better performance compared with others

*Note: All models use 10-fold CV and random_state=2

5/5

Conclusion

5 Conclusion

Final Model

- **Random Forest** is chosen for classification and regression problem
- We use **Random Forest** model to make prediction for Value (C) and BS on the 1,120 options in the test dataset.

NOT using our train model to predict option values for Tesla stocks

- Option values are influenced by a variety of factors, such as industry trends, economic conditions or company-specific news
- Elon Musk has a strong influence on the company's stock price and investor sentiment towards the company