DSO 530

Options Pricing Summary Report

May 2, 2024

Group 16 Authors:
Gabriele Monti
Matteo Mennini
HyeongSeok (Nick) Lee
Farhan Sikder
Seongjae (John) Liew
Mihir Sabnis

Executive Summary

The Options Pricing Group Project analyzes the European call option pricing data, comparing the traditional Black-Scholes methodology and modern statistical and machine learning techniques.
The project aims to explore alternative approaches to option valuation without prior reliance on the Black-Scholes formula.

$$C_{pred} = S\Phi(d_1) - Ke^{-rT}\Phi(d_2)$$

The training dataset consists of 5,000 options with key variables including **current option value (C)**, **asset value (S)**, **strike price (K)**, **annual interest rate (r)**, **time to maturity (tau)**, and **Black-Scholes predicted value (BS)**. The test dataset, comprising 500 options, lacks actual option values and Black-Scholes predictions.

The project methodology encompasses the following steps:

1. **Data Exploration and Cleaning:**

   We meticulously examined the training dataset, identifying its structure and key variables, and ensuring data integrity by:

   - **Missing Data and Transformations**: Transforming categorical labels into binary format ('1' for 'Over', '0' for 'Under') to simplify computations.
   - **Outliers and Scaling**: Utilizing RobustScaler to effectively manage outliers, crucial for the skewed 'value' variable.
2. **Model Development**

   We explored both regression and classification techniques:

   - **Regression Models**: Employed linear regression, decision trees, support vector regression (SVR) and stacked model to predict actual option values.
   - **Classification Models**: Used logistic regression, decision trees, and SVC, and stacked model to determine if options were priced 'Over' or 'Under' according to the Black-Scholes model.
3. **Evaluation and Model Selection**

   We critically assessed model performance using:

   - **Regression Evaluation**: Metrics like MSE, R-squared, and plots comparing actual vs. predicted values assessed regression models' accuracy.
   - **Classification Evaluation**: Accuracy scores and error rates evaluated classification models' ability to correctly categorize options.
4. **Prediction on Test Data**

   Selected models were applied to a test dataset of 500 options:

   - **Model Application**: Predicted option values and their categorization relative to Black-Scholes estimates.
   - **Comparison and Submission**: Predictions were compiled and compared against actual market values to assess performance improvement over traditional methods.
5. **Summary of Insights and Business Applicability**

   The project demonstrated the effectiveness of modern statistical and machine learning techniques in option valuation, offering deep insights into option pricing dynamics and enhancing financial decision-making. The exploration of various modeling approaches underlined the potential of these techniques in real-world applications.

Data Exploration and Cleaning

In the initial phase of the project, we conducted a thorough exploration and cleaning of the provided dataset to ensure its quality and reliability:

1. **Data Investigation**: We began by inspecting the training dataset to understand its structure and variables.
2. **Missing Data and DataTransformation**: The dataset did not contain missing values. We changed the 'Over' and 'Under labels into binary '1' and '0'.
3. **Exploratory Data Analysis**: We checked the distributions of each variable to understand whether we had strong outliers and created a correlation heatmap(figures 1.1 - 1.7).
4. **Scaling and Outliers**: Each predictor variable had a different scale, "tau" and the dependent variable "value" was heavily skewed to the right variable "value". For some models, outliers can affect the model generalization and final prediction. We utilized visualization tools to identify outliers and used the RobustScaler function to scale and manage outliers. RobustScaler manages outliers by centering the data around the median and scaling it based on the interquartile range (IQR).

Model Development

Our approach to model development involved considering a diverse range of regression and classification techniques to predict option values and Black-Scholes estimates. Key considerations during model selection included:

- **Regression Models:** We evaluated linear regression, tree-based models, and SVR to predict option values (C) based on predictor variables.
- **Classification Models**: For predicting Black-Scholes values (BS), we explored logistic regression, decision tree-based models, SVC, and a stacked model. These models enabled us to categorize options as either "Over" or "Under" (encoded in 1 and 0) predicted based on the Black-Scholes formula.

Model Evaluation

The Evaluation of the developed models was crucial to assess their performance and identify the most accurate and robust approaches. We employed a range of evaluation metrics tailored to the specific tasks:

- **Regression Evaluation:** Models predicting option values (C) were evaluated using metrics such as mean squared error (MSE), R-squared, and mean absolute error (MAE) to quantify predictive accuracy and goodness of fit. We also visualized feature importance.
- **Classification Evaluation:** For Black-Scholes prediction, classification models were assessed using classification accuracy score (1 - classification error) to measure the model's ability to correctly classify options as "Over" or "Under" predicted. We also visualized feature importance, confusion matrices, and ROC curves.

Regression Models

Linear Regression

We first used linear regression as a baseline model. It models the relationship between the dependent variable and one or more independent variables by fitting a linear equation to observed data.

After splitting the training dataset 80/20 and using the 20% set for validation, we got the following results:

R-Squared: 0.928; Root Mean Squared Error (RMSE): 33.967; Mean Squared Error (MSE): 1153.749

The scatter plot of actual vs. predicted values shows a strong linear relationship, especially in the middle range of values (figures 2.1 - 2.2). However, it seems less accurate at the extreme ends, particularly with higher actual values. Linear regression assumes a linear relationship between the dependent and independent variables, which might not hold where interactions and nonlinear relationships can exist.

Random Forest Regression

Random Forest combines multiple decision trees to improve prediction accuracy and prevent overfitting. By averaging the outputs of individual trees, it provides a prediction that balances out errors, managing intricate feature relationships and reducing variance.

For hyperparameter tuning, we tested different configurations to optimize for R^squared. We split the dataset for training and validation, using 20% for validation (figure 3.1). Results of the Random Forest regressor include (figure 3.1):

R-squared: 0.996; RMSE: 7.519

While Random Forest regression models are highly effective, they can be computationally intensive and may experience overfitting. Their complexity may also pose challenges in interpretability.

XGBoost For Regression

XGBoost uses decision trees as base learners and optimizes them using gradient-boosting frameworks, being particularly beneficial for regression tasks because it can model non-linear relationships and interactions between variables very effectively. It is also robust against overfitting due to its regularized model formalization to control complexity, and it handles various types of data.

For hyperparameter tuning we selected estimators=100, learning rate=0.1, max depth=5. We used 5-fold cross-validation. Using 20% for validation, we achieved the following results(figure 4.1):

RMSE: 6.854; R^2 Score: 99.706% Cross-Validation RMSE: 7.219; Cross-Validation R^2 Score: 99.661%

Cross-validation results confirm the model's robustness and its ability to generalize well on unseen data, with slight variations indicating good stability across different subsets of the dataset. Despite its high performance, XGBoost can be computationally expensive, especially with large datasets and when tuning multiple hyperparameters.

Support Vector Regression (SVR)

SVR with randomized search CV aims to predict continuous target variables by identifying the optimal hyperplane that maximizes the margin between data points and the regression line. It provides robustness against outliers but might not be the most useful model for this data as it is better at capturing complex non-linear relationships and high-dimensional data.

We performed a randomized search for hyperparameters with 5-fold cross-validation and 50 iterations. The best hyperparameters were kernel = rbf, gamma = 0.1, epsilon = 0.0424, and C = 88.428 which gave us the following results (figure 5.1): **R^2 Score: 99.721%; RMSE: 6.675.**
This model performed marginally worse than other decision tree-based regression models.

Gradient Boosting Regression

Gradient Boosting with Grid Search enhances predictive models by optimizing a differentiable loss function, sequentially building decision trees to correct prior errors.

For hyperparameter tuning in our project, we utilized Grid Search with parameters set for the number of trees (estimators=200), learning rate (0.1), and max depth (5), employing 3-fold cross-validation to refine the model's performance. Using the 20% for validation, the Gradient Boosting model delivered the following results(Figure 6.1):

RMSE: 7.22; R^2 Score: 99.758%; Cross-Validation RMSE: 6.854; Cross-Validation R^2 Score: 99.76%

Cross-validation results confirm the model's robustness and its generalization capability across unseen data, demonstrating consistency and stability across different subsets of the dataset.

Classification Models

Logistic Regression For Classification

Logistic regression is employed as a baseline classification model because of its effectiveness for binary classification tasks. It models the probability of a binary response based on one or more predictor variables.

After splitting the training dataset 80/20 and using the 20% set for validation, we achieved **88.6% accuracy** (figure 7.1-7.3).

While logistic regression has performed well, it assumes linearity between the log odds of the dependent variable and each independent variable, which might not capture more complex relationships or interactions between variables.

Random Forest Classification

Random Forest excels in classification by using a multitude of decision trees to vote on the outcome, thereby providing the class that is the mode of the classes predicted by individual trees. This method corrects for any decision tree's tendency to overfit.

After the 80/20 training/testing split. The Random Forest classifier gave an **accuracy of 93.3%** (figures 8.1-8.3).

Despite its advantages, the RF classifier may imply reduced transparency due to model complexity.

XGBoost Classifier

XGBoost uses gradient-boosted decision trees designed for speed and performance. XGBoost is suitable for classification because it can handle imbalanced datasets well, efficiently differentiate between classes, and provide probabilities of class memberships.

For hyperparameter tuning, we selected estimators=100, learning_rate=0.1, and max_depth=5. We also used 5-fold cross-validation. After the usual 80/20 split for training and validation, we achieved the following results (figures 9.1-9.3):

Classification Accuracy: 93.8%; Cross-Validation Accuracy: 93.35%; Error Rate: 6.2%

Support Vector Classifications (SVC)

SVR with randomized search CV aims to classify the target variable into binary classes by finding the optimal hyperplane that maximizes the margin between data points and the regression line. It provides robustness against outliers but might not be the most useful model for this data as it is better at capturing complex non-linear relationships and high-dimensional data. It is also susceptible to class imbalance.

We performed a randomized search for hyperparameters with 5-fold cross-validation and 50 iterations. The best hyperparameters were kernel = rbf, gamma = auto, and C = 100 which gave us the following results (figures 10.1-10.3): **Classification Accuracy: 91.60%; Test Error: 8.39%**

This model performed worse than all the other complex decision tree-based models.

Gradient Boosting Classification

Gradient Boosting with GridSearch for classification iteratively refines a series of decision trees using a differential loss function, with GridSearch employed to optimize hyperparameters like learning rate, tree depth, and number of estimators to maximize classification accuracy

For hyperparameter tuning we selected 200 estimators, a learning rate of 0.1, and a maximum depth of 5 through Grid Search, applying 3-fold cross-validation to determine the best configuration.
Results show (figures 11.1-11.3): **Classification Accuracy: 93.2%; Error Rate: 6.8%**

Ensemble Stacked Regression and Classification Models

Stacking is an advanced ensemble method that enhances the accuracy of predictions by combining multiple models via a meta-learner. It is particularly beneficial for complex predictive tasks as it leverages the strengths of various base models and mitigates their weaknesses. We decided to use a stacking model as we wanted to reduce overfitting while still trying to improve the overall model performance.

For hyperparameter tuning, we chose our Random Forest, SVM, and Gradient Boosting models as base learners for regression and classification and used a meta-learner for final predictions. 80/20 split was used, dedicating 20% for validation. This setup allowed us to optimize the meta-learner's performance based on the outputs from the base models (figures 12.1;13.1-13.4).

R^2 Score: 99.75%; Accuracy: 94.1%; Cross-Validation R^2 Score: 99.72%; Cross-Validation Accuracy: 94.0%

These metrics indicate that the stacked model is also extremely accurate, achieving a **94.1% classification accuracy rate** for Black Scholes classification. This was our most effective model to implement for classification.

Predicting on Test Data

Following model development and evaluation, we applied the selected models to predict option values and Black-Scholes estimates for the 500 options in the test dataset (option_test_nolabel.csv). The predictions were compiled into a test data CSV file for submission, enabling comparison with the actual values to assess model performance.

Conclusion and Business Applicability Considerations

Our final approach used the stacked ensemble model to predict both option values and Black-Scholes estimates. We chose this approach due to its demonstrated effectiveness in capturing complex relationships and improving predictive accuracy.

For option pricing, prediction accuracy, and interpretability are crucial, although it depends on the model use. In high-frequency trading, accuracy is prioritized, while in regulatory settings where transparency is vital, interpretability might be prioritized. For BS classification, prediction accuracy might be more important as we are trying to beat the Black-Scholes formula. Accurate predictions are essential for informed decision-making, interpretability enables stakeholders to understand the underlying factors driving the model's predictions.

Machine learning models may outperform the Black-Scholes formula because they capture complex nonlinear relationships and adapt to changing market dynamics. The Black-Scholes formula relies on simple assumptions, machine learning techniques can leverage richer datasets and learn from historical patterns.

From a business perspective, including all four predictor variables (asset value, strike price, interest rate, time to maturity) in option pricing predictions may be justified to capture the full range of factors influencing option values. Variable selection should be guided by empirical evidence and domain knowledge.

We would not use these models directly on option values for Tesla Stock. We do not know how the model would generalize to Tesla stock options as they were not trained on market dynamics specific to Tesla. Tesla is also quite a volatile stock, so our model would need to be trained and the performance tested on Tesla stock options before making such a decision.

Appendix

Data Exploration Analysis

Figure 1.1 - 1.4: Distributions of the predictor variables S, K, tau, and r

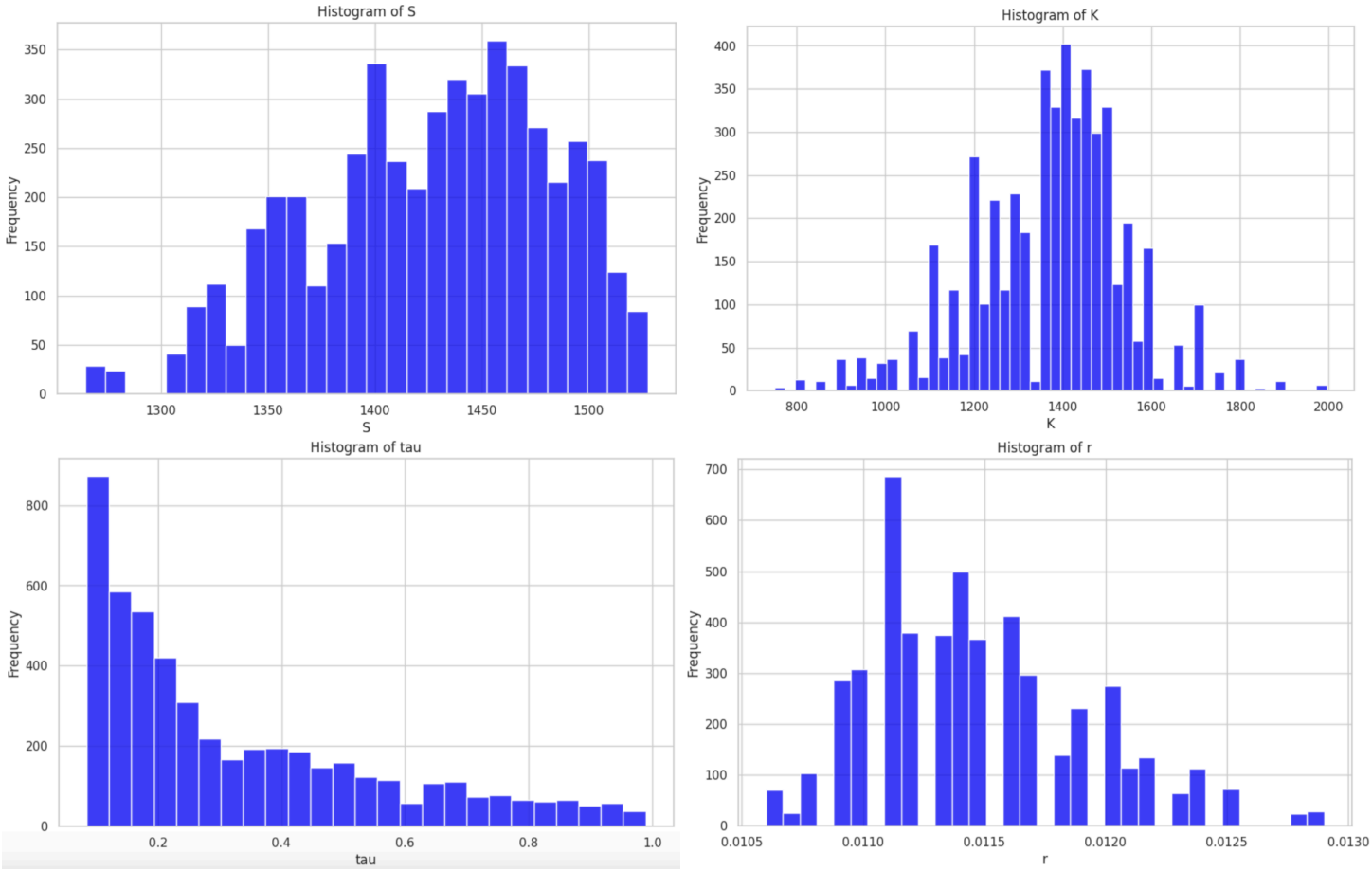


Figure 1.5 - 1.6: Distributions of the dependant variables ‘Options Value’ and ‘Black Scholes’

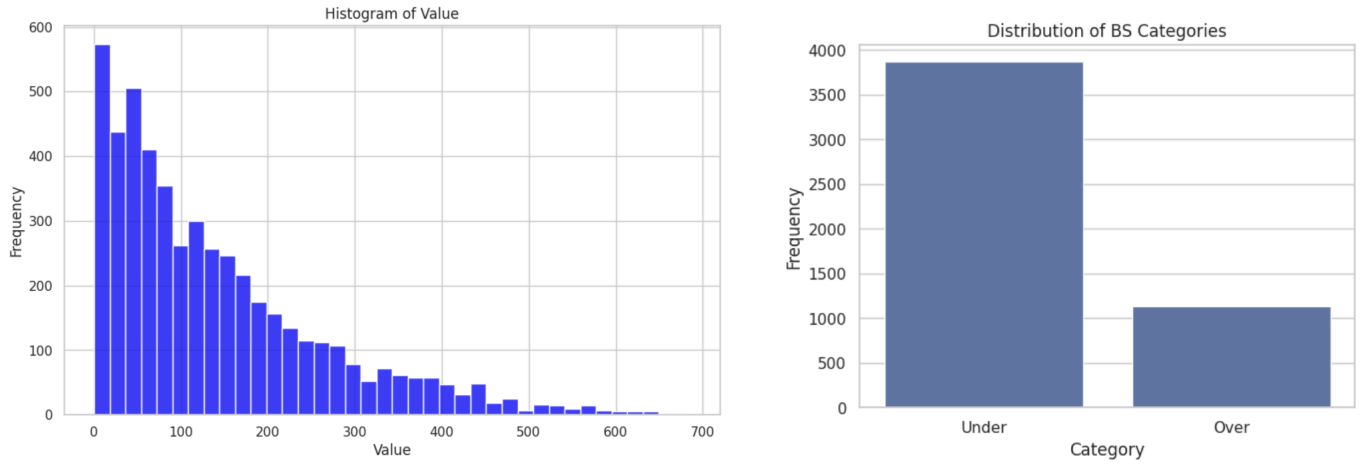
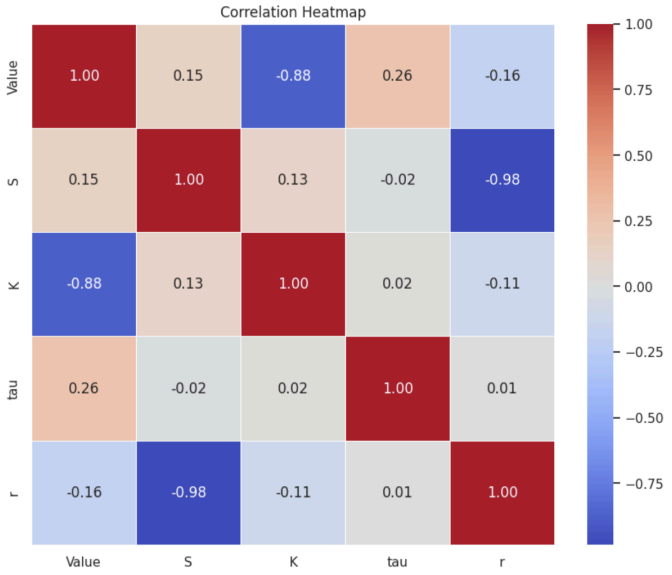


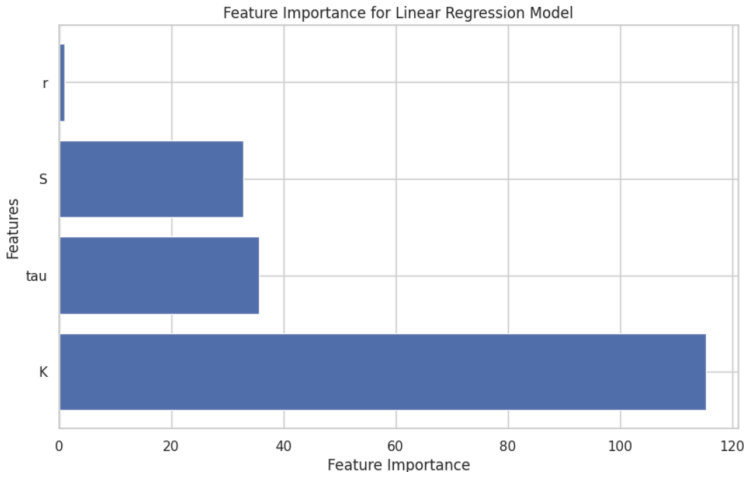
Figure 1.7: Correlation heatmap



Linear Regression

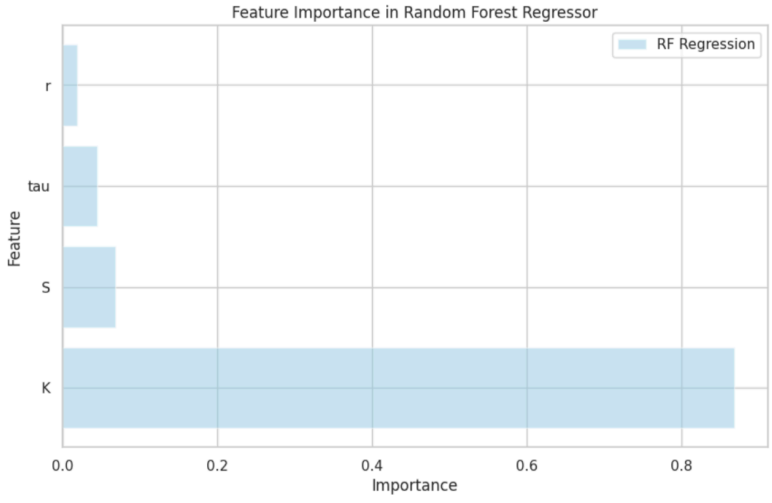
Figure 2.1: Actual vs Predicted Value Graph

Figure 2.2: Multiple Linear Regression Feature Importance



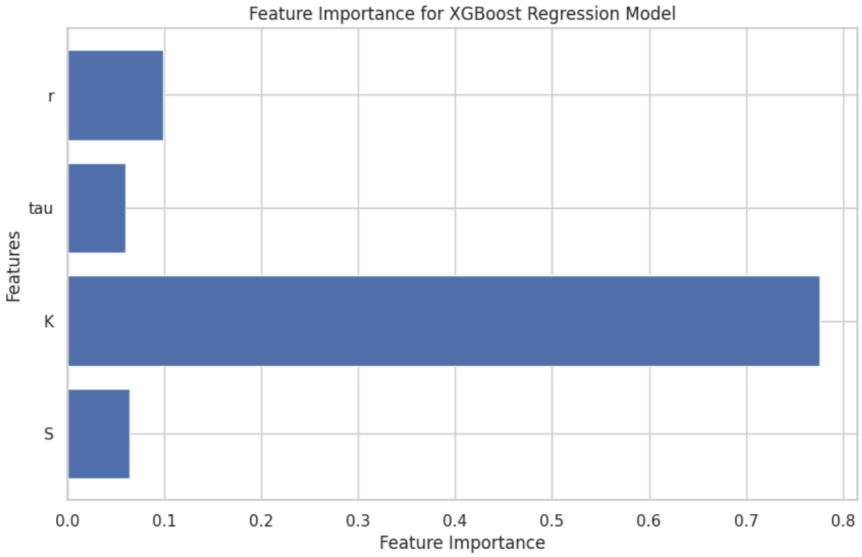
Random Forest Regression

Figure 3.1: Random Forest Regression Feature Importance



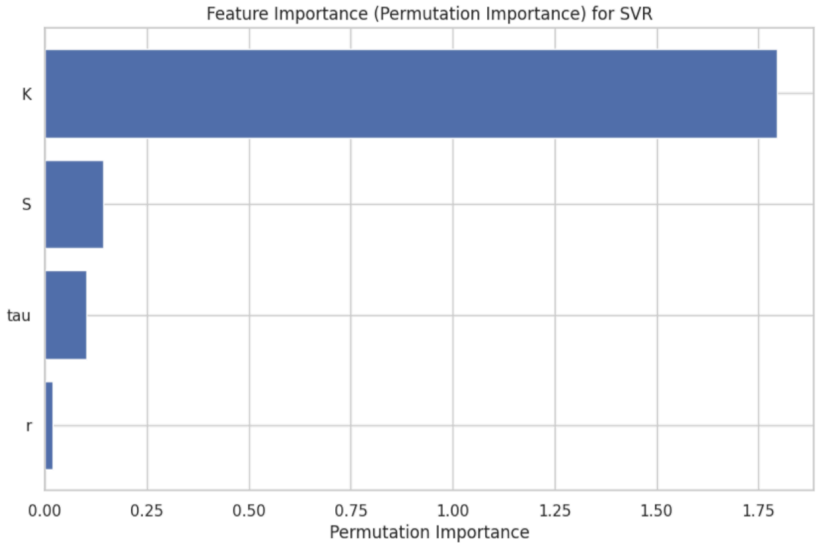
XGBoost Regression

Figure 4.1: XGBoost Regression Feature Importance



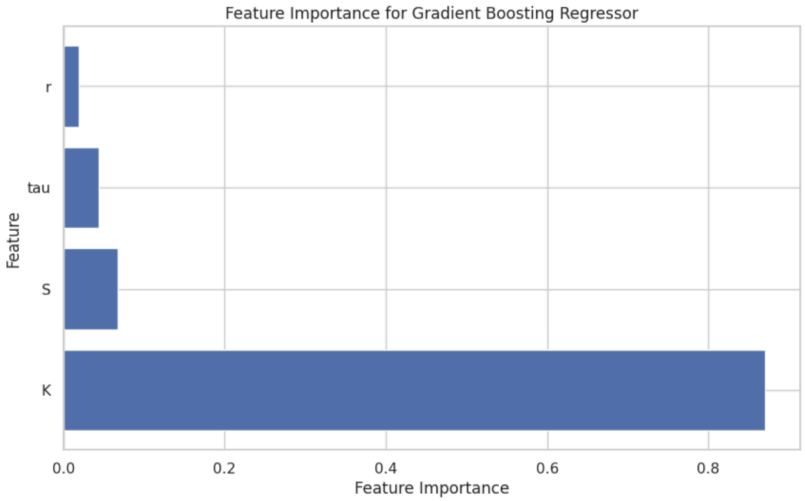
Support Vector Regression

Figure 5.1: SVR Feature Importance



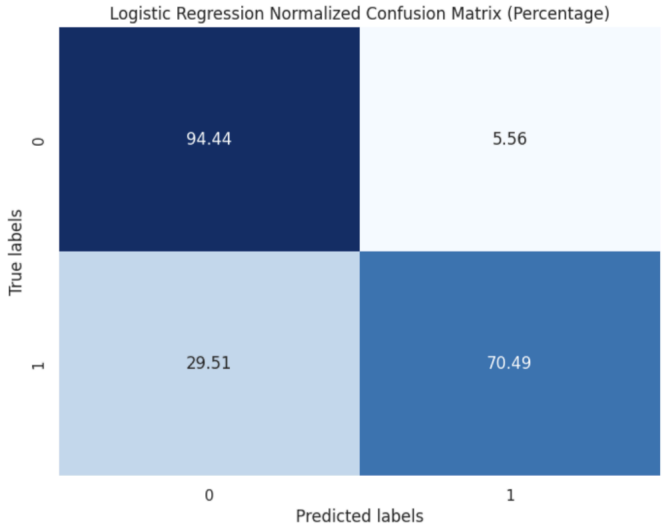
Gradient Boosting Regression

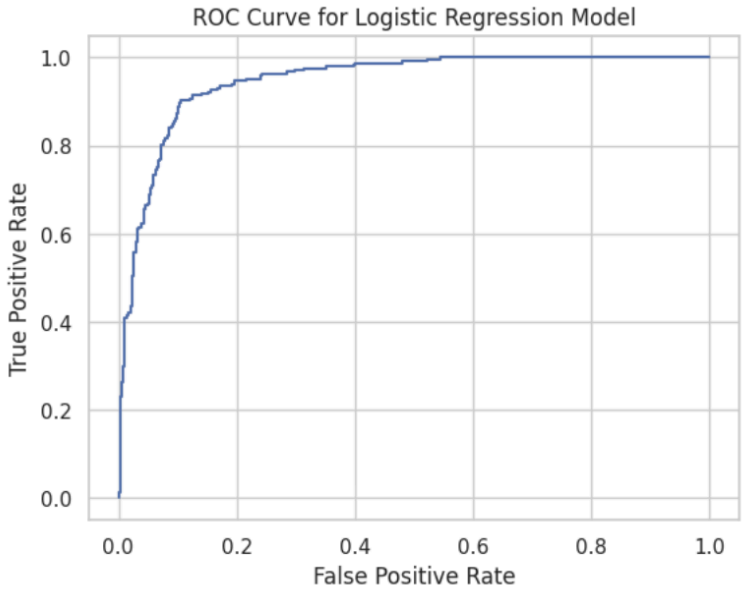
Figure 6.1: Gradient Boosted Regression Feature Importance

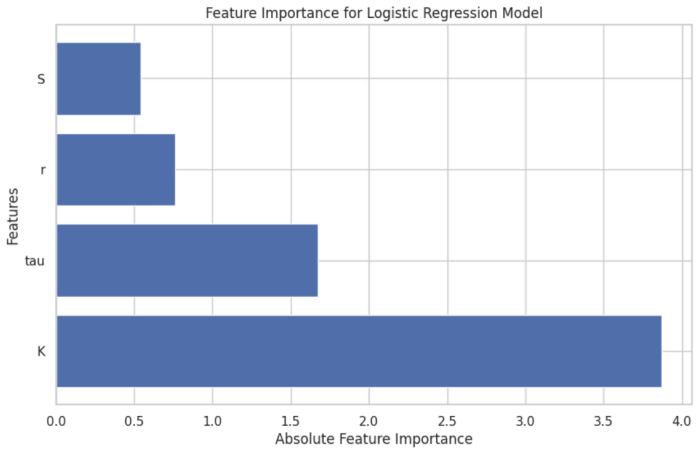


Logistic Regression for Classification

Figure 7.1: Normalized Confusion Matrix
Figure 7.2: ROC Curve
Figure 7.3: Feature Importance

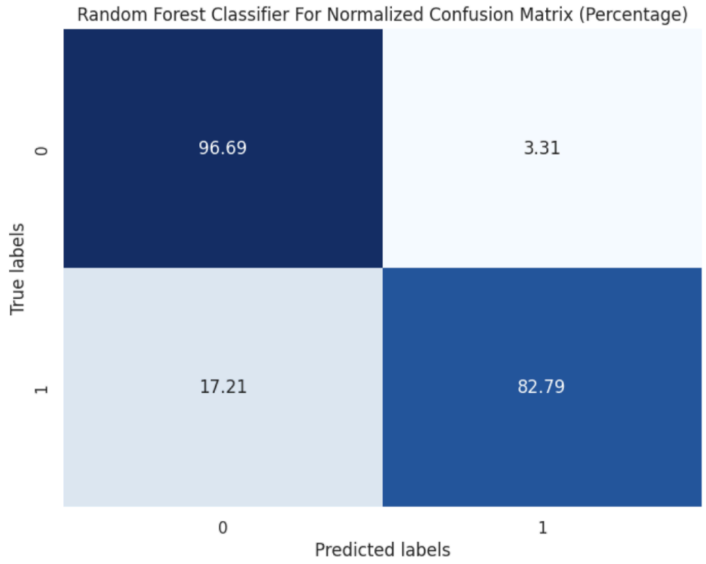


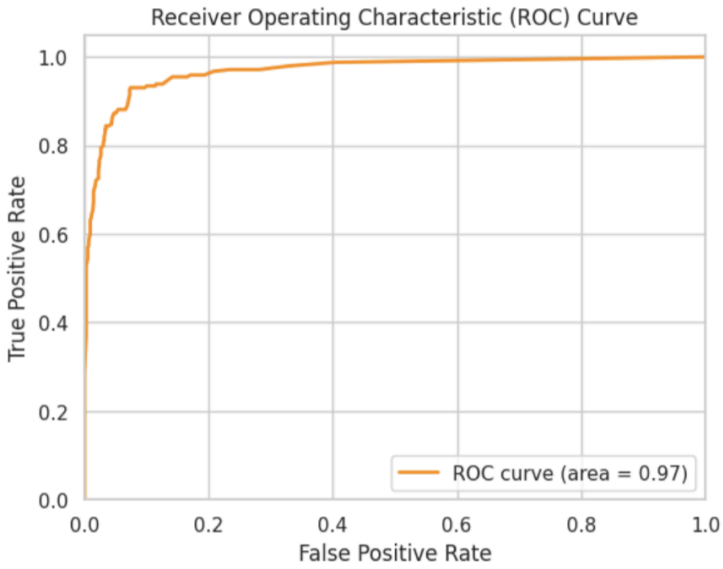


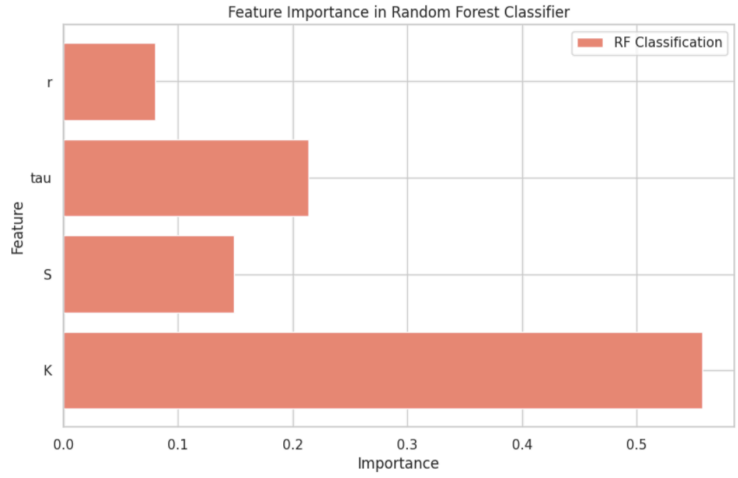


Random Forest Classification

Figure 8.1: Normalized Confusion Matrix
Figure 8.2: ROC Curve
Figure 8.3: Feature Importance

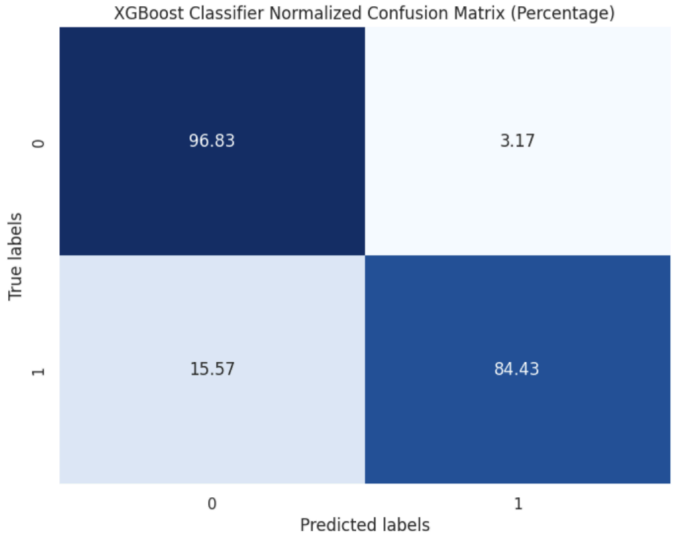


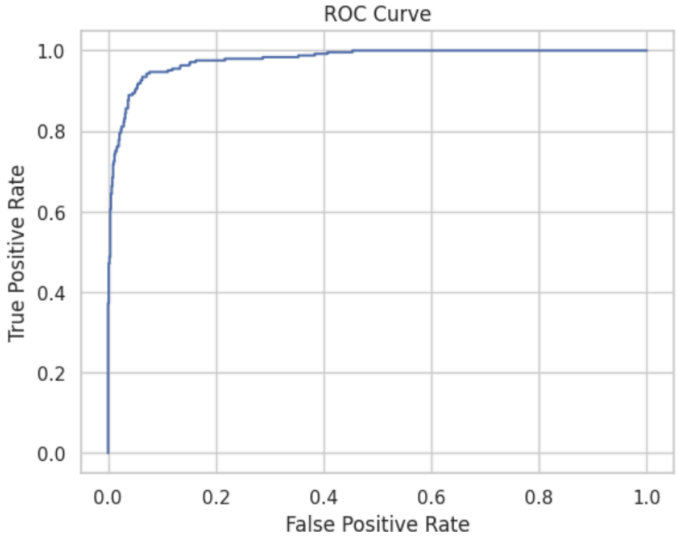


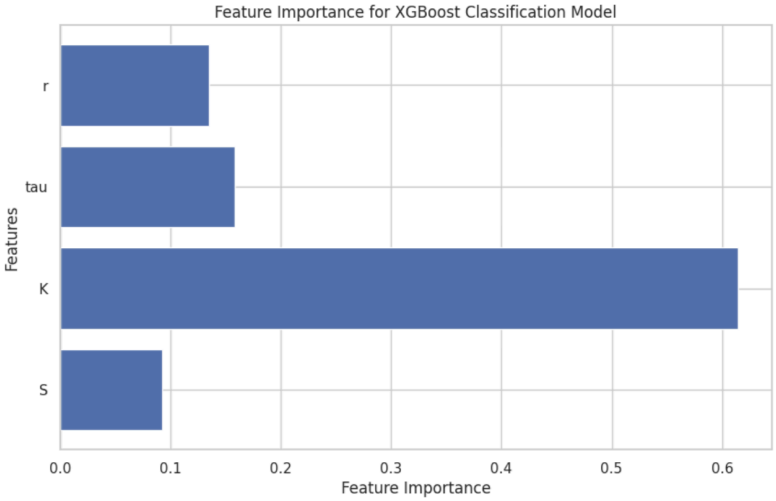


XGBoost Classification

Figure 9.1: Normalized Confusion Matrix
Figure 9.2: ROC Curve
Figure 9.3: Feature Importance

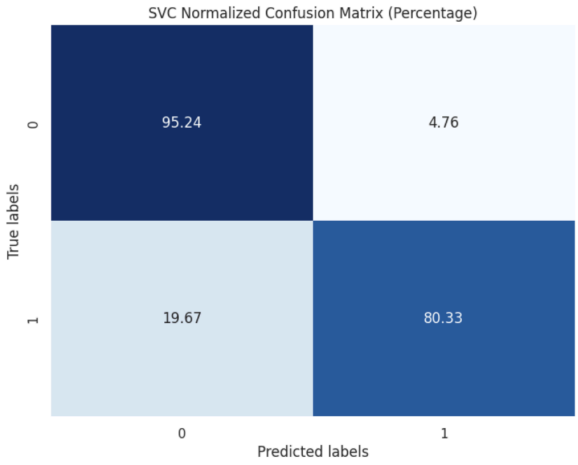


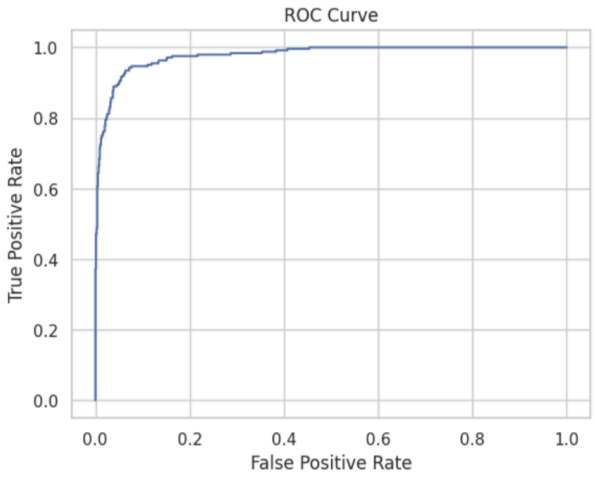


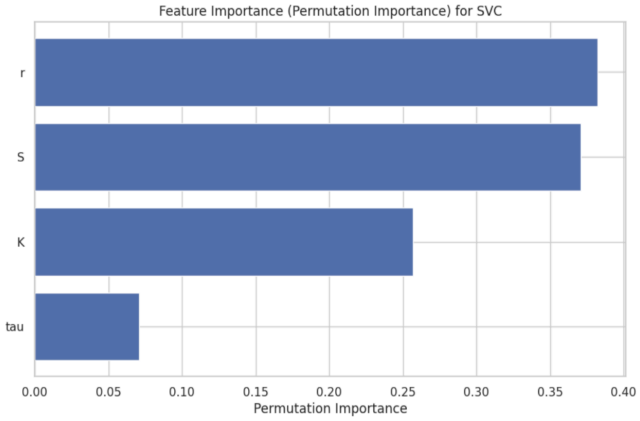


Support Vector Classification

Figure 10.1: Normalized Confusion Matrix
Figure 10.2: ROC Curve
Figure 10.3: Feature Importance

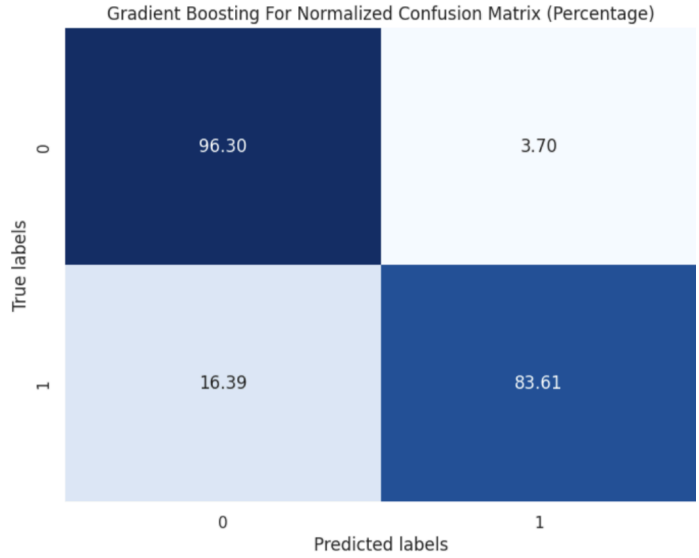


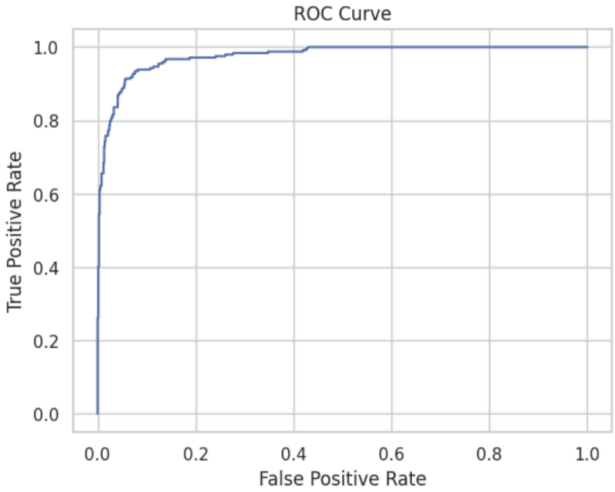


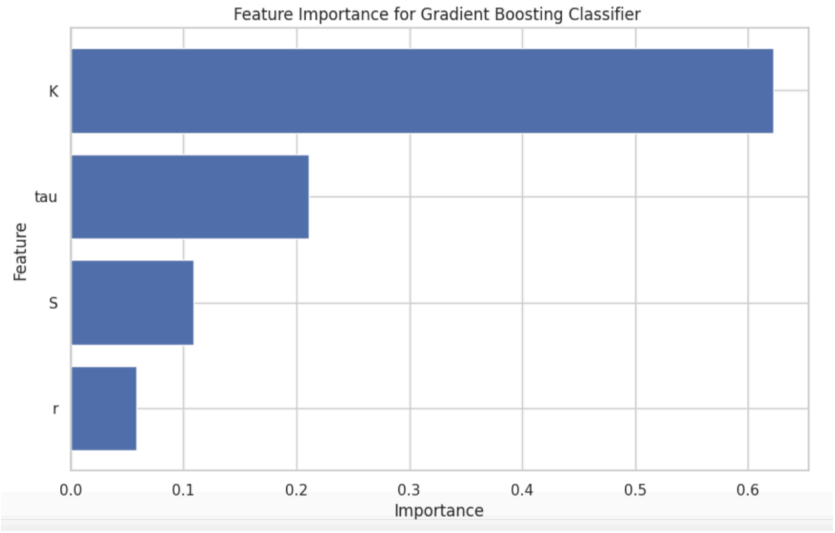


Gradient Boosting Classification

Figure 11.1: Normalized Confusion Matrix
Figure 11.2: ROC Curve
Figure 11.3: Feature Importance

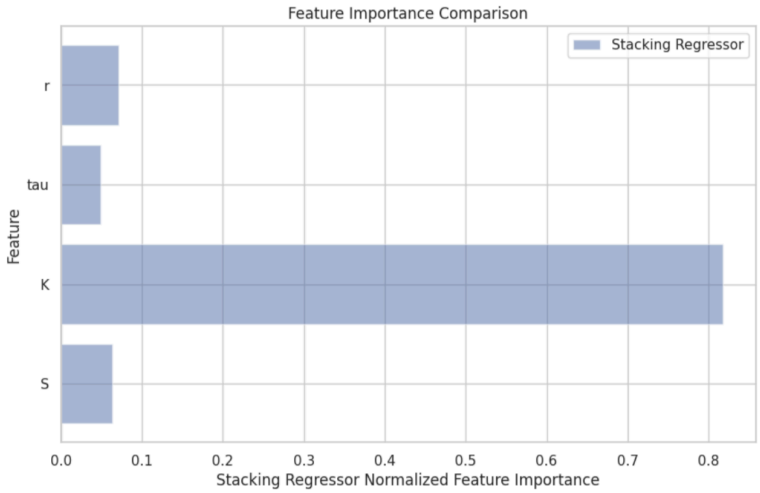






Ensemble Stacked Regressor

Figure 12.1: Stacked Regressor Feature Importance



Ensemble Stacked Classifier

Figure 13.1: Normalized Confusion Matrix
Figure 13.2: ROC Curve (each model)
Figure 13.3: ROC Curve
Figure 13.4: Feature Importance