Options Pricing Prediction

Prepared by - Group 11:

Abhinav Chadha	1543592016
Sumeet G Duddagi	1657480168
Puneet Singh Maini	2846862569
Mane Mirijanyan	2417844547
Sanjeet Walia	8054931174
Rizabek Zhumkenov	6937295872

For questions, please contact Sanjeet Walia (sanjeetw@usc.edu)

Executive Summary

Problem Statement

The objective of this project was to develop two machine learning models by examining European call option pricing data on the S&P 500 to predict the current option value (C) and to check if Black-Scholes option pricing model prediction overestimated the option value or underestimated the option value.

The Big Picture

A European call option gives the holder the right (but not the obligation) to purchase an asset at a given time for a given price. Valuing such an option is tricky because it depends on the future value of the underlying asset. The Black-Scholes option pricing formula provides an approach for valuing such options. The Black-Scholes formula states:

$$C\_pred = S\Phi(d_1) - Ke^{-r\tau}\Phi(d_2)$$

where,

C_pred: predicted option value

d_1=log(S/K)+(r+σ^2)tau/($\sigma\sqrt{}$tau)

d_2=d_1-(sÖTau)

S: Current asset value

Φ(x): represents the probability that a standard normal random variable will take on a value less

than or equal to x.

K: Strike price of Option

r: Annual interest rate

tau: time to maturity (in years)

Using a dataset of 1680 options, we built statistical/ML models. For training our models, we used 70% of this data and the rest 30% was used to test out the models. We explored regression models like linear Regression and Random Forest Regressor for predicting the Value and classification models like Logistic Regression, SVM, KNN, Random Forest Classifier and Naïve Bayes Classifier for classifying if the option value has been over-estimated or under-estimated.

Results

We found Random Forest Regressor to be our best model for predicting the Value. On the test data we created, we see out-of-sample R^2 as 99.82%.

For classification we used a different approach which we call Majority Principle. We used all 5 classification models that we had built and predicted BS for each record. Basis the majority prediction of these 5 models we classified our final BS as either ‘Over’ or ‘Under’. The test accuracy we achieved using this approach on the test data we created is 93.04%.

Methodology

We follow the following procedure to build prediction models for both "Value" and "BS" features.

A. Data Cleaning

This process involved looking for Missing Values and Outlier detection. We were able to find two records with missing values and by looking at the distribution of Tau and S we were able to remove three more records since they had values more than 100 for Tau, or 0 for S.

B. Feature Engineering

a. Trivial Features

Since we intuitively know that Value and BS do not have a linear relation with tau, we introduce more features such as √tau, tau^2, tau^4. We also introduced 1+ r into our dataset.

b. Non-Trivial Features

To introduce some scaling into our dataset, we introduce two more features, S/K and |S-K|.

C. Data Preparation

We introduce **min-max scaling** into our dataset. Since the features S, K, Tau, and r only take positive values in real life, it does not make sense to introduce Standard Scaler. Once we scale the data, we will use this scaler to transform the Options dataset without Labels for final production as well. After scaling, we divide our data into **Training and Test dataset** in 70-30 ratio with stratified on BS Flag. To have a uniform index for both prediction models, we have same set of test dataset for both "Value" prediction and "BS" prediction.

D. Prediction Models

To predict the "Value" feature we use Regression models and to predict "BS" feature we use Classification models.

a. Regression Model – Value Prediction

We look at two models, Linear Regression and Random Forest Regressor. We do feature selection using Forward Selection criteria using R^2 as the scoring metric. We then evaluate its performance on Test dataset using **5-fold cross validation**. We also tune hyperparameters for Random Forest Regressor and re-evaluate the model performance. This whole analysis can be summarized in the tables below:

MODEL	**FEATURES**	**R^2 on Test Data**
Linear Regression*	tau, S/K, |S-K| and r	99.45%
Random Forest Regressor*	S/K, √tau and r	99.82%

MODEL	**HYPER-PARAMETERS**	**R^2 on Test Data**
Random Forest Regressor+	N-estimators = 194, Max-depth = 5, Min-samples-split = 3, Min-samples-leaf = 1, Bootstrap = True	99.44%

* => After completing Forward Stepwise selections for both Linear Regression and Random Forest Regression, we clearly see that, by using K = 3 and K = 2, respectively, R^2 peaks, however the best features in Forward Selection Algorithm lose r, so introduce it manually.

+ => For, Random Forest Regressor we see, the hyperparameter grid we define does not help us in gaining more R^2, therefore, we will be using Random Forest Regressor where we do not specify any parameters except random state, i.e., no hyperparameter specified.

b. Classification Model – BS Prediction

We look at five different classification models for BS Prediction. These are logistic regression, SVM, KNN, Random Forest and Naïve Bayes Classifier. We start with looking at their top features using **Forward Stepwise Selection** criteria using **ROC-AUC scoring**. We plotted Accuracy at different levels of K, and selected K where accuracy hits the peak. We then look at the best **hyperparameters** for SVM, KNN, and Random Forest via Random Grid Search CV and used accuracy as the scoring metric. The following table summarizes our methodology:

MODEL	**FEATURES**	**ACCURACY on Test Data**
Logistic Regression	S, r, S/K, tau^2 and tau^4	92.24%
SVM	S, K, tau, r, S/K, |S – K| and √tau	92.64%
KNN*	S, S/K, tau^2, √tau and r	91.25%
Random Forest Classifier	S, r, S/K and √tau	93.24%
Naïve Bayes	S, r, S/K, |S – K| and tau^4	91.65%

MODEL	**HYPER-PARAMETERS**	**ACCURACY on Test Data**
SVM	C = 10, Gamma = 0.01, Kernel = 'linear'	92.64%
KNN	p = 2, N-Neighbors = 11, Leaf-Size = 5	91.25%
Random Forest Classifier	N-Estimators = 115, Min-Samples-Split = 5, Min-Samples-Leaf = 4, Max-Depth = 9, Bootstrap = True	93.84%

* => After completing Forward Stepwise selections for KNN, we clearly see that, by using K = 4, respectively, accuracy peaks, however the best features in Forward Selection Algorithm lose r, so introduce it manually.

Final Approach

For Value Prediction:

- We have a clear winner, where we select Random Forest Regressor without specifying any parameters except random state.

For BS Prediction:

- Since all classification models have almost similar results, it is difficult to choose one out of five. Therefore, we will be using Majority Principle to predict BS. We will be using all 5 classification models and predicting BS for each record. Based on the prediction by majority models we will be predicting BS.
  The following table sums up the whole process,

	Models					
Record	**Logistic Regression**	**SVM**	**KNN**	**Random Forest Classifier**	**Naïve Bayes Classifier**	**Final Prediction**
1	1	1	1	1	1	1
2	1	1	0	1	0	1
3	1	0	1	0	0	0
4	0	1	0	0	0	0
5	0	0	0	1	0	0

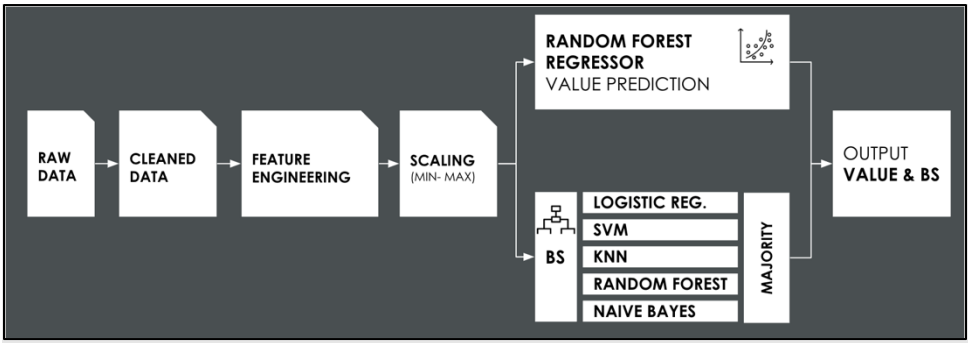
- Using this criterion, the model accuracy on Test Dataset is 93.04%.

Final Prediction for Options Data without Label:

Before we proceed with predicting anything for Options Data without Labels, we perform the following operations:

1. **Feature Engineering** – Similar to Primary Data, we add Trivial and Non-Trivial Features to our Options Data without Labels.
2. **Scaling** – We use the same feature scaler (Min-Max Scaler) as we did for our Primary Data and transform our Options Data without Labels.
3. **Model Re-Training** – for both Value and BS prediction as discussed above, we retrain the model, now on complete 100% of Primary Data for better results. For reference, earlier we were using 70% of it.

Conclusion



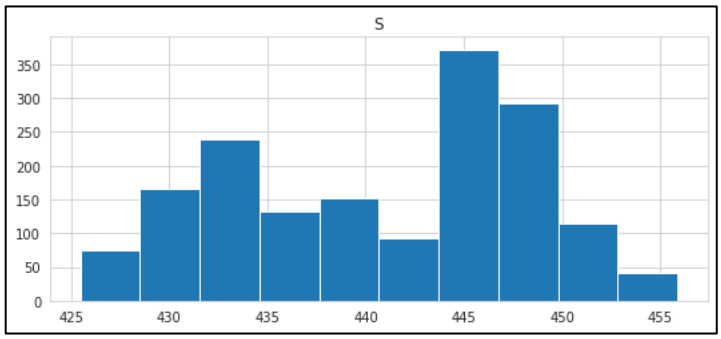
Our whole analysis can be summed up in this visual graphic. We clean our data, add trivial and non-trivial features, scale our data, and run Random Forest Regressor for Value Prediction and run these five models - Logistic Regression, SVM, KNN, Random Forest Classifier, Naïve Bayes Classifier to get BS Prediction and then use majority criteria for our final BS Prediction. We gave preference to Accuracy instead of Model Interpretation. The higher the prediction accuracy of our classification model, the better chances we have of classifying BS.

In our analysis, we found that volatility can be modeled using ML Regression and Classification algorithms. Our models could describe a relation between S, K, tau, and r, and final asset prices (Value) & BS variable. These findings are validated using actual market models i.e. Black-Scholes-Merton model. Furthermore, we arrived at the Model of European, which is a variation of Black-Scholes formula and is computationally efficient. This model can be successfully deployed for other stocks while validating for their excess volatility (if any).

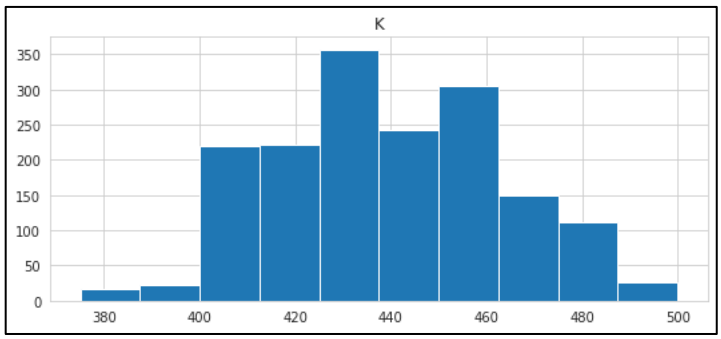
Appendix

1. EDA (Exploratory Data Analysis) after Data Cleaning

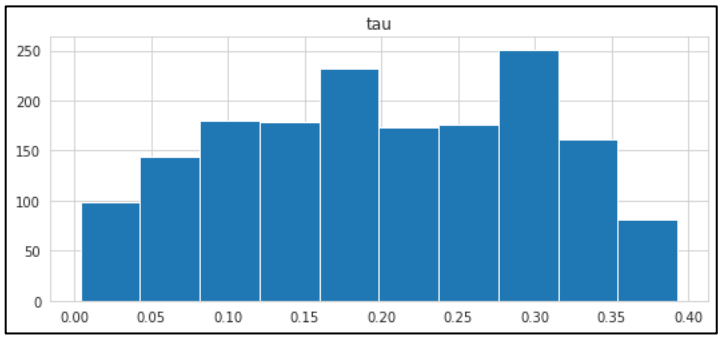
a. S



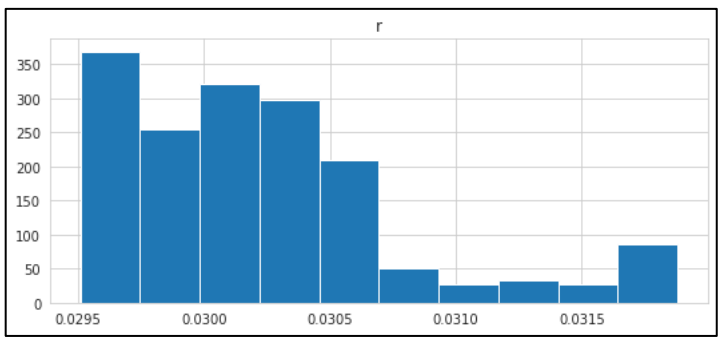
b. K



c. tau



d. r

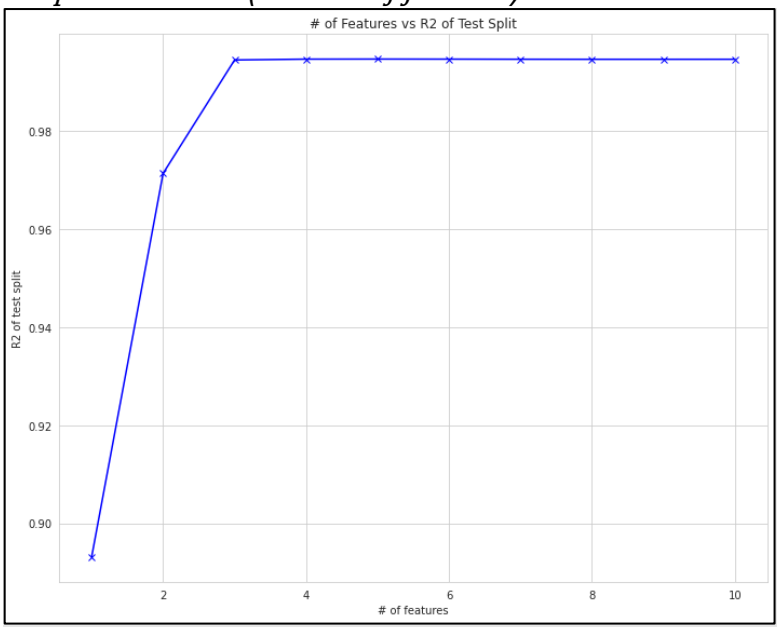


2. Value Prediction:

a. Linear Regression

i. Feature Selection

Graph between K (number of features) and R^2 on Test Data



ii. Model Performance

5-fold Cross Validation R^2= [0.99574382, 0.9945653, 0.9947298, 0.99404035, 0.99517379]

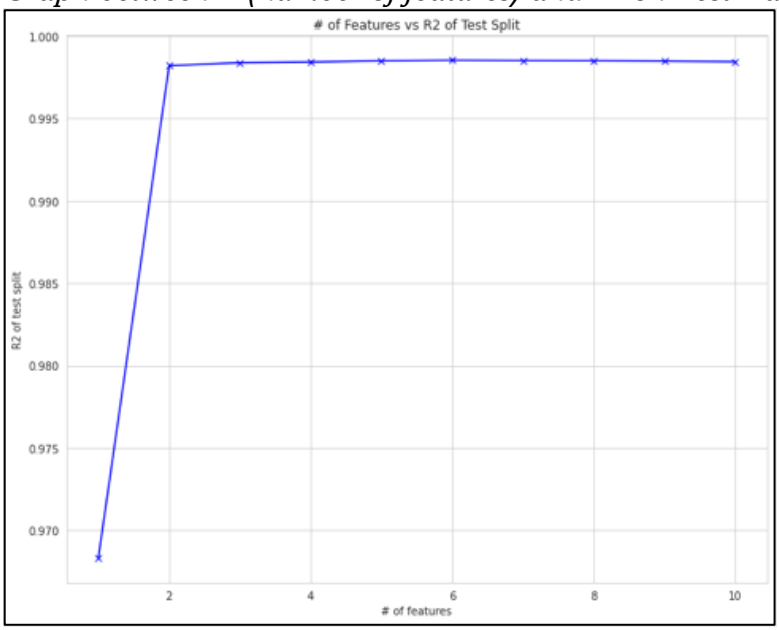
Mean R^2 on 5-fold CV: 0.9948506115337015

R^2 on test Dataset: 0.9945275988505234

b. Random Forest Regressor

i. Feature Selection

Graph between K (number of features) and R^2 on Test Data



ii. Model Performance without Hyperparameter Tuning

5-fold Cross Validation R^2= [0.99772576, 0.99752309, 0.99746519, 0.99791723, 0.99740396]

Mean R^2 on 5-fold CV: 0.9976070480078164

R^2 on test Dataset: 0.998211544820166

iii. Model Performance with Hyperparameter Tuning

5-fold Cross Validation R^2= [0.99279578, 0.99242773, 0.9935035, 0.99371717, 0.99439174]

Mean R^2 on 5-fold CV: 0.9933671838969798

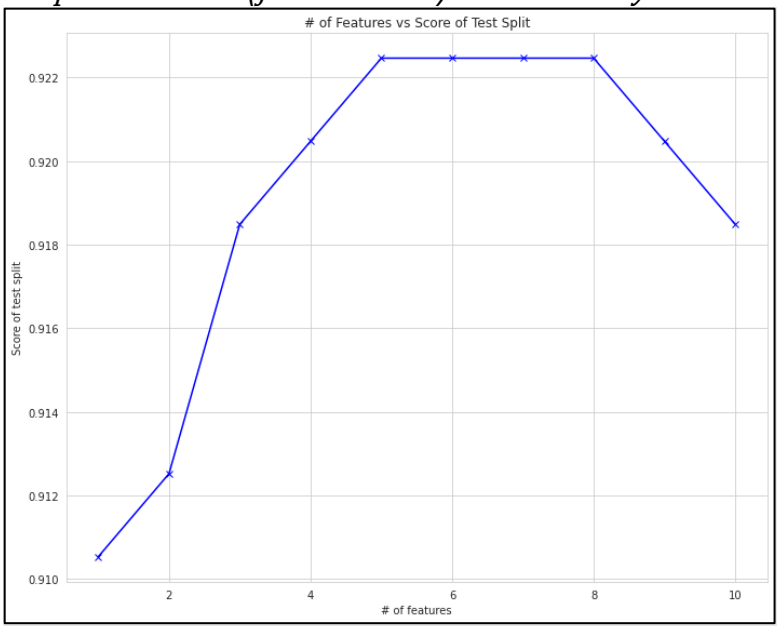
R^2 on test Dataset: 0.994440353117182

3. BS Prediction:

a. Logistic Regression

i. Feature Selection

Graph between K (feature count) and Accuracy Score on Test Data



ii. Model Performance

Confusion Matrix on Test Dataset:

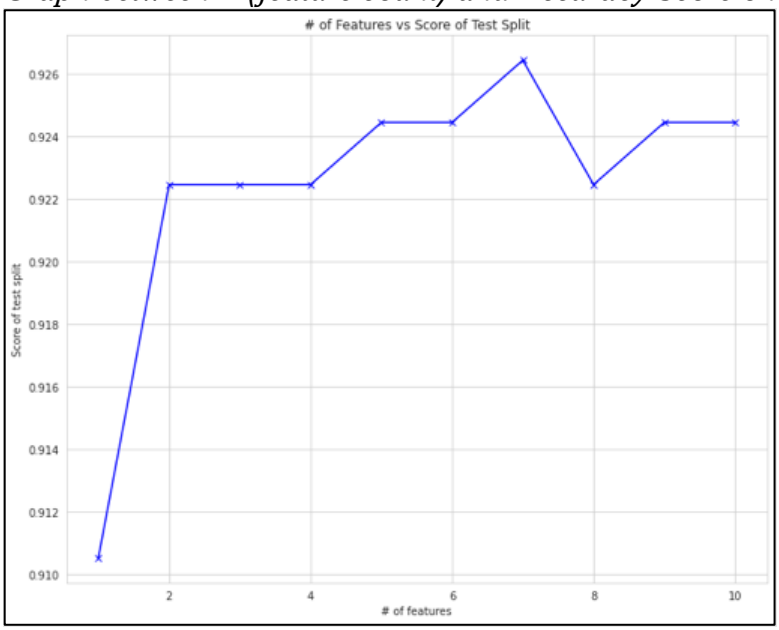
	Predicted 0	Predicted 1
Actual 0	264	20
Actual 1	19	200

Accuracy on test Dataset: (264 + 200)/ (264 + 200 + 19 + 200) = 92.24%

b. SVM (Support Vector Machine)

i. Feature Selection

Graph between K (feature count) and Accuracy Score on Test Data



ii. Model Performance without Hyperparameter Tuning

Confusion Matrix on Test Dataset:

	Predicted 0	Predicted 1
Actual 0	269	15
Actual 1	22	197

Accuracy on test Dataset: 92.64%

iii. Model Performance with Hyperparameter Tuning

Confusion Matrix on Test Dataset:

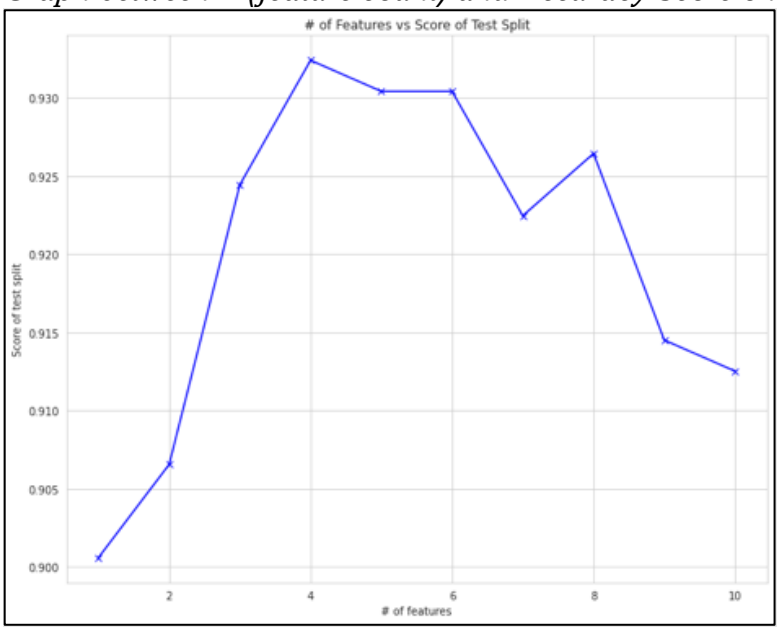
	Predicted 0	Predicted 1
Actual 0	263	21
Actual 1	16	203

Accuracy on test Dataset: 92.64%

c. KNN (K Nearest Neighbors)

i. Feature Selection

Graph between K (feature count) and Accuracy Score on Test Data



ii. Model Performance without Hyperparameter Tuning

Confusion Matrix on Test Dataset:

	Predicted 0	Predicted 1
Actual 0	267	17
Actual 1	27	192

Accuracy on test Dataset: 91.25%

iii. Model Performance with Hyperparameter Tuning

Confusion Matrix on Test Dataset:

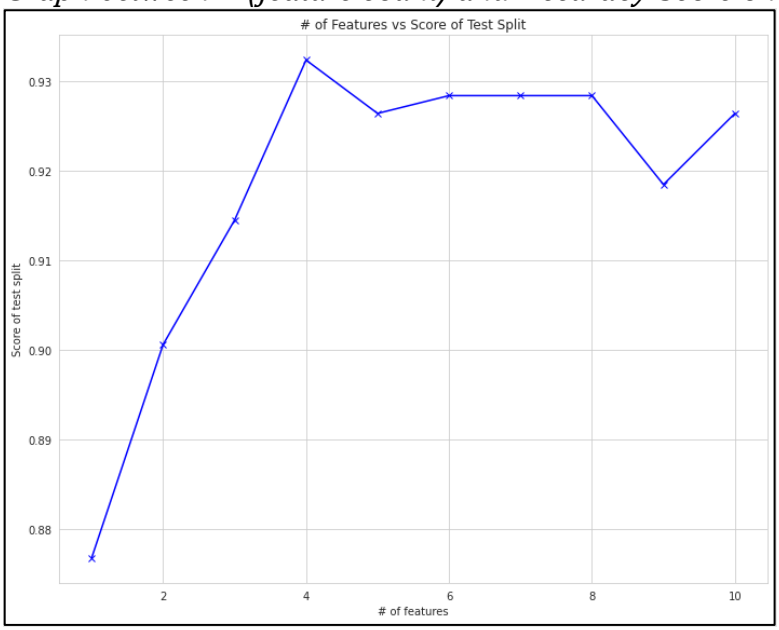
	Predicted 0	Predicted 1
Actual 0	264	20
Actual 1	24	195

Accuracy on test Dataset: 91.25%

d. Random Forest Classifier

i. Feature Selection

Graph between K (feature count) and Accuracy Score on Test Data



ii. Model Performance without Hyperparameter Tuning

Confusion Matrix on Test Dataset:

	Predicted 0	Predicted 1
Actual 0	265	19
Actual 1	15	204

Accuracy on test Dataset: 93.24%

iii. Model Performance with Hyperparameter Tuning

Confusion Matrix on Test Dataset:

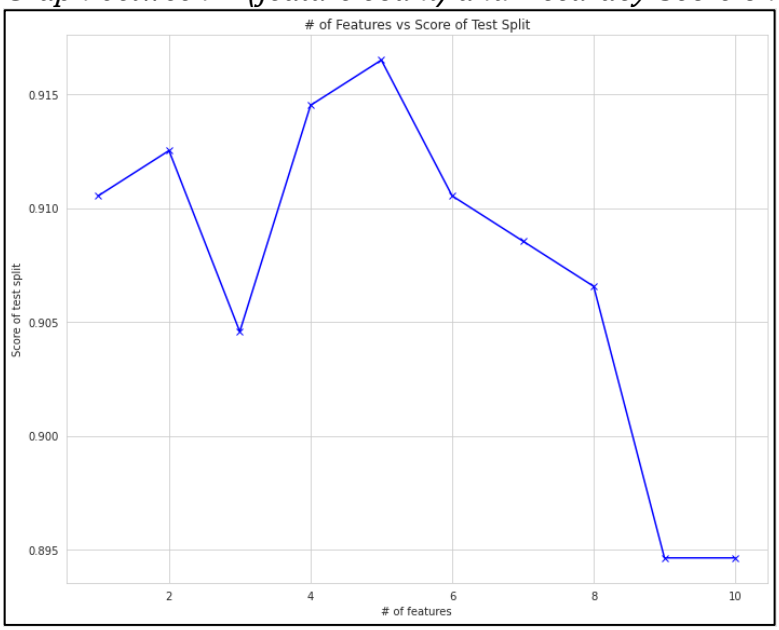
	Predicted 0	Predicted 1
Actual 0	266	18
Actual 1	13	206

Accuracy on test Dataset: 93.84%

e. Naïve Bayes Classifier

i. Feature Selection

Graph between K (feature count) and Accuracy Score on Test Data



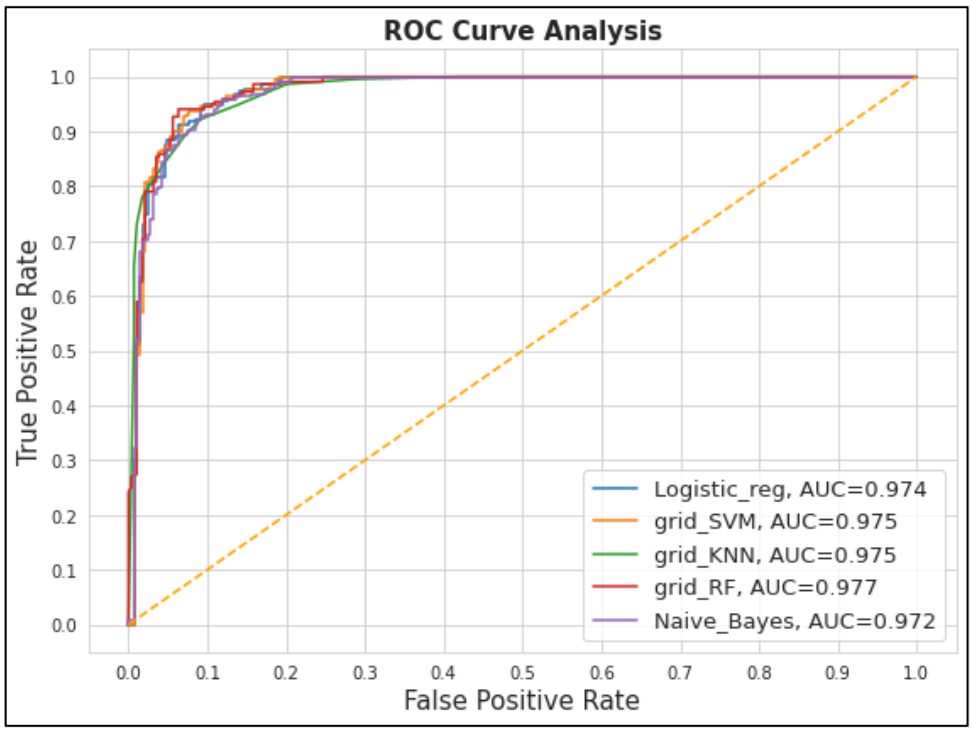
ii. Model Performance

Confusion Matrix on Test Dataset:

	Predicted 0	Predicted 1
Actual 0	259	25
Actual 1	17	202

Accuracy on test Dataset: 91.65%

4. Classification Model Comparison (ROC-AUC Curve)



5. Majority Criteria for Prediction

Confusion Matrix on Test Dataset:

	Predicted 0	Predicted 1
Actual 0	266	18
Actual 1	17	202

Accuracy on test Dataset: 93.04%