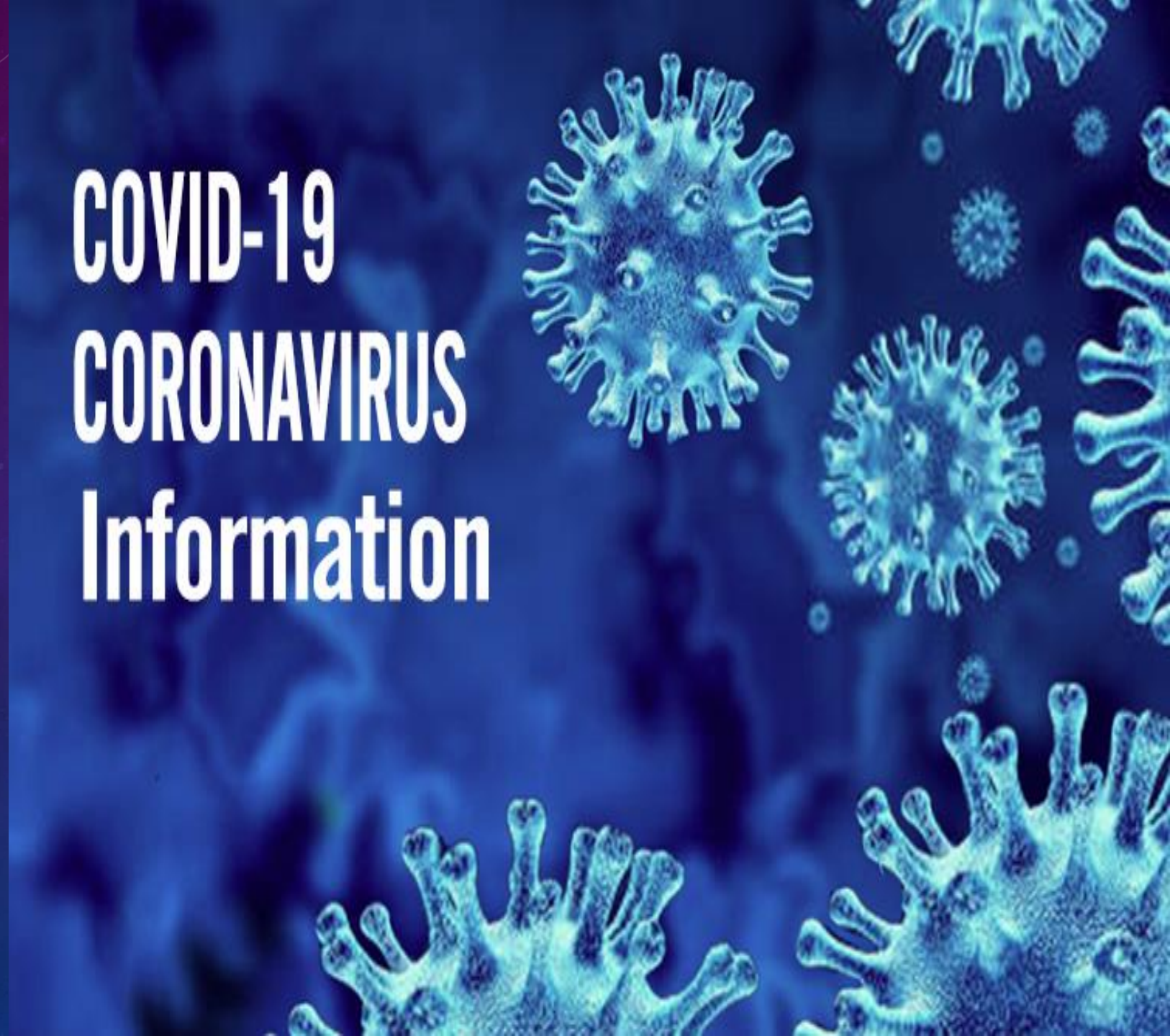
CSAI 801 PROJECT:
COVID-19 OUTCOME PREDICTION

Dr. Marwa Elsayed

Eng. Amr Zaki



- NAME: Bilal Mohamed Fetouh Mohamed Morsy
- ID : 20398551

PROJECT DESCRIPTION

- The data used in this project will help to identify whether a person is going to recover from coronavirus symptoms or not based on some pre-defined standard symptoms.
- These symptoms are based on guidelines given by the World Health Organization (WHO). This dataset has daily level information on the number of affected cases, deaths and recovery from 2019 novel coronavirus.
- The data is available from 22 Jan,2020.

DATA OF THE PROJECT

- Our data in this project is used for classifying the death and recovery of Covid-19 patients. It consists of (14) features which are:
  - 1. Country: where the person resides
  - 2. Location: which part in the Country
  - 3. Age: Classification of the age group for each person, based on WHO Age Group Standard
  - 4. Gender: Male or Female
  - 5. Visited_Wuhan: whether the person has visited Wuhan, China or not
  - 6. From_Wuhan: whether the person is from Wuhan, China or not
  - 7. Symptoms: there are six families of symptoms that are coded in six fields (symptoms 1 till 6)
  - 8. Time_before_symptoms_appear: period of time before being suspected as a Covid-19 patient
  - 9. Result: death (1) or recovered (0)

DATA PREPROCESSING

- As mentioned in the description that, The data is already cleaned and preprocessed.
- The shape of the data is Row 863 Column 14.
- There are 13 features and 1 label column (target data).
- In the training data, there is a column called (Unnamed: 0) will be dropped.
- Split the data into Train and Test by 80% for training to 20% for testing.
- Normalize the data values using StandardScaler which most algorithms need that.

MODELS TO ANALYSIS THE PROJECT

- You need to design the following classifiers:
  - 1. K-Nearest Neighbors (Without SMOTE and With SMOTE)
  - 2. Logistic Regression (Without SMOTE and With SMOTE)
  - 3. Naïve Bayes (Without SMOTE and With SMOTE)
  - 4. Decision Trees (Without SMOTE and With SMOTE)
  - 5. Support Vector Machines (Without SMOTE and With SMOTE)
- For each classifier, find the optimal hyperparameters.

THE OPTIMAL HYPERPARAMETERS FOR EACH CLASSIFIER

- We used GridSearchCV to find the optimal hyperparameters.
- What is grid search?
  - The idea that hyperparameter tuning using (scikit-learn's GridSearchCV) was the greatest invention of all time.
  - It runs through all the different parameters that is fed into the parameter grid and produces the best combination of parameters, based on a scoring metric of your choice (Accuracy, F1, etc.).
- The parameters in GridSearchCv are :
  - **estimator**: estimator object being used.
  - **param_grid**: dictionary that contains all of the parameters to try.
  - **scoring**: evaluation metric to use when ranking results.
  - **cv**: cross-validation, the number of cv folds for each combination of parameters.

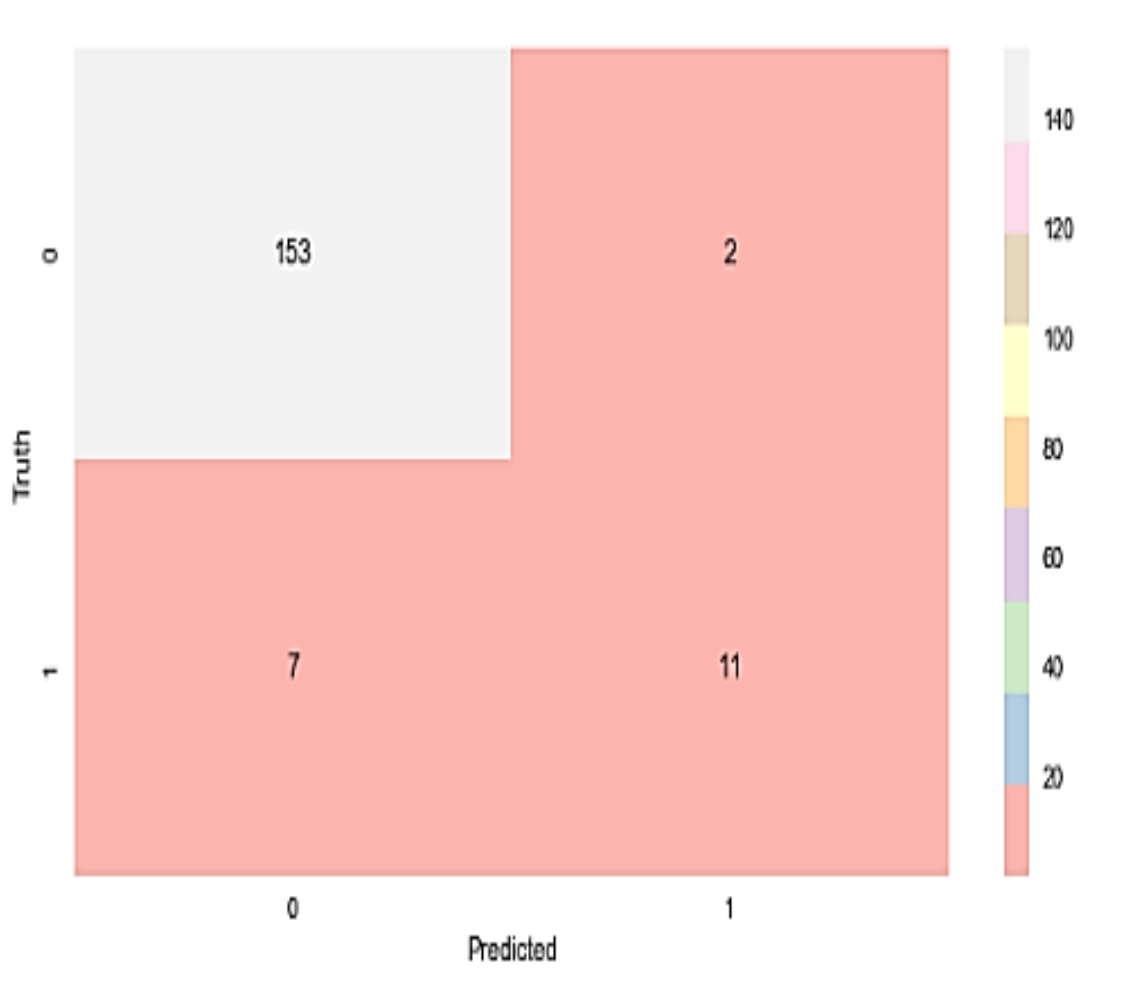
PERFORMANCE OF ALL CLASSIFIERS USING DIFFERENT METRICS

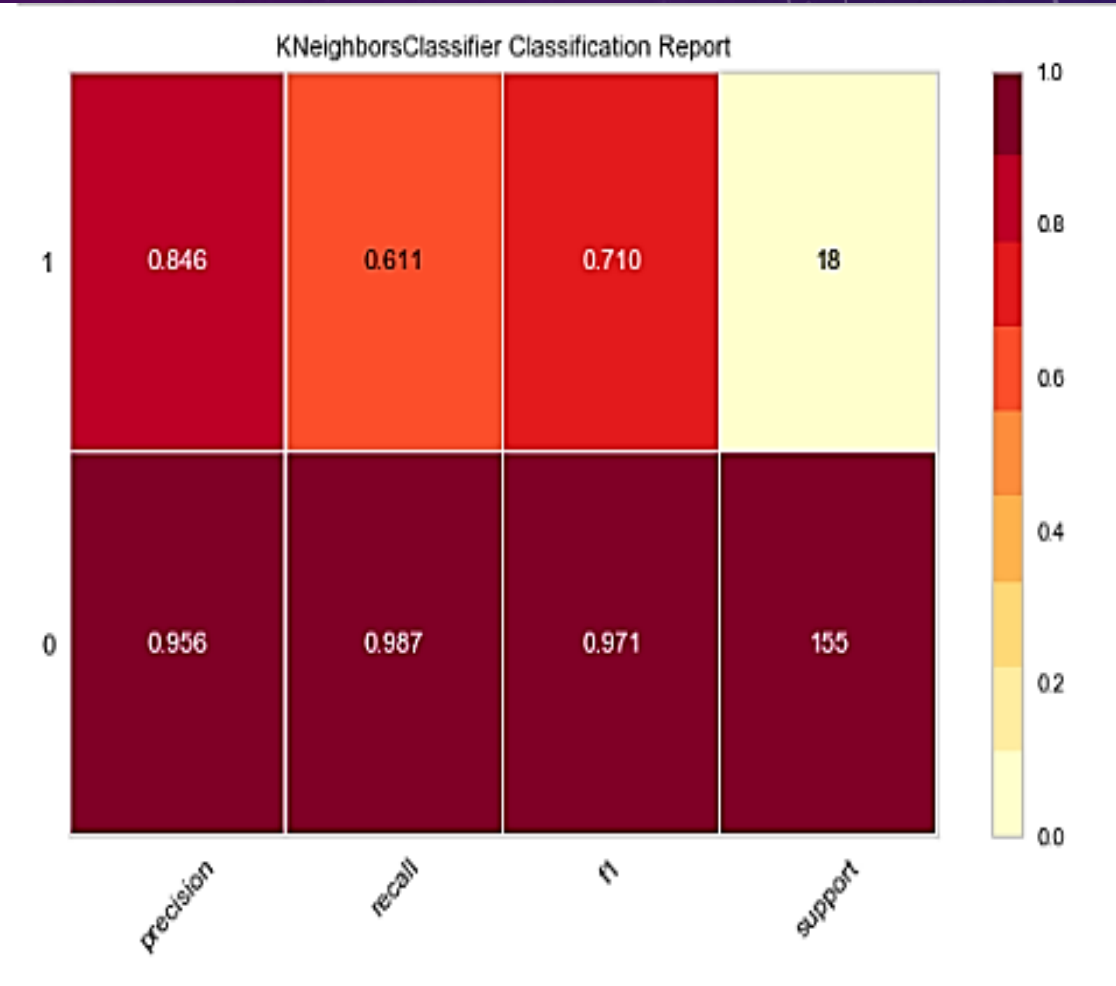
- **Confusion Matrix:** a table showing correct predictions and types of incorrect predictions.
- **Precision:** the number of true positives divided by all positive predictions. Precision is also called Positive Predictive Value. It is a measure of a classifier's exactness. Low precision indicates a high number of false positives.
- **Recall:** the number of true positives divided by the number of positive values in the test data. The recall is also called Sensitivity or the True Positive Rate. It is a measure of a classifier's completeness. Low recall indicates a high number of false negatives.
- **F1: Score:** the weighted average of precision and recall.
- **Area Under ROC Curve** (AUROC): AUROC represents the likelihood of your model distinguishing observations from two classes.

1. K-NEAREST NEIGHBORS (WITHOUT SMOTE)

	KNN	KNN with Best parameters
Data Splitting	Training = 80 % Testing = 20 %	Cv= 10
Hyperparameters	'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]}	N_Neighbors = 6
	Metric = 'minkowski'	Metric = 'minkowski'
	P = 2	P = 2
	Weights = ['uniform', 'distance']	Weights = 'distance'
	estimator=KNeighborsClassifier()	estimator=KNeighborsClassifier()
	scoring='accuracy'	scoring='accuracy'

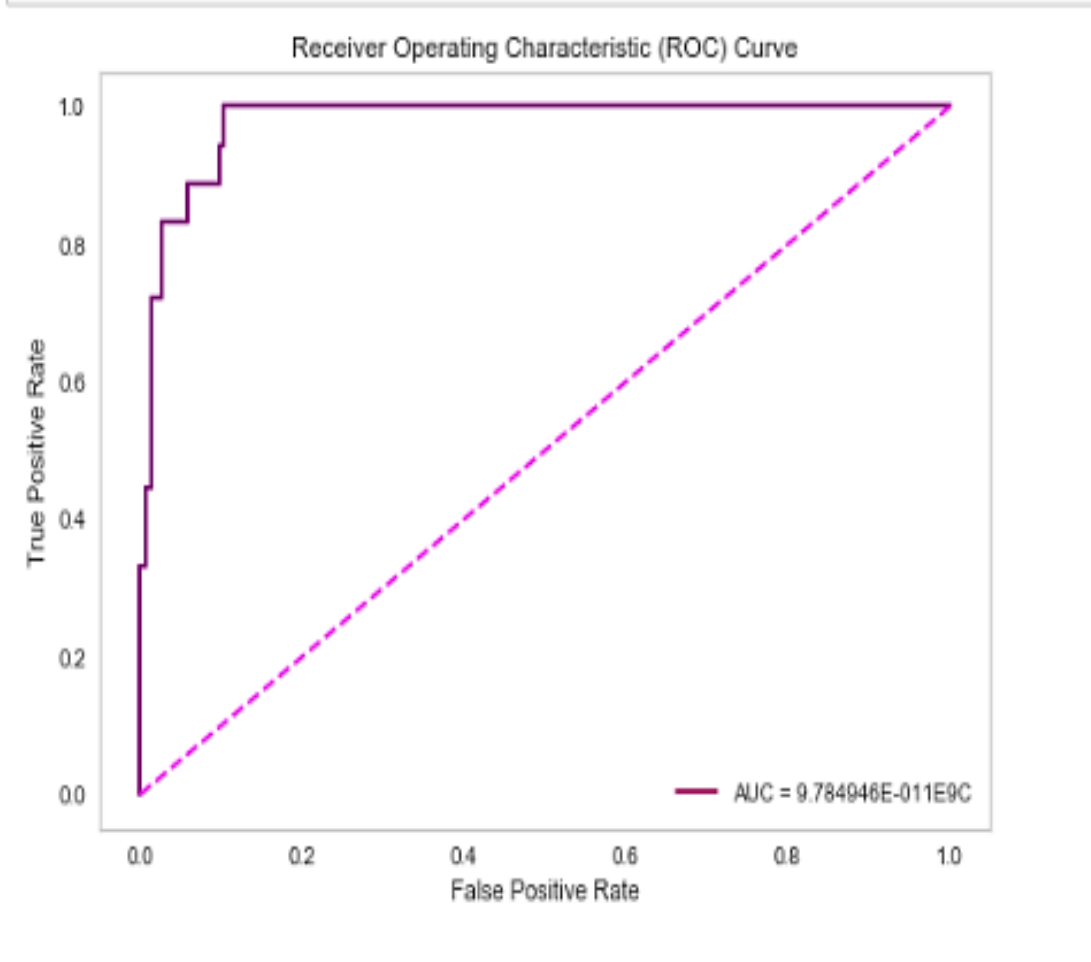
CONFUSION MATRIX AND CLASSIFICATION REPORT(KNN)

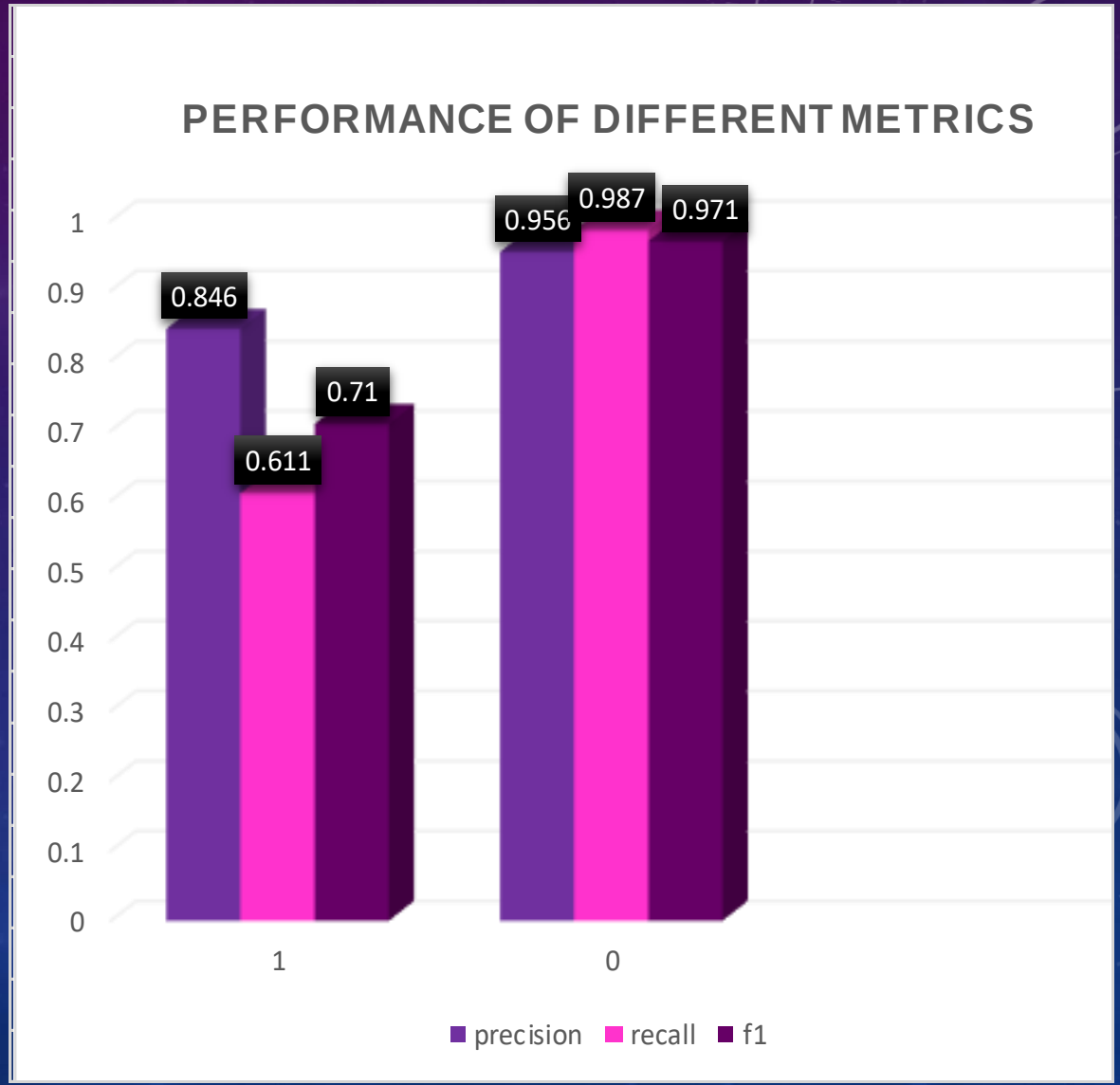




THE ROC CURVE(KNN)

The AUC = 97.8 %

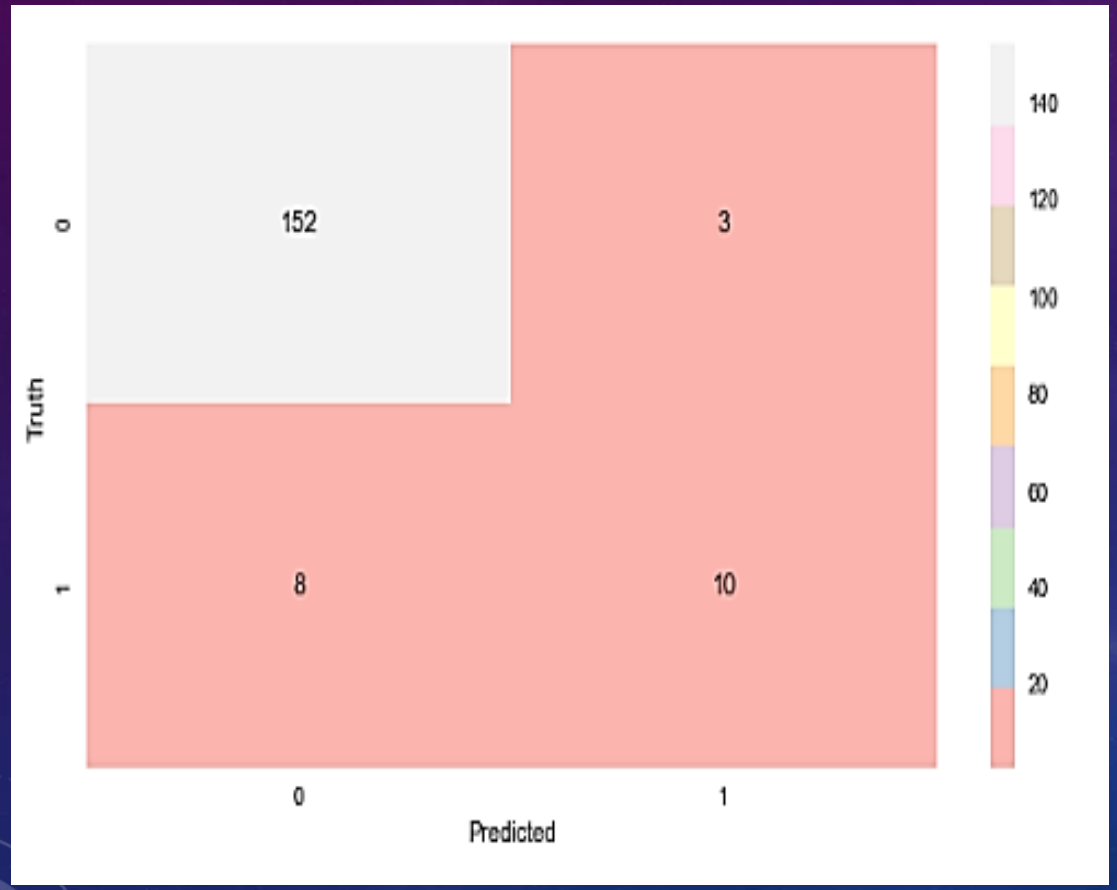


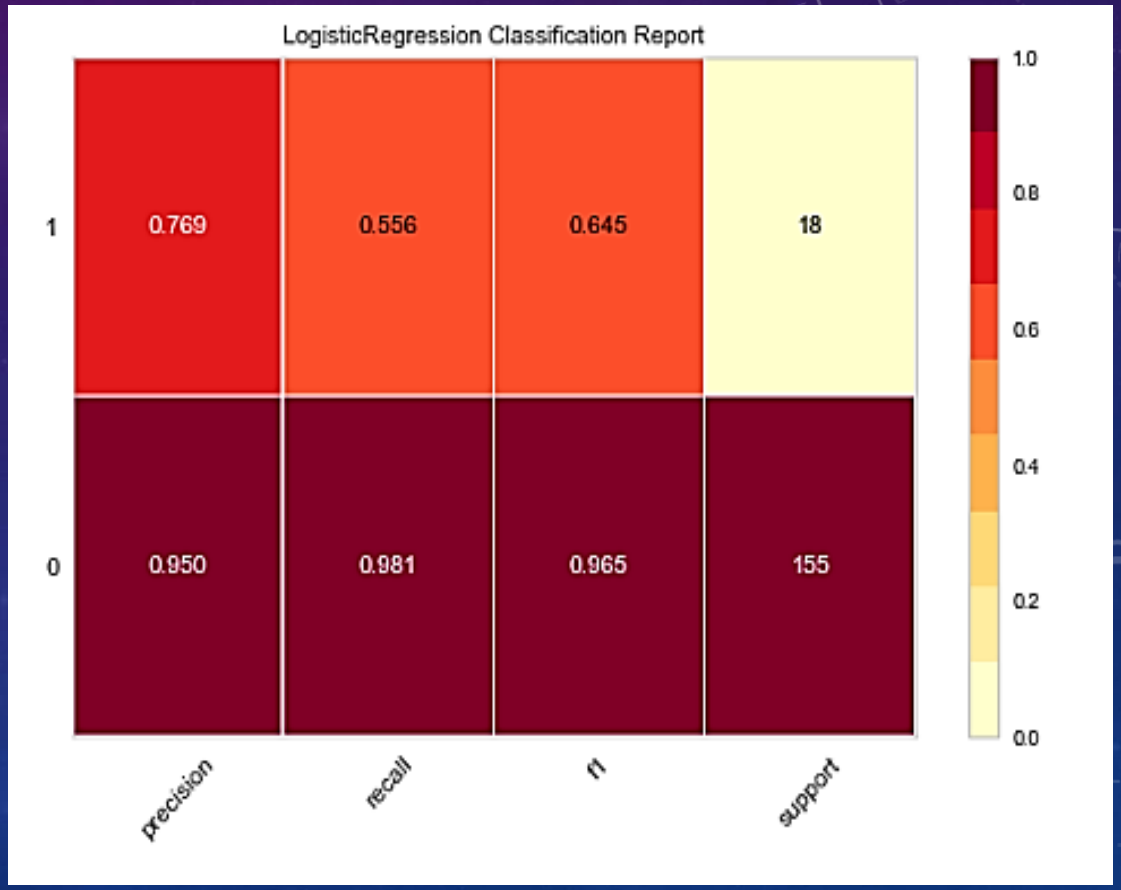


2. LOGISTIC REGRESSION (WITHOUT SMOTE)

	Logistic Regression	Logistic Regression with Best parameters
Data Splitting	Training = 80 % Testing = 20 %	Cv= 10 , max_iter = 300
Hyperparameters	'C':[0.001, 0.005, 0.08, 0.1, 0.5, 0.8, 1.2, 1, 5, 10, 25,100000.0]	'C': 5
	'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']	'solver': 'liblinear'
	estimator=LogisticRegression()	estimator=LogisticRegression ()
	scoring='accuracy'	scoring='accuracy'

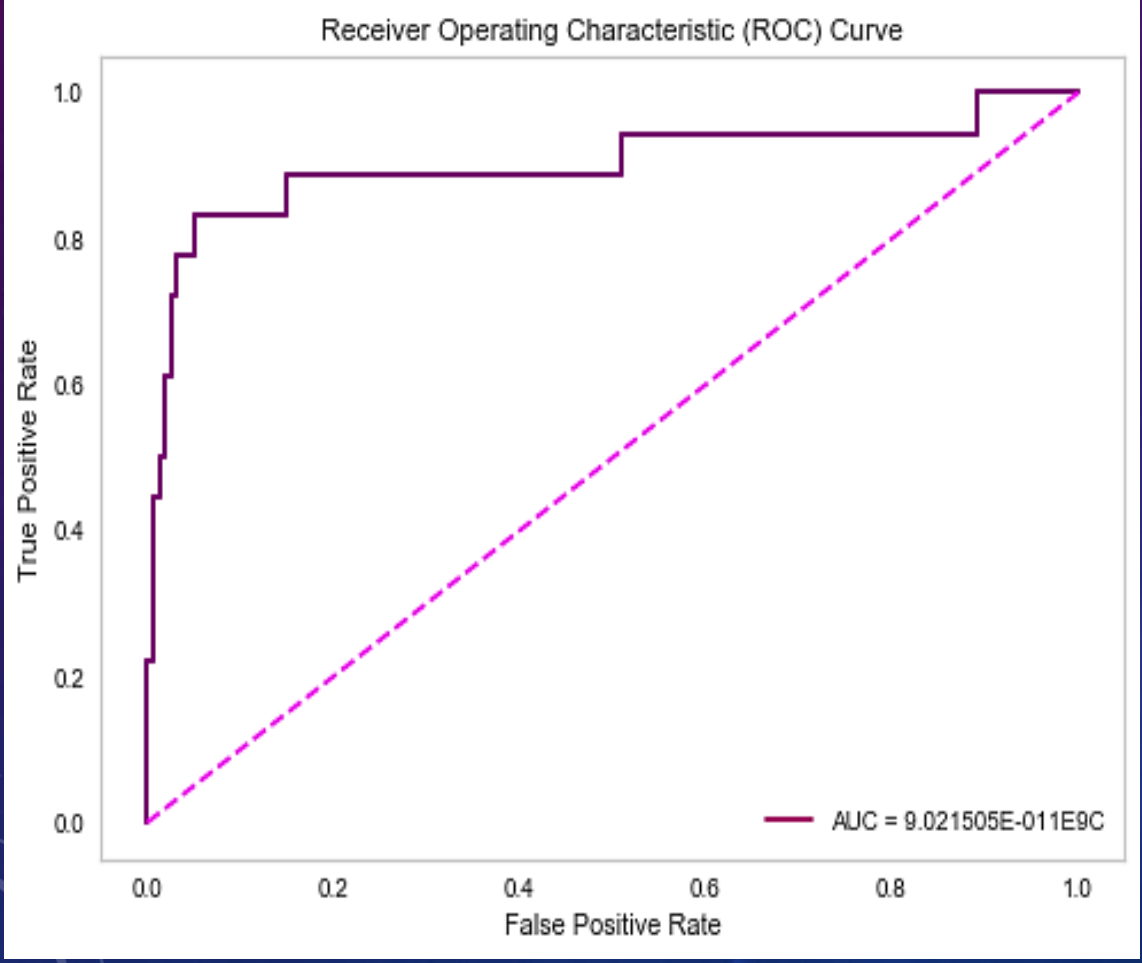
CONFUSION MATRIX AND CLASSIFICATION REPORT(LOGREG)

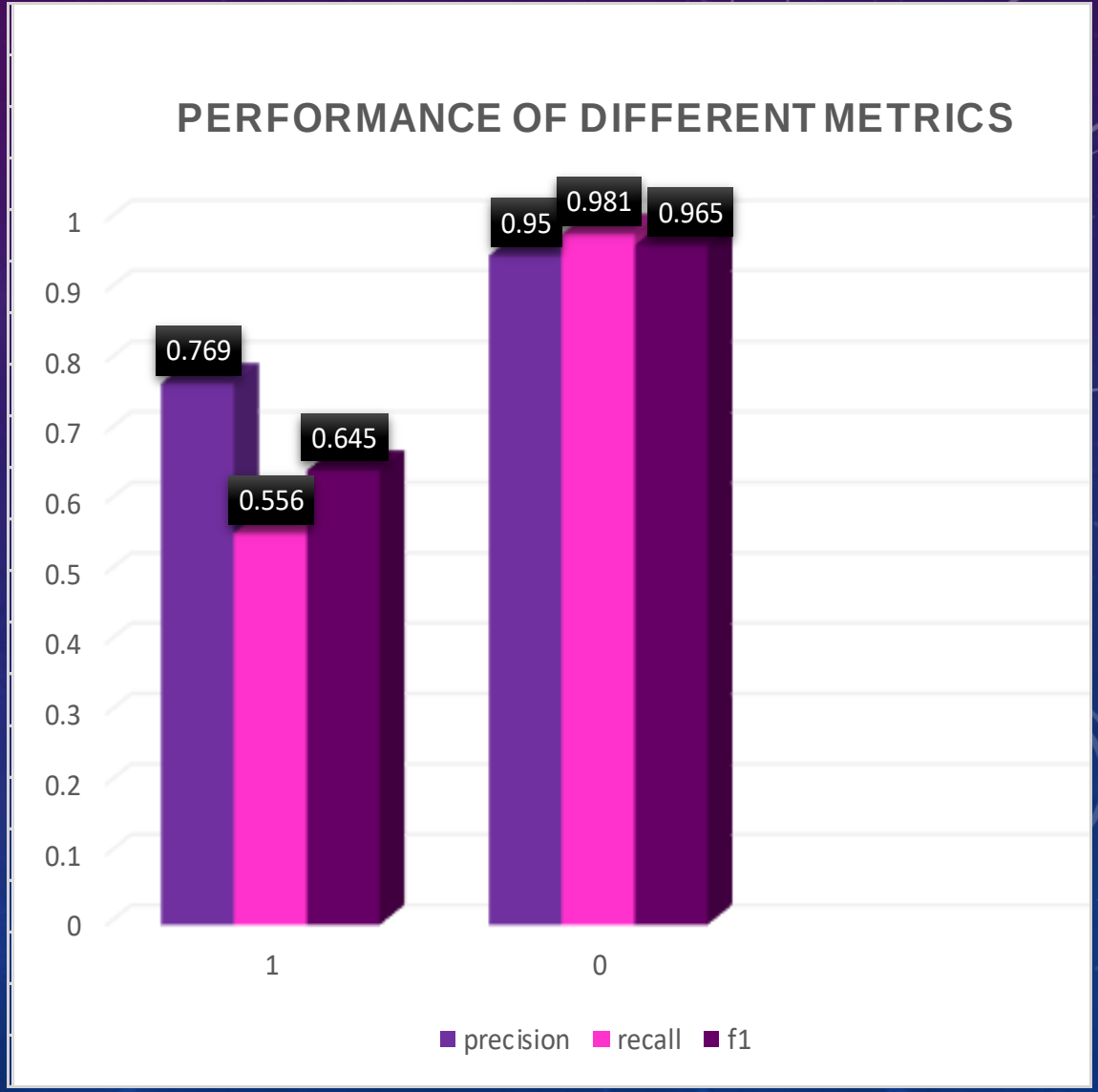




THE ROC CURVE(LOGREG)

The AUC = 90.2%

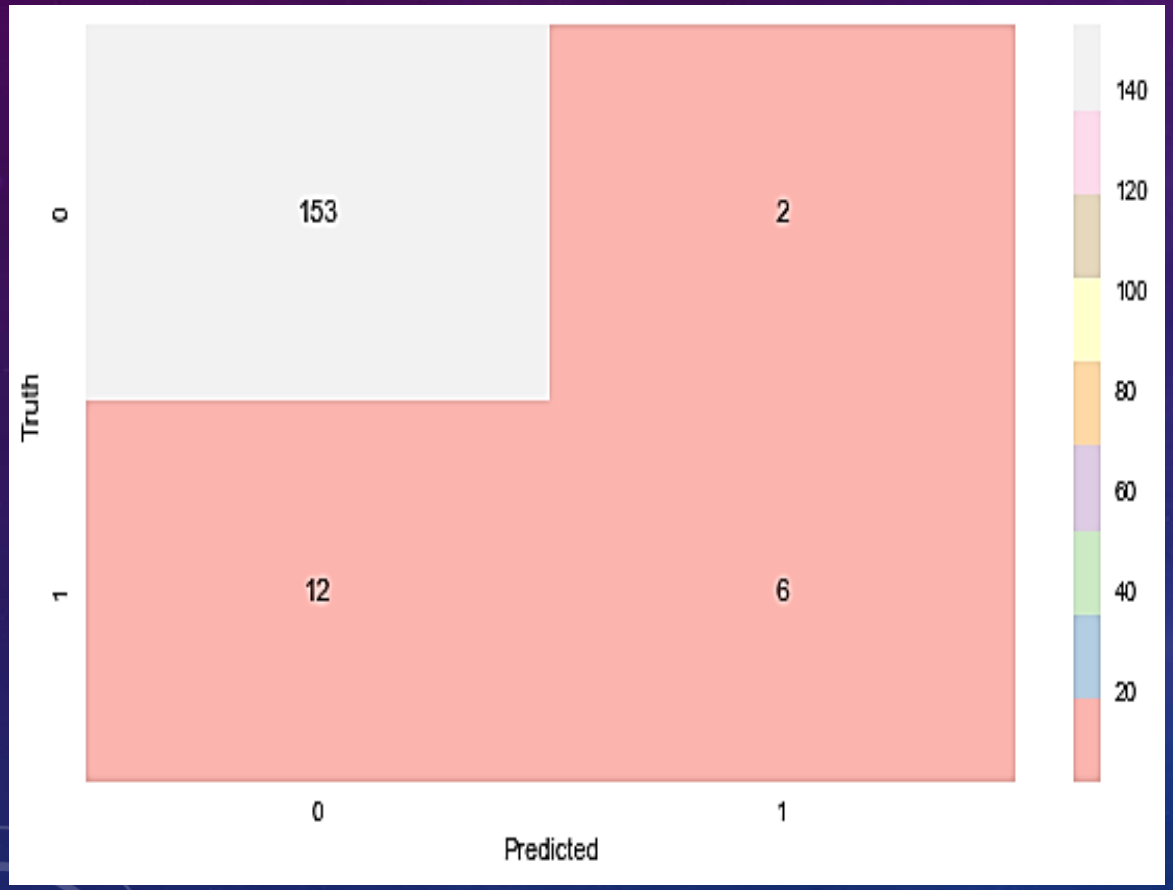


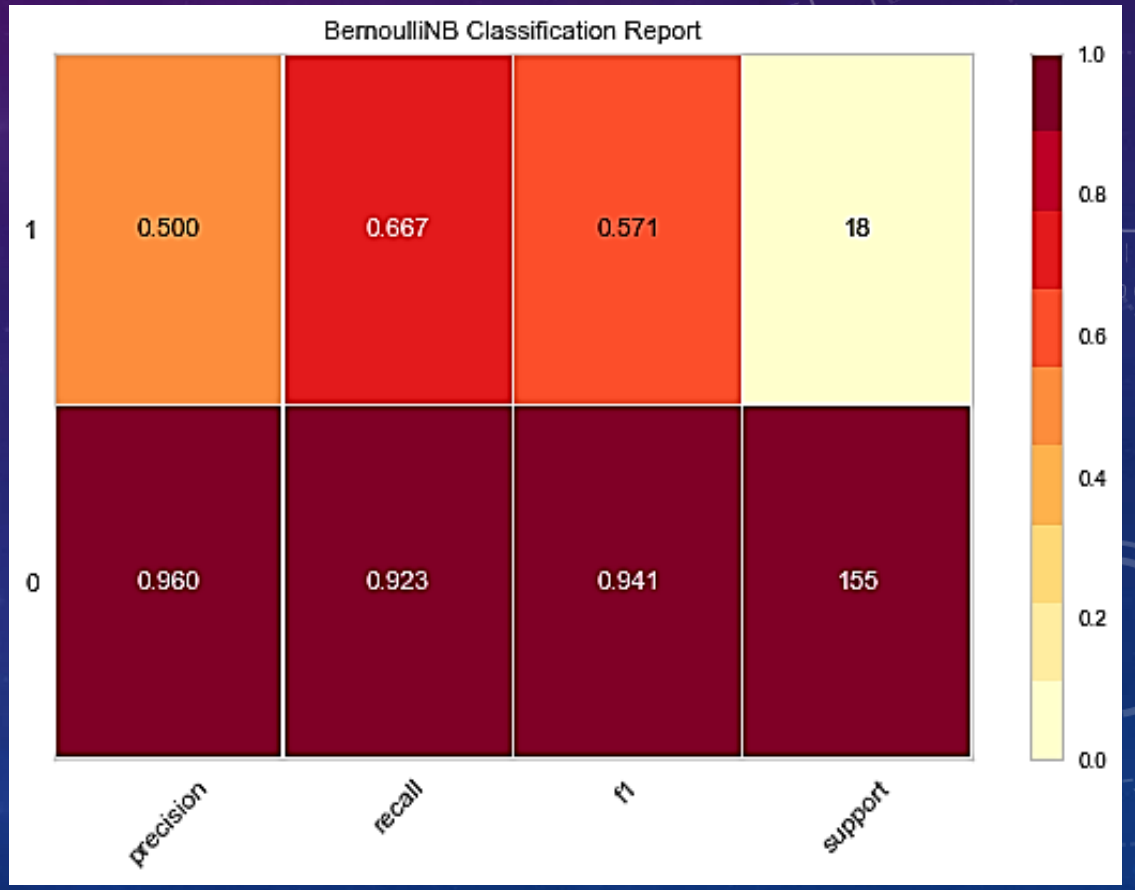


3. BERNOULLI NAIVE BAYES (WITHOUT SMOTE)

	Bernoulli Naive Bayes	Bernoulli Naive Bayes with Best parameters
Data Splitting	Training = 80 % Testing = 20 %	Cv= 10
Hyperparameters	'alpha': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,14, 15, 16, 17, 18, 19]	'alpha': 5,
	'binarize': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,14, 15, 16, 17, 18, 19]	'binarize': 1,
	estimator=BernoulliNB(alpha=1, binarize=1)	BernoulliNB(alpha = 5, binarize=1, fit_prior = True)
	scoring='accuracy'	scoring='accuracy'
	n_jobs=-1	n_jobs=-1
	'fit_prior': [True, False]	'fit_prior': True

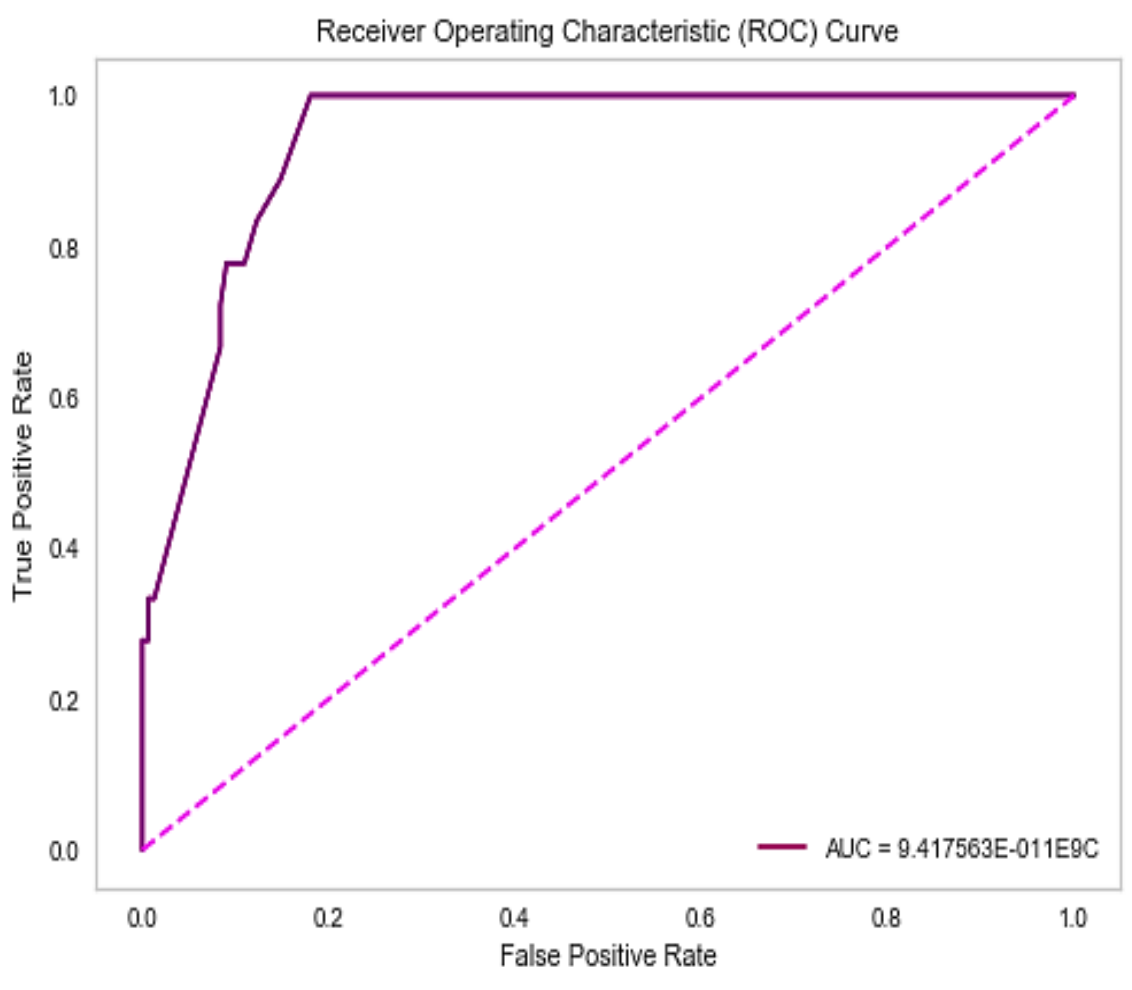
CONFUSION MATRIX AND CLASSIFICATION REPORT(NAIVE BAYES)

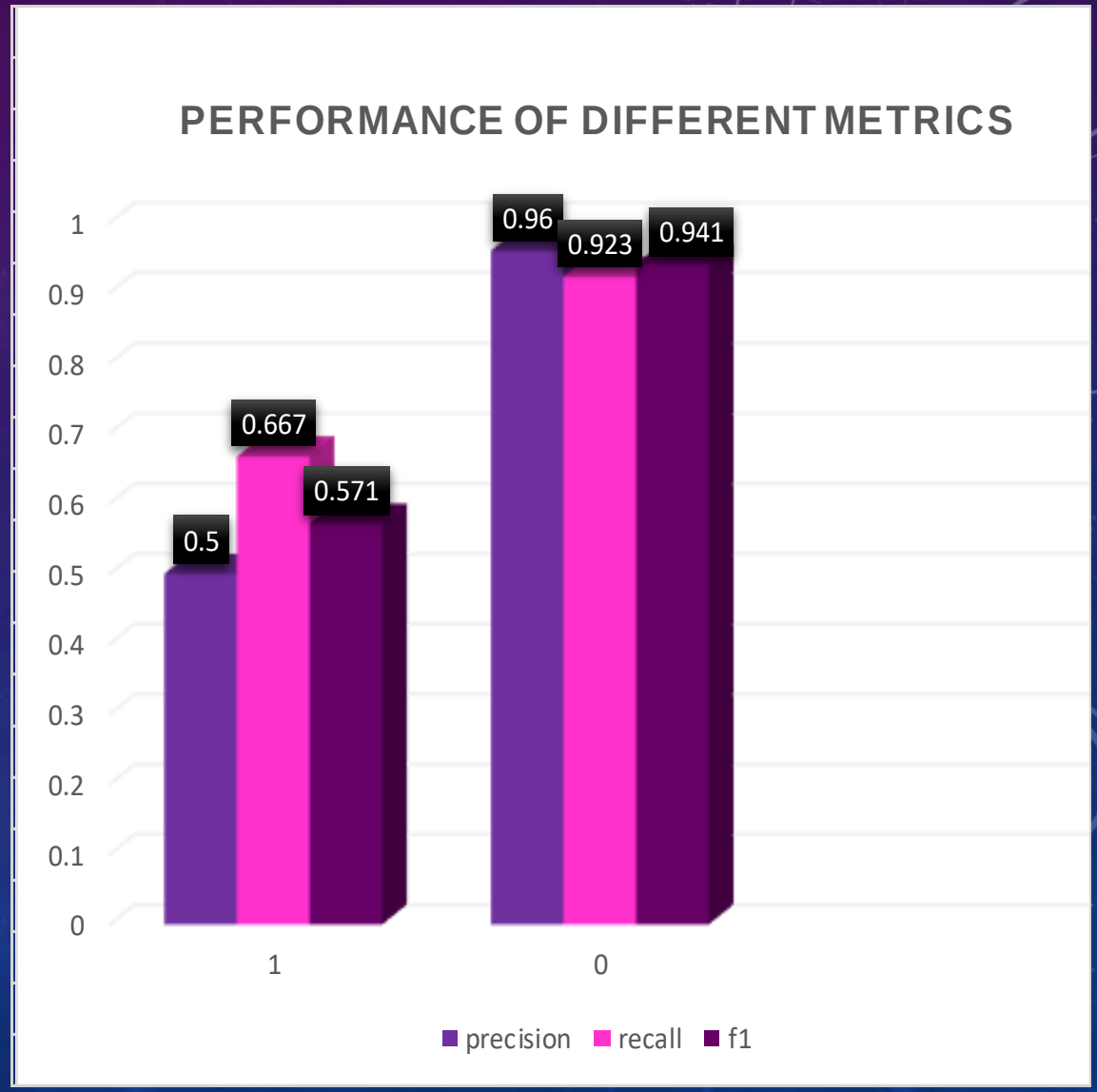




THE ROC CURVE(NAIVE BAYES)

The AUC = 94.2%

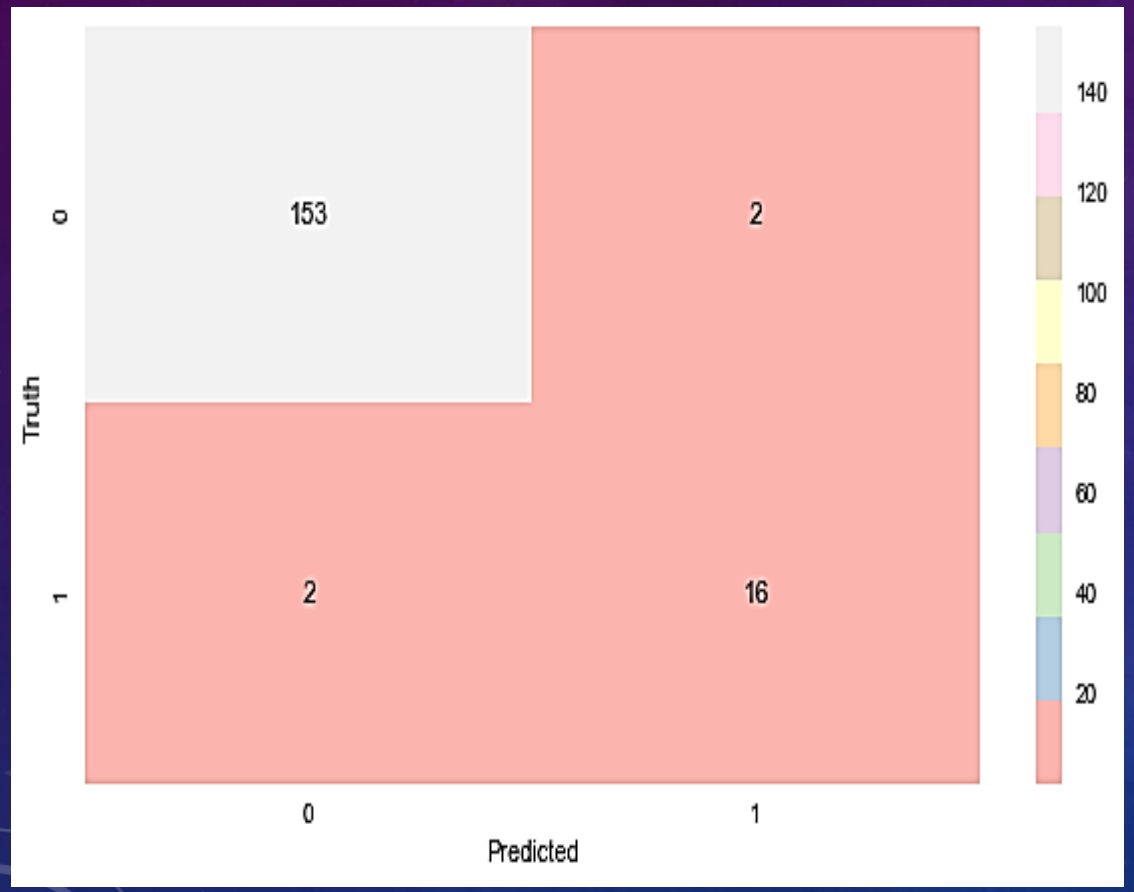


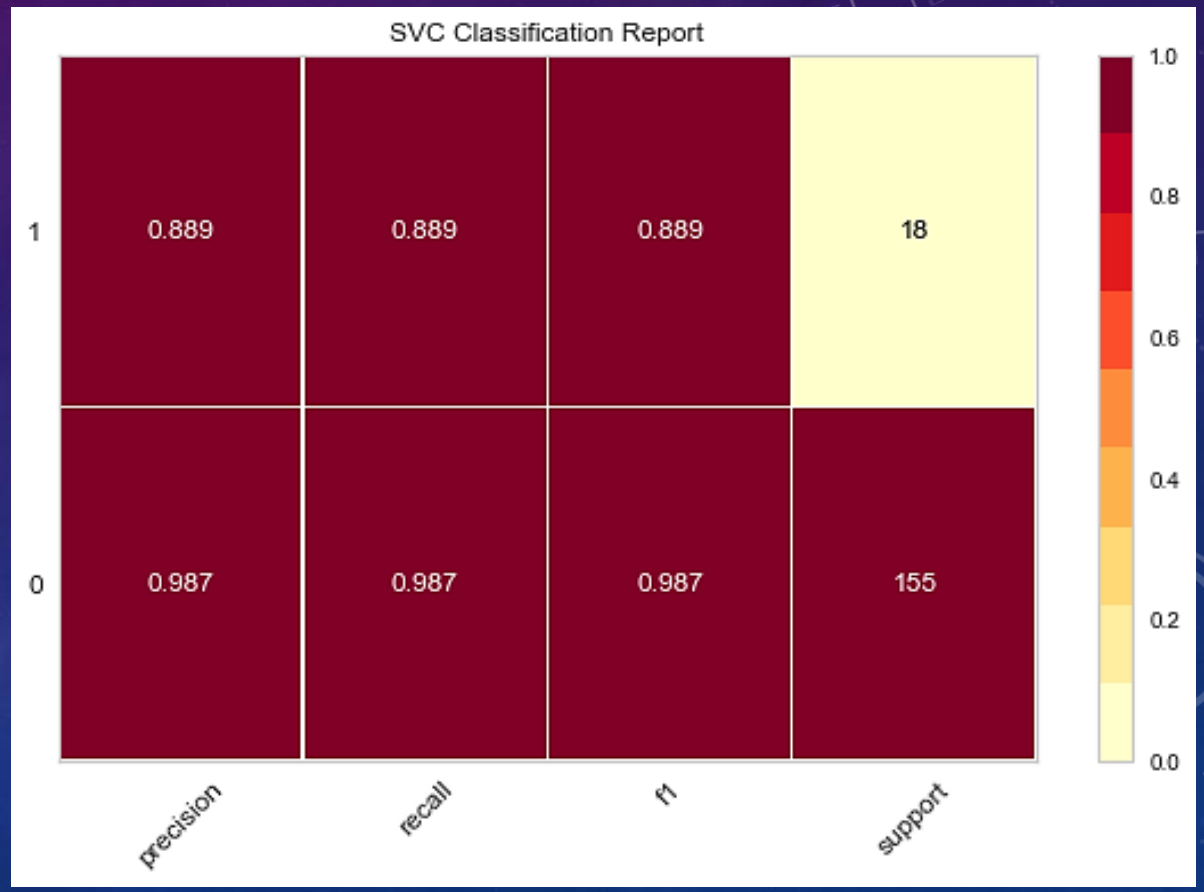


4. SUPPORT VECTOR MACHINE(SVM) (WITHOUT SMOTE)

	Support vector machine	Support vector machine with Best parameters
Data Splitting	Training = 80 % Testing = 20 %	Cv= 10
Hyperparameters	'C': [0.1, 1, 10, 100]	'C': 10
	'gamma': [1, 0.1, 0.01, 0.001, 0.0001]	'gamma': 0.1
	estimator=SVC(random_state=3)	estimator=SVC(C=10, gamma=0.1, probability=True, random_state=0)
	'kernel': ['linear', 'rbf']	'kernel': 'rbf'
	'probability': [True, False]	probability': True
	scoring='accuracy'	scoring='accuracy'
	n_jobs=-1	n_jobs=-1

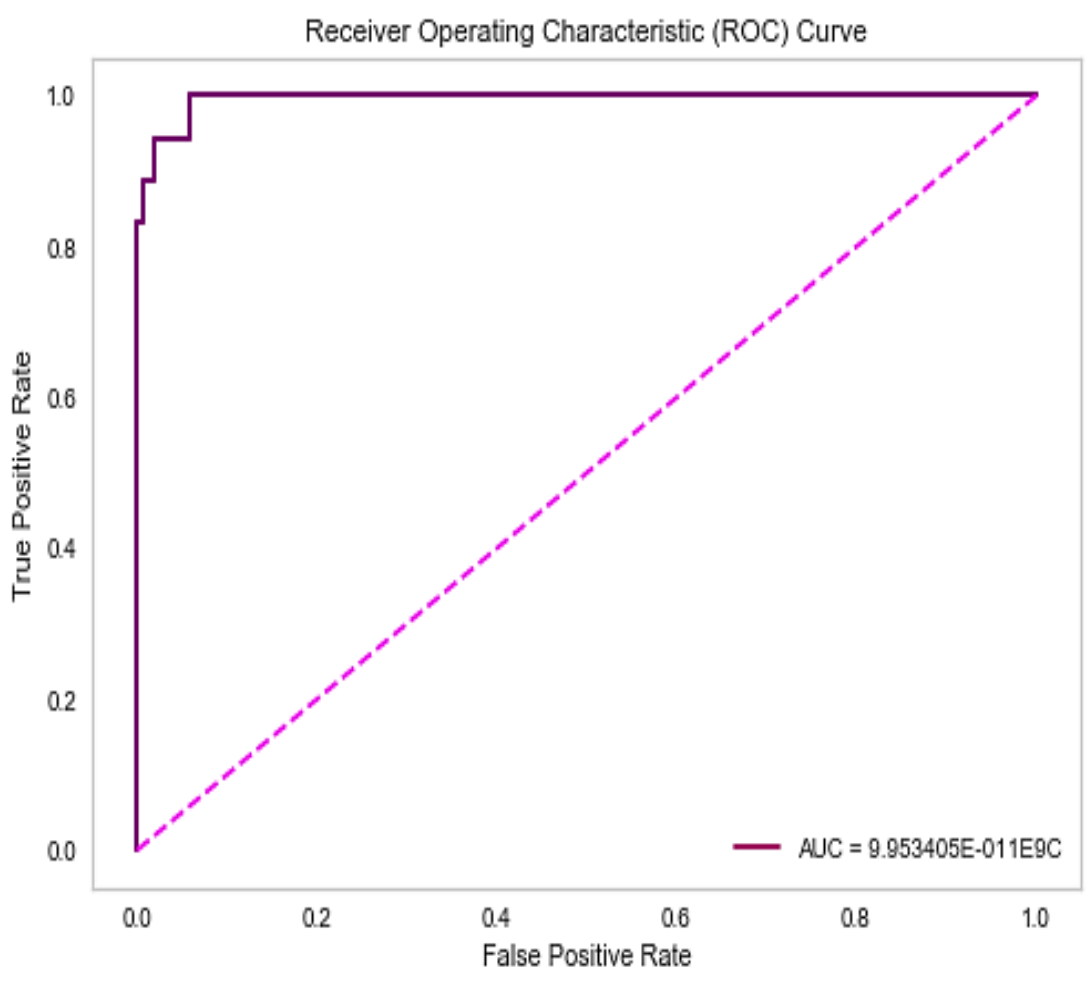
CONFUSION MATRIX AND CLASSIFICATION REPORT(SVM)

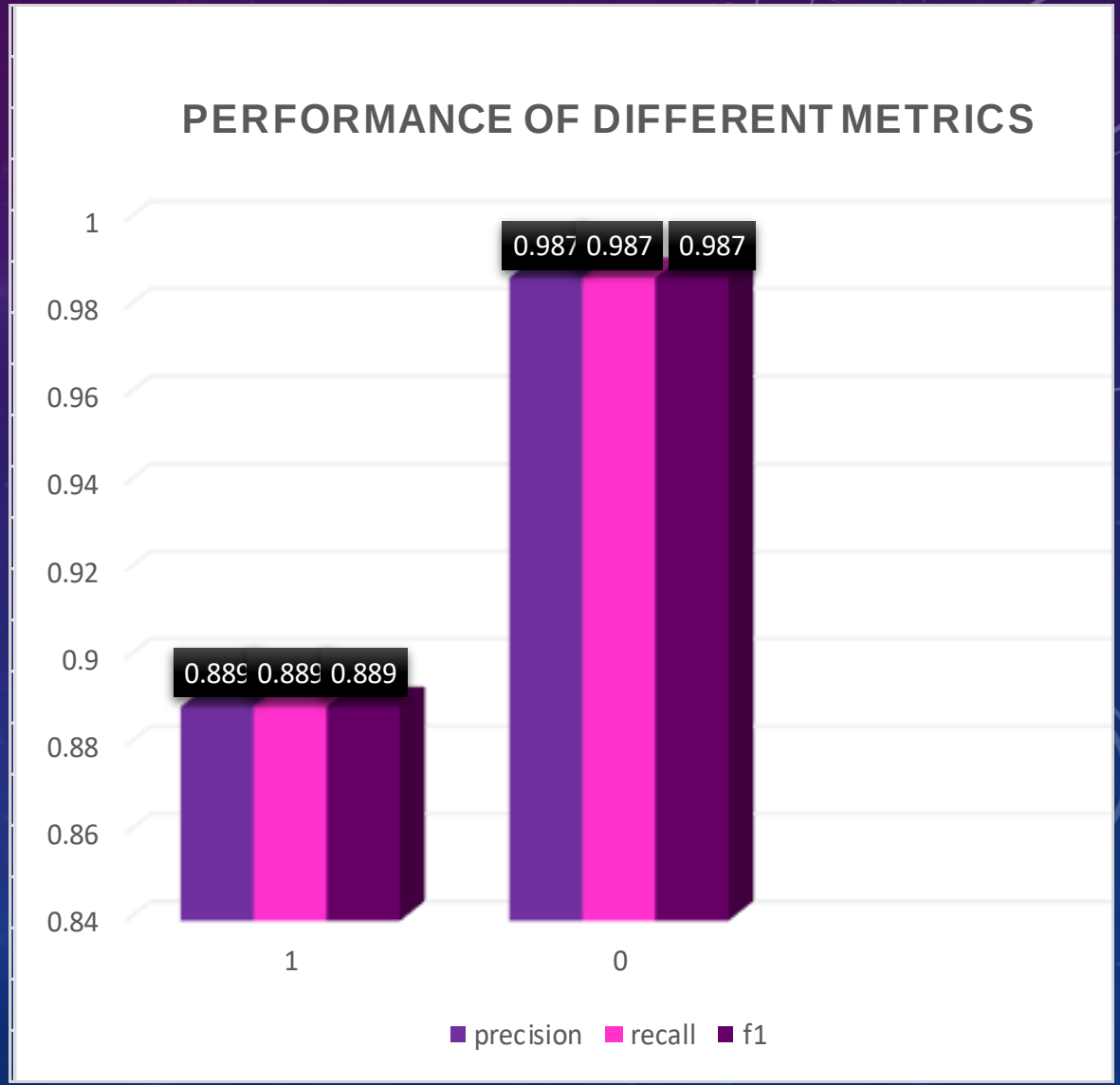




THE ROC CURVE(SVM)

The AUC = 99.5%

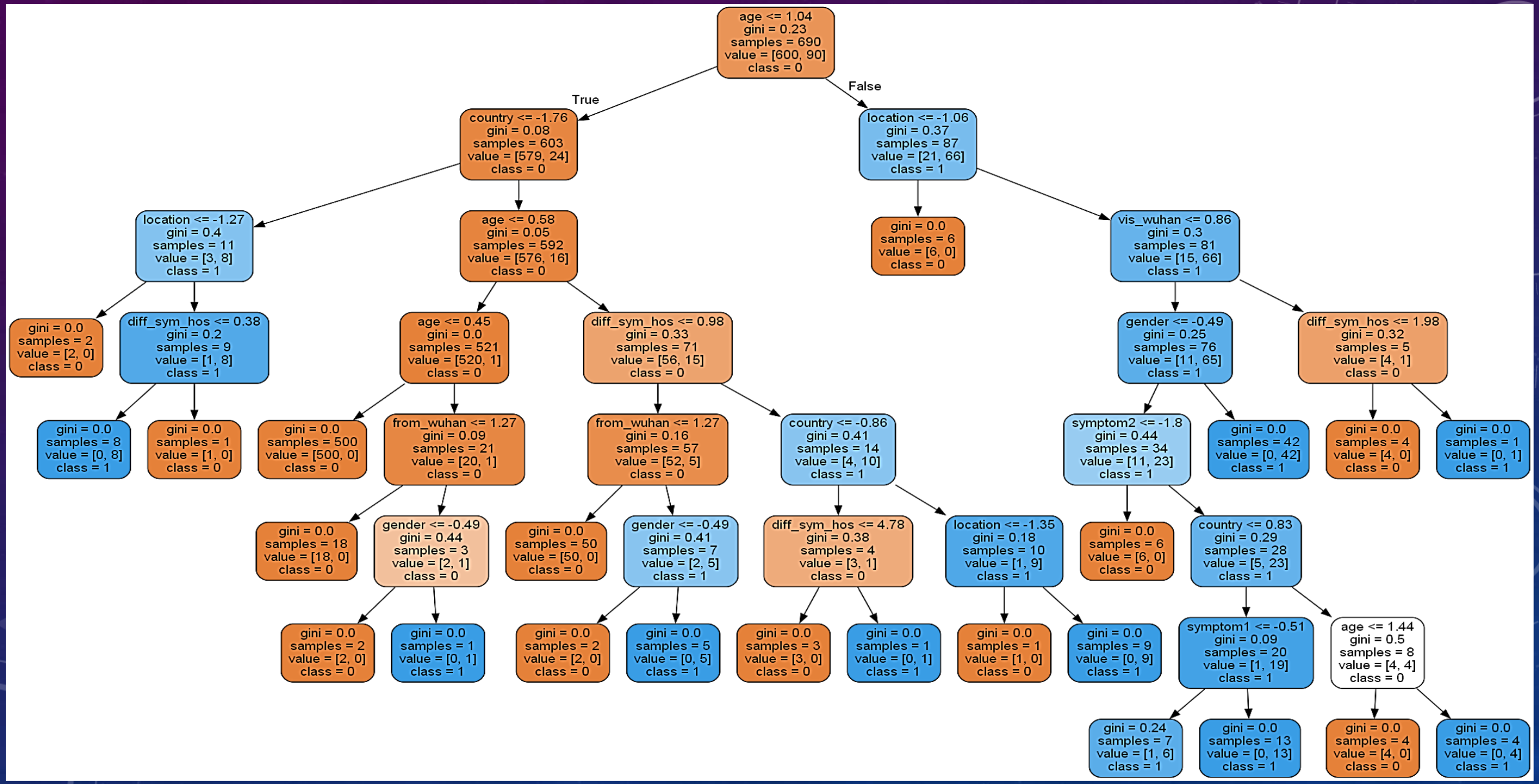




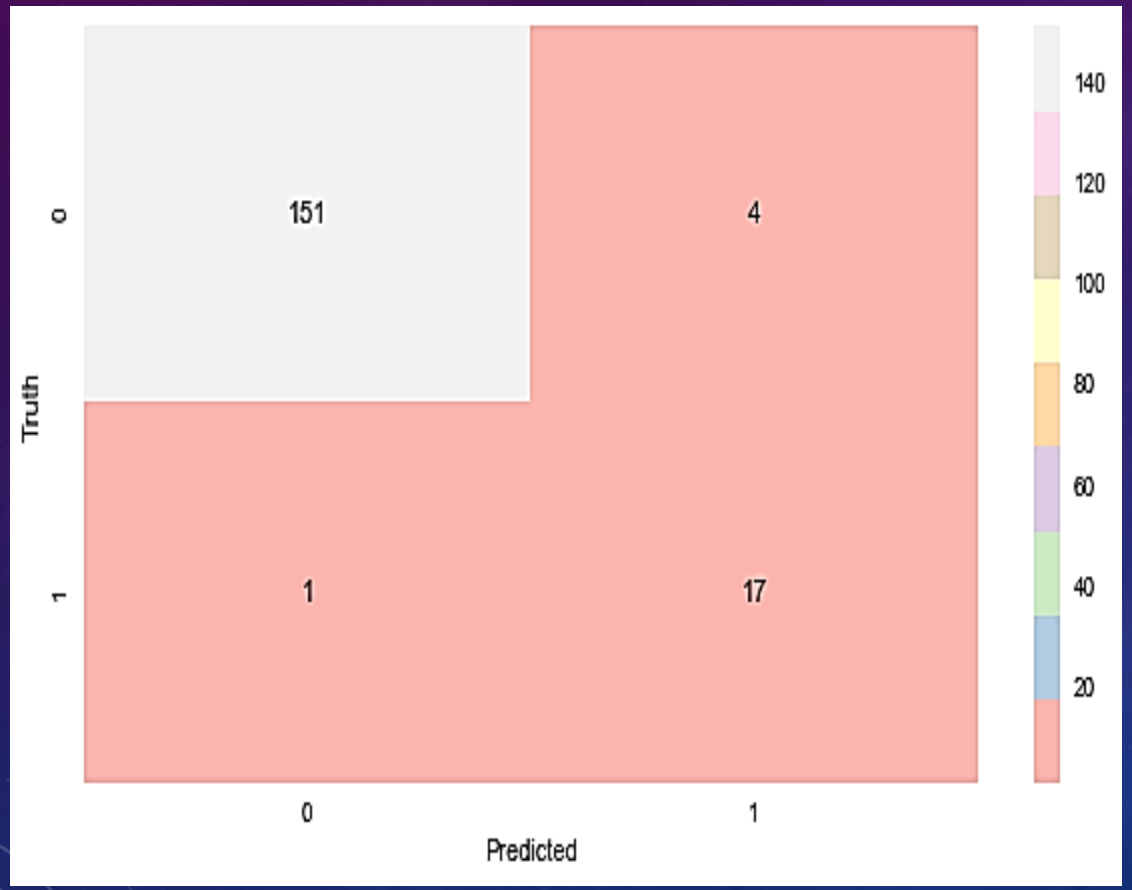
5. DECISION TREE (WITHOUT SMOTE)

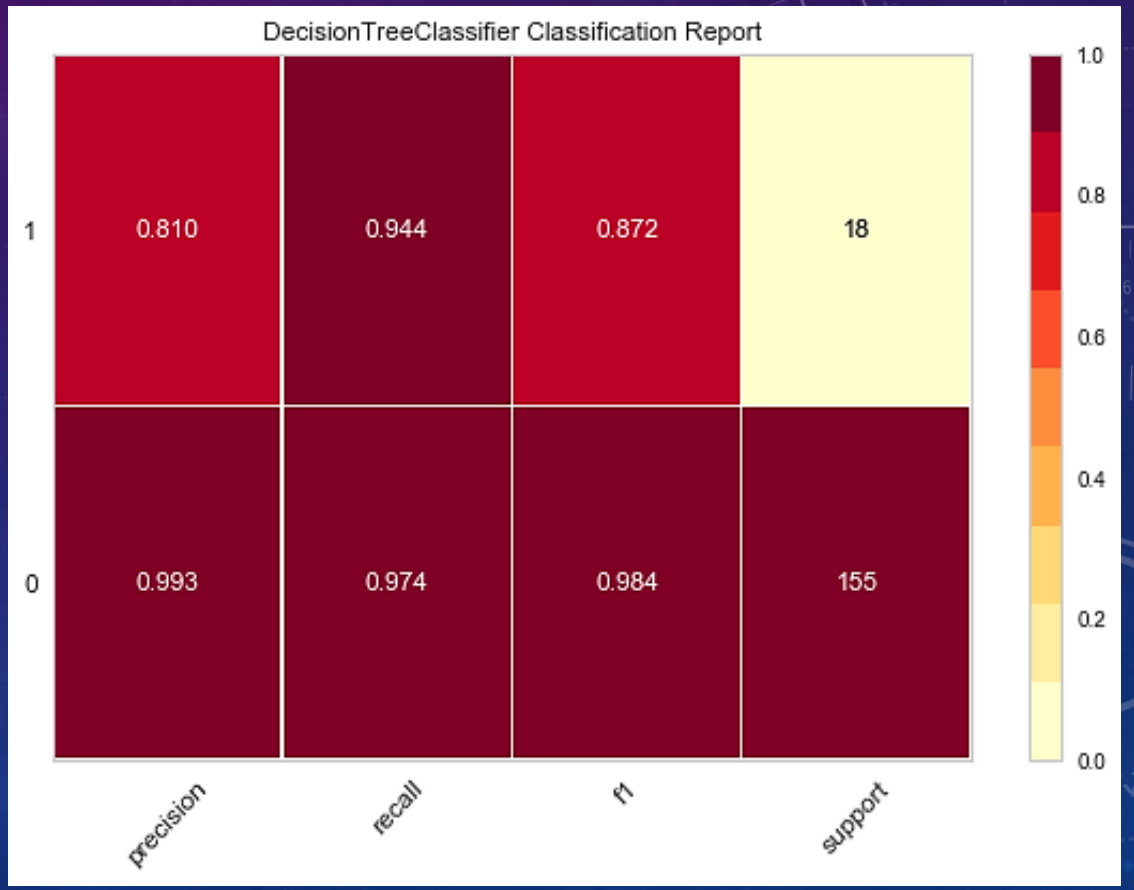
	Decision Tree	Decision Tree with Best parameters
Data Splitting	Training = 80 % Testing = 20 %	Cv= 10
Hyperparameters	'criterion': ['gini', 'entropy']	criterion= "gini"
	'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	max_depth= 7
	estimator=DecisionTreeClassifier(random_state=3)	estimator=DecisionTreeClassifier(criterion= "gini", max_depth= 7 , min_samples_leaf= 1 , random_state= 3)
	scoring='accuracy'	scoring='accuracy'
	'min_samples_leaf': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]	min_samples_leaf= 1

THE VISUALIZATION FOR THE TREE



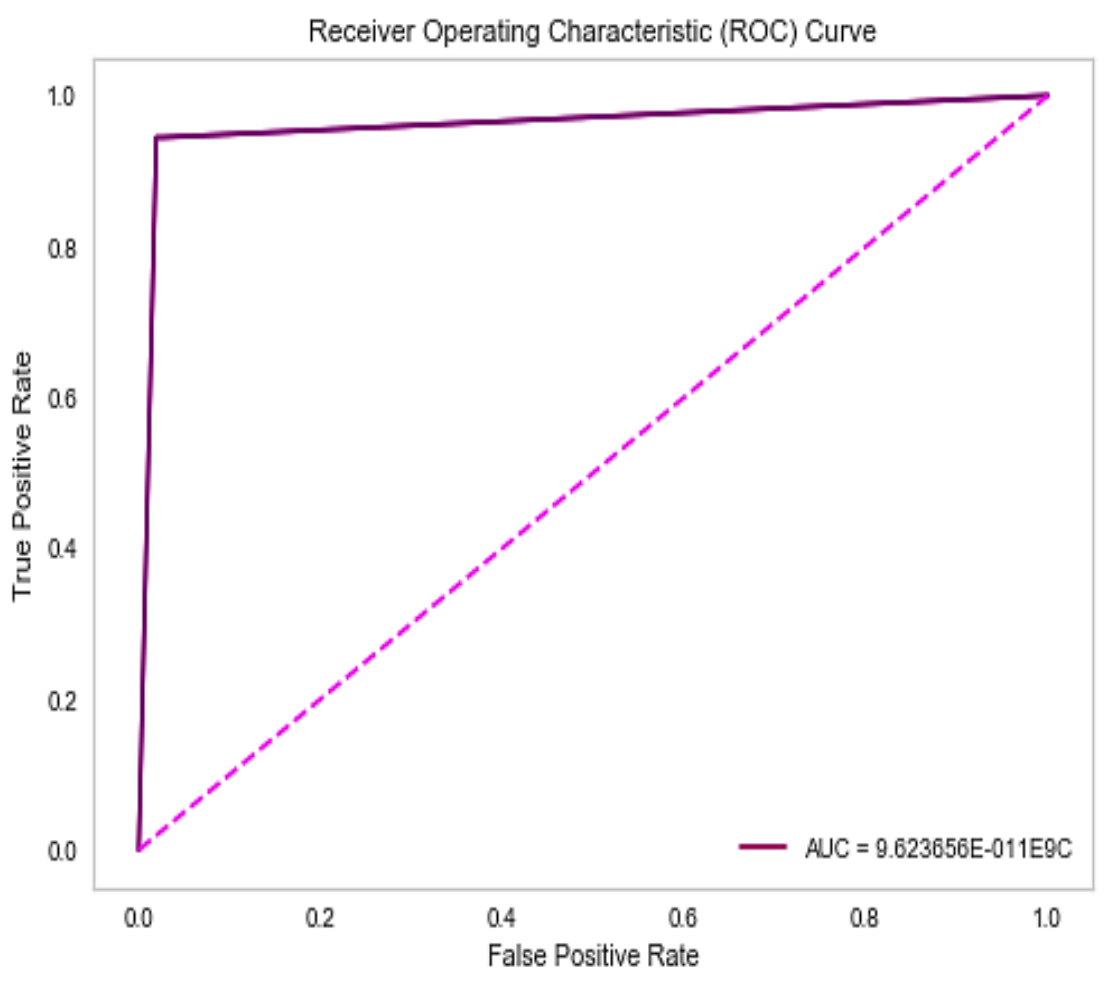
CONFUSION MATRIX AND CLASSIFICATION REPORT(DT)

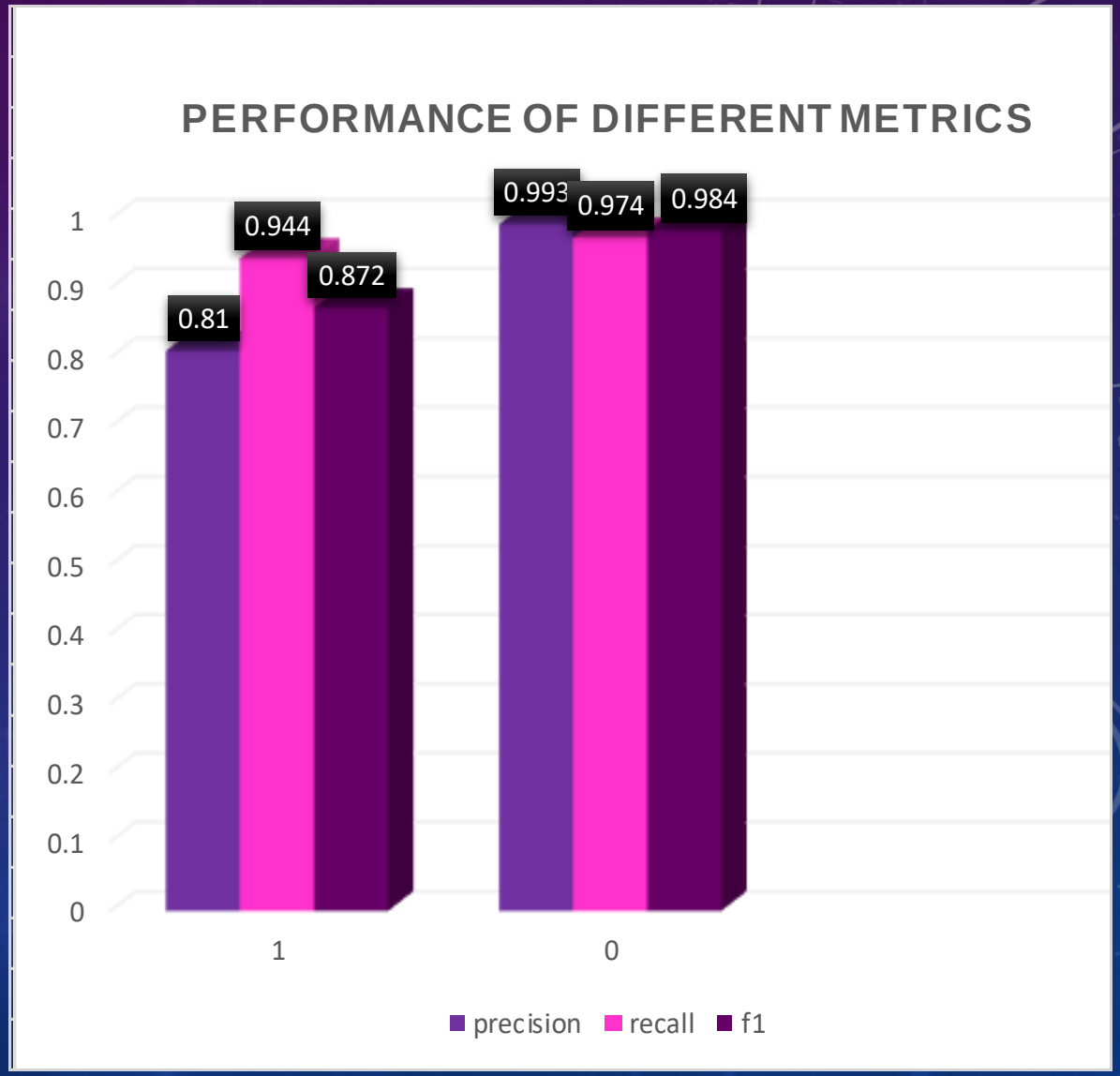




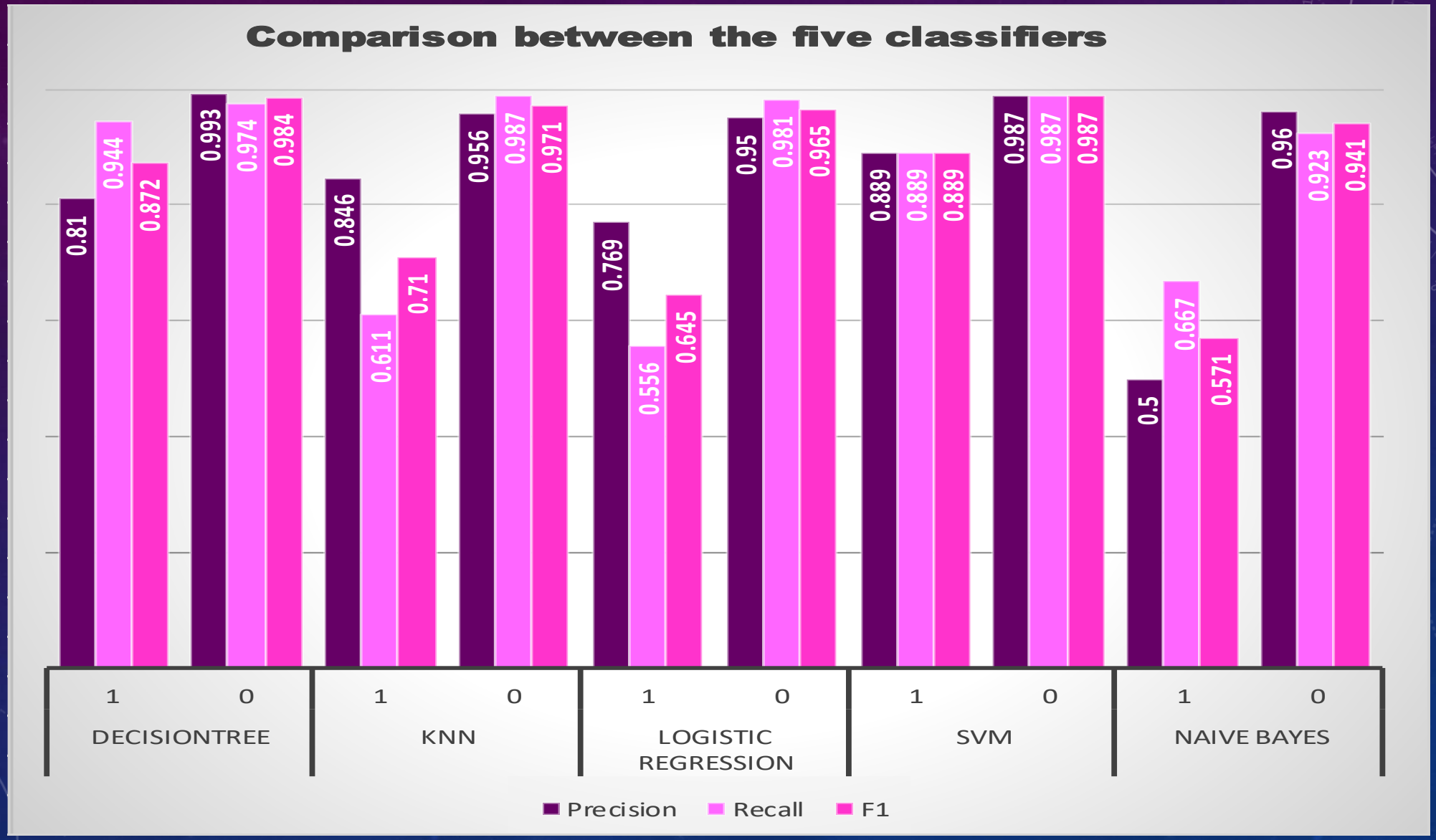
THE ROC CURVE(DT)

The AUC = 96.24%



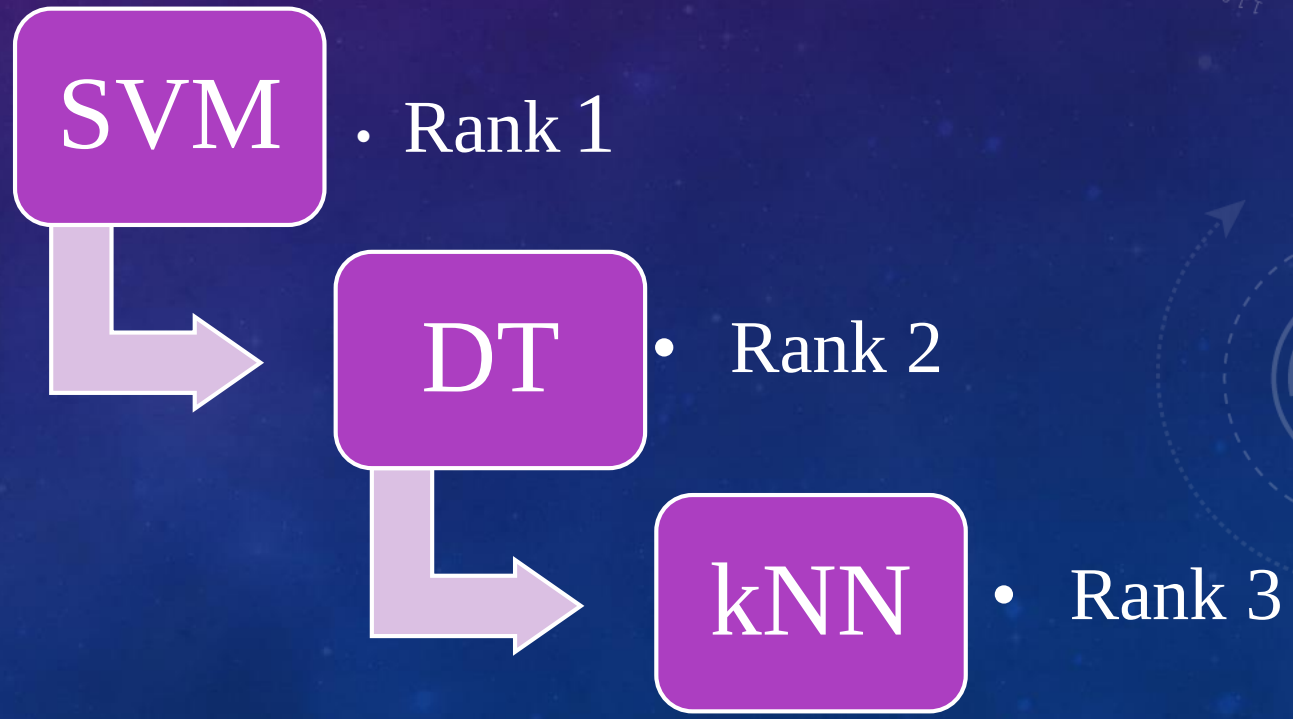


COMPARISON BETWEEN THE FIVE CLASSIFIERS



THE BEST MODEL

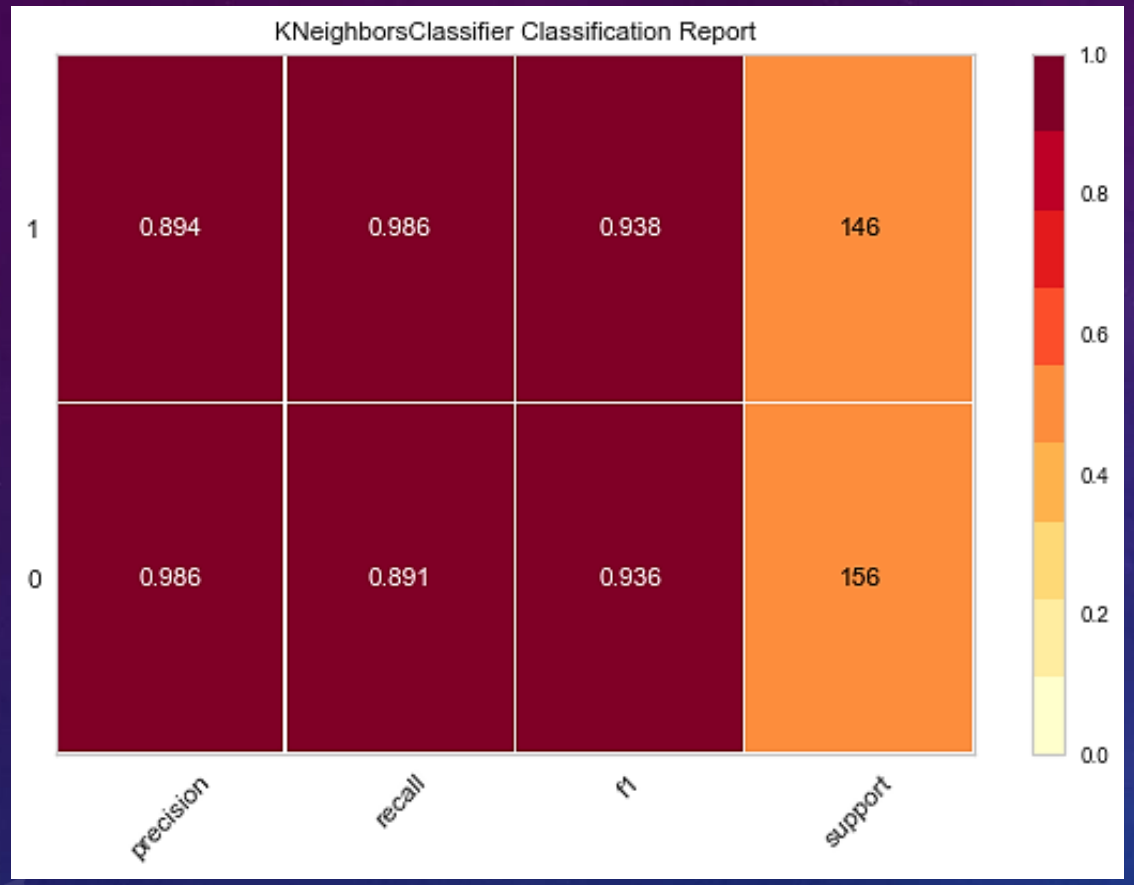
- The good news is that all models are above 90% accuracy.
- From visualization, we found that the SVM model **has** the highest one. Then Decision Tree with respect to
  - **Precision**
  - **Recall**
  - **F1: Score**
  - **Area Under ROC Curve** (AUROC)



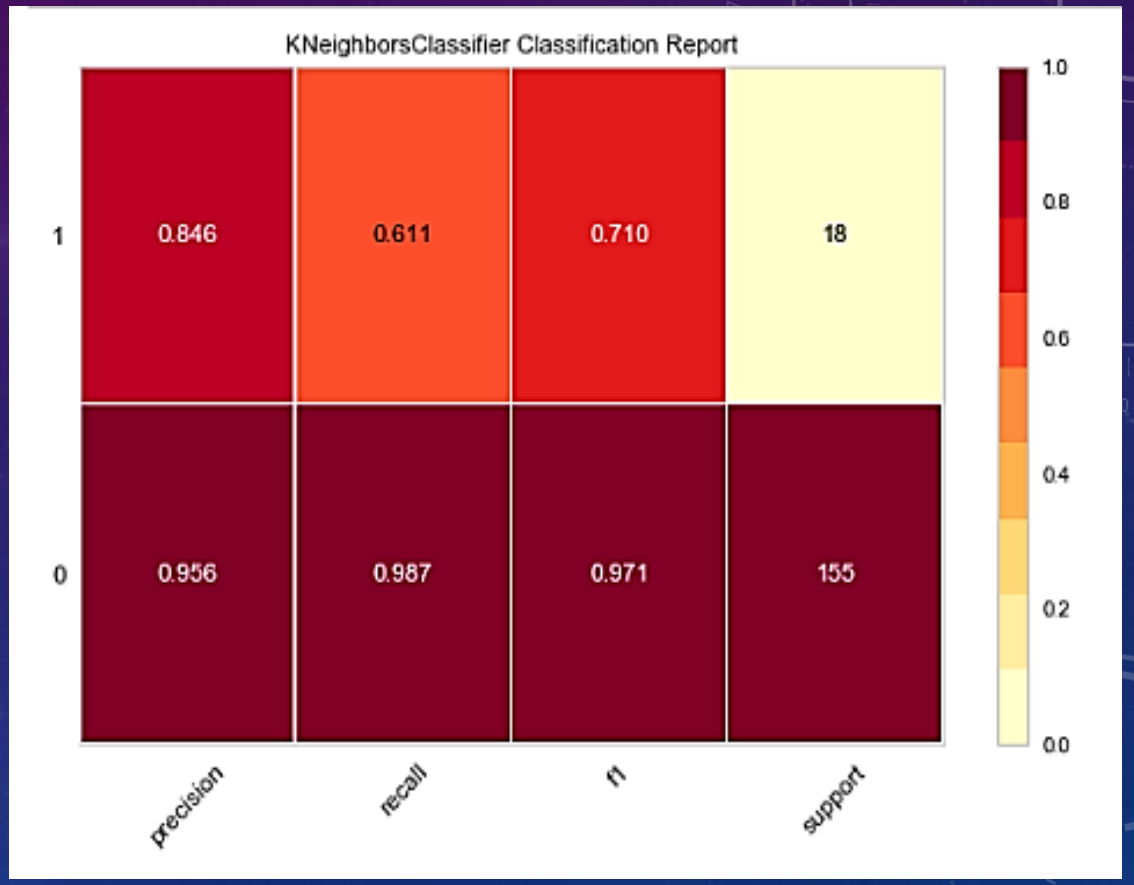
CLASS IMBALANCE IN MACHINE LEARNING

- Most machine learning algorithms work best when the number of samples in each class are about equal. This is because most algorithms are designed to maximize accuracy and reduce errors.
- However, if the data set in imbalance then In such cases, you get a pretty high accuracy just by predicting the **majority class**, but you fail to capture the **minority class**, which is most often the point of creating the model in the first place.
- Change the performance metric
  - Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be misleading.
- Metrics that can provide better insight are:
  - **Confusion Matrix**
  - **Precision**
  - **Recall**
  - **F1: Score**
  - **Area Under ROC Curve** (AUROC)

1. K-NEAREST NEIGHBORS (WITH SMOTE)

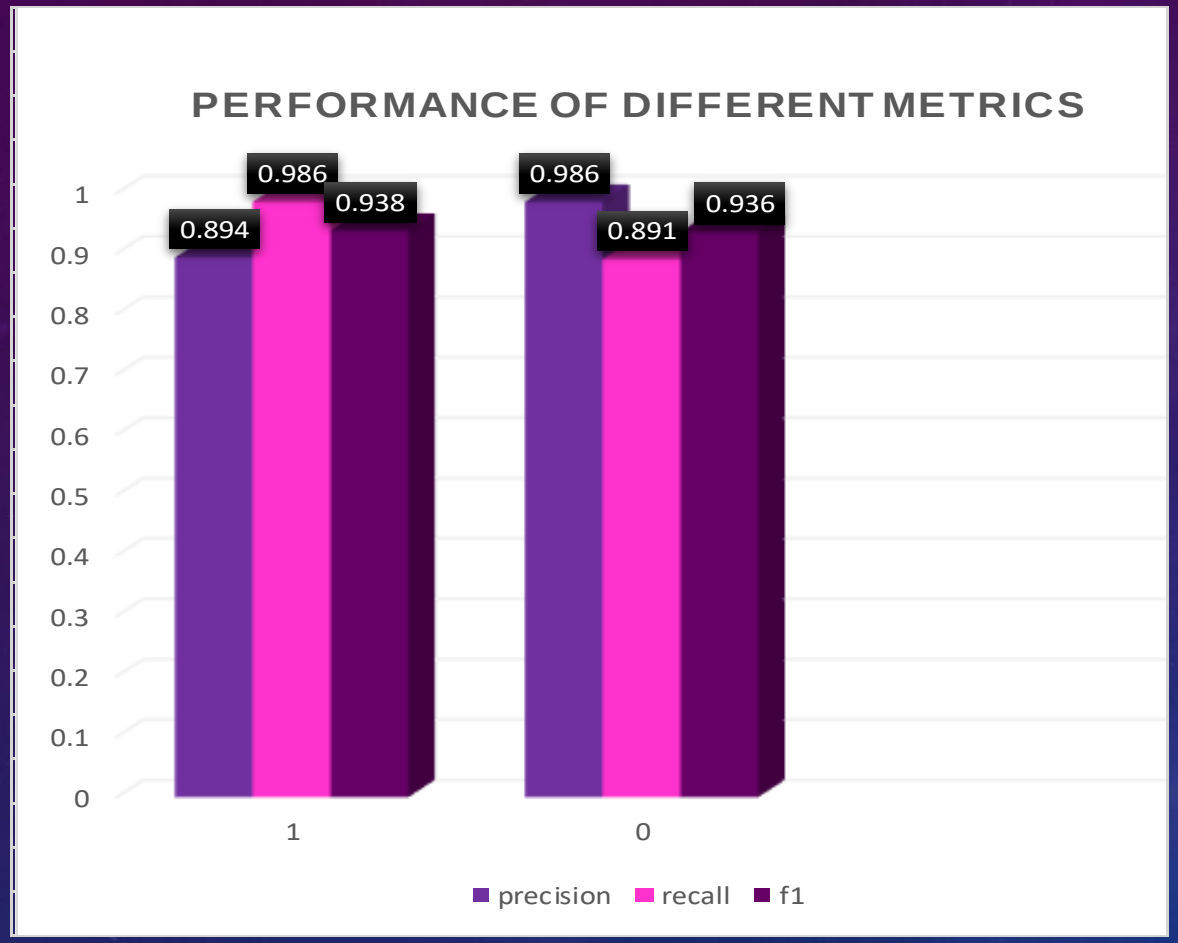


After Applying The SMOTE

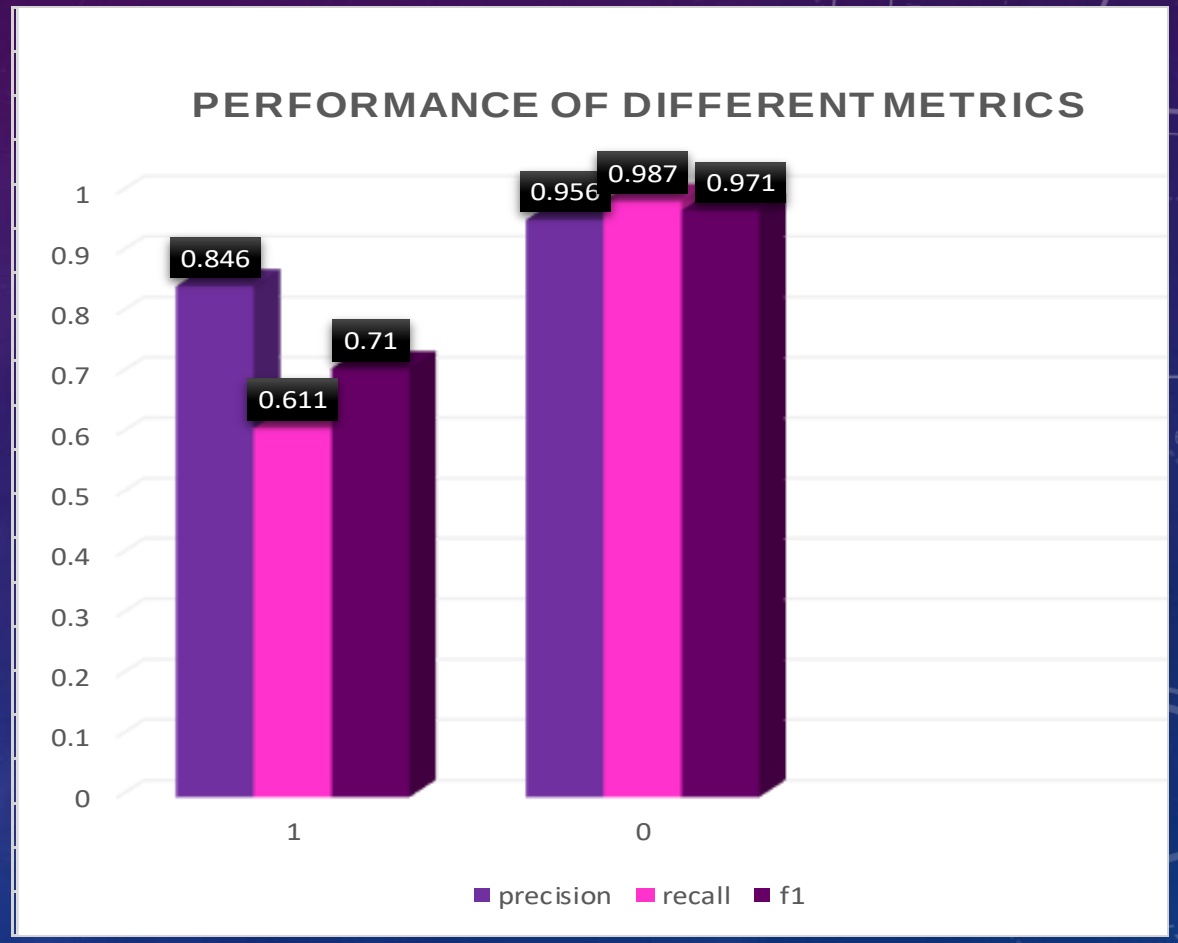


Before Applying The SMOTE

1. K-NEAREST NEIGHBORS (WITH SMOTE)



After Applying The SMOTE(The Best)

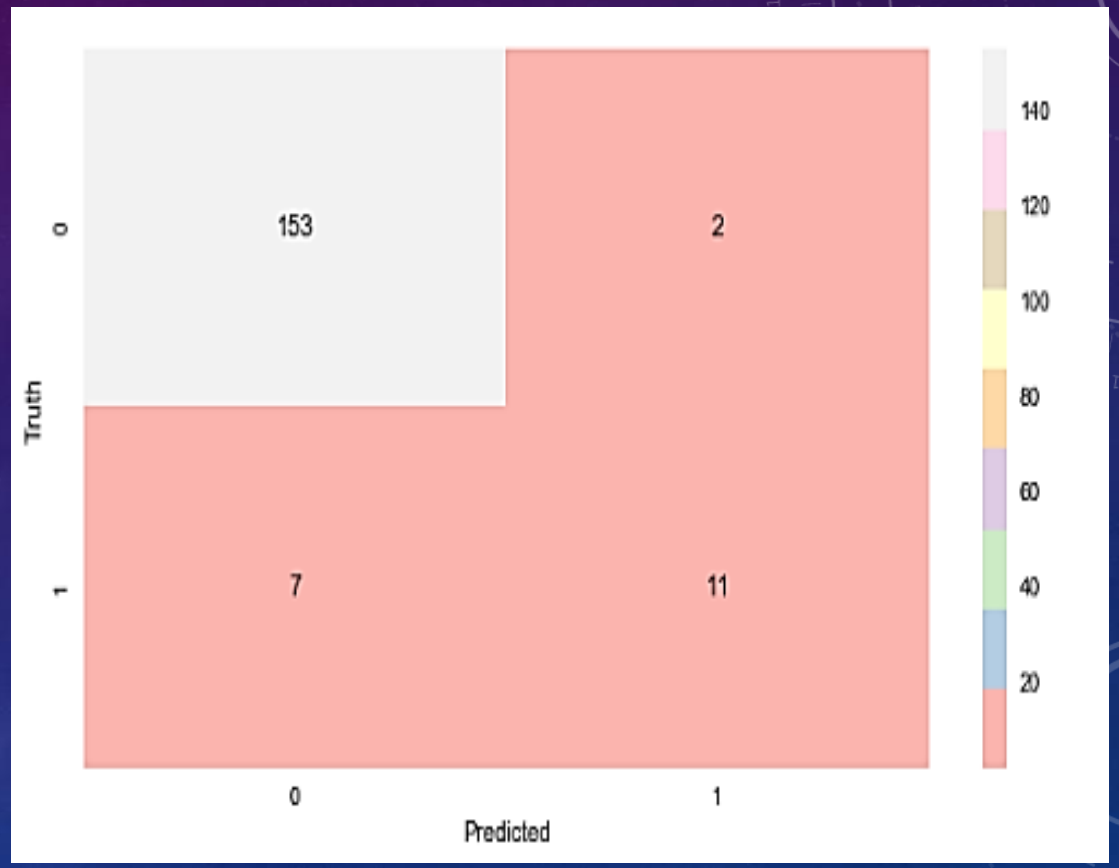


Before Applying The SMOTE

Performance	KNN(With SMOTE)	KNN(Without SMOTE)
Precision	0 => 98.6 % / 1 => 89.4 %	0 => 95.6 % / 1 => 84.6 %
Recall	0 = > 89.1 % / 1 => 98.6 %	0 = > 98.7 % / 1 => 61.1 %
F1-score	0 => 93.6% / 1 => 93.8 %	0 => 97.1 % / 1 => 71 %
ROC/AUC	98.2%	97.8%

1. K-NEAREST NEIGHBORS (WITH SMOTE)

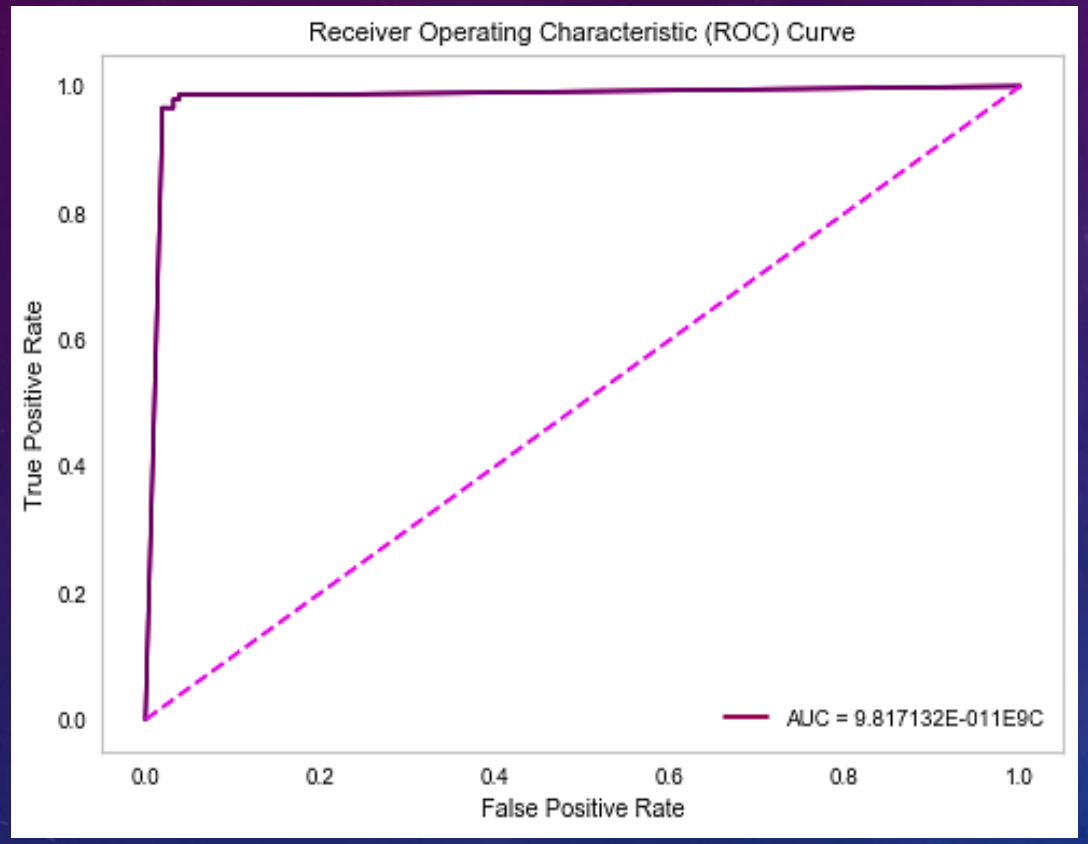
After Applying The SMOTE



Before Applying The SMOTE

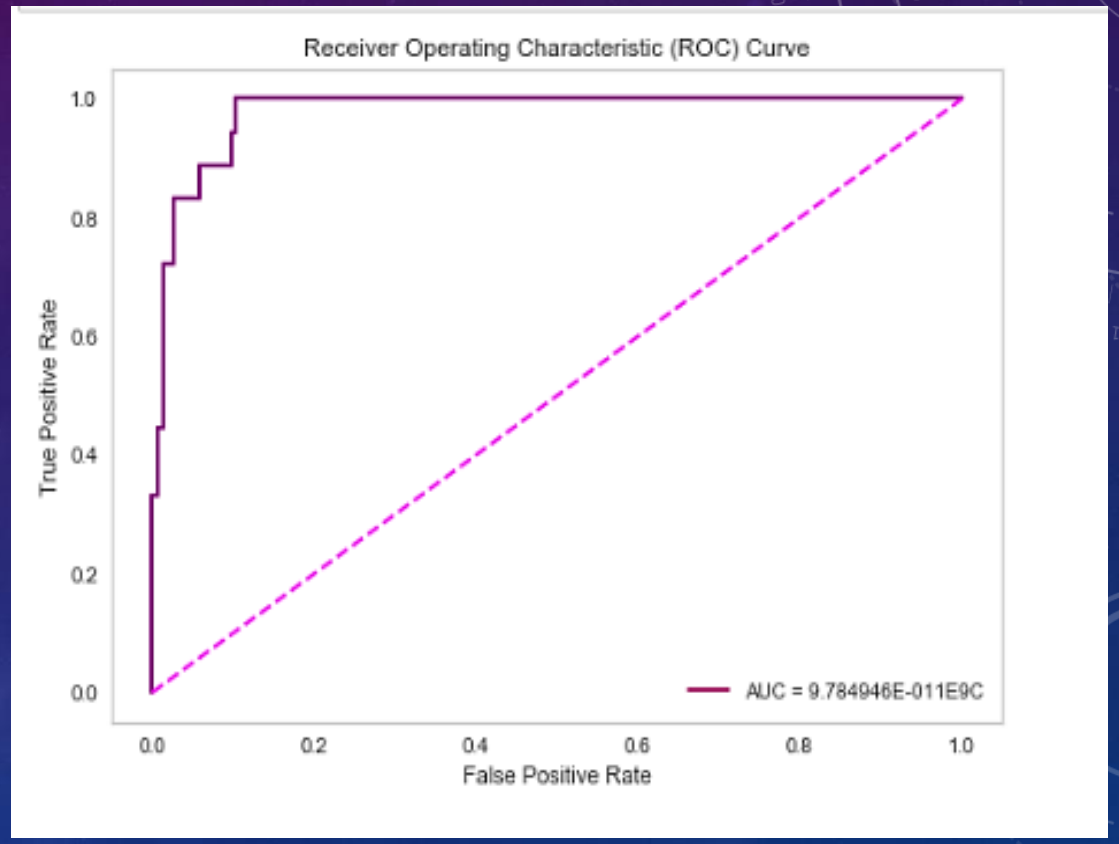
1. K-NEAREST NEIGHBORS (WITH SMOTE)

The AUC = 98.2 %



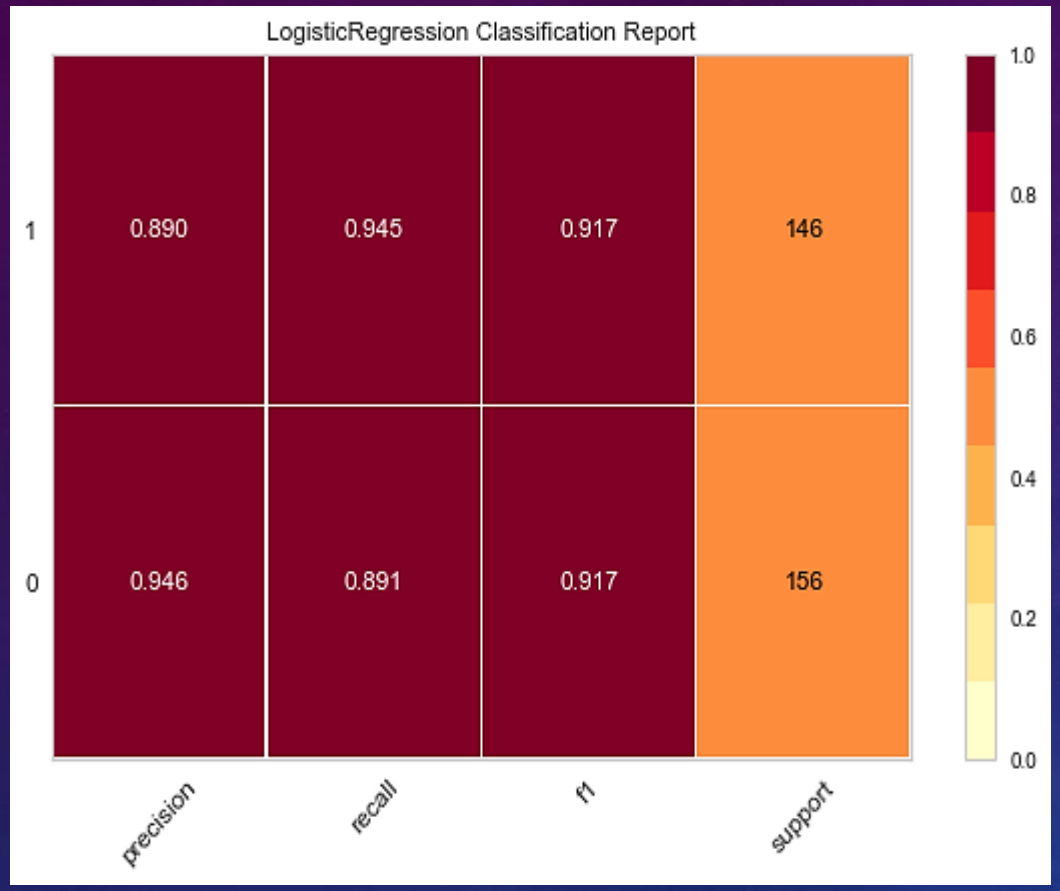
After Applying The SMOTE(The Best)

The AUC = 97.8 %

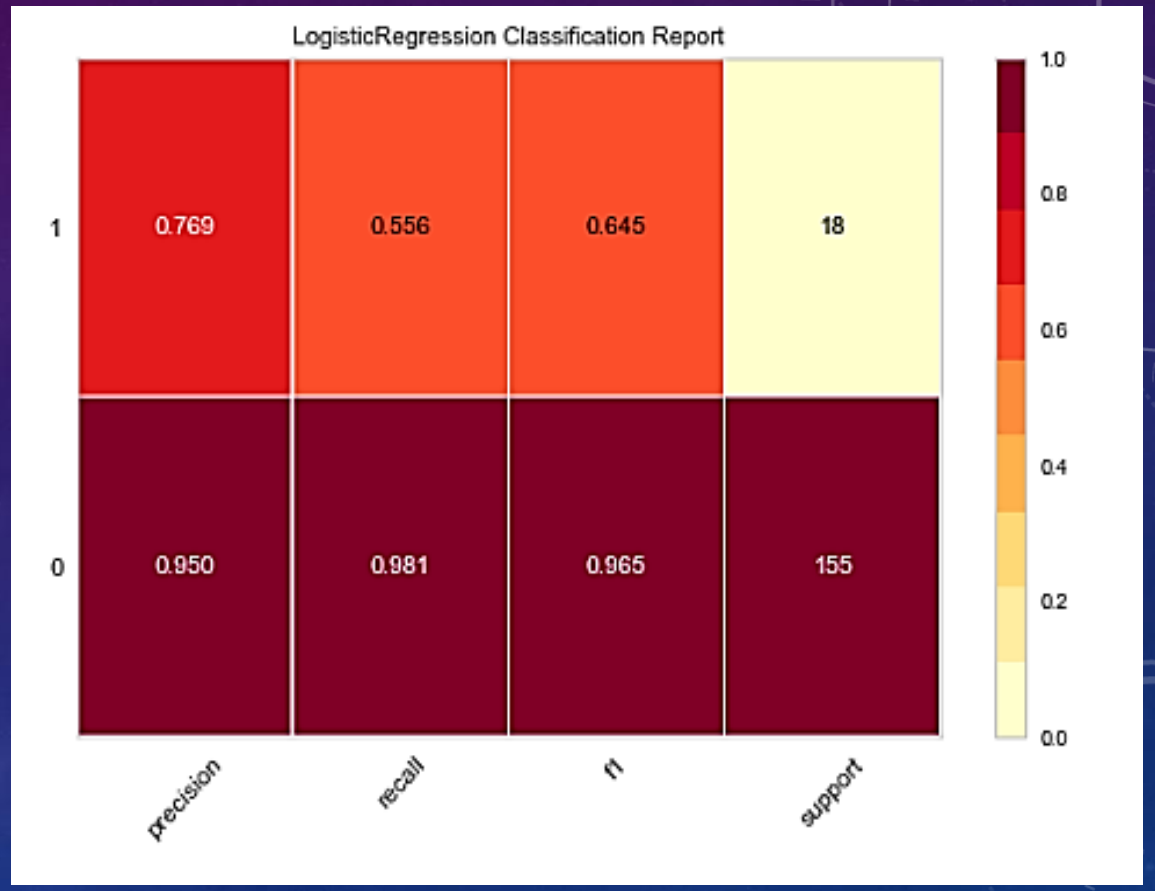


Before Applying The SMOTE

2. LOGISTIC REGRESSION (WITH SMOTE)

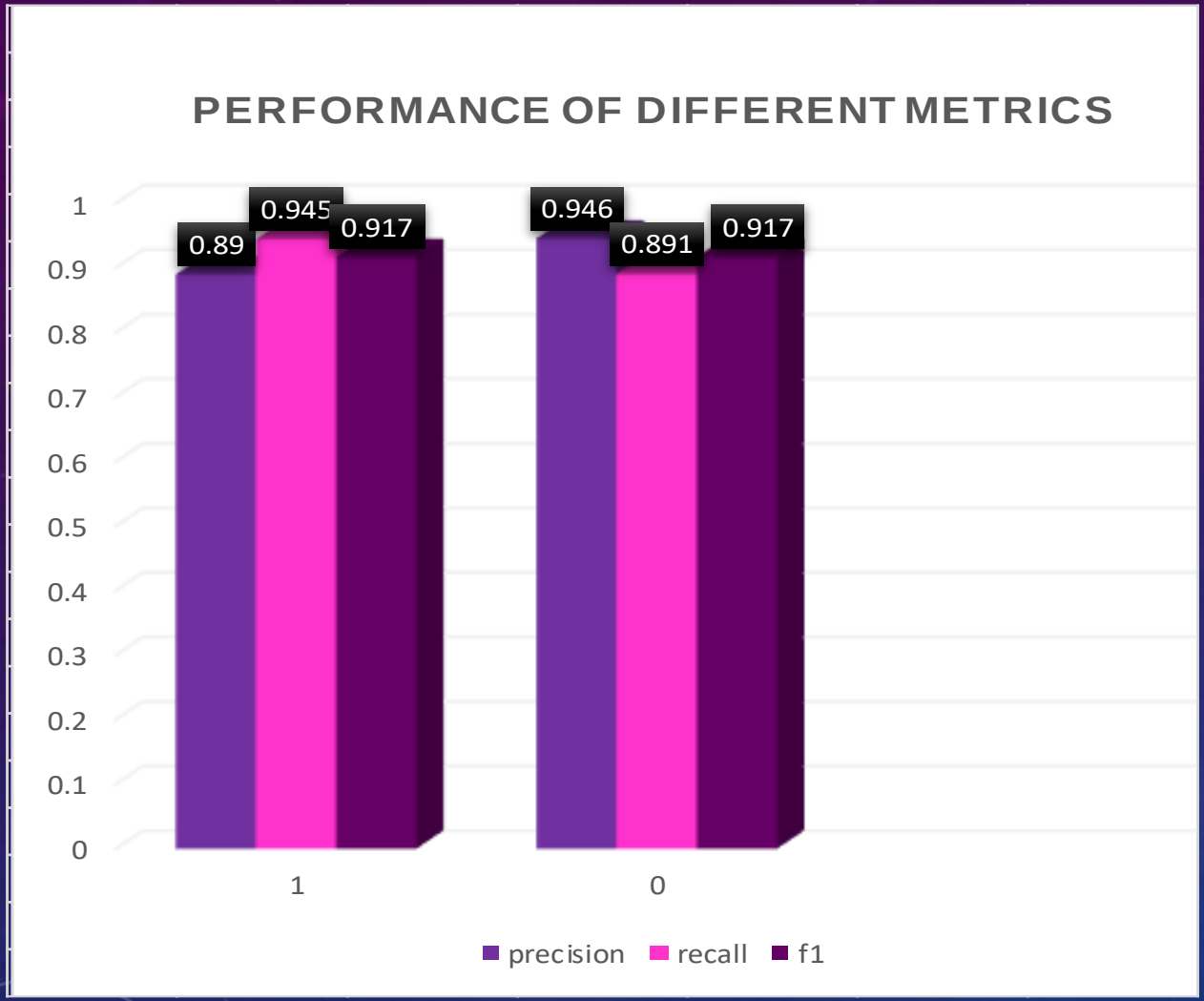


After Applying The SMOTE(The Best)

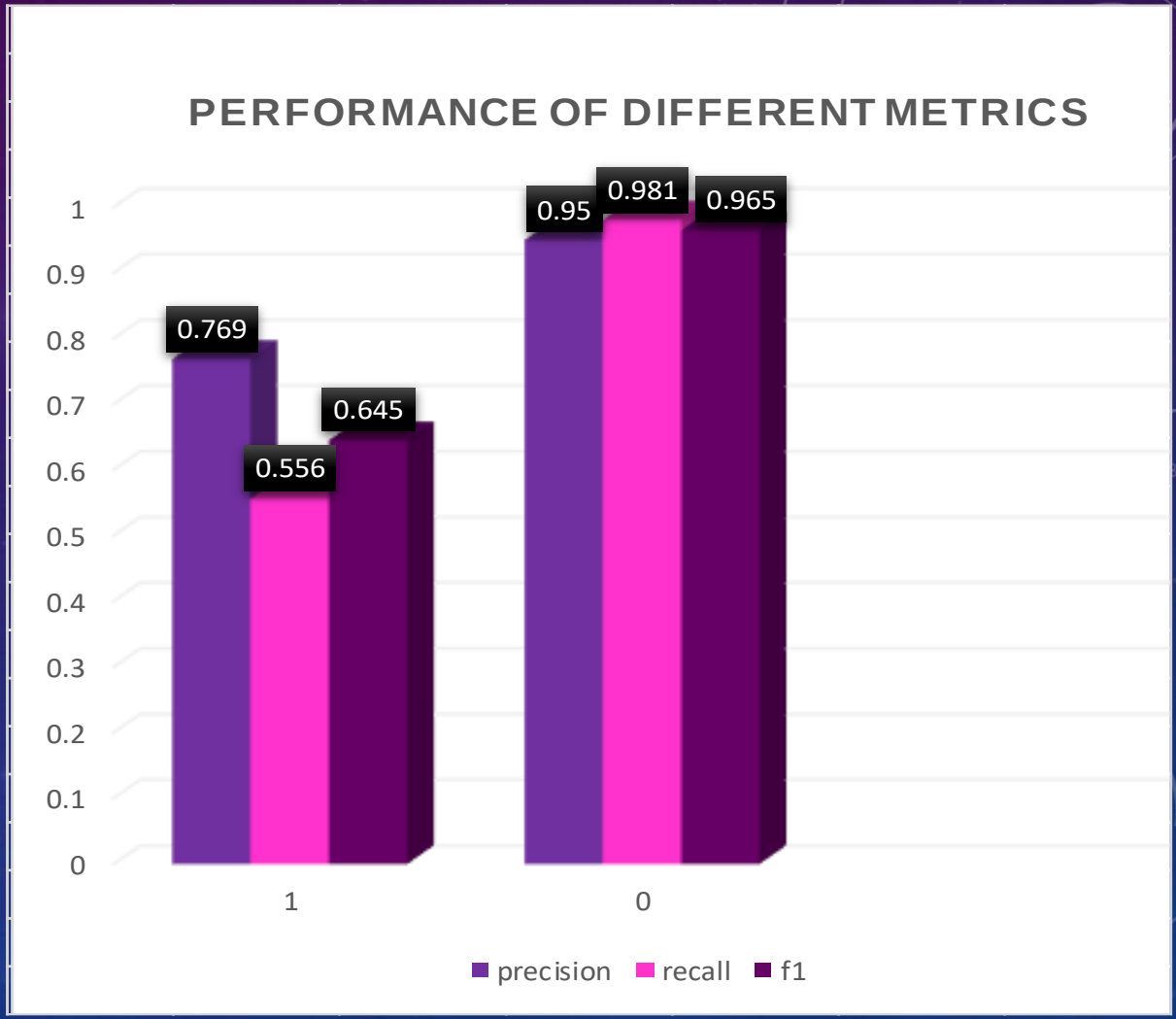


Before Applying The SMOTE

2. LOGISTIC REGRESSION (WITH SMOTE)



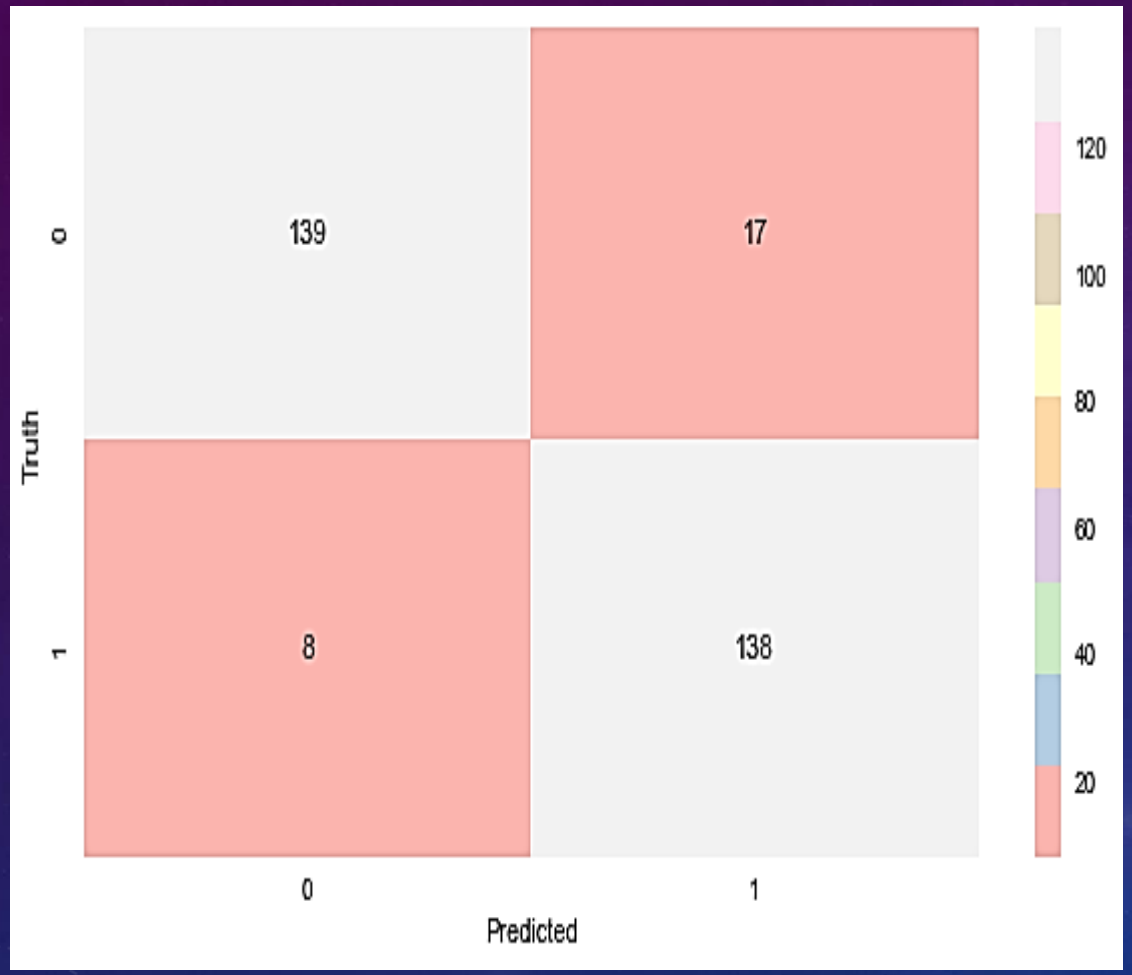
After Applying The SMOTE(The Best)



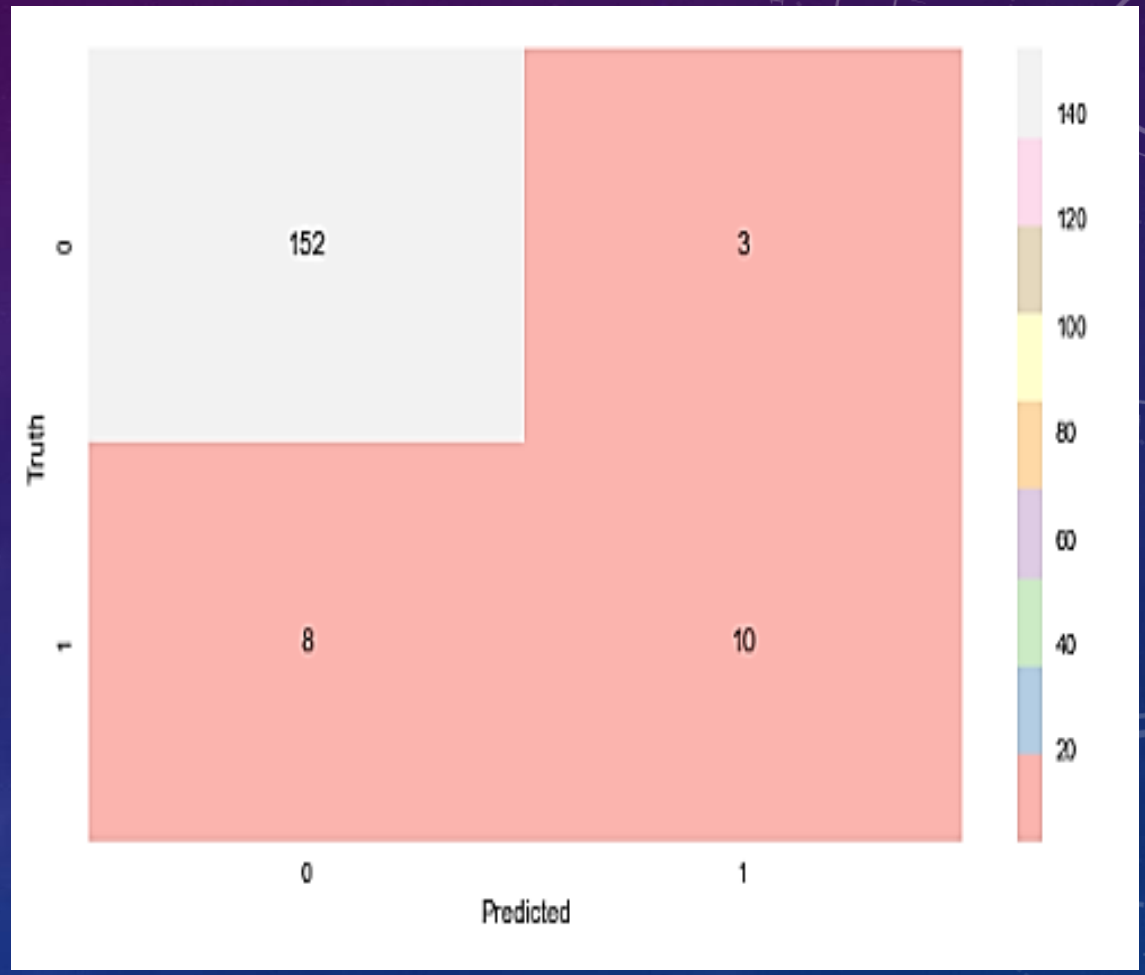
Before Applying The SMOTE

Performance	LogReg(With SMOTE)	LogReg(Without SMOTE)
Precision	0 => 94.6 % / 1 => 89 %	0 => 95 % / 1 => 76.9 %
Recall	0 = > 89.1 % / 1 => 94.5 %	0 = > 98.1% / 1 => 55.6 %
F1-score	0 => 91.7% / 1 => 91.7%	0 => 96.5 % / 1 => 64.5 %
ROC/AUC	95.5%	90.2%

2. LOGISTIC REGRESSION (WITH SMOTE)



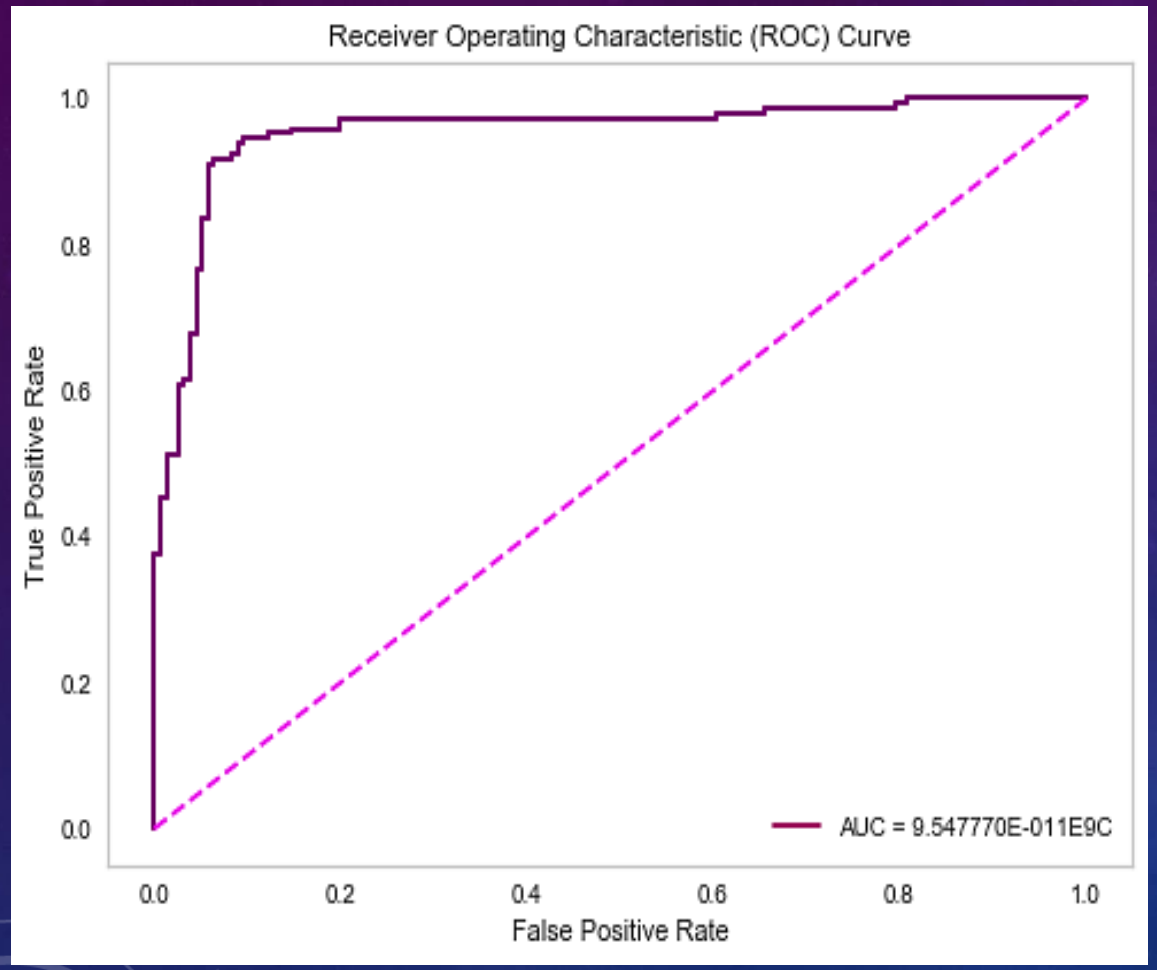
After Applying The SMOTE(The Best)



Before Applying The SMOTE

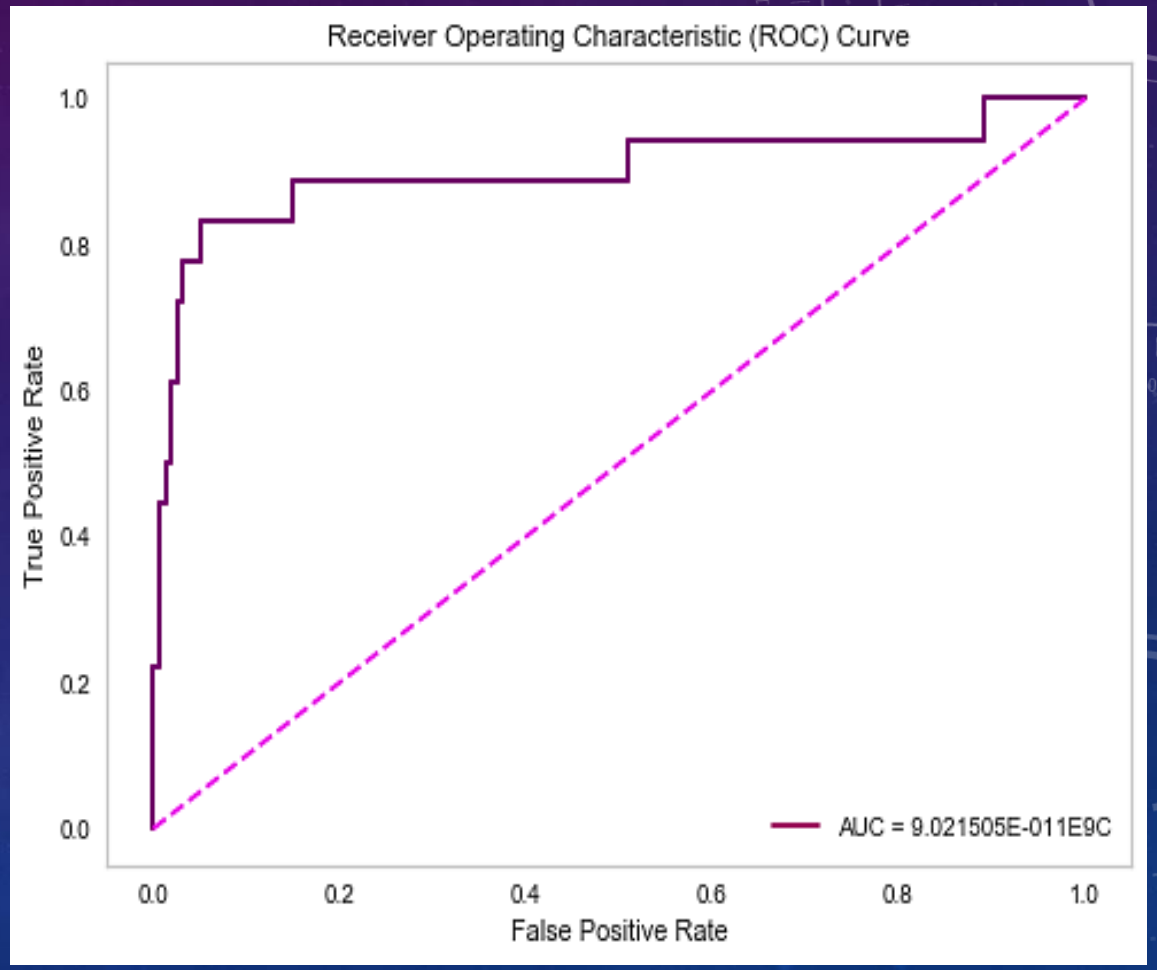
2. LOGISTIC REGRESSION (WITH SMOTE)

The AUC = 95.5%



After Applying The SMOTE(The Best)

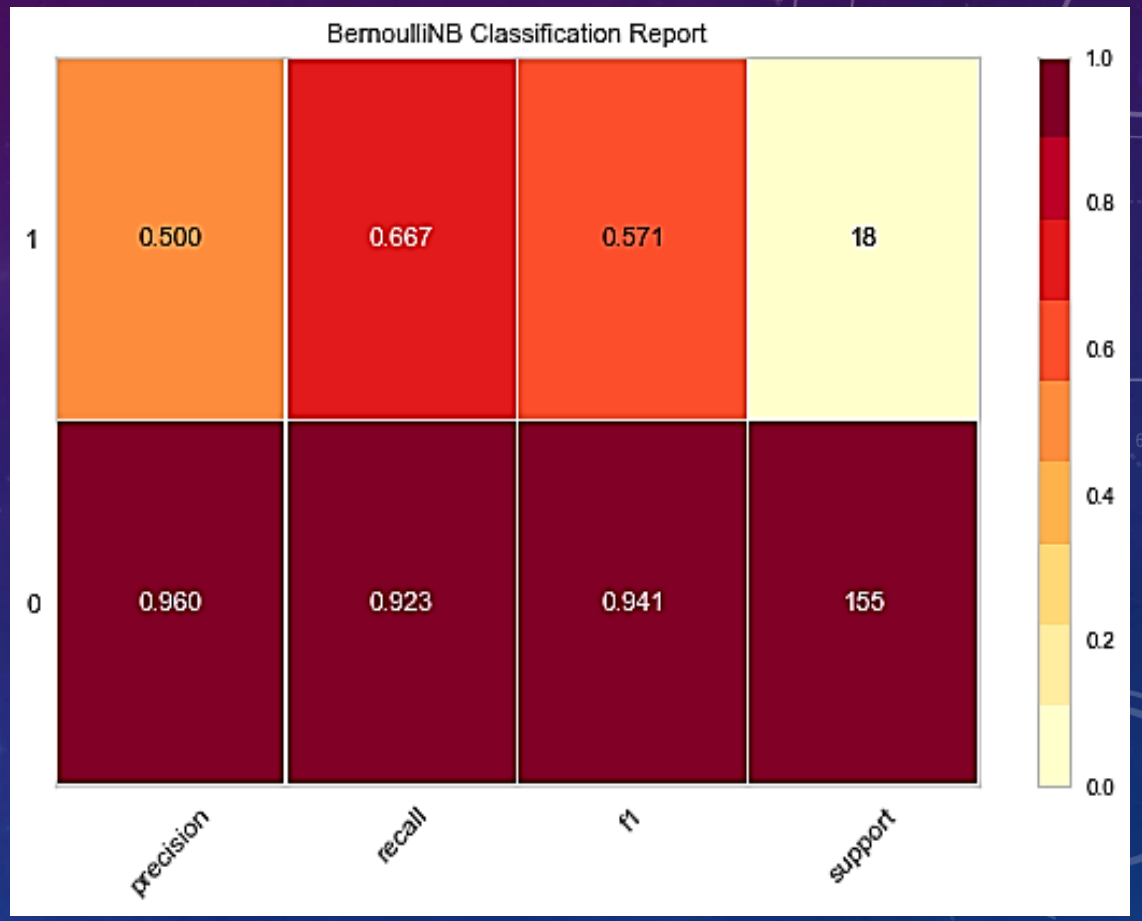
The AUC = 90.2%



Before Applying The SMOTE

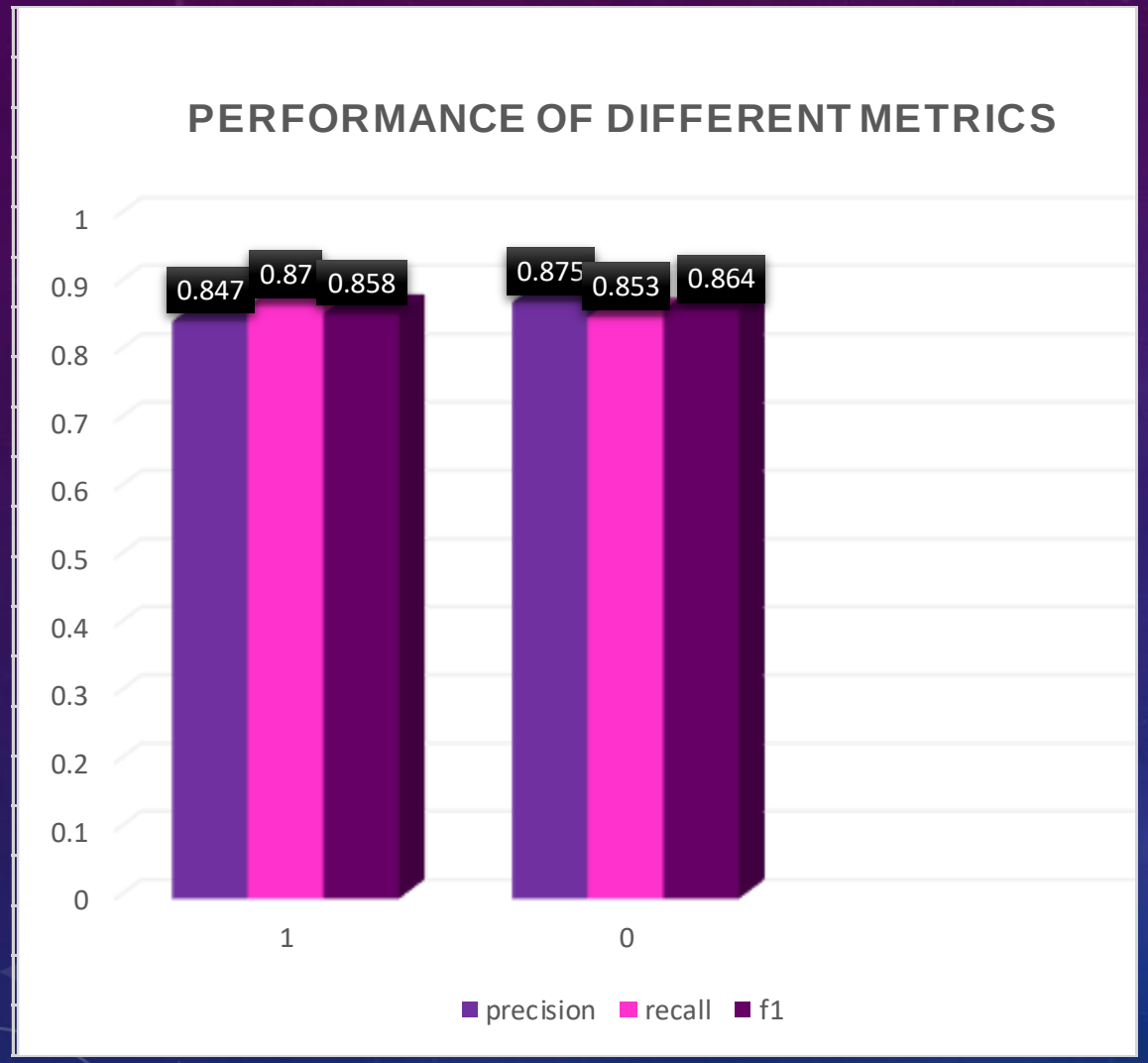
3. BERNOULLI NAIVE BAYES (WITH SMOTE)

After Applying The SMOTE(The Best)

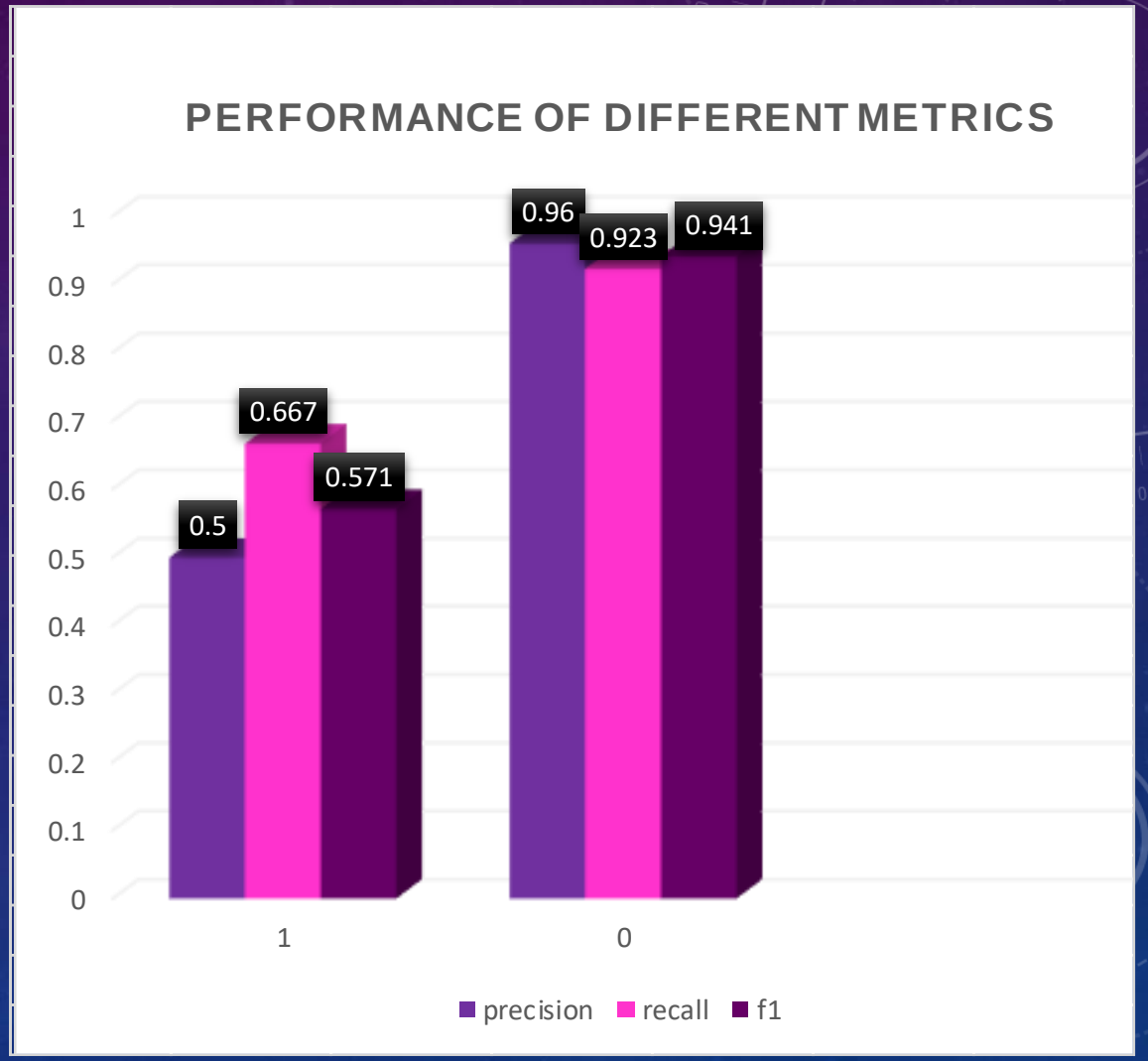


Before Applying The SMOTE

3. BERNOULLI NAIVE BAYES (WITH SMOTE)



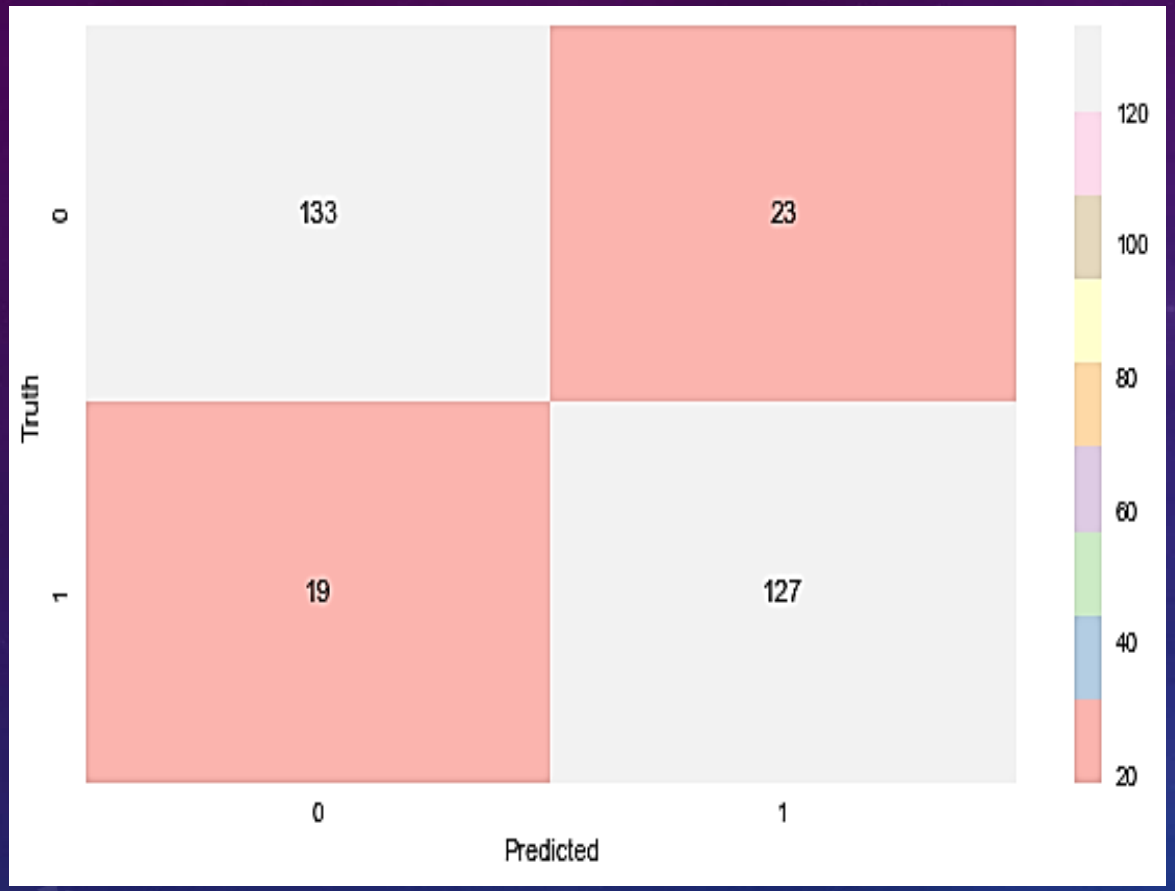
After Applying The SMOTE(The Best)



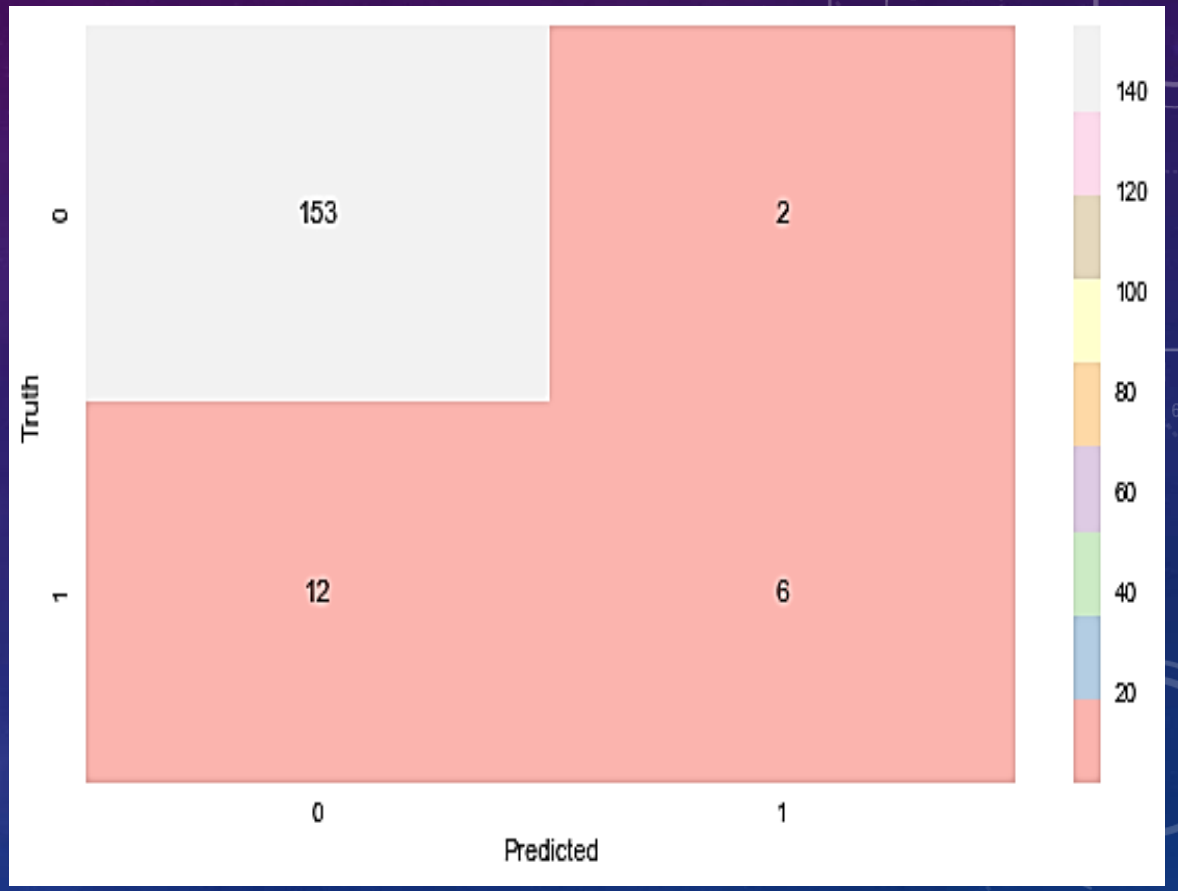
Before Applying The SMOTE

Performance	Naive Bayes(With SMOTE)	Naive Bayes(Without SMOTE)
Precision	0 => 87.5 % / 1 => 84.7 %	0 => 96 % / 1 => 50 %
Recall	0 = > 85.3 % / 1 => 87 %	0 = > 92.3% / 1 => 66.7 %
F1-score	0 => 86.4% / 1 => 85.8%	0 => 94.1 % / 1 => 57.1 %
ROC/AUC	92.1%	94.2%

3. BERNOULLI NAIVE BAYES (WITH SMOTE)



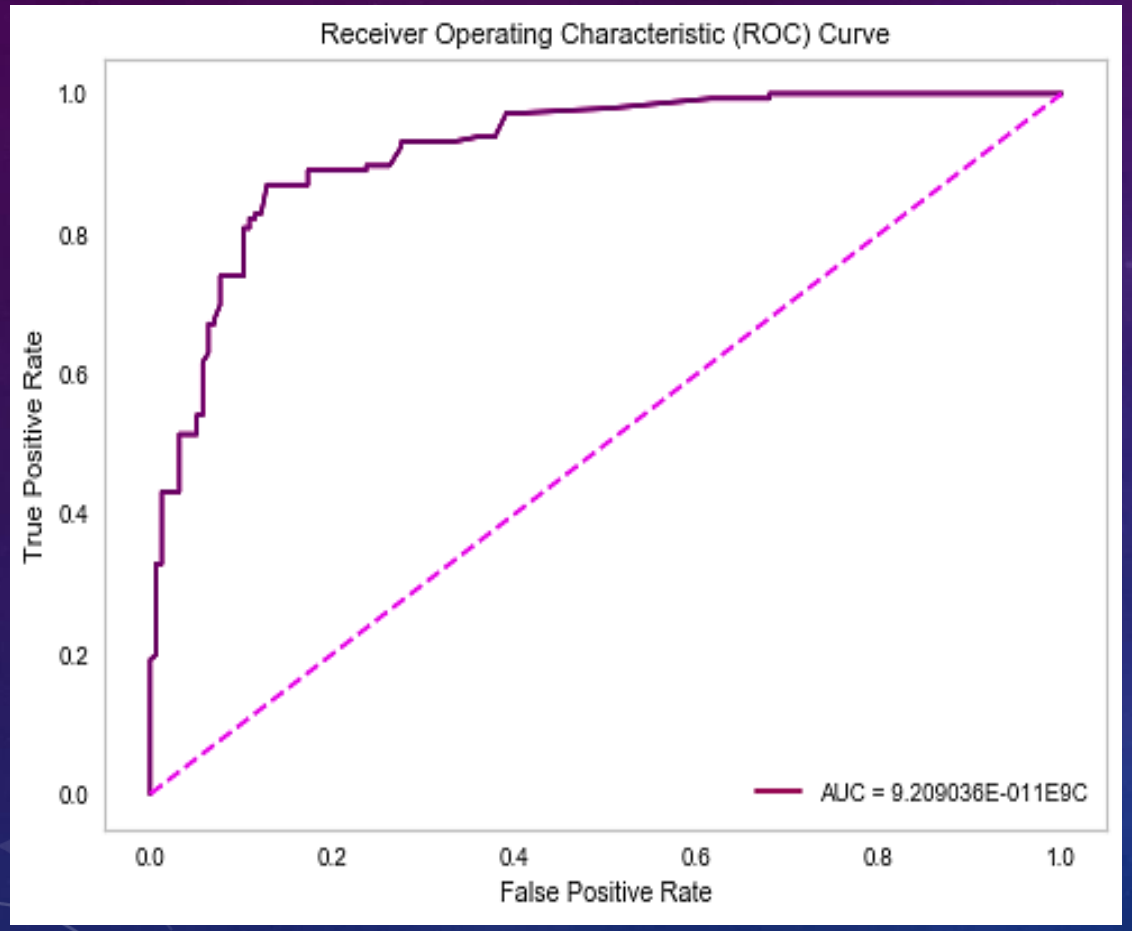
After Applying The SMOTE(The Best)



Before Applying The SMOTE

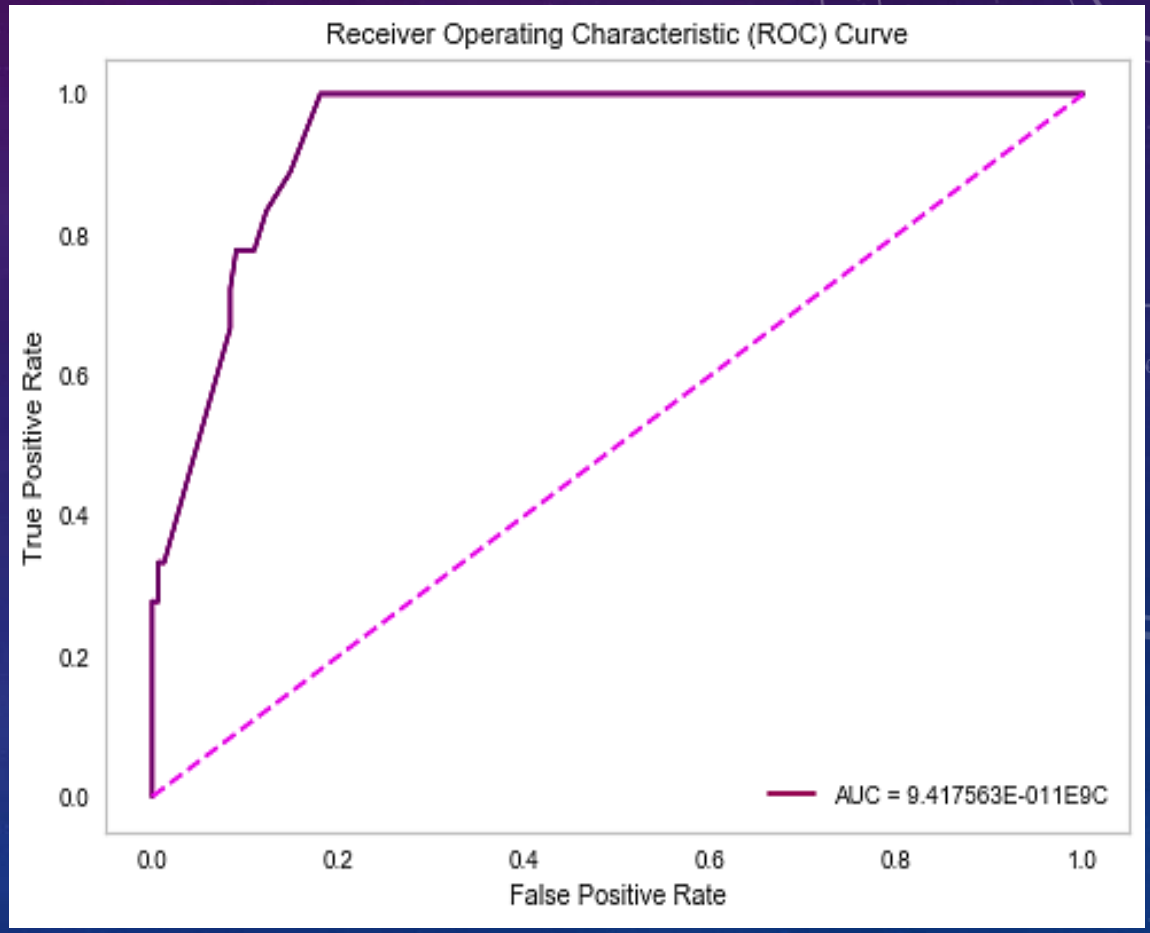
3. BERNOULLI NAIVE BAYES (WITH SMOTE)

The AUC = 92.1%



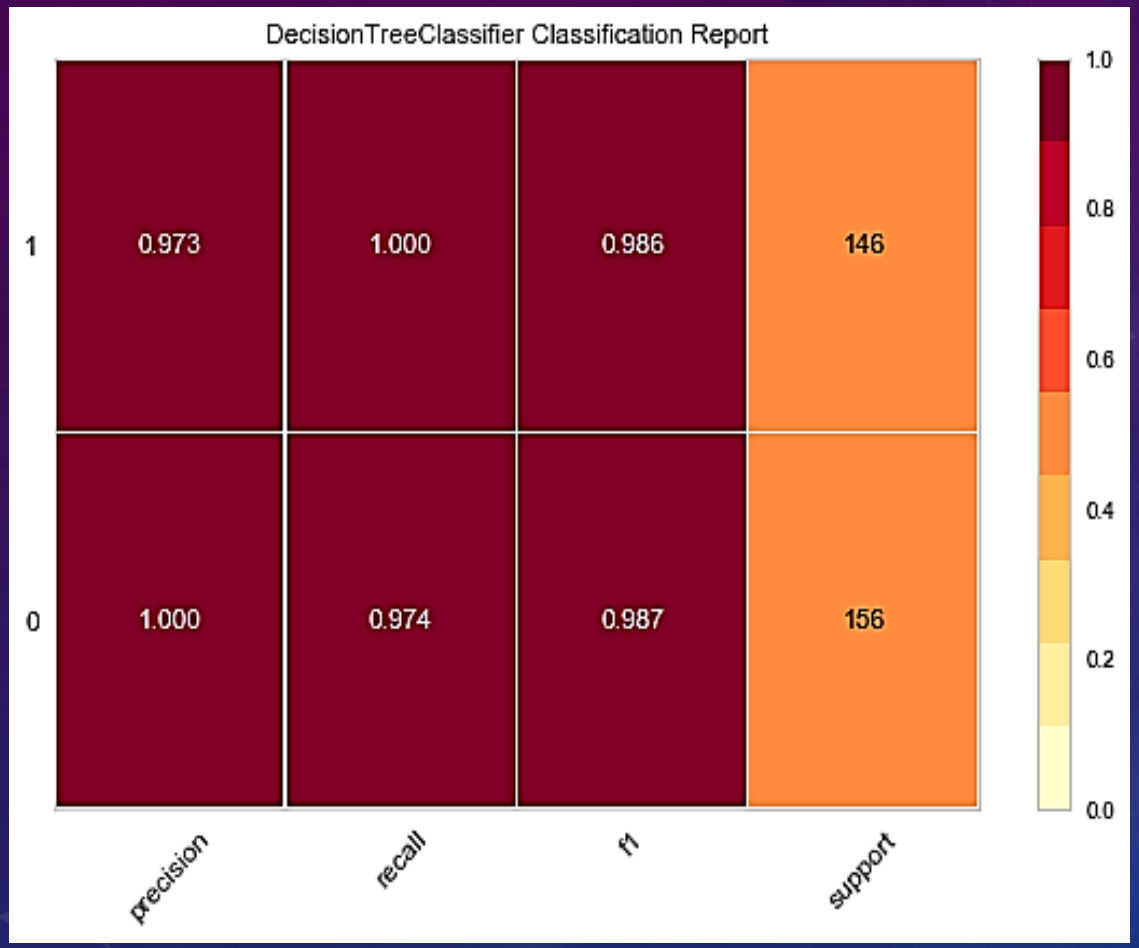
After Applying The SMOTE

The AUC = 94.2%

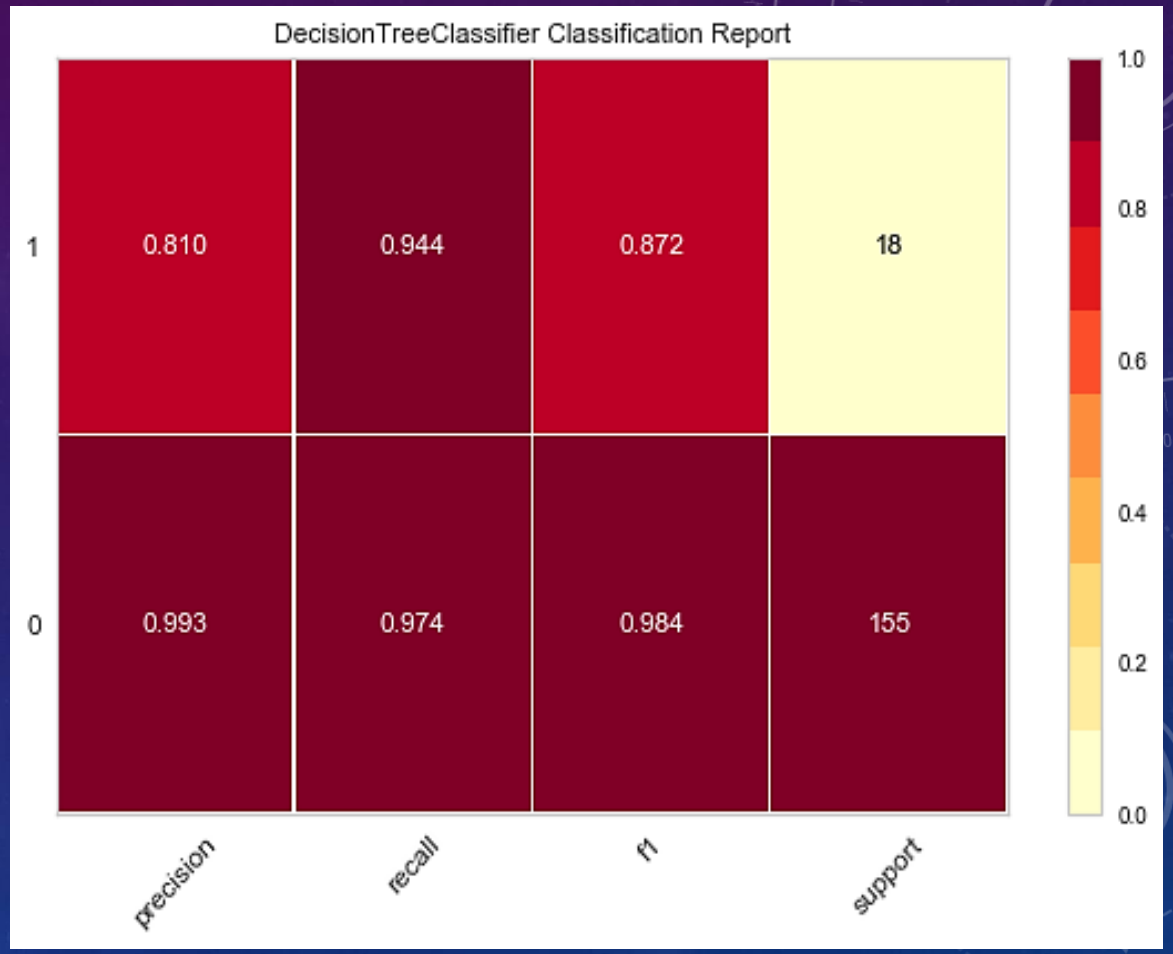


Before Applying The SMOTE (The Best)

4. DECISION TREE (WITH SMOTE)

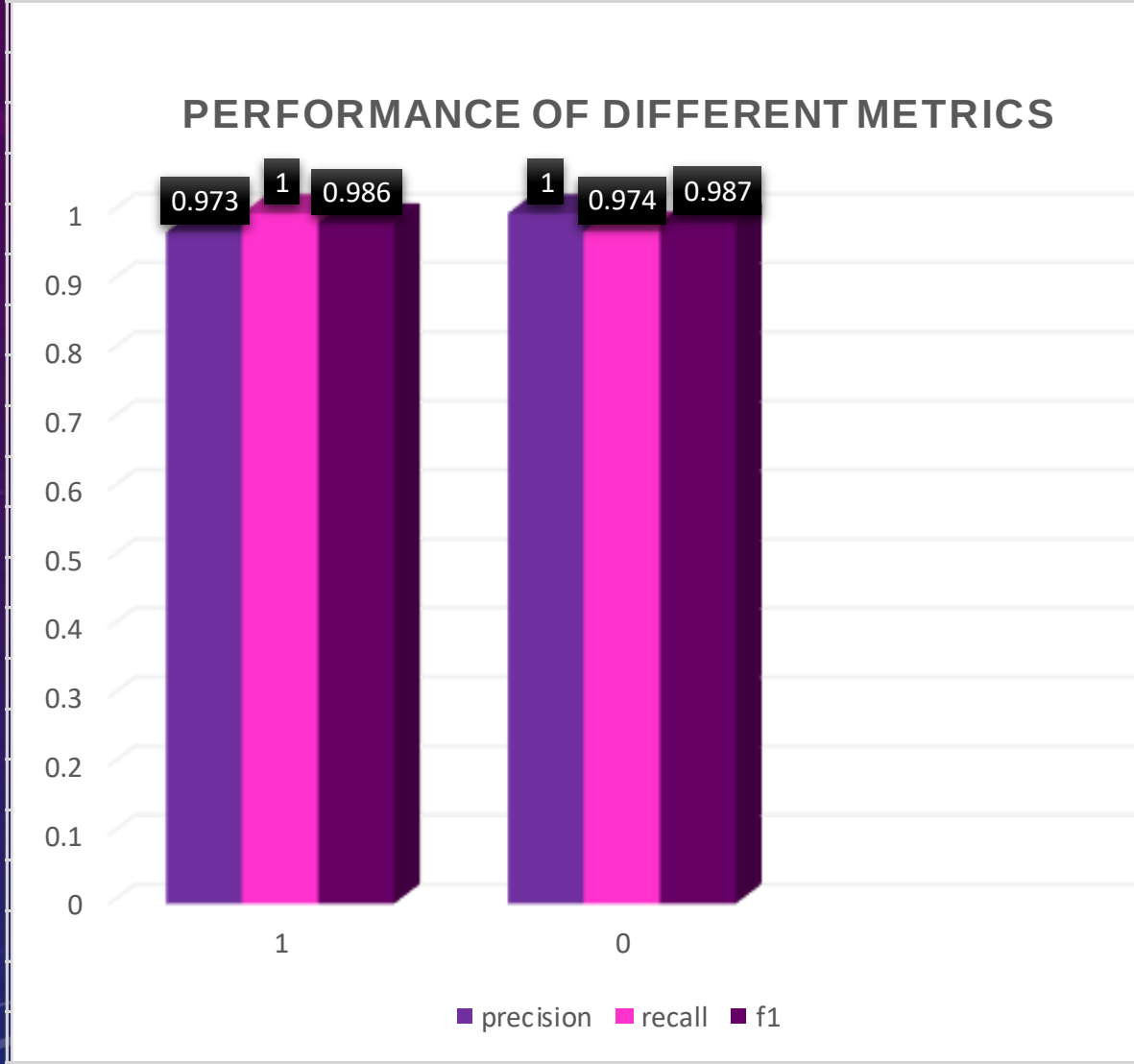


After Applying The SMOTE(The Best)

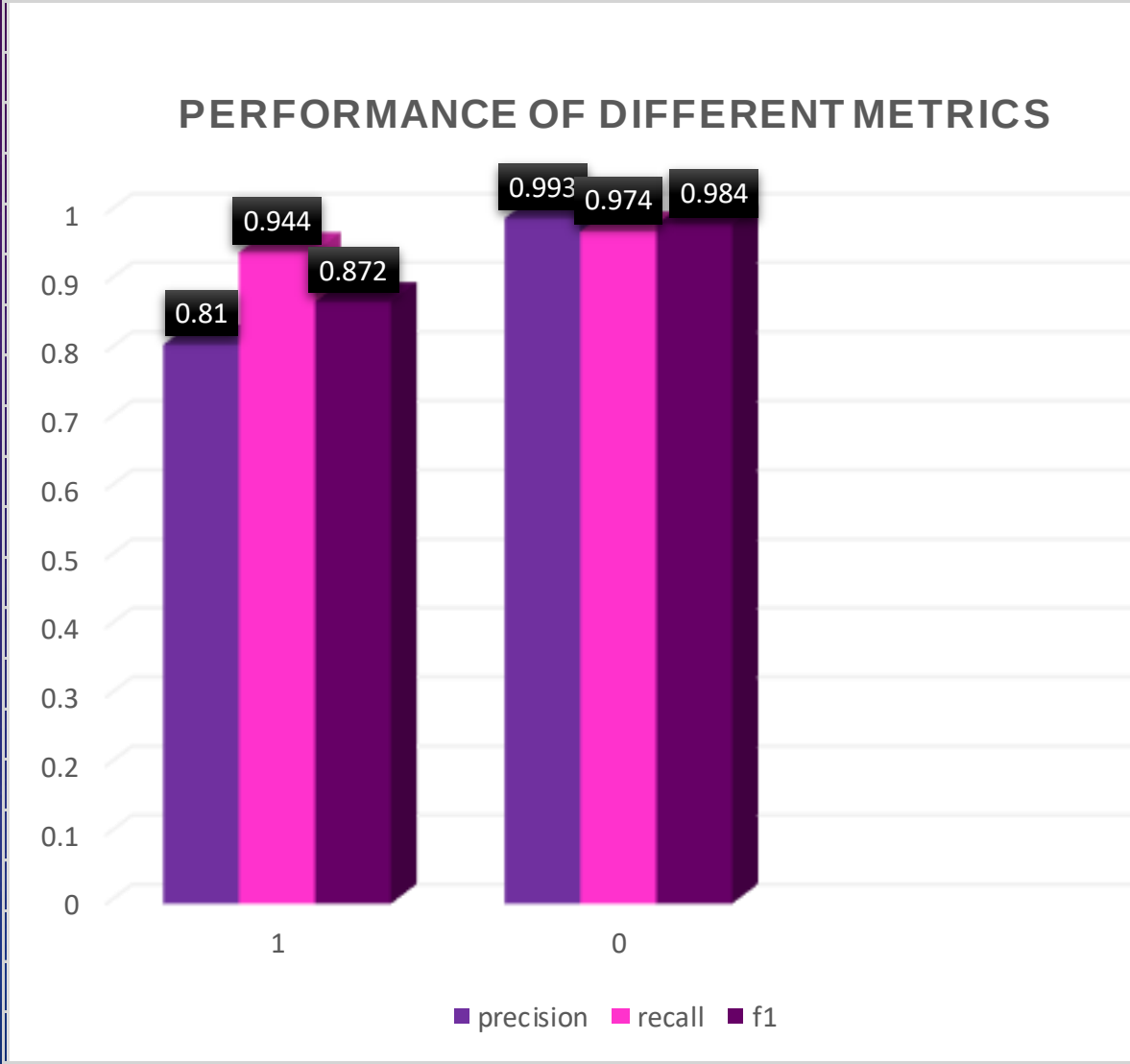


Before Applying The SMOTE

4. DECISION TREE (WITH SMOTE)



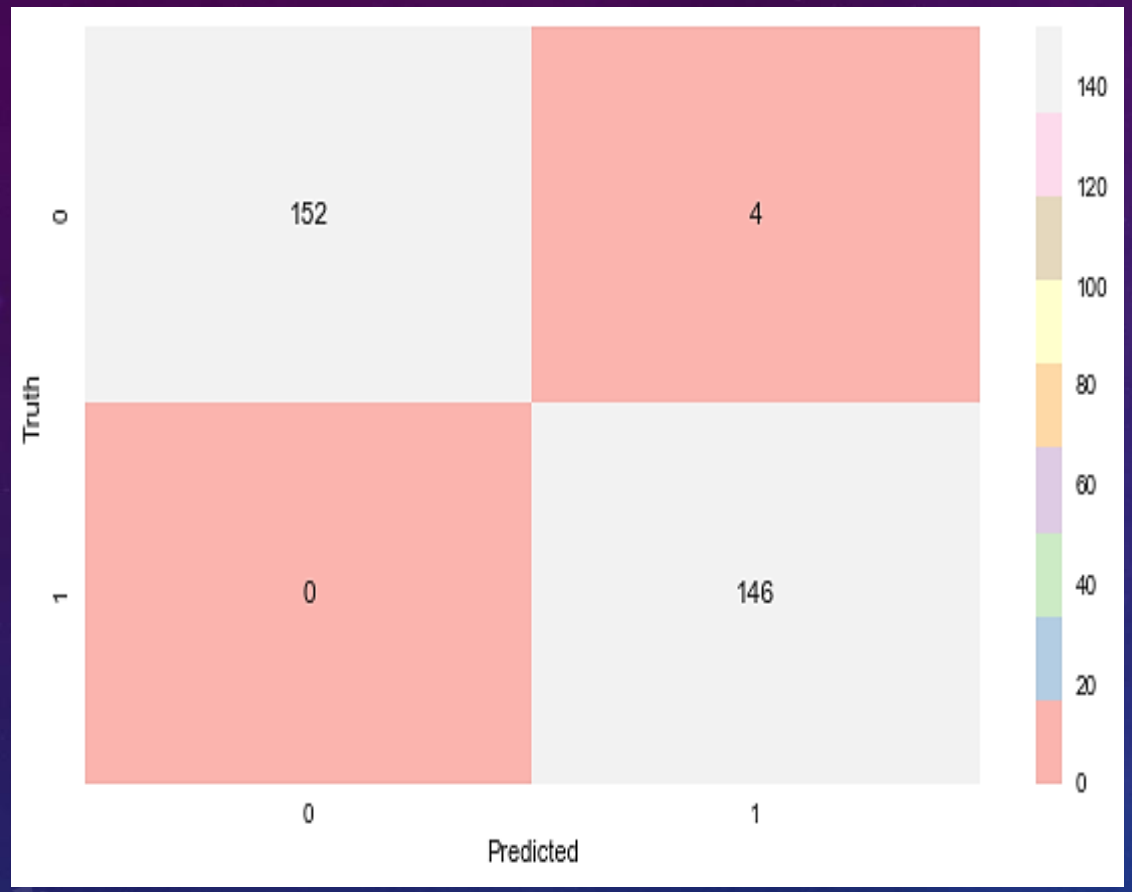
After Applying The SMOTE(The Best)



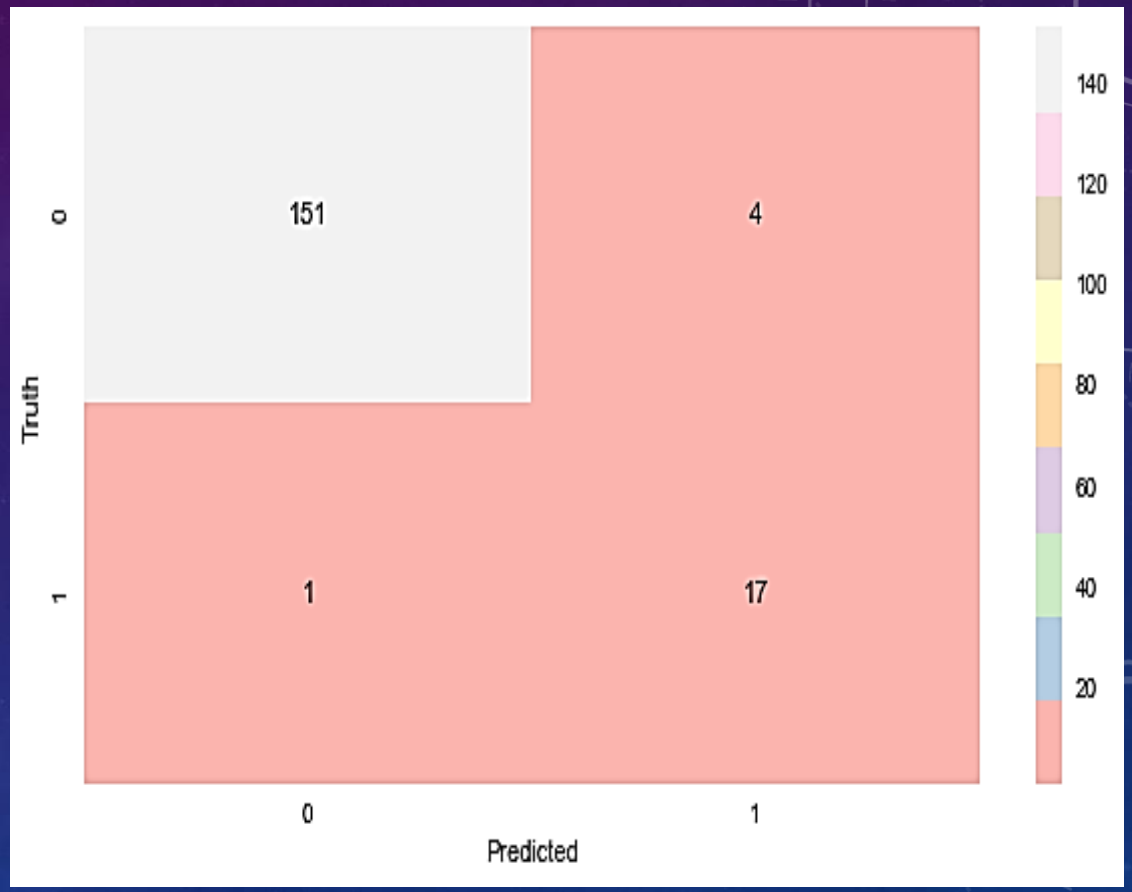
Before Applying The SMOTE

Performance	DT(With SMOTE)	DT(Without SMOTE)
Precision	0 => 100 % / 1 => 97.3%	0 => 99.3 % / 1 => 81 %
Recall	0 = > 97.4 % / 1 => 100 %	0 = > 97.4% / 1 => 94.4 %
F1-score	0 => 98.7% / 1 => 98.6%	0 => 98.4 % / 1 => 87.2%
ROC/AUC	98.98%	96.24%

4. DECISION TREE (WITH SMOTE)



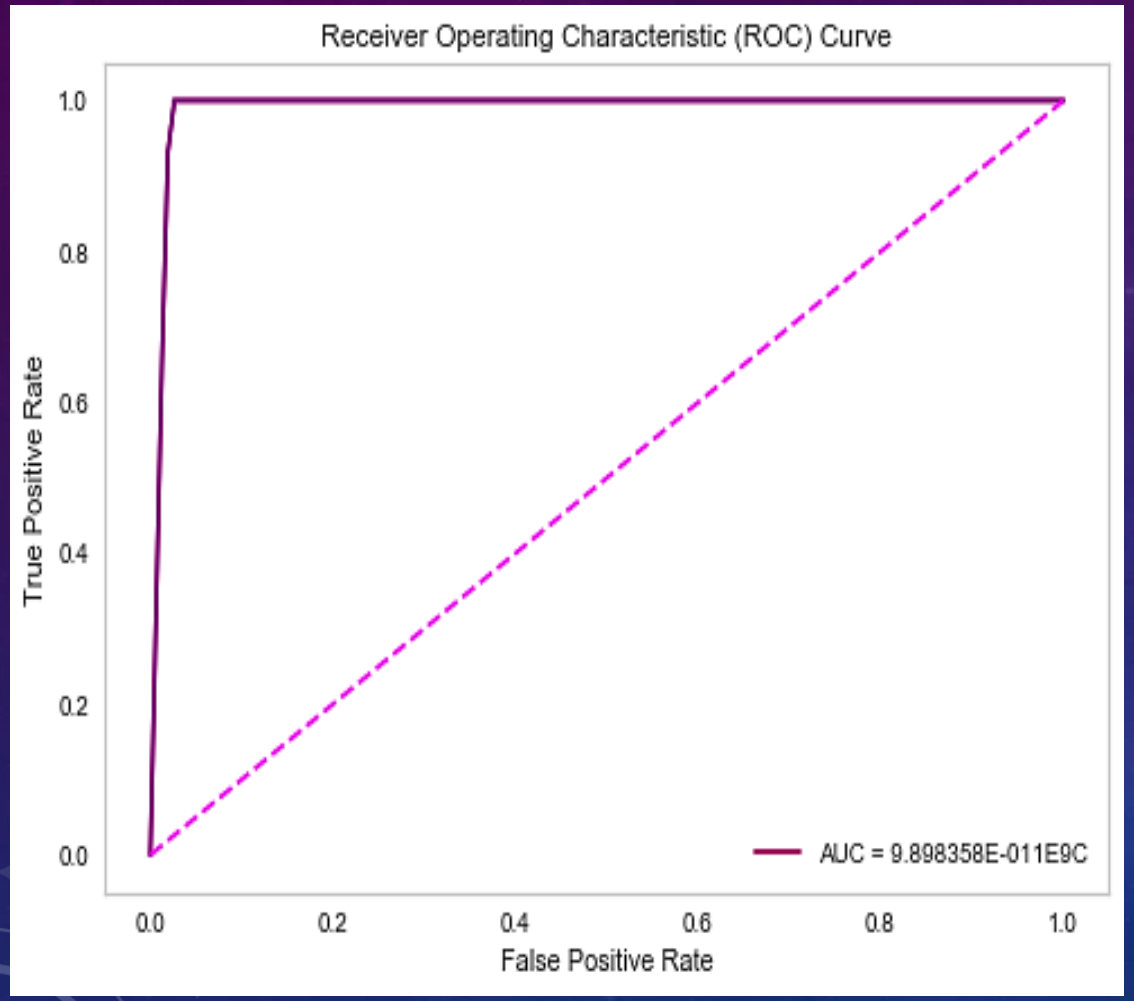
After Applying The SMOTE(The Best)



Before Applying The SMOTE

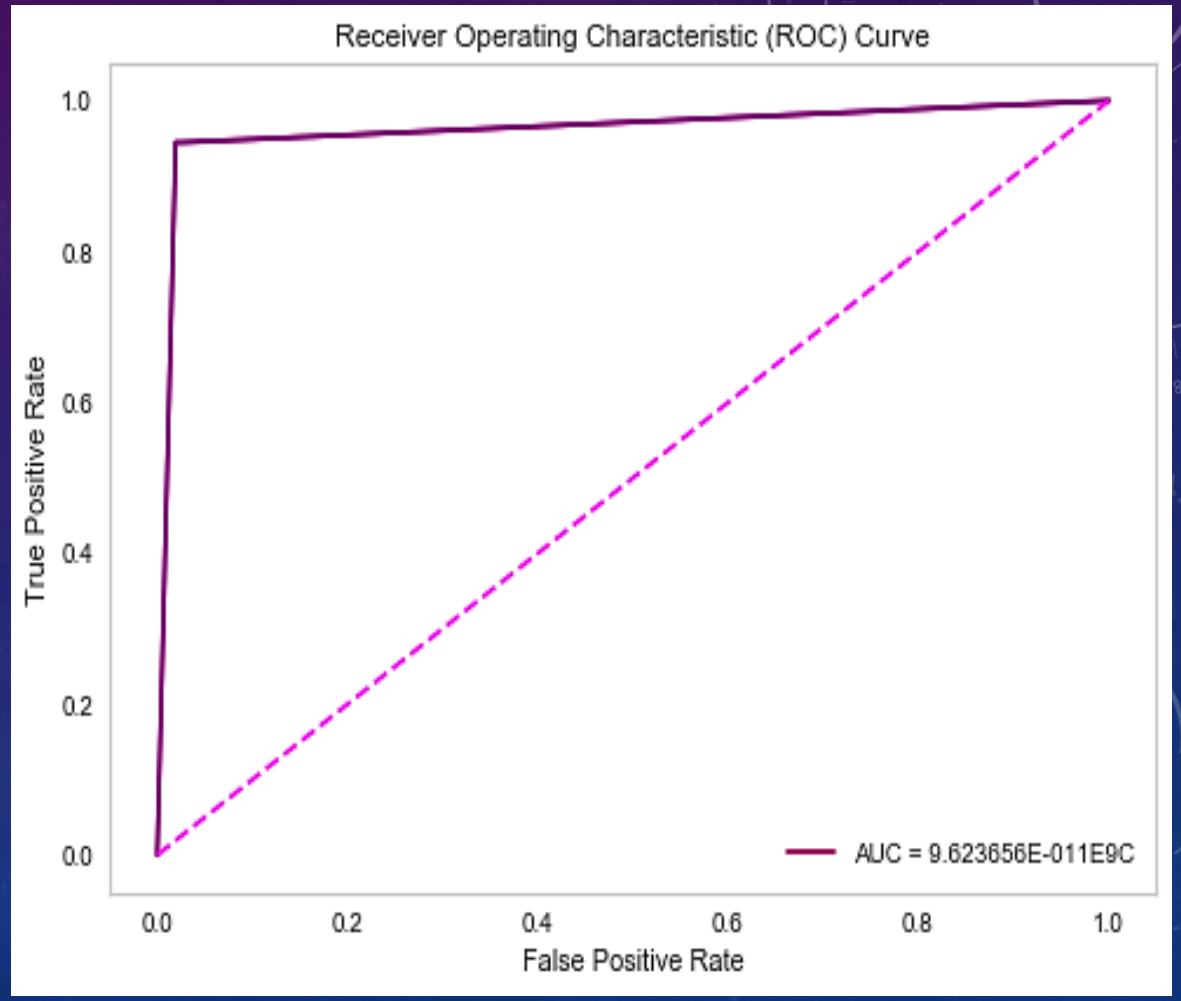
4. DECISION TREE (WITH SMOTE)

The AUC = 98.98%



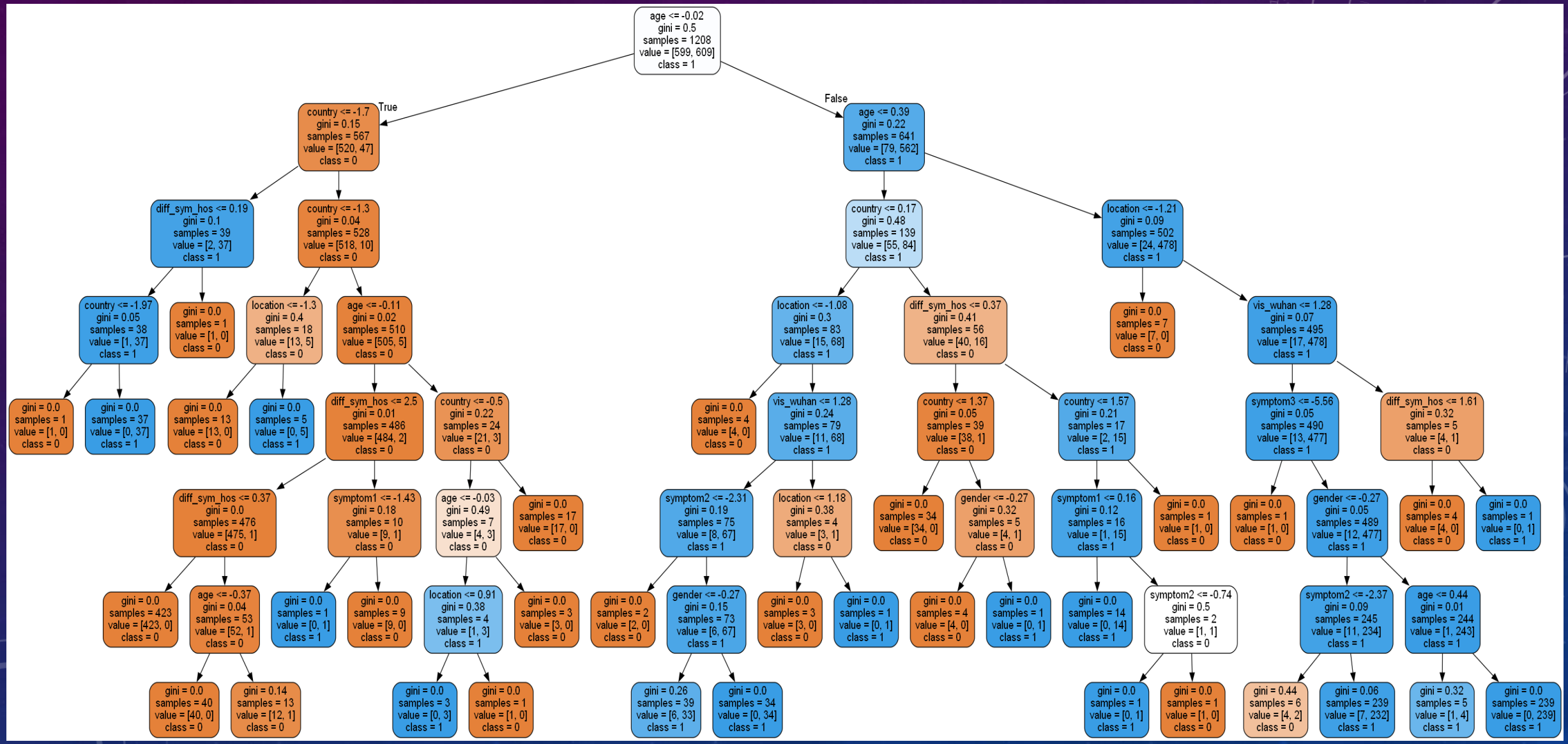
After Applying The SMOTE (The Best)

The AUC = 96.24%

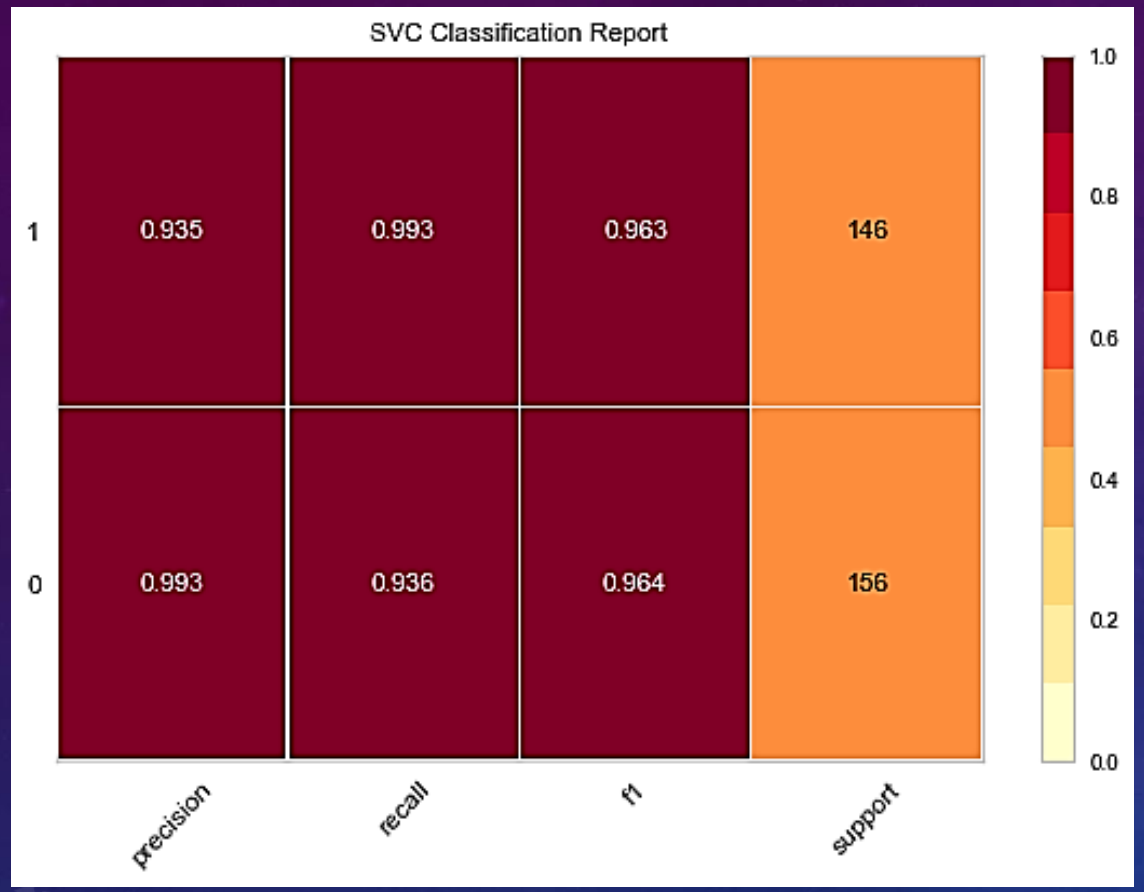


Before Applying The SMOTE

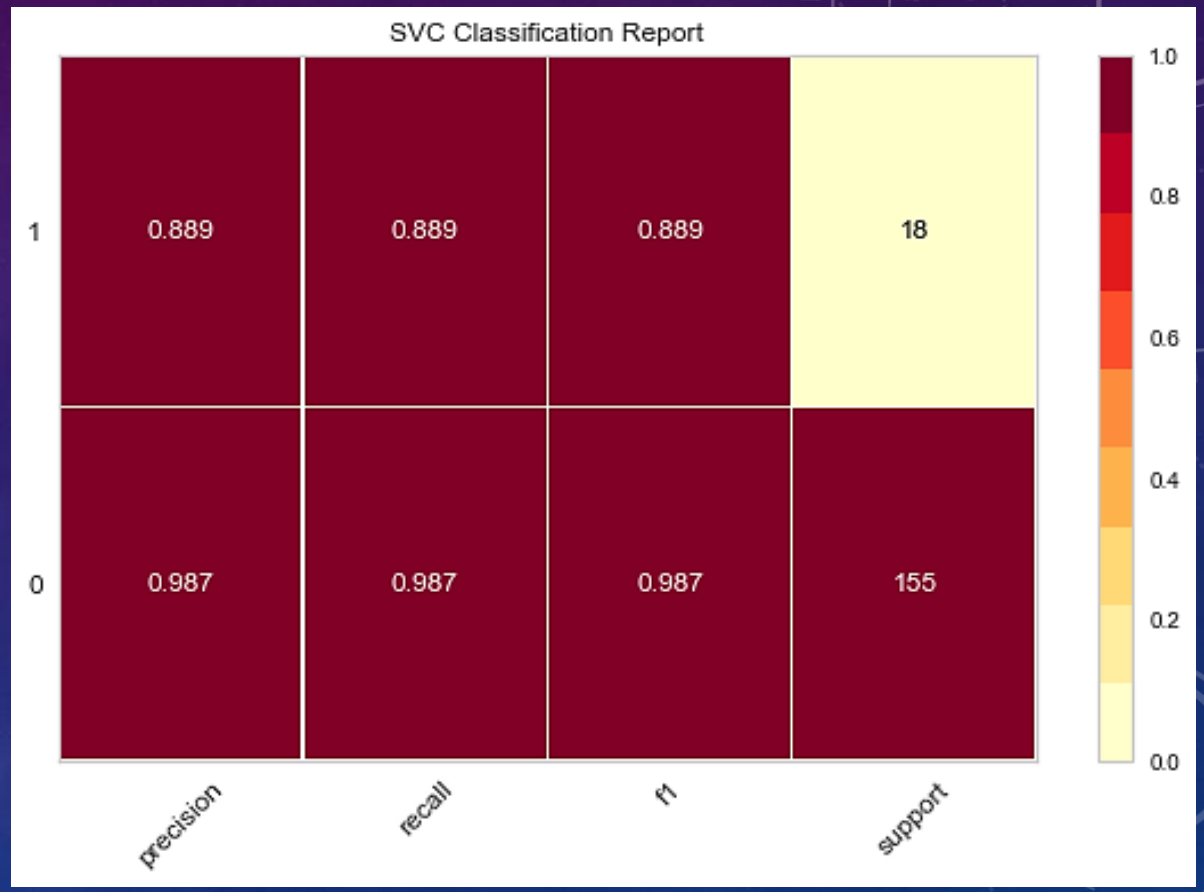
THE VISUALIZATION FOR THE TREE(WITH SMOTE)



5. SUPPORT VECTOR MACHINE(SVM) (WITH SMOTE)

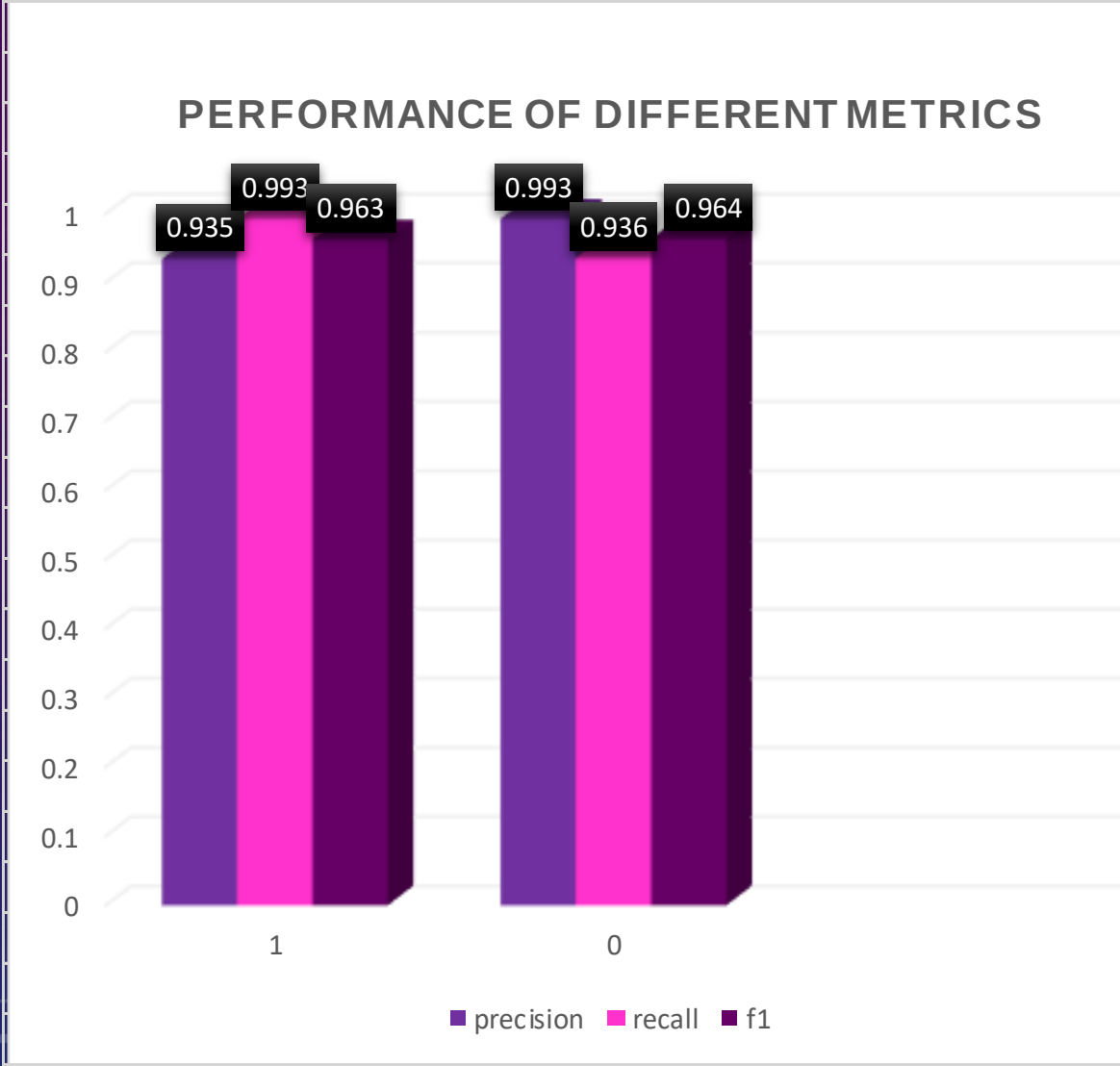


After Applying The SMOTE(The Best)

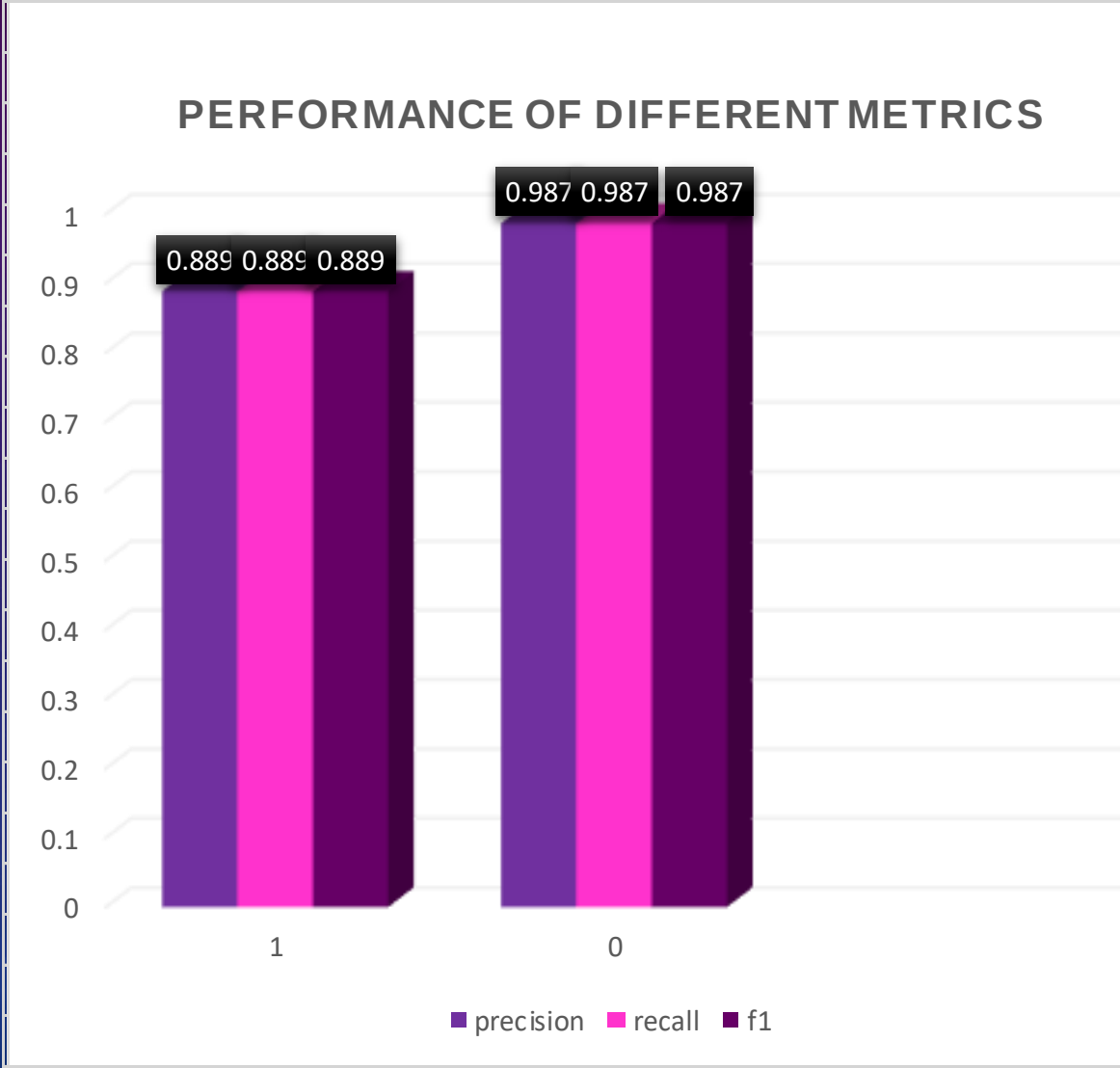


Before Applying The SMOTE

5. SUPPORT VECTOR MACHINE(SVM) (WITH SMOTE)



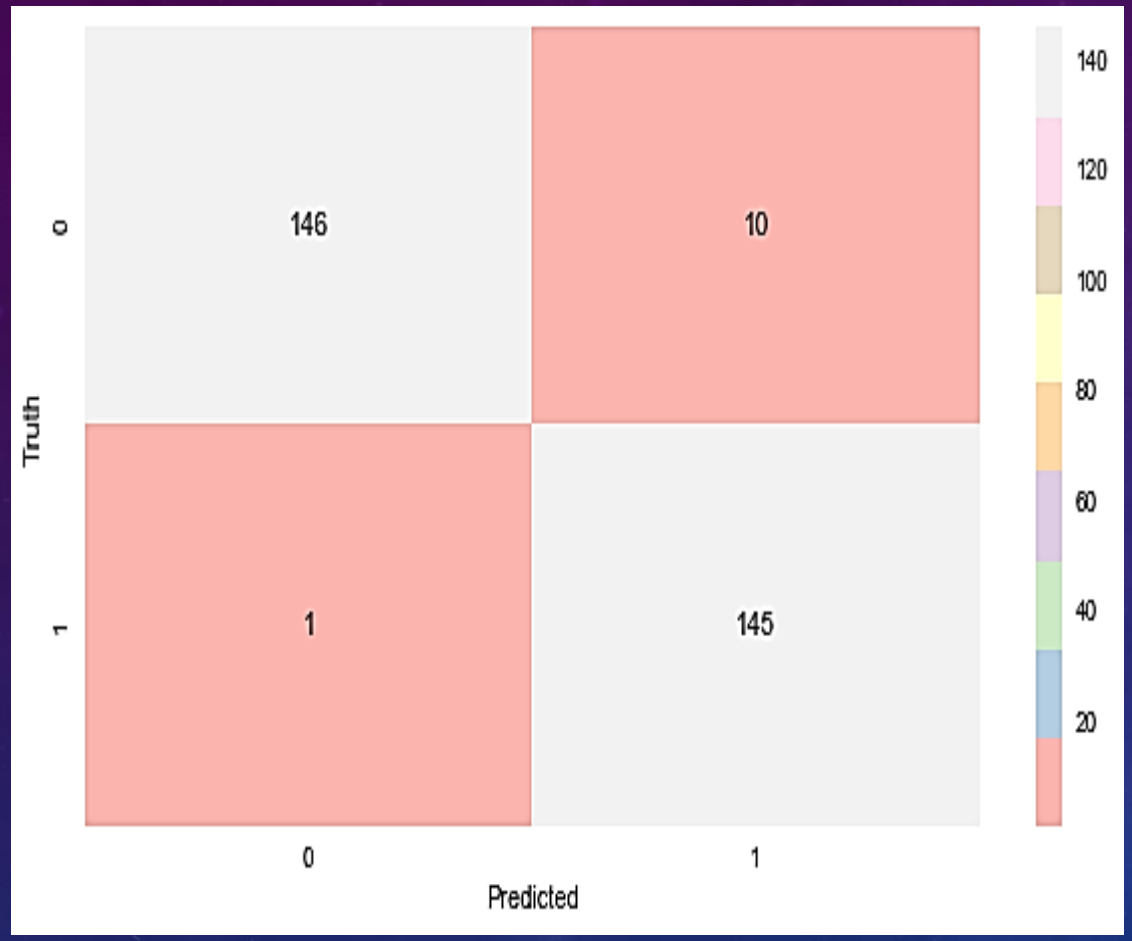
After Applying The SMOTE(The Best)



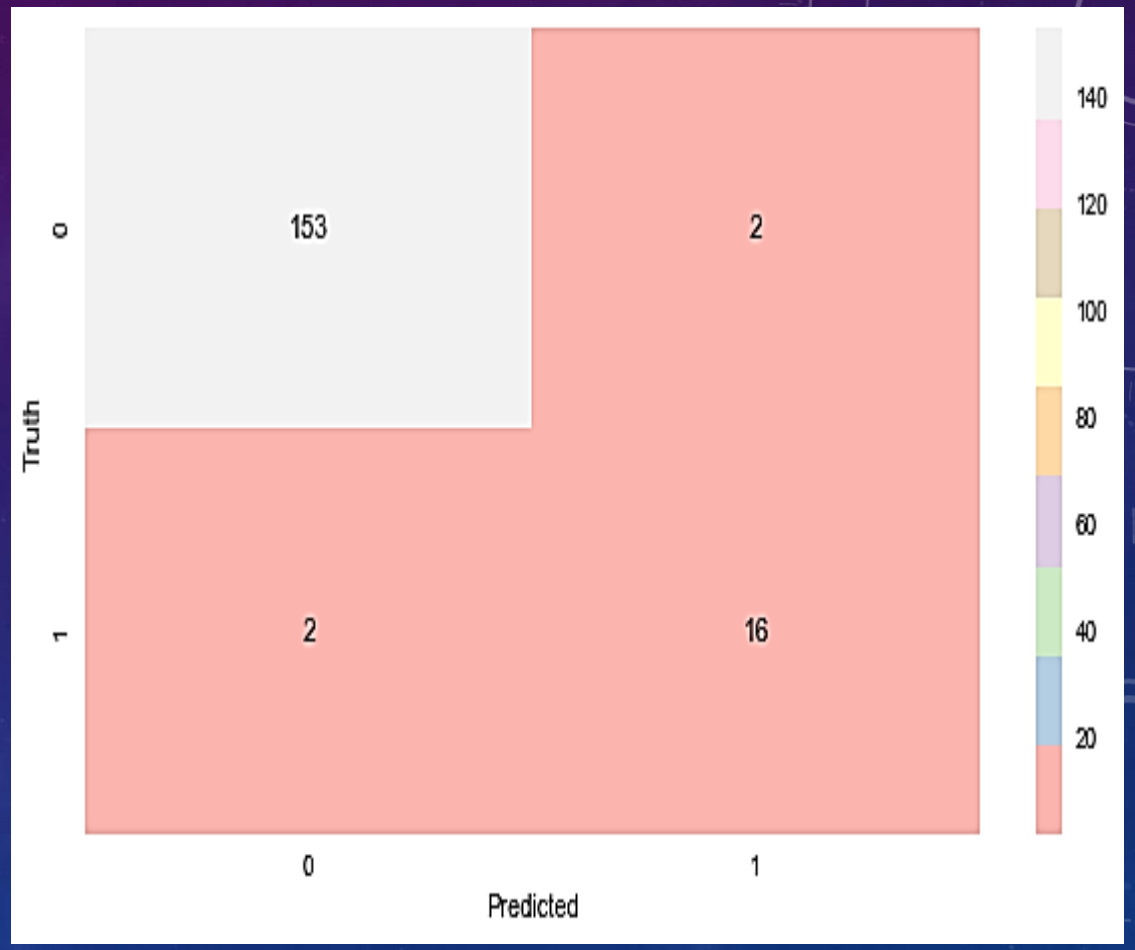
Before Applying The SMOTE

Performance	SVM(With SMOTE)	SVM(Without SMOTE)
Precision	0 => 99.3 % / 1 => 93.5%	0 => 98.7 % / 1 => 88.9 %
Recall	0 = > 99.3 % / 1 => 93.6 %	0 = > 98.7% / 1 => 88.9 %
F1-score	0 => 96.4% / 1 => 96.3%	0 => 98.7 % / 1 => 88.9%
ROC/AUC	99.47%	99.5%

5. SUPPORT VECTOR MACHINE(SVM) (WITH SMOTE)



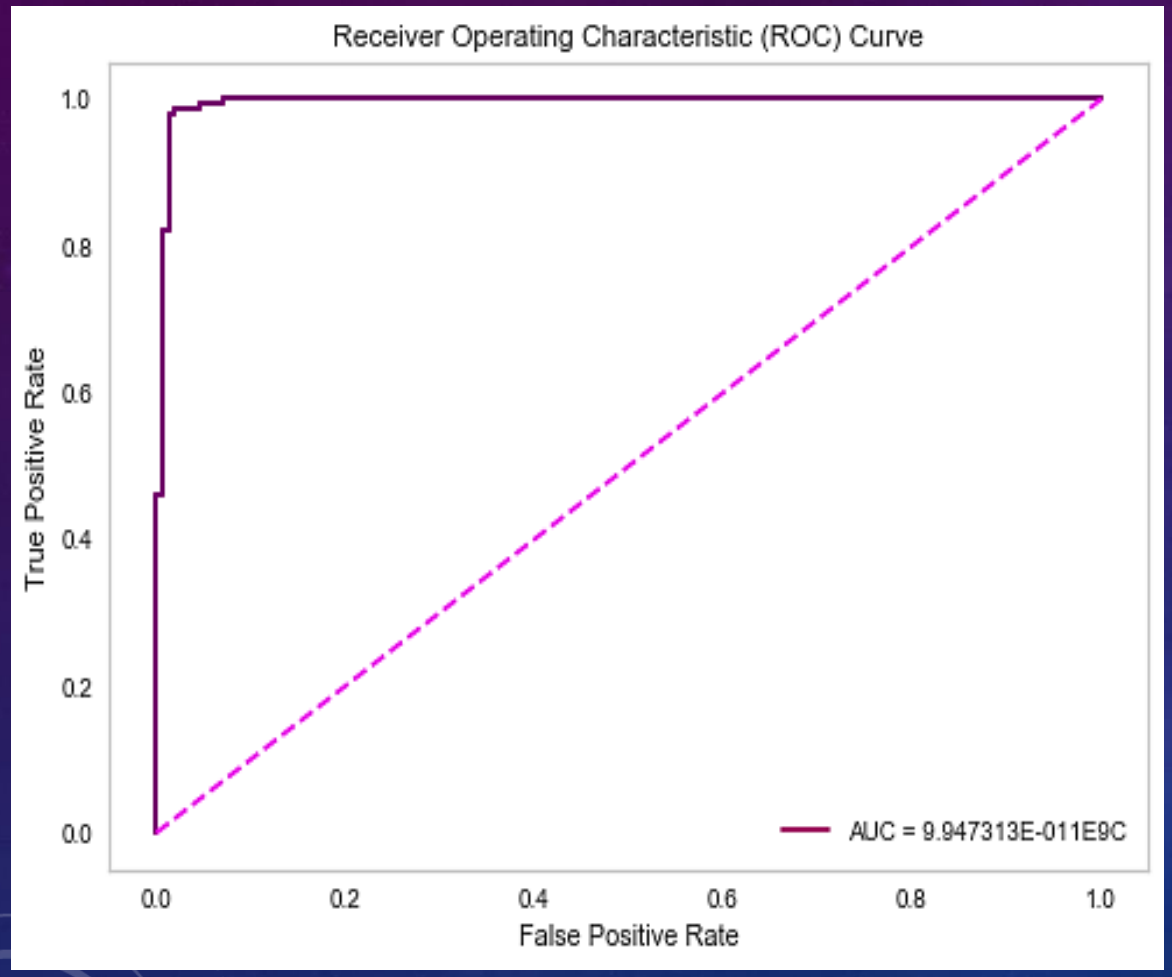
After Applying The SMOTE(The Best)



Before Applying The SMOTE

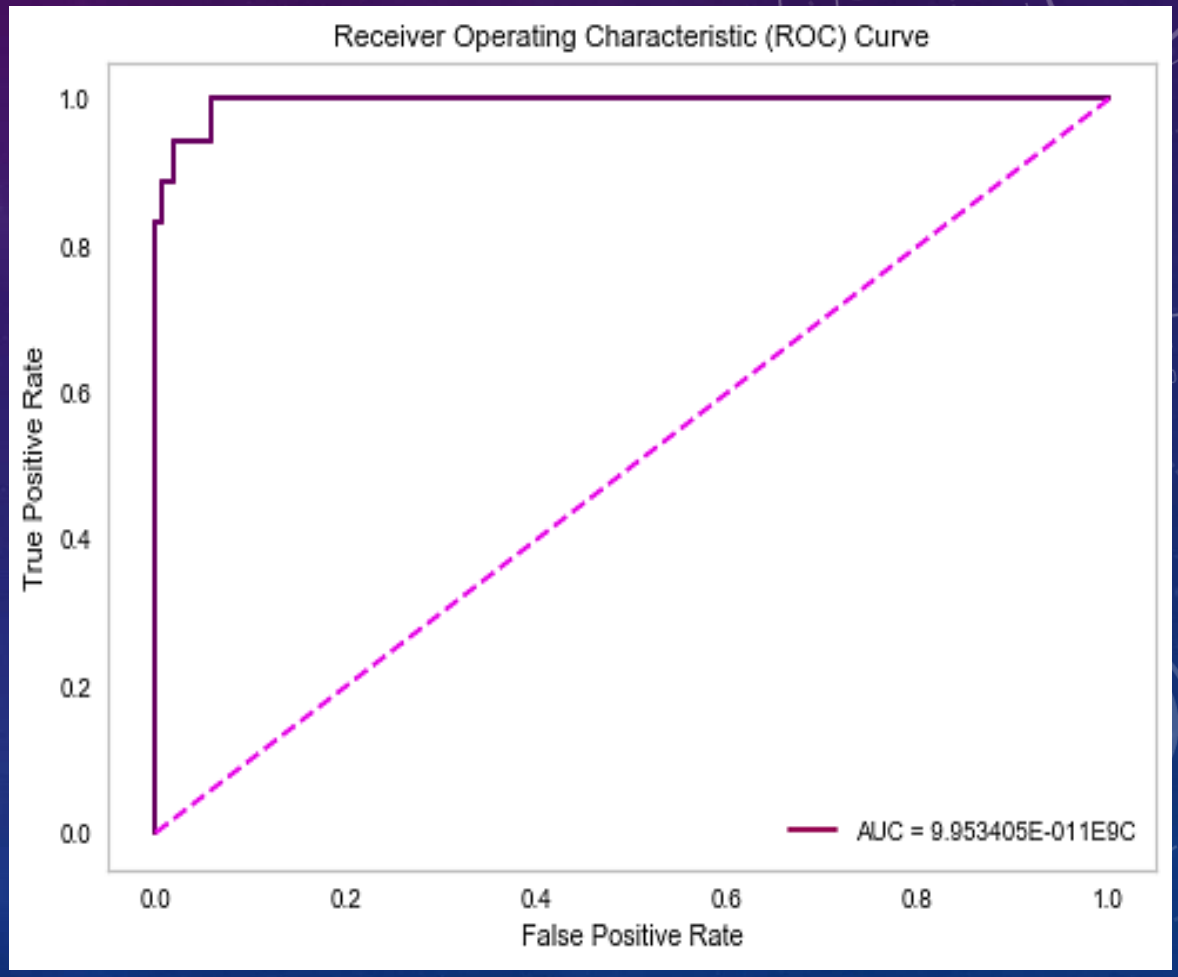
5. SUPPORT VECTOR MACHINE(SVM) (WITH SMOTE)

The AUC = 99.47%



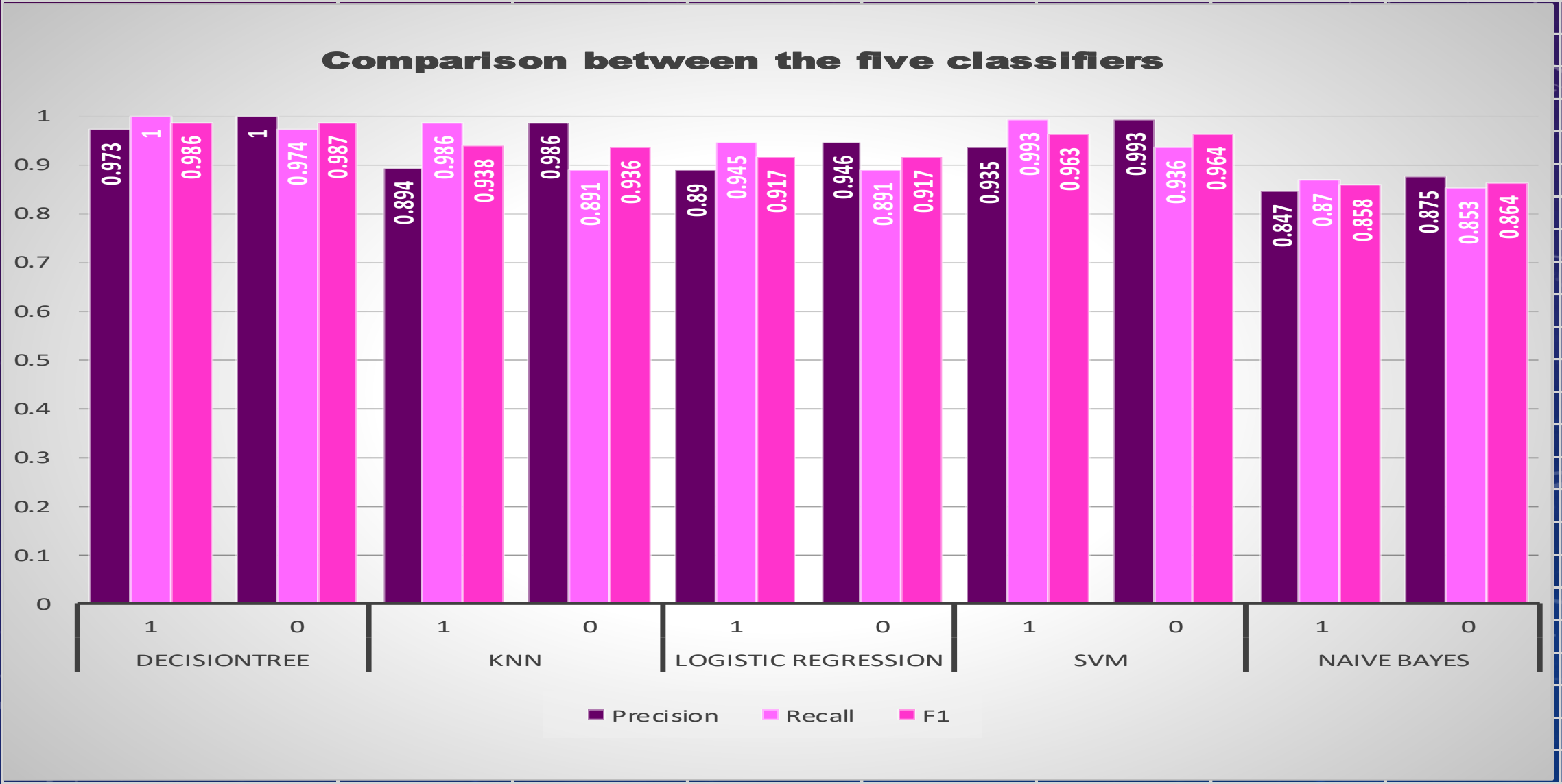
After Applying The SMOTE

The AUC = 99.5%



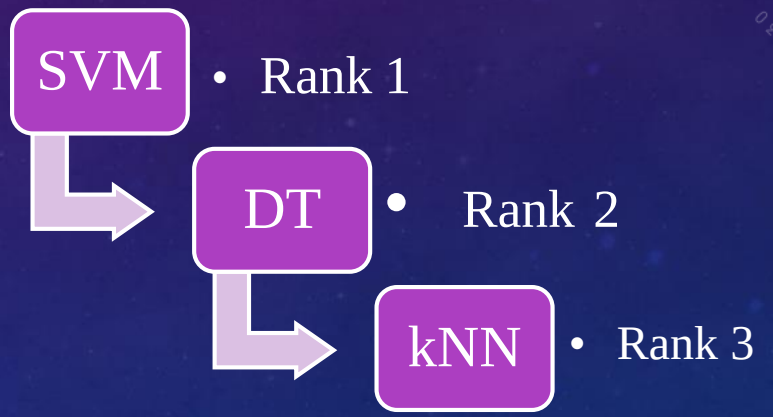
Before Applying The SMOTE

COMPARISON BETWEEN THE FIVE CLASSIFIERS(WITH SMOTE)



THE BEST MODEL(WITH SMOTE)

- The good news is that all models are above 92% accuracy.
- From visualization, we found that the SVM model **has** the highest one. Then Decision Tree with respect to
  - **Precision**
  - **Recall**
  - **F1: Score**
  - **Area Under ROC Curve** (AUROC)



- Compared the performance of the models before and after applying SMOTE, The models have a big enhancement and high improvement specially in logistic regression.

RESOURCES

- Sklearn Documentation
- Stackoverflow
- scikit-learn package
- Classification Report from yellowbrick.classifier