Early Stroke Risk Prediction Using Machine Learning Models

Name:Shah Sanzida Masiat
ID: 20-42937-1
Dept Name:CSE
Institute Name:AIUB
Dhaka,Bangladesh
email :masiat.shah.42937@gmail.com
Phone:01794835232

Name:S.M.Kamrul Hasan Koche
ID: 20-43405-1
Dept Name:CSE
Institute Name:AIUB
Dhaka,Bangladesh
email :kamrulkoche@gmail.com
Phone :01798135862

Student Name: MD. Afif All Reza
ID: 20-43409-1
Dept Name:CSE
Institute Name:AIUB
Dhaka, Bangladesh,
email address:
afifalreza30497@gmail.com
Phone Number: 01701758757

Name: MD TANBIN TUSHAR
ID: 20-43573-1
Dept Name:CSE
Institute Name:AIUB
Dhaka,Bangladesh
email : tanbintushar433@gmail.com
Phone Number: 01762290516

Abstract

The "Stroke Prediction Healthcare Dataset" is a useful and thorough set of medical information created to make precise predictions about the likelihood of having a stroke based on a variety of demographic, clinical, and lifestyle characteristics. Effective prediction tools are required for stroke, a significant cause of morbidity and death globally, to enable early intervention and individualized care, thereby lowering the burden of this crippling illness on people and healthcare systems.

The patient demographics, risk factors (such as hypertension, heart disease, diabetes, smoking, and alcohol use), clinical measurements (such as blood pressure, glucose levels, and BMI), stroke specifics (ischemic or hemorrhagic), treatment information, lifestyle factors (physical activity, diet), and follow-up information (recovery status, long-term effects) are all included in this dataset. Together, these elements create a rich setting for in-depth investigation of the factors influencing stroke development.

This dataset's main goal is to enable academics, healthcare providers, and data analysts to create reliable models for stroke incidence prediction. These models seek to identify people who are more likely to have a stroke by revealing the complex connections between risk factors, clinical signs, and lifestyle variables. The dataset gives insights into the efficiency of various treatment approaches, the effect of lifestyle changes on stroke prevention and recovery, and the identification of major risk factors linked to stroke in addition to serving as a basis for stroke prediction.

This dataset can promote stroke research, public health initiatives, policy formulation, and patient-centered treatment as well as its prediction powers. It acts as a helpful tool for internal team projects.
The "Stroke Prediction Healthcare Dataset" must be used with strict data protection and ethical standards in mind. To safeguard the identity of the people represented in the dataset, appropriate anonymization and security procedures should be upheld. This will guarantee that the data is handled responsibly and in compliance with established norms.

In summary, the "Stroke Prediction Healthcare Dataset" provides a substantial advancement in the management and prevention of stroke. It is an important tool in the fight to detect and prevent strokes, enhance patient outcomes, and advance the area of stroke-related healthcare due to its comprehensive nature and the potential for effective research and therapies.

I. INTRODUCTION

The dataset called "Stroke Healthcare Dataset" is a collection of information that focuses on stroke cases, patient characteristics, risk factors and different health indicators. Stroke is a condition that happens when there is an interruption or reduction in blood flow to a specific area of the brain resulting in damage to brain cells. It is one of the leading causes of disability and death. It is vital to understand the factors contributing to stroke incidence and the outcomes for individuals to enhance prevention strategies, treatment protocols and overall healthcare.

It is essential to highlight how important it is to utilize this dataset responsibly. Data security and patient privacy must be given careful consideration. Working with this sensitive healthcare data requires careful attention to anonymization procedures and ethical standards.

This dataset's main goal is to make it possible to create reliable predictive models that take advantage of the abundance of data it contains. Researchers and healthcare practitioners can develop sophisticated models that can identify people at increased risk, frequently before the development of overt symptoms, by studying the correlations between the various traits and the likelihood of strokes.

This dataset holds value for researchers’ healthcare professionals and data analysts who want to explore the factors associated with strokes. It can help identify high risk groups and develop interventions. The dataset contains variables that can be used to investigate the relationships between factors and stroke occurrence. Additionally, it provides insights into the impact of treatments and lifestyle changes.

II. MOTIVATION OF THE PROJECT

In the realm of healthcare every set of data has the potential to inspire transformation. There are certain datasets that hold exceptional significance. The "Stroke Healthcare Dataset" is more than a compilation of numbers and variables; it serves as a guide for comprehending and combating one of the impactful medical challenges we face today – stroke. Imagine a world where we possess the ability to accurately predict, prevent and provide personalized care for stroke. Picture a future where strokes, those life altering occurrences are not just treated but significantly reduced in frequency.

Such progress would save lives. Alleviate the burden of disability on individuals, families, and communities alike. This dataset offers hope rather than just data. The aim is that we will be able to sort out the complex web of risk factors, lifestyle variables, and clinical indicators that affect strokes. In-depth analysis of this data is hoped to reveal any hidden patterns, early warning indicators, or effective treatment approaches that might alter the course of this severe illness.

III. OBJECTIVE OF THE PROJECT

These objectives highlight the Stroke Healthcare Dataset's transformational potential, motivating academics, medical professionals, politicians, and activists to work together to bring about significant changes in stroke prevention, treatment, and patient well-being.

Based on input characteristics including gender, age, numerous illnesses, and smoking status, this dataset is used to determine whether a patient is likely to get a stroke. Each row of the data contains pertinent patient information.

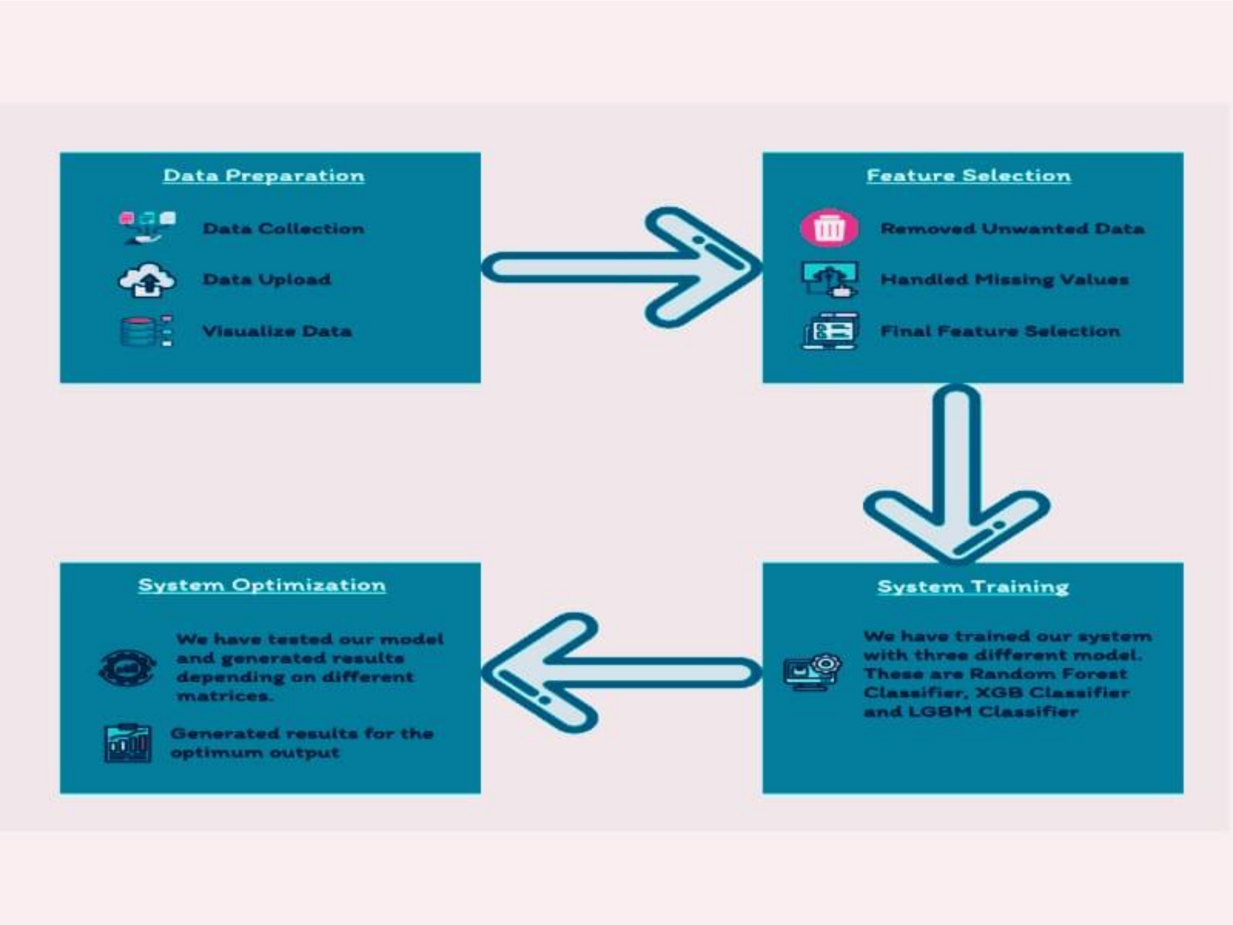
To acquire a thorough grasp of their influence on stroke development, look at the well-known risk factors for stroke, such as hypertension, heart disease, diabetes, smoking, and alcohol use. To improve treatment protocols and find successful interventions that are suited to specific patient profiles, analyze the information to find trends in stroke kinds, severity, and treatment results.

We can use the dataset to offer data-supported insights for public health initiatives that aim to increase understanding of stroke risk factors, preventative tactics, and the value of early medical intervention. We can Share the dataset's anonymized and compiled findings with the scientific community to promote cooperation in the advancement of stroke research, which will ultimately result in ground-breaking discoveries and advancements in stroke prevention and treatment.

we will be able to Empower healthcare practitioners with data-driven insights to provide individualized care plans for stroke patients, taking into account individual risk profiles, treatment histories, and recovery trajectories.

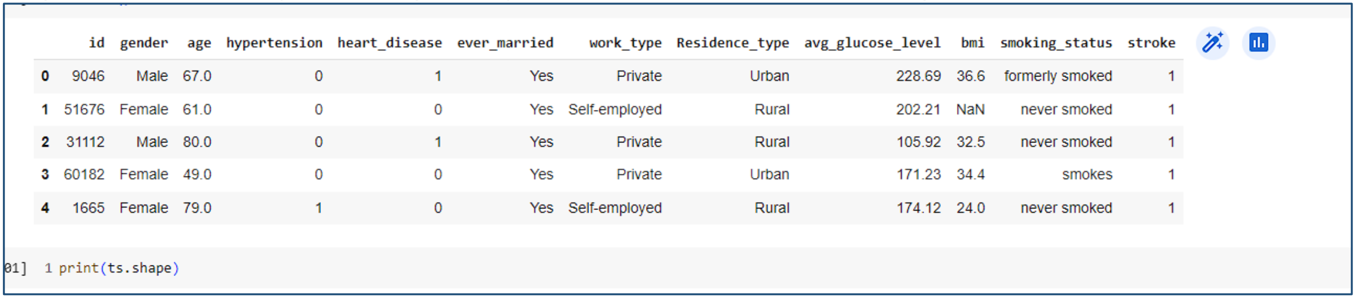
IV. Methodology

The dataset was originally collected through the Kaggle website. After that, we upload it to Google Drive. The Google Drive and Google Colab are then connected. Next, we illustrate several dataset insights. After that, we carried out feature engineering, dealing missing data, handling categorical features, handling feature scaling, and eliminating outliers. Following feature importance calculation, the model is built. Following that, we tested our model and produced results based on several matrices.


Data Preparation
Data Collection
Data Upload
Visualize Data
Feature Selection
Removed Unwanted Data
Handled Missing Values
Final Feature Selection
System Training
We have trained our system with three different model. These are Random Forest Classifier, XGB Classifier and LGBM Classifier
System Optimization
We have tested our model and generated results depending on different matrices.
Generated results for the optimum output

A. Data Collection

The dataset was originally collected through the Kaggle website. The dataset may be used to assess the effectiveness of various stroke treatment plans. Researchers can determine which therapies result in greater recovery and fewer long-term side effects by examining therapy specifics and results, so enhancing patient care.



	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

```
01]  1 print(ts.shape)
```

B. Data processing

Data Loading: Use pd.read_csv() from Pandas to load the dataset. Exploratory Data Analysis (EDA): Utilize Pandas, Seaborn, and Matplotlib to understand the data's structure, attributes, and patterns. Data Preprocessing: Prepare data for analysis and modeling. Handle missing values using Pandas functions. Encode categorical variables with one-hot or label encoding. Scale numerical features for consistent scales. Split data into training and testing subsets using train_test_split. Address outliers that impact model performance.

C. Dataset description

The "Stroke Healthcare Dataset" contains a rich set of attributes that provide insights into stroke occurrences and related factors. Some key(12)features included in the dataset are:
Attribute Information

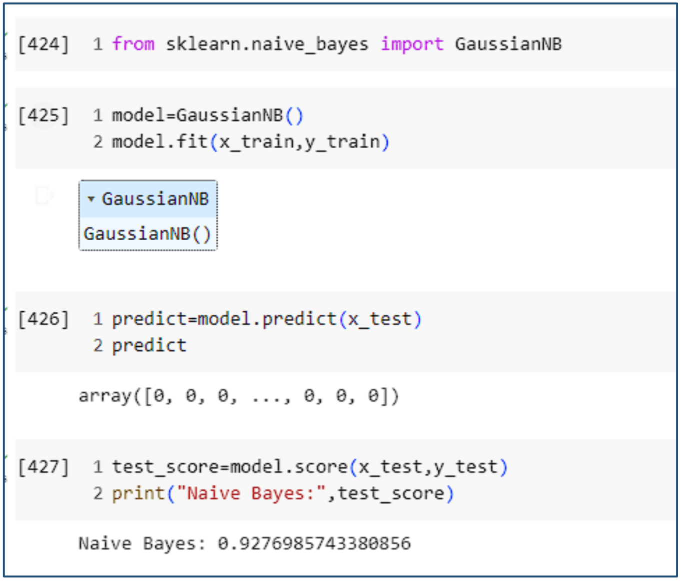
1) id: unique identifier
2) gender: "Male", "Female" or "Other"
3) age: age of the patient
4) hypertension: 0 if the patient does not have hypertension, 1 if the patient has hypertension
5) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
6) ever_married: "No" or "Yes"
7) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
8) Residence_type: "Rural" or "Urban"
9) avg_glucose_level: average glucose level in blood
10) bmi: body mass index
11) smoking status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
12) stroke: 1 if the patient had a stroke or 0 if not

D. Machine Learning model development and evaluation

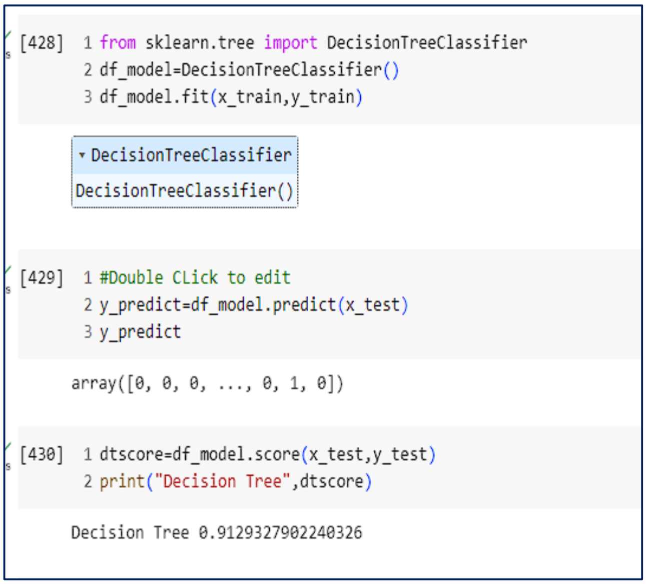
Following preprocessing, we create machine learning models applying a variety of techniques, and we evaluate their effectiveness using metrics like accuracy, precision, recall, and F1-score.
Basically, the code creates the environment, and then the data loading, exploratory analysis, preprocessing, modeling, and assessment stages come next.

V. Results

We have used five algorithms. They are GaussianNB, DecisionTreeClassifier, RandomForestClassifier, LogisticRegression, and KNeighbourClassifier . Among them, RandomForestClassifier performs well. So, we finally chose RandomForestClassifier to train and test our model. We have used different measurement matrices to evaluate our model. They are described below:

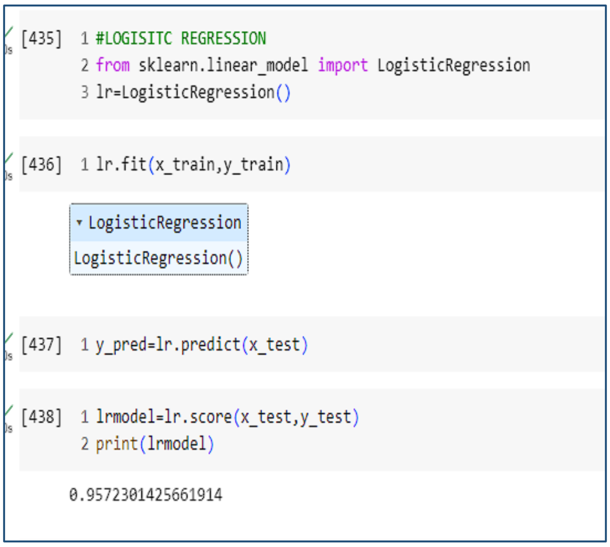


```
[424]  from sklearn.naive_bayes import GaussianNB

[425]  model=GaussianNB()
       model.fit(x_train,y_train)

       ▾ GaussianNB
       GaussianNB()

[426]  predict=model.predict(x_test)
       predict

       array([0, 0, 0, ..., 0, 0, 0])

[427]  test_score=model.score(x_test,y_test)
       print("Naive Bayes:",test_score)

       Naive Bayes: 0.9276985743380856
```

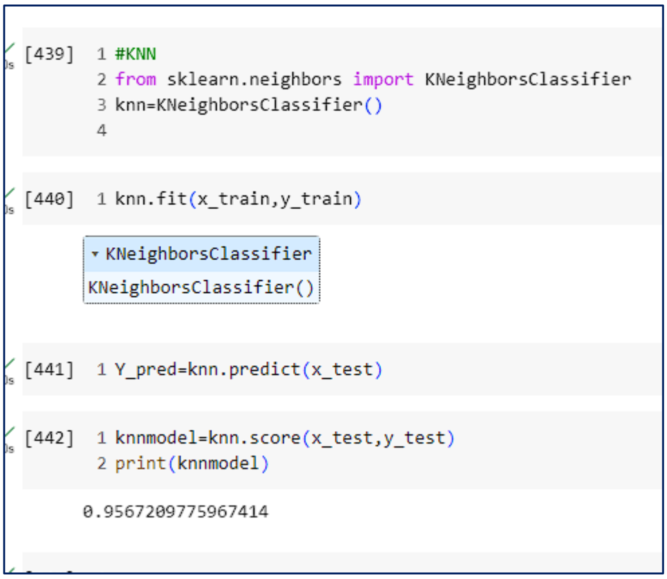


```
[428]  from sklearn.tree import DecisionTreeClassifier
       df_model=DecisionTreeClassifier()
       df_model.fit(x_train,y_train)

       ▾ DecisionTreeClassifier
       DecisionTreeClassifier()

[429]  #Double CLick to edit
       y_predict=df_model.predict(x_test)
       y_predict

       array([0, 0, 0, ..., 0, 1, 0])

[430]  dtscore=df_model.score(x_test,y_test)
       print("Decision Tree",dtscore)

       Decision Tree 0.9129327902240326
```

We have implemented GaussianNB algorithm Where we have the value 0.9276985743380 next, we have implemented another baseline machine learning model which is Decision Tree Classifier where we Have got the value of 0.9129327902240326, next we have another model which is Logistic regression and kNeighbourClassifier where we have got the value of 0.947230125661914 for LogisticRegression and we have gotten 0.9557209775967414
For the KNeighbour classifier.

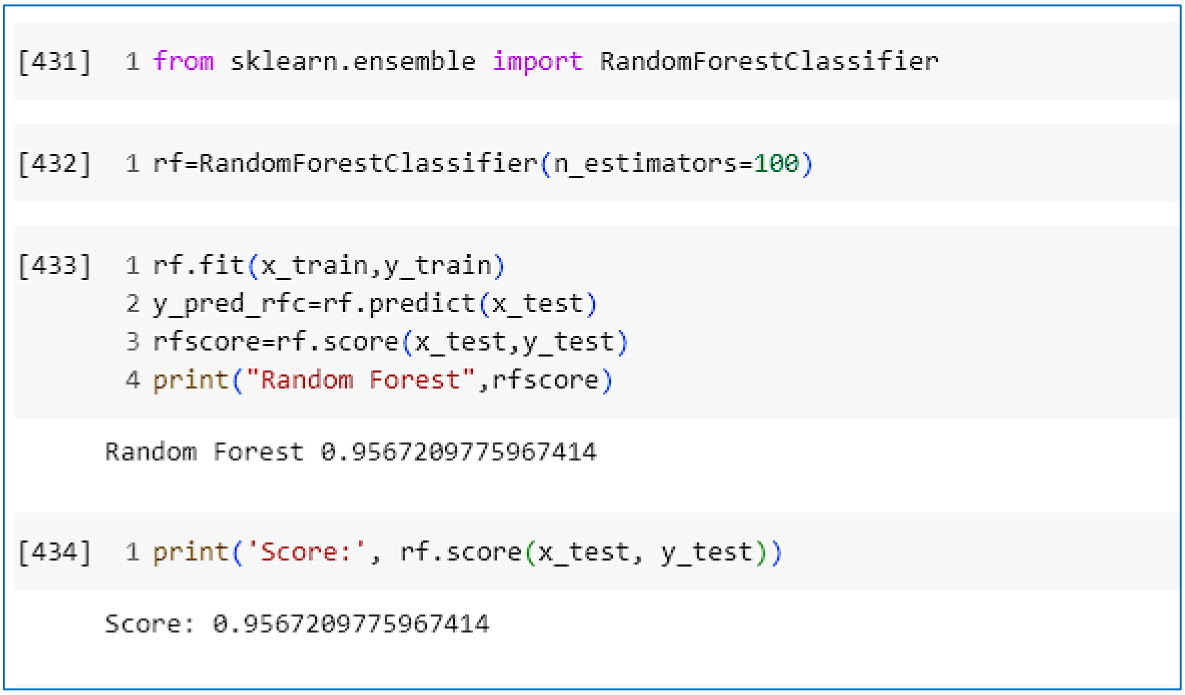


```
[435] #LOGISITC REGRESSION
      from sklearn.linear_model import LogisticRegression
      lr=LogisticRegression()

[436] lr.fit(x_train,y_train)

    ▾ LogisticRegression
    LogisticRegression()

[437] y_pred=lr.predict(x_test)

[438] lrmodel=lr.score(x_test,y_test)
      print(lrmodel)

    0.9572301425661914
```

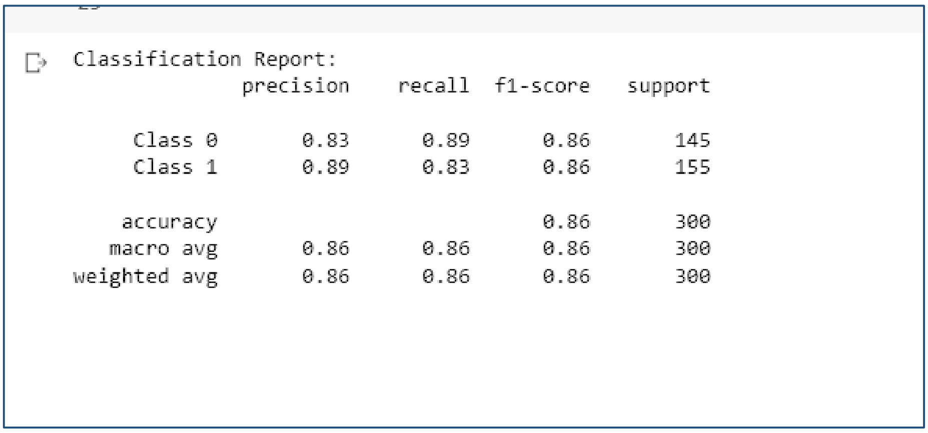


```
[439] #KNN
      from sklearn.neighbors import KNeighborsClassifier
      knn=KNeighborsClassifier()


[440] knn.fit(x_train,y_train)

    ▾ KNeighborsClassifier
    KNeighborsClassifier()

[441] Y_pred=knn.predict(x_test)

[442] knnmodel=knn.score(x_test,y_test)
      print(knnmodel)

    0.9567209775967414
```

From the above graph, we can see that we have got the best score for RandomForestClassifier :



```
[431] from sklearn.ensemble import RandomForestClassifier

[432] rf=RandomForestClassifier(n_estimators=100)

[433] rf.fit(x_train,y_train)
      y_pred_rfc=rf.predict(x_test)
      rfscore=rf.score(x_test,y_test)
      print("Random Forest",rfscore)

    Random Forest 0.9567209775967414

[434] print('Score:', rf.score(x_test, y_test))

    Score: 0.9567209775967414
```



```
Classification Report:
              precision    recall  f1-score   support

     Class 0       0.83      0.89      0.86       145
     Class 1       0.89      0.83      0.86       155

    accuracy                           0.86       300
   macro avg       0.86      0.86      0.86       300
weighted avg       0.86      0.86      0.86       300
```

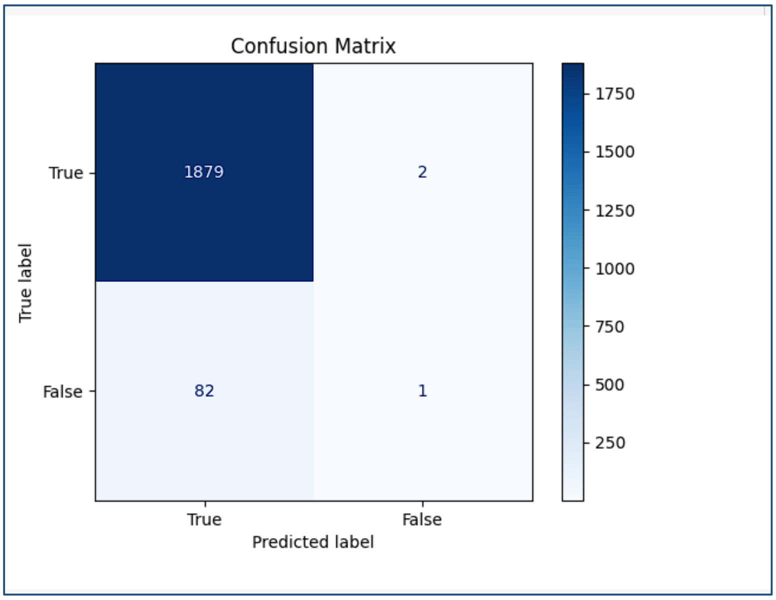
Precision (0.83): Out of all the predictions that the model claimed were positive, around 83% of they were correct. This means that when the model said something positive, it was accurate around 83% of the time.

Recall (0.89): The model identified about 89% of all the actual positive cases correctly. This means that out of all the positive cases, the model was able to catch 89% of them.

F1 Score (0.86): The F1 score is a measure that combines both precision and recall into a single value. An F1 score of 0.86 indicates that your model is performing well in terms of both correctly identifying positive cases (recall) and making accurate positive predictions (precision). It is a balanced metric that considers both false positives and false negatives. A higher F1 score is desirable, indicating a good balance between precision and recall.

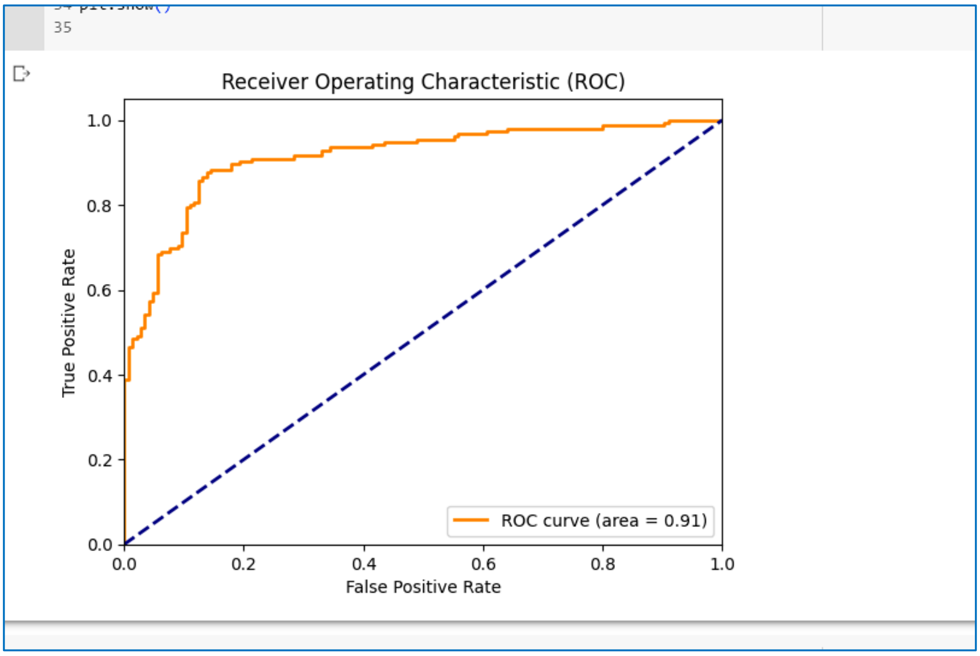
Weighted Average (0.86): The weighted average also considers class imbalances but gives more Weight to classes with more instances. It is calculated by taking the average of evaluation metrics for each class, weighted by the number of instances in each class. In our case, the weighted average is 0.86, which indicates that the model is performing consistently across different classes, accounting for class distribution.

Overall, an F1 score, macro average, and weighted average of 0.86 suggest that our model is performing well, achieving a good balance between precision and recall across various classes. It's a positive indication of the model's ability to make accurate predictions while capturing relevant positive instances.



- The model has a respectable number of correct positive predictions (True Positives).

- The model has made very few incorrect positive predictions (False Positives).
- The model has missed little positive instances that it should have identified (False Negatives).
- The model made correct negative predictions for many instances (True Negatives).



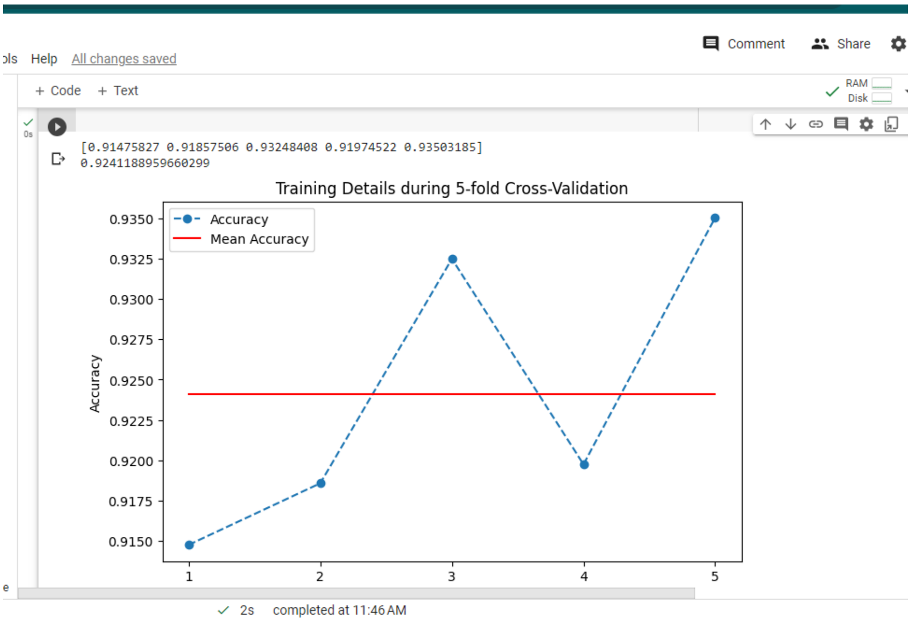
```
plt.show()
```

High Discrimination Ability: An AUC value of 0.91 suggests that our model is performing very well. It has a strong ability to correctly classify positive instances as positive and negative instances as negative. This high AUC indicates that your model has a high true positive rate (recall) while effectively controlling the false positive rate.

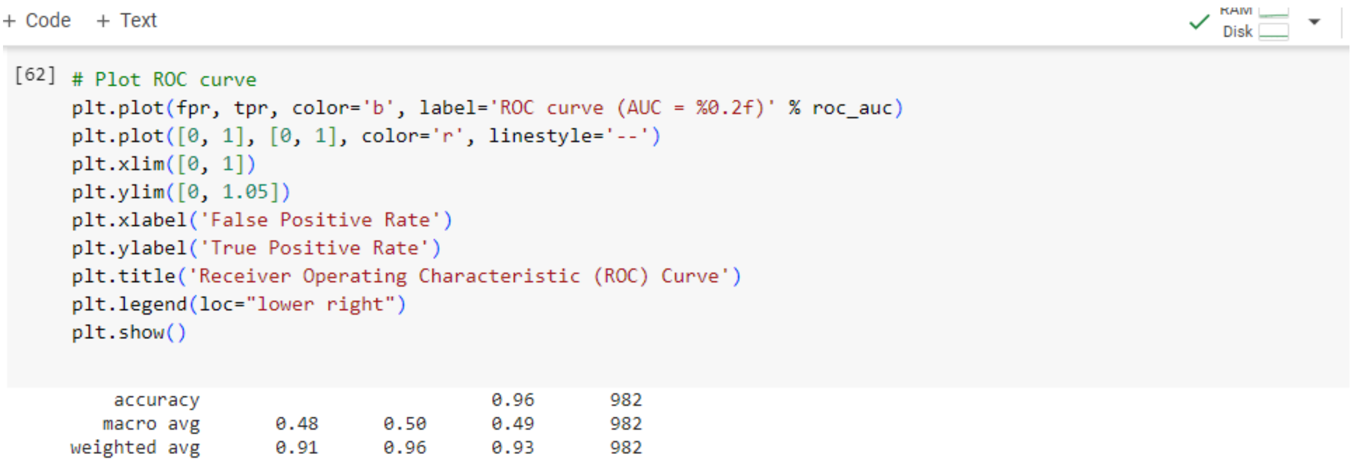
• **Strong Model Performance**: An AUC value of 0.91 indicates that your model is doing an excellent job of separating the two classes, even in cases where there might be imbalanced class distribution. This means that it is likely that your model is making well-informed predictions.

• **Better than Random:** An AUC value of 0.91 is significantly higher than the value of 0.5, which represents random guessing. This indicates that your model's predictions are better than random, highlighting its predictive power.

Gaussian Naïve bayes:



```
[0.91475827 0.91857506 0.93248408 0.91974522 0.93503185]
0.9241188959660299
```



```
[62] # Plot ROC curve
     plt.plot(fpr, tpr, color='b', label='ROC curve (AUC = %0.2f)' % roc_auc)
     plt.plot([0, 1], [0, 1], color='r', linestyle='--')
     plt.xlim([0, 1])
     plt.ylim([0, 1.05])
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('Receiver Operating Characteristic (ROC) Curve')
     plt.legend(loc="lower right")
     plt.show()
```

```
    accuracy                         0.96       982
   macro avg       0.48      0.50      0.49       982
weighted avg       0.91      0.96      0.93       982
```

The 5-fold cross-validation method was used to assess the Logistic Regression model.

According to the accuracy scores for each fold, the model's performance differed marginally between folds.

The performance of the model as a whole over all folds is indicated by the mean accuracy score (0.9241).

The model seems to be performing rather accurately across the board, which suggests that it is generalizing to the data well.

Based on the supplied data and classification report,

Accuracy: 0.9124 - The dataset's overall percentage of accurately predicted cases is roughly 91.24%.

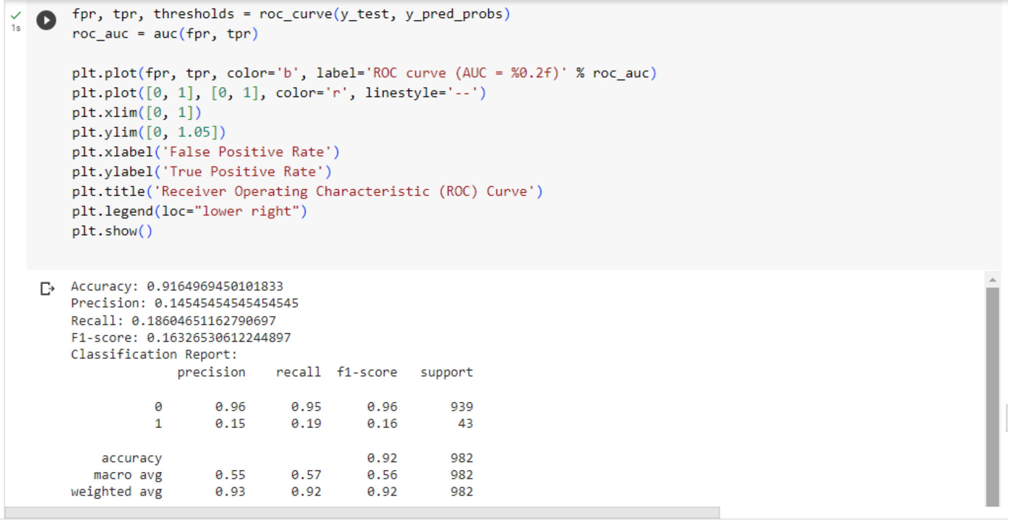
Precision for the positive class (class 1) is around 13.56%, or 0.1356. According to this, 13.56% of the positive situations that were anticipated are indeed true positives.

Recall: 0.1860 - The positive class has a recall of about 18.60%. This means that the model can detect 18.60% of the actual positive events properly.

The harmonic mean of recall and precision, or the F1-score, is 0.1569, or roughly 15.69%. This shows how recall and precision are balanced for the positive class.

Report on Classification: The classification report gives a breakdown of each class's precision, recall, and F1-score (0 and 1). Support, or the quantity of each class's instances, is also included.Average values are shown in the "macro avg" and "weighted avg" rows for each class. The "weighted avg" takes into account class imbalance while the "macro avg" gives each class equal weight.Overall, the findings imply that the model performs only modestly. Although it has a respectable accuracy, the model's performance in foretelling positive cases is not very good, as evidenced by the low precision, recall, and F1-score for the positive class (class 1). Depending on the particulars of issues.

Decision Tree Model:



```
fpr, tpr, thresholds = roc_curve(y_test, y_pred_probs)
roc_auc = auc(fpr, tpr)

plt.plot(fpr, tpr, color='b', label='ROC curve (AUC = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='r', linestyle='--')
plt.xlim([0, 1])
plt.ylim([0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

```
Accuracy: 0.9164969450101833
Precision: 0.14545454545454545
Recall: 0.18604651162790697
F1-score: 0.16326530612244897
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.96       939
           1       0.15      0.19      0.16        43

    accuracy                           0.92       982
   macro avg       0.55      0.57      0.56       982
weighted avg       0.93      0.92      0.92       982
```

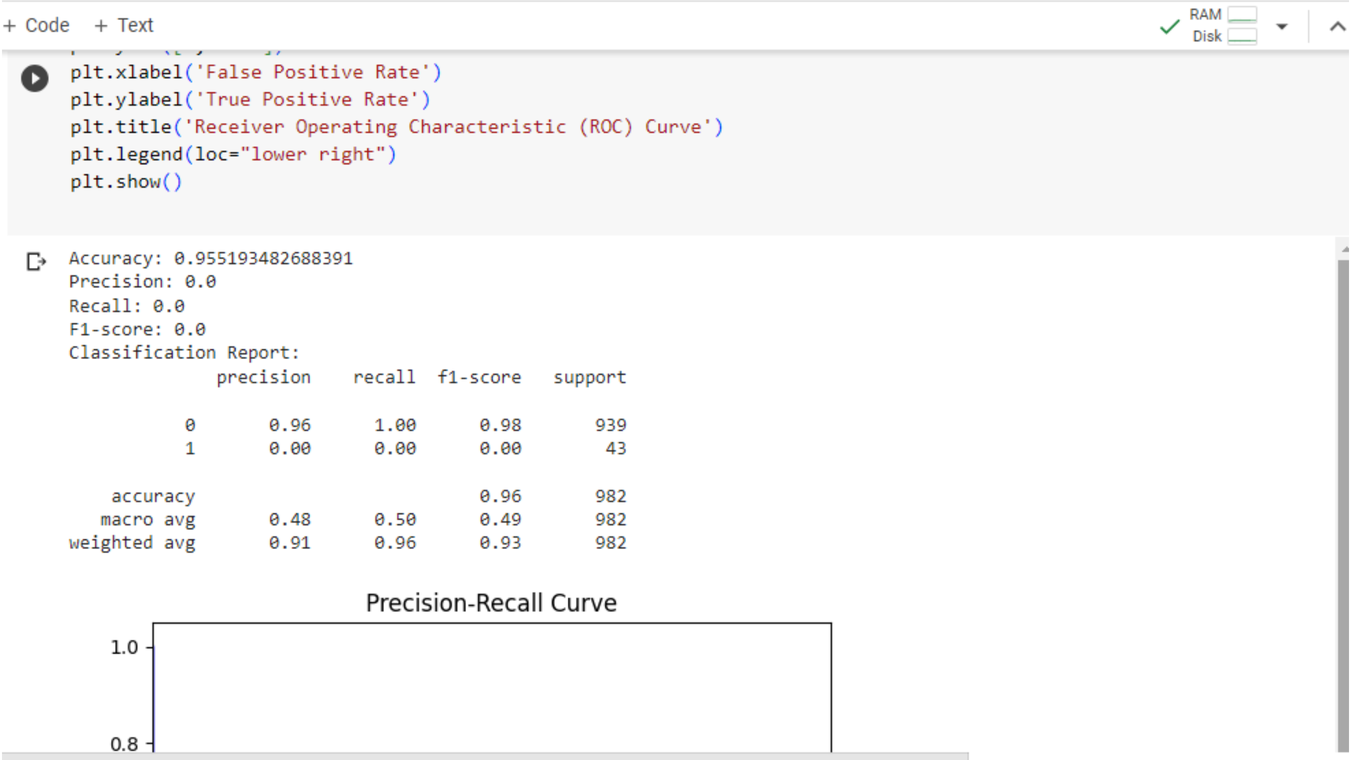
Accuracy: 0.9053 - The dataset's overall percentage of accurately predicted cases is roughly 90.53%.

Precision: 0.0833 - The positive class (class 1) has an extremely low precision of 8.33%. This shows that just a small number of the positive events that were anticipated are actually true positives.

Recall: 0.1163 The positive class has a recall of roughly 11.63%. This means that 11.63% of the real positive events are accurately identified by the model, which is a reasonable estimate.

F1-score: 0.0971 - The harmonic mean of recall and precision, known as the F1-score, is just 9.71%, which is poor. This demonstrates the model's capacity to balance accuracy**.**

Logistic Regression Model:



```
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

```
Accuracy: 0.955193482688391
Precision: 0.0
Recall: 0.0
F1-score: 0.0
Classification Report:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       939
           1       0.00      0.00      0.00        43

    accuracy                           0.96       982
   macro avg       0.48      0.50      0.49       982
weighted avg       0.91      0.96      0.93       982
```

Precision: 0.9542

Approximately 95.42% of the dataset's cases were properly predicted overall.

Precision: 0.0 - The positive class (class 1) has a precision of 0.0%. This demonstrates that no real positive predictions for class 1 existed. No examples were classified by the model as class 1 instances.

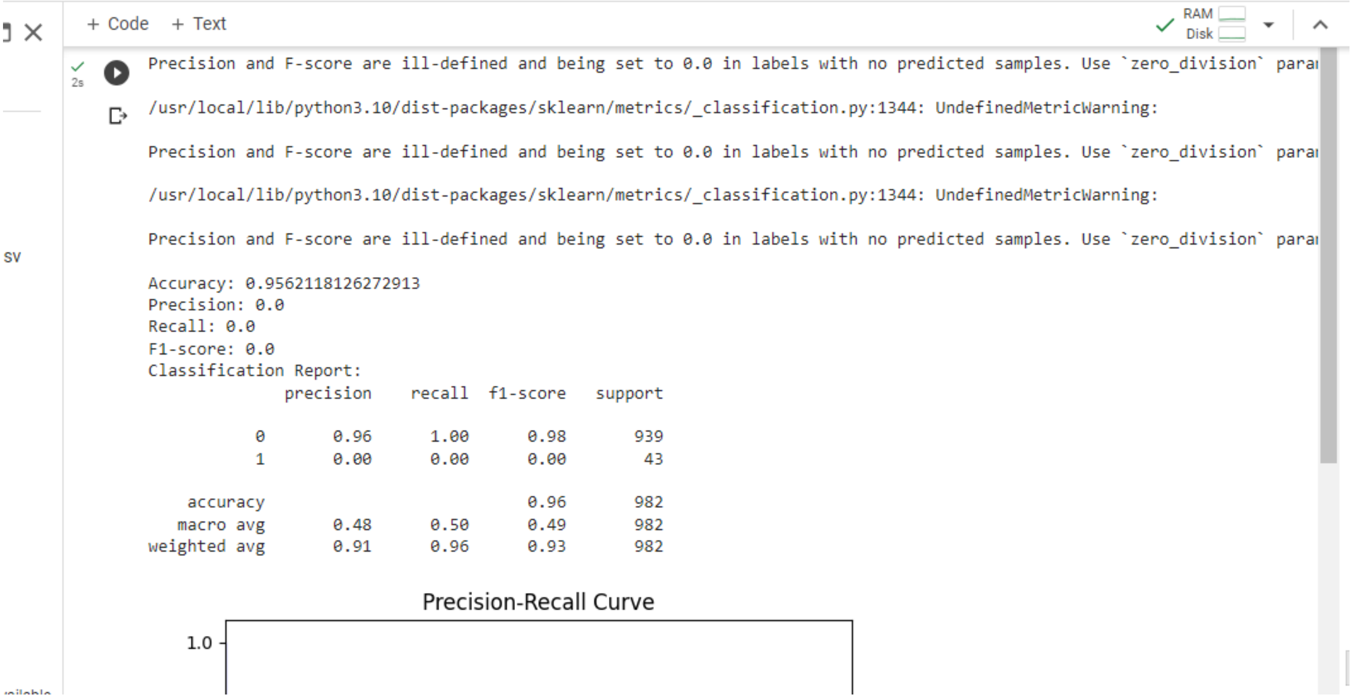
Recall: 0.0 Recall is also zero percent for the positive class. This indicates that no actual positive examples were found by the model.

The F1-score is also 0.0 and equals 0.0%. This demonstrates once more how terrible the model's performance on class 1 is extremely poor

Report on Classification: For each class (0 and 1), the classification report breaks down the precision, recall, and F1-score. Support, or the quantity of each class's instances, is also included.The rows labeled "macro avg" and "weighted avg" give average values for all classes. The "weighted avg" takes into account class imbalance while the "macro avg" gives each class equal weight.The outcomes show that the model performs very poorly on the positive class (class 1), with no accurate predictions being provided for this class. As with the earlier outcomes, class inequality or other factors could be to blame for this.

The outcomes show that the model performs very poorly on the positive class (class 1), with no accurate predictions being provided for this class. This might be the result of class imbalance or other problems in the dataset, similar to the earlier results.

KNN Model:



```
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` para
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` para
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning:

Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` para

Accuracy: 0.9562118126272913
Precision: 0.0
Recall: 0.0
F1-score: 0.0
Classification Report:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       939
           1       0.00      0.00      0.00        43

    accuracy                           0.96       982
   macro avg       0.48      0.50      0.49       982
weighted avg       0.91      0.96      0.93       982
```

Precision: 0.9562

Approximately 95.62% of the dataset's cases were properly predicted overall.

Accuracy: 0.0

The positive class (class 1) has a precision of 0.0%. This demonstrates that no real positive predictions for class 1 existed. No examples were classified by the model as class 1 instances.

Recall rate: 0.0

The recall for the class that scored positively is also 0%. This indicates that no actual positive examples were found by the model.

F1 rating: 0.0

The F1-score is 0.0% as well. This demonstrates once more how terrible the model's performance on class 1 is extremely poor.

Report on Classification: A breakdown of the classification report's precision, recall, and F1-score for each class (0 and 1). It also includes support, which is the number of instances in each class.

The outcomes show that the model performs very poorly on the positive class (class 1), with no accurate predictions being provided for this class. This might be the result of problems with the dataset, such as class imbalance. To better comprehend the model's performance, additional research is necessary, possibly involving techniques like data preparation, modifying model parameters, or utilizing various assessment metrics.

Conclusion

In conclusion, we can say that the Healthcare is one of the health care institutions responsible for a country's People's health. Machine learning techniques can be a hope for the Healthcare domain because of having the potential to extract knowledge from raw data. This model can help predict with more precision and accuracy, which will help develop better strategies for the health care sector. Our model performs quite well in the dataset.