Stroke Detection: Prediction with Multiple Classification Models

Group 13 Project Progress Report

University of Wisconsin - Madison
Computer Science 539 - Introduction to Artificial Neural Networks

Tony Wang - ywang3298@wisc.edu
College of Letters and Science - Undergraduate
Hanchi Chen - Hchen794@wisc.edu
College of Letters and Science - Undergraduate

Submission date: Nov.18, 2023

Abstract

The goal of this project is to use multiple machine learning classification algorithms to predict whether a patient have a stroke based on several parameters like gender, age, various diseases, and smoking status. The Stroke Detection project is progressing as planned. We have rigorously explored our dataset, implementing preprocessing measures and conducting a thorough visualization and analysis. Following baseline research, we successfully validate an example algorithm on Kaggle[1]. Subsequently, we used several models including Support Vector Classification (SVC), Logistic Regression, Random Forest, and XGBoost. A portion of these models were further refined using k-fold cross-validation. The performance of these algorithms was evaluated using confusion matrix, precision, and other relevant metrics. In the following weeks, we would like to develop algorithms including CNN or RNN for further test and evaluation.

1 Introduction

1.1 Overview

Machine learning algorithms can analyze large amounts of data and identify patterns and correlations that can be used to make predictions with a high degree of accuracy. Our aim is to optimize the predictive power of various machine-learning classifiers for stroke detection. The ultimate goal will be finding an effective model to predict if the patient has a stroke or not.

1.2 Motivation and Related Works

According to the data from the World Health Organization (WHO) in 2019, stroke is the second leading cause of death, which is responsible for approximately 11% of total deaths respectively[2]. What's more, every 40 seconds, someone in the United States has a stroke. Every 3 minutes and 14 seconds, someone dies of stroke[3]. Stroke should be paid attention to more than thought.

The current workflow when patients want to check if they have a stroke typically begins with a series of examinations, including Computerized tomography (CT), Blood tests, physical exams, and so on. Not only is this approach time-consuming, but it also poses significant financial burdens. Rather than focusing predominantly on stroke treatment, the emphasis should shift towards prevention, especially considering the abrupt onset of stroke symptoms which often renders patients incapacitated, and unable to seek timely assistance.

In fact, many recent studies focus on training deep learning models for medical purposes, for instance, brain tumor detection and heart attack detection, are already successfully used as efficient and friendly methods. A promising solution for stroke detection may also lie in leveraging neural networks to analyze pertinent data, thereby predicting individuals at heightened risk of experiencing a stroke.

2 Methods

2.1 Datasets

We found the data from https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/data. The data contains 5110 observations. There are 12 features in the dataset and some missing data for BMI. We conduct data analysis to check the correlation of each feature to the target, and currently decided to just drop out the "id" of each sample since it does not influence detection. For data pre-processing, we use many techniques like Label-Encoder for encoding, SMOTE for data imbalance, imputation for missing data, and feature scaling for normalized data. We are initially planning to 80/20 split the dataset and test accuracy against the 20% test sets.

S.No.	Attribute	Code given	Value ranges	Data type
1	Gender	gender	1, 0	Binary
2	Age	age	in years	Numeric
3	Hypertension binary feature	hypertension	1, 0	Binary
4	Heart disease binary feature	heart_disease	1, 0	Binary
5	Has the patient ever been married	ever_married	1, 0	Binary
6	Work type of the patient	work_type	0, 1, 2	Nominal
7	Residence type of the patient	Residence_type	0, 1	Binary
8	Average glucose level in blood	avg_glucose_level	55.1–272	Numeric
9	Body Mass Index	bmi	10.3-97.6	Numeric
10	Smoking status of the patient	smoking_status	0,1,2	Nominal
11	Stroke event	stroke	0,1	Binary

Table 1: Description of Dataset Attributes

2.2 Data Analysis and Pre-processing

The following image shows basic information of features of this dataset, with 5110 valid samples.

```
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   gender             5110 non-null   object
 1   age                5110 non-null   float64
 2   hypertension       5110 non-null   int64
 3   heart_disease      5110 non-null   int64
 4   ever_married       5110 non-null   object
 5   work_type          5110 non-null   object
 6   Residence_type     5110 non-null   object
 7   avg_glucose_level  5110 non-null   float64
 8   bmi                4909 non-null   float64
 9   smoking_status     5110 non-null   object
 10  stroke             5110 non-null   int64
dtypes: float64(3), int64(3), object(5)
memory usage: 439.3+ KB
```

Figure 1, Dataset Info.

1. Data Imputation: We use KNNImputer to deal with missing data in BMI, and for the "Unknown" state in the feature "Smoking status of the patient", we assume it is a virtual state and encode it separate from other states.

```
gender                0
age                   0
hypertension          0
heart_disease         0
ever_married          0
work_type             0
Residence_type        0
avg_glucose_level     0
bmi                 201
smoking_status        0
stroke                0
dtype: int64
```

Figure 2, missing data.

2. Data Imbalance: This dataset includes ten times more negative samples than positive ones, which is common in real-world diseases. However, This can be a problem because the algorithm might only learn from the majority of examples and not pay enough attention to the minority examples, thus the model can not perform well to predict. Therefore, we use SMOTE to fix this problem by creating new, fake examples of the minority type so that they become more evenly represented in the dataset. After Oversampling, we get a new dataset with 7788 samples, which includes 3894 stroke samples and 3894 healthy samples.

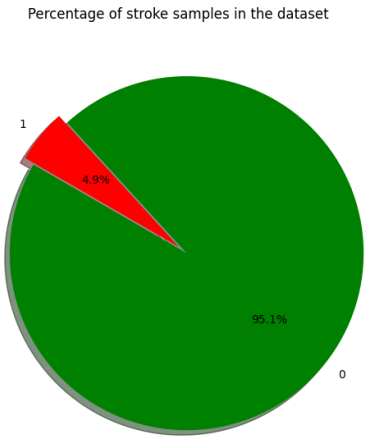


Figure 3, imbalance sample.

3. Data Importance: We test the correlation of each feature to stroke in the following image. The main factor to stroke is 'Age', which is significant and intuitive here. However, other factors are not enough influential and are quite similarly important, which may cause problems in prediction.

	feature	importance
1	age	0.557018
9	avg_glucose_level	0.133431
5	work_type	0.088479
8	bmi	0.063563
6	Residence_type	0.061046
0	gender	0.043540
7	smoking_status	0.026112
4	ever_married	0.018408
2	hypertension	0.004438
3	heart_disease	0.003965

Figure 4, data importance.

The following analysis for detailed features referenced the code in baseline research[1].

For 'work type' and 'smoke status', they seem to be less significant for stroke, since the stroke ratios among all jobs and smoking groups are surprisingly similar.

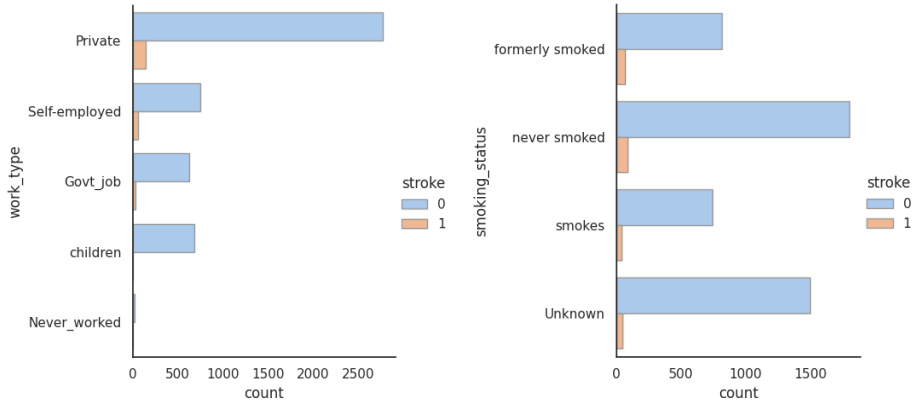


Figure 5.

We separate the stroke samples of different genders when comparing the stroke density of some features, which shows that gender is less discriminate in stroke detection. Patients in Urban places are more likely to have stroke. Meanwhile, hypertension and heart disease increase the possibility of stroke.

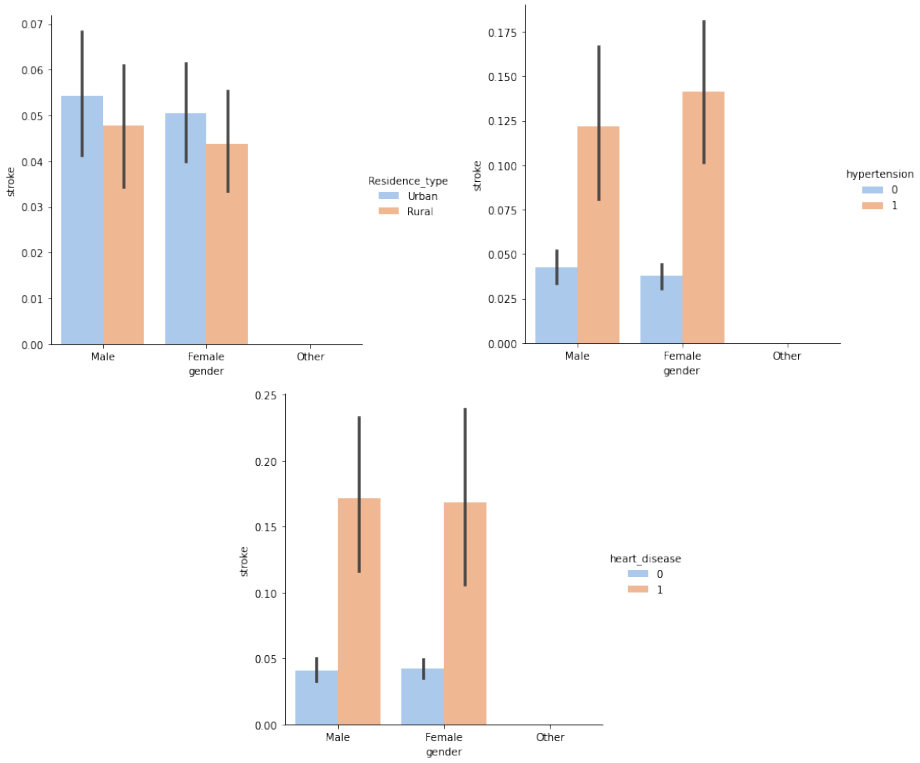


Figure 6.

4. Data encoding: We use LabelEncoder to encode the categorical columns: like 'gender', 'work type', and 'smoking status'.

5. Data scaling: We noticed that these data are not standard, for instance, after encoding, features like heart disease become binary integers, but bmi can be 100 times larger than binary. This may cause some problems with model training. However, we used the StanderdScalar in sklearn to standardize the features, but we didn't get good performance, so we gave up scaling currently.

2.3 Algorithm and Program

Stroke prediction models predict an individual's risk of having a stroke based on various factors such as age, blood pressure, cholesterol levels, smoking status, and other medical history. Machine learning algorithms can analyze large amounts of data and identify patterns and correlations that can be used to make predictions with a high degree of accuracy. Our aim is to optimize the predictive power of various machine-learning classifiers for stroke detection.

Some existing work already reached more than 90 percent predictive accuracy in Kaggle. For example, RachidYZ achieved an accuracy of more than 95 percent.[1] We considered it as a baseline example result for us. For this project, we explore multiple models such as Logistic regression, Support Vector Classifier (SVC), Random Forest, and XGBoost. We are considering implementing specific Artificial Neural Network (ANN) models (may include some pre-trained models), in the later weeks. We compare the predictive accuracy and other metrics for stroke detection.

We also utilize k-fold cross-validation for models for further performance improvement. Cross-validation is a more robust technique that involves partitioning the dataset into k-folds, and currently we choose 5-fold considering computation and performance. Cross-validation provides a more reliable estimate of the model's performance and helps to reduce the risk of over-fitting.

2.4 Platform

In our project, we primarily utilize GitHub for version control and Google Colab for collaborative development of our IPython Notebooks (.ipynb files). Colab's cloud-based environment facilitates simultaneous access and simplifies the code execution environment, eliminating the need for local setup.

The integration of GitHub ensures a clear version history, enhancing the maintainability and scalability of our project. Additionally, we are considering incorporating Google Cloud Platform (GCP) for its advanced computational resources and flexible data storage and processing capabilities.

2.5 Results and Performance Evaluation Metrics

Our primary objective is to design an efficient model for stroke detection by exploring a wide spectrum of classification models and determining the most proficient model based on predictive accuracy. To evaluate our model's performance, we will use both confusion matrices and predictive accuracy as our performance metrics. Past works in this field have set a high benchmark in terms of prediction accuracy. Given these implementations, our project aspires to achieve, or even surpass, a predictive accuracy of at least 95 percent.

We also use additional performance matrices like recall, f1-score, precision, confusion matrix, ROC, and AOC. ROC (Receiver Operating Characteristic) and AUC (Area Under the ROC Curve)

are evaluation metrics commonly used in machine learning for binary classification problems. We use ROC/AUC as evaluation metrics for binary classification models because they provide a more comprehensive view of the model's performance than accuracy alone.

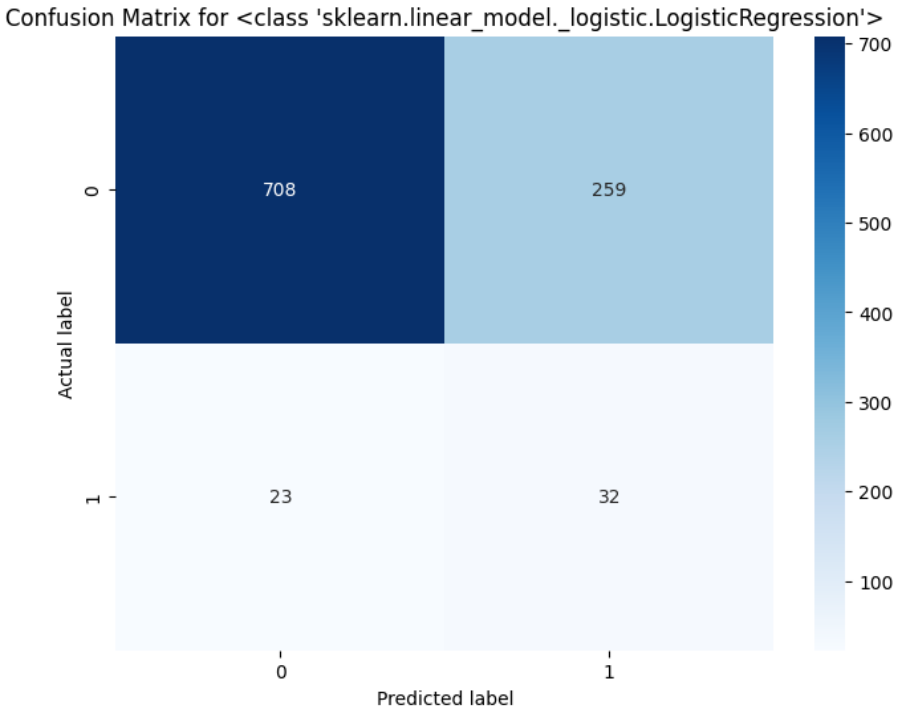
3 Results

The following images show the main performance metrics, the confusion matrix, and the ROC/AUC. Random forest achieves the highest accuracy 100 percent, which surpasses the performance of baseline research. It is important to notice that the performance of Logistic Regression and SVM are limited to around 70-80 percent because the precision at positive samples is very low, just above 10 percent, which must not be accepted. The reason behind this may be the imbalance of original datasets is not totally solved, which makes these two models tend to predict negative.

1. Logistic Regression

	precision	recall	f1-score	support
0	0.97	0.73	0.83	967
1	0.11	0.58	0.18	55
accuracy			0.72	1022
macro avg	0.54	0.66	0.51	1022
weighted avg	0.92	0.72	0.80	1022

Validation Accuracy: 0.7240704500978473
Training Accuracy: 0.7441291585127201



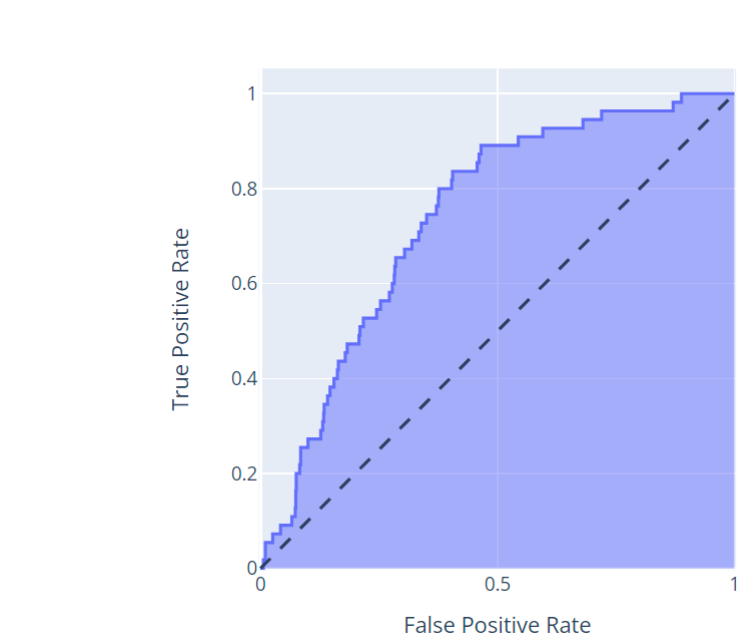


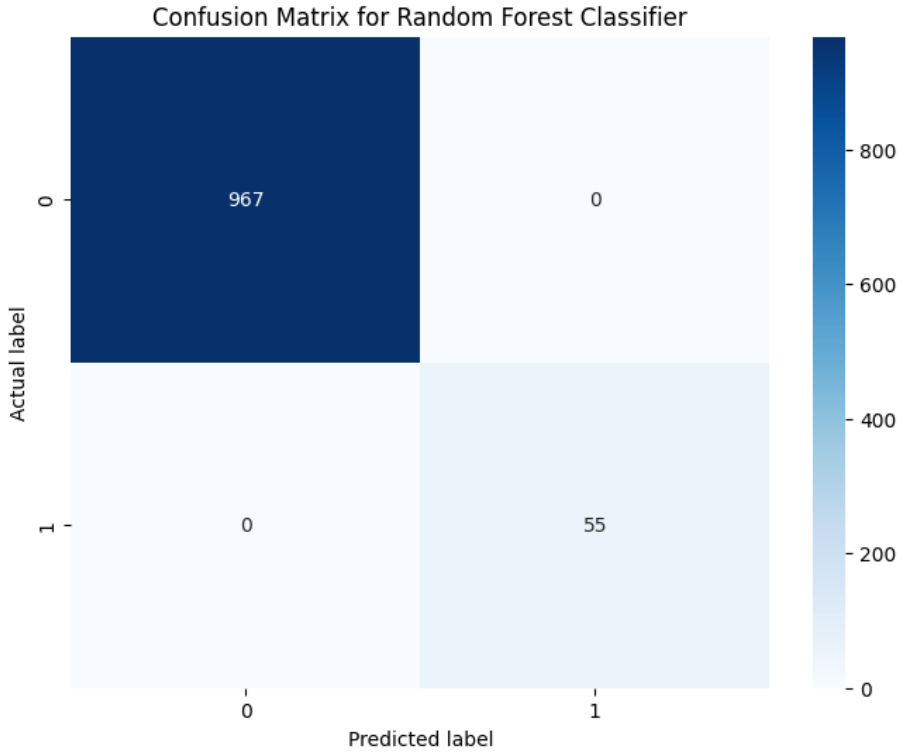
Figure 7, Logistic Regression Results.

2. Random Forest

Random Forest Classifier

	precision	recall	f1-score	support
0	1.00	1.00	1.00	967
1	1.00	1.00	1.00	55
accuracy			1.00	1022
macro avg	1.00	1.00	1.00	1022
weighted avg	1.00	1.00	1.00	1022

Validation Accuracy: 1.0
Training Accuracy: 1.0



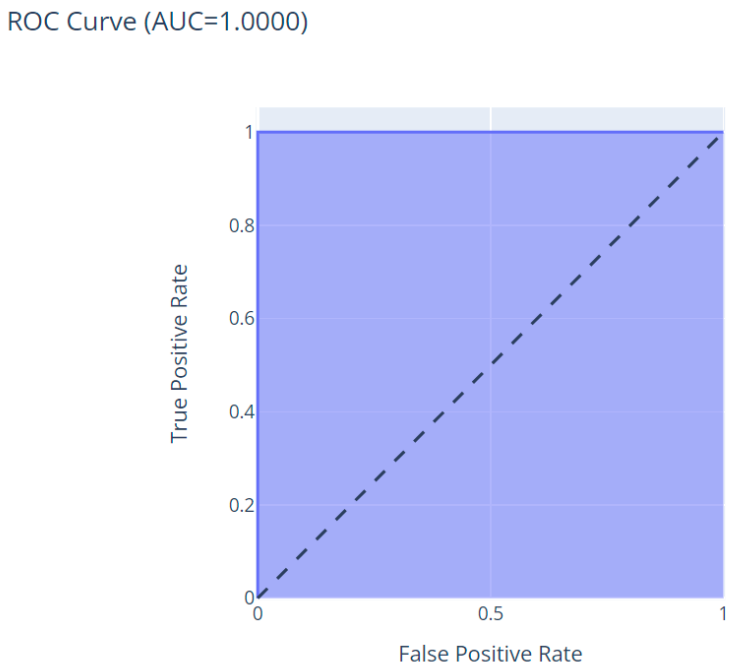


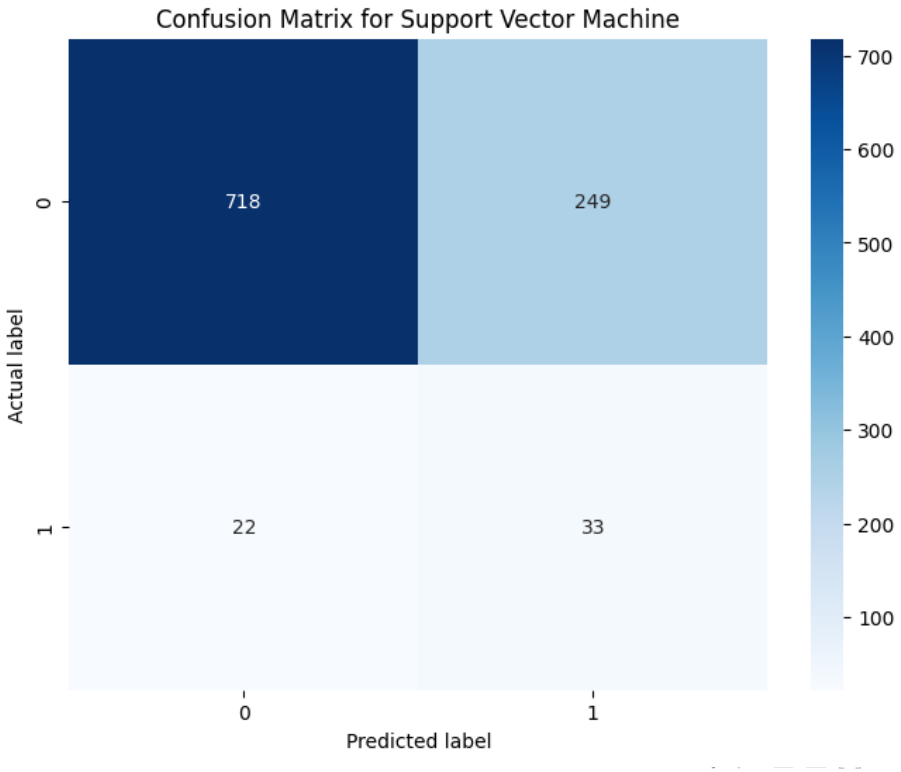
Figure 8, Random Forest Results.

3. Support Vector Machine

Support Vector Machine

	precision	recall	f1-score	support
0	0.97	0.74	0.84	967
1	0.12	0.60	0.20	55
accuracy			0.73	1022
macro avg	0.54	0.67	0.52	1022
weighted avg	0.92	0.73	0.81	1022

Validation Accuracy: 0.7348336594911937
Training Accuracy: 0.7566046966731899



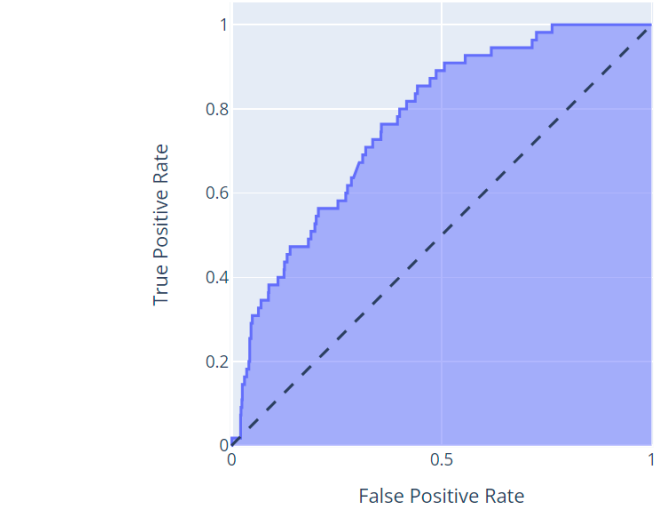


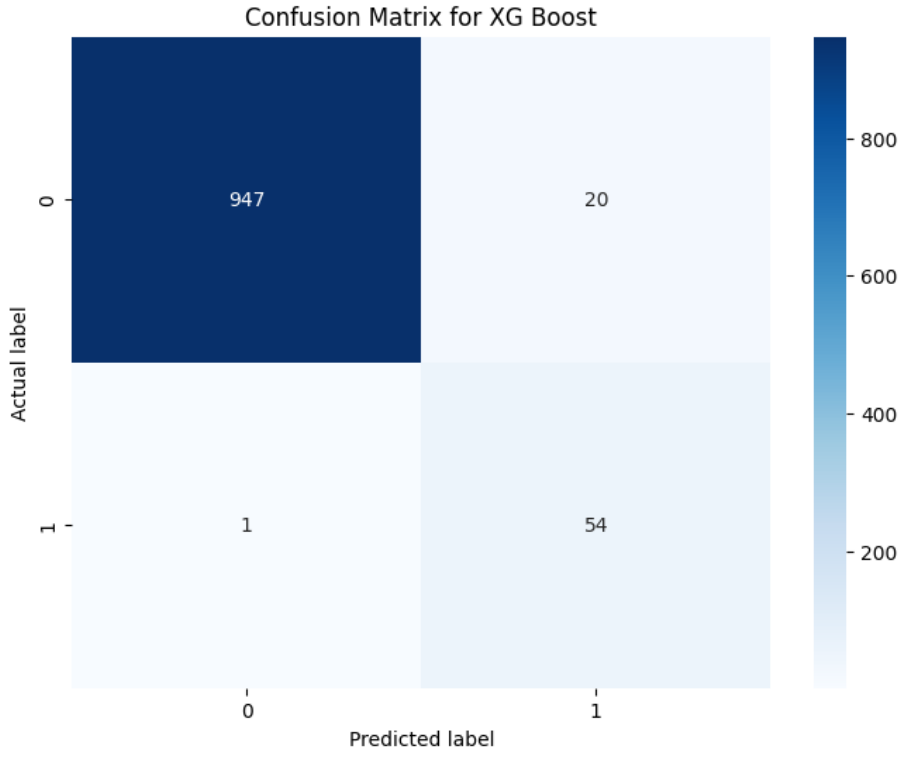
Figure 9, SVM Results.

4. XG Boost

XG Boost

	precision	recall	f1-score	support
0	1.00	0.98	0.99	967
1	0.73	0.98	0.84	55
accuracy			0.98	1022
macro avg	0.86	0.98	0.91	1022
weighted avg	0.98	0.98	0.98	1022

Validation Accuracy: 0.9794520547945206
Training Accuracy: 0.9796966731898239



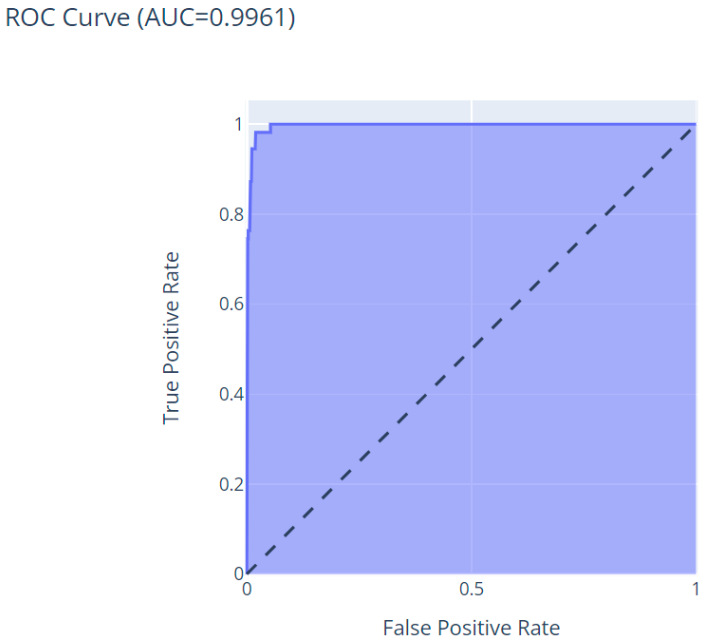


Figure 10, XG Boost Results.

4 Discussion

Many models perform with several problems. For example, XGBoost shows high training accuracy 98.39% but lower validation accuracy (89.14%), indicating potential overfitting. Also, there's a significant imbalance in class performance, with poor precision (18%) and recall (29%) for the minority class. This could be due to data imbalance, bad configuration with model parameters, or insufficient features. The overall accuracy is high, but the F1 score for the minority class is low, highlighting challenges in handling imbalanced datasets.

Critical factors in this performance discrepancy are attributed to the inherent class imbalance within our dataset and the imbalance value between features. The minority class, being underrepresented, has led to the model's inadequate learning of its characteristics, thus impairing its predictive capability on the validation set. Also, the model may pay over attention to those features that have large original values.

Furthermore, the application of SMOTE on our test samples has notably influenced the outcomes. When SMOTE is generally employed for training datasets only, it performs worse. In this

context, SMOTE may have inadvertently introduced a bias towards the dataset, thereby skewing the model's ability to generalize on unseen data. This underscores the need for careful consideration when applying over-sampling techniques, as they can significantly affect the model's performance, especially in the context of highly imbalanced datasets.

In light of these findings, future work should focus on optimizing the model parameters specifically tailored to address data imbalance. Also, more advanced algorithms and better hyperparameters will also be explored in the next few weeks.

5 References

[1]RACHIDYZ. (2020, May 22). EDA and modeling for predicting stroke. Kaggle. Retrieved October 19, 2023, from https://www.kaggle.com/code/rachidyz/eda-and-modeling-for-predicting-stroke

[2]"The Top 10 Causes of Death." World Health Organization, World Health Organization, www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death. Accessed 21 Oct. 2023.

[3]"How Many People Are Affected by/at Risk for Stroke?" Eunice Kennedy Shriver National Institute of Child Health and Human Development, U.S. Department of Health and Human Services, www.nichd.nih.gov/health/topics/stroke/conditioninfo/risk. Accessed 21 Oct. 2023.