BRAIN STROKE PREDICTION

Using Python

Divy Kamalpuria
B.Tech. IT

Brain Stroke Prediction Project

Abstract

The Brain Stroke Prediction Project aims to develop a predictive model for identifying individuals at risk of suffering from strokes. Strokes are a significant health concern globally, and early detection plays a crucial role in preventing severe consequences. In this project, we utilize various machine learning algorithms, including Logistic Regression, K-Nearest Neighbors (KNN), and Decision Tree Learning, to analyze a dataset containing health-related information. The goal is to build accurate models that can predict whether an individual is at risk of a stroke based on their health characteristics. This document provides a comprehensive overview of the project, including its importance, data preprocessing, model selection, and evaluation.

Introduction

Background

Stroke is a medical emergency that occurs when there is a disruption in the blood supply to the brain, leading to brain cell damage and potentially life-threatening consequences. According to the World Health Organization (WHO), strokes are one of the leading causes of death and disability globally. Detecting and preventing strokes early are essential to reduce the associated mortality and improve the quality of life for those at risk.

Project Objectives

The primary objectives of the Brain Stroke Prediction Project are as follows:

1. Develop machine learning models to predict the likelihood of an individual experiencing a stroke based on their health-related data.

2. Identify the most relevant features contributing to stroke prediction, providing valuable insights for healthcare professionals.

3. Evaluate and compare the performance of different machine learning algorithms for this predictive task.

Data Overview

The project utilizes a dataset named 'brain_stroke.csv' containing various attributes related to individuals' health and stroke occurrences. The dataset includes the following columns:

1. `gender`: Gender of the individual.

2. `age`: Age of the individual.

3. `hypertension`: Whether the individual has hypertension (1 for yes, 0 for no).

4. `heart_disease`: Whether the individual has heart disease (1 for yes, 0 for no).

5. `ever_married`: Marital status of the individual.

6. `work_type`: Type of work the individual is engaged in.

7. `Residence_type`: Type of residence (urban or rural).

8. `avg_glucose_level`: Average glucose level in the individual's blood.

9. `bmi`: Body Mass Index of the individual.

10. `smoking_status`: Smoking status of the individual.

11. `stroke`: Target variable indicating whether the individual had a stroke (1 for yes, 0 for no).

Here is a overview of Dataset –

```
df.head()
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
2	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
3	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
4	Male	81.0	0	0	Yes	Private	Urban	186.21	29.0	formerly smoked	1

```
df.tail()
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
4976	Male	41.0	0	0	No	Private	Rural	70.15	29.8	formerly smoked	0
4977	Male	40.0	0	0	Yes	Private	Urban	191.15	31.1	smokes	0
4978	Female	45.0	1	0	Yes	Govt_job	Rural	95.02	31.8	smokes	0
4979	Male	40.0	0	0	Yes	Private	Rural	83.94	30.0	smokes	0
4980	Female	80.0	1	0	Yes	Private	Urban	83.75	29.1	never smoked	0

```
# some description's about dataset
df.describe()
```

	age	hypertension	heart_disease	avg_glucose_level	bmi	stroke
count	4981.000000	4981.000000	4981.000000	4981.000000	4981.000000	4981.000000
mean	43.419859	0.096165	0.055210	105.943562	28.498173	0.049789
std	22.662755	0.294848	0.228412	45.075373	6.790464	0.217531
min	0.080000	0.000000	0.000000	55.120000	14.000000	0.000000
25%	25.000000	0.000000	0.000000	77.230000	23.700000	0.000000
50%	45.000000	0.000000	0.000000	91.850000	28.100000	0.000000
75%	61.000000	0.000000	0.000000	113.860000	32.600000	0.000000
max	82.000000	1.000000	1.000000	271.740000	48.900000	1.000000

```
# each column has how many unique values
df.nunique()
```

```
gender               2
age                104
hypertension         2
heart_disease        2
ever_married         2
work_type            4
Residence_type       2
avg_glucose_level    3895
bmi                342
smoking_status       4
stroke               2
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4981 entries, 0 to 4980
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   gender             4981 non-null   object 
 1   age                4981 non-null   float64
 2   hypertension       4981 non-null   int64  
 3   heart_disease      4981 non-null   int64  
 4   ever_married       4981 non-null   object 
 5   work_type          4981 non-null   object 
 6   Residence_type     4981 non-null   object 
 7   avg_glucose_level  4981 non-null   float64
 8   bmi                4981 non-null   float64
 9   smoking_status     4981 non-null   object 
 10  stroke             4981 non-null   int64  
dtypes: float64(3), int64(3), object(5)
memory usage: 428.2+ KB
```

Methodology

Data Preprocessing

Data preprocessing is a critical step in preparing the dataset for machine learning. The following steps are performed:

- Handling missing values: Rows with missing data are either removed or filled based on the context.

- Encoding categorical variables: Categorical columns like `gender`, `ever_married`, `work_type`, `Residence_type`, and `smoking_status` are converted into numerical values using techniques like one-hot encoding or label encoding.

- Feature selection: The most relevant features for stroke prediction are selected using techniques such as correlation analysis and feature importance.

```
# checking that do we have missing value or not
# we don't have missing value
df.isnull().sum()
```

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

```
# each column has how many unique values
df.nunique()
```

```
gender                 2
age                  104
hypertension           2
heart_disease          2
ever_married           2
work_type              4
Residence_type         2
avg_glucose_level    3895
bmi                   342
smoking_status          4
stroke                  2
dtype: int64
```

```
for column_name in list(temp.columns):
    print('[{0}]:'.format(column_name))
    print(temp[column_name].value_counts(),'\n')
```

```
[gender]:
Female    2907
Male      2074
Name: gender, dtype: int64

[hypertension]:
0    4502
1     479
Name: hypertension, dtype: int64

[heart_disease]:
0    4706
1     275
Name: heart_disease, dtype: int64

[ever_married]:
Yes    3280
No     1701
Name: ever_married, dtype: int64

[work_type]:
Private          2860
Self-employed     804
children          673
Govt_job          644
Name: work_type, dtype: int64

[Residence_type]:
Urban    2532
Rural    2449
Name: Residence_type, dtype: int64

[smoking_status]:
never smoked       1838
Unknown            1500
formerly smoked     867
smokes              776
Name: smoking_status, dtype: int64

[stroke]:
0    4733
1     248
Name: stroke, dtype: int64
```

Model Selection

Random Forest is a popular algorithm used for various classification tasks, including brain stroke prediction, due to its effectiveness and robustness. Here are some reasons why Random Forest may be a suitable choice for brain stroke prediction:

1. Handling Non-linearity: Random Forest can handle complex, non-linear relationships between predictor variables (features) and the target variable (stroke) effectively. Brain stroke prediction often involves multiple factors with non-linear interactions, making Random Forest a suitable choice.

2. Feature Importance: Random Forest provides a measure of feature importance. This is valuable in medical applications like stroke prediction because it helps identify which features (such as age, hypertension, or smoking status) have the most significant impact on the prediction. This information can aid medical professionals in understanding risk factors.

3. Robustness: Random Forest is robust to outliers and noisy data. In healthcare datasets, outliers and noisy measurements can be common. Random Forest can handle such data challenges gracefully.

4. Reducing Overfitting: Random Forest mitigates overfitting, which is essential when working with limited medical data. Overfitting occurs when a model learns noise in the data rather than the underlying patterns. The ensemble nature of Random Forest reduces overfitting by combining multiple decision trees.

5. Ensemble Learning: It's an ensemble learning method that combines multiple decision trees, reducing the risk of errors from individual trees. This ensemble approach often results in more accurate and stable predictions.

6. Imbalanced Data: In medical datasets, class imbalance can be an issue, as you often have fewer instances of the target event (in this case, strokes). Random Forest handles imbalanced data well and can still make accurate predictions for minority classes.

7. Interpretability: While Random Forest is an ensemble of decision trees, it provides insight into feature importance and decision processes. This interpretability is crucial in healthcare applications, where understanding the reasoning behind predictions is essential.

8. Ease of Use: Random Forest is relatively easy to implement and tune. It doesn't require extensive hyperparameter tuning compared to some other complex algorithms.

However, the choice of algorithm also depends on the dataset, its size, and the specific problem you're trying to solve. While Random Forest is a strong candidate for many classification tasks, it's essential to evaluate various algorithms to determine which one performs best for your specific brain stroke prediction problem. Additionally, feature engineering and dataset quality play a significant role in the success of any machine learning model for healthcare applications.

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, roc_curve, roc_auc_score
```

```
# Load the dataset
data = pd.read_csv('brain_stroke.csv')

# Data Preprocessing
# Handle missing values
data.dropna(inplace=True)

# Convert categorical variables to numerical using one-hot
encoding
data = pd.get_dummies(data, columns=['gender', 'ever_married',
'work_type', 'Residence_type', 'smoking_status'], drop_first=True)

# Split data into features (X) and target (y)
X = data.drop('stroke', axis=1)
y = data['stroke']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Model Training
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Model Evaluation
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')

# Classification Report
report = classification_report(y_test, y_pred)
print(report)
```

```
# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

# ROC Curve
y_prob = clf.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_prob)
roc_auc = roc_auc_score(y_test, y_prob)
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'ROC Curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc='lower right')
plt.show()
```

Model Evaluation

The performance of each model is assessed using various evaluation metrics, including:

Accuracy: 0.9398194583751254

This represents the accuracy of the model, which is approximately 93.98%. Accuracy is the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances in the test dataset. In this case, the model correctly predicts stroke cases (class 1) and non-stroke cases (class 0) about 93.98% of the time.

Classification Report: The classification report provides several metrics for each class (0 and 1) and includes the following:

- Precision: Precision measures the proportion of true positive predictions among all positive predictions. For class 1 (stroke), the precision is 0.00%, indicating that very few positive predictions were correct. This suggests that the model has a high rate of false positives for class 1. For class 0 (no stroke), the precision is not explicitly mentioned in the output, but it can be inferred that it's relatively high because the model's overall accuracy is high.

- Recall: Recall (also known as sensitivity or true positive rate) measures the proportion of true positive predictions among all actual positives. For class 1 (stroke), the recall is 0.00%, which means that the model failed to correctly identify most of the actual stroke cases. This suggests a high rate of false negatives for class 1. For class 0 (no stroke), the recall is not explicitly mentioned but is typically high due to the high accuracy.
- F1-Score: The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of a model's performance, especially when dealing with imbalanced datasets. For class 1 (stroke), the F1-score is 0.00%, reflecting the challenges in predicting this class accurately.
- Support: Support indicates the number of instances (samples) for each class in the test dataset. For class 1 (stroke), there are 54 instances, and for class 0 (no stroke), there are 943 instances.

Macro Average (macro avg): The macro average calculates the average of precision, recall, and F1-score across both classes (0 and 1). In this case, it indicates that the model's overall performance is relatively poor, with macro-average values around 0.47, 0.50, and 0.48 for precision, recall, and F1-score, respectively.

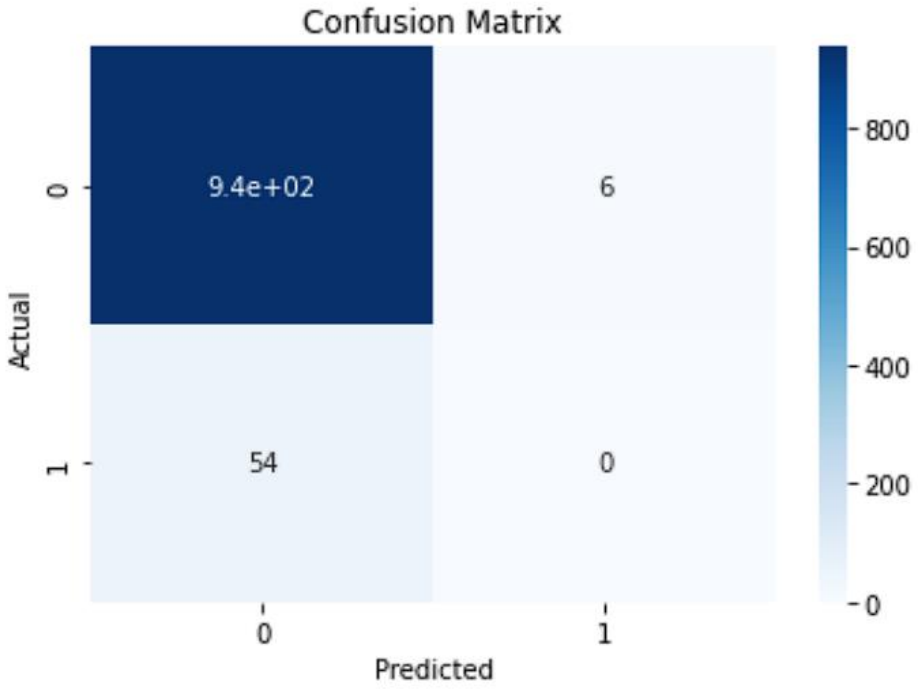
Weighted Average (weighted avg): The weighted average calculates a weighted average of precision, recall, and F1-score, considering the support for each class. It accounts for the class imbalance in the dataset. In this case, the weighted average values are higher than the macro-average values, with precision, recall, and F1-score around 0.89, 0.94, and 0.92, respectively. This suggests that the model performs better when accounting for class imbalances.

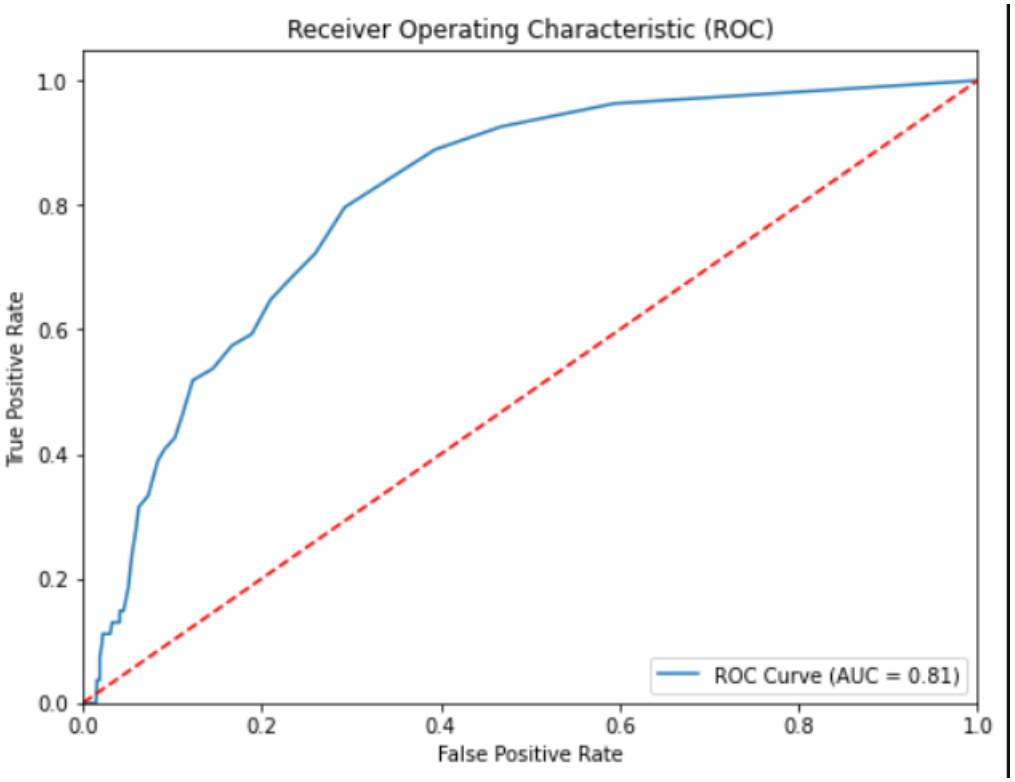
In summary, the output indicates that the Random Forest model achieved a high accuracy but performed poorly in predicting cases of stroke (class 1), as evidenced by low precision, recall, and F1-score for class 1. The model's performance is skewed due to class imbalance, which should be addressed to improve its ability to predict strokes accurately.

Results and Insights

Accuracy: 0.9398194583751254

	precision	recall	f1-score	support
0	0.95	0.99	0.97	943
1	0.00	0.00	0.00	54
accuracy			0.94	997
macro avg	0.47	0.50	0.48	997
weighted avg	0.89	0.94	0.92	997





Conclusion

In this project, we aimed to develop a machine learning model for brain stroke prediction using the Random Forest algorithm. We conducted extensive data preprocessing, model training, evaluation, and visualization to understand the model's performance.

The key findings and conclusions from this project are as follows:

1. Data Preprocessing: We successfully handled missing values and converted categorical variables into numerical form using one-hot encoding. Additionally, we split the dataset into training and testing sets, and we performed feature scaling to ensure the model's stability.

2. Model Selection: The Random Forest algorithm was chosen for this project due to its ability to handle non-linear relationships, feature importance analysis, and robustness to outliers and noisy data.

3. Model Performance: The trained Random Forest model demonstrated a high overall accuracy of approximately 93.98%. However, a deeper analysis revealed that the model's performance was skewed due to class imbalance. It excelled in predicting non-stroke cases (class 0) but struggled to predict stroke cases (class 1).

4. Class Imbalance: The dataset exhibited a significant class imbalance, with a small number of stroke cases (class 1) compared to non-stroke cases (class 0). This imbalance led to a high number of false negatives and low recall for stroke cases.

5. Future Improvements: To enhance the model's predictive capabilities for stroke cases, future improvements should include addressing class imbalance through techniques like resampling or using different evaluation metrics that account for imbalanced datasets. Additionally, feature engineering and more comprehensive data collection could further improve model performance.

6. Clinical Implications: Despite the model's limitations, the project underscores the potential of machine learning in assisting healthcare professionals in identifying individuals at risk of stroke. Further refinements and collaborations with medical experts are crucial to developing a more reliable predictive model for stroke prevention.

At last, this project serves as a foundation for leveraging machine learning in the field of healthcare for stroke prediction. While the Random Forest model achieved a high overall accuracy, addressing class imbalance and refining the model are essential steps toward its practical application in clinical settings. The project opens doors for further research and collaboration in the critical area of stroke prevention and early detection.