< Decision Analysis Project >

stroke classification model



20011813 Park jeongwon

20011796 Kim kyungmin

20011819 Kim sunjae

Table of Contents

1. Introduction

- A. Background

- B. Objectives

- C. Data analysis flow

- D. Version

2. Dataset Description

- A. feature description

3. Building a Classification Model

- A. Data preprocessing 1(before split)

- a. null

- b. outlier

- c. categorical data

- B. Data Split

- C. Basic model evaluation

- a. logistic regression

- D. Data preprocessing 2(after split)

- a. Sampling

- b. Feature Selection(info gain)

- c. Standardization

- E. Model Optimization

- a. MLP

- b. Logistic Regression

- c. Decision Tree

- d. SVC (SVM)

- e. Gaussian Naïve Bayes

- f. K-Neighbor

- g. Random Forest

- h. AdaBoost

- i. Gradient Boosting

- j. Histogram Gradient Boosting

- k. Bagging

- l. Extra Tree

- m. Stacking

- n. Voting

- o. CNN

- p. Hybrid CNN + Logistic Rregression

- F. Determination of the Optimal Model

- G. Evaluation of Optimal Model

4. Performance comparison between optimal model and additional methods

- A. Feature Selection

- a. Genetic Algorithm

- b. RFE

- c. Random Forest Important

- d. Variance Threshold

- e. Select K Best

- f. PCA

-g. Comparison between optimal and additional preprocessing models

- B. Additional ML model

- a. GAN

- b. GAN + CNN

- c. GAN + CNN + LSTM

5. Conclusion

6. Reflections and Limitations

1. Introduction

A. Background

Stroke, a leading cause of mortality and long-term disability on a global scale, poses significant challenges to public health worldwide. The early identification and prevention of stroke are paramount in minimizing its adverse impact. Machine learning, with its capacity to analyze extensive medical datasets, holds promise in predicting the probability of stroke occurrence. Such predictions enable timely interventions and the implementation of effective management strategies, thereby potentially reducing the burden of stroke-related morbidity and mortality.

B. Objective

This study endeavors to develop a sophisticated machine learning model tailored to accurately predict the likelihood of stroke based on patient data. To achieve this, the study will engage in comprehensive feature selection, carefully considering which patient characteristics are most relevant for accurate prediction. Furthermore, the study will meticulously select and evaluate appropriate machine learning algorithms, ensuring that the chosen models demonstrate high efficacy in stroke prediction tasks. Through these efforts, the study aims to contribute to the advancement of stroke prediction methodologies, ultimately enhancing early intervention and patient outcomes in stroke care.

C. Data analysis flow

We evaluated the model in as many ways as possible at every stage of data analysis and completed the model by selecting the model or preprocessing method that performs best.

In the preprocessing stage, standardization and feature selection were conducted through information gain, and the preprocessing method showing the best performance was selected

In the model selection stage, the best model was determined by selecting the model with the best performance by comparing the performance of the models that conducted grid search.

D. Version

In this study, experiments were conducted using the following software versions:

Scikit-Learn Version: 1.3.2

Python version: 3.8.5 (default, Sep 3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]

TensorFlow Version: 2.11.0

Note that experiments have been conducted in the above software versions and environments, and may not run smoothly in different versions or environments. Therefore, it is recommended to use the exact same software version to reproduce the same results.

2. Dataset Description

A. feature description

Let me explain the feature used in predicting stroke.

1. Gender: Sex of the Patient

- Male

- Female

2. Age : Age of the patient in Years

3. Hypertension

- 0 = if the patient doesn't have hypertension

- 1 = if the patient has hypertension

4. Heart\_disease

- 0 = if the patient doesn't have any heart diseases

- 1 = if the patient has a heart disease

5. ever\_married: whether the patient is married or not

- No

- Yes

6. work\_type: the type of employment of a patient

- children

- Govt\_jov

- Never\_worked

- Private

- Self-employed

7. Residence\_type:

- Rural

- Urban

8. avg\_glucose\_level: average glucose level in blood

9. bmi: body mass index ((weight) / (height)²)

10. smoking\_status

- never smoked

- formerly smoked

- smokes

- Unknown

11. stroke [Target data]

- 0 = if the patient had not a stroke

- 1 = if the patient had a stroke

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

3. Building a Classification Model

Data pre-processing was divided into pre-processing before data split and post-processing after data split. Pre-processing before data split was performed [removing Null data, removing outliers, and encoding categorical data].

Through this process, normal data were allowed to enter the training set and test set.

Preprocessing after data segmentation was performed by [imbalanced data sampling, feature selection, standardization, PCA], etc., increasing the effectiveness of the data.

- A. Data preprocessing 1(before split)

- a. null

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

It was found that only the bmi feature had null values, so the null values were treated by replacing them with the mean of bmi.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

- b. outlier

The Isolation Forest model was used to remove outliers in integer features.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 도표, 평행, 라인이(가) 표시된 사진

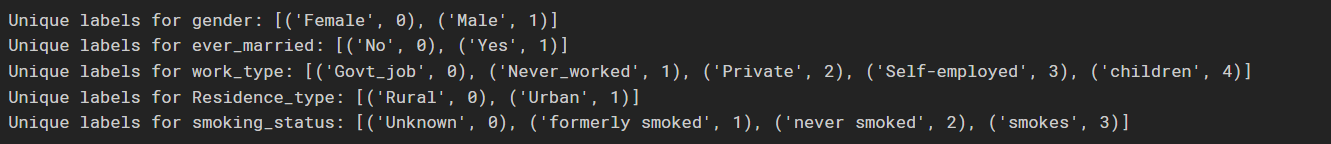
자동 생성된 설명There were 5110 rows before outlier removal, but 1379 rows were removed through outlier removal.

- c. categorical data

Categorical data needs to be encoded into numerical format for use in classification models. Therefore, We performed encoding to convert the categorical data into integer format. There are various encoding methods available, such as one-hot encoding and label encoding. However, We chose label encoding over one-hot encoding because one-hot encoding significantly increases the number of columns, leading to longer execution times and unclear interpretations when feeding the data into classification models.

Label encoding assigns a unique integer to each label, which can introduce the disadvantage of weighted learning based on the size of the labels. However, since the maximum label size is 4 in this case, the impact of weighted learning is minimal. Hence, we proceeded with label encoding.

The categorical features ['gender', 'ever\_married', 'work\_type', 'Residence\_type', 'smoking\_status'] were label encoded as they contain categorical data.



스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

- B. Data Split(Train, Test)

Since we will evaluate performance through cross-validation (k-fold), we divided the data into only train/test splits.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

- C. Basic model evaluation

- a. logistic regression

Before comparing various data preprocessing methods and models, we calculated the metrics for the current dataset using the most basic model, logistic regression. During the data preprocessing stage, we will use the metrics of the basic model as a reference point and select only those preprocessing methods that improve upon these metrics.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 직사각형, 도표이(가) 표시된 사진

자동 생성된 설명

We have not yet completed the preprocessing, and since we have not optimized the parameters of the logistic regression model, it classified all data as (stroke=0). Therefore, we consider these metrics to be meaningless and proceed with preprocessing and model optimization.

- D. Data preprocessing 2(after split)

Before proceeding with data preprocessing, let's examine the distribution of the X and y data.

원, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 도표, 그래프, 라인이(가) 표시된 사진

자동 생성된 설명

The y data is binary, with an overwhelming number of 0s. Additionally, the ranges of values in the X data vary significantly, indicating a need for standardization. Besides standardization, we will also perform feature selection to create an optimal dataset.

- a. Sampling

We'll start by addressing the bias issue in the y data. To mitigate the bias problem, we employed a sampling technique. There are two main sampling techniques: OverSampling and UnderSampling. For this project, we utilized OverSampling to address the bias issue in the y data.

OverSampling is a technique used to balance the dataset by increasing the number of minority class instances in an imbalanced dataset. It involves creating synthetic instances of the minority class or duplicating instances until the dataset is balanced. This technique helps improve the performance of machine learning models by ensuring that the minority class is adequately represented in the training data.

텍스트, 스크린샷, 도표, 원이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Before sampling, both Recall and F1 score were 0, but after sampling, these metrics increased significantly. Although accuracy slightly decreased, we considered the post-sampling dataset to be superior. Therefore, we decided to proceed with the oversampled data.

- b. Feature Selection(info gain)

We created a dataset by extracting the top 5 columns based on information gain and evaluated its performance.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Top 5 features with highest average information gain:

['avg\_glucose\_level', 'bmi', 'age', hypertension', 'heart\_disease']

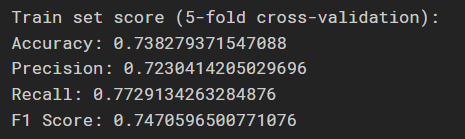
텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

The average metrics (accuracy, precision, recall, F1 score) for the dataset before feature selection were 0.745323467113913. After feature selection, the average metrics improved slightly to 0.745387210768993. Additionally, feature selection reduced the model execution times. Therefore, we chose the dataset after feature selection. (Preprocessing steps so far: OverSampling + Feature Selection)

- c. Standardization

Since the range of the X data varies significantly across features, there is a high likelihood of introducing errors in interpreting the data. Therefore, I will standardize the data to align the scales of all features, ensuring that each feature's weight is equal, and then train the model.



Even though standardization did not result in differences in evaluation metrics, I will still proceed with the standardized data to enhance interpretability. (Preprocessing steps so far: OverSampling + Feature Selection + Standardization)

Final preprocessing steps: Over Sampling + Feature Selection + Standardization.

- E. Model Optimization

Each model was trained using a training dataset and evaluated using a test dataset. Performance evaluations were based on ROC curves and AUC (Area Under the Curve) scores.

Since the training set's performance may not necessarily be a good model, we used the test set to prevent overfitting and evaluate generalization performance. Overfitting refers to a phenomenon in which the model is too suited to the training data, resulting in poor predictive performance on new data

- a. MLP

MLP Model and Grid Search Overview

This section details the process of training data using multilayer perceptron (MLP) models and finding optimal hyperparameters through grid search. MLP is a neural network based model that is powerful enough to learn complex nonlinear relationships well. Grid search is a method to systematically explore different combinations of hyperparameters to find optimal model settings.

Performing grid search

GridSearchCV is used to find the optimal hyperparameters through grid search. Evaluate the optimal model performance. After finding the optimal hyperparameters through grid search, the optimal model is used to evaluate the performance.

텍스트, 스크린샷, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

- b. Logistic Regression

Logistic Regression Model and Grid Search Overview

This section details the process of training data using a logistic regression model and finding the optimal hyperparameters through grid search. Logistic regression is a widely used linear model in binary classification problems, which is very useful for predicting class probabilities. Grid search is a method of systematically exploring different combinations of hyperparameters to find the optimal model settings.

라인, 스크린샷, 그래프, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- c. Decision Tree

The decision tree is a nonparametric supervised learning model used for data classification and regression analysis. The model forms a tree that makes decisions based on the features of the data. Here are the main points about the decision tree:

Structure:

Node: A point at which a segmentation is determined based on a particular feature of the data.

Root Node: The top node in the tree, determines the first partition from the overall data.

Internal Node: An intermediate node that continues to be split.

Terminal Node (Leaf Node): A final node that is no longer divided, where it has a final prediction value.

Branch: Indicates the connection between nodes and branches according to the conditions of the particular feature.

스크린샷, 라인, 그래프, 텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- d. SVC (SVM)

SVM is a method of mapping given data points into a high-dimensional space and finding the optimal boundary line between two classes in that space. In this study, an SVM model was implemented using the SVC class of the Scikit-learn library. The main hyperparameters are C, gamma, and kernel, and GridSearchCV was used to find the optimal value for them.

텍스트, 라인, 도표, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- e. Gaussian Naïve Bayes

Gaussian Naive Bayes is a Naive Bayes model that uses Gaussian distributions to handle continuous data. Each feature in the given data independently follows a Gaussian distribution, which is used to compute class probabilities. In this work, a model was implemented using the GaussianNB class in the Scikit-learn library. The main hyperparameters are var\_smoothing, which is used to increase the stability of variance estimation.

텍스트, 라인, 도표, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- f. K-Neighbor

The K-Nearest Neighbors (KNN) algorithm is a simple yet effective method used for supervised learning classification and regression problems. When classifying or predicting new data points, KNN refers to the nearest K data points (neighbors).

스크린샷, 라인, 그래프, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- g. Random Forest

Random forests are methods to enhance prediction performance by ensembleing multiple decision trees. Each tree is trained independently, and the final prediction is a combination of the predictions from these trees by means (in the case of regression) or majority vote (in the case of classification). Random forests are advantageous for preventing overfitting and capturing different patterns of data.

텍스트, 라인, 도표, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- h. AdaBoost

Adaptive Boosting (AdaBoosting) is an algorithm that creates a Strong Learner by sequentially learning several weak Learners. At each stage, the data points that were incorrectly predicted at the previous stage are weighted and reflected in the next learner. This progressively improves predictive performance.

라인, 그래프, 도표, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- i. Gradient Boosting

Gradient Boosting is a method to train several weak learners sequentially to reduce Residual Errors. At each step, a new model is trained to correct the errors of the previous model. This process is performed similarly to Gradient Descent, and the final model is created by combining the predictions of all weak learners..

라인, 도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- j. Histogram Gradient Boosting

Histogram Gradient Boosting is a variant of Gradient Boosting, which splits continuous variables into histogram bins to speed up learning, reducing memory usage and allowing them to work efficiently on large datasets.

라인, 도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

‘

- k. Bagging

Bagging is a method of randomly generating multiple training data samples to train a model for each sample, and combining the predictions of these models. Bagging is effective in reducing the variance of the model and improving the prediction performance. Random forests are a typical example of bagging

라인, 도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- l. Extra Tree

The Extra Tree (Extremely Randomized Trees) algorithm is similar to random forests, but introduces more randomness when learning each tree. By constructing a tree by randomly selecting segmentation criteria, it reduces the model's bias and speeds up the learning

. 텍스트, 라인, 스크린샷, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- m. Stacking

Stack is a method of combining the prediction results of several different models through a meta-model. We train a meta-model using the prediction results of each model as a new feature. This allows us to combine the advantages of different models.

라인, 도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

- n. Voting

Voting is a method of combining prediction results from multiple models to make final predictions. Majority Voting is used for classification problems, and Averaging is used for regression problems. Voting is more stable than a single model and generally provides higher prediction performance.

텍스트, 라인, 도표, 그래프이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명

-o. CNN

CNN (Convolutional Neural Networks)  
Convolutional Neural Networks (CNNs) are deep learning models designed to handle spatial structures of data, such as image processing. Here are some of the key things about CNNs:  
Structure:  
Convolutional Layer: Extracts regional features of an image. Filters (or kernels) perform convolutional operations while sliding over the image.  
Pooling Layer: Shrink the dimensions of the data and increase computational efficiency, usually with Max Pooling or Average Pooling.  
Fully Connected Layer: Performs final classification based on extracted features.

라인, 스크린샷, 그래프, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명

-p. hybrid CNN + Logistic Regression

라인, 도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- F. Determination of the Optimal Model

We analyze the four key evaluation metrics used to evaluate the performance of both machine learning and deep learning models: Accuracy, Precision, Recall, and F1 Score. A bar graph was created to visually compare the average performance scores of each model.

다채로움, 스크린샷, 그래픽, 직사각형이(가) 표시된 사진

자동 생성된 설명

In this analysis, the strength of each model was identified by comparing the performance of various models. In particular, Random Forest and Histogram-based Gradient Boosting models showed excellent overall performance. This suggests that these models have strong generalization capabilities for various data distributions.

Visualize the loss values for each model (16 in total).

Use plt.bar to represent the bar graph.

The x-axis is the model number (1-16), and the y-axis is the Loss value.

스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

Visualize the ROC AUC value for each model.

Similarly, use plt.bar to graph the bar.

The x-axis is the model number (1-16), and the y-axis is the ROCAUC value.

스크린샷, 그래픽, 그래픽 디자인, 디자인이(가) 표시된 사진

자동 생성된 설명

Weigh each performance indicator to calculate the weighted sum score.

In this example, we assigned 20% accuracy, 10% precision, 30% recall, 30% F1 score, and 10% ROCAUC. Since the task involves predicting the presence of a stroke, We have determined that a classification model with higher Recall is preferable over one with higher Precision. Consequently, We have decided on the following weights for each metric. Finds the index for the model with the highest weighted sum score.

Use that index to get the best model name from the models list.

Outputs the name of the optimal model.



- G. Evaluation of Optimal Model

In important medical prediction models such as stroke, Recall is very important to protect the patient's life. This refers to the ability to find all of the actual positive cases without missing them. However, after evaluating the final model, the test set showed that TPR was 1/32 and FPR was 7/714. We prioritized recall in the model selection process, but the model focused on lowering FPR was selected as the final model. This is somewhat regrettable in that we did not select a model that could identify more real positive cases. For stroke prediction, since it is important to reduce false negatives rather than false positives, there is a lot of regret about the final model selection in that it is more desirable to choose a model with high recall.

텍스트, 스크린샷, 직사각형, 도표이(가) 표시된 사진

자동 생성된 설명스크린샷, 직사각형, 사각형, 그래픽이(가) 표시된 사진

자동 생성된 설명

A comprehensive evaluation of the final model's performance showed relatively high accuracy, precision, recall, and F1 scores on the test dataset. The confusion matrix and classification reports allowed us to see how the model performs in each class. This confirmed that the final model provides stable and reliable performance on the test dataset.

4. Performance comparison between optimal model and additional methods

- A. Feature Selection

Based on the optimal model derived from comparing various models, I will evaluate the performance of additional preprocessing methods by comparing them with the preprocessing methods used previously.

- a. Genetic Algorithm

Genetic Algorithm (GA) is an optimization algorithm that mimics the principles of natural selection and genetics, used to find the optimal solution for complex problems. GA generates an initial population randomly and then evaluates the fitness of each individual. Individuals with higher fitness are selected to undergo crossover and mutation to create new offspring, and this process is repeated to explore the optimal solution. GA is advantageous in finding global optima and can be applied to a variety of problems, including feature selection.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- b. RFE

Recursive Feature Elimination (RFE) is a feature selection technique used in machine learning. It begins by training a model with all available features, then evaluates the importance of each feature and iteratively removes the least significant ones, aiming to select the optimal subset of features. This iterative process helps in reducing the complexity of the model, improving its performance, preventing overfitting, and enhancing interpretability. RFE is model-dependent but effective when used with appropriate models in various machine learning tasks.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- c. Random Forest Important

Random Forest is a powerful algorithm used in machine learning that operates by ensembling multiple decision trees. Each decision tree is trained on a randomly selected subset of features and data samples. During this process, it computes the importance of each feature. Feature importance is evaluated based on how much each feature reduces impurity in predictions within each tree and is averaged across all trees. This allows us to understand the contribution of each feature, prioritize them based on their importance, and perform feature selection accordingly.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- d. Variance Threshold

Variance Threshold is a technique used in feature selection where features are filtered based on their variance (or the square of the standard deviation). This method removes features whose variance falls below a certain threshold. Essentially, features with low variance are likely to contain less useful information for model training because they exhibit little variation in the dataset. Therefore, they are removed to reduce noise and potentially improve the model's performance, especially in high-dimensional datasets.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

- e. Select K Best

SelectKBest is a feature selection method that selects the top K most important features from a given set of features. It typically uses statistical methods to evaluate the importance of each feature and selects those with the highest scores. Common statistical methods used include ANOVA (Analysis of Variance) and chi-squared tests. In this case, ANOVA was used to conduct feature selection. This approach helps reduce the complexity of models, improve prediction performance, and enhance interpretability by focusing on the most relevant features for the task at hand.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

-f. PCA

Principal Component Analysis (PCA) is a technique used to reduce the dimensionality of a high-dimensional dataset to a lower-dimensional space. It aims to minimize the loss of information while preserving the essential features of the data. We will use PCA to compare whether each dimension shows better performance compared to the original dataset.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

-g. Comparison between optimal and additional preprocessing models

I compared the performance of the existing preprocessing method (oversampling + top 5 attributes based on information gain + standardization) with additional preprocessing methods. Among them, the performance of data processed using Genetic Algorithm stood out as the best. It showed comparable performance to the optimal model, with minimal difference.

. 스크린샷, 그래픽, 그래픽 디자인, 다채로움이(가) 표시된 사진

자동 생성된 설명

- B. Additional ML model

-a. GAN

Generative Adversarial Networks (GANs) are models in which two neural networks, the generator and the discriminator, compete with each other to learn. The generator generates fake data, and the discriminator tries to distinguish whether this data is real or fake. The generator learns to produce more and more authentic data, and the discriminator learns to distinguish it more accurately. This is how the two networks compete and develop at the same time. As a result, the generator can generate very realistic data. GAN is utilized in various fields, such as image generation, video synthesis, and data augmentation.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

. -b. Hybrid CNN +GAN

This code builds a Generative Adversarial Network (GAN) model and evaluates its performance by cross-validation. We first define a generator and a CNN-based discriminator, and combine the two models to create a GAN model. We wrap the GAN model with KerasClassifier and evaluate accuracy, precision, recall, and F1 score using Scikit-Learn's cross\_validate. Finally, we output an average performance metric.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

-c. Hybrid CNN + LSTM + MLP

Long Short-Term Memory (LSTM) is a variant of Recurrent Neural Networks (RNNs) designed to handle long-term dependencies in time series data or sequential data. Here are the main points about LSTMs:

Structure:

Cell State: Remember the long-term dependence of time series data.

Input Gate: Determines how much the current input will be reflected in the cell state.

Forget Gate: Determines how much information from the previous cell state is to be forgotten.

Output Gate: Generates an output value based on the current cell state.

This code combines the CNN, LSTM, and MLP models to construct a hybrid model. The build\_hybrid\_model function combines these three models to form a final output layer. Then, we wrap the hybrid model with KerasClassifier, and evaluate its performance with cross-validation. The cross-validation results are summarized as the average of accuracy, precision, recall, and F1 scores. Finally, the average value of each performance metric is output.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

-d. Comparison between optimal and additional ML/DL models

The performance of the final model is better than the additional three models

5. Conclusion

In the process of determining the optimal model, We deliberated extensively on which evaluation metrics to prioritize. For disease prediction, We considered models with high TPR to be better performers, and We used this criterion to select the optimal model. Additionally, We experimented with various models and preprocessing methods to find the best fit. Given that the dataset We prepared was not complex, machine learning models outperformed more complex models like deep learning.

6. Reflections and Limitations

**Necessity of Various Models**:

Analyzing stroke data with 16 different models highlighted the importance of leveraging a diverse range of models. Each model has unique strengths and weaknesses, and some performed better than others on this specific dataset. This experience reinforced the value of experimenting with multiple models.

**Importance of Model Selection and Tuning**:

The performance of each model significantly depended on hyperparameter tuning. Utilizing methods such as Grid Search and Random Search to find optimal hyperparameters was essential and significantly improved model performance.

**Significance of Data Preprocessing**:

This experience underscored how crucial data preprocessing is to model performance. Proper handling of missing values, scaling, and encoding had a profound impact on the accuracy and generalization capability of the models.

**Diversity of Evaluation Metrics**:

Evaluating model performance using only accuracy is insufficient. It was important to also consider precision, recall, F1 score, and other metrics. Especially in medical data, certain metrics may be more critical, and choosing the right ones for the context is necessary.

Points of Regret

Despite recognizing the 'unknown' values in the smoke column as missing data, I was unable to handle them effectively. Instead of simply ignoring or dropping these values, more appropriate methods could have been employed. For instance, imputing the missing values using predictive models or substituting them based on patterns in other data could have been explored. Missing values can contain crucial information, especially in medical data, making it important to handle them properly to enhance the accuracy and reliability of the model. In the future, I need to be better prepared with a variety of methods for handling missing data and apply these techniques to ensure the completeness and integrity of the dataset.