International Journal of Engineering Research and Development

DOI: 10.29137/ijerad.1432162

Volume:17 Issue:1 Pages:11-27 March/2025

Research Article

Comparison of Machine Learning and Deep Learning Techniques for Stroke Prediction

Süphan YAKUT[1*], Necaattin BARIŞÇI[2]

[1] *Gazi University, Faculty of Technology, Department of Computer Engineering, 06560 Ankara, Türkiye*
[2] *Gazi University, Faculty of Technology, Department of Computer Engineering, 06560 Ankara, Türkiye*

***Received**: 09/02/2025. **Accepted**:29/09/2024 **Published Online:** 15/03/2025*
***Final Version**: 01/03/2025*

Abstract

Early diagnosis of diseases is crucial in the healthcare field, and early detection of stroke significantly contributes to the success of the treatment process. This study aims to identify the most accurate and effective model for early stroke diagnosis. While the literature primarily focuses on machine learning methods for early stroke detection, this research combines machine learning techniques with deep learning and hybrid models to achieve better results and compare them. The study employs machine learning methods such as Logistic Regression, Decision Trees, Random Forest, and Support Vector Machines, and deep learning methods including Convolutional Neural Networks, Long Short-Term Memory, and Bidirectional Long Short-Term Memory algorithms. Hybrid models integrating machine learning and deep learning algorithms are also explored. Analyses are performed on a dataset including clinical parameters like age, gender, hypertension, and heart disease. The highest accuracy rate is achieved with Logistic Regression following the Least Absolute Shrinkage and Selection Operator (LASSO) method, (96% accuracy) Support Vector Machines following the LASSO method (96% accuracy), and Ensemble Learning of Machine Learning Algorithms (Logistic Regression, Decision Trees, Random Forest, Support Vector Machines) (96% accuracy).

Keywords

"Health, Stroke, Machine learning, Deep learning, Data analysis"

**** Corresponding Author:** yakutsuphan@gmail.com*

1. Introduction

Humanity continuously seeks ways to improve health and quality of life, and advancements in medical diagnosis and treatment methods are of vital importance. In this context, the early diagnosis and effective management of neurological disorders, such as stroke, are among the greatest challenges of modern medicine. Stroke is a prevalent health issue worldwide that can seriously affect individuals' quality of life and even pose a life-threatening risk (World Health Organization, 2022).

Stroke is a widespread health problem globally, with approximately 13.7 million new stroke cases reported annually. According to the World Health Organization (WHO), stroke is the leading cause of death and disability worldwide, with around 5.5 million people dying each year due to stroke. Additionally, one in four people is at risk of experiencing a stroke during their lifetime. Stroke not only poses a risk of death but can also lead to long-term and serious health consequences for survivors (World Health Organization, 2022).

This study aims to identify stroke risk more effectively using advanced algorithms and data analysis techniques and to enable early intervention in the disease's early stages.
One of the studies in the literature, conducted by Singh et al. (2017), examined the use of artificial intelligence methods in stroke prediction by performing an in-depth analysis of the Cardiovascular Health Study dataset. The study reported that artificial neural networks were able to predict stroke with an accuracy rate of 97.7%, demonstrating the potential of artificial intelligence in medical diagnosis and prognostic predictions (Singh et al., 2017).

In a study by Lee et al. (2020), machine learning techniques were used to detect the onset of stroke symptoms within 4.5 hours in stroke patients, and the performance of the model was evaluated. The machine learning algorithms used in the study included Decision Trees, Support Vector Machines, and Random Forest models. The results showed that machine learning algorithms provided higher accuracy compared to human readers and improved the ability to detect symptoms early. The best results achieved were 92.4% accuracy and 90.7% sensitivity in detecting stroke symptoms (Lee et al., 2020).

Emon et al. (2020) studied machine learning methods such as Random Forest, Decision Trees, Naive Bayes, and Support Vector Machines (SVM) for stroke diagnosis and early detection. The study used the dataset obtained from the UCI Machine Learning Repository. It emphasized the potential of these techniques in stroke diagnosis and evaluated the performance of each method in detail. The results showed the effectiveness of machine learning methods in stroke diagnosis and their accuracy rates (Emon et al., 2020).

In Sevli's (2021) study, the Stroke Prediction 2021 dataset and the machine learning technique of Random Forest were used to achieve an accuracy rate of 98.84% for early detection of stroke risk. The study highlighted the importance of factors such as age, body mass index, and average blood glucose levels in determining stroke risk (Sevli, 2021).

In a study by Oğuz et al. (2021), 13 different machine learning methods were used to determine stroke risk, and a comparative analysis was conducted. The dataset used (Stroke Prediction 2021) was based on risk factors such as age, gender, smoking, and hypertension related to stroke. In this study, algorithms like Logistic Regression (LR), Classification and Regression Trees (CART), Random Forest Classifier (RFC), Support Vector Machines (SVM), and k-Nearest Neighbors (KNN) were tested. The results showed that Random Forest Classifier (RFC) was the most successful model with an accuracy rate of 99.425% (Oğuz et al., 2021).

In the study by K. Pallavi and Saravananthirunavakarasu (2022), data from 507 patients' case sheets from Kumbakonam Sugam Multispecialty Hospital in Tamil Nadu, India, were used to classify stroke disease with various machine learning algorithms, including artificial neural networks, support vector machines, bagging, and boosting methods. The study found that artificial neural networks showed the highest success with a 95% accuracy rate (Pallavi & Saravananthirunavakarasu, 2022).

In the study by Srinivas, A., & Mosiganti (2023), a ensemble machine learning model for stroke diagnosis and prediction was proposed, and the model demonstrated a high accuracy rate of 96.88% (Srinivas & Mosiganti, 2023).

In Kuksal et al.'s (2023) study, the applicability of the Extreme Gradient Boosting Classifier on the Stroke Prediction 2021 dataset for stroke detection and prediction was explored, and successful results were obtained with high accuracy, precision, and F-measure metrics. The highest accuracy, 96%, was achieved using the XGBoost technique (Kuksal et al., 2023).

In Chandra et al.'s (2023) study, a machine learning model using Random Forest, k-Nearest Neighbors, and Logistic Regression was developed for stroke risk prevention and early treatment with the Stroke Prediction 2021 dataset. The model demonstrated that it could provide effective healthcare services for stroke detection, and the study performed a performance comparison among various machine learning models, identifying the beneficial features. The best result was obtained with the Decision Tree model (DVM) at an accuracy rate of 93.93% (Chandra et al., 2023).

In the literature, the use of machine learning techniques in stroke diagnosis and early detection has proven to be significant. Various machine learning methods, such as Random Forest, Decision Trees, Naive Bayes, and Support Vector Machines, have yielded

successful results in stroke diagnosis and detection. These techniques have shown their potential in solving complex problems in the healthcare field.

This study goes beyond machine learning techniques by employing deep learning techniques and a ensemble learning method to create hybrid models for stroke prediction. The results obtained from these models are compared. The machine learning algorithms examined include Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), and Decision Trees (DT). On the deep learning side, 1D Convolutional Neural Networks (CNN 1-D), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (Bi-LSTM) algorithms are used. Additionally, hybrid models obtained from the mentioned four machine learning models and three separate deep learning models are created and compared. This approach aims to perform a more comprehensive performance analysis in stroke prediction, distinguishing it from studies in the literature.

The results of this study provide valuable insights that can guide healthcare professionals and clinical applications in developing stroke prevention and early intervention strategies. In the following sections, the second section presents the materials and methods, the third section covers the research findings, and the fourth and final section provides the conclusions.

2. Materials and Methods

In this study, four machine learning and three deep learning methods were examined to enable risk assessment in the early stages of stroke diagnosis. The machine learning methods used were Logistic Regression (LR), Random Forest (RF), Decision Trees (DT), and Support Vector Machines (SVM). In the context of deep learning techniques, Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (Bi-LSTM) models were explored. The results provide a detailed comparison to identify the most effective model for stroke diagnosis and risk assessment.

2.1 Hardware and Software Specifications Used

2.1.1 Hardware specifications used

The model training was conducted on a device running Windows 11 Pro 64-bit operating system. The device is equipped with an 11th-generation Intel(R) Core(TM) i7-1165G7 processor, with 8 cores and a base clock speed of 2.80 GHz, and an Intel Iris Xe Graphics card. Additionally, it has 32 GB of RAM (32768 MB), which has been sufficient for efficiently completing the training process. Since no visual data was used, no additional resources were required. The current hardware capacity of the system successfully supported the model training process.

2.1.2 Software specifications used

The project was developed using the Python (3.11.5) programming language. Python libraries such as NumPy, Pandas for data processing, and Matplotlib.pyplot and Seaborn for graphical analysis and plotting were utilized. During the development process, Visual Studio Code (version 1.92) was predominantly used, with occasional use of Jupyter Notebook. These tools provided a flexible and efficient development environment at various stages of the project.

2.2 Dataset

The dataset used in the study is called the **Stroke Prediction Dataset (2021)** and was obtained from the Kaggle platform (Federisino, 2021).

2.2.1 Content of the Dataset and Classification Features:

The dataset contains data from 5510 individuals and includes the features shown in Table 1. As an example, the features of the first five individuals in the dataset are presented in Table 2.

Table 1. Dataset Features Table

1. **ID**: A unique identifier for each individual.
2. **Gender**: The gender of the individual ("Male", "Female", "Other").
3. **Age**: The age of the individual.
4. **Hypertension**: The individual's hypertension status (0 = no, 1 = yes).
5. **Heart Disease**: The individual's heart disease status (0 = no, 1 = yes).
6. **Marital Status**: Whether the individual is married ("No", "Yes").
7. **Job Type**: The type of job the individual works in ("Children", "Govt_job", "Never worked", "Private", "Self-employed").
8. **Residence Type**: The type of area where the individual lives ("Rural", "Urban").
9. **Average Glucose Level**: The individual's average glucose level in the blood.
10. **Body Mass Index (BMI)**: The individual's body mass index.
11. **Smoking Status**: The individual's smoking status ("Previously smoked", "Never smoked", "Smokes").
12. **Stroke**: Whether the individual has had a stroke (0 = no, 1 = yes).

Table 2. Data for the First Five Individuals in the Dataset

	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

2.2.2 Classification in the Dataset

In the dataset, classification is based on the "Stroke" feature. This classification indicates whether individuals have had a stroke or not, and it consists of two classes: No stroke (0) and Stroke (1).

2.2.3 Missing data and measures taken

The dataset contains missing values for the "Body Mass Index" (BMI) feature. To handle the missing data, the method of filling the missing values with the mean has been used. This approach has been applied to complete the missing data and include it in the analysis process. The steps taken will be detailed under the methodology section in the following parts.

2.3 Methodology

The study began with exploratory data analysis. First, the dataset was loaded, and then the data types of the variables in the dataset were determined. Subsequently, missing value analysis, normalization, and outlier analysis were conducted. These steps were performed to understand the overall structure of the dataset and to train the models more accurately.

2.3.1 Missing value analysis

As shown in Figure 1, missing value analysis revealed that there were missing values for 201 individuals in the 'bmi' column. These missing values were filled by calculating the average bmi value of the other individuals in the dataset, thus resolving this issue in the dataset.

id	0	id	0
gender	0	gender	0
age	0	age	0
hypertension	0	hypertension	0
heart_disease	0	heart_disease	0
ever_married	0	ever_married	0
work_type	0	work_type	0
Residence_type	0	Residence_type	0
avg_glucose_level	0	avg_glucose_level	0
bmi	201	bmi	0
smoking_status	0	smoking_status	0

Figure 1. Number of Missing Values Before and After Missing Value Analysis

2.3.2 Normalization process

Different variables in datasets often have different scales. For instance, one variable may take values between 0 and 1000, while another could range between 0 and 1. This discrepancy can lead to serious issues, particularly for machine learning algorithms. Data with varying scales can create bias during the learning process, causing some features to carry more weight than others. To eliminate this problem and ensure that all features are treated equally by the model, a normalization process is applied (Jain et al., 2000). In this study, MinMaxScaler was used to bring the numerical features of the dataset to the same scale. MinMaxScaler scales each feature to a range between 0 and 1, based on its minimum and maximum values. This ensures that all numerical variables in the dataset are within the same range, allowing the model to work more effectively and evenly on the data. Some data examples scaled to the 0-1 range using MinMaxScaler are shown in Figure 2.

```
array([[0.81689453, 0.80126489, 0.30126002],
       [0.74365234, 0.67902317, 0.21298095],
       [0.97558594, 0.23451205, 0.25429553],
       ...,
       [0.42626953, 0.12865848, 0.2325315 ],
       [0.62158203, 0.51320284, 0.17525773],
       [0.53613281, 0.13922999, 0.18213058]])
```

Figure 2. Example Data Scaled Between 0 and 1

This process helps eliminate potential issues arising from the dataset having extremely large or small values, contributing to the model producing more consistent and stable results. Therefore, normalization was applied in this study to ensure that the model generates more accurate and reliable outcomes.

2.3.3 Outlier analysis

Outliers are data points that are significantly different from other observations in a dataset. These values may arise due to data collection, measurement errors, extreme conditions, or unusual situations in the data structure. Outliers can negatively impact the predictive performance of a model, making outlier analysis an important step in the data processing process. Detecting and appropriately handling outliers contributes to the model producing more accurate and reliable results (Iglewicz & Hoaglin, 1993).

When performing outlier analysis, the Interquartile Range (IQR) method is commonly used. The IQR represents the difference between the first quartile (Q1) and the third quartile (Q3), and it measures the central spread of the dataset. Using IQR, outliers are defined as values below Q1 - 1.5 * IQR and above Q3 + 1.5 * IQR. Observations outside these values are considered outliers, and the method for handling these outliers in the analysis is determined (Iglewicz & Hoaglin, 1993).

Before training the models, outlier analysis was performed on the dataset, and these outliers were adjusted to fit the dataset. Various methods were used to handle the outliers, including removing outliers and replacing them with the mean value. As will be explained and discussed in detail in Section 3, different approaches to handling outliers were explored to optimize the model's accuracy. The highest accuracy was achieved when outliers were removed from the dataset.

After the outliers were removed using the IQR method, the dataset size was reduced from 5110 to 4748 data points. This step helped reduce the noise in the dataset, leading to improved model performance.

2.4 Machine Learning Techniques

2.4.1 Logistic regression

Logistic regression is a statistical modeling technique commonly used in binary or multi-class classification problems. Its main purpose is to model how independent variables (features) affect an outcome, and to predict the probability of an event occurring. It is a preferred method in various fields such as medicine, social sciences, and machine learning (Hosmer et al., 2013).

The mathematical expression of the logistic regression model is based on a logit function, known as the sigmoid function. As shown in Figure 3, the sigmoid function transforms any real number into a probability value between 0 and 1. This property allows the output of the model to be interpreted as a probability. The model calculates the z-score, which is a linear combination of the independent variables, and then inputs this z-score into the sigmoid function to obtain the predicted probability (Hosmer et al., 2013).

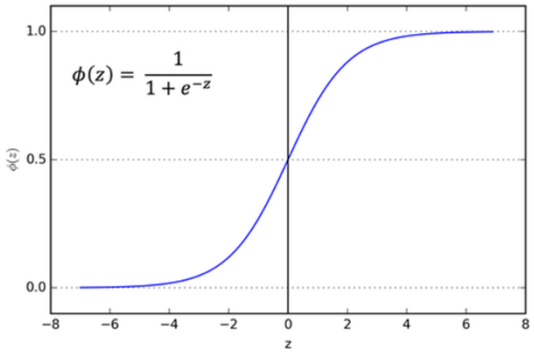

Figure 3. Sigmoid Function

In the study, the following parameters were used when applying LR:

- **solver='lbfgs'**: The 'lbfgs' optimization algorithm is used.
- **max_iter=100**: The maximum number of iterations is set to 100.
- **multi_class='auto'**: Based on the number of classes, 'ovr' or 'multinomial' is automatically selected.
- **penalty='l2'**: The L2 norm is used as the penalty.
- **C=1.0**: The inverse of the regularization strength, C, is set to 1.0.

2.4.2 Decision trees

The decision tree algorithm is a classification method used in data mining, and its basic principles are based on information theory. This algorithm helps users handle complex datasets by automatically creating interpretable models. Decision trees are artificial intelligence models used to understand and classify relationships within a dataset. As shown in Figure 4, the root node represents the starting point of the algorithm, while the branches and leaf nodes represent decision tests and classification outcomes based on the values of the variables in the dataset. With this structure, decision tree algorithms analyze datasets using the principles of information theory and generate understandable, interpretable models that provide valuable insights to users (Breiman et al., 1986).

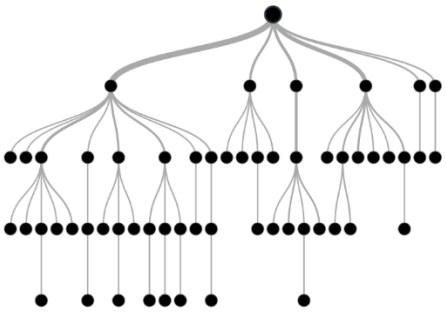

Figure 4. Visualization of Decision Tree

In the study, the following parameters were used when applying DC:

- **criterion='gini'**: The Gini impurity is used as the criterion for branching the tree.
- **splitter='best'**: The best split is chosen at each node.
- **max_depth=None**: There is no limitation on the depth of the tree; nodes are expanded until all leaves are pure or until all leaves contain fewer examples than min_samples_split.
- **min_samples_split=2**: The minimum number of samples required to split an internal node.

2.4.3 Random forest

Random Forest is a powerful ensemble learning algorithm that is highly effective in solving both classification and regression problems by leveraging the advantages of decision trees. This method creates a single model by combining the predictions of many decision trees. As shown in Figure 5, each decision tree is trained on different subsets of the dataset, and the predictions of these trees are combined to produce the final prediction. The core of the Random Forest algorithm lies in generating a large number of decision trees using randomly selected examples from the dataset (bootstrap sampling). Each decision tree is trained independently on these randomly selected examples. Then, in classification tasks, the class predictions of each tree are combined, and the class with the most votes is chosen as the final prediction. In regression tasks, the results of various trees are averaged to produce a final output (Breiman, 2001).

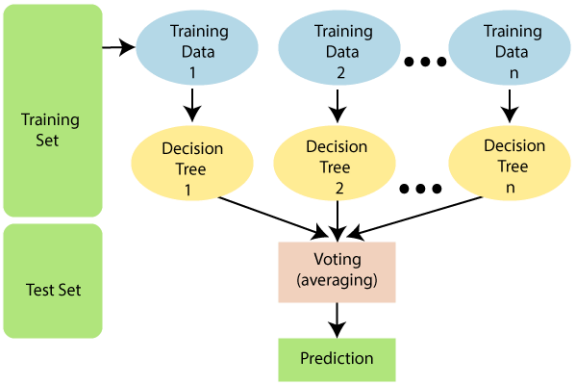

Figure 5. Visualization of the Working Principle of the Random Forest Algorithm

In the study, the following parameters were used when applying RF:

- **n_estimators=100**: Specifies the number of trees to be created in the forest.
- **criterion='gini'**: The function used to measure the quality of splits. By default, Gini impurity is used.
- **max_depth=None**: There is no limit on the maximum depth of the trees.
- **min_samples_split=2**: The minimum number of samples required to split an internal node.

2.4.4 Support vector machines

Support Vector Machines (SVM) are a supervised machine learning algorithm with strong foundations based on the Vapnik-Chervonenkis theory, and they often perform better than neural networks and radial basis function artificial neural networks. SVMs are widely used in many real-life problems such as marketing, text recognition, and image classification. The popularity of SVMs is based on advantages such as computational ease, scalability, and robustness against outliers. SVMs perform well even in situations with a small amount of training data and a large number of features, and they stand out for not having an upper limit on dataset size. As shown in Figure 6, SVMs select the hyperplane that separates data points from different classes, ensuring that the distance from the closest data points to this hyperplane (support vectors) is maximized. This results in the widest margin between classes, which increases the model's ability to generalize. Support vectors are critical points that define this optimal separating hyperplane and define the decision boundary of the model (Cortes & Vapnik, 1995).

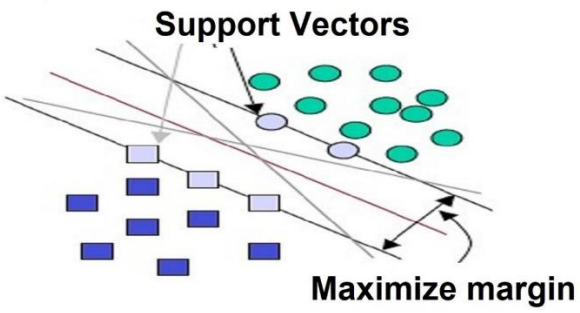

Figure 6. Creation of Support Vectors

In the conducted study, the following parameters were used while employing the SVM algorithm:

- **C=1.0**: The parameter that determines the penalty for misclassifications.
- **kernel='rbf'**: The type of kernel used. By default, 'rbf' (Radial Basis Function) is used.
- **degree=3**: If a polynomial kernel is used (kernel='poly'), this parameter defines the degree of the kernel.
- **Gamma='scale'**: The kernel coefficient.

2.5 Deep Learning Methods

2.5.1 Convolutional neural network

Convolutional Neural Network (CNN) is one of the deep learning architectures, particularly standing out as a multilayer feedforward artificial neural network. This neural network is notable for requiring less training data and fewer parameters in large network models. CNNs find wide application, especially in object recognition, image analysis, and natural language processing, as shown in an example in Figure 7 (LeCun et al., 2015). One of the significant advantages of CNNs is their ability to work effectively on large and complex datasets. Known for their success in image analysis, these networks achieve high accuracy rates in tasks such as object recognition. Additionally, CNNs have been successfully used in natural language processing, where they demonstrate impressive applicability in tasks such as text analysis, sentence analysis, and classification (Krizhevsky et al., 2012).

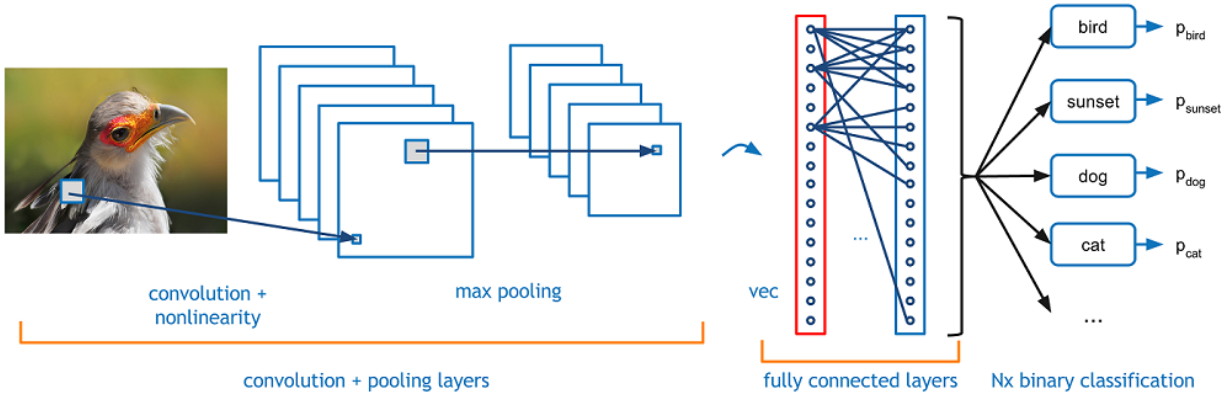

Figure 7. Image Analysis with Convolutional Neural Networks

In this study, the random_state=42 parameter was used to ensure a balanced split of the data and minimize the effect of randomness. The data was converted to the float32 type to be used in the format required by the model.

The model architecture was designed to perform deep feature extraction on sequential data. In the first layer, a 1D Convolutional layer with 64 filters and a kernel size of 3 units was used. This layer was used with the ReLU activation function to extract important features from the input data. After the convolution operation, a 2-unit MaxPooling layer was applied to reduce the dimensions of the data and preserve the important features. Then, the data was flattened using the Flatten layer and passed on to fully connected layers

In the fully connected layers, a Dense layer with 128 neurons was added first, and the ReLU activation function was used in this layer as well. The output layer, due to the binary nature of the classification problem, consists of a single neuron with a sigmoid activation function.

The model was compiled using the Adam optimization algorithm and the binary_crossentropy loss function. The Adam optimizer was set with a learning rate of 0.001. The performance of the model was tracked during the training process using the accuracy metric. The model was trained first for 10 epochs, then for 50 epochs, with a mini-batch size of 32, and validation was performed on the test data at the end of each epoch. The results obtained in Section 3 will be compared and discussed in detail in the following sections.

In the final stage, the predictions of the model on the test data were evaluated, and the accuracy of the predictions was calculated using the accuracy_score metric.

2.5.2 Long short-term memory (LSTM)

LSTM, or Long Short-Term Memory, is a deep learning model specifically designed to work with sequential data. LSTMs are widely used to learn long-term dependencies in sequential and time-dependent data structures such as time series data, text, and speech. While traditional neural networks face issues with the backward flow of information over time (such as the vanishing gradient problem), LSTMs overcome this issue, allowing them to effectively learn information that spans long time periods. A 1D LSTM essentially has an architecture that processes information at each time step while preserving the temporal order of the data and taking into account the information from previous steps. This model consists of a series of cells, each of which contains structures known as input, forget, and output gates (Hochreiter & Schmidhuber, 1997). As shown in the visual in Figure 8, these gates are used to regulate the flow of information within the cell, forget unnecessary information, or retain important information.

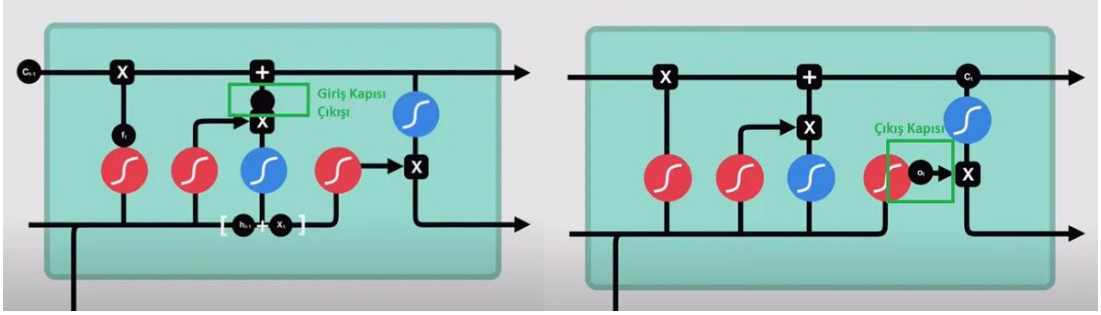

Figure 8. Long Short-Term Memory Input and Output Gates

In this study, the balanced division of the data and the minimization of the effects of randomness were controlled using the random_state=42 parameter. The data were converted to the float32 type to be used in the LSTM model's required format.

The architecture of the model was designed to capture sequential dependencies in time series data. In the first layer, an LSTM layer with 64 neurons was used. This layer was combined with the ReLU activation function to learn the temporal dependencies in the data. After the LSTM layer, a Dense layer with 128 neurons was added as a fully connected layer. Again, the ReLU activation function was used in this layer. The output layer consists of a single neuron with a sigmoid activation function for the binary classification problem.

The model was compiled with the Adam optimization algorithm. The learning rate of the Adam optimizer was set to 0.001, and the model's loss was minimized using the binary_crossentropy function. Accuracy was used as the performance metric. The model was trained for 10 epochs initially, followed by 50 epochs, with a mini-batch size of 32, and validation was performed on the test data after each epoch. The results obtained will be compared and discussed in detail in Section 3.

2.5.3 Bidirectional long short-term memory (Bi-LSTM)

Bidirectional Long Short-Term Memory (Bi-LSTM) is a variation of LSTM, specifically designed to learn sequential data both from past to future and future to past. Bi-LSTMs retain the advantages of LSTM while processing sequential data in two directions, which allows them to learn richer contextual information, especially in areas such as text and speech recognition. This enables the model to consider both previous and subsequent information at each point in a sequence, resulting in a more comprehensive understanding and prediction ability (Schuster & Paliwal, 1997).

A one-dimensional Bi-LSTM, as shown in Figure 9, essentially consists of two separate LSTM layers. One layer processes the data in temporal order (from past to future), while the other layer processes the data in reverse order (from future to past). The information from both directions is then combined or appropriately arranged for the next layer. This process allows the model to access both previous and subsequent contextual information at each time step, thus enabling it to learn dependencies in sequential data structures more effectively.

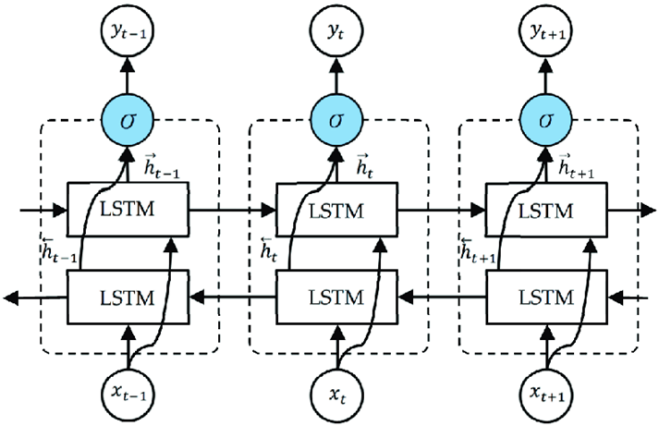

Figure 9. Visualization of Bidirectional Long Short-Term Memory

The model architecture is designed to capture both forward and backward dependencies in time series data using a bidirectional LSTM layer. The first layer consists of a bidirectional LSTM with 64 neurons. This layer is configured to learn both forward and backward dependencies in the time series data and is used with the ReLU activation function. Following this layer, a Dense layer with 128 neurons is added, where the ReLU activation function is also preferred. Finally, the output layer consists of a single neuron with a sigmoid activation function, which is suitable for binary classification problems.

The model is compiled using the Adam optimization algorithm and the binary_crossentropy loss function. The learning rate for the Adam optimizer is set to 0.001. The model's performance is monitored using the accuracy metric. During the training process, the model is trained for 10 epochs with a mini-batch size of 32 units, and at the end of each epoch, the model's performance on the test data is evaluated using a validation set.

2.6 Dimension Reduction and Model Simplification Techniques

2.6.1 Least absolute shrinkage and selection operator (LASSO)

The Least Absolute Shrinkage and Selection Operator (LASSO) is a regularization technique used in regression analysis. This method aims to prevent overfitting by reducing the complexity of the model and creating simpler, more interpretable models. LASSO adds an L1 norm (absolute value) penalty to the regression coefficients, which allows some coefficients to be reduced to zero. As a result, only the most significant variables remain in the model, while the irrelevant ones are eliminated (Tibshirani, 1996).

In the study, the value of the alpha parameter in LASSO regression was chosen as 0.01. A small alpha value limits the regularization effect, allowing the model to include more features and better fit the data. However, very small alpha values can increase the risk of overfitting. The chosen value of alpha=0.01 aims to optimize the model's ability to select and retain important features while minimizing the inclusion of unnecessary ones.

2.6.2 Principle component analyses (PCA)

Principal Component Analysis (PCA) is a powerful dimensionality reduction technique used to reduce the complexity of datasets. Its primary objective is to create a new set of components (principal components) by using linear combinations of the original variables in the dataset. These new components consist of linear combinations that explain the maximum variance in the data, with each component being selected to be independent (orthogonal) of the previous ones. PCA facilitates the projection of high-dimensional data into a lower-dimensional space, making the dataset simpler and more interpretable. Throughout this process, information loss is minimized, preserving the overall structure of the original data to a great extent (Jolliffe, 2002).

PCA is particularly effective in addressing multicollinearity issues and reducing model complexity in high-dimensional datasets. It is widely used in machine learning and statistical analyses to reduce the number of features, thereby decreasing the computational load and enhancing interpretability (Hotelling, 1933).

In the study, the parameter n_components in PCA was set to 10. These 10 components effectively preserve the main variance of the dataset while maintaining the data's dimensionality at a manageable level. This choice reduces dimensionality while retaining a significant portion of the dataset's information. The value of **n_components=10** was chosen to explain a substantial portion of the dataset's variance (e.g., 95%), achieving a compact representation of the data. This parameter was optimized based on the application requirements and the characteristics of the dataset, aiming to minimize information loss while ensuring a manageable data size during the dimensionality reduction process.

2.7 Ensemble Learning

2.7.1. Application of ensemble learning methods to machine learning techniques

Ensemble learning is a strategy formed by combining multiple models, aiming to improve model performance. Ensemble methods typically consist of the combination of different learning algorithms or various variations of the same algorithm. These methods can enhance overall model performance by compensating for the weaknesses of individual models (Dietterich, 2000).

In this study, an ensemble model was created using the **VotingClassifier**, which combines four different classification algorithms: Logistic Regression, Decision Trees, Random Forest, and Support Vector Machines. The model was configured with the option **voting='hard'**. This option uses majority voting based on the predictions of each model. The performance of the ensemble model was evaluated using the accuracy score calculated on the test data.

The results show that the test accuracy of the ensemble model reflects the overall success achieved by combining various classification methods. These findings will be explained and interpreted in the results section, accompanied by tables.

2.7.2 Application of Ensemble Learning Methods to Deep Learning Techniques

In this study, three different deep learning models were applied: CNN, LSTM, and Bi-LSTM. These models were used to learn various aspects of the data, aiming to improve the results.

2.7.2.1 Combination of CNN and LSTM

In this model, CNN and LSTM layers were combined. The CNN layers extract features from the time series data, while the LSTM layers learn temporal dependencies. The outputs of these two models were combined, and a common dense layer was added.

CNN Layer: 64 filters, a kernel size of 3, and the ReLU activation function were used.
LSTM Layer: Configured with 64 units and the ReLU activation function.

2.7.2.2 Combination of CNN and Bi-LSTM Model
In this model, CNN and Bi-LSTM layers were combined. The CNN layers learn the local features of the data, while the Bi-LSTM layers learn temporal dependencies in both directions.
CNN Layer: 64 filters and a kernel size of 3 were used.
Bi-LSTM Layer: Configured with 64 units and the ReLU activation function.

2.7.2.3 CNN, LSTM, and Bi-LSTM Models
Finally, the ensemble learning method was applied to three separate deep learning models, and the results were combined.
CNN Layer: 64 filters, a kernel size of 3, and the ReLU activation function were used.
LSTM Layer: Configured with 64 units and the ReLU activation function.
Bi-LSTM Layer: Configured with 64 units and the ReLU activation function.

3. Result and Discussion

In the machine learning section, the dataset was trained using Logistic Regression (LR), Decision Trees (DT), Random Forest (RF), and Support Vector Machines (SVM). The models were developed using ensemble learning methods, and performance improvements were attempted by applying techniques such as LASSO and PCA. After LASSO and PCA, the models were reevaluated using ensemble learning methods, and their efficiencies were compared.
In the deep learning section, the dataset was trained using Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (Bi-LSTM). Initially, training was done for 10 epochs, and then for 50 epochs, with the results recorded in a table. As hybrid model approaches, models were trained using CNN and LSTM, followed by CNN and Bi-LSTM, and accuracy scores were compared. Finally, three deep learning models were combined to create an ensemble learning hybrid model, and the performance of this model was recorded in a table.
This section provides a detailed explanation and discussion of the results obtained from the applied models and methods, reflecting on how this study contributes to stroke risk prediction and exploring its potential future applications.

3.1 Results Obtained from Machine Learning Algorithms

The data in the dataset were split at different ratios, as shown in Table 4, and various combinations for testing and training were tried in order to obtain the best result:

Table 4. Accuracy Values at Different Test and Training Ratios

Accuracy	Logistic Regression	Decision Tree	Random Forest	SVM
10% Test- 90% Training	0.9491	0.8884	0.9491	0.9491
20% Test- 80% Training	0.9500	0.9080	0.9540	0.9559
30% Test- 70% Training	**0.9562**	**0.9145**	**0.9543**	**0.9562**
50% Test- 50% Training	0.9495	0.9021	0.9487	0.9495

In Table 4, the accuracy values of four different models are compared at different training and test ratios. Overall, it is observed that the Support Vector Machines (SVM) model achieved the highest accuracy values across various training and test ratios. This suggests that SVM is better at distinguishing the features present in the dataset compared to other models, and it shows higher generalization capabilities. One of the reasons SVM outperforms the other models in terms of accuracy is that it maximizes the margin between classes, ensuring the best separation. This makes the distinctions in the data more prominent and improves its generalization ability. Furthermore, the regularization parameters in SVM prevent overfitting, allowing the model to perform well on both training and test sets. These features enable SVM to learn patterns and relationships in the dataset more effectively than other models.

When examining the impact of training and test ratios on model performance, it was observed that at the 10% Test- 90% Training ratio, the Logistic Regression (LR), Random Forest (RF), and SVM models all achieved the same accuracy values. However, the Decision Tree model performed worse in comparison. Moving to the 20% Test- 80% Training ratio, the SVM model achieved the highest accuracy, while the Decision Tree model performed notably lower than the other models. This suggests that the Decision Tree may not be learning the relationships in the dataset adequately or might struggle with more complex structures.

At the 30% Test- 70% Training ratio, an overall increase in accuracy values across all models was observed. Both SVM and Logistic Regression models achieved the same accuracy, indicating that they have strong generalization capabilities on this dataset. However, when transitioning to the 50% Test- 50% Training ratio, a slight decrease in accuracy was observed. This could indicate that with less training data, the models reached their limits in terms of generalization ability.

Logistic Regression and Random Forest models generally exhibited similar accuracy values. However, the Decision Tree model stood out with lower and more inconsistent accuracy values compared to the other models. These findings suggest that the Decision Tree

model struggled to learn the complexities in the dataset. One potential reason for this could be that the Decision Tree tends to overfit the details in the dataset, which may result in good performance on training data but difficulties in learning the general patterns on test data.

As a result, SVM, Random Forest and Logistic Regression models performed better across different training and test ratios, demonstrating superior generalization capabilities. The findings in Table 4 show that training and test ratios have a significant impact on model performance, but this effect may vary depending on the model type.

In another study, the impact of different outlier handling methods on accuracy values was examined. Outliers in the dataset were detected before training, and the best accuracy values obtained by processing these outliers according to the strategies presented in Table 5 were determined.

Table 5. Accuracy Values Changing Based on Outliers

Accuracy Values	Logistic Regression (LR)	Decision Tree (DT)	Random Forest (RF)	SVM
Before Outliers are Removed	0.9562	0.9145	0.9543	0.9562
After Outliers are Removed	**0.9577**	**0.9305**	**0.9578**	**0.9578**
When the Mean of Outliers is Taken	0.9566	0.9178	0.9554	0.9563

Table 5 shows the accuracy values of four different machine learning models after processing outliers with various methods. Models such as Logistic Regression, Decision Tree, Random Forest, and Support Vector Machines have shown an increase in accuracy when outliers were removed. Specifically, after removing outliers, the Decision Tree model displayed improved accuracy compared to the scenario where outliers were not removed. This suggests that outliers may have a negative impact on the model's generalization ability, and their removal allows the models to produce more accurate results.

Additionally, replacing outliers with the mean of the dataset resulted in minimal changes in accuracy. This indicates that replacing outliers with the mean may have a neutral effect on model performance, neither significantly improving nor worsening it.
Possible reasons for these differences include the varying degrees of sensitivity each model has to outliers. Tree-based models, such as Decision Trees, may be more sensitive to outliers, while models like Logistic Regression and SVM may be more robust to such values. This explains why the removal or replacement of outliers leads to varying degrees of improvement or changes in model performance.

3.1.1 Application of LASSO and PCA methods to machine learning models
LASSO and PCA methods were applied to the machine learning models with the aim of increasing accuracy rates. In this study, the accuracy rates before the application of LASSO and PCA (Accuracy 1) and after their application are compared in Table 6:

Table 6. Accuracy Values After Applying Lasso and PCA

Accuracy Values	Logistic Regression	Decision Tree Model	Random Forest Model	SVM
Accuracy 1	0.9577	0.9305	0.9578	0.9578
Accuracy 2 (LASSO)	**0.96**	**0.95**	**0.95**	**0.96**
Accuracy 3 (PCA)	0.9586	0.9158	0.9579	0.9586

Table 6 demonstrates the impact of applying LASSO and PCA methods on the accuracy values of machine learning models. After applying LASSO, a significant increase in accuracy was observed, particularly in the Logistic Regression and SVM models, where their accuracy values increased to 0.96. In the Decision Tree model, the accuracy value also rose to 0.95 with LASSO, indicating a marked improvement in the model's performance. On the other hand, in the Random Forest model, the accuracy decreased from 0.9578 to 0.95. These results suggest that LASSO enhances the accuracy of models by eliminating unnecessary features, but in some models, due to their structural properties, it can lead to a decrease in accuracy.

On the other hand, the accuracy values obtained after applying PCA showed minimal changes compared to the original values. After applying PCA, the accuracy values for Logistic Regression and SVM models increased to 0.9586, while the accuracy for the Random Forest model remained almost unchanged. However, the Decision Tree model experienced a decline in accuracy after PCA was applied. This decrease in accuracy observed in the Decision Tree model after applying PCA may have occurred due to the nature of the model. Decision trees classify data points by splitting based on certain threshold values of features in the dataset. PCA, on the other hand, reorganizes the original features into linear combinations that explain the maximum variance. This transformation may cause the decisive split points previously used by the decision tree to disappear or change. As a result, the performance based on the decision tree's significant variable relationships may decrease, leading to a drop in accuracy. This suggests that the Decision Tree model may perform more effectively when working with the original features.

The effect of LASSO, due to its nature of eliminating unnecessary features, provided a more noticeable performance improvement, while PCA's dimensionality reduction capabilities resulted in a more limited improvement in accuracy. This suggests that LASSO

could be a more effective method, especially for large datasets with a high number of features, when the goal is to optimize accuracy rates.

3.1.2 Application of ensemble learning method to machine learning models

In the first scenario, an ensemble model was created using four different machine learning algorithms, and this model achieved an accuracy rate of 95.85%. In the second trial, the machine learning algorithms were subjected to another ensemble learning process; however, this time, the LASSO method was applied before the ensemble learning method, resulting in an accuracy of 96%. In the third case, ensemble learning performed after PCA application reached an accuracy rate of 95.86%. Finally, deep learning models, which will be discussed in more detail in the following sections, were combined into an ensemble and achieved an accuracy of 94.42%. These results are shown in Table 7.

Table 7. Ensemble Learning Accuracy Values

Ensemble Learning	Accuracy Rate
Accuracy Value (LR, KA, RO, DVM)	0.9585
Accuracy Value (LR, KA, RO, DVM with LASSO)	**0.96**
Accuracy Value (LR, KA, RO, DVM with PCA)	0.9586
Accuracy Value (ESA, UKSB, İY-UKSB)	0.9442

These data show that ensemble learning methods are effective in improving model performance by combining various machine learning algorithms. However, the inability to achieve higher accuracy values is believed to be due to the preprocessing techniques used and the significant impact that the hyperparameter settings of each model have on performance.

3.1.3 10-fold cross-validation method as an alternative approach

10-fold cross-validation is a commonly used method for evaluating the performance of machine learning models. This method divides the dataset into 10 equal parts, using each part as a test set while the remaining 9 parts are used for training. This process is repeated 10 times, with a different part being used as the test set each time. As a result, the model's performance is calculated for each test set, and an average performance value is obtained. This method provides a more reliable way to measure the model's generalization ability and reduces the risk of overfitting (Kohavi, 1995).

In the study conducted, in addition to different test and training dataset splitting methods, the 10-fold cross-validation method was also applied to increase the reliability of the study, and the resulting accuracy values were recorded in Table 8.

Table 8. 10-Fold Cross-Validation Results

Model	10-Fold Cross-Validation Results	Average Accuracy
Logistic Regression	[0.9558, 0.9558, 0.9579, 0.9558, 0.9579, 0.9558, 0.9537, 0.9579, 0.9578, 0.9578]	95.66%
Random Forest	[0.9558, 0.9579, 0.9579, 0.9579, 0.9579, 0.9558, 0.9558, 0.9579, 0.9578, 0.9578]	95.72%
Decision Trees	[0.9011, 0.9200, 0.9179, 0.9347, 0.9221, 0.9137, 0.9263, 0.9116, 0.9241, 0.9051]	91.76%
SVM	**[0.9558, 0.9579, 0.9579, 0.9579, 0.9579, 0.9579, 0.9579, 0.9579, 0.9578, 0.9578]**	**95.77%**

In the Logistic Regression model, the 10-fold cross-validation results ranged between 95.37% and 95.79%, with an average accuracy of 95.66%. The Random Forest model achieved an average accuracy of 95.72%, with accuracy rates ranging from 95.58% to 95.79%, making it one of the best-performing models. The Support Vector Machine model, with accuracy rates ranging from 95.58% to 95.79%, achieved an average accuracy of 95.77%, providing the highest average accuracy among the models.

In contrast, the Decision Trees model showed accuracy rates between 90.11% and 93.47%, with an average accuracy of 91.76%. The Decision Trees model exhibited a lower accuracy rate compared to the other models, suggesting that its generalization capacity might be more limited than the others.

In conclusion, the 10-fold cross-validation method has proven to be an effective tool for more reliably measuring model performance. The results show that the SVM model generally provided the highest accuracy, while the Random Forest model also displayed highly competitive performance. The lower accuracy rate of the Decision Trees model compared to the others indicates that additional tuning

and improvement steps might be needed to enhance its generalization ability. These findings emphasize the need for further investigation of machine learning models using different dataset splitting strategies and cross-validation methods.

3.2 Results Obtained from Deep Learning Algorithms

In the deep learning section of the study, Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BiLSTM) methods were chosen. The experiments initially began with 10 epochs and were later adjusted to use 50 epochs in order to achieve the highest accuracy. The obtained results have been recorded in detail in Table 10:

Table 10. Accuracy Values Changing with Different Epoch Values

Epoch Value	CNN	LSTM	BiLSTM
Epoch-10	0.9431	0.9442	0.9421
Epoch-50	**0.9442**	**0.9442**	**0.9442**

In the deep learning section of the study, Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BiLSTM) methods were chosen. These methods were selected to evaluate the performance of deep learning models on sequential and spatial data. The experiments were initially conducted with 10 epochs, and then the number of epochs was increased to 50 to maximize the accuracy of the model.

According to the results in Table 10, for Epoch-10, the CNN, LSTM, and BiLSTM models achieved accuracies of 94.31%, 94.42%, and 94.21%, respectively. At Epoch-50, the accuracy rates of all three models equalized at 94.42%, with a slight improvement observed. These findings suggest that as the number of epochs increases, the accuracy rates of deep learning models tend to stabilize, with the LSTM model demonstrating stable results even with a smaller number of epochs.

These results highlight that increasing the number of epochs has a certain impact on performance, but this effect remains limited, and no significant increase in accuracy should be expected beyond a certain point. This emphasizes the importance of carefully selecting the number of epochs during the training of deep learning models.

3.2.1 Application of Ensemble Learning Method to Deep Learning Models

Hybrid models were created using these three models, with 50 epochs, and efforts were made to achieve the best possible results with these models. The best obtained values are recorded in Table 11:

Table 11. Accuracy Values of Hybrid Models in Deep Learning Algorithms

Model	Accuracy Value
CNN, LSTM	0.9442
CNN, BiLSTM	0.9442
CNN, LSTM, BiLSTM	0.9442

In this section, the ensemble learning method was applied to the Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BiLSTM) models. The goal was to create hybrid models that maximize accuracy through the combination of these models. The experiments were conducted using 50 epochs, and the accuracy values of hybrid models with different combinations were compared.

Upon examining the accuracy values of the hybrid models presented in Table 11, it was observed that, despite using different combinations of CNN, LSTM, and BiLSTM, the accuracy rates remained constant. There are several possible reasons for this. First, the similar performance characteristics of the models used could have led to results that were close to each other. Since CNN, LSTM, and BiLSTM models respond similarly to the dataset, each model may have maintained the same level of accuracy.
Moreover, the combination of models did not provide additional improvement as expected, and the result of each combination remained close to the results of the individual models. Additionally, the dataset used may not have sufficient diversity or complexity to differentiate the performance of these models. As a result, the accuracy rates of the hybrid models remained unchanged, and no significant difference in model performance was observed.

3.3 Literature Table

Upon analyzing the literature, it is observed that similar studies using machine learning algorithms have achieved high success rates. However, the uniqueness of this study lies in its departure from traditional approaches by integrating deep learning methods, advanced techniques such as LASSO and PCA, and ensemble learning methods. This approach differentiates itself from other studies in the literature by adding a new dimension to the existing body of knowledge and providing researchers with an alternative perspective.

Table 12. Literature Comparison Table

Authors	Title of the Study	Dataset Used	Algorithms Used	Accuracy Values
Özer Oğuz, Suat Bayır, Hasan Badem (OĞUZ et al., 2021)	Performance of Machine Learning Methods in Determining Stroke Risk: A Comparative Study	Stroke Prediction Dataset (2021)	Logistic Regression, Decision Trees, Random Forest Classifier, Support Vector Machine	Random Forest Classifier: 99.425% accuracy, the best performing method
Onur Sevli (Sevli, 2021)	Detection of Stroke Risk Using Machine Learning	Stroke Prediction Dataset (2021)	Random Forest	Accuracy: 0.9884, Precision: 0.9776, Sensitivity: 1.0, F1 Score: 0.9886
Chandra, K. Kausalya, Kanaga Suba Raja, B. Ciddarth RM, Gokul Ranjith V (Kanaga Suba Raja et al., 2023)	Prognosis of Stroke using Machine Learning Algorithms	Stroke Prediction Dataset (2021)	Logistic Regression, Random Forest, K-Nearest Neighbors, Support Vector Machines, Decision Trees	KA: 86.45%, LR: 92.83%, K-Nearest: 93.44%, RO: 93.73%, DVM: 93.93%
Rashmi Kuksal, Musheer Vaqur, Ashish Bhatt (Kuksal et al., 2023)	Stroke Disease Detection and Prediction using Extreme Gradient Boosting	Stroke Prediction Dataset (2021)	Random Forest, XGBoost, and SVC	XGBoost: 96%, Random Forest: Not specified, below 96%, SVC: Not specified, below 96%
This Study	Stroke Prediction Dataset (2021)	Logistic Regression, Decision Trees, Random Forest, DVM, CNN, LSTM, BiLSTM	After LASSO, Logistic Regression, Support Vector Machines, and Ensemble Learning methods in Machine Learning resulted in 0.96 accuracy.	0.96 accuracy

The literature comparison presented in Table 12 compares the success rates of machine learning methods used for stroke risk prediction using the same dataset (Stroke Prediction 2021) and highlights how this study differentiates itself from others in the literature. This comparison provides important insights into the contributions and potential shortcomings of our study, helping to better understand its position within the existing body of research.

The most notable advantage of this study is the use of a broader and more advanced range of algorithms compared to other works in the literature. By going beyond traditional approaches, methods such as LASSO, PCA, and Ensemble Learning have been applied. Specifically, the LASSO method has enabled the elimination of unnecessary features in the model, creating a more efficient and faster system. The Ensemble Learning approach, on the other hand, improves prediction performance by combining multiple models, offering a unique and innovative contribution compared to methods in the literature.

This methodological diversity ensures that our study stands out from other studies in the literature and provides researchers with an alternative perspective on stroke risk prediction.

However, the study's relatively lower accuracy compared to other studies in the literature is noteworthy. For example, in the study by Özer Oğuz et al. (2021), the Random Forest algorithm stands out with an accuracy of 99.425%, while the highest accuracy in this study is 96%. This discrepancy may stem from hyperparameter tuning or optimization of model combinations.

In studies with higher accuracy rates, it has been observed that simpler algorithms (e.g., using only Random Forest) yield better results. This can be explained by the fact that simpler models have fewer parameters, thus reducing the risk of overfitting and better fitting the dataset. Additionally, it should be considered that techniques like LASSO and PCA may lead to the loss of some important information during the learning process, which could negatively impact accuracy.

In this context, the methodological innovation of this study provides a significant contribution to the literature. Even though the advanced techniques used in this study resulted in a lower accuracy compared to other studies, the alternative perspectives and methods presented by this research are valuable. In conclusion, the study offers a new and valuable approach to researchers in the field of stroke risk prediction and enriches the existing literature in this domain.

4. Conclusion

In this study, the effectiveness of various machine learning and deep learning algorithms for stroke prediction was evaluated, and accuracy rates were compared. The research findings showed that machine learning algorithms such as Logistic Regression, Support Vector Machines, Random Forest, and Decision Trees, as well as deep learning algorithms like Convolutional Neural Networks, Long Short-Term Memory, and Bidirectional Long Short-Term Memory, have significant potential in stroke prediction. Dimension reduction techniques such as LASSO and PCA, along with Ensemble Learning methods, were applied to enhance the model's accuracy and generalizability. As a result of the experiments, the highest accuracy rate of 96% was achieved with Logistic Regression following the LASSO (Least Absolute Shrinkage and Selection Operator) method, Support Vector Machines following the LASSO method, and the Ensemble Learning method of Machine Learning Algorithms (Logistic Regression, Decision Trees, Random Forest, Support Vector Machines).

The successful application of the model in clinical settings depends on the proper selection of the model and its alignment with patient demographics. Furthermore, the effectiveness of the model should be tested on a broad patient population using complex datasets. Future studies may aim to examine the performance of these models on different disease types and stages in more detail and focus on improving their integration into clinical applications. Additionally, hyperparameter tuning on machine learning and deep learning algorithms will help improve the effectiveness of the models used.

In clinical and educational settings, providing healthcare professionals with knowledge about how machine learning and deep learning models work, their advantages, and limitations will enable more effective use of these technologies in clinical environments. Based on this information, experts can make more accurate and faster diagnoses and improve treatment planning. Educational programs should focus on how these models can be integrated, interpreted, and incorporated into patient care processes in clinical environments.

References

Breiman, L. (2001). Random forests. Machine Learning, 45, 5-32.

Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (1986). Classification and Regression Trees. Wadsworth & Brooks/Cole.

Chandra, B., Kausalya, K., & Ciddarth, R. M. (2023, February). Prognosis of stroke using machine learning algorithms. In 2023 7th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 1-6). IEEE.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20, 273–297.

Dietterich, T. G. (2000, June). Ensemble methods in machine learning. In International Workshop on Multiple Classifier Systems (pp. 1-15). Springer Berlin Heidelberg.

Emon, M. U., Keya, M. S., Meghla, T. I., Rahman, M. M., Al Mamun, M. S., & Kaiser, M. S. (2020, November). Performance analysis of machine learning approaches in stroke prediction. In 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA) (pp. 1464-1469). IEEE.

Federisino (2021). Stroke prediction dataset [Data set]. Kaggle. https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression (3rd ed.). Wiley.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24(6), 417-441.

Iglewicz, B., & Hoaglin, D. C. (1993). How to detect and handle outliers. Sage Publications.

Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 4-37.

Jolliffe, I. T. (2002). Principal Component Analysis. Springer Series in Statistics.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the 14th International Joint Conference on Artificial Intelligence (Vol. 2, pp. 1137-1143).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097-1105.

Kuksal, R., Vaqur, M., Bhatt, A., Chander, H., & Joshi, K. (2023, January). Stroke disease detection and prediction using extreme gradient boosting. In 2023 International Conference on Artificial Intelligence and Smart Communication (AISC) (pp. 187-191). IEEE. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

Lee, H., Lee, E. J., Ham, S., Lee, H. B., Lee, J. S., Kwon, S. U., ... & Kang, D. W. (2020). Machine learning approach to identify stroke within 4.5 hours. Stroke, 51(3), 860-866.

Oğuz, Ö., Bayır, S., & Badem, H. (2021). Makine öğrenmesi yöntemlerinin felç riskinin belirlenmesinde performansı: Karşılaştırmalı bir çalışma. Computer Science, (Special), 274-287.

Pallavi, K., & Saravananthirunavakarasu. (2022). Classification of stroke disease using machine learning algorithms. https://ijcrt.org/papers/IJCRT2208395.pdf

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

Sevli, O. (2021). İnme (Felç) riskinin makine öğrenmesi kullanılarak tespiti (Determining the risk of stroke using machine learning). https://www.researchgate.net/publication/351776808

Singh, M. S., & Choudhary, P. (2017, August). Stroke prediction using artificial intelligence. In 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON) (pp. 158-161). IEEE.

Srinivas, A., & Mosiganti, J. P. (2023). A brain stroke detection model using soft voting based ensemble machine learning classifier. Measurement: Sensors, 29, 100871.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1), 267-288.

World Health Organization. (2022, October 29). World Stroke Day 2022. Retrieved from https://www.who.int/srilanka/news/detail/29-10-2022-world-stroke-day-2022