**A project report on**

**Stroke Risk Prediction Using**

**Artificial Neural Networks**

**By**

**S Sai Akhil**

**Contact details:-**

Email:- saiakhil0711@gmail.com

Contact No:- 9959410761

**ABSTRACT**

Stroke is a significant public health concern, causing substantial mortality and long-term disability worldwide. Early detection and prevention are vital for mitigating the impact of this life-threatening condition. In recent years, the application of Artificial Neural Networks (ANNs) has gained prominence in healthcare for predictive modeling and risk assessment.

This project presents a comprehensive analysis of stroke risk prediction using ANNs. Leveraging a diverse Stroke Prediction Dataset, we explore the potential of machine learning techniques to identify individuals at high risk of stroke. The dataset encompasses a wide range of attributes, including demographic, medical history, and lifestyle factors.

The methodology involves the development of an ANN model, encompassing data preprocessing, model architecture design, and thorough evaluation. Performance metrics as accuracy is employed to assess the model's predictive capabilities. Results and visualizations provide insights into the model's accuracy and clinical implications.

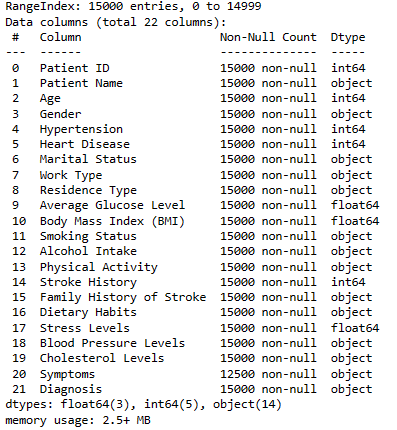
The findings of this study contribute to the field of healthcare by offering an advanced approach to stroke risk assessment, aiding healthcare professionals in making informed decisions and implementing preventive measures. The documentation serves as a valuable resource for understanding the project's methodology and results. Ultimately, the project underscores the potential of ANNs in enhancing the early detection and prevention of strokes, leading to improved patient outcomes and a reduction in stroke-related morbidity and mortality.

**Table of contents**

1. Introduction
2. Data cleansing
3. Exploratory Data Analysis ( EDA )
4. Model Building
5. Prediction
6. **Introduction**

In recent years, the application of artificial intelligence and machine learning techniques has gained significant traction in the field of healthcare. One critical area of focus is the early prediction and prevention of life-threatening medical conditions, such as strokes. Stroke is a leading cause of mortality and long-term disability worldwide, making its early detection and risk assessment of utmost importance.

* Artificial Neural Networks (ANNs) have emerged as a powerful tool for predictive modeling in healthcare. ANNs are a subset of machine learning algorithms that are particularly well-suited for handling complex and high-dimensional data. When applied to healthcare datasets, ANNs can assist medical professionals in identifying individuals at high risk of stroke, allowing for timely intervention and prevention strategies.
* This Dataset is a prime example of leveraging ANNs for stroke risk assessment. This dataset typically comprises a variety of features related to an individual's demographic, medical history, lifestyle, and more. By training an ANN on this dataset, we can build a predictive model that learns complex patterns and relationships within the data. This model can then be used to predict an individual's risk of suffering a stroke in the future.
* In this project, we aim to explore the potential of ANNs in predicting strokes by employing a comprehensive Stroke Prediction Dataset. This dataset includes a range of attributes, such as :- age, hypertension, heart disease, smoking status, etc., which can be used as inputs to the ANN I.e., (15000 rows and 22 columns). By analyzing and training the ANN on this data, we can develop a robust predictive model capable of offering insights into an individual's stroke risk.
* The significance of this work lies in its potential to assist healthcare professionals in making more informed decisions, offering personalized care, and implementing preventive measures for those at risk. Additionally, the use of ANNs in stroke prediction has the potential to enhance the efficiency and accuracy of diagnoses and risk assessments, ultimately contributing to improved patient outcomes and the reduction of stroke-related morbidity and mortality.
* First we will import the Required libraries as per the requirement.
* Then we will the load the data from the local disk to the Data frame (df).
* Performing the Data cleaning process : Removing the special characters, outliers detection, Fillling out the Nan values, removing the unwanted columns.
* The below mentioned is the summary of the data.



**2 Data Cleaning**

**Patient ID and Patient Name:**

The "Patient ID" and "Patient Name" columns are often specific identifiers and personal information related to the individuals in the dataset. In a stroke prediction model, such personally identifiable information is typically not relevant and might even pose privacy concerns. Therefore, these columns are commonly removed to maintain data privacy and to ensure the focus remains on relevant medical and demographic features.

**Work Type:**

The "Work Type" column might include information about the occupation or employment status of the patients. While occupation could be considered a potential factor in stroke risk, it is often not a direct predictor and may not significantly contribute to the predictive power of the model. Given this, "Work Type" is sometimes removed during data cleaning to simplify the dataset and improve the model's efficiency.

**Symptoms:**

The "Symptoms" column may contain information about the symptoms experienced by patients, which can be subjective and highly variable. For stroke prediction, the focus is typically on more objective and measurable factors like age, hypertension, heart disease, and lifestyle choices. Subjective symptoms might not be as reliable for predictive modeling, and therefore, this column is often excluded from the dataset.

**Blood Pressure Levels:**

"Blood Pressure Levels" are a crucial health indicator, but as we do have the Hypertension column so we can remove this column.

**Residence Type:**

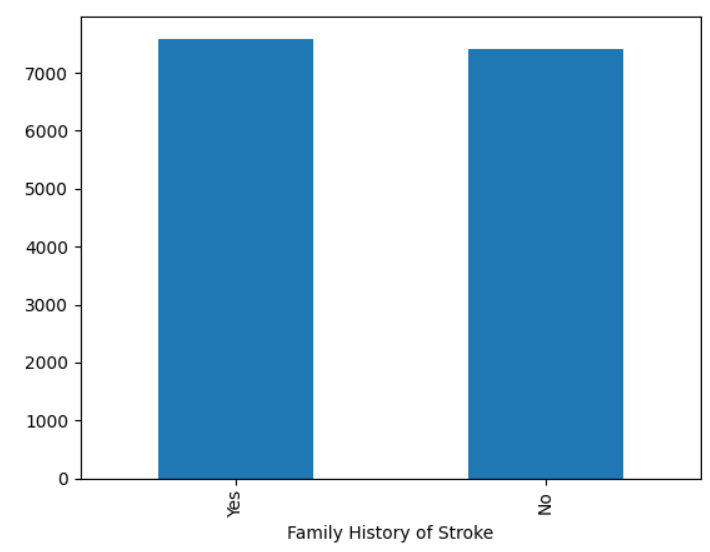
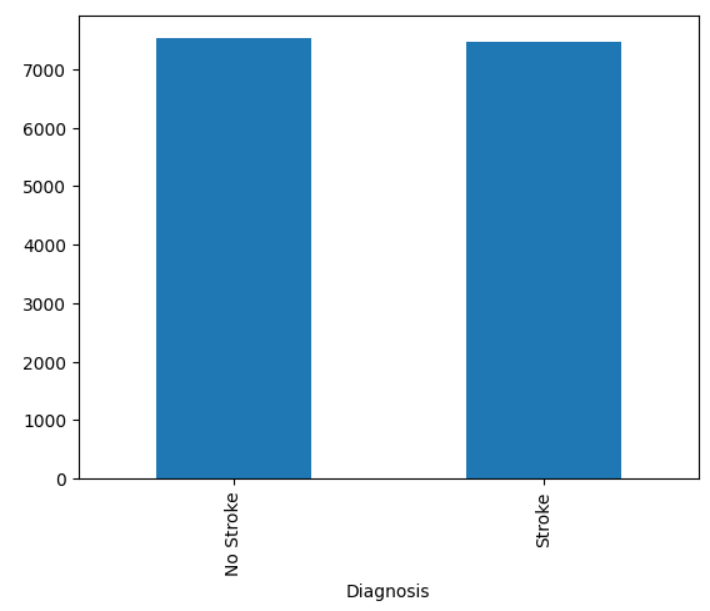
"Residence Type" typically denotes whether a patient lives in an urban or rural area. While this might have some influence on health outcomes, it is not a direct risk factor for stroke. Removing this column simplifies the dataset and focuses on factors more directly associated with stroke risk.

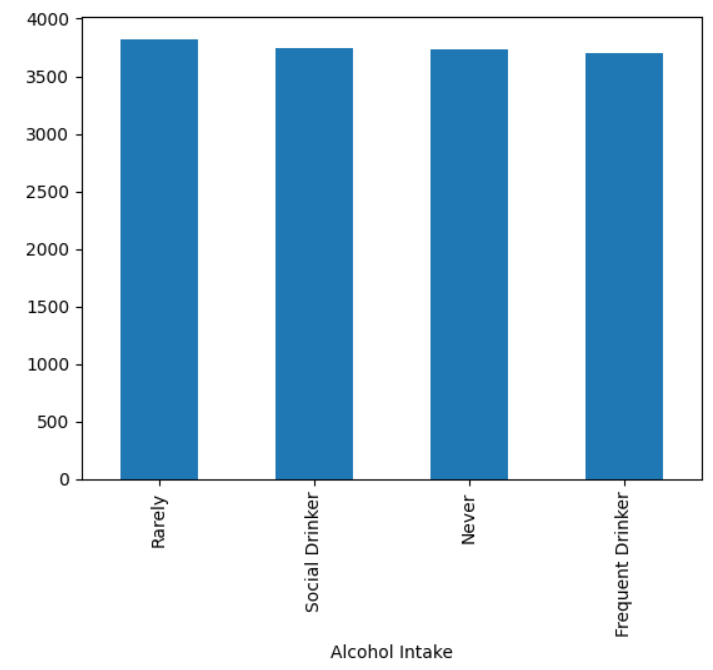
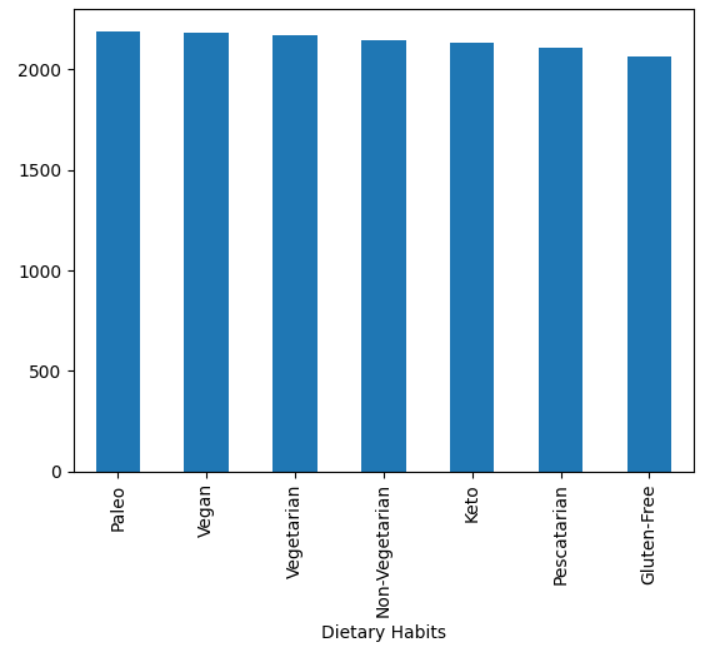
* As the HDL and LDL are in single column so we need to separate both of them as HDL is a good cholesterol and LDL is the bad cholesterol.
* Using the lambda functions and regex we are going to separate them.
* Age column is being divided into the age groups.
* At last we do the datatype conversion to their respective datatypes.
* Which will be ready for preprocessing and visualization.

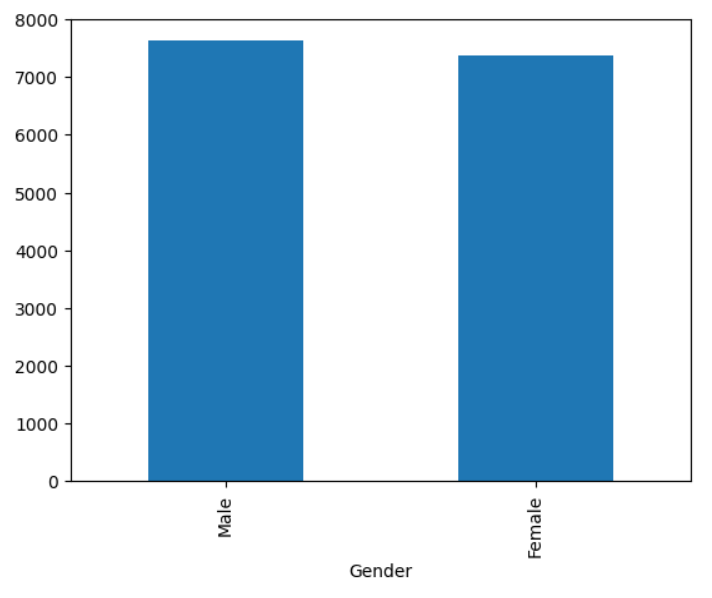
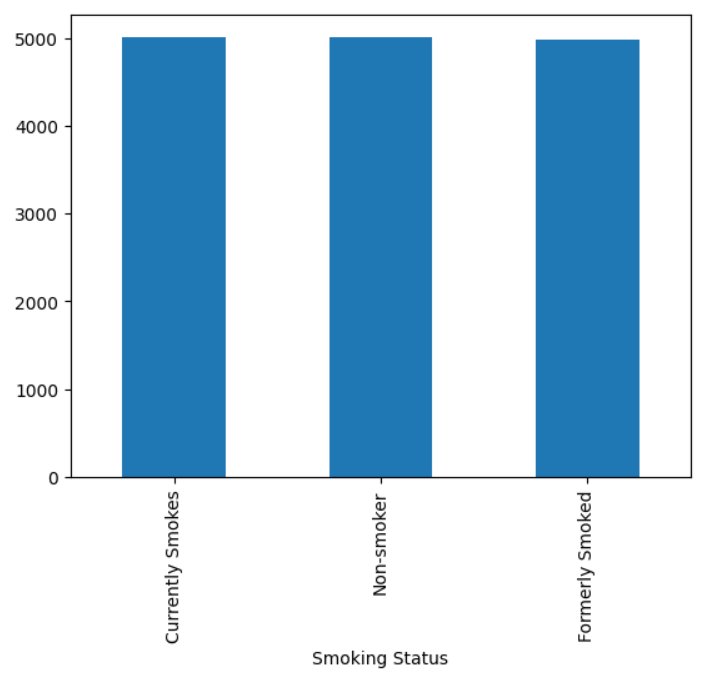
**3 Exploratory Data Analysis**

Exploratory Data Analysis (EDA) is a critical step in understanding and preparing the dataset for modeling. The main objectives of EDA are as follows :

From the Below Bar chart’s we can conclude that the data is uniformly distributed among all the columns equally

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



* On going through the EDA of the Data we can conclude that all the data in columns are equally distributed and even in the target variable column the data is equally distributed.
* Hence upon doing the analysis I can conclude that the below are the columns which are mainly causing the stroke:
  + - 1. Hypertension
      2. Family Stroke History
      3. Heart Diseases
      4. LDL Cholesterol levels
      5. Heart Diseases

**4 Model Building**

Model building is the process of creating a mathematical representation or algorithm that can make predictions or decisions based on input data, allowing machines to perform specific tasks, such as classification, regression, or pattern recognition. It involves selecting an appropriate architecture and training the model using data to optimize its performance.

Deep learning model building is utilized to leverage the power of artificial neural networks, which can capture complex patterns and relationships in data. This is particularly valuable for tasks where traditional machine learning models may struggle, such as image and speech recognition, natural language processing, and other intricate, high-dimensional data problems. Deep learning models can automatically learn and extract features from data, making them highly effective for a wide range of tasks.

The following steps involved in the Model building:

* **Data Splitting:** Data was split into training and testing sets using a 85-15% ratio (train-test split) to evaluate the model’s performance. This separation ensures that the model is trained on one subset of the data and tested on an independent subset, allowing us to assess its generalization capabilities and avoid overfitting.
* **Data Preprocessing:**
* **Separating Numerical and Categorical Columns:** In the initial data, we had a dataset with 15000 rows and 16 columns. To prepare the data for deep learning, we first separated the dataset into two types of columns: numerical and categorical. This separation is crucial because different preprocessing techniques are applied to each type.
* **Preprocessing the data:** Before training and evaluating any model, it’s essential to preprocess both the training and test data. This ensures that the model performs optimally and consistently across all datasets. The preprocessing steps for both the training and test data include the following:
* **Scaling the Numerical Columns:** We standardized the numerical columns in both the training and test data using the StandardScaler from the scikit-learn library. Standardization transforms the data to have a mean of 0 and a standard deviation of 1. This step is crucial because it ensures that all numerical features have the same scale, preventing some features from dominating the learning process.
* **One-Hot Encoding for Categorical Features:** For the categorical columns in both the training and test data, we used one-hot encoding. One-hot encoding coverts categorical variables into binary (0/1) format, where each category becomes a separate binary column. This technique ensures that the deep learning algorithm can work with categorical data effectively.
* **Concatenating Numerical and Categorical Features:** After scaling and one-hot encoding, we concatenated the processed numerical and categorical features back together for both the training and test datasets. This step ensures that both datasets are prepared in the same way and are ready for model training and evaluation.
* **Balancing the data:** In deep learning, an imbalanced dataset can lead to model bias, where the model may perform poorly on the minority class. To address this issue, we used the Synthetic Minority Over-Sampling Technique (SMOTE) to balance the dataset. SMOTE generates synthetic samples for the minority class by interpolating between existing samples. This balancing step ensures that the model has an equal representation of both classes (attended and not attended) and helps improve its performance.

After preprocessing, our training data now contains a balanced set of 12750 samples, with 2250 samples for both attended and not attended categories.

**Building a Model Using Keras-Tuner:**

* **Introduction to Keras Tuner:** Keras Tuner is a powerful library that automates the process of hyper parameter tuning, helping to find the optimal configuration for your deep learning models. In this documentation, we’ll walk through the process of building a deep learning model using Keras Tuner for the given dataset.
* **Hyper parameter Optimization Strategy:** We willuse Keras Tuner’s Random Search strategy to explore different hyper parameter combinations. This strategy randomly samples from the defined hyper parameter space to find the best- performing model.
* **Hyper parameter Explored and Tuned:**

1. Number of Hidden Layers: We vary the number of hidden layers between 10 and 300 to explore the network’s depth.
2. Neuron’s per Hidden Layers: For each hidden layer, the number of neurons is tuned between 4 and 200.
3. Activation Function: We explore three activation functions – **sigmoid**, **tanh**, and **ReLU** – for the hidden layers.
4. Weight Initialization: We experiment with weight initialization methods, including **glorot\_uniform**, **glorot\_normal**, **he\_uniform**, **he\_normal**.
5. Optimizer: We choose between four optimization algorithms – **Stochastic Gradient Descent (SGD)**, **Adam, RMSprop, and Adadelta**.

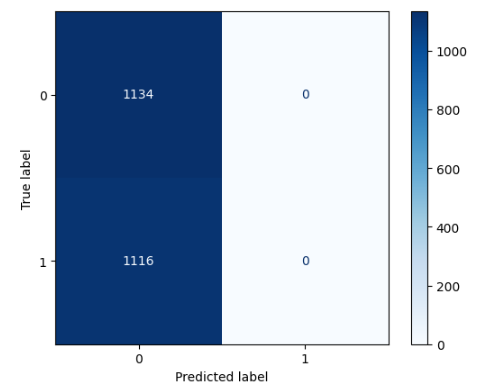
* **Objective and Metrics:** Our objectiveis to maximize validation accuracy while minimizing the binary cross-entropy loss. We aim to find the model configuration that best balances these two metrics.
* **Hyper parameter Tuning Process:**

1. We define a function called ‘**best\_model’** that constructs the neural network based on the hyper parameters selected by Keras Tuner.
2. We use Keras Tuner to perform a random search over the hyper parameter space to find the best model configuration.
3. The search is conducted with 10 training epochs, and we validate the models using a separate validation dataset.

* **Best Model Selection:** We select the best model based on the highest validation accuracy.
* **Optimal Hyper parameters:** The optimal hyper parameters are determined by Keras Tuner and may vary from one run to another. The selected hyper parameters will be used to build the final model.
* **Model Training with Optimal Hyper parameters:** We train the selected model using the optimal hyper parameters for 10 epochs, with a batch size of 10, and a 15% validation split.
* **Model Configurations and Results:** The best model configuration may include a varying number of hidden layers, neurons per layer, and different activation functions. It’s essential to understand that the model’s architecture is determined by the tuning process.

The model achieved an accuracy of 50% on the validation dataset. The training and validation loss trends are plotted to visualize the model’s learning process.

* **Conclusion:** In this document, we have outlined the process of building a deep learning model using Keras Tuner. The model’s hyper parameters have been efficiently optimized to achieve a balance between accuracy and loss, resulting in a high-performing model for the given dataset.

****