SUPERVISED LEARNING – CLASSIFICATION MODEL (LOGISTIC REGRESSION)

Stroke Prediction Model

Prepared By Kehinde

TABLE OF CONTENTS

Contents

Business Understanding

We are building a machine learning model on Stroke predictions for patients. To comprehensively grasp the extent, business rationale, and advantages of developing this model to assist businesses, individuals, and the key sectors that will depend on our machine-learning model, built on historical data introduced for this project, for their daily operations and diagnostics, the following are the key characteristics of this model or project.

Industry

This project is intended for the ***healthcare industry***, with a focus on early detection and prevention of diseases. Stroke, a leading cause of death and disability worldwide, can be effectively addressed with timely intervention to save lives and reduce long-term complications.

Problem Statement

Strokes are a major cause of death and long-term disability worldwide, impacting millions annually. Early detection and prevention are essential for reducing mortality and minimizing impact. Some scientifically proven contributing factors include age, glucose levels, BMI, type of work, heart disease, gender, hypertension, etc. The main idea behind our problem statement is as follows: "***Subsequently, current methods for predicting stroke risk often lack the necessary accuracy and efficiency for timely intervention.***"

What Goals Do We Intend to Achieve?

This project aims to develop a reliable machine-learning model that predicts the likelihood of a stroke, facilitating early detection and preventive care in healthcare settings. We have divided this into four key sections.

Four (4) summaries of our goals.

1. Develop a model that accurately classifies stroke risk based on patient data.
2. Improve healthcare outcomes by ensuring timely interventions for high-risk individuals.
3. Analyze and identify the key risk factors contributing to strokes.
4. Provide a tool that assists healthcare providers in making data-driven decisions concerning patient care.

Why is it Worth Solving?

Strokes are a major cause of death and disability, but many can be prevented through early detection. Accurately predicting stroke risk allows healthcare providers to take timely action, improving patient outcomes and reducing costs. Addressing this issue can save lives and promote healthier living.

Who are the Stakeholders?

The key stakeholders for this project include:

A. **Patients**: Individuals who are at risk of stroke and could benefit from early detection and prevention.
B. **Healthcare Providers**: Doctors, nurses, and healthcare teams who can use the model to make informed decisions and provide better care.
C. **Hospitals and Clinics**: Institutions that can improve patient outcomes and allocate resources more efficiently.
D. **Public Health Organizations**: Groups focused on reducing stroke incidence and improving community health.

Data Collection and Data Understanding

This section presents an overview of the historical data gathered and used for this project.

Data Source

The data was collected from Kaggle, an open-source project datasets hub for data scientists. The data comprises 5,111 rows and 12 columns. Here is a list of the columns or features, along with brief descriptions for each.

1. Id (numerical) – Contains unique identifiers for each row, patient, or observation.
2. Gender (text) – Gender of patients.
3. Age (numerical) – Age of patients.
4. Hypertension (numerical) – Categorical variable describing the hypertension status of patients.
5. Heart disease (numerical) - Categorical variable describing the heart disease status of patients.
6. Ever married (text) – Marriage history of patients.
7. Work type (text) – Patient's employment types.
8. Residence type (text) - Patient's residential address status.
9. Average glucose level (numerical) - Glucose level rates for patients.
10. BMI (numerical) – Body mass index rates for patients.
11. Smoking status (text) – Signals indicating whether patients smoke or not.
12. Stroke (numerical) – Categorizes patients with stroke and not with stroke.

Methodology

In this project, we will be implementing a supervised classification machine learning model. The main concept behind this approach is that our data is more discrete than continuous. It is important

to note that there are numerous classification models available for a data science team to choose from. However, for our purposes, we will focus on using the ***logistics regression classification model***. Throughout this project, we will be working on the Stroke prediction dataset for our analysis.

We will perform data wrangling and exploratory data analysis (EDA) to clean the dataset and analyze each feature's impact on the target variable.

Data Wrangling and Exploratory Analysis

Data wrangling typically means cleaning the data. The steps below are the wrangling steps carried out on the data.

1. We reviewed the data distributions for each feature (columns) and the output (target) of our dataset. This typically involves examining the five-point summary for each column, which includes the minimum value, first quartile (Q1: 25%), median (Q2), third quartile (Q3: 75%), and maximum value (highest value). Additionally, we analyzed the frequency distribution of our features and the output using visual graphs such as histograms.
2. **Removing Irrelevant Columns:** We removed columns that do not contribute to the performance or model predictions. For example:
    a) **id**: The id column is irrelevant to our prediction and can be dropped.
3. **Handling missing values (especially BMI) using imputation of the Mean value**: We filled in missing values using the mean of the BMI column for better model performance.

Image 1 - Removing the ID column

Dropping a column: The ID column isn't useful in developing our model. Let's get rid of it.

```
[10]: df.drop('id', axis = 1, inplace = True)

      df.head()
```

[10]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

Image 2 – Replacing Null values in BMI with the mean

```
[19]: # Third Step, fill the missing rows in the BMI column with the calculated mean.

      df['bmi'].fillna(mean_bmi, inplace=True)

      print(df['bmi'])

      0       36.600000
      1       28.893237
      2       32.500000
      3       34.400000
      4       24.000000
                ...
      5105    28.893237
      5106    40.000000
      5107    30.600000
      5108    25.600000
      5109    26.200000
      Name: bmi, Length: 5110, dtype: float64
```

Data Modeling

Data modeling is the process of using mathematical and statistical techniques to train a machine-learning model that can make predictions. In this project, we use patient data (such as age, medical history, and lifestyle factors) to build a model that predicts the likelihood of stroke. By learning patterns from the data, the model can help healthcare providers identify individuals at higher risk and take preventive actions.

This process involves several steps, which are listed below.

Step 1 - Target (Output) Sampling.

We observed significant differences in the "stroke" column, the target output we want to predict, where 1 indicates a stroke and 0 indicates no stroke. To address this, we used a Random Over Sampler to oversample the minority class (stroke -1) for data balancing, increasing samples in the underrepresented category. This helps create a more even distribution, allowing our model to learn effectively from balanced input.

Image 3 - Applying Random Over-Sampler to Oversample the Minority Class

Using RandomOverSampler to Oversample the Minority Class (stroke = 1)

```
[40]: # 'df' is our DataFrame and 'stroke' is our target column
X = df.drop('stroke', axis=1)  # All Features will remain (except for the 'stroke' column)
y = df['stroke']               # Target column ('stroke')

# Defining the oversampling
ros = RandomOverSampler(sampling_strategy='auto', random_state=42)

# Apply oversampling
X_resampled, y_resampled = ros.fit_resample(X, y)

# Putting resampled arrays back into DataFrame
df_resampled = pd.concat([pd.DataFrame(X_resampled, columns=X.columns), pd.DataFrame(y_resampled, columns=['stroke'])], axis=1)

# Checking the new distribution of target column
print("Original target distribution:")
print(df['stroke'].value_counts())
print("\nResampled target distribution:")
print(df_resampled['stroke'].value_counts())
```

```
Original target distribution:
stroke
0    3814
1     112
Name: count, dtype: int64

Resampled target distribution:
stroke
1    3814
0    3814
Name: count, dtype: int64
```

Step 2: Calculating and Removing Outliers Within Some Features

In this step, we identified and removed outliers from several numerical features (age, hypertension, heart disease, Avg glucose level, and BMI) to improve the quality of the dataset.

1. **Outlier Calculation**: For each numerical feature, we calculated the Interquartile Range (IQR), which is the range between the 25th percentile (Q1) and the 75th percentile (Q3). Outliers were identified as values falling below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR. These outliers were replaced with NaN (null values).

Image 4 – Calculating Outliers

```
[22]: # Calculating our Interquartile and Finding/displaying the Outliers for all numerical columns 1 - 6. All Features

for label in ['age', 'hypertension', 'heart_disease', 'avg_glucose_level', 'bmi']:

    q3, q1 = np.percentile(df[label], [75 ,25])
    iqr = q3 - q1
    lower_lim = q1 - 1.5*iqr
    upper_lim = q3 + 1.5*iqr
    df[label] = df[label].mask((df[label]<lower_lim)|(df[label]>upper_lim), np.nan)
```

2. **Removing Outliers**: After detecting outliers, all rows containing NaN values were removed from the dataset. Finally, we reset the index to maintain consistency after the removal.

By removing these extreme values, we ensure that our model training is less affected by potential anomalies in the data.

Image 5 – Removing Outliers

```
[25]: # Now, we know there are some nan (null values) from our outliers. We will remove the null (nan) values and also reset our column index.

      df.dropna(inplace = True)
      df.reset_index(inplace = True, drop = True)

[26]: # Now, let's see if we have successfully removed the null values (nan).
      df.info()
```

Step 3 - Preparing Data for Modeling

The Third step in our model design phase involves splitting the data into two main groups:

1. **Training Dataset (80%)**: This portion of the data will be used to actively train the model.
2. **Test or Validation Dataset (20%)**: This portion of the data will be used to evaluate the model's performance.

Image 6 – Data Splitting for Our Model (Train & Test)

PART 4: BUILD OUR MODEL

Model Type - Logistic Regression
We are using all independent features of our data to train and predict our target (stroke).

```
[76]: # Define the feature columns and target. Here we are using our resampled dataset.

X = df_resampled[['gender', 'age', 'hypertension', 'heart_disease', 'ever_married', 'work_type', 'Residence_type', 'avg_glucose_level', 'bmi', 'smoking_
y = df_resampled[['stroke']]

# We are One-hot encoding categorical columns and also fitting Logistic Regression in a pipeline
pipeline = Pipeline([
    ('preprocessor', ColumnTransformer(transformers=[
        ('cat', OneHotEncoder(drop='first'), ['gender', 'ever_married', 'work_type', 'Residence_type', 'smoking_status'])
    ], remainder='passthrough')),
    ('classifier', LogisticRegression(solver='liblinear'))
])

# Split resampled data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)

# Train the model and make predictions
pipeline.fit(X_train, y_train)
y_pred = pipeline.predict(X_test)

# Evaluate the model
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Step 4 – Machine Learning Model and Why is it important?

For this project, we developed a ***Logistic Regression classification model*** that utilizes patient data to predict the likelihood of stroke in future patients, ultimately aiming to save lives.

Why is it important?

The Logistic Regression model is vital for this project as it predicts the likelihood of stroke in patients using key health indicators. By identifying high-risk individuals early, healthcare providers can implement preventive measures. This model excels in binary classification, determining if the outcome is a stroke (1) or no stroke (0). Its simplicity and effectiveness in analyzing patient features make it a valuable tool for informed healthcare decisions.

Results

The model was developed using historical data from stroke patients, comprising 11 columns to accurately predict stroke outcomes. Currently, the model achieves an accuracy of 78%.

Breakdown of Model's Performance

This is a complete overview of the model's performance including computation speed, accuracy, and other analysis.

Computation & Accuracy.

The model takes a few seconds to compute, with a storage size of 1 MB. It's important to note that the faster the execution speed, the better the model performs. Currently, the model achieves an accuracy of 78% which has been achieved through continuous iterations and parameter tuning.

Image 7 – Model's Accuracy Report

```
Accuracy: 0.78

Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.77      0.77       766
           1       0.77      0.79      0.78       760

    accuracy                           0.78      1526
   macro avg       0.78      0.78      0.78      1526
weighted avg       0.78      0.78      0.78      1526
```

Analysis

Next, we will provide a step-by-step summary of our model's performance.

A. Target Stroke Confusion Matrix

The confusion matrix below shows how well our model predicted stroke outcomes. It has two key sections:

1. The top-left box shows 586 correct predictions for patients who did not have a stroke (True Negatives).
2. The bottom-left box shows 162 cases where the model missed predicting a stroke (False Negatives).

This matrix highlights that while the model performs well in predicting no-stroke and stroke cases, it still misses some stroke cases, which will require further improvements.

Image 8 - Confusion Matrix

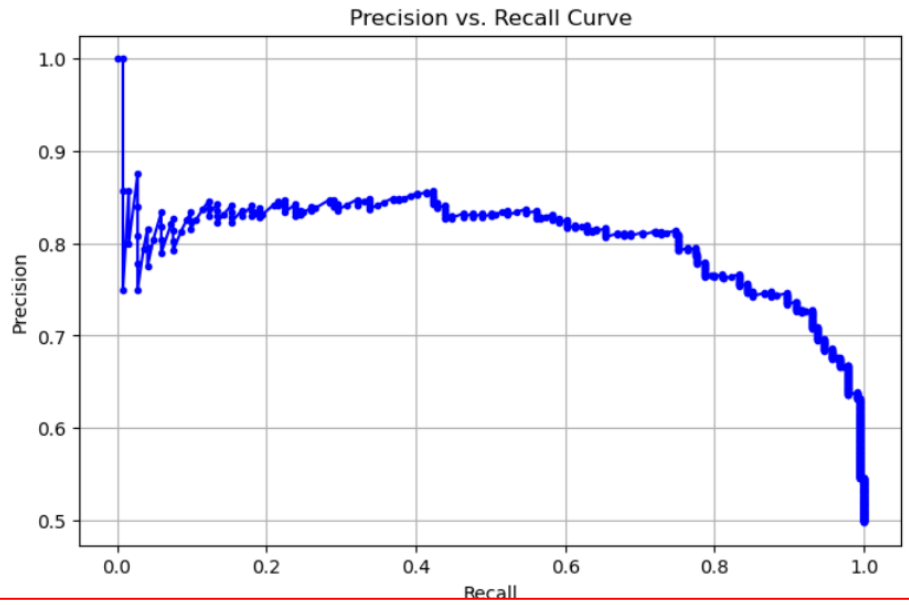
B. Target Stroke Precision vs. Recall

The Precision-Recall Curve shows the trade-off between precision and recall for our stroke prediction model. Precision indicates the percentage of actual strokes among those predicted, while recall measures the model's ability to identify all stroke cases.

As recall increases—meaning more strokes are correctly identified—precision often decreases, resulting in more non-stroke cases being misclassified as strokes. Finding the right balance between these metrics is crucial in healthcare, particularly when it comes to minimizing missed stroke cases. At the same time, it's beneficial that the model can predict a significant number of patients who actually have strokes.

Image 9 - Precision vs. Recall

```
# Plotting the Precision-Recall Curve
plt.figure(figsize=(8, 5))
plt.plot(recall, precision, marker='.', color='blue')
plt.title('Precision vs. Recall Curve')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.grid(True)
plt.show()
```



Conclusion

In this project, we developed a machine learning model using logistic regression to predict stroke risk based on patient data. By focusing on key health indicators such as age, glucose levels, and medical history, the model aims to assist healthcare providers in making informed decisions, enabling early detection and prevention of strokes. Our approach involved thorough data cleaning, balancing the dataset, and removing outliers to improve model performance. With an accuracy of 78%, this model shows promise in helping reduce stroke-related mortality and disability. However, further optimization is required to improve its ability to catch all potential stroke cases.