MZ20177 (5).docx

Turnitin

Document Details

Submission ID

trn:oid:::20206:419571529

Submission Date

Jan 7, 2025, 10:28 AM UTC

Download Date

Jan 7, 2025, 10:29 AM UTC

File Name

MZ20177_5.docx

File Size

1.5 MB

20 Pages

3,234 Words

18,055 Characters

0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Detection Groups

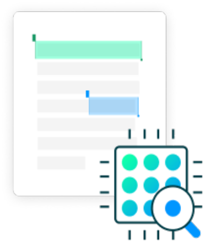
1 AI-generated only 0%
Likely AI-generated text from a large-language model.

2 AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.



False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

PROGRAMMING FOR DATA ANALYTICS AND AI

Student ID-

Table of Contents

Introduction

In this project, we are discussing a very crucial and fatal physical condition called stroke. This fatal physical problem works with brain blockage. In this data set, we will work with different machine learning algorithms like (regression, classification, and clustering algorithms).To this day, one of the major causes of death in today's society worldwide remains stroke. The World Health Organization (WHO) accords that stroke accounts for approximately 11 percent of all health issues and 11 percent of all deaths. This research examines data collected from 5,110 patients about various factors including health, and lifestyle. The main goal of this dataset is to analyse certain variables such as body mass index (BMI) and glucose levels along with stroke occurrence predicting factors.

The objectives of this report:

Apply visualizations and simple statistics for descriptive analytics of the dataset in order to obtain its characteristics. Prepare the dataset to be used with the algorithms by including feature engineering, normalizing data, and removing any missing values. Carry out a thorough analysis of several machine learning algorithms with classification analysis in order to predict the probability of having a stroke. Use a regression model to estimate one's BMI given the other health characteristics. Explore groupings and hidden patterns in the data by investigating clustering techniques. The report is analysed in such a way that its distribution presentation becomes easy. Descriptive analytics follows this introduction by describing the dataset. Data preparation techniques include many ways of cleaning and preparing the data. Later, classification, regression, and clustering analyses are conducted with each one presenting methodologies, outcomes, and discussions of all results. This report finally ends with a discourse on the main findings, perceived ramifications, and suggestions for further work.

Important libraries:

For numerical computations and data manipulation: NumPy: NumPy is a library that is open source and free to use for the Python programming language that is used to implement arrays in data analysis in this project, it also specializes in for multi-dimensional arrays and mathematical functions Pandas: Pandas is a popular, open-source software in Python. It mainly works on data cleaning and also for data analysis and manipulation in Python Matplotlib: Pandas is a popular, open-source library in Python. It mainly works on data representation. Machine learning libraries: scikit-learn: Scikit-learn is a comprehensive library machine learning library, which includes tools for: Model Selection and

Preprocessing: train_test_split (splitting data), StandardScaler (feature scaling). Classification Algorithms: LogisticRegression, RandomForestClassifier, SVC (Support Vector Classifier), GradientBoostingClassifier.

Regression Algorithms: Linear regression: a statistical is a regression method used for predicting a dependent variable which are continuous on the basis of independent variables(predictor).

Clustering Algorithms:

K-Means A clustering technique is an information retrieval method that is part of unsupervised learning algorithms, which categorizes a given dataset into different clusters. It is actually an iterative algorithm that separates the complete dataset into k distinct clusters in a manner that any single dataset is a member of only one cluster, which is similar to it. k different clusters in a way that each dataset belongs to only one group that has similar properties.

It helps us group the data into different clusters and it is one of the simplest form of self-organizing that can help us to find out what the form of the clusters of the groups in the dataset are without need for training.

It is an algorithm that is based on centroid, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

Evaluation Metrics:

accuracy_score,

confusion_matrix,

classification_report,

mean_squared_error,

r2_score.

Utility Libraries

math: This library provides standard mathematical functions.

warnings: Used for managing warnings that might arise during code execution.

Descriptive Analysis of the 'Stroke' Dataset

The core of this project is a dataset called "healthcare-dataset-stroke-data.csv" containing masses of information relating to individuals and their health features. Maybe that dataset has come from some sort of healthcare service or perhaps there is some other public availability point. Before any kind of analysis of it, there are a couple things to appreciate its structure and contents.

Overview of the Dataset

The Stroke Dataset contains the information to detect potential patterns and factors related to stroke occurrences. Both numerical and categorical attributes are included: Numerical Attributes: Age, BMI (Body Mass Index), average glucose level. Categorical Attributes: Gender, smoking status, work type, residence type, and whether or not the individual has ever had a stroke.

The data is analysed for basic statistical properties, trends, and visualization of key characteristics in the data.

We first used some descriptive analytics in order to explore the data by first checking how many rows and columns were within the dataset; data types used, and all descriptive statistics describing each feature are presented below as steps taken **Data Loading and Exploratory Analysis**: The dataset was loaded in Python using pd.read_csv() and other functions such as head(), tail(), columns, and info() were used to explore a broad outline of its structure and the types of data present. **Descriptive Statistics:** The describe() function was used to calculate simple statistics about the data distribution, including mean, standard deviation, min, max, and quartiles for numerical features.

Histograms: Histograms were created for each feature to plot the data distribution. The boxplots helped identify potential outliers and understand the spread of data. These visualizations provided insight into the characteristics of individual features. •Correlation Matrix: A correlation matrix was calculated to understand the interdependencies between the numerical features. This helped to identify potential multicollinearity between variables.

Basic Statistical Analysis

Numerical Attributes

Age: The dataset consists of people from the youngest children to the oldest in the population. The mean and median values would give a picture of the central tendency of the dataset, whereas the standard deviation would show variability.
BMI: Body mass index is also important. Summary statistics would tell us the general health profile of people.

Mean Blood Glucose Level: High blood glucose level is considerably linked with stroke risk. Descriptive statistics are useful for rendering a sense of glucose profile from this dataset.
Categorical Attributes
Gender: This dataset covers an equal number of both genders or shows gender imbalance .gender is categorized into "male", "female" etc.

Smoking Status: Under this attribute Analyses the Categories are "never smoked", "previously smoked", and "smoking currently", to distinguish both smoker and non-smoker individuals
Stroke Cases: In this attribute Analyses the proportion of stroke vs. non-stroke cases. This is an essential category from where it's easily the proportion of non-stroke cases and their conditions

Identification of Outliers

Outliers can significantly impact the analysis. Box plots are used to identify extreme values in numerical attributes, such as:

Age: Older peoples with unusually high ages could be outliers.
BMI: Very high or low BMI could be outliers or a few data points.
Average Glucose Level: Extremely high glucose level might imply an extreme illness. Outliers found in the above attributes are highlighted and explained in relation to the implications they would have on the data.

Graphical Representation

Histogram

Histogram is the graphical representation of of data. For this project, several histograms are added to make the project more understandable with many valuable insights .Here is an example -

• **Age Group Distribution:** Histogram presents the count for each age group. This can be used to establish the stroke prevalence within a particular age group. Scatter

Plot

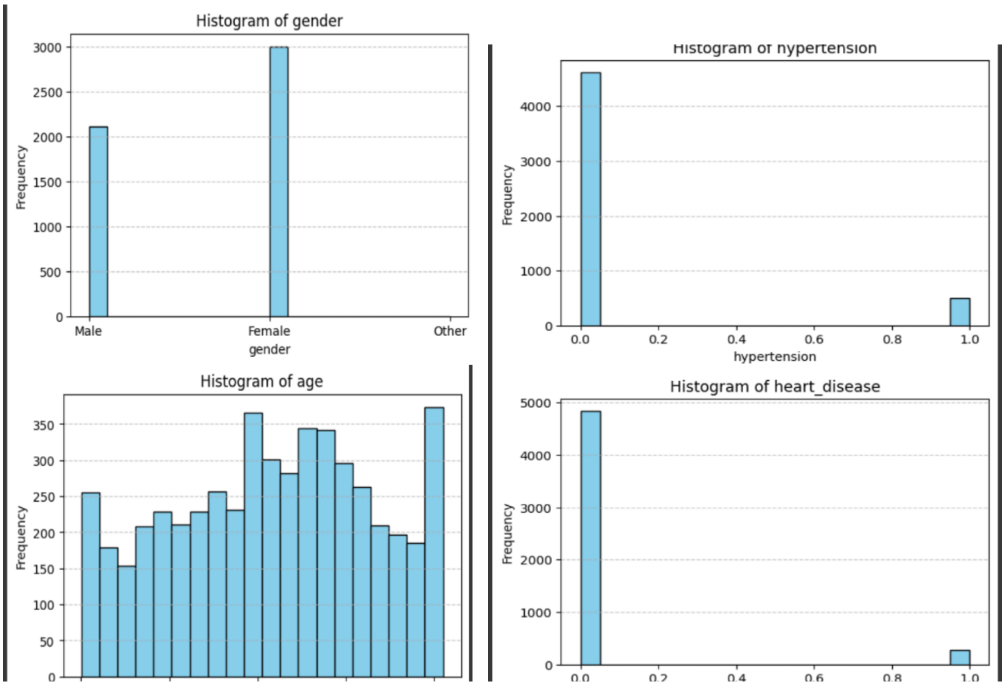


Figure 1histograms

(Source: Self-Created)

Scatter plot

Scatter plots are used to showcase the potential relationship between columns. several scatter plots are also used in this project for showing the distinct relationships. 5. Data Characteristics

```
# Now select numeric columns only for correlation
numeric_columns = df.select_dtypes(include=['int64', 'int32', 'float64'])

# Compute the correlation matrix
correlation_matrix = numeric_columns.corr()

# Display the correlation matrix
print(correlation_matrix)
```

```
                         id       age  hypertension  heart_disease  \
id                 1.000000  0.003538      0.003550      -0.001296
age                0.003538  1.000000      0.276398       0.263796
hypertension       0.003550  0.276398      1.000000       0.108306
heart_disease     -0.001296  0.263796      0.108306       1.000000
avg_glucose_level  0.001092  0.238171      0.174474       0.161857
bmi                0.003084  0.333398      0.167811       0.041357
stroke             0.006388  0.245257      0.127904       0.134914

                   avg_glucose_level       bmi    stroke
id                          0.001092  0.003084  0.006388
age                         0.238171  0.333398  0.245257
hypertension                0.174474  0.167811  0.127904
```

Figure 2: correlation matrix

Correlations

Age and Stroke Cases: It seems older patients are more prone to having strokes. Smoking Status and Stroke: Smokers may be associated with more strokes. BMI and Glucose Levels: The relationship of BMI and glucose levels is also discussed Imbalances and Anomalies

The data could be imbalanced regarding stroke cases or other categorical attributes, so these would have to be adjusted during modelling.Missing values in the attributes, such as BMI, may skew analysis.

Key Findings

Aging above a certain age is more likely to lead to stroke. High levels of glucose are found to relate to stroke. Smoking seems to be a very crucial risk factor. Outliers in both BMI and glucose levels indicate that data preprocessing should be done with extra care.

The descriptive analysis lays down a base for further explorations to be done in future sections about data preparation, classification, regression, and clustering.

Data Preparation

Data Cleanings

Data cleaning is one of the major steps to ensure that the quality and reasonability of the analysis are proper. We performed the following data-cleaning processes: **Missing Value Handling:** starting with replacing missing values in the 'BMI' column with the mean value of the column. The method is one of the most frequently used methods to handle missing data without losing the overall distribution.

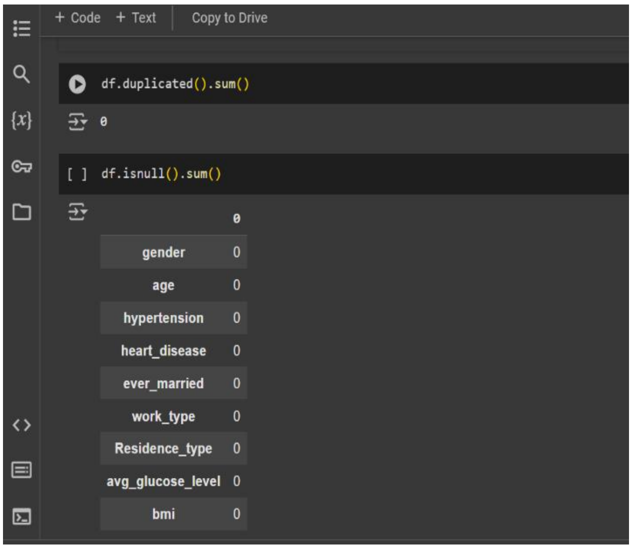


Figure 3: data cleaning 1

Removing Duplicate Data Rows: The duplicate rows present in the data had been removed as there was an execution of duplicated() function in it. As a result, no duplicate data existed that could contribute bias.

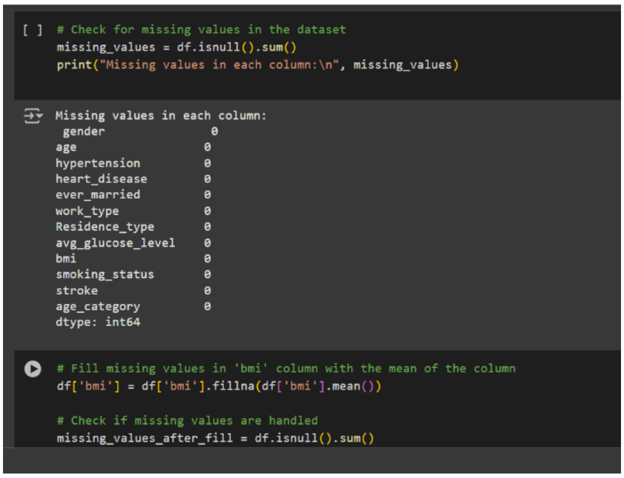


Figure 4: data cleaning 2

Delete Unused Columns: The 'id' column would likely be one and only an identification number to a person in that dataset hence could be completely neglected for that predictive task.

Feature Engineering. Unused columns can create a nuisance while running through algorithms.so deleting them was crusial work to be done.

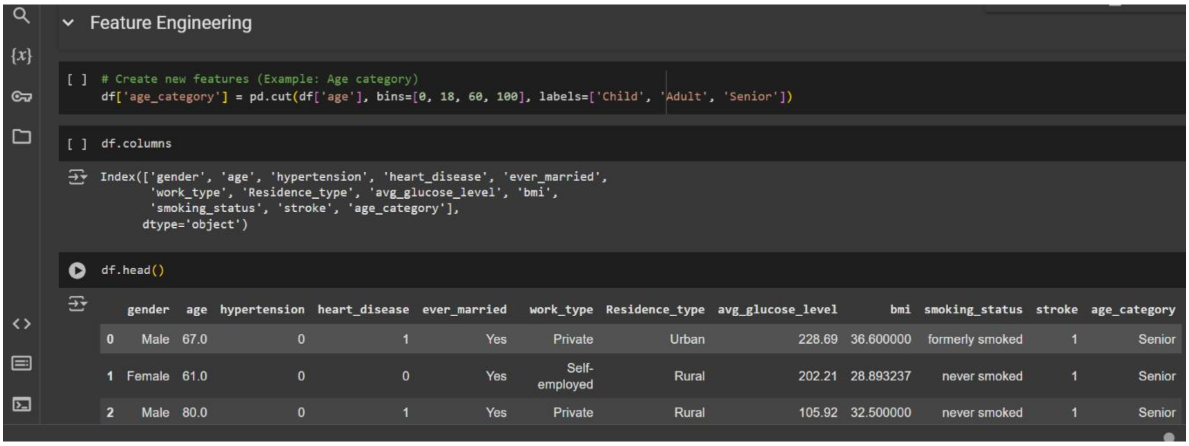


Figure 5: data cleaning 3

Feature engineering actually generates new features from existing ones. This is all done to further improve the model's performance. To be specific to this project, we have thought of an 'age category' feature that caters to making the age column into three types: 'Child', 'Adult', and 'Senior'. New feature thus possibly gives a more representative view to age for prediction of strokes.

Classification Analysis

This section, mainly used three machine learning algorithms: the first one is Logistic Regression, the second one is Random Forest Classifier and the third one is Support Vector Machine (SVM). The main goal of this section is to predict a patient, getting a stroke based on a set of health and demographic features.

Algorithm Used

In this portion there are three algorithms are used.

First of all, used Logistic Regression, which is the linear model, in binary classification tasks this algorithm is being used. It is used in estimating the probability of a binary outcome, setting a logistic curve to the data.

And the second one is Random Forest Classifier, which is an advanced type model which combines many decision trees for making more accuracy in predictions. Also it known for handling the large amount of datasets including avoiding overfitting.

And Third one is Support Vector Machine, which is a very powerful algorithm for finding optional hyperplane that mainly separates the classes. Also it can handle separable data but this data have to be non-linear.

Model Training Evaluation

The given datasets were already preprocessed by handling missing values, encoding categorical variables, and normalizing the feature set using StandardScaler to ensure the compatibilities with the algorithms. Then split the dataset into training 80% and testing 20% sets using train_test_split. All the models were trained by the training sets and then take it for testing on the unseen test sets. For evaluating the models used several matrices, including accuracy, precision, recall F1-score, confusion matrix. That means correct prediction percentages, also used some matrices for understanding the matrixes mainly when to dealing with imbalance data. Also used some table, for showing the models are giving the correct or incorrect prediction.

Logistic Regression:

```
# Initialize the Logistic Regression model
log_reg = LogisticRegression()

# Train the model
log_reg.fit(X_train, y_train)

# Make predictions
y_pred_logreg = log_reg.predict(X_test)

# Evaluate the model
print("Logistic Regression Accuracy: ", accuracy_score(y_test, y_pred_logreg))
print("Logistic Regression Report:\n", classification_report(y_test, y_pred_logreg))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_logreg))
```

Figure 6:Logistic Regression Code

Source: Self-Created

Random Forest Classifier:

```
# Initialize the Logistic Regression model
log_reg = LogisticRegression()

# Train the model
log_reg.fit(X_train, y_train)

# Make predictions
y_pred_logreg = log_reg.predict(X_test)

# Evaluate the model
print("Logistic Regression Accuracy: ", accuracy_score(y_test, y_pred_logreg))
print("Logistic Regression Report:\n", classification_report(y_test, y_pred_logreg))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_logreg))
```

Figure 7:Random Forest Classifier Code

Source: Self-Created

Support Vector Machine:

```
# Initialize the Support Vector Machine model
svm_clf = SVC(kernel='linear', random_state=42)

# Train the model
svm_clf.fit(X_train, y_train)

# Make predictions
y_pred_svm = svm_clf.predict(X_test)

# Evaluate the model
print("SVM Accuracy: ", accuracy_score(y_test, y_pred_svm))
print("SVM Report:\n", classification_report(y_test, y_pred_svm))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_svm))
```

Figure 8: Support Vector Machine Code

Source: Self-Created

Result and Discussion

Logistic Regression:

This logistic regression model gives almost 93.93 % accuracy and the classification report are given bellow for more better understanding of the result

Accuracy: 93.93%

```
Logistic Regression Accuracy:  0.9393346379647749
Logistic Regression Report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       960
           1       0.00      0.00      0.00        62

    accuracy                           0.94      1022
   macro avg       0.47      0.50      0.48      1022
weighted avg       0.88      0.94      0.91      1022

Confusion Matrix:
 [[960   0]
 [ 62   0]]
```

Figure 9: Logistic Regression Output

Source: Self-Created

This logistic regression model achieved an accuracy of 93.93% but also it unable to describe few predictions correctly to the stroke cases for class 1. It reflected in the poor recall and precision for predicting stroke. Although, in this model favoured the majority class of no stroke because of imbalance in the data.

Random Forest Classifier:

This random forest classifier model gives almost 93.93 % accuracy and the classification report are given bellow for more better understanding of the result

Accuracy: 93.93%

```
Random Forest Accuracy:  0.9393346379647749
Random Forest Report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       960
           1       0.00      0.00      0.00        62

    accuracy                           0.94      1022
   macro avg       0.47      0.50      0.48      1022
weighted avg       0.88      0.94      0.91      1022

Confusion Matrix:
 [[960   0]
 [ 62   0]]
```

Figure 10: Output

Source: Self-Created

Like the pervious Logistic Regression, the Random Forest Classifier also achieved an accuracy of 93.93%, model but also it unable to predictions correctly to the stroke cases but this model performed good on the majority class, no stroke class, but struggling with the minority class means the stroke class.

Support Vector Machine:

This logistic regression model gives almost 93.93 % accuracy and the classification report are given bellow for more better understanding of the result

Accuracy: 93.93%

```
SVM Accuracy:  0.9393346379647749
SVM Report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97       960
           1       0.00      0.00      0.00        62

    accuracy                           0.94      1022
   macro avg       0.47      0.50      0.48      1022
weighted avg       0.88      0.94      0.91      1022

Confusion Matrix:
 [[960   0]
 [ 62   0]]
```

Figure 11: Output

Source: Self-Created

Like the pervious model Logistic Regression, the Random Forest Classifier, support vector machine also achieved an accuracy of 93.93%, model but also this model not predicting the stroke cases correctly. But this model mainly focused on the class of majority means the no-stroke class, it leading foor performance to the minority class.

Comparison:

All these three models Random Forest Classifier, Logistic Regression, and the Support Vector Machine all are achieved the high, overall accuracy of 93.93%. also all of them struggling to predict correctly in stroke cases because of the imbalance in the datasets, where there are more no-stroke cases than stroke cases. although all of the model achieved the high accuracy but all were poor to predicting the stroke cases. All because of the misleading datasets.

```
# Collect accuracy scores
model_names = ['Logistic Regression', 'Random Forest', 'SVM']
accuracy_scores = [
    accuracy_score(y_test, y_pred_logreg),
    accuracy_score(y_test, y_pred_rf),
    accuracy_score(y_test, y_pred_svm)
]

# Plot the accuracy comparison
plt.figure(figsize=(10, 5))
sns.barplot(x=model_names, y=accuracy_scores)
plt.title('Accuracy Comparison of Models')
plt.ylabel('Accuracy')
plt.show()
```

Figure 12: Comparison Code

Source: Self-Created

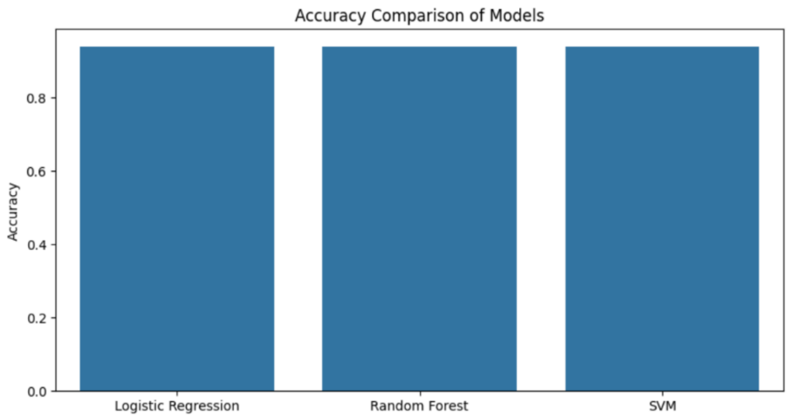


Figure 13: Bar Chart

Source: Self-Created

In conclusion, all of the model performed well for the majority class means the no stroke class. But all are unable to predict the stroke cases effectively. Due to the imbalance issue is very crucial, although it should be improved for better performing the models.

Regression Analysis

In this section, used liner regression for predicting the BMI (Body Mass Index) based on various factor like age hypertension, heart disease, glucose levels, and more. Linear regression is technique which tries to found out the best line which explain the relationship between the BMI and the other features.

Algorithm Used

Here Linear Regression uesd, which is a very simple and common algorithm to predict a continuous value. The main goal is to predicting a person's BMI using the given features, such as age, gender, and health conditions.

Model Training Evaluation

Firstly, prepared the data by cleaning it, and do all necessary things to then training the model from the given datasets. Basically, in two matrices one is mean squared error and the second is R-squared. These two matrices are told how the model does well with the data. High value means a better fit, it would be ideal if it closer to 1.

```
# Initialize and train the Linear Regression model
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# Make predictions on the test set
y_pred = lin_reg.predict(X_test)

# Evaluate the model using R-squared and Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared: {r2}")

# Print the model's coefficients
print("Model Coefficients:")
for feature, coef in zip(X.columns, lin_reg.coef_):
    print(f"{feature}: {coef}")
```

Figure 14: Linear Regression

Source: Self-Created

Result and Discussion

The MSE was found to be 0.0516. This means that on average, the model is predictions are fair and also close to the actual BMI values, though there is still some error. The R-squared value of this model was 0.095. That means the model explains only **9.53%** of the variation in BMI. This is quite low, suggesting that the model doesn't fully capture the factors that influence BMI. There could be other important factors missing from the data.

```
Mean Squared Error (MSE): 0.05155500489618247
R-squared: 0.09528935258762994
Model Coefficients:
id: 0.0027718306777794193
age: 0.060767424276923825
hypertension: 0.009125447797386362
heart_disease: 0.010316404787004541
avg_glucose_level: 0.011261978867839131
bmi: -0.00262415797276628
gender_Male: -0.0010887632767394165
gender_Other: -2.2551405187698492e-17
ever_married_Yes: -0.014019896722245299
work_type_Never_worked: 0.0010347332226965623
work_type_Private: 0.007356269645908101
work_type_Self-employed: -0.002606645858377041
work_type_children: 0.012892598455574797
Residence_type_Urban: 0.0017811331429334585
smoking_status_formerly smoked: -0.0014430719680910843
smoking_status_never smoked: -0.0022994187426371064
smoking_status_smokes: 0.001321226031839634
age_category_Adult: -0.012182034896790352
age_category_Senior: -0.002173255333258295
```

Figure 15: Output

Source: Self-Created

These coefficients are told us about how each factor affects the BMI. For example, as the age increases, BMI tends to increase. On the other hand, being male or being married seems to slightly lower BMI in this dataset. This model's R-squared value of 9.53% shows that it doesn't explain much of the variation in BMI. This means that there are probably other important factors for affecting BMI which are not included in this model. The Mean Squared Error is relatively small, but there is still hope for improvement.

Clustering Analysis

In this segment, the K-Mean clustering algorithm is oredered in order to cluster the data based on the various similar attributes. This one is the types of clustering that assist in identifying the patterns within the data without making use of the specified labels.

Algorithm Used

The algorithm used for this task is K-Means Clustering. K-Means is a popular clustering algorithm that separate the data into K clusters.

1. Randomly selecting the K initial of centroids.
2. Assigning each of the data point to the nearest centroid.
3. Recomputing the centroids based on the current assignments.

4. Repeating steps 2 & 3 until the convergence when the centroids no longer move significantly.

Model Training Evaluation

```
# Standardize the numerical features
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)

# Apply K-Means clustering
kmeans = KMeans(n_clusters=3, random_state=42)  # You can try different numbers of clusters
kmeans.fit(df_scaled)

# Get cluster labels for each data point
cluster_labels = kmeans.labels_

# Evaluate the clustering using the Within-Cluster Sum of Squares (WCSS)
wcss = kmeans.inertia_
print(f"Within-Cluster Sum of Squares (WCSS): {wcss}")
```

Figure 16: K-Means

Source: Self-Created

For Applying the K-Means, it needs to perform, Data Preprocessing: where the data was cleaned. Second the Choosing the Number of Clusters (K), for determining an optimal number of the clusters. Also, in this method involving in plotting the Within-Cluster Sum of Squares against different values of K. And lastly the Model Fitting where the K-Means algorithm was then trained on the dataset to intending the clusters.

Result and Discussion

```
Within-Cluster Sum of Squares (WCSS): 83016.06785252143
```

Figure 17: Output

Source: Self-Created

After running the K-Means clustering algorithm with the no of clusters, obtained the Sum of Squares value of 83016.07 Within-Cluster. This value indicates the compactness of the clusters. A smaller WCSS suggests that the data points within each cluster are closer to the centroid, implying better clustering. And with the high WCSS suggests that the clusters are less well-defined, and the model should be improved. The relatively high WCSS value of 83016.07 suggests that the clusters are somewhat spread out. While K-Means works well when the clusters are distinct and well-separated. for improving the results, we could experiment with different values of K.

Conclusion

In this analysis, several machine learning tasks are performed and precdict the outcomes using the different techniques. First of all, applied the three machine learning algorithm in the classification on stroke data. All the three model achieved the same accuracy of **93.93%**. although, despite the high accuracy, the classification reports told us that both precision and recall for predicting the stroke class one were zero, indicating that the models failed to identify correctly for stroke cases. This suggests that the dataset may be imbalanced. On the Linear Regression model was used to predict **BMI** based on various factors like age, gender, smoking status, and more. This model achieved a Mean Squared Error value of 0.0516 and a R-squared value of 0.0953. The mainly low R-squared value means the model does not explain much of the variability in BMI. And in the last K-Means clustering for grouping the data points that based on similarities in the features. The Within-Cluster Sum of Squares was calculated to be 83016.07. This suggesting that the number of clusters chosen might not have been optima. In conclusion, while many models are used in this analysis that provides many useful insights, there is room for improvement.

References

"Data Analysis Tutorial." *GeeksforGeeks*, 5 May 2023, www.geeksforgeeks.org/data-analysis-tutorial/.

Syedkamranhashmi. "Performing Exploratory Data Analysis on Stroke Dataset." *Geek Culture*, 17 May 2021, medium.com/geekculture/performing-exploratory-data-analysis-on-stroke-dataset-1b4885989030.

"What Is Data Analysis?" *GeeksforGeeks*, 20 Sept. 2021, www.geeksforgeeks.org/what-is-data-analysis/.