



IN3062: Introduction to Artificial Intelligence Coursework Stroke Prediction

By
Abarna Ahilan
Rahim Ahmed
Shajeevan Chandrasekaran

https://github.com/ItsRahim/IN3062_CW

| | |
|---|----|
| Contents | |
| What is your dataset, problem domain? | 4 |
| Define questions and analysis tasks | 5 |
| Initial investigation of the dataset and the characteristics of the data | 5 |
| Plan as to how you might transform the data to make it useable | 6 |
| Is your model classification or regression? | 6 |
| Did you have any missing, corrupt, or misleading data? If so, how did you cope with it? | 6 |
| Have you omitted some data? If so, why? | 6 |
| Did you apply techniques to understand your dataset? | 6 |
| What models did you use? | 8 |
| Analysis and Results/Outputs | 8 |
| Decision Tree | 8 |
| Random Forest | 9 |
| Logistic Regression | 9 |
| Naïve Bayes | 10 |
| Did you have any problems or difficulties working with the dataset? | 10 |
| SMOTE | 11 |
| Confusion Matrix Models | 11 |
| Accuracy Metrics Comparison (weighted average used) | 11 |
| Evaluation | 11 |
| Conclusion | 12 |
| References | 12 |

What is your dataset, problem domain?

The dataset used for this project will be the dataset from the website known as Kaggle (<https://www.kaggle.com/datasets>). The specific dataset used from Kaggle is the stroke dataset (<https://www.kaggle.com/fedesoriano/stroke-prediction-dataset>). The dataset provides information regarding different types of factors that can influence the cause of stroke. It is a large dataset that initially contains 12 attributes and approximately 5000 observations. This means that there is a lot of data to work with.

The problem domain the project is focusing on is predicting stroke based on the different attributes that the dataset presents, and the valuable data included. There are some columns and rows that are required to be removed and edited to increase the accuracy of the prediction through means such as removing unnecessary data and replacing null with data acquired from the median amount. This will further increase the accuracy of the prediction regarding the problem domain.

Define questions and analysis tasks

The domain is focused on predicting stroke based on various factors such as age, gender, diseases, and if the person is a smoker or not. These factors can cause a stroke. The reason for choosing stroke as the problem domain out of the similar issues is because stroke is one of the highest leading causes of death in the world. By undergoing stroke prediction, doctors can find out the chances of someone getting a stroke in the earlier stages. This will provide them with a foundation to apply more advanced checks on the patients to rapidly help them with recovery.

The analytical questions that are being asked regarding the stroke prediction are:

- What factors can increase the likeliness of stroke?
- What is the cause of stroke in individuals?
- Can chances of stroke be identified at an earlier stage?
- What can people do to reduce the chances of stroke?
- How does smoking affect the chances of a stroke?

The objectives are as followed:

- To create clear visualizations for the user to clearly understand the dataset
- For the machine to predict, as accurately as possible, whether a person is at risk of a stroke given variables such as smoking, age, BMI, etc.
- To use the correct and most accurate machine learning model to train the stroke dataset i.e., Linear Regression, Random Forest Regression, Clustering, or Classification.

The expected outputs are:

- From the dataset, we have 95% of the patients within it are unhealthy and are at risk of a stroke. In a perfect scenario, we would expect the AI to successfully predict to a degree of accuracy as close to 95% as possible.
- The real accuracy of our machine learning will be illustrated through a graph that would show actual vs predicted data

Initial investigation of the dataset and the characteristics of the data

The dataset used includes 12 columns of data in which after discussing, we decided that most data was useful, but some were not required for the prediction as it would not affect the accuracy and precision of the prediction if it were not included. For example, the column which had data on if the person was married or not, was not required as stroke is based on many individual factors and not an external factor such as being married. Similarly, the work type of the patient was not required as the data was general and not specific enough to affect the prediction. By being self-employed or working in a private field was not useful as there were no specific jobs. However, the other data were key information regarding predicting strokes such as smoking status, BMI, age, gender, and glucose levels. The characteristics of the data in the dataset are that it consists of numerical figures as well as descriptions. For example, the BMI and if a person is a smoker or not.

Plan as to how you might transform the data to make it useable

Some data were either null or missing. To make these useful for the prediction of stroke these data will be transformed. In some cases, some figures were missing from the body mass index which we decided to use the median as it provides better precision and accuracy rather than using the mean. This will allow the end prediction to be more accurate. The data from the columns which are not required will be removed from the dataset as it can have negative influence on the prediction. The data such as the ID from the dataset will be removed as it is not useful as we are only focusing on the factors which affect stroke.

Is your model classification or regression?

The model we use is classification because we want to predict a discrete outcome-based of multiple variables through unsupervised learning. We also split the dataset into training and test samples through train test split as well as improving the models as we go when we test the outcomes.

We cannot use regression as a machine learning model such as linear regression because linear regression focuses on one variable whereas ours uses multiple independent variables to predict an outcome. Regression provides outputs that are continuous which our model does not. Also, linear regression is supervised learning, and we are trying to implement unsupervised learning.

Did you have any missing, corrupt, or misleading data? If so, how did you cope with it?

Within the BMI column, there were several records or 'N/A' values for the BMI. It is always better to have more data to train the AI and because we had already removed the unknown smoking status. We believed it would have been better to replace unknown values with the median.

Have you omitted some data? If so, why?

We have omitted data such as ID, Work Type, and Residence Type as we believe that these are factors that do not affect the risk of strokes. Whilst cleaning the data we had noticed there were values such as the Smoking Status being 'Unknown' and Gender being 'Other'. We removed that as this does play a very large factor in stroke and it is an unknown would cause inaccuracy whilst training.

Did you apply techniques to understand your dataset?

The first step we took to understanding our dataset was using a confusion matrix to visualize how smoking and gender impacts the likelihood of having a stroke. This matrix allowed us to easily picture the dataset and better understand that both smoking and gender influenced the probability of stroke according to our data. Using this matrix, we were able to find out that the highest percentage of people with a stroke in our dataset were females that had never smoked at 45% along with males that had quit smoking at 33%.

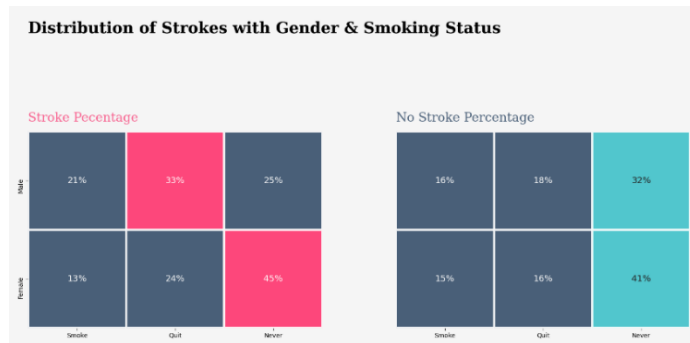


Figure 1 - Confusion matrix

Along with the matrix, we also loaded the data in a data structure using pandas Data Frame. With this table, we were able to analyze the data by splitting the data into mean, minimum, maximum, etc. The table shows the mean age of people in the data is 48 yet has a wide range of age from 10 to 82 years old. We discovered that for the mean from the data, it can be assumed that most people have not smoked since the smoking status is around -0.02.

| | gender | age | ... | smoking_status | stroke |
|-------|-------------|-------------|-----|----------------|-------------|
| count | 3565.000000 | 3565.000000 | ... | 3565.000000 | 3565.000000 |
| mean | 0.605330 | 48.860309 | ... | -0.026648 | 0.056662 |
| std | 0.488848 | 18.873140 | ... | 0.684621 | 0.231228 |
| min | 0.000000 | 10.000000 | ... | -1.000000 | 0.000000 |
| 25% | 0.000000 | 34.000000 | ... | 0.000000 | 0.000000 |
| 50% | 1.000000 | 50.000000 | ... | 0.000000 | 0.000000 |
| 75% | 1.000000 | 63.000000 | ... | 0.000000 | 0.000000 |
| max | 1.000000 | 82.000000 | ... | 1.000000 | 1.000000 |

Figure 2 - Data as a table (features matrix)

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
df = pd.read_csv('stroke_data.csv')

# Display the first few rows
df.head()

# Check the data types
df.dtypes

# Check for missing values
df.isnull().sum()

# Convert the 'stroke' column to a categorical variable
df['stroke'] = df['stroke'].astype('category')

# Create a confusion matrix
cm = pd.crosstab([df['gender'], df['smoking_status'], df['stroke']],
                 df['gender'], df['smoking_status'], df['stroke'])

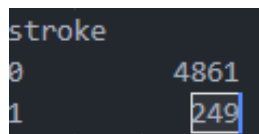
# Print the confusion matrix
print(cm)

# Create a heatmap of the confusion matrix
sns.heatmap(cm, annot=True, cmap='magma')
plt.show()

```

Figure 3 - (code used for printing the data into the matrix)

We also printed the stroke vs non-stroke percentage and noticed that we had a much larger number of patients without stroke than with stroke.



| | | |
|--------|------|--|
| stroke | | |
| 0 | 4861 | |
| 1 | 249 | |

Figure 4 - Stroke/Non-Stroke

What models did you use?

Since we're predicting whether someone is more likely to get a stroke using our dataset, we, therefore, have a classification problem. To handle this problem, we will use several models including decision tree, random forest, logistic regression, linear regression, Naïve Bayes.

The first step we took before applying any of these techniques was cleaning our dataset. With the current dataset we had, there were a few columns in the table that would not affect the prediction and were unnecessary for us to use, hence we removed the ID and work-type, and residence type columns. Therefore, while inspecting our dataset, we also noticed a few BMI records contained N/A instead of a useable figure, therefore we disregarded those rows from the dataset. Moreover, to clean up the data, we changed any text to figures, for example, for the gender, changing male to 0 and female to 1. The smoking status column also contained 3 categories "formerly smoked", "never smoked" and "smokes", which we translated to the numbers -1,0,1, as well as removing rows that contained "Unknown" under smoking status. We are using K-Fold to split our data into training and test data.

Analysis and Results/Outputs

In this section, we will be analyzing and providing results for the different models that are implemented. We will be also showing the accuracy and comparing it for the problem domain which was predicting stroke based on various factors. These models are all classification algorithms except for linear regression which we will talk about here, but it is not part of the code as it is a regression algorithm.

Decision Tree

A decision tree creates questions and then continuously splits/classifies the data until it divides all the data from each class e.g., true, and false statements. It starts at the first node, the root node, and adds on a node for each time a question is asked, where it creates sub nodes at decision nodes and eventually stops at a leaf node. The reason for choosing a decision tree model is because it is a classification model that allowed us to represent the factors such as gender, smokes & age as nodes.

For our decision tree, we set our criterion as 'gini' since it improved the accuracy output more than 'entropy'. For the other values such as max_depth, min_samples_split and min_samples_leaf all as default values. Displayed below is the confusion matrix we created for the decision tree as well as the

results that were outputted from the decision tree. Using the decision tree, we were able to reach an accuracy of 90%.

| Decision Tree Classifier | | | | | |
|--------------------------|--------------|-----------|--------|----------|---------|
| | | precision | recall | f1-score | support |
| | 0 | 0.96 | 0.94 | 0.95 | 2247 |
| | 1 | 0.18 | 0.25 | 0.21 | 130 |
| | accuracy | | | 0.90 | 2377 |
| | macro avg | 0.57 | 0.59 | 0.58 | 2377 |
| | weighted avg | 0.91 | 0.90 | 0.91 | 2377 |

Figure 5 - Decision tree result

Random Forest

Another model we tested out was random forest since it works similarly to decision tree and could give us a higher accuracy. We decided to use it as it was similar to decision tree so that we could compare the results. For random forest we also kept the default values as they gave us our highest accuracy score at 94%. We then tested out other combinations for `n_estimators` and criterion to see if we could increase the accuracy any further. After setting the `n_estimators` we received an accuracy of 95% as well as an increase in our other values such as macro average and weighed average. We also tested out criterion for which 'gini' seemed the best fit, whilst 'entropy' seemed to decrease the accuracy.

| Random Forest Classifier | | | | | |
|--------------------------|--------------|-----------|--------|----------|---------|
| Confusion Matrix | | | | | |
| [[2244 3] | | | | | |
| [127 3]] | | | | | |
| | | precision | recall | f1-score | support |
| | 0 | 0.95 | 1.00 | 0.97 | 2247 |
| | 1 | 0.50 | 0.02 | 0.04 | 130 |
| | accuracy | | | 0.95 | 2377 |
| | macro avg | 0.72 | 0.51 | 0.51 | 2377 |
| | weighted avg | 0.92 | 0.95 | 0.92 | 2377 |

Figure 6 - Random Forest result

Logistic Regression

Initially, we tried to do linear regression as we wanted to check if the data, we wanted to analyse could be analysed using linear regression. However, we realized that we were trying to do classification metrics in a regression model which was not going to work and therefore pointless. Therefore, we implemented Logistic regression which is like linear regression but can be used in a classification problem. This is because instead of predicting data to be continuous it predicts one thing or another.

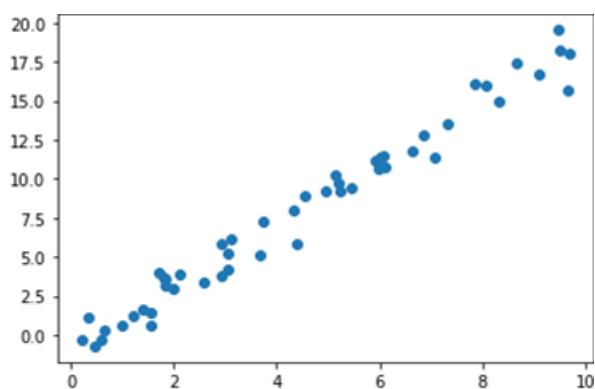


Figure 7 - scatter plot for logistic regression

For linear regression, the model fit line would be a straight continuous line whereas for the logistic regression the model fit would be shaped as curved S. The logistic regression produced an accuracy rating score of 95% which is the highest out of all the models. Logistic regression also produced errors during its outputs, however it did not interrupt the predictions and so we did not discard it.

| Confusion Matrix | | | | | |
|------------------|---|-----------|--------|----------|---------|
| [[2247 0] | | | | | |
| [130 0]] | | | | | |
| | | precision | recall | f1-score | support |
| | 0 | 0.95 | 1.00 | 0.97 | 2247 |
| | 1 | 0.00 | 0.00 | 0.00 | 130 |
| accuracy | | | | 0.95 | 2377 |
| macro avg | | 0.47 | 0.50 | 0.49 | 2377 |
| weighted avg | | 0.89 | 0.95 | 0.92 | 2377 |

Figure 8 - logistic regression result

Naïve Bayes

Naïve Bayes is based on the Bayes theorem with the strong 'naïve' assumptions to calculate conditional probabilities. The accuracy score received for this model was 87% which was the third-best out of the other models. When running this model, we found that it seemed to be the best model of all our models. Although the accuracy was at 87% and lower than other models, the predictions that were outputted seemed to be the best results of all the models thus far.

| Naïve Bayes | | | | | |
|------------------|---|-----------|--------|----------|---------|
| Confusion Matrix | | | | | |
| [[2007 240] | | | | | |
| [78 52]] | | | | | |
| | | precision | recall | f1-score | support |
| | 0 | 0.96 | 0.89 | 0.93 | 2247 |
| | 1 | 0.18 | 0.40 | 0.25 | 130 |
| accuracy | | | | 0.87 | 2377 |
| macro avg | | 0.57 | 0.65 | 0.59 | 2377 |
| weighted avg | | 0.92 | 0.87 | 0.89 | 2377 |

Figure 9 - Naive Bayes result

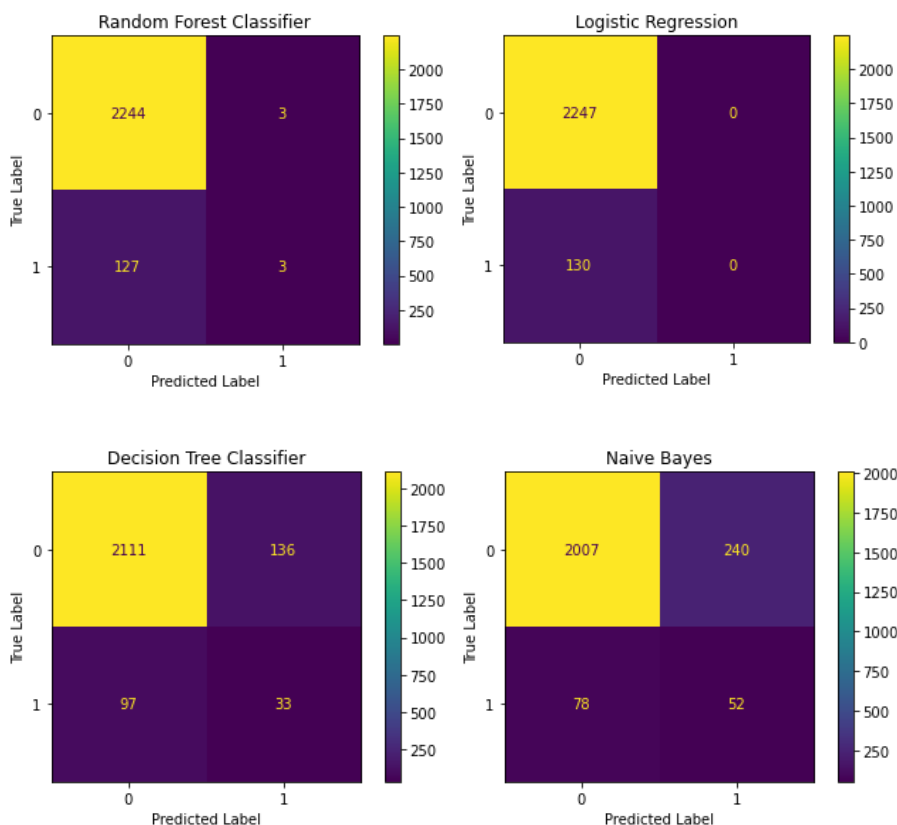
Did you have any problems or difficulties working with the dataset?

Problems with the dataset included missing BMI, unknown smoking status, and unknown/another gender. For BMI we solved this problem by replacing the unknown values with the median. As for the gender and smoking status, any rows with unknown factors, we had completely removed from the dataset. Due to the reason that our data was under sampled in the stroke, we decided to use Synthetic Minority Oversampling Technique (SMOTE) to increase the number of minority cases which means it would balance the data out.

SMOTE

After analysis of the dataset, we had found out that the dataset was very unbalanced. 95% of the data was about non stroke patients leaving a small 5% of it to be stroke patients. From this we had concluded this was not enough data to train the machine learning model to accurately predict people who could have strokes. We used Smote to oversample the dataset to make it more balanced. By doing so, random state was not an option causing the matrix to change with every iteration of the code.

Confusion Matrix Models



Accuracy Metrics Comparison (weighted average used)

| Model | precision | Recall | f1-score | accuracy |
|---------------------|-----------|--------|----------|----------|
| Decision Tree | 91% | 90% | 91% | 90% |
| Logistic Regression | 89% | 95% | 92% | 95% |

| | | | | |
|---------------|-----|-----|-----|-----|
| Random Forest | 89% | 95% | 92% | 95% |
| Naïve Bayes | 92% | 87% | 89% | 87% |

Evaluation

The training and testing data was split into 1/3 to 2/3 of the whole data. By doing so we could analyse how different models would perform. The most valuable score from the accuracy metrics comparison would be the accuracy and the recall. This is because accuracy shows the percentage of how well the model would classify the patient's input and predict if they have a chance of either stroke or no stroke. The recall percentage shows how many of those predictions were correct. The models that seemed to have performed consistently well were the decision tree, Naïve Bayes, and random forest. Since both the recall and accuracy were 95% for both logistic regression and random forest, we can assume that many of the positive predictions were correct. On closer inspection, the confusion matrix for the random forest did not produce the expected number of negative predictions therefore it was not an appropriate model. Also, the confusion matrix logistic regression was not appropriate either so decision tree with an accuracy of 90% is the best model although it had lower accuracy than both random forest and Naïve Bayes.

Conclusion

In conclusion, this prediction can be used by hospitals to automatically predict if their patient has a chance of getting a stroke or no stroke. The hospitals just need to input the required data from the patients. In the future, we could collect data from hospitals so that we have a much larger sample which would increase the accuracy of the prediction. Through AI machine learning, doctors would be able to treat and provide valuable insight to their patients regarding stroke. There will be situations where this could help a doctor treat a patient before they have a stroke which means lives could be saved. Finally, it was very intriguing to learn and implement AI as well as work with the dataset. We were able to observe different models and their outputs. It was an interesting module as we were able to try to make something that would have a positive effect on the world through machine learning.

References

Scikit-learn.org. (2019). 1.9. *Naive Bayes* — *scikit-learn 0.21.3 documentation*. [online] Available at: https://scikit-learn.org/stable/modules/naive_bayes.html. (for definiton)

Laerd Statistics (2018). Linear Regression Analysis in SPSS Statistics - Procedure, Assumptions and Reporting the output. [online] Laerd.com. Available at: <https://statistics.laerd.com/spss-tutorials/linear-regression-using-spss-statistics.php>.

Brownlee, J. (2020). *SMOTE for Imbalanced Classification with Python*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>.