**ANL 488 PROJECT FINAL REPORT**

**Identifying Risk of Stroke**

**Using Predictive Modelling**



**Submitted by**

**Diane Yeo Su Ting**

**H1882431**

**SCHOOL OF BUSINESS Singapore University of Social Sciences**

**Presented to Singapore University of Social Sciences in partial fulfilment   
of the requirements for the**

**Degree of Bachelor of Science in Business Analytics**

**2021**

**Abstract**

Stroke is one of the leading causes of death worldwide. Cardiovascular related deaths such as heart disease or stroke, make up 31.7% of all deaths in Singapore in 2020. Stroke can affect anyone at any age, more so in elderly. In Singapore, the number of people having stroke is rising, especially so among adults aged 40-59 years of age. Statistically, 1 in 10 stroke patients in Singapore are below 50 years of age.

The aim of this project is to be able to predict an individual’s risk of having stroke. This is done by analysing demographics, past and current medical history, and lifestyle habits. This project adopted the CRISP-DM framework for predictive modelling, and the tool of choice was Python. Due to Python’s library Scikit-learn inability to handle categorical features, the dataset’s categorical features were then transformed into numeric types using *OneHotEncoding*. Initial data exploration found that the dataset was largely imbalanced, skewed towards non-incidence of stroke, and data balancing had to be conducted before modelling can be done. This project carried out three plausible methods in dealing with the data imbalance – over-sampling, under-sampling, and combination of both over- and under-sampling. Six predictive modelling techniques were chosen to perform predictive modelling, and the best model was chosen based on the highest Recall score returned.

The recommended model for this project was Random Forest, with combination sampling that dealt with the data imbalance. The recommended model returned age, average glucose levels, BMI, and surprisingly individuals who were not married, were more susceptible to having stroke. The model returned results that were aligned with previous research where age and glucose levels were determinants in predicting an individual’s risk of having stroke.

Table of Contents

[Chapter 1 Introduction 1](#_Toc87232420)

[Chapter 1.1 Business Problem 2](#_Toc87232421)

[Chapter 1.2 Business Analytics Problem 2](#_Toc87232422)

[Chapter 2 Literature Review 3](#_Toc87232423)

[Chapter 3 Data Understanding and Preparation 9](#_Toc87232424)

[Chapter 3.1 Data Understanding 9](#_Toc87232425)

[Chapter 3.1.1 Data Description 9](#_Toc87232426)

[Chapter 3.1.2 Data Exploration and Data Quality Verification 10](#_Toc87232427)

[Chapter 3.2 Data Preparation 15](#_Toc87232428)

[Chapter 3.2.1 Removing Duplicated Samples 15](#_Toc87232429)

[Chapter 3.2.2 Missing Values Replacement 16](#_Toc87232430)

[Chapter 3.2.3 Categorical Features Transformation 18](#_Toc87232431)

[Chapter 4 Modelling and Evaluation 21](#_Toc87232432)

[Chapter 4.1 Evaluation Metric 21](#_Toc87232433)

[Chapter 4.2 Train-Test Split 22](#_Toc87232434)

[Chapter 4.2.1 Imbalanced Data 22](#_Toc87232435)

[Chapter 4.3 Dealing with Data Imbalance 23](#_Toc87232436)

[Chapter 4.3.1 Over-sampled Data 23](#_Toc87232437)

[Chapter 4.3.2 Under-sampled Data 27](#_Toc87232438)

[Chapter 4.3.3 Combination of Over- and Under-sampled Data 29](#_Toc87232439)

[Chapter 4.4 Model Evaluation 30](#_Toc87232440)

[Chapter 4.5 Recommendation 34](#_Toc87232441)

[Chapter 5 Conclusion 38](#_Toc87232442)

[References 40](#_Toc87232443)

[Appendix A 42](#_Toc87232444)

[Appendix B 45](#_Toc87232445)

[Appendix C 63](#_Toc87232446)

[Appendix D 72](#_Toc87232447)

List of Tables

[Table 1. Data Dictionary for Raw Dataset 10](#_Toc87232532)

[Table 2. Features Descriptive Statistics 11](#_Toc87232533)

[Table 3. Summary of Target Variable 11](#_Toc87232534)

[Table 4. Recall scores of Imbalanced Dataset 22](#_Toc87232535)

[Table 5. Summary of Recall scores for SMOTE and Random Over-sampling 26](#_Toc87232536)

[Table 6. Summary of Recall scores for Random Over-sampling 26](#_Toc87232537)

[Table 7. Summary of Recall scores for Random Under-sampling 28](#_Toc87232538)

[Table 8. Summary of Recall scores for Combined Sampling 30](#_Toc87232539)

[Table 9. Summary of Recall scores for balanced data with 80-20 train-test split 31](#_Toc87232540)

[Table 10. Summary of Recall scores for Over-sampling and Combination sampling 32](#_Toc87232541)

[Table 11. Summary of Recall scores for 80-20 split Combination sampling (30-90) 33](#_Toc87232542)

List of Figures

[Figure 1. CRISP-DM Framework (Data Science Process Alliance, n.d.) 9](#_Toc87232254)

[Figure 2. Bar plot of the target variable (stroke) 12](#_Toc87232255)

[Figure 3. Gender distribution with Incidence of Stroke. 13](#_Toc87232256)

[Figure 4. Age distribution with Incidence of Stroke. 13](#_Toc87232257)

[Figure 5. Heart disease and Hypertension with Incidence of Stroke. 14](#_Toc87232258)

[Figure 6. Glucose levels with Incidence of Stroke. 14](#_Toc87232259)

[Figure 7. Smoking status with Incidence of Stroke. 15](#_Toc87232260)

[Figure 8. BMI distribution with Incidence of Stroke. 16](#_Toc87232261)

[Figure 9. Heatmap of continuous features. 17](#_Toc87232262)

[Figure 10. Import various Libraries before Over-sampling and Modelling. 24](#_Toc87232263)

[Figure 11. Define Input features and Target. 24](#_Toc87232264)

[Figure 12. Over-sampling training data using RandomOverSampler. 25](#_Toc87232265)

[Figure 13. Over-sampling training data using SMOTE. 25](#_Toc87232266)

[Figure 14. Computation of Evaluation metrics and breakdown. 25](#_Toc87232267)

[Figure 15. Import various Libraries before Under-sampling and Modelling. 27](#_Toc87232268)

[Figure 16. Under-sampling training data. 28](#_Toc87232269)

[Figure 17. Combination sampling (over-sampling). 29](#_Toc87232270)

[Figure 18. Combination sampling (under-sampling). 29](#_Toc87232271)

[Figure 19. Target count of Combination sampling. 30](#_Toc87232272)

[Figure 20. Recall score of RF model with default parameter values. 35](#_Toc87232273)

[Figure 21. Recall score of RF model with specified parameter values. 36](#_Toc87232274)

[Figure 22. Python code for optimized RF model. 36](#_Toc87232275)

[Figure 23. Features Importance for RF model with Combination Sampling. 37](#_Toc87232276)

# Chapter 1 Introduction

There are many causes of death, with stroke being the second leading cause (Frank et al., 2021). (Ischemic) Stroke is defined as a neurological attack whereby the brain is deprived of blood flow and oxygen required to function. It is often mixed up with heart attack (medically known as ischemic heart disease), where the coronary artery has complete blockage or blockage so narrow that it restricts blood flow to the heart muscle.

Stroke does not occur overnight. Often, there are factors and demographics that would lead up to the likely occurrence of a stroke. Factors such as old age, current or past medical conditions, family history, gender, current, ex- or non-smoker. Stroke has since become the second leading cause of death (Frank et al., 2021) from the third leading cause of death in 2006 (Khan & Vohra, 2006). Advanced technology, especially prevalent in the medical field, together with big data and analytics, could help us identify individuals at higher risk of stroke. To help identify individuals at greater risk of stroke, we can use *predictive modelling*.

*Predictive modelling* uses current or past data to build models that predict future values or the state of certain variables (Singapore University of Social Sciences [SUSS], 2020). Some examples of predictive modelling are *Logistic Regression* (LR), *Decision Trees* (DT), *Artificial Neural Networks* (ANN). There are two types of predictive models – an *estimator* that predicts a numeric target or a *classifier* that predicts a categorical target (SUSS, 2020).

*Predictive modelling* has been used in many scenes in the healthcare industry. Most are used to predict if a patient would be re-admitted after being discharged. Furthermore, how long after discharge before returning to the hospital. Another common area where predictive modelling is used in the healthcare industry is the prediction of certain illnesses based on patients’ symptoms and family history.

This project would be looking into predicting the incidence of stroke – looking into various factors such as – age, gender, the medical history that affect the prediction. Chapter 2 reviews published articles related to this project, and Chapter 3 presents an overview of the dataset, exploration of the data, preparation of the data before it is used for modelling. Chapter 4 discusses models that the data has been fitted into and the returned results, the evaluation metrics looked at that determined model performance and the recommendation on the model for this project. Chapter 5 concludes the findings and analysis of this project.

## Chapter 1.1 Business Problem

Unfortunately, stroke moved from being the third to second largest cause of death (Frank et al., 2021) and is a worrying health issue in today’s society. This report aims to identify individuals who are at risk of stroke and provide them with the necessary treatments as soon as possible.

## Chapter 1.2 Business Analytics Problem

The business analytics problem is to build and identify the best performing predictive modelling techniques that could help determine the incidence of stroke with the highest Recall rate – based on an individual’s demographical profile, past and current medical history, and lifestyle choices. In addition, which features affect the incidence of stroke the most would also be assessed.

# Chapter 2 Literature Review

As with all major illnesses, stroke has risk factors that may or may not have contributed to the outcome. Risks factors such as being a(n) (ex) smoker, family history, prior or current illnesses are some factors that affect the occurrence of a stroke.

Khan and Vohra (2006) conducted a study between 1st April 1997 and 31st March 1998 in Dr Ziauddin Medical University Hospital, North Nazimabad Campus, Karachi. 281 patients who had their first-ever stroke was admitted and recorded during one year. The patient profiles considered patients of both genders, aged 20 to 70 and who had their first stroke confirmed by CT scan. Khan and Vohra (2006) found seven risk factors – hypertension, smoking, diabetes mellitus, underlying cardiac diseases, positive family history, high cholesterol, and past transient ischemic attack history- contributing factors to a stroke occurrence. Other than risk factors that we should note when determining the occurrence of stroke, Khan and Vohra (2006) also noted that determinants such as age, gender, ethnicity cannot be modified, unlike those factors mentioned. However, these are considered risk markers and need to be considered in the assessments.

Khan and Vohra (2006) analysed the data using the SPSS software package. The data’s qualitative variables were analysed by finding the frequencies and percentages. A chi-squared test was used to compare the risk factors in the types of strokes (Khan & Vohra, 2006). Quantitative variables were analysed by calculating the mean, standard deviation, and t-test (Khan & Vohra, 2006). Khan and Vohra (2006) applied the t-test to find the differences between the two types of strokes – ischemic and haemorrhage. This project will be conducted in Python rather than SPSS Modeler.

Between the two genders, Khan and Vohra (2006) found that males are at greater risk of stroke than females, but it is a slight difference. Khan and Vohra (2006) also found that age is the most vital determinant among new stroke cases each year. Patients with diabetes mellitus are four times more likely at risk of incidence of stroke than individuals who do not have diabetes (Khan & Vohra, 2006). As we all know that smoking is bad for our health, Khan and Vohra (2006) found that smokers are 1.5 to 2.9 times at risk of incidence of stroke than non-smokers. Smokers who have quit for 5 to 10 years are seen to reduce their risk of stroke than non-smokers (Khan & Vohra, 2006).

There is a complex relationship between age and gender which complicates the assessment of gender differences in stroke-related mortality (Roy-O’Reilly & McCullough, 2018). Roy-O’Reilly and McCullough (2018) recognised that female patient would be at higher risk of stroke mortality than males, but gender alone does not independently predict mortality. Stroke risk factors are important determinants of incidence and pathophysiology of ischemic stroke (the artery that supplies oxygen-rich blood to the brain becomes blocked) (Roy-O’Reilly & McCullough, 2018). Roy-O’Reilly and McCullough (2018) found that women have several unique stroke risks such as – oral contraceptive pill (OCP) use, pregnancy, menopause, and hormone replacement therapy (HRT). The use of OCP significantly increases the risk of stroke, with the highest risk conferred by high-oestrogen OCPs. Oestrogens has many positive cardiovascular effects, but they also enhance coagulation, elevating the risk of blood clotting in women. Roy-O’Reilly and McCullough (2018) found that during pregnancy is when the incidence of stroke increases, especially during the last trimester and early postpartum period. The risk of stroke is further increased in women who develop gestational hypertension and preeclampsia during pregnancy (Roy-O’Reilly & McCullough, 2018). Due to a decline in female sex hormones, menopause, and high oestrogen levels, middle-aged women face an increased risk of stroke incidence compared to their male counterparts (Roy-O’Reilly & McCullough, 2018). However, women who experience early menopause (younger than 42 years) are faced with a twofold increased risk of stroke incidence.

Roy-O’Reilly and McCullough (2018) found that men tend to have a higher incidence of stroke in childhood and early adult years than women. However, in the middle- and elderly-aged women, women are reported with higher stroke incidence, naturally accompanied by menopause and loss of female sex hormones (Roy-O’Reilly & McCullough, 2018). Atrial fibrillation (AF) is a major risk factor for ischemic strokes. In addition, age is a major risk factor for AF. Stroke risk conferred by AF increases with age (Roy-O’Reilly & McCullough, 2018). The prevalence of AF and cardioembolic ischemic stroke are significantly higher in women than in men. Women tend to be older at the time of stroke onset, besides the prevalence of AF, which results in increased stroke severity than their male counterparts (Roy-O’Reilly & McCullough, 2018).

A large-scale meta-analysis in recent years demonstrated that type 2 diabetes confers a greater risk for stroke in women (Roy-O’Reilly & McCullough, 2018). The risk of a fatal stroke is higher in diabetic female patients than males. Roy-O’Reilly and McCullough (2018) found a recent study that reported diabetic women are at a higher risk of 5-year mortality after stroke than non-diabetic women, a relationship that is not found in men. Overall, diabetes is a vital risk factor for stroke in women than in men, and women who have diabetes have poorer outcomes after a stroke than men (Roy-O’Reilly & McCullough, 2018). Roy-O’Reilly and McCullough (2018) cannot find the driving factors behind this gender disparity, and it remains largely unknown.

Frank et al. (2021) presented that smoking is a modifiable risk factor contributing to approximately 20% of stroke occurrences. The term “smoking paradox” suggests a link between smoking and favourable clinical outcomes following a stroke (Frank et al., 2021). A recent meta-analysis that includes 21 studies found no difference in the prognostic outcome of smokers and ischemic stroke (Frank et al., 2021). Frank et al. (2021) studied 90 adult patients (above 21 years), of which 49 were female. On average, Frank et al. (2021) found that smokers suffered a stroke 10 years earlier than non-smokers, consistent with published findings. An increase in age is a known negative correlation of stroke outcomes. The significant early onset may account for the controversial smoking paradox, as younger patients recover more quickly from stroke (Frank et al., 2021).

Although stroke is the second largest cause of death (Frank et al., 2021), it is rare. Data collected tend to be highly imbalanced – with non-occurrence of stroke being the majority class and occurrence of stroke the minor class. With a highly imbalanced dataset, the model would return skewed results towards the majority class because of their high frequency. There are two routes to deal with class imbalance – data level and algorithmic level (Kotsiantis, Kanellopoulos, & Pintelas, 2006). The Kotsiantis, Kanellopoulos, and Pintelas (2006) study guide how I should deal with the largely imbalanced dataset I have.

At the data level, Kotsiantis et al. (2006) proposed different forms of re-sampling such as – random over-sampling with replacement, random under-sampling, a combination of the above techniques and feature selection. Over-sampling is a non-heuristic method that aims to balance the classes through random replication of the minority class samples (Kotsiantis et al., 2006). The drawback of this method is that the replication of samples can increase the chances of overfitting the model. On the other hand, under-sampling is also a non-heuristic method that aims to balance the classes through random elimination of the majority class samples (Kotsiantis et al., 2006). The major drawback of this method is that it can eliminate potentially valuable data that could form interesting rules from the model. Feature selection is where the significant features for the model are selected, and those should not be removed (Ashfaq, Booma, & Mafas, 2020). Ashfaq, Booma, and Mafas (2020) study on – Managing Student Performance structured upon Knowledge Discovery in Databases (KDD) presented an approach for over-sampling where the synthetic samples are created for the minority class, Synthetic Minority Over-sampling (SMOTE), instead of replacing with the samples generated from over-sampling. SMOTE creates synthetic samples by analysing features rather than data (Ashfaq et al., 2020). Ashfaq et al. (2020) also presented Adaptive Synthetic Sampling (ADASYN), which utilises the distribution of weights for various samples of the minority class. ADASYN enhances the distribution of the data points by minimising the data bias by shifting the sample weights (Ashfaq et al., 2020). ADASYN only alters the minority class compared to SMOTE (Ashfaq et al., 2020). Ashfaq et al. (2020) study has returned the Random Forest (RF) model as the best model (highest accuracy of 86.74%), together with ADASYN and Recursive Feature Elimination (RFE).

At the algorithmic level, Kotsiantis et al. (2006) proposed solutions such as – adjusting the costs of the various classes to counter the class imbalance, adjusting the probabilistic estimate at tree leaf (DT), adjusting the decision threshold. Weighted distance function can be proposed in the classification stage of k-Nearest Neighbours (kNN) to compensate for the imbalance in the training sample without altering the class distribution (Kotsiantis et al., 2006). Weights are assigned to the respective classes, not the individual samples. As such, the weighting factor is further and greater than the minority class, producing a tendency for new patterns to find their nearest neighbour among the minority class (Kotsiantis et al., 2006). Some classifiers, such as Naïve Bayes classifier or Neural Networks (NN), return a score representing the extent to which a sample is a class member (Kotsiantis et al., 2006).

Often a mixture of the abovementioned techniques by Kotsiantis et al. (2006) and Ashfaq et al. (2020) is used to handle the class-imbalance problems. These methods combine the results of many classifiers, usually induced after over-sampling or under-sampling the data with different thresholds (Kotsiantis et al., 2006).

The proposed models will lookout for interesting relationships based on the abovementioned risk factors and markers. The implementation of the various techniques and algorithms related to data mining mentioned above in Chapter 4 will be looked out.

# Chapter 3 Data Understanding and Preparation

In this project, the Cross-Industry Standard Process for Data Mining (CRISP-DM) framework, as shown in Figure 1. CRISP-DM involves six phases which are adopted for this data mining project.

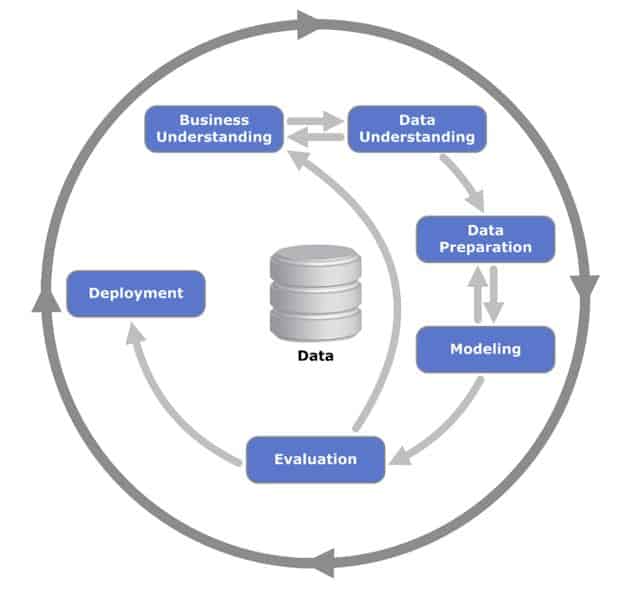


Figure 1. CRISP-DM Framework (Data Science Process Alliance, n.d.)

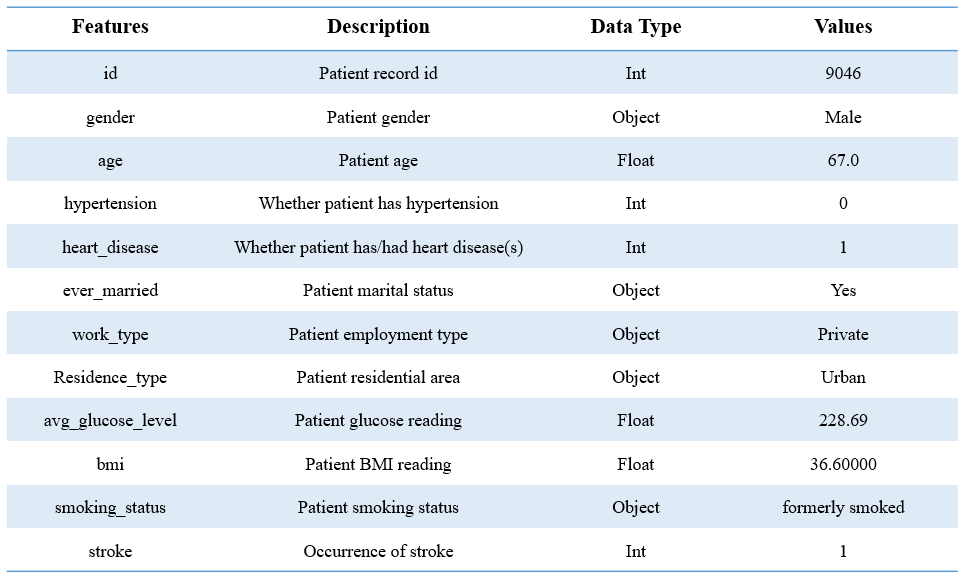
## Chapter 3.1 Data Understanding

The data understanding phase is where data collection, analysis, and verification of data quality (SUSS, 2020) begins.

### Chapter 3.1.1 Data Description

The dataset was obtained from Fedesoriano (2021) on Kaggle titled “Stroke Prediction Dataset”. The dataset consists of 5,110 samples with 11 input features and one target variable, stroke. The dataset contains features that represent the demographics and risk factors of each patient recorded. Our target would be identifying an individual at-risk of stroke, represented by value 1 under the stroke feature in our dataset. The 11 features would be input variables to predict if an individual is at risk of stroke. Table 1 provides a summary of the features and target in the dataset using strokeData.info().

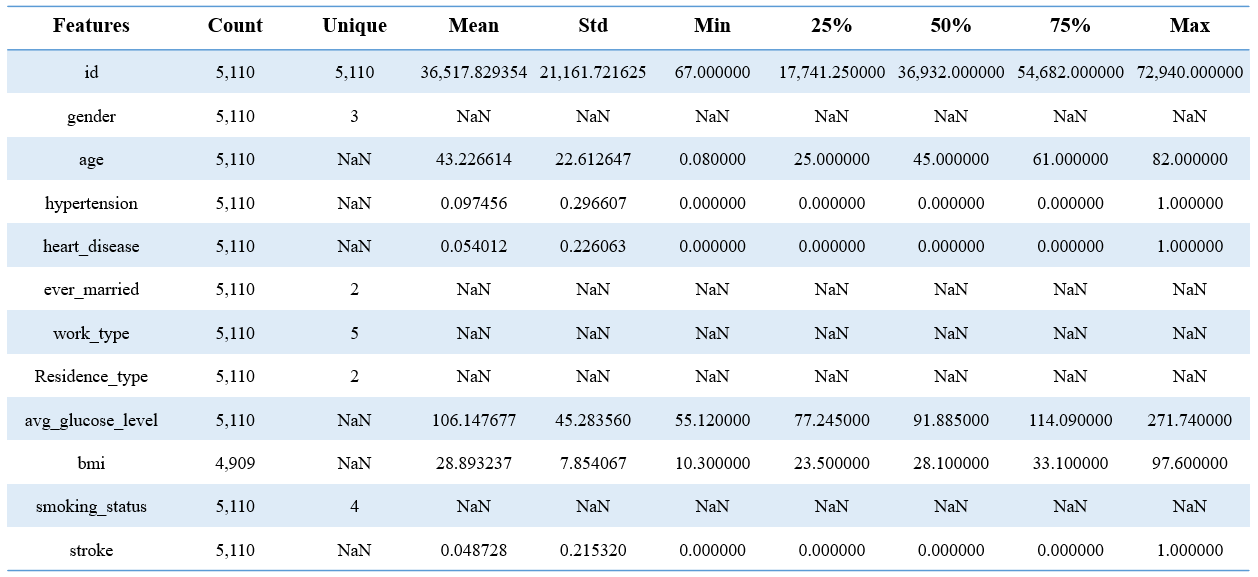
Table 1. Data Dictionary for Raw Dataset



### Chapter 3.1.2 Data Exploration and Data Quality Verification

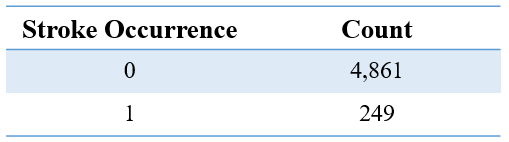
Before any data preparation can be done, it is best to know the full descriptive statistics of the dataset (see Table 2). Table 2 summarises the dataset’s features – the total number of samples, the min and max of each feature, the mean, median, and standard deviation, as Ill as unique counts for categorical values. Table 2 is derived from the dataset using strokeData.describe(include=‘all’).transpose().

Table 2. Features Descriptive Statistics



Preliminary findings suggest that the dataset has more not at-risk of stroke than at-risk. This is not surprising as stroke is a rare event. As seen in Table 2, the stroke mean is much lower (0.0487) than 0.05, lower than the average of 0 and 1 (not at-risk and at-risk, respectively). This indicates that the at-risk of stroke in the dataset is very little (see Table 3), and data preparation is needed so that results returned are not skewed towards the majority class.

Table 3. Summary of Target Variable



Chart

Description automatically generated

Figure 2. Bar plot of the target variable (stroke)

Based on demographics, we have more data on females than males (see Figure 3). Preliminary findings suggest that there is not much difference between females and males for incidence of stroke. As seen in Figure 3, the difference is so minute that the result can go either way. In contrast, age plays a part in the incidence of stroke (see Figure 4). Figure 4 shows the relationship between age and their relationship with the incidence of stroke – an increased risk with age.

Chart, bar chart

Description automatically generated

Figure 3. Gender distribution with Incidence of Stroke.

Chart

Description automatically generated

Figure 4. Age distribution with Incidence of Stroke.

Based on the individual’s medical history, preliminary findings show that hypertension and heart disease does not affect stroke incidence (see Figure 5). However, keeping in mind that the dataset is largely imbalanced and skewed towards the non-incidence of stroke. On the other hand, glucose levels show a possible relationship with the incidence of stroke (see Figure 6) – increased levels of glucose, increased risk of incidence of stroke.

A picture containing text

Description automatically generated

Figure 5. Heart disease and Hypertension with Incidence of Stroke.

Chart

Description automatically generated

Figure 6. Glucose levels with Incidence of Stroke.

Based on lifestyle habits, preliminary findings showed not much difference amongst current smokers, past smokers, non-smokers, and unknown smoking status (see Figure 7). An individual’s relationship with smoking and the incidence of smoking is not definite.

Chart, bar chart

Description automatically generated

Figure 7. Smoking status with Incidence of Stroke.

## Chapter 3.2 Data Preparation

In this third stage, Data Preparation occurs. The dataset was cleaned, transformed, and possibly reduced or oversampled (SUSS, 2020).

When the dataset is filled with duplicated, missing or incomplete samples, it will cause our modelling results to be inaccurate. The data must be transformed and cleaned to reduce inaccuracy caused by uncleaned data.

The data preparation stage involved these phases:

1. Removed duplicated samples, ensuring that each sample was unique.
2. Replaced missing values
3. Transformed categorical features into numeric types

### Chapter 3.2.1 Removing Duplicated Samples

The first step is removing any duplicated samples to prevent the results from being skewed. When there is a sample that is duplicated, it has a higher frequency than the other samples. With a higher frequency, the model will read the sample “more than once”, thinking they are distinct samples. This will cause the model to return inaccurate results.

After performing this phase, there were no duplicated samples in this dataset which means zero samples have been removed. It was essential to carry out this step when a dataset is retrieved, especially when it is large. Time is saved looking through each sample, ensuring they are unique.

### Chapter 3.2.2 Missing Values Replacement

As seen in (see Table 2), *bmi* has 4,909 samples, meaning 201 missing values. There were two ways to deal with missing data – remove or replace. Before either one was carried out, the distribution of *bmi* with the incidence of stroke was explored (see Figure 8). The majority have BMI within 20 to 30, with the incidence of stroke having a slightly higher BMI mean. However, BMI above 30 did not indicate a higher chance of stroke.

Chart

Description automatically generated

Figure 8. BMI distribution with Incidence of Stroke.

A heatmap was created to explore how each continuous feature correlated to the incidence of stroke (see Figure 9). Figure 9 showed that *age* had the highest correlation with the incidence of stroke, and *bmi* had the lowest correlation. Although *bmi* had the lowest correlation compared to the other features, it had a positive correlation which indicated some form of relationship with the incidence of stroke.

Chart, treemap chart

Description automatically generated

Figure 9. Heatmap of continuous features.

Replacing the missing values with the BMI mean rather than removing them. Since the dataset was largely imbalanced – leaning towards non-incidence of stroke, it would be risky to delete any samples that could potentially be useful for the models.

However, the limitation to replacing the missing BMI values with the mean could risk generalising these 201 individuals as the BMI’s mean physical build, when it may not be the case. Also, BMI is relative to an individual’s age as well as gender. This means the individual may be an overweight adult but has an average adult’s BMI. This would categorise him/her in the normal/healthy range. This is dangerous as these individuals may be at-risk of stroke but are left out due to generalisation.

### Chapter 3.2.3 Categorical Features Transformation

This project would be carried out using Python, and one of the Python libraries that were used – Scikit-learn, is unable to take categorical features. Due to the library’s limitation, categorical features would have to be transformed into numeric types so that the library can read the dataset to proceed with the modelling.

I had chosen to use *OneHotEncoding* library to transform the dataset as it was suitable for the dataset that was stored in a DataFrame format. Before deploying *OneHotEncoding* library, I first imported it from its library – sklearn.preprocessing (**from** sklearn.preprocessing **import** OneHotEncoder). Followed by creating an instance for OneHotEncoder and setting it to ignore unknown samples (enc = OneHotEncoder(handle\_unknown=‘ignore’, sparse=**False**)). Following, I extracted the categorical features individually, paired with *id* to make it a 2D-array (i.e., stroke\_gender = strokeData[[‘id’, ‘gender’]]). This will then allow *OneHotEncoding* library to carry out the transformation (i.e., stroke\_gender\_transformed = pd.DataFrame(enc.fit\_transformed(stroke\_gender[[‘gender’]]))). Not forgetting to rename the transformed features’ columns with their original feature header (i.e., stroke\_gender\_transformed.columns = enc.get\_feature\_names([‘gender’])).

Once all the categorical features had been transformed, I then concatenated the numeric features and transformed categorical features together to return the ‘cleaned’ dataset (i.e., strokeData\_transformed = pd.concat([strokeData, stroke\_gender\_transformed],axis=1)).

The cleaned dataset with ‘new’ features was summarized in Table 4. More features (20 features, excluding target variable stroke) are now compared to Table 1 (11 features) due to transforming the categorical features into numeric types.

Table 4. Summary of Cleaned Dataset



# Chapter 4 Modelling and Evaluation

In this project, Python was the tool used to carry out the predictive modelling for the risk of incidence of stroke. The dataset used was largely imbalanced even after cleaning, was skewed towards the non-incidence of stroke.

Six models were chosen to perform the modelling – *Logistic Regression* (LR), *K-Nearest Neighbours* (kNN), *Random Forest* (RF), *Support Vector Machine* (SVM), *Decision Tree* (DT), and *Neural Network* (NN).

LR was selected to predict the probability of an event, in this case, stroke. SVM was selected as it works well, in most cases, with two possible target values – in this case, the at-risk or not at-risk of stroke. DT can be used for both classification and regression problems. They are constructed by recursively evaluating different features. RF is built on an ensemble of DTs, and they are usually trained using the ‘bagging’ method – meaning the combination of learning models would help to increase the overall result (Donges, 2021). RF can be used in both classification and regression problems, and RF would eliminate the problem of overfitting. Just like RF, kNN can be used in both classification and regression problems. kNN predicts the value of new data points based on their training data and how similar they are (Singh, 2018). NNs tend to be better at predictive modelling because of their hidden layers, which help to make predictions more accurate. Inside the hidden layers, the model ‘learns’, similarly to a person.

## Chapter 4.1 Evaluation Metric

In this project, the models’ performance is based on the **Recall score**. Recall score is a measure of the model correctly identifying true positives. It showed how many were correctly predicted at-risk of stroke vs those who had a stroke.

## Chapter 4.2 Train-Test Split

Before inputting any dataset into any model, splitting of data must be done first. Splitting of data is essential when it comes to training the model. This is to ensure that the model is not overfitted or underperforms. If the data is not split, the model is trained with 100% seen data, returning overfitted results. When the model is deployed on unseen data, it may underperform, and the model will have to be retrained. This will cost time and money, a situation that should be avoided.

Modelling was performed on three splits – 70-30, 80-20, and 90-10 in hopes to find the train-test split that best fits.

### Chapter 4.2.1 Imbalanced Data

Before data imbalance was dealt with, six models were performed with a train-test split of 70-30. Table 4 showed the summary of Recall scores of the six models when imbalanced data was inputted.

Table 4. Recall scores of Imbalanced Dataset

Table

Description automatically generated

## Chapter 4.3 Dealing with Data Imbalance

The data was largely imbalanced, skewed towards the non-incidence of stroke. When modelling was performed on the imbalanced data, **0%** of at-risk individuals predicted almost every time. Data balancing had been carried out to avoid this from happening. Data imbalance has been tackled using these three plausible methods – over-sampling, under-sampling, and a combination of over-sampling and under-sampling.

### Chapter 4.3.1 Over-sampled Data

Over-sampling is duplicating random samples from the minority class, incidence of stroke. After random duplication, the number of samples in the minority class will equal those in the majority class. However, the flaw would be that duplication of samples would increase the chances of overfitting the model.

Over-sampling was carried out in Python using the imbalanced-learn library, relying on scikit-learn – from imblearn.over\_sampling import RandomOverSampler, SMOTE.

Over-sampling was first carried out using SMOTE. SMOTE is riskier compared to random over-sampling as it creates synthetic samples by analysing features rather than data. This means that SMOTE creates its own samples based on the analysis of the dataset’s features. Samples created by SMOTE can be misleading when training the models. Random over-sampling was then carried out to duplicate samples from the minority class. Over-sampling was performed using both methods, with random over-sampling generally returning higher Recall scores than SMOTE (see Table 5).

Before over-sampling was carried out, the dataset had to be split into training and testing sets. The six models will each go through three different modelling for the three different splits – 70-30, 80-20, and 90-10. The project was carried out in Python, libraries and packages were imported before any modelling or splitting of data were done. Libraries such as Pandas, Scikit-learn, Imblearn, and Matplotlib were imported with some packages (see Figure 10).

Text

Description automatically generated

Figure 10. Import various Libraries before Over-sampling and Modelling.

The cleaned dataset was read as a DataFrame, followed by defining the input features and target column-wise (i.e., X = strokeData.drop(columns=‘stroke’, axis=1’) (see Figure 11). The dataset was split for training and testing by indicating its test size (i.e., X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42, stratify=y). The random state was given a number to ensure the split was always split at the same splitting point rather than randomly chosen. This was to ensure consistency for this project. In the train-test split, stratify was also given a value of the target. This was to ensure that at least one sample of the target prediction would occur to prevent 0% of the incidence of stroke. Once the dataset had been split, SMOTE and RandomOverSampler were instantiated to begin over-sampling and input the training data of X and y to undergo over-sampling (see Figure 12, Figure 13).

Text

Description automatically generated

Figure 11. Define Input features and Target.

Text

Description automatically generated

Figure 12. Over-sampling training data using RandomOverSampler.



Figure 13. Over-sampling training data using SMOTE.

Classifier was created for each model before fitting the ‘new’ training data into the model for modelling (i.e., rf = RandomForestClassifier(n\_estimators=10, min\_sample\_leaf=34)). The under-sampled training data was then fitted into the models for training (i.e, rf.fit(X\_oversample, y\_oversample)). After training, prediction of the original test data was carried out (i.e., y\_pred = rf.predict(X\_test)).

Recall score is the evaluation metric considered for this project to determine how well the model performed. Measures of performance such as confusion matrix, classification report, and Recall scores were computed to understand better the breakdown of the results returned (see Figure 14).

Text

Description automatically generated

Figure 14. Computation of Evaluation metrics and breakdown.

Over-sampling of the imbalanced dataset, using random over-sampling and SMOTE, were performed on all six models. The over-sampled dataset results have been collated in a table for easy comparison (see Table 5).

Table 5. Summary of Recall scores for SMOTE and Random Over-sampling

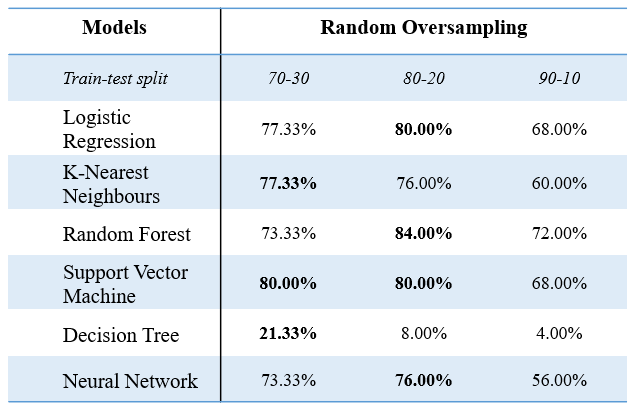
Table

Description automatically generated

As mentioned, SMOTE was a riskier approach to over-sampling than randomly over-sampling the minority class. After performing both approaches for over-sampling, RandomOverSampler yielded higher Recall scores for most of the models and splits (see Table 5). For this project, random over-sampling was selected rather than SMOTE for the over-sampling approach of the imbalanced dataset.

From Table 6, the ideal train-test split for random over-sampling is 80-20, with LR, RF, SVM, and NN returning the highest Recall scores.

Table 6. Summary of Recall scores for Random Over-sampling



### Chapter 4.3.2 Under-sampled Data

Under-sampling is the elimination of random samples in the majority class, non-incidence of stroke. After randomly eliminating samples in the majority class, the samples left will equal the number in the minority class. The major drawback of this is that the random elimination of samples may eliminate potentially useful data that would form interesting rules from the model.

Under-sampling was carried out in Python using the imbalanced-learn library – from imblearn.under\_sampling import RandomUnderSampler.

Before any modelling is done, the dataset was first split according to the three different train-test splits – 70-30, 80-20, and 90-10. Like over-sampling, various libraries and packages were used in the modelling were imported before modelling was carried out. The same libraries used in over-sampling (i.e., Pandas, Scikit-learn) were also imported for under-sampling (see Figure 14).

Text

Description automatically generated

Figure 15. Import various Libraries before Under-sampling and Modelling.

The cleaned dataset was first read as a DataFrame and defined the target's input features (see Figure 11). The dataset was then split based on the train-test split set, assigning random state a fixed number, so the split was not randomized, and stratifying the target to ensure at least one sample will return the incidence of stroke (i.e., X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42, stratify=y). Once the dataset had been split, RandomUnderSampler was instantiated to begin under-sampling and inputting the training data of X and y to undergo under-sampling (see Figure 16).

A screenshot of a computer

Description automatically generated with medium confidence

Figure 16. Under-sampling training data.

A classifier was created for each model before fitting the ‘new’ training data into the model for modelling (i.e., knn = KNeighborsClassifier(n\_neighbors=60)). The under-sampled training data was then fitted into the models for training (i.e, knn.fit(X\_undersample, y\_undersample)). After training, the original test data was predicted (i.e., y\_pred = knn.predict(X\_test)).

Recall score is the evaluation metric that was considered when evaluating how well the models performed. Confusion matrix, classification report and recall scores were computed to understand better the breakdown of the results returned (see Figure 14).

Under-sampling of the imbalanced dataset had been performed on all six models, and their results are collated in a table for easy comparison (see Table 7).

Table 7. Summary of Recall scores for Random Under-sampling

Table

Description automatically generated

From Table 7, the ideal train-test split for under-sampling was 80-20, with LR, kNN, SVM, and DT returning the highest Recall scores.

### Chapter 4.3.3 Combination of Over- and Under-sampled Data

Over-sampled data may overfit the model, and under-sampled data may potentially eliminate interesting rules from the model. Both methods were combined and implemented to find a balance, with under-sampling getting a higher percentage than over-sampling, as under-sampling returned higher Recall scores than over-sampling (see Table 7 and Table 6, respectively).

Before the combination of over-sampling and under-sampling was carried out, the dataset was first split into training and testing, just like over- and under-sampling (i.e., X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42, stratify=y). The combination of over-sampling and under-sampling was first carried out by over-sampling the minority class (see Figure 17), followed by under-sampling the majority class (see Figure 18). A range of sampling strategies was carried out to find which combination would return the highest Recall scores (see Table 8).

A screenshot of a computer

Description automatically generated with medium confidence

Figure 17. Combination sampling (over-sampling).

A screenshot of a computer

Description automatically generated with medium confidence

Figure 18. Combination sampling (under-sampling).

Table 8. Summary of Recall scores for Combined Sampling

Background pattern, table

Description automatically generated

From Table 8, the ideal combination of over-sampling and under-sampling was 30% and 90%, respectively. From Table 8, five out of six models, except DT, returned the highest Recall scores for combination sampling of 30-90 (over-sampling – under-sampling).

## Chapter 4.4 Model Evaluation

After carrying out random over-sampling, under-sampling, and combination of both, the ideal train-test split was 80-20, with combination having minority class over-sampled by 30% and majority class under-sampled by 90% (see Figure 19).

Chart

Description automatically generated

Figure 19. Target count of Combination sampling.

Out of the three data balancing methods explored, the decision was made to eliminate under-sampling. Even though under-sampling returned Recall scores that were generally greater than over-sampling (see Table 9), the dataset was just too small to interpret interesting rules from the model. It would risk generalising the whole dataset which could be dangerous.

Table 9. Summary of Recall scores for balanced data with 80-20 train-test split

Table

Description automatically generated

Between over-sampling and the combination of over-sampling and under-sampling, the combination of over-sampling and under-sampling dealt with the data imbalance. Reason being that firstly, the Recall scores returned by the combination was generally higher than that of over-sampling alone (see Table 10). Secondly, as mentioned above, over-sampling duplicate random samples from the minority class to make up samples to match the quantity in the majority class. When the model reads duplicated samples, overfitting tends to surface in terms of evaluation metrics. The model may return high evaluation metrics (i.e., accuracy score) that could mislead the reader into interpreting that the model is performing better than it is.

Table 10. Summary of Recall scores for Over-sampling and Combination sampling

Table

Description automatically generated

As seen from Table 8, a range of combination sampling was carried out to find the combination that would return the highest Recall score. When the percentage of over-sampling increased, the six models tended to show a decline in Recall scores instead of having a lower percentage of the over-sampled minority class. The opposite occurred when it came to the under-sampled majority class. The six models returned higher Recall scores when the percentage of under-sampled majority class increased. The data was resampled where the incidence and non-incidence of stroke are not equal, and stroke being an extremely rare event, the incidence of stroke to have lesser samples than non-incidence of stroke. There is no perfect data where the target variable is split equally in the real world, therefore replicating that in this project.

Amongst the six models, LR and RF returned the highest Recall scores with 80-20 train-test split for combination sampling with 30% over-sampling and 90% under-sampling of minority and majority classes, respectively (see Table 11).

Table 11. Summary of Recall scores for 80-20 split Combination sampling (30-90)

Table

Description automatically generated

For this project, models were narrowed down to LR and RF, based on the Recall scores returned for combination sampling. These two models returned the highest Recall scores amongst the six models built and ran on.

Both LR and RF are popular and efficient techniques that can generate reliable models for predictive modelling, as seen in Table 11. The advantages of LR are that it is simple and linear, reliable, and there are no parameters to tune. However, still do note the maximum iterations it performs for it to reach optimization before it converges. LR has a default maximum iterations of 100. However, it was not enough for this project when it first ran. The model returned a warning of ‘converging before optimization was obtained’. Thus I specified the maximum iterations for LR to 99,999 (i.e., logreg = LogisticRegression(max\_iter=99999)). LR can handle extreme values and outliers better than RF. LR is more flexible in classifying the output by changing the probability threshold dependent on the project. The general cons of LR are that it cannot handle non-linearities in data.

The advantage of RF is that it is reliable and has the ability to handle non-linearities in data. When the RF model is fine-tuned to suit the dataset, it will likely outperform an LR model. Not forgetting that RF is built on an ensemble of DTs, trained using the bagging method. This takes care of the problem of overfitting and returning an improved overall score. Whereas RF’s cons are that its parameters need to be specified, although their default values may work fine as they are. RF model does not perform well if the numerical features of the test data are not within or close to the range of the training data. Should a sample in the test data does not fall within the range of the training data’s specified range, it will be classified as a non-occurrence of the event.

## Chapter 4.5 Recommendation

The model recommended it would be **RF** with a combination sampling with 30-90 over-under-sampling, using 80-20 train-test split as the best model for this project and dataset, based on the Recall score returned – **84.00%**. This means that 42 out of 50 individuals in the testing dataset, predicted to have a stroke, actually *had* a stroke.

In my opinion, LR is unable to handle datasets that have many variables. This means that LR will perform poorly when given data with high dimensionality conditions. LR is better at handling continuous variables as opposed to categorical ones, which this project’s dataset has more of the latter. Both LR and RF models only accept numerical data. Should the dataset have categorical features, simply encoding categorical features with numerical values will allow the RF model to perform with categorical features. However, LR relies on the calculation based on weights. Applying the same method on RF to LR will not return the expected outcome. Depending on how the numbers are assigned, the algorithm in LR will treat certain categories that carry a more significant number to have higher importance. This can be solved by using one-hot-encoding, which was used in this project in the data preparation stage. This ensures that all categorical features are given an equal weightage to perform unbiased modelling.

As mentioned above, specifying RF parameters are thought of as a con for the general model. I think that this con is a pro – be it for this project or other projects. If the RF default parameters work for that given dataset, well and good. However, if the default parameter values return less than optimal results, fine-tuning of parameters may return a better performing RF model, rather than dismissing the RF model and concluding that it is a bad fit for the dataset. RF models allow users to specify the parameters, which helps fine-tune and customise the model to different projects. Not just different projects, but within the same project, possibly on another train-test split. RF model is suitable for datasets with high dimensionality conditions. This means that the RF model is highly suitable for datasets that contain many variables, such as this project.

Before specifying RF parameters and using its default parameter values, the over-sampled RF model with 70-30 train-test split returned a Recall score of **4.00%** (see Figure 20). Compared to specifying RF parameters after (i.e., stopping measures such as the minimum number of samples in leaf node), with the same over-sampled 70-30 train-test split RF model, the returned Recall score is **73.33%** (see Figure 21). Therefore, the recommended RF model has parameters values of n\_estimators=10 and min\_samples\_leaf=34. These parameters have been fine-tuned to fit the dataset for this project to return the highest Recall score, which returned the best model (see Figure 22).

Calendar

Description automatically generated with medium confidence

Figure 20. Recall score of RF model with default parameter values.

Calendar

Description automatically generated

Figure 21. Recall score of RF model with specified parameter values.

Text

Description automatically generated

Figure 22. Python code for optimized RF model.

The recommendation would be the RF model with combination sampling as the best model for this project because of its features importance. Most of the features had a positive correlation with the risk of incidence of stroke. The most prominent features were age, avg\_glucose\_level, bmi, and married\_No (see Figure 23).

Graphical user interface, text

Description automatically generated

Figure 23. Features Importance for RF model with Combination Sampling.

*Age* was the most important feature in this RF model, followed by *avg\_glucose\_level*, then *bmi*. As mentioned by Khan and Vohra (2006), the older the individual, the higher that individual is at risk of incidence of stroke. This was reflected in the RF model in Figure 23. Preliminary findings from the data exploration stage showed that *avg\_glucose\_level* and *bmi* had a positive correlation, feature *bmi* to a certain extent, with the increased risk of incidence of stroke. This was also reflected in features importance in the RF model in Figure 23.

Contrary to Khan and Vohra (2006) and Roy-O Reilly and McCullough (2018), findings of individuals who have a history of past ischemic attack and smokers were more at risk of incidence stroke. Figure 23 showed otherwise – individuals who had a history of heart diseases or attacks, hypertension, and current smokers were not at-risk of incidence of stroke. Figure 23 reaffirmed the preliminary findings from Chapter 3’s Data Exploration stage that features gender, hypertension, history with heart attacks, and an individual’s relationship with smoking did not determine if an individual was more at risk or increased an individual’s risk of incidence of stroke.

What is interesting was that the RF model had a positive correlation with the feature of *married\_No*. There was no mention in past research about the marital status that affects an individual’s risk of incidence of stroke.

# Chapter 5 Conclusion

This project’s focus was to build and identify the best performing predictive modelling techniques that could help determine the incidence of stroke with the highest Recall rate. Six predictive modelling techniques, namely LR, kNN, RF, SVM, DT, and NN, were implemented to determine which models performed the best amongst them.

The dataset obtained was largely imbalanced and skewed towards the non-incidence of stroke. Data balancing had to be carried out to deal with the data imbalance before any modelling was done. Plausible data balancing methods such as over-sampling, under-sampling, and combination of both over-sampling and under-sampling were carried out to deal with the data imbalance. The models were then fine-tuned to return the highest Recall score – the evaluation metric used to determine the best performing model in this project.

After many rounds of modelling and fine-tuning of models’ parameters, the best performing model amongst the six models was **RF with a combination sampling** that dealt with the data imbalance; returned the highest Recall score of **84.00%**.

(Word Count: 7,158)

# References

Ashfaq, U., Booma, P. M., & Mafas, R. (2020). Managing Student Performance: A Predictive Analytics using Imbalanced Data. *International Journal of Technology and Engineering (IJRTE), 8*(6), 2277-2283. doi:10.35940/ijrte.E7008.038620

Data Science Process Alliance. (n.d.). CRISP-DM framework [Image]. Retrieved from <https://www.datascience-pm.com/crisp-dm-2/>

Donges, N. (2021). *A Complete Guide to the Random Forest Algorithm*. Retrieved from <https://builtin.com/data-science/random-forest-algorithm>

Fedesoriano. (2021). Stroke Prediction Dataset. Retrieved June 20, 2021, from <https://www.kaggle.com/fedesoriano/stroke-prediction-dataset>

Frank, J. A., Swafford, K. J., Roberts, J. M., Trout, A. L., Stowe, A., Lukins, D. E., Grupke, S., Pennypacker, K., & Fraser, J. F. (2021). Smoking-Induced Sex Differences in Clinical Outcomes in Patients Undergoing Mechanical Thrombectomy for Stroke. *World Neurosurgery*, *153*, e365-e372.   
doi:[10.1016/j.wneu.2021.06.108](https://doi.org/10.1016/j.wneu.2021.06.108)

Khan, S. N., & Vohra, E. A. (2006). Risk Factors for Stroke: A hospital based study. *Pakistan Journal of Medical Sciences, 23*(1), 17-22. Retrieved August 7, 2021, from <http://www.pjms.com.pk/issues/janmar07/pdf/stroke.pdf>

Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering, 30*, 25-36. Retrieved July 26, 2021, from <https://www.researchgate.net/publication/228084509_Handling_imbalanced_datasets_A_review>

Roy-O’Reilly, M., & McCullough, L.D. (2018). Age and sex are critical factors in ischemic stroke pathology. *Endocrinology, 159*(8)*,* 3120-3131.   
doi:[10.1210/en.2018-00465](https://doi.org/10.1210/en.2018-00465)

Singapore University of Social Sciences. (2020). ANL303 Fundamentals of Data Mining Study Guide (5CU). Singapore, Singapore: Educational Technology & Production

Singapore University of Social Sciences. (2020). ANL307 Predictive Modelling Study Guide (5CU). Singapore, Singapore: Educational Technology & Production

Singh, A. (2018). *A Practical Introduction to K-Nearest Neighbours Algorithm for Regression (with Python code)*. Retrieved from <https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/>

# Appendix A

*Imbalanced sampling Results*

Logistic Regression

Graphical user interface, text

Description automatically generated with medium confidence

K-Nearest Neighbours

Graphical user interface, text

Description automatically generated with medium confidence

Random Forest

Graphical user interface, text

Description automatically generated with medium confidence

Support Vector Machine

A screenshot of a computer

Description automatically generated

Decision Tree

A screenshot of a computer

Description automatically generated with medium confidence

Neural Network

Text

Description automatically generated

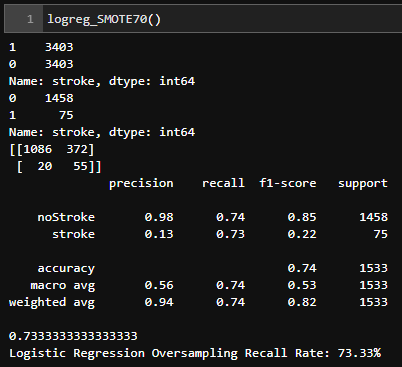
# Appendix B

*Over-sampling Results*

Logistic Regression

SMOTE

70-30



80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Random Over-sampling

70-30

Text

Description automatically generated

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A picture containing text

Description automatically generated

K-Nearest Neighbours

SMOTE

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

Text

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Random Over-sampling

70-30

Text

Description automatically generated with low confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Random Forest

SMOTE

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

Text

Description automatically generated

90-10

A picture containing calendar

Description automatically generated

Random Over-sampling

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Support Vector Machine

SMOTE

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Random Over-sampling

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

Text

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Decision Tree

SMOTE

70-30

A picture containing text

Description automatically generated

80-20

Text

Description automatically generated

90-10

A picture containing text

Description automatically generated

Random Over-sampling

70-30

A picture containing text

Description automatically generated

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Neural Network

SMOTE

70-30

Text

Description automatically generated

80-20

Text

Description automatically generated

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Random Over-sampling

70-30

Text

Description automatically generated with medium confidence

80-20

A picture containing text

Description automatically generated

90-10

A screenshot of a computer

Description automatically generated with medium confidence

# Appendix C

*Under-sampling Results*

Logistic Regression

70-30

Text

Description automatically generated

80-20

Text

Description automatically generated

90-10

A screenshot of a computer

Description automatically generated with medium confidence

K-Nearest Neighbours

70-30

Text

Description automatically generated with low confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

Text

Description automatically generated with medium confidence

Random Forest

70-30

Text

Description automatically generated

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

Text

Description automatically generated

Support Vector Machine

70-30

A picture containing text

Description automatically generated

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Decision Tree

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

Neural Network

70-30

A screenshot of a computer

Description automatically generated with medium confidence

80-20

A screenshot of a computer

Description automatically generated with medium confidence

90-10

A screenshot of a computer

Description automatically generated with medium confidence

# Appendix D

*Combination sampling Results*

Logistic Regression

|  |  |
| --- | --- |
| 20-80 | 20-90 |
|  |  |
| 30-80 | 30-90 |
|  |  |
| 40-80 | 40-90 |
|  |  |
| 50-80 | 50-90 |
|  |  |
| 60-80 | 60-90 |
|  |  |
| 70-80 | 70-90 |
|  |  |
|  | 80-90 |
|  |  |

K-Nearest Neighbours

|  |  |
| --- | --- |
| 20-80 | 20-90 |
|  |  |
| 30-80 | 30-90 |
|  |  |
| 40-80 | 40-90 |
|  |  |
| 50-80 | 50-90 |
|  |  |
| 60-80 | 60-90 |
|  |  |
| 70-80 | 70-90 |
|  |  |
|  | 80-90 |
|  |  |

Random Forest

|  |  |  |
| --- | --- | --- |
| 20-80 | 20-90 | |
|  |  | |
| 30-80 | | 30-90 |
|  | |  |
| 40-80 | | 40-90 |
|  | |  |
| 50-80 | | 50-90 |
|  | |  |
| 60-80 | | 60-90 |
|  | |  |
| 70-80 | | 70-90 |
|  | |  |
|  | | 80-90 |
|  | |  |

Support Vector Machine

|  |  |  |
| --- | --- | --- |
| 20-80 | | 20-90 |
|  | |  |
| 30-80 | 30-90 | |
|  |  | |
| 40-80 | 40-90 | |
|  |  | |
| 50-80 | 50-90 | |
|  |  | |
| 60-80 | 60-90 | |
|  |  | |
| 70-80 | 70-90 | |
|  |  | |
|  | 80-90 | |
|  |  | |

Decision Tree

|  |  |
| --- | --- |
| 20-80 | 20-90 |
|  |  |
| 30-80 | 30-90 |
|  |  |
| 40-80 | 40-90 |
|  |  |
| 50-80 | 50-90 |
|  |  |
| 60-80 | 60-90 |
|  |  |
| 70-80 | 70-90 |
|  |  |
|  | 80-90 |
|  |  |

Neural Network

|  |  |  |
| --- | --- | --- |
| 20-80 | | 20-90 |
|  | |  |
| 30-80 | 30-90 | |
|  |  | |
| 40-80 | 40-90 | |
|  |  | |
| 50-80 | 50-90 | |
|  |  | |
| 60-80 | 60-90 | |
|  |  | |
| 70-80 | 70-90 | |
|  |  | |
|  | 80-90 | |
|  |  | |