ARCADA

Dataset Imbalance treatment with re-samplers pipeline

Ndifreke OKORIE

Master's Thesis
Master of Engineering - Big Data Analytics
June 2, 2022

MASTER'S THESIS	
Arcada University of Applied Sciences	
Degree Programme:	Master of Engineering - Big Data Analytics
Identification number:	27993
Author:	Ndifreke OKORIE
Title:	Dataset Imbalance treatment with re-samplers pipeline
Supervisor (Arcada):	Amin Majd
Commissioned by:	

Abstract:
Imbalance in dataset is an age long thing and it is receiving lots of attention because of how it impacts the outcome of models. Imbalance in the sample simply means a class of sample is over-represented while the other is under-represented. A lot of data analyst use over-sampling and under-sampling, which are some of the methods used in balancing the samples. In this thesis we present both, a derivative of one of them (called SMOTE -Synthetic Minority Oversampling Technique-) so has to see their performance as against that of combining two re-sampling techniques in pipeline. This work compares and contrasts between these standalone and ensembled techniques result, note the pros and cons of each method and show that the ensembled method works to the advantage of the minority class by repeating the experiment with a second severely imbalanced dataset which gives credence to the result because of consistency.

Keywords:	Imbalance Sampling Pipeline
Number of pages:	49
Language:	English
Date of acceptance:	30.04.2022

CONTENTS

FIGURES

TABLES

ABBREVIATIONS

AI	Artificial Intelligence
ROS	Random Over Sampling
RUS	Random Under Sampling
SMOTE	Synthetic Minority Over-sampling Technique
RUSAMP	Random Under Sampling
ROSAMP	Random Over Sampling
RORUSAMP	Random Over and Random Under Samplings
RUROSAMP	Random Under and Random Over Samplings
SMRUSAMP	SMOTE and Random Under Samplings
RUSMSAMP	Random Under and SMOTE Samplings
G-Mean	Geometric Mean
AUC	Area Under the ROC Curve
ETC.	Etcetera
SASYNO	Self-Adaptive Synthetic Over-Sampling
MCDM	Multi-Criteria Decision Making

FOREWORD

With respect to this thesis, it has been shown that when sampling techniques are ensembled at the data level, they can significantly improve the classification results of the positive class. Though the negative class may remain high but not changed positively and in most cases drops a little.

This is because, judging from the results obtained in this work where classification accuracies of the minority classes were consistently elevated by as much as 4% for two different severely imbalanced datasets. One could then draw an inference that the tweaking of the sampling-strategy parameter of ensembled random over-sampling and random under-sampling algorithms, with a basic model like XGBoost could fetch encouraging results than their standalone or the variant SMOTE, when python pipeline library is in use.

1 INTRODUCTION

1.1 Research Background

In many real-world classification domains, most examples are from one of the classes. In binary classification, it is typically the minority (positive) class that the practitioner is interested in. Imbalance in the class distribution often causes machine learning algorithms to perform poorly on the minority class. In addition, the cost of misclassifying the minority class is usually much higher than the cost of other misclassifications. Therefore, a natural question in machine learning research is how to improve upon the performance of classifiers when one class is relatively rare.

This kind of scenario cannot be avoided as we now leave in a data driven world; where it must be mined, pre-process and modeled. This informs why we are looking for a better way to handle these imbalances before they are passed through models.

A common solution is to sample the data, either randomly or intelligently, to obtain an altered class distribution. Numerous techniques have been used by experts, but we are set out in this thesis not to see which one is best but how to handle the datasets (harnessing the power of both early known techniques) especially as we handle life threatening datasets which may have very little or no tolerance for error.

Some researchers have experimentally evaluated the use of sampling in data-sampling arena as they attempt to handle imbalance in datasets or learn from it Barandela et al. (2004), Barandela et al. (2003). And the techniques they used will always be categorize into the two broad groups names:

a) Over-sampling (replicates examples in) the minority-class

b) Under-sampling (eliminates examples in) the majority class

Though some have tried to add a third which is "Internally biasing the discrimination-based process so as to compensate for the class imbalance", but basically it will still lean towards one of the groups listed above.

As such to tackle class imbalance, long-established methods such as random under-sampling, SMOTE (Vuttipittayamongkol & Elyan (2020)), were still used in many recent studies. Although improvements in results were reported, they have been constantly outperformed by newer methods. Which is observed by how Vuttipittayamongkol et al. [cited by: Vuttipittayamongkol & Elyan (2020)] designed a scoring function that assigned ranking to differentiate between minority class and majority class instances. One of the latest techniques, Generative Adversarial Net (GAN), was employed to synthesize minority class instances.

1.2 Problem Statement

And as upheld empirically by Barandela et al. (2004); Results indicate that, when the imbalance is not very severe, techniques for appropriately under-sampling the majority class is the best option. Only when the majority/minority ratio is very high it is required to over-sampling the minority class. Notwithstanding all these efforts, whether the sampling technique employs up-sampling or down-sampling; the drawbacks of each is infused into the data.

Critical datasets for example history of various patients' heart disease pre-symptoms which is needed to build models to effectively predict if future patients having certain symptoms will eventually have heart disease, needs to be very accurate. But majority of the times these samples come with lots of class imbalances and needs to be treated. Now, the popular way of handing this according to sampling technique listed by Van Hulse et al. (2007); is data analysts randomly using one of those techniques even if they do not have domain expertise.

The challenge is that if they happen to use the over-sampling techniques, dummy values will be randomly added (which were not originally in the controlled data collection phase), hence adding noise to the sample which may mislead the model. And if under-sampling is employed important sample may be thrown away, either ways havoc causing is inevitable.

1.3 Research Questions

This work will seek to answer the questions:

i. Will models perform better when each sampling techniques (Over and Under Sampling) are placed in a pipeline to handle the imbalance together before passing the result onto a model? ii. Is it possible to further tweak this process for better model efficiencies? iii. Could there be a way to infuse some domain specific parameter tweaking as some compensating factor during this re-sampling?

We will hope that this will be better than the current way of handling imbalance at the data level using standalone (one of the techniques), as it is done by most people today.

1.4 Objective of Research

There has been python libraries and codes that allow the piping of these sampling techniques individually as they are passed through models.

Seeing that imbalance is a problem that affect generally machine learning algorithms in every field (Mena & Gonzalez 2006); Analysts have been known to use either of the techniques to handle imbalances in production, but the result is somewhat biased by the single technique used for obvious reasons. This study is geared towards showing how to effectively resolve this issue using an approach that is unbiased. In this light, the objective of this study is to harness the power of every technique of handling imbalances collectively as it concerns very important datasets (bearing in mind the impact errors in models can cause to lives), so models can predict better

1.5 Significance of Work

If question one and any other in our Research Questions section is successfully answered, this work would have greatly and positively impacted: i. Critical datasets industries such as the medicals, financial, auto, etc., which have minute or zero error tolerance because of human lives. ii. Domain critical datasets; needing domain expertise, which many data analytics may lack.

1.6 Scope of Research

This thesis without any other contribution would have exclusively leaned towards the use of majority class intelligently and randomly under-sampled (by removing redundant and noisy datasets, from the sample), bringing the majority class at par with the minority (original sample) without adding any extraneous data).

This technique may not be without its downsides, as the minority dataset may not be enough to make the model generalize effectively; but nonetheless, the benefits of under-sampling instead of over-sampling is that we know and trust our dataset and can depend on the models trained by this dataset. The result will be a reasonable prediction upon an original dataset that was carefully collected and preserved.

In the most basic form according to Van Hulse et al. (2007); in one of the earliest attempts to improve upon the performance of random re-sampling, Kubat and Matwin Kubat et al. (1997), proposed a technique called one-sided selection (OSS). One-sided selection attempts to intelligently under-sample the majority class by removing majority class examples that are considered either redundant or 'noisy.' In the light of the foregoing, you will agree with me that the scope of this research is already portrayed as binary classification.

2 RELATED WORK

Past works on imbalance dataset classification has been around data level and algorithm level (Lin et al. 2020). Data level seeks to balance the class distribution by modifying the training samples using the already mention up or down techniques and their variants. While the algorithm level methods highlight the importance of minority class by making better the existing algorithms, using techniques like cost-sensitive learning, ensemble learning, and decision threshold adjustment.

2.1 Conceptual Framework

Having been in the field of data science for about four years now, I have come to the realization that data acquisition seems to be the most challenging of all parts of data analytic. The reason being that real world data in most cases does not come complete in its class distribution according to Zhang et al. (2018), that is, there exists imbalances a lot of the times. That is why this project has been embarked upon to fine a proper way to handle this better than the existing conventional ways. And despite high interests in classification of critical life changing data, the common issue of imbalanced class distributions is not often addressed. This is evidenced by a review paper discussing existing methods used for dataset classification, which shows that Only 1 out of 71 proposed solutions considered the class imbalanced issue, (Vuttipittayamongkol & Elyan 2020). But there are two basic methods for re-sampling according to Makki et al. (2019) which cause the class distribution to become more balanced. Nevertheless, both strategies have shown important drawbacks, (Barandela et al. 2003).

By way of explanation, Imbalanced training sample means that one class is represented by a large number of examples while the other is represented by only a few. It has been observed that this situation, which arises in several practical domains, may produce an important deterioration of the classification accuracy, with patterns belonging to the less represented classes (Barandela et al. 2004).

Zhang et al. (2018) in trying to look for a better imbalance handling techniques using a cost-sensitive deep belief network recognizes SMOTE as one of the data level re-sampling techniques as divides the re-sampling into two broad types namely: data level and algorithm level re-sampling.

2.2 Empirical Reviews

In their work, Gu et al. (2020) identified the three ways imbalance in dataset could be handled as 1) data sampling, 2) cost-sensitive learning and 3) algorithmic modification. Their paper stated that data sampling approaches are the dominant solutions to address the class imbalance problem because they are more generic and can be employed by standard classification methods; as such a variant of SMOTE, named SASYNO (Self-Adaptive Synthetic Over-Sampling), was used to obtain a better metric result.

The financial domain known for how delicate it is, Song & Peng (2019) proposed the usage of multi-criteria decision making (MCDM)-based approach to evaluate imbalanced classifiers in credit and bankruptcy risk prediction by considering multiple performance metrics simultaneously. In doing this SMOTE was used in conjunction with other techniques.

Quite a number of research recommended over-sampling though they emphasized on the variant of which are: Yan et al. (2019) and Rahman & Davis (2013), in fact Akbani et al. (2004) Used SMOTE to make the positive instances more densely distributed in order to make the boundary more well defined.

But there are others who took the under-sampling way, as noted when using "One-Sided Selection" (OSS) which is a combination of Tomek Link (TL) followed by the application of the Condensed Nearest Neighbour (CNN) rule. This TL method is used to remove noise, and borderline majority class samples and then CNN is used to discard samples from the majority class that is redundant and far away from the decision border, Maheshwari et al. (2018). Even Tang et al. (2009) compared under-sampling with various state-of-the-art approaches on a variety of datasets.

Monotonic class imbalance problem refers to a severe loss of classification accuracy of certain classes due to their under- representation in the training dataset. That is, some classes have a lot less instances than others, which affects their identification by standard classifiers. These under-represented classes are known as minority or positives, whilst the rest are referred to as majority or negative classes. In many real-life applications, the most important classes are usually the most imbalanced. Therefore, the misclassification of these classes entails greater costs. This was put forth by González et al. (2019); and he used under-sampling in his work of solving this problem.

Empirical studies have shown that there are arising issues with these two basic approaches of re-sampling imbalances in data sampling; Under-sampling the majority class and also internally biasing the discrimination process according to Barandela et al. (2004) (resulting to overall number of records in the training set being greatly reduced), even though Sun et al. (2019) and Basso et al. (2021), argued that after different oversampling methodologies are analyzed to balance the training data, it was found that the Deep Convolutional Generative Adversarial Networks technique with random under-sampling presented better results. This means that during classification, training time is also greatly reduced. Since we may at times be dealing with very high dimensional datasets, there is a significant savings in memory as well. However, because we are eliminating members from the majority class, it is possible that we will lose a lot of valuable information if we eliminate documents that could be useful to our classifier in building an accurate model.

2.3 Theoretical Framework

Theoretically, one of the problems with random under-sampling is that one cannot control what information about the majority class is thrown away. In particular, very important information about the decision boundary between the minority and majority class may be eliminated. Despite this controversy, random under-sampling has empirically been shown to be one of the most effective re-sampling methods. In particular, only a few of the more sophisticated under-sampling methods have outperformed random under-sampling in empirical studies Liu (2004). The challenges also with over-sampling is that we greatly increase the size of the training set. Thus, we also increase training time and the amount of memory required to hold the training set. Since we are dealing with very high dimensional datasets, we need to be careful how we proceed to keep time complexity and memory complexity under reasonable constraints (Liu 2004). Furthermore, since over-sampling typically replicates examples in the minority class, over-fitting is more likely to occur.

However, a suggestion was put forward by Barandela et al. (2003) which is combining under-sampling and over-sampling methods and instead of over-sampling by merely replicating positive prototypes, they form new minority instances by interpolating between several positive examples that lie close together. This sounds like a brilliant idea worthy of further pursuit.

As such this study has a design and approach intent or resolution to truly following both basic techniques. That is apply both basic techniques on every dataset that is intended for this analytic. But thereafter, build a generator or pipeline, that will combine both results before modeling. Whereby harnessing the potentials of each method of re-sampling and one will compensate for the other in the final model result.

2.4 Study Gap

The throwing away of samples when datasets are under-sampled and the adding of noise or useless and approximated samples to the data when datasets are over-sampled, is the gap in imbalanced dataset classification which this work hopes to fill by proving that a methodical, technical and an unusual ensemble of up and down sampling techniques Dou et al. (2022) will produce a model that does better than standalone samplings.

When data experts treat imbalance in dataset they most times just use on of the said re-sampling techniques perhaps because of the ease of usage or a good metric result achieved by it. Not minding if the inherent bias of the techniques (the weakness) has been infused in the process. But as better ways of harnessing the combined strengths of both methods are used, it is believed that the pros will out-weigh the weaknesses in the sampling methods.

That is why the thesis intends to run several experiments using the standalone re-sampling techniques alone, then combined them while looking out for the outcome on the minority class. So that recommendation could be made to what process produced the best effect on greatly imbalanced dataset. So, this thesis will test to see that ensembling traditional data-level re-sampling techniques like RandomOverSampling, RandomUnderSampling and SMOTE in a pipeline, will increase the accuracy metric of the negative class (or minority) better than when any one of them is used alone.

3 RESEARCH METHODOLOGY

As the trail of thought in this work goes datasets with class imbalances are been dealt with, meaning one of its binary class is a lot represented than the other (or what could be described as being highly imbalanced) Turlapati & Prusty (2020). The focus then in this research treats this imbalance at the data level and not the algorithm level.

First, there is need to develop a method to correctly measure model accuracy. Since the target variable has a class imbalance, there is need for something a little bit more complex than simple cross-validation. The following scheme will be used:

i. Randomly split data on train and test sets.

ii. Handle class imbalance in train set with one of three available methods: random under-sampling, random over-sampling or over-sampling with SMOTE algorithm.

iii. Train and test the model, separately measuring the accuracy of determining a recession and accuracy of determining the absence of a recession (and stroke/absence of stroke for the second dataset)

iv. Repeat this multiple times and average the results.

XGBoost, RandomForest and Ridge will be used as models of choice.

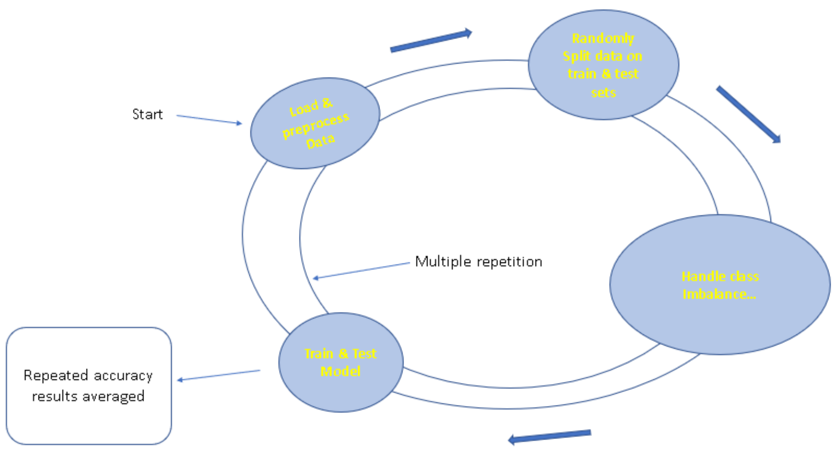

Start
Load & preprocess Data
Randomly Split data on train & test sets
Handle class Imbalance...
Train & Test Model
Multiple repetition
Repeated accuracy results averaged

Figure 1. Research Methodology Flowchart

3.1 Research Design

3.1.1 Initial baseline

To make the design more objective and clearer two highly imbalanced datasets will be used. The first will be used in the process of creating the sampling system and algorithm metric evaluation, which in this case will be model accuracy since accuracy improvement is the reason for re-sampling an imbalance dataset (Xia et al. 2021) and not to increase or reduce the data size or improve efficiency. When the program flow is deemed ok the second dataset will be passed through the same process (perhaps with a little more preprocessing, because of it features structure) to ascertain the consistency of the results obtained, as encouraged by Vanhoeyveld & Martens (2018) experiments with multiple datasets.

What is looked for here as re-sampled dataset is passed through each of the three models is to have a baseline accuracy belonging to r-esampling the dataset using only one technique at a time.

As portrayed in figure 2 below, the data is to be separately passed through an up-sampling technique (in this case, ROS and SMOTE) as shown by the red arrow from the data source. The next red arrow is the output that is fed into the models and out comes the result shown by the last red arrow and the red accuracy metric.

The same goes for the green arrow through the down-sampler which is a RUS, and the green accuracy is obtained.

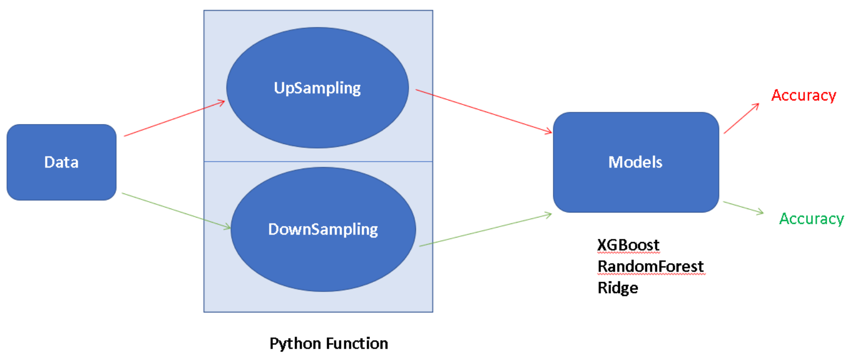


Figure 2. Single sampling technique

This is done to get the baseline metric results of using the over-sampling and under-sampling techniques separately.

3.1.2 Final baseline for comparison

To answer our research question one a pipeline is built also to ensemble these basic techniques of up-sampling and down-sampling, and the results from them passed through each of the three models to compare the new accuracies with the first baseline accuracies. For clarity they are bulleted as:

i. A python pipeline to ensemble ROS and RUS techniques will be built ii. Anther to ensemble SMOTE and RUS techniques will be built iii. Two more will be built reversing the positions in point i. and ii. above (i.e. RUS and ROS, then RUS and SMOTE)

Each of the four built pipelines will be passed through the three selected models to also see their accuracies, as depicted in Figure 3 below.

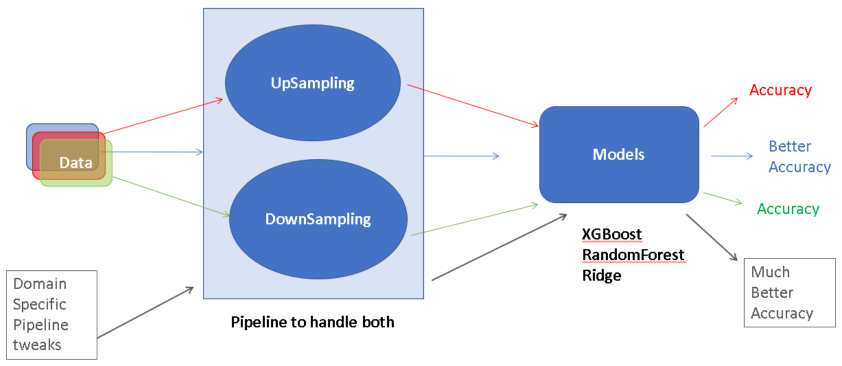


Figure 3. Re-sampling techniques ensembled

3.2 Datasets Acquisition and structure

The datasets were obtained from the public domain (Kaggle), at the following locations:

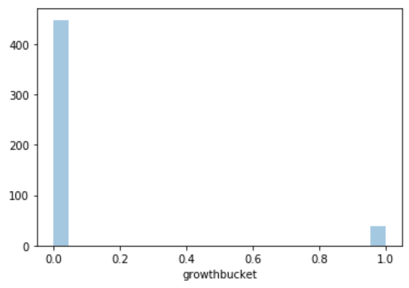
i. Africa recession dataset https://www.kaggle.com/chirin/african-country-recession-dataset-2000-to-2017 and

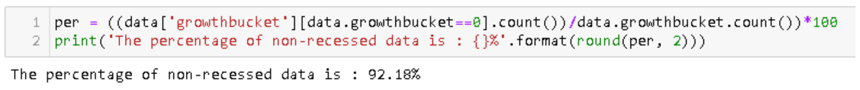
ii. Cerebral Stroke dataset https://www.kaggle.com/shashwatwork/cerebral-stroke-predictionimbalaceddataset.

And the datasets can be visualized as shown in the next section.

3.2.1 Datasets description

AFRICAN RECESSION: Thus, we can say that if you are from a government, investment fund or an international company, it will be very important for you to know whether the economy of the country you are interested in is in a state of recession.





```
per = ((data['growthbucket'][data.growthbucket==0].count())/data.growthbucket.count())*100
print('The percentage of non-recessed data is : {}%'.format(round(per, 2)))
```

```
The percentage of non-recessed data is : 92.18%
```

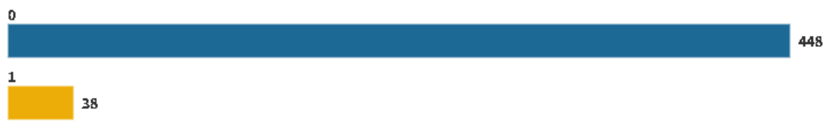


Figure 4. African recession dataset

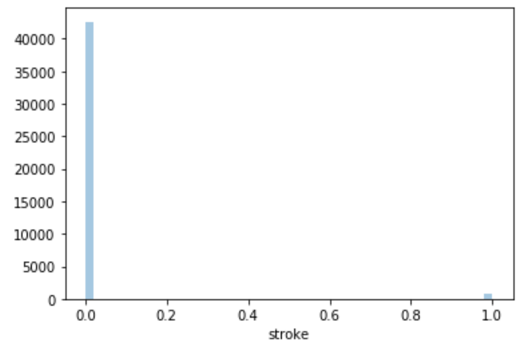
A few words about the success criteria of the model. When this analysis commenced, building a model as accurate as possible was thought to be the task. However, as will be seen shortly, imbalance in target variable classes and the small size of a dataset leads to searching for a compromise between the accuracy of determining the recession and the accuracy of determining the absence of a recession. The optimal relationship between them will depend on the cost of the mistake in both cases, which is difficult to evaluate due to poor knowledge of the domain. Reflecting on how this model can be used, a conservative strategy was settled for: the accuracy of determining the absence of a recession should be quite high (at about 85%, since it has most of the records), and the accuracy of determining a recession should be as high as possible.

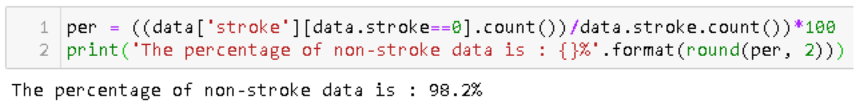
The African recession dataset contains 486 samples and 50 features including the target column, which can be seen in figure 4 to be greatly biased.

CEREBRAL STROKE: A stroke, also known as a cerebrovascular accident or CVA is when part of the brain loses its blood supply and the part of the body that the blood-deprived brain cells control stops working. This loss of blood supply can be ischemic because of lack of blood flow, or hemorrhagic because of bleeding into brain tissue. A stroke is a medical emergency because strokes can lead to death or permanent disability. There are opportunities to treat ischemic strokes, but that treatment needs to be started in the first few hours after the signs of a stroke begin. That is why it is essential to correctly predict a condition by handling the dataset in the right way.

The cerebral Stroke dataset consists of 43,400 samples and 12 features including the target column which is also greatly imbalanced.

The same strategy of evaluation will be adopted as in the first dataset since their imbalance ratio is almost the same.





```
per = ((data['stroke'][data.stroke==0].count())/data.stroke.count())*100
print('The percentage of non-stroke data is : {}%'.format(round(per, 2)))
```

```
The percentage of non-stroke data is : 98.2%
```

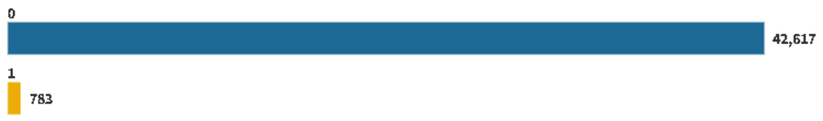


Figure 5. Cerebral stroke dataset

3.3 Method Justification

Data level as one of the ways of handling class imbalance was chosen over the algorithm level, because according to Nnamoko & Korkontzelos (2020) :

i. It is classifier-independent and relatively easy to apply because it focuses on data preprocessing techniques ii. It does not have relatively a high computational cost like the algorithm level iii. It is generally the largest used method, even learners go there first so that if it is improved upon will impact a lot of users iv. Data level modifies the training set such that it is suitable for any standard learning algorithm

Since classification problems with imbalanced datasets widely exist in real world Zhu et al. (2020), models will be chosen that can broadly generalize as one of the easily learnable and standard ones. And the choices of XGBoost, RandomForest and Ridge models for our metric evaluation is that they are known to mitigate the influence of skewed classes in training sets (that is, able to handle imbalance data).

4 EXPERIMENTS

The experimental environment needs to be noted according to Feng et al. (2020), perhaps for reproducible reasons, and it is:

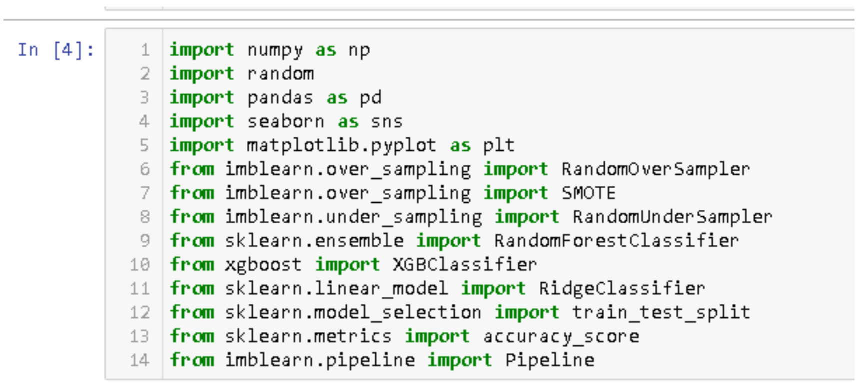
i. Dell x64-based PC, Latitude E7250 with Intel core i7, 16GB RAM, CPU@2.60GH, 2601Mhz 2 Cores(s) 4 Logical processor(s)

ii. Windows 10 Pro Operative System, Version 10.0. 19043 Build 19043

iii. Jupyter Notebook installed via Anaconda 3 package

4.1 Exploratory data analysis

Important python 3 libraries were loaded in Jupyter Notebook including the re-sampler libraries (RandomOverSampler, SMOTE and RandomUnderSampler). The three models libraries were loaded too.



```
In [4]:
import numpy as np
import random
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from imblearn.over_sampling import RandomOverSampler
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.linear_model import RidgeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from imblearn.pipeline import Pipeline
```

Figure 6. Python libraries

4.1.1 African Recession Dataset

Since the data had no null value and are all numeric (all float except for the target that is integer), the structure was checked with a seaborn distribution plot and found the un-recessed percentage to be 92.18, which means the recessed percentage is 7.82 .

4.1.2 Cerebral Stroke Dataset

The dataset was found to have 98.2 percent non-stroke data after visualization leaving a meagre 1.8 percent to stroke data.

4.2 Preprocessing

Much preprocessing was not done on the African recessed dataset except for correlation visualization and the removal of features with correlation coefficient less than 0.86 (as shown below), after which the data shape was (486, 34) .

```
In [16]:  plt.figure(figsize=(15, 13))
          sns.heatmap(data=data.corr())

Out[16]:  <matplotlib.axes._subplots.AxesSubplot at 0x27af7af1dc8>
```

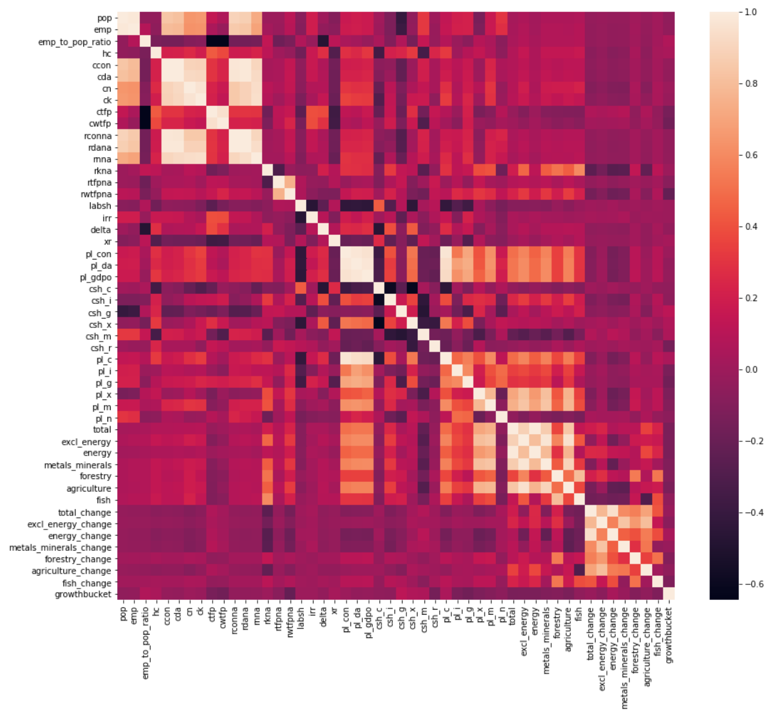


Figure 7. African Recession dataset correlation heat map

But Cerebral stroke dataset had null values and samples containing them were dropped. Five features had ordinal data and had to be treated by replacement with integers. Correlation (as shown below) was used to eliminate features with coefficient less than 0.81, leaving the shape as (29072, 11) .

```
In [17]: plt.figure(figsize=(10, 8))
         sns.heatmap(data=data.corr())
Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x19fb82e2208>
```

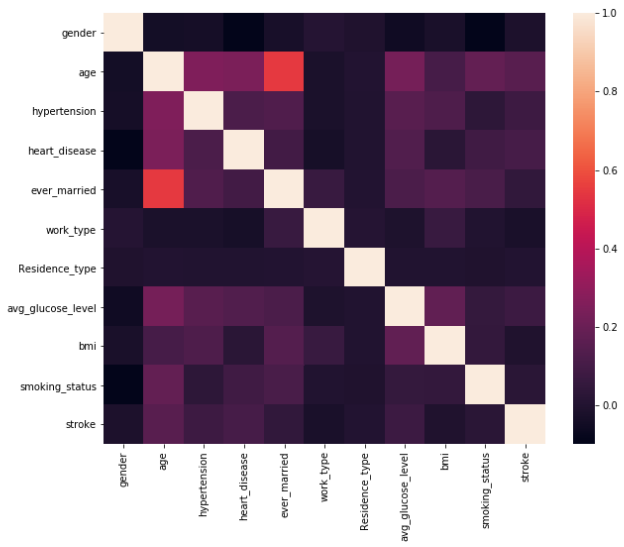
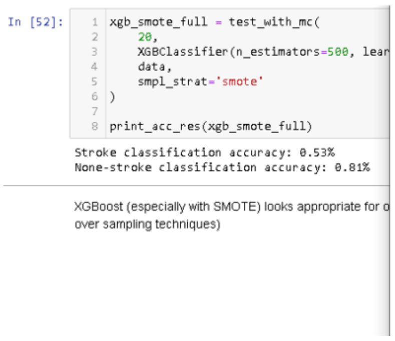


Figure 8. Cerebral Stroke dataset correlation heat map

4.3 Model building

Python functions for the modelling and printing process was written and data was passed through the samplers and the models in this fashion.

Firstly, random under-sampling, random-over sampling and smote was done and passed through xgboost classifier, secondly, randomforest classifier and lasted ridge classifier. This gave the baseline for comparison that will be done much later after the ensembling (or conbining) of sampling techniques phase; It was found that randomforest did best using smote sampling technique for African recession dataset but xgboost using smote was found to be well performing for Cerebral stroke data.

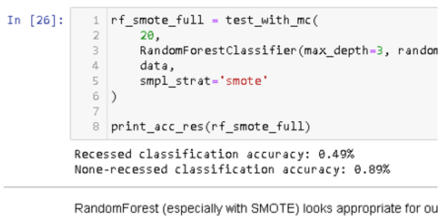


```
In [52]: xgb_smote_full = test_with_mc(
             20,
             XGBClassifier(n_estimators=500, lear
             data,
             smpl_strat='smote'
         )

         print_acc_res(xgb_smote_full)

         Stroke classification accuracy: 0.53%
         None-stroke classification accuracy: 0.81%
```

XGBoost (especially with SMOTE) looks appropriate for o
over sampling techniques)



```
In [26]: rf_smote_full = test_with_mc(
             20,
             RandomForestClassifier(max_depth=3, random
             data,
             smpl_strat='smote'
         )

         print_acc_res(rf_smote_full)

         Recessed classification accuracy: 0.49%
         None-recessed classification accuracy: 0.89%
```

RandomForest (especially with SMOTE) looks appropriate for ou

Figure 9. Outstanding results

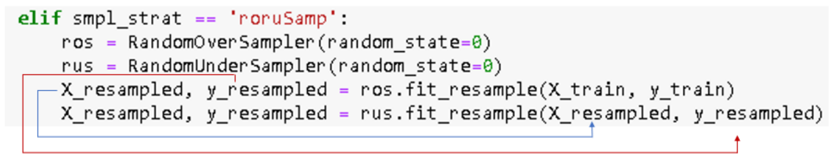
Since the aim is to keep the majority class fairly stable but improve upon the minority according to Zhang & Chen (2019) the next step of ensembling of the three sampling data level techniques that is under consideration, in the order: randomOver-sampling with randomUnder-sampling, randomUnder-sampling with randomOver-sampling, smote-sampling with randomUnder-sampling, randomUnder-sampling with smote-sampling (making four orders of ensembles). Care was taken to make sure the ordering was always with an up-sampler and a down-sampler. Again, RandomForest classifier was found to be best performing (using smote-sampling with randomUnder-sampling) for African Recession dataset after passing these ensembled techniques outputs into the models (but for Cerebral stroke data, it was Ridge classifier with the same ensemble type).

It will be interesting to mention that the process of ensembling the techniques was done using a python function. So, after realizing that RandomForest and Ridge were best through this ensembling method for African Recessed data and Cerebral stroke data respectively, to be more objective, they were adopted for used for the different datasets in the next section of experiment. Which was passing the datasets through an ensembling done with python pipeline library. Once again RandomForest got the better of the three models with (smote-sampling with randomUnder-sampling) for African Recession and Ridge did so for Cerebral stroke dataset (with the same ensemble type as that of the African recessed dataset) so a twist was thought of, which is expounded upon in the method adjustment section.

It was mentioned earlier that the evaluation metric will be accuracy, this is for better judgement as two results (accuracy of minority class prediction and that of majority class) were retrieved every time a model ran; as contrasted to the common way of measuring imbalance via G-mean and AUC according to Cao et al. (2013).

4.4 Method adjustment

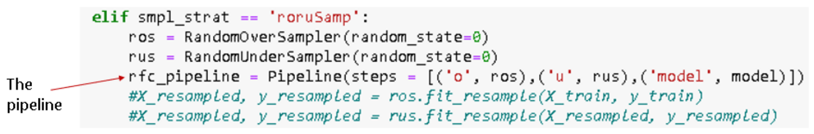
It was found that during the ensembling of the three used sampling techniques in data level sampling used in this work, data flows linearly. This here means that, the output sampled data coming out of the first sampler after it had been fed with the original dataset is being fed into the second sampling techniques in an ensemble so it can in turn perform its sampling. Therefore, it was decided that to see what happens, an adjustment to this experiment will be done by feeding the original dataset into two samplers separately then the outputs are combined before passing it through the models. Codes to the three approaches are displayed below.



```
elif smpl_strat == 'roruSamp':
    ros = RandomOverSampler(random_state=0)
    rus = RandomUnderSampler(random_state=0)
    X_resampled, y_resampled = ros.fit_resample(X_train, y_train)
    X_resampled, y_resampled = rus.fit_resample(X_resampled, y_resampled)
```

Figure 10. Function mimicking pipeline

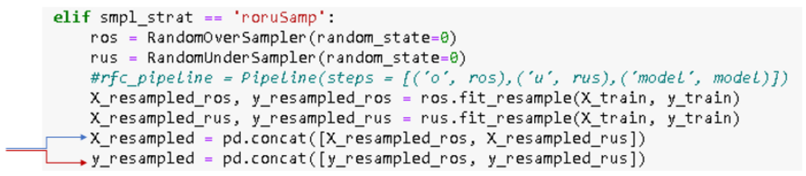
——————————-



```
elif smpl_strat == 'roruSamp':
    ros = RandomOverSampler(random_state=0)
    rus = RandomUnderSampler(random_state=0)
    rfc_pipeline = Pipeline(steps = [('o', ros),('u', rus),('model', model)])
    #X_resampled, y_resampled = ros.fit_resample(X_train, y_train)
    #X_resampled, y_resampled = rus.fit_resample(X_resampled, y_resampled)
```

Figure 11. Python pipeline

——————————-



```
elif smpl_strat == 'roruSamp':
    ros = RandomOverSampler(random_state=0)
    rus = RandomUnderSampler(random_state=0)
    #rfc_pipeline = Pipeline(steps = [('o', ros),('u', rus),('model', model)])
    X_resampled_ros, y_resampled_ros = ros.fit_resample(X_train, y_train)
    X_resampled_rus, y_resampled_rus = rus.fit_resample(X_train, y_train)
    X_resampled = pd.concat([X_resampled_ros, X_resampled_rus])
    y_resampled = pd.concat([y_resampled_ros, y_resampled_rus])
```

Figure 12. Python concatenation

Three ensemble ways in this experiment

In doing this using RandomForest and Ridge classifiers (for African Recession and Cerebral Stroke datasets respectively) since they were adopted for doing well previously, it was found that the combination of smote-sampling with randomUnder-sampling outputs did best for both datasets and so, it is therefore imperative to note that all these were achieved using the sampling techniques with their default parameters initialized. The models were use with their default parameters since according to Shah (2020) RandomForest classifier works well and best in this status.

But for the sake of being thorough, the process above was pursued in a further experiment, according to Huang et al. (2016) stating that other researchers did some parameter adjustment in their algorithm level imbalance treatment. A leave was borrowed from this as the sampling-strategy of the sampling techniques were adjusted.

5 RESULTS

5.1 African Recession Dataset Results

5.1.1 Default parameters used

Data Class Imbalance [Recessed: 92.18%, Non-Recessed: 7.82%]

Data shape: 486, 50

Model Metric: Accuracy

Running the experiment as previously explained in the last section, a series of results were obtained as portrayed by table 1. As seen the baseline after treating the dataset in question with RUSAMP, ROSAMP and SMOTE and passing the output through the three models, was Recession 49% and non-Recession 89% (with SMOTE and RF being the best sampling technique and Model). This result is consistent with our dataset seeing the gross imbalance that initially existed.

The next phase was to do an ensembled of these techniques and using them to treat the imbalance, and again RandomForest was best (with SMRUSAMP ensemble done by writing a python function), with the results 49% and 88%. The result was not better than the baseline (our yardstick of comparison like Zolanvari et al. (2018) also did) as expected by the research question, so, the same set of ensembled techniques were done using a real python pipeline but the best result was not different from the last one.

The last batch of results shows a little improvement in the recessed accuracy at 51% supporting our hypothesis that ensembles of re-sampling techniques should do better than standalone (perhaps because the ensembled re-sampling techniques each acted on the original data before being concatenated, a process that logically should be better than passing the output of one techniques as an import to the other (which was clearly what the first function and the pipeline were doing). But it is noteworthy to say that the minority sample (or the positive class) only benefited from the approach, while the other fell short of expectation.

Do find the link to the jupyter notebook code in Appendix A

Table 1. Used sampling techniques default parameter on African Recessed dataset imbalance treatment

Sampling Techniques	**XGBoost**		**RandomForest**		**Ridge**	
	Recessed	Non-Recessed	Recess.	Non-Recess.	Recess.	Non-Recess.
Single Techniques						
rusamp(Under Sampling)	66%	64%	68%	68%	66%	63%
rosamp(Over Sampling)	33%	90%	38%	92%	56%	76%
smote	31%	89%	49%	89%	58%	76%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	26%	97%	41%	92%	56%	76%
ruroSamp(Under+Over Sampling)	66%	64%	60%	69%	66%	63%
smruSamp(smote+Under Sampling)	30%	96%	49%	88%	54%	78%
rusmSamp(Under+smote Sampling)	66%	64%	60%	69%	66%	63%
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)			41%	92%		
ruroSamp(Under+Over Sampling)			60%	69%		
smruSamp(smote+Under Sampling)			49%	88%		
rusmSamp(Under+smote Sampling)			60%	69%		
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)			37%	92%		
ruroSamp(Under+Over Sampling)			39%	92%		
smruSamp(smote+Under Sampling)			51%	88%		
rusmSamp(Under+smote Sampling)			47%	88%		

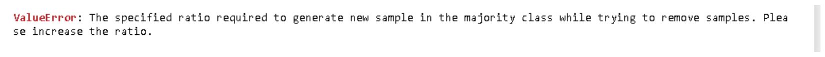
5.1.2 Adjusted parameters 1

To Answer the second research question the re-samplers sampling-strategy parameter was adjusted in the form laid out in the two points below.

i. Over Sampling Stategy = 0.5 when in front of the ensemble and 0.6 when at the back

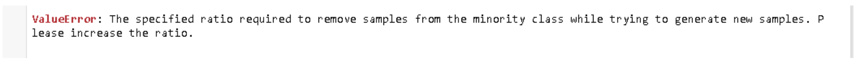
ii. Under Sampling Stategy = 0.5 when in front of the ensemble and 0.6 when at the back

This is so as to place the smaller percentage in front, so as not to spit-out errors as shown in figure 12 and 13



```
ValueError: The specified ratio required to generate new sample in the majority class while trying to remove samples. Plea
se increase the ratio.
```

Figure 13. Error when RO=0.6 and RU=0.5 (and RO is in front)



```
ValueError: The specified ratio required to remove samples from the minority class while trying to generate new samples. P
lease increase the ratio.
```

Figure 14. Error when RU=0.6 and RO=0.5 (and RU is in front)

Errors produced by these ratios

RUSAMP and Xgboost gave the best baseline (50% and 84%), RUROSAMP combined techniques via function with Xgboost gave the next results 52% and 83% improving the minority class prediction by 2% which is really our concern. Pipeline delivered the same as via function but the last batch which is concatenation failed in our expectations. But essentially, answering the second research question of tweaking our sampler with these sets of parameters chosen arbitrarily only produced in the minority class prediction an improvement of 1% over the un-tweaked ensembled samplers

Do find the link to the jupyter notebook code in Appendix A

Table 2. Adjusted techniques sampling-strategy parameter to 0.5 or 0.6 in African Recessed dataset imbalance treatment

Sampling Techniques	**XGBoost**		**RandomForest**		**Ridge**	
	Recessed	Non-Recessed	Recess.	Non-Reccess.	Recess.	Non-Recess.
Single Techniques						
rusamp(Under Sampling)	50%	84%	43%	88%	45%	84%
rosamp(Over Sampling)	31%	93%	23%	98%	38%	89%
smote	39%	93%	28%	97%	34%	90%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	28%	96%	28%	97%	45%	86%
ruroSamp(Under+Over Sampling)	52%	83%	39%	90%	49%	80%
smruSamp(smote+Under Sampling)	35%	95%	33%	95%	43%	86%
rusmSamp(Under+smote Sampling)	54%	81%	38%	90%	47%	81%
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)	28%	96%				
ruroSamp(Under+Over Sampling)	52%	83%				
smruSamp(smote+Under Sampling)	35%	95%				
rusmSamp(Under+smote Sampling)	54%	81%				
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)	26%	97%				
ruroSamp(Under+Over Sampling)	25%	98%				
smruSamp(smote+Under Sampling)	31%	96%				
rusmSamp(Under+smote Sampling)	32%	96%				

5.1.3 Adjusted parameters 2

In the last phase of tweaking the re-samplers parameters, the following measures were chosen.

i. Over Sampling Stategy = 0.1
ii. Under Sampling Stategy = 0.5

It is a bit relaxed in the percentages of re-sampling but stricter because the strategy was consistent over the processes and batched without adjusting the figures when any of the techniques go in front or behind. The strategy figures could be explained as:

(a) Parameter adjusting for over sampler
The over sampled data should not be exactly equal, to avoid overfitting the aim is to over sample the minority class to 10 percent of the majority class (hence the sampling-strategy=0.1; in the oversampling code)

(b) Parameter adjusting for under sampler
The under sampled data should not be exactly equal, to avoid overfitting the aim is to under sample the majority class to 50 percent more than the minority class (hence the sampling-strategy=0.5; in the under-sampling code)

So, the baseline with RUSAMP and Xgboost was 50% and 84%, ensembled via function and Pipeline were the same at 53% and 85% (an obvious improvements of 3% and 1% over the baseline even though the last batch didn't produce any). And as could be seen RORUSAMP was the ensemble that earned this, while the other spat-out the said error (in red, in table 3) mentioned in the previous section. As postulated, further tweaks yielded a better improvement in the minority class prediction and a visible improvement in the majority too. Do find the link to the jupyter notebook code in Appendix A

Table 3. Adjusted techniques sampling-strategy parameter to 0.5 for under-sampling and 0.1 for over-sampling in African Recessed dataset imbalance treatment

Sampling Techniques	**XGBoost**		**RandomForest**		**Ridge**	
	Recessed	Non-Recessed	Recess.	Non-Recess.	Recess.	Non-Recess.
Single Techniques						
rusamp(Under Sampling)	50%	84%	43%	88%	45%	84%
rosamp(Over Sampling)	20%	97%	12%	99%	6%	100%
smote	21%	97%	14%	99%	0%	100%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	53%	85%	34%	94%	44%	84%
ruroSamp(Under+Over Sampling)	err	err	err	err	err	err
smruSamp(smote+Under Sampling)	50%	84%	38%	93%	44%	85%
rusmSamp(Under+smote Sampling)	err	err	err	err	err	err
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)	53%	85%				
ruroSamp(Under+Over Sampling)	err	err				
smruSamp(smote+Under Sampling)	50%	84%				
rusmSamp(Under+smote Sampling)	err	err				
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)	21%	98%				
ruroSamp(Under+Over Sampling)	21%	98%				
smruSamp(smote+Under Sampling)	22%	98%				
rusmSamp(Under+smote Sampling)	22%	98%				

5.2 Cerebral Stroke Dataset Results

5.2.1 Default parameters used

Data Class Imbalance [Stroke: 98.2%, Non-Stroke: 1.8%]

Data shape: 43400, 12

Model Metric: Accuracy

This experimental result was to see if a different dataset and a more seriously imbalanced one for that matter could replicate the little improvement noticed in the experiment conducted using the African recession dataset.

Here the baseline results went to SMOTE and Xgboost at 53% and 81%. The Function and Pipeline ensembles had 59% and 78% with SMRUSAMP again but this time using the Ridge model and the last batch concatenate was even better at 60% and 78%, improving the minority class prediction by 7% even though the majority prediction dropped a little (as shown in table 4).

Do find the link to the jupyter notebook code in Appendix B

Table 4. Used sampling techniques default parameter on Cerebral Stroke dataset imbalance treatment

Sampling Techniques	XGBoost		RF		Ridge	
	Stroke	Non-Stroke	Stroke	Non-Stroke	Stroke	Non-Stroke
Single Techniques						
rusamp(Under Sampling)	74%	69%	79%	71%	81%	72%
rosamp(Over Sampling)	71%	75%	84%	68%	81%	72%
smote	53%	81%	70%	74%	59%	78%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	23%	93%	84%	68%	81%	72%
ruroSamp(Under+Over Sampling)	74%	69%	83%	69%	81%	72%
smruSamp(smote+Under Sampling)	8%	98%	70%	74%	59%	78%
rusmSamp(Under+smote Sampling)	74%	69%	83%	69%	81%	72%
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)					81%	72%
ruroSamp(Under+Over Sampling)					81%	72%
smruSamp(smote+Under Sampling)					59%	78%
rusmSamp(Under+smote Sampling)					81%	72%
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)					81%	72%
ruroSamp(Under+Over Sampling)					81%	72%
smruSamp(smote+Under Sampling)					60%	78%
rusmSamp(Under+smote Sampling)					60%	72%

5.2.2 Adjusted parameters 1

i. Over Sampling Stategy = 0.5 when in front and 0.6 when at the back

ii. Under Sampling Stategy = 0.5 when in front and 0.6 when at the back

This first level of tougher tweak was done with baseline result 52% and 86% going to ROSAMP and Xgboost. Via Function and Pipeline both went to Xgboost again (a powerful python classifier, which can be extended and indeed has been done in Imbalance-XGBoos package according to Wang et al. (2020)) using RUROSAMP at 57% and 81% with a minority class prediction improvement of 5% where concatenation batch fell short.

There is need to be careful here not to assume excellence if accuracy is more than 90% for majority class, because it could fail at some point for severely imbalance data as explained by Brownlee (2020).

Do find the link to the jupyter notebook code in Appendix B

Table 5. Adjusted techniques sampling-strategy parameter to 0.5 or 0.6 in Cerebral Stroke dataset imbalance treatment

Sampling Techniques	**XGBoost**		**RF**		**Ridge**	
	Stroke	Non-Stroke	Stroke	Non-Stroke	Stroke	Non-Stroke
Single Techniques						
rusamp(Under Sampling)	56%	81%	59%	83%	63%	85%
rosamp(Over Sampling)	52%	86%	48%	89%	63%	85%
smote	36%	89%	30%	91%	38%	88%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	17%	95%	63%	84%	70%	82%
ruroSamp(Under+Over Sampling)	57%	81%	65%	83%	70%	82%
smruSamp(smote+Under Sampling)	6%	98%	44%	87%	46%	85%
rusmSamp(Under+smote Sampling)	59%	80%	64%	83%	70%	81%
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)	17%	95%				
ruroSamp(Under+Over Sampling)	57%	81%				
smruSamp(smote+Under Sampling)	6%	98%				
rusmSamp(Under+smote Sampling)	59%	80%				
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)	15%	96%				
ruroSamp(Under+Over Sampling)	16%	96%				
smruSamp(smote+Under Sampling)	6%	99%				
rusmSamp(Under+smote Sampling)	7%	98%				

5.2.3 Adjusted parameters 2

i. Over Sampling Stategy = 0.1

ii. Under Sampling Stategy = 0.5

A further tweaking of the sampler here did not produce improved results for this dataset as the baseline was 59% and 83% and the next two ensemble batches produced 48% and 89%, while the last over fitted on the data (table 5). This could be because of the level of high imbalance that is existing in this data causing a high bias creating a difficulty for system consistency or human explanation Ruf & Detyniecki (2021).

Do find the link to the jupyter notebook code in Appendix B

Table 6. Adjusted techniques sampling-strategy parameter to 0.5 for under-sampling and 0.1 for over-sampling in Cerebral Stroke dataset imbalance treatment

Sampling Techniques	**XGBoost**		**RF**		**Ridge**	
	Stroke	Non-Stroke	Stroke	Non-Stroke	Stroke	Non-Stroke
Single Techniques						
rusamp(Under Sampling)	56%	81%	59%	83%	63%	85%
rosamp(Over Sampling)	3%	99%	0%	1%	0%	1%
smote	4%	99%	0%	1%	0%	1%
Combined Techniques via function						
roruSamp(Over+Under Sampling)	27%	93%	48%	89%	63%	85%
ruroSamp(Under+Over Sampling)	err	err	err	err	err	err
smruSamp(smote+Under Sampling)	20%	94%	36%	90%	47%	86%
rusmSamp(Under+smote Sampling)	err	err	err	err	err	err
Combined Techniques via pipeline						
roruSamp(Over+Under Sampling)			48%	89%		
ruroSamp(Under+Over Sampling)			err	err		
smruSamp(smote+Under Sampling)			36%	90%		
rusmSamp(Under+smote Sampling)			err	err		
Combined Techniques via fn, but manually concatenating results of each technique on original dataset						
roruSamp(Over+Under Sampling)			0%	100%		
ruroSamp(Under+Over Sampling)			0%	100%		
smruSamp(smote+Under Sampling)			0%	100%		
rusmSamp(Under+smote Sampling)			0%	100%		

Using the sampling techniques in it default parametric state saw SMOTE championing either as the baseline standalone or during ensembling because as observed by Kim et al. (2020), it is an up-sampling variant and does data augmentation also. As a result, the main point of our research was answered which essentially is, will models perform better when data sampling techniques are ensembled for data treatment?

6 CONCLUSIONS

As could be seen in the experiment following the methodology, the design was intense (just like Krawczyk (2016) also noted of other researchers) and process followed strictly which led to the results answering the first research question that ensemble data level sampling techniques which is widely used, could improve the positive class:

Baseline (African recessed data) = 49% and 89% Adjusted/ensembled parameters 2 (African recessed data) = 53% and 85% [which is a better ratio] And was achieved by passing into XGBoost model the output of the python pipeline ensembling random over-sampling and random under-sampling.

Baseline (Cerebral stroke data) = 53% and 81% Adjusted/ensembled parameters 2 (Cerebral stroke data) = 57% and 81% [which is a better ratio also] This was achieved by passing into XGBoost model the output of the python pipeline ensembling random under-sampling and random over-sampling.

But care is to be taken because the negative class (or majority) could suffer a little. This is because one of the problems that can arise in classification is the small sample size. This issue is related to the "lack of density" or "lack of information", where induction algorithms do not have enough data to make generalizations about the distribution of samples, a situation that becomes more difficult in the presence of high dimensional and imbalanced data Ramyachitra & Manikandan (2014), which is the condition we have experienced.

It should be said that the three different pipelines used (the written pipeline mimicking function, python pipeline library and the concatenating function) produced great results in different situations. And also noted is the realization that tweaking the re-samplers produced some improvements which if further studied and experimented upon in conjunction with its other parameters (e.g., replacement) could lead to some groundbreaking results. Though a lot of data scientist defaults to using SMOTE which as we have seen can produce stunning results, but there are other methods as well as ensembles that can do better if explore further Luo et al. (2019).

The limitation of this work just like Lu et al. (2019) also had, is that the third research question was not explicitly answered, perhaps because the experimenter lacks some life impacting domains (i.e. medicals, Auto, Financial, construction, etc.) expertise according to Korhonen (2022). but it was shown that this stricter tweaking measures spoken of, coupled with the allowed percentages of up and down samplings strategy (along with other parameters) with domain knowledge could be a further study (as well as the granular minority points noted by Krawczyk (2016) in his concluding recommendations) to get a better handling algorithm for treating highly imbalanced dataset at the data level effectively.

REFERENCES

Akbani, Rehan; Kwek, Stephen & Japkowicz, Nathalie. 2004, Applying support vector machines to imbalanced datasets, In: *European conference on machine learning*, Springer, pp. 39–50.

Barandela, Ricardo; Sánchez, José Salvador; García, Vicente & Ferri, Francesc J. 2003, Learning from imbalanced sets through resampling and weighting, In: *Iberian Conference on Pattern Recognition and Image Analysis*, Springer, pp. 80–88.

Barandela, Ricardo; Valdovinos, Rosa; Sánchez, Josep & Ferri, Francesc. 2004, The Imbalanced Training Sample Problem: Under or over Sampling?, p. 806.

Basso, Franco; Pezoa, Raúl; Varas, Mauricio & Villalobos, Matías. 2021, A deep learning approach for real-time crash prediction using vehicle-by-vehicle data, *Accident Analysis Prevention*, vol. 162, , p. 106409.

Brownlee, Jason. 2020, *Imbalanced classification with python: Better metrics, balance skewed classes, cost-sensitive learning*, Machine Learning Mastery.

Cao, Peng; Zhao, Dazhe & Zaiane, Osmar. 2013, An Optimized Cost-Sensitive SVM for Imbalanced Data Learning, In: Jian Pei; Vincent S. Tseng; Longbing Cao; Hiroshi Motoda & Guandong Xu, eds., *Advances in Knowledge Discovery and Data Mining*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 280–292.

Dou, Jun; Song, Yan; Wei, Guoliang & Zhang, Yameng. 2022, Fuzzy information decomposition incorporated and weighted Relief-F feature selection: When imbalanced data meet incompletion, *Information Sciences*, vol. 584, , pp. 417–432. Available: https://www.sciencedirect.com/science/article/pii/S0020025521010793.

Feng, Fang; Li, Kuan-Ching; Shen, Jun; Zhou, Qingguo & Yang, Xuhui. 2020, Using cost-sensitive learning and feature selection algorithms to improve the performance of imbalanced classification, *IEEE Access*, vol. 8, , pp. 69979–69996.

González, Sergio; García, Salvador; Li, Sheng-Tun & Herrera, Francisco. 2019, Chain based sampling for monotonic imbalanced classification, *Information Sciences*, vol. 474, , pp. 187–204.

Gu, Xiaowei; Angelov, Plamen P & Soares, Eduardo A. 2020, A self-adaptive synthetic over-sampling technique for imbalanced classification, *International Journal of Intelligent Systems*, vol. 35, no. 6, pp. 923–943.

Huang, Chen; Li, Yining; Loy, Chen Change & Tang, Xiaoou. 2016, Learning deep representation for imbalanced classification, In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5375–5384.

Kim, Jaehyung; Jeong, Jongheon & Shin, Jinwoo. 2020, M2m: Imbalanced classification via major-to-minor translation, In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13896–13905.

Korhonen, Satu. 2022, Building Trustworthy AI. Main questions, possible solutions and a case study as example.

Krawczyk, Bartosz. 2016, Learning from imbalanced data: open challenges and future directions, *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232.

Kubat, Miroslav; Matwin, Stan et al.. 1997, Addressing the curse of imbalanced training sets: one-sided selection, In: *Icml*, vol. 97, Citeseer, p. 179.

Lin, Enlu; Chen, Qiong & Qi, Xiaoming. 2020, Deep reinforcement learning for imbalanced classification, *Applied Intelligence*, vol. 50, no. 8, pp. 2488–2502.

Liu, Alexander Y. 2004, The Effect of Oversampling and Undersampling on Classifying Imbalanced Text Datasets.

Lu, Huijuan; Xu, Yige; Ye, Minchao; Yan, Ke; Gao, Zhigang & Jin, Qun. 2019, Learning misclassification costs for imbalanced classification on gene expression data, *BMC bioinformatics*, vol. 20, no. 25, pp. 1–10.

Luo, Menghua; ke, Wang; Cai, Zhiping; Liu, Anfeng; Li, Yangyang & Cheang, Chak. 2019, Using Imbalanced Triangle Synthetic Data for Machine Learning Anomaly Detection, *Computers, Materials Continua*, vol. 58, , pp. 15–26.

Maheshwari, Satyam; Jain, R.C. & Jadon, R.S. 2018, An Insight into Rare Class Problem: Analysis and Potential Solutions, *Journal of Computer Science*, vol. 14, no. 6, pp. 777–792. Available: https://thescipub.com/abstract/jcssp.2018.777.792.

Makki, Sara; Assaghir, Zainab; Taher, Yehia; Haque, Rafiqul; Hacid, Mohand-Said & Zeineddine, Hassan. 2019, An experimental study with imbalanced classification approaches for credit card fraud detection, *IEEE Access*, vol. 7, , pp. 93010–93022.

Mena, Luis & Gonzalez, Jesus. 2006, Machine Learning for Imbalanced Datasets: Application in Medical Diagnostic., vol. 2006, pp. 574–579.

Nnamoko, Nonso & Korkontzelos, Ioannis. 2020, Efficient treatment of outliers and class imbalance for diabetes prediction, *Artificial Intelligence in Medicine*, vol. 104, , p. 101815.

Rahman, Mostafizur & Davis, Darryl N. 2013, Addressing the Class Imbalance Problem in Medical Datasets, *International Journal of Machine Learning and Computing*, vol. 3, , p. 224.

Ramyachitra, D. & Manikandan, Parasuraman. 2014, IMBALANCED DATASET CLASSIFICATION AND SOLUTIONS : A REVIEW.

Ruf, Boris & Detyniecki, Marcin. 2021, Towards the right kind of fairness in AI, *arXiv preprint arXiv:2102.08453*.

Shah, Sejal. 2020, *Automatic Ticket Assignment using Machine Learning and Deep Learning Techniques*, Ph.D. thesis, Dublin, National College of Ireland.

Song, Yongming & Peng, Yi. 2019, A MCDM-based evaluation approach for imbalanced classification methods in financial risk prediction, *IEEE Access*, vol. 7, , pp. 84897–84906.

Sun, Chengfa; Cui, Hui; Zhou, Weidong; Nie, Weiwei; Wang, Xiuying & Yuan, Qi. 2019, Epileptic seizure detection with EEG textural features and imbalanced classification based on EasyEnsemble learning, *International journal of neural systems*, vol. 29, no. 10, p. 1950021.

Tang, Yuchun; Zhang, Yan-Qing; Chawla, Nitesh V. & Krasser, Sven. 2009, SVMs Modeling for Highly Imbalanced Classification, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281–288.

Turlapati, Venkata Pavan Kumar & Prusty, Manas Ranjan. 2020, Outlier-SMOTE: A refined oversampling technique for improved detection of COVID-19, *Intelligence-Based Medicine*, vol. 3-4, , p. 100023. Available: https://www.sciencedirect.com/science/article/pii/S2666521220300235.

Van Hulse, Jason; Khoshgoftaar, Taghi & Napolitano, Amri. 2007, Experimental Perspectives on Learning from Imbalanced Data, vol. 227, pp. 935–942.

Vanhoeyveld, Jellis & Martens, David. 2018, Imbalanced classification in sparse and large behaviour datasets, *Data Mining and Knowledge Discovery*, vol. 32, no. 1, pp. 25–82.

Vuttipittayamongkol, Pattaramon & Elyan, Eyad. 2020, Overlap-based undersampling method for classification of imbalanced medical datasets, In: *IFIP International Conference on Artificial Intelligence Applications and Innovations*, Springer, pp. 358–369.

Wang, Chen; Deng, Chengyuan & Wang, Suzhen. 2020, Imbalance-XGBoost: Leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost, *Pattern Recognition Letters*, vol. 136, , pp. 190–197.

Xia, Shuyin; Zheng, Shaoyuan; Wang, Guoyin; Gao, Xinbo & Wang, Binggui. 2021, Granular ball sampling for noisy label classification or imbalanced classification, *IEEE Transactions on Neural Networks and Learning Systems*.

Yan, Yuanting; Liu, Ruiqing; Ding, Zihan; Du, Xiuquan; Chen, Jie & Zhang, Yanping. 2019, A parameter-free cleaning method for SMOTE in imbalanced classification, *IEEE Access*, vol. 7, , pp. 23537–23548.

Zhang, Chong; Tan, Kay Chen; Li, Haizhou & Hong, Geok Soon. 2018, A cost-sensitive deep belief network for imbalanced classification, *IEEE transactions on neural networks and learning systems*, vol. 30, no. 1, pp. 109–122.

Zhang, Jue & Chen, Li. 2019, Clustering-based undersampling with random over sampling examples and support vector machine for imbalanced classification of breast cancer diagnosis, *Computer Assisted Surgery*, vol. 24, no. sup2, pp. 62–72.

Zhu, Honghao; Liu, Guanjun; Zhou, Mengchu; Xie, Yu; Abusorrah, Abdullah & Kang, Qi. 2020, Optimizing Weighted Extreme Learning Machines for imbalanced classification and application to credit card fraud detection, *Neurocomputing*, vol. 407, , pp. 50–62. Available: https://www.sciencedirect.com/science/article/pii/S0925231220306639.

Zolanvari, Maede; Teixeira, Marcio A & Jain, Raj. 2018, Effect of imbalanced datasets on security of industrial IoT using machine learning, In: *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, IEEE, pp. 112–117.

APPENDIX A

African Recessed Dataset:

1. For Default parameters jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/africaRecession.ipynb`
or `click`
2. For Adjusted parameters 1 jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/africaRecessionAdjust56.ipynb`
or `click`
3. For Adjusted parameters 2 jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/africaRecessionAdjust15.ipynb`
or `click`

APPENDIX B

Cerebral Stroke Dataset:

1. For Default parameters jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/cerebralStroke.ipynb`
or `click`
2. For Adjusted parameters 1 jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/cerebralStrokeAdjust56.ipynb`
or `click`
3. For Adjusted parameters 2 jupyter-notebook code, copy and paste in your browser:
`https://github.com/packetech/Arcada-BDA/blob/main/cerebralStrokeAdjust15.ipynb`
or `click`