Handling Imbalanced Classification Problem for the Prediction of Stroke in Patients

Rahul Shah, Fan Wang
Department of Mechanical and Industrial Engineering,
University of Illinois at Chicago, Chicago, IL, 60607

***Abstract* — In healthcare data analytics, medical events are often not balanced in the class labels of their response variable. Detection of minority events in imbalanced class distribution is often challenging. Most classification models tend to perform poorly on such data because they do not consider relative distribution of each class and are designed to optimize the overall classification accuracy. The Python programming language is one of the many tools available for data analytics. In this study, we develop a framework for learning healthcare stroke data set with imbalanced class distribution via incorporating different classification algorithms and resampling strategies in Python. The classifiers logistic regression, kNN and SVM were applied to the original, under-sampled, over-sampled, and combination of over and under-sampled data sets. The evaluation results showed that kNN combined with the combination of over and under-sampling method using SMOTE-Tomek exhibited the best stroke detection results. The AUC (Testing: 0.9943), accuracy (Testing: 97.19%), precision (Testing: 94.78%) and recall (Testing: 99.88%) were achieved with the best performing classifier and resampling strategy.**

***Keywords* — Machine Learning, Binary Classification, Class Imbalance, Under-sampling, Over-sampling, SMOTE, AUC-ROC**

I. INTRODUCTION

In this paper, a framework for learning a healthcare data set with imbalanced class distribution via incorporating different classification algorithms and resampling strategies are developed. Today there are plentiful collected data in cases of various diseases in medical sciences. Physicians can assess new findings about diseases and procedures in dealing with them by probing these data. This study was performed to predict stroke incidence in patients. According to the World Health Organization, of the 56.9 million deaths worldwide in 2016, ischemic heart disease and stroke accounted for a combined 15.2 million deaths in 2016 [1]. These diseases have remained the leading causes of death globally in the last 15 years. The data from U.S. National Stroke Association states that each year nearly 800,000 people experience a new or recurrent stroke [2]. At the same time, stroke is the fifth leading cause of death in the U.S., and every 4 minutes someone dies from stroke. The risk factors that cause stroke can be divided into two categories: the first is the disease factor, and the second is the behavioral factors, including diet, smoking, drinking, exercise, etc. Actually, up to 80 percent of strokes can be prevented.

For an effective healthcare data analytics, highly skewed class distribution of the response variable poses a major challenge, which is referred to as the imbalanced classification problem [3]. An imbalanced classification problem occurs when the classes in a dataset have a highly unequal number of samples. For instance, in binary classification, the imbalanced classification problem is present when one class has significantly fewer observations than the other class. The former is usually called a minority class, and the latter, a majority class. In this study, we develop a method for detecting stroke incidence events in a data set where this data challenge is present. We use the data set from Kaggle, the world's largest community of data scientists and machine learning [4]. This is a classification problem where the primary goal is to train the model and predict whether the patient will have a stroke through machine learning. In other words, the goal is to predict the stroke probability using the given information of patients.

TABLE I
DATA DICTIONARY

Attributes	Description
ID	Unique identifier for patient.
Gender	Gender of patient: Male/Female/Other
Age	Age of patient
Hypertension	0: no hypertension, 1: suffering hypertension
Heart disease	0: no heart disease, 1: suffering from heart disease
Ever married	Yes/No
Work type	Type of occupation
Residence type	Area type of residence: Urban/Rural
Average glucose level	Average Glucose level (measured after meal)
BMI	Body mass index (BMI) is a measure of body fat based on height and weight that applies to adult men and women.
Smoking status	Patient's smoking status
Stroke (Target)	0: no stroke, 1: suffered stroke

The data dictionary, as shown in TABLE I, consists of 12 attributes describing the patient's basic information such as blood sugar levels, smoking status, marriage etc. on the patient's impact on stroke. The first attribute is patient ID, which is not considered in the feature set, as it doesn't have any value in model building. Thus, the feature set consists of 10 predictor variables, and one target variable- Stroke. The total number of patient records in the data set is over 43,000.

There were various data preprocessing steps performed, which will be discussed in greater details in subsequent sections, before applying classification methods to predict stroke. Sampling strategies have been used to overcome the class imbalance problem. As shown in Fig. 1, the majority class (no stroke) occurs in about 98.2% of the records in the data set, while the minority class occurrence is only 1.8%. The classifiers logistic regression, kNN and SVM were applied to the original, under-sampled, over-sampled, and combination of

over and under-sampled data sets. The evaluation metric used to assess the performance of each classification models under each of the sampling techniques is 'Area Under the Receiver Operating Characteristics (AUC-ROC) Curve'.

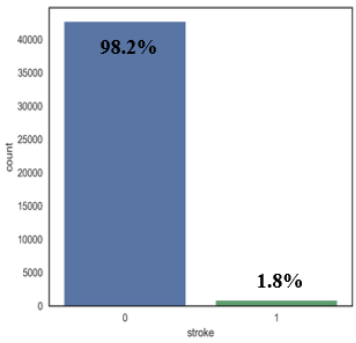

Fig. 1. Class imbalance in target variable- Stroke

The rest of this paper is organized as follows. Section 2 presents the data preprocessing procedure. In Section 3, an overview of different resampling strategies is presented to handle imbalanced data. In Section 4, various classification models are illustrated. Section 5 explains in detail the results obtained. Finally, the conclusions are drawn, and future scope is discussed in Section 6.

II. DATA PREPROCESSING

Data preprocessing aspect of machine learning is a technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors [5]. As the number of patient records is over 43,000, it is safe to assume that the data would have to be processed before passing it through different classification models. So, it is vital that operations like identification and replacement of missing values, outlier detection, feature encoding, feature selection, and normalization be done.

A. Missing Value Imputation and Outlier Detection

The first process was to identify if there were any missing values in the data set. There were two features, BMI and smoking status, with missing values. BMI had a total of 1462 missing values. We used the average BMI value grouped by age group (defined in the bins of 10 corresponding to the age of the patient) and gender to impute the missing values in BMI. The heatmap in Fig. 2 shows these average BMI values. Next, in feature- smoking status, there were about 13,292 missing values. According to US law, young people under the age of 18 are prohibited from buying tobacco products. So, for patients below the age of 18, we assigned a category 'never smoked' as an imputed value for smoking status. For all other patients we introduced a new category 'no information' to be used as an imputed value for smoking status.

Next, we performed an outlier detection analysis for continuous features using boxplots. An outlier is usually an observation which typically lies farthest from the mean. According to a Statistical theory, if any observation is 3 * IQR (Inter-Quartile Range) from the mean then it's called an Outlier [5]. From this analysis, we observed that we didn't have any significant outliers in the data set.

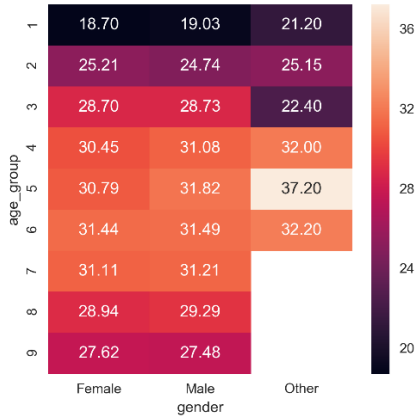

Fig. 2. Heatmap of average BMI grouped by age group and gender

B. Encoding of Categorical Features

Most of the machine learning algorithms; such as logistic regression, kNN, SVM; cannot work directly with categorical data. The encoding allows algorithms which expect continuous features to use categorical features [6]. There are two widely used methods of converting a categorical feature into numerical feature, which are Label Encoding and One-Hot Encoding. We performed Label Encoding for categorical features with two categories. These features are ever married and residence type. Here, the encoding scheme used was '1' and '0' to represent categories 'Yes' and 'No', respectively. For features with more than two categories, One-Hot Encoding scheme was utilized. These features included gender, work type and smoking status. One-Hot Encoding transforms each categorical feature with *n* possible categories into *n* binary features, with only one active [6]. This encoding is used when there is no ordinal relationship between elements. After the categorical features were encoded, the feature set increased from 10 features to 19 features. Next, feature selection procedure was implemented to achieve a reduced subset of features.

C. Feature Selection

Feature selection using Random Forest Regressor (RFE) was implemented in Python, on the training set, which attempts to weight the importance of each feature. Random forests allow to compute a heuristic for determining how "important" a feature is in predicting a target. This heuristic measures the change in prediction accuracy of a given feature and permutes it across the data points in the training set. The more the accuracy drops when the feature is permuted, the more "important" is the feature. The bar plot in Fig. 3 shows the relative importance of the named features. From this plot, features such as 'work type children', 'work type never worked' and 'gender other' displayed negligible importance relative to the other features, and accordingly were removed. As a result, a reduced subset of 16 features was achieved.

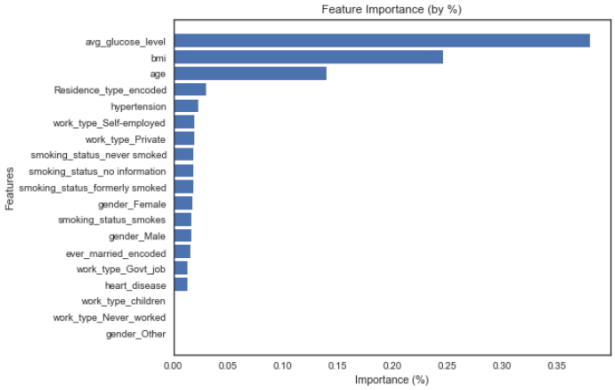

Fig. 3. Feature Importance (by %)

D. Feature Scaling

The last step for the data preprocessing is feature scaling. Normalization using *z*-score standardization was implemented to scale the features. This was performed by using StandadScalar() module in Python, where the data is rescaled such that each feature column has the property of a standard normal distribution with a mean of zero and a standard deviation of one. This is necessary to avoid scale effect of features on the classification models (such as logistic regression, kNN and SVM). One more advantage of using this approach is that, we end up with smaller standard deviations, which suppresses the effects of outliers, if any present.

III. RESAMPLING STRATEGIES

In this section we analyze the importance of appropriate sampling technique in a classification model. In the dataset as mentioned earlier, there is a very distinct imbalance in the classes. Modelling the classifier by training on the mere dataset alone may result in incorrect classification of the under sampled class. In cases such as ours where an incorrect prediction is undesired since it can have negative repercussions on various medical treatment procedures, it is important that the under sampled class be identified appropriately. The model was initially trained with the raw data without any class balancing technique. Though this method helped predict the "NO" class well, the recall and precision in predicting the "YES" class i.e. the patients who suffered a stroke, was considerably low. Hence, class balancing techniques were considered and the following sampling techniques were applied to the model by using imblearn (imbalanced-learn) library in Python.

A. Random Under-sampling

The simplest implementation of under-sampling is through random elimination of majority class instances through a non-heuristic method that aims to balance class distribution [7]. However, this method has a major drawback that it can discard potentially useful information that could be important for the training process. For our case, this method randomly selects the patients without a stroke to be matched with the number of patients who suffered a stroke. The implementation of this strategy is shown in Fig. 4.

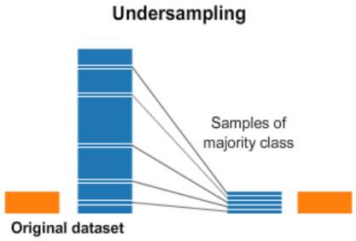

Fig. 4. Implementation of Under-sampling Method

B. Random Over-Sampling

The most straightforward implementation of over-sampling is to duplicate random records from the minority class through a non-heuristic method that aims to balance class distribution [7]. It is possible that this method may cause some degree of overfitting. For our case, this method randomly selects the patients who suffered a stroke to match with the number of patients without a stroke. The implementation of this strategy is shown in Fig. 5.

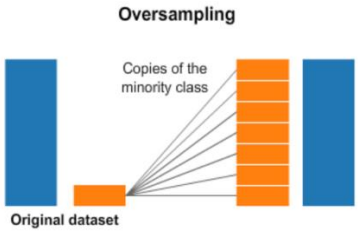

Fig. 5. Implementation of Over-sampling Method

C. Over-sampling using SMOTE

SMOTE (Synthetic Minority Oversampling technique) consists of synthesizing elements for the minority class, based on those that already exist [8]. It works by randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbors. Our implementation currently uses five nearest neighbors. The execution of this strategy is shown in Fig. 6.

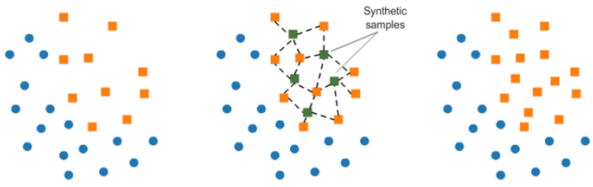

Fig. 6. Implementation of Over-sampling Method using SMOTE

D. Combination of over and under-sampling using SMOTE-Tomek

We also analyze the combination of over and under-sampling method, where over-sampling is followed by under-sampling strategy. The over-sampling is performed by using SMOTE as shown in Fig. 6, and under-sampling is performed using Tomek links procedure as described in Fig. 7. Tomek links are pairs of very close instances, but of opposite classes. Removing the instances of the majority class of each pair increases the space

between the two classes, facilitating the classification process [9].

.

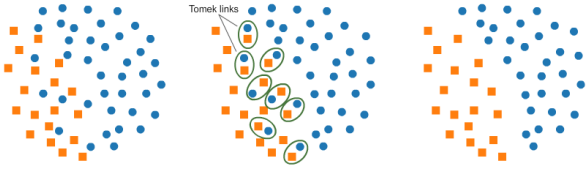

Fig. 7. Implementation of Under-sampling Method using Tomek links

IV. METHODOLOGY

We have used three classification models (logistic regression, kNN and SVM) to examine how they perform under each of the resampling strategies mentioned in Section 3. We have applied Stratified K-Fold Cross Validation and used AUC-ROC Curve as the evaluation metric for the performance of each classifier. The analysis is performed using sklearn library in Python.

A. Classification Models

Logistic Regression is the most popular classifier in healthcare research. This is due to the fact that logistic regression delivers the important information for doctors and medical practitioners such as the odds ratio, relative risk and predictions of dichotomous outcomes [13]. TABLE II shows the hyperparameters and their settings that were considered for tuning using the grid search to get the best parameters for Logistic Regression under each sampling method.

TABLE II
HYPERPARAMETERS FOR LOGISTIC REGRESSION

Hyper-parameter	Description	Settings
C	Inverse of regularization strength	[0.001, 0.003, 0.005, 0.01, 0.05, 0.1, 0.3, 0.5, 1, 2, 3, 4, 5, 10, 20]
Class weight	Weight associated with the classes	{'none', 'balanced'}

The second classifier implemented was k-Nearest Neighbors (kNN). The strategy of kNN is to classify the observations based on the surrounding neighbors, where *k* in kNN refers to the number of neighbors to be considered in the sampling process [13]. The distance used to calculate the closest neighbors are based on the Euclidean distance formula. TABLE III shows the hyperparameters and their settings that were considered for tuning using the grid search to get the best parameters for kNN under each sampling method.

TABLE III
HYPERPARAMETERS FOR KNN

Hyper-parameter	Description	Settings
k	Number of neighbors	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Weight	Weight function used in prediction	{'uniform', 'distance'}

Lastly, the third classifier employed was Support Vector Machine (SVM). The architecture of SVMs uses the general category of kernel methods [13]. These kernels depend on the distribution of the data mapped through some dot-products, whereby these dot-products are then transformed into kernel functions which computes a dot-product in high dimensional feature space. As an efficient classifier, SVM is able to generate non-linear decision boundaries using methods designed for linear classifiers. In addition, the kernel functions used in SVM allows the researchers to apply the classifier to any data sets which have no obvious representation in terms of patterns. TABLE IV shows the hyperparameters and their settings that were considered for tuning using the grid search to get the best parameters for SVM under each sampling method.

TABLE IV
HYPERPARAMETERS FOR SVM

Hyper-parameter	Description	Settings
C	Penalty parameter for error term	{'rbf', 'linear'}
Kernel	Kernel type	[0.001, 0.0001]
Gamma	Kernel coefficient	[1, 10, 100, 1000]

B. Cross Validation

All the previously mentioned resampling strategies and classification models (along with the various hyperparameter settings) were implemented using a Stratified K-Fold Cross Validation method. Stratified K-Fold allows the model to preserve the ratio of classes in each fold, thereby achieving model robustness [10]. For our analysis, K was set to 10 which means that every time the model runs, it uses 90% of the data as the training set and 10% as the testing set. This process was repeated 10 times to give a chance to all data points to exist in both training and testing sets at least once. Using K-Fold cross validation ensures the integrity and unbiasedness of the model towards a specific segment of the data set.

C. Performance Evaluation Metric: AUC-ROC Curve

We used Area Under the Receiver Operating Characteristic (AUC-ROC) Curve as the evaluation metric to assess the performance of each classification model under each resampling strategy. ROC is a graphical way to show how good the performance of a classifier is [11]. Essentially, it is a probability curve plotted with True Positive Rate (TPR) against False Positive Rate (FPR). The AUC represents the degree or measure of separability, i.e., capability of a model to distinguish between the classes of the response variable [12]. It can take any value between 0 and 1. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. For our case, higher the AUC, better the model is at distinguishing between patients with stroke and no stroke. When AUC is approximately 0.5, model has no discrimination capacity to distinguish between positive class and negative class, i.e. it makes random predictions.

V. RESULTS AND DISCUSSION

This section discusses and compares how different classification models performed under various resampling strategies. The results are also compared to a case when no sampling was considered, and the original imbalance ratio of the class was kept while implementing different classifiers. Since, a 10-fold cross validation was used, the ROC curves

displayed in the figures below is the mean of the 10 ROC curves (one curve from each K-Fold). Also, the AUC values displayed is the mean of the 10 AUC values coming from each fold. Also, the results discussed below are of the best hyperparameter settings achieved by performing grid search for various models under different resampling strategies.

Fig. 8 shows how Logistic Regression model performs under different resampling strategies. The performance of this model, as seen, is not largely affected by the resampling strategy used. Overall, over-sampling followed by under-sampling using SMOTE-Tomek provided a marginally better AUC of 0.8587 for Logistic Regression. This suggests that Logistic Regression has about 86% chance to be able to distinguish between patients with stroke and no stroke.

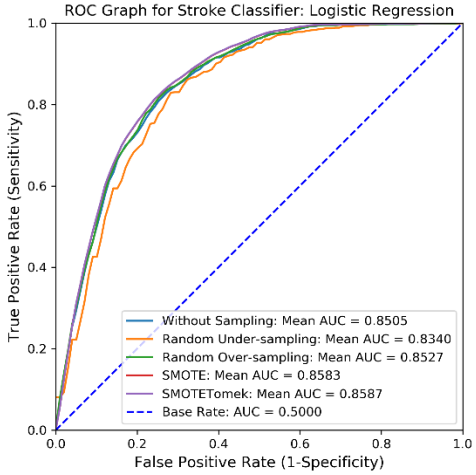

Fig. 8. Comparison of AUC-ROC for Logistic Regression Under Various Resampling Strategies

Fig. 9 shows how kNN model performs under different resampling strategies. We see that the performance is the worst when no sampling is employed. Further, kNN performs the best with both the over-sampling strategies and also with the combination of over and under-sampling strategy. The SMOTE-Tomek performs marginally better with an AUC score of 0.9943. This suggests that kNN model is almost always able to distinguish between patients with stroke and no stroke.

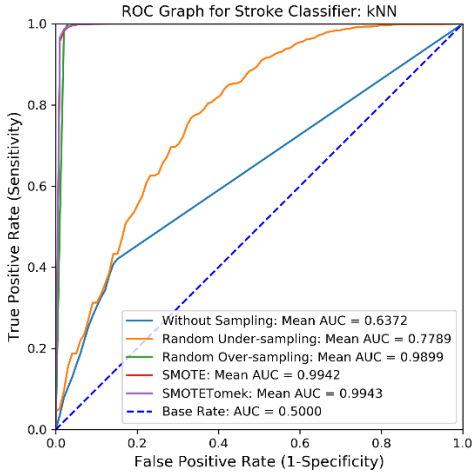

Fig. 9. Comparison of AUC-ROC for kNN Under Various Resampling Strategies

Fig. 10 shows how SVM model performs under different resampling strategies. Again, the performance is the lowest when no sampling is employed. The best performance of the SVM is achieved with SMOTE-Tomek resampling strategy which attains an AUC score of 0.9582. This suggests that SVM has about 96% chance to be able to distinguish between patients with stroke and no stroke.

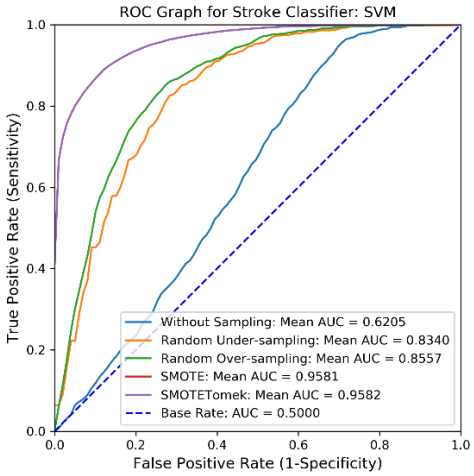

Fig. 10. Comparison of AUC-ROC for SVM Under Various Resampling Strategies

In Fig. 11, the results are summarized for the best performing classifier under each resampling strategy. We observe that, Logistic Regression performed best with the original imbalanced data set. Further, SVM had the best performance with the random under-sampled data set. For the remaining three resampling strategies, kNN had the best performance.

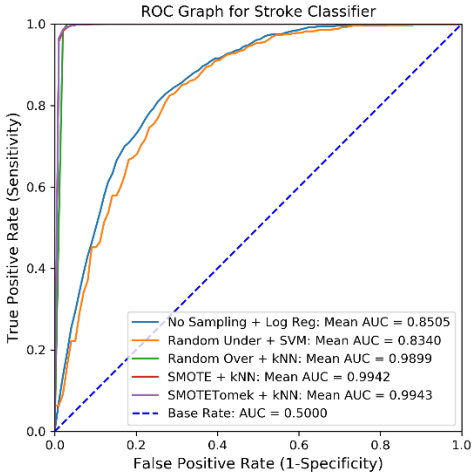

Fig. 11. AUC-ROC of the Best Performing Model Under Various Resampling Strategies

TABLE V shows the confusion matrix for the best performing model (Logistic Regression) when no resampling strategy was employed. This confusion matrix is the average of the 10 confusion matrices generated from each K-Fold. It can be observed from the table that the accuracy of the model is 73.20%, it has a recall value of 83.33% for the positive class and a precision of just 5.35% which is significantly low. In the medical field it is very important to correctly predict the positive instances as that is what would decide treatment

procedures for a medical condition and subsequent lifestyle changes in a patient.

TABLE V
CONFUSION MATRIX: LOGISTIC REGRESSION WITHOUT SAMPLING IMPLEMENTED

		Predicted		
		NO	YES	Recall
Actual	NO	3112	1150	73.02%
	YES	13	65	83.33%
	Precision	99.58%	5.35%	
	Accuracy = 73.20%			

Further, Table VI shows the average confusion matrix for the best classifier (kNN) and the best resampling strategy (SMOTE-Tomek). It can be observed from the table that the accuracy of the model is 97.19%, it has a recall value of 99.88% for the positive class and a precision of 94.78% which is significantly better than the previous confusion matrix. This shows how important it is to handle an imbalanced data set like the one used in this study in order to achieve better predictions for the minority class.

TABLE VI
CONFUSION MATRIX: KNN WITH SMOTE-TOMEK SAMPLING IMPLEMENTED

		Predicted		
		NO	YES	Recall
Actual	NO	4024	234	94.50%
	YES	5	4252	99.88%
	Precision	99.87%	94.78%	
	Accuracy = 97.19%			

The performance of the two confusion matrices can also be compared by calculated their total misclassification cost. If we assume a unit cost of \$500/misclassification for misclassifying a YES as a NO and a unit cost of \$100/misclassification for misclassifying a NO as a YES, then the weighted misclassification costs of the two confusion matrices can be calculated as shown in equation (1) and (2), respectively. The cost in misclassifying a minority class is higher than that of the majority class, because in medical datasets high risk patients tend to be the minority class. It can be clearly seen from the cost analysis that cost of misclassification is nine times less in the second confusion matrix, thereby suggesting that resampling strategy using SMOTE-Tomek together with kNN performs significantly better compared to when no sampling is performed.

$$(\text{Misclassification Cost})^{I} = \frac{(1150\times100+13\times500)}{4340} = \$28 \quad (1)$$

$$(\text{Misclassification Cost})^{II} = \frac{(234\times100+5\times500)}{8515} = \$3 \quad (2)$$

VI. CONCLUSION

Class imbalance is a common problem with most medical datasets. Most existing classification methods tend not to perform well on minority class examples when the dataset is extremely imbalanced. Sampling strategies have been used to overcome the class imbalance problem by either over-sampling or under-sampling. We examined random under-sampling, random over-sampling, popular over-sampling strategy SMOTE, and combination of over and under-sampling using SMOTE-Tomek. Classification models such as Logistic Regression, kNN and SVM were examined. We used AUC as a measure of performance to evaluate different classifiers. After analyzing various combinations of sampling methods and classification models for the given imbalanced data set, we can state that the two best models were SMOTE with kNN classification (0.9942) and SMOTE-Tomek with kNN classification (0.9943). Overall, the best results were obtained by the kNN classifier with SMOTE-Tomek resampling strategy. The ROC graph shows the models with the best AUC values. Accuracy as high as 97.19% was achieved compared to the 73.2% we see in Logistic Regression method without sampling.

Considering most of the medical datasets today are a part of the imbalanced class problem, the proposed framework can be incorporated with different classification algorithms and resampling strategies to achieve a desired result which helps us in prediction of the future outcomes more precisely. This application can be used in multiple industries and various cases as per one's requirement. Such timely predictions can help one in making the necessary changes to alter the future outcome and lead a better lifestyle.

REFERENCES

[1] World Health Organization, & World Health Organization. (2015). The top 10 causes of death. 2012. *Availble at: http://www. who. int/mediacentre/factsheets/fs310/en/*.

[2] Ischemic Stroke, National Stroke Association, March 2016, [online] Available: http://www.stroke.org/understand-stroke/what-stroke/ischemic-stroke.

[3] Zhao, Y., Wong, Z. S. Y., & Tsui, K. L. (2018). A Framework of Rebalancing Imbalanced Healthcare Data for Rare Events' Classification: A Case of Look-Alike Sound-Alike Mix-Up Incident Detection. *Journal of Healthcare Engineering*, *2018*.

[4] https://www.kaggle.com/asaumya/healthcare-dataset-stroke-data/activity

[5] Birant, D. (2011). Data mining using RFM analysis. In *Knowledge-oriented applications in data mining*. InTech.

[6] Zhang, W., Du, T., & Wang, J. (2016, March). Deep learning over multi-field categorical data. In *European conference on information retrieval* (pp. 45-57). Springer, Cham.

[7] Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, *30*(1), 25-36.

[8] Rahman, M. M., & Davis, D. N. (2013). Addressing the class imbalance problem in medical datasets. *International Journal of Machine Learning and Computing*, *3*(2), 224.

[9] Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, *6*(1), 20-29.

[10] Mullin, M. D., & Sukthankar, R. (2000, June). Complete Cross-Validation for Nearest Neighbor Classifiers. In *ICML* (pp. 639-646).
[11] Acharya, A. (2017). Comparative Study of Machine Learning Algorithms for Heart Disease Prediction.
[12] Fan, J., Upadhye, S., & Worster, A. (2006). Understanding receiver operating characteristic (ROC) curves. *Canadian Journal of Emergency Medicine*, *8*(1), 19-20.
[13] Rahman, H. A. A., Wah, Y. B., He, H., & Bulgiba, A. (2015, September). Comparisons of ADABOOST, KNN, SVM and logistic regression in classification of imbalanced dataset. In *International Conference on Soft Computing in Data Science*(pp. 54-64). Springer, Singapore.