Survey of Several Sampling methods of imbalanced dataset Classification

Karthik Jallepalli [#1] **, Yashwanth Manglarapu**[#2]
[#1,#2] B.Tech Student, Department of Computer Science and Information Technology (CSIT),
CVR College of Engineering, Vastunagar, Mangalpalli (v), Ibrahimpatnam (M),
R.R.Dist-501510, Telangana state.

ABSTRACT

On classification datasets with an unbalanced distribution of classes, machine learning approaches frequently perform poorly or with falsely optimistic results. This is because a lot of machine learning algorithms are made to work on classification data that has an equal number of observations for each class. When this isn't the case, algorithms can discover that a small number of samples aren't crucial and can be disregarded in order to perform well.In order to balance or improve the balance of the class distribution, a training dataset can be transformed using a variety of approaches provided by data sampling. Once the dataset has been balanced, normal machine learning methods can be trained directly on it without any changes. This makes it possible to use a data preparation strategy to handle the problem of imbalanced classification, even with substantially unbalanced class distributions. There are many various types of data sampling techniques available, and there is no one ideal technique to apply to all classification issues and models. To find what works best for your proposed work, careful experimentation is necessary, just like with selecting a predictive model.In this proposed work we try to discuss about the problems of imbalanced class classification and what are the several data sampling methods available in the real world dataset classification.Also we disucss the pros and cons of oversampling and undersampling techniques.

Index Terms:

Machine Learning, Unbalanced Distribution, Machine Learning Algorithms, Data Sampling, Undersampling, Oversampling, Class Distribution.

1. INTRODUCTION

If the categorization classes are not roughly equal and one class has significantly more samples than the other classes, the data set is said to be imbalanced. Due to the increased influence of the majority class, classifiers may perform poorly on the minority classes in this case but perform well on the majority class. Thus, the conventional modeling. The class-imbalance techniques are proposed to increase the prediction performance of the modelling algorithms because algorithms do not always work well on the unbalanced data[1]. In this article, we examine performance metrics and strategies for handling unbalanced data. Then, using empirical trials, we assess the effects of class-imbalance methods on a variety of conventional modelling algorithms[2]-[5]. The issue of unbalanced data has received a lot of attention in both theoretical and empirical investigations. The class-imbalance strategies can be divided into data pre-processing methods and modelling algorithm-specific methods depending on the modelling stages used. The approaches used for data pre-processing before modelling typically involve under- or oversampling the training set. The approaches that are special to modelling algorithms are stand-alone algorithms that operate directly on imbalanced training data. 15 data sets from the UCI/Keel databases are tested, with each data set containing at least 500 samples and the range of the imbalance ratio being broad in order to assess the effect of class-imbalance approaches on the model performance. Eight modelling algorithms are subjected to four class-imbalance experiments, and the performance is evaluated using the F-score and AUC. The findings indicate that the choice of modelling algorithms has a greater influence on performance, but class-imbalance approaches are more efficient with straightforward linear algorithms like Logistic Regression and Linear SVC[6].

Unbalanced data sets are thought to be a unique instance of a data science issue. The original data science problem has been extended or restricted to take the form that it does in these situations. Another extended supervised learning approach is this issue.This problem arises when one class' distribution is significantly lower than that of the other classes. Typically, we deal with the phrases "positive class" and "negative class" (minority class and majority class, respectively). Because the standard machine learning algorithms deal with unbalanced data without sensitivity to the unbalanced distribution of the classes, which leads to a bias towards the (negative class/majority class), this type of data creates a new problem in the field of data mining across all of the cosmos[1-4]. Because the balanced distributions of the class are required by the norm classification techniques, this problem is occasionally unimportant for some applications. However, it is a real problem for real-world applications like text classification [5], telecommunications, finance, oil spills detection using radar[2], e-mail foldering[6], the fraudulent calls detection[3], medical diagnosis[7], etc. because in this situation, the learning is more interested in the minority classes than the majority classes, which are needed to be correctly identified in these applications.

2. LITERATURE SURVEY

In this section we will mainly discuss about the background work that is carried out in order to prove the performance of our proposed Method. Literature survey is the most important step in software development process. In this we are going to discuss about some papers which are already published by several authors about the data sets sampling methods and what are the pros and cons regarding improper dataset cleaning while we classify any unbalanced datasets for real world problems.

MOTIVATION

A well known author Shrouk El-Amir, et.al, proposed a paper" Classification Imbalanced Data Sets: A Survey". In this paper the author said that uunbalanced data, a problem that frequently arises in real-world applications, can significantly reduce the classification performance of machine learning algorithms. Unbalanced data sets are classified in a number of different ways. We should use machine learning classifiers to artificially equalise the imbalanced data sets by oversampling and/or undersampling. Here he mainly concentrated in oversampling and undersampling techniques and if we want to achieve accurate results after training one should clearly follow these techniques, or else the result may not be accurate[8].

A well known author Lian Yu, et.al, proposed a paper" Survey of Imbalanced Data Methodologies". In this paper the author said that, in the current financial sector, unbalanced data sets are a common issue that has received extensive research. In this study, we examined and contrasted several widely used approaches for dealing with data imbalance. The under-sampling/over-sampling approaches were then used to a number of modelling algorithms on the UCI and Keel data sets. Analysis of the performance included comparison of grid search criteria, modelling techniques, and class-imbalance approaches[9]-[11].

A well known author Jason Brownlee, et.al, proposed a paper" Random Oversampling and under sampling for Imbalanced Classification". In this article the author discussed an extreme skew in the class distribution, such as a ratio of 1:100 or 1:1000 samples from the minority class to the majority class, is a sign that a dataset is unbalanced. Many machine learning algorithms may be affected by this bias in the training dataset, and some may completely overlook the minority class. This is problematic because forecasts are often more crucial for the minority class. Randomly resampling the training dataset is one way to deal with the issue of class imbalance. Undersampling, or removing examples from the majority class, and oversampling, or duplicating examples from the minority class, are the two main strategies for randomly resampling an unbalanced dataset[12].

A well known author Yunqian Ma, et.al, proposed a paper" Imbalanced Learning: Foundations, Algorithms, and Applications". In this paper the author said that Class imbalance impacts datasets with more than two classes that may include numerous minority classes or multiple majority classes, despite the fact that it is frequently discussed in terms of two-class categorization issues. The ineffectiveness of typical machine learning methods on imbalanced classification datasets is a major issue. Many machine learning algorithms use the class distribution in the training dataset to determine how likely it is that users of the model will see examples from each class when making predictions. Since the majority class is more important than the minority class, many machine learning algorithms, including decision trees, k-nearest neighbours, and neural networks, will learn this and focus more on it. These algorithms will also perform better on it[13].

A well known author Fernández, et.al, proposed a paper” Learning from Imbalanced Data Sets”. In this paper the author said that the most common remedy for an unbalanced classification issue is to alter the training dataset's makeup. We are sampling an existing data sample, therefore techniques intended to alter the class distribution in the training dataset are more commonly known as sampling methods or resampling methods. Sampling methods appear to be the most popular strategy in the community because they directly address uneven learning[14].

A well known author Max Kuhn, et.al, proposed a paper” Applied Predictive Modeling 1st Edition”. In this paper the author said that the training dataset is used by machine learning algorithms like the Naive Bayes Classifier to determine how likely it is to observe samples from each class. It is possible for these models to learn a less biased prior probability and instead concentrate on the particulars (or evidence) from each input variable to discriminate the classes by fitting these models on a sampled training dataset with an artificially more equal class distribution. Some models, including naive Bayes and discriminant analysis classifiers, use prior probabilities. These models normally obtain the priors' value from the training data, unless otherwise specified manually. An unbalanced class can be addressed by using more balanced priors or a balanced training set[15].

3. EXISTING SYSTEM & ITS LIMITATIONS

In the existing system there was no proper method to identify the imbalanced datasets accurately and hence all the existing systems try to upload the datasets for classification. The imbalanced datasets may sometimes lead to generate in accurate or in appropriate results compared with accurate and balanced datasets. In general if we want to classify health science related datasets related to disease prediction, and then we need to process the dataset keenly in order to avoid the mistakes during prediction. The existing system follows manual approach and hence the following are the limitations of the existing system.

1. More Time Delay in finding the unbalanced datasets.
2. There is a huge delay in finding the unbalanced datasets
3. All the existing approaches are in accurate while predicting the data.
4. There is no recommendation system in the existing system.
5. The primitive systems fail in classifying the data when the dataset is unbalanced.

4. PROPOSED SYSTEM & ITS ADVANTAGES

In the proposed system, we try to conduct a survey on several sampling methods on imbalanced datasets to identify the unbalanced datasets and how effectively we can classify the given dataset on particular task. In general the real world datasets are almost imbalanced and if the user fails to identify the imbalanced data properly the classification[16] may generate in accurate results. Hence in this proposed system we try to identify the best method for sampling the imbalanced datasets. The following are the advantages of the proposed system. They are as follows:

1) By using several datasets ,we can predict which sampling technique is best to classify/
2) The proposed work is used by machine learning algorithms for generating optimized results
3) The proposed work is greatly help in optimization.
4) In this proposed work, we try to classify the unbalanced data accurately.
5) By using proposed system we can accurately identify the best data sampling technique.

5. PROPOSED METHODOLOGY

In this proposed system we try to use five main steps to make an unbalanced datasets to get into balanced datasets by using some Undersampling and over sampling techniques. The procedure is explained clearly in this article. The proposed methodology is divided into 5 steps, as follows:

1. Random Resampling Inconsistent Datasets
2. Identify the Incomplete Library
3. Imbalanced datasets with Random Oversampling
4. Datasets with Random Undersampling and Imbalances
5. Mixing random under- and oversampling

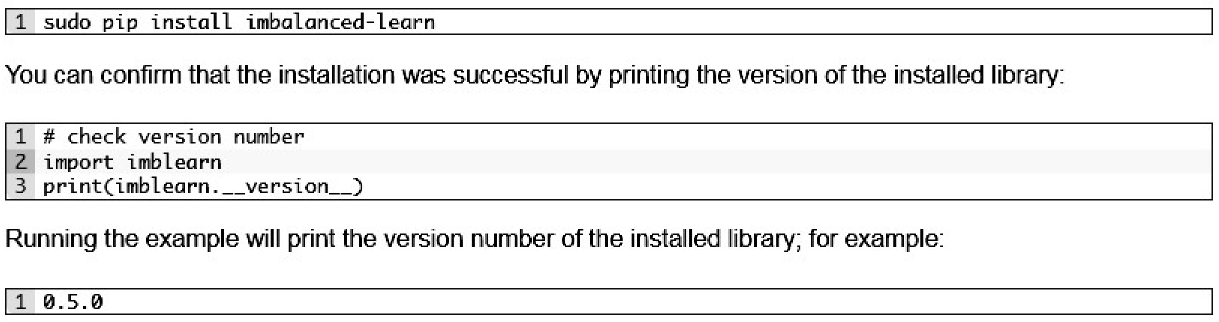
1) RANDOM RESAMPLING INCONSISTENT DATASETS

Resampling entails transforming the training dataset into a new version where the chosen samples have a different class distribution. This is a quick and reliable solution to imbalanced classification issues. The imbalance issue can be effectively solved by using re-sampling techniques to get a more equal data distribution. **Oversampling** and **Undersampling** are the two basic strategies for random resampling for unbalanced categorization. Oversample at random by duplicating instances from the minority class. Undersampling at random: Remove examples from the majority class at random. Random oversampling includes adding replacement samples drawn at random from the minority class to the training dataset. In random undersampling, instances from the majority class are randomly chosen and removed from the training dataset.

All the above methods can be applied repeatedly until the training dataset exhibits the desired class distribution, such as an equal distribution of classes. They are known as "**naive resampling"** techniques since they make no assumptions about the data and don't employ any heuristics. For highly large and complex datasets, this makes them easy to implement and quick to execute. Both methods can be applied to problems involving binary classification of two classes as well as multi-class classification issues involving one or more majority or minority classes. Importantly, the training dataset is the only one to which the class distribution has been altered. The goal is to alter how well the models fit. The test or holdout dataset used to assess a system's performance does not undergo resampling.

2) IDENTIFY THE INCOMPLETE LIBRARY

Here we try to discuss incomplete library by using some implementation steps which is implemented in Python Language.



```
sudo pip install imbalanced-learn
```

You can confirm that the installation was successful by printing the version of the installed library:

```
# check version number
import imblearn
print(imblearn.__version__)
```

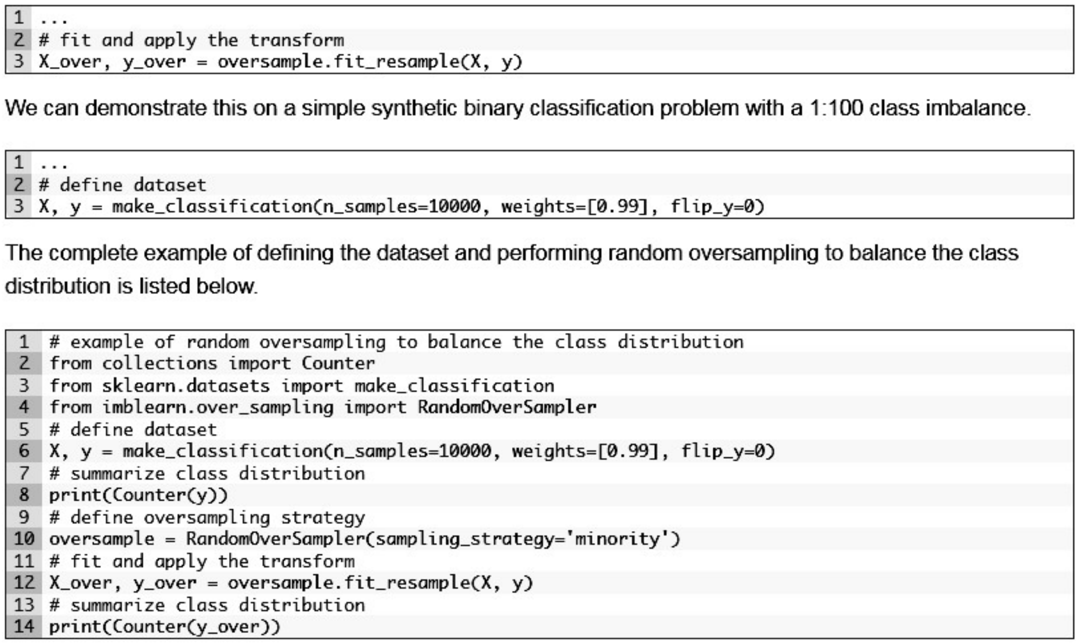
Running the example will print the version number of the installed library; for example:

```
0.5.0
```

In the above steps we can identify how the libraries are included for finding any incomplete library during data sampling.

3) IMBALANCED DATASETS WITH RANDOM OVERSAMPLING

The training dataset is oversampled at random by adding duplicate cases from the minority class. Examples are chosen at random with replacement from the training dataset. This implies that samples from the minority class can be picked from the initial training dataset, added to the new, "more balanced," training dataset, and then returned or "replaced" in the initial dataset, allowing them to be selected once more. This method can be useful for machine learning methods where the model fit can be influenced by several duplicate samples for a particular class and if the distribution is skewed. This could involve iteratively learning coefficients-based techniques like stochastic gradient descent-based artificial neural networks. Support vector machines and decision trees are two examples of models that may be impacted. Adjusting the desired class distribution can be helpful. Some algorithms may overfit the minority class as a result of trying to find a balanced distribution for a dataset with a marked imbalance, which increases generalization error. Better performance on the training dataset may result, however worse performance on the holdout or test dataset may result.



```
...
# fit and apply the transform
X_over, y_over = oversample.fit_resample(X, y)
```

We can demonstrate this on a simple synthetic binary classification problem with a 1:100 class imbalance.

```
...
# define dataset
X, y = make_classification(n_samples=10000, weights=[0.99], flip_y=0)
```

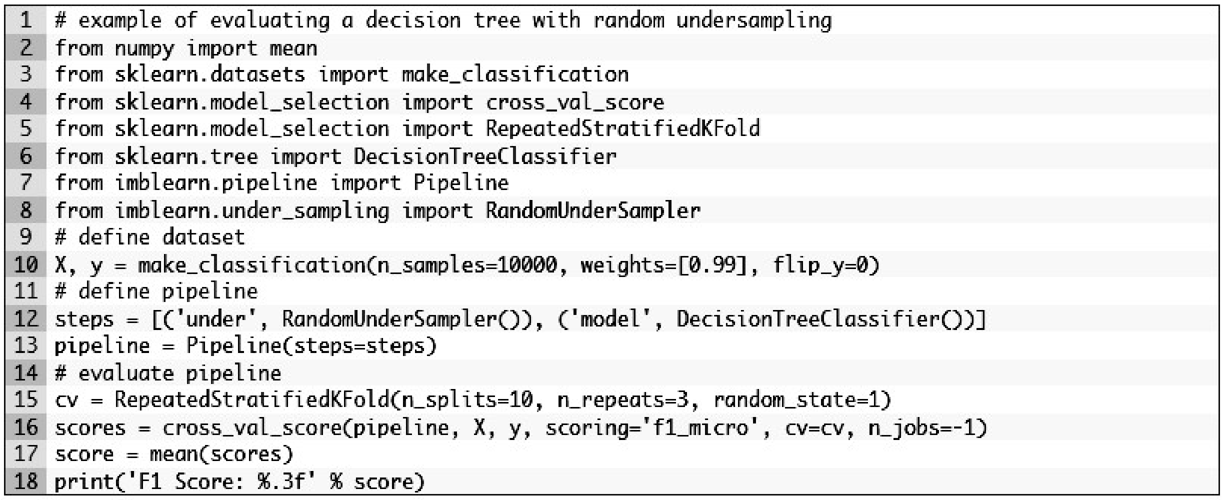
The complete example of defining the dataset and performing random oversampling to balance the class distribution is listed below.

```
# example of random oversampling to balance the class distribution
from collections import Counter
from sklearn.datasets import make_classification
from imblearn.over_sampling import RandomOverSampler
# define dataset
X, y = make_classification(n_samples=10000, weights=[0.99], flip_y=0)
# summarize class distribution
print(Counter(y))
# define oversampling strategy
oversample = RandomOverSampler(sampling_strategy='minority')
# fit and apply the transform
X_over, y_over = oversample.fit_resample(X, y)
# summarize class distribution
print(Counter(y_over))
```

4) DATASETS WITH RANDOM UNDERSAMPLING AND IMBALANCES

In random undersampling, instances from the majority class are randomly chosen and removed from the training dataset. As a result, there are fewer examples in the majority class in the training dataset that has been modified. Repeat this procedure until the desired class distribution—for instance, an equal number of samples for each class—is reached. Although a significant number of cases in the minority class can be found, this strategy may be better appropriate for datasets with a class imbalance. Undersampling has the drawback of removing samples from the majority class that might be essential, useful, or even crucial for fitting a strong decision boundary. There is no method to identify or save "excellent" or more informative instances from the majority class because examples are randomly removed.

The complete example is listed below.



```
# example of evaluating a decision tree with random undersampling
from numpy import mean
from sklearn.datasets import make_classification
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.tree import DecisionTreeClassifier
from imblearn.pipeline import Pipeline
from imblearn.under_sampling import RandomUnderSampler
# define dataset
X, y = make_classification(n_samples=10000, weights=[0.99], flip_y=0)
# define pipeline
steps = [('under', RandomUnderSampler()), ('model', DecisionTreeClassifier())]
pipeline = Pipeline(steps=steps)
# evaluate pipeline
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
scores = cross_val_score(pipeline, X, y, scoring='f1_micro', cv=cv, n_jobs=-1)
score = mean(scores)
print('F1 Score: %.3f' % score)
```

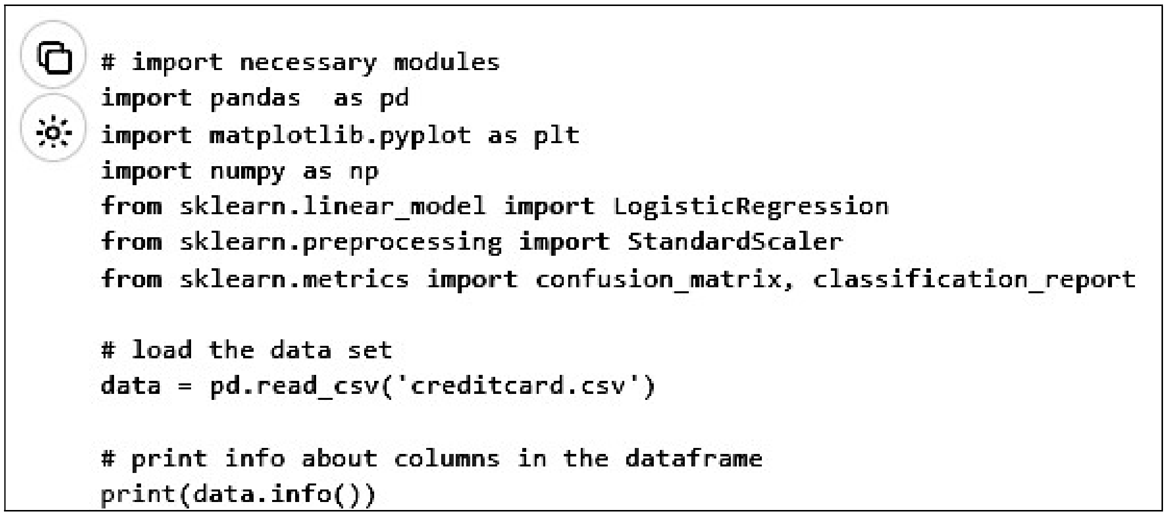
5) MIXING RANDOM UNDER- AND OVERSAMPLING

Combining random under- and over-sampling can produce some interesting outcomes. For instance, applying a small bit of undersampling to the majority class can lessen the bias on that class while applying a small amount of oversampling to the minority class to boost the bias towards these examples. Compared to using just one or the other technique alone, this can lead to better overall performance. For instance, if we had a dataset with a 1:100 class distribution, we may use oversampling to duplicate samples from the minority class in order to improve the ratio to 1:10, then use undersampling to eliminate examples from the majority class in order to further improve the ratio to 1:2.This might be done by employing an imbalanced-learn system with a RandomUnderSampler with a sampling strategy of 0.5 (50%), followed by a RandomOverSampler with a sampling strategy of 0.1 (10%).

6. EXPERIMENTAL REPORTS

In this application we try to use Python as coding platform and take SMOTE as sampling model to train the unbalanced datasets. In general we try to take credit card transactions as unbalanced dataset and now we apply SMOTE to clean that unbalanced dataset and then apply classification. The dataset comprises of credit card transaction data. Out of the 284, 807 transactions in this dataset, 492 were fraud transactions. With the positive class (frauds) accounting for 0.172% of all transactions, it is very lopsided.

Import Necessary Libraries

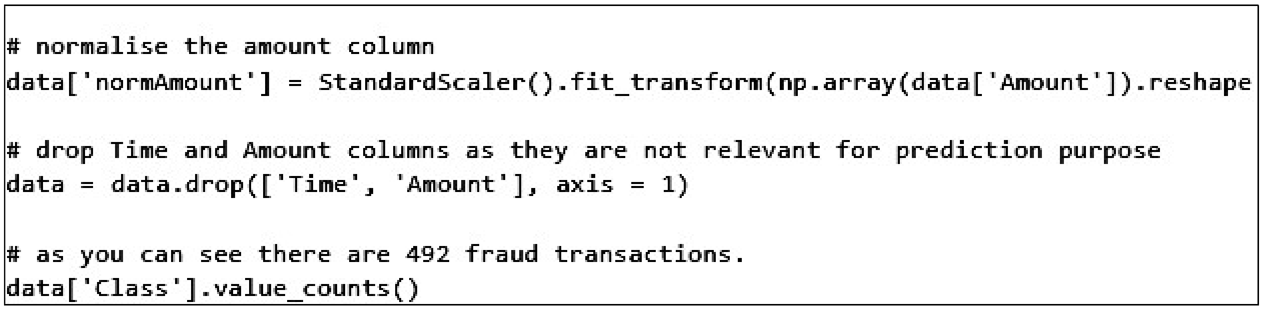


```
# import necessary modules
import pandas  as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report

# load the data set
data = pd.read_csv('creditcard.csv')

# print info about columns in the dataframe
print(data.info())
```

Normalize the Data

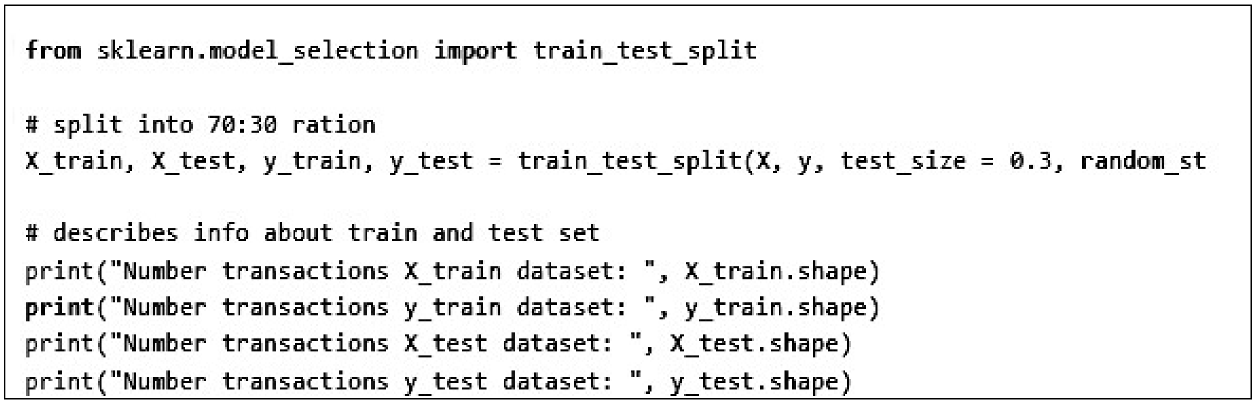


```
# normalise the amount column
data['normAmount'] = StandardScaler().fit_transform(np.array(data['Amount']).reshape

# drop Time and Amount columns as they are not relevant for prediction purpose
data = data.drop(['Time', 'Amount'], axis = 1)

# as you can see there are 492 fraud transactions.
data['Class'].value_counts()
```

SPLIT THE DATA INTO TEST & TRAIN

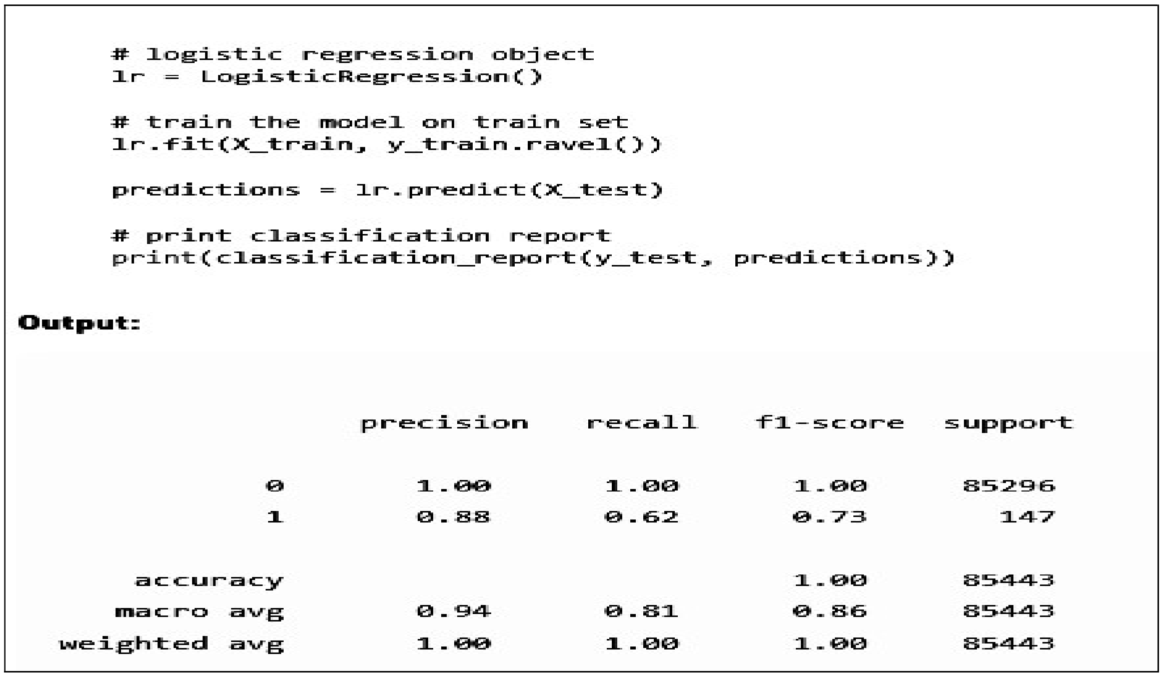


```
from sklearn.model_selection import train_test_split

# split into 70:30 ration
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_st

# describes info about train and test set
print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)
```

TRAIN THE MODEL WITHOUT HANDLING IMBALANCED CLASSIFICATION



```
# logistic regression object
lr = LogisticRegression()

# train the model on train set
lr.fit(X_train, y_train.ravel())

predictions = lr.predict(X_test)

# print classification report
print(classification_report(y_test, predictions))
```

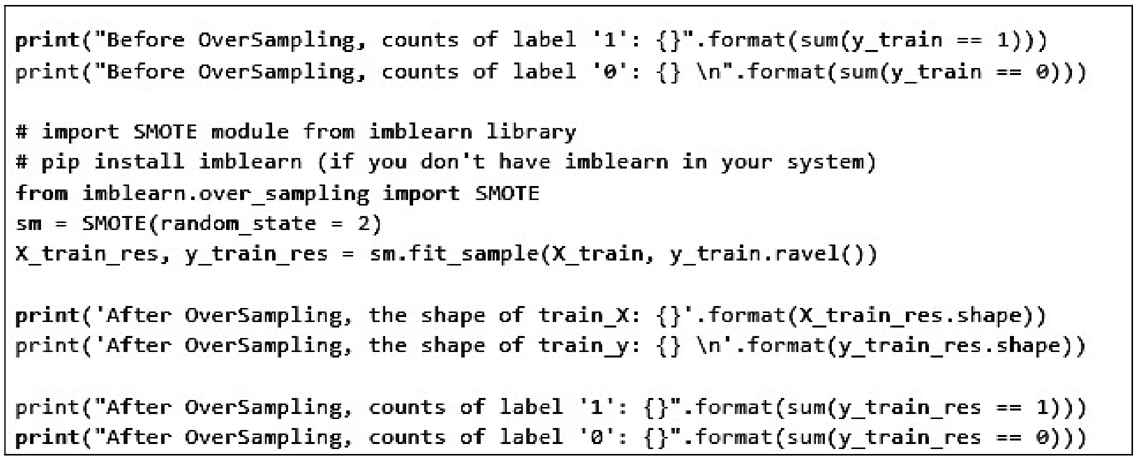
Output:

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     85296
           1       0.88      0.62      0.73       147

    accuracy                           1.00     85443
   macro avg       0.94      0.81      0.86     85443
weighted avg       1.00      1.00      1.00     85443
```

USING SMOTE



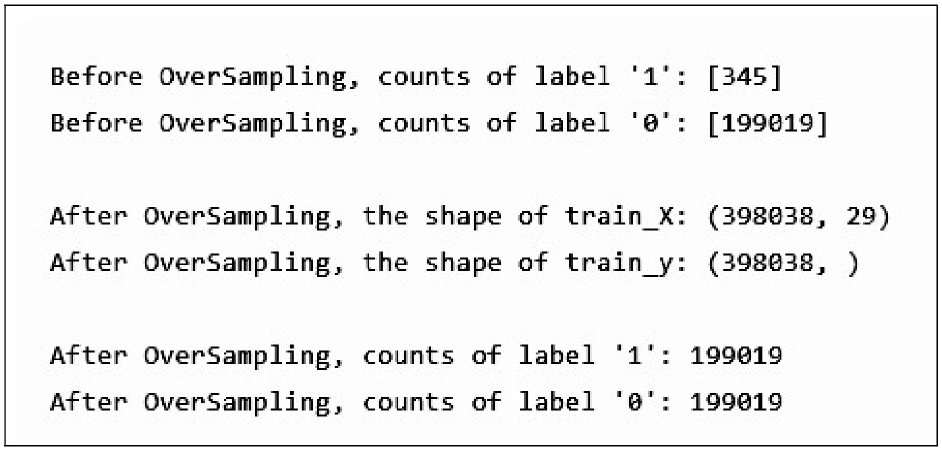
```
print("Before OverSampling, counts of label '1': {}".format(sum(y_train == 1)))
print("Before OverSampling, counts of label '0': {} \n".format(sum(y_train == 0)))

# import SMOTE module from imblearn library
# pip install imblearn (if you don't have imblearn in your system)
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
X_train_res, y_train_res = sm.fit_sample(X_train, y_train.ravel())

print('After OverSampling, the shape of train_X: {}'.format(X_train_res.shape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_res.shape))

print("After OverSampling, counts of label '1': {}".format(sum(y_train_res == 1)))
print("After OverSampling, counts of label '0': {}".format(sum(y_train_res == 0)))
```

From the above code block we can see small differences in data classification before applying SMOTE and after applying this model. This can be clearly seen in below output window.



```
Before OverSampling, counts of label '1': [345]
Before OverSampling, counts of label '0': [199019]

After OverSampling, the shape of train_X: (398038, 29)
After OverSampling, the shape of train_y: (398038, )

After OverSampling, counts of label '1': 199019
After OverSampling, counts of label '0': 199019
```

7. CONCLUSION

The most frequent problem is data imbalance. We need to prepare and balance the data since unbalancing data is difficult for standard classification algorithms to classify accurately. In this suggested work, our main focus is on how to use cost-sensitive and sampling approaches to apply data sampling techniques on such unbalanced datasets. These techniques are solutions for overcoming the problem of imbalanced data. Sampling is the most often used method for addressing uneven data at the data level. While some under-sampled algorithms outperform over-sampled ones when employing global learning classifiers, over-sampling is unquestionably more effective when utilising locally based classifiers. Future plans call for expanding the same idea to additional data sampling methods in order to improve the precision of our suggested application.

8. REFERENCES

[1] Wang, H., Utilizing Imbalanced Data and Classification Cost Matrix to Predict Movie Preferences. arXiv preprint arXiv:1812.02529, 2018.

[2] Maheshwari, S., R. Jain, and R. Jadon, A Review on Class Imbalance Problem: Analysis and Potential Solutions. International Journal of Computer Science Issues (IJCSI), 2017. 14(6): p. 43-51.

[3] Bermejo, P., J.A. Gámez, and J.M. Puerta, Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets. Expert Systems with Applications, 2011. 38(3): p. 2072-2080.

[4] Huang, C., et al., Deep imbalanced learning for face recognition and attribute prediction. IEEE transactions on pattern analysis and machine intelligence, 2019.

[5] Chan, R., et al., Application of decision rules for handling class imbalance in semantic segmentation. arXiv preprint arXiv:1901.08394, 2019.

[6] Potharlanka, J.L. and M.P. Turumella, Weighted SVMBoost based Hybrid Rule Extraction Methods for Software Defect Prediction. International Journal of Rough Sets and Data Analysis (IJRSDA), 2019. 6(2): p. 51-60.

[7] Raskutti, B. and A. Kowalczyk, Extreme re-balancing for SVMs: a case study. ACM Sigkdd Explorations Newsletter, 2004. 6(1): p. 60-69.

[8] Folorunso, S. and A. Adeyemo, Empirical Study of Enhanced Sampling Schemes with Ensembles to Alleviate the Class Imbalance Problem.

[9] Schubach, M., et al., Variant relevance prediction in extremely imbalanced training sets. F1000Research, 2017. 6: p. 1392.

[10] Abdullah, Z., et al. 2M-SELAR: A Model for Mining Sequential Least Association Rules. in Proceedings of the International Conference on Data Engineering 2015 (DaEng-2015). 2019. Springer.

[11] Wu, G.P. and K.C. Chan. Clustering driving trip trajectory data based on pattern discovery techniques. in 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA). 2018. IEEE.

[12] Breiman, L., Random forests. Machine learning, 2001. 45(1): p. 5-32.

[13] Freund, Y. and R. Schapire, A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence, 1999. 14(771-780): p. 1612.

[14] Liu, H. and M. Cocea, Granular computing-based approach of rule learning for binary classification. Granular Computing, 2019. 4(2): p. 275-283.

[15] Xia, Y., et al., A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. Expert Systems with Applications, 2017. 78: p. 225-241.

[16] Fan, Q., et al., Entropy-based fuzzy support vector machine for imbalanced datasets. Knowledge-Based Systems, 2017. 115: p. 87-99.

[17] Chawla, N.V., et al., SMOTE: synthetic minority oversampling technique. Journal of artificial intelligence research, 2002. 16: p. 321-357.