Inverse Random Under Sampling (IRUS) For Class Imbalance Problem

PROJECT TEAM

Minhaj Uddin Meraj	K180177
Hamza Edhi	K180335
Zain Shaikh	K180331

SECTION - 8A

NAME OF FACULTY

Dr. Atif Tahir

FAST SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES KARACHI CAMPUS

June 2022

CONTENTS

Page

RESEARCH GOAL 3

RETRIEVING DATA 3

DATA PREPARATION 3

DATA EXPLORATION 4

DATA MODELING 4

RESULTS 6

APPENDICES 7

REFERENCES 10

RESEARCH GOAL

Our research goal is to solve the problem of class imbalance on a data set that is true to the concept of class imbalance, with the majority of samples in major classes and very few in minor classes, and we have used a novel approach known as Inverse random under sampling [1] technique to accomplish this. The fundamental concept is to significantly under-sample the majority class, resulting in a large number of unique training sets. We next find a decision boundary that divides the minority class from the majority class for each training set. We create a composite barrier between the majority and minority classes by fusing the numerous designs together. This approach has previously been used to overcome the problem of class imbalance in binary classification. This approach will be used on our data collection.

RETRIEVING DATA

Credit Card Fraud data set was the data set we were seeking for. This data set contains a binary class study of two distinct types of labels. The credit card fraud transactions may be utilized as evidence if correctly recognized, which was the motive behind the compilation of this data set for criminological research. This data collection contains twenty-one attributes that are then utilized to determine that the transaction case is either fraud or normal based on those twenty-one qualities.

This data set obtained from Kaggle (https://www.kaggle.com/datasets/kartik2112/fraud-detection), and this credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. It covers credit cards of 1000 customers doing transactions with a pool of 800 merchants. This is a highly imbalanced class that we need to handle.

DATA PREPARATION

The first thing we saw during the correlation testing was that the attributes ‘Transaction Date Time’, ‘Credit Card Number Of Customer’. ‘First Name’, ’Last Name’ and ‘Transaction Number’ had no relationship with the target variable i.e., that does not provide us any additional information, giving us a cause to remove it and minimize the size of our data collection.

Now that our data frame has only the required data, we will now proceed to the Data processing step, i.e., handling missing values, encoding variables and scaling variables. Since, there are no missing data in our dataset, therefore we move towards the encoding step. We encoded every categorical feature which includes 'merchant', 'category’, 'gender', 'street', 'city', 'state', 'job' one by one using one-hot encoding technique and also replace the date of birth column with age column by converting every date of birth into age. After the on hot encoding steps, now we remove all the duplicates from our data. As of now the data is in its cleanest form, of course apart from the class imbalance problem, we start moving towards the data modeling step. In this we divide our data into (70/30) and prepare our training set (70%) while focusing all the issues of imbalance class and remain hidden the test set from our model.

DATA EXPLORATION

Following the extraction of the data, it was rigorously examined for any unexpected or unusual findings. 'merchant', 'category', 'amt', 'gender', 'street', 'city', 'state', 'zip', 'lat', 'long', 'city_pop', 'job', 'unix_time', 'merch_lat', 'merch_long', 'age' were the sixteen properties in our data set. Continuous values were used for all of the attributes. In total, 1296674 samples were available in the data set.

We began to observe after probing a little deeper. In our data set, there is a class imbalance. As There were two distinct classes in a perfect scenario. would have been if each class had been made up of equals samples, however, this is not the case with our set of data So, before we go any further, let's have a look at Examine the sample size per class. Our first class which consider as normal class have 84% ratio while fraud class have 16%. This shows how large of a disparity exists between the classes.

Following that, we dug deeper into our data set. Pearson was used. The variables are correlated, and a heat map is used to clearly separate each correlation. This gave us a rough sense of the variables and their relationships with one another. Apart from that, there was a good correlation between the factors and the target variable. Missing values are another stumbling block in a data collection that has a detrimental impact on model training. There were no missing values in any of the columns in our data collection, thus there was no need to fill it in. We began to comprehend the distribution of each variable in the data set after discovering correlation. The distribution of a data set's values across a curve is described by its distribution.

DATA MODELING

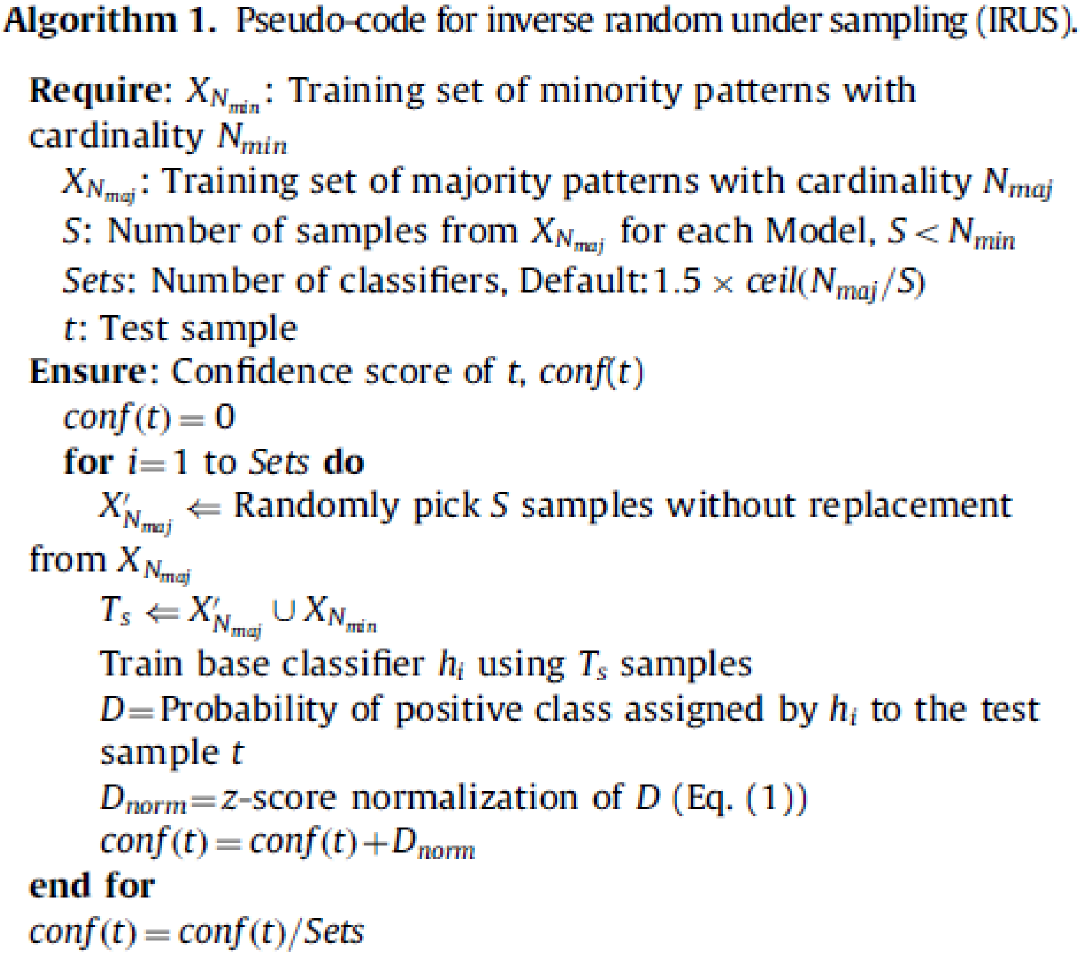
The cardinalities of the majority and minority classes are inverted in the inverse random under-sampling approach [1]. As a result, the majority class is divided into subsets, each of which is smaller than the minority class. The basic idea underlying Inverse Random is as follows: The goal of sampling is to keep the rate of sampling high. Inversion of the imbalance rate yields a true positive rate; however, this also leads to a high percentage of false positives. For this Bagging is a method used in ensembles to achieve this [2], by merging the developed detectors with the help of this we can control the rate of false positives through fusion.

Now we'll take a look at the issue of class imbalance problem. We'll use the Inverse Random Under Sampling approach to solve this problem. Let allow us to comprehend how this technique's mechanism is effective. Using the bagging approach [4], this technique enhances the true positive rate while lowering the false positive rate, which rises as a result of the higher true positive rate.

Bagging is an ensemble classifier based on a straightforward concept. Bagging creates a large number of training subsets from a bigger data set. Each subset of samples is created at random, with each sample in the subset being chosen with equal chance and replacement. A prediction algorithm is used to each subgroup. The final prediction is then determined by a hard/soft voting.

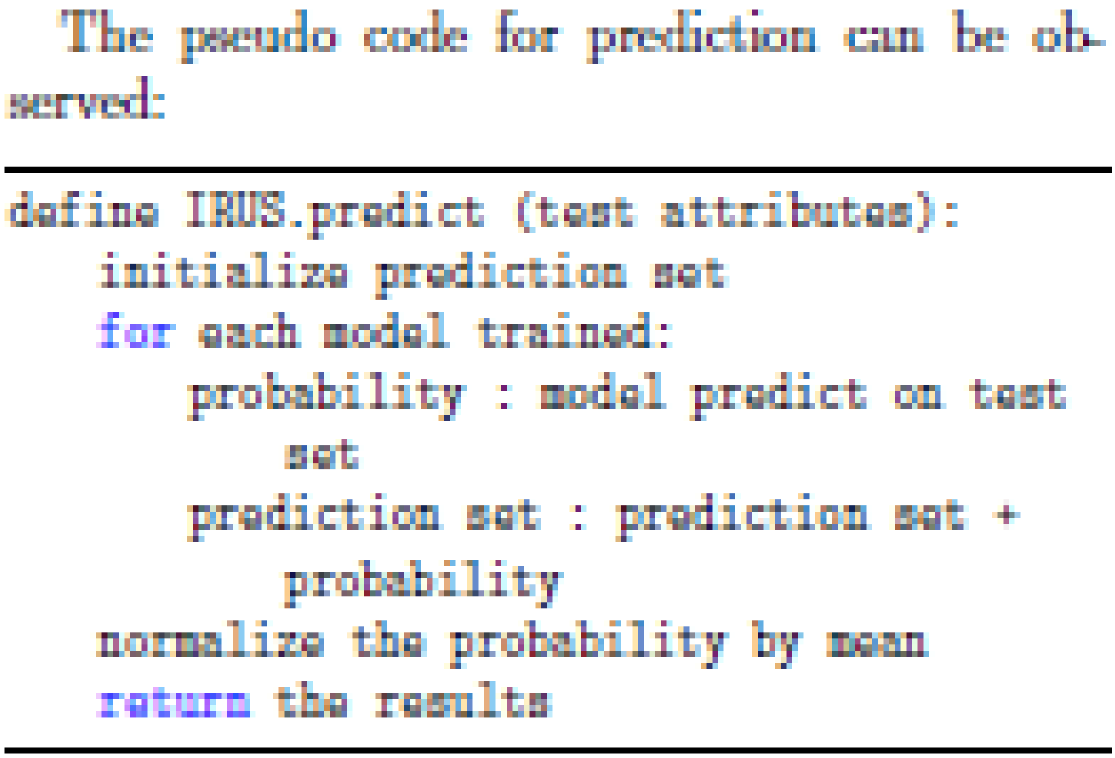
We go through all of the minority classes one by one, examining them on a binary level. The fraud class is defined as a collection of minority classes. In the publication "Inverse Random Under Sampling for class imbalance problem and its application to multi-label classification," which was handed to us, we used the Inverse Random Under Sampling pseudo code. We employed a decision tree as our base classifier, and the mean approach (Soft Voting) for combining was used for bagging. The training pseudo code can be found below.

Inverse Random is usually the positive class with a relatively small number of samples. Data manipulation occurs during sampling in such a way that the sample size of the majority class, which is the negative class, is reduced to the point where the sample size of the positive class surpasses the sample size of the negative class. This is accomplished by repeatedly under sampling until each subset is less than the positive classes. Now that the positive class has more samples than the negative class, the focus changes to the positive class. Each training set produces a single classifier design that prioritizes the positive class. Then, via fusion, a composite border can be created by integrating the individual designs.



Algorithm 1. Pseudo-code for inverse random under sampling (IRUS).

```
Require: $X_{N_{min}}$: Training set of minority patterns with
cardinality $N_{min}$
  $X_{N_{maj}}$: Training set of majority patterns with cardinality $N_{maj}$
  $S$: Number of samples from $X_{N_{maj}}$ for each Model, $S < N_{min}$
  $Sets$: Number of classifiers, Default: $1.5 \times ceil(N_{maj}/S)$
  $t$: Test sample
Ensure: Confidence score of $t$, $conf(t)$
  $conf(t) = 0$
  for $i = 1$ to $Sets$ do
    $X'_{N_{maj}} \Leftarrow$ Randomly pick $S$ samples without replacement
from $X_{N_{maj}}$
    $T_s \Leftarrow X'_{N_{maj}} \cup X_{N_{min}}$
    Train base classifier $h_i$ using $T_s$ samples
    $D =$ Probability of positive class assigned by $h_i$ to the test
    sample $t$
    $D_{norm} = z$-score normalization of $D$ (Eq. (1))
    $conf(t) = conf(t) + D_{norm}$
end for
$conf(t) = conf(t)/Sets$
```



The pseudo code for prediction can be observed:

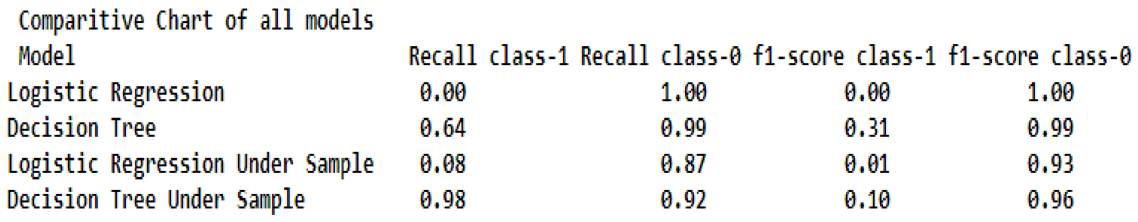
```
define IRUS.predict (test attributes):
    initialize prediction set
    for each model trained:
        probability : model predict on test
            set
        prediction set : prediction set +
            probability
    normalize the probability by mean
    return the results
```

RESULTS

We employed the F1 score [4] and the ROC-AUC curve [5] to test our model using better techniques of evaluation.

The ROC-AUC is a performance indicator for classifying issues at different thresholds. The term AUC stands for Area Under the ROC Curve, and it refers to the complete two-dimensional area beneath the entire ROC curve. The harmonic mean of precision and recall is also known as the F1 score. It is a metric for determining how accurate a model is on a given data set.

To further understand how the technique of Inverse Random Under Sampling helps us, we utilized various different machine learning models that did not use this technique to see how well this technique performed in comparison to these models that did not use Inverse Random Under Sampling. For this purpose, the model we used logistic regression, decision tree, without sampling and with sampling. First, we test our results on imbalanced class and then we test our models by using random under sampling techniques.

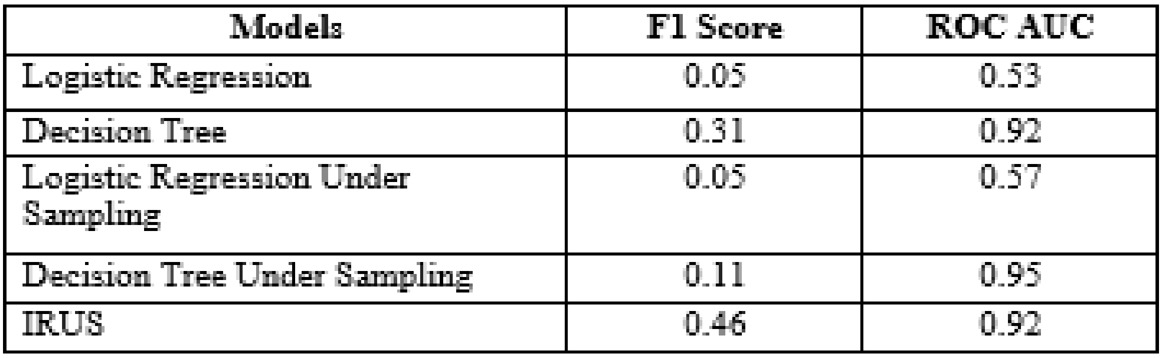


```
 Comparitive Chart of all models
 Model                              Recall class-1 Recall class-0 f1-score class-1 f1-score class-0
Logistic Regression                  0.00             1.00           0.00             1.00
Decision Tree                        0.64             0.99           0.31             0.99
Logistic Regression Under Sample     0.08             0.87           0.01             0.93
Decision Tree Under Sample           0.98             0.92           0.10             0.96
```

Comparative Chart

Finally, we put the Inverse Random Model to the test. Other models that do not use this technique are sampled with other models that do not use this technique. The IRUS model performed well as compared to the other models

The result is evidence enough that the technique Inverse Random Under Sampling provides a huge increase in performance. It helps to solve the problem of class imbalance and provides and evident jump in results then other conventional method. The full table of all the models with their F1 score and ROC score are shown below



Models	F1 Score	ROC AUC
Logistic Regression	0.05	0.53
Decision Tree	0.31	0.92
Logistic Regression Under Sampling	0.05	0.57
Decision Tree Under Sampling	0.11	0.95
IRUS	0.46	0.92

APPENDICES

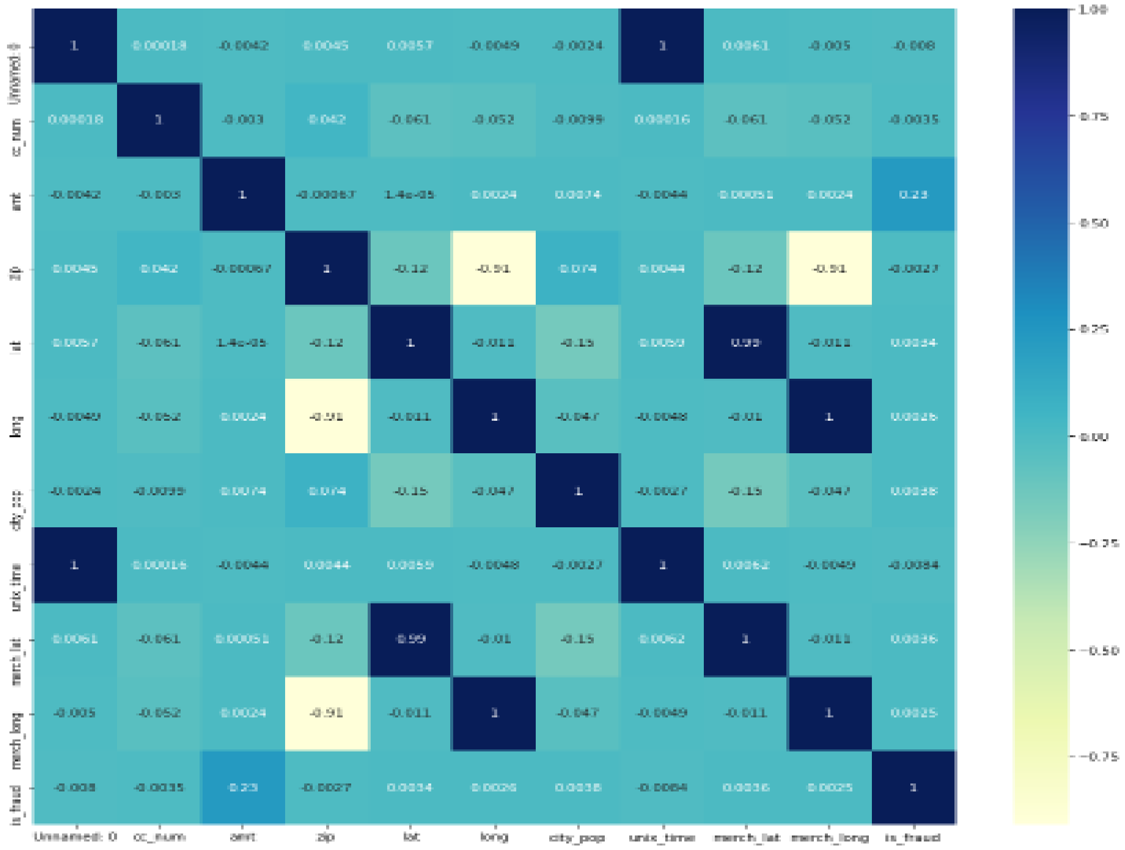


Figure 1: Correlation Heatmap

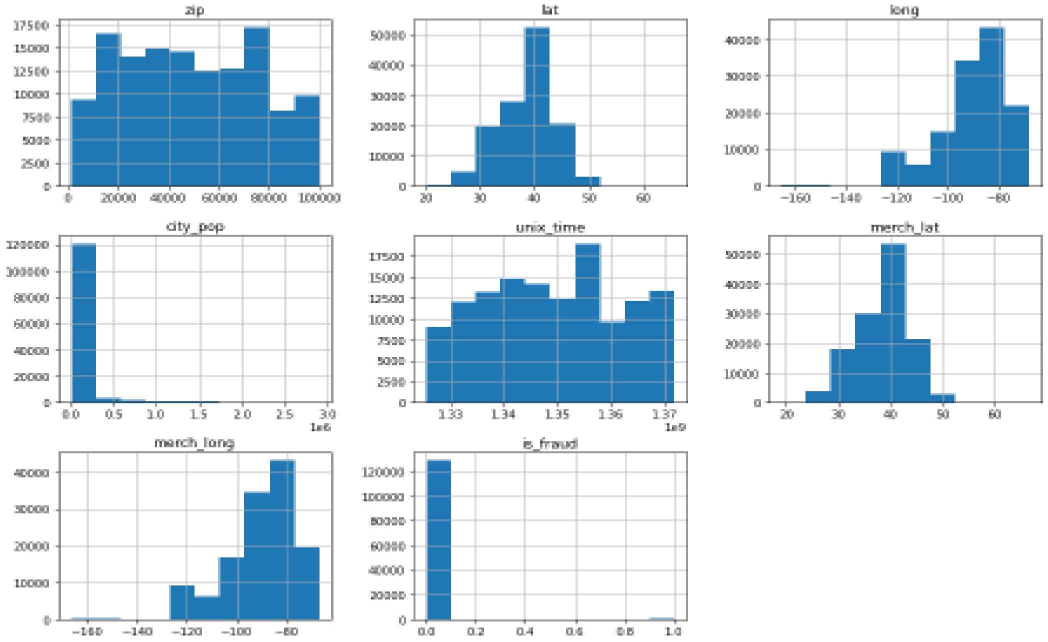


Figure 2: Data Distributions

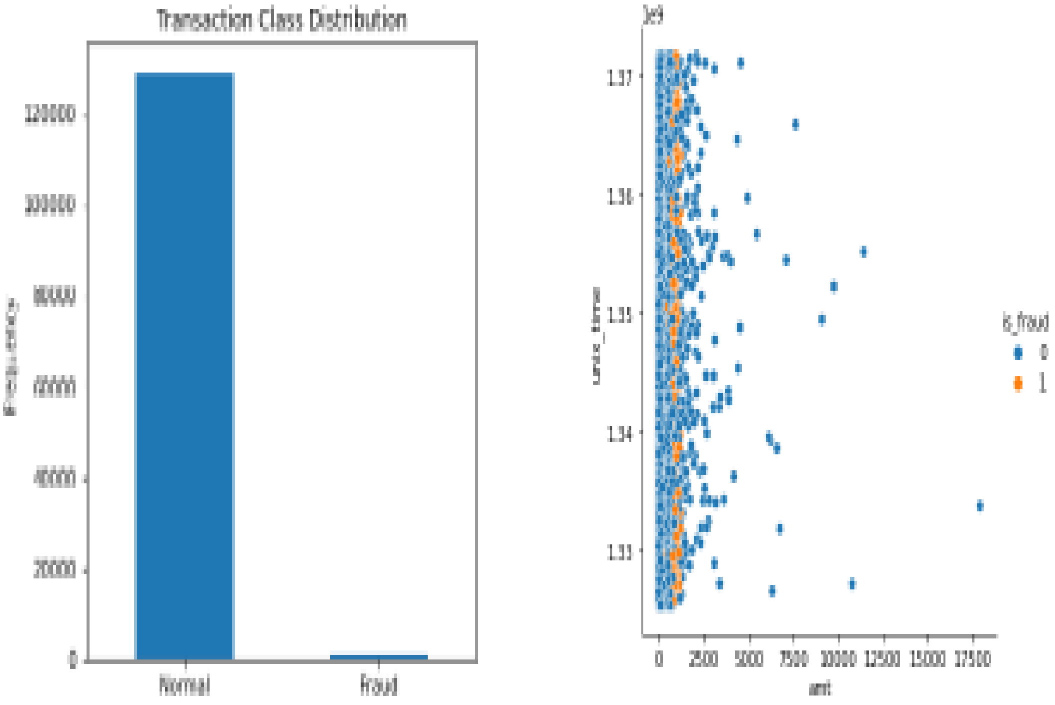


Figure 3: Target Data Distributions

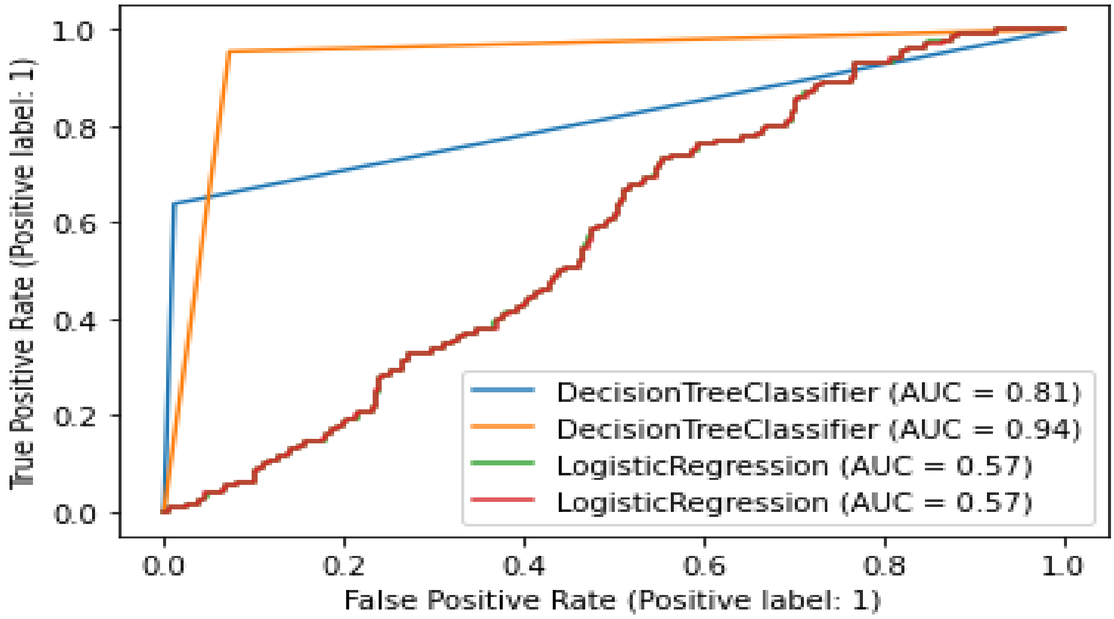


Figure 4: ROC Curve Without IRUS

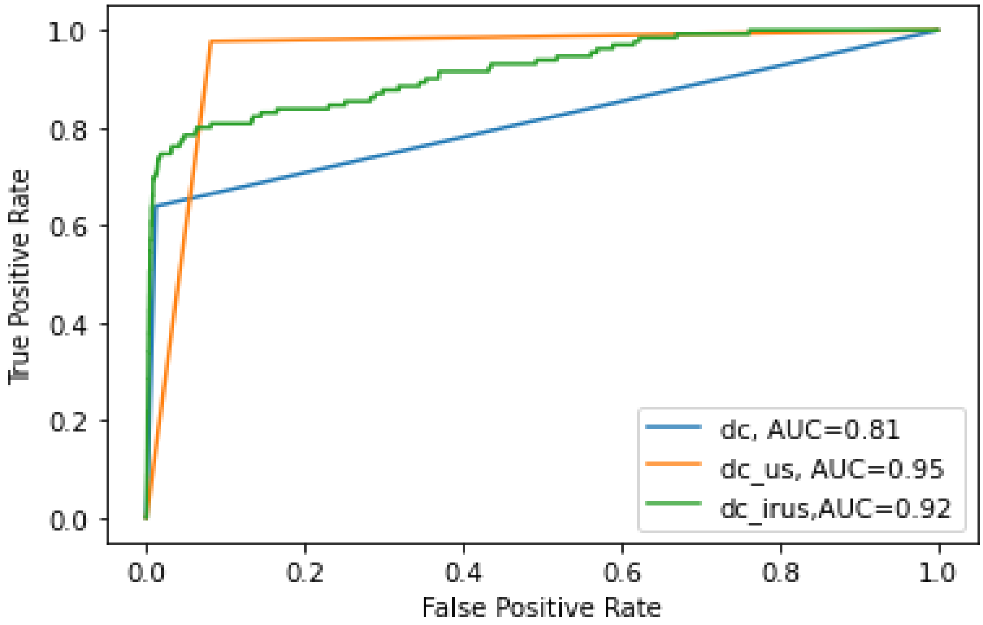


Figure 4: ROC Curve With IRUS

REFERENCES

[1] Tahir, M.A., Kittler, J. and Yan, F., 2012. Inverse random under sampling for class imbalance problem and its application to multi-label classification. Pattern Recognition, 45(10), pp.3738-3750.

[2] Breiman, L., 1996. Bagging predictors. Machine learning, 24(2), pp.123-140.

[3] Swain, P.H. and Hauska, H., 1977. The decision tree classifier: Design and potential. IEEE Transactions on Geoscience Electronics, 15(3), pp.142-147.

[4] Fujino, A., Isozaki, H. and Suzuki, J., 2008. Multi-label text categorization with model combination based on f1-score maximization. In Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II.

[5] Fawcett, T., 2006. An introduction to ROC analysis. Pattern recognition letters, 27(8), pp.861-874.