Pattern Recognition and Machine Learning
Indian Institute of Technology, Jodhpur

MINOR PROJECT
CREDIT CARD FRAUD DETECTION

Anupam Singh Bhadouriya (B21CS086)
Vidit Garg (B21AI045)
Vidit Agrawal (B21AI058)

Abstract:

This paper describes various techniques for handling imbalanced dataset problems. Imbalanced data sets exist in many real-world domains,
such as spotting cancer patients, fraud detection, and many other real-world problems. These types of datasets are often imbalanced where one class or category of data is represented by significantly fewer instances (also known as the minority class) than the other class. Imbalanced datasets tend to become biased toward the majority class, and not learn to distinguish the minority class effectively.

Introduction:

Imbalanced learning strategies are mainly divided into three broad categories -
Data level method: This includes resampling the datasets which can be achieved by either oversampling where we over-sample the minority class or undersampling where we undersample the majority class to re-balance the class distribution and remove the biases in the classification of data.
Ensemble method: This includes using ensemble techniques to calculate weighted average of different base classifiers based on a majority vote. The classifiers with higher values of g-mean and F1 score are given more weight in comparison to others so that the correct prediction of minority class is emphasized.
Algorithm level method: This includes finding the cost functions for our classifier model and training model based on cost function giving more priority to minority class. This can be done by using various classifier models such as ANN, AdaBoost etc.

<u>Dataset:</u>

The provided dataset consists of credit card transactions out of which 99.83% of the data is classified as non-fraud transactions and the rest 0.17% is classified as fraud transactions. The dataset is an example of an imbalanced dataset where the general classifier models have a bias towards the majority class (non-fraud transactions) since the minority class (fraud transactions) is less in number.

<u>Preprocessing of Dataset:</u>

The preprocessing involved identification of null values,outliers, value count of the label class. Scaling of the unscaled feature columns is also done using Standard and Robust Scaler. The visualization of all the feature columns is done by plotting histograms for each of them. The box plot is also plotted for features to visualize the outliers. A correlation heatmap is also plotted for visualizing the correlation between different columns of feature set. The preprocessed data is then splitted into training,testing and validation dataset to be used by different classifier models.

1. <u>Data Level Processing:</u>

As we have already mentioned, data-level solutions include many different forms of re-sampling such as random oversampling with replacement, random undersampling, combinations of the above techniques, and other techniques too so we will be discussing the implementation of these techniques on our dataset and their corresponding results.

1.1 <u>UnderSampling:</u>

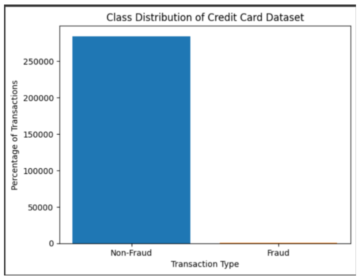
Undersampling is a technique that involves the removal of random data from the majority class and keeping all of the data in the minority class this is done to balance the dataset and reduce the biases of the majority class. But the disadvantage of undersampling involves the loss of information that could be valuable while training the model.

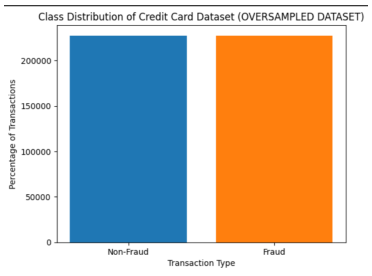
1.2 <u>Oversampling:</u>

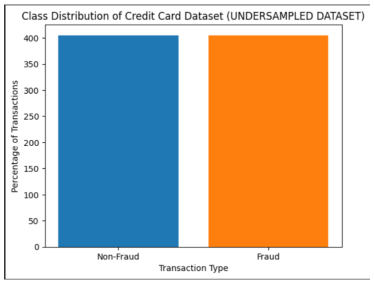
Oversampling is a technique that involves the selection of random data points from the minority class and duplicating them into the dataset in order to increase the size of the minority class and finally obtain a balanced dataset. But the disadvantage of oversampling is that it overfits the model since it makes exact copies of the minority class examples. In this way, a traditional classifier might construct rules that cover a single, replicated, example. And thus making the model inefficient and more prone to overfitting.

1.3 <u>SMOTE</u> (Synthetic Minority Oversampling Technique):

SMOTE is a technique that involves the generation of random samples for the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling. The working of SMOTE involves, the selection of a random minority class instance say ‘a’, and then using the KNN algorithm to find its K nearest neighbors then synthetic instances are created by choosing one of the K neighbors and connecting ‘a’ and ‘b’ to form a line segment in the feature space. And then drawing random data points in between this line and thus generating a synthetic dataset.







Ensemble Method(Ensemble-KNN):

The En-KNN algorithm utilizes the K-nearest neighbor algorithm as the base classifier. It calculates the weight coefficients of each base classifier by combining the G-mean and f-measure, which are effective measures for evaluating the performance of classification methods for imbalanced data. The En-KNN trains different KNN base classifier models using different subsets of the training dataset and then integrates these models into En-KNN using the weight of each base classifier. The g-mean is the geometric mean of the recall and specificity for both positive and negative classes, while the f-measure is a weighted harmonic mean of the recall and precision of the positive class. g-mean and f-measure have their best and worst values at 1 and 0, respectively. If both g-mean and f-measure are greater than 0.5, the ensemble weight of each base classifier is set to 1, otherwise, the weights are set to 0. Finally, the base classifiers are integrated using a weighted voting strategy.

G-mean and F- measure can be obtained respectively by

$$g\ mean\ =\ \sqrt{specificity\ *\ recall}$$

$$f\ measure\ =\ \frac{(\alpha^2+1)*precision*recall}{\alpha^2(precision+recall)}$$

$$\omega_t\ \Rightarrow 1\ if\ g\ mean\ >\ 0.5\ and\ f\ measure\ >\ 0.5$$

$$\omega_t\ \Rightarrow 0\ \ otherwise$$

When α=1, f-measure becomes F1-score.

By using a weighted voting strategy,more priority will be given to classifiers with better g-mean and f1score i.e.the classifiers predicting the minority class better in comparison to other classifiers. Using this ensemble technique, we get the final prediction for the label class using these weights.

Cost-Sensitive Methods:

Cost-Sensitive AdaBoost:

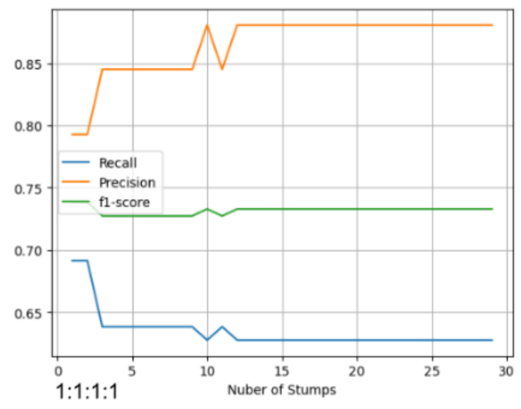
The AdaBoost algorithm uses a weighting strategy that increases the weights of misclassified samples and decreases the importance of correctly classified samples. However, this approach treats samples from different classes equally, which is not ideal for addressing imbalanced class problems. In such cases, the objective is to improve the identification of the minority class. To achieve this objective, it is essential to significantly weigh the minority class. A better-boosting strategy should prioritize samples that are more important for accurate identification.

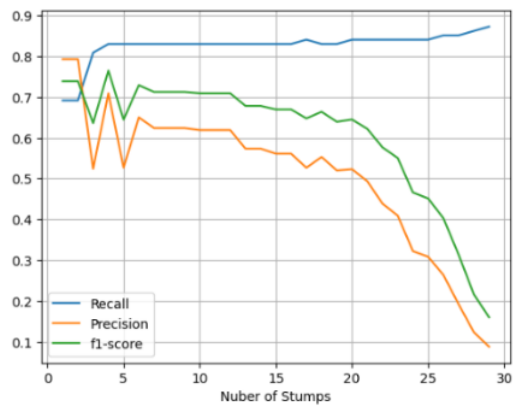
We have changed the formula by which weights are updated.

$$W_{new} = W_{old} * exp(\alpha * C)$$

Where α is the weight of our base classifier (Decision stump), the calculation of this term is not changed.

The intuition behind assigning the weights to different types of error is that we want to give the highest priority to the sample in our next base model if it was positive(fraud transaction) and classified as negative (False Negative - FN). At the same time, we do not want to decrease the weight of the sample if it was classified correctly as fraud (true positive) as we have very less samples in our data, So we will increase the weightage of both FN and TP but FN will be given more weight. Similarly, if a non-fraud transaction is classified as a fraud (false positive - FP) we will increase its importance but the increment would be less than the increment in the importance of TP and TN. For the correctly classified non-fraud transactions (True Negative - TN) we would decrease the importance of the sample.





After doing some experiments, the best value of weights was found using the intuition discussed above, final assigned weights are,
$C_{TP} = 50, C_{TN} = -50, C_{FP} = 50, C_{FN} = 500$. negative
sign denotes that we are decreasing the weight for this type of error. We were getting the best f1-score for 4 stumps. The graph on the left shows results for the traditional ADABoost. **F1-score_fraud - 0.76, Recall_fraud - 0.82, Precision_fraud - 0.70**. We can also observe a tradeoff between precision and recall for the fraud transaction class(minority class). As compared to traditional ADABoost, F1-score_fraud - 0.72, Recall_fraud - 0.63, Precision_fraud - 0.84, there is no considerable difference between the f1 scores, but there is a considerable difference in the recall scores, which is the most important metric after the f1-score.

Cost-Sensitive Neural Network

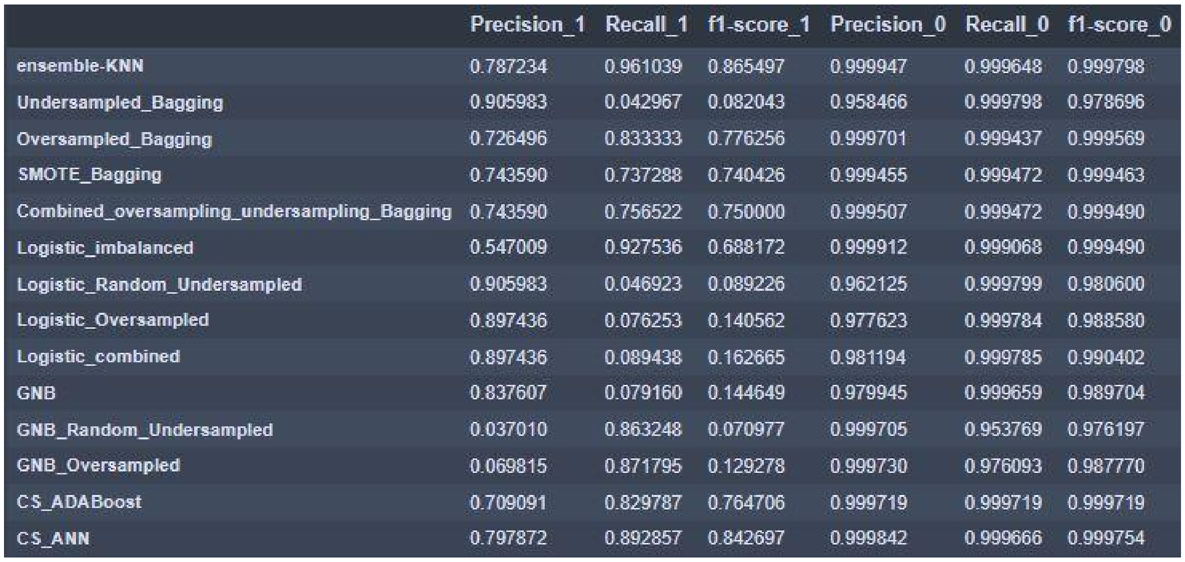
A neural network using Tensorflow was trained. It is having an input layer of size 28 and two hidden layers each of them having 32 perceptrons, and an output layer contains only one perceptron. As it is a binary classification problem, we have used sigmoid as an activation function for the output layer. While training of neural network it calculates the loss in each epoch and tries to update all the trainable parameters such that the loss is minimized. We have defined a new loss function, which adds loss to total loss based on the true class of the sample. As we have used a sigmoid activation function we will be getting a float value as output (n). New loss for a sample

$$L(n) = \begin{array}{l} w * (n - 1)^2 \quad , if\ true\ class\ is\ 1 \\ (1 - w) * (n - 0)^2, if\ true\ class\ is\ 0 \\ where\ w\ is\ the\ weight\ assigned\ to\ class\ 1,\ for\ traditional\ loss\ function\ w\ is\ 0.5 \end{array}$$

Prediction of the class of a sample will be based on a threshold, if n is greater than this threshold then the prediction will be classified as fraud transaction, else non-fraud transaction. We have to find the best value of w and threshold.

We have split our data into 70:10:20 ratios named train, validation, and test dataset respectively. Using the grid search method, the optimal values of both parameters were found by testing the models on our validation dataset. The final w is 0.9 and the threshold is 0.9 which gave the following results **Precision_1: 0.80, Recall_1 0.89, f1-score_1: 0.84, Precision_0: 0.999, Recall_0: 0.999, f1-score_0: 0.999**.

Compared to the traditional ANN Precision_1: 0.8191489361702128, Recall_1: 0.8105263157894737, f1-score_1: 0.8148148148148148, Precision_0: 0.9996834775269045, Recall_0: 0.9997010568519528, f1-score_0: 0.9996922671121466



	Precision_1	Recall_1	f1-score_1	Precision_0	Recall_0	f1-score_0
ensemble-KNN	0.787234	0.961039	0.865497	0.999947	0.999648	0.999798
Undersampled_Bagging	0.905983	0.042967	0.082043	0.958466	0.999798	0.978696
Oversampled_Bagging	0.726496	0.833333	0.776256	0.999701	0.999437	0.999569
SMOTE_Bagging	0.743590	0.737288	0.740426	0.999455	0.999472	0.999463
Combined_oversampling_undersampling_Bagging	0.743590	0.756522	0.750000	0.999507	0.999472	0.999490
Logistic_imbalanced	0.547009	0.927536	0.688172	0.999912	0.999068	0.999490
Logistic_Random_Undersampled	0.905983	0.046923	0.089226	0.962125	0.999799	0.980600
Logistic_Oversampled	0.897436	0.076253	0.140562	0.977623	0.999784	0.988580
Logistic_combined	0.897436	0.089438	0.162665	0.981194	0.999785	0.990402
GNB	0.837607	0.079160	0.144649	0.979945	0.999659	0.989704
GNB_Random_Undersampled	0.037010	0.863248	0.070977	0.999705	0.953769	0.976197
GNB_Oversampled	0.069815	0.871795	0.129278	0.999730	0.976093	0.987770
CS_ADABoost	0.709091	0.829787	0.764706	0.999719	0.999719	0.999719
CS_ANN	0.797872	0.892857	0.842697	0.999842	0.999666	0.999754

Table1: Precision, Recall, F1-Score values for classifiers trained on the credit card dataset. Class 1 represents fraud transactions, Class 0 represents non-fraud transactions.

Results:

From Table1, we can see that Ensemble-KNN, Bagging Classifier on Oversampled data and Cost-sensitive ANN are giving high F1-score when trained over the imbalance credit card dataset.

- Recall for the minority class is the most important factor for analysing the prediction of the classifiers. F1-Score for the minority class(f1-score_1) is the next important since we are focusing on the correct prediction of the minority (fraud transaction) class.
- The recall for the Gaussian Naive Bayes classifier on undersampled data is 95% which is significantly lower than the any other classifier. The loss of information due to undersampling results in the decrease of recall for the majority class.
- In the Logistic Regression Classifier on undersampled data, the majority class data is reduced and hence the false positive(the number of negative samples classified as positive) gets reduced. As a result,precision for the minority class(Precision_1) improves.
- Ensemble KNN gives exceptionally high recall and comparatively better F1-Score for the minority class.
- Cost sensitive ANN is performing well on the imbalanced dataset and giving a good recall and F1-Score.
- SMOTE is giving better results than undersampling and oversampling because it does not duplicate minority class like oversampling. SMOTE makes new points based on KNN and hence, provides average precision and recall for classifier models.

Conclusion:

In this paper, we performed a comparative analysis study of different methods used for training of models for imbalanced datasets.
We can conclude from the study that almost all different kinds of machine learning classifiers like Ensemble-KNN and Cost sensitive Adaboost and Cost sensitive ANN provide quite promising results over the provided dataset.
Ensemble-KNN method is giving **best results** for the recall and f1-score since more weights are given to classifiers with considerably high g-mean and f1-score. So, finally the predictions based on the majority vote are more influenced by models predicting the minority class better.
Cost-sensitive Neural network deals with the imbalanced dataset on algorithmic level. The CS-ANN model is the next best classifier for the imbalanced dataset. In addition, the Cost sensitive ADABoost also performs well on the dataset in which we have tuned four parameters intuitively.

Contributions:

All the team members contributed equally in the literature review and implementation of algorithms.