Classification of Imbalanced Data: A Comparison of Logistic Regression and Gaussian Mixture Model in Conjunction with Resampling

Chongqi Wu*
California State University, East Bay, Hayward, California, USA

Aishwarya Choudhary
California State University, East Bay, Hayward, California, USA

Steve Peng
California State University, East Bay, Hayward, California, USA

Jia Guo
California State University, East Bay, Hayward, California, USA

This paper considers multiple approaches for classifying imbalanced data. We compare logistic regression and Gaussian Mixture Model (GMM) for classification in conjunction with over-sampling and under-sampling techniques. When applied to a credit card fraud dataset combined with over-sampling, both logistic regression and GMM demonstrate reliable performance. Over-sampling tends to overperform under-sampling. A combination of resampling and clustering methods, such as GMM, is a legitimate alternative to handle imbalanced classification problems. Furthermore, we propose a framework to help define a repertoire of tools for combating imbalanced learning problems and improving model performance.

* Corresponding Author. E-mail address: chongqi.wu@csueastbay.edu

I. INTRODUCTION

The issue of imbalanced data in classification problems, also known as the imbalanced class distribution of a dataset, refers to the fact that datasets often have many more observations or instances in some classes than in others. Data imbalance, sometimes severe, is prevalent in machine learning models of classification. According to Sun et al. (2009) and Maheshwari et al. (2017), the issues of imbalanced data have been observed in many fields, from text classification, image recognition, and anomaly detection, to medical diagnosis, detection of fraudulent calls, detection of oil spills, and risk management.

Class imbalance, particularly extreme class imbalance, causes a tremendous challenge for machine learning classification models. One of the authors worked on a dataset of tens of thousands of computer game players. The game players belong to two classes: regular players who play the games for free (negative class, majority class in this case), and VIP members or paying customers who pay for advanced game features (positive class, also the minority class). However, of all the players, less than 1% are VIP, paying customers. That is, the ratio of negative (majority) and positive (minority) classes is 99:1. The goal is to build classification models that generalize well. A naïve baseline model can predict any game player in the negative class, i.e., non-paying customers. This naïve model will have a prediction accuracy of 99%. However, it has a false negative ratio of 100% and thus is effectively and practically useless. It is tough to find a classification model that performs well on most key performance metrics with severe class imbalance, including but not limited to accuracy, false positive and false negative ratios, F-score, and ROC curve.

Resampling and its variants are the dominant methods employed to battle the issue of class imbalance. The Literature Review section below provides more information on other techniques such as ensemble learning, kernel-based approach, and cost-sensitive learning. It is not uncommon to combine two or more of these methods. Indeed, Wu (2017) combined ensemble learning and resampling methods to predict future NBA all-stars. Performance is mixed, dependent on the nature of datasets. Different techniques and their combinations are typically experimented to identify a better approach for a particular imbalanced dataset.

Clustering methods, when used to combat class imbalance, are considered a subcategory of the resampling method, according to He and Garcia (2009). They reported the usage of k-means, a commonly used clustering method, to classify imbalanced data. On the other hand, our work proposes and adopts a more sophisticated clustering algorithm in Gaussian Mixture Model (GMM), combined with the over-sampling technique, to handle imbalanced data with two classes. The idea is relatively straightforward. After training and test data split, we train two Gaussian clusters, one for each class. Each class or cluster is then described by a Gaussian distribution, with parameters identified in model training. Then test data are fed into the two Gaussians, which classify the data according to the resulting probability density. If an observation has a higher probability density from the first Gaussian than from the second Gaussian, we predict that it belongs to the first class and vice versa. We can easily generate such methods for multi-class problems by training more Gaussians, one for each class. We also consider the classical logistic regression method, combined with resampling techniques. Indeed, logistic regression with oversampling has provided the best performance for the credit fraud dataset. It is worth noting that the performance can be dataset sensitive. Besides, we can much more easily apply GMMs to multi-class problems than logistic regression.

II. LITERATURE REVIEW

Sun et al. (2009) offered an excellent review of research related to imbalanced data. They reported the prevalence of class imbalance in practice, ranging from fraud detection and medical diagnosis to detection of oil spills and manufacturing plants. They identified learning difficulties with standard classification models, including decision

trees, neural networks, Bayesian classification, Support Vector Machine, associative classifiers, and KNN. They reported solutions at both data and algorithm levels. At the data level, a popular remedy is resampling, oversampling the small class. It invites the questions of what is or how to decide the optimal class distribution. At the algorithm level, a common strategy is to choose an appropriate inductive bias. Another alternative is cost-sensitive learning, taking into consideration the costs of different misclassification types. Boosting and ensemble learning are also popular choices, which combine many a classifier trained from the original data to improve generalization ability. Performance results are mixed. Sun et al. (2009) also suggested future research direction in multi-class imbalance problems.

He and Garcia (2009), another review on imbalanced learning, provided a different categorization framework than Sun et al. (2009). He and Garcia (2009) summarized solutions to imbalanced learning into four broad categories: (1) resampling methods, (2) cost-sensitive methods, (3) kernel-based methods, and (4) other methods. According to He and Garcia, the technique used in this paper belongs to a subcategory under resampling methods: Cluster-Based Sampling Method, usually a Cluster-Based Oversampling method (CBO). The CBO algorithm, however, only makes use of the k-means clustering technique. Our methods, on the other hand, focus on GMM. He and Garcia (2009) also reviewed assessment metrics for imbalanced learning. They argued that metrics such as F-measure, G-Mean of Precision and Recall, ROC curves, and Precision-Recall curves combined could effectively evaluate imbalanced learning methods. Synthetic generation of minority class data (SMOTE), one resampling technique, has been used to generate a more balanced dataset and offset imbalanced data limitations. In a more recent review, Maheshwari et al. (2017) studied factors such as feature selection that influence the dataset and lead to data imbalance.

Rastogi et al. (2018) implemented SMOTE in a distributed environment under spark. An uncorrelated cost-sensitive multiset learning (UCML) approach proposed in Wu et al. (2017) is an under-sampling method. It partitions the majority class into multiple blocks, balanced to the minority class, and combines each block with the minority class to construct a balanced sample set. The approach was applied to some experimental datasets and outperformed some other methods. It is a resampling method in nature and does not differ from oversampling that duplicates minority class. Indeed, it can be troublesome when the minority class is minimal, and the resulting block is small. On the other hand, Krawczyk (2016) offered several unbalanced learning research areas focusing on applications. It extended the scope to a more general imbalanced domain.

Xiang and Xie (2018) proposed an ensemble learning approach to handling imbalanced data. They balanced datasets with SMOTE, selected SVM, KNN, and Logistic Regression as the base classifier, and generated the final result by weighted voting. The approach demonstrated improvement in their experiments with six UCI datasets. A similar work is Lu and Wozniak (2019) that applied dynamic selection and weighted voting to ensemble learning with imbalanced data. Chen et al. (2020) proposed an alternative cost-sensitive learning approach and ensemble learning approach in random forest. Chakraborty (2017) combined multiple clustering and classification methods using an optimization function

called EC3 to support both binary and multi-class classification. He, however, did not directly address imbalanced learning. Huang et al. (2016) specifically focused on imbalanced learning in deep neural network. They demonstrated that enforcing tight constraints in a standard deep learning framework could reduce class imbalance in local data neighborhood.

Methodologically, Prabakaran et al. (2019) are more relevant to our work. They applied GMMs to small data with more than 300 clinical samples of breast cancer. There are four classes in their model, with a ratio of sample sizes being 40% - 39% - 19% - 2%. It is a concern that the smallest class has a sample size of less than 8, casting doubt on the estimation of Gaussian parameters. Huang et al. (2005) employed GMMs for multiple limb motion classification using continuous myoelectric signals and examined algorithmic issues such as model order selection and variance limiting in GMMS. Calo (2007) used GMMs to reduce the number of free parameters and developed a method called projection pursuit for dimensionality reduction. Stepanek et al. (2015) modified the Expectation-Maximization algorithm and applied GMMs to the task of signal separation from background in high energy physics. Ling and Zhu (2017) used a GMM-based classifier to tell whether precipitation events will happen on a certain day at a certain time from historical meteorological data. They achieved 75% accuracy, 30% precision, and 80% recall. Dixit et al. (2011) formulated a generic, topic-independent GMM known as the background GMM for generative and discriminative classification.

Fernandez et al. (2013) attempt to address the class imbalance issue in a multi-class problem. Experimentally, they proposed binarization schemes such as one-versus-one and one-versus-all, in addition to some ad-hoc procedures applied to several well-known algorithms, for example, decision trees and support vector machines.

III. DATA AND DATA PREPROCESSING

We sourced the data from Kaggle (https://www.kaggle.com/mlg-ulb/creditcardfraud). The dataset recorded 284,807 credit card transactions in September 2013 in Europe, out of which 492 are frauds. The objective is to build classification models that predict frauds well. The dataset is highly skewed and imbalanced, with the positive class (fraud) accounting for only 0.172% of all transactions.

The dataset contains 31 columns:

- Time: seconds elapsed between each transaction and the first transaction in the dataset
- Amount: transaction amount
- V1 – V28: 28 principal components obtained with PCA
- Class: response variable (1, fraud; 0, otherwise)

The dataset is very clean without missing values or outliers. Time is not included in our analysis as the time elapsed from the first transaction to the current transaction is independent of fraud. We thus have 29 features in our model: the transaction amount and 28 principal components. Feature engineering (feature expansion and selection) is irrelevant because we want to compare different models with the same features. Data is normalized to avoid scale bias.

The data was divided randomly into training and test set with a ratio of 75:25. For uniformity, the train and test sets remain the same for all the models considered.

IV. METHODOLOGIES

There are several traditional methods to handle data imbalance as aforementioned. Most of them involve data resampling one way or the other. In this study, we consider the following models for comparison.

1. Logistic regression without resampling
2. Logistic regression with oversampling minority class
3. Logistic regression with under-sampling majority class
4. GMM for classification without resampling
5. GMM for classification with oversampling minority class

While evaluating model performance, we focus on precision, recall, and F1-score since accuracy is a flawed performance metric in imbalanced learning. These metrics are directly from the confusion matrix where TP is the number of true positives, FP is the number of false positives, FN is the number of false negatives, and TN is the number of true negatives. Therefore, the numbers of elements in the actual positive class, the actual negative class, the predicted positive class, and the predicted negative class are TP + FN, FP + TN, TP + FP, and TN + FN, respectively. Precision, recall, and F1-score are then defined as follows:

- Precision = $\frac{TP}{TP+FP}$
- Recall = $\frac{TP}{TP+FN}$
- F1 = $\frac{2TP}{2TP+FP+FN}$ (the harmonic mean of precision and recall).

Precision is the ratio of true positive among all predicated positive. Higher precision means that more predicted positives are indeed true positives. Recall is the ratio of true positive among all actual positive. Higher recall indicates that more true positives are indeed predicted as positive. A model likely has high precision but low recall or vice versa. F1 score strikes a balance between precision and recall. F1 is not perfect in that it treats precision and recall equally. In other words, a false positive is considered as bad as a false negative. More often than not, one may be a worse outcome and should be weighted more than the other. For example, people would not mind as much a smoke detector going off when there is no fire as the smoke detector not sounding an alarm when there is indeed a fire. It is possible to take different weights of precision and recall into account by adopting metrics such as F_β, where β is the times that recall is as important as precision. F1 measure is a special case of F_β with $\beta = 1$. We employ F1 measure instead of F_β to provide equal footing in comparing the performance of different classification models.

4.1. Logistic Regression without Resampling

We adopt a logistic regression model without resampling as a benchmark for comparison. Although it delivers an accuracy of 0.999 on the test dataset, Table 1 below demonstrates poor performance measured by the recall because FN is high. The model incorrectly classifies many true positive elements as negative. It is particularly troublesome as the model does not catch many actual frauds.

The software generates precision, recall, F1-score, and support for each class. Support is the number of elements in each class. Micro average for each metric is computed across both classes, as defined earlier. Macro average computes each metric independently for each class and then take the average. The macro average of recall, for example, is the simple average of two recall scores, one for each class: (1.00 + 0.64)/2 = 0.82. The weighted average of, for example,

precision is the weighted average of the two precision scores weighted by the element counts in each class. The majority (in our case, negative) class dominates the dataset. Values of each performance metric under micro average, the weighted average, and the negative class are effectively equal after we rounded them to the nearest hundredth. Performance scores under macro average are therefore more meaningful here.

TABLE 1. PERFORMANCE OF LOGISTIC REGRESSION WITHOUT RESAMPLING.

	precision	recall	F1-score	support
0 (negative)	1.00	1.00	1.00	717070
1 (positive)	0.91	0.64	0.76	132
Micro avg	1.00	1.00	1.00	71202
Macro avg	0.96	0.82	0.88	71202
Weighted avg	1.00	1.00	1.00	71202

4.2. Logistic Regression with Oversampling Minority Class

In this model, we adopt the oversampling method by duplicating the minority class. In the end, the support of the minority (positive) class is roughly equal to that of the majority class. We then run logistic regression. The overall accuracy on the test set decreases to 0.950. Table 2 below shows the oversampling method's performance under other metrics: precision, recall, and F1 score.

TABLE 2. PERFORMANCE OF LOGISTIC REGRESSION WITH OVERSAMPLING.

	precision	recall	F1-score	support
0 (negative)	0.93	0.98	0.95	70996
1 (positive)	0.98	0.92	0.95	71162
Micro avg	0.95	0.95	0.95	142158
Macro avg	0.95	0.95	0.95	142158
Weighted avg	0.95	0.95	0.95	142158

As expected, the precision, recall, and F1 scores of the negative class dropped slightly. We observe, on the other hand, remarkable increases in the scores of the positive class. Overall, the macro averages of recall and F1 scores improve markedly, whereas the macro average of precision dipped by just one percentage point. The main disadvantage of oversampling is that duplicating existing examples makes overfitting more likely. Furthermore, it increases learning time.

4.3. Logistic Regression with Under-Sampling Majority Class

To implement the under-sampling method, we decided to select samples from the majority class at random such that the support of the majority class roughly matches that of the minority class. The overall accuracy deteriorated dramatically down to 0.9145. The Table below summarizes the model's performance under other metrics.

TABLE 3. PERFORMANCE OF LOGISTIC REGRESSION WITH UNDERSAMPLING.

	precision	recall	F1-score	support
0 (negative)	0.92	0.91	0.92	128
1 (positive)	0.91	0.92	0.91	118
Micro avg	0.91	0.91	0.91	246
Macro avg	0.91	0.91	0.91	246
Weighted avg	0.91	0.91	0.91	246

Table 3 shows that, compared with the oversampling method, the under-sampling approach underperforms in every category. One primary reason is that it discards many samples in the majority class that potentially contains plenty of useful information or pattern.

4.4. Classification Using Gaussian Mixture Models (GMMs)

In Gaussian generative models, we consider any data point or instance in the dataset a realization of a Gaussian random variable. Each data point is a feature vector x of dimension d generated from a Gaussian with mean μ (must also be of dimension d) and standard deviation σ. In the dataset, each feature vector is of 29 dimensions because there are 29 features. An underlying assumption is that each feature independently follows a univariate Gaussian distribution with a common standard deviation σ. The mean of each Gaussian distribution may differ. Therefore, each feature vector x follows a multivariate Gaussian distribution with the following probability density function (pdf).

$$p(x|\mu,\ \sigma^2) = \frac{1}{(2\pi\sigma^2)^{d/2}} exp\left(\frac{-\|x-\mu\|^2}{2\sigma^2}\right). \quad (1)$$

Note that equation (1) provides the likelihood of feature vector x being generated from the Gaussian distribution described above. Both x and μ are vectors of size 29 whereas σ^2, the variance, is a positive number. A Gaussian distribution represents a cluster.

All features except one in this dataset are principal components. To generate principal components, the original input features are standardized. The only non-principal-component feature, transaction amount, is also normalized. Therefore, all the features are on the same scale and their standard deviations are highly comparable. In addition, allowing different standard deviations is much more computationally expensive, if not prohibitive.

Consider a set of n data points generated from the same Gaussian distribution described by equation (1). These data points are $(x^{(1)}, \cdots, x^{(n)})$. The joint likelihood that these data points are from equation (1) is thus

$$\prod_{i=1}^{n} p\left(x^{(i)}\middle|\mu,\ \sigma^2\right) = \prod_{i=1}^{n} \frac{1}{(2\pi\sigma^2)^{d/2}} exp\left(\frac{-\|x^{(i)}-\mu\|^2}{2\sigma^2}\right). \quad (2)$$

We then apply the maximum likelihood to maximize the logarithm of equation (2) and obtain the following optimal estimates of μ and σ^2.

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x^{(i)} \quad (3)$$

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}\left(\|x^{(i)}-\mu\|^2\right)}{nd} \quad (4)$$

When the positive class and the negative class are generated from two Gaussian distributions, we can apply the formulas above to find the optimal solutions. When predicting a new data point, we compute the likelihood of this data point

generated from either Gaussian distribution. If it is more likely to be generated from the Gaussian that describes the positive class, we then predict or label this data point as positive; otherwise, we predict it as negative.

In either positive or negative class, more than one clusters are likely to exist. Suppose there exist K clusters, and a multinomial distribution captures the probability that a data point is from one of these K clusters. Each cluster is then described by a Gaussian distribution, characterized by Equations (1) – (4). We end up with a Gaussian mixture model (GMM). In the GMM, there are $3K$ parameters: $p_1, \cdots, p_K; \mu^{(1)}, \cdots, \mu^{(K)}; \sigma_1^2, \cdots, \sigma_K^2$, where p_i is the probability of a data point generated from the i^{th} cluster or Gaussian distribution, and $\mu^{(i)}$ and σ_i^2 are the mean and variance of the i^{th} Gaussian distribution, respectively.

Suppose, once again, there are n data points: $X = (x^{(1)}, \cdots, x^{(n)})$. Let vector θ denote all the $3K$ parameters. The joint likelihood of these n data points generated from the K Gaussians are as follows:

$$p(X|\theta) = \prod_{i=1}^{n} \sum_{j=1}^{K} p_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I) \quad (5)$$

where $N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I)$ represents the probability density function of the j^{th} Gaussian. It is, however, impossible to find the optimal solution simultaneously to all the $3K$ parameters. EM (Expectation-Maximization) algorithm can computationally find a local optimum. To implement the algorithm, we begin with random initialization of θ. Then in the E-step, we obtain the posterior probability that data point i belongs to cluster j:

$$p(j|i) = p_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I) / p(x^{(i)}|\theta). \quad (6)$$

In the M-step, we maximize the likelihood to find the following:

$$n_j = \sum_{i=1}^{n} p(j|i) \quad (7)$$

$$\hat{p}_j = \frac{n_j}{n} \quad (8)$$

$$\hat{\mu}^{(j)} = \frac{1}{n_j} \sum_{i=1}^{n} p(j|i) x^{(i)} \quad (9)$$

$$\hat{\sigma}_j^2 = \frac{\sum_{i=1}^{n} \left(p(j|i) \| x^{(i)} - \mu \|^2 \right)}{n_j d} \quad (10)$$

The E-step and M-step are repeated until the algorithm converges. We tried different initializations to identify a better local optimum.

Once we identify the clusters in each class, we apply them to classify a new data point. Compute the sum of probability densities that this data point belongs to each cluster in positive and negative class, respectively. We label the data point positive if the sum of probability densities from positive clusters is higher than that from negative clusters, vice versa. We have tried different cluster numbers: K =1, K = 3, and K = 6. In our studies, K = 1 outperformed the other two.

Application of GMM to classification is a form of under-sampling of majority class (negative class in our case). As observed in 4.3, under-sampling may potentially discard useful information in the dataset. We thus consider a combination of GMM and oversampling. We first oversample the minority class as done in 4.2. Then apply GMM to each class.

4.4.1. Classification Using Gaussian Mixture Models (GMMs) without Resampling

In this subsection, we consider GMM for classification without data resampling. We start with training one cluster or Gaussian distribution for each class. The accuracy is 0.967. Table 4 below summarizes the performance results for this approach.

TABLE 4. PERFORMANCE OF GMM WITH 1 CLUSTER PER CLASS.

	precision	recall	F1-score	support
0 (negative)	1.00	0.97	0.98	717070
1 (positive)	0.05	0.86	0.09	132
Micro avg	0.97	0.97	0.97	71202
Macro avg	0.52	0.91	0.54	71202
Weighted avg	1.00	0.97	0.98	71202

The performance is not ideal. In particular, the macro averages of precision and f1-score are 0.52 and 0.54, respectively. That is, false positives are high.

One possibility is that one cluster per class is not sufficient to capture each class's distinctive underlying characteristics. We look into an appropriate number of clusters for each class. Note that we use the sum of probability densities for classification. It is necessary to keep the number of clusters in each class the same. We understand that the "optimal" number of clusters in each class may differ. We decide not to explore this for two reasons. It requires using a different criterion for classification instead of the sum of probability densities. It is not apparent what criterion serves our purpose. Also, our study shows that multiple clusters have similar performance compared with a single cluster.

We use BIC (Bayesian Information Criterion), one of the most popular criteria for choosing hyperparameter (the number of clusters in our case), to identify the "optimal" number of clusters. The analysis shows that 3-cluster and 6-cluster are two better choices. The performance results of GMM with 3-cluster and 6-cluster are summarized in Table 5 and 6 below,

TABLE 5. PERFORMANCE OF GMM WITH 3 CLUSTERS PER CLASS.

	precision	recall	F1-score	support
0 (negative)	1.00	0.97	0.98	717070
1 (positive)	0.04	0.85	0.09	132
Micro avg	0.97	0.97	0.97	71202
Macro avg	0.52	0.91	0.53	71202
Weighted avg	1.00	0.97	0.98	71202

TABLE 6. PERFORMANCE OF GMM WITH 6 CLUSTERS PER CLASS.

	precision	recall	F1-score	support
0 (negative)	1.00	0.99	1.00	717070
1 (positive)	0.15	0.83	0.26	132
Micro avg	0.99	0.99	0.99	71202
Macro avg	0.58	0.91	0.63	71202
Weighted avg	1.00	0.99	0.99	71202

The respective accuracies of GMM with 3 and 6 clusters are 0.966 and 0.991. Overall, the GMM with 6 clusters slightly outperforms those with 1 or 3 clusters. But it pays a price of relatively low recall.

GMM for classification is essentially under-sampling the data because the same number of parameters are estimated with a vastly different number of data points in each class. Consequently, we do not consider the

combination of GMM with the under-sampling technique. Instead, we investigate the combinations of GMM with the oversampling approach.

4.4.2. Classification Using Gaussian Mixture Models (GMMs) with Over-Sampling

To offer a fair comparison, we use the same training and test sets generated by the over-sampling process described in Section 4.2. We begin with 1-cluster GMM. And then use BIC to select the "optimal" number of clusters. It turns out that 3-cluster and 6-cluster once again are two better choices. The accuracies of 1-cluster, 3-cluster, and 6-cluster GMMs are 0.931, 0.917, and 0.917. Tables 7-9 summarize the performance results of those three GMMs.

The three GMMs render very similar performance results, with 1-cluster GMM being slightly better. The original training dataset has only 132 positive data points. After they have been duplicated many times in the oversampling process, having more clusters in positive class is more likely to catch some noise as signal. It potentially leads to overfitting, thus underperforming on the test set.

TABLE 7. PERFORMANCE OF GMM WITH 1 CLUSTER PER CLASS.

	precision	recall	F1-score	support
0 (negative)	0.90	0.97	0.93	70996
1 (positive)	0.96	0.90	0.93	71162
accuracy	NA	NA	0.93	142158
Macro avg	0.93	0.93	0.93	142158
Weighted avg	0.93	0.93	0.93	142158

TABLE 8. PERFORMANCE OF GMM WITH 3 CLUSTERS PER CLASS.

	precision	recall	F1-score	support
0 (negative)	0.88	0.96	0.92	70996
1 (positive)	0.96	0.87	0.91	71162
accuracy	NA	NA	0.92	142158
Macro avg	0.92	0.92	0.92	142158
Weighted avg	0.92	0.92	0.92	142158

TABLE 9. PERFORMANCE OF GMM WITH 6 CLUSTERS PER CLASS.

	precision	recall	F1-score	support
0 (negative)	0.88	0.96	0.92	70996
1 (positive)	0.96	0.87	0.91	71162
accuracy	NA	NA	0.92	142158
Macro avg	0.92	0.92	0.92	142158
Weighted avg	0.92	0.92	0.92	142158

GMMs with over-sampling far outperform GMMs without resampling. It is not a surprise as over-sampling, to a large degree, makes up for the inherent under-sampling effect of GMM. The performance results are highly comparable with those of logistic regression with oversampling. GMMs for classification, however, are much easier to generalize to multi-class classification problems.

V. A FRAMEWORK FOR IMBLANCED LEARNING AND POSITIONING OF THE PAPER

We propose a framework of six aspects in helping scholars and practitioners determine effective approaches for handling

imbalanced learning problems. These aspects are (1) data preprocessing; (2) feature engineering; (3) resampling choice; (4) model and algorithm choice; (5) ensemble methods; and (6) performance metrics choice.

In data preprocessing, the choices available include, but not limited to, data standardization and normalization, principal components, and independent components. Our paper adopts principal components and data standardization. In feature engineering, feature expansion with or without kernel tricks is essential, along with feature selection that follows. Our paper does not utilize feature engineering. The choice of resampling methods is always a critical decision in an imbalanced learning problem. We consider both oversampling and under-sampling techniques. Depending on each problem's nature, scholars and practitioners must also decide the models and corresponding algorithms that will be used. Our paper selects logistic regression and GMM models. Ensemble methods are proved to be effective. A combination of multiple models tends to outperform each model. Our paper does not consider ensemble methods. The choice of different performance metrics can also be worthwhile in an imbalanced learning problem. Our paper chooses precision, recall, and f1-score as performance metrics. Our paper is uniquely positioned at the conjunction of principal components, resampling, and GMM for classification in handling a severe data imbalance.

Not only does this framework help define a repertoire of tools for combating imbalanced learning problems, but it also guides for improving model performance. Exceptional performance on the selected metrics is not a concern of our paper. If we were to improve the performance, we could explore alternatives in each of those six aspects, particularly the ones we did not consider. First, featuring engineering is worth a try. Techniques like kernel methods are shown to be effective. Second, we only considered logistic regression and GMM models. There are plenty of other classification and clustering models to consider. Additionally, an ensemble of some of these models will likely outperform our current choices. Last but not least, false negative is more severe a problem in credit card fraud detection. Thus, it helps to consider some cost-sensitive methods to penalize misclassification of actual fraud more severely.

VI. CONCLUSIONS

This paper uses a highly imbalanced credit card fraud dataset and investigates the effectiveness of combining classification and clustering algorithms with resampling techniques on handling imbalanced learning. One algorithm is logistic regression, a classic classification method. The other, GMM, is a sophisticated clustering method applied to classification. Our study indicates that exploring combinations of different algorithms and techniques is a reasonable and probably necessary approach to combat severe data imbalance in classification problems. When combined with the oversampling method, both logistic regress and GMM render rather satisfying performance results. Oversampling tends to outperform under-sampling.

We also propose a framework of six aspects for selecting an effective approach to handle imbalanced learning problems and for improving model performance. To improve performance on credit card fraud detection, additional considerations include, but not limited to, feature engineering, various classification and clustering models,

ensemble methods, and heavier penalty on false negative.

Deep learning techniques are, in some sense, overtaking the field of machine learning. It, however, usually demands a vast dataset. With the capabilities of deep neural networks increasing day by day, it would be fascinating to investigate the effectiveness of deep learning in dealing with imbalanced learning. We look forward to it.

REFERENCES

Calo, D. G., "Gaussian Mixture Model Classification: A Projection Pursuit Approach", *Computational Statistics & Data Analysis*, 52(1), 2007, 471-482.

Chakraborty, T., "EC3: Combining Clustering and Classification for Ensemble Learning", arXiv: 1708.08591, 2017.

Chen, C., Liaw, A., Breiman, L., "Using Random Forest to Learn Imbalanced Data," https://statistics.berkeley.edu/sites/default/files/tech-reports/666.pdf (accessed October 1, 2020)

Dixit, M., Rasiwasia, N., Vasconcelos, N., "Adapted Gaussian Models for Image Classification", *IEEE Conference on Computer Vision and Pattern Recognition Proceedings*, 2011.

Fernandez, A., Lopez, V., Galar, M., Jose del Jesus, M., Herrera, F., "Analysing the Classification of Imbalanced Datasets with Multiple Classes: Binarization Techniques and ad-hoc Approaches", *Knowledge-Based System*, 42, 2013, 97-110.

He, H., Garcia, E. A., "Learning from Imbalanced Data", *IEEE transactions on Knowledge and Data Engineering*, 21(9), 2009, 1263–1284.

Huang, C., Li, Y., Loy, C. C., Tang, X., "Learning Deep Representation for Imbalanced Classification", *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, 18-24

Huang, Y., Englehart, K. B., Hudgins, B., Chan, A. D. C., "A Gaussian Mixture Model Based Classification Scheme for Myoelectric Control of Powered Upper Limb Prostheses", *IEEE Transactions on Biomedical Engineering*, 52(11), 2005, 1801-1811.

Krawczyk, B., "Learning from Imbalanced Data: Open Challenges and Future Directions", *Progress in Artificial Intelligence*, 5, 2016, 221-232.

Ling, H., Zhu, K., "Predicting Precipitation Events Using Gaussian Mixture Model", *Journal of Data Analysis and Information Processing*, 81(1), 2018, 1476-1483.

Lu, L., Wozniak, M., "Imbalanced Data Classification Using Weighted Voting Ensemble", *International Conference on Image Processing and Communications*, 2019, 5375-5384.

Maheshwari, S., Jain, R. C., Jadon, R. S., "A Review on Class Imbalance Problem: Analysis and Potential Solutions", 172, *International Journal of Computer Science Issues*, 14(6), 2017, 43-51.

Prabakaran, I., Wu, Z., Lee, C., Tong, B., Steeman, S., Koo, G., Zhang, P. J., Guvakova, M. A., "Gaussian Mixture Models for Probabilistic Classification of Breast Cancer", *Cancer Research*, 79(13), 2019, 3492-3502.

Rastogi, A. K., Narang, N., Siddiqui, Z. A., "Imbalanced Big Data Classification: A Distributed Implementation of SMOTE", *Proceedings of the Workshop Program of the 19th International Conference on Distributed Computing and*

Networking, Article No. 14, 2018, page 1-6.

Stepanek, M., Franc, J., Kus, V., "Modification of Gaussian Mixture Models for Data Classification in High Energy Physics", *Journal of Physics: Conference Series*, 574, 2015, 012150.

Sun, Y., Wong, A., Kamel, M. S., "Classification of Imbalanced Data: A Review," *International Journal of Pattern Recognition and Artificial Intelligence*, 23(4), 2009, 687-719.

Wu, C., Du, H., "Catch Shooting Stars – Predict NBA All Stars from Rookie Performance," *Journal of Supply Chain and Operations Management*, 15(1), 2017, 34-54.

Wu, F., Jing, X., Shan, S., Zuo, W., Yang, J., "Multiset Feature Learning for Highly Imbalanced Data Classification", *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 2017, 1583-1589.

Xiang, Y., Xie, Y., "Imbalanced Data Classification Method Based on Ensemble Learning", *International Conference in Communications, Signal Processing, and Systems*, 2018, 18-24.