# Impact of Class Imbalance on Unsupervised Label Generation for Medicare Fraud Detection

Robert K.L. Kennedy and Taghi M. Khoshgoftaar
Florida Atlantic University
Boca Raton, Florida 33431
rkennedy@fau.edu, khoshgof@fau.edu

***Abstract*—In this work, we use an unsupervised method for generating binary class labels in a novel context to create class labels for Medicare fraud detection. We examine how class imbalance influences the quality of these new labels and how it affects supervised classification. We use four different Medicare Part D fraud detection datasets, with the largest containing over 5 million instances. The other three datasets are sampled from the original dataset. Using Random Under-Sampling (RUS), we subsample from the majority class of the original data to produce three datasets with varying levels of class imbalance. To evaluate the performance of the newly created labels, we train a supervised classifier and evaluate its classification performance and compare it to an unsupervised anomaly detection method as a baseline. Our empirical findings indicate that the generated class labels are of high enough quality and enable effective supervised classifier training for fraud detection. Additionally, supervised classification with the new labels consistently outperforms the baseline used for comparison across all test scenarios. Furthermore, we observe an inverse relationship between class imbalance in the dataset and classifier performance, with AUPRC scores improving as the training dataset becomes more balanced. This work not only validates the efficacy of the synthesized class labels in labeling Medicare fraud but also shows its robustness across different degrees of class imbalance.**

***Index Terms*—class labeling, unsupervised learning, class imbalance, Medicare fraud, machine learning**

## I. Introduction

Fraud detection is a critical task across various real-world scenarios such as credit card transactions, identity theft, and Medicare fraud. It involves identifying fraudulent activities amid a multitude of legitimate activities. This process represents a form of anomaly detection, which identifies events that occur infrequently and bear similarities to the rest of the data. Fraud detection datasets, the type of dataset used in this work, often suffer from a lack of consistent and accurate labels. Due to the nature of anomalies, these datasets are often class imbalanced or highly class imbalanced. Put another way, the labeled examples of one class are significantly outnumbered by the other class.

Data labeling is a resource-intensive and error-prone task that is a critical component to the effective use of machine learning models [1]. Labeling datasets can be cost prohibitive and can often have noise or inaccurately labeled instances. These low-quality labels can drastically reduce a machine learning models' performance [2]. Furthermore, a large proportion of newly generated datasets are unlabeled by default [3]. This presents both an opportunity and a challenge to the machine learning community. While large datasets typically produce better performing models than smaller datasets [4], the absence of labels undermines this performance boost. This is especially evident in privacy-sensitive or expert-dependent domains like medical imaging or fraud detection. Consequently, unsupervised learning, a paradigm that does not use class labels, is appealing. However, unsupervised models typically do not perform as well as their supervised counterparts when labels are present [5], highlighting a gap in machine learning research of data labeling.

Class imbalance presents a significant challenge in machine learning, particularly in binary classification problems like the one we focus on in this paper, where the minority class is vastly outnumbered by the majority class. Though we focus on the binary problem, all the concepts can be extended to the multi-class problem via class decomposition [6]. A dataset is typically considered imbalanced if the ratio between the classes exceeds 1:4 [7] and is considered to have high class imbalance when there is an imbalance ratio of 1:1000 or [8], [9]. Such extreme imbalances between the class representation in the datasets add complexity and requires tailored strategies for effective model training and classification. Fraud detection datasets are often plagued by class imbalance since fraudulent examples, by nature, are few and far between and are often the class of interest [10]–[12]. Our research aims to improve upon the existing research of methodologies and strategies designed to address the challenges of both class imbalance and unlabeled data in the context of fraud detection.

In this paper, we evaluate an unsupervised method for generating binary class labels for Medicare fraud detection and examine how varying levels of class imbalance affect the labeling method. Importantly, our work focuses on the scenario where fraudulent data is unlabeled. In such scenarios, to identify fraudulent data, class labels need to be manually collected, which can be costly and error prone, or unsupervised anomaly detection methods can be used. We employ an automated class labeling technique to label fraudulent Medicare data, train a supervised classifier on the new labels, and measure its classification performance. We compare its results to an unsupervised anomaly detection method as a baseline comparison. Our method uses an artificial neural network to learn from the dataset's features. Based on an error that is calculated for each instance, detailed in Section III, every instance is labeled as either fraudulent or non-fraudulent. Our empirical findings show that our labeling technique significantly enhances data quality for supervised learning. Models built with the data labeled with our technique outperform an unsupervised model used as a baseline for comparison. Additionally, our results show that the classification performance increases as the class imbalance ratio decreases when training on data labeled with our novel technique.

The remainder of this paper is organized as follows. Section II provides a review of related works and highlights gaps in the existing research where our research fits in. In Section III, we discuss the label synthesis methodology and the method in which we measure the performance of the labeling technique. Section IV presents and discusses the experimental results. Section V concludes the work and provides areas for potential future work.

## II. Related Work

After reviewing the relevant existing literature, it becomes evident that our automated class labeling technique for highly imbalanced big data represents novel research for labeling big fraud detection datasets. Given the unique nature of our research, to the best of our knowledge, there are no existing methodologies or comparable studies currently available.

Baek et. al [13] aim to create a method for detecting network anomalies in a supervised manner without requiring extensive analysis of network traffic by human experts. Their strategy involves three main steps. Initially they use a clustering technique to assign labels to the training data. Next, they train supervised models with these estimated labels. Finally, they employ these models to differentiate between anomalous and non-anomalous network behaviors. The first step used K-means clustering to group the data. Then the labeling of the clusters is based on two criteria. First a cluster is anomalous if it is small or sparse. The second criterion, which aimed to improve the performance, was based on their observations of the data after extensive investigation. Many network attacks exhibit similar patterns, which form dense clusters. This insight allowed them to improve the performance of the supervised models trained on the labels but required extensive expert human involvement. This is in contrast with our approach in that theirs relies heavily on expertise and a thorough investigation of the datasets with prior knowledge. This makes it unable to find new anomalous patterns that may manifest in the future without additional human involvement. Our approach is more automated and would not be susceptible to such a scenario.

Moslehi et al. [14] also use K-means clustering to refine a label assignment. They acknowledge the impracticality of labeling the entire dataset and instead propose assigning labels to a representative subset based on the generated clusters. Their method involves matching the cluster statistics from their dataset with those from a secondary, previously labeled dataset, effectively using it as a reference to assign labels. This approach significantly differs from ours, as it relies on existing labels from an external source rather than synthesizing new class labels without any provided class labels as a starting basis. Further, their dataset only contains 385 samples, making it significantly smaller than all the datasets we use in our paper, by a factor of at least 48.

Maqbool et. al [15] introduce an automated method for labeling clusters by leveraging keyword identifiers in their dataset. They use two ranking schemes, frequency and inverse frequency of keywords. This prioritizes the keywords for labeling, underscoring the importance of automation in enhancing cluster interpretability. In [16], Rauber presents a labeling technique of self-organizing maps applied to small text datasets. Their cluster labels were derived from the words in the dataset features. Our approach differs from theirs in that our dataset lacks such keywords for labeling. Instead, we use an autoencoder that only processes numeric features to calculate an error statistic. This error statistic is then used to generate binary class labels for each instance.

These related works differ from our approach in four important ways. First, the reviewed clustering methods do not address highly imbalanced datasets, and the studies often used balanced data, such as in [13]. Second, these methods are applied to a significantly smaller dataset than ours, with examples like the 385-sample dataset used in [14]. Third, our method does not derive class labels from dataset features, which contrasts with the keyword-based labeling methods seen in [15] and [16]. Fourth, and most importantly, our method is automated, eliminating the need for the extensive human involvement, making it suitable for big data and highly imbalance datasets, in contrast to [13]–[15].

## III. Methodology

Our methodology presents a new approach for synthesizing binary class labels for fraud detection data, specifically designed for datasets that are characterized by having high class imbalance [17]. In this paper, we test and evaluate its efficacy on fraud detection datasets with varying levels of class imbalance. The labeling methodology uses an autoencoder [18] to learn from the features in the datasets. Once the autoencoder is fully trained, a reconstruction error is produced, specifically the mean squared error, for all instances in a provided dataset. The instances are then sorted, by error, from greatest to least. The underlying premise is that instances with higher reconstruction errors are more likely to belong to the minority class, whereas those with lower reconstruction errors tend to be part of the majority class. Once sorted, the instances are arranged in such a way that the likelihood of belonging to the minority class decreases from the top to the bottom.

An error threshold needs to be established such that the instances with errors above this threshold are classified as positive, and those below as negative. In the context of our datasets, positive instances belong to the minority or fraudulent class, while negative instances belong to the majority or non-fraudulent class. Initially, we set this threshold based on domain knowledge and an understanding of the data's population characteristics. We label a subset of instances just below the threshold as positive. This has been shown to enhance model generalization and performance in highly imbalanced datasets

[17]. Moreover, instances around the threshold are regarded as potentially noisy. An alternative to labeling these uncertain instances as positive would be to exclude them entirely from the data set, potentially losing valuable information. Therefore, to preserve as much information as possible, we choose to label these uncertain instances as positive. For a given dataset, the total number of instances labeled as positive is denoted as *P*. We test our methodology across various *P* values for multiple datasets with various levels of class imbalance.

We examine a wide range of *P* values that represent fewer and greater numbers of positive-labeled instances than the dataset population. Where applicable, we limit the *P* value so that we do not create a dataset where the number of fraudulent-labeled instances is greater than the number of non-fraudulent. Increasing *P* beyond this point would start to resemble how one-class classifiers are trained [19]. One-class classifiers are typically trained on only the majority class and are out of scope of this paper. Our findings indicate that labeling more instances as positive, even those with a low probability of being in the positive class, or increasing *P*, can boost supervised learning classification performance.

## A. Experimental Datasets

We utilize Medicare Part D data spanning from 2013 through 2019. Part D data contains Medicare claims data describing a medical provider's prescription drug activity, a given year and drug name [20]. Annually, the Centers for Medicare and Medicaid Services (CMS) releases one year's worth of data [21]. However, there is a delay between the end of the year and when CMS releases data. The Part D dataset contains approximately 172 million records, making it one of the more extensive Medicare datasets released by CMS. The features include provider specific details, such as their National Provider Identifier (NPI), medical specialty, gender, geographical information, and claims-level data like the number of beneficiaries per drug, cost, and how many prescriptions are written per NPI.

Initially, the raw Part D data is cleaned and processed. This involves merging annual data, normalizing columns, filling in missing values, and removing duplicates. Missing values are handled using CMS-provided dictionary files, as well as specific rules for imputing or excluding data based on its nature. For example, the column named Tot Bene, or total beneficiaries for a given drug per NPI, has missing rows if the number is less than 10. A median value of 5 is inputed for these missing values. Subsequent data aggregation steps reduced the number of instances in the dataset to approximately 6 million rows with 31 features. This process involved consolidating data over NPI, year, and provider type, and calculating summary statistics for numerical attributes. This reduction in size reduces the computational and storage requirements. It also helps with the challenges of its highly class imbalanced nature [22], [23].

The preprocessed Part D data was then enriched with another CMS dataset, the Medicare Part D summary by Provider, also spanning from 2013 through 2019. This enrichment step added 51 new features by combining the data on NPI and year. All of the categorical features were one-hot encoded. This resulted in a final dataset with 328 features. This enriched and preprocessed Medicare Part D data forms the basis for our experiments and is the largest of the Part D datasets used, called Part D Full. When we train the autoencoder on this dataset, and any of the randomly undersampled datasets described below, we scale all the numeric features before autoencoder training. Once we label each instance, the features are reverted to the original non-scaled version, to avoid any potential data leakage due to scaling the entire dataset at once.

TABLE I
DATASET CLASS CHARACTERISTICS

| Dataset | Minority Count | Majority Count | Total Count | Minority Imbalance |
|---|---|---|---|---|
| Part D Full | 3,700 | 5,340,406 | 5,344,106 | 0.0692% |
| Part D RUS-1 | 3,700 | 366,300 | 370,000 | 1% |
| Part D RUS-5 | 3,700 | 70,300 | 74,000 | 5% |
| Part D RUS-20 | 3,700 | 14,800 | 18,500 | 20% |

To effectively examine the effects of class imbalance on the synthesized class labels, we generated three additional, smaller, Part D datasets with varying levels of class imbalance. The dataset characteristics, including class counts, for all datasets used are shown in Table I. We used random undersampling (RUS) to randomly undersample from the majority class of the Part D Full dataset. RUS is a widely used and effective machine learning technique to reduce the class imbalance in a given dataset. We apply RUS to generate three additional datasets with a class imbalance of 1%, 5%, and 20% minority, namely, RUS-1, RUS-5, and RUS-20, respectively. Additionally, we run RUS ten times for each of those dataset sizes. The performance results shown in Table II and III are the average across the 10 different datasets. We repeat RUS multiple times so we can get high quality, representative under-sampled results. This approach prevents the possibility of obtaining a dataset split that may not be representative of the overall distribution.

Autoencoders typically perform better with scaled features largely due to their architectural design and the gradient descent optimization used for training [18]. When features are scaled, they contribute equally to the error gradient, enhancing the model's convergent speed instability during training. In contrast, unscaled features with widely varying minimum and maximum values can disrupt training. The learning algorithm might focus on minimizing errors in features with larger scales at the expense of those with smaller scales leading to sub-optimal performance. Therefore, we scale the numeric features in our datasets while generating new class labels to optimize the training of the auto coder. However, to prevent any potential data leakage in subsequent steps between the training and test data splits, we reverse the scaling after all new labels were generated and before proceeding to use supervised classification to measure labeling performance.

## B. Measuring Performance

Our work focuses on the unlabeled scenario where it is only possible to use unsupervised learning. To use supervised classification with a given unlabeled dataset, class labels must be manually collected or otherwise created. Our methodology generates class labels, using only the dataset features, for subsequent use in supervised learning. To assess the impact of class imbalance on the synthesized class labels, we generate new class labels for each dataset across various levels of *P*. *P* is the total number of positive-labeled instances. Although as *P* varies, the number of positive instances changes, the total number of instances does not change. Only the distribution between the two classes changes. Throughout this labeling process no instance is removed. We then train a supervised machine learning classifier using these newly created labels and evaluate its classification performance. The classification performance is measured using the ground truth labels in the original data set. It is important to note that we use the generated class labels with supervised learners solely to evaluate the performance of the new labels and the label generation method. We exclude supervised learning with original labels from our study, as our focus is on the unlabeled scenario, making the use of the original labeled data for supervised learning beyond the scope of our work.

We use a Decision Tree (DT) as our supervised classifier when evaluating the generated labels. DTs are a popular and well-known supervised learning method that can be utilized for both classification and regression tasks. They are structured hierarchically, comprising of a root node, branches, internal nodes, and leaf nodes. The internal nodes serve as decision points based on data attributes. Leaf nodes are the DT's final output that can output either a numeric value for regression tasks, or a class label prediction for classification tasks. The DT classifier trains using a divide-and-conquer approach, where the algorithm uses a greedy search to identify optimal split points in the training dataset. When training on a binary labeled dataset, this splitting process is done recursively until all instances are categorized in either of the two classes. The depth or size of the trained DT model has an impact on its performance. Often, the leaves of the tree are pure nodes, meaning that all data within a node is in a single class.

We train and evaluate the DT using five-fold cross validation. The classifier is trained on data labeled with the technique described above and uses the original class labels in the test folds. We perform one round of five-fold cross validation on each dataset and repeat this for ten rounds. The random undersampling is included in each round. I.e., for each of the ten rounds, each randomly undersampled dataset consists of a different randomly selected set of instances, while maintaining the same imbalance ratio. The performance results shown in the tables are the average value across the ten iterations of five-fold cross validation for each of the full Part D, RUS-1, RUS-5, and RUS-20 datasets.

The classification performance of the DT, when trained on the new labels, serves as an effective indicator of the labels' overall quality and utility, reflecting their potential real-world application. Datasets are often completely unlabeled. Thus, our approach would be employed to generate labels in an unsupervised fashion for use in supervised learning. The performance metrics of the DTs trained our experimental datasets show expected results on new, unseen fraud detection datasets of varying class imbalances. In real-world situations where datasets lack labels, one can either use an unsupervised approach or manually collect class labels, which has its significant drawbacks, or use an automated labeling approach such as the one used in this paper. Thus, it is essential to benchmark our labeling technique against a baseline unsupervised model. By outperforming this unsupervised baseline, we demonstrate the practicality and superiority of our method.

We employ the area under the precision-recall curve (AUPRC) as our primary classification performance metric. The AUPRC values are calculated based on the true positive (TP), false positive (FP), false negative (FN), and true negative (TN) counts from a standard confusion matrix. This metric effectively captures the balance between precision and recall. They are defined as: $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$. AUPRC was selected as our key metric because it is an optimal metric when evaluating classification performance in the presence of class imbalance [24], [25].

## C. Baseline Comparison

We train the DT on the class labels synthesized by our method. This evaluates both the effectiveness and quality of the labels. In the unlabeled scenario, it is possible to use an unsupervised anomaly detection method to directly use the unlabeled data. Therefore, a performance comparison between the classification results obtained from our synthesized labels and those from an unsupervised anomaly detection approach is essential. To accurately demonstrate the advantage to our labeling technique, it must outperform this baseline. For this purpose, we utilize the Isolation Forest (IF) method, initially introduced by Liu et al. in [26], as the baseline for comparison.

IF is a popular unsupervised method for anomaly detection. This tree-based algorithm works by isolating anomalies from the dataset through recursively partitioning the data into smaller subsets. The core idea behind IF, as described in [26], is that anomalies are "few and different", making them more likely to be isolated earlier in the tree building process than normal, non-anomalous instances. Anomalies are identified by the IF algorithm when an instance has consistently shorter paths, across multiple trees in the forest. We train the IF algorithm using five-fold cross validation on each of the four datasets. We repeat this ten times for each dataset, and the performance metrics shown in Table II are the average across the five folds and ten repetitions. Similar to how DT is used, the original class labels are only used in the test folds to calculate classification performance. If the DT trained on synthesized labels yields higher AUPRC scores than IF, trained only on features, we can conclude our labeling method is better than using IF and is able to produce robust classifier training. Our results show that in the unsupervised scenario, when class labels are not available, when it is possible to use IF with no labels or the binary classifier with synthetic labels, our methodology outperforms.

### D. Implementation Details

Using the implementations in Scikit-learn [27], version 1.3.0, we define and train DT, as well as IF. We use the library defaults for DT. According to the original design [26], there are only two available parameters for IF: the number of trees in the forest and the contamination rate. We set both parameters to the library defaults. However, during experimentation, we observed that the contamination rate did not have any effect on threshold-based performance metrics, such as AUPRC.

For the autoencoder [18], which is used for the labeling process, we use Keras [28], version 2.8.0. The autoencoder contains an encoder section that is connected to a decoder section. The encoder section first has a 100-neuron layer fully connected to the second layer containing 50 neurons. Both layers use the ReLu activation function. These are then connected to the decoder section, which first has 50 neurons fully connected to 100 neurons. These layers use the Tanh activation function. The final output layer of the autoencoder uses ReLu. We train using a learning rate of 0.0001, a mini-batch size of 256, and MSE as the loss function. We use the Adam optimizer function and set the implementation to use a 20% validation set size for a maximum of 250 epochs with EarlyStopping set to 25 epochs.

## IV. Experimental Results

We apply the unsupervised labeling method to four datasets, Part D Full, Part D RUS-1, Part D RUS-5, and Part D RUS-20. We train a DT on them and generate an AUPRC score using the original class labels, only to calculate the performance metric. We train IF on the same features used by the DT, and only use the class labels for performance metric calculations. Additionally, we vary P, the number of positive-labeled instances. For the full dataset, RUS-1, and RUS-5 we vary *P* from 1,000 through 15,000 in increments of 1,000. As described in previous sections, we limit the number of positive-labeled instances for the more balanced dataset, RUS-20. We vary *P* from 1,000 to 8,000 in increments of 500. The number of positives does not affect the IF results since it does not train using the synthesized labels.

TABLE II
Baseline IF Results

| Dataset | AUPRC |
|---|---|
| Part D Full | 0.00220 |
| Part D RUS-1 | 0.02331 |
| Part D RUS-5 | 0.09169 |
| Part D RUS-20 | 0.24614 |

For the Part D Full dataset, IF achieved a baseline AUPRC of 0.00220. This is drastically lower than DT for all levels of *P*. The lowest AUPRC score of 0.02908, on the full dataset, occurs when there are 1,000 labeled instances, as can be seen in Table III. This is a factor of roughly 13. For all the experiments completed with the full Part D datasets, the synthesized labels produce a DT that is more performant. Further, as *P* increases, the AUPRC score for DT increases as well. It reaches a maximum value of 0.05518 when training on 14,000 positive labeled instances. This shows that for all levels of *P* on the full Part D dataset, the DT trained on the synthesized labels outperforms the baseline IF. A similar pattern can be seen when applying the methodology to the RUS-1 dataset. RUS-1 starts with a class imbalance of 1% as opposed to 0.0692% of the full Part D dataset, as can be seen in Table I. The AUPRC score for the Decision Tree (DT) exceeds that of the Isolation Forest (IF) across all values of *P*, reaching 0.22464 with 8,000 positive instances compared to IF's score of 0.02331. Although the highest performance typically occurs when *P* is in the middle of its range, the trend of increasing AUPRC scores with larger values of *P* is less pronounced when using the entire dataset. The AUPRC fluctuates as *P* increases. However, at all *P* values, DT outperforms IF.

TABLE III
DT AUPRC Part D Results

| *P* | Part D Full | RUS-1 | RUS-5 | *P* | RUS-20 |
|---|---|---|---|---|---|
| 1000 | 0.02908 | 0.17756 | **0.35693** | 1000 | 0.49003 |
| 2000 | 0.02927 | 0.21253 | 0.33945 | 1500 | **0.49801** |
| 3000 | 0.03535 | 0.21268 | 0.33984 | 2000 | 0.44806 |
| 4000 | 0.03358 | 0.21166 | 0.28898 | 2500 | 0.45728 |
| 5000 | 0.03629 | 0.18541 | 0.30183 | 3000 | 0.42995 |
| 6000 | 0.03500 | 0.19101 | 0.29194 | 3500 | 0.42710 |
| 7000 | 0.04391 | 0.17844 | 0.28226 | 4000 | 0.42960 |
| 8000 | 0.04686 | **0.22464** | 0.31228 | 4500 | 0.45307 |
| 9000 | 0.04142 | 0.20159 | 0.29056 | 5000 | 0.41837 |
| 10000 | 0.04442 | 0.20581 | 0.28365 | 5500 | 0.40985 |
| 11000 | 0.05060 | 0.20412 | 0.29115 | 6000 | 0.41410 |
| 12000 | 0.05062 | 0.21676 | 0.30118 | 6500 | 0.43164 |
| 13000 | 0.05445 | 0.20915 | 0.30005 | 7000 | 0.41863 |
| 14000 | **0.05518** | 0.19346 | 0.30562 | 7500 | 0.44780 |
| 15000 | 0.05452 | 0.20328 | 0.30536 | 8000 | 0.44047 |

The results of RUS-5 and RUS-1, which have 5% and 1% minority respectively, are more similar to each other than they are to the full datasets results. Specifically, the AUPRC value fluctuates as *P* increases, though in this case, the highest AUPRC score occurs when *P* is 1,000. Like the previous datasets, however, the AUPRC score for every *P* value outperforms that of IF. The last dataset examined, RUS-20, has a 20% class imbalance, and the results are more similar to RUS-5 than the others. In this case we increase *P* by a smaller amount since the stopping point for *P* is lower than the others due to it being a more balanced dataset with fewer instances. With this level of imbalance, we observe that the highest AUPRC of 0.49801 occurs early on when increasing *P*, as can be seen in Table III. Again, with this imbalance ratio, the AUPRC score for DT is higher than IF for all values of *P*.

DT outperforms IF when measuring AUPRC for all values of *P* and for all levels of class imbalance as shown when using the full dataset, RUS-1, RUS-5, and RUS-20. This shows that the synthesized class labels are of high enough quality to produce a supervised classifier that outperforms the baseline unsupervised model. We also observe an inverse relationship between the class imbalance and the resulting AUPRC classification performance, i.e., the AUPRC score for DT increases as the class imbalance decreases. With each dataset used, the AUPRC score generally increases as we decrease the imbalance with RUS-1, RUS-5, then RUS-20. This pattern is also evident with the IF results, but we conclude that IF underperforms our method at all levels of class imbalance.

## V. Conclusion

In this paper, we assess an unsupervised technique for generating binary class labels for Medicare fraud detection and explore the impact of different levels of class imbalance on classification performance of a supervised classifier trained on the new labels. We evaluate using four different Medicare Part D fraud detection datasets. The largest dataset contains over 5 million instances, and the other three datasets are randomly undersampled versions of that original dataset. We produce three smaller datasets with varying levels of class imbalance using RUS. To evaluate the labels, we generate class labels on each of these datasets and train a supervised classifier using the new labels and compare it to a baseline unsupervised model that is trained on the features alone. Our empirical results show that the generated class labels produce a trained classifier that outperforms the unsupervised IF model in every test case. We also show that there is an inverse relationship between the classification performance of a model using our labels and the class imbalance of the underlying dataset. As the dataset becomes more balanced, the AUPRC scores increase. This work both evaluates and validates the performance and usability of the synthesized class labels across various levels of class imbalance in the context of Medicare fraud detection. Future work includes evaluating the method with other supervised classifiers and datasets from other domains.

## References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[2] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[3] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *IEEE intelligent systems*, vol. 24, no. 2, pp. 8–12, 2009.

[4] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[5] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*. PMLR, 2016, pp. 478–487.

[6] S. Wang and X. Yao, "Multiclass imbalance problems: Analysis and potential solutions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 4, pp. 1119–1130, 2012.

[7] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[8] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[9] T. M. Khoshgoftaar, C. Seiffert, J. Van Hulse, A. Napolitano, and A. Folleco, "Learning with limited minority class data," in *Machine Learning and Applications, 2007. ICMLA 2007. Sixth International Conference on*. IEEE, 2007, pp. 348–353.

[10] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.

[11] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine learning*, vol. 30, no. 2, pp. 195–215, 1998.

[12] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets." in *GrC*, 2006, pp. 732–737.

[13] S. Baek, D. Kwon, S. C. Suh, H. Kim, I. Kim, and J. Kim, "Clustering-based label estimation for network anomaly detection," *Digital Communications and Networks*, vol. 7, no. 1, pp. 37–44, 2021.

[14] F. Moslehi, A. Haeri, and M. R. Gholamian, "A novel selective clustering framework for appropriate labeling of clusters based on k-means algorithm," *Scientia Iranica*, vol. 27, no. 5, pp. 2621–2634, 2020.

[15] O. Maqbool and H. A. Babri, "Automated software clustering: An insight using cluster labels," *Journal of Systems and Software*, vol. 79, no. 11, pp. 1632–1648, 2006.

[16] A. Rauber, "Labelsom: On the labeling of self-organizing maps," in *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, vol. 5. IEEE, 1999, pp. 3527–3532.

[17] R. K. Kennedy, F. Villanustre, T. M. Khoshgoftaar, and Z. Salekshahrezaee, "Synthesizing class labels for highly imbalanced credit card fraud detection data," *Journal of Big Data*, vol. 11, no. 1, pp. 1–22, 2024.

[18] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[19] P. Oliveri, "Class-modelling in food analytical chemistry: development, sampling, optimisation and validation issues–a tutorial," *Analytica chimica acta*, vol. 982, pp. 9–19, 2017.

[20] U.S. Government, U.S. Centers for Medicare & Medicaid Services. What's medicare. Https://www.medicare.gov/sign-up-change-plans/decide-how-to-get-medicare/whats-medicare/what-is-medicare.html.

[21] Centers For Medicare & Medicaid Services. (2019) Medicare provider utilization and payment data. [Online]. Available: https://www.cms.gov/research-statistics-data-and-systems/statistics-trends-and-reports/medicare-provider-charge-data

[22] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, p. 42, 2018.

[23] R. A. Bauder and T. M. Khoshgoftaar, "The effects of varying class distribution on learner behavior for medicare fraud detection with imbalanced big data," *Health information science and systems*, vol. 6, pp. 1–14, 2018.

[24] J. L. Leevy, T. M. Khoshgoftaar, and J. Hancock, "Evaluating performance metrics for credit card fraud classification," in *2022 IEEE 34th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2022, pp. 1336–1341.

[25] J. T. Hancock, T. M. Khoshgoftaar, and J. M. Johnson, "Evaluating classifier performance with highly imbalanced big data," *Journal of Big Data*, vol. 10, no. 1, p. 42, 2023.

[26] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[28] F. Chollet *et al.*, "Keras," https://keras.io, 2015.