Journal of Big Data

RESEARCH Open Access

Data reduction techniques for highly imbalanced medicare Big Data

John T. Hancock[1*], Huanjing Wang[2], Taghi M. Khoshgoftaar[1] and Qianxin Liang[1]

*Correspondence:
jhancoc4@fau.edu

[1] College of Engineering and Computer Science, Florida Atlantic University, Boca Raton, USA
[2] Ogden College of Science and Engineering, Western Kentucky University, Bowling Green, USA

Abstract

In the domain of Medicare insurance fraud detection, handling imbalanced Big Data and high dimensionality remains a significant challenge. This study assesses the combined efficacy of two data reduction techniques: Random Undersampling (RUS), and a novel ensemble supervised feature selection method. The techniques are applied to optimize Machine Learning models for fraud identification in the classification of highly imbalanced Big Medicare Data. Utilizing two datasets from The Centers for Medicare & Medicaid Services (CMS) labeled by the List of Excluded Individuals/Entities (LEIE), our principal contribution lies in empirically demonstrating that data reduction techniques applied to these datasets significantly improves classification performance. The study employs a systematic experimental design to investigate various scenarios, ranging from using each technique in isolation to employing them in combination. The results indicate that a synergistic application of both techniques outperforms models that utilize all available features and data. Moreover, reduction in the number of features leads to more explainable models. Given the enormous financial implications of Medicare fraud, our findings not only offer computational advantages but also significantly enhance the effectiveness of fraud detection systems, thereby having the potential to improve healthcare services.

Keywords: Random undersampling, Ensemble supervised feature selection, Big Data, Medicare fraud detection

Introduction

Data reduction techniques for the classification of highly imbalanced Big Data are desirable since they may improve performance, and smaller data sizes generally lead to faster model training times and therefore accelerate research. We systemically investigate the application of Random Undersampling (RUS) and our novel ensemble supervised feature selection technique to the Machine Learning task of Medicare insurance fraud detection. Both feature selection and RUS are data reduction techniques. We perform experiments on two imbalanced Big Medicare Datasets. In the experiments, the data reduction techniques are applied alone, and in combination. The statistical analysis of the experimental outcomes indicates that the data reduction techniques, in combination, yield the best performance in terms of Area Under the Precision Recall Curve (AUPRC) [1]. Furthermore, that performance is significantly

better than using all features. We prefer to report results in terms of AUPRC since it is threshold agnostic, and the only other widely used, threshold-agnostic metric, Area Under the Receiver Operating Characteristic Curve (AUC) [2], is shown in previous research to be misleading for evaluating classification of imbalanced Big Data [3, 4]. Our contribution is to show that intelligent data reduction techniques improve the classification of highly imbalanced Big Medicare data.

Medicare is the United States' public health insurance program. Its mission is to provide insurance for people aged 65 and over. It is important to note that Medicare is sporadically compromised by fraudulent insurance claims. These illicit activities often go undetected, allowing unscrupulous healthcare providers to exploit weaknesses in the system. The Department of Justice managed to reclaim approximately $3 billion dollars from such fraudulent activities in 2019, as cited in their recovery report [5]. However, it is essential to recognize that this figure only represents a fraction of the total monetary loss, the full extent of which remains indeterminate. In 2019, the Centers for Medicare & Medicaid Services (CMS) estimated that improper payments, a category that includes both fraudulent and erroneous payments, amounted to roughly $100 billion [6]. Therefore, automated Medicare fraud detection has the potential to discover more fraudulent activity.

In the application domain of fraud detection, Machine Learning aids in pinpointing the small percentage data related to fraudulent activity in a vast sea of Big Medicare data. Since identification of fraud is the first step in stopping it, Machine Learning techniques may conserve substantial resources for the Medicare system by preventing fraud. In our study, we compile Medicare insurance claims datasets from several sources. The sources originate with the CMS. Furthermore, we label the datasets with the List of Excluded Individuals and Entities (LEIE). The LEIE is provided by of the United States Office of the Inspector General [7].

The performance of a classifier can be swayed by multiple effects. Two factors that can make data more difficult to classify are dimensionality, and class imbalance. Class imbalance in labeled data happens when the overwhelming majority of instances in the dataset have one particular label. This imbalance presents obstacles, as since it is possible for a classifier, optimized for a metric such as accuracy, will mislabel fraudulent activities as non-fraudulent to boost overall scores in terms of the metric. Yet, data reduction techniques provide promising avenues for addressing these challenges in Big Data. These methods, if properly applied to detect and stop Medicare insurance fraud, could substantially elevate the standard of healthcare service. This would be made possible by reducing costs related to fraud.

To address the challenges of imbalanced Big Data and high dimensionality, feature selection and data sampling are often utilized as initial data preparation steps. On one hand, data sampling is adopted to tackle the issue of class imbalance. It entails adjusting the training dataset by adding or subtracting examples to ensure a more even balance between fraudulent and non-fraudulent entries. On the other hand, feature selection, which addresses high dimensionality, focuses on choosing a specific group of attributes from the training data, and only these chosen attributes are used to construct the final model. This not only streamlines the learning process but can also enhance classification accuracy by discarding less relevant attributes.

The data sampling technique we use is RUS. RUS is a straightforward yet potent data sampling technique. It also has the added benefit of data reduction. RUS works by randomly removing samples from the majority class until a specific balance between the minority and majority classes is met. A crucial point to emphasize is that we applied RUS solely to the training datasets. In one set of experiments, we employ RUS to generate datasets with several minority to majority class proportions, such as 1:1, 1:3, 1:9, 1:27, and 1:81. It is also worth mentioning that we conduct experiments where the training datasets are left at their original class ratios. Following this approach, we developed six classification models from the datasets used in this study. This research enabled us to delve deeper into the influence of RUS on the efficiency of models, with an emphasis on how class ratios affect experimental outcomes.

For feature selection, we incorporated a supervised feature selection method based on feature ranking lists. Subsequently, through the implementation of an innovative approach, these lists are combined to yield a conclusive feature ranking. Upon the derivation of this consolidated ranking, features are selected based on their position in the list. Specifically, we select subsets comprising a top number of features, as dictated by their rank. Using these subsets, we build additional classification models. To furnish a benchmark, models were also built utilizing all features of the datasets. This systematic approach granted us a deeper comprehension regarding the interplay between feature selection and model robustness within the context of multiple learning algorithms.

In further experiments, we move on to combine data sampling and feature selection. The order of data sampling and feature selection is important, and different techniques are implemented in order to investigate the effect of applying the data reduction techniques in different orders. Therefore, an additional contribution of this study is a detailed explanation of how the techniques are applied in four distinct scenarios: RUS only, feature selection only, feature selection followed by RUS, and RUS followed by feature selection. There is an additional fifth scenario incorporated in experiments, where we do not apply data reduction techniques as a control. The remainder of our study is organized in the following sections: Related Work, Datasets, Classifiers, Methodology, Results, Statistical Analysis, and Conclusions. Following the conclusions, we include an appendix with a statistical analysis of additional experiments.

Related work

In the context of health insurance fraud detection, this section reviews existing literature that explores the classification of data with high dimensionality which is also highly imbalanced. Techniques such as data sampling and feature selection are commonly employed to mitigate the challenges associated with this specialized type of data. In our review of existing literature, we found an opportunity to contribute a study on the effect of RUS and feature selection. Since our supervised feature selection technique is novel, previous work does not cover it. Moreover, we did not find a study that provides the comprehensive analysis of the impact of feature selection techniques and sampling techniques on the performance of insurance fraud detection with Machine Learning models.

Sateesh et al. present a supervised learning framework for fraud detection in Big Medicare Data [8]. In their study, they use publicly available Medicare claims data from 2012 through 2015. We use data spanning a larger number of years, from 2013 through 2019.

Sateesh et al. utilize the same LEIE dataset for labeling their dataset that we use here. They compile a highly imbalanced dataset with over 37 million records. They employ RUS to induce multiple class ratios. Sateesh et al. use only three classifiers: Decision Tree, Logistic Regression and Support Vector Machine, whereas we use six. Furthermore, they employ multiple performance metrics including AUC, false positive rate (FPR) and false negative rate (FNR) to assess experimental outcomes. However, as stated previously, we find AUC to be a misleading metric for evaluating the performance of classifiers on highly imbalanced, Big Data. We prefer not to publish results in terms of FPR and FNR since these are threshold-dependent metrics. Sateesh et al. find that the Decision Tree and Logistic Regression classifiers perform the best overall. Particularly they found that with the 1:4 class ratio, Decision Tree has the lowest FNR, which they deem critical for detecting fraud. We maintain that threshold-agnostic metrics, such as AUPRC, give a more informative picture of overall classifier performance. While Sateesh et al.'s findings are in alignment with ours as far as data reduction techniques improving performance, our study is far more extensive, since we use five ensemble classifiers, whereas they employ three single-learner classifiers. Another key element we found missing from Sateesh et al. is that they do not report results for experiments conducted at the original class ratio, so it is unclear whether their application of RUS yields better results than leaving the data at the original class ratio. More importantly, we consider an additional data reduction technique, feature selection, and we demonstrate a precise methodology for applying the data reduction techniques in a manner that is not covered by Sateesh et al. Our study extends beyond the work done by Sateesh et al. because we also investigate the combination of RUS and feature selection, and we provide a detailed exposition of our experimental methodology for combining data reduction techniques.

Focusing on the task of identifying Medicare fraud, Hancock et al. [4] employed datasets from the CMS, which varied considerably in size, ranging from approximately 12 million to 175 million instances. All the datasets exhibited a pronounced class imbalance, prompting the use of RUS to induce the minority-to-majority class ratios 1:1, 1:3, 1:9, 1:27, and 1:81. In their methodology, five ensemble classifiers were used, and their performance was assessed using both AUC and AUPRC metrics. The study revealed that irrespective of the degree of RUS applied, the classifiers produced high AUC but low AUPRC scores. Hancock et al. emphasized the superiority of the AUPRC metric over AUC for gauging the efficacy of classifiers on imbalanced datasets. However, their findings also indicated that RUS had an adverse impact on AUPRC scores. We believe the improvement in AUPRC scores we find when applying RUS to our data is the result of our data preprocessing step of aggregation, which Hancock et al. do not employ. Another significant difference between our studies is that Hancock et al.'s study did not involve any feature selection methodology. Not only do we cover feature selection, but also we cover RUS, and we explain the various ways RUS and feature selection can be combined. Furthermore, we provide a statistical analysis on the impact of the variations of the techniques on experimental outcomes.

In a study that introduces a novel deep learning architecture for Big Medicare Data fraud detection, Mayaki and Riveill [9] propose MINN-AE. MINN-AE is a specialized Medicare fraud detection model featuring a multiple-input deep neural network supplemented by a Long-short Term Memory autoencoder component. The model's

efficacy was evaluated against nine baseline models, which included traditional Machine Learning algorithms like Logistic Regression and Random Forest, as well as various forms of artificial neural networks. Evaluation metrics employed in the study included geometric mean, precision, AUC, and AUPRC. The authors underscored the significance of the AUPRC metric over the AUC metric, particularly when dealing with highly imbalanced datasets. We were unable to locate discussion of feature selection in Mayaki and Riveill's study. Hence, Mayaki and Riveill's study is another example of a study that does not combine feature selection and sampling techniques in the Machine Learning task of fraud detection. In our study, we portray four distinct scenarios for applying data reduction techniques, and two of the scenarios involve multiple data reduction techniques.

Similar to our study, Herland et al. use publicly available data from the CMS, in a study on Medicare fraud detection [10]. Their study employs a diverse set of Medicare data, including Part B data spanning 2012 to 2015, Part D data covering 2013 to 2015, and Medicare Durable Medical Equipment, Devices & Supplies (DMEPOS) [11] data from the same years. In addition to these three datasets, they create a fourth, dubbed the Combined dataset, by merging the aforementioned datasets. Their methodology involves comparing the performance of three classification models. These are Logistic Regression, Random Forest, and Gradient Boosted Trees. The Combined dataset, when paired with a Logistic Regression model, delivers the best performance in fraud detection, according to their results. However, they do not use any of the data reduction techniques we use in this study. Unlike Herland et al., we introduce a novel feature selection technique. Furthermore, their results are exclusively reported in terms of the Area Under the Curve (AUC) metric, a measure we find unsatisfactory for evaluating classifiers on imbalanced Big Data sets. This serves as a key point of departure between our research and that conducted by Herland et al. Finally, our study also expands the range of classifiers used, since we include results from six classifiers, two of which are representatives of the Bagging Family, and three of which are Gradient Boosted Decision Tree implementations.

Lopo and Hartomo [12] compare multiple sampling techniques: RUS, Random Oversampling (ROS), Synthetic Minority Over-sampling Technique (SMOTE) [13], and Instance Hardness Threshold (IHT), for addressing class imbalance in healthcare insurance fraud detection. Using a real-world Indonesian healthcare dataset with over 2 million records and a 6:94 class imbalance ratio, sampling methods are applied to induce 1:1, 3:7 and 1:9 class ratios. Lopo and Hartomo use one classifier, XGBoost, in conjunction with the sampling techniques. Multiple evaluation metrics including AUC, AUPRC are used to evaluate experimental outcomes. Results indicate RUS and ROS perform best with 1:1 distribution, while SMOTE and IHT are more effective at the 3:7 and 1:9 class ratios, respectively. SMOTE at the 3:7 distribution level demonstrates consistently high scores across all metrics. Longer computation times are a trade-off of SMOTE and IHT. Key predictive features identified include costs, diagnoses codes, healthcare service types, gender and disease severity. While providing insights on sampling techniques for imbalanced healthcare data, limitations of a single dataset and classifier indicate opportunities for the research we document here. We use multiple data sources and additional algorithms. In this study we use

two datasets, and six learners. Moreover, we investigate the combination of RUS, and feature selection, whereas Lopo and Hartomo only perform experiments with sampling techniques.

In their investigation into the classification of imbalanced Medicare Big Data, Johnson and Khoshgoftaar apply Deep Learning algorithms and evaluate performance using Geometric Mean and Area Under the Curve (AUC) metrics [14]. Employing a dataset of approximately five million instances and increasing levels of RUS, their study finds that performance metrics deteriorate when the minority class exceeds one percent of the training data. Our research diverges from Johnson and Khoshgoftaar's in several significant ways. Firstly, we use AUPRC for evaluation of experimental outcomes, in contrast to their use of Geometric Mean And AUC. Secondly, Johnson and Khoshgoftaar do not employ feature selection techniques as we do here. Hence, their study does not include a detailed exposition of the methodologies available for combining feature selection and RUS. A third major difference in our studies is the learners we employ. Johnson and Khoshgoftaar employ neural networks, one type of Bagging classifier, Random Forest, and one type of Gradient Boosting Decision Tree classifier. Here, we use three instances of Gradient Boosting Decision Tree classifiers, and two types of Bagging classifiers. These methodological choices create a distinct difference between our study and the research conducted by Johnson and Khoshgoftaar.

Hasanin et al. investigate RUS and feature selection for addressing class imbalance in bioinformatics Big Data [15]. The Evolutionary Computation for Big Data and Learning 2014 (ECBDL'14) competition dataset with approximately 32 million records is used in their study. The dataset is imbalanced, and has numerous features. Feature selection, based on feature importance with a single learner, Random Forest, is used to do feature selection. Here, we use a more sophisticated feature selection technique that involves six learners. Hasanin et al. employ RUS to address class imbalance. Random Forest, Logistic Regression and Gradient Boosted Trees are evaluated using true positive rate (TPR), true negative rate (TNR) and their product. Their study demonstrates RUS with feature selection can effectively address class imbalance and high dimensionality in Big Data, outperforming prior methods on the ECBDL'14 dataset while lowering computational costs. Our study reaches beyond the work of Hasanin et al. not only because of our more sophisticated feature selection technique, but also because of our evaluation of more classifiers from the Bagging and Boosting families of classifiers, as well as our use of two highly imbalanced, Big Medicare Data datasets. A significant difference between our studies is in their application domains. Our study is in the healthcare insurance fraud detection domain, whereas Hasanin et al.'s use data from the protein structure prediction application domain. A more important difference between our studies is the methodologies used. Hasanin et al. employ only one scenario of applying RUS and feature selection. Therefore, it seems more preliminary, and not comprehensive. We investigate four data reduction scenarios, two of which involve RUS and feature selection combined, and we provide results that indicate which scenario yields the best results.

"Explainable machine learning models for Medicare fraud detection" by Hancock et al. [16] is a study which focuses on the use of feature selection to build more explainable models. Feature selection engenders simpler Machine Learning models, which are thus easier to explain. While this study employs feature selection, and is in the same

application domain as Hancock et al.'s, this study has a different focus, and provides a different contribution. First, this study is focused on data reduction. Hence, we employ sampling techniques as well as feature selection. Second, we cover the options one faces when designing experiments with multiple data reduction techniques. The study by Hancock et al. does not involve the different methodologies practitioners may use when designing experiments that involve multiple data reduction techniques. We provide a detailed explanation of the methodologies one may employ to use more than one data reduction technique. The statistical analysis we provide is an example of how to compare experimental outcomes involving multiple data reduction techniques. Such an analysis is not provided by Hancock et al. For these reasons, this study is vastly different from Hancock et al.

Our literature review reveals an opportunity to extend the field of research in the application of Machine Learning to the task of Medicare insurance fraud detection. Of the related studies we surveyed, we did not find a study that provides the in-depth coverage of the application of an ensemble feature selection technique. Moreover, we found some studies did not contain results of experiments combining feature selection and RUS. Of the studies that use a combination of RUS and Feature Selection, they do not contain an investigation of the different scenarios of RUS and Feature selection, the effects of applying them singly or combined, and the order of combination. Other studies only made use of one dataset. Many of the studies failed to provide the extensive array of classifiers that we use here. In summary, our review of literature exposed the need for a comprehensive study into the effects of feature selection and sampling techniques in the classification of multiple highly imbalanced Big Data datasets where experimental outcomes are documented in terms of AUPRC, and backed by the statistical analyses to prove the efficacy of the data reduction techniques.

Datasets

In our research, we employ datasets synthesized from two US government agencies, specifically the CMS and the United States Office of Inspector General (OIG). Data from the CMS is constituted by Medicare Health insurance plans, known as Part B Part D. To prepare these data for supervised Machine Learning applications, data aggregation is employed for both Part B and Part D datasets. Our datasets derive from publicly accessible Medicare information for the years 2013 to 2019. These datasets are procured in a comma-separated format from the CMS website. Within the sphere of health insurance, a plan delineates the coverage agreement between the insurer and the insured, specifying the range of treatments and medications that the insurance provider will pay for. Part B is oriented towards covering treatments and procedures, whereas Part D is tailored to cover prescription medications. After this preprocessing phase, the datasets are completed through a labeling process. This additional information is sourced from the OIG's LEIE [7].

The methodologies for compiling and preprocessing these datasets have been previously detailed in [17]. Before elaborating on the data aggregation and labeling processes, we discuss the Medicare Parts B and D datasets in detail. To bolster our understanding of the attributes present within these datasets, we consult data dictionaries supplied by CMS, which are available in the public domain [18–20], and [21].

For our study's focus on Medicare Part D data, we leverage two distinct datasets. We refer to the first dataset as the "provider-drug-level Part D data". The CMS has designated this dataset as the Medicare Part D Prescribers – by Provider and Drug dataset [22]. This source offers granular data that captures each unique combination of healthcare provider, prescribed medication, and the year when the medications were prescribed. Contrastingly, the second source, which we call the "provider-level Part D data", is collected from the Medicare Part D Prescribers – by Provider dataset [23]. This latter dataset provides a broader view, furnishing only one record for each healthcare provider per year. Thus, difference between the provider-drug-level Part D data and the provider-level Part D data lies in their respective degrees of specificity, with the former offering a more detailed account of prescription practices than the latter.

The provider-drug-level Part D data has 22 attributes. A subset of these attributes, specifically those pertaining to provider names and addresses, are excluded from Machine Learning processes to circumvent model memorization that would compromise generalization. However, we retain the National Provider Identifier (NPI) as an exception for future labeling tasks. The dataset contains two specific categorical features that denote the class of medication being prescribed. These features are eliminated during the data aggregation phase. Descriptive statistics are introduced that serve as proxies for the information removed when discarding the medication identifier features. Furthermore, while the attribute indicating the year of the claim is instrumental for aggregation, it is not used as an independent variable in the supervised Machine Learning process. Several numeric features in the dataset are directly pertinent to our Machine Learning application. These comprise metrics on the frequency and volume of prescription claims, the total associated costs, and the number of patients involved. Additional granular features are also available, focusing specifically on patients aged 65 and above. The provider-drug-level Part D records encompass approximately 174 million entries. It is worth noting that upon aggregation, we maintain a categorical feature that identifies the type of healthcare provider, which is essential for subsequent analyses. The final provider-drug-level Part D features we use are listed in Table 1. This concludes our summary of the provider-drug-level Part D data.

Now we move on to describe the provider-level Part D data. Records in this dataset have 46 fields. These attributes are based on insurance claims submitted by providers across all medications prescribed within a given year. A detailed listing of these features is provided in Table 3. The descriptions of the features are taken from the data dictionary [21]. Among these features, ten are summary statistics that characterize the beneficiaries for whom claims are submitted by the provider. Additionally, an average beneficiary risk

Table 1 Provider-drug-level part D base features, descriptions from [20]

Feature	Description
Prscrbr_Type	The Medicare specialty code, describes the type of practice
Tot_Clms	The number of Medicare Part D claims this includes original prescriptions and refills.
Tot_30day_Fills	The aggregate number of Medicare Part D standardized 30-day fills
Tot_Day_Suply	The aggregate number of day's supply for which this drug was dispensed
Tot_Drug_Cst	The aggregate drug cost paid for all associated claims
Tot_Benes	The total number of unique Medicare Part D beneficiaries with at least one claim for the drug

score is included using a risk-adjustment model based on Hierarchical Condition Categories (HCC). According to CMS guidelines, an HCC score exceeding the mean value of 1.08 is indicative of projected Medicare expenditures that surpass the average.

The provider-level Part D data also furnishes attributes capturing subtotals for a myriad of claim categories, such as Low-Income Subsidy (LIS) claims, Medicare Advantage Prescription Drug Plan (MAPD) coverage claims, and Medicare Prescription Drug Plan (PDP) claims. These subtotals are enumerated in various forms: the overall number of claims, the aggregate 30-day prescription orders, cumulative drug costs, the total day's supply dispensed, and the grand total of beneficiaries involved. Furthermore, this dataset is also segmented by drug type categories, offering statistics specifically for claims related to opiate drugs, long-acting (LA) opiate drugs, antibiotics, and antipsychotic medications.

In addition to the datasets focused on Medicare Part D, our research also incorporates two distinct datasets pertaining to Medicare Part B. Similar to the Part D datasets, these Part B datasets also vary in their level of granularity. The first dataset, which we refer to as the "provider-service-level Part B data," emanates from the Medicare Physician & Other Practitioners – by Provider and Service dataset [25]. This dataset offers a granular perspective, featuring an individual record for each distinct treatment or procedure submitted as a claim to Medicare by a provider for a given year. Conversely, the second dataset, which we refer to as "provider-level Part B data" originates from the Medicare Physician & Other Practitioners – by Provider [19] dataset. This dataset provides an aggregate view, containing records that summarize a provider's entire claim activity for all treatments and procedures over the course of a year. Thus, the main distinction between these two datasets lies in the scope of claim-related activities they encapsulate, with the former offering specific treatment or procedure information and the latter delivering a comprehensive annual summary.

The provider-service-level Part B data contains a total of 29 attributes. Similar to the Part D data, a portion of these attributes are related to provider demographics, which we intentionally exclude from Machine Learning models to avoid overfitting. Attributes denoting specific treatments or procedures are also present but are not included in the final aggregated dataset, which is tailored to focus on provider-level characteristics. However, we do retain several categorical attributes, specifically those related to the provider's location type, gender, and provider type. In terms of numerical features, this dataset is rich in data about submitted claims related to specific treatments and procedures. These numerical attributes cover a variety of metrics, including the total frequency of the rendered service, the aggregate number of patients benefiting from the service, the average daily beneficiary count, and statistical data concerning both the provider's average charges and Medicare's average payments for the service. The provider-service-level Part B dataset has a volume of approximately 68 million records. The features used in generating the final Part B dataset are documented in Table 2.

The provider-level Part B data is detailed in Table 4. It has a total of 47 attributes. Within this dataset, one subset of features focuses on the demographic distribution of treated patients, categorized by age brackets: under 65, 65–75, 75–84, and 85 or above. An additional attribute calculates the average number of patients across these age groups. The dataset includes seven core features related to annual claims for

Table 2 Provider-service-level part B base features, descriptions from [24]

Rndrng_Prvdr_Type	Derived from the provider specialty code reported on the claim
Place_Of_Srvc	Identifies whether the place of service submitted on the claims is a facility
Rndrng_Prvdr_Gndr	The provider's gender
Tot_Srvcs	Number of services provided; note that the metrics used to count the number provided can vary from service to service
Tot_Benes	Number of distinct Medicare beneficiaries receiving the service
Tot_Bene_Day_Srvcs	Number of distinct Medicare beneficiary/per day services
Avg_Sbmtd_Chrg	Average of the charges that the provider submitted for the service
Avg_Mdcr_Pymt_Amt	Average amount that Medicare paid after deductible and coinsurance amounts have been deducted for the line item service

treatments and procedures sent by providers to Medicare. These features encompass variables such as the total types of distinct procedures conducted, the overall patient count, the annual amount billed to Medicare by the provider, the allowable Medicare payment, and the actual payment disbursed by Medicare. A distinct attribute standardizes these payment amounts by adjusting for regional cost differences, facilitating cross-regional financial comparisons.

Furthermore, 14 attributes of the provider-level part B data are derived by segmenting these seven core features into two distinct classifications: medication-related services and all other services. Additionally, the dataset contains features that enumerate the gender distribution among patients. The dataset also includes 18 attributes that quantify the prevalence of specified chronic conditions, such as Alzheimer's, kidney disease, and asthma, among the provider's patient population. Moreover, an HCC risk score feature, analogous to that found in the provider-level Part D data, is also incorporated. To summarize, the provider-level Part B dataset contains attributes incorporating both demographic and financial attributes of the provider's patient population along with information about their chronic conditions.

In our methodology for both Part B and Part D datasets, we aggregate information at the level of individual drugs or services from the initial source. Subsequently, this aggregated data is augmented with attributes from the second source, which is organized at the provider level. As stated previously, for the provider-service-level Part B data we retain attributes such as provider type, service location, and provider gender, given that they pertain to the provider level. Features specific to individual services, like the Healthcare Common Procedure Coding System (HCPCS) code, are excluded from our final dataset.

Here we transition to a discussion of how we combine the Part B and Part D datasets provided by the CMS into the finalized datasets. To further distill the datasets, we generate a set of summary statistics–namely sum, mean, median, minimum, maximum, and standard deviation–for each remaining numerical attribute. These summary statistics are calculated for each provider for the entire year. In the final Part D dataset, all attributes listed in Table 1, barring Prscbr_Type, have six corresponding summary statistics. A similar set of base features for the Part B data is illustrated in Table 2. Excepting Rndrng_Prvdr_Gndr, in the final Part B dataset, each feature listed in Table 2, is also accompanied by the six summary statistics. The aggregated Part D

Table 3 Provider-level part D features, descriptions copied from [21]

Feature	Description
GE65_Tot_Clms	The number of Medicare Part D claims for beneficiaries age 65 and older
GE65_Tot_30day_Fills	The number of Medicare Part D standardized 30-day fills for beneficiaries age 65 and older
GE65_Tot_Drug_Cst	The aggregate total drug cost paid for all associated claims for beneficiaries age 65 and older
GE65_Tot_Day_Suply	The aggregate number of day's supply for which this drug was dispensed, for beneficiaries age 65 and older
GE65_Tot_Benes	The total number of unique Medicare Part D beneficiaries age 65 and older with at least one claim for the drug
Brnd_Tot_Clms	Total claims of brand-name drugs, including refills
Brnd_Tot_Drug_Cst	Aggregate drug cost paid for brand-name drugs
Gnrc_Tot_Clms	Total claims of generic drugs, including refills
Gnrc_Tot_Drug_Cst	Aggregate cost paid for generic drugs
Othr_Tot_Clms	Total claims of other drugs, including refills. A drug is classified as "other" using any FDA approval categories not included in the brand or generic definitions
Othr_Tot_Drug_Cst	Aggregate cost paid for all other drugs not classified as brand or generic
MAPD_Tot_Clms	The number of claims for beneficiaries covered by (Medicare Advantage plan that includes Medicare Part (MDAPD)
MAPD_Tot_Drug_Cst	Aggregate cost paid for claims filled by beneficiaries in MAPD plans
PDP_Tot_Clms	The number of claims for beneficiaries covered by standalone Prescription Drug Plans (PDPs)
PDP_Tot_Drug_Cst	Aggregate drug cost paid for claims filled by beneficiaries in standalone PDPs
LIS_Tot_Clms	Total number of claims from this prescriber, including refills, for beneficiaries with a Part D low-income subsidy (LIS)
LIS_Drug_Cst	Aggregate drug cost paid for claims for beneficiaries with a Part D low-income subsidy
NonLIS_Tot_Clms	Total number of claims from this prescriber, including refills, for beneficiaries without a Part D low-income subsidy
NonLIS_Drug_Cst	Aggregate drug cost paid for claims for beneficiaries without a Part D low-income subsidy
Opioid_Tot_Clms	Total claims of opioid drugs, including refills
Opioid_Tot_Drug_Cst	Aggregate cost paid for opioid drugs
Opioid_Tot_Suply	The aggregate number of day's supply for opioid drugs
Opioid_Tot_Benes	The total number of unique Medicare Part D beneficiaries with at least one opioid claim
Opioid_Prscrbr_Rate	The percent of the Tot_Clms represented by the Opioid_Tot_Clms
Opioid_LA_Tot_Clms	The aggregate number of day's supply for long-acting (LA) opioid drugs
Opioid_LA_Tot_Drug_Cst	Aggregate cost paid for long-acting opioid drugs
Opioid_LA_Tot_Suply	The aggregate number of day's supply for long-acting opioid drugs
Opioid_LA_Tot_Benes	The total number of unique Medicare Part D beneficiaries with at least one long-acting opioid claim
Opioid_LA_Prscrbr_Rate	The percent of the Opioid_Tot_Clms represented by the Opioid_LA_Tot_Clms
Antbtc_Tot_Clms	Total claims of antibiotic drugs, including refills
Antbtc_Tot_Drug_Cst	Aggregate cost paid for antibiotic drugs
Antbtc_Tot_Benes	The total number of unique Medicare Part D beneficiaries with at least one antibiotic claim
Antpsyct_GE65_Tot_Clms	Total claims of antipsychotic drugs, including refills, for beneficiaries age 65 and older
Antpsyct_GE65_Tot_Drug_Cst	Aggregate cost paid for antipsychotic drugs for beneficiaries age 65 and older
Antpsyct_GE65_Bene_Suprsn_Flag	A flag indicating the reason the Antpsyct_GE65_Tot_Benes variable is suppressed

Table 3 (continued)

Feature	Description
Antpsyct_GE65_Tot_Benes	The total number of unique Medicare Part D beneficiaries age 65 and older with at least one antipsychotic claim
Bene_Avg_Age	Average age of beneficiaries
Bene_Age_LT_65_Cnt	Number of beneficiaries under the age of 65
Bene_Age_65_74_Cnt	Number of beneficiaries between the ages of 65 and 74
Bene_Age_75_84_Cnt	Number of beneficiaries between the ages of 75 and 84
Bene_Age_GT_84_Cnt	Number of beneficiaries over the age of 84
Bene_Feml_Cnt	Number of female beneficiaries
Bene_Male_Cnt	Number of male beneficiaries
Bene_Dual_Cnt	Number of Medicare beneficiaries qualified to receive Medicare and Medicaid benefits
Bene_Ndual_Cnt	Number of Medicare beneficiaries qualified to receive Medicare only benefits
Bene_Avg_Risk_Scre	Average Hierarchical Condition Category (HCC) risk score of beneficiaries

dataset comprises roughly 6.3 million instances, while the aggregated Part B dataset contains approximately 8.7 million instances.

Following the aggregation of both Part D and Part B data at the respective service and drug levels, we proceed to integrate these datasets with their corresponding provider-level data. Specifically, the National Provider Identifier (NPI) serves as the linkage criterion for merging the aggregated Part D data with its provider-level counterpart. The NPI plays a similar role for the Part B data. This merging process culminates in the creation of unlabeled datasets. Put another way, we join the aggregated provider-drug-level Part D data with the provider-level Part D data, and we join the aggregated provider-treatment-level Part B data with the provider-level part B data.

In the case of Part B data, the size of the unlabeled dataset remains consistent with the previously aggregated dataset. Conversely, the unlabeled Part D dataset is reduced by approximately one million records, attributable to the absence of corresponding provider-level records for certain NPIs and specific years. As a result of this data integration, the attribute count stands at 82 for the unlabeled Part B dataset and 80 for the unlabeled Part D dataset.

In the final stage of dataset preparation for both Part B and Part D data, we incorporate labels sourced from the LEIE. Administered on a monthly basis by the Office of Inspector General (OIG), the LEIE serves as an authoritative source for healthcare providers disallowed from submitting Medicare insurance claims due to legal convictions. We use the same methodology for labeling both the Part D and Part B datasets. We align with the fraud indicators utilized by Bauder and Khoshgoftaar, as described in their 2016 publication, to categorize types of exclusions that trigger a fraud label [26]. Notably, the National Provider Identifier (NPI) is the key for joining the LEIE to the Part B or Part D datasets. Whenever a healthcare provider appears on the LEIE under the exclusion types for Medicare Fraud, all records affiliated with that provider, and dated prior to the conclusion of the exclusion time frame, are classified as fraudulent.

Table 4 Provider level part B features, descriptions copied from [18]

Tot_HCPCS_Cds	Total number of unique HCPCS codes
Tot_Benes	Total Medicare beneficiaries receiving services from the provider
Tot_Srvcs	Total provider services
Tot_Sbmtd_Chrg	The total charges that the provider submitted for all services
Tot_Mdcr_Alowd_Amt	The Medicare allowed amount for all provider services
Tot_Mdcr_Pymt_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for all the provider's line item service
Tot_Mdcr_Stdzd_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for the line item service and after standardization of the Medicare payment has been applied
Drug_Tot_HCPCS_Cds	Total number of HCPCS codes for drug services
Drug_Tot_Benes	Total Medicare beneficiaries receiving drug services
Drug_Tot_Srvcs	Total drug services
Drug_Sbmtd_Chrg	The total charges that the provider submitted for drug services
Drug_Mdcr_Alowd_Amt	The Medicare allowed amount for drug services
Drug_Mdcr_Pymt_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for all the provider's line item drug services
Drug_Mdcr_Stdzd_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for the line item drug service
Med_Tot_HCPCS_Cds	Total number of HCPCS codes associated with medical services
Med_Tot_Benes	Total Medicare beneficiaries receiving medical services
Med_Tot_Srvcs	Total medical services
Med_Sbmtd_Chrg	The total charges that the provider submitted for medical services
Med_Mdcr_Alowd_Amt	The Medicare allowed amount for medical services
Med_Mdcr_Pymt_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for all of the provider's line item medical services
Med_Mdcr_Stdzd_Amt	Total amount that Medicare paid after deductible and coinsurance amounts have been deducted for the line item medical service
Bene_Avg_Age	Average age of beneficiaries. Beneficiary age is calculated at the end of the calendar year or at the time of death
Bene_Age_LT_65_Cnt	Number of beneficiaries under the age of 65. Beneficiary age is calculated at the end of the calendar year or at the time of death
Bene_Age_65_74_Cnt	Number of beneficiaries between the ages of 65 and 74. Beneficiary age is calculated at the end of the calendar year or at the time of death
Bene_Age_75_84_Cnt	Number of beneficiaries between the ages of 75 and 84
Bene_Age_GT_84_Cnt	Number of beneficiaries over the age of 84. Beneficiary age is calculated at the end of the calendar year or at the time of death
Bene_Feml_Cnt	Number of female beneficiaries
Bene_Male_Cnt	Number of male beneficiaries
Bene_Dual_Cnt	Number of Medicare beneficiaries qualified to receive Medicare and Medicaid benefits
Bene_Ndual_Cnt	Number of Medicare beneficiaries qualified to receive Medicare only benefits
Bene_CC_AF_Pct	Percent of beneficiaries meeting the Chronic Conditions Data Warehouse chronic condition algorithm for atrial fibrillation
Bene_CC_Alzhmr_Pct	Percent of beneficiaries meeting the Chronic Conditions Data Warehouse (CCW) chronic condition algorithm for Alzheimer's, related disorders, or dementia
Bene_CC_Asthma_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for Asthma
Bene_CC_Cncr_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithms for cancer
Bene_CC_CHF_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for heart failure
Bene_CC_CKD_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for chronic kidney disease
Bene_CC_COPD_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for chronic obstructive pulmonary disease
Bene_CC_Dprssn_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for depression

Table 4 (continued)

Bene_CC_Dbts_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for diabetes
Bene_CC_Hyplpdma_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for hyperlipidemia
Bene_CC_Hyprtnsn_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for hypertension
Bene_CC_IHD_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for ischemic heart disease
Bene_CC_Opo_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for osteoporosis
Bene_CC_RAOA_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for rheumatoid arthritis/osteoarthritis
Bene_CC_Sz_Pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for schizophrenia and other psychotic disorders
Bene_cc_strok_pct	Percent of beneficiaries meeting the CCW chronic condition algorithm for stroke
Bene_Avg_Risk_Scre	Average Hierarchical Condition Category (HCC) risk score of beneficiaries

Table 5 Summary of part B and part D datasets

Dataset	Instance count	Fraudulent	Ratio fraudulent	Number of features
Part D	5,344,106	3,700	0.0693%	80
Part B	8,669,497	3,954	0.0456%	82

The datasets for Part B and Part D span complete calendar years, whereas exclusion periods in the LEIE terminate at specific months. To reconcile this disparity, we round the conclusion of an exclusion period to the nearest year-end. Following the lapse of an exclusion term, healthcare providers are consequently removed from the LEIE, making them eligible to submit Medicare claims anew. All subsequent claim data for these providers are, therefore, deemed non-fraudulent.

It is worth noting that the present version of the LEIE will not incorporate records of providers who were once listed but are no longer excluded. Therefore, for a comprehensive dataset spanning multiple years, one may need to resort to archival services, such as the Internet Archive Tool, to consult previous LEIE editions for accurately labeling older CMS data. A summary of this labeling undertaking for the Part B and Part D datasets is provided in Table 5.

Classifiers

To ensure the repeatability of our findings, we have employed a combination of both ensemble and linear learning algorithms for our classification experiments. The algorithms are open-source, and widely available via the Internet. Specifically, our ensemble methods comprise XGBoost [27], LightGBM [28], Extremely Randomized Trees (ET) [29], Random Forest [30], and CatBoost [31]. For linear classification, we utilized Logistic Regression [32]. Additionally, for the feature selection process, we incorporated the aforementioned ensemble methods along with the Decision Tree algorithm [33]. Here, we provide an in-depth overview of the fundamental characteristics of each Machine Learning method employed in our research.

Given its inherent simplicity relative to other classifiers, Logistic Regression, [32] serves as an apt starting point. At its core, Logistic Regression revolves around tailoring a sigmoid function to a dataset. This is accomplished by setting the sigmoid function's parameters to the maximum likelihood estimate of their value, given the training data. A Logistic Regression model is characterized by one parameter for each independent variable. This attribute renders it considerably simpler than many ensemble methods which have a larger number of parameters due to their being composed of instances of other models. Our rationale behind employing Logistic Regression in our study is to ensure we check for the potential efficacy of a simpler model before recommending more sophisticated algorithms that consume far more computing resources.

In this research, we employ an array of ensemble algorithms that can be broadly classified into Bagging techniques and Gradient Boosted Decision Tree (GBDT) techniques. Within these categorizations, ET and Random Forest are examples of the Bagging paradigm. CatBoost, XGBoost, and LightGBM exemplify the GBDT framework. Our selection is underpinned by a desire to bolster the robustness of our findings across diverse algorithmic methodologies. Bagging and GBDT techniques take distinct approaches as ensemble techniques for harnessing the power of multiple learners in classification scenarios.

The landscape of Machine Learning changed with the introduction of the Bagging concept by Breiman in 1996 [34]. Breiman's work illuminated the applicability of Bagging in classification and regression scenarios. Given that our investigative efforts are channeled towards binary classification, we delve deeper into Breiman's explanation of Bagging's for binary classification. At its core, Bagging entails the training of multiple instances of a Machine Learning algorithm to an array of bootstrap samples procured from the training dataset, thus building an ensemble of learners. Bootstrap samples are instances sampled, with replacement, from the training data, as defined in [35]. An intriguing facet of the Bagging technique is the allowance for the instances of the algorithms to be weak learners. A weak learner is a Machine Learning algorithm that would yield suboptimal performance in isolation. However, in an ensemble setting, the aggregate results of the weak learners can be much better than their individual performance. The underlying logic for Bagging's proficiency in bolstering classification performance is probabilistic in nature. Suppose it is the case that there is a better than 50% chance a weak learner makes an accurate classification. Then, as we augment the count of such weak learners in an ensemble, the likelihood of a majority leaning towards accurate classification amplifies. Therefore, with a larger ensemble size, Bagging exhibits an increased propensity to render correct classification results. The culmination of the Bagging process is a classification output, which is determined by the consensus–or the class identified by the majority–among the ensemble learners.

In our study, one of the Bagging techniques we employ is Random Forest, an algorithm conceived by Breiman [30]. The foundational architecture of Random Forest is deeply rooted in the Bagging principle but is distinguished by an innovative enhancement. To fully appreciate this enhancement, it is imperative to first elucidate the concept of a "split" in the context of Decision Trees. Within a Decision Tree structure, the non-leaf nodes encapsulate rules. These rules specify which subsequent node should be navigated to, based on the comparison of an attribute of the data to a value which was

optimally determined during the model's training phase. The value used in the comparison is referred to as a split. Consequently, the essence of training a Decision Tree model is contingent upon astutely determining the optimal values for these splits. Since it is an instance of the Bagging paradigm, the weak learners in Random Forest are Decision Trees. Therefore, split calculation is important for Random Forest. Random Forest amplifies the traditional Decision Tree algorithm with a novel twist. In lieu of relying on the entire feature set, when calculating the optimal value for a split, Random Forest adopts a strategy of considering only a randomly chosen subset of features. This approach differentiates Random Forest from a plain Bagging approach applied to Decision Trees.

In a further exploration of the Bagging technique, we added the Extremely Randomized Trees (ET) classifier [29] to the collection of algorithms used in this study. The ET classifier, while emerging from the same lineage as Random Forest, manifests a distinct methodological approach, especially in its strategy for Decision Tree splits. For comparison, Random Forest adopts deterministic logic to calculate splits in its constituent Decision Trees. Therefore, the process is driven by an algorithm for identifying the optimal split values for Decision Trees. In contrast, ET introduces a change by forgoing this deterministic methodology. Instead, it embarks on a route of randomly selecting split values. While this deviation might appear counter-intuitive at first glance, our empirical research offers intriguing insights. Specifically, in the domain of Medicare fraud detection, especially when handling highly imbalanced Big Data sets, ET's random split selection strategy frequently exhibits competitive classification performance.

GBDT classifiers emerge from the foundational work of Friedman on the Gradient Boosted Machine algorithm [36]. Central to Friedman's ensemble approach is its iterative nature. Beginning with a preliminary learner, an initial prediction set $\hat{\mathbf{y}}$ for the dependent variable $\mathbf{y}$ is generated. Discrepancies arising between this predicted set $\hat{\mathbf{y}}$ and the genuine values $\mathbf{y}$ are used to calculate the residual vector $\mathbf{y} - \hat{\mathbf{y}}$. This residual is then treated as a dependent variable, estimated using a subsequent learner. The sum of the outputs of the two models, an ensemble, has improved accuracy in predicting $\mathbf{y}$. As this iterative model-building extends, every subsequent learner is attuned to the residuals of the preceding ensemble, thus refining the estimation at each juncture. Contemporary algorithms such as XGBoost, LightGBM, and CatBoost refine Friedman's original concept. Their shared reliance on Decision Trees warrants their collective designation as Gradient Boosted Decision Trees (GBDTs).

In our study, we employ CatBoost, a novel GBDT framework proposed by Prokhorenkova et al. in 2018 [31]. Central to CatBoost's research ethos, as articulated in its foundational paper, is the challenge of overfitting. Addressing this concern, Prokhorenkova et al. advanced two distinct strategies. Ordered Boosting, the primary strategy, emphasizes the judicious selection of training instances for fitting Decision Trees within the CatBoost ensemble. The entire process of incorporating a Decision Tree into the GBDT ensemble is bifurcated into two phases. Initially, prospective Decision Trees are fit to the dependent variable present in the training dataset. Subsequently, these candidate trees undergo an evaluation process, with the prime objective being the identification and selection of a tree that best amplifies the

ensemble's cumulative efficacy. Crucially, within the Ordered Boosting paradigm, there is a stringent guideline ensuring that instances utilized for fitting a particular Decision Tree are excluded from its evaluation phase. Such a meticulous approach acts as a bulwark against the ensemble's propensity to overfit to the training dataset.

The designers of CatBoost took another pivotal measure to curtail overfitting. This is the Ordered Target Statistics method, designed for the encoding of categorical features. This approach is underpinned by the foundational concept of target encoding. At the heart of target encoding is the process wherein the encoded value for a specific categorical feature is ascertained based on the mean value of the dependent variable it correlates with. Notably, such a straightforward encoding mechanism is fraught with potential pitfalls, a prominent one being "target leakage", as delineated by Prokhorenkova et al. We define target leakage with an illustrative scenario. Imagine a circumstance where an encoded feature's value associates with one dependent variable value in the training and a completely different value in the test data. Under conventional target encoding, this feature's encoded value conveys information regarding the target value it co-occurs with in the training data, but not the test data. Therefore, the target encoding undermines the feature's efficacy as a predictor for the dependent variable. To safeguard against such pitfalls, the Ordered Target Statistics method is engineered to ensure that the encoding of a categorical feature for a particular instance is predicated solely on data from other instances. This intrinsic design criterion eliminates the potential for an instance's encoded feature value to be intrinsically linked to its corresponding dependent variable value. In summary, the Ordered Target Statistics method, with its approach to categorical feature encoding, fortifies CatBoost's defense mechanisms against overfitting.

In recent studies, the Gradient Boosting Decision Tree (GBDT) method has seen significant advancements with the introduction of the LightGBM framework. Established by Ke et al. in their 2017 paper [28], LightGBM was conceived to rival the performance of XGBoost, yet with an emphasis on reducing resource demands. Central to LightGBM's efficacy are two novel advancements pioneered by Ke et al.: Exclusive Feature Bundling (EFB) and Gradient-based One-Side Sampling (GOSS). EFB provides a mechanism to diminish the dataset's dimensions by combining pairs of attributes. The crux of this method hinges on the observation that certain attributes in a dataset, especially those exhibiting sparsity, can occasionally demonstrate mutually exclusive occurrences in their infrequent values. Sparse data is characterized by predominantly static values punctuated by infrequent variations. In datasets with multiple sparse features, these attributes can be combined into a singular feature, ensuring minimal information loss. EFB's application, especially pertinent to sparse data, leads to a reduction in the dataset's dimensions, translating to expedited training. Conversely, GOSS offers an approach to optimize the number of training instances during LightGBM's training phase. The principle behind GOSS is to prioritize instances based on their contribution to the aggregate loss function, which is integral to fitting the GBDT ensemble to the training dataset. This procedure ensures that instances contributing beyond a modifiable threshold to the model's overall loss are retained for ensuing iterations. Conversely, instances falling below this threshold find themselves excluded. Collectively, through the combined

capabilities of GOSS and EFB, Ke et al. delivered a GBDT framework in LightGBM that demonstrates an appreciable reduction in computational overhead.

In 2016, Chen and Guestrin unveiled XGBoost, marking it as the pioneering GBDT among the three we employ in this study. Beyond the conventional GBDT framework, XGBoost introduces several enhancements. Notably, during its training phase, an advanced loss function is added that integrates an additional regularization term, acting as a countermeasure against overfitting. This enhancement is further complemented by XGBoost's refined approach to determining splits within its Decision Tree ensemble. Chen and Guestrin's innovative "approximate algorithm" is another enhancement to Friedman's original approach. It facilitates the estimation of optimal split values, proving useful when the full dataset surpasses the constraints of available memory and exhibiting benefits in distributed computing scenarios. Additionally, XGBoost addresses challenges posed by sparse data. With its "sparsity aware split finding" mechanism, XGBoost can fit more efficiently to sparse data.

The next algorithm we discuss is Decision Tree. This algorithm holds paramount significance in our study given its foundational role in the ensemble techniques we employ, as well as its utilization in our ensemble feature selection method. Hence, we expand on our comments regarding Decision Trees above. Decision tree algorithms iteratively construct a hierarchical structure that captures a decision process. Here we refer to the Decision Tree structure, as well as the algorithm that builds it both as "Decision Tree." Initially, Decision Tree establishes a rule based on an attribute value compared to a threshold, or split. This rule is visualized as a node from which two edges emerge, leading to nodes representing either outcome of the binary classification. These end nodes, denoting specific class assignments, are referred to as leaf nodes. The decision rule divides a dataset in two classes. Decision Tree will choose the value for the split that optimizes the change in the value of a metric evaluated on the full sample, as well as the two subsamples that the decision rule divides the data into. Two such metrics are Shannon Entropy and Gini Impurity. Intuitively, both of these metrics measure the homogeneity of elements in a set. A well-chosen value for a split increases the homogeneity of the elements in the subsets, meaning the split is the best value for dividing the dataset into two distinct classes. During the Decision Tree construction process, the paths to leaf nodes undergo further subdivisions by incorporating additional decision rules, which further increase the homogeneity of the subsets of classes that the Decision Tree divides the dataset into. This iterative enhancement is represented by the introduction of intermediary nodes, effectively increasing the number leaf nodes in the decision tree. Please see Table 6 for the hyperparameters used for all algorithms described here. If an algorithm is not mentioned, we used default values for all hyperparameters.

This concludes our discussion of the classifiers used in our study. Now we move on to discuss the methodology of our experiments.

Methodology

Here we describe our experimental methodology. We conduct experiments with the Part B and Part D data by following five scenarios. One element in common to each scenario is ten iterations of five-fold cross validation, so we explain that first.

Table 6 Hyperparameter settings used in experiments

Classifier	Parameter name	Parameter setting
CatBoost	`task_type`	`'GPU'*`
	`max_ctr_complexity`	`1`
	`max_depth`	`5`
ET	`max_depth`	`8`
XGBoost	`max_depth`	`3`
	`tree_method`	`'gpu_hist'*`
LightGBM	`max_depth`	`4`
Random Forest	`max_depth`	`4`

*Setting selects Graphics Processing Unit (GPU) implementation of the classifier

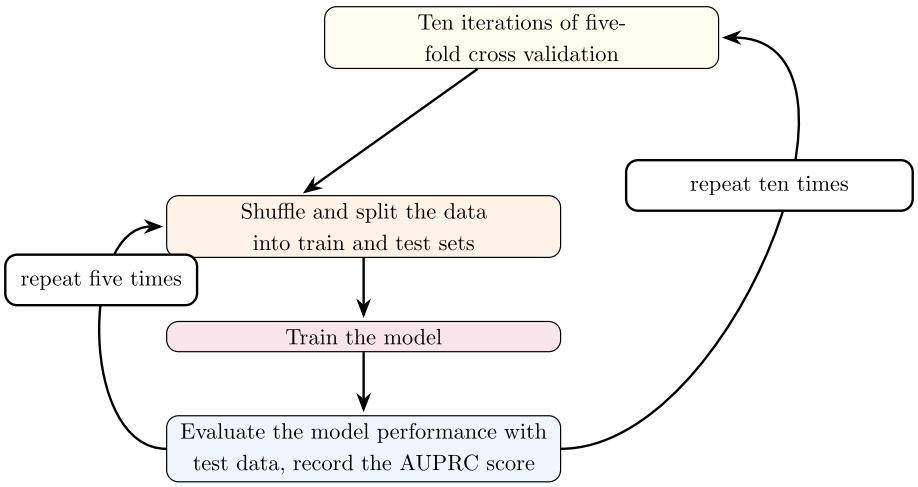

Fig. 1 Five-fold cross validation

Five-fold cross validation is a process of five iterations. In five-fold cross-validation, we systematically divide the data into five parts with an approximately equal size and distribution of class membership. In each iteration, we shuffle the data. Then we use four fifths, or 80% of the data, to train the model. We use the remaining 20% to evaluate the performance of the trained model. Performance evaluation begins with feeding the trained model the test data. The model then assigns class membership probabilities to each instance of the test data. We refer to this assignment of probabilities as a classification. To complete evaluation of performance, we compute the AUPRC score of the classification. In every scenario, we perform ten iterations of five-fold cross validation, which results in 50 AUPRC scores, all of which are recorded for later analysis. Ten iterations are performed to mitigate the effects of random chance impacting experimental outcomes. Please see Fig. 1 for a flow chart of the ten iterations of five-fold cross validation process.

Calculation of the AUPRC score is also something in common to each scenario, since it is performed in each iteration of five-fold cross validation. The precision-recall curve represents the trade-off between precision and recall. Precision is defined as

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

where TP is the number of true positives and FP is the number of false positive. The definition of recall is

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

where FN is the number of false negatives. To calculate AUPRC, the threshold probability value for deciding class membership is successively reduced. For each threshold value, precision and recall values are plotted as points on a curve. There is a tendency for recall values to increase as the classification threshold decreases, due to a greater chance of an instance being assigned to the positive class. However, the increase in recall is often at the sacrifice of precision, also due to the higher chance of an instance being assigned to the positive class. Therefore, the multiple values of precision and recall recorded form a characteristic curve. The area under this curve is a composite metric, covering the model's proficiency over a range of threshold values. Numerical integration techniques are used to calculate the area under the curve.

We have covered the ten iterations of five-fold cross validation, and calculation of AUPRC, which are elements in common to all five scenarios. Now, we proceed to discuss the aspects that are unique for each scenario. The scenarios exhaust the ways one could apply RUS and Feature Selection as data preprocessing steps. The first scenario, which we call "Scenario One" is the scenario where we apply RUS only. While other sampling techniques are available, results documented in previous research on sampling techniques applied to highly imbalanced Big Data show that RUS yields the best performance [37]. Since RUS is only applied to the training data, it must be applied during each fold of five-fold cross validation. In order to apply RUS, we select a target class ratio of minority to majority instances, then we randomly remove instances of

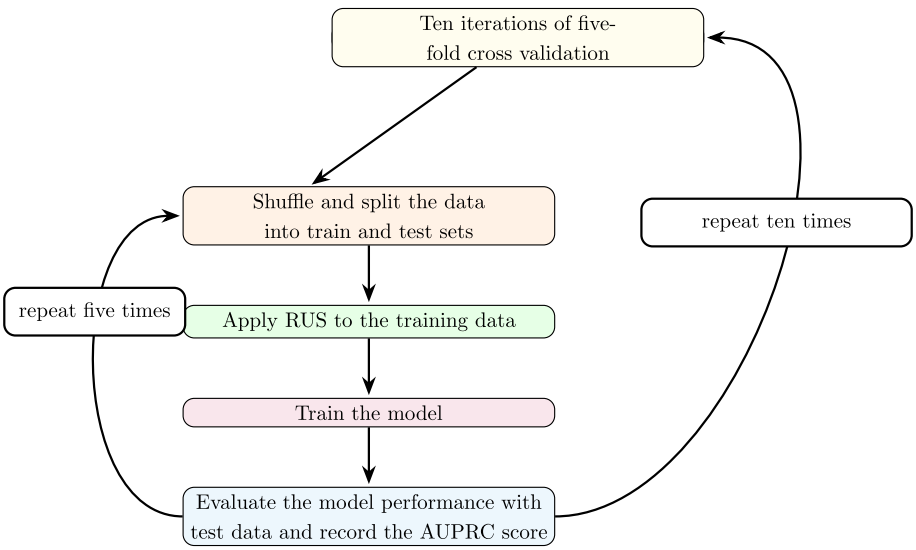

Fig. 2 Procedure for scenario one experiments, RUS only

the majority class until the target class ratio is reached. For the experiments in this study, we use target class ratios of 1:1, 1:3, 1:27, 1:81. Moreover, we include experiments where the data is left at its original class ratio to validate the effect of RUS. The stage in Scenario One experiments where RUS is applied is depicted in Fig. 2.

This concludes our discussion of Scenario One, now we move on to discuss Scenario Two. Since Scenario Two involves our supervised feature selection technique, we must describe it first. It is important to note that supervised feature selection takes place prior to the start of the ten iterations of five-fold cross validation. The crux of the feature selection process is a feature ranking technique. Our feature ranking technique leverages six learners to generate an ordered list of the features of a dataset. The six learners are CatBoost, XGBoost, LightGBM, Decision Tree, Random Forest, and ET. Each of these learners builds a feature importance list as a side effect of the training process. Therefore, we train the six learners to obtain six feature importance lists. To apply our feature selection technique we merge the six lists according to the following logic: we assign each feature the median value of its rank in each of the six feature importance lists. This places an order on the features in the dataset, and we can select a number of features in this order. Figure 3 depicts the feature ranking process.

Scenario Two experiments are experiments where we apply our novel, ensemble supervised feature selection technique as the only data preprocessing step. The feature ranking process must be performed only once per dataset. In Fig. 4, we show how feature selection fits into Scenario Two. At the beginning of Scenario Two, we reuse the work depicted in Fig. 3 by selecting the top k features from the list that the feature ranking process produces. For the Scenario Two with Part D data, we use this technique to build feature sets of sizes 7, 10, 15, 20, 25, and 30 features. We include experiments where all features are used as a check for the effectiveness of the feature selection technique. One may notice in our results for experiments with Part D data that we have feature

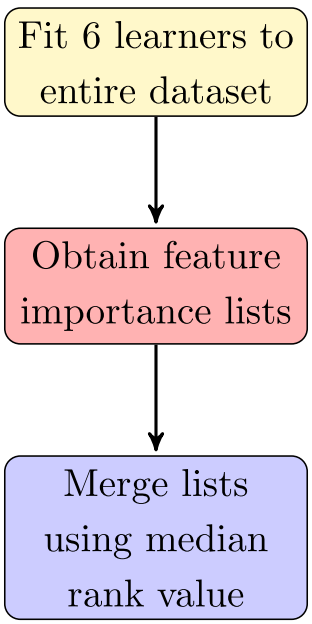

Fig. 3 Procedure for ensemble feature ranking

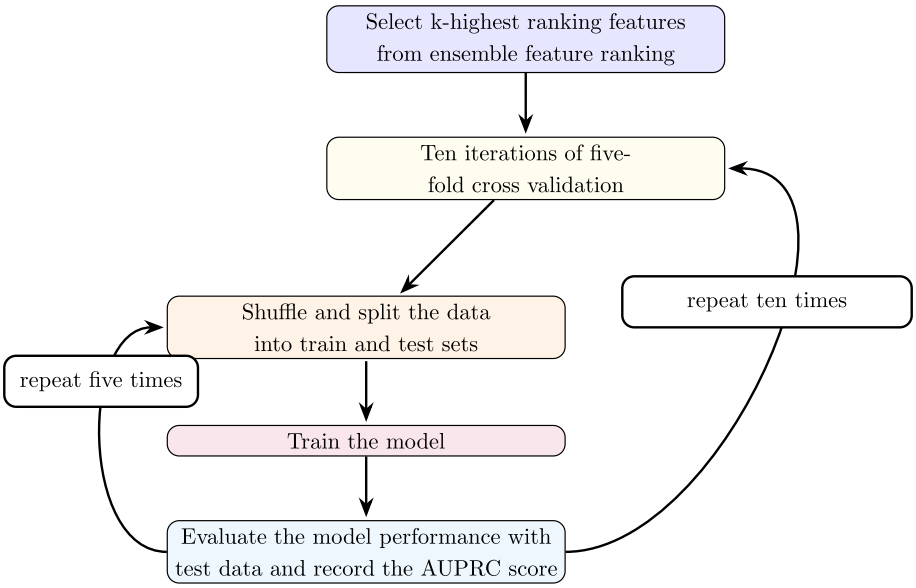

Fig. 4 Methodology for scenario two experiments, feature selection only

sets named "7a" and "7b". This is because it is possible for two features to be assigned the same rank, because they have the same median rank. Hence, we build datasets with seven features with one of each feature that has the same rank at position 7.

For experiments with Part B data, we use feature sets of size 10, 15, 20, 25, 30 as well as the set of all features. Experiments with Part D data showed that models built with ten features yielded performance that is significantly better than the performance of models built with any other number of features, hence the experiments with 7 features for the Part D data. For experiments with Part B data, models built with 10 features did not significantly outperform other models in terms of AUPRC scores, so it was not necessary to conduct experiments with fewer features.

The next scenario, Scenario Three is the first scenario where we use a combination of RUS and feature selection. In Scenario Three, we do feature selection first, then apply RUS. Before the start of the ten iterations of five-fold cross validation, the ensemble feature selection technique is applied to the dataset to rank the features. At the start of the experiment, a decision is made on how many of the highest ranking features will be used. Since our results from the Scenario Two experiments with the Part D data show that models built with 7 features do not outperform models built with ten features, for our Scenario Three experiments we use feature set sizes of 10, 15, 20, 25, 30, and all features. The same numbers of features are used for experiments with the Part D and the Part B data. Scenario Three experiments are similar to Scenario One experiments, since we apply RUS to the training data before the training step of each fold of five-fold cross validation. However, since the Scenario One experiments with the Part D and Part B data both show that the 1:81 class ratio yields the best performance, we only apply RUS to induce class ratios of 1:81 in the Scenario Three experiments. Scenario Three is depicted in Fig. 5.

In Scenario Four, we apply RUS prior to feature ranking. We refer to the modified feature ranking process as RUS prior to feature ranking. RUS is applied to the data

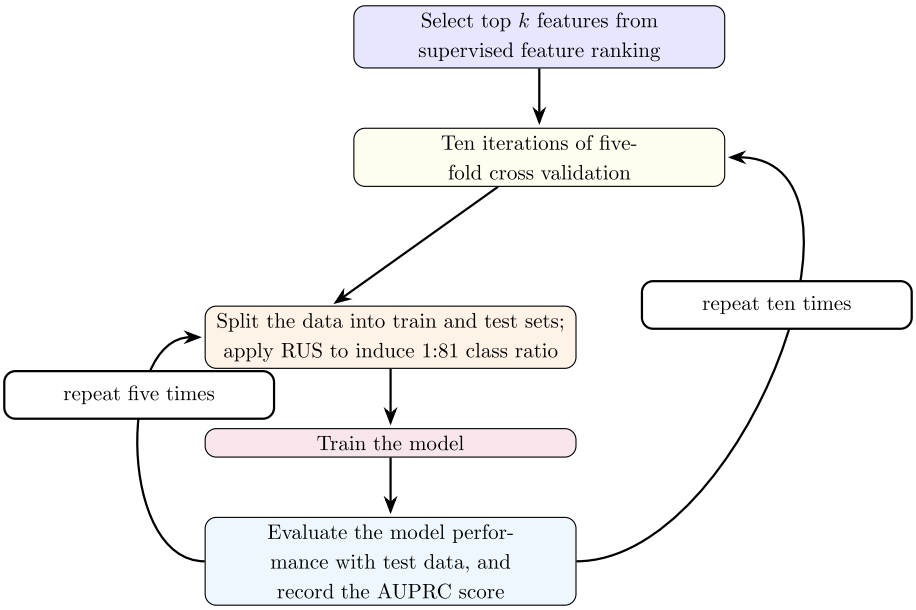

Fig. 5 Methodology scenario three experiments, for feature selection, then RUS

prior to the start of the ensemble feature ranking technique. We only apply RUS to induce a 1:81 class ratio in the data prior to feeding the data to the classifiers to obtain the feature importance lists. As stated previously, we use the 1:81 class ratio because we found that the 1:81 ratio yields the best performance in the Scenario One experiments. In order to perform RUS prior to feature selection we make a modification to the ensemble feature selection technique. We make the modification in order to mitigate the impact of RUS on the ensemble feature ranking process. The modification is that for each learner, we repeat the feature ranking process ten times. That is, we apply RUS to the data, to generate six ranked lists of all features, ten times. Then, in the merge step we merge all the ranked lists. In our study we use six learners, so we generate sixty ranked lists of features. We then merge the sixty ranked lists according to the same logic as the feature selection technique as it is described in figure 6.

After generating the ranked list of features from the RUS then feature selection process depicted in Fig. 6, we are ready to conduct the Scenario Four experiments. We use the same numbers of features, 10, 15, 20, 25, 30, and all features, for the Scenario Four experiments as used in the Scenario Three experiments. In these experiments, we perform RUS during cross validation. It is important to note that the features in each subset of features selected in Scenario Four may be different from the features selected in Scenario Two or Three because the feature ranking process for Scenario Four is different from that used in Scenarios Two and Three. Please see Fig. 7 for a graphical description of Scenario Four. In Scenario Four, RUS to induce a class ratio of 1:81 is also applied during the training phase as well. We induce the 1:81 class ratio since this ratio yields the best results in Scenario One experiments.

We have Scenario Five as a control so that we can be certain there is some benefit to our data reduction techniques. In Scenario Five, we apply no preprocessing

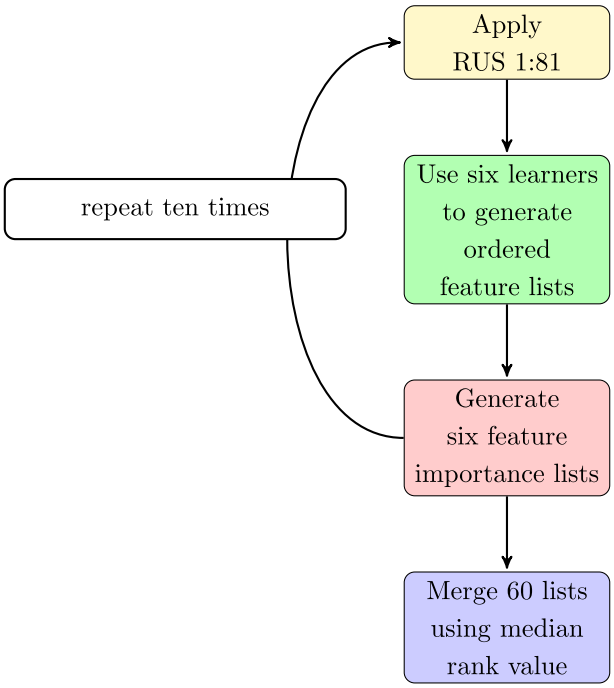

Fig. 6 Methodology for RUS followed by supervised feature selection

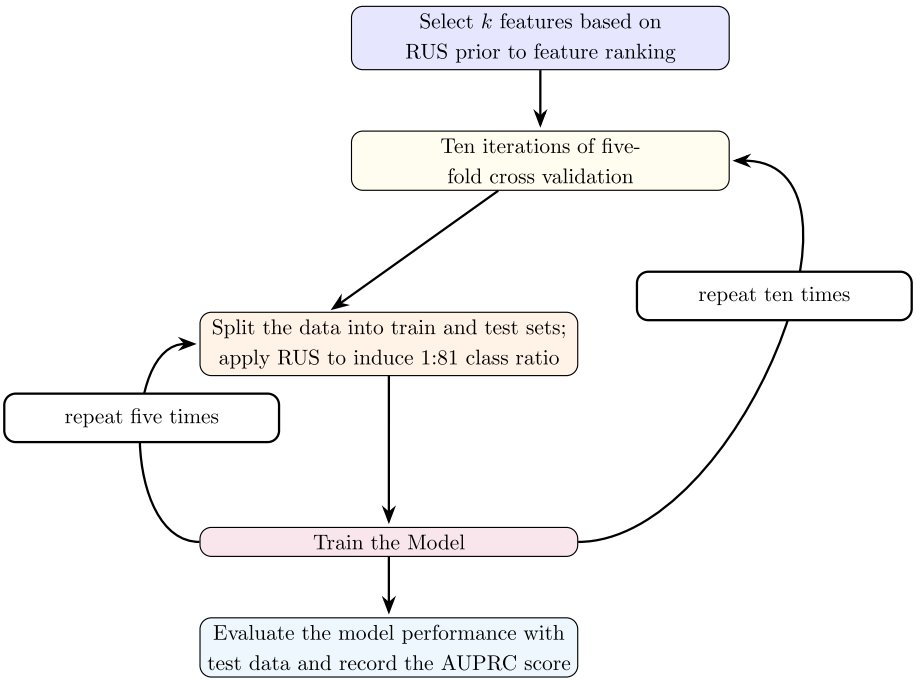

Fig. 7 Methodology scenario four experiments, for RUS then feature selection

to the data before using it to train the classifiers. Therefore, in Scenario Five, we do ten iterations of five-fold cross validation, with all features and the data at its original class ratio. Hence, Scenario Five is depicted in Fig. 1. This concludes our discussion of methodology, now we move on to present our results.

Table 7 Mean AUPRC values by classifier and induced class ratio for ten iterations of five-fold cross validation, for part D scenario one

Ratio Classifier	1:1	1:3	1:9	1:27	1:81	1:1,429
CatBoost	0.6304	0.7047	0.7498	0.7662	**0.7798**	0.7793
ET	0.0937	0.1199	0.1635	0.2029	0.2401	**0.3254**
LightGBM	0.5786	0.6588	0.7025	**0.7183**	0.6783	0.5132
Logistic regression	0.1189	0.1701	0.2173	0.2455	0.2700	**0.3060**
Random forest	0.1784	0.2208	0.2212	0.2240	0.2199	**0.2469**
XGBoost	0.6065	0.6768	0.7164	**0.7377**	0.7351	0.7372

The bold values indicates the maximum value for the classifier

Table 8 Mean AUPRC values by classifier and number of features (Part 1) for ten iterations of five-fold cross validation, for part D scenario two

Features classifier	7a	7b	8	9	10
CatBoost	0.7575	0.7582	0.7570	0.7558	**0.7585**
ET	0.5941	0.5171	0.5585	**0.6006**	0.5878
LightGBM	0.4548	0.4533	0.4689	0.5116	**0.5529**
Logistic Regression	0.3468	0.3368	0.3497	0.3482	**0.3537**
Random Forest	0.5873	0.4937	**0.5927**	0.5393	0.5903
XGBoost	0.7533	0.7539	0.7533	0.7514	**0.7571**

The bold values indicates the maximum value for the classifier

Results

In this section, we present results for classification experiments. All the numeric figures in tables in this section are the mean values of ten iterations of five-fold cross validation. We present classification results first for experiments on the Part D data, and then for experiments with the Part B data. We have subsections for Scenarios One through Four. Results for Scenario Five are included as the columns in the tables for Scenario One where the original class ratios are used. In the tables of this section, the figures in bold text indicate the maximum value for the classifier.

Part D scenario one: RUS only

Table 7 holds results from experiments where we vary the induced class ratio with RUS. In Table 7, we notice an upward trend in AUPRC scores as the class ratio approaches the original ratio. However, there are some cases, such as that of LightGBM's, when experiments where RUS is applied yield higher AUPRC scores than cases where RUS is not applied. As can be seen in Table 7 for LightGBM, this is the 1:27 ratio. The substantial improvement in LightGBM's performance is significant, and highlights the importance of including RUS in experiments with highly imbalanced Big Data for evaluating classifiers.

Part D scenario two: feature selection only

The experimental outcomes in terms of AUPRC are listed in Tables 8 and 9. Interestingly, the classifiers yield higher AUPRC scores when feature selection is applied.

Table 9 Mean AUPRC values by classifier and number of features (Part 2) for ten iterations of five-fold cross validation, for part D scenario two

Features classifier	15	20	25	30	82
CatBoost	**0.8016**	0.7953	0.7962	0.7949	0.7797
ET	**0.4954**	0.4647	0.4605	0.4391	0.3275
LightGBM	0.4447	0.4661	0.4603	0.4841	**0.4982**
Logistic regression	**0.3669**	0.3536	0.3519	0.2939	0.3047
Random forest	**0.6097**	0.5398	0.5519	0.5249	0.2429
XGBoost	**0.7889**	0.7448	0.7589	0.7548	0.7376

The bold values indicates the maximum value for the classifier

Table 10 Mean AUPRC values by classifier and number of features for ten iterations of five-fold cross validation, for part D scenario three

Features classifier	10	15	20	25	30	82
CatBoost	0.7589	**0.8051**	0.7973	0.7984	0.7938	0.7798
ET	**0.4986**	0.4151	0.3933	0.4009	0.3668	0.2401
LightGBM	0.7070	**0.7400**	0.7100	0.7019	0.6937	0.6783
Logistic regression	0.3117	**0.3189**	0.3117	0.3170	0.2486	0.2700
Random forest	**0.4820**	0.4753	0.3983	0.4120	0.3545	0.2199
XGBoost	0.7473	**0.7860**	0.7588	0.7554	0.7491	0.7351

The bold values indicates the maximum value for the classifier

CatBoost, Random Forest, Logistic Regression, and XGBoost yield the highest AUPRC scores with 15 features. ET yields the highest AUPRC score with nine features. LightGBM yields the best performance with ten features. Therefore, the results in Tables 8 and 9 are strong evidence that feature selection can improve the performance of classification results with Medicare Part D data.

Part D scenario 3: feature Selection, then RUS 1:81

Table 10 contains the mean AUPRC scores for the same experiments where we perform feature selection, then sample the training data to induce a 1:81 class ratio. It is interesting to note that all classifiers trained on preprocessed data yield higher AUPRC scores when classifying data in the test set than classifiers trained on the original data. In both the Scenario two and Scenario Three results we see better performance with fewer features. Models with fewer features are easier to explain because they are simpler and there is a reduced chance for complex interactions.

Part D scenario four: RUS 1:81 then feature selection

The last results we report are for experiments performed with another hybrid approach. The experiments in this scenario use RUS and feature selection in a different combination that in Scenario Three. In Scenario Four, data is sampled to a 1:81 level, then supervised feature selection is applied to rank features. A clear impact of the preprocessing treatment is apparent in terms of the AUPRC scores. In Table 11, we see that all classifiers exhibit a response to the preprocessing treatment. Models trained on the preprocessed data yield higher scores than models trained on the original dataset.

Table 11 Mean AUPRC values by classifier and number of features for ten iterations of five-fold cross validation, for part D scenario four

Features classifier	10	15	20	25	30	82
CatBoost	0.7546	**0.7992**	0.7914	0.7965	0.7926	0.7798
ET	**0.4765**	0.4228	0.3857	0.3967	0.3639	0.2401
LightGBM	0.7073	**0.7268**	0.6971	0.6974	0.6857	0.6783
Logistic regression	0.2609	**0.2785**	0.2358	0.2461	0.2613	0.2700
Random forest	0.4209	**0.4639**	0.3311	0.3553	0.3392	0.2199
XGBoost	0.7471	**0.7743**	0.7550	0.7524	0.7476	0.7351

The bold values indicates the maximum value for the classifier

Part B scenario one: RUS only

Here, we report the results of experiments in which we repeated the scenarios we conducted in experiments with the Part D data, with the Part B data. Table 12 holds the AUPRC scores that are the outcomes of experiments where we induce various class ratios in the training data. Here the impact of the treatment is not as clear, however, there is an improvement in the performance of LightGBM when trained on the preprocessed data over LightGBM when trained with data at its original class ratio.

Part B scenario two: feature selection only

Table 13 contains the AUPRC scores that are the result of training models on the Part B data when it is preprocessed with feature selection. We find it is important to point out that all classifiers yield better performance when trained on fewer than all features.

Table 12 Mean AUPRC values by classifier and induced class ratio for ten iterations of five-fold cross validation, for part B scenario one

Ratio classifier	1:1	1:3	1:9	1:27	1:81	1:2,500
CatBoost	0.4428	0.5320	0.6228	0.6569	0.6812	**0.6817**
ET	0.0125	0.0135	0.0184	0.0272	0.0336	**0.0433**
LightGBM	0.4001	0.4859	0.5563	**0.5967**	0.5766	0.4146
Logistic regression	0.0058	0.0069	0.0076	0.0086	0.0099	**0.0103**
Random forest	0.0791	0.1210	0.1596	0.1829	0.2017	**0.2462**
XGBoost	0.4240	0.5104	0.5783	0.6234	0.6536	**0.6886**

The bold values indicates the maximum value for the classifier

Table 13 Mean AUPRC values by classifier and number of features for ten iterations of five-fold cross validation, for part B scenario two

Features classifier	10	15	20	25	30	80
CatBoost	0.6581	0.6792	**0.7069**	0.7009	0.7016	0.6817
ET	0.0400	0.0462	0.0443	**0.0524**	0.0424	0.0433
LightGBM	0.3939	0.3830	0.4261	**0.4589**	0.4293	0.4146
Logistic regression	0.0093	0.0326	**0.0338**	0.0065	0.0064	0.0103
Random forest	**0.4356**	0.3990	0.3736	0.3800	0.3395	0.2462
XGBoost	0.6611	0.6715	**0.6995**	0.6956	0.6955	0.6886

The bold values indicates the maximum value for the classifier

Table 14 Mean AUPRC values by classifier and number of features for ten iterations of five-fold cross validation, for part B scenario three

Features classifier	10	15	20	25	30	80
CatBoost	0.6600	0.6850	**0.7143**	0.7047	0.7023	0.6812
ET	0.0233	0.0261	0.0317	**0.0429**	0.0358	0.0336
LightGBM	0.5756	0.5904	**0.6185**	0.6009	0.6030	0.5766
Logistic regression	0.0076	0.0205	**0.0217**	0.0074	0.0076	0.0099
Random forest	0.2820	0.2758	0.3168	**0.3225**	0.2990	0.2017
XGBoost	0.6436	0.6535	**0.6798**	0.6708	0.6681	0.6536

The bold values indicates the maximum value for the classifier

Table 15 Mean AUPRC values by classifier and number of features for ten iterations of five-fold cross validation, for part B scenario four

Features classifier	10	15	20	25	30	80
CatBoost	0.6787	0.6725	**0.6994**	0.6978	0.6975	0.6812
ET	0.0289	0.0352	0.0495	**0.0512**	0.0462	0.0336
LightGBM	0.5968	0.5803	**0.6063**	0.5935	0.5938	0.5766
Logistic regression	0.0078	0.0065	0.0067	0.0069	0.0090	**0.0099**
Random forest	**0.3313**	0.3036	0.3161	0.3120	0.2892	0.2017
XGBoost	0.6560	0.6406	0.6644	0.6630	**0.6662**	0.6536

The bold values indicates the maximum value for the classifier

Part B scenario three: feature selection then RUS 1:81

Table 14 contains the AUPRC scores from the results of experiments in Scenario Three. In Scenario Three, Part B data that is preprocessed by feature selection, followed by RUS. The results in Table 14 show that all models respond well to being trained with data that is preprocessed with the Scenario Three approach. That is to say, the maximum scores each classifier yields is when the classifier is trained with the preprocessed data.

Part B scenario four: RUS 1:81 then feature selection

Lastly for the Part B data, we review the AUPRC scores of classifiers trained on data which is first undersampled to the 1:81 class ratio, and then feature selection is applied. An inspection of these scores in Table 15 reveals that all models yield higher AUPRC scores in the classification of the training data when they are trained on data preprocessed with the Scenario Four technique. We find this result is noteworthy and should be considered for further analysis.

Statistical analysis

In the Statistical analysis that follows, we first present an analysis of results for the four scenarios separately to determine which levels of experimental factors yield the best performance. Then, we do a combined analysis of the outcomes of each scenario to determine which scenario(s) yield the best performance. We coin the term "inter-scenario analysis" for the combined analysis.

Table 16 ANOVA for ratio and classifier as factors of performance in terms of AUPRC, for part D scenario one

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Ratio	5	3.34	0.67	239.59	*
Classifier	5	106.28	21.26	7614.63	*
Residuals	1789	4.99	0.00		

* indicates the value is less than 1×10^{-4}

Table 17 HSD test groupings after ANOVA of AUPRC for the Ratio factor, for part D scenario one

Group a consists of: 1:81, 1:1,429, 1:27
Group b consists of: 1:9
Group c consists of: 1:3
Group d consists of: 1:1

Part D scenario one: RUS only

The first analysis we perform is for Scenario One, where we do RUS only. Our methodology for statistical analysis is as follows: first we conduct an analysis of variance ANOVA test to determine whether the experimental factors have a significant impact on experimental outcomes [38]. We set a significance level of $\alpha = 0.01$ for the ANOVA test so that if the Pr(>F), or p-value of the significance test is less than or equal to 0.01, we reject the null hypothesis that the factor has no impact on experimental outcomes. If we reject the null hypothesis, then we perform a Tukey's Honestly Significant Difference (HSD) test to rank the levels of experimental factors in terms of their impact on the experimental outcomes [39]. The outcome of the Tukey HSD test is a labeling of experimental factors such that the alphabetical order of the label on the factor corresponds to the experimental outcome it is associated with. Therefore, the factors in the group labeled 'a' are associated with the highest values of experimental outcomes, and so on. For the purpose of this study, an experimental outcome is the mean AUPRC score recorded as the outcome of ten iterations of five-fold cross validation for a particular combination of experimental factors.

Table 16 holds the result of the ANOVA test for experiments in Scenario One. In Scenario One we select a classifier and a class ratio for the training data. As stated previously, we use six classifiers in our study, therefore, there are 5 degrees of freedom (DF) in Table 16. Similarly, since we use six different class ratios in our experiments, DF for the Ratio Factor in Table 16 is also 5.

Since the Pr(>F) values for both the ratio and classifier factors in Table 16 are less than our selected significance level of 0.01, according to our statistical analysis procedure, we conduct Tukey HSD tests to group the levels of the factors by their performance. Table 17 contains the HSD test result for the ratio factor in the experiments in Scenario One. The result shows that there is no significant difference in the AUPRC scores of models built with data that is not preprocessed with RUS and models built with data that is preprocessed with RUS to make the class ratio 1:81 or 1:27. Hence, we find that that RUS, a data reduction technique, maintains the best performance.

Table 18 contains the HSD result for the classifier factor. The HSD test reveals that CatBoost yields the best performance in Scenario One. Moreover, we would like to point out that the three best performing classifiers, CatBoost, XGBoost, and LightGBM, are all GBDT techniques.

Part D scenario two: feature selection Only

The next statistical analysis we perform is for Scenario Two, where we use our ensemble feature selection technique, by itself. Table 19 contains the ANOVA test result for the impact of the number of features (Features), and the choice of classifier (Classifier) factors on AUPRC scores. The Pr(>F) value for both factors is practically zero, so we conclude that both factors have a significant effect on experimental outcomes.

Since the ANOVA test shows that both the classifier and number of features have a significant effect on experimental outcomes, we run HSD tests to determine which levels of the factors are associated with the highest AUPRC scores. The HSD result in Table 20 show that the use of ten features is associated with the best performance. Similar to the result for Scenario One, we find that an additional data reduction technique of feature selection also improves performance.

The HSD result for the classifier factor is in Table 21. Similar to the HSD result in Scenario One, the HSD result shows CatBoost and XGBoost yield the best performance. However, unlike the result for Scenario One, the third GBDT classifier, LightGBM, is not among the top three performers.

Table 18 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part D scenario one

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: Logistic Regression, Random Forest
Group e consists of: ET

Table 19 ANOVA for Features and Classifier as factors of performance in terms of AUPRC, for part D scenario two

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	9	2.97	0.33	47.15	*
Classifier	5	71.64	14.33	2050.15	*
Residuals	2985	20.86	0.01		

* indicates the value is less than 1×10^{-4}

Table 20 HSD test groupings after ANOVA of AUPRC for the Features factor, for part D scenario two

Group a consists of: '10'
Group ab consists of: '15', '9', '7a', '8'
Group bc consists of: '25', '20'
Group c consists of: '7b', '30'
Group d consists of: '82'

Table 21 HSD test groupings after ANOVA of AUPRC for the Classifier factor, for Part D Scenario Two

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: Random Forest
Group d consists of: ET
Group e consists of: LightGBM
Group f consists of: Logistic Regression

Part D scenario 3: feature selection, then RUS 1:81

Now we move on to the statistical analysis of the Scenario Three experiments with the Part D data, where we do feature selection, then apply RUS to induce a 1:81 class ratio in the training data. The ANOVA test result in Table 22 is similar to the results for Scenarios One and Two in that the Pr(>) F values both factors is practically zero. Since we use only one sampling ratio in Scenario Three, only the choice of classifier and number of features are treated as factors in the ANOVA test.

Since the ANOVA test shows that both the choice of classifier, and the number of features selected have a significant impact on AUPRC scores, we do the HSD tests to determine which levels of the factors yield the best performance. Table 23 contains the HSD result for the number of features factor. In it we see that, similar to the result in Scenario Two, models built with ten features are in the group that yields the best performance. This is a noteworthy result, since it demonstrates that a data reduction technique can yield better performance. In the results of both Scenario Two and Scenario Three, models with fewer features demonstrated superior performance. Such models are more straightforward, making them easier to interpret due to their inherent simplicity and decreased potential for intricate interactions.

Next we move on to the HSD result for the classifier factor. Here we see that the three GBDT techniques, CatBoost, XGBoost, and LightGBM yield the best performance. It is interesting to note that this is similar to the HSD test result for the classifier factor in Scenario One, and that in the experiments in Scenario One and Scenario Three, RUS is applied to the training data (Table 24).

Table 22 ANOVA for Features and Classifier as factors of performance in terms of AUPRC, for part D scenario three

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	5	2.17	0.43	178.58	*
Classifier	5	72.01	14.40	5933.71	*
Residuals	1789	4.34	0.00		

* indicates the value is less than 1×10^{-4}

Table 23 HSD test groupings after ANOVA of AUPRC for the Features factor, for part D scenario three

Group a consists of: '15', '10'
Group b consists of: '25', '20'
Group c consists of: '30'
Group d consists of: '82'

Table 24 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part D scenario three

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: Random Forest, ET
Group e consists of: Logistic Regression

Table 25 ANOVA for features and classifier as factors of performance in terms of AUPRC, for part D scenario four

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	5	1.43	0.29	127.51	*
Classifier	5	80.72	16.14	7209.10	*
Residuals	1789	4.01	0.00		

* indicates the value is less than 1×10^{-4}

Table 26 HSD test groupings after ANOVA of AUPRC for the Features factor for, part D scenario four

Group a consists of: '15'
Group b consists of: '10'
Group c consists of: '25', '20', '30'
Group d consists of: '82'

Table 27 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part D scenario four

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: ET
Group e consists of: Random Forest
Group f consists of: Logistic Regression

Part D Scenario Four: RUS 1:81 then feature selection

Finally, we come to the last scenario for the experiments with Part D data. Table 25 contains the result of the ANOVA test for the Scenario Four experiments. Similar to Scenario Three, we only use one level of undersampling, 1:81. Therefore, we treat the number of features, and the choice of classifiers as experimental factors. The Pr(>F) values indicate that both factors have a significant effect on AUPRC scores.

Since the ANOVA test for Scenario Four indicates that the number of features used to train the model has a significant effect on experimental outcomes, we use the HSD test to determine which number of features yields the best performance. For the Scenario Four experiments, we see in Table 26, that 15 features yields the best performance.

The HSD test for experiments with Part D data is documented in Table 27. Again we have the result that CatBoost yields the best performance, and the three GBDT techniques yield the top three mean AUPRC scores.

Part D, inter-scenario analysis

Now that we have documented the outcomes of the individual scenarios, we do a further analysis to determine which scenario(s) yield the best performance over all. To get started we take note of which levels of factors yield the best performance in a scenario. We then select data from experiments to include for this analysis based on which experiments included factors set at the levels that yield the best performance. For Scenario One, RUS only, we select RUS 1:81. For Scenario Two, feature selection only, we select experiments where ten features are used. For Scenario Three, feature selection, then RUS, we select experiments where 10 features are used, and RUS 1:81 is used. For Scenario Four, RUS, then feature selection, we select experiments where RUS 1:81 is used, and 10 features are used. We then treat each scenario, and the classifier as an experimental factor and conduct an ANOVA test, followed by HSD tests to determine which scenario(s) yield the best performance, and which classifiers yield the best performance. We consider the case where no preprocessing is done to the data to be the fifth scenario. Table 28 summarizes the levels of factors used in all scenarios.

Table 29 contains the result of the ANOVA test for the scenario and classifier factors. The Pr(>F) values indicate that both the scenario, and the classifier have a significant impact on experimental outcomes.

The ANOVA test result in Table 29 indicates that the scenario has a significant impact on experimental outcomes. Therefore, we can conduct a Tukey HSD test to determine which scenario yields the highest AUPRC scores. The HSD test result in Table 30 indicates that either feature selection, or feature selection followed by RUS yield the best performance.

Table 28 Summary of part D scenarios, and optimal levels of features selected from each scenario

Scenario	Description
Scenario One	RUS only: RUS 1:81 selected
Scenario Two	Feature selection only: 10 features selected
Scenario Three	Feature selection, then RUS: 10 features and RUS 1:81 selected
Scenario Four	RUS, then feature selection: RUS 1:81 and 10 features selected
Scenario Five	No preprocessing done on the data

Table 29 ANOVA for scenario and classifier as factors of performance in terms of AUPRC, for part D experiments

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Scenario	4	3.65	0.91	125.08	*
Classifier	5	48.41	9.68	1328.40	*
Residuals	1490	10.86	0.01		

* indicates the value is less than 1×10^{-4}

Table 30 HSD test groupings after ANOVA of AUPRC for the Scenario factor in part D experiments

Group a consists of: Scenario Two, Scenario Three
Group b consists of: Scenario Four
Group c consists of: Scenario One, Scenario Five

Table 31 contains the HSD test result for the classifier factor. Since we find that for the majority of the scenarios individually that the GBDT classifiers yield the best performance, it is not surprising that the three GBDT classifiers are members of the best-performing groups.

Part B statistical analysis

We conduct a statistical analysis similar to the one performed here, for the Part B data. Please see Appendix A for the report of the analysis. After performing the analysis for each scenario, and then the inter-scenario analysis for the Part B data, we determined that both feature selection, followed by RUS, and RUS followed by feature selection yield the best performance. These two scenarios are by themselves in the HSD group 'a' for the scenario factor in the inter-scenario analysis for experiments with the Part B data. Moreover, the analysis of Part B experiments confirms that the GBDT classifiers yield the best performance, with CatBoost the best among all learners.

Conclusions

We presented an in-depth statistical analysis of the experimental outcomes in terms of AUPRC for results for Medicare insurance fraud detection in Big Medicare Data datasets. For both Medicare Part B and Part D datasets, we carry out experiments in five scenarios that exhaust the possible ways to utilize, or omit, the RUS and feature selection data reduction techniques. For both datasets, we found that data reduction techniques also improve classification results. We also found that the three GBDT classifiers, LightGBM, XGBoost, and CatBoost yield the best performance in all experiments, with one exception. In any case, CatBoost consistently yields the best performance.

In experiments with the Part D data, we conducted separate statistical analyses of the four scenarios where at least one data reduction technique was used, to determine which technique yields the best performance in each scenario. We then conducted a statistical analysis of results between all scenarios. The result of the analysis shows that our supervised feature selection technique alone, or our feature selection technique, followed by RUS, yields the best performance.

We performed a similar statistical analysis for experiments involving the Medicare Part B data. After doing the analysis of the individual scenarios, we again selected the techniques which yielded the best results, and performed an analysis of results between all scenarios. We find that either combination of using our feature selection technique, followed by RUS, or RUS followed by our feature selection technique both yield the best performance.

Therefore, in the classification of either dataset, we find that a technique with the largest amount of data reduction also yields the best performance. That is the technique of doing feature selection, then applying RUS. The same statistical analysis

Table 31 HSD test groupings after ANOVA of AUPRC for the classifier factor in part D experiments

Group a consists of: CatBoost, XGBoost
Group b consists of: LightGBM
Group c consists of: ET
Group d consists of: Random Forest
Group e consists of: Logistic Regression

shows that the GBDT classification techniques, XGBoost, CatBoost and LightGBM outperform the other three learners we use in our experiments. These other learners are Logistic Regression, Random Forest, and Extremely Randomized trees. An added benefit of our feature selection technique is model explainability. It is easier to reason about how a model performs classifications when it is built with fewer features. Overall, the key conclusion one should draw from our results is that intelligent data reduction techniques, applied in combination, may improve the results in classifying highly imbalanced, Big Data.

Appendix A statistical analysis of experiments with part B Data

Here we report individual scenario results for the experiments with Part B data. We do the analysis of scenarios in the same order for experiments with Part B data as we do for the Part D data, so we start with the scenario where only RUS is applied to preprocess the data. Table 32 contains the ANOVA test result for the Part B Scenario One experiments. The ANOVA test result shows that both the classifier and RUS ratio have a significant impact on experimental outcomes.

Since both the RUS ratio and the choice of classifier have a significant impact on AUPRC scores, we perform HSD tests to determine which levels of the factors are associated with the highest AUPRC scores. The HSD result in Table 33 indicates that classifiers trained on data preprocessed with RUS to induce class ratios of 1:81, 1:27, or the original data all yield the highest AUPRC scores.

Next we look at the HSD result for the analysis of the classifier factor in Scenario One. The result for the classifier factor continues the same trend we see in the majority of scenarios for Part D data as well. The GBDT techniques yield the best performance, and CatBoost yields the best performance over all (Table 34).

Part B scenario 2: feature selection only

Next we do the analysis for the Part B Scenario Two experiments, where only feature selection is performed as a preprocessing step. The ANOVA test result in Table 35

Table 32 ANOVA for ratio and classifier as factors of performance in terms of AUPRC, for part B scenario one

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Ratio	5	3.99	0.80	259.76	*
Classifier	5	116.98	23.40	7613.74	*
Residuals	1789	5.50	0.00		

* indicates the value is less than 1×10^{-4}

Table 33 HSD test groupings after ANOVA of AUPRC for the Ratio factor, for part B scenario one

Group a consists of: 1:81, 1:27, 1:2,500
Group b consists of: 1:9
Group c consists of: 1:3
Group d consists of: 1:1

Table 34 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part B scenario one

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: Random Forest
Group e consists of: ET
Group f consists of: Logistic Regression

shows that both the number of features used to train a classifier, and the choice of classifier have a significant effect on experimental outcomes.

Since the ANOVA test indicates that both the classifier and the number of features used have a significant effect on experimental outcomes, we conduct HSD tests to determine which levels of these factors yield the best performance. The HSD test result in Table 36 indicates that any level of feature selection yields the best performance, whereas using all features yields significantly worse performance.

The second HSD test we perform for Scenario Two is to determine which classifiers yield the best performance in the Part B Scenario Two experiments. As it has been the case for most scenarios we have analyzed thus far, the three GBDT techniques yield the best performance. This time CatBoost and XGBoost both yield the best performance, since they are both in HSD group 'a'(Table 37)

Part B scenario three, feature selection, then RUS

We continue to the first hybrid approach for experiments with Part B data, where we apply feature selection, then RUS. Although we apply RUS to make the class ratio 1:81 in Scenario Three, RUS is not an experimental factor since it does not change

Table 35 ANOVA for Features and Classifier as factors of performance in terms of AUPRC, for Part B Scenario Two

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	5	0.24	0.05	13.16	*
Classifier	5	130.10	26.02	7242.04	*
Residuals	1789	6.43	0.00		

* indicates the value is less than 1×10^{-4}

Table 36 HSD test groupings after ANOVA of AUPRC for the features factor, for part B scenario two

Group a consists of: '25', '20', '30', '15', '10'
Group b consists of: '80'

Table 37 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part B scenario two

Group a consists of: CatBoost, XGBoost
Group b consists of: LightGBM
Group c consists of: Random Forest
Group d consists of: ET
Group e consists of: Logistic Regression

throughout the course of the Part B Scenario Three experiments. Since we build models with different numbers of features and different classifiers, these are experimental factors. The ANOVA test results in Table 38 show that both the choice of classifier and the number of features selected have a significant impact on experimental outcomes.

Since the ANOVA test result in Table 38 shows that the number of features selected has a significant impact on the AUPRC scores recorded in the Part B Scenario Three experiments, we perform an HSD test to determine which number of features can be used to build models that yield the best performance. The HSD test result in Table 39 indicates that models built with 20 features yield the best performance, since this set of experiments is in the HSD group 'a'.

Next, we turn to the result of the HSD test for the effect of the classifier on experimental outcomes in the Part B Scenario Three experiments. The result confirms the pattern we have seen deviated from only once. That pattern is that the GBDT methods yield the best AUPRC scores, with CatBoost yielding the best AUPRC scores (Table 40).

Part B, scenario four, RUS 1:81, then feature selection

Here we proceed to the analysis of the last of the Scenario Four experimental results with Part B data. This is the scenario where we apply RUS to induce a class ratio of 1:81 prior to doing feature selection.

Table 38 ANOVA for features and classifier as factors of performance in terms of AUPRC, for part B scenario three

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	5	0.34	0.07	61.20	*
Classifier	5	146.24	29.25	26678.81	*
Residuals	1789	1.96	0.00		

* indicates the value is less than 1×10^{-4}

Table 39 HSD test groupings after ANOVA of AUPRC for the features factor, for part B scenario three

Group a consists of: '20'
Group ab consists of: '25'
Group b consists of: '30'
Group c consists of: '15'
Group d consists of: '10', '80'

Table 40 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part B scenario three

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: Random Forest
Group e consists of: ET
Group f consists of: Logistic Regression

Following our procedure for statistical analysis, we conduct an ANOVA test to determine which experimental factors have a significant effect on experimental outcomes. In Table 38, we confirm that both the choice of classifier and the number of features used in the experiment have a significant impact on experimental outcomes. Although RUS is employed prior to feature selection, it is not an experimental factor since we only apply RUS to induce a single class ratio.

The ANOVA test result in Table 41 indicates that the number of features used has a significant impact on experimental outcomes, so we conduct an HSD test. The test result is in Table 42. The test result indicates that RUS followed by feature selection with any number of features, except 15, yields the best performance, and also that applying RUS and then feature selection yields better performance than not applying feature selection.

The second HSD test we conduct for the Part B Scenario Four experiments is for the effect of the classifier on experimental outcomes. In Table 43 we confirm the clear pattern in the HSD results we have observed in all but one scenario, and that is that the GBDT techniques outperform all others, and CatBoost yields the best performance.

Part B inter-scenario analysis

Here we begin the analysis of results between scenarios. We do the same inter-scenario analysis for the Part B scenarios that we did for the Part D scenarios. As stated previously in the context of the Part D data inter-scenario analysis, though we report

Table 41 ANOVA for features and classifier as factors of performance in terms of AUPRC, for part B scenario four

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Features	5	0.20	0.04	33.72	*
Classifier	5	143.27	28.65	24577.10	*
Residuals	1789	2.09	0.00		

* indicates the value is less than 1×10^{-4}

Table 42 HSD test groupings after ANOVA of AUPRC for the features factor, for part B scenario four

Group a consists of: '20', '25', '30', '10'
Group b consists of: '15'
Group c consists of: '80'

Table 43 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part B scenario four

Group a consists of: CatBoost
Group b consists of: XGBoost
Group c consists of: LightGBM
Group d consists of: Random Forest
Group e consists of: ET
Group f consists of: Logistic Regression

results for four scenarios, there is a latent fifth scenario, which is the case where do not do any preprocessing to the data. Also, similar to the Part D inter-scenario analysis, we select the levels of factors that yield the best performance in each scenario. Table 44 contains a summary of the levels of factors selected for each scenario.

First, we conduct an ANOVA test to confirm that the scenario and the choice of classifier have a significant impact on experimental outcomes. The Pr(>F) values for both the scenario and classifier factors in Table 45 imply that both factors have a significant impact on AUPRC values.

Since we have confirmed that both the scenario and the classifier have a significant impact on experimental outcomes, we conduct an HSD test to determine which levels of the factors yield the best performance. The HSD test result in Table 46 implies that the two hybrid approaches yield the best performance. This is noteworthy since it means the two techniques that yield the largest data reduction also yield the strongest performance.

Finally, we take a look at the classifiers that perform the best over all scenarios. Here, we find CatBoost and XGBoost yield the best performance, followed by LightGBM. The consistent pattern we find in the results for the individual scenarios in Part B carries over into the results for the Part B inter-scenario analysis (Table 47)

Table 44 Summary of Part B Scenarios, and optimal levels of features selected from each scenario

Scenario	Description
Scenario One	RUS only: RUS 1:81 selected
Scenario Two	Feature selection only: 10 features selected
Scenario Three	Feature selection, then RUS: 20 features and RUS 1:81 selected
Scenario Four	RUS, then feature selection: RUS 1:81 and 10 features selected
Scenario Five	No preprocessing done on the data

Table 45 ANOVA for scenario and classifier as factors of performance in terms of AUPRC, for part B experiments

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Scenario	4	0.46	0.12	28.70	*
Classifier	5	113.33	22.67	5621.43	*
Residuals	1490	6.01	0.00		

* indicates the value is less than 1×10^{-4}

Table 46 HSD test groupings after ANOVA of AUPRC for the scenario factor, for part B experiments

Group a consists of: Scenario Three, Scenario Four
Group b consists of: Scenario Two
Group bc consists of: Scenario One
Group c consists of: Scenario Five

Table 47 HSD test groupings after ANOVA of AUPRC for the classifier factor, for part B experiments

Group a consists of: CatBoost, XGBoost
Group b consists of: LightGBM
Group c consists of: Random Forest
Group d consists of: ET
Group e consists of: Logistic Regression

Abbreviations

ANOVA	Analysis of variance
CMS	Centers for Medicare and Medicaid Services
ET	Extremely Randomized Trees
GBDT	Gradient Bosted Decision Trees
HCC	Hierarchical Condition Categories
HCPCS	Healthcare Common Procedure Coding System
HSD	Honestly Significant Difference
LA	Long-acting
LEIE	List of Excluded Individuals and Entities
LIS	Low-Income Subsidy
MAPD	Medicare Advantage Prescription Drug Plan
NPI	National Provider Identifier
OIG	Office of the Inspector General
PDP	Prescription Drug Plan
ROS	Random Oversampling
RUS	Random Undersampling

Acknowledgements
The authors would like to thank the various members of the Data Mining and Machine Learning Laboratory, Florida Atlantic University, for their assistance with the reviews.

Author contributions
JTH and QL conducted experiments, and contributed to the manuscript. HW contributed to the manuscript. TMK provided oversight of experiments, coordinated research, and contributed to the manuscript.

Funding
Not applicable.

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 22 September 2023 Accepted: 14 December 2023
Published online: 03 January 2024

References

1. Boyd K, Eng KH, Page CD. Area under the precision-recall curve: point estimates and confidence intervals. Joint European conference on machine learning and knowledge discovery in databases, 451–466. Springer 2013
2. Bekkar M, Djemaa HK, Alitouche TA. Evaluation measures for models assessment over imbalanced data sets. J Inf Eng Appl. 2013;3(10).
3. Hancock JT, Khoshgoftaar TM, Johnson JM. Evaluating classifier performance with highly imbalanced big data. J Big Data. 2023;10(1):42.
4. Hancock J, Khoshgoftaar TM, Johnson JM. Informative evaluation metrics for highly imbalanced big data classification. In: 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1419–1426, 2022
5. Civil Division, U.S. Department of Justice: Fraud Statistics, Overview. https://www.justice.gov/opa/press-release/file/1354316/download, 2020
6. Centers for Medicare and Medicaid Services: 2019 Estimated Improper Payment Rates for Centers for Medicare & Medicaid Services (CMS) Programs (2019). https://www.cms.gov/newsroom/fact-sheets/2019-estimated-improper-payment-rates-centers-medicare-medicaid-services-cms-programs

7. LEIE: Office of Inspector General Leie Downloadable Databases. https://oig.hhs.gov/exclusions/index.asp
8. Sateesh N, Kumar BP, Jyothi P. Supervised learning framework for healthcare fraud detection system with excluded provider labels. J Crit Rev. 2020;7:4785–94.
9. Mayaki MZA, Riveill M. Multiple inputs neural networks for fraud detection. In: 2022 International Conference on Machine Learning, Control, and Robotics (MLCR), pp. 8–13,2022. IEEE
10. Herland M, Khoshgoftaar TM, Bauder RA. Big data fraud detection using multiple medicare data sources. J Big Data. 2018;5(1):1–21.
11. The Centers for Medicare and Medicaid Services: Medicare Durable Medical Equipment, Devices & Supplies – by Referring Provider and Service (2021). https://data.cms.gov/provider-summary-by-type-of-service/medicare-durable-medical-equipment-devices-supplies/medicare-durable-medical-equipment-devices-supplies-by-referring-provider-and-service Accessed 2 July 2022.
12. Lopo JA, Hartomo KD. Evaluating sampling techniques for healthcare insurance fraud detection in imbalanced dataset. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI). 2023;9(2):223–38.
13. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. Smote: synthetic minority over-sampling technique. J Artif Intell Res. 2002;16:321–57.
14. Johnson JM, Khoshgoftaar TM. The effects of data sampling with deep learning and highly imbalanced big data. Inform Syst Front. 2020;22(5):1113–31.
15. Hasanin T, Khoshgoftaar TM, Leevy J, Seliya N. Investigating random undersampling and feature selection on bioinformatics big data. In: 2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService), pp. 346–356, 2019. IEEE
16. Hancock JT, Bauder RA, Wang H, Khoshgoftaar TM. Explainable machine learning models for medicare fraud detection. J Big Data. 2023;10(1):154.
17. Johnson JM, Khoshgoftaar TM. Data-centric ai for healthcare fraud detection. SN Comp Sci. 2023;4(4):389.
18. The Centers for Medicare and Medicaid Services: Medicare Physician & Other Practitioners – by Provider Data Dictionary. https://data.cms.gov/resources/medicare-physician-other-practitioners-by-provider-data-dictionary 2021.
19. The Centers for Medicare and Medicaid Services: Medicare Physician & Other Practitioners – by Provider (2021). https://data.cms.gov/provider-summary-by-type-of-service/medicare-physician-other-practitioners/medicare-physician-other-practitioners-by-provider Accessed 2 July 2022.
20. The Centers for Medicare and Medicaid Services: Medicare Part D Prescribers – by Provider and Drug Data Dictionary (2021). https://data.cms.gov/resources/medicare-part-d-prescribers-by-provider-and-drug-data-dictionary Accessed 16 April 2022.
21. The Centers for Medicare and Medicaid Services: Medicare Part D Prescribers – by Provider Data Dictionary (2020). https://data.cms.gov/resources/medicare-part-d-prescribers-by-provider-data-dictionary Accessed 27 May 2023.
22. The Centers for Medicare and Medicaid Services: Medicare Part D Prescribers – by Provider and Drug (2021). https://data.cms.gov/provider-summary-by-type-of-service/medicare-part-d-prescribers/medicare-part-d-prescribers-by-provider-and-drug Accessed 16 April 2022.
23. The Centers for Medicare and Medicaid Services: Medicare Part D Prescribers - by Provider (2021). https://data.cms.gov/provider-summary-by-type-of-service/medicare-part-d-prescribers/medicare-part-d-prescribers-by-provider Accessed 16 April 2022.
24. The Centers for Medicare and Medicaid Services: Medicare Physician & Other Practitioners – by Provider and Service Data Dictionary. https://data.cms.gov/resources/medicare-physician-other-practitioners-by-provider-and-service-data-dictionary 2021.
25. The Centers for Medicare and Medicaid Services: Medicare Physician & Other Practitioners – by Provider and Service (2021). https://data.cms.gov/provider-summary-by-type-of-service/medicare-physician-other-practitioners/medicare-physician-other-practitioners-by-provider-and-service Accessed 2 July 2022.
26. Bauder RA, Khoshgoftaar TM. A novel method for fraudulent medicare claims detection from expected payment deviations (application paper). In: 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI), pp. 11–19 2016. IEEE.
27. Chen T, Guestrin C. Xgboost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16, 2016.
28. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y. Lightgbm: a highly efficient gradient boosting decision tree. Adv Neural Inform Proc Syst. 2017;30:3146–54.
29. Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. Mach Learn. 2006;63(1):3–42.
30. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
31. Prokhorenkova L, Gusev G, Vorobev A, Dorogush AV, Gulin A. Catboost: unbiased boosting with categorical features. Advances in neural information processing systems 2018;31.
32. Le Cessie S, Van Houwelingen JC. Ridge estimators in logistic regression. J Royal Stat Soc Series C Appl Stat. 1992;41(1):191–201.
33. Breiman L, Friedman J, Stone CJ, Olshen RA. Classification and Regression Trees. US: Taylor & Francis; 1984.
34. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–40.
35. Efron B, Tibshirani RJ. An Introduction to the Bootstrap. Boca Raton: CRC Press; 1994. p. 5–6.
36. Friedman JH. Greedy function approximation: a gradient boosting machine. Ann stat. 2001;29:1189–232.
37. Hasanin T, Khoshgoftaar TM, Leevy JL, Bauder RA. Severely imbalanced big data challenges: investigating data sampling approaches. J Big Data. 2019;6(1):1–25.
38. Iversen GR, Norpoth H. Analysis of Variance, vol. 1. Newbury Park: Sage; 1987.
39. Tukey JW. Comparing individual means in the analysis of variance. Biometrics. 1949;5:99–114.

Publisher's Note