Journal of Big Data

RESEARCH Open Access

The effects of class rarity on the evaluation of supervised healthcare fraud detection models

Matthew Herland*, Richard A. Bauder and Taghi M. Khoshgoftaar

*Correspondence:
mherlan1@fau.edu
Florida Atlantic University,
777 Glades Road, Boca Raton,
FL, USA

Abstract

The United States healthcare system produces an enormous volume of data with a vast number of financial transactions generated by physicians administering healthcare services. This makes healthcare fraud difficult to detect, especially when there are considerably less fraudulent transactions (documented and readily available) than non-fraudulent. The ability to successfully detect fraudulent activities in healthcare, given such discrepancies, can garner up to $350 billion in recovered monetary losses. In machine learning, when one class has a substantially larger number of instances (majority) compared to the other (minority), this is known as class imbalance. In this paper, we focus specifically on Medicare, utilizing three 'Big Data' Medicare claims datasets with real-world fraudulent physicians. We create a training and test dataset for all three Medicare parts, both separately and combined, to assess fraud detection performance. To emulate class rarity, which indicates particularly severe levels of class imbalance, we generate additional datasets, by removing fraud instances, to determine the effects of rarity on fraud detection performance. Before a machine learning model can be distributed for real-world use, a performance evaluation is necessary to determine the best configuration (e.g. learner, class sampling ratio) and whether the associated error rates are low, indicating good detection rates. With our research, we demonstrate the effects of severe class imbalance and rarity using a training and testing (Train_Test) evaluation method via a hold-out set, and provide our recommendations based on the supervised machine learning results. Additionally, we repeat the same experiments using Cross-Validation, and determine it is a viable substitute for Medicare fraud detection. For machine learning with the severe class imbalance datasets, we found that, as expected, fraud detection performance decreased as the fraudulent instances became more rare. We apply Random Undersampling to both Train_Test and Cross-Validation, for all original and generated datasets, in order to assess potential improvements in fraud detection by reducing the adverse effects of class imbalance and rarity. Overall, our results indicate that the Train_Test method significantly outperforms Cross-Validation.

Keywords: Big Data, Medicare, LEIE, Fraud detection, Cross-Validation, Test set, Class imbalance, Rarity, Random Undersampling

Introduction

The healthcare system in the United States (US) contains an extremely large number of physicians, who perform numerous services for an even larger number of patients. Every day, there are a massive number of financial transactions generated by physicians administering healthcare services, such as hospital visits, drug prescriptions, and other medical procedures. The vast majority of these financial transactions are conducted without any fraudulent intent, but there are a minority of physicians who maliciously defraud the system for personal gain. In machine learning, when a dataset portrays this discrepancy in class representation (i.e. a low number of actual fraud cases), it is known as class imbalance [1, 2]. The main issue attributed to class imbalance is the difficulty in discriminating useful information between classes due to the over-representation of the majority class (non-fraud) and the limited amount of information available in the minority class (fraud). In real-world medical practice, the number of fraudulent physicians is in the minority, where even fewer are confirmed, well-documented. To further confound the situation, we found that the number of these known fraudulent physicians are becoming less frequent each year, trending towards class rarity. Class rarity is when the Positive Class Count (PCC), or number of minority class instances, becomes extremely small. We argue that qualifying for severe class imbalance and rarity does not necessarily rely on the proportion between classes, but on PCC. A common class imbalance percentage is 2% [3], but, for example, a dataset with 10,000,000 instances still has a PCC of 200,000. A machine learning model would most likely be able to effectively determine qualities and patterns in the minority class by using 200,000 instances, and would not be effected by issues normally attributed to class imbalance, especially with data sampling. However, these issues would be present if this dataset had an extremely small PCC, such as 100, where a machine learning model would have to contend with 9,999,900 negative instances during training. This latter example indicates the concerns presented with class rarity. These issues are further exacerbated when applied to Big Data, which can dramatically increase the number of majority instances while leaving the minority class representation relatively unchanged [4].

Throughout the literature, the task of defining Big Data has proven rather complicated, without a universally accepted definition [5]. Recently, Senthilkumar et al. [5] provided a definition specifically for healthcare, categorizing Big Data into six V's: Volume, Variety, Velocity, Veracity, Variability, and Value. We employ three publicly available Medicare 'Big Data' datasets released by The Centers for Medicare and Medicaid Services (CMS): 1. Medicare Provider Utilization and Payment Data: Physician and Other Supplier (Part B), 2. Medicare Provider Utilization and Payment Data: Part D Prescriber (Part D), and 3. Medicare Provider Utilization and Payment Data: Referring Durable Medical Equipment, Prosthetics, Orthotics and Supplies (DMEPOS). The data begins in either 2012 or 2013 and ends in 2016 (which was released June 2018). Note that neither 2017 or 2018 is currently available for these Medicare datasets. These CMS datasets include payment information for claims submitted to Medicare, payments made by Medicare, and other data points related to procedures performed, drugs administered, or supplies issued. Both individually and combined, the Medicare datasets provide an extensive view into a physician's annual claims, across three major parts of Medicare. Furthermore, we utilize

the the Office of Inspector General's (OIG) List of Excluded Individuals and Entities (LEIE) [6] to generate fraud labels since CMS does not provide any fraud information. For further detail related to Medicare and Medicare fraud, we refer the reader to [7–11].

Even though there are far fewer fraudulent physicians, they still contribute to large financial losses. Many citizens depend on healthcare, especially the elderly population, which comprises the majority of Medicare beneficiaries. The Federal Bureau of Investigation (FBI) concluded that fraud accounts for 3–10% of healthcare costs [12]. Throughout the United States, in 2017, healthcare generated approximately $3.5 trillion in spending [13], with Medicare contributing around 20% [14], receiving roughly $592 million in federal funding while spending up to $702 billion. This translates to around $105 billion to $350 billion lost from fraud per year, with $21 billion to $70 billion from Medicare alone. In order to relate these monetary values to number of affected citizens, the average healthcare cost in the United States, per person, per year, is about $10,000 [15]. Therefore, if healthcare fraud was eliminated altogether, a potential 35 million ($350 billion/$10,000) more Americans could receive full medical assistance per year, with about 6 million from Medicare alone. This demonstrates the potential impact of minimizing fraud. Note, healthcare spending is forecast to increase more than 4% per year through 2026 [13].

A number of studies employed the CMS Medicare datasets to detect fraudulent physician behavior through data mining, machine learning and other analytical methods [16, 17], with a large portion of these studies using only Part B data [18–22]. Even though the Part B dataset is comprehensive in its own right, employing only one Medicare part limits the comprehensive assessment of fraud detection performance available to a machine learning model compared to multiple parts. There are a few studies that utilized multiple parts of Medicare including [23–25]. Branting et al. [23] used the Part B (2012–2014) and Part D (2013) datasets. They utilize the LEIE for determining fraud labels through their identity-matching algorithm centered around a physician's National Provider Identifier (NPI) [26]. Through this algorithm, they matched over 12,000 fraudulent physicians, but the authors were not as discriminatory in fraud label mapping as we are in this study, leading to the possible inclusion of physicians excluded for charges unrelated to fraud. The authors employed sampling, balancing their dataset to a 50:50 class ratio. The authors may have benefited from examining other class ratios, possibly without removing as many non-fraudulent instances. They developed a method for discriminating fraudulent behavior by determining the fraud risk using graph-based features in conjunction with a decision tree learner resulting in good overall fraud detection. In [24], Sadiq et al. employ Part B, Part D, and DMEPOS datasets, but limit their study to Florida only. They venture to find anomalies that possibly indicate fraudulent or other interesting behavior. The authors use an unsupervised method (Patient Rule Induction Method based bump hunting) to try to detect peak anomalies by spotting spaces of higher modes and masses within the dataset. They conclude that their method can accurately characterize the attribute space of CMS datasets. In [25], we conducted an exploratory study to determine which part of Medicare and which learner allows for the better detection of fraudulent behavior. We used all available years from 2015 and before, while fraud labels were generated through LEIE mapping. However, in [25], our experiments were conducted to show the feasibility of Medicare fraud detection, without focusing on the

problem of class imbalance, rarity, or any methods to mitigate adverse effects on model performance. Unfortunately, even with these and the other research currently available, according to [27], current methods are not significantly decreasing monetary losses facing the US healthcare system. Therefore, we continue the efforts to detect Medicare fraud in order to decrease monetary loss due to real-world fraud.

In real-world practice, machine learning models are built on a full training dataset and evaluated on a separate test dataset consisting of new, unseen data points (i.e. hold-out set). We denote this evaluation method as Train_Test, and emulate this process by splitting the CMS datasets into training datasets (all years prior to 2016) and test datasets (the full 2016 year). In essence, the problem can be summarized as: can a model accurately detect new, known fraudulent physicians (from 2016) based on historical patterns of fraud (prior to 2016)? The results from the Train_Test method will provide a clear evaluation of fraud detection performance with Medicare claims data using machine learning. Furthering this sentiment, Rao et al. [28] discuss that: "Any modeling decisions based upon experiments on the training set, even cross validation estimates, are suspect, until independently verified [by a completely new Test dataset]." Unfortunately, in practice, the means to generate both a separate training and test set is limited, such as when the number of positive cases are too few or when only prior data is available. Therefore, we also conduct all of our experiments using Cross-Validation (CV). CV emulates the Train_Test method by splitting a single dataset into smaller training and test datasets for building and evaluation. This allows practitioners to assess prediction performance without a separate test dataset. For our experiments, we apply CV to our training datasets, and through comparisons with Train_Test results, we determine if CV can be a useful substitute. There are different variants of CV including: k-fold, stratified k-fold, leave-p-out, and holdout [29]. In this study, our experiments employ stratified k-fold CV due to its usefulness and ubiquity in machine learning as well as the its focus on creating balanced class distribution across folds.

We present our novel procedure for data processing and fraud label mapping, to create our Medicare datasets. These Medicare datasets, with the added LEIE fraud labels, have severely imbalanced class distributions. In order to assess the effects of severe class imbalance and rarity, in addition to these original datasets, we generate datasets with growing class imbalance and increasing degrees of rarity by randomly removing positive class instances (i.e. lowering PCC). We also perform data sampling, specifically random undersampling (RUS), to determine whether sampling can effectively mitigate the negative effects of severe class imbalance and class rarity. For each original and generated dataset, we created five additional datasets with varying class ratios, ranging from balanced (50:50) to highly imbalanced (1:99). We evaluate our results over three different learners, across all Medicare parts and the combined dataset, using the area under the receiver operating characteristic (ROC) curve (AUC) and significance testing. Our study has several goals. Primarily, we are determining the effects severe class imbalance and rarity have on the Train_Test evaluation method in Medicare fraud detection, which to the best of our knowledge, we are the first to study. We also compare the Train_Test method to CV, across all experimental configurations. Lastly, we examine overall trends across Train_Test and CV to determine the optimal model configuration in terms of data sampling ratio and learner. From the Train_Test results, we determine that for

the severely imbalanced Medicare claims, machine learning was able to discriminate between real-world fraudulent and non-fraudulent physician behavior reasonably well, but as the PCC trended toward rarity, the results generally decreased over all experiments. Overall, the results show that Train_Test significantly outperforms CV. Even so, we conclude that when necessary CV can be a viable substitute, but a practitioner should note that estimates may be conservative. Moreover, CV was similarly affected by severe class imbalance and rarity. Data sampling was demonstrated to mitigate the effects of having such a limited number of positive class instances, where the best class ratios were those with slightly larger negative class representation (i.e. less balanced).

The rest of the paper is organized as follows. The "Related works" section presents related works, focusing on studies that employ datasets with class rarity. We discuss the Medicare and LEIE datasets in the "Data" section, and summarize our data processing and fraud label mapping. We discuss the concepts of severe class imbalance and rarity in the "Severe class imbalance and rarity" section and explain the Train_Test and CV evaluation methods in "Train_Test and Cross-Validation" section. In "Methods/experimental" section, we outline the learners, performance metrics, and significance testing. We then present our experimental results in "Discussion and results" section. Finally, we conclude our study and present future work.

Related works

Throughout the previous academic literature, class imbalance has been widely studied, and as in the real-world, there are many cases where there is a large disparity between classes, such as online shopping (making purchase/not making purchase) [30] and healthcare fraud (fraud/non-fraud). The majority of these studies employ smaller datasets [31–36]. Experiments using class imbalance with smaller data, could provide a basic understanding of the effects that class imbalance has on Big Data, but will be limited in understanding specific concerns when using Big Data. For instance, we determined throughout our research, when applying sampling in machine learning, a balanced ratio (50:50) is not as beneficial for Big Data as it is for smaller datasets, at least in the Medicare fraud detection domain. Studies that focus on class rarity are far less common [2, 37]. With regards to the research presented in this paper, we limit our discussion to studies that employ Big Data for studying class imbalance in relation to rarity.

In [38], Hasanin et al. use four real-world Big Data sources from the sentiment140 text corpus and the UCI Machine Learning Repository in order to assess the impacts of severe class imbalance on Big Data. To supplement to the original datasets, the authors generate additional datasets ranging from imbalanced to severely imbalanced with the positive class percentages of 10%, 1%, 0.1%, 0.01%, and 0.001%. They use the Random Forest (RF) learner on both Apache Spark and H2O Big Data frameworks to assess classification performance, determining that that 0.1% and 1.0% can provide adequate results. They also experimented with data sampling, specifically RUS, and determined that balanced class ratios provided no benefit over the full datasets. Even though 0.001% is a very large disparity between classes, the authors do not generate any datasets that qualify as a rarity. Fernandez et al. [3] provide a literature survey and experimentation focused on Big Data and class imbalance. They employ Hadoop with MapReduce using the Spark Machine Learning Library (MLlib) [39] versions of undersampling (RUS) and

oversampling (ROS), and Synthetic Minority Over-sampling Technique (SMOTE). The authors compare RUS, ROS, and SMOTE over two Big Data frameworks using two imbalanced datasets, derived from the ECBDL14 dataset, which consist of 12 million and 600,000 instances, 90 features and class ratios of 98:2 (majority:minority). They compare these methods across two learners, RF and Decision Tree. They determine that as the number of partitions decreases, RUS has better performance, while ROS performs better with a larger number of partitions, while SMOTE performed inadequately across all experiments. They also recommend that newer, more advanced Big Data frameworks, such as Apache Spark, should be used compared to more dated frameworks. The authors do not remove any positive class instances in order to study the effects of rarity nor severe class imbalance. Another work by Rastogi et al. [40], use the ECBDL14 dataset, with 1.7% positive class representation (PCC = 48,637) and a total of 2.8 million instances and 631 features. They split the data 80% for training and 20% for testing. They compare Python SMOTE to their own version of SMOTE based on Locality Sensitive Hashing implemented in Apache Spark, demonstrating their model is superior. The authors do not test their method on a dataset with a rare number of positive class members.

Two studies that employ relatively Big Data with rarity are [41] and [42]. Dong et al. [41] develop a deep learning model for very imbalanced datasets, employing batch-wise incremental minority class rectification along with a scalable hard mining principle. They evaluate their method's performance on a number of datasets, including a clothing attribute benchmark dataset (X-domain) with a PCC of 20 and 204,177 negative class instances (clothing attributes dataset). This dataset contains multiple classes, but they also test a binary dataset with 3713 positive instances and 159,057 negative. Through their experiments, they found their method was superior, with a minimum 3–5% increase in accuracy, with the additional benefit of being up to seven times faster. In [42], Zhai et al. utilize seven different datasets from various data sources including one artificial dataset with 321,191 negative class instances and 150 positive. They developed an algorithm based on MapReduce and ensemble extreme learning machine (ELM) classifiers, and determine their method superior compared to three different versions of SMOTE. Through Zhai et al's. research, it is hard to infer the effects of rarity of the aforementioned dataset because their datasets are gathered from multiple sources. In order to determine the effects of rarity and the ability of their model, it would have been beneficial if datasets from the same source were tested with varying PCC values.

In a study conducted by Tayal et al. [43], the authors perform experiments with real-world datasets derived from the standard KDD Cup 1999 data, where the largest contained 812,808 instances. The positive class made up 0.098%, with a PCC of around 800, qualifying this dataset as severely imbalanced, but not rarity. This latter point is because 800 instances could still provide a reasonable level of discrimination for a machine learning model. They determined that their RankRC method was able to outperform several SVM methods and was more efficient with processing speed and space required. Maalouf et al. [44] present a truncated Newton method in prior correction logistic regression (LR) including an additional regularization term to improve performance. They also employ the KDD Cup 1999 dataset, along with six others. The largest dataset they use has 304,814 instances with the positive representation at 0.34%, translating to a PCC of a

little over 1000. In [45], Chai et al. generate a dataset from a manufacturer and user facility device experience database with the goal of automatically identifying health information technology incidents. The subset consists of 570,272 instances with a PCC of 1534. They generate two additional subsets, one balanced (50:50) and another with 0.297% class representation. They employ statistical text classification through LR. These studies utilize data that can be described as relatively big, but do not assess the effects of rarity.

Zhang et al. [30] discuss the vast amounts of data created from online shopping websites and the large imbalance between purchases made versus visits made without a purchase. The level of imbalance quickly escalates when considering high spending customers (i.e. over $100). They found that in a week, a retail website had 42 million visits with only 16,000 purchases resulting in a ratio of 1:2,500, while high spending customers were 1:10,000. They developed an adaptive sampling scheme that samples from severely imbalanced data. Through this method, the authors ensure that when sampling data, they obtain a satisfactory number of positive class instances by searching through the original data. We would argue that when the effects of imbalance can be solved by searching for more available positive class instances, then the domain in question does not suffer from the traditional effects attributed to class imbalance even if that ratio is (1:10,000) or worse. The real effects of severe class imbalance and rarity are felt when, throughout all available data, the resultant PCC is so minimal, a machine learning algorithm cannot discriminate useful patterns from the positive class. As mentioned, we note that the number of available real-world fraudulent physicians matching with the CMS Medicare datasets are decreasing, moving the Medicare fraud detection domain towards rarity. To the best of our knowledge, we are the first study to assess the effects of class rarity using the Train_Test evaluation method with Medicare Big Data using real-world fraud labels.

Data

In this section, we summarize the Medicare datasets, LEIE, and our data preparation and feature engineering. In conjunction to our brief summary, we provide discussions in [25] to cover all data-related details not specifically covered in this work. Since this study aims to predict fraudulent behavior as it appears in real-world medical practice, we utilize the LEIE, which currently contains the most comprehensive list of real-world fraudulent physicians throughout the United States. To the best of our knowledge, there is no publicly available database containing both provider claims activity and fraud labels, and therefore, we use the LEIE to supplement the Medicare datasets, allowing for an accurate assessment of fraud detection performance. Additionally, we detail our training and test datasets, outlining the differences, and discuss our processes.

We utilize three publicly available Medicare datasets maintained by the CMS: Part B, Part D, and DMEPOS [46–48]. CMS is the Federal agency within the US Department of Health and Human Services that manages Medicare, Medicaid, and several other health related programs. These Medicare datasets are derived from administrative claims data for Medicare beneficiaries enrolled in the Fee-For-Service program, where all claims information is recorded after payments are made [49–51], and thus we assume these datasets are already reasonably cleansed. We employ all years currently available for all three parts, where Part B is available for 2012 through 2016 and Part D

and DMEPOS are available for 2013 through 2016. The Part B dataset provides claims information for each procedure a physician performs. The Part D dataset provides information pertaining to the prescription drugs they administer under the Medicare Part D Prescription Drug Program. The DMEPOS dataset provides claims information about medical equipment, prosthetics, orthotics, and supplies that physicians referred patients to either purchase or rent from a supplier. Physicians are identified using their unique NPI established by CMS [26]. Part B and DMEPOS have all procedures labeled by their Healthcare Common Procedure Coding System (HCPCS) code [52], whereas Part D has each drug labeled by its brand and generic name. We create a training and test dataset for each Medicare part. The training datasets encompass all available years of Medicare data prior to 2016, for each Medicare part, by appending each annual dataset, aggregated over matching features. The test datasets were created using the same process, but only include the latest 2016 data. We also develop training and test combined datasets, which integrate features from all three Medicare parts, joined by NPI, provider type, and year (excluding the gender variable from DMEPOS). We develop the combined dataset because, in practice, a physician could submit claims to multiple Medicare parts with no dependable way of determining within which part a physician will target their fraud behavior. Through combining information relating to procedures, drugs and equipment, we are utilizing a more encompassing view of a physician's behavior for machine learning. One limitation to combining Medicare datasets is that it is only applicable to physicians who submit claims to multiple Medicare parts. The combined training dataset consists of the years 2013 through 2015, while the test set is only 2016.

From these Medicare datasets, we select the features specifically related to claims information and a select physician-specific data points, as we believe they provide value and are readily usable by machine learning models. Table 1 demonstrates the features chosen for our study. Note the exclusion feature is generated through mapping to the LEIE, creating the fraud or non-fraud labels for classifying physicians. We excluded repetitious features including physician names, addresses, or code descriptions as they provide no extra value. We also did not include several features containing missing or constant values. NPI was used for identification purposes but not for building the models, and other features, such as Medicare participation, were used for data filtering. Also features, like standardized payments and standard deviation values, are removed since they are not present in all of the Medicare years. Details on all of the available Medicare features can be found in the "Public Use File: A Methodological Overview" documents, for each respective dataset, available at [49–51].

The LEIE was established and is maintained by the Office of Inspector General (OIG) [53] under the authority of Sections 1128 and 1156 of the Social Security Act [6]. The LEIE [54] contains information such as reason for exclusion, date of exclusion, and reinstate/waiver date for all current physicians who violated established rules and were found unsuitable to practice medicine. The LEIE, unfortunately, contains the NPI values for only a small percentage of physicians and entities within its database, contributing to the large class imbalance found after fraud labels are added to the Medicare datasets. We note that 38% of providers convicted of fraud continue practicing medicine and 21% of providers with fraud convictions were not suspended from practicing medicine, despite being convicted [55]. There are different categories

Table 1 Description of features chosen from the Medicare datasets

Datasets		Feature	Description	Type
Combined	Part B	npi*	Unique provider identification number	Categorical
		provider_type	Medical provider's specialty (or practice)	Categorical
		nppes_provider_gender	Provider's gender	Categorical
		line_srvc_cnt	Number of procedures/services the provider performed	Numerical
		bene_unique_cnt	Number of distinct Medicare beneficiaries receiving the service	Numerical
		bene_day_srvc_cnt	Number of distinct Medicare beneficiary / per day services performed	Numerical
		average_submitted_chrg_amt	Average of the charges that the provider submitted for the service	Numerical
		average_medicare_payment_amt	Average payment made to a provider per claim for the service performed	Numerical
	Part D	npi*	Unique provider identification number	Categorical
		specialty_description	Medical provider's specialty (or practice)	Categorical
		bene_count	Number of distinct Medicare beneficiaries receiving the drug	Numerical
		total_claim_count	Number of drug the provider administered	Numerical
		total_30_day_fill_count	Number of standardized 30-day fills	Numerical
		total_day_supply	Number of day's supply	Numerical
		total_drug_cost	Cost paid for all associated claims	Numerical
	DMEPOS	referring_npi*	Unique provider identification number	Categorical
		referring_provider_type	Medical provider's specialty (or practice)	Categorical
		referring_provider_gender**	Provider's gender	Categorical
		number_of_suppliers	Number of suppliers used by provider	Numerical
		number_of_supplier_beneficiaries	Number of beneficiaries associated by the supplier	Numerical
		number_of_supplier_claims	Number of claims submitted by a supplier due to an order by a referring order	Numerical
		number_of_supplier_services	Number of services/products rendered by a supplier	Numerical
		avg_supplier_submitted_charge	Average payment submitted by a supplier	Numerical
		avg_supplier_medicare_pmt_amt	Average payment awarded to suppliers	Numerical
	All	Exclusion	Fraud labels from the LEIE database	Categorical

* Not used in building models

** Not used in the combined dataset

of exclusions, based on severity of offense. As shown in Table 2, we chose only the mandatory non-permissive exclusions. We use these excluded providers as fraud labels in all of our training and test datasets. Note, the LEIE does not provide within which program a physician perpetrated their offenses (i.e. Medicare), meaning these excluded physicians were not necessarily convicted for committing criminal activities within Medicare, but we assume that a physician who commits such acts would continue their fraudulent behavior when submitting claims to Medicare.

The LEIE is aggregated at the provider-level (i.e. a single recorded exclusion per provider by NPI) and does not contain information regarding procedures, drugs or

Table 2 Selected LEIE rules

Rule number	Description	Exclusion period
1128(a)(1)	Conviction of program-related crimes	5 years
1128(a)(2)	Conviction due to patient abuse or neglect	5 years
1128(a)(3)	Felony conviction due to healthcare fraud	5 years
1128(b)(4)	License revocation or suspension	5 years
1128(b)(7)	Fraud, kickbacks and other prohibited activities	5 years
1128(c)(3)(g)(i)	Conviction of two mandatory exclusion offenses	10 years
1128(c)(3)(g)(ii)	Conviction of 3 mandatory exclusion offenses	Indefinite

Table 3 Summary of final datasets: train and test

	Dataset	Features	Non-fraud	Fraud	% Fraud
Train	Part B	126	3,691,146	1409	0.038
	Part D	126	2,098,715	1018	0.048
	DMEPOS	145	862,792	635	0.074
	Combined	173	759,267	473	0.062
Test	Part B	126	999,815	99	0.010
	Part D	123	744,918	135	0.018
	DMEPOS	119	290,548	75	0.026
	Combined	171	256,529	55	0.021

equipment related to fraudulent activities. Therefore, we transform the Medicare data to the provider- (or NPI-) level, and then map LEIE exclusion labels (fraud and non-fraud) to each Medicare dataset by NPI and year. The transformation process consists of grouping the data by provider type (such as Cardiology), NPI, and gender (if available), and aggregating over each procedure/drug name and place of service/rental type. In order to minimize information loss from aggregation, we generate additional numeric features for each original numeric feature located in Table 1, including: mean, sum, median, standard deviation, minimum, and maximum. In preparing our categorical variables for data mining, we utilized one-hot-encoding, which creates new binary features for each option within a categorical variable, assigning a one or zero based on membership. Physicians are labeled as fraudulent for claims within their exclusion period and the period prior to their recorded exclusion start date. The reason we decided to include claims submitted during the exclusion period is that these are payments that should not have been fulfilled by Medicare, and could be considered fraudulent per the federal False Claims Act (FCA) [11]. We include claims prior to the exclusion start date due to these potentially consisting of the fraudulent activities that resulted in the physician being placed on the LEIE, including criminal convictions, patient abuse or neglect, or revoked licenses.

Table 3 summarizes each dataset used in our study, detailing the number of features, number of fraudulent and non-fraudulent instances, and the percentage of fraudulent cases after aggregation, one-hot-encoding and fraud labeling. The main difference between the training and test datasets are in the provider type labels within the 2016 CMS datasets, as they were either entered incorrectly, slightly different, or completely changed. We adjusted as many of these these provider type labels as possible when processing the 2016

datasets (test datasets) to match them to the training dataset labels. A few examples are: Obstetrics and Gynecology to Obstetrics/Gynecology, Radiation Therapy Center to Radiation Therapy, and Oral Surgery (Dentists only) to Oral Surgery (dentists only). In addition, there were other provider types that were added or removed, which we were unable to match between the training and test datasets. To the best of our knowledge, there is no documentation discussing differences in provider types between years. For the Train_Test evaluation method, we removed any non-matching physician type from both the training and test datasets (after one-hot encoding), such as Hospitalist which was added in 2016 and Physical Therapist which was removed starting in 2016. The removal of these non-matching physicians does present some information loss in the Train_Test results, but we believe there is no significant impact on the results as this is a minimal modification. We document these non-matching provider types in Table 7 in Appendix A.

Severe class imbalance and rarity

Having a large difference between the number of majority and minority class instances can create bias towards the majority class when building machine learning models. This is known as class imbalance [56], which presents issues for machine learning algorithms when attempting to discriminate, often complex, patterns between classes, particularly when applied to Big Data. Rarity is an exceptionally severe form of class imbalance. In real-world situations, when severe class imbalance and rarity are present, the minority class is generally the class of interest [2]. When employing Big Data in machine learning, severe class imbalance and rarity exhibit a large volume of majority class instances, increased variability, and disjuncts. Small disjuncts are associated with issues, such as between- and within-class imbalance [1] and [57]. Generally, a learner will provide more accurate results for large disjuncts, which are created based on a large volume of instances. Large disjuncts can overshadow small disjuncts, leading to overfitting and misclassification of the minority class due to the under-representation of subconcepts [58].

Table 4a demonstrates the level of class imbalance present in each Medicare dataset, split by year. We observe that the number and percentage of fraudulent instances matching between the LEIE and the Medicare datasets decreases every year, across each dataset. There are a few possibilities explaining this decrease, including the continued efforts to remove fraudulent physicians from practice, fraudulent physicians more efficiently avoiding detection, law enforcement shifting focus from physician fraud, or the deterrent effect of technological advances in fraud detection. We also note from the labeled Medicare datasets that each year, the non-fraudulent instances generally increase at a faster rate than the fraudulent cases are decreasing. These two observances are pushing the imbalance in fraud instances from severe to rarity. Therefore, rarity is an important topic to study in Medicare fraud detection, and in order to study rarity, we generate additional training datasets as shown in Table 4b. All non-fraudulent instances are kept, while we remove a number of fraudulent instances, achieving further levels of severe class imbalance and rarity. The PCCs in these new generated datasets range from 1000 to 100, based on original number of fraudulent instances. These PCCs were further chosen, based on preliminary results, which demonstrate that these adequately represent class rarity in Big Data. In order to get a thorough representation of fraudulent instances, we generate ten different

Table 4 Summary of Medicare datasets

Year	Part B		Part D		DMEPOS		Combined	
	Fraud	%Fraud	Fraud	%Fraud	Fraud	%Fraud	Fraud	%Fraud
(a) By year								
2012	546	0.062	–	–	–	–	–	–
2013	403	0.044	465	0.069	323	0.110	229	0.090
2014	285	0.030	329	0.047	193	0.068	154	0.061
2015	175	0.018	224	0.031	119	0.041	90	0.035
2016	99	0.010	135	0.018	75	0.026	55	0.021
PCC	**Part B**		**Part D**		**DMEPOS**		**Combined**	
	Fraud	**%Fraud**	**Fraud**	**%Fraud**	**Fraud**	**%Fraud**	**Fraud**	**%Fraud**
(b) Class imbalance and rarity								
All	1409	0.038	1018	0.048	635	0.074	437	0.062
1000	1000	0.027	–	–	–	–	–	–
400	400	0.011	400	0.019	400	0.046	–	–
200	200	0.005	200	0.010	200	0.023	200	0.026
100	–	–	100	0.005	100	0.012	100	0.013

datasets by re-sampling for each dataset/PCC pair. For example, with regard to the 200 PCC for the Part D, we randomly select 200 instances from the original 1018, with this process repeated ten times. The final result for each dataset/PCC pair is the average score across all ten generated rarity subsets.

Data sampling is used to minimize the effects caused by severe class imbalance and rarity, from which there are two main branches: oversampling and undersampling. Oversampling generates new minority instances while undersampling removes majority instances. The goal of data sampling is adjusting the datasets to a given ratio of majority and minority representation. Oversampling has a few disadvantages, including decreased model generalization due to its process of duplicating existing minority class instances [59] and the increased processing time due to these additional instances. For these reasons and based on our prior research where oversampling has been shown to decrease fraud detection [60], we select RUS. The main drawback of RUS is the potential removal of useful information, but it is beneficial when applied to Big Data as removing instances decreases both required computing resources and build time, as well as being supported by [38] and [56]. As we employ RUS, our goal is to incur minimal information loss while simultaneously removing the maximum number of majority instances (i.e. determine which ratio delivers the best fraud detection). Therefore, we chose the following class ratios: 1:99, 10:90, 25:75, 35:65, and 50:50 (minority:majority), including the full, non-sampled datasets as the baseline (labeled as Full). In applying these ratios, we generate ten datasets for each original and generated training dataset, to reduce bias due to poor random draws. These ratios were chosen because they provide a good distribution, ranging from balanced 50:50 to highly imbalanced 1:99 [61]. Note, for Train_Test, when applying RUS or creating the severe class imbalance and rare subsets, only the training datasets are sampled, as they build the model, while test datasets are kept unaltered for model evaluation.

Train_Test and Cross-Validation

In this study, we employ the Train_Test evaluation method, which uses a training dataset for building the model, and evaluate this model using a separate, distinct test dataset, as demonstrated in Fig. 1a. Instances from the test dataset are completely new, and never used for model building. Examining the Train_Test method's performance is necessary for assessing whether, based on past occurrences, a model can accurately predict new occurrences. Through our experimentation, Train_Test will determine, based on prior Medicare data (years < 2016), whether a physician can be accurately classified as fraudulent or non-fraudulent given Medicare data from the most recently released 2016 datasets. We are also assessing how rarity effects the Train_Test method's results, which is especially important since known fraudulent instances are decreasing year-over-year.

Additionally, we also use CV in order to compare results and determine whether employing CV estimates are similar to results from the Train_Test evaluation method. For CV, we use training datasets that were not altered to match the test datasets, as CV does not employ the test dataset. CV is very popular among the Data Mining and Machine Learning community [62] as an evaluation method for prediction performance in almost every application domain, and can be useful when a researcher only has access to prior data. We are performing this comparison with CV due to its popularity and potential drawbacks versus the Train_Test method. Rao et al. [28] recommend validating CV results with a separate test dataset. They also mention that when a model is tuned by a test dataset, this is no longer an accurate simulation of the real-world event. A few other drawbacks of CV, as found in the literature, are CV can result in large errors using small sample sizes [63], the error introduced by bias or variance [29, 64], and CV being vulnerable to high levels of variability. Therefore, by employing the 2016 test datasets with Train_Test, we are evaluating the viability of CV for providing estimates that lead to model selection in Medicare fraud detection.

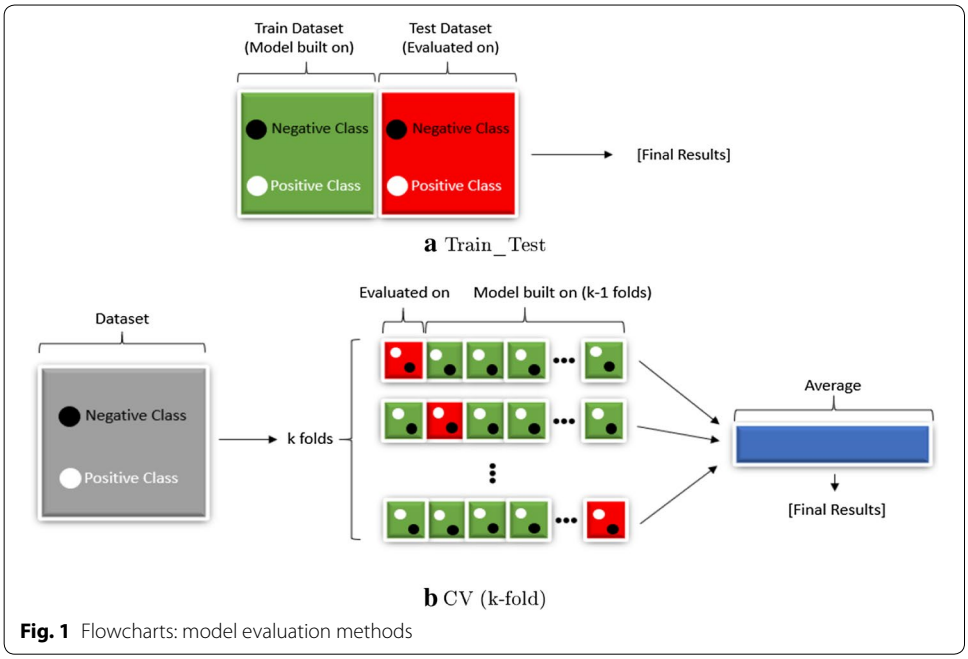

Fig. 1 Flowcharts: model evaluation methods

As mentioned, there are a number of different versions of CV and for this study, we chose stratified k-fold CV with $k = 5$. k-fold CV evenly splits a dataset into k-folds, allowing a learner to evaluate a single dataset (training datasets). The model is then built on $(k - 1)$fold and evaluated on the remaining fold. This is repeated until each fold has been used for evaluation. The results from each repeat are averaged together to give the final result. The process for k-fold CV is demonstrated in Fig. 1b. This allows that every instance located in the training dataset will be used for both model building and evaluation. The process of stratification is important when applying CV to highly imbalanced data, and even more so with rare data, which could result in a fold without a single fraudulent instance. Stratification ensures that all k-fold have approximately the same ratio of class representation as the original data. To avoid any bias caused by bad random draws when creating folds, we repeat the CV process 10 times for each learner/dataset pair. The final detection performance score is the average of all 10 CV repeats.

Methods/experimental

In this section, we discuss the learners (machine learning models), as well as the performance metric and significance testing which will be used to evaluate the influence of severe class imbalance and rarity on Medicare fraud detection. Since our Medicare datasets have such a large volume of data, we required a machine learning network that can handle Big Data. Therefore, we employ Apache Spark [65] on top of a Hadoop [66] YARN cluster for running and validating our models using their implemented MLlib. Apache Spark is a unified analytics engine capable of handling Big Data, offering dramatically quicker data processing over traditional methods or other approaches using MapReduce. The MLlib provided by Apache is a scalable machine learning library built on top of Spark.

Learners

From Apache Spark 2.3.0 [67] MLlib [68], we chose LR [69], and two tree-based models: RF and Gradient Tree Boosting (GTB) [70]. As of this study, Spark's MLib has eight available classifiers. We chose these three based on preliminary research where other learners provided relatively worse fraud detection, such as Multilayer Perceptron or Naive Bayes. We used default configurations for each learner, unless noted otherwise. In Appendix B, we provide detailed descriptions for each learner, as well as indicate any configuration modifications.

Performance metric

In order to evaluate the fraud detection performance of each learner, we use the Area Under the Receiving Operator Curve (ROC) Curve (AUC) [71, 72]. AUC has demonstrated itself quite capable as a metric for quantifying results for machine learning studies employing datasets with class imbalance [73]. AUC shows performance over all decision thresholds, representing the ROC curve as a single value ranging from 0 to 1. An AUC of 1 denotes a classifier with perfect prediction for both the positive and negative classes, 0.5 represents random guessing, and any score under 0.5 means a learner demonstrated predictions worse than random guessing. The ROC curve is a plot comparing false positive rate (1 − specificity) against true positive rate (sensitivity),

and is commonly used to visually represent binary classification results. The false positive rate is calculated by $\frac{FP}{FP+TN}$ and measures the number of negative instances (non-fraud) incorrectly classified as positive (fraudulent) in proportion to the total number of instances labeled as negative, also known as a false alarm rate. True positive rate is calculated by $\frac{TP}{TP+FN}$ and measures the number of positive instances correctly classified as positive in proportion to the total number of instances labeled as positive. This relationship between (1 − specificity) and sensitivity, as portrayed in the ROC curve, illustrates a learner's capability to discriminate between both classes.

Significance testing

We perform hypothesis testing to demonstrate the statistical significance around our AUC results through ANalysis Of VAriance (ANOVA) [74] and Tukey's HSD tests [75]. ANOVA is a statistical test determining whether the means of several groups (or factors) are equal. Tukey's HSD test determines factor means that are significantly different from each other. This test compares all possible pairs of means using a method similar to a t-test, where statistically significant differences are grouped by assigning different letter combinations (e.g. group 'a' is significantly better than group 'b' in correlation to the issue). Both ANOVA and the Tukey's tests explore the differences between the following factors: datasets, learners, PCCs, and class ratios.

Discussion and results

In this section, we discuss our experimental results, assessing the impacts of rarity on Medicare fraud detection, as well as provide recommendations for practitioners based on these results. Table 5 presents the average AUC scores for Train_Test across PCC, consisting of original class distribution (All) and the selected severe class imbalance and rarity values, split by dataset (sub-tables), learner, and class ratio. The boldfaced values indicate the learner/ratio pair producing the best fraud detection performance per PCC. The effects of class rarity are demonstrated across each dataset, where the boldfaced values decrease as the PCCs decrease, and persist across nearly every learner/ratio pair. We notice that across all datasets, LR frequently presents the best scores, where the only outlier is DMEPOS, with GBT having better results for higher PCCs. Even though DMEPOS demonstrates better results with tree-based learners, we observe that as PCC decreases, LR begins to have better results. We believe that LR's results are due to a more successful strategy for handling class imbalance and rarity through regularization, which penalizes large coefficients (Ridge Regression) minimizing the adversities of noise and overfitting, leading to increased model generalization. Both GBT and RF also employ mechanisms to curtail the effects of noise and overfitting, but appear less robust to class imbalance and especially rarity, for Medicare fraud detection. Among the boldfaced values, we notice the less balanced ratios have the highest scores. Upon closer inspection, the 10:90 ratio most frequently scores higher across PCC/ratio pairs followed closely by 1:99 and Full, especially for the combined dataset. Note that the Full (non-sampled) results indicate good detection performance, again, showing that a good representation of the majority class is beneficial. We believe the diminishing results when approaching a balanced configuration are due to the removal of too many negative class instances, deterring the learner's ability to discriminate the details of the non-fraudulent class. The

Table 5 AUC results for Train_Test

Learner	Ratio	200	400	1000	All
(a) Part B					
GBT	[Full]	0.77604	0.78207	0.79080	0.78636
	[1:99]	0.78453	0.79440	0.80158	0.79523
	[10:90]	0.78575	0.79954	0.81324	0.81566
	[25:75]	0.75562	0.78570	0.80824	0.81476
	[35:65]	0.74548	0.78412	0.80523	0.81057
	[50:50]	0.72767	0.76659	0.79002	0.80626
LR	[Full]	0.80406	0.81686	0.82063	0.82133
	[1:99]	*0.80803*	0.82062	0.82441	0.82542
	[10:90]	0.80501	0.81933	*0.82601*	*0.82901*
	[25:75]	0.79251	*0.82101*	0.82347	0.82675
	[35:65]	0.77845	0.81197	0.82049	0.82768
	[50:50]	0.76491	0.80030	0.81863	0.82109
RF	[Full]	0.70754	0.73765	0.75130	0.75725
	[1:99]	0.73952	0.75813	0.76869	0.77081
	[10:90]	0.76287	0.77653	0.78856	0.78527
	[25:75]	0.75999	0.77324	0.78079	0.78683
	[35:65]	0.75139	0.76720	0.77863	0.78505
	[50:50]	0.74320	0.76689	0.77646	0.78136
Learner	**Ratio**	**100**	**200**	**400**	**All**
(b) Part D					
GBT	[Full]	0.69437	0.70885	0.74099	0.74969
	[1:99]	0.69356	0.71879	0.74821	0.76157
	[10:90]	0.68142	0.70006	0.75261	0.78212
	[25:75]	0.65184	0.69628	0.73418	0.77174
	[35:65]	0.64126	0.67560	0.71313	0.77054
	[50:50]	0.62193	0.65445	0.70517	0.74277
LR	[Full]	*0.74339*	0.76832	0.78592	0.79566
	[1:99]	0.73766	*0.76943*	*0.78623*	0.79714
	[10:90]	0.72541	0.76436	0.78491	*0.79858*
	[25:75]	0.71602	0.75240	0.77726	0.79431
	[35:65]	0.69825	0.74023	0.77341	0.79031
	[50:50]	0.68920	0.72922	0.75804	0.78866
RF	[Full]	0.60202	0.62445	0.64317	0.69302
	[1:99]	0.66243	0.68387	0.69370	0.73303
	[10:90]	0.70282	0.72050	0.73803	0.77433
	[25:75]	0.69181	0.71398	0.74121	0.76349
	[35:65]	0.68118	0.70447	0.73372	0.75418
	[50:50]	0.66406	0.68312	0.71714	0.75602
(c) DMEPOS					
GBT	[Full]	0.72688	0.75808	0.78221	0.78281
	[1:99]	0.73083	0.75805	*0.78426*	0.79202
	[10:90]	0.71749	0.74800	0.77617	*0.79683*
	[25:75]	0.67911	0.73027	0.76314	0.78660
	[35:65]	0.66527	0.69776	0.75773	0.77678
	[50:50]	0.65155	0.66424	0.74161	0.75944

Table 5 (continued)

Learner	Ratio	100	200	400	All
LR	[Full]	0.75220	*0.76024*	0.77622	0.78088
	[1:99]	*0.74545*	0.75646	0.77403	0.77819
	[10:90]	0.73002	0.75079	0.76687	0.77482
	[25:75]	0.70978	0.73345	0.75993	0.76963
	[35:65]	0.68578	0.72187	0.75741	0.76715
	[50:50]	0.67933	0.70508	0.74394	0.75723
RF	[Full]	0.65576	0.71649	0.75220	0.77105
	[1:99]	0.67756	0.72877	0.76250	0.78803
	[10:90]	0.70214	0.73717	0.77153	0.78861
	[25:75]	0.71244	0.74737	0.76301	0.78914
	[35:65]	0.70956	0.72159	0.76735	0.78076
	[50:50]	0.69299	0.73423	0.75302	0.77333
Learner	**Ratio**	**100**	**200**		**All**
(d) Combined					
GBT	[Full]	0.76056	0.78431		0.83654
	[1:99]	0.75698	0.79823		0.84513
	[10:90]	0.74038	0.79609		0.84929
	[25:75]	0.73296	0.78145		0.83126
	[35:65]	0.69390	0.77744		0.82757
	[50:50]	0.70015	0.75962		0.81149
LR	[Full]	*0.81514*	*0.85430*		*0.86888*
	[1:99]	0.80496	0.84899		0.86829
	[10:90]	0.76965	0.82737		0.86157
	[25:75]	0.73072	0.80287		0.84583
	[35:65]	0.71753	0.77810		0.83743
	[50:50]	0.69273	0.74712		0.81778
RF	[Full]	0.62150	0.72501		0.80122
	[1:99]	0.71805	0.78308		0.82193
	[10:90]	0.74251	0.78836		0.82896
	[25:75]	0.74442	0.77432		0.81791
	[35:65]	0.73639	0.77206		0.81273
	[50:50]	0.72878	0.76666		0.81375

combined dataset has higher scores compared to the individual datasets, as indicated by the boldfaced values. Possible contributing factors are that the combined dataset contains a larger selection of attributes, which facilitates a broader view of physician behavior over the individual parts, and that it only concentrates on providers who submitted claims to all three parts of Medicare.

Additionally, we perform this same experiment for Train_CV, and provide the results in Table 8 in Appendix C. Figure 2 presents multiple bar graphs, comparing the differences in average AUC scores between Train_Test and Train_CV across each learner, dataset, PCC, and ratio configuration, where each bar represents the average AUC score for Train_Test minus the average AUC score for Train_CV. These bar graphs demonstrate that Train_Test outperforms Train_CV in almost all cases, the only notable contradiction being for the Part B dataset when employing the tree-based learners, in particular RF. We note that Train_Test had superior results for every configuration when

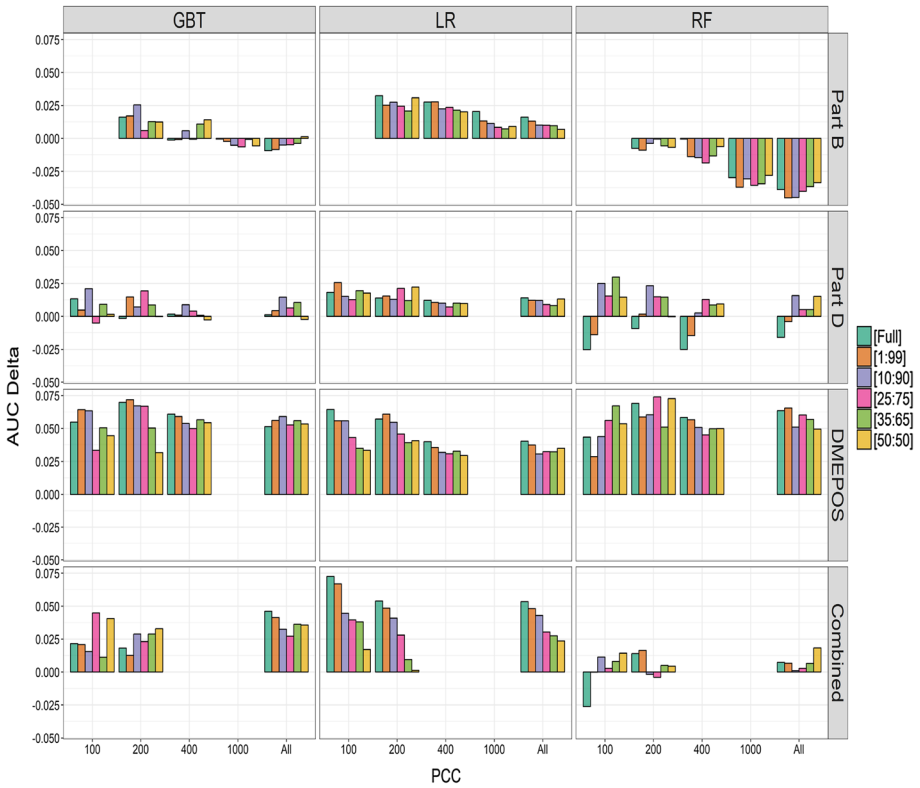

Fig. 2 Average AUC: comparing Train_Test–Train_CV

employing LR. This signifies that the models built using previous years (training dataset) and evaluated on a separate, new year (test dataset) provide better fraud detection over applying CV on the training dataset alone. As mentioned above, CV is susceptible to bias and variance, which could contribute to the moderate results compared to Train_Test. We observe that as PCC decreases, becoming more rare, the delta between Train_Test and Train_CV generally increases. We surmise that Train_Test handles imbalanced data and class rarity better due to the models being trained with all available positive class (fraudulent) instances. Thus, with Train_Test, there is a decreased chance of overfitting, compared to CV, where models are built using a sub-sample of instances in each fold, bringing the already small PCC even lower for each training dataset.

Additionally, we performed hypothesis testing to demonstrate the significance of our results. We used a one-factor ANOVA test for Evaluation Method (Train_Test and Train_CV), and assess significance over learners, datasets, ratios and PCC as shown in Table 9c in Appendix C. Evaluation method was significant at a 95% confidence interval, and therefore, we further conducted a Tukey's HSD test, presented in Table 6, to determine the significance between fraud detection results garnered from Train_Test and Train_CV. The Tukey's HSD test placed Train_Test in group 'a' and Train_CV in group 'b' signifying that evaluating a model on a segregated test set provides significantly better results over building and evaluating a model through CV.

Even though the Tukey's HSD test determined the evaluation methods are one group apart, we can argue that CV provides comparable results to Train_Test, albeit

Table 6 Tukey's HSD test results for evaluation methods

Factor	Level	AUC	Std	r	Min	Max	Group
Evaluation method	Train_Test	0.74604	0.05212	1980	0.56483	0.87266	a
Evaluation method	Train_CV	0.72954	0.06841	10,620	0.40944	0.88532	b

conservative. Therefore, we present the following results for both Train_Test and Train_CV in order to provide a thorough claim as to which learner and ratio yield the best results for class imbalance and rarity, as well as provide further insight into comparing these evaluation methods. We perform a 4-factor ANOVA test for both Train_Test and Train_CV, in Table 9a and b (in Appendix C), and evaluate the differences between datasets, learners, class ratios and PCCs. All factors and their interactions are shown as significant, at a 5% significance level. We perform further Tukey's HSD tests for each PCC, and assess any significant differences for learners (across ratios and datasets) and ratios (across learners and datasets), shown in Fig. 3a and b, respectively. We concentrate only on the results for the best scoring group (group 'a'). For these graphs, there can be multiple combinations within a group designation, such as in Fig. 3a, for the Part D dataset with 200 and 400 PCC, Train_Test and Train_CV with a class ratio of 10:90, are each in group 'a', respectively. The results in Fig. 3a, show that LR contains the vast majority of group 'a' members, while Train_Test and Train_CV both have almost an identical distribution with the same number of group 'a' members. Therefore, regardless of model configuration across all PCCs, LR is able to provide the highest levels of discernment between fraudulent and non-fraudulent behavior patterns within each of our Medicare datasets. Figure 3b shows that the less balanced ratios contain the majority of group 'a' membership, with 10:90 having more representation than all other ratios combined. The more balanced ratios have significantly less group 'a' representation, where 50:50 has zero members. As seen with the learner results, Train_Test and Train_CV have similar distributions. However, Train_Test handles the more balanced datasets better, which is potentially due to the fact that Train_Test employs the entire training datasets whereas Train_CV splits the data, minimizing the already small fraud and non-fraud instances. Overall, from these results, we observe that for PCC, although the rarity experiments have similar group 'a' representation compared to the original class distribution, the overall AUC scores diminish as the level of rarity is increased. The main difference between evaluation methods from the learner and ratio Tukey's test is that the Train_Test generally has higher average AUC scores over comparable configurations. The complete Tukey's HSD results for all configurations for both learners and ratios are listed in Tables 10 and 11 in Appendix C.

In summary, we assessed the effects that class imbalance and rarity have on Medicare claims data, and compared how various machine learning techniques handle detecting fraudulent behavior when being subjected to these effects. Machine learning was able to improve results, with RUS, for the majority of the class imbalance and rarity experiments presented in this work. However, we found that the rarer fraudulent instances

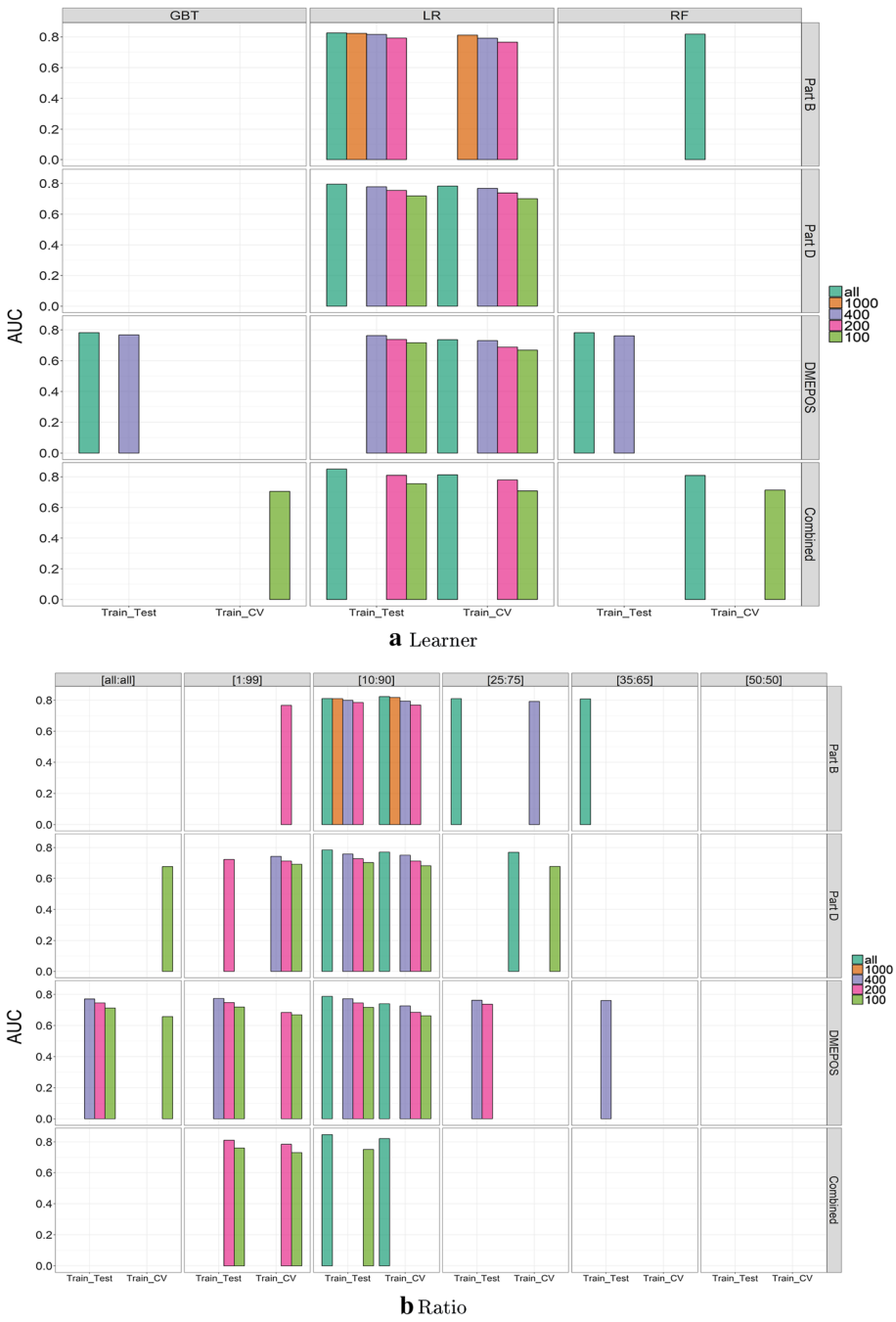

Fig. 3 Tukey's HSD test results for group 'a'

become, the less machine learning can effectively discern fraudulent behavior from non-fraudulent behavior. Therefore, if the PCC qualifies a Medicare claims dataset as rare, we recommend that a practitioner gather additional, quality data until there is a sufficient PCC. The Train_Test and Train_CV method, in general, had similar results, but the latter's results were somewhat conservative in comparison. We recommend practitioners

employ the Train_Test evaluation method with LR, after applying RUS with a 10:90 class distribution.

Conclusion

Two significant challenges facing healthcare fraud detection are the large amounts of data generated (Big Data) and the significant imbalance in fraudulent versus non-fraudulent behavior (class imbalance). The combination of these two issues leads to datasets that contain an extremely large volume of negative class instances (non-fraudulent) and very small numbers of positive class instance (fraudulent). In the case of fraud detection, the data is severely imbalanced. We focus our study on three 'Big Data' datasets released by CMS, specifically Part B, Part D, and DMEPOS (individually and combined), as well as the LEIE from the OIG in order to map our real-world fraud labels. We notice that the fraudulent physicians from the LEIE had less matching physicians between the Medicare datasets, each year, since CMS started releasing these datasets. Because of this, we experimented with further severe class imbalance leading into class rarity. We do this by generating additional datasets and randomly remove fraudulent instances in order to determine the effects of increasing rarity on real-world fraud detection performance (Train_Test). In order to minimize the effects of severe class imbalance and rarity, we also employ data sampling, with various class ratios. In applying RUS, we created a new dataset for each original, severe class imbalanced and rare dataset. Detecting fraudulent behavior is the first step towards eliminating, or at least minimizing, fraud in healthcare, which would allow programs such as Medicare the ability to provide medical funding to a larger number of beneficiaries in the United States.

Throughout our study, we employ three learners and assess model performance using AUC and significance testing. When utilizing the Train_Test evaluation method for severely imbalanced and rare datasets, we recommend building the model with LR and applying RUS with a 10:90 ratio. We noticed that as ratios approached balance (i.e. 50:50), performance decreased, and as such, determine that larger non-fraudulent representation is beneficial, with 10:90 being optimal. In practice though, a separate test dataset to evaluate a machine learning model may not be available due to a shortage of positive cases or lack of new data, and thus requires the use of other methods. To address this, we re-ran all experiments with CV, using the training datasets. CV emulates the Train_Test method, providing model generalization and error estimates on a single dataset by sub-setting the dataset into smaller training and test datasets, allowing all instances to both build and evaluate performance. We found that Train_Test results were significantly better than CV, but we determine that CV can be a reliable substitute, when necessary, but a practitioner should keep in mind that results will be conservative. CV also showed similar patterns to Train_Test in terms of observed effects due to severe class imbalance and rarity, as well as the improvement garnered upon applying RUS. Overall, we noticed that prediction performance decreased as the number of fraudulent instances trended towards rarity, and therefore, we recommend that when PCC becomes too small (rare), then a practitioner should search for more quality data in order to appropriately allow for proper discrimination between fraudulent and non-fraudulent instances when applying machine learning. Future work will consist of employing other Big Data sources from other branches of Medicare or other healthcare

programs, including misclassification costs, and determining methods for obtaining more quality real-world fraudulent physicians.

Abbreviations

Train_Test: model built on training data and evaluated with test data; Train_CV: model built and evaluated using training data through Cross-Validation; US: United States; PCC: Positive Class Count; CMS: Centers for Medicare and Medicaid Services; Part B: Medicare Provider Utilization and Payment Data: Physician and Other Supplier; Part D: Medicare Provider Utilization and Payment Data: Part D Prescriber; DMEPOS: Medicare Provider Utilization and Payment Data: Referring Durable Medical Equipment, Prosthetics, Orthotics and Supplies; OIG: Office of Inspector General; LEIE: List of Excluded Individuals and Entities; FBI: Federal Bureau of Investigation; NPI: National Provider Identifier; CV: Cross-Validation; RUS: Random Undersampling; ROC: receiver operating characteristic; AUC: area under the ROC curve; RF: Random Forest; MLlib: Machine Learning Library; ROS: Random Oversampling; SMOTE: Synthetic Minority Over-sampling Technique; ELM: extreme learning machine; LR: Logistic Regression; HCPCS: Healthcare Common Procedure Coding System; FCA: False Claims Act; GBT: Gradient Tree Boosting; ANOVA: Analysis of Variance.

Authors' contributions

MH and RAB performed the primary literature review, experimentation and analysis for this work, and also drafted the manuscript. TMK worked with MH to develop the article's framework and focus. TMK introduced this topic to MH, and to complete and finalize this work. All authors read and approved the final manuscript.

Acknowledgements

We would like to thank the reviewers in the Data Mining and Machine Learning Laboratory at Florida Atlantic University. Additionally, we acknowledge partial support by the NSF (CNS-1427536). Opinions, findings, conclusions, or recommendations in this paper are solely of the authors' and do not reflect the views of the NSF.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

Not applicable.

Consent for publication

Not applicable.

Ethics approval and consent to participate

Not applicable.

Funding

Not applicable.

Appendix A

See Table 7.

Table 7 Provider type labels removed from train and test datasets

Dataset	Not in Train	Not in Test
Part B	Hospitalist	Psychologist (billing independently)
	Dentist	Pharmacy
Part D	Dentist	Medical supply company, other
	Hospitalist	All other suppliers
	Individual certified prosthetist-orthotist	Ambulance service supplier
		Pharmacy
		Voluntary Health or Charitable Agencies
		Centralized flu
DMEPOS	Hospitalist	All other suppliers
		Ambulatory Surgical Center
		Anesthesiologist assistants
		Audiologist (billing independently)
		Centralized flu
		Clinical laboratory
		HHA (Dmercs only)
		Independent diagnostic testing facility
		Individual certified orthotist
		Individual certified prosthetist
		Mass immunization roster biller
		Medical supply company, other
		Medical supply with certified orthotist
		Medical supply with certified prosthetist-orthotist
		Medical supply with prosthetist
		Medical supply with resp. therapist (Dmercs only)
		Occupational therapist
		Ocularist
		Optician
		Pharmacy (Dmercs only)
		Physical therapist
		Public Health Welfare Agency
		Slide preparation facility
		SNF (Dmercs Only)
		Speech language pathologist
		Supplier of oxygen and/or oxygen related equip.
		Voluntary Health or Charitable Agency
Combined	Hospitalist	Clinical Psychologist
		Occupational therapist
		Physical therapist

Appendix B

Logistic Regression predicts probabilities for which class a categorical dependent variable belongs to by using a set of independent variables employing a logistic function. This learner employs a sigmoidal (logistic) function to generate values between [0,1] representing class probabilities. LR is similar to linear regression but uses a different hypothesis class to predict class membership [76–79]. The bound matrix was set to match the shape of the data (number of classes and features) allowing the algorithm to know the number of classes and features the dataset contains and the bound vector size was set to 1 signifying binomial regression and no thresholds set for binary classification.

Random Forest employs sampling with replacement, creating a number of randomized datasets to build each tree, where features are selected automatically, at each node, through entropy and information gain. Each tree within the forest is dependent upon the values dictated by a random vector that is independently sampled and where each tree is equally distributed among the forest [78, 80]. The generation of random datasets minimizes overfitting. We build each RF learner with 100 trees. The parameter that caches node IDs for each instance, was set to true and the maximum memory parameter was set to 1024 MB in order to minimize training time. The setting that manipulates the number of features to consider for splits at each tree node was set to one-third, since this setting provided better results upon initial investigation. The maximum bins parameter determines the max number of bins to be used for discretizing continuous features, and is set to 2 since we converted our categorical features through one-hot encoding.

Gradient Boosted Trees is an ensemble of decision trees which trains each decision tree individually in order to minimize loss determined by the algorithm's loss function. During each iteration, the current ensemble is used to predict the class for each instance in the training data. The predicted values are compared with the actual values allowing the algorithm to detect and improve upon previously mislabeled instances. The parameter that caches node IDs for each instance, was set to true and the maximum memory parameter was set to 1024 MB to minimize training time.

Appendix C

See Tables 8, 9, 10, 11.

Table 8 AUC Results for Train_CV

Learner	Ratio	200	400	1000	All
(a) Part B					
GBT	[Full]	0.75982	0.78328	0.79120	0.79569
	[1:99]	0.76740	0.79520	0.80378	0.80373
	[10:90]	0.76032	0.79377	0.81847	0.82064
	[25:75]	0.74964	0.78624	0.81464	0.81948
	[35:65]	0.73271	0.77326	0.80600	0.81434
	[50:50]	0.71530	0.75244	0.79563	0.80499
LR	[Full]	0.77162	0.78921	0.80019	0.80516
	[1:99]	*0.78282*	0.79295	0.81119	0.81238
	[10:90]	0.77752	0.79680	0.81465	0.81881
	[25:75]	0.76797	*0.79746*	0.81507	0.81686
	[35:65]	0.75771	0.79061	0.81336	0.81806
	[50:50]	0.73414	0.78012	0.80964	0.81415
RF	[Full]	0.71510	0.73806	0.78110	0.79604
	[1:99]	0.74846	0.77197	0.80579	0.81586
	[10:90]	0.76661	0.79117	*0.81933*	*0.83012*
	[25:75]	0.76031	0.79187	0.81641	0.82703
	[35:65]	0.75699	0.78061	0.81299	0.82156
	[50:50]	0.74994	0.77298	0.80448	0.81496
Learner	**Ratio**	**100**	**200**	**400**	**All**
(b) Part D					
GBT	[Full]	0.68101	0.71044	0.73932	0.74851
	[1:99]	0.68871	0.70412	0.74731	0.75727
	[10:90]	0.66033	0.69299	0.74381	0.76756
	[25:75]	0.65692	0.67700	0.73008	0.76538
	[35:65]	0.63219	0.66694	0.71228	0.75996
	[50:50]	0.62040	0.65461	0.70773	0.74506
LR	[Full]	0.72516	*0.75436*	0.77369	0.78164
	[1:99]	*0.71200*	0.75396	*0.77575*	0.78486
	[10:90]	0.71031	0.75129	0.77481	*0.78657*
	[25:75]	0.70331	0.73115	0.77009	0.78540
	[35:65]	0.67880	0.72835	0.76340	0.78216
	[50:50]	0.67158	0.70696	0.74834	0.77557
RF	[Full]	0.62721	0.63364	0.66818	0.70888
	[1:99]	0.67627	0.68215	0.70816	0.73706
	[10:90]	0.67777	0.69735	0.73538	0.75857
	[25:75]	0.67634	0.69916	0.72832	0.75838
	[35:65]	0.65126	0.68992	0.72510	0.74904
	[50:50]	0.64951	0.68343	0.70771	0.74088
Learner	**Ratio**	**100**	**200**	**400**	**All**
(c) DMEPOS					
GBT	[Full]	0.67203	0.68827	0.72125	0.73129
	[1:99]	0.66654	0.68611	0.72516	0.73591
	[10:90]	0.65411	0.68073	0.72241	0.73777
	[25:75]	0.64571	0.66342	0.71327	0.73389
	[35:65]	0.61468	0.64740	0.70118	0.72090
	[50:50]	0.60699	0.63259	0.68728	0.70598

Table 8 (continued)

Learner	Ratio	100	200	400	All
LR	[Full]	0.68783	*0.70311*	0.73615	0.74063
	[1:99]	*0.68960*	0.69565	*0.73853*	*0.74085*
	[10:90]	0.67423	0.69604	0.73498	0.74421
	[25:75]	0.66667	0.68769	0.72912	0.73715
	[35:65]	0.65088	0.68259	0.72463	0.73488
	[50:50]	0.64590	0.66432	0.71445	0.72225
RF	[Full]	0.61229	0.64745	0.69381	0.70756
	[1:99]	0.64896	0.66998	0.70598	0.72245
	[10:90]	0.65829	0.67671	0.72066	0.73767
	[25:75]	0.65636	0.67337	0.71790	0.72889
	[35:65]	0.64239	0.67054	0.71756	0.72390
	[50:50]	0.63938	0.66152	0.70306	0.72379

Learner	Ratio	100	200	All
(d) Combined				
GBT	[Full]	0.73906	0.76623	0.79047
	[1:99]	0.73626	0.78562	0.80373
	[10:90]	0.72482	0.76730	0.81675
	[25:75]	0.68806	0.75833	0.80405
	[35:65]	0.68275	0.74855	0.79127
	[50:50]	0.65960	0.72675	0.77587
LR	[Full]	0.74260	0.80043	0.81554
	[1:99]	0.73814	*0.80060*	0.82011
	[10:90]	0.72508	0.78653	0.81868
	[25:75]	0.69117	0.77479	0.81553
	[35:65]	0.67940	0.76854	0.80998
	[50:50]	0.67567	0.74588	0.79415
RF	[Full]	0.64769	0.71098	0.79383
	[1:99]	0.71813	0.76663	0.81515
	[10:90]	0.73110	0.79011	*0.82793*
	[25:75]	*0.74162*	0.77822	0.81503
	[35:65]	0.72834	0.76699	0.80619
	[50:50]	0.71446	0.76228	0.79546

Table 9 ANOVA tests

Term	Df	Sum Sq	Mean Sq	F value	Pr(>F)
(a) Four factor: Train_Test					
Datasets	3	1.96516	0.65505	1600.558663	0
Learner	2	0.66409	0.33205	811.3215437	1.29E−270
Pos count	4	2.33669	0.58417	1427.367028	0
Ratio	5	0.30092	0.06018	147.0525251	1.17E−136
Dataset:learner	6	0.22145	0.03691	90.18221558	2.05E−102
Dataset:pos count	7	0.05788	0.00827	20.2044874	1.61E−26
Learner:pos count	8	0.03291	0.00411	10.0523189	7.04E−14
Dataset:ratio	15	0.08724	0.00582	14.21108373	2.17E−35
Learner:ratio	10	0.46537	0.04654	113.7087576	2.69E−194
Pos count:ratio	20	0.05443	0.00272	6.649302133	3.98E−18
Dataset:learner:pos count	14	0.02401	0.00171	4.189763574	2.52E−07
Dataset:learner:ratio	30	0.11577	0.00386	9.428942232	2.99E−40
Dataset:pos count:ratio	35	0.01306	0.00037	0.911606914	0.61790442
Learner:pos count:ratio	40	0.10682	0.00267	6.52539348	3.70E−32
Dataset:learner:pos count:ratio	70	0.04795	0.00068	1.673573178	4.56E−04
Residuals	2430	0.99452	0.00041	–	–
(b) Four factor: Train_CV					
Datasets	7	26.98719	3.85531	2641.872038	0
Learner	2	1.34356	0.67178	460.3389767	5.33E−194
Pos count	3	6.30292	2.10097	1439.702651	0
Ratio	5	1.25714	0.25143	172.2919289	3.65E−178
Dataset:learner	14	0.79698	0.05693	39.00963177	2.86E−105
Dataset:pos count	4	0.34344	0.08586	58.83679308	2.58E−49
Learner:pos count	6	0.07181	0.01197	8.201310311	7.04E−09
Dataset:ratio	35	0.32743	0.00936	6.410645904	3.43E−29
Learner:ratio	10	1.36440	0.13644	93.49620948	1.03E−187
Pos count:ratio	15	0.13786	0.00919	6.298014086	1.64E−13
Dataset:learner:pos count	8	0.04741	0.00593	4.061371638	7.71E−05
Dataset:learner:ratio	70	0.42329	0.00605	4.143770726	3.52E−28
Dataset:pos count:ratio	20	0.01735	0.00087	0.594299086	0.919852682
Learner:pos count:ratio	30	0.08452	0.00282	1.9304991	0.001663456
Dataset:learner:pos count:ratio	40	0.07660	0.00192	1.312296909	0.089594623
Residuals	13230	19.30669	0.00146	–	–
(c) Two factor: Train_Test and Train_CV					
Evaluation method	1	0.45439	0.45439	103.9411081	2.59E−24
Residuals	12598	55.07319	0.00437	–	–

Table 10 Tukey's HSD: learner

Part B				Part D				DMEPOS				Combined			
Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group
(a) Train_Test															
All	LR	0.82521	a	All	LR	0.79411	a	All	GBT	0.78241	a	All	LR	0.84996	a
1000	LR	0.82227	a	400	LR	0.77763	a	All	RF	0.78182	a	200	LR	0.80979	a
400	LR	0.81501	a	200	LR	0.75399	a	400	GBT	0.76752	a	100	LR	0.75512	a
200	LR	0.79216	a	100	LR	0.71832	a	400	LR	0.76307	a	All	GBT	0.83355	b
All	GBT	0.80481	b	All	GBT	0.76307	b	400	RF	0.76160	a	200	GBT	0.78286	b
1000	GBT	0.80152	b	400	GBT	0.73238	b	200	LR	0.73798	a	100	GBT	0.73082	b
400	GBT	0.78540	b	200	GBT	0.69234	b	100	LR	0.71709	a	100	RF	0.71527	b
200	GBT	0.76252	b	200	RF	0.68840	b	200	RF	0.73094	ab	All	RF	0.81608	c
All	RF	0.77776	c	100	RF	0.66739	b	All	LR	0.77132	b	200	RF	0.76825	c
1000	RF	0.77407	c	100	GBT	0.66406	b	200	GBT	0.72607	b	–	–	–	–
400	RF	0.76327	c	All	RF	0.74568	c	100	GBT	0.69519	b	–	–	–	–
200	RF	0.74408	c	400	RF	0.71116	c	100	RF	0.69174	b	–	–	–	–
(b) Train_CV															
All	RF	0.81760	a	All	LR	0.78270	a	All	LR	0.73666	a	All	LR	0.81233	a
1000	LR	0.81068	a	400	LR	0.76768	a	400	LR	0.72964	a	All	RF	0.80893	a
400	LR	0.79119	a	200	LR	0.73768	a	200	LR	0.68823	a	200	LR	0.77946	a
200	LR	0.76530	a	100	LR	0.70019	a	100	LR	0.66918	a	100	RF	0.71356	a
All	LR	0.81424	b	All	GBT	0.75729	b	All	GBT	0.72763	b	100	LR	0.70868	a
1000	RF	0.80668	b	400	GBT	0.73009	b	All	RF	0.72404	b	100	GBT	0.70509	a
1000	GBT	0.80495	b	200	GBT	0.68435	b	400	GBT	0.71176	b	All	GBT	0.79702	b
400	GBT	0.78070	b	200	RF	0.68094	b	400	RF	0.70983	b	200	RF	0.76253	b
200	RF	0.74957	b	100	RF	0.65973	b	200	RF	0.66659	b	200	GBT	0.75880	b
200	GBT	0.74753	b	100	GBT	0.65659	b	200	GBT	0.66642	b	–	–	–	–
All	GBT	0.80981	c	All	RF	0.74213	c	100	GBT	0.64334	b	–	–	–	-
400	RF	0.77444	c	400	RF	0.71214	c	100	RF	0.64295	b	–	–	–	–

Table 11 Tukey's HSD: ratio

Part B				Part D				DMEPOS				Combined			
Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group
(a) Train_Test															
All	[10:90]	0.80998	a	All	[10:90]	0.78501	a	All	[10:90]	0.78675	a	All	[10:90]	0.84661	a
All	[25:75]	0.80945	a	400	[10:90]	0.75851	a	400	[1:99]	0.77359	a	200	[1:99]	0.81010	a
1000	[10:90]	0.80927	a	200	[10:90]	0.72831	a	400	[10:90]	0.77152	a	100	[1:99]	0.76000	a
All	[35:65]	0.80777	a	200	[1:99]	0.72403	a	400	[All:all]	0.77021	a	100	[10:90]	0.75085	a
400	[10:90]	0.79846	a	100	[10:90]	0.70322	a	400	[25:75]	0.76203	a	All	[1:99]	0.84512	ab
200	[10:90]	0.78454	a	400	[25:75]	0.75088	ab	400	[35:65]	0.76083	a	200	[10:90]	0.80394	ab
1000	[25:75]	0.80416	ab	200	[25:75]	0.72089	ab	200	[1:99]	0.74776	a	100	[25:75]	0.73603	ab
400	[25:75]	0.79332	ab	100	[1:99]	0.69788	ab	200	[10:90]	0.74532	a	100	[All:all]	0.73240	abc
400	[1:99]	0.79105	ab	100	[25:75]	0.68656	abc	200	[All:all]	0.74494	a	All	[All:all]	0.83554	bc
200	[1:99]	0.77736	ab	All	[25:75]	0.77651	b	200	[25:75]	0.73703	a	200	[All:all]	0.78787	bc
All	[50:50]	0.80290	b	All	[35:65]	0.77168	b	100	[1:99]	0.71795	a	100	[35:65]	0.71594	bc
1000	[35:65]	0.80145	bc	400	[1:99]	0.74272	b	100	[10:90]	0.71655	a	All	[25:75]	0.83167	c
400	[35:65]	0.78776	bc	400	[35:65]	0.74009	b	100	[All:all]	0.71162	a	All	[35:65]	0.82591	c
200	[25:75]	0.76937	bc	200	[35:65]	0.70677	bc	All	[1:99]	0.78608	ab	200	[25:75]	0.78622	c
All	[1:99]	0.79715	c	100	[All:all]	0.67992	bc	100	[25:75]	0.70045	ab	200	[35:65]	0.77587	c
200	[All:all]	0.76255	c	All	[1:99]	0.76392	c	All	[25:75]	0.78179	abc	100	[50:50]	0.70722	c
200	[35:65]	0.75844	c	All	[50:50]	0.76248	c	400	[50:50]	0.74619	b	All	[50:50]	0.81434	d
1000	[1:99]	0.79823	cd	400	[50:50]	0.72678	c	200	[35:65]	0.71374	b	200	[50:50]	0.75780	d
400	[All:all]	0.77886	cd	400	[All:all]	0.72336	c	200	[50:50]	0.70118	b	–	–	–	–
1000	[50:50]	0.79504	d	200	[All:all]	0.70054	cd	all	[All:all]	0.77825	bc	–	–	–	–
All	[All:all]	0.78832	d	100	[35:65]	0.67357	cd	100	[35:65]	0.68687	bc	–	–	–	–
400	[50:50]	0.77793	d	All	[All:all]	0.74612	d	All	[35:65]	0.77490	c	–	–	–	–
200	[50:50]	0.74526	d	200	[50:50]	0.68893	d	100	[50:50]	0.67462	c	–	–	–	–
1000	[All:all]	0.78758	e	100	[50:50]	0.65839	d	All	[50:50]	0.76333	d	–	–	–	–

Table 11 (continued)

Part B				Part D				DMEPOS				Combined			
Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group	Pos count	Ratio	AUC	Group
(b) Train_CV															
All	[10:90]	0.82319	a	All	[10:90]	0.77090	a	All	[10:90]	0.73988	a	All	[10:90]	0.82112	a
1000	[10:90]	0.81748	a	All	[25:75]	0.76972	a	400	[10:90]	0.72602	a	200	[1:99]	0.78428	a
400	[10:90]	0.79391	a	400	[10:90]	0.75133	a	200	[10:90]	0.68450	a	100	[1:99]	0.73084	a
400	[25:75]	0.79186	a	400	[1:99]	0.74374	a	200	[1:99]	0.68391	a	200	[10:90]	0.78131	ab
200	[10:90]	0.76815	a	200	[10:90]	0.71388	a	100	[1:99]	0.66837	a	100	[10:90]	0.72700	ab
200	[1:99]	0.76623	a	200	[1:99]	0.71341	a	100	[10:90]	0.66221	a	All	[1:99]	0.81300	b
All	[25:75]	0.82113	ab	100	[1:99]	0.69233	a	100	[All:all]	0.65738	a	All	[25:75]	0.81154	b
1000	[25:75]	0.81538	ab	100	[10:90]	0.68280	a	400	[1:99]	0.72322	ab	200	[25:75]	0.77045	bc
400	[1:99]	0.78671	ab	100	[25:75]	0.67886	a	400	[25:75]	0.72010	ab	100	[All:all]	0.70978	bc
200	[25:75]	0.75931	ab	100	[All:all]	0.67779	a	400	[All:all]	0.71707	ab	100	[25:75]	0.70695	bc
All	[35:65]	0.81799	b	400	[25:75]	0.74283	ab	200	[All:all]	0.67961	ab	All	[35:65]	0.80248	c
400	[35:65]	0.78149	b	200	[25:75]	0.70243	ab	200	[25:75]	0.67483	ab	All	[All:all]	0.79995	c
200	[35:65]	0.74914	b	All	[35:65]	0.76372	b	100	[25:75]	0.65625	ab	200	[35:65]	0.76136	c
200	[All:all]	0.74885	b	All	[1:99]	0.75973	b	All	[25:75]	0.73331	b	200	[All:all]	0.75921	c
1000	[35:65]	0.81078	bc	200	[All:all]	0.69948	b	All	[1:99]	0.73307	b	100	[35:65]	0.69683	cd
All	[50:50]	0.81137	c	100	[35:65]	0.65408	b	400	[35:65]	0.71446	b	All	[50:50]	0.78849	d
All	[1:99]	0.81066	c	100	[50:50]	0.64716	b	200	[35:65]	0.66684	b	200	[50:50]	0.74497	d
400	[All:all]	0.77018	c	400	[35:65]	0.73359	bc	100	[35:65]	0.63598	bc	100	[50:50]	0.68324	d
400	[50:50]	0.76851	c	200	[35:65]	0.69507	bc	All	[35:65]	0.72656	c	–	–	–	–
200	[50:50]	0.73313	c	All	[50:50]	0.75384	c	All	[all:all]	0.72649	c	–	–	–	–
1000	[1:99]	0.80692	cd	200	[50:50]	0.68167	c	400	[50:50]	0.70160	c	–	–	–	–
1000	[50:50]	0.80325	d	400	[All:all]	0.72706	cd	200	[50:50]	0.65281	c	–	–	–	–
all	[All:all]	0.79896	d	All	[All:all]	0.74634	d	100	[50:50]	0.63076	c	–	–	–	–
1000	[All:all]	0.79083	e	400	[50:50]	0.72126	d	All	[50:50]	0.71734	d	–	–	–	–

Publisher's Note

Received: 11 December 2018 Accepted: 13 February 2019
Published online: 28 February 2019

References

1. He H, Garcia EA. Learning from imbalanced data. IEEE Trans Knowl Data Eng. 2008;9:1263–84.
2. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A. Mining data with rare events: a case study. In: Ictai, 2007. Piscataway: IEEE. 2007; p. 132–139.
3. Fernández A, del Río S, Chawla NV, Herrera F. An insight into imbalanced big data classification: outcomes and challenges. Complex Intell Syst. 2017;3(2):105–20.
4. Katal A, Wazid M, Goudar R. Big data: issues, challenges, tools and good practices. In: Sixth international conference on Contemporary computing (IC3), 2013. Piscataway: IEEE; p. 404–409.
5. Senthilkumar S, Rai BK, Meshram AA, Gunasekaran A, Chandrakumarmangalam S. Big data in healthcare management: a review of literature. Am J Theor Appl Business. 2018;4(2):57–69.
6. OIG: Office of Inspector General Exclusion Authorities. https://oig.hhs.gov/exclusions/index.asp Accessed 14 July 2016.
7. US Government, US Centers for Medicare & Medicaid Services: The Official US Government Site for Medicare. https://www.medicare.gov/. Accessed 21 Jan 2017.
8. CMS: Research, Statistics, Data, and Systems. https://www.cms.gov/research-statistics-data-and-systems/research-statistics-data-and-systems.html. Accessed 18 Nov 2018.
9. Medicare.gov: What's Medicare.
10. Bauder R, Khoshgoftaar TM, Seliya N. A survey on the state of healthcare upcoding fraud analysis and detection. Health Serv Outcomes Res Methodol. 2017;17(1):31–55.
11. CMS: Medicare fraud & abuse: prevention, detection, AND reporting. https://www.cms.gov/Outreach-and-Education/Medicare-Learning-Network-MLN/MLNProducts/downloads/Fraud_and_Abuse.pdf. Accessed 28 Sept 2018.
12. Morris L. Combating fraud in health care: an essential component of any cost containment strategy. Health Affairs. 2009;28(5):1351–6.
13. CMS: National Health Expenditure Projections 2017–2026. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/NationalHealthExpendData/Downloads/ForecastSummary.pdf Accessed 27 Sept 2018.
14. Kaiser Family Foundation: The facts on Medicare spending and financing. https://www.kff.org/medicare/issue-brief/the-facts-on-medicare-spending-and-financing/ Accessed 27 Sept 2018.
15. Bradley Sawyer and Cynthia Cox Kaiser Family Foundation: how does health spending in the US compare to other countries? https://www.healthsystemtracker.org/chart-collection/health-spending-u-s-compare-countries/#item-relative-size-wealth-u-s-spends-disproportionate-amount-health. Accessed 27 Sept 2018.
16. Sheshasaayee A, Thomas SS. A purview of the impact of supervised learning methodologies on health insurance fraud detection. In: Information systems design and intelligent applications. Singapore: Springer; 2018. p. 978–984.
17. Waghade SS, Karandikar AM. A comprehensive study of healthcare fraud detection based on machine learning. Int J Appl Eng Res. 2018;13(6):4175–8.
18. Feldman K, Chawla NV. Does medical school training relate to practice? Evidence from big data. Big Data. 2015;3(2):103–13.
19. Herland M, Bauder RA, Khoshgoftaar TM. Approaches for identifying us Medicare fraud in provider claims data. Health Care Manag Sci. 2018. https://doi.org/10.1007/s10729-018-9460-8.
20. Bauder RA, Khoshgoftaar TM. The detection of Medicare fraud using machine learning methods with excluded provider labels. In: FLAIRS conference, 2018. p. 404–409.
21. Chandola V, Sukumar SR, Schryver JC. Knowledge discovery from massive healthcare claims data. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining. New York: ACM; 2013. p. 1312–1320.
22. Khurjekar N, Chou C-A, Khasawneh MT. Detection of fraudulent claims using hierarchical cluster analysis. In: IIE Proceedings annual conference. Peachtree Corners: Institute of Industrial and Systems Engineers (IISE); 2015. p. 2388.
23. Branting LK, Reeder F, Gold J, Champney T. Graph analytics for healthcare fraud risk estimation. In: IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM), 2016. Piscataway: IEEE; 2016. p. 845–851.
24. Sadiq S, Tao Y, Yan Y, Shyu M-L. Mining anomalies in Medicare big data using patient rule induction method. In: IEEE third international conference On multimedia Big Data (BigMM), 2017. Piscataway: IEEE; 2017; p. 185–192.
25. Herland M, Khoshgoftaar TM, Bauder RA. Big data fraud detection using multiple Medicare data sources. J Big Data. 2018;5(1):29. https://doi.org/10.1186/s40537-018-0138-3.
26. CMS: National Provider Identifier Standard (NPI). https://www.cms.gov/Regulations-and-Guidance/Administrative-Simplification/NationalProvIdentStand/. Accessed 11 Aug 2016.
27. Aetna: the facts about rising health care costs. http://www.aetna.com/health-reform-connection/aetnasvision/facts-about-costs.html. Accessed 2015.
28. Rao RB, Fung G, Rosales R. On the dangers of Cross-Validation. An experimental evaluation. In: Proceedings of the 2008 SIAM international conference on data mining, 2008. Philadelphia: SIAM; p. 588–596.
29. Prashant Gupta: Cross-Validation in machine learning. https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f. Accessed 10 Sept 2018.

30. Zhang W, Kobeissi S, Tomko S, Challis C. Adaptive sampling scheme for learning in severely imbalanced large scale data. In: Asian conference on machine learning, 2017. p. 240–247.
31. Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G. Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl. 2017;73:220–39.
32. Lin S-C, Wang C, Wu Z-Y, Chung Y-F. Detect rare events via mice algorithm with optimal threshold. In: Seventh international conference on innovative mobile and internet services in ubiquitous computing (IMIS), 2013. Piscataway: IEEE; 2013. p. 70–75.
33. Alhammady H, Ramamohanarao K. Using emerging patterns and decision trees in rare-class classification. In: Null, 2004. Piscataway: IEEE; 2004. p. 315–318.
34. Li J, Liu L-S, Fong S, Wong RK, Mohammed S, Fiaidhi J, Sung Y, Wong KK. Adaptive swarm balancing algorithms for rare-event prediction in imbalanced healthcare data. PloS ONE. 2017;12(7):0180830.
35. Zhang X, Li Y, Kotagiri R, Wu L, Tari Z, Cheriet M. Krnn: k rare-class nearest neighbour classification. Pattern Recogn. 2017;62:33–44.
36. Zaharia M, Xin RS, Wendell P, Das T, Armbrust M, Dave A, Meng X, Rosen J, Venkataraman S, Franklin MJ, et al. Apache spark: a unified engine for big data processing. Commun ACM. 2016;59(11):56–65.
37. Dongre SS, Malik LG. Rare class problem in data mining. Int J Adv Res Comput Sci. 2017;8(7).
38. Hasanin T, Khoshgoftaar TM. The effects of random undersampling with simulated class imbalance for big data. In: 2018 IEEE international conference on information reuse and integration (IRI). Piscataway: IEEE; 2018. p. 70–79.
39. Meng X, Bradley J, Yavuz B, Sparks E, Venkataraman S, Liu D, Freeman J, Tsai D, Amde M, Owen S, et al. Mllib: Machine learning in apache spark. J Mach Learn Res. 2016;17(1):1235–41.
40. Rastogi AK, Narang N, Siddiqui ZA. Imbalanced big data classification: a distributed implementation of smote. In: Proceedings of the workshop program of the 19th international conference on distributed computing and networking, 2018. New York: ACM; 2018. p. 14.
41. Dong Q, Gong S, Zhu X. Imbalanced deep learning by minority class incremental rectification. IEEE Trans Pattern Anal Mach Intell. 2018. https://doi.org/10.1109/TPAMI.2018.2832629.
42. Zhai J, Zhang S, Wang C. The classification of imbalanced large data sets based on mapreduce and ensemble of elm classifiers. Int J Mach Learn Cybern. 2017;8(3):1009–17.
43. Tayal A, Coleman TF, Li Y. Rankrc: Large-scale nonlinear rare class ranking. IEEE Trans Knowl Data Eng. 2015;27(12):3347–59.
44. Maalouf M, Homouz D, Trafalis TB. Logistic regression in large rare events and imbalanced data: a performance comparison of prior correction and weighting methods. Comput Intell. 2018;34(1):161–74.
45. Chai KE, Anthony S, Coiera E, Magrabi F. Using statistical text classification to identify health information technology incidents. J Am Med Inform Assoc. 2013;20(5):980–5.
46. CMS: Medicare provider utilization and payment data: physician and other supplier. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Physician-and-Other-Supplier.html. Accessed 16 Aug 2016.
47. CMS: Medicare provider utilization and payment data: Part D prescriber. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Part-D-Prescriber.html. Accessed 21 Dec 2017.
48. CMS: Medicare provider utilization and payment data: referring durable medical equipment, prosthetics, orthotics and supplies. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/DME.html. Accessed 21 Dec 2017.
49. CMS Office of Enterprise Data and Analytics: Medicare Fee-For-Service Provider Utilization & Payment Data Physician and other supplier. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Downloads/Medicare-Physician-and-Other-Supplier-PUF-Methodology.pdf. Accessed 04 Sept 2018.
50. CMS office of Enterprise Data and Analytics: Medicare Fee-For Service Provider Utilization & Payment Data Part D Prescriber public use file: a methodological overview. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Downloads/Prescriber_Methods.pdf. Accessed 04 Sept 2018.
51. CMS Office of Enterprise Data and Analytics: Medicare Fee-For-Service Provider Utilization & Payment Data referring durable medical equipment, prosthetics, orthotics and supplies public use file: a methodological overview. https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/Medicare-Provider-Charge-Data/Downloads/DME_Methodology.pdf. Accessed 04 Sept 2018.
52. CMS: HCPCS—general information. https://www.cms.gov/Medicare/Coding/MedHCPCSGenInfo/index.html. Accessed 13 Jan 2018.
53. OIG: Office of Inspector General Exclusion Authorities US Department of Health and Human Services. https://oig.hhs.gov/. Accessed 05 Jan 2018.
54. LEIE: Office of Inspector General LEIE Downloadable Databases. https://oig.hhs.gov/exclusions/authorities.asp. Accessed 14 July 2016.
55. Pande V, Maas W. Physician Medicare fraud: characteristics and consequences. Int J Pharm Healthcare Market. 2013;7(1):8–33.
56. Van Hulse J, Khoshgoftaar TM, Napolitano A. Experimental perspectives on learning from imbalanced data. In: Proceedings of the 24th international conference on machine learning, 2007. New York: ACM; 2007. p. 935–942.
57. Japkowicz N. Concept-learning in the presence of between-class and within-class imbalances. In: Conference of the Canadian society for computational studies of intelligence, 2001. Berlin: Springer; 2001. p. 67–77.
58. Weiss GM. Mining with rarity: a unifying framework. ACM Sigkdd Explorations Newslett. 2004;6(1):7–19.
59. Chawla NV. Data mining for imbalanced datasets: an overview. In: Data mining and knowledge discovery handbook. Boston: Springer; 2009. p. 875–886.

60. Bauder RA, Khoshgoftaar TM, Hasanin T. Data sampling approaches with severely imbalanced big data for Medicare fraud detection. In: IEEE 30th international conference On tools with artificial intelligence (ICTAI), 2018. Piscataway: IEEE; 2018.
61. Bauder RA, Khoshgoftaar TM. Medicare fraud detection using Random Forest with class imbalanced big data. In: 2018 IEEE International conference on information reuse and integration (IRI), 2018. Piscataway: IEEE; 2018. p. 80–87.
62. Bengio Y, Grandvalet Y. No unbiased estimator of the variance of k-fold Cross-Validation. J Mach Learn Res. 2004;5(Sep):1089–105.
63. Varoquaux G. Cross-validation failure: small sample sizes lead to large error bars. Neuroimage. 2018;180:68–77.
64. Justin D: Overfitting, model selection, cross validation, bias-variance. https://people.cs.umass.edu/~domke/courses/sml2011/02overfitting.pdf. Accessed 27 Sept 2018.
65. Apache: Apache Spark. http://spark.apache.org/. Accessed 24 May 2018.
66. Apache: Apache Hadoop. http://hadoop.apache.org/. Accessed 24 May 2018.
67. Apache: Spark Release 2.3.0. http://spark.apache.org/releases/spark-release-2-3-0.html. Accessed 29 Nov 2018.
68. Apache: classification and regression. https://spark.apache.org/docs/2.3.0/ml-classification-regression.html. Accessed 19 Sept 2018.
69. Apache: linear methods-RDD-based API. https://spark.apache.org/docs/latest/mllib-linear-methods.html. Accessed 10 Sept 2017.
70. Apache Spark: ensembles-RDD-based API. https://spark.apache.org/docs/2.3.0/mllib-ensembles.html. Accessed 24 May 2018.
71. Bekkar M, Djemaa HK, Alitouche TA. Evaluation measures for models assessment over imbalanced data sets. J Inf Eng Appl. 2013;3(10).
72. Seliya N, Khoshgoftaar TM, Van Hulse J. A study on the relationships of classifier performance metrics. In: 21st international conference on tools with artificial intelligence, ICTAI'09, 2009. Piscataway: IEEE; 2009. p. 59–66.
73. Jeni LA, Cohn JF, De La Torre F. Facing imbalanced data–recommendations for the use of performance metrics. In: Humaine association conference on affective computing and intelligent interaction (ACII), 2013. Piscataway: IEEE; 2013. p. 245–251.
74. Gelman A, et al. Analysis of variance-why it is more important than ever. Ann Stat. 2005;33(1):1–53.
75. Tukey JW. Comparing individual means in the analysis of variance. Biometrics. 1949;5(2):99–114.
76. Le Cessie S, Van Houwelingen JC. Ridge estimators in logistic regression. J Roy Stat Soc: Ser C (Appl Stat). 1992;41(1):191–201.
77. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The weka data mining software: an update. ACM SIGKDD Explorations Newslett. 2009;11(1):10–8.
78. Witten IH, Frank E, Hall MA, Pal CJ. Data mining: practical machine learning tools and techniques. Burlington: Morgan Kaufmann; 2016.
79. le Cessie S, van Houwelingen JC. Ridge estimators in logistic regression. Appl Stat. 1992;41(1):191–201.
80. Breiman L. Random Forests. Mach Learn. 2001;45(1):5–32.