Florida Atlantic University

Data Mining & Mach Learning

CAP 6673-002

# Course term project
# Medicare fraud detection using neural networks*

*as the base line paper

*Author:*
Temirlan Kdyrkhan
Z23757665

April 17, 2024

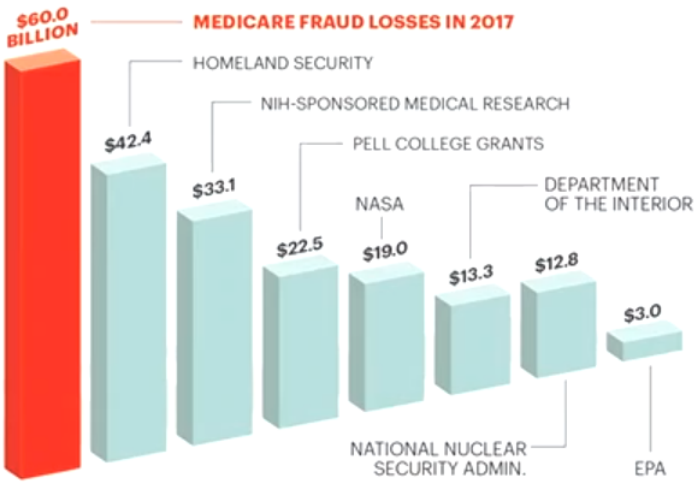
This report begins with summary of "Medicare fraud detection using neural networks" by Justin M. Johnson and Taghi M. Khoshgoftaar, then there will be provided details, program and results from the test performing the methods explained in original work, and finally some other experiments. Goal of program implementation is to get similar results stated in paper work (or different with possible reasons) and admit some critics and ideas. Real time code implementation always gives **more understanding** of research paper which was done on older data.

1. "Medicare fraud detection using neural networks" by Justin M. Johnson and Taghi M. Khoshgoftaar.
2. Program implementation and ideas.
3. Experiment 1.
4. Experiment 2.
5. Experiment 3.

Links for programs will be provided in each section below. Link:
https://github.com/TemirlanKN/cap6673_medicare_fraud_detection

# 1 "Medicare fraud detection using neural networks" by Justin M. Johnson and Taghi M. Khoshgoftaar

Medicare fraud detection using neural networks paper provides analysis, focusing on the challenge of class imbalance in machine learning models. The authors, Justin M. Johnson and Taghi M. Khoshgoftaar, highlight the significant problem of fraud, waste, and abuse within the Medicare system, which costs billions of dollars annually. Medicare, a critical component of the United States' healthcare system, provides insurance to individuals over 65 and those with specific disabilities. The total spending on Medicare and Medicaid exceeded $1.6 trillion in 2021 and continues to grow. To 2017 estimates suggest that fraud accounts for 3–10% of all Medicare billings, translating to financial losses between $21 and $71 billion annually.




In the study there were used 2 datasets:

1. Centers for Medicare & Medicaid Services (CMS) with Medicare Provider Utilization and Payment Data (PUF) dataset. Part B.

2. U.S. Department of Health and Human Services' Office of Inspector General (OIG) with List of Excluded Individuals and Entities (LEIE).

The primary data source used in the study is the **Medicare Provider Utilization and Payment Data (PUF)**, which contains information about services and procedures provided to Medicare beneficiaries by many providers. This dataset includes details on the utilization, payment and submitted charges organized by National Provider Identifier (NPI), Healthcare Common Procedure Coding System (HCPCS) code, and place of service. This sort of data offers a detailed overview of billing activities within the Medicare system, making it important for identifying cases with fraudulent behavior.

To label instances for fraud detection, the researchers use data from the **List of Excluded Individuals and Entities (LEIE)**, provided by the U.S. Department of Health and Human Services' Office of Inspector General (OIG). The LEIE database includes information about individuals and entities excluded from participation in Medicare, Medicaid, and all other Federal health care programs. Exclusions are handled for a variety of reasons, including convictions for Medicare or Medicaid fraud, patient abuse or neglect, felony convictions for other health care-related fraud, theft, or other financial misconduct.

**The main challenges in this problem are**:

1. Class Imbalance: A significant challenge in fraud detection research is the imbalance between the number of fraudulent cases and non-fraudulent. Fraudulent cases are much rarer compared to non-fraudulent ones, leading to imbalanced training data that can make machine learning model predict the majority class, reducing the effectiveness of fraud detection. Prepossessed dataset in research has class imbalance ratio is $99.97 : 0.03$.

2. Data Quality and Availability: The effectiveness of model in detecting fraud also depends on the quality of the dataset used for training. Issues such as missing data (will appear as NaN values), inaccuracies, and limited access to detailed fraud case information (some NPI numbers don't exist in the list of fraudulent cases) can prevent the development of robust models. In research authors mentioned that data quality can be improved by leveraging the NPPES registry to look up NPI numbers that are currently missing.

3. Feature Selection: medicare fraud include a wide range of practices and ways, making it hard to identify all possible patterns of fraud through a single model. Sometimes specific feature selection gives misleading results, for example predicting whether the transaction is fraudulent or not based on first and last name of the provider can make unpredicted results and give random answers.

4. Need for Specialized Data preprocessing techniques: with large volume and high dimensionality of healthcare data, there is a need for proper data preprocessing and feature extraction techniques to ensure that neural networks can learn effectively from the data. For example some numeric features like average submitted charge (Avg_Sbmtd_Chrg) can be easily handled by Neural Network, however categorical

features like provider type (Rndrng_Prvdr_Type), which contains more than 50 class variables needs some methods to be applied before giving it to model. Naive converting all provider type variables to list of numbers (for example: 1, 2, ..., 50) was proven to be wrong method.

The study explores **six deep learning methods** to address the issue of class imbalance, which severely affects the learning process and model performance. These methods are divided into data-level and algorithm-level approaches. Data-level techniques, including random over-sampling (ROS), random under-sampling (RUS), and a hybrid ROS-RUS approach, manipulate the training data to balance the class distribution. Algorithm-level techniques, such as cost-sensitive loss functions, mentioned in paper aim to adjust the learning process itself to be more sensitive to the minority class. The research evaluates these methods using the Medicare Part B dataset, labeling the dataset with fraud labels from the List of Excluded Individuals and Entities (LEIE) to train and test the neural network models.

The research paper is based on using Deep Neural Network architecture. They used the artificial neural network (ANN) with hidden layers to approximate some function f. In this case function f is constructed with two and four hidden layers, each one using 32 neurons, batch normalization, ReLU activation functions and Sigmoid activation function in the output layer.

Decision threshold whether the case is fraudulent or not is was held by threshold moving (a procedure to identify optimal decision boundaries using validation data). The optimal decision threshold is calculated for each of the ten validation models to balance TPR and TNR, averaged, and then applied to the test set.

The research performed evaluation of cost-sensitive learner, MFE loss and FL, as a modifications to the loss function that influence network weight updates by increasing the impact of the minority class during training.

**Various methods for addressing problems above have been introduced up until now**, including algorithm-level methods, data level methods, and hybrid methods:

1. Thresholding strategies for deep learning with highly imbalanced big data.

   Explores the significance of output thresholding in deep neural networks (DNNs) and suggests that the default threshold of 0.5 is not optimal. It was shown that Optimal thresholds (one that maximizes G-Mean, TPR, TNR) are strongly correlated with the positive class prior (prior probability of the positive class). **Main finding:** Prior thresholds perform consistently well across all distributions. And it was particularly noted for its simplicity and direct access from data without the need for optimization, making it a good preliminary choice for experiments with imbalanced data

2. Semantic embeddings for medical providers and fraud detection.

   Medical provider's specialty (significant predictor) was used to explore three (GloVe, Med-Word2Vec, HcpcsVec) techniques for representing medical provider types with dense, semantic embeddings that capture specialty similarities. Performed Principal Component Analysis to compare the performance of embedding sizes between 32-128. By using Logistic Regression (LR), Random Forest (RF), Gradient Boosted

Tree (GBT), and Multilayer Perceptron (MLP) learners each embedding technique was evaluated. **Main finding:** it was found that all three semantic embeddings significantly outperform one-hot representations when using RF and GBT learners.

3. Encoding techniques for high-cardinality features and ensemble learners.

   Evaluation of five encoding techniques for high-cardinality categorical features (healthcare procedure code feature with 7,752 unique values) using ensemble methods. It was found that inclusion of the categorical feature significantly improves performance for all ensemble learners, except when using one-hot representation. XGBoost learner with Hcpcs2Vec encodings achieved the highest average AUC of 0.8715.

4. Output thresholding for ensemble learners and imbalanced big data.

   Main focus is on output thresholding as a method to improve classification by adjusting the decision threshold for assigning class labels based on class probabilities. Ensemble Learners Performance: The study evaluates four tree-based ensemble learners (Random Forest, XGB, LightGBM, and CatBoost). The findings suggest that non-default thresholds significantly outperform the default threshold, and small changes to the threshold can lead to substantial differences in classification performance. The study recommends validating threshold strategies using a hold-out dataset to meet application requirements.

5. Cost-Sensitive Ensemble Learning for Highly Imbalanced Classification.

   Paper evaluates data-level and algorithm-level methods (by using ensemble learning algorithms: XGBoost, CatBoost, Random Forest) for handling class imbalance using cost-sensitive learning, which assigns different misclassification costs to classes. The results suggest that random undersampling and class weighting can improve classification when using a default threshold, but may decrease the discriminative power of models.

6. Random over sampling and random under sampling techniques.

   The effectiveness of RUS and ROS was proven in many researches combined with different types of learners. It was found that ROS and ROS–RUS performed significantly better than other methods, with ROS–RUS being the most efficient in terms of training time.

7. Also the same techniques were used in other field: threshold optimization approach for classifying imbalanced datasets, specifically using the Credit Card Fraud Detection Dataset. The findings suggest that the best results for selecting an optimal threshold are achieved without the use of RUS, and that the default threshold performs well at a balanced class ratio but not when the dataset is imbalanced (which was noted also in medicare fraud detection papers).

## 2 Program implementation

All program implementations were done in Python, using libraries and packages as Numpy, Pandas, Tensorfolw, sklearn, imblearn, matplotlib, seaborn, keras. PC specifications: 12th Gen Intel(R) Core(TM), i9-12900H, 2.50 GHz, 16GB RAM, shared GPU

memory 8GB. Programs names used in this section: "preprocessing_1.ipynb", "preprocessing_2.ipynb", "ROS_RUS.ipynb".

In the report we will provide only part of the code to highlight main idea of implementation, in order to see the whole code please check the original code.

As it was mentioned before datasets used in research are Medicare Provider Utilization and Payment Data from CMS and LEIE which are available in corresponding websites(press on name to open the link quickly). First one includes 3 datasets with providers information for each year between 2013 and 2021:

1. Medicare Physician & Other Practitioners - by Geography and Service
2. Medicare Physician & Other Practitioners - by Provider
3. Medicare Physician & Other Practitioners - by Provider and Service

For this specific experiment we will be working on Medicare **Physician & Other Practitioners - by Provider and Service** dataset, since it has the biggest number of instances in dataset, names of features in that dataset follows with features used in baseline paper and less number of NaN values for features used in research. **Idea:** use newly updated dataset and **compare the results**.

First we use dataset's columns and features listed in paper.

1. Provider Identification: Unique provider identification number (NPI).
2. Provider Specialty: Medical provider's specialty or practice type.
3. Provider Gender: Gender of the medical provider.
4. Service Metrics: Includes the number of services performed (line_srvc_cnt), number of distinct Medicare beneficiaries receiving the service (bene_unique_cnt), and number of distinct beneficiary/per day services (bene_day_srvc_cnt).
5. Charges and Payments: Average charges submitted for the service (average_submitted_chrg_amt) and average payment made to a provider per claim for the service (average_medicare_payment_amt).

These features are used to detect fraudulent activities by analyzing annual claims data grouped by provider. This study focuses on **procedure-level** attributes and excludes provider-level attributes like name and address.

When performing the experiment it was found to do preprocessing in 2 steps, for the reason of limited memory of PC.

## 2.1 First preprocessing step

In "preprocessing_1.ipynb".

When analyzing part B dataset from 2013 to 2021, all features names (columns in df) are identical. So first thing to do was to get all data with features (columns in data frame) for each year between 2013 and 2021. And since each of those datasets in CMS website include more than 8M instances, again it was found to perform preprocessing for each year separately (running the program or Jupyter kernel separately for each year).

1. First thing was data extraction with features listed in research paper.

```
df_features = ['Rndrng_NPI', 'Rndrng_Prvdr_Type', ...]
```

2. All data files were renamed in format:
Medicare_Physician_Other_Practitioners_by_Provider_and_Service_{year}.csv
and loaded with dropping all instances with provider gender feature having non-values.

```
file_path = base_path.format(...)
df_final.dropna(subset=['Rndrng_Prvdr_Gndr'], ...)
```

3. LEIE dataset was extracted by adjusting nearest year of the exclusion date since claims labeled as fraudulent dated prior to the provider's exclusion date. As in research we label providers as fraudulent for a given year if they are on the exclusion list for the majority of that year. And note instances in dataframe as fraudulent if their nearest year greater than or equal to year of the part B dataset was extracted.

```
df_leie['Nearest Year'] = df_leie.apply(lambda row: row['Year'] +
1 if row['Month'] >= 7 else row['Year'], axis=1)
...
df_leie['fraud'] = (df_leie['Nearest Year'] >= int(year_of_df))
...
```

4. Then we label part B dataset instances as fraudulent if their NPI number was found in LEIE fraud list.

```
filtered_leie = df_leie[df_leie['fraud'] == 1]
...
df_final['fraud'] = df_final['NPI'].isin(unique_npi_leie_filtered)
```

5. Provider type and Provider gender must be properly encoded before giving the data to neural network. We use one-hot encoding for provider type and provider gender.

```
df_final = pd.get_dummies(df_final, columns=columns_to_encode)
```

However we want to emphasize that one-hot encoding performed in research is risky to use since it increases the dimensionality of the dataset ("curse of dimensionality" extremely sparse and high-dimensional data leads risk of overfitting in some models), which can make the model training process more computationally expensive and slower. Resulting data matrix after one-hot encoding is very sparse (mostly zeros), which is inefficient for storage and computation. One-hot encoding treats each category as equally similar to others. This means that any ordinal relationships or hierarchies inherent in the categorical data (e.g., 'high', 'medium', 'low') are lost, as the encoded features are independent of each other.

6. Then we apply aggregation on numeric values with {mean, sum, median, std, min, max} and normalize them. Then we apply maximum function on categorical values, since we want all providers to have their gender listed in dataset.

```
agg_funcs = {
    'Tot_Benes': ['mean', 'sum', 'median', 'std', 'min', 'max'],
    ...
```

```
}
...
df_aggregated_2 = df_final.groupby('Rndrng_NPI').agg(agg_funcs)...
```

7. When having data ready we save it to csv file for the second preprocessing step.

```
merged_df.to_csv(year_of_df+".csv", index=False)
```

## 2.2 Second preprocessing step

In "preprocessing_2.ipynb".

When all data preprocessed in first step we have "{year}.csv" data for each year between 2013 and 2021.

1. Merge all data collected in first step into one df.

```
files = [
    '2013.csv', '2014.csv', '2015.csv',
    '2016.csv', '2017.csv', '2018.csv',
    '2019.csv', '2020.csv', '2021.csv'
...
df_final = pd.concat(dataframes, ignore_index=True)
]
```

2. There was found that when aggregating numeric data with standard deviation in first step some values become NaN. **It's not clearly stated in paper if authors faced this situation, but we apply dropping non-values considering if research paper experiments also dropped instances with NaN values.**

```
df_final.fillna(0, inplace=True)
...
```

Finally after all steps performed we have **8665935 rows and 153 columns with 8118 instances labeled as fraudulent (0.0937%)** . In paper research **stated 125 features, with 4,692,370 samples, 1508 labeled as fraudulent**, we assume differences in code implementation and availability newer data, having new provider class type variables for the new data to be the reasons for different numbers.

Data was saved as csv file in "data_set_result_yearly.csv".

**Idea:** after having some memory errors we also aggregated the data again by NPI (basically having new df with all unique NPI) to have smaller dataset, and perform experiment. Basically it means that we don't care anymore if provider had exclusion date, since we mark all providers who had been ever listed in LEIE with fraud activity. Data was saved in "data_set_result_aggregated_by_npi.csv" with **1492937 rows and 153 columns.**

## 2.3 Model training and evaluation on data

In "ROS_RUS.ipynb".

Neural network model used in study have 2 layers with **rectified linear unit activation functions**, **and sigmoid function in the output layer**. We applied the

Sequential model in Keras as it is very straightforward to use and suitable for a stack of layers where each layer has exactly one input tensor and one output tensor.

In paper choice of 32 neurons in hidden layers gave enough capacity to overfit the model to training data, **ideally** would be good to test different number of hidden layers and number of neurons to see how model would perform and demonstrate the results. Another idea is to choose another activation functions, since except the ReLU and Sigmoid functions, we can apply "tanh", "Leaky ReLU", "ELU", etc.

```
...
model.fit(X_train_scaled, y_train, epochs=50,
          batch_size=32, validation_split=0.1)
...
```

```
Epoch 1/50
194984/194984 — 136s 686us/step — accuracy: 0.9983 —
loss: 0.0112 — val_accuracy: 0.9991 — val_loss: 0.0094
...
Epoch 50/50
194984/194984 — 136s 696us/step — accuracy: 0.9989 —
loss: 0.0181 — val_accuracy: 0.9991 — val_loss: 0.0083
...
ROC AUC Score: 0.6923290024577556
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 1731563 |
| 1 | 0.00 | 0.00 | 0.00 | 1624 |

The model was trained for 50 epochs. During training, both training and validation accuracy (accuracy), and loss (loss) were tracked. The same was done for the validation dataset (val_accuracy and val_loss). The accuracy remains consistently high at around 0.9990 across all epochs for both training and validation datasets. However model achieved a ROC-AUC Score of approximately 0.6923. Given the high accuracy reported, this suggests that the accuracy might indeed was **misleading** due to class imbalance. The precision, recall, and F1-score for the minority class (1) are all zero, which indicates that model failed to correctly predict any of the minority class instances with having 0.5 as the default threshold.

## 2.4 RUS

In this section we will be using under sampling technique on majority class to discard millions instances and increase the model training time. However it needs no be noted that in research paper it was shown that both the class imbalance level and the representation of the majority class are important. When performing RUS technique after majority class size equal to 99 the ROC-AUC score started to decrease:

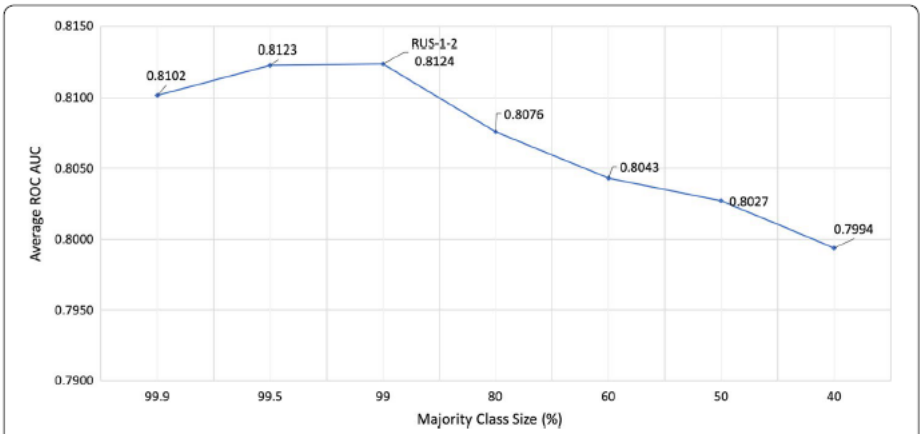



**Fig. 1** RUS: majority class size vs average AUC (30 runs)

We expect to see the same performance. We apply RUS methods on 2 layer neural network with class imbalance ratio $n_{neg} : n_{pos} = 99 : 1, 80 : 20, 60 : 40, 50 : 50, 40 : 60$.

Additionally we apply thresholding technique to determine which threshold is effective for each under sampled data. Author observed in research paper that the **most optimal threshold is approximately the same as the minority class size**, so we apply that for all models.

| Method | n_neg | n_pos | n_neg:n_pos | Decision Threshold | AUC | TPR | TNR | G-Mean |
|---|---|---|---|---|---|---|---|---|
| RUS_1 | 803,682 | 8,118 | 99:1 | 0.01 | 0.8204 | 0.7444 | 0.7444 | 0.7444 |
| RUS_2 | 32,472 | 8,118 | 4:1 | 0.268 | 0.8315 | 0.6336 | 0.8433 | 0.7309 |
| RUS_3 | 12,177 | 8,118 | 3:2 | 0.42 | 0.8354 | 0.7611 | 0.7486 | 0.7548 |
| RUS_4 | 8,118 | 8,118 | 1:1 | 0.497 | 0.8378 | 0.7573 | 0.7598 | 0.7585 |
| RUS_5 | 5,412 | 8,118 | 2:3 | 0.573 | 0.8289 | 0.7727 | 0.7393 | 0.7558 |

We can see almost the same pattern that happened in original paper, with decreasing number of majority class AUC score starts increase and after certain number it showed decreasing AUC score (in paper it was after RUS-2-2, here we see after RUS_4). Another point is that in this experiment decision threshold we choose led us to better TNR, ROC-AUC and G-Mean scores than in research paper. For example: for RUS_3 AUC here is 0.8378 compared with RUS-3-2 AUC score equal to 0.8043, however in experiment we have TPR as 0.7611, in research paper it was stated 0.7783 which is slightly better. Main reason for better performance might be bigger and "richer" dataset (almost 2 times more instances than one showed in research paper).

## 2.5 ROS

Random over sampling technique requires to increase the size of minority class. ROS method was done on PC with specifications mentioned before, and it was not possible to get reasonable performance results that could give better understanding due to limited memory and high dimensionality of data. So we skipped this part. Same situation for ROS-RUS.

# 3 Experiment 1

In "Experiment 1 and 2.ipynb".

In this section we will be using data saved in "data_set_result_aggregated_by_npi.csv" with 1492937 rows and 153 columns. All providers information collected for each year was grouped by NPI number and labeled as 'fraud' if they were ever listed in LEIE dataset. The goal is to evaluate the performance and compare it. It is important to mention that model training and performance we will get might be misleading since not all providers who provided any service after their exclusion date would make any more fraudulent services, or otherwise some providers after their exclusion date might still continue perform fraudulent activity.

Like in previous experiment we performed 5 RUS methods:

| Method | n_neg | n_pos | n_neg:n_pos | Decision Threshold | AUC | TPR | TNR | G-Mean |
|---|---|---|---|---|---|---|---|---|
| initial | 1,492,937 | 1,937 | 999:1 | 0.001 | 0.8261 | 0.8552 | 0.5834 | 0.7064 |
| RUS_1 | 191,763 | 1,937 | 99:1 | 0.01 | 0.8628 | 0.7803 | 0.7979 | 0.7891 |
| RUS_2 | 7,748 | 1,937 | 4:1 | 0.2 | 0.8315 | 0.6336 | 0.8433 | 0.7309 |
| RUS_3 | 2,905 | 1,937 | 3:2 | 0.4 | 0.8651 | 0.7835 | 0.7486 | 0.7986 |
| RUS_4 | 1,937 | 1,937 | 1:1 | 0.5 | 0.8715 | 0.8372 | 0.7598 | 0.7912 |
| RUS_5 | 1,937 | 1,291 | 2:3 | 0.573 | 0.8527 | 0.8067 | 0.7596 | 0.7828 |

Results show that all the metrics performed pretty good, having the best AUC score with RUS_4 equal to 0.8715 and TPR equal to 0.8372. To better understand if we trained the correct model which really can identify fraudulent cases, it would be reasonable to check performance on "data_set_result_yearly.csv". Since the current model was build on cases where we labeled all instances from part B dataset that ever appeared in LEIE dataset.

Additionally it is **important** to choose and find optimal random seed when splitting the data, since on RUS_3 with random_state parameter in train_test_split function we had 0.5 AUC score.

## 4 Experiment 2

In "Experiment 1 and 2.ipynb".

Here we evaluate the model performance trained in previous experiment, test it on "data_set_result_yearly.csv" dataset which contains 8665935 rows and 153 columns with 8118 instances labeled as fraudulent (0.0937%).

1. First we load the both data and train the model on data "data_set_result_aggregated_by_npi.csv" lets say **Data A**.

```
df_final = pd.read_csv("data_set_result_aggregated_by_npi.csv")
df_final_1 = pd.read_csv("data_set_result_yearly.csv")
...
```

2. Then we get predict model on "data_set_result_yearly.csv" lets say **Data B**.

```
X_1 = df_final_1.drop(['fraud', 'rndrng_npi'], axis=1)
y_1 = df_final_1['fraud']
...
predictions_1 = model.predict(X_1_scaled)
...
```

3. As the result we have:

```
ROC AUC Score: 0.7627935523863829
True Positive Rate (TPR): 0.7099039172209904
True Negative Rate (TNR): 0.6913404383576137
Geometric Mean (G-Mean): 0.7005606935329348
              precision    recall  f1-score   support

           0       1.00      0.69      0.82   8657817
           1       0.00      0.71      0.00      8118
```

   ROC AUC Score 0.76 happens to be the second worst we had after AUC=0.692 where we had the model which predicted only negative class. We can come to conclusion that model evaluation on data that was labeled differently was wrong choice. We will keep this report for future work in case if we validate the idea that providers who have ever caught with fraudulent transaction will likely perform fraudulent transactions again, or if they are performing fraudulent services without even being caught.

## 5 Experiment 3

In "Experiment 3.ipynb".

In this experiment we will evaluate the same 2 layer NN we applied before on "Medicare Physician & Other Practitioners - by Provider" dataset, which includes some provider information and additionally include information of beneficiaries: NPI number, provider type and gender, total submitted charge, total medicare payment amount (after deductible), beneficiaries age, age ranges, total number of male, female for corresponding provider NPI and number of beneficiaries with specific chronic deceases, race, average risk score.

Our main goal here is to properly perform preprocessing of data, evaluate the performance of the model on 3 groups selected features.

(**This work is for experimental purposes and to understand the impact on the model's performance.** When using sensitive features like race this step is strictly for research and aims to contribute to the development better models, algorithms. We understand the sensitivity of data and are committed to keeping the highest ethical standard to prevent any form of discrimination.)

1. Main features: NPI, provider type, provider gender, total submitted charge, total medicare payment amount, beneficiaries average age, beneficiaries average risk score.

2. Additional features 1: Number of beneficiaries with age ranges less than 65 y.o., from 65 to 74, from 75 to 84, above the 84 y.o., number of male and female beneficiaries, non-Hispanic white, non-Hispanic black or African American, Asian Pacific Islander, Hispanic, American Indian or Alaska Native and other beneficiaries with race not elsewhere classified.

3. Additional features 2: Percent of beneficiaries meeting the CCW chronic condition algorithm for atrial fibrillation, Alzheimer, Asthma, chronic condition algorithms for cancer, heart failure, chronic kidney disease, chronic obstructive pulmonary disease, depression, diabetes, hyperlipidemia, hypertension, ischemic heart disease,

osteoporosis, rheumatoid arthritis/osteoarthritis, schizophrenia and other psychotic disorders, stroke.

The algorithm follows almost everything of what was in first experiment. We loaded Medicare Physician & Other Practitioners - by Provider dataset from 2021 to 2013, descending order is important since we want to consider only new instances (providers) with new unseen data. If applied data from 2013 to 2021 it would save old data of provider. Then we loaded LEIE dataset and labeled data with fraud label, dropped NaN values for provider gender attribute and ended up with replaced values in numeric values (except charges amount and payment amount since there were no NaN values) with **median** value. One-hot encoding was performed on features provider gender and provider type. Normalization (max-min) for numeric attributes. In total we had 1555318 instances with 1987 (0.001 of total, same as 0.1%) being labeled as fraudulent, and 152 features.

Results:

| Data | AUC | TPR | TNR | G-Mean |
|---|---|---|---|---|
| Main | 0.7291 | 0.7052 | 0.7398 | 0.7223 |
| Main + features 1 | 0.7357 | 0.7531 | 0.604 | 0.6744 |
| Main + features 2 | 0.7898 | 0.7884 | 0.6459 | 0.7136 |
| All | 0.7893 | 0.7405 | 0.6968 | 0.7183 |

It shows that main features of provider with included provider type, gender performed as worst in terms of AUC, when then including features of beneficiaries ages and race, it shows slight better result, however main features with beneficiaries information about their decease highly increased the performance. And there is no significant difference between all features included and main features combined with beneficiaries decease features together, suggesting that information about race and ages of provider's customers doesn't impact so much to performance. As the possible future projects we think that feature selection we proposed would greatly work with ROS-RUS techniques.

We want note that authors of research mostly used Gradient Boosted Decision trees in their recent projects, it would be suggested to continue research with NN by applying different techniques, hidden layers and activation functions manipulation and to look up the performance change.

## 6 Conclusion

As a conclusion we covered issues, analysis and techniques about class imbalance in machine learning models, in the context of Medicare fraud, which involves billions of dollars annually. Addressed key challenges including class imbalance, data quality and availability, feature selection, and the need for specialized data preprocessing techniques. We also implemented the model training following the research on new fresh Part B dataset resulting better AUC score 0.8378 compared to 0.8043 taken from research base paper. By step by step code implementation we highlighted

importance of data quality. We employed under-sampling technique to handle class imbalance. Showed different way than in research paper of aggregating the data with given list of features. Demonstrated performance metrics comparison across them, showing the importance of feature selection about providers information, race and age of beneficiaries and chronic decease information of beneficiaries. In each section we provided critics and out thoughts of possible methods that could make better performance.

## References


[1] Leevy, J.L., Johnson, J.M., Hancock, J. et al. Threshold optimization and random undersampling for imbalanced credit card data. J Big Data 10, 58 (2023). https://doi.org/10.1186/s40537-023-00738-z

[2] Johnson, J.M., Khoshgoftaar, T.M. (2021). Thresholding Strategies for Deep Learning with Highly Imbalanced Big Data. In: Wani, M.A., Khoshgoftaar, T.M., Palade, V. (eds) Deep Learning Applications, Volume 2. Advances in Intelligent Systems and Computing, vol 1232. Springer, Singapore. https://doi.org/10.1007/978-981-15-6759-9_9

[3] J. M. Johnson and T. M. Khoshgoftaar, "Semantic Embeddings for Medical Providers and Fraud Detection," 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), Las Vegas, NV, USA, 2020, pp. 224-230, doi: 10.1109/IRI49571.2020.00039.

[4] J. M. Johnson and T. M. Khoshgoftaar, "Encoding Techniques for High-Cardinality Features and Ensemble Learners," 2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI), Las Vegas, NV, USA, 2021, pp. 355-361, doi: 10.1109/IRI51335.2021.00055.

[5] J. M. Johnson and T. M. Khoshgoftaar, "Output Thresholding for Ensemble Learners and Imbalanced Big Data," 2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI), Washington, DC, USA, 2021, pp. 1449-1454, doi: 10.1109/ICTAI52525.2021.00230.

[6] J. M. Johnson and T. M. Khoshgoftaar, "Cost-Sensitive Ensemble Learning for Highly Imbalanced Classification," 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 2022, pp. 1427-1434, doi: 10.1109/ICMLA55696.2022.00225.

[7] J. Hancock, T. M. Khoshgoftaar and J. M. Johnson, "The Effects of Random Undersampling for Big Data Medicare Fraud Detection," 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE), Newark, CA, USA, 2022, pp. 141-146, doi: 10.1109/SOSE55356.2022.00023.

[8] Johnson, J.M., Khoshgoftaar, T.M. Encoding High-Dimensional Procedure Codes for Healthcare Fraud Detection. SN COMPUT. SCI. 3, 362 (2022). https://doi.org/10.1007/s42979-022-01252-4

[9] https://github.com/TemirlanKN/cap6673_medicare_fraud_detection

[10] Medicare provider utilization and payment data.

[11] List of Excluded Individuals and Entities.