PROJECT REPORT

FRAUD DETECTION IN MEDICARE

SUBJECT: DAB322 CAPSTONE PROJECT I

Submitted by:

Davinder Kaur (W0813395)

Ronaldo Chan Fu Yi (W0831976)

Simerjeet Kaur (W0814800)

1 INTRODUCTION

Health is a fundamental human right: “The enjoyment of the highest attainable standard of health is one of the fundamental rights of every human being without distinction of race, religion, political belief, economic or social condition” (WHO, 2017).

Advances in technology, medicine, new drugs and vaccines is propelling populations to have a higher life expectancy. United States doesn’t have universal health coverage provided by the government, but for a few eligible beneficiaries, it is offered federal health insurance program: Medicare.

As a federal funded program, Medicare relies on tax contributions from the population to foment its service. The significant issue of fraud, waste, and abuse in the U.S. healthcare system estimate it to cost around $700 billion, hence fraud in healthcare systems increases the healthcare cost for the population, thus fraud detection and prevention are present-day challenges.

Our project is to propose a machine learning model to streamline the process to identify fraud and eligibility using publicized datasets made available by US federal agencies: Center for Medicare and Medicaid Services (CMS) and Office of Inspector General (OIG).

1.1 MEDICARE

Medicare stands as a federal health insurance program for: people who are 65 or older, certain younger people with disabilities and people with end-stage renal disease. Established in 1965, Medicare is administered by the Centers for Medicare & Medicaid Services (CMS) and provides essential healthcare coverage to millions of Americans.

Medicare is composed of several parts, each addressing different aspects of healthcare:

Medicare Part A (Hospital Insurance):

Part A covers inpatient hospital care, skilled nursing facility care, hospice care, and some home health services.

Medicare Part B (Medical Insurance):
Helps cover: Services from doctors and other health care providers, outpatient care, home health care, durable medical equipment (like wheelchairs, walkers, hospital beds, and other equipment) and many preventive services (like screenings, shots or vaccines, and yearly “Wellness” visits)

Medicare Part C (Medicare Advantage): Medicare Advantage is a Medicare-approved plan from a private company that offers an alternative to Original Medicare for your health and drug coverage. These “bundled” plans include Part A, Part B, and usually Part D.

Medicare Part D (Prescription Drug Coverage): Helps cover the cost of prescription drugs (including many recommended shots or vaccines). You join a Medicare drug plan in addition to Original Medicare, or you get it by joining a Medicare Advantage Plan with drug coverage. Plans that offer Medicare drug coverage are run by private insurance companies that follow rules set by Medicare.

"**Fee-for-service**" (FFS) is a payment model in healthcare where providers are paid by the government pays directly the providers for the health care for each service they delivered. In the context of Medicare Part B, which covers outpatient care and various medical services, fee-for-service means that healthcare providers are reimbursed by Medicare for each individual service or procedure they perform.
Here's a breakdown of the fee-for-service model in the context of Medicare Part B:

1. Individual Payments for Each Service: In a fee-for-service system, healthcare providers bill Medicare for each specific service they provide to a beneficiary. These services can include office visits, laboratory tests, surgeries, and other medical procedures.
2. Payment Based on Fee Schedule: Medicare sets a fee schedule that outlines the maximum amount it will reimburse for each covered service. Providers are paid up to the established fee for each service they deliver. The fee schedule may vary based on factors such as geographic location.

3. Flexibility for Beneficiaries: Fee-for-service allows Medicare beneficiaries to choose their healthcare providers and access a wide range of medical services. They can see any healthcare professional who accepts Medicare, and the program will pay for the covered services.
4. Potential for Overutilization: One criticism of the fee-for-service model is that it may incentivize overutilization of services. Since providers are reimbursed for each service, there's a concern that they may be motivated to perform unnecessary tests or procedures to increase their reimbursement.
5. Traditional Medicare vs. Medicare Advantage: Medicare Part B operates on a fee-for-service basis, but it's important to note that there are alternatives, such as Medicare Advantage plans. These plans may use different payment models, such as capitation, where the plan receives a fixed amount per beneficiary, regardless of the number of services provided.

1.2 MEDICARE FRAUD

Some of the fraud schemes are as follows (not exhaustive list):

- Billing for Unnecessary Services or Items: Intentionally billing for unnecessary medical services or items.
- Billing for Services or Items Not Provided (Phantom billing): Intentionally billing for services or items not provided.
- Billing for mutually exclusive procedures: Billing procedures that cannot reasonably be performed together based on code definition or anatomic considerations.
- Duplicate claims: Multiple claims for the same healthcare service they provide to a patient on a specific date of service.
- Unbundling: Billing for multiple codes for a group of procedures that are covered in a single global billing code.
- Upcoding: Billing for services at a higher level of complexity than provided.
- Card Sharing: Knowingly treating and claiming reimbursement for someone other than the eligible beneficiary.

- Collusion: Knowingly collaborating with beneficiaries to file false claims for reimbursement.
- Drug Diversion: Writing unnecessary prescriptions, or altering prescriptions, to obtain drugs for personal use or to sell them.
- Overprescribing: Where patients are being prescribed medicines that are inappropriate or no longer necessary, or where harms outweigh benefits.
- Kickbacks: Offering, soliciting, or paying for beneficiary referrals for medical services or items.
- Multiple Cards: Knowingly accepting multiple Medicaid ID cards from a beneficiary to claim reimbursement.
- Program Eligibility: Knowingly billing for an ineligible beneficiary.

1.3 LEIE

The Office of Inspector General (OIG) has the authority to exclude individuals and entities from Federally funded health care programs for a variety of reasons, including a conviction for Medicare or Medicaid fraud. Those that are excluded can receive no payment from Federal health care programs for any items or services they furnish, order, or prescribe. This includes those that provide health benefits funded directly or indirectly by the United States (other than the Federal Employees Health Benefits Plan). OIG maintains a list of all currently excluded individuals and entities called: List of Excluded Individuals/Entities (LEIE).

1.4 STATE MEDICAID EXCLUSION LIST

Medicaid is a joint federal and state program that helps cover medical costs for some people with limited income and resources. The federal government has general rules that all state Medicaid programs must follow, but each state runs its own program.
In addition to LEIE, we have states Medicaid exclusion list. Therefore, if a provider or entity is excluded under any state Medicaid program, that provider should be excluded

from participating in all state Medicaid programs. Out of 50 states in the US, we gathered 42 datasets.

2 OBJECTIVE

Create a supervised machine learning model to predict Medicare providers eligibility enlisted in Medicare PART B FFS data based on a target feature based on LEIE, OnR and Medicaid State Board exclusion lists.

3 METHODS

For this project we will be working with a big volume dataset from Medicare part B dataset: 9,886,177 rows. Our project is limited by computing power, hence working will a huge load of data will be an additional challenge for our project.

Since this dataset is solely information on services and procedures provided to Original Medicare (fee-for-service) Part B (Medical Insurance) beneficiaries by physicians and other healthcare professionals, we will use additional datasets to create our target feature eligibility. Integrating the additional datasets with Medicare dataset is expected to create few matches relative to the entirety of the data, hence working with an imbalanced dataset will be another difficulty. For this issue, we will try under sampling and over sampling techniques to enhance our model metrics.

Working with big volume dataset it is expected huge variability for categorical and numerical features, creating additional challenge for data cleaning and wrangling. For categorical features, we need to transform the data to a more suitable form for use, in this case, we will try Label encoding and One Hot encoding. When using One Hot encoding we might face the hyperdimensional problem that will result in a sparse dataset. The next step is screening supervised machine learning models with the following conditions:

- Label encoding x Under sampling.
- Label encoding x Over sampling.
- One Hot encoding x Under sampling.

- One Hot encoding x Over sampling.

A model with the best performance will be selected for further improvements through hyperparameter tuning and lastly, it will be trained and tested with the compiled dataset.

3.1 DATASETS

3.1.1 MEDICARE DATASET

For this project, we will be using the public available dataset: Medicare Physician & Other Practitioners - by Provider and Service with information on services and procedures provided to Original Medicare (fee-for-service) Part B (Medical Insurance) beneficiaries by physicians and other healthcare professionals; aggregated by provider and service.

It is important to notice that Medicare dataset has 29 columns with National Provider Identifier (NPI) and FFS data. NPI will be our key to match with the other datasets when creating target variable and FFS will give numbers to identify possible frauds or improper payments.

Dataset shape: 9.886.177 rows, 29 columns

Data dictionary for Medicare Part B:

Feature	Description
Rndrng_NPI	National Provider Identifier (NPI) for the rendering provider on the claim. The provider NPI is the numeric identifier registered in NPPES.
Rndrng_Prvdr_Last_Org_Name	When the provider is registered as an organization (entity type code = ‘O’), this is the organization name.
Rndrng_Prvdr_First_Name	When the provider is registered in NPPES as an individual (entity type code=’I’), this is the provider’s first name.

Rndrng_Prvdr_MI	When the provider is registered in NPPES as an individual (entity type code='I'), this is the provider's middle initial.
Rndrng_Prvdr_Crdntls	When the provider is registered in NPPES as an individual (entity type code='I'), these are the provider's credentials.
Rndrng_Prvdr_Gndr	When the provider is registered in NPPES as an individual (entity type code='I'), this is the provider's gender.
Rndrng_Prvdr_Ent_Cd	type of entity reported in NPPES. An entity code of 'I' identifies providers registered as individuals and an entity type code of 'O' identifies providers registered as organizations.
Rndrng_Prvdr_St1	The first line of the provider's street address, as reported in NPPES.
Rndrng_Prvdr_St2	The second line of the provider's street address, as reported in NPPES.
Rndrng_Prvdr_City	The city where the provider is located, as reported in NPPES.
Rndrng_Prvdr_State_Abrvtn	The state where the provider is located, as reported in NPPES. The fifty U.S. states and the District of Columbia are reported by the state postal abbreviation.
Rndrng_Prvdr_State_FIPS	FIPS code for rendering provider's state.
Rndrng_Prvdr_Zip5	The provider's zip code, as reported in NPPES.
Rndrng_Prvdr_RUCA	Rural-Urban Commuting Area Codes (RUCAs), are a Census tract-based classification scheme that utilizes the standard Bureau of Census Urbanized

	Area and Urban Cluster definitions in combination with work commuting information to characterize all of the nation's Census tracts regarding their rural and urban status and relationships.
Rndrng_Prvdr_RUCA_Desc	Description of Rural-Urban Commuting Area (RUCA) Code
Rndrng_Prvdr_Cntry	The country where the provider is located, as reported in NPPES.
Rndrng_Prvdr_Type	Derived from the provider specialty code reported on the claim.
Rndrng_Prvdr_Mdcr_Prtcptg_Ind	Identifies whether the provider participates in Medicare and/or accepts assignment of Medicare allowed amounts.
HCPCS_Cd	HCPCS code used to identify the specific medical service furnished by the provider
HCPCS_Desc	Description of the HCPCS code for the specific medical service furnished by the provider
HCPCS_Drug_Ind	Identifies whether the HCPCS code for the specific service furnished by the provider is a HCPCS listed on the Medicare Part B Drug Average Sales Price (ASP) File.
Place_Of_Srvc	Identifies whether the place of service submitted on the claims is a facility (value of ‘F’) or non-facility (value of ‘O’).
Tot_Benes	Number of distinct Medicare beneficiaries receiving the service for each

	Rndrng_NPI, HCPCS_Cd, and Place_Of_Srvc.
Tot_Srvcs	Number of services provided; note that the metrics used to count the number provided can vary from service to service.
Tot_Bene_Day_Srvcs	Number of distinct Medicare beneficiary/per day services. Since a given beneficiary may receive multiple services of the same type (e.g., single vs. multiple cardiac stents) on a single day, this metric removes double-counting from the line service count to identify whether a unique service occurred.
Avg_Sbmtd_Chrg	Average of the charges that the provider submitted for the service.
Avg_Mdcr_Alowd_Amt	Average of the Medicare allowed amount for the service; this figure is the sum of the amount Medicare pays, the deductible and coinsurance amounts that the beneficiary is responsible for paying, and any amounts that a third party is responsible for paying.
Avg_Mdcr_Pymt_Amt	Average amount that Medicare paid after deductible and coinsurance amounts have been deducted for the line item service.
Avg_Mdcr_Stdzd_Amt	Average amount that Medicare paid after beneficiary deductible and coinsurance amounts have been deducted for the line item service and after standardization of the Medicare payment has been applied.

Table 1 Medicare Part B dictionary

In figure 1 we can notice that the dataset is highly detailed, showing each and every service by the provider, thus we will simplify the dataset grouping the columns. A provider can have the same name as another, therefore we will be grouping as per NPI, since it is unique and will be used as our primary key to link with the other datasets.

	Rndrng_NPI	Rndrng_Prvdr_Last_Org_Name	Rndrng_Prvdr_First_Name	HCPCS_Cd	HCPCS_Desc	Avg_Mdcr_Pymt_Amt
0	1003000126	Enkeshafi	Ardalan	99213	Established patient outpatient visit, total ti...	83.908220
1	1003000126	Enkeshafi	Ardalan	99214	Established patient outpatient visit, total ti...	118.570638
2	1003000126	Enkeshafi	Ardalan	99217	Hospital observation care on day of discharge	61.066923
3	1003000126	Enkeshafi	Ardalan	99220	Hospital observation care, typically 70 minutes	141.442857
4	1003000126	Enkeshafi	Ardalan	99222	Initial hospital inpatient care, typically 50 ...	105.700833
5	1003000126	Enkeshafi	Ardalan	99223	Initial hospital inpatient care, typically 70 ...	170.388889
6	1003000126	Enkeshafi	Ardalan	99226	Subsequent observation care, typically 35 minu...	84.338125
7	1003000126	Enkeshafi	Ardalan	99231	Subsequent hospital inpatient care, typically ...	31.255862
8	1003000126	Enkeshafi	Ardalan	99232	Subsequent hospital inpatient care, typically ...	58.462108
9	1003000126	Enkeshafi	Ardalan	99233	Subsequent hospital inpatient care, typically ...	84.875327
10	1003000126	Enkeshafi	Ardalan	99238	Hospital discharge day management, 30 minutes ...	59.603478
11	1003000126	Enkeshafi	Ardalan	99239	Hospital discharge day management, more than 3...	86.786378
12	1003000126	Enkeshafi	Ardalan	99454	Remote monitoring of physiologic parameters, i...	63.303729
13	1003000126	Enkeshafi	Ardalan	99457	Remote physiologic monitoring treatment manage...	48.005374
14	1003000126	Enkeshafi	Ardalan	99458	Remote physiologic monitoring treatment manage...	38.270000

Figure 1 Medicare dataset

For Categorical features it will keep unique values, i.e., if HCPCS_Cd is more than one is present, it will join the unique entries creating a new entry. For Numerical features, Tot_Benes, Tot_Srvcs and Tot_Bene_Day_Srvcs we will sum and for the other numerical features, we are using weighted average with Tot_Bene_Day_Srvcs as weight. Figure 2 demonstrates how the compilation turns out.

Aggregated dataset shape: 1.123.589 rows, 29 columns

	NPI	Last_Org_Name	First_Name	HCPCS_Cd	HCPCS_Desc	Avg_Mdcr_Pymt_Amt
0	1003000126	Enkeshafi	Ardalan	99213, 99214, 99217, 99220, 99222, 99223, 9922...	Established patient outpatient visit, total ti...	61.441327
1	1003000134	Cibull	Thomas	88304, 88305, 88312, 88313, 88321, 88341, 8834...	Pathology examination of tissue using a micros...	27.620868
2	1003000142	Khalil	Rashid	62323, 64483, 64484, 64490, 64491, 64493, 6449...	Injection of substance into spinal canal of lo...	71.649996
3	1003000423	Velotta	Jennifer	81002, G0101, Q0091	Urinalysis, manual test, Cervical or vaginal c...	31.303448
4	1003000480	Rothchild	Kevin	99202, 99203, 99212, 99213	New patient outpatient visit, total time 15-29...	48.870698
5	1003000530	Semonche	Amanda	81002, 90662, 90670, 90732, 93000, 99213, 9921...	Urinalysis, manual test, Vaccine for influenza...	96.499037
6	1003000597	Kim	Dae	50590, 51102, 51700, 51702, 51705, 51798, 5200...	Shock wave crushing of kidney stones, Aspirati...	73.625755
7	1003000639	Benharash	Peyman	99205	New patient outpatient visit, total time 60-74...	161.090000
8	1003000704	Gatton	Zachary	00142	Anesthesia for lens surgery	118.130500
9	1003000720	Hernandez	Otniel	81003, 99203, 99204, 99205, 99213	Automated urinalysis test, New patient outpati...	68.720287
10	1003000738	Zumwalt	Juliette	20610, 20611, 29823, 29827, 73030, 73562, 9920...	Aspiration and/or injection of large joint or ...	40.744730
11	1003000795	O'neill	Michael	90832	Psychotherapy, 30 minutes	40.528673
12	1003000829	Kochanek	Michelle	97110, 97112, 97140, 97161	Therapeutic exercise to develop strength, endu...	25.300159
13	1003000902	Lohano	Jaivanti	81003, 90662, 90732, 99204, 99213, 99214, 9949...	Automated urinalysis test, Vaccine for influen...	75.950196
14	1003000936	Stellingworth	Mark	36415, 85610, 93000, 93010, 93228, 93270, 9327...	Insertion of needle into vein for collection o...	46.623078

Figure 2 Aggregated Medicare dataset.

3.1.2 LEIE DATASET

LEIE dataset is publicly available at OIG website and is constantly updated.

LEIE dataset shape: 78034 rows, 18 columns

Data dictionary for LEIE:

FIELD VALUE	Description
LASTNAME	Last name
FIRSTNAME	First name
MIDNAME	Mid name
BUSNAME	Business name
GENERAL	Business description
SPECIALTY	Business specialty
UPIN	Unique physician identification number
NPI	National provider identifier
DOB	Date of birth
ADDRESS	Address
CITY	City
STATE	State
ZIP CODE	Zip code

EXCLTYPE	Exclusion type code
EXCLDATE	Exclusion date
REINDATE	Reinstated date
WAIVERDATE	Waiver date
WAIVERSTATE	Waiver state

Table 2 LEIE dictionary.

LEIE dataset comprise 78034 entries. Among these entries, 6713 entries contain NPI. Crossing with Medicare dataset, only 109 yielded (0.14% from LEIE dataset or 0.01% from Medicare dataset) matches with Medicare dataset.

3.1.2.1 EXCLUSION CODES

Exclusion codes are specific numerical or alphanumeric identifiers used to categorize and record the reasons for an individual or entity's exclusion from participation in government programs, particularly in the context of the List of Excluded Individuals and Entities (LEIE) maintained by the U.S. Department of Health and Human Services (HHS) Office of Inspector General (OIG). These codes help to categorize and specify the exact nature of the exclusion. The specific codes may vary depending on the program or agency involved, but they generally serve to categorize the exclusion reasons.

Section	Exclusion Authority
1128(b)(4)	License revocation, suspension, or surrender. Minimum Period: Period imposed by the state licensing authority.
1128(a)(2)	Conviction relating to patient abuse or neglect. Minimum Period: 5 years
1128(b)(6)	Claims for excessive charges, unnecessary services or services which fail to meet professionally recognized

	standards of health care, or failure of an HMO to furnish medically necessary services. Minimum Period: 1 year
1128(a)(3)	Felony conviction relating to health care fraud. Minimum Period: 5 years
1128(a)(4)	Felony conviction relating to controlled substance. Minimum Period: 5 years
1128(a)(1)	Conviction of program-related crimes. Minimum Period: 5 years
1128(b)(7)	Fraud, kickbacks, and other prohibited activities. Minimum Period: None
1128b1	Misdemeanor conviction relating to health care fraud. Baseline Period: 3 years

Table 3 LEIE exclusion codes

As per exclusion codes, our focus shifted from fraud to eligibility due to the broad scope of exclusions nature.

3.1.3 ORDER AND REFERRING DATASET

The Order and Referring (OnR) dataset provide information on all physicians and non-physician practitioners, by their National Provider Identifier (NPI), who are of a type/specialty that is legally eligible to order and refer in the Medicare program and who have current enrollment records in Medicare.

Order and Referring dataset shape: 1.785.839 rows, 7 columns.

Data dictionary for Order and Referring:

Field value	Description
NPI	National Provider Identifier (NPI) of the Order and Referring Provider
LAST NAME	Last Name of the Order and Referring Provider
FIRST NAME	First Name of the Order and Referring Provider
PART B	Indicates that provider can refer to Part B
DME	Indicates that provider can order Durable Medical Equipment
HHA	Indicates that provider can refer to Home Health Agency
PMD	Indicates that provider can order Power Mobility Devices

Table 4 Order and Referring dictionary.

Order and Referring dataset is a compilation of the providers and respective eligibility, in this case we will only consider part B eligibility.

	NPI	LAST_NAME	FIRST_NAME	PARTB	DME	HHA	PMD
0	1558467555	.MCINDOE	THOMAS	Y	Y	Y	Y
1	1417051921	A BELLE	N	Y	Y	Y	Y
2	1972040137	A NOVOTNY	ELIZABETH	Y	Y	Y	Y
3	1760465553	A SATTAR	MUHAMMAD	Y	Y	Y	Y
4	1295400745	A'NEAL	BROGAN	Y	Y	N	N
5	1700562584	AAB	BAILEY	Y	Y	Y	N
6	1467482471	AAB	BARRY	Y	Y	Y	N
7	1245971480	AABEDI	ALEXANDER	Y	Y	Y	Y
8	1164905659	AABEL	SAMANTHA	Y	Y	N	N
9	1255630869	AABERG	MAURA	Y	Y	N	N
10	1801093968	AABERG	MELANIE	Y	Y	Y	Y
11	1346991064	AABERG	MICHAEL	Y	Y	Y	Y
12	1588763981	AABERG	RANDAL	Y	Y	Y	Y
13	1194753186	AABERG	THOMAS	Y	Y	Y	Y
14	1891993317	AABIDA	AFEERA	Y	Y	Y	Y
15	1659765857	AABO	MEGHAN	Y	Y	Y	Y
16	1306810700	AABOE	STELLA	Y	Y	Y	Y
17	1164404232	AABY	AAZY	N	Y	N	Y
18	1487775912	AABY	ROYAL	Y	Y	Y	N
19	1508817040	AACH	DOUGLAS	Y	Y	Y	Y

Figure 3 Order and Referring dataset.

Order and Referring dataset comprise 1785839 entries. Among these entries, 55027 appears as N for PARTB. Crossing with Medicare dataset, only 9330 yielded (16.96% from Order and Referring PARTB as N dataset or 0.83% from Medicare dataset) matches with Medicare dataset.

3.1.4 STATE MEDICAID EXCLUSION LIST

After collecting and making some basic standardized procedures for all 42 states list, it did a quick check on how many lists contained NPI information. Following states presented the information: Arizona, Colorado, Delaware, Florida, Georgia, Hawaii, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maryland, Massachusetts, Michigan, Mississippi, Missouri, Nebraska, Nevada, New Hampshire, New Jersey, New

York, North Carolina, North Dakota, Ohio, Pennsylvania, South Carolina, Tennessee, Texas, Vermont, Washington, West Virginia, Wyoming; 33 in total. Each state had a different spreadsheet, columns and standard to insert the values, hence for the sake of simplicity only NPI was extracted and created a column referring the state which was extracted.

```
states.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 60109 entries, 0 to 60108
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   NPI     16107 non-null  object
 1   State   60109 non-null  object
 2   Text    60109 non-null  bool  
dtypes: bool(1), object(2)
memory usage: 998.0+ KB
```

Figure 4 States exclusion list information

```
states.describe()
```

	NPI	State	Text
count	16107	16107	16107
unique	12560	33	1
top	1316370950	Florida	False
freq	11	4802	16107

Figure 5 States exclusion list information cont'd.

As we can see in figure 4 and 5, yielded 60109 entries, but only 16107 was not null and solely 12560 NPI unique entries, which means that one provider might appeared more than once in a single state or appear in more than one state. When crossing with Medicare dataset, it did not yield any match, hence this information collected will be disregarded.

3.1.5 TARGET VARIABLE

LEIE Dataset and OnR Dataset were used to create the target variable eligibility. Eligibility is a broader concept than fraud, it also considers exclusions due to convictions for program-related fraud as well as offenses such as patient abuse or simply for not being enrolled to Part B but eligible to receive from other federal programs as we could check with OnR dataset.

Target Variable:

- **Positive Class (1):** Providers in the LEIE or in OnR PartB as N.
- **Negative Class (0):** Providers not in the LEIE or in OnR PartB as N.

In this context:

- **True Positive (TP):** A model correctly identifies a provider that should not receive funds from government.
- **False Positive (FP):** A model incorrectly identifies a provider that should not receive funds from government when they are actual eligible.

3.1.5.1 FALSE POSITIVE

- **Incorrectly Accusing a Provider:** False positives mean that the model wrongly flags a provider as excluded from federal healthcare programs. This can have serious consequences for the falsely accused provider, including damage to reputation, legal complications, and potential disruption to their ability to participate in Medicare and other programs.
- **Waste of Resources:** False positives may lead to unnecessary investigations and audits, diverting resources away from genuine cases of fraud. This can be costly both for the healthcare providers and the agencies involved in fraud detection and prevention.
- **Chilling Effect on Healthcare Providers:** The fear of being falsely accused and the potential consequences may have a chilling effect on healthcare providers, affecting their willingness to participate in federal healthcare programs or provide certain services.

It's crucial to strike a balance between sensitivity and specificity in fraud detection models. While it's important to identify fraudulent activity, minimizing false positives is equally important to avoid unnecessary harm to legitimate healthcare providers. Continuous monitoring, regular updates to the model, and incorporating additional context and features can help improve the accuracy of fraud detection systems. Additionally, a robust review process and mechanisms for providers to appeal or correct inaccuracies are essential components of a fair and effective fraud detection system.

3.2 DATA PROCESSING AND WRANGLING

3.2.1 DATA CLEANING

Cleaning a hyperdimensional dataset might take few steps.

```
df.duplicated().sum()
```

```
0
```

```
df.isna().sum()
```

```
NPI                    0
Last_Org_Name          0
First_Name         64187
MI                405734
Crdntls           139077
Gndr               64187
Ent_Cd                 0
St1                    0
St2               839121
City                   0
State_Abrvtn           0
State_FIPS             0
Zip5                   1
RUCA                 703
RUCA_Desc            703
Cntry                  0
Type                   0
Mdcr_Prtcptg_Ind       0
HCPCS_Cd               0
HCPCS_Desc             0
HCPCS_Drug_Ind         0
Place_Of_Srvc          0
Tot_Benes              0
Tot_Srvcs              0
Tot_Bene_Day_Srvcs     0
Avg_Sbmtd_Chrg         0
Avg_Mdcr_Alowd_Amt     0
Avg_Mdcr_Pymt_Amt      0
Avg_Mdcr_Stdzd_Amt     0
fraud                  0
partb_n                0
EXCLTYPE         1123480
EXCLDATE         1123480
eligibility            0
dtype: int64
```

Figure 6 Total null values.

In figure 6 shows that we have none duplicated values and 10 columns with null values. EXCLTYPE and EXCLDATE are derived from LEIE dataset, so we can disregard these values. Last_Org_name, First_Name and MI (middle name) information are different

ways to check provider identity, therefore we can disregard as we will use NPI, which is unique and does not present null or duplicated values. We will apply the same principle to St2 as we have other columns if address information.

For Gender:

```
len(df[(df.Gndr.isna()) & (df.Ent_Cd == 'O')])
```

```
64187
```

All null values for gender are related to Organization.

```
df.Gndr.fillna('O', inplace = True)
```

Figure 7 Cleaning gender column.

Gender and first name have the same number of null values, hence we checked the possibility that the missing gender are due to being organizations and not individuals. Once that confirmed as per figure 7, created another value: “O”.

For Credentials:

```
df.Crdntls.isna().sum()
```

```
139077
```

```
df[(~df.Crdntls.notna()) & (df.Ent_Cd == 'O')].shape[0]
```

```
64187
```

```
df.loc[df['Ent_Cd'] == 'O', ['Crdntls']] = 'O'
```

```
df.Crdntls.isna().sum()
```

```
74890
```

```
df[(~df.Crdntls.notna()) & (df.Ent_Cd == 'I')].shape[0]
```

```
74890
```

All blanks remaining Crdntls blanks are Individuals

```
df.Crdntls.fillna('I', inplace = True)
```

Figure 8 Cleaning credentials column.

The first assumption is that empty values are due to organizations, once confirmed as per figure 8 it was assigned “O”. Next it was confirmed that the remaining are individuals, hence assigned “I”.

For RUCA:

```
df.RUCA.value_counts()
```

```
1.0     948304
4.0      72957
2.0      29920
7.0      26691
1.1      15218
10.0     11171
5.0       5092
4.1       4201
7.1       1848
3.0       1671
8.0       1367
99.0      1002
7.2        631
6.0        612
9.0        522
10.2       508
2.1        467
10.1       396
10.3       139
5.1        131
8.2         22
8.1         16
Name: RUCA, dtype: int64
```

```
df.RUCA.fillna(0, inplace = True)
```

Figure 9 Cleaning RUCA column.

As per figure 9, we assigned “0” for the blanks.

3.2.2 DIMENSIONALITY REDUCTION

3.2.2.1 DROPPING COLUMNS

Feature	Null count	Type	Relevance
NPI	0	Categorical	Keep
Last_Org_Name	0	Categorical	Drop
First_Name	64187	Categorical	Drop
MI	405734	Categorical	Drop

Crdntls	0	Categorical	Drop
Gndr	0	Categorical	Keep
Ent_Cd	0	Categorical	Drop
St1	0	Categorical	Drop
St2	839121	Categorical	Drop
City	0	Categorical	Drop
State_Abrvtn	0	Categorical	Keep
State_FIPS	0	Categorical	Drop
Zip5	1	Categorical	Drop
RUCA	0	Categorical	Keep
RUCA_Desc	703	Categorical	Drop
Cntry	0	Categorical	Drop
Type	0	Categorical	Keep
Mdcr_Prtcptg_Ind	0	Categorical	Keep
HCPCS_Cd	0	Categorical	Drop
HCPCS_Desc	0	Categorical	Drop
HCPCS_Drug_Ind	0	Categorical	Keep
Place_Of_Srvc	0	Categorical	Keep
Tot_Benes	0	Numerical	Keep
Tot_Srvcs	0	Numerical	Keep
Tot_Bene_Day_Srvcs	0	Numerical	Keep
Avg_Sbmtd_Chrg	0	Numerical	Keep
Avg_Mdcr_Alowd_Amt	0	Numerical	Keep
Avg_Mdcr_Pymt_Amt	0	Numerical	Keep
Avg_Mdcr_Stdzd_Amt	0	Numerical	Keep
fraud	0	Categorical	Drop
partb_n	0	Categorical	Drop
EXCLTYPE	1123480	Categorical	Drop
EXCLDATE	1123480	Date	Drop
eligibility	0	Categorical	Keep

Table 5 Feature relevance.

In table 5 features are separated per categories:

Blue – ID

ID won't contribute with the model, but we kept NPI as reference for now.

Red – ID classification

Credentials has too many unique values, therefore disregard due to high cardinality. Ent_Cd provide almost the same information as Gender, hence we keep only Gender.

Yellow – Geolocation

St1, St2, City, State_FIPS, Zip5 and Cntry are implicit in State_ Abrvtn. RUCA_Desc is implicit in RUCA, hence we keep State_ Abrvtn and RUCA, that might bring a different perception for the model.

Grey – Medical service related

This is one of the most relevant categories, which we will drop only HCPCS_Cd and HCPCS_Desc that apparently is a more detailed version of Type column.

White – FFS

Another relevant categories since we are trying to predict eligibility based on FFS data.

Green – Target feature

Feature “fraud” is named after matching LEIE dataset, partb_n is named after matching OnR dataset and eligibility is the merged columns, hence we drop the first two. EXCLTYPE and EXCLDATE are does not look like relevant data since almost all the values are null.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1123589 entries, 0 to 1123588
Data columns (total 19 columns):
 #   Column              Non-Null Count    Dtype
---  ------              --------------    -----
 0   NPI                 1123589 non-null  int64
 1   Crdntls             1123589 non-null  object
 2   Gndr                1123589 non-null  object
 3   Ent_Cd              1123589 non-null  object
 4   State_Abrvtn        1123589 non-null  object
 5   RUCA                1123589 non-null  float64
 6   Type                1123589 non-null  object
 7   Mdcr_Prtcptg_Ind    1123589 non-null  object
 8   HCPCS_Cd            1123589 non-null  object
 9   HCPCS_Drug_Ind      1123589 non-null  object
 10  Place_Of_Srvc       1123589 non-null  object
 11  Tot_Benes           1123589 non-null  int64
 12  Tot_Srvcs           1123589 non-null  float64
 13  Tot_Bene_Day_Srvcs  1123589 non-null  int64
 14  Avg_Sbmtd_Chrg      1123589 non-null  float64
 15  Avg_Mdcr_Alowd_Amt  1123589 non-null  float64
 16  Avg_Mdcr_Pymt_Amt   1123589 non-null  float64
 17  Avg_Mdcr_Stdzd_Amt  1123589 non-null  float64
 18  eligibility         1123589 non-null  int64
dtypes: float64(6), int64(4), object(9)
memory usage: 162.9+ MB
```

Figure 10 Dataset after dimensionality reduction.

3.2.2.2 CHI SQUARE

For further feature selection, it was implemented Chi Squared. We already selected 7 features, but we want to further reduce the dimensionality selecting the 5 most relevant.

```
df.RUCA = df.RUCA.astype(object)
```

```
cat_col = ['Gndr', 'State_Abrvtn', 'RUCA','Type',
    'Mdcr_Prtcptg_Ind', 'HCPCS_Drug_Ind', 'Place_Of_Srvc']
```

```
X = df[cat_col]
y = df.eligibility
```

```
X.describe()
```

	Gndr	Ent_Cd	State_Abrvtn	RUCA	Type	Mdcr_Prtcptg_Ind	HCPCS_Drug_Ind	Place_Of_Srvc
count	1123589	1123589	1123589	1123589.0	1123589	1123589	1123589	1123589
unique	3	2	61	23.0	103	4	4	4
top	M	I	CA	1.0	Nurse Practitioner	Y	N	O
freq	559187	1059402	87424	948304.0	149911	1122070	901567	561001

```
label_encoder = LabelEncoder()
for col in cat_col:
    X[col] = label_encoder.fit_transform(X[col])
```

Figure 11 Converting categorical features.

Figure 11 shows the columns selected for categorical feature selection and transforming the data.

```
chi2_stats, p_values = chi2(X, y)

results_df = pd.DataFrame({'Feature': cat_col, 'Chi-squared': chi2_stats, 'p-value': p_values})

print(results_df)
```

```
             Feature  Chi-squared       p-value
0               Gndr    74.781743  5.257365e-18
1       State_Abrvtn   104.697243  1.422989e-24
2               RUCA    23.407752  1.310494e-06
3               Type  1806.206803  0.000000e+00
4  Mdcr_Prtcptg_Ind     0.008807  9.252312e-01
5    HCPCS_Drug_Ind    16.911569  3.916217e-05
6     Place_Of_Srvc   144.343798  2.988307e-33
```

```
X_new = SelectKBest(score_func=chi2, k=5).fit_transform(X, y)
```

```
selector = SelectKBest(score_func=chi2, k=5)
X_new = selector.fit_transform(X, y)

selected_feature_indices = selector.get_support(indices=True)

selected_features = X.columns[selected_feature_indices]

print("Selected Features:")
print(selected_features)
```

```
Selected Features:
Index(['Gndr', 'State_Abrvtn', 'RUCA', 'Type', 'Place_Of_Srvc'], dtype='object')
```

Figure 12 Chi Squared results.

The second step is running Chi Squared and selecting the best values. As we can see in the first result shown in Figure 12, Type, Ent_cd, Place_Of_Srvc, State_Abrvtn and Gndr had the highest Chi-squared values with p-values < 0.05. We used SelectKBest to confirm our interpretation.

3.2.2.3 MULTICOLLINEARITY

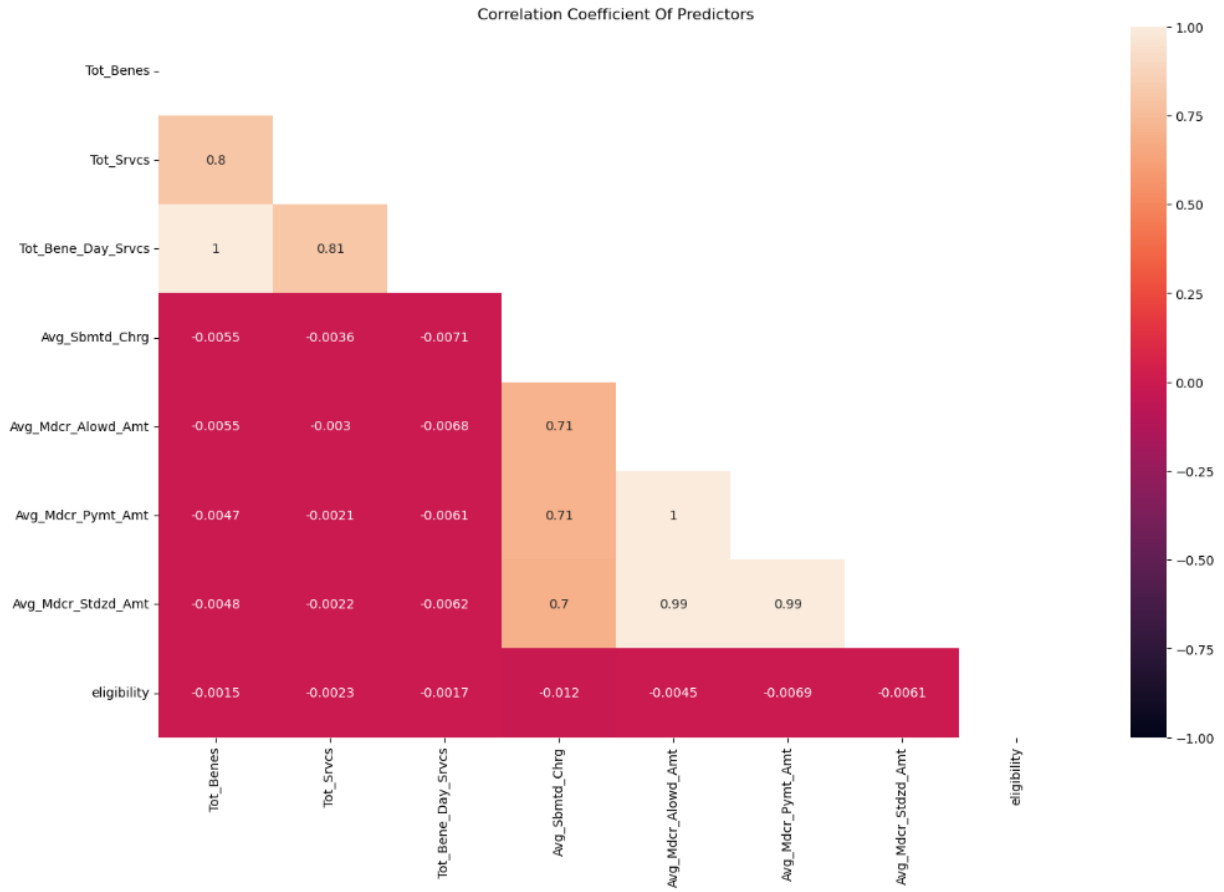

Figure 13 Numerical features correlation.

There is low to no correlation between the independent features and target feature, however it shows collinearity between some independent features. To deal with collinearity, was verified Variance Inflation Factor (VIF) and four iterations was performed until we reach values lower than 5 as per figures 14-17.

```
X = data_num.drop(columns=['eligibility'])
vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif_data
```

	Feature	VIF
0	Tot_Benes	113.146093
1	Tot_Srvcs	2.909676
2	Tot_Bene_Day_Srvcs	118.421456
3	Avg_Sbmtd_Chrg	2.009628
4	Avg_Mdcr_Alowd_Amt	269.597469
5	Avg_Mdcr_Pymt_Amt	340.426661
6	Avg_Mdcr_Stdzd_Amt	86.906657

Figure 14 VIF first iteration.

```
X = data_num.drop(columns=['eligibility','Tot_Benes'])
vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif_data
```

	Feature	VIF
0	Tot_Srvcs	2.870310
1	Tot_Bene_Day_Srvcs	2.870301
2	Avg_Sbmtd_Chrg	2.009421
3	Avg_Mdcr_Alowd_Amt	269.591667
4	Avg_Mdcr_Pymt_Amt	340.422573
5	Avg_Mdcr_Stdzd_Amt	86.906383

Figure 15 VIF second iteration.

```
X = data_num.drop(columns=['eligibility','Tot_Benes','Avg_Mdcr_Alowd_Amt'])
vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif_data
```

	Feature	VIF
0	Tot_Srvcs	2.870179
1	Tot_Bene_Day_Srvcs	2.870291
2	Avg_Sbmtd_Chrg	2.009356
3	Avg_Mdcr_Pymt_Amt	88.346743
4	Avg_Mdcr_Stdzd_Amt	86.635528

Figure 16 VIF third iteration.

```
X = data_num.drop(columns=['eligibility','Tot_Benes', 'Avg_Mdcr_Alowd_Amt', 'Avg_Mdcr_Pymt_Amt'])
vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif_data
```

	Feature	VIF
0	Tot_Srvcs	2.870176
1	Tot_Bene_Day_Srvcs	2.870291
2	Avg_Sbmtd_Chrg	1.967601
3	Avg_Mdcr_Stdzd_Amt	1.967593

Figure 17 VIF fourth iteration.

If we check the dictionary, we can see that Tot_Benes_Day_Srvcs is related to Tot_Benes and Avg_Mdcr,_Stdzd_Amt is related to Avg_Mdcr_Alowd_Amt and Avg_Mdcr_Pymt_Amt.

3.2.3 FEATURES TRANSFORMATION

3.2.3.1 STANDARDIZATION

Next, we need to transform all the data to suitable forms in which our machine learning models will be able to access and interpretate. Standardization process was chosen for numerical features transformation as per following figure 18.

```
numerical_columns = ['Tot_Bene_Day_Srvcs', 'Tot_Srvcs','Avg_Sbmtd_Chrg', 'Avg_Mdcr_Stdzd_Amt']
```

```
data_num[numerical_columns] = StandardScaler().fit_transform(data_num[numerical_columns])
```

Figure 18 Numerical features transformation.

3.2.3.2 LABEL ENCODING

For categorical features, we will try two approaches: label encoding and one-hot encoding. Label encoding is a process of assigning numerical labels to categorical data values, which might give an implicit ordinality that is not ideally for our case. Figure 19 demonstrate label encoding transformation.

```
df_label = df.copy()
```

```
label_encoder = LabelEncoder()
for col in categorical_columns:
    df_label[col] = label_encoder.fit_transform(df[col])
```

```
df_label[numerical_columns] = StandardScaler().fit_transform(df_label[numerical_columns])
```

```
df_label = df_label[categorical_columns+numerical_columns+target]
```

Figure 19 Label encoding transformation.

3.2.3.3 ONE-HOT ENCODING

The other categorical feature transformation selected for screening is one-hot encoding. One-hot encoding is a technique used to represent categorical variables as binary vectors. It involves converting a categorical variable with k distinct categories into k separate binary features, each representing one category. Each binary feature takes the value of 1 if the category is present and 0 if it is not.

For example, one-hot encoding gender would first consist of generating extra features equaling the number of options, in this case three (male, female and organization). If the physician is male, the new male feature would be assigned a 1, the female and organization feature would be 0; while for female, the male and organization would be

assigned a 0 and the female assigned a 1. Figure 20 demonstrate one-hot encoding transformation.

```
df_reduced = df[categorical_columns+numerical_columns+target]
```

```
df_reduced.shape
```

```
(1123589, 10)
```

```
encoder = OneHotEncoder(sparse = False)
```

```
encoder.fit(df_reduced[categorical_columns])
df_reduced_encoded = encoder.transform(df_reduced[categorical_columns])
df_reduced_encoded_df = pd.DataFrame(df_reduced_encoded, columns=encoder.get_feature_names(categorical_columns))
df_reduced_encoded_df.index = df_reduced.index
```

```
df_reduced_encoded_df.shape
```

```
(1123589, 194)
```

```
df_combined = df_reduced_encoded_df.join(data_num)
```

```
df_combined = df_combined.drop(columns=['Tot_Bene_Day_Srvcs', 'Avg_Mdcr_Alowd_Amt', 'Avg_Mdcr_Pymt_Amt'])
```

Figure 20 One-hot encoding transformation.

3.3 IMBALANCE DATA

Imbalanced dataset refers to a situation where the distribution of the target variable (in this case, whether a provider is eligible or not) is significantly skewed. In other words, one class has much fewer instances compared to the other class, consequently, pose challenges for machine learning models because models may become biased toward the majority class. In fraud detection, where the occurrence of fraud is typically rare, a model trained on imbalanced data might struggle to accurately identify instances of fraud.

Traditional machine learning algorithms can be biased towards the majority class, leading to high accuracy but poor performance in identifying the minority class (fraud). The model might be too conservative and tend to predict non-fraudulent cases more frequently. Strategies to address imbalanced data include resampling techniques (oversampling the minority class or under sampling the majority class), using different evaluation metrics (precision, recall, F1 score), or employing specialized algorithms designed for imbalanced datasets.

3.3.1 SAMPLING

There are two basic sampling methods: oversampling and under sampling. Oversampling is a method for balancing classes by adding instances to the classification for class, whereas under sampling removes samples from the majority class.

Oversampling can increase processing time by increasing the overall size. More critically, oversampling can overfit the data by making identical copies of the minority class. On the contrary, with under sampling, we retain all the original fraud-labeled instances and randomly sample without replacement from the remaining majority class instances.

3.3.1.1 OVER SAMPLING

We used the SMOTE (Synthetic Minority Over-sampling Technique) algorithm to oversample the minority class in your training data (X_train, y_train).

```
y_train_val.value_counts()
```

```
0    44553
1      390
Name: eligibility, dtype: int64
```

```
smote = SMOTE(random_state=42)
X_train_smote_val, y_train_smote_val = smote.fit_resample(X_train_val, y_train_val)
```

```
y_train_smote_val.value_counts()
```

```
0    44553
1    44553
Name: eligibility, dtype: int64
```

Figure 21 SMOTE sampling.

Figure 21 shows the outcome when using SMOTE: we are creating entries for the imbalanced data to match with the one in majority.

3.3.1.2 UNDER SAMPLING

For under sampling, it was retained the same amount of the eligibility label rows for the majority label, thus ensuring that we have a balanced dataset but with the risk of information loss.

```
df_under = pd.concat([X_train_val, y_train_val], axis=1)
```

```
df_under.eligibility.value_counts()
```

```
0    44553
1      390
Name: eligibility, dtype: int64
```

```
fraud_df = df_under.loc[df['eligibility'] == 1]
non_fraud_df = df_under.loc[df['eligibility'] == 0][:390]
```

```
normal_distributed_df = pd.concat([fraud_df, non_fraud_df])
# Shuffle dataframe rows
df_new = normal_distributed_df.sample(frac=1, random_state=42)
# split out validation dataset for the end
y_train_under= df_new["eligibility"]
X_train_under = df_new.loc[:, df.columns != 'eligibility']
```

```
df_new.eligibility.value_counts()
```

```
0    390
1    390
Name: eligibility, dtype: int64
```

Figure 22 Random under sampling.

Figure 22 shows the outcome when using under sampling: we are excluding entries from the majority label to match with eligibility label.

3.3.2 MODELS METRIC

In fraud detection, the interpretation of classification metrics is crucial for understanding how well a model performs in identifying fraudulent transactions. As mentioned before, it is also important to have low False Positive outcome and since we are running machine

learning model screening with a highly imbalanced dataset, it is key to properly define our metrics priority.

Basic classification metrics:

- Precision: is the ratio of true positive predictions to the total predicted positives. It measures the accuracy of positive predictions.
- Recall (Sensitivity or True Positive Rate): is the ratio of true positive predictions to the total actual positives. It measures the ability of the classifier to capture all the positive instances.
- F1-Score: is the harmonic mean of precision and recall. It provides a balance between precision and recall.
- Accuracy: represents the number of correctly classified data instances over the total number of data instances.

A high recall means that the model can correctly identify “not eligible” instances with few false negatives, but not choosing precision might yield lots of false positive cases which is not ideal. Usually, accuracy is the metric to go with, but for our project the aim is to identify the most cases labeled as “1”, i.e., not eligible, therefore we will use a multimetric system with recall as the main metric followed by accuracy. The goal is to achieve both metrics above 0.60.

3.4 MODEL SCREENING

Machine learning models screening will be performed using ten distinguished algorithms: Logistic Regression, Linear Discriminant Analysis, KNN, Decision Tree, Support Vector Machine, Naïve Bayes, Gradient Boosting, Adaboost, Random Forest and Extra Trees.

```
models = []
models.append(('LR', LogisticRegression(solver = 'saga', max_iter = 5000)))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('SVM', SVC()))
models.append(('NB', GaussianNB()))
models.append(('GBM', GradientBoostingClassifier()))
models.append(('AB', AdaBoostClassifier()))
models.append(('RF', RandomForestClassifier()))
models.append(('ET', ExtraTreesClassifier()))
```

Figure 23 Screening models.

We have prepared two datasets: label encoded and one-hot encoded. For each dataset, three screenings will be performed: No sampling techniques, under sampling and over sampling. Screening will be done on 5% validation subset from the original dataset.

5% Validation

```
df_val, X_test, y_train, y_test = train_test_split(df, df.eligibility, test_size=0.95, stratify = df.eligibility, random_state=42
```

```
X_train_val, X_test_val, y_train_val, y_test_val = train_test_split(df_val.drop(columns = ['eligibility'], axis = 1), df_val.elig
```

Figure 24 Screening validation subset.

The best models selected from screening using under sampling and over sampling will be trained on a 10% subset from the original dataset to verify the best models performance on a train and test subset.

10%Train and Test

Spliting 10% for further spliting into train and test 80%

```
df_val_10, X_test, y_train, y_test = train_test_split(df, df.eligibility, test_size=0.90, stratify = df.eligibility, random_state
```

```
X_train, X_test, y_train, y_test = train_test_split(df_val_10.drop(columns = ['eligibility'], axis = 1), df_val_10.eligibility, t
```

Figure 25 Screening 10% train and test subset.

Last step is to train the best models with 50% subset to verify the performance between encoding and model selected, if there is more than one plausible model.

50% Train and Test

```
df_val_50, X_test, y_train, y_test = train_test_split(df, df.eligibility, test_size=0.5, stratify = df.eligibility, random_state=
```

```
X_train, X_test, y_train, y_test = train_test_split(df_val_50.drop(columns = ['eligibility'], axis = 1), df_val_50.eligibility, t
```

Figure 26 Screening 50% train and test subset.

3.5 HYPERPARAMETER TUNING

To achieve a desirable model with sufficient score, we will try to improve the best model selected from the model screening performing a grid search for hyperparameter tuning. Given we are evaluating the model based on multimetric scoring, we will perform two grid searches, the first one using refit set as recall and the second grid set as accuracy. Grid search will be performed on 20% subset the hyperparameters shown in figure 27 and thereafter we will train and test using the whole dataset for:

- Best parameters from recall grid search;
- Best parameters from accuracy grid search; and
- Base model.

```
scoring = ('recall', 'accuracy')

param_grid = {"min_samples_split": range(2, 300, 50),
              "max_features": range(1, 50,5),
              "max_depth": range(1, 20,5),
              'subsample':[0.7,0.75,0.8,0.85,0.9,1]}
```

Figure 27 Hyperparameters and scoring selected for tuning.

4 RESULTS

4.1 LABEL ENCODING

4.1.1 NO SAMPLING TECHNIQUES

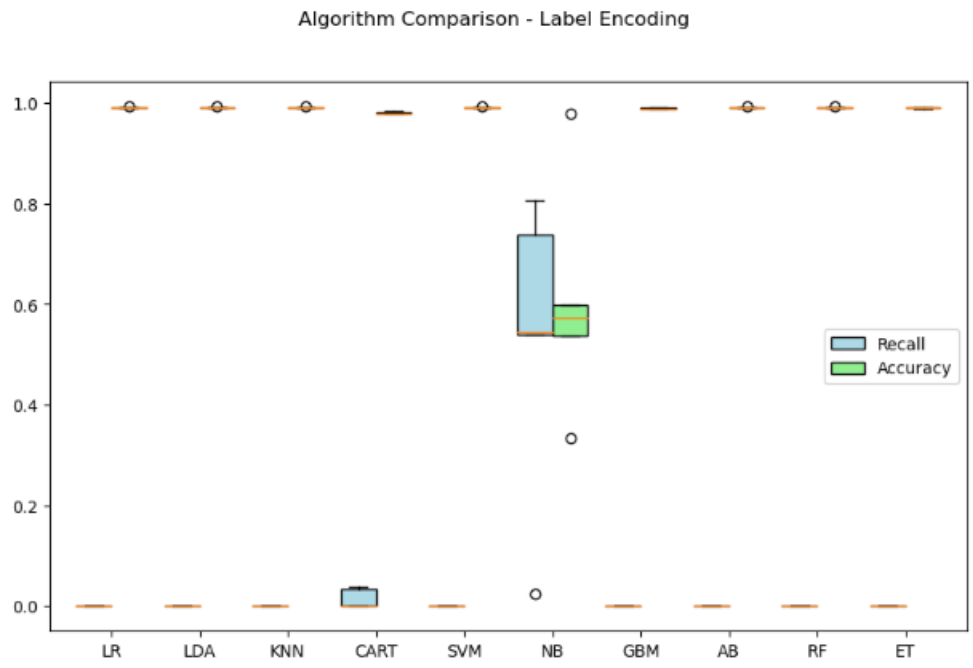

Figure 28 Screening boxplot for label encoding without sampling techniques.

Figure 28 is the result of label encoding without sampling techniques. Yielded high accuracy and low recall for most of the models, which is not relevant to be considered.

4.1.2 UNDER SAMPLING

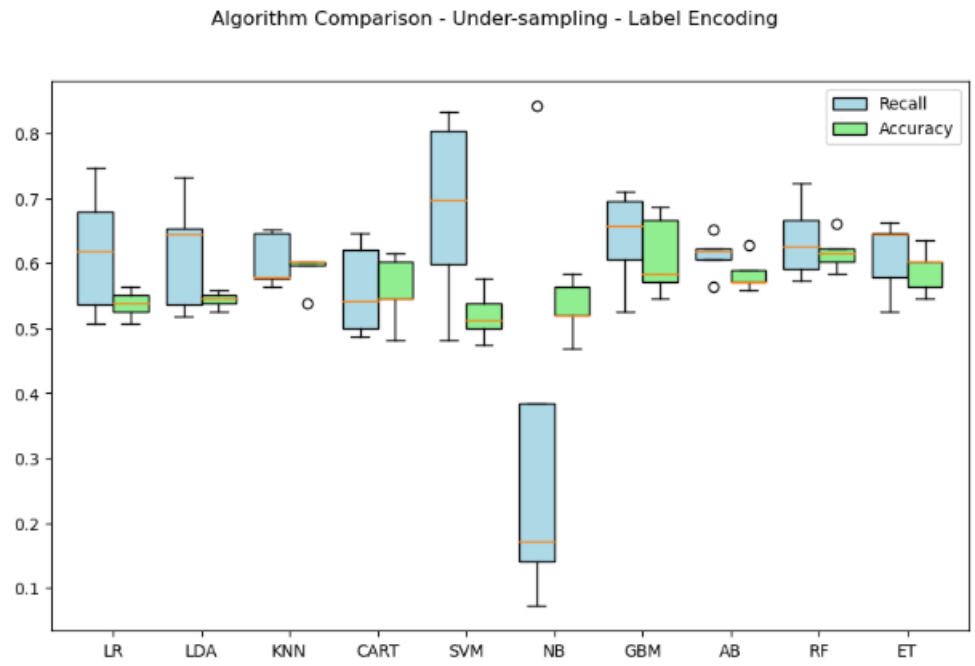

Figure 29 Screening boxplot for label encoding with under sampling.

Figure 29 is the result of label encoding with under sampling. For this screening we found that KNN, SVM, gradient boost and random forest could be suitable, hence those models were chosen for further assessment. Although SVM has accuracy below 0.6, it will be tested due to high recall.

4.1.3 OVER SAMPLING

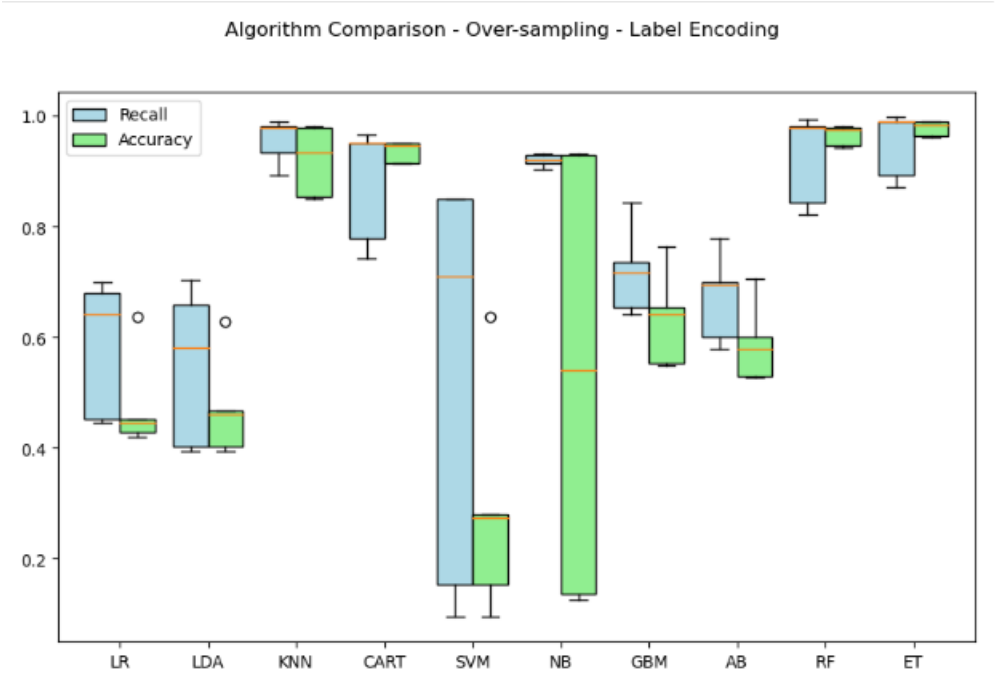

Figure 30 Screening boxplot for label encoding with over sampling.

Figure 30 is the result of label encoding with over sampling. For this screening we found that KNN, CART, SVM, gradient boost, adaboost, random forest and extra trees could be suitable, hence those models were chosen for further assessment. Although SVM has accuracy below 0.6, it will be tested due to high recall.

4.2 ONE-HOT ENCODING

4.2.1 NO SAMPLING TECHNIQUES

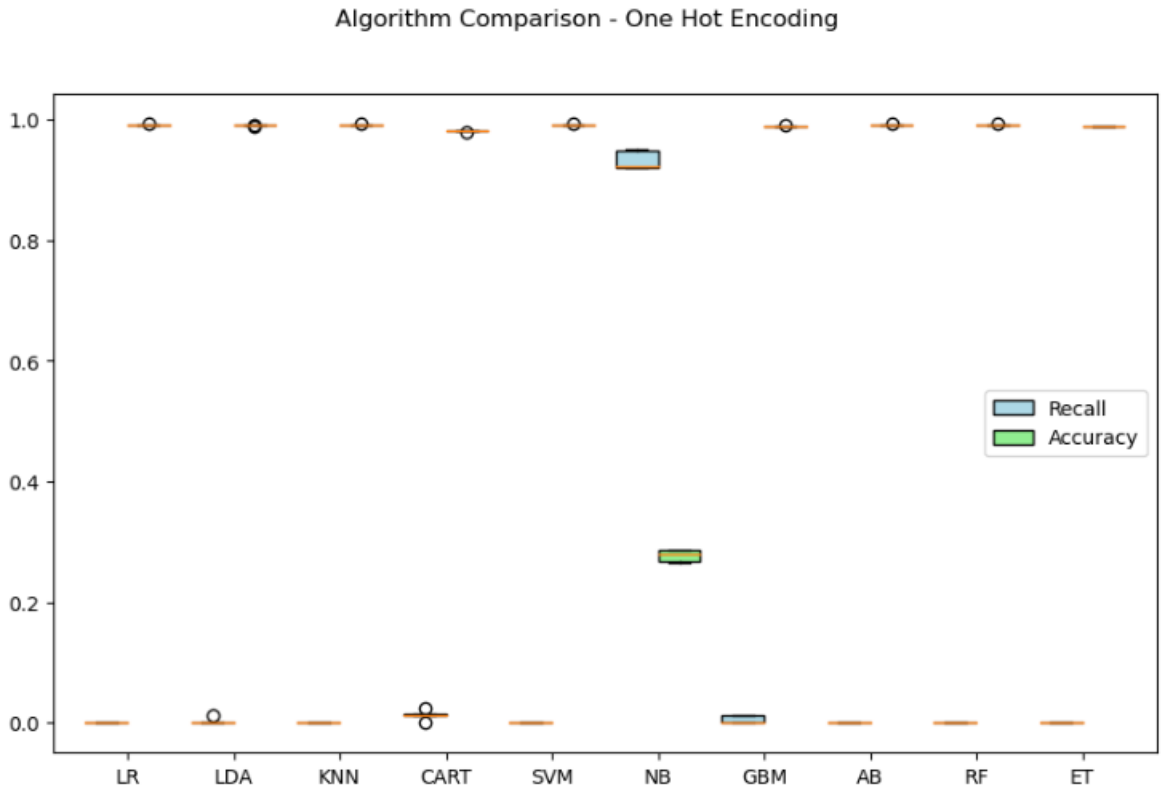

Figure 31 Screening boxplot for one-hot encoding without sampling techniques.

Figure 31 is the result of one-hot encoding without sampling techniques. Yielded high accuracy and low recall for most of the models, which is not relevant to be considered.

4.2.2 UNDER SAMPLING

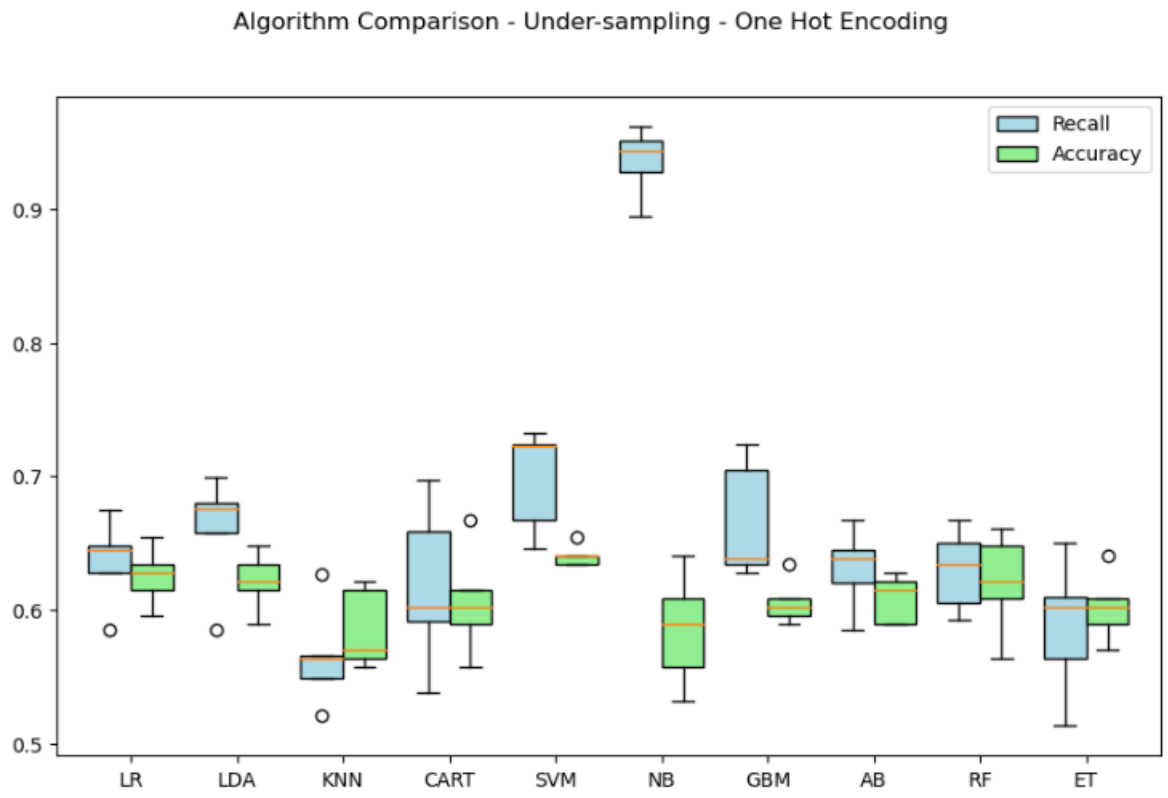

Figure 32 Screening boxplot for one-hot encoding with under sampling.

Figure 32 is the result of one-hot encoding with under sampling. For this screening we found that logarithmic regression, LDA, SVM, gradient boost, adaboost and random forest could be suitable, hence those models were chosen for further assessment.

4.2.3 OVER SAMPLING

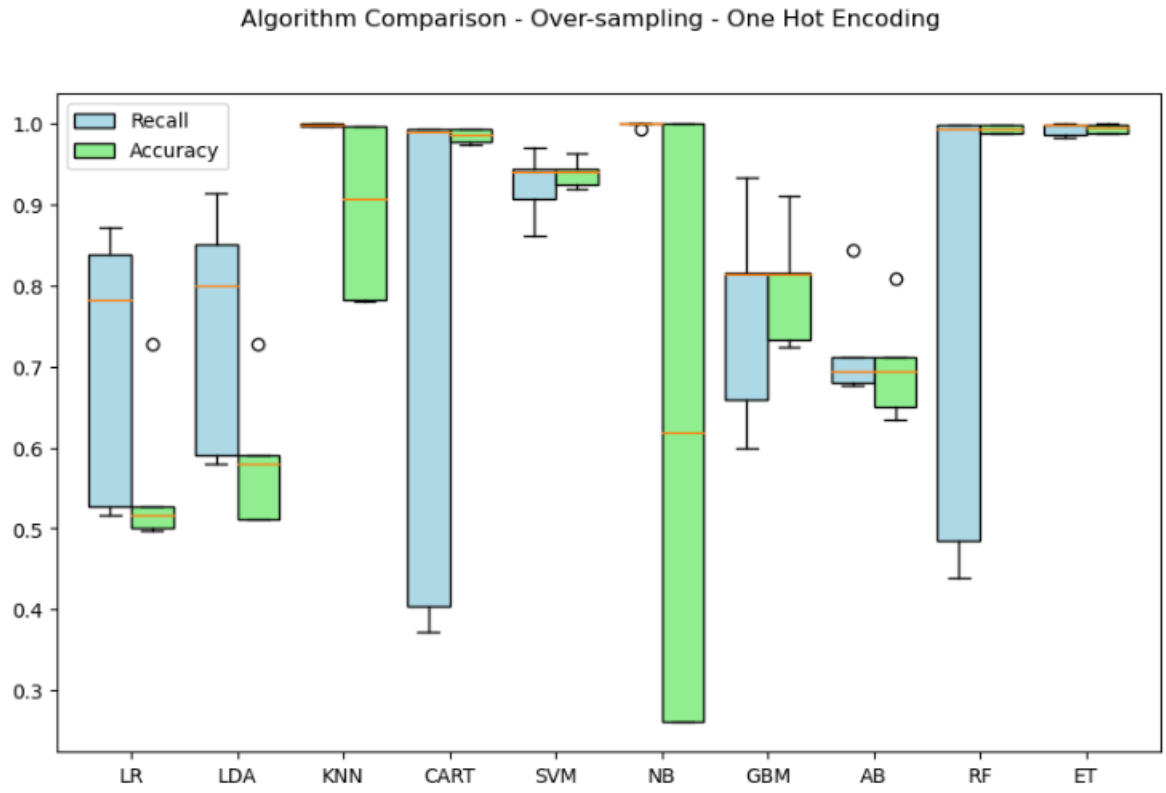

Figure 33 Screening boxplot for one-hot encoding with over sampling.

Figure 33 is the result of one-hot encoding with under sampling. For this screening we found that KNN, SVM, gradient boost, adaboost, random forest and extra trees could be suitable, hence those models were chosen for further assessment. Although logarithmic regression and LDA has accuracy below 0.6, it will be tested due to high recall.

4.3 BEST MODELS

Encoding	Sampling	Model	Recall	Accuracy
Label	Under	KNN	0.59	0.53
Label	Under	SVM	0.72	0.36
Label	Under	Gradient Boost	0.70	0.58
Label	Under	Random Forest	0.66	0.60
Label	Over	KNN	0.13	0.89
Label	Over	CART	0.14	0.91

Label	Over	SVM	0.47	0.61
Label	Over	Gradient Boost	0.58	0.65
Label	Over	Random Forest	0.05	0.96
Label	Over	Adaboost	0.56	0.62
Label	Over	Extra Trees	0.05	0.97
One-Hot	Under	Log Regression	0.72	0.58
One-Hot	Under	LDA	0.75	0.57
One-Hot	Under	SVM	0.71	0.59
One-Hot	Under	Gradient Boost	0.77	0.57
One-Hot	Under	Adaboost	0.73	0.59
One-Hot	Under	Random Forest	0.72	0.60
One-Hot	Over	KNN	0.21	0.87
One-Hot	Over	SVM	0.10	0.95
One-Hot	Over	Gradient Boost	0.33	0.87
One-Hot	Over	Adaboost	0.53	0.77
One-Hot	Over	Random Forest	0.01	0.99
One-Hot	Over	Extra Trees	0.02	0.98

Table 6 10% Train and Test subset models performance.

From table 6, at least one model with recall higher than 0.7 and accuracy higher than 0.55 from each encoding methods were further tested.

Encoding	Sampling	Model	Recall	Accuracy
Label	Under	Gradient Boost	0.74	0.59
One-Hot	Under	Gradient Boost	0.76	0.58
One-Hot	Under	LDA	0.74	0.58

Table 7 50% Train and Test subset models performance.

From table 7, gradient boost model using one-hot encoding dataset with under sampling had the best outcome, thus this setup will be used for hyperparameter tuning.

4.4 HYPERPARAMETER TUNING

Scoring and hyperparameters grid

Model	Recall	Accuracy
Recall refit	0.75	0.56
Accuracy refit	0.75	0.61
Base model	0.76	0.59

Table 8 Hyperparameter tuning model results.

From table 8, all the models had similar recall and accuracy, but since our goal is to achieve at least 0.60 for both metrics, accuracy refit grid search parameters were selected as the best model.

```
best_model_ac = GradientBoostingClassifier(**best_params_ac, random_state=42)
best_model_ac.fit(X_train_under, y_train_under)

predictions_ac = best_model_ac.predict(X_test)
print(classification_report(y_test, predictions_ac))
```

```
              precision    recall  f1-score   support

           0       1.00      0.61      0.75    222852
           1       0.02      0.75      0.03      1866

    accuracy                           0.61    224718
   macro avg       0.51      0.68      0.39    224718
weighted avg       0.99      0.61      0.75    224718
```

Figure 34 Accuracy refit classification report.

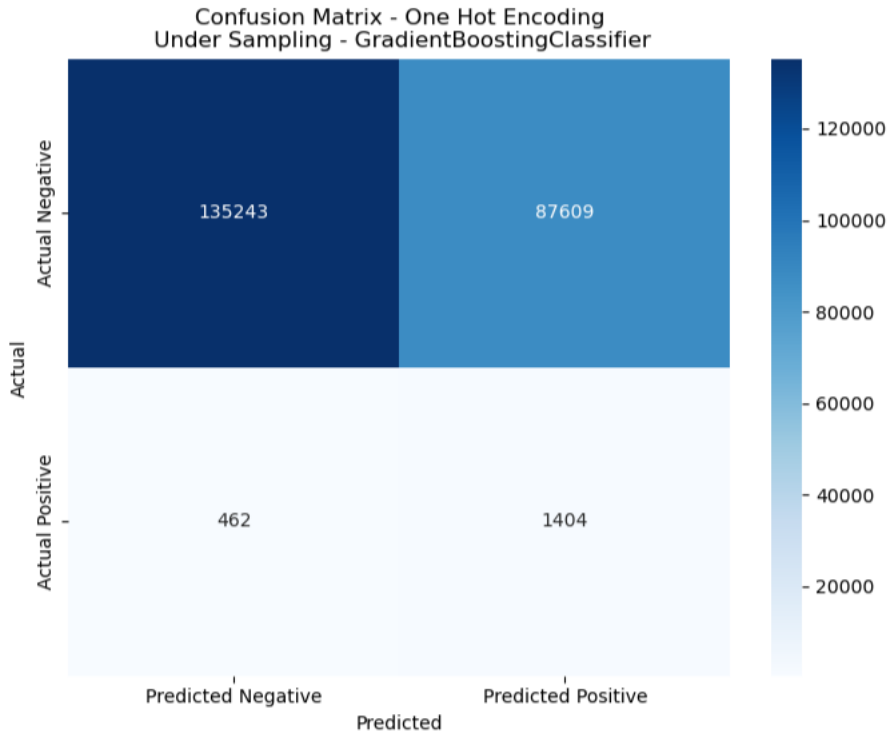

Figure 35 Accuracy refit confusion matrix.

Figure 34 and 35 demonstrate that the best model achieved the minimum requirements, although it has high false positive rate.

5 DISCUSSION

Using different encoding methods and coupled with sampling techniques the model performance varied significantly. In general, under sampling enhanced recall, but had lower accuracy, in the other hand over sampling had higher accuracy but recall below desired thresholds for all models screened.

An assessment with a 10% subset showcased that one-hot encoding paired with under sampling delivered the best set of results, with all models screened recall higher than 0.70 and accuracy ranging from 0.57-0.60. Gradient Boost using One-Hot Encoding with under sampling emerged as the top performer, presenting a balanced trade-off between recall and accuracy.

Hyperparameter tuning aimed at achieving at least 0.60 for both recall and accuracy. Although models exhibited similar performance post-tuning, the accuracy refit grid search parameters were chosen as they met the desired thresholds. The best-performing model met the minimum requirements for both recall and accuracy, yet displayed a higher false positive rate, which might affect its feasibility in practical applications.

6 CONCLUSION

The study evaluated various encoding methods and sampling techniques. Under sampling consistently bolstered recall but at the expense of lowered accuracy, while over sampling displayed higher accuracy but struggled to meet desired recall thresholds across all models assessed.

Upon carefully examination a 10% subset, one-hot encoding combined with under sampling proved all screened models achieved recall levels surpassing 0.70, with accuracy ranging from 0.57 to 0.60. Notably, Gradient boost employing one-hot encoding with under sampling showcased a well-balanced trade-off between recall and accuracy, standing out as the top-performing model.

The subsequent hyperparameter tuning attempt aimed at achieving a minimum of 0.60 for both recall and accuracy. Although the models exhibited comparable performance post-tuning, the accuracy refit grid search parameters were preferred due to meeting the desired thresholds. The best-performing model met the minimum requirements for both recall and accuracy, yet unveiled a higher false positive rate, potentially impacting its practical applicability in real-world scenarios.

In summary, the evaluation highlights the significance of encoding methods and sampling strategies in enhancing model performance for imbalanced datasets. The identified approach of one-hot encoding with under sampling using gradient boost showcase a promising model for achieving a balanced performance trade-off.

7 REFERENCES

1. How Medicare and Medicaid fraud became a $100B problem for the U.S. (cnbc.com)
2. Healthcare Fraud Data Mining Methods: A Look Back and Look Ahead - PMC (nih.gov)
3. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9013219/
4. Medicare PART B dataset (cms.gov)
5. Order and referring dataset (cms.gov)
6. LEIE Downloadable Databases | Office of Inspector General | U.S. Department of Health and Human Services (hhs.gov)
7. All Exclusion Databases & More - Streamline Verify
8. Big Data fraud detection using multiple Medicare data sources | Journal of Big Data | Full Text (springeropen.com)
9. Identifying Physician Fraud in Healthcare with Open Data | SpringerLink