Name: SWATI SHARMA

Email address: swatisharma890929@gmail.com

Contact number: +91-9643-077-602

Anydesk address: **876 627 812**

Years of Work Experience: **2.4**

Date: **26th Nov 2021**

Self Case Study -1: Healthcare Provider Fraud Detection Analysis

Kaggle Link: https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis

---

"After you have completed the document, please submit it in the classroom in the pdf format."

Please check this video before you get started:
https://www.youtube.com/watch?time_continue=1&v=LBGU1_JO3kg

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** ***

1. Introduction

Healthcare Fraud is stated to be a white-collar crime, which means it is nothing but *a nonviolent crime which is often seen in financial situations for monetary profit*. And often many of such crimes are difficult to prosecute because the perpetrators use sophisticated means to conceal their activities through a series of complex transactions.

Healthcare Fraud does involve practitioners kicking schemes such as:

- Providing patients fully or partially covered medicines, which not only involves increasing the cost on the patient's head but the health-insurance companies are also at a loss.
- Duplicate claims, altering dates, description of services, billing of non-covered service as a covered service, etc are some of the more fraudulent ways of fooling health-insurance companies.

1.1. Business Problem

The Business problem has been taken from one of the Kaggle posts, which can be found at this [weblink].

- As per the problem, we need to *"predict the potential fraudulent providers"*, based on the claims filed by them.
- Also, to *"discover important variables helpful in detecting the behaviour of potentially fraudulent providers"*.
- If certain patterns are discovered in providers' claims to understand the future trend of frauds, then it would add a lot of value to the health-insurance companies.

1.1.1. ML formulation of the Business Problem

It is a pure binary classification task.

1.1.2. Business Constraints

- Cost of mis-classification can be very high, as it leads to a fraudulent provider being provided with the reimbursement of the medical bills s/he could have added to the insurance company.
- Probability for a provider to be non/fraudulent, if s/he follows the pattern figured out, then that could be flagged.

1.1.3. Data Columns Analysis

There are 8 files in total. 4 each of train and test, viz.

1. Test-1542969243754.csv
2. Test_Beneficiarydata-1542969243754.csv
3. Test_Inpatientdata-1542969243754.csv
4. Test_Outpatientdata-1542969243754.csv
5. Train-1542865627584.csv
6. Train_Beneficiarydata-1542865627584.csv
7. Train_Inpatientdata-1542865627584.csv
8. Train_Outpatientdata-1542865627584.csv

1.1.3.1. Beneficiary(Rows=, Columns=25)

Name	Data Type	Description
1. BeneID	obj	Healthcare Beneficiary Identity
2. DOB	obj	Date of Birth
3. DOD	obj	Date of Discharge
4. Gender	int	Gender(Male/Female)
5. Race	int	Ethnicity of the patient
6. RenalDiseaseIndicator	obj	If the beneficiary has a renal/kidney disease
7. State	int	State to which the beneficiary

		belongs
8. Country	int	Country to which the beneficiary belongs
9. NoOfMonths_PartACov	int	Number of months for which Part A of Medicare is covered
10. NoOfMonths_PartBCov	int	Number of months for which Part B of Medicare is covered
11. ChronicCond_Alzheimer	int	Does the beneficiary have alzhemiers
12. ChronicCond_Heartfailure	int	Does the beneficiary had a heart failure
Name	**Data Type**	**Description**
13. ChronicCond_KidneyDisease	int	Does the beneficiary have kidney disease
14. ChronicCond_Cancer	int	Does the beneficiary have cancer
15. ChronicCond_ObstrPulmonary	int	Does the beneficiary have Obstructive Pulmonary disease
16. ChronicCond_Depression	int	Does the beneficiary have Depression
17. ChronicCond_Diabetes	int	Does the beneficiary have Diabetes
18. ChronicCond_IschemicHeart	int	Does the beneficiary have Ischemic Heart Disease
19. ChronicCond_Osteoporosis	int	Does the beneficiary have Osteoporosis
20. ChronicCond_rheumatoid arthritis	int	Does the patient have rheumatoid arthritis
21. ChronicCond_stroke	int	Does the patient have a stroke
22. IPAnnualReimbursementAmt	int	Yearly reimbursement amount for the beneficiary when s/he is admitted as an inpatient
23. IPAnnualDeductibleAmt	int	Yearly deductible amount for the beneficiary when s/he is admitted as

		an inpatient
24. OPAnnualReimbursementAmt	int	Yearly reimbursement amount for the beneficiary when s/he is admitted as an outpatient
25. OPAnnualDeductibleAmt	int	Yearly deductible amount for the beneficiary when s/he is admitted as an outpatient

1.1.3.2. Inpatient (Rows=, Columns=30)

Name	Data Type	Description
1. BeneID	obj	Healthcare Beneficiary Identity
2. ClaimID	obj	Claim Identity entered by the provider
3. ClaimStartDt	obj	Date from which claim for the beneficiary was filed by the healthcare provider
4. ClaimEndDt	obj	Date till which claim for the beneficiary was filed by the healthcare provider
5. Provider	obj	Healthcare provider
6. InscClaimAmtReimbursed	int	Insurance claim amount reimbursed by the insurer
7. AttendingPhysician	obj	Attending physician
8. OperatingPhysician	obj	Physician under which surgery or operation was done
9. OtherPhysician	obj	Any other physician under which beneficiary was taken care
10. AdmissionDt	obj	Admission date of the beneficiary

11. ClmAdmitDiagnosisCode	obj	Diagnosis code for the admit claim
12. DeductibleAmtPaid	float	Deductible amount paid by the insurer
13. DischargeDt	obj	Discharge date of the beneficiary
14. DiagnosisGroupCode	obj	Group code for the diagnosis

There are 17 more columns, for which metadata as per the source website is not described.

1.1.3.3. Outpatient (Rows=, Columns=27)

Name	Data Type	Description
1. BeneID	object	Healthcare Beneficiary Identity
2. ClaimID	object	Claim Identity entered by the provider
3. ClaimStartDt	object	Date from which claim for the beneficiary was filed by the healthcare provider
4. ClaimEndDt	object	Date till which claim for the beneficiary was filed by the healthcare provider
5. Provider	object	Healthcare provider
Name	Data Type	Description
6. InscClaimAmtReimbursed	int	Insurance claim amount reimbursed by the insurer
7. AttendingPhysician	object	Attending physician
8. OperatingPhysician	object	Physician under which surgery or operation was done
9. OtherPhysician	object	Any other physician under which beneficiary was taken care
10. DeductibleAmtPaid	int	Deductible amount paid by the insurer

11. ClmAdmitDiagnosisCode	object	Diagnosis code for the admit claim

There are 16 more columns for which metadata is not available at the source website.

1.1.3.4. Target (Rows=, Columns=2)

Name	Data Type	Description
1. Provider	object	Healthcare Provider who is filing the claim
2. PotentialFraud	object	Flag whether the provider was identified as fraudulent or not

1.1.3.5. All Possible Performance Metrics

- Confusion matrix: If we go for finding the confusion matrix, we can go ahead with creating precision, recall and F1-score, which will give us a more detailed analysis of the performance of our models.

Actual ➔ **Predicted ⬇**	**0**	**1**
0	**True negative**	**False negative**
1	**False positive**	**True positive**
	Total negatives	**Total positives**

- Precision: As per the confusion matrix results, the precision findings can give us a measure of out of *all the points declared as positive*, what percent of them are *actually positives*.

$$Precision \ = \ true\,positives/(true\,positives\ +\ false\,positives)$$

- Recall: As per the confusion matrix results, the recall findings can give us a measure of out of all the points *belonging to class positive*, how many of them are *predicted to be positive*.

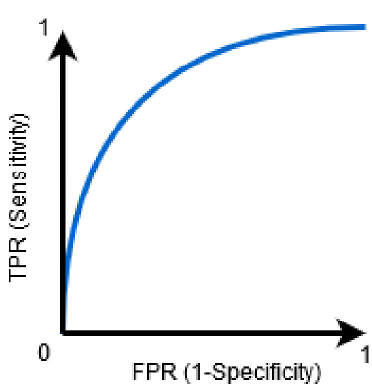
$$recall \ = \ true\,positives/(true\,positives\ +\ false\,negatives)$$

- F1-score: It is helpful in figuring out the correct balance between the two measures of precision and recall. It is nothing but the harmonic mean of precision and recall.

$$F1 - Score \ = \ 2\,.\,(precision\,.\,recall)/(precision\ +\ recall)$$

- Accuracy: If we apply an upsampling technique to balance the dataset, we can go for accuracy as one of the measures to check the performance of our classification models.

$$Accuracy \ = \ total\ \#\ of\ points\ correctly\ classified/total\ \#\ of\ points\ in\ the\ test\ set$$



- AUC-ROC: it is used for binary classification tasks. Assuming that the model outputs a score which can be interpreted similar to the probability of a class 1 score. The AUC values lie between 0 and 1, where 0 is the worst and 1 being the best. *Image credits: Analytics Vidhya*

Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed, you can include images or explain with your own diagrams. it is mandatory to write a brief description about that paper. Without understanding of the resource please don't mention it***

2. Research

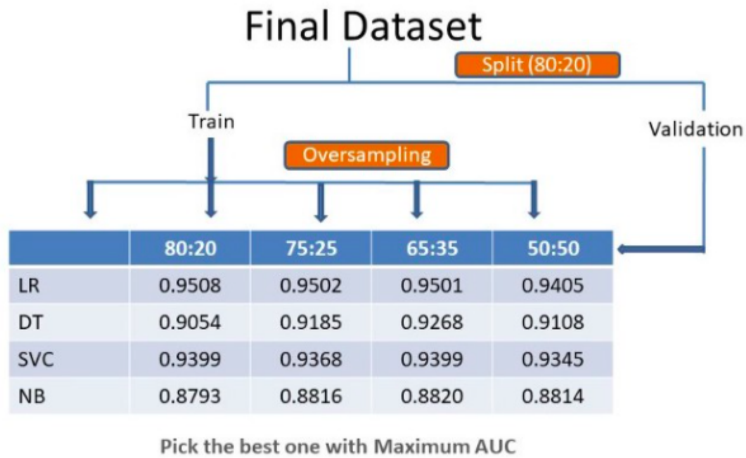
2.1. Resources

2.1.1. https://medium.com/analytics-vidhya/healthcare-provider-fraud-detection-analysis-using-machine-learning-81ebf09ed955

Observations:

a. This blog describes the cost of misclassification, which as per the author is very high. It says that False negatives and False positives should be as low as possible. As the misclassification can happen, even if the model is too brilliant, the insurer must provide a reason as to why it declared a particular claim as fraudulent. And then they can go for manual investigation. And since the insurer has promised to reimburse all the genuine claims, they must pass it within 30 days or as stipulated. So, there is a strict latency constraint, but it should not take more than a day.

b. As per the blog, after splitting the dataset, oversampling should be done only on the train and not on the validation. The test data is already given separately. It seems that the author has chosen only the SMOTE oversampling method, which is too simple in its approach.

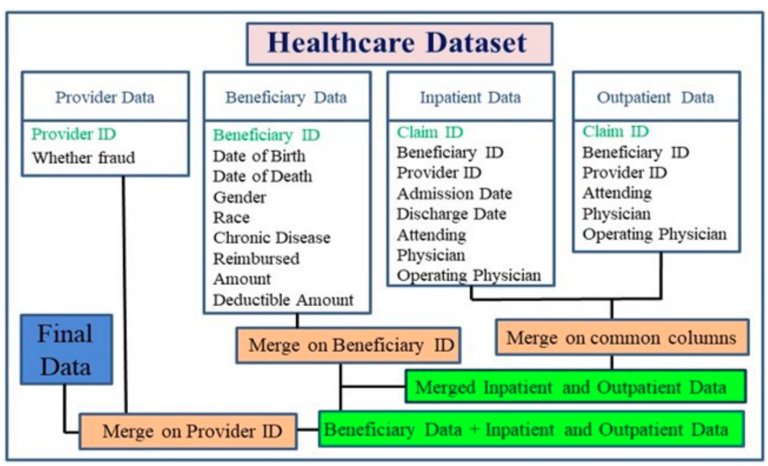


	80:20	75:25	65:35	50:50
LR	0.9508	0.9502	0.9501	0.9405
DT	0.9054	0.9185	0.9268	0.9108
SVC	0.9399	0.9368	0.9399	0.9345
NB	0.8793	0.8816	0.8820	0.8814

Pick the best one with Maximum AUC

Source: Analytics Vidhya blog on medium.com

c. The author has also merged all the three datasets. For this, the author has first created some features such as 'age', category - 'if dead', 'claim duration', 'hospital stay duration'. The author has also grouped and then taken mean or average in several circumstances.



Overall Representation of The Dataset

Source: Analytics Vidhya blog on medium.com

Takeaway:

a. Except for the SMOTE technique, all of the rest seem to be meaningful. The 'merging after feature engineering' and the 'business constraints' described in point **a** are very useful.

2.1.2. https://rohansoni-jssaten2019.medium.com/healthcare-provider-fraud-detection-and-analysis-machine-learning-6af6366caff2

Observations: How to raise suspicion on a claim submitted for adjudication:

a. Excessive price charged on treatment/medicine
b. Unusual high number of invoices for a beneficiary in a short time period
c. Billing for a service which was not prescribed for the beneficiary.
d. Duplicate claim submission
e. Misrepresentation of the service provided
f. Charging for a more complex service than what was actually provided

Takeaway:

a. All the above key points seem important for EDA to understand the claim fraud, in-depth.

2.1.3. https://nycdatascience.com/blog/student-works/healthcare-fraud-detecting-inconsistencies-in-provider-data/

Observation:

a. It has a really good EDA description as well as feature engineering techniques.
b. Diagrams are very much noticeable and worth the effort of making it understandable to a viewer.

Takeaway:

a. The process of EDA can take inspiration from this blog.

2.1.4. https://www.datasciencecentral.com/profiles/blogs/deep-learning-detecting-fraudulent-healthcare-provider-using

Observations:

a. KNN, Autoencoders(Deep Neural Networks), K-Means, Support Vector Machines, Naive Bayes – Could be very useful for anomaly detection.
b. Unsupervised algorithms of ML would be better for highly imbalanced datasets.

2.1.5. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6120851/

Observations:

a. Random Forest produced the highest AUC of 0.87302 for the 90:10 class distribution.
b. Naive Bayes being the worst performing learner
c. Except the C4.5 Decision Trees, the trend across class distribution, shows that most learners have decreasing performance as the classes become more imbalanced with less minority class representation.
d. K-Fold Cross-Validation can be incorporated.

Takeaway:

a. With the above findings, we need to experiment with a few models in such a manner to have the best distribution of data in a good ratio such as 90:10 or 80:20 for Train:Validation.
b. Experimenting a bit can give us better results.

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

*** When you are doing the basic EDA and building the First Cut Approach you should not refer any blogs or papers ***

1. We need a good oversampling method and not a SMOTE. Because it is very simple in its approach. Applying SMOTE might not give us good results although we can try.
2. We need feature engineering to build good features such as age, if_dead, claim and hospital-stay duration.
3. Combining the 3 different datasets to work for getting best results.
4. We can also use features which prove more useful in predicting targets by plotting graphs and with the help of that we can create more features.
5. For encoding categorical features, mean/target encoding could be more useful. Although, mean-encoding can lead to overfitting, use of a regularizer can be beneficial.

6. We can use percentile methods to remove extreme outliers.
7. Models such as Random-Forest, GBDT-XGBoost would be tried first and then would go to Support Vector Machines, logistic Regression and Naive Bayes at last.

Notes when you build your final notebook:

1. You should not train any model either it can be a ML model or DL model or Countvectorizer or even simple StandardScalar
2. You should not read train data files
3. The function1 takes only one argument "X" (a single data points i.e 1*d feature) and the inside the function you will preprocess data point similar to the process you did while you featurize your train data
   a. Ex: consider you are doing taxi demand prediction case study (problem definition: given a time and location predict the number of pickups that can happen)
   b. so in your final notebook, you need to pass only those two values
   c. def final(X):
      preprocess data i.e data cleaning, filling missing values etc
      compute features based on this X
      use pre trained model
      return predicted outputs
      final([time, location])
   d. in the instructions, we have mentioned two functions one with original values and one without it
   e. final([time, location]) # in this function you need to return the predictions, no need to compute the metric
   f. final(set of [time, location] values, corresponding Y values) # when you pass the Y values, we can compute the error metric(Y, y_predict)
4. After you have preprocessed the data point you will featurize it, with the help of trained vectorizers or methods you have followed for your train data
5. Assume this function is like you are productionizing the best model you have built, you need to measure the time for predicting and report the time. Make sure you keep the time as low as possible
6. Check this live session: https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4148/

hands-on-live-session-deploy-an-ml-model-using-apis-on-aws/5/module-5-feature-engineering-productionization-and-deployment-of-ml-models