Fraud Detection in Credit Card Transactions

...

Mentor - Dr. Bhaskar Biswas
Vivek Kumar Yadav 416
Kunal Gupta 347
Utsav Raj 411

Problem statement

The aim of the project is to predict fraudulent credit card transactions using machine learning models.

This is crucial from the bank's as well as customer's perspective. The banks cannot afford to lose their customers' money to fraudsters. Every fraud is a loss to the bank as the bank is responsible for the fraud transactions.

The dataset contains transactions made over a period of two days by credit cardholders.

Prerequisites

Most of the libraries like sklearn, pandas, seaborn, matplotlib, numpy, scipy.

We will be Training the model with various algorithm such as Logistic regression, Decision Tree, Random forest & XGboost also their comparison.

We are working on jupyter Notebook

Steps to be taken

Reading, understanding and visualising the data (EDA)

- It is a CSV file, contains 31 features, the last feature is used to classify the transaction whether it is a fraud or not.

Preparing the data for modelling

Building the model (with different model)

Evaluate the model

DataSet

It is a CSV file, contains 31 features, the last feature is used to classify the transaction whether it is a fraud or not

This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. Which is 0.17%.

It contains only numeric input variables. Due to secrecy issues original features are not provided Features V1, V2, … V28 are the principal components obtained is the only features

link :https://www.kaggle.com/mlg-ulb/creditcardfraud

Using ROC-AUC Receiver Operating characteristic curve

As we have seen that the data is heavily imbalanced, where only 0.17% transactions are fraudulent, we should not consider accuracy as a good measure for evaluating the model. Because in the case of all the data points return a particular class(1/0) irrespective of any prediction, still the model will result very high Accuracy.

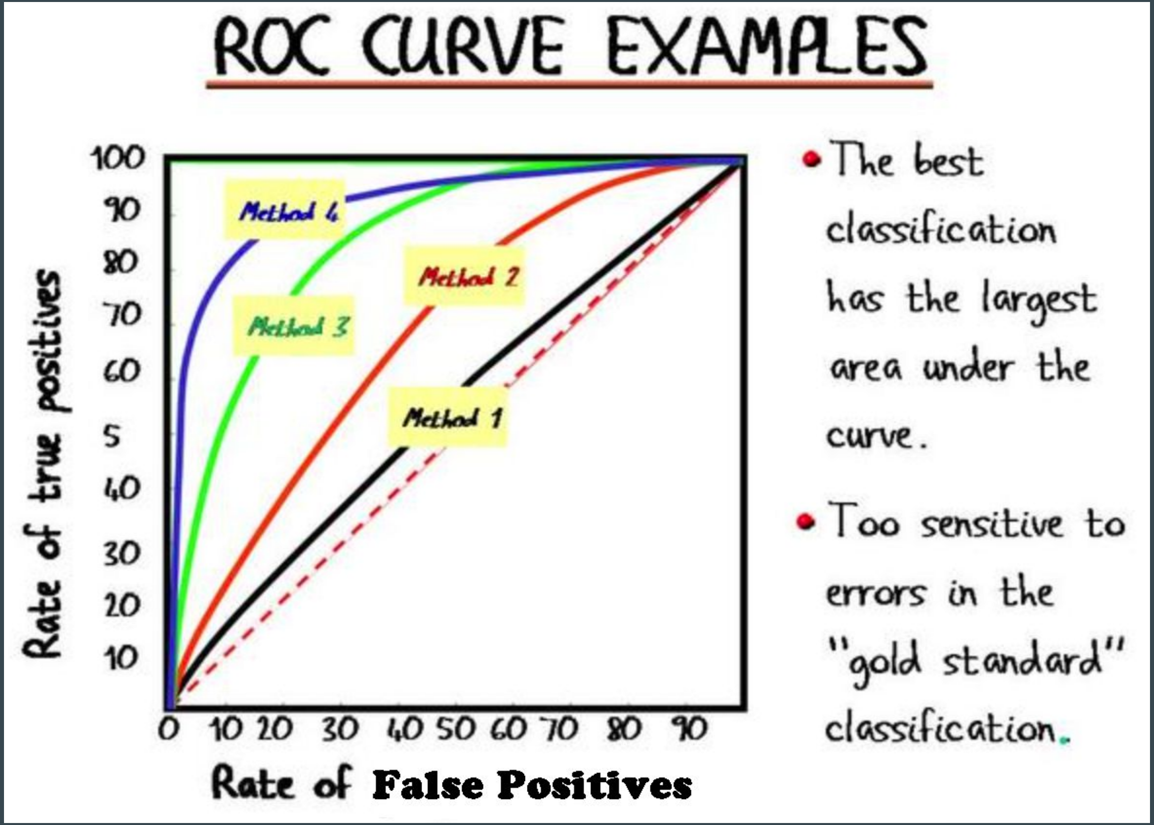
Hence, we have to measure the ROC-AUC score for fair evaluation of the model.

The ROC curve is used to understand the strength of the model by evaluating the performance of the model at all the classification thresholds.

Overall accuracy is based on one specific threshold, while ROC tries all of the threshold and plots the sensitivity and specificity. So when we compare the overall accuracy, we are comparing the accuracy based on some threshold.

ROC curve

fig.



- The best classification has the largest area under the curve.
- Too sensitive to errors in the "gold standard" classification.

Using ROC-AUC

The ROC curve is measured at all thresholds, the best threshold would be one at which the TPR(True Positive Rate) is high and FPR(False Positive Rate) is low, i.e., misclassifications are low. After determining the optimal threshold, we can calculate it.

F1 Score(is a measure of a model's accuracy on a dataset) of the classifier to measure the precision and recall at the selected threshold.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Accuracy , Precision , Recall

Accuracy - it is simply a ratio of correctly predicted observation to the total observations.

Accuracy = TP+TN/TP+FP+FN+TN

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Precision = TP/TP+FP

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class

Recall = TP/TP+FN

What we are Learning.

ALL different models discussed above,

Along with implementation.

THANK YOU

References:

https://spd.group/machine-learning/credit-card-fraud-detection/

https://www.kaggle.com/mlg-ulb/creditcardfraud

https://towardsdatascience.com/credit-card-fraud-detection-using-machine-learning-python-5b098d4a8edc

AFTER EDA MODEL SELECTION AND PREDICTION.

CONTINUE FROM HERE

MODEL Demonstration

Classification Models we are using over imbalanced data set.

1 .LOGISTIC Regression.

2 .DECISION TREE

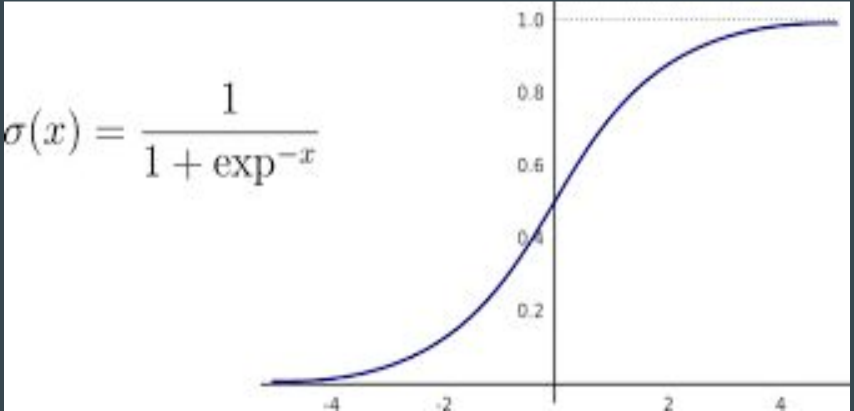
3 .XGBOOST

If needed we can check with some other classification models.

But our goal is to find best suited model that consumes less computational resources and build on infrastructure that takes less deploying cost.

Logistic Regression

- It is the Supervised Learning technique used for predicting the categorical dependent variable(**predicted variable (class)**) using a given set of independent variables(**predictors (our independent variables**)(v1 to v28)
- It predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either **Yes or No, 0 or 1, true or False**, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.



Logistic Function (Sigmoid Function):

The sigmoid function is a mathematical function used to map the predicted values to probabilities.

It maps any real value into another value within a range of 0 and 1.

The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.

The mathematical steps to get Logistic Regression equations are given below:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

Cross validation & GridSearchCV

The most common is the k-fold cross-validation technique, where you divide your dataset into k distinct subsets. By choosing 1 of the k subsets to be the validation set, and the rest k–1 subsets to be the training set, we can repeat this process k times by choosing a different subset to be the validation set every time. This makes it possible to repeat the training-validation process k times, eventually going through the entire original training set as both training and validation set.

GridSearchCV -It is the process of performing hyperparameter tuning in order to determine the optimal values for a given model.

Prediction on the Train & Test set (LOGISTIC)

Train set

Accuracy = 0.99

Sensitivity = 0.70 Specificity = 0.99 F1-Score = 0.76

ROC = 0.99

Test set

Accuracy = 0.99 Sensitivity = 0.77 Specificity = 0.99 F1-Score = 0.65

ROC = 0.97

Decision Tree

Decision Tree's are an excellent way to classify classes, unlike a Random forest they are a transparent or a whitebox classifier which means we can actually find the logic behind decision tree's classification.

Next we're going to initialise our classifier and **GridSearchCv** which is the main component which will help us find the best hyperparameters.

We simply create a tuple (kind of non edit list) of hyperparameters we want the machine to test with as save them as parameters.

We then give the classifier and list of params as paramters to gridsearchcv.

GridSearchCV & tuning of hyperparameters.

GridSearchCV -It is the process of performing hyperparameter tuning in order to determine the optimal values for a given model.

The performance of a model significantly depends on the value of hyperparameters.

Note that there is no way to know in advance the best values for hyperparameters so ideally, we need to try all possible values to know the optimal values.

Doing this manually could take a considerable amount of time and resources and thus we use GridSearchCV to automate the tuning of hyperparameters.

Prediction on the Train & Test set(DECISION TREE)

Train set

Accuracy = 0.99 Sensitivity = 1.0 Specificity = 1.0 F1-Score = 0.75

ROC-AUC = 0.95

Test set

Accuracy = 0.99 Sensitivity = 0.58 Specificity = 0.99 F-1 Score = 0.75

ROC-AUC = 0.92

XGBOOST

Made a separate ppt

Take it from there.

XGBoost Model.

•••

Mentor - Dr. Bhaskar Biswas
Vivek Kumar Yadav 416
Kunal Gupta 347
Utsav Raj 411

What we have done till now.

1. EDA over data set
2. Power transformation
3. Logistic regression (As Classification)
4. Decision tree (Classifier)
5. Covering in this ppt :- XGboost Classification
6. Next task mentioned below.

Choosing best model on the balanced data using these techniques.
-Undersampling,
-Oversampling,

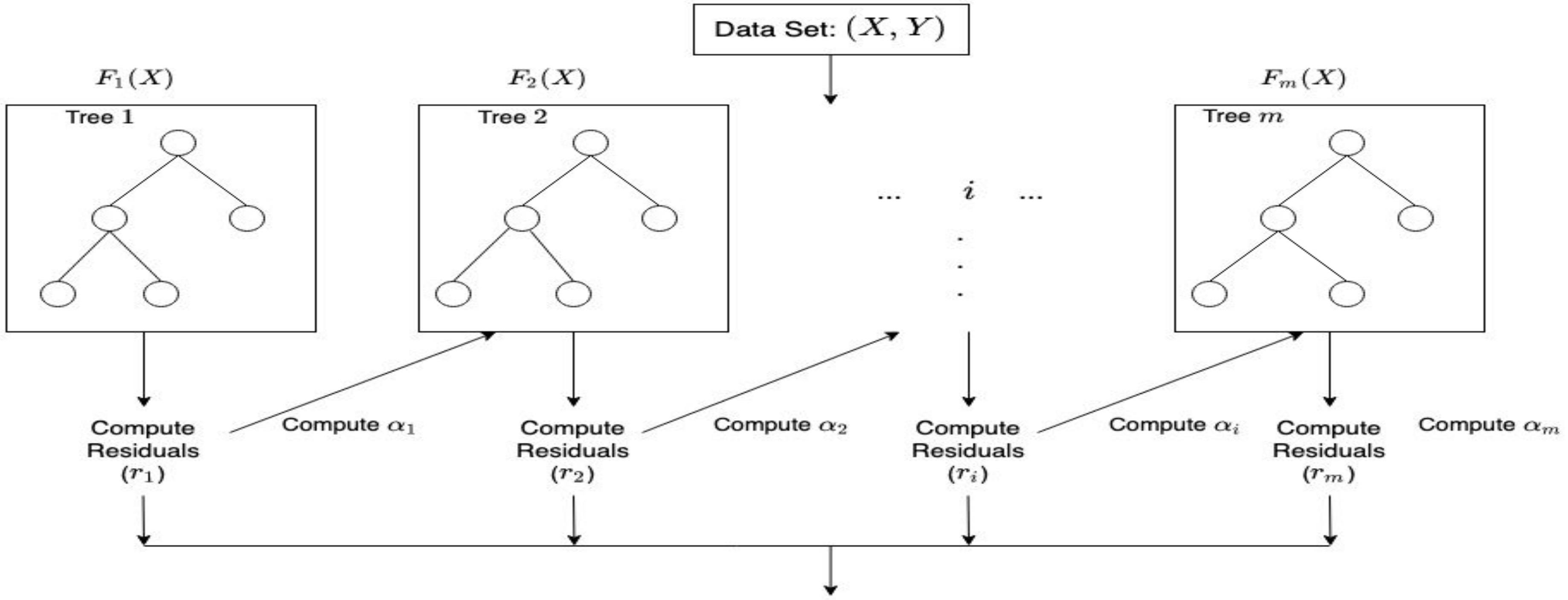
About XGboost

XGBoost is boosting sequential technique which works on the principle of an ensemble(combine several base model to produce one optimal predictive model). It combines a set of weak learners and delivers improved prediction accuracy.

when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.

This learning algorithm whose **individual learning units are decision trees** and trees have two favorable features which, again, render feature engineering unnecessary.

How it works internally



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

Hyperparameter tuning with XGBoost

Hyper-parameter tuning can considerably improve the performance of learning algorithms.

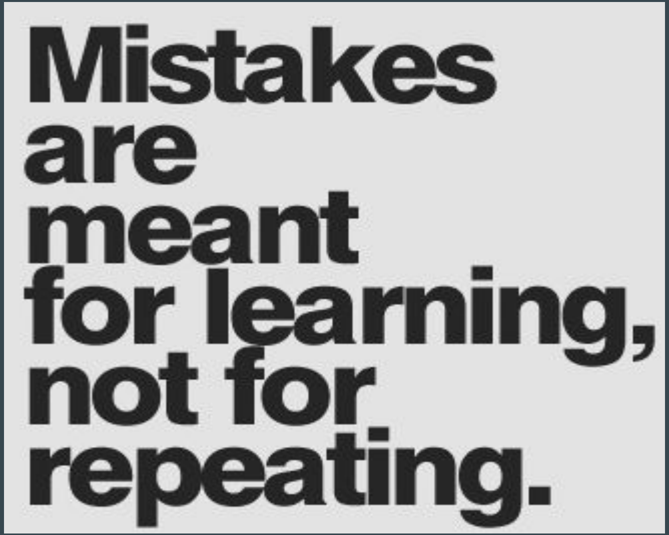
XGBoost has many hyper-parameters which make it powerful and flexible, but also very difficult to tune due to the high-dimensional parameter space. Instead of the more traditional tuning methods

(i.e. grid search and random search) that perform a search through the parameter space,

Challenges & Solutions

The main challenge in fraud detection is the extreme class imbalance in the data which makes it difficult for many classification algorithms to effectively separate the two classes. Only 0.172% of transactions are labeled as fradulent in this dataset. So we will address the class imbalance by reweighting the data before training XGBoost

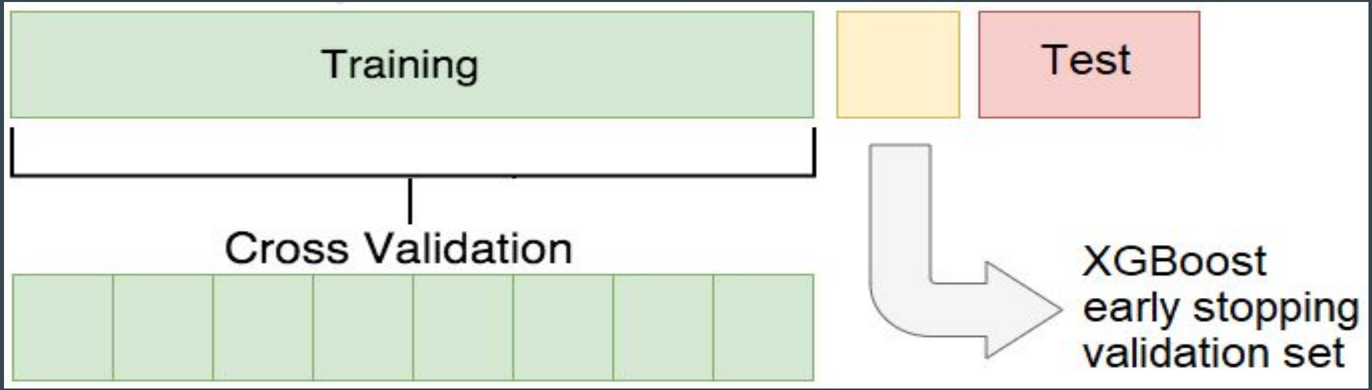
MOTIVATION



Mistakes are meant for learning, not for repeating.

Tuning by Cross Validation

Here we are using most common is the k-fold cross-validation technique, where you divide your dataset into k distinct subsets. By choosing 1 of the k subsets to be the validation set, and the rest k–1 subsets to be the training set, we can repeat this process k times by choosing a different subset to be the validation set every time. This makes it possible to repeat the training-validation process k times, eventually going through the entire original training set as both training and validation set.



Why tuning by Cross Validation

To prevent the early stopping feature from overfitting, in addition to splitting the data into a training and a test set, I also set aside a small portion of the training set to be used as the early stopping validation set. The figure below shows how the data is split.

In above fig.

Splitting the dataset. split the dataset into train and test sets. Then use a small part (of data) of the training set as the early stopping validation set.

Hyperparameter Tuning with GridSearchCV

It is the process of performing hyperparameter tuning in order to determine the optimal values for a given model cause the performance of a model significantly depends on the value of hyperparameters. So we use GridSearchCV to automate the tuning of hyperparameters.

This helps us to loop through predefined hyperparameters and fit estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyperparameters.

Specifying the range of hyperparameters {'learning_rate': [0.2, 0.6],

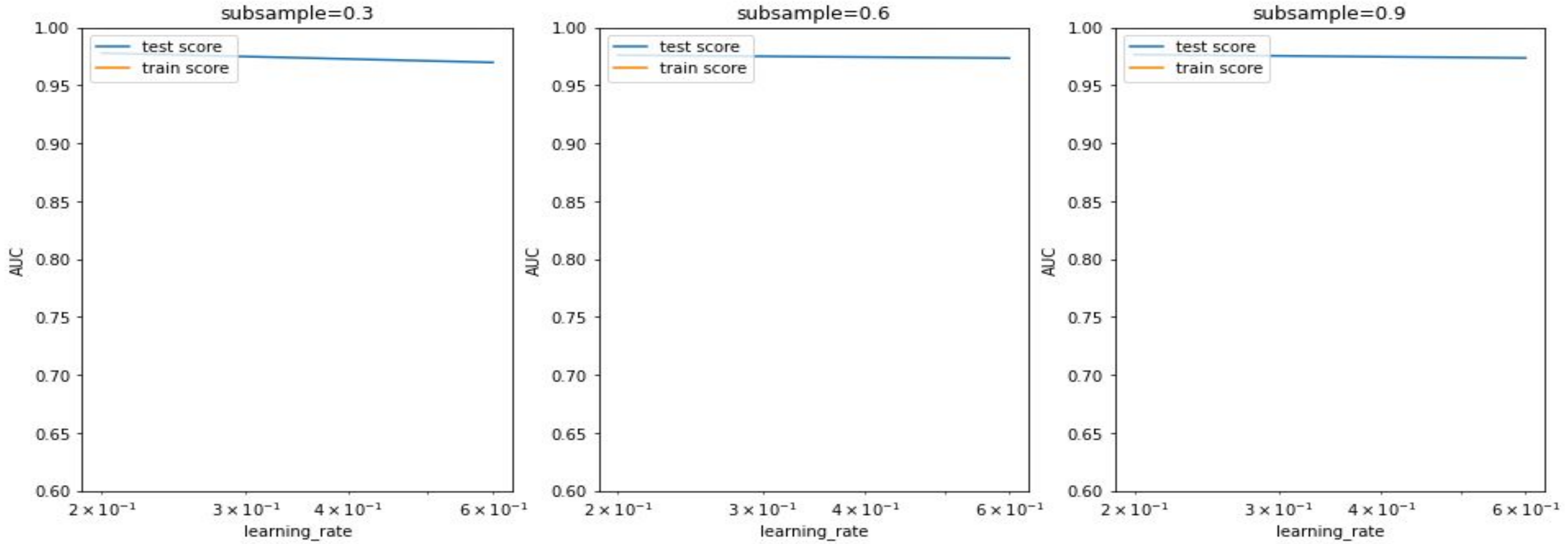
'subsample': [0.3, 0.6, 0.9]}

Arguments taken by GridSearchCV.

1. **estimator:** Pass the model instance for which you want to check the hyperparameters.
2. **params_grid:** the dictionary object that holds the hyperparameters you want to try
3. **scoring:** evaluation metric that you want to use, you can simply pass a valid string/ object of evaluation metric
4. **cv:** number of cross-validation you have to try for each selected set of hyperparameters
5. **verbose:** you can set it to 1 to get the detailed print out while you fit the data to GridSearchCV
6. **n_jobs:** number of processes you wish to run in parallel for this task if it -1 it will use all available processors.

Model with optimal hyperparameters

We see that the train score almost touches to 1. Among the hyperparameters, we can choose the best parameters as learning_rate : 0.2 and subsample: 0.3



Metrics not covered.

As metrics are explained in all previous slides so not covering in this,

As we are calculating all those things in every models

"**Accuracy**:-",metrics.accuracy_score(y_test, y_test_pred

"**Sensitivity:**-",TP / float(TP+FN

"**Specificity:**-", TN / float(TN+FP)

"F1-Score:-", f1_score(y_test, y_test_pred

Conclusion Till now & Preparing for next Task.

We can see that between the models we tried (Logistic,, Decision Tree, and XGboost), almost all of them have performed well.

More specifically Logistic regression and XGBoost performed best in terms of ROC-AUC score.

But as we have to choose one of them, we can go for the best as XGBoost, which gives us ROC score of 1.0 on the train data and 0.98 on the test data.

Keep in mind that XGBoost requires more resource utilization than Logistic model. Hence building XGBoost model is more costlier than the Logistic model. But XGBoost having ROC score 0.98, which is 0.01 more than the Logistic model. The 0.01 increase of score may convert into huge amount of saving for the bank.

Finally our task is to predict accuracy using ROC-AUC curve after choosing best balancing technique mentioned. So we are gearing up for that.

End

THANK YOU

Refrences:https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

Bergstra, James, Daniel Yamins, and David Daniel Cox. 2013. “Making a science of model search: Hyperparameter optimization in hundreds of dimensions forvision architectures”.

Chawla, Nitesh V, et al. 2002. “SMOTE: synthetic minority over-sampling technique”. Journal of artificial intelligence research 16:321–357.

Davis, Jesse, and Mark Goadrich. 2006. “The relationship between Precision-Recall and ROC curves”. In Proceedings of the 23rd international conferenceon Machine learning, 233–240. ACM.

Aim of the Project

Choosing best model on the balanced data.

Here we are trying to balance the data with various approach such as

- Undersampling,
- Oversampling,
- SMOTE ->Synthetic minority oversampling technique
- Adasy.

With every data balancing technique we built several models such as Logistic, XGBoost, Decision Tree, and Random Forest.

WHY NEED TO BALANCE DATA

We have seen that the data is heavily imbalanced, where only **0.17% transactions are fraudulent**, we should not consider simply a accuracy as a good measure for evaluating the model. Because in the case of all the data points return a particular class(1/0) irrespective of any prediction, still the model will result very high Accuracy.

Hence, we have to measure the ROC-AUC score for fair evaluation of the model,along with Balancing techniques given below.

Aim of the Project to choose best balancing technique.

Undersampling :- Here for balancing the class distribution, the non-fraudulent transactions count will be reduced to 396 (similar count of fraudulent transactions)

Oversampling :- Here we will make the same count of non-fraudulent transactions as fraudulent transactions.

SMOTE :- Synthetic minority oversampling technique. It is another oversampling technique, which uses nearest neighbor algorithm to create synthetic data.

Adasyn:- This is similar to SMOTE with minor changes that the new synthetic data is generated on the region of low density of imbalanced data points.

Our Foremost Task, Predict accuracy of model and Feasibility

Finally our task is to predict accuracy using ROC-AUC curve after choosing best balancing technique mentioned above.

Like we can perform all three classification models in **Undersampling, Oversampling , SMOTE, and in Adasyn respectively.**

Only to get best suited models over tried classification model.

Reason : *We also have to consider that for little change of the ROC score how much monetary loss of gain the bank incur..*

Balancing technique Undersampling & SMOTE

...

Mentor - Dr. Bhaskar Biswas

Vivek Kumar Yadav 416

Kunal Gupta 347

Utsav Raj 411

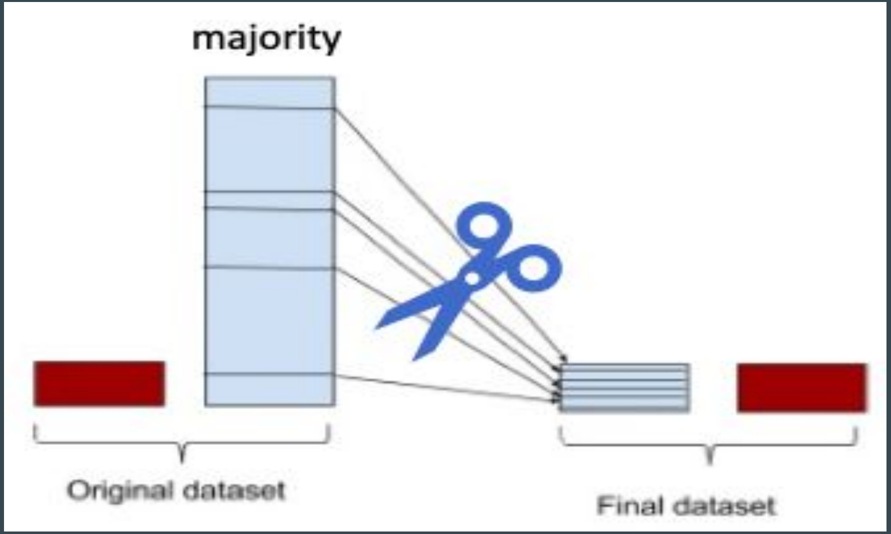
UNDERSAMPLING

Idea behind & implementation of Undersampling

In this we reduces data in majority class so that it can match with data of minority class. Basically we will shrink down the majority to minority ,

If we don’t do this model will be biased towards majority data set so we are doing balancing

It will increase classification performance.



Idea behind & implementation of Undersampling

Working with libraries.

Imblearn -provides tools when dealing with classification with imbalanced classes.

Using undersampler library which under-sample the majority class(es) by randomly picking samples with or without replacement

Accuracy (ROC-AUC) Comparison

Logistic -without any balancing technique

Train set : Accuracy = 0.99 Sensitivity = 0.70 Specificity = 0.99 ROC = 0.99

Test set : Accuracy = 0.99 Sensitivity = 0.77 Specificity = 0.99 ROC = 0.97

Decision tree -without any balancing technique

Train set: Accuracy = 0.99 Sensitivity = 1.0 Specificity = 1.0 ROC-AUC = 0.95

Test set : Accuracy = 0.99 Sensitivity = 0.58 Specificity = 0.99 ROC-AUC = 0.92

Logistic with Undersampling

Logistic

Train set Accuracy = 0.95 Sensitivity = 0.92 Specificity = 0.98

ROC = 0.99

Test set

Accuracy = 0.97 Sensitivity = 0.86 Specificity = 0.97

ROC = 0.96

Decision Tree with Undersampling

Train set

Accuracy = 0.93 Sensitivity = 0.88 Specificity = 0.97

ROC-AUC = 0.98

Test set

Accuracy = 0.96 Sensitivity = 0.85 Specificity = 0.96

ROC-AUC = 0.96

SMOTE

Smote

Synthetic minority overlapping technique uses **nearest neighbour algorithm** to create synthetic data.

Smote takes the entire dataset as input and increases the percentage of only the minority cases.

To double the percentage of minority set the module property to 200%

To triple set to 300% and so on.

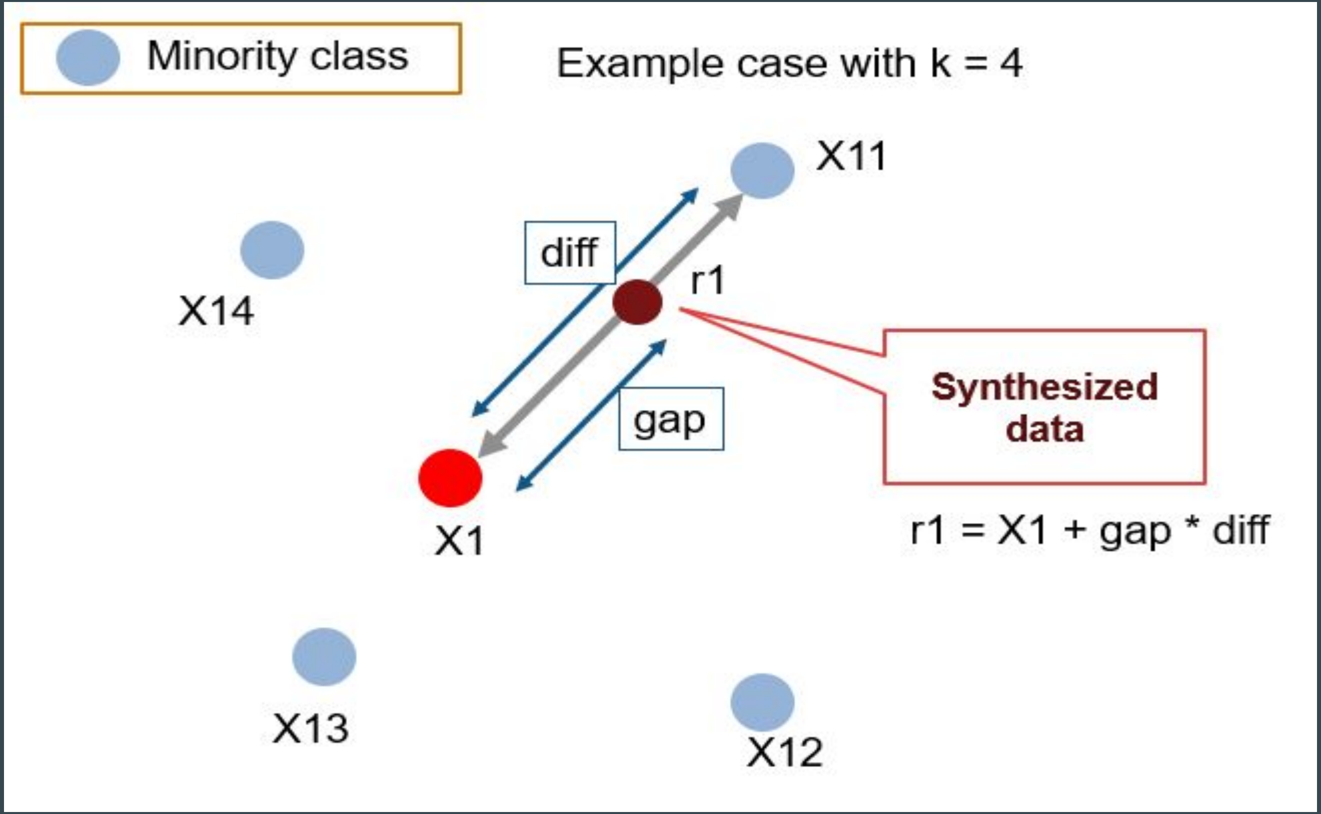
Working Procedure

At first the total no. of oversampling observations, N is set up. Generally, it is selected such that the binary class distribution is 1:1. But that could be tuned down based on need. Then the iteration starts by first selecting a positive class instance at random.

Next, the KNN's (by default 5) for that instance is obtained. At last, N of these K instances is chosen to interpolate new synthetic instances. To do that, using any distance metric the difference in distance between the feature vector and its neighbors is calculated. Now, this difference is multiplied by any random value in (0,1] and is added to the previous feature vector.

This is pictorially represented below:

•



Logistic with SMOTE

Train set

Accuracy = 0.95 Sensitivity = 0.92 Specificity = 0.98

ROC = 0.99

Test set

Accuracy = 0.97 Sensitivity = 0.90 Specificity = 0.99

ROC = 0.97

Decision tree with SMOTE balancing technique

Train set

Accuracy = 0.99Sensitivity = 0.99Specificity = 0.98

ROC-AUC = 0.99

Test set

Accuracy = 0.98Sensitivity = 0.80Specificity = 0.98

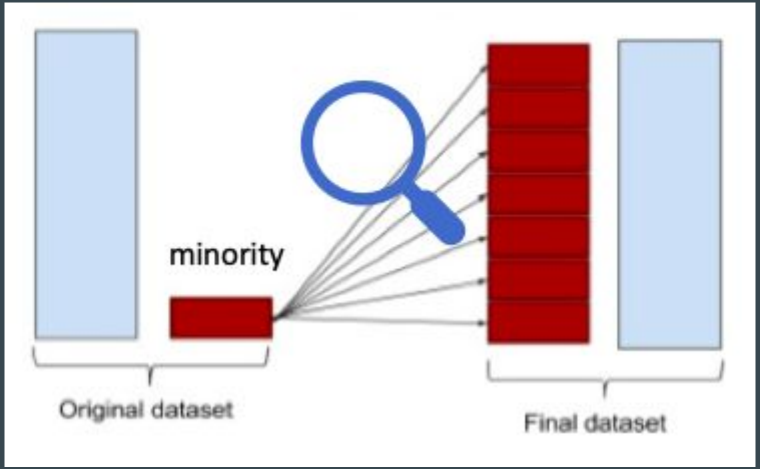
ROC-AUC = 0.86

Additional Technique - OVERSAMPLING

In this we increase the data of minority class so that it can match with data of majority class. Basically we will expand the minority to majority.

If we don't do this, model will be biased towards majority data set so we are doing balancing

It will increase classification performance.



Additional Technique - Adasyn

It is an algorithm that generates synthetic data and its advantages are not copying the same minority data and generating more data for “harder to learn” examples.

After creating those samples it adds random values so as to make it more realistic.

ADASYN implements a methodology that detects those samples of the minority class found in spaces dominated by the majority class,

Cause it focuses on those samples of the minority class that are difficult to classify because they are in a low-density area.

Choosing best model on the balanced data

Here we balanced the data with various approach such as Undersampling, Oversampling, (can also apply SMOTE and Adasy). With every data balancing technique we built several models such as Logistic, XGBoost, Decision Tree.

We can see that almost all the models performed more or less good. But we should be interested in the best model.Though the Undersampling technique models performed well, we should keep mind that by doing the undersampling some imformation were lost. Hence, it is better not to consider the undersampling models.

Whereas the SMOTE performed well. Among those models the simplist model Logistic regression has ROC score 0.98 in the train set and 0.97 on the test set. We can consider the Logistic model as the best model to choose because of the easy interpretation of the models and also the resoursce requirements to build the model is lesser than the other heavy models such as Random forest(NOT USING) or XGBoost.

Hence, we can conclude that the Logistic regression model with SMOTE is the best model for its simlicity and less resource requirement

Cost benefit analysis.

We have tried several models till now with both balanced and imbalanced data. We have noticed most of the models have performed more or less well in terms of ROC score, Precision and Recall.

But while picking the best model we should consider few things such as whether we have required infrastructure, resources or computational power to run the model or not. For the models such as Random forest, SVM, XGBoost we require heavy computational resources and eventually to build that infrastructure the cost of deploying the model increases. On the other hand the simpler model such as Logistic regression requires less computational resources, so the cost of building the model is less.

We also have to consider that for little change of the ROC score how much monetary loss of gain the bank incur. If the amount if huge then we have to consider building the complex model even though the cost of building the model is high.

Summary & Conclusion

For banks with smaller average transaction value, we would want high precision because we only want to label relevant transactions as fraudulent. For every transaction that is flagged as fraudulent, we can add the human element to verify whether the transaction was done by calling the customer. However, when precision is low, such tasks are a burden because the human element has to be increased.

For banks having a larger transaction value, if the recall is low, i.e., it is unable to detect transactions that are labelled as non-fraudulent. So we have to consider the losses if the missed transaction was a high-value fraudulent one.

So here, to save the banks from high-value fraudulent transactions, we have to focus on a high recall in order to detect actual fraudulent transactions.

After performing several models, we have seen that in the balanced dataset with SMOTE technique the simplest Logistic regression model has good ROC score and also high Recall. Hence, we can go with the logistic model here. It is also easier to interpret and explain to the business.

END

THANK YOU