IEEE-CIS Fraud Detection

Comparing different Gradient boosting techniques

Team:

Akshaykumar Rao Racherla

Akhil Koppera

Rachith Ramaswamy

Table of contents

Introduction

The gradient boosting decision tree (GBDT) is one of the best performing classes of algorithms in machine learning competitions.

Among the 29 challenge winning solutions published at Kaggle's blog during 2015, 17 used XGBoost.

Though XGBoost seemed to be the go-to algorithm in Kaggle for a while, a new contender is quickly gained traction: LightGBM - claimed to be more efficient (better predictive performance for the same running time) than XGBoost?

What makes these GBM's so popular and effective algorithms in machine learning ?

Decision Trees and moving towards forests and gradient boosting

- Decision trees are a method of splitting the data based on features to either classify or predict some value.
- A single decision tree can easily overfit the data. (Building a deep decision tree is prone to outlier observations)
- This is overcome by using ensemble techniques like bagging(parallel) and boosting(sequential)

Gradient boosting decision trees(GBDT)

- Multiple decision trees are used together and their predictions are added together to give the final prediction.
- GBDTs are trained iteratively:

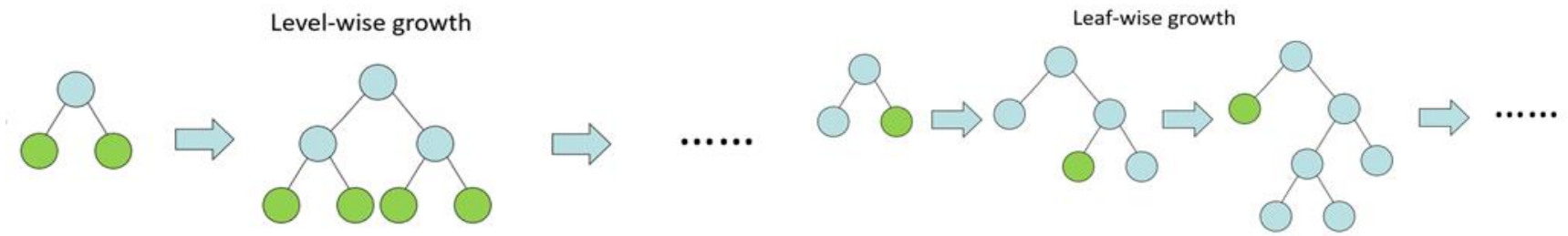
➢A weak decision is trained first -to minimize a loss function – such as the mean squared error.

➢ Done by recursively splitting the data in a way that maximizes some criterion until some limit – such as the depth of the tree – is met.

➢The next tree is then trained to minimize the loss function when its outputs are added to the first tree.

Motivation for Extreme Gradient Boosting and Light Gradient Boosting

- Computing the best split requires the model to go through various splits and compute the criterion for each split. There is no analytical solution for determining the best split at each stage -one of the major challenges of training GBDTs.
- A different approach approximates the split by building histograms of the features. That way, the algorithm doesn't need to evaluate every single value of the features to compute the split, but only the bins of the histogram, which are bounded. This approach turns out to be much more efficient for large datasets, without adversely affecting accuracy.
- Both XGBoost and lightGBM use the **leaf-wise** growth strategy when growing the decision tree.

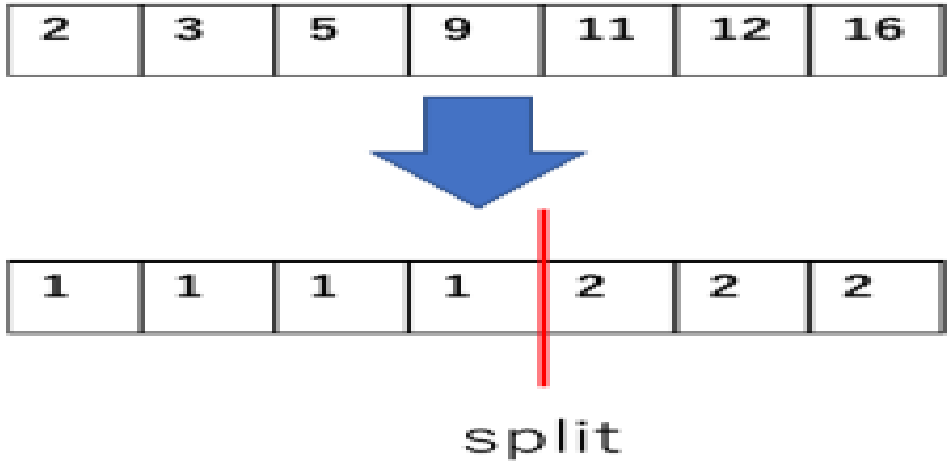


Working of XG boost and lightGBM

Histogram-based methods (Both XGBoost and LightGBM)

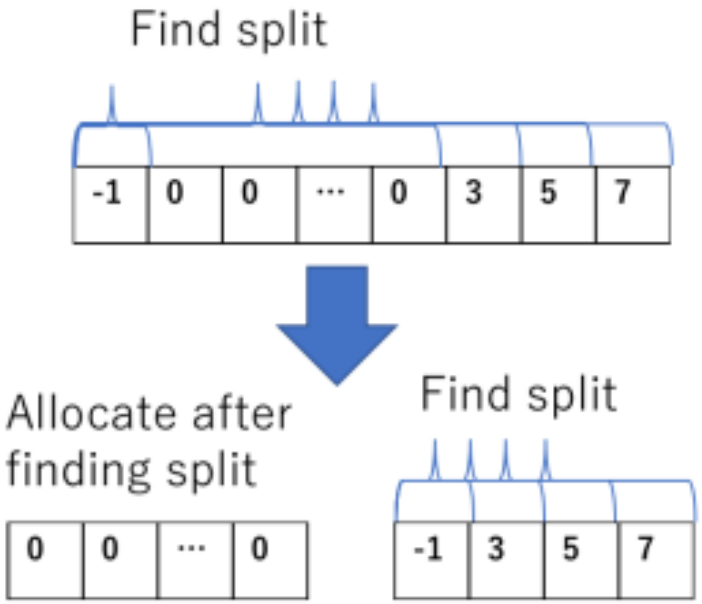
Histogram-based methods groups features into a set of bins and perform splitting on the bins instead of the features.

Since the features can be binned before building each tree, this method can greatly speed up training, reducing the computational complexity.



Ignoring sparse inputs (XGBoost and LightGBM)

XGBoost & LightGBM ignore the missing values when computing the split, then allocates all the data with missing values to whichever side of the split reduces the loss more.
This reduces the number of samples that have to be used when evaluating each split, speeding up the training process.



Gradient-based One-Side Sampling (exclusively in LightGBM)

Gradient-based One-Side Sampling is a shortcut to extract the most information from the dataset as fast as possible. This is achieved by adding a hyper parameter, alpha.
Alpha is the ratio of the samples to retain as the most informative samples. The remaining samples are duplicated so the complexity of the algorithm is reduced to a linear search over alpha samples.
The duplication of the less informative samples helps maintain the original data distribution.

Exclusive Feature Bundling (LightGBM)

Exclusive Feature Bundling (EFB) combines similar columns into a single feature without losing any information.

How to use LightGBM and XGBoost in practice?

IEEE-CIS Fraud Detection

Dataset Description

Dataset of credit card transactions provided by vesta corporation

Dataset contains identity and transaction CSV for both test and train.

Train data : 590540 x 433. Fraud transactions 20663

Size of Raw data:

```
[1] "train_identity: 144233 Rows and 41 Columns"
[1] "test_identity: 141907 Rows and 41 Columns"
[1] "train_transaction: 590540 Rows and 394 Columns"
[1] "test_transaction: 506691 Rows and 393 Columns"
```

Important Features and Variables OF DATA SET

TransactionDT: timedelta from a given reference datetime (not an actual timestamp)

TransactionAMT: transaction payment amount in USD

ProductCD: product code, the product for each transaction

card1 - card6: payment card information, such as card type, card category, issue bank, country, etc.

addr: address

dist: distance

P_ and (R__) emaildomain: purchaser and recipient email domain

C1-C14: counting, such as how many addresses are found to be associated with the payment card, etc.

Size of the data after performing inner joints using TransactionID:

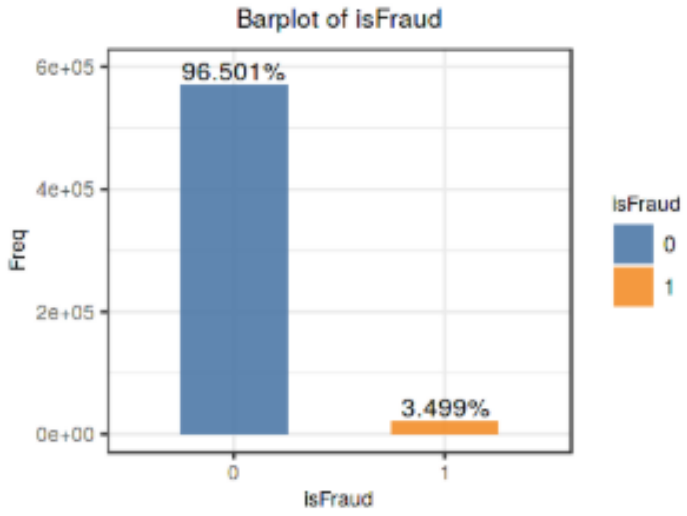
```
Joining, by = "TransactionID"
Joining, by = "TransactionID"


[1] 590540    434
[1] 506691    433
```

Number of Columns having at least one NA value.

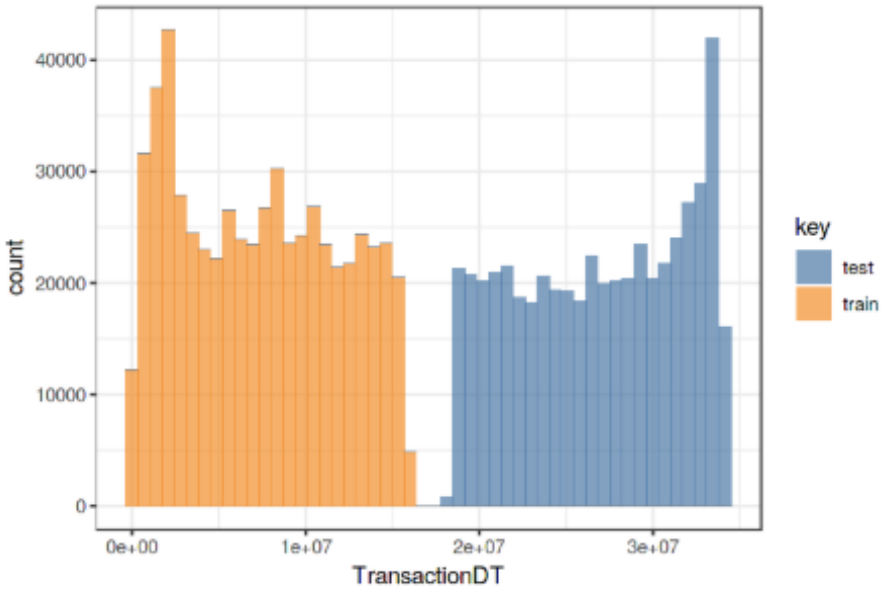
```
[1] "414 columns out of 434 have missing values in train"
[1] "385 columns out of 433 have missing values in test"
```

EDA

Target: isFraud

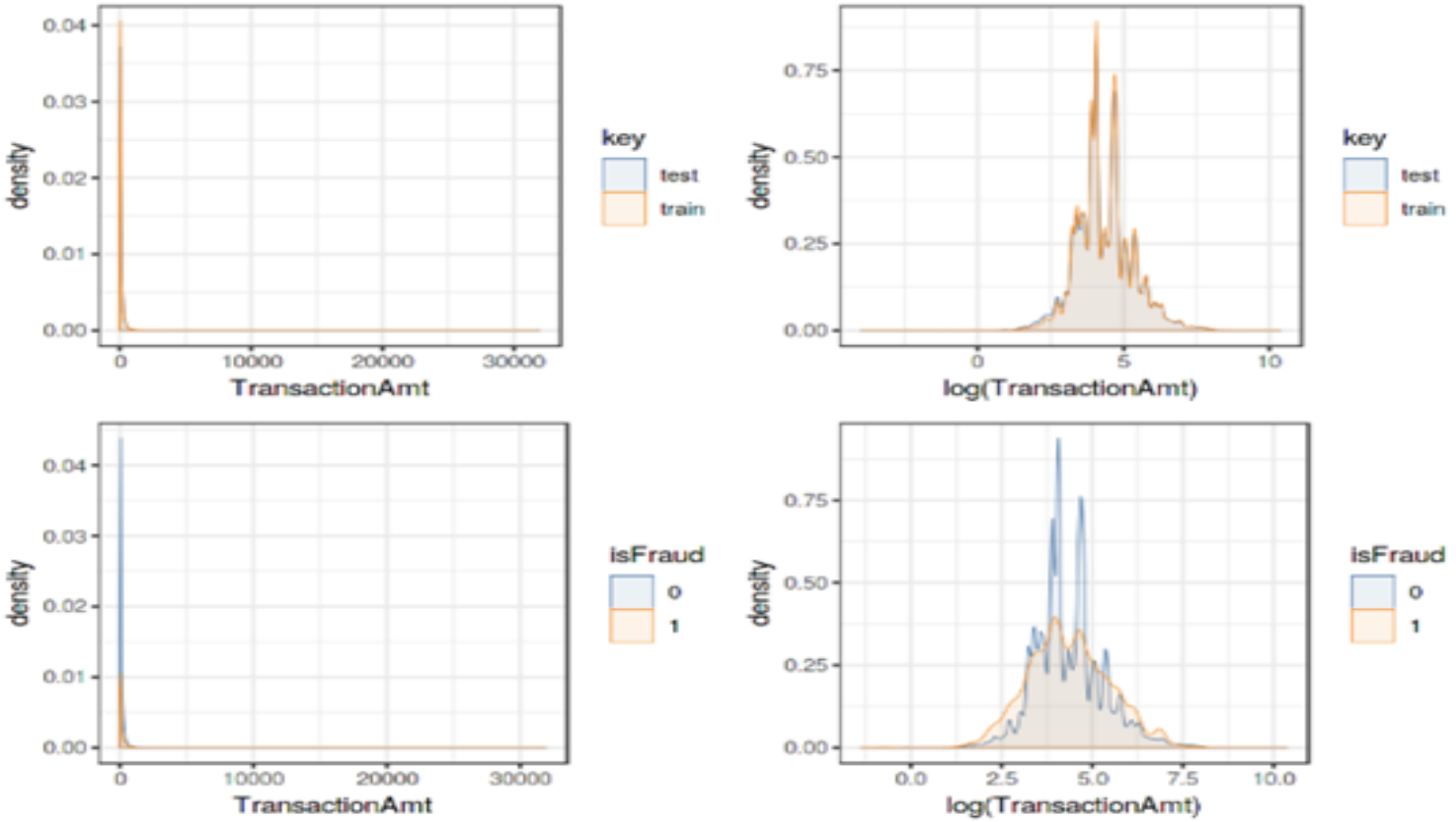


TransactionDT



EDA

TransactionAmt



Modelling XGBoost and LightGB on the dataset

LightGBM	XGBOOST
Time taken to create model : 20-30 min	Time taken to create model: 2 hours (aprx)
Accuracy attained: 67.99%	Accuracy attained : 74.44%

Summary

Xgboost and lightGBM are very powerful and effective algorithms that can be used out of the box without much tuning.

LightGBM improves on XGBoost.

The LightGBM paper uses XGBoost as a baseline and outperforms it in training speed and the dataset sizes it can handle.

LightGBM in some cases reaches it's top accuracy in under a minute and while only reading a fraction of the whole dataset.

This goes to show the power of approximation algorithms and intelligently sampling a dataset to extract the most information as fast as possible.

References:

Xgboost

The original paper
The docs for xgboost
A blog post on how to use xgboost

LightGBM

The original paper
The docs for lightGBM
A blog post by Microsoft on lightGBM