Handling Imbalanced Datasets

By - Abhishek Periwal

About Me

- IIT Kanpur 2013 Graduate
- Currently working as Data Scientist at Flipkart
- Have worked at Paytm, Lendingkart as Data Scientist
- Majorly worked in Fintech Domain and problems like
    - Credit Risk Models
    - Credit Recovery Models
    - Product Adoption Models

What is Imbalanced Dataset ?

Scenario which has unequal distribution of classes in the datasets (number of observations of one class is significantly lower than other classes) like credit default data, fraud detection datasets

Example :

- Total customers - 284,807
- Paid Back - 284,315 (99.8%)
- Defaulted - 492 (0.172%)

If I predict all customers as non-defaulter, Accuracy of my model is 99.8% (very high …) but it fails in predicting other class

Why Imbalance is the problem ?

- Cost function of some machine learning Algorithms are usually designed to improve accuracy by reducing the error.
- Thus, they work best when the number of samples in each class are about equal
- Standard classifier algorithms like Logistic Regression have a bias towards classes which have number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored.

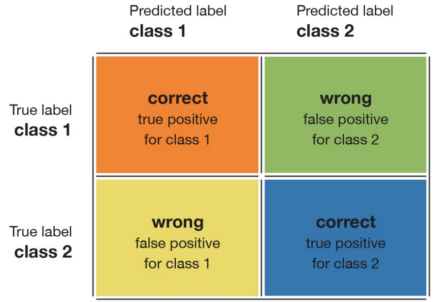
Solution!!

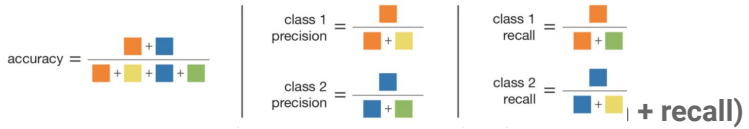
- Right Evaluation Metrics
  - Precision, Recall
  - F1 Score
  - AUROC
- Data Level approach
  - Sampling Techniques
- Better Algorithmic Techniques
  - Bagging
  - Boosting

Evaluation Metrics

Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be very misleading.

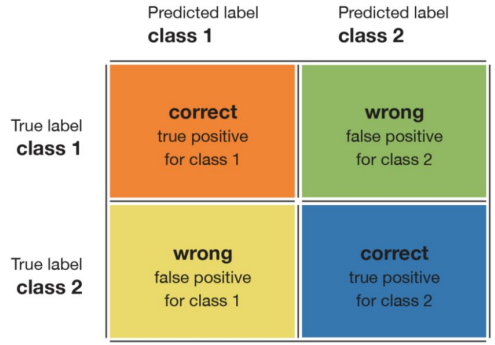
- **Confusion Matrix**





- **+ recall)**
- **ROC (AUC)**

Confusion Matrix



high recall + high precision : the class is perfectly handled by the model

low recall + high precision : the model can't detect the class well but is highly trustable when it does

high recall + low precision : the class is well detected but the model also include points of other classes in it

low recall + low precision : the class is poorly handled by the model

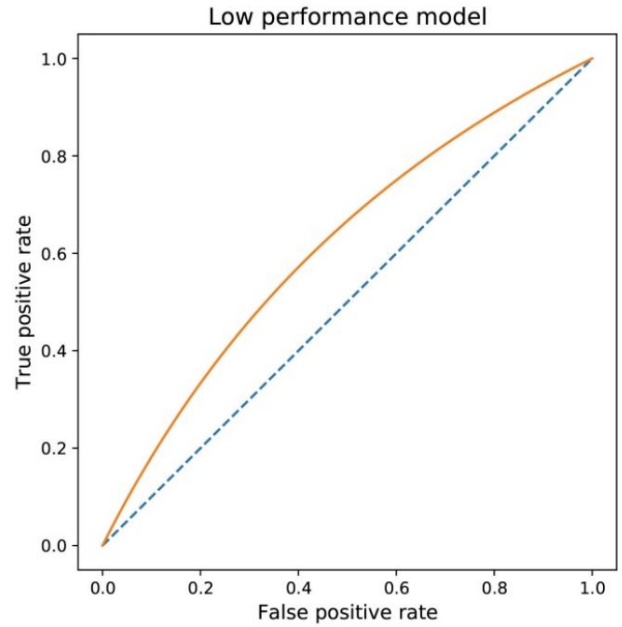
F1 Score

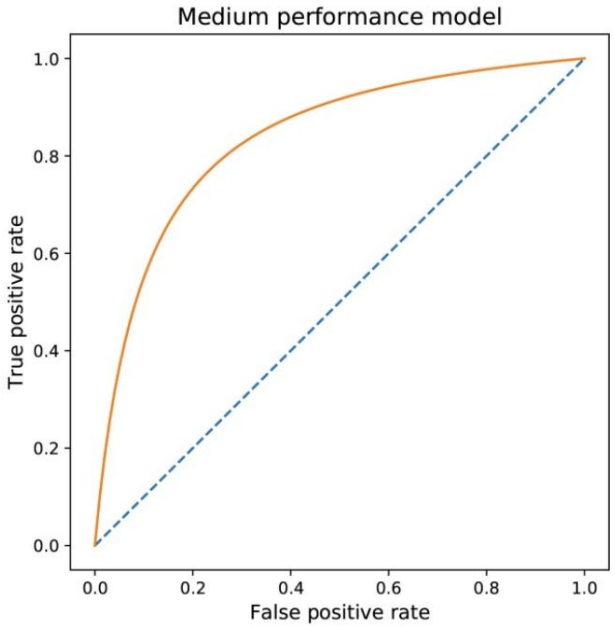
$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

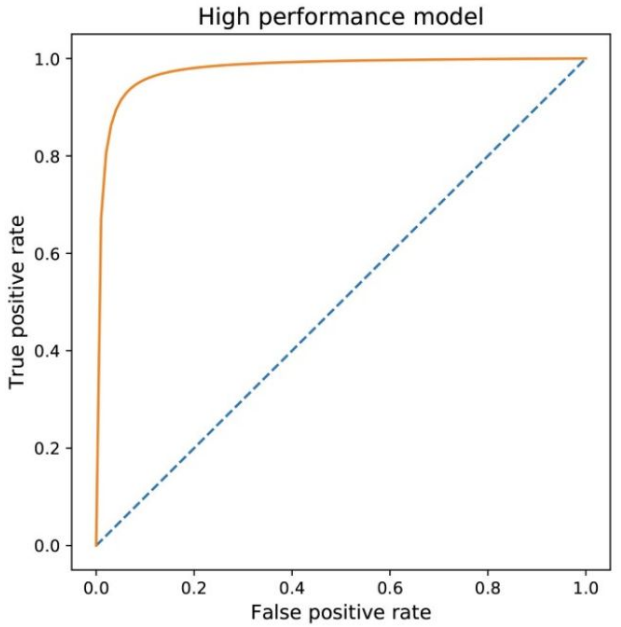
- F1 score gives equal weightage to precision and recall
- F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall AND there is an imbalanced class distribution.

ROC AUROC

ROC is a probability curve and AUC represents degree or measure of separability. For each threshold (between [0,1]) a point is generated with coordinates (x, y) → (False positive, True Positive). Then, the ROC curve is the curve described by the ensemble of points generated.







Data Approach

Resampling

- **Undersampling** consists in sampling from the majority class in order to keep only a part of these points
- **Oversampling** consists in replicating some points from the minority class in order to increase its cardinality
- **Generating Synthetic Data** consists in creating new synthetic points from the minority class to increase its cardinality

Random Under Sampling

- **What :** Balancing class distribution by randomly eliminating some data points from Majority Class
- **How :** Reduce imbalance by reducing majority class
- **Pros**
  - Improve Runtime
  - Less memory consumption
- **Cons**
  - Discard potentially useful information
  - Not a accurate representation of the population

Random Over Sampling

- **What** : Balancing class distribution by increasing the number of instances in the minority class by randomly replicating them
- **How :** Reduce imbalance by reducing majority class
- **Pros**
  - No Information Loss
  - Outperform under sampling
- **Cons**
  - Increases the chances of overfitting
  - Increase Runtime

Cluster-Based Over Sampling

- **What** : Independent Clusters for classes created using k-means clustering & oversampling in each cluster
- **Pros**
  - Solve unbalanced problem
  - Examples representing positive class differs from the Examples representing a negative class
  - Overcome challenges within class imbalance, where a class is composed of different sub clusters. And each sub cluster does not contain the same number of examples
- **Cons**
  - Possibility of over-fitting the training data

Synthetic Minority Over-sampling Technique

- **Method**
  - A subset of data is taken from the minority class as an example and then new synthetic similar instances are created.
  - These synthetic instances are then added to the original dataset.
  - The new dataset is used as a sample to train the classification models.
- **Pros**
  - Mitigates the problem of overfitting
  - No loss of useful information
- **Cons**
  - Does not take into consideration neighboring examples from other classes. This can result in increase in overlapping of classes and can introduce additional noise
  - Not very effective for high dimensional data

Algorithmic Techniques

Algorithmic Techniques

1. Bagging
2. Boosting
    a. Adaptive Boosting
    b. Gradient Tree Boosting
    c. XGBoost
    d. Lightgbm

Balanced Bagging Classifier

- **What :** It is special case of bagging. It allows the resampling of each subset of the dataset before training each estimator of the ensemble.

  (`sampling_strategy` & `replacement`)
- **Pros**
  - Improves stability & accuracy
  - Reduces variance, Overcomes overfitting
  - Improved misclassification rate of the bagged classifier
  - In noisy data environments bagging outperforms boosting
- **Cons**
  - Bagging works only if the base classifiers are not bad to begin with. Bagging bad classifiers can further degrade performance

Random Forest

Normal Random forest also gives biased results to majority class. As it is constructed to minimize the overall error rate, it will tend to focus more on the prediction accuracy of the majority class, which often results in poor accuracy for the minority class.

To handle this problem

1. Balanced Random Forest
2. Weighted Random Forest - `class_weight` parameter can be used

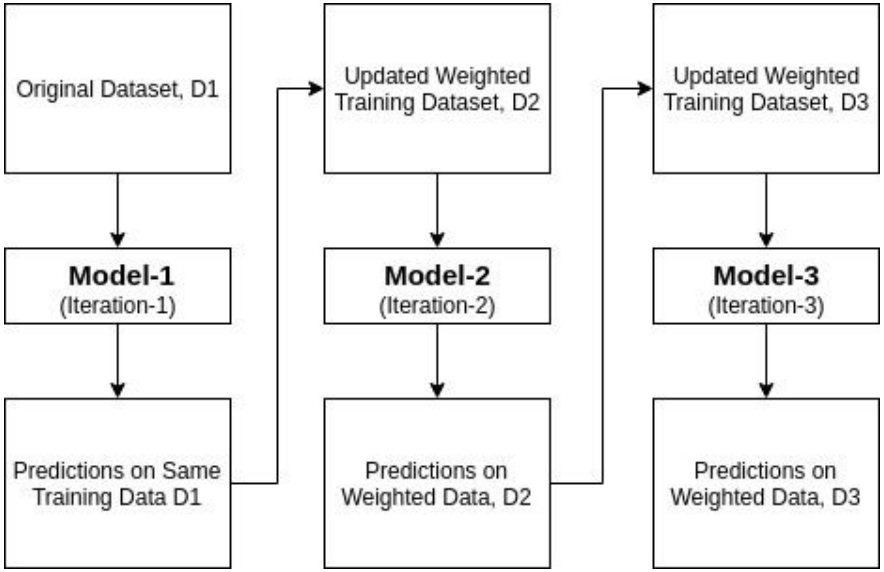
Boosting

- **What :** Boosting is a family of algorithms which ensemble weak learners to create a strong learner. The idea of boosting is to train weak learners sequentially, each trying to correct its predecessor.

Types of Boosting :

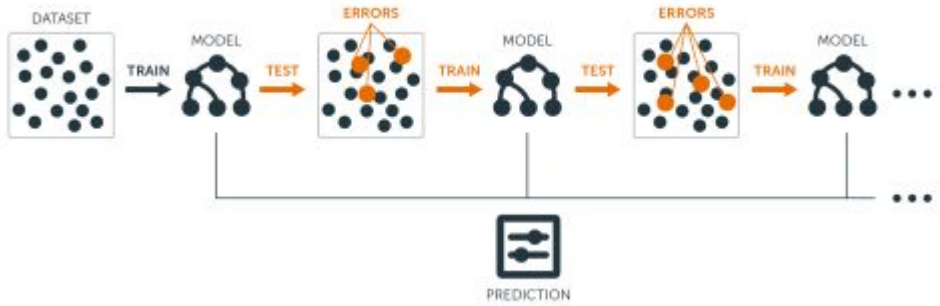
- **Adaboost**
- **Gradient Boosting**
  - XGboost

Adaboost



➔ **How :** If minority class is misclassified the observation points will get higher weightage in subsequent model

Gradient Boosting



- Works by sequentially adding predictors to an ensemble, each one correcting its predecessor.
- Gradient Boosting method tries to fit the new predictor to the residual errors made by the previous predictor.

➔ **How :** Gradient Boosting is a sequential process and thus every time it makes an incorrect prediction, it focuses more on that incorrectly predicted data point

➔ **Cons**

- Very slow in implementation
- Prone to overfitting

XGBoost

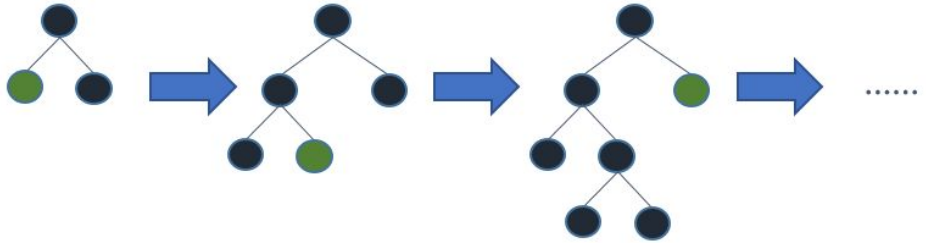
XGBoost is an implementation of gradient boosted decision trees designed for speed and performance

Xgboost also has parameter for sampling too which is useful for imbalanced dataset

`scale_pos_weight` - Control the balance of positive and negative weights

- **Pros**
  - Fast & Efficient
  - Can check Variable Importance.
  - Do not require feature engineering (missing values imputation, scaling and normalization)
- **Cons**
  - Does not handle categorical variables
  - Leads to overfitting if hyperparameters are not tuned properly

LightGBM



Decision tree based algorithms, it splits the tree leaf wise with the best fit

`scale_pos_weight` - weight of labels with positive class

`pos_bagging_fraction` - will randomly sample positive samples in bagging

`neg_bagging_fraction` - will randomly sample positive samples in bagging

➔ **Pros**

- Fast & Efficient
- Low Memory usages
- Better Accuracy
- Handel Categorical variables

Key Takeaways

- Accuracy is not the right measure in case of imbalanced datasets
- AUROC, F1 Score, Precision, Recall can be used to evaluate model
- Resampling Techniques improve model performance
- Bagging and Boosting improve model performance better than Resampling
- Combining Bagging & Boosting with Resampling techniques further improves the model

References

- https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/
- https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28
- https://towardsdatascience.com/methods-for-dealing-with-imbalanced-data-5b761be45a18
- https://pdfs.semanticscholar.org/95df/dc02010b9c390878729f459893c2a5c0898f.pdf
- https://pdfs.semanticscholar.org/fb76/4c7bdaa19550e520f7f0eeb8003c11b1b0fb.pdf

Q & A ?

Thank You