Robust Machine Learning with Imbalanced Data: Expanding Mixup

AIMS Sénégal
Institut Africain des Sciences Mathématiques

Andriamarolahy Rabetokotany

African Institute for Mathematical Sciences

- Supervisor -
Dr. Vukosi Marivate

August 6, 2019

Outline

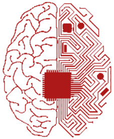
Outline

Introduction

Generality

- More and more data are generated:
  - social media
  - mobile, sensors (IOT)
  - Large Hadron Collider (LHC)
  - ...
- Imbalanced data: unequal classes.
- Problem: Machine Learning algorithm will be biased towards majority classes.

IMBALANCED LEARNING



Introduction

Goal

- Find a more robust technique for approaching this problem.
- Compare the new technique with the related works.

Outline

Definitions

Machine Learning

- A subfield of computer science.
- Give computers the ability to learn and make predictions on data.
- Classified into 2 groups:
  - Supervised
  - Unsupervised

Definitions

Imbalanced Data

- A data set that exhibits a significant unequal distribution between its classes.
  - Majority classes: the dominant groups in the data (negative classes).
  - Minority classes: the underrepresented groups in the data (positive classes).

Definitions

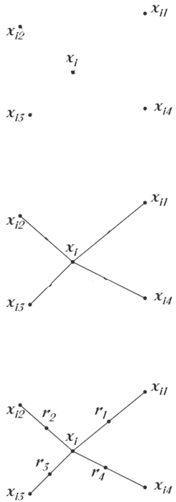
Imbalanced Learning

- The process of learning from imbalanced data
- Two methods to approach it:
  - Data level methods.
  - Algorithm level methods.

Outline

SMOTE

- $SMOTE$: Synthetic Minority Oversampling Technique:
  - Select x_i in minority class.
  - Select x_{i1}, x_{i2}, x_{i3} and x_{i4} randomly.
  - Take the difference between x_i and its nearest neighbour.
  - Multiply this difference by a random number between 0 and 1.
  - Add it to the feature vector x_i.
  - We get r_1, r_2, r_3 and r_4.
- Advantages:
  - simple: easy, implemented in python,
  - avoid overfitting.



mixup

- Empirical Risk Minimization
  - Minimize (1) over P
  - Problem: P is unknown.
  - Solution: ERM: $(x_i, y_i)_{i=1}^n$, where $(x_i, y_i) \sim P_{emp}$
  - (1) $\Rightarrow$ (3).
- Drawbacks:
  - memorize instead of generalize
  - poor performance outside $(x_i, y_i)_{i=1}^n$
- Solution: VRM

$$R(f) = \int \ell(f(x), y) dP(x, y) \quad (1)$$

$$P_{emp}(x, y) = \frac{1}{n} \sum_{i=1}^{n} \delta(x = x_i, y = y_i) \quad (2)$$

$$R_{emp}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) \quad (3)$$

mixup

- Vicinal Risk minimization
  - Use (4) defined from $\Omega(x_i) = x : d(x, x_i) \leq r_{x_i}$
  - $\Omega(x_i)$: the vicinity of the point x_i.
  - ν is a vicinity distribution that measures the probability of finding the virtual feature-target pair $(\tilde{x}, \tilde{y})$ in the vicinity of the training feature-target pair (x_i, y_i).
  - (3) $\Rightarrow (5)$
- Advantage: Improves the generalization of the training data.
- Drawback: requires big understanding of the distribution of the dataset.
- Solution: mixup

$$P_{est}(\tilde{x}, \tilde{y}) = \frac{1}{n} \sum_{i=1}^{n} \nu(\tilde{x}, \tilde{y} | x_i, y_i) \quad (4)$$

$$R_{vic}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\tilde{x}_i), \tilde{y}_i) \quad (5)$$

mixup

- ν can be chosen arbitrary
- use *mixup vicinal distribution* which is defined in (6)
- $\lambda \sim Beta(\alpha, \alpha)$, for $\alpha \in (0, \infty)$.
- Improves the efficiency of the algolirthm in term of adversarial examples.

$$\nu(\tilde{x}, \tilde{y}|x_i, y_i) = \frac{1}{n}\sum_{j}^{n} \mathbb{E}_\lambda[\delta(\tilde{x} = \lambda.x_i + (1-\lambda).x_j, \tilde{y} = \lambda.y_i + (1-\lambda).y_j)] \quad (6)$$

Outline

Proposed method

- Combine `the advantages` of `SMOTE` and `mixup`.

Methods

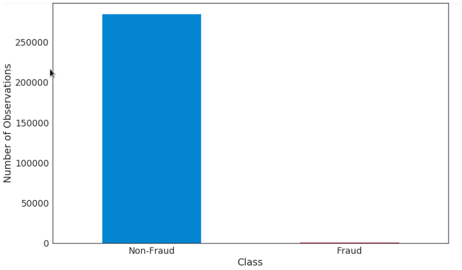
Data samplers and Classifiers

- Data samplers
  - $SMOTE$
    - $random - state = 42$
    - $ratio = 1 : 25000$
  - mixup
    - $alpha = 0.1$
  - mixup $+SMOTE$
    - $alpha = 0.1$
    - $random - state = 42$
    - $ratio = 1 : 25000$
- Classifiers
  - Random Forest
    - default parameters
  - Gradient Boost
    - $'n\_estimators' : 500,$
    - $'max\_depth' : 3,$
    - $'subsample' : 0.5,$
    - $'learning\_rate' : 0.01,$
    - $'min\_samples\_leaf' : 1,$
    - $'random\_state' : 3$

Methods

Dataset

- European credit card transactions.
- $284,807$ rows: $284,315$ normal and 492 fraudulant transactions,
- 31 columns: $V1, V2, \ldots, V28, Time, Amount$ and $Class$
- features: $V1, V2, \ldots, V28, Time, Amount$
- targets: $Class$: normal , fraudulant



Methods

Evaluation metrics

Confusion matrix

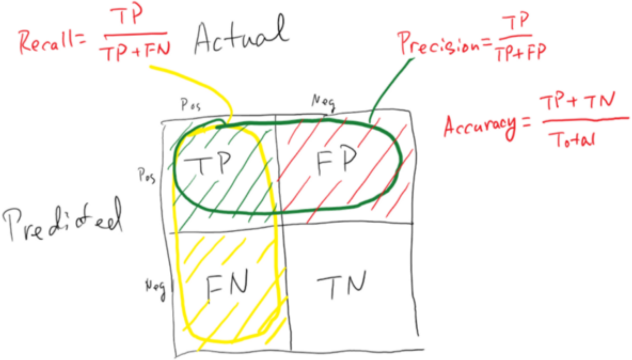
- a performance measurment for classification.
  - TP: predicted Positive and it's True.
  - TN: predicted Negative and it's True.
  - FP (*type I error*): predicted Positive and it's False.
  - FN (*type II error*): predicted Negative and it's False.

Predicted Values \ Actual Values	Positive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

Methods

Evaluation metrics

- Accuracy:
  - the percentage of correct predictions.
- Precision:
  - the fraction of positive predictions that are correct.
- Recall:
  - the fraction of the truly positive instances that the classifier recognizes.

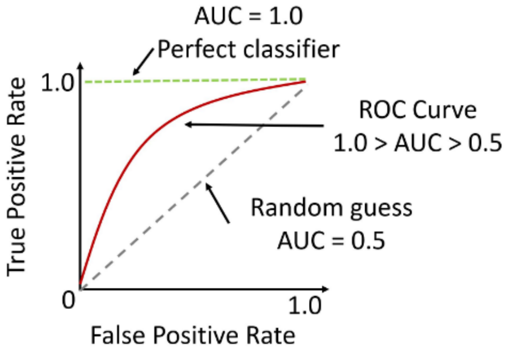


Methods

Evaluation metrics

AUC - ROC

- a performance measurement for classification problem at various thresholds settings.
- Receiver Operating Characteristic (ROC) curve visualizes a classifier's performance.
- AUC represents degree or measure of separability.
- It tells how much model is capable of distinguishing between classes.
- Higher the AUC, better the model is.



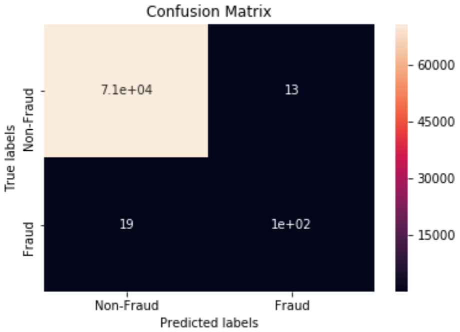
ROC curve with values of AUC for balanced two-class problem

Outline

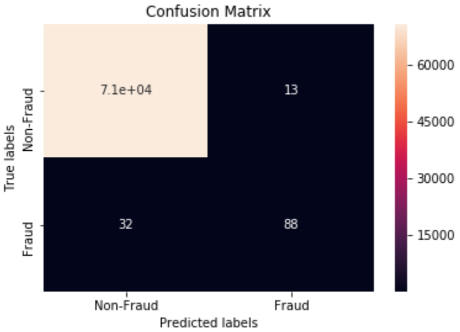
Experimental results

Confusion matrices

Random Forest



(a) $SMOTE$

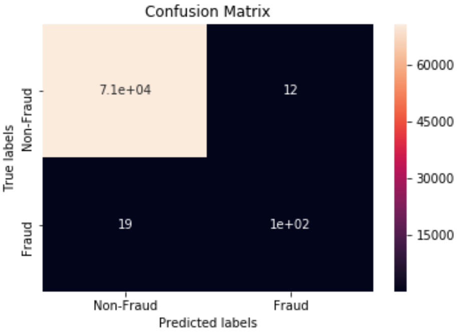


(b) mixup

Experimental results

Confusion matrices

Random Forest

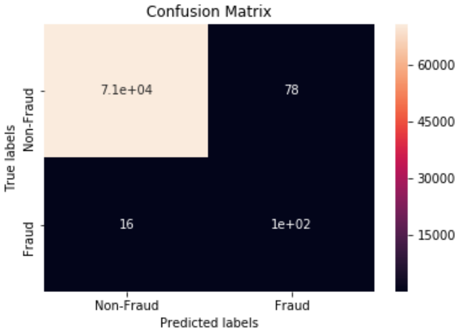


(c) mixup + $SMOTE$

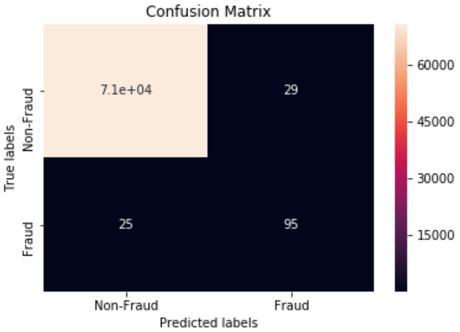
Experimental results

Confusion matrices

Gradient Boost



(d) $SMOTE$

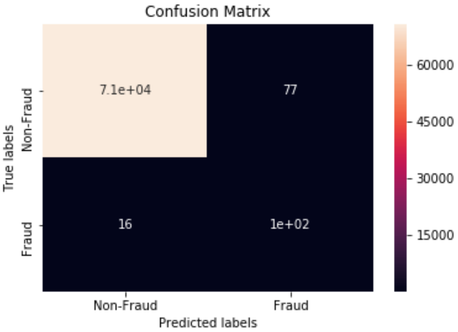


(e) mixup

Experimental results

Confusion matrices

Gradient Boost



(f) mixup + $SMOTE$

Experimental results

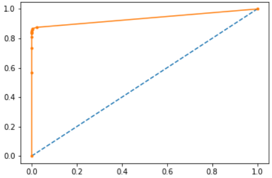
Accuracies

	$SMOTE$	mixup	mixup + $SMOTE$
RF	0.999508	0.999157	0.999438
GB	0.9986798	0.999073	0.996854

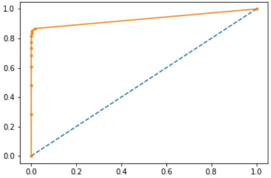
Experimental results

AUC-ROC

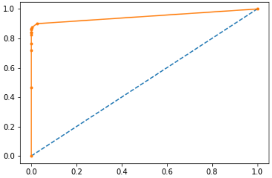
Random Forest



(g) $SMOTE$: 0.944



(h) mixup: 0.932

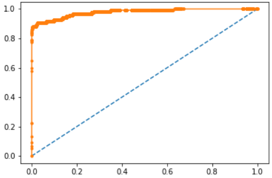


(i) mixup + $SMOTE$: 0.948

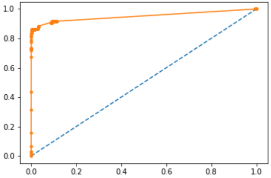
Experimental results

AUC-ROC

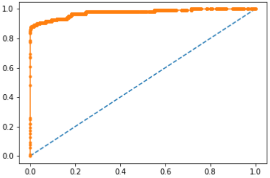
Gradient Boost



(j) $SMOTE$: 0.979



(k) mixup: 0.951



(l) mixup + $SMOTE$: 0.976

Outline

Conclusion

- Imbalanced data is one of the big issues in Machine Learning.
- $SMOTE$ is simple and efficient.
- mixup improved the efficiency towards adversarial examples.
- mixup + $SMOTE$ exhibits prominent results.
- Future works:
  - Tune the best values of hyperparameters.
  - study mixup + $SMOTE$ with multiclass problems.

THANK YOU!!!