
wroc.ai

Imbalanced data

Wroc.ai [0]

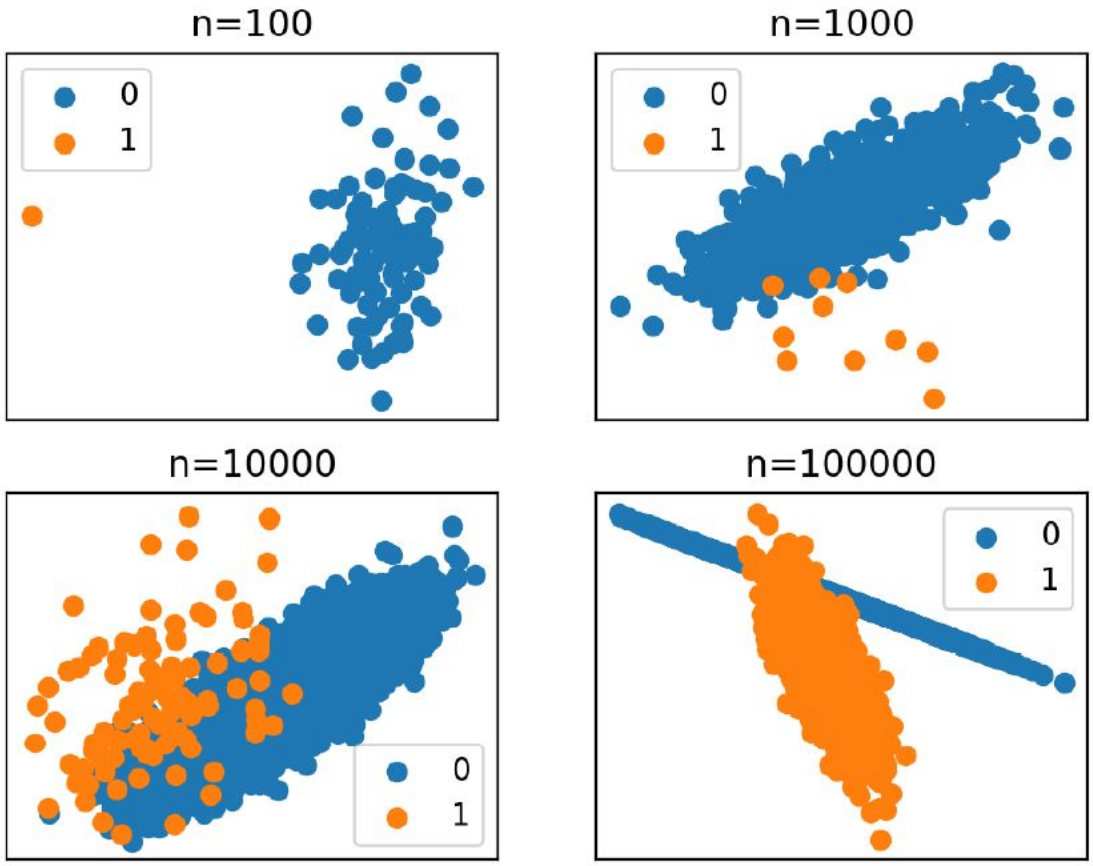
Agenda

- problem definition, intuition and challenges
- how to evaluate models
- how to split dataset
- resampling: SMOTE, Tomek-Links
- Example - Ecoli data set
- Further reading
- Q&A

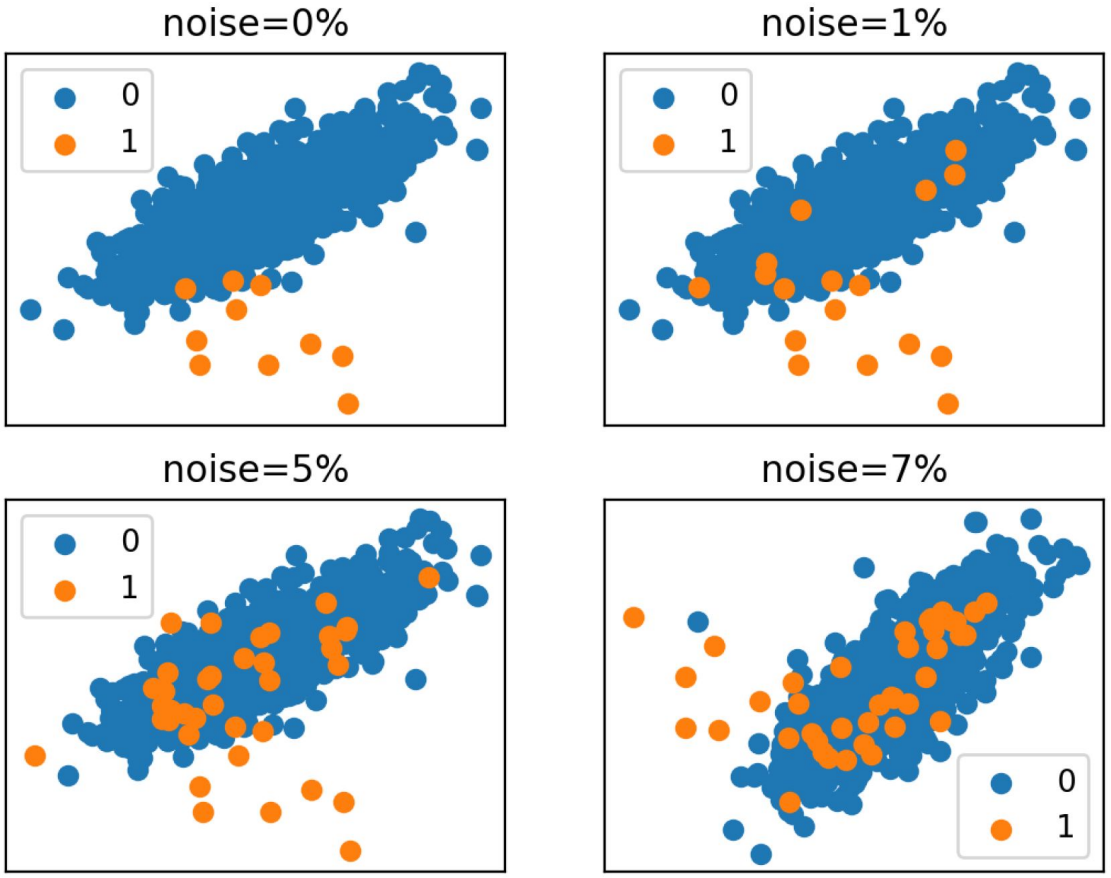
Overview

- unequal distribution of classes in the dataset
- unequal misclassification costs
- minority class often more important
- vanilla algorithms designed with no bias in mind

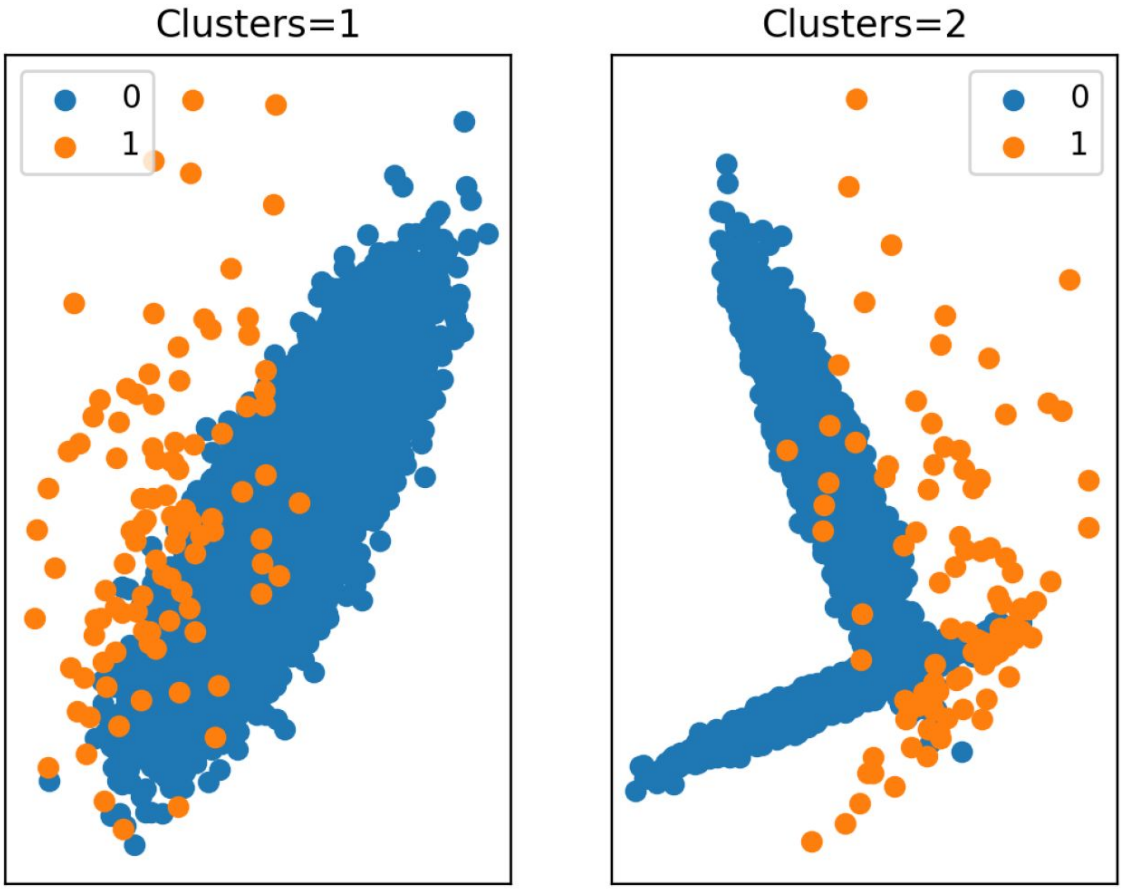
Skewness



Label noise



Distribution



Evaluation metrics

Threshold metrics

- Accuracy
- Precision
- Recall
- F-measure

Ranking metrics

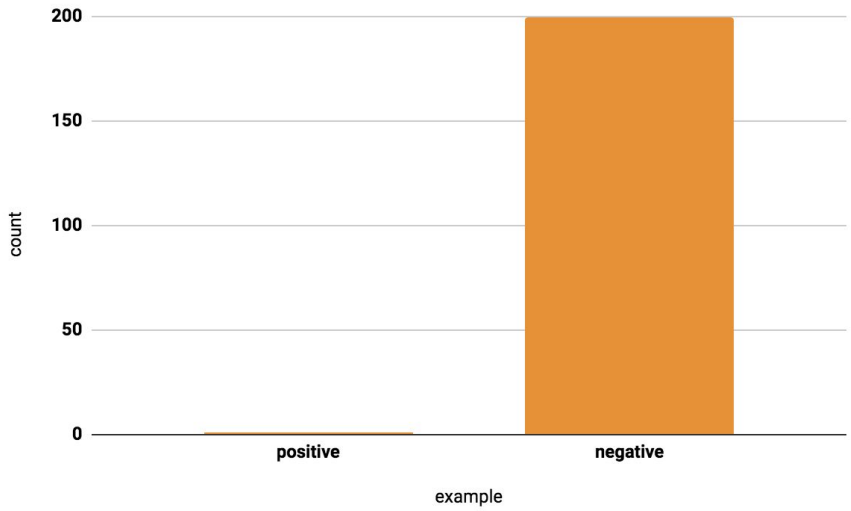
- ROC
- Precision-Recall
- AUC

Probabilistic metrics

- log loss / cross-entropy
- brier score

Threshold metrics - accuracy fail

$$Accuracy = \frac{Correct}{Total}$$



- model of constant negative answer will get a score of nearly 100%
- fails on problems with skewed distribution

Threshold metrics

$$Precision = TP/(TP + FP)$$

$$Recall = TP/(TP + FN)$$

$$F1 = \frac{2 * Precision * Recall}{(Precision + Recall)}$$

- how to deal with multiclass?
- how to deal with imbalanced data?
- assumes train/test class distribution won't change on prod

Threshold metrics - modes

Micro

- uses global number of TP, FN, FP

$$F1_{class1+class2+...+classN}$$

Macro

- bigger penalisation on minority classes
- calculates scores separately for each class

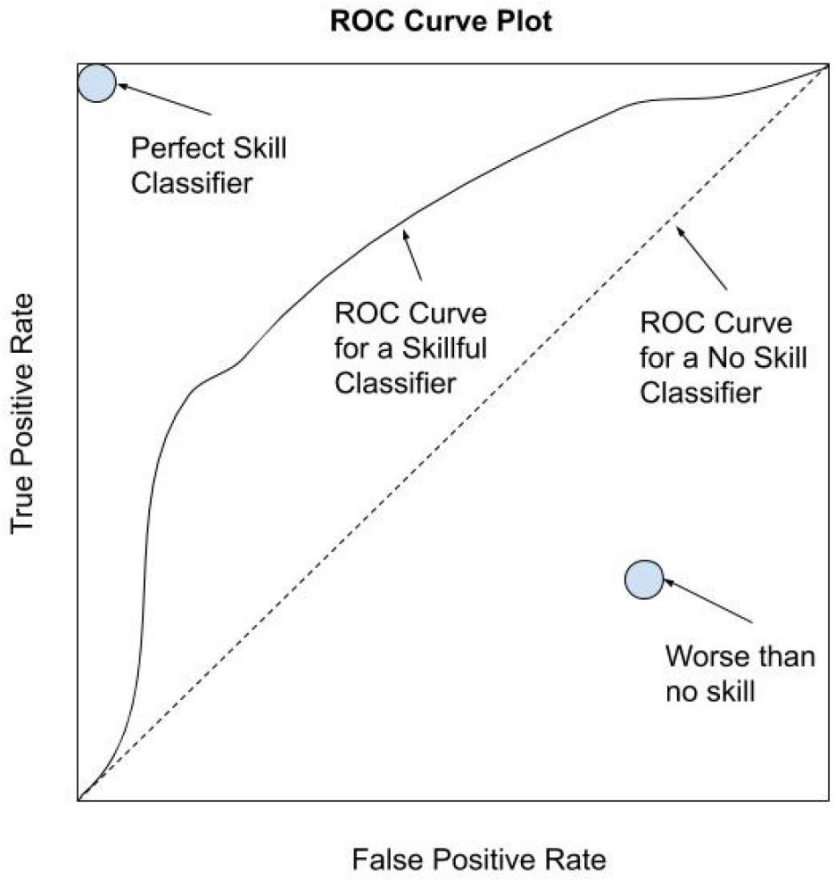
$$F1_{class1} + F1_{class2} + \ldots + F1_{classN}$$

Weighted

- weights depend on the number of label for each class
- favours majority class

$$\sum_{n=1}^{N} (F1_n * w_n)$$

Ranking metrics - ROC

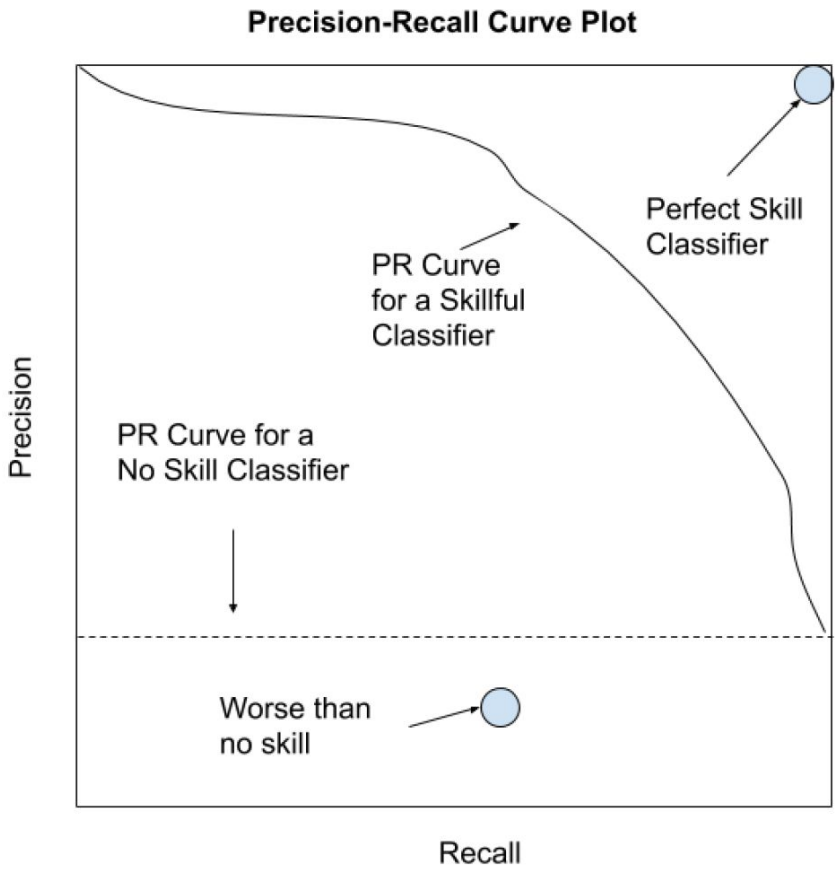


$$TP_{rate} = \frac{TP}{TP+FN}$$

$$FP_{rate} = \frac{FP}{FP+TN}$$

- no assumptions about class balance
- can be too optimistic under a severe class imbalance
- to compare classifiers calculate Area Under Curve (AUC)
- most *sklearn* models have *predict_proba()*

Raking metrics - Precision-Recall Curve

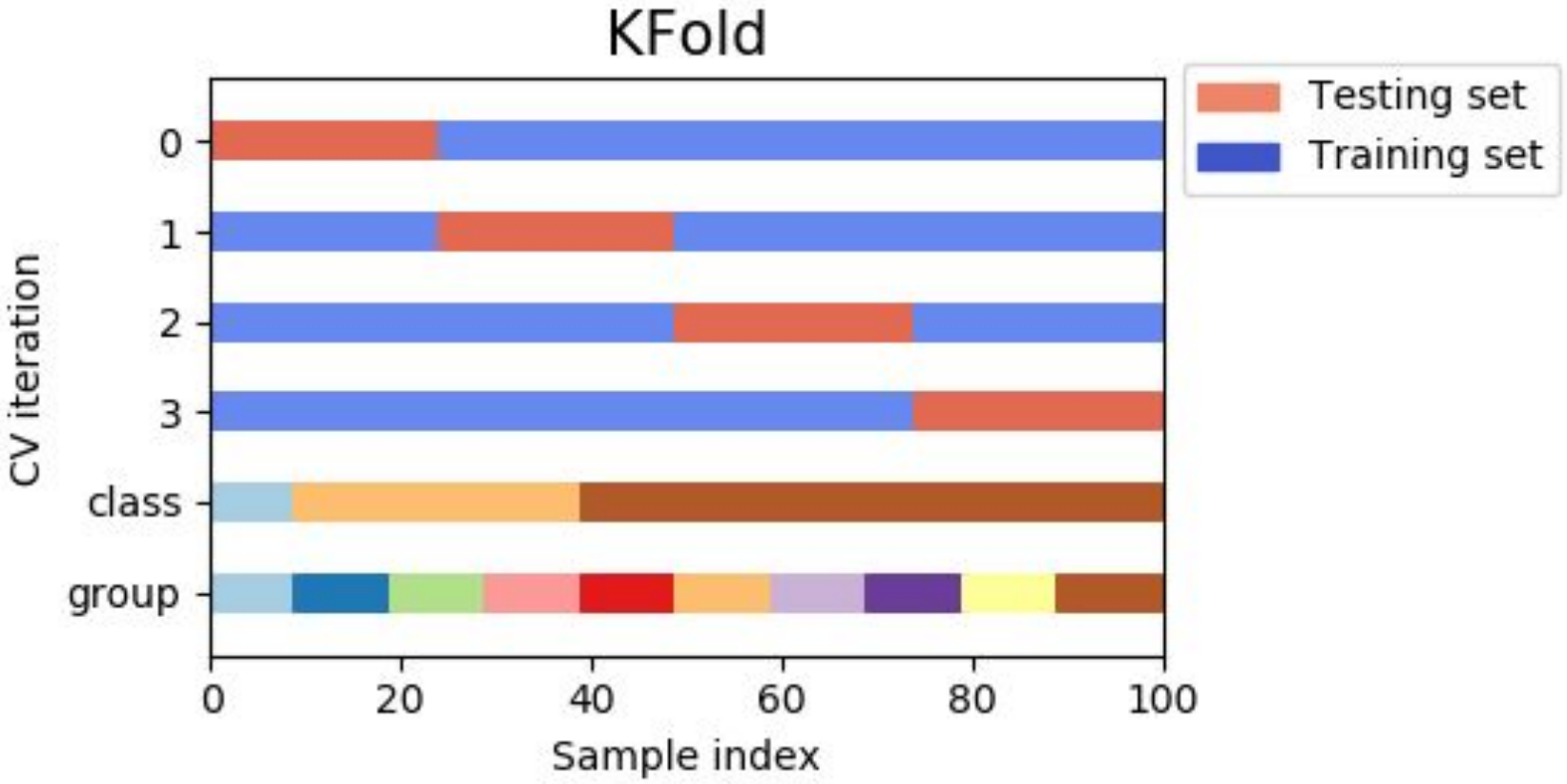


$$Precision = TP/(TP + FP)$$

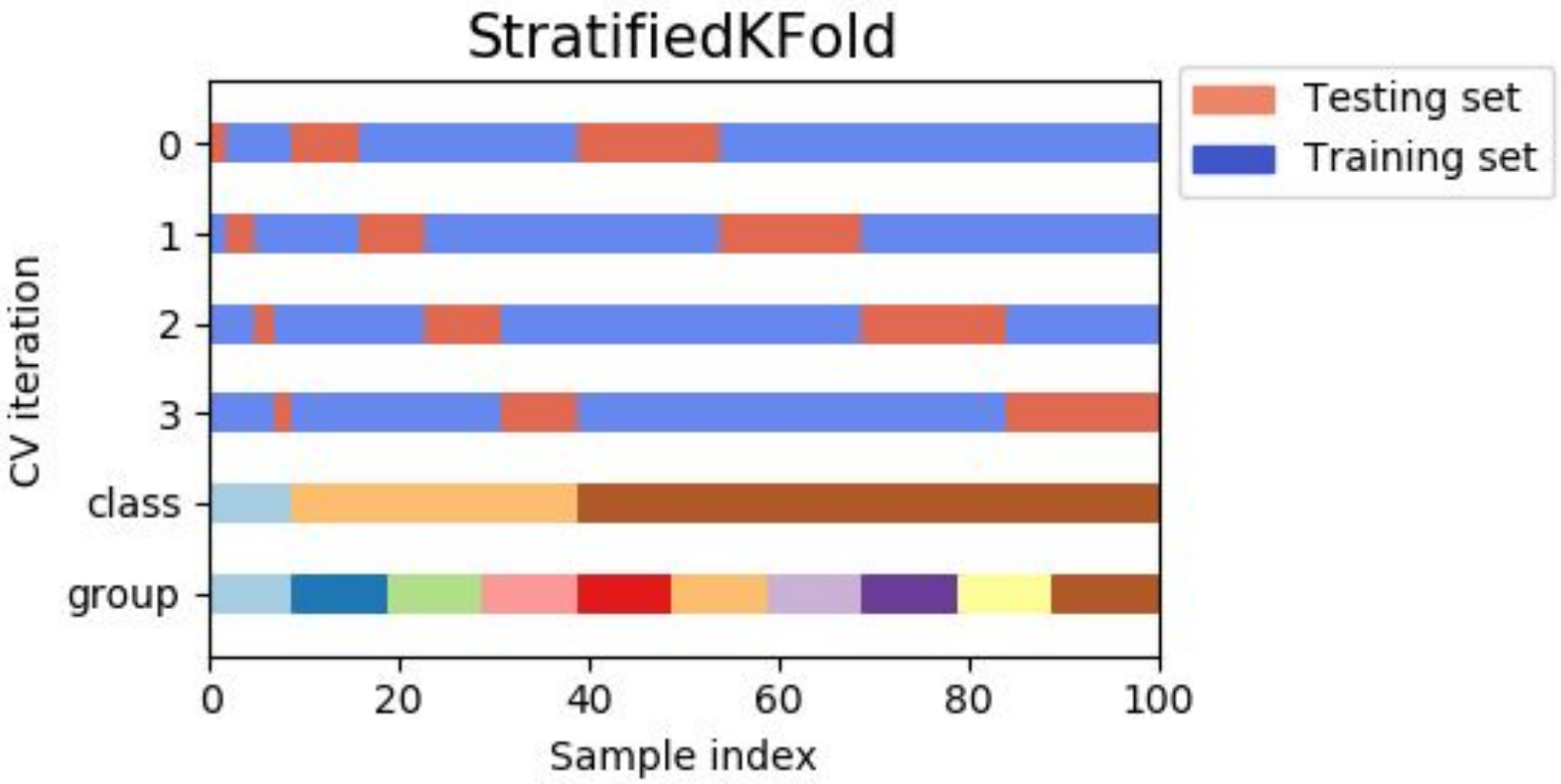
$$Recall = TP/(TP + FN)$$

- to compare classifiers calculate Area Under Curve (AUC)

How to split?

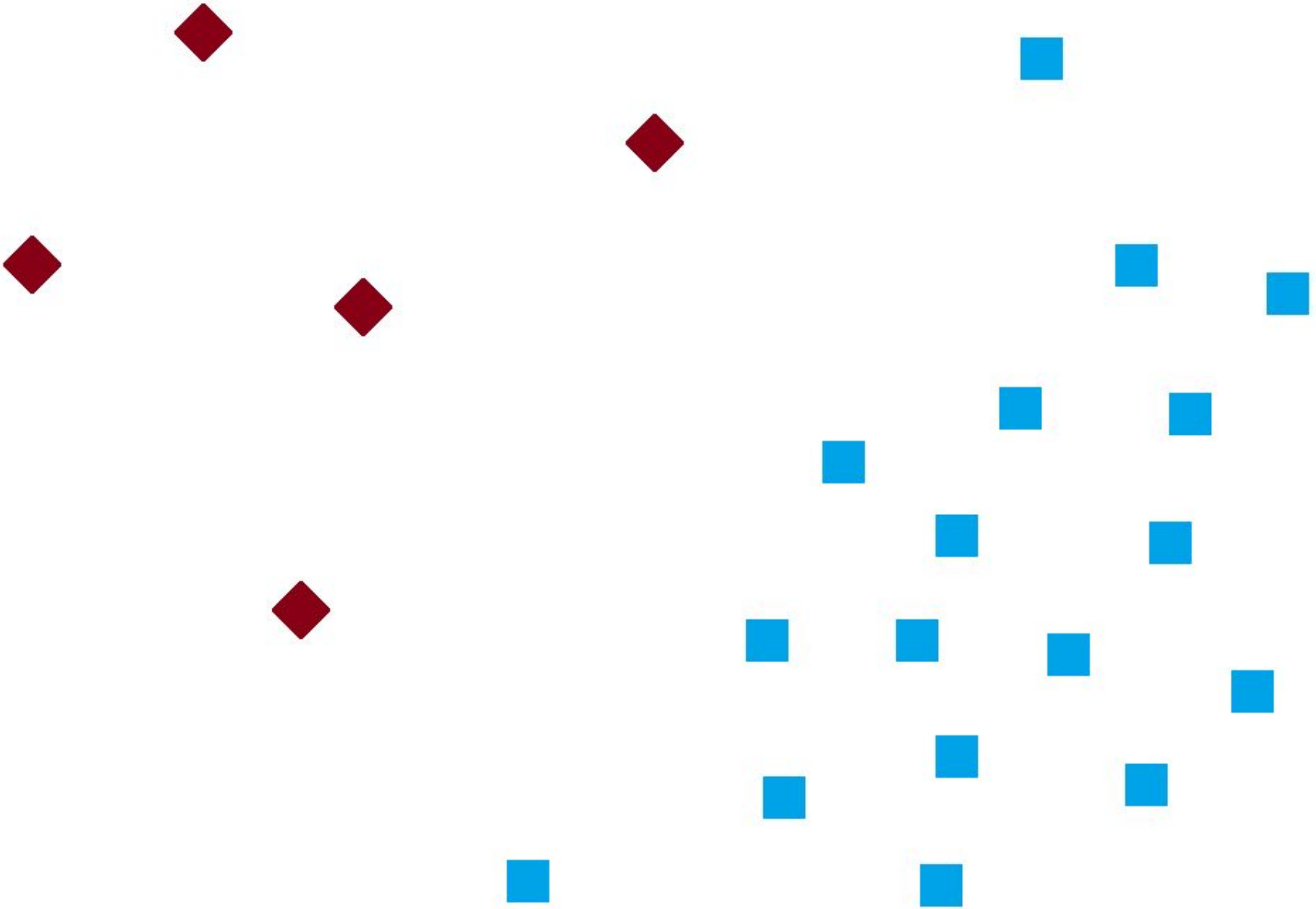

KFold
Testing set
Training set
CV iteration
0
1
2
3
class
group
0
20
40
60
80
100
Sample index

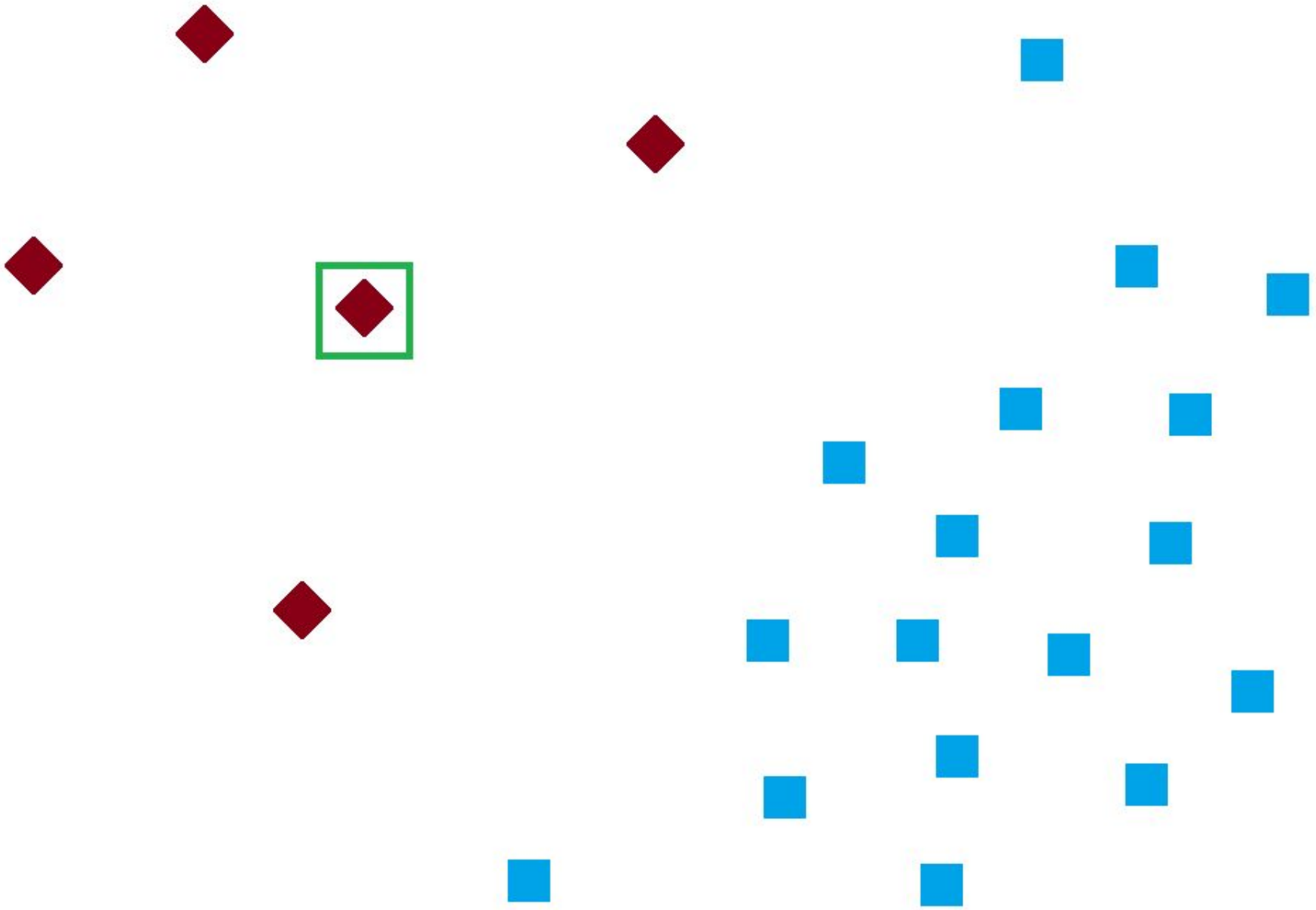
How to split?

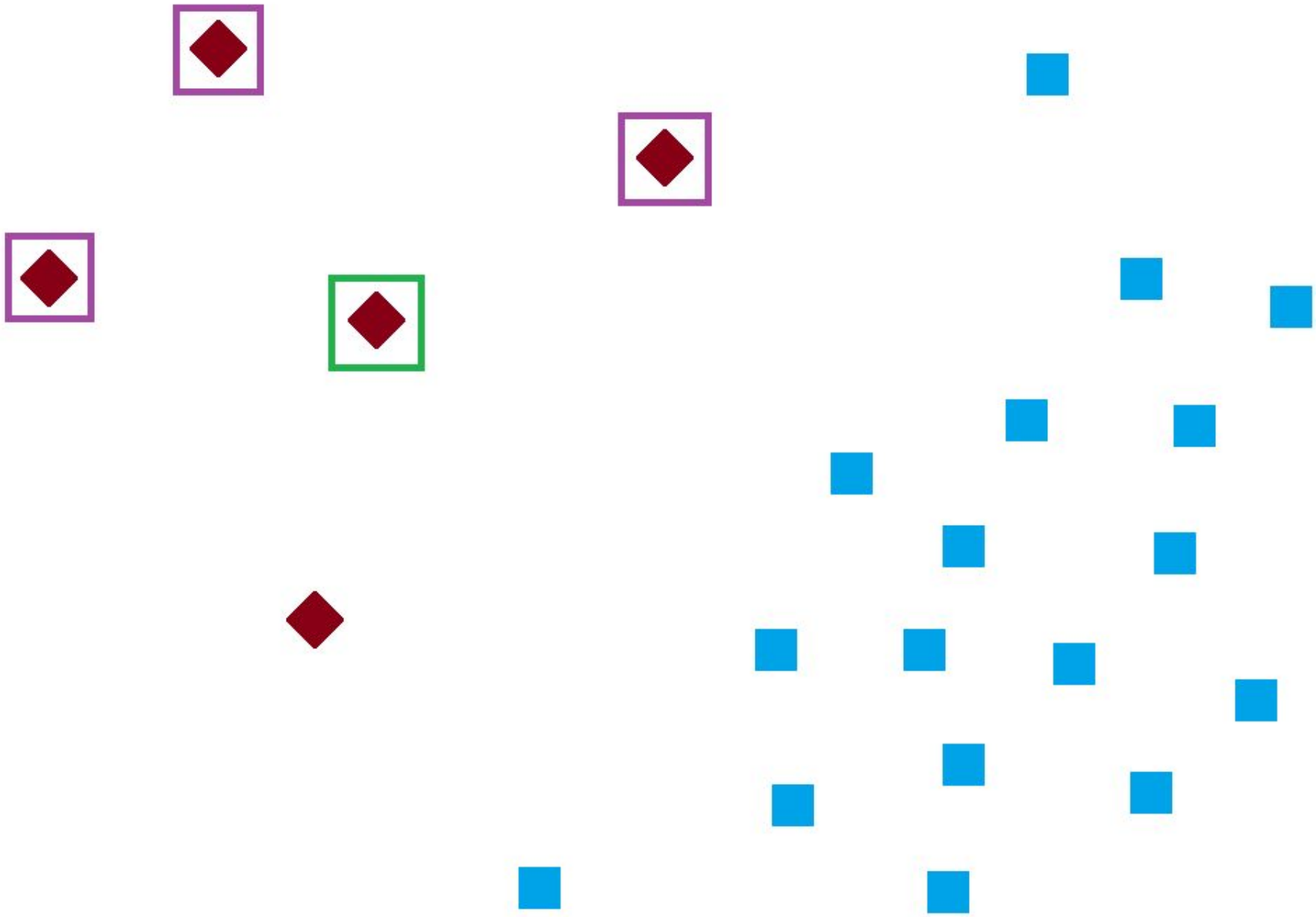

StratifiedKFold
Testing set
Training set
CV iteration
0
1
2
3
class
group
0
20
40
60
80
100
Sample index

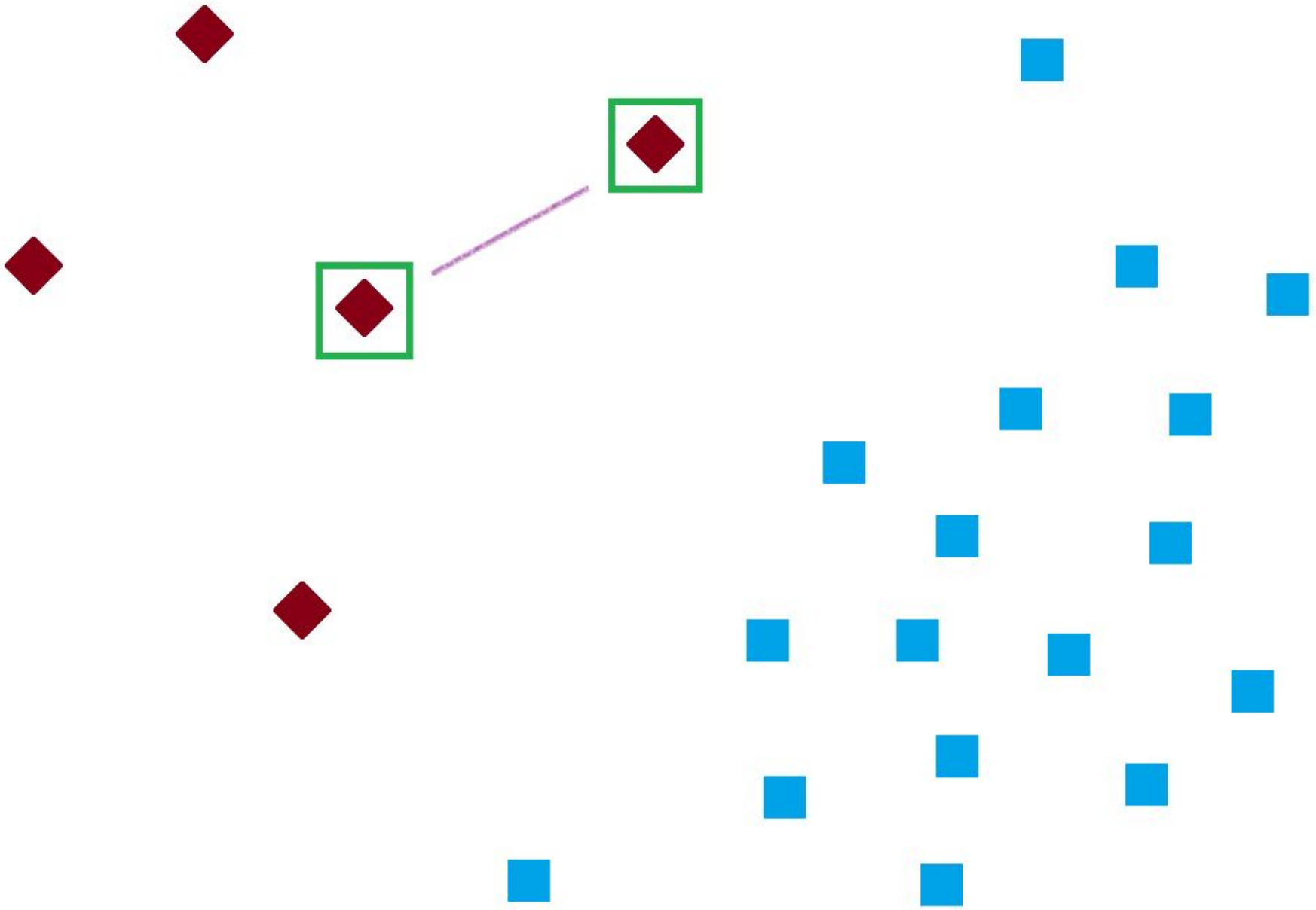
SMOTE

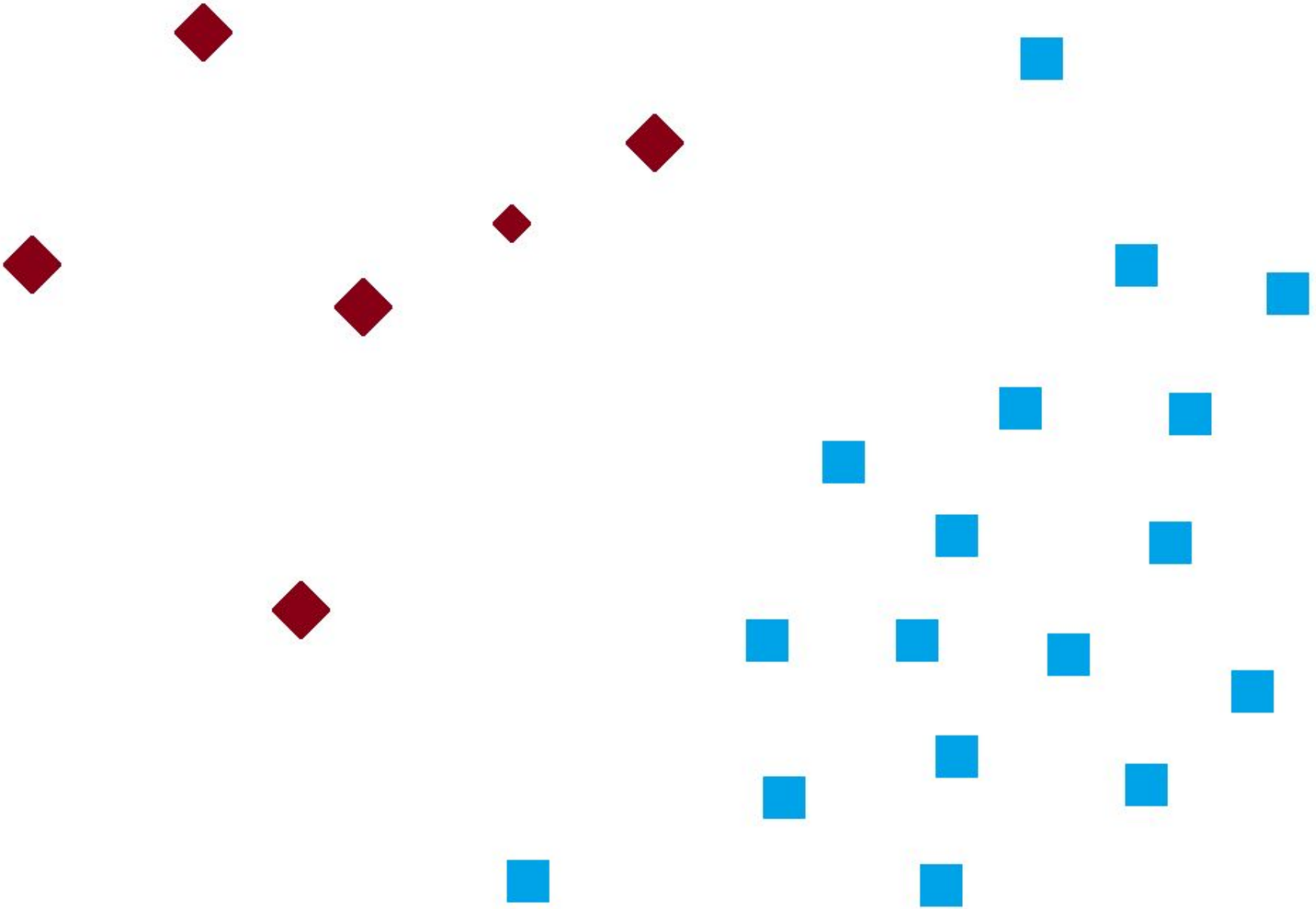
synthetic minority over-sampling technique

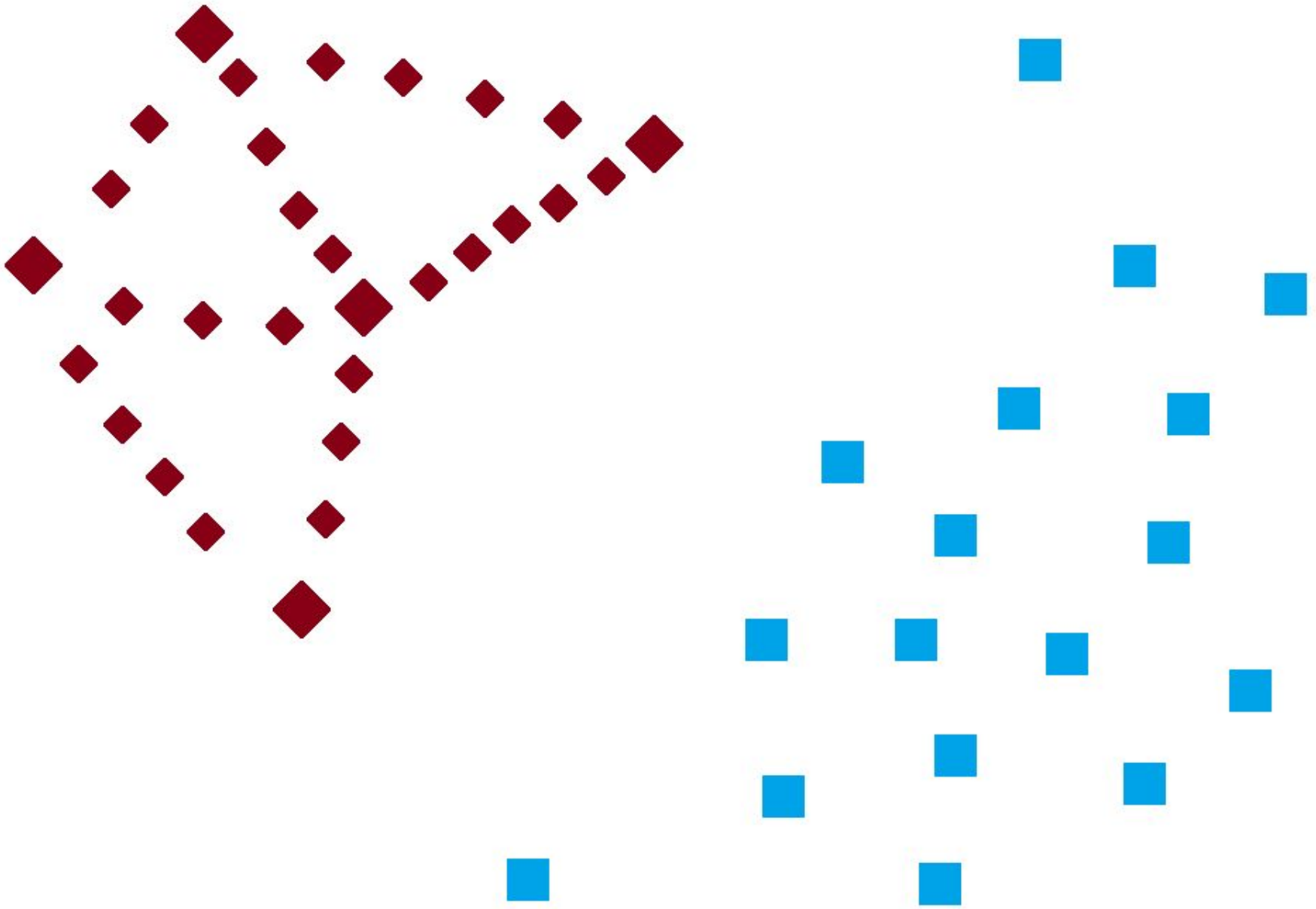












Tomek-Links

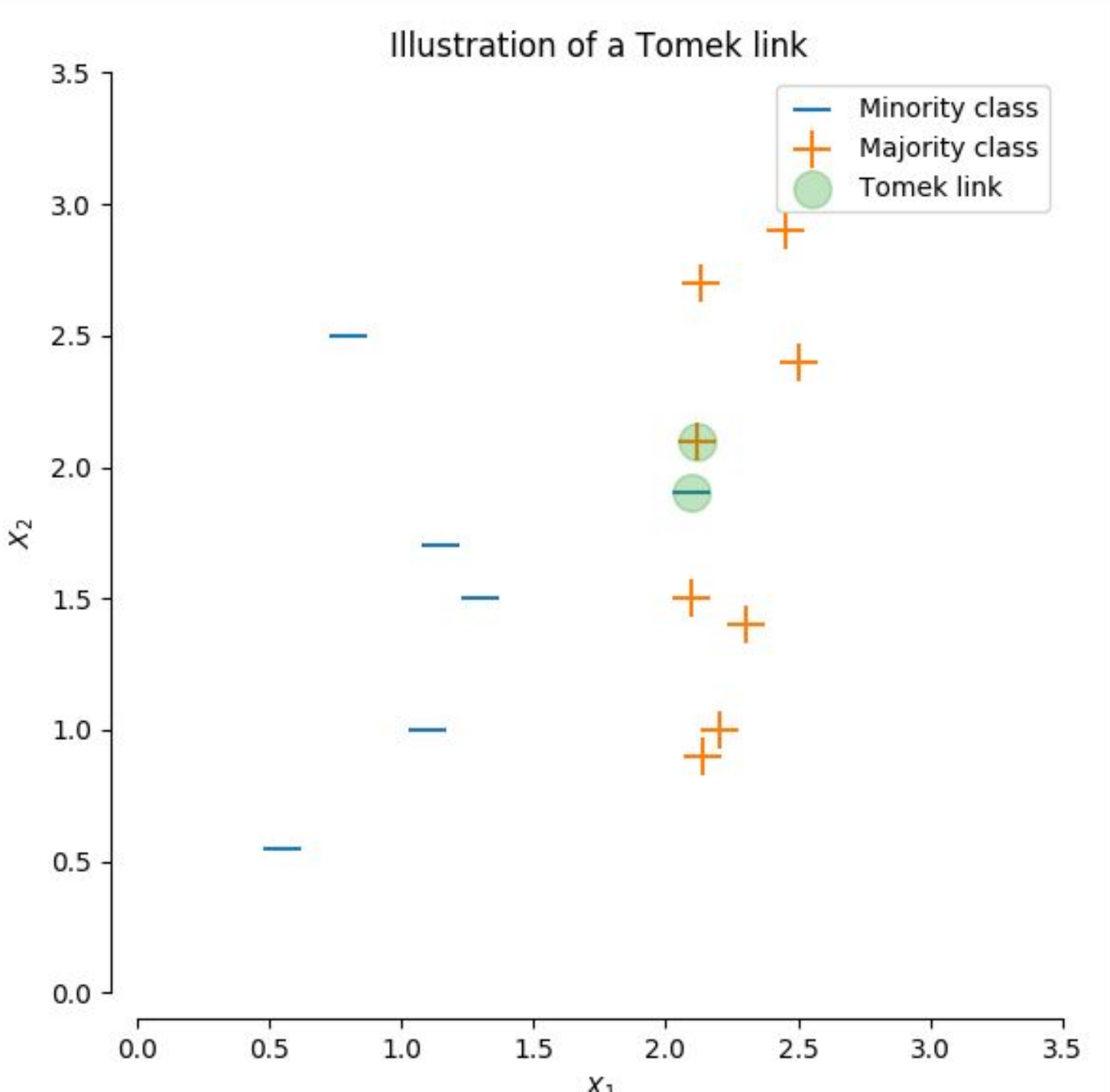
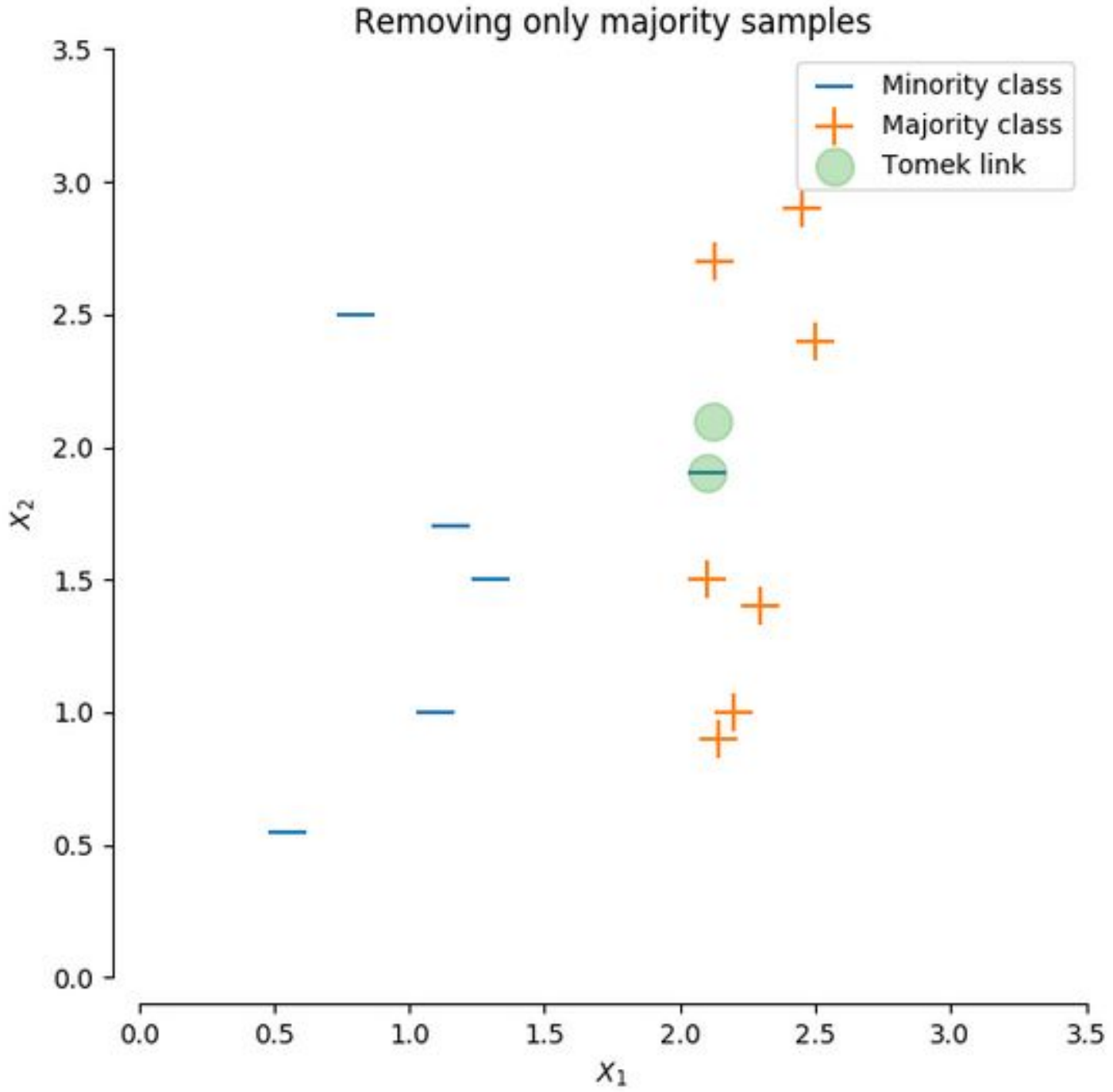
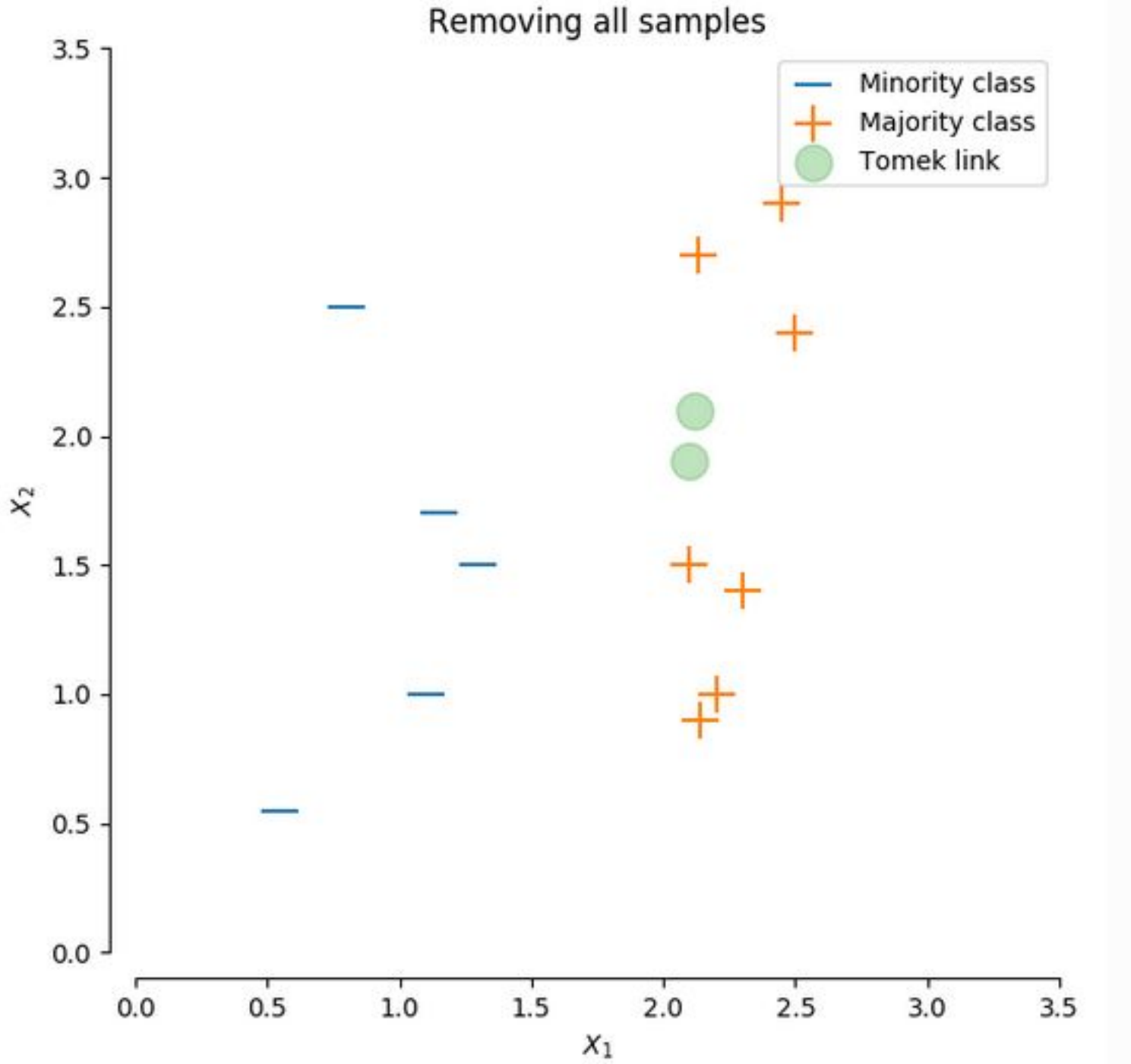


Illustration of a Tomek link
Minority class
Majority class
Tomek link
X_1
X_2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5


Removing only majority samples
Minority class
Majority class
Tomek link
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
X_2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
X_1


Removing all samples
Minority class
Majority class
Tomek link
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
X_2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
X_1

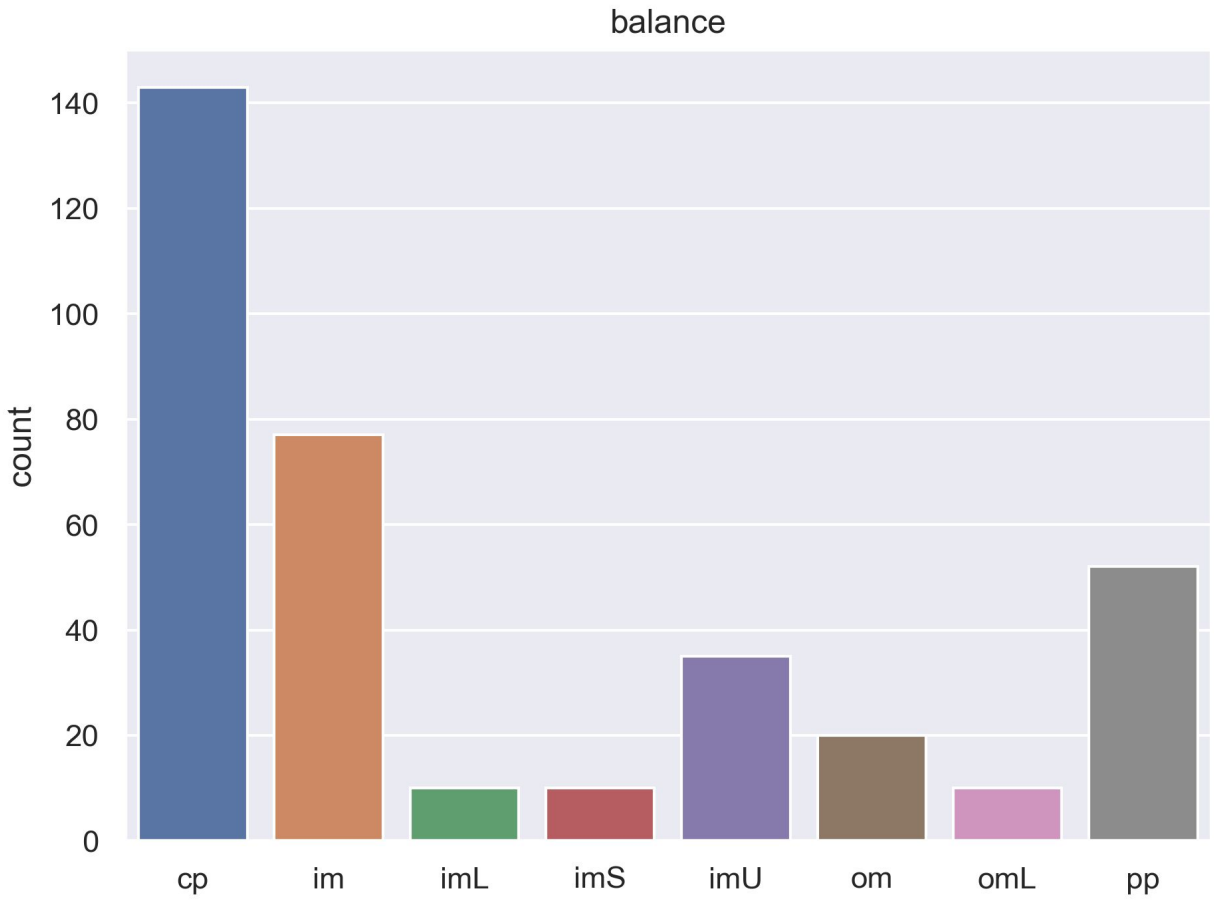
Evaluation

- We resample only training set
- We evaluate metrics only on stratified sample
- There's dedicated *Pipeline* in *imblearn,* which follow *sklearn* API

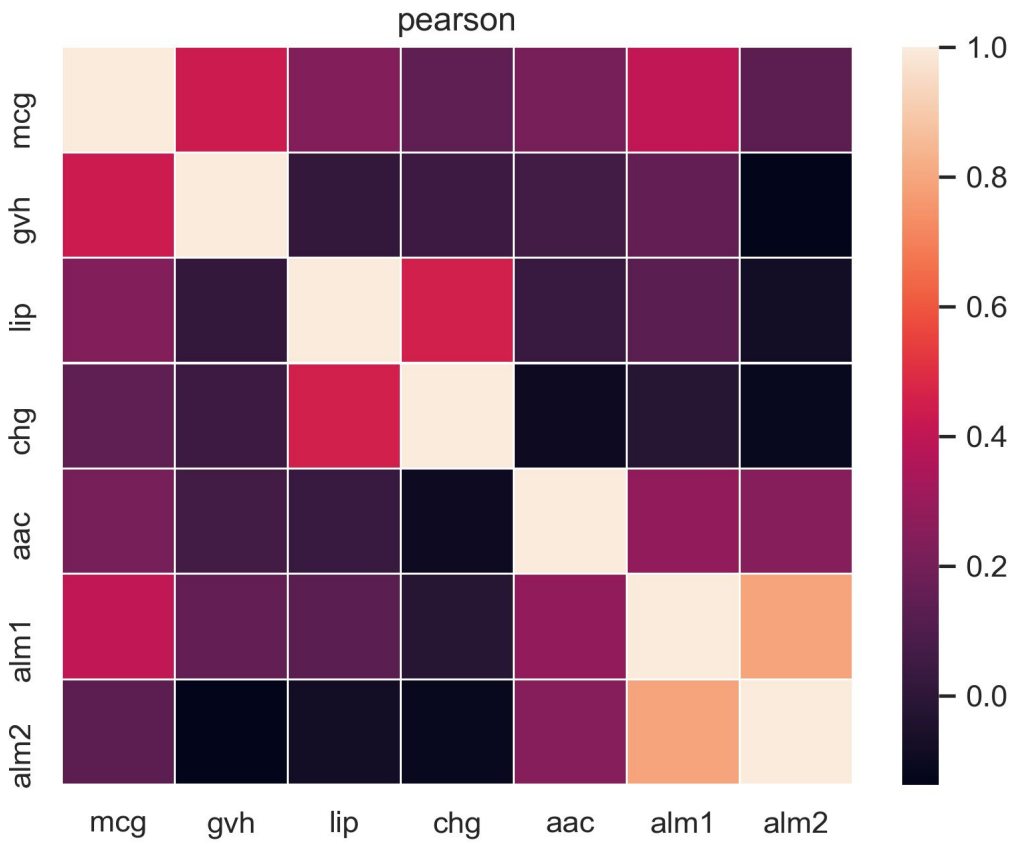
Ecoli dataset - overview

- hosted at UCI ML Repository - download
- classification
- 7 numeric attributes
- 357 records
- 8 target classes

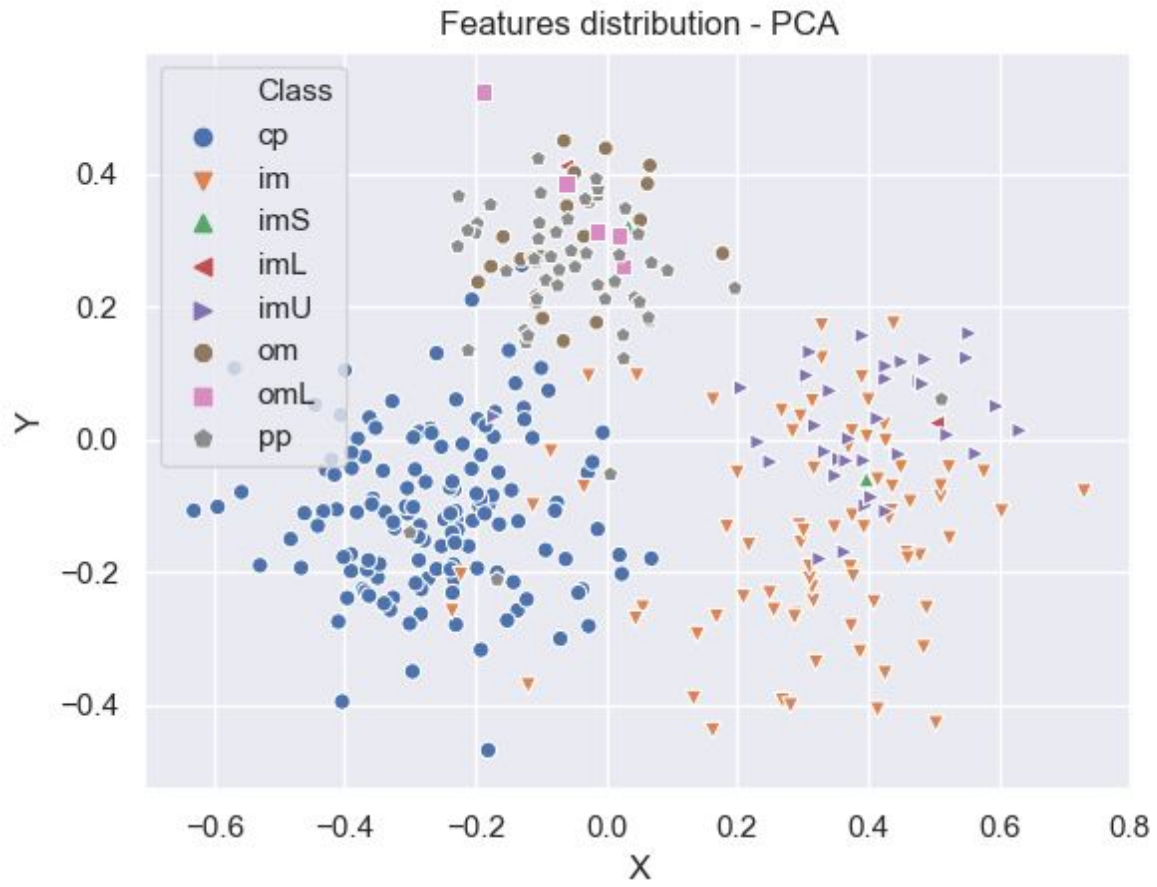
Ecoli dataset - class balance



Ecoli dataset - attribute correlation



Ecoli dataset - PCA



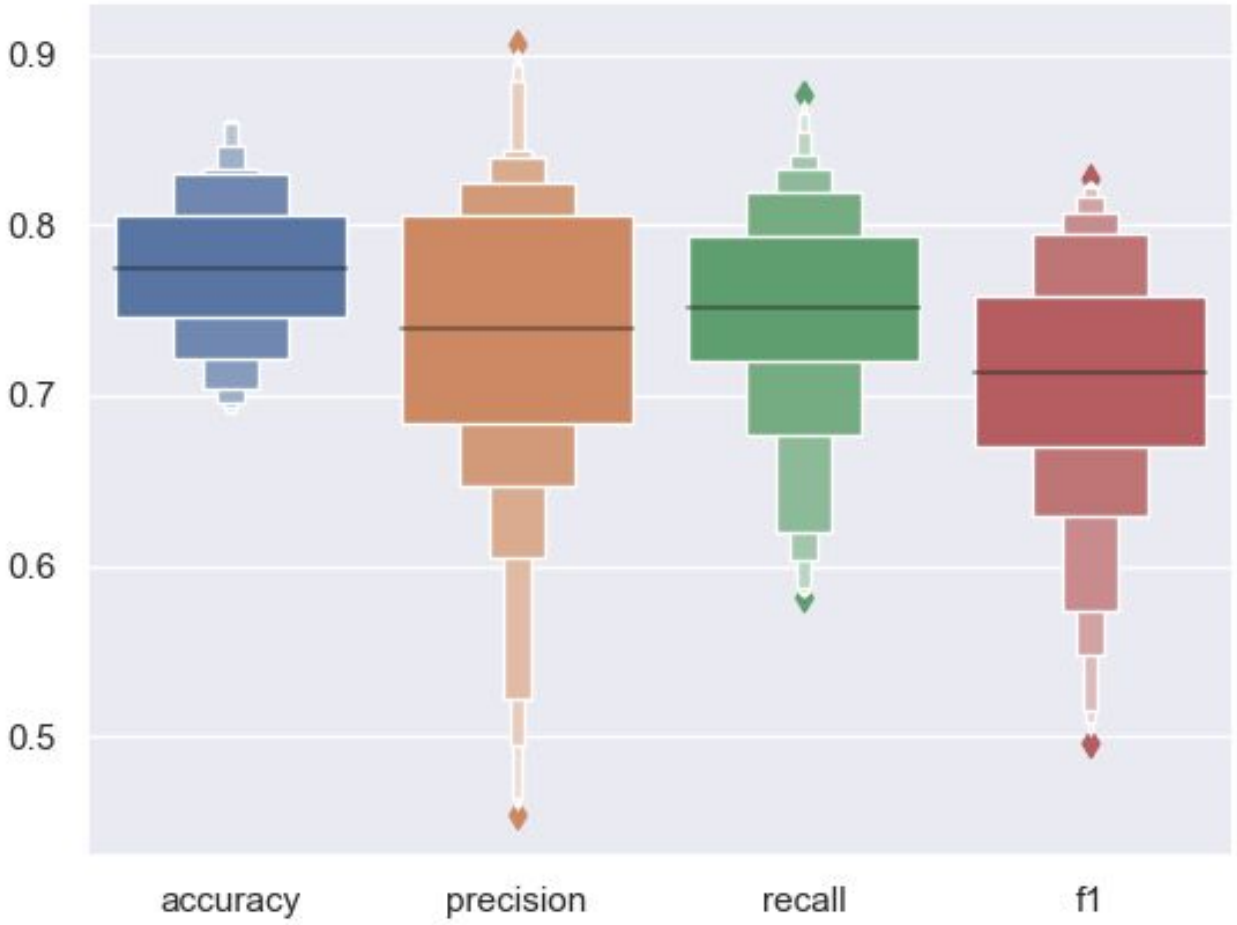
Pipelines

StandardScaler

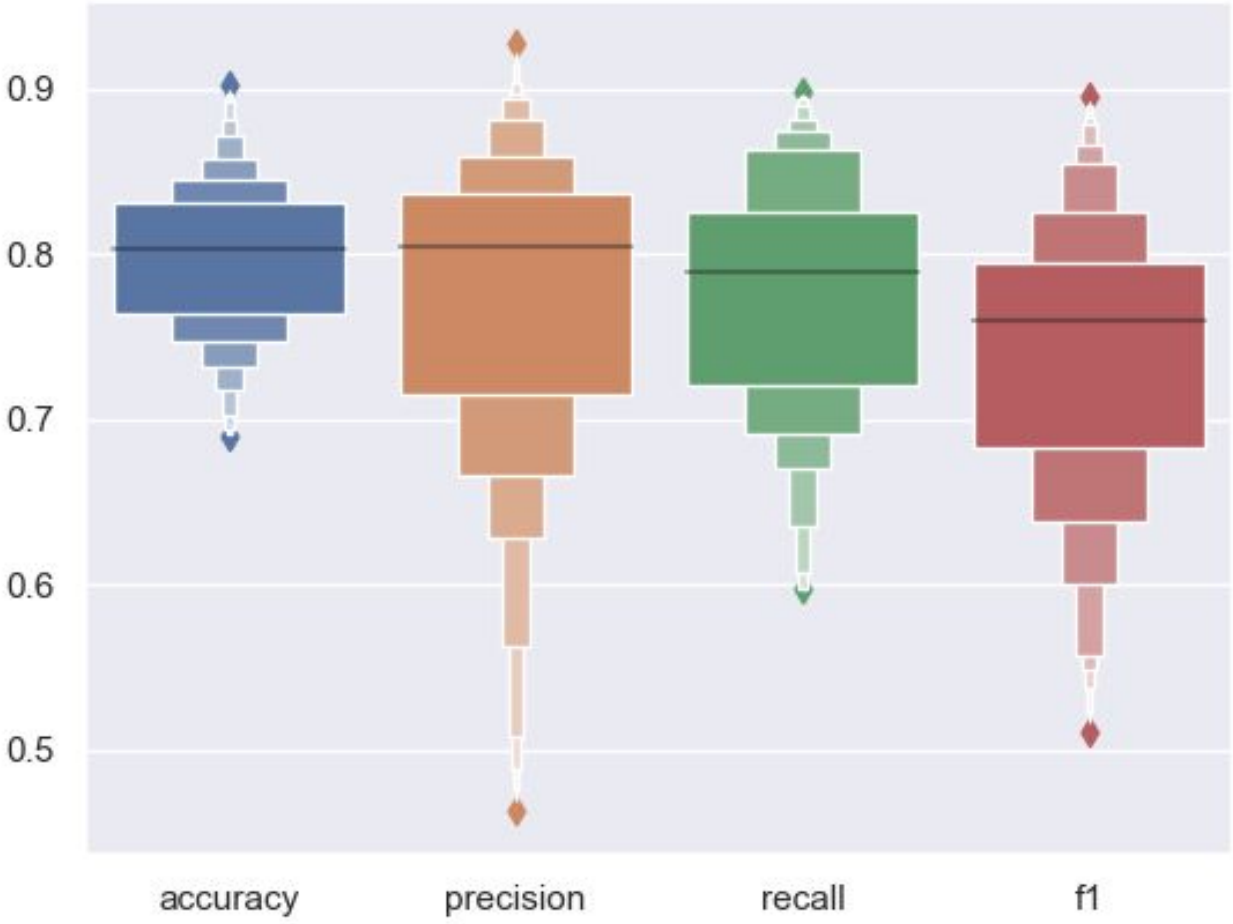
1. *GaussianNB*
2. *SMOTE >> GaussianNB*
3. *SMOTE >> Tomek-Links >> GaussianNB*

RepeatedStratifiedKFold(n_splits=5, n_repeats=20)

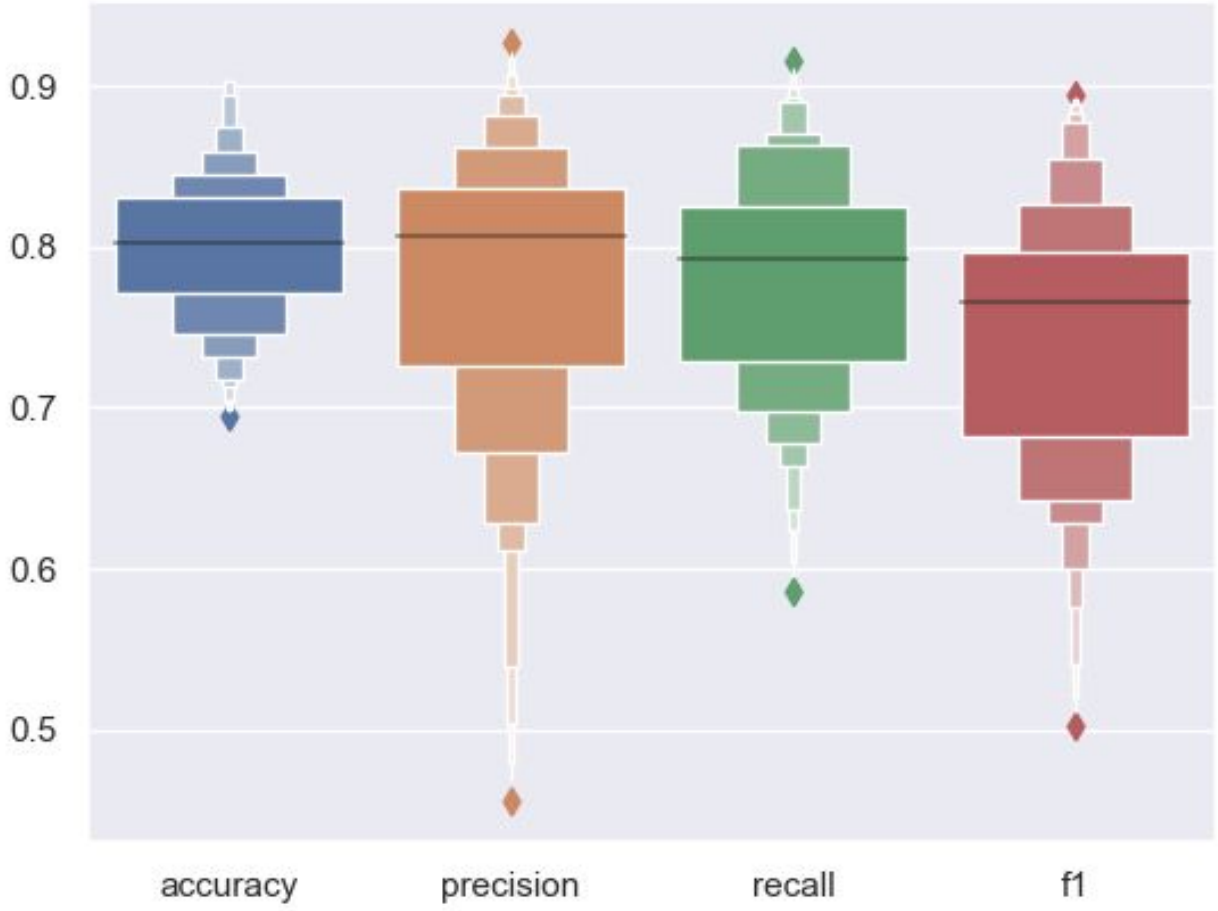
1. GaussianNB



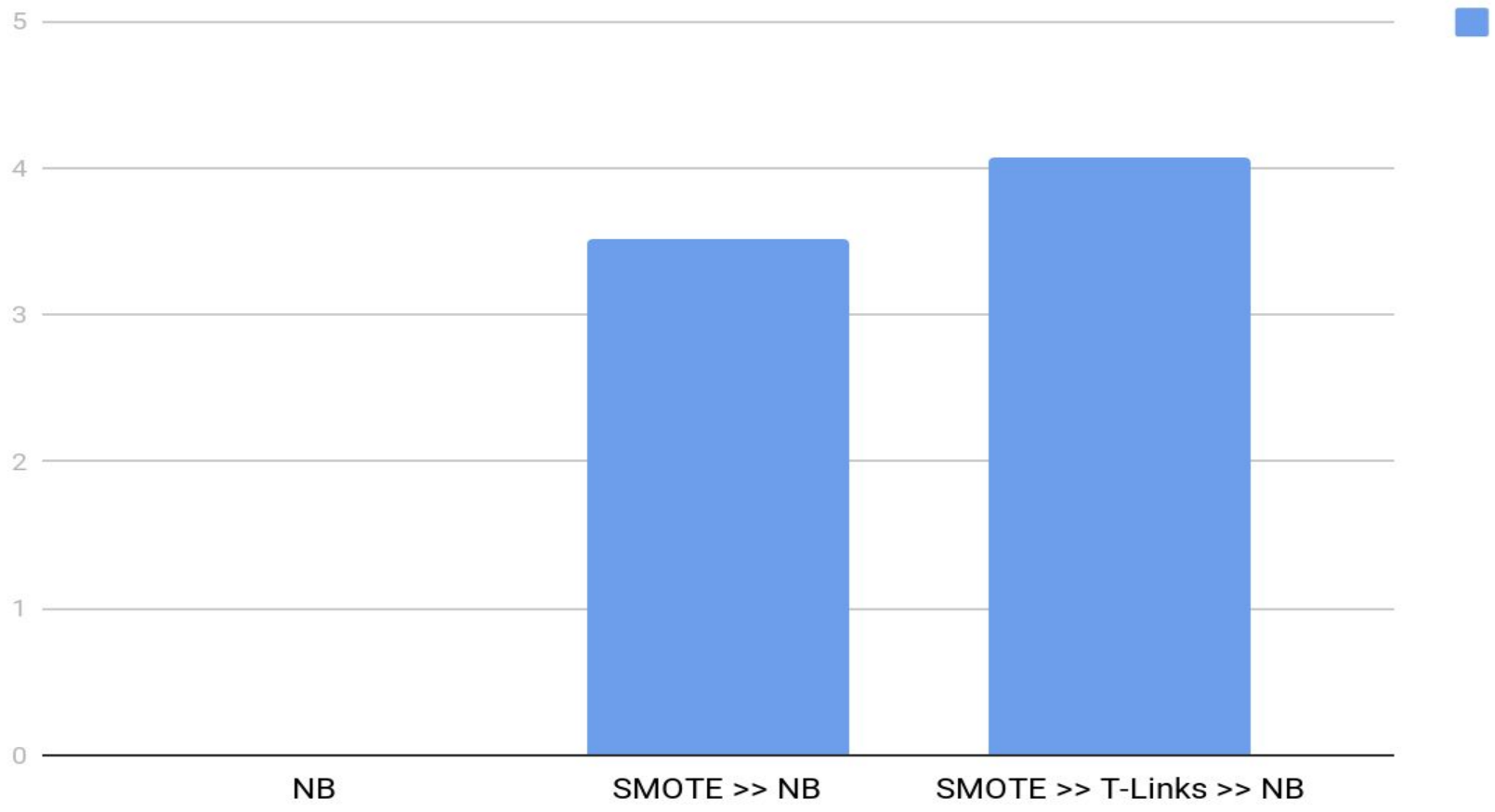
2. SMOTE >> GaussianNB



3. SMOTE >> Tomek-Links >> GaussianNB



KL divergence of F1



So what?

- obtain meaningful metrics before prod
- tune model for prod, not for nice scores to put in docs/wiki
- possible to squeeze few % from resampling

Cost sensitive learning

- Many of *sklearn* methods can be parametrized with custom class weights.
- In *keras* you can pass weights to *fit* method
- In *pytorch* you can instantiate your *loss* with specific weights

Advanced

- probability threshold moving
- probability calibration
- ensemble methods for imbalanced data
- one-class classification: svm, isolation forest, elliptic envelope, local outlier factor

Further reading

- https://github.com/scikit-learn-contrib/imbalanced-learn
- https://machinelearningmastery.com/imbalanced-classification-with-python

Q&A


SOFTWARE
HUT
GRUPA TENDERHUT

Thanks for coming!

- let us know your opinion and your needs
- call for speakers
- call for sponsors