Machine Learning 2

# Imbalanced Data & SMOTE

Dept. SW and Communication Engineering

Prof. Giseop Noh (kafa46@hongik.ac.kr)

# Contents

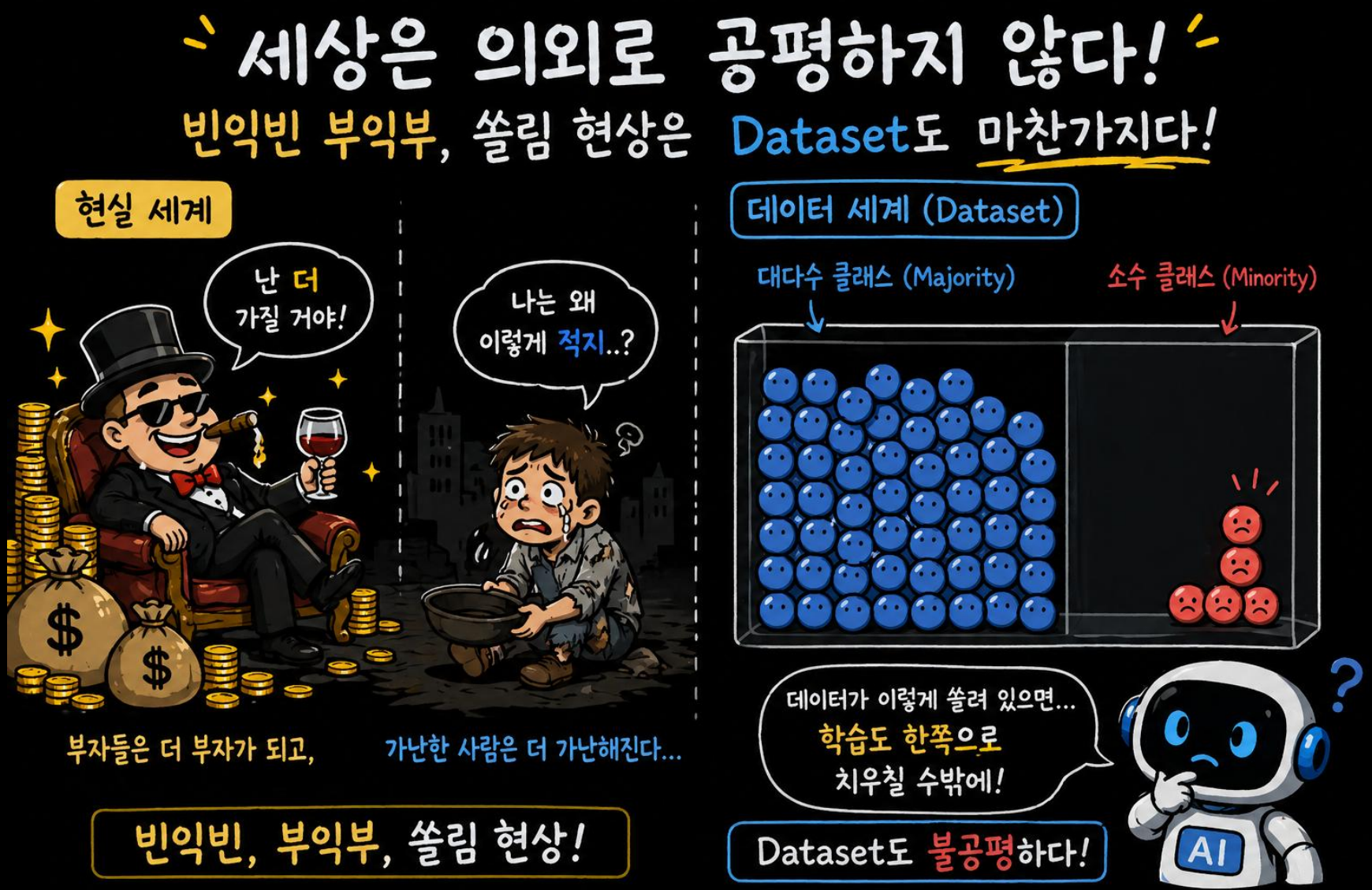
## Class Imbalance

## Sampling Techniques

- SMOTE
- Borderline SMOTE
- DBSCAN SMOTE

# Class Imbalance

# The Real World is Not Fair!



# Class Imbalance in Real-World Data

## What is Class Imbalance?

A situation where one class (majority) significantly outnumbers the other (minority)

Occurs in many real-world applications:

- Medical diagnosis → rare disease
- Semiconductor defect detection → low failure rate
- Fraud detection → very small number of fraud cases

Real-world datasets are rarely balanced (e.g., 5:5 or 6:4)

## Imbalance Ratio

$$IR\ (Imbalanced\ Ratio) = \frac{\#\ of\ majority\ class}{\#\ of\ minority\ class}$$

# Accuracy Can Be Misleading

## Accuracy Paradox in Imbalanced Data

In highly imbalanced datasets, accuracy becomes unreliable

Example:

- Total transactions = 1,000
- Fraud cases = 10 (1%)
- If Model prediction:
  - → Predict almost as normal
  - → Only hit only one abnormal by chance

**Result:**

- TP = 1 (fraud hit)
- FP = 0 (No false alarm)
- FN = 9 (fraud misses)
- TN = 990 (normal hits)

### Problem

- Looks excellent
- But detects only 1 out of 10
- High accuracy does NOT mean good performance

$$Accuracy = \frac{TP + TN}{\#\ All\ \ Predictions}$$

$$= \frac{1 + 990}{1000} = 0.991 = 99.1\%$$

# Precision Can Be Misleading

## Precision with Imbalanced Data

In highly imbalanced datasets, precision becomes unreliable

Example:

- Total transactions = 1,000
- Fraud cases = 10 (1%)
- If Model prediction:
  → Predict almost as normal
  → Only hit only one abnormal by chance

**Result:**

- TP = 1 (fraud hit)
- FP = 0 (No false alarm)
- FN = 9 (fraud misses)
- TN = 990 (normal hits)

### Problem

- Model predicts very few positives
- Precision becomes artificially high (100%)
- High precision does NOT mean good detection

$$Precision = \frac{TP}{TP + FP}$$

$$= \frac{1}{1 + 0} = 1.0 = 100\%$$

# Also, Recall Can Be Misleading

## Recall with Imbalanced Data

In highly imbalanced datasets, recall becomes unreliable

Example:

- Total transactions = 1,000
- Fraud cases = 10 (1%)
- If Model prediction:
  - → Predict almost as normal
  - → Only hit only one abnormal by chance

**Result:**

- TP = 1 (fraud hit)
- FP = 0 (No false alarm)
- FN = 9 (fraud misses)
- TN = 990 (normal hits)

### Problem

- Model misses most minority (fraud) cases
- Recall is extremely low (10%)
- Model fails to detect important events

$$Recall = \frac{TP}{TP + FN}$$

$$= \frac{1}{1 + 9} = 0.1 = 10\%$$

# Perfect Scores, Useless Model

Example:

- Total transactions = 1,000
- Fraud cases = 1 (0.1%)
- If Model prediction:
  - → Predict almost as normal
  - → Randomly detects the only fraud case

Result:

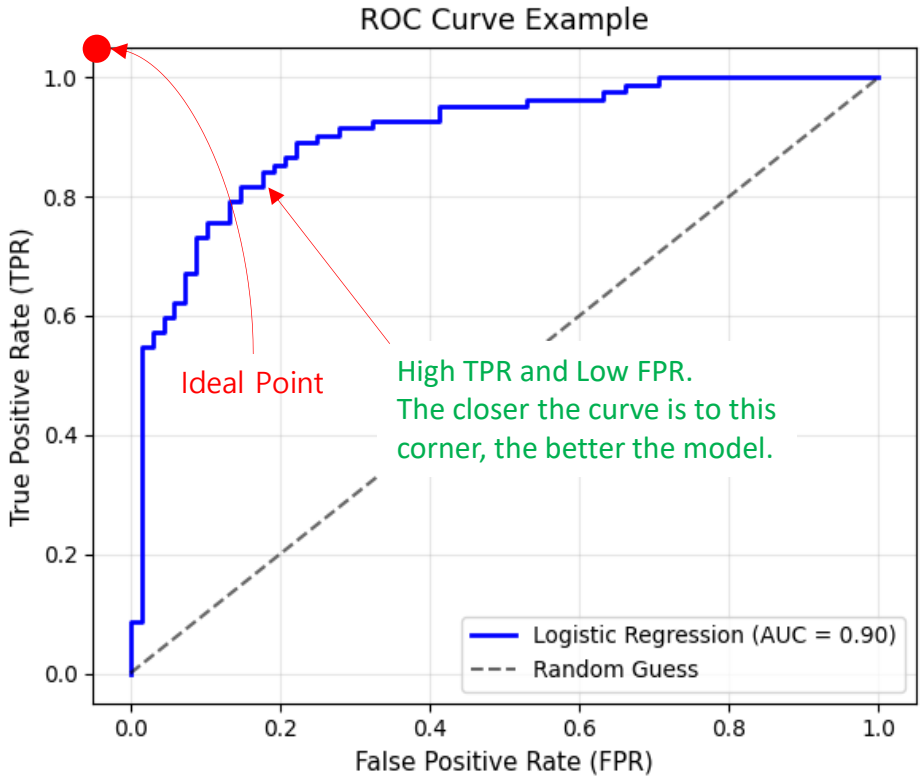
- TP = 1 (fraud hit)
- FP = 0 (No false alarm)
- FN = 0 (fraud misses)
- TN = 990 (normal hits)

$$Accuracy = \frac{1 + 999}{1000} = 100\%$$

$$Precision = \frac{1}{1 + 0} = 100\%$$

$$Recall = \frac{1}{1 + 0} = 100\%$$

# Recap: ROC & AUC




| Metric | Obser-vation | Interpretation |
|---|---|---|
| **TPR (Sensitivity)** | High | The model detects most of the actual positives. $$TPR = \frac{TP}{TP + FN}$$ |
| **FPR (1-Specificity)** | Low | The model rarely misclassifies negatives as positives. $$FPR = \frac{FP}{TN + FP} = 1 - \frac{TN}{TN + FP} = 1 - specificity$$ |
| **AUC = 0.90** | Excellent | The model has strong discriminative power. |

# ROC Collapses to a Single Point

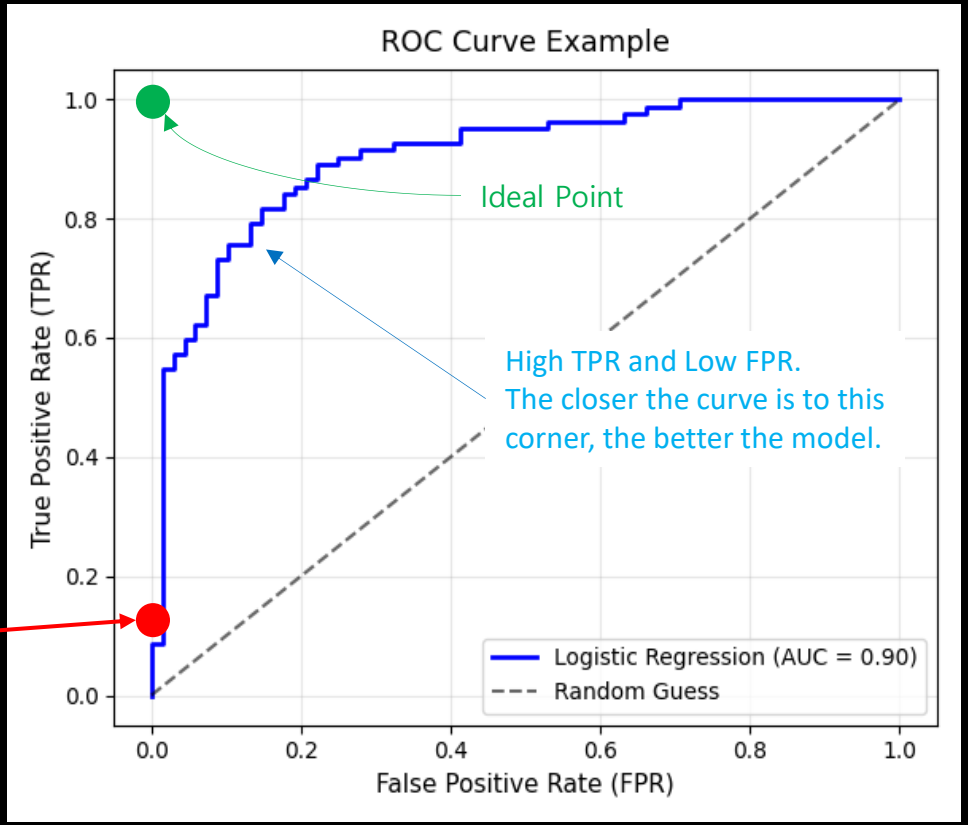
Our model is highly skewed toward predicting normal.

Most fraud cases receive low scores, except one by chance

⇩

Only one point is observed.

Current model, however, operates at a single point with low TPR despite having almost no false positives.

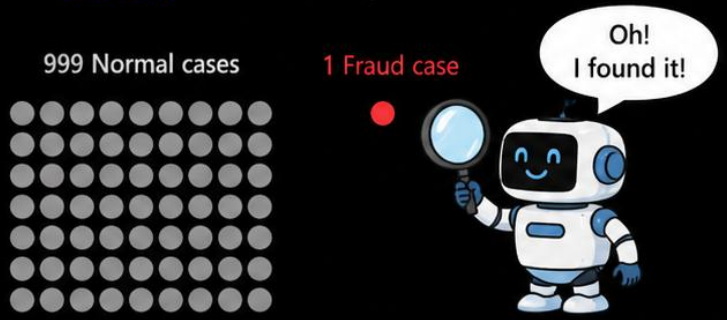


# Lucky Success vs. Real Ability

**A model that looks perfect by chance is like someone who got lucky once.**

## The Model

- Fraud cases = 1 out of 1,000
- Predict almost everything as normal
- Randomly detects the only fraud




| Result (by chance) | | Looks perfect! |
|---|---|---|
| | Accuracy = 100% | |
| | Precision = 100% | |
| | Recall = 100% | |

## In Real Life

- Many people invest without much analysis
- One person happens to make a huge profit
- Was it skill? Or just luck?


**One-time success does NOT mean real ability.**

→ The model (or person) must prove consistent performance, not just accidental success.

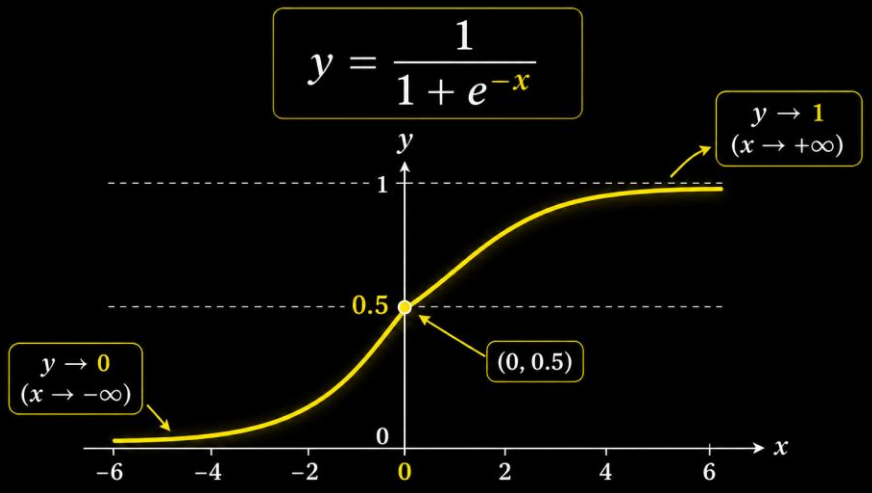
# Threshold Problem in Imbalanced Data

## What is the Class Imbalanced Problem?

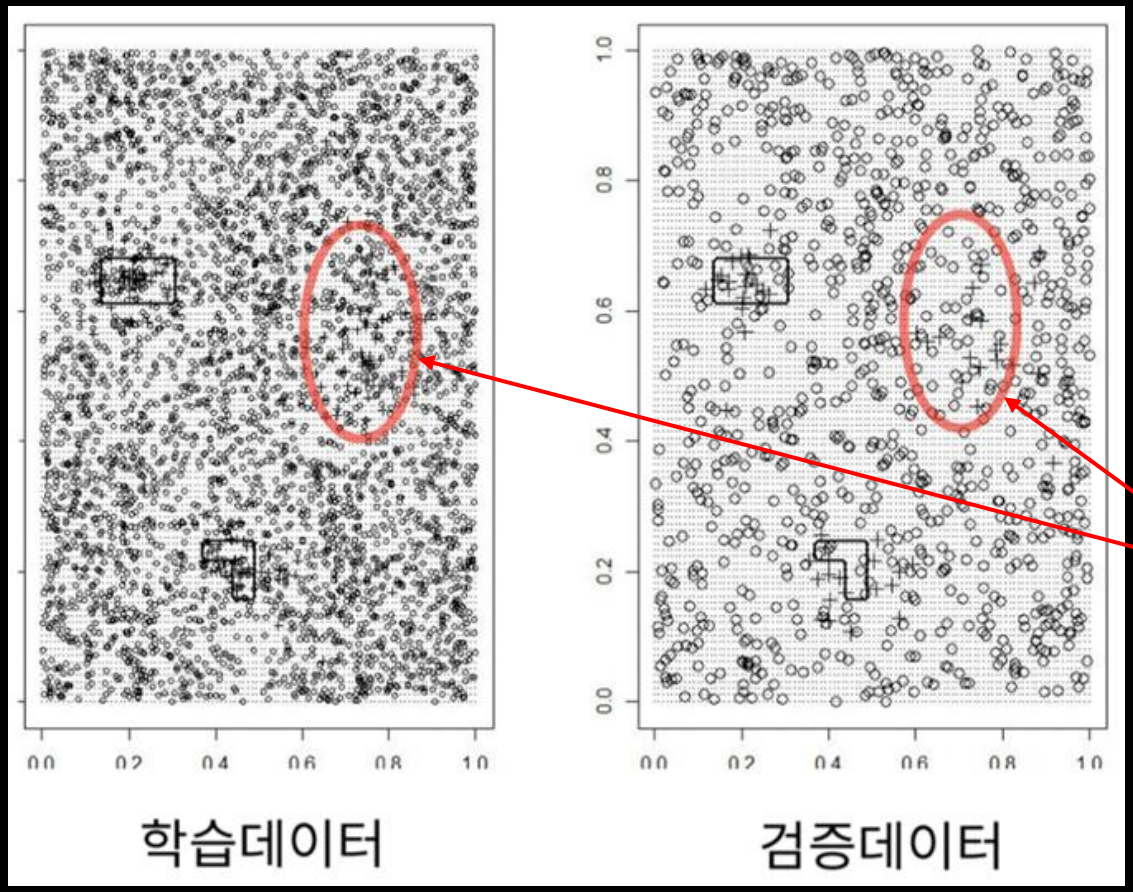
In binary classification, models typically output a probability value for each sample

When the imbalance ratio (IR) is high, the model tends to predict probabilities close to 0 for most samples

Prediction threshold (default: 0.5) is truly reliable?



# Model Behavior on Imbalanced Data




- $IR = 20$, 2D dataset,
- Decision Tree Training

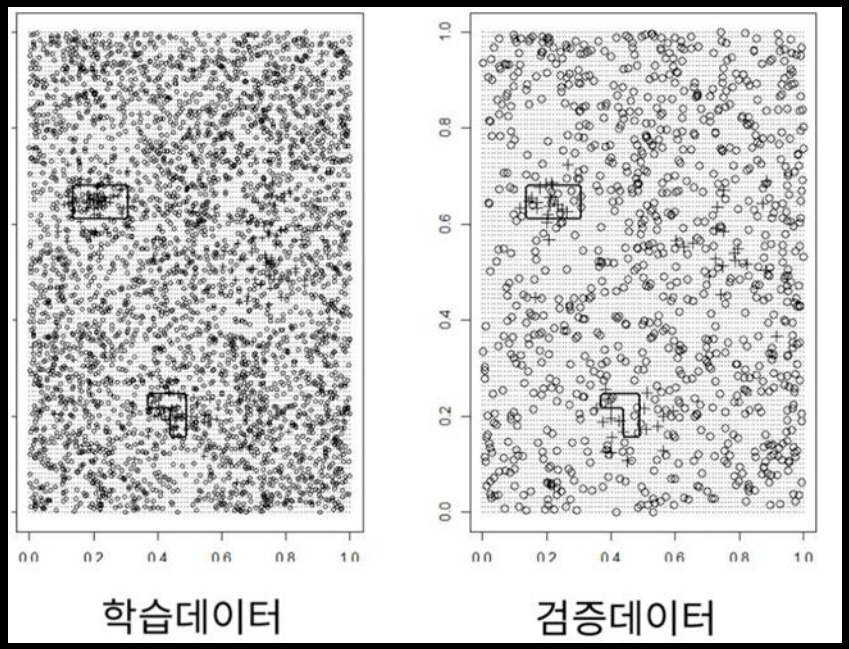
  → Decision Boundary

- Minority 데이터가 모여 있는 영역
- 하지만 주변은 대부분 majority 데이터로 둘러싸임

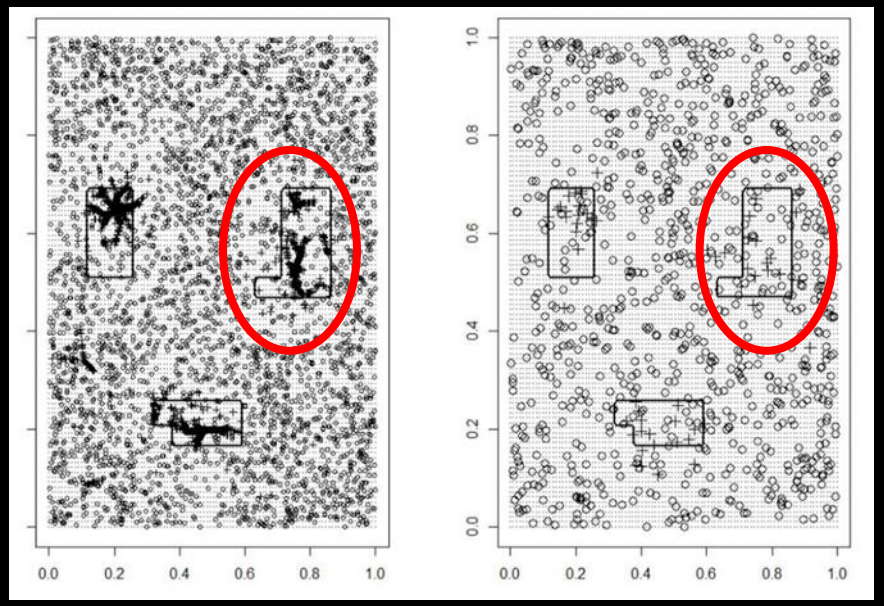
→ 모델이 이 영역을 제대로 학습하지 못함

그림출처: https://heung-bae-lee.github.io/2020/06/06/machine_learning_21

# The Problem is the Data, Not the Model




Can we change the decision boundary as follow?



Problem is not overfitting!

Problem is inherent data imbalance!

# How Can We Fix This Problem?

The problem is data imbalance

Models cannot learn minority patterns properly

→ Changing the model is NOT enough

→ Must modify the data distribution

Solution? Fix the data, not just the model

Resampling Methods
- Over-sampling
- Under-sampling

# Sampling Techniques

# How Do We Fix Class Imbalance?

## Class Imbalanced Problem: Solutions

### Resampling: Data-based Apporach

- Over sampling: Increase minority class data
- Under sampling: Reduce majority class data
- Hybrid resampling: Combination of over- & under-sampling

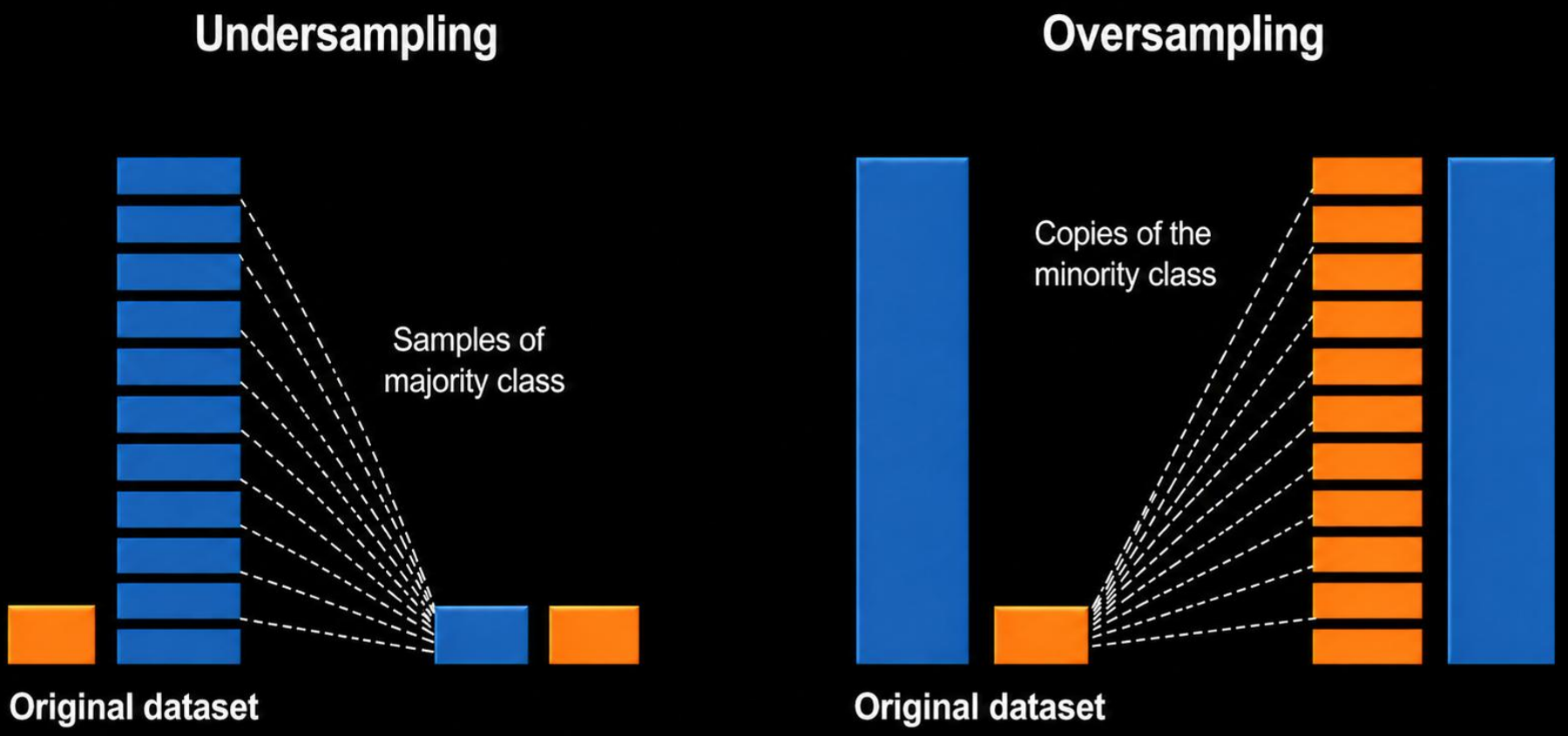
### Cost-sensitive learning: Model-based Apporach

Assign higher penalties to misclassification of important classes

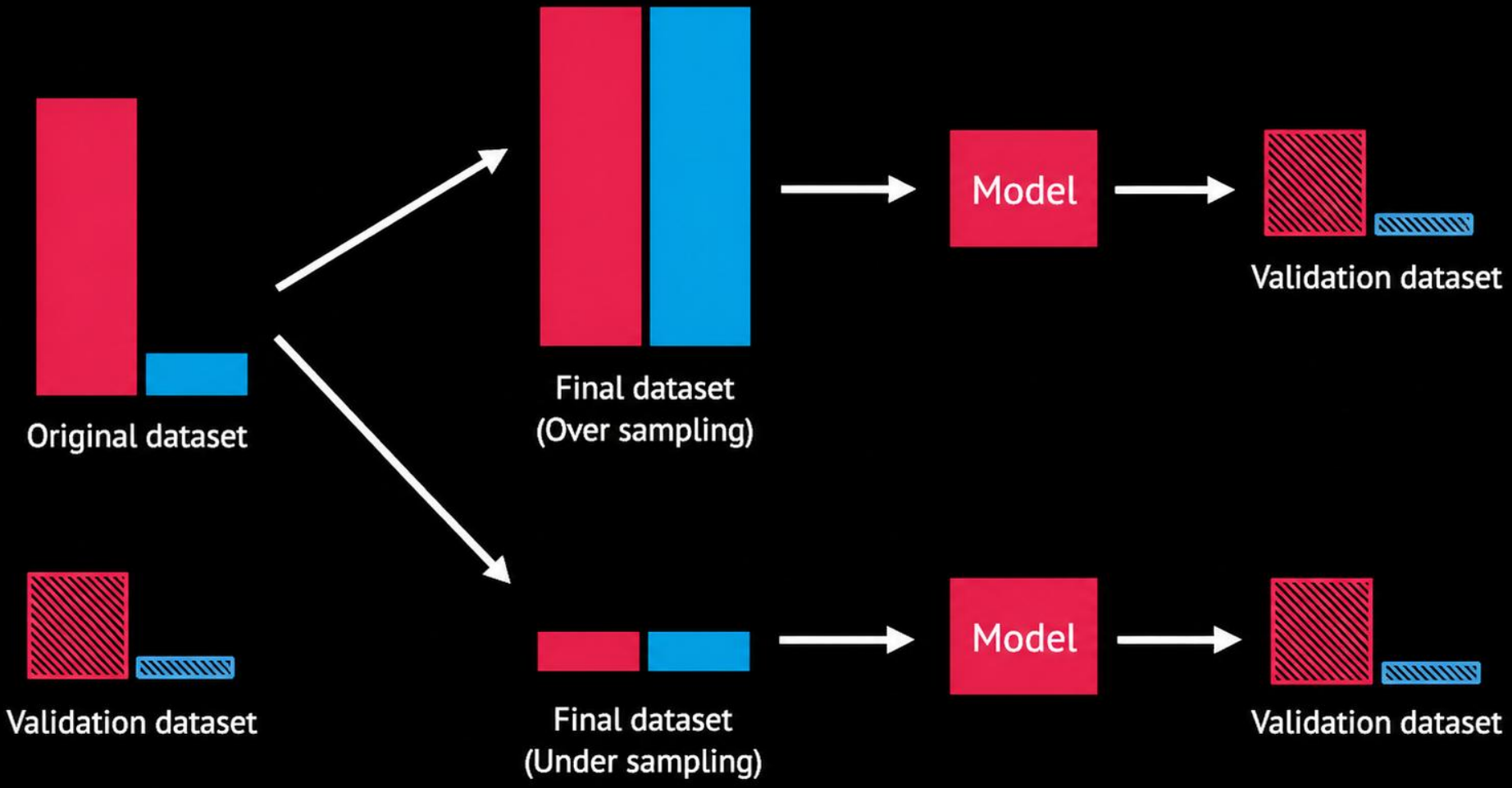
- Introduce class weights in the loss function
- Penalize false negatives more heavily
- Boosting

⋮

# Balancing Data: Undersampling vs Oversampling



# Balance Training, Keep Evaluation Realistic

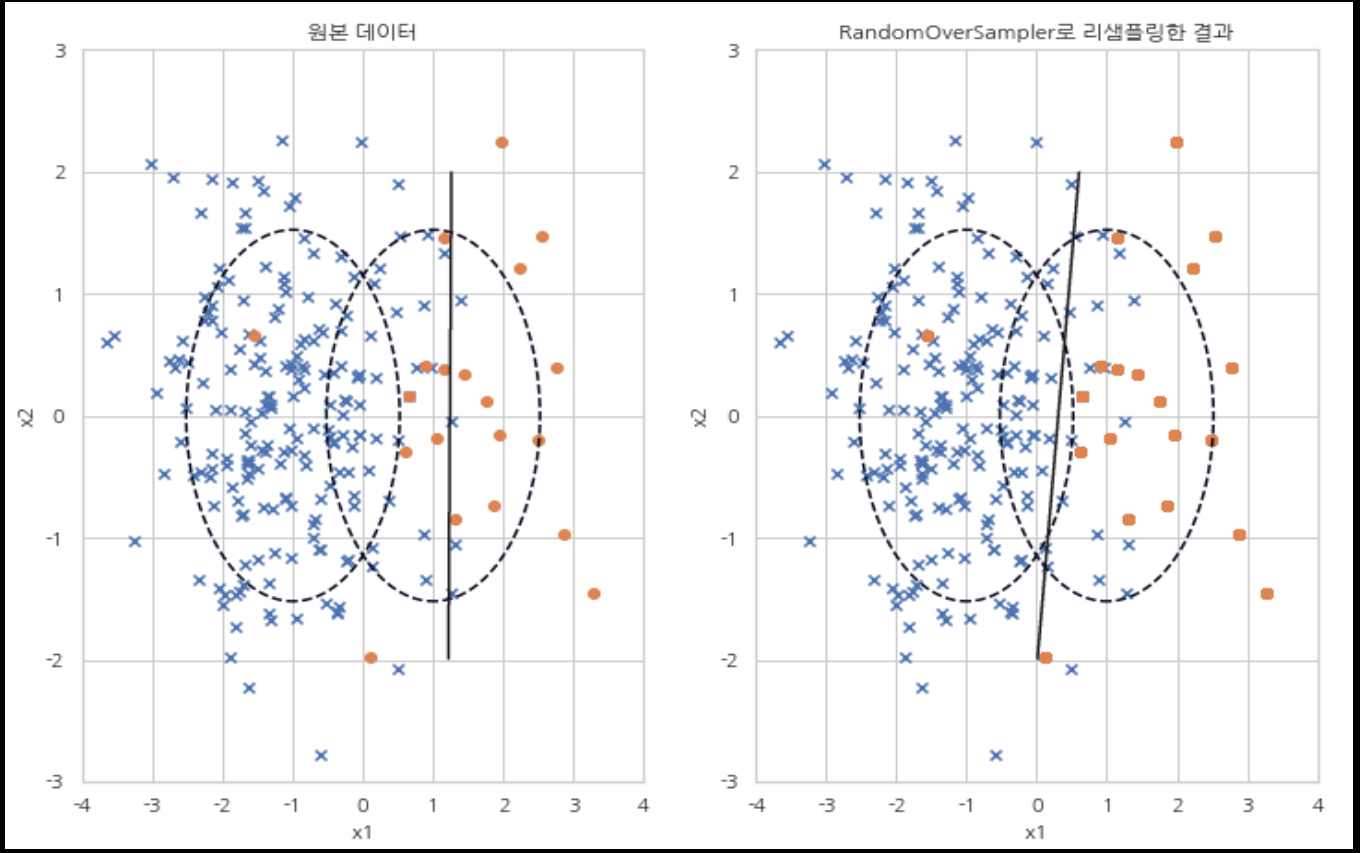


# Over Sampling Techniques

- Random
- SMOTE
- BLSMOTE
- DBSMOTE

# Technique 1. Random Over Sampling

- Randomly duplicates samples from the minority class and adds them to the original dataset
- A simple yet effective method that often performs well






그림출처: https://heung-bae-lee.github.io/2020/06/06/machine_learning_21

# Technique 2. SMOTE (1/2)

## SMOTE (Synthetic Minority Over-sampling Technique)

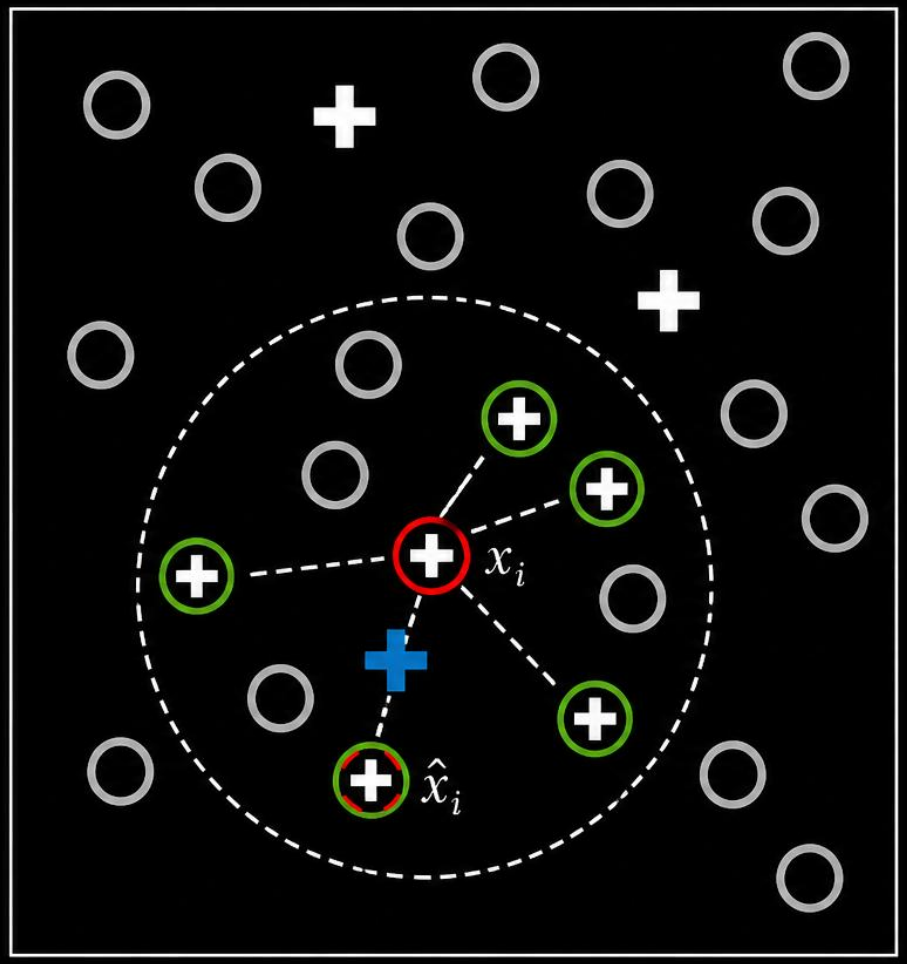
- A fundamental and widely used oversampling method
- After applying KNN to minority samples, new data points are generated randomly between a sample and its nearest neighbors
- Many variants of SMOTE have been proposed

## KNN (K-Nearest Neighbors)

makes predictions by finding the K closest data points (neighbors) to a given input based on a distance metric (e.g., Euclidean distance).

- Classification: Assigns the class most common among the K neighbors.
- Regression: Predicts the average value of the K neighbors.

# Technique 2. SMOTE (2/2)




Majority class samples

Minority class samples

Randomly selected minority class sample $x_i$

5 $K$-nearest neighbors of $x_i$

Randomly selected sample $\hat{x}_i$ from the 5 neighbors

Generated synthetic minority instance

Between two minority samples (random interpolation)

# Technique 3. BLSMOTE (1/3)

# Technique 3. BLSMOTE (2/3)

## BLSMOTE (Borderline SMOTE)

- Focus on samples near the decision boundary because those are more difficult to classify.
- Borderline: majority neighbors > 50%
  - A minority sample is considered borderline
    if more than half of its neighbors belong to the majority class
- Apply SMOTE only to borderline samples

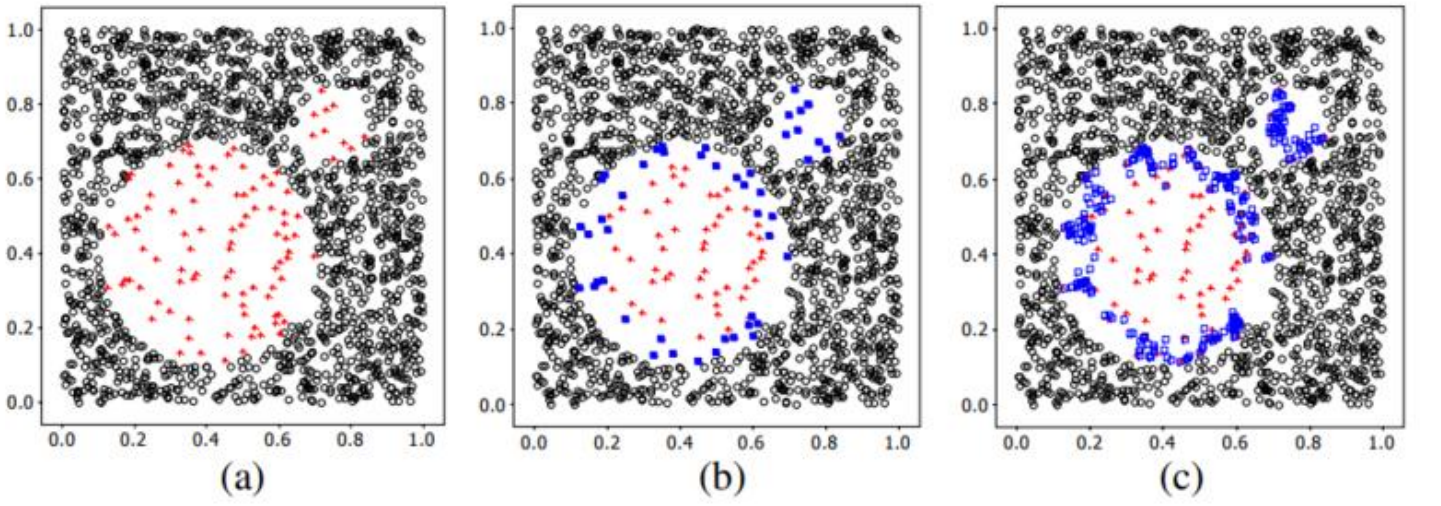
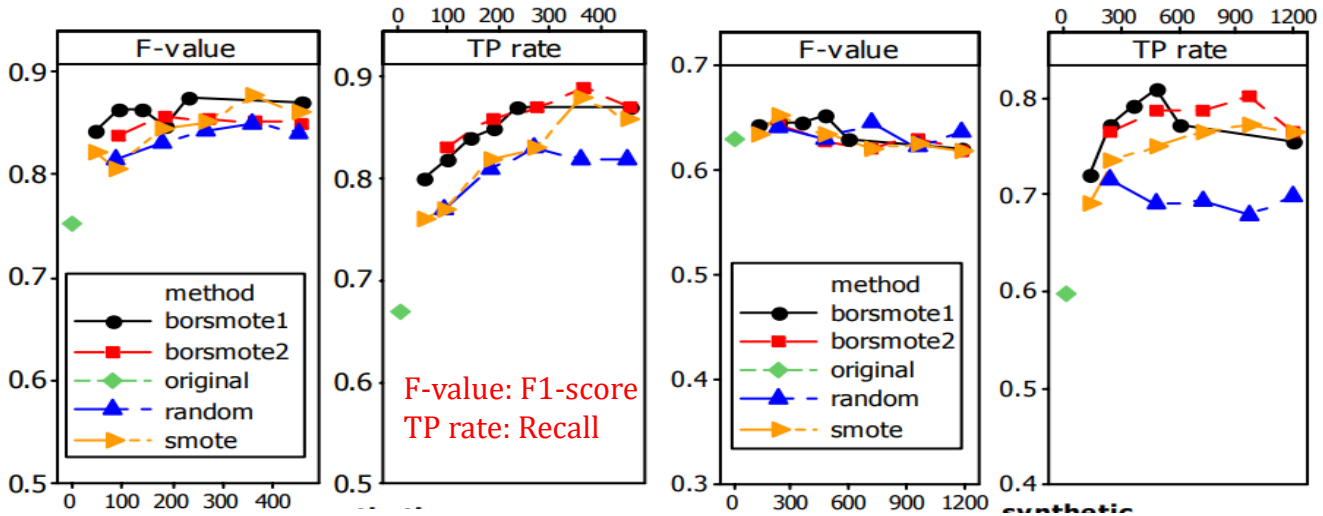



**Fig. 1.** (a) The original distribution of Circle data set. (b) The borderline minority examples (*solid squares*). (c) The borderline synthetic minority examples (*hollow squares*).

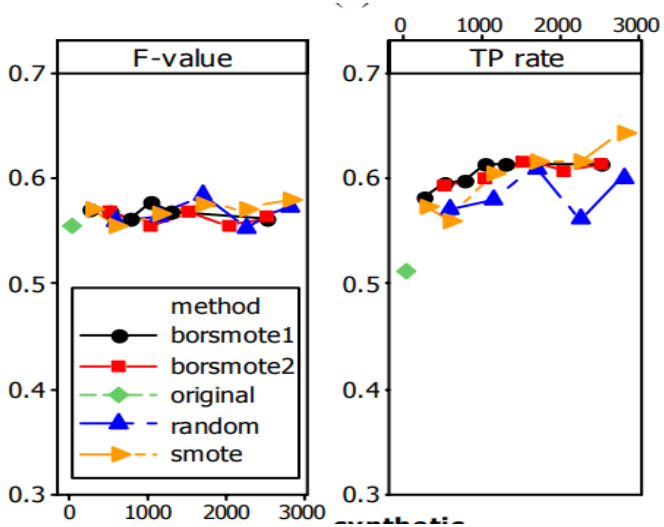
# Technique 3. BLSMOTE (3/3)



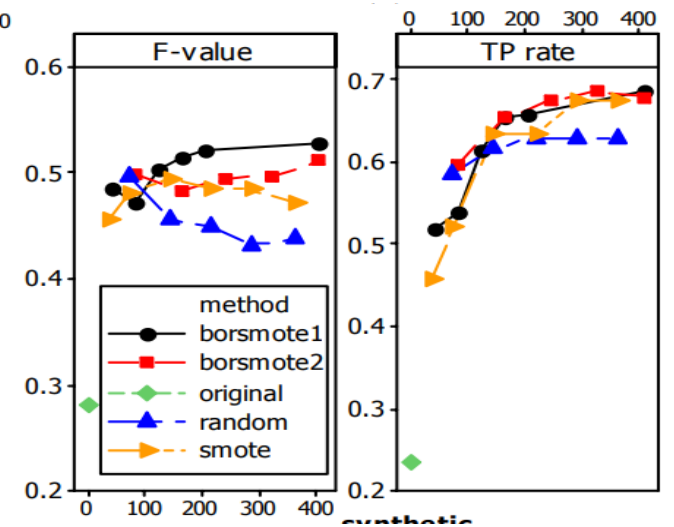

Circle dataset (minority class: 6.25%)




Pima dataset (minority class: 34.77%)




Satimage dataset (minority class: 9.73%)




Haberman dataset (minority class: 26.47%)

Fig. 2. (a), (b), (c) and (d) illustrate the F-value and TP rate for minority class when proposed over-sampling methods are applied on Circle, Pima, Satimge and Haberman respectively with C4.5. "borsmote1" and "borsmote2" denote borderline-SMOTE1 and borderline-SMOTE2, "random" denotes random over-sampling, and "original" denotes the values of the original data sets. The x-axis is the number of synthetic examples

그림출처: Hui Han et al. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning, ICIC, 2005

https://sci2s.ugr.es/keel/keel-dataset/pdfs/2005-Han-LNCS.pdf

# Technique 4. DBSMOTE (1/3)

## DBSMOTE (DBSCAN SMOTE)

"무작정 늘리지 말고, 구조를 먼저 찾고 그 안에서 생성하자"

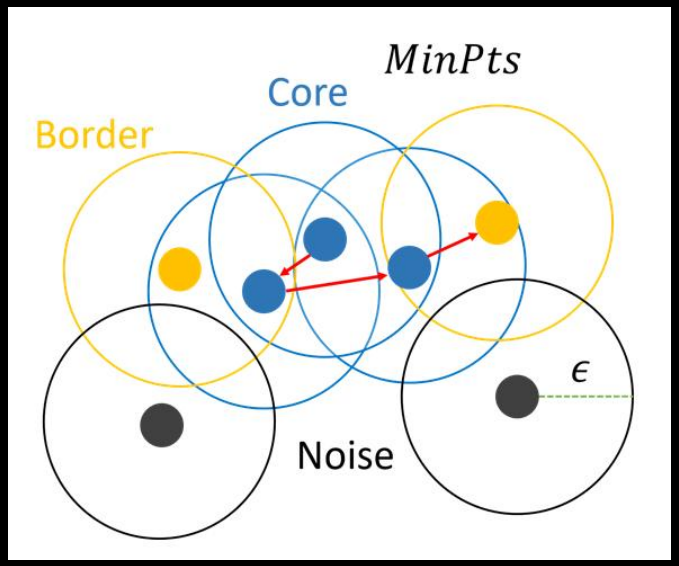
| Existing SMOTE | DBSCAN SMOTE |
| --- | --- |
| • Including Noise<br>• Can cross decision boundary | • Exclude Noise<br>• Preserve data structure<br>• Produces stable distribution |

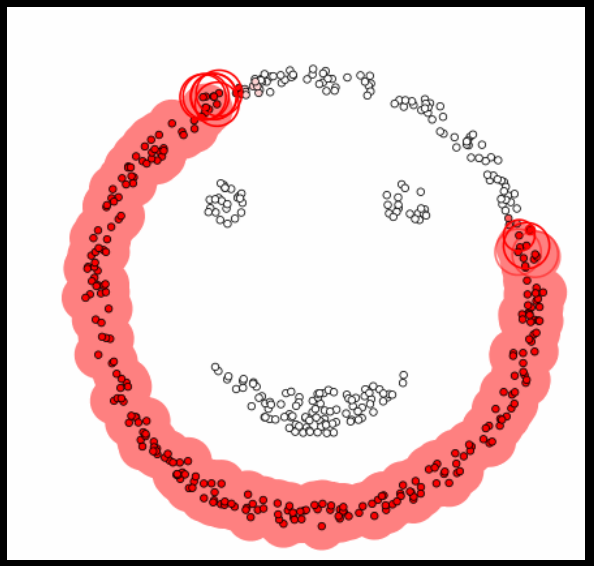
Instead of K-means, applying DBSCAN algoritm!

# Technique 4. DBSMOTE (2/3)

## DBSCAN (Density-Based Spatial Clustering)

1. 임의의 점을 선택한다.
2. 각각의 객체들은 반경 $\epsilon$을 기준으로 최소 이웃 minPts 이상을 충족하면 군집
3. 최소 이웃 minPts를 충족하지 못하면 잡음으로 판단
4. 군집되었다면 군집된 이웃 객체를 대상으로 1번 반복

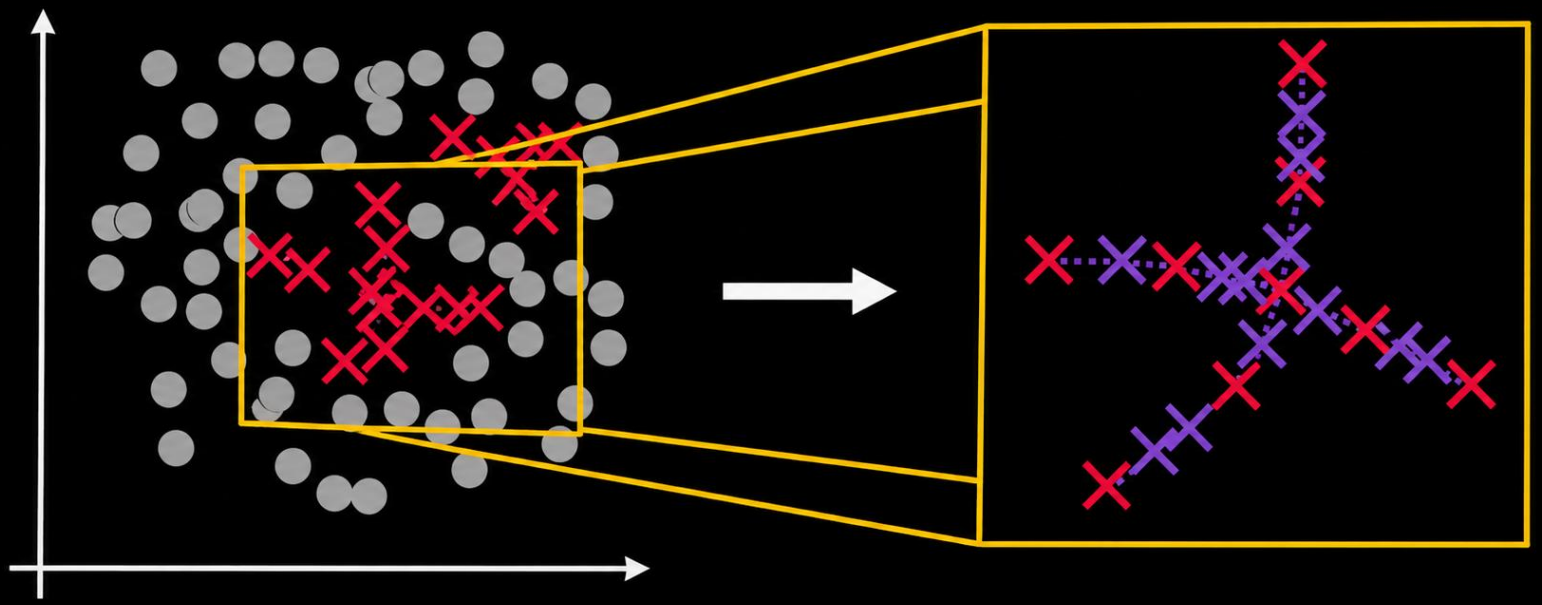


- $\epsilon$ : neighborhood radius
- MinPts: minimum number of points to form a cluster

# Technique 4. DBSMOTE (3/3)

- Apply DBSCAN to the minority data → form clusters
- Generate new samples randomly along the line between each cluster centroid and its data points
- Advantage: Enables sampling while removing noise (benefit of DBSCAN clustering)



# Under Sampling Techniques

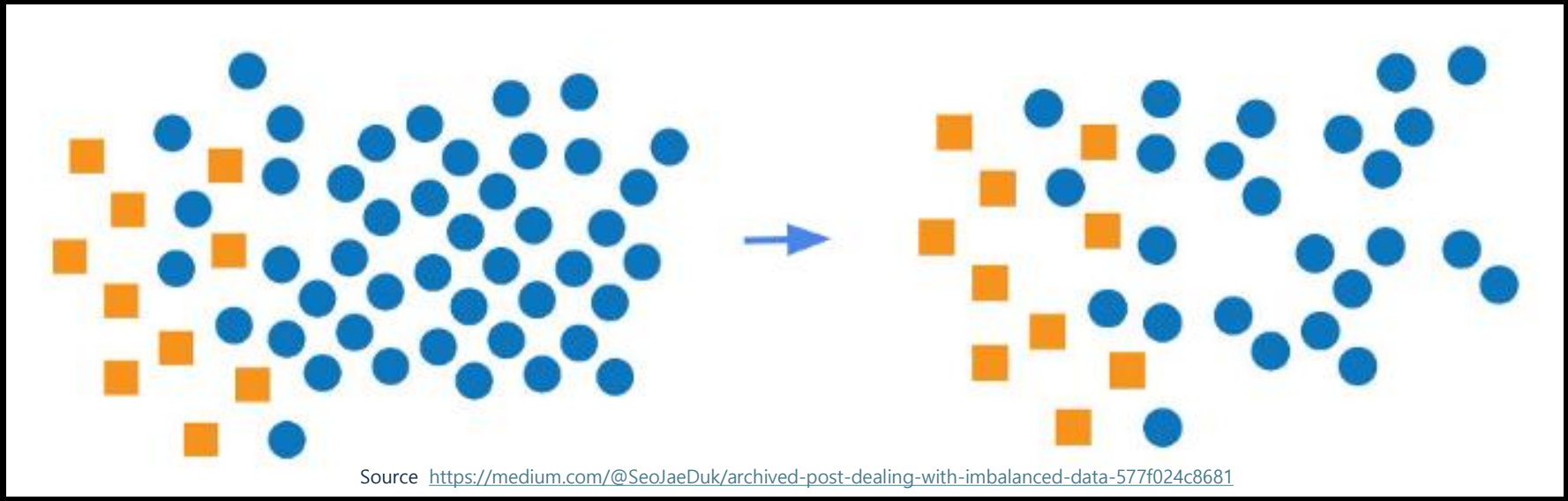
- Random
- Tomek Links
- EasyEnsemble

# Random Under Sampling

## Random Under Sampling

- Randomly remove samples from the majority class
- The simplest approach

Limitation: Not consider cluster structure or borderline characteristics



Source https://medium.com/@SeoJaeDuk/archived-post-dealing-with-imbalanced-data-577f024c8681

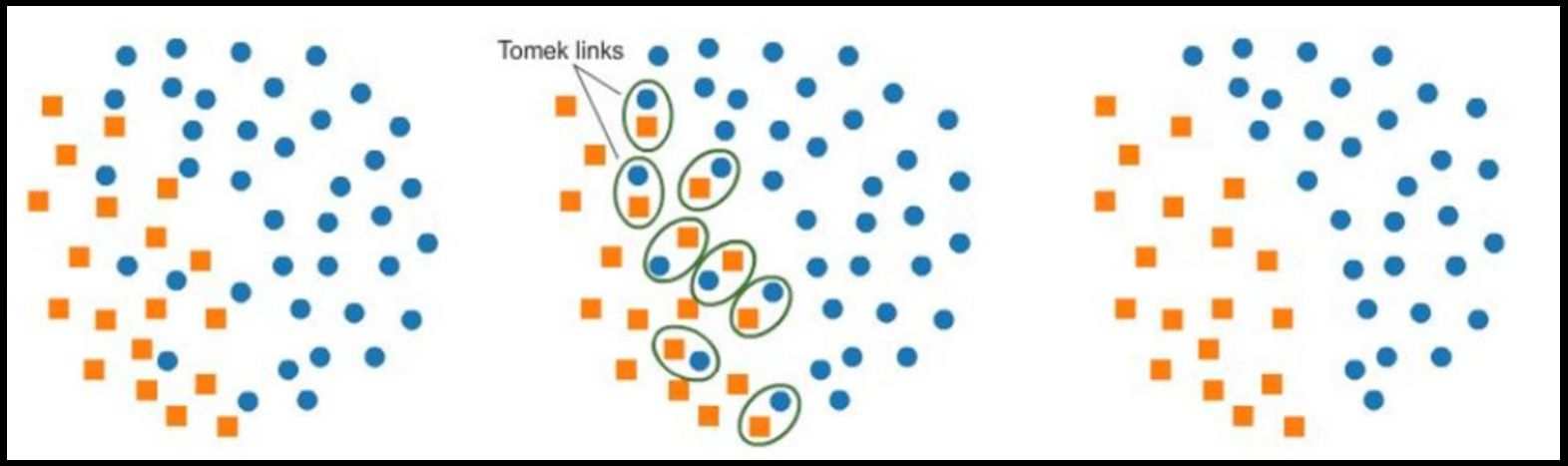
# Tomek Links

## Random Under Sampling

- Remove majority class samples that lie on the borderline between majority and minority classes

Limitations:

- Works well in low-dimensional spaces (2D or 3D)
- Becomes difficult to interpret distances and links in high-dimensional data (4D and above)



# EasyEnsemble (1/2)

# EasyEnsemble (2/2)

실제로 삭제하는 것은 아님 → 논리적으로 삭제함

Minority는 그대로 유지

(소중한 데이터는 모두 사용)

Majority를 여러 번 랜덤 샘플링

Minority 개수만큼 맞춰서 뽑음

Subset 1: Majority 일부
+ Minority 전체

Subset 2: Majority 다른 일부
+ Minority 전체

Subset 3: …

각각 따로 모델 학습 (모델 여러 개 생성)

마지막에 결과 합치기 (Ensemble): Voting / averaging

**Algorithm 1** The `EasyEnsemble` algorithm.

1: {Input: A set of minor class examples $\mathcal{P}$, a set of major class examples $\mathcal{N}$, $|\mathcal{P}| < |\mathcal{N}|$, and $T$, the number of subsets to be sampled from $\mathcal{N}$.}
2: $i \leftarrow 0$
3: **repeat**
4: $\quad i \leftarrow i + 1$
5: $\quad$ Randomly sample a subset $\mathcal{N}_i$ from $\mathcal{N}$, $|\mathcal{N}_i| = |\mathcal{P}|$.
6: $\quad$ Learn $H_i$ using $\mathcal{P}$ and $\mathcal{N}_i$. $H_i$ is an AdaBoost ensemble with weak classifiers $h_{i,j}$ and corresponding weights $\alpha_{i,j}$, i.e.

$$H_i(\boldsymbol{x}) = \text{sgn}\left(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(\boldsymbol{x}) - \theta_i\right).$$

7: **until** $i = T$
8: Output: An ensemble:

$$H(\boldsymbol{x}) = \text{sgn}\left(\sum_{i=1}^{T}\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(\boldsymbol{x}) - \sum_{i=1}^{T} \theta_i\right).$$

자료출처: Liu, Xu-Ying, Jianxin Wu, and Zhi-Hua Zhou. "Exploratory undersampling for class-imbalance learning." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2 (2008): 539-550.

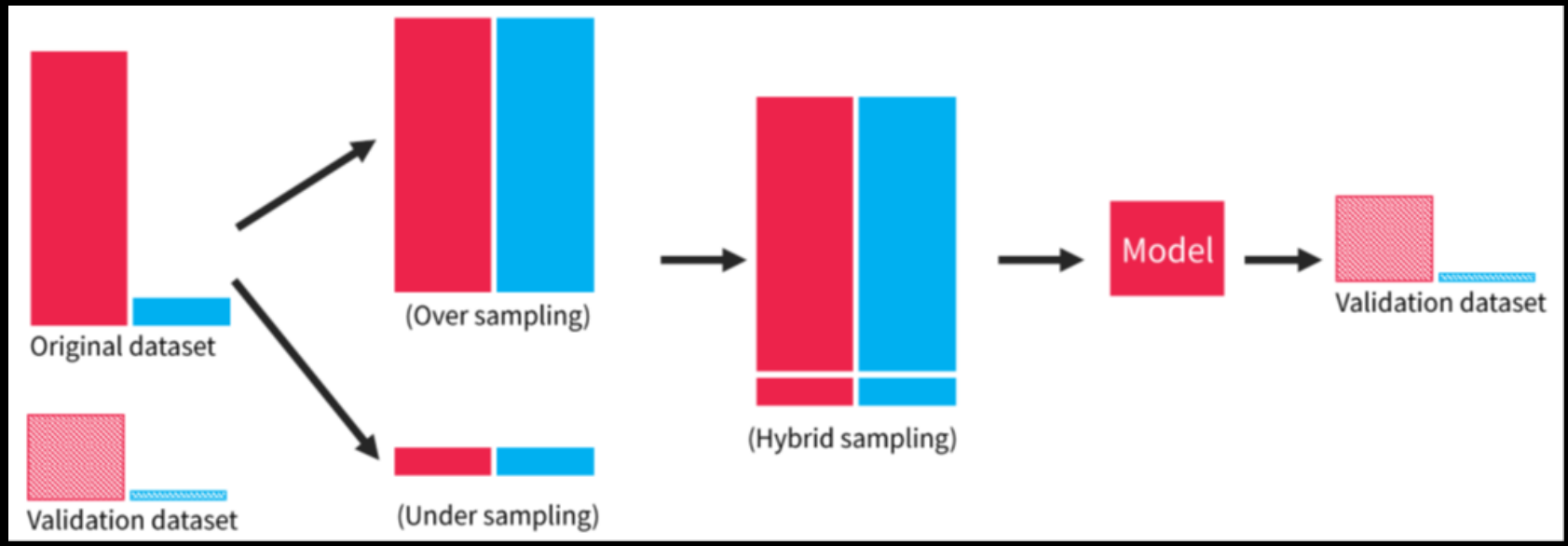
# Hybrid Resampling

## Hybrid Resampling

- A method that combines over-sampling and under-sampling
- Commonly uses a combination of SMOTE + Tomek Links

SMOTE: Performs over-sampling on minority class data

Tomek Links: Performs under-sampling by removing majority class



# Play with Examples

Codes:

https://www.deepshark.org/courses/machine_learning_2/w/09_imbalanced_data_SMOTE


Thank you!