# Imbalance Problem in Machine Learning

Shreya Sharma

# Contents

# What is imbalance problem?

- An imbalance problem with respect to an input property occurs when the distribution of that property affects the model performance. For example, imbalance in classes can make the model biased towards majority class
- If not addressed, imbalance causes adverse effect on the final detection performance
- Types of imbalance problems
  - Class – number of labels → Classification
  - Scale – object sizes
  - Spatial – object locations
  - Objective – contribution of different tasks to overall loss

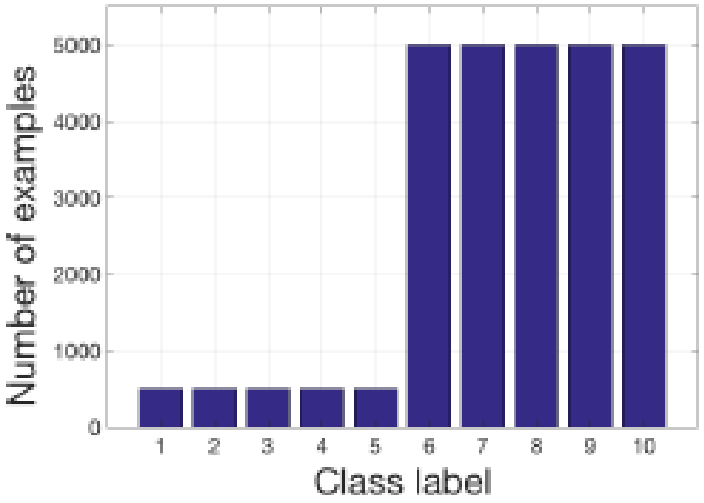
Object Detection

# Forms of Imbalance

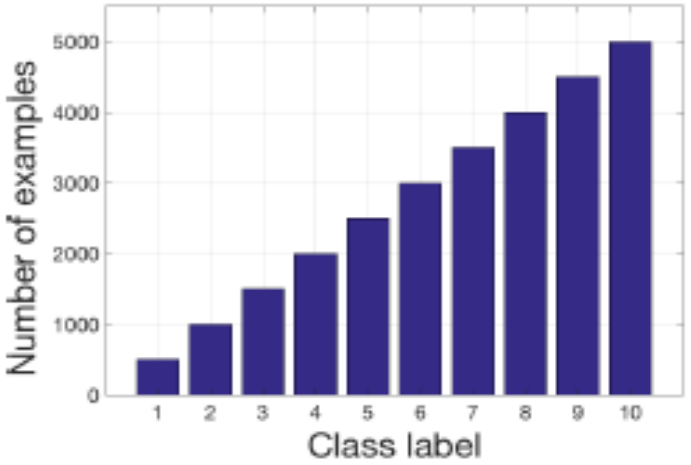
- Step Imbalance
  - Number of examples is equal within minority classes and equal within majority classes but differs between majority and minority classes
  - Defined by 2 parameters
    1. Fraction of minority class (μ)

$$\mu = \frac{|\{i \in \{1, \ldots, N\} : C_i \text{ is minority}\}|}{N},$$

    2. Ratio between number of examples in majority class to minority class (ρ)
- Linear Imbalance
  - Number of examples vary linearly
  - Defined by parameter ρ



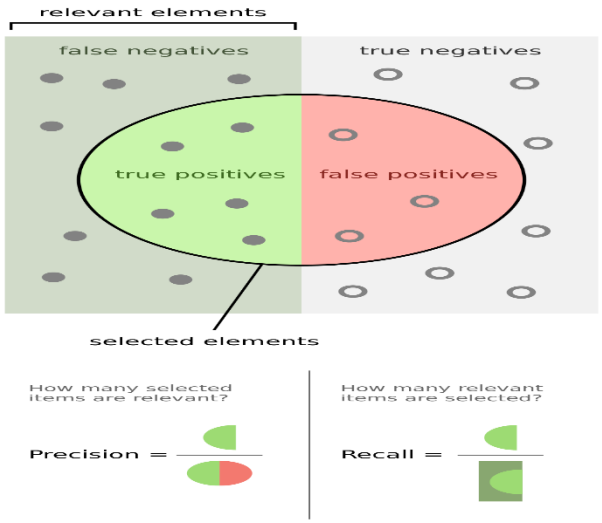

$(\mathrm{a})\ \rho = 10, \mu = 0.5$

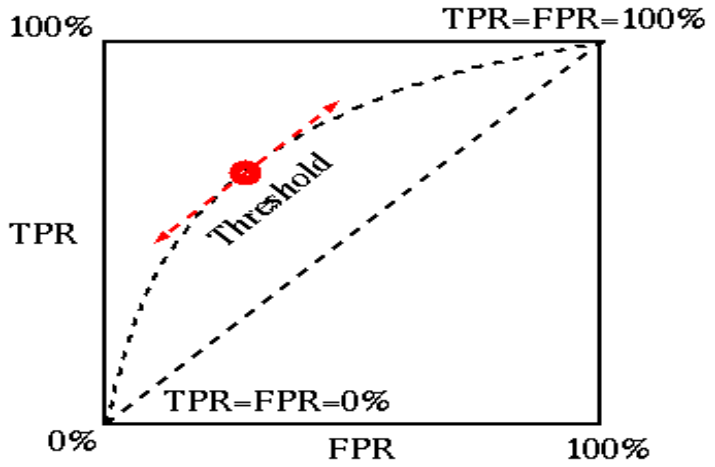



$(\mathrm{c})\ \rho = 10$

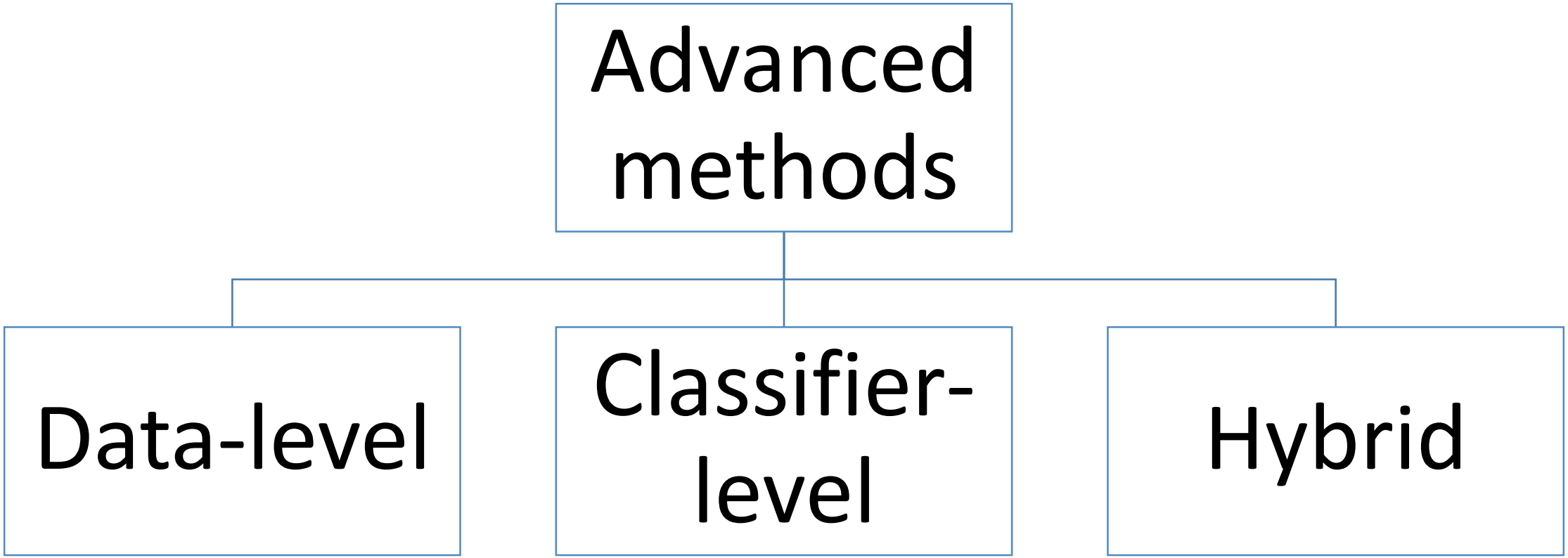
# Initial methods to handle imbalance

- Collect more data
- Change evaluation metric from accuracy to f-measure and ROC-AUC



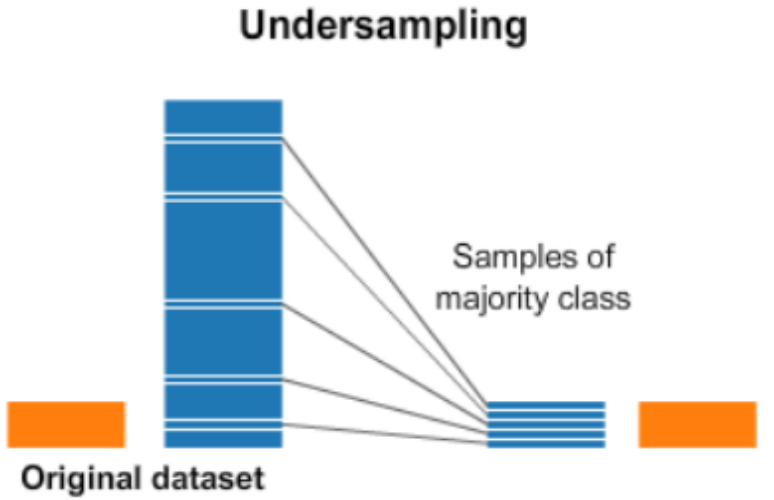



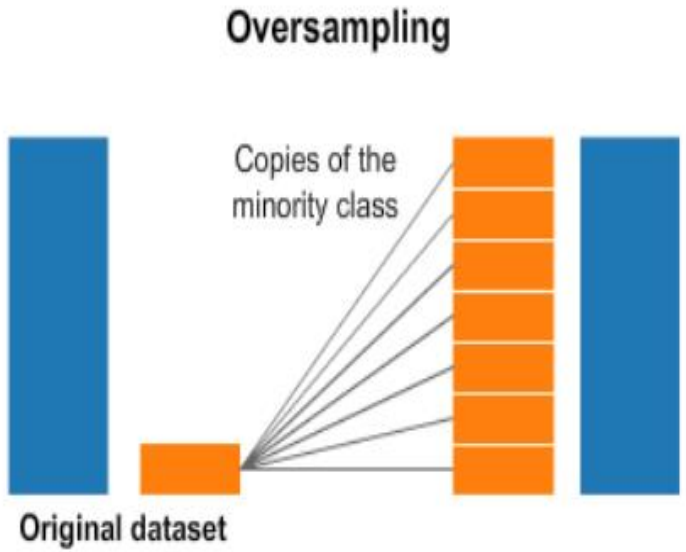

- Try tree-based models or boosting method
  - Decision tree perform well on imbalanced data. It learns a hierarchy of if/else questions that force both classes to be addressed
  - XGBoost implicitly balances the data before classification


Advanced methods
Data-level
Classifier-level
Hybrid

# Data-level Methods

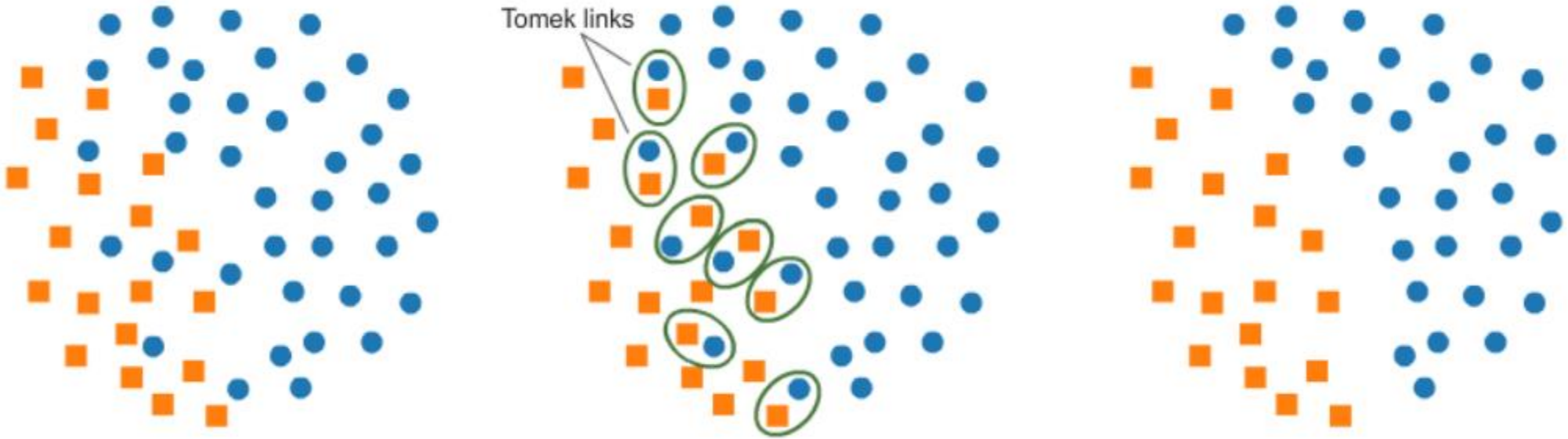
- Undersampling
  - Random majority undersampling
  - Tomek Links
  - Clustering-based
  - Removing data redundancy
- Oversampling
  - Random minority oversampling
  - SMOTE
  - Modified SMOTE
  - Adasyn
  - Clustering-based
  - Data Boost IM
  - Class-aware sampling
  - Data Augmentation of minority class



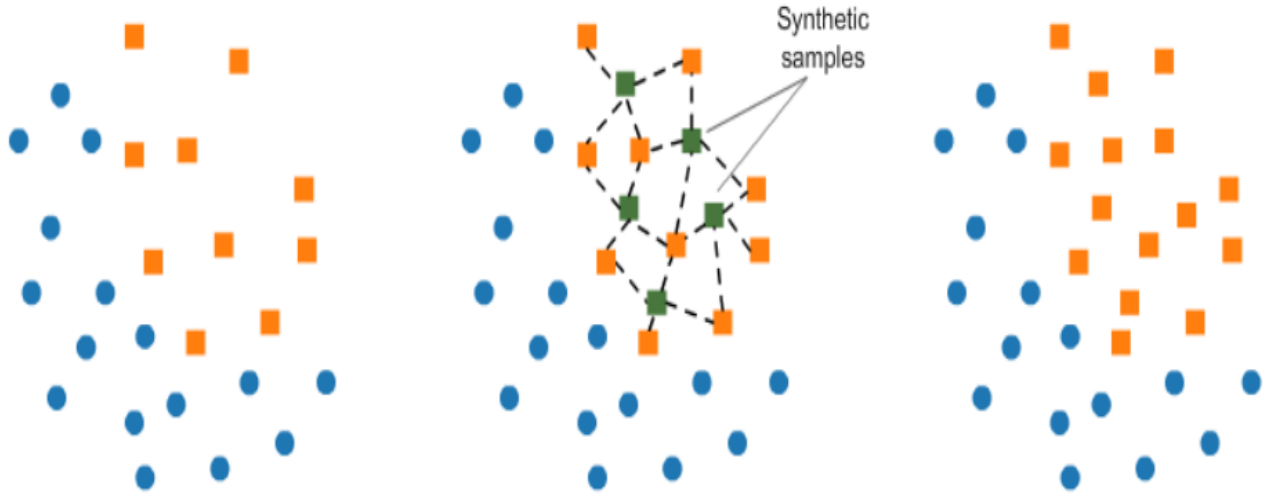
# Tomek Links

- Pairs of very close instances, but of opposite classes. Removing the instances of the majority class of each pair increases the space between two classes, facilitating the classification process



# SMOTE

- Synthetic Minority Oversampling Technique
- Synthesize elements for the minority class, based on those that already exist. It works by randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbors
- Drawbacks
  - While generating s
    other classes. This
    additional noise
  - Not very effective




s from
troduce

# Cluster-based oversampling

- first clusters the dataset and then oversamples each cluster separately.
- In this case, the K-means clustering algorithm is independently applied to minority and majority class instances to identify clusters in the dataset.
- Subsequently, each cluster is oversampled such that all clusters of the same class have an equal number of instances and all classes have the same size.

# Classifier-level

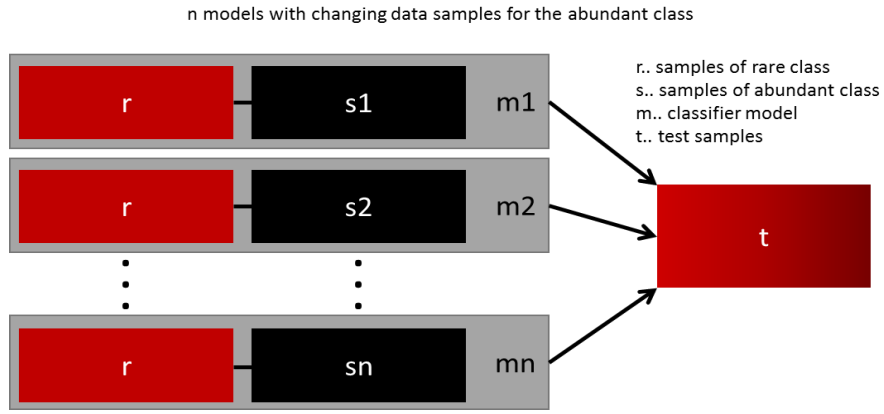
- Weighing of loss functions
  - Inverse class frequency
  - Focal loss
  - Class-balanced loss
- Thresholding
  - Applied in test phase and involves changing the output class probabilities
  - Correct class probability by dividing the network output of each class by its prior prob.
- Cost-sensitive learning
  - Assign different cost to misclassified samples from different classes
  - Harder examples are given higher cost
  - Modify the learning rate such that the higher cost examples contribute more to the update of weights
  - Train the network by minimizing the misclassification cost instead of the standard loss function
  - Result similar to oversampling
  - Circular learning approach: first train the classifier on easy examples and then on hard examples

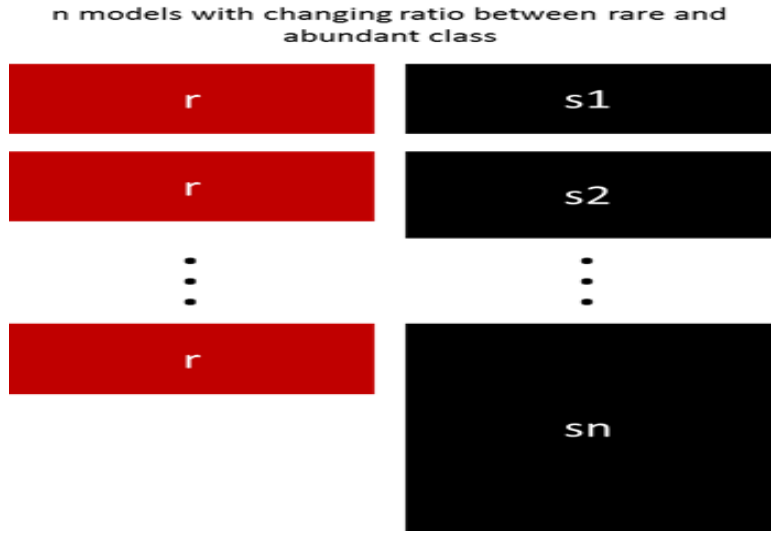
# Classifier-level

- Transfer learning
  - Design a model (meta-learner) to learn how the model evolves as the size of training data increases
  - The meta-learner model is trained gradually by increasing the number of examples from the classes with the large number of examples.
  - The resulting meta-model is able to transform another model trained with less examples to a model trained with more examples which makes it useful to be exploited by an underrepresented class
- One-class classification
  - From the perspective of change detection or anomaly detection
  - Concept learing technique that recognized positive instances rather than discriminating between two classes
  - Common technique is to use autoencoders to learn identity mapping
  - Works well for extremely high imbalance
- Weakly supervised learning
  - using unlabeled samples whose labels are generated using a classifier or manually. Noisy labels can be used

# Hybrid

- Ensembling
  - With different resampled datasets- same ratio
  - With different resampled datasets- different ratio
- EasyEnsemble and BalanceCascade
  - train a committee of classifiers on undersampled subsets
- SMOTEBoost
  - boosting + SMOTE
- Two-level training
  - pre-training on balanced dataset and Then fine-tuning the last output layer before softmax on the original imbalanced data



# Comparison

- Oversampling – best in deep learning
- Undersampling – not suitable especially when number of minority classes are less. When minority classes are high, oversampling and underdamping becomes equivalent
- Cost-sensitive learning- performs similar to oversampling
- Thresholding- worked particularly well when applied jointly with oversampling. Please note that thresholding does not have an actual effect on the ability of the classifier to discriminate between a given class from another but rather helps to find a threshold on the network output that guarantees a large number of correctly classified cases.
- Ensembling – huge computational time so not practical
- One-class method- works in case of extremely high imbalance
- Two-phase training – with oversampling is better than baseline

# Common Pitfalls

- Train-test split
  - Always split into test and train sets BEFORE trying any resampling techniques! Oversampling before splitting the data can allow the exact same observations to be present in both the test and train sets! This can allow our model to simply memorize specific data points and cause overfitting.
- K-fold cross-validation
  - Keep in mind that over-sampling takes observed rare samples and applies bootstrapping to generate new random data based on a distribution function. If cross-validation is applied after over-sampling, basically what we are doing is overfitting our model to a specific artificial bootstrapping result. That is why cross-validation should always be done before over-sampling the data, just as how feature selection should be implemented. Only by resampling the data repeatedly, randomness can be introduced into the dataset to make sure that there won't be an overfitting problem.

# Deep Learning

- Adjust class-weights
- Increase batch size to ensure each batch has sufficient samples from all classes
- Use precision, recall, f-measure, ROC as evaluation metric

# Initial Things to try

- Get more data
- Change evaluation metric
- Try tree-based method

# Code

- Sklearn.utils resample()
  - https://scikit-learn.org/stable/modules/generated/sklearn.utils.resample.html
- Imbalanced-learn library
  - https://imbalanced-learn.readthedocs.io/en/stable/

# References


- https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets
- https://www.curiousily.com/posts/practical-guide-to-handling-imbalanced-datasets/
- https://www.ele.uri.edu/faculty/he/PDFfiles/ImbalancedLearning.pdf
- https://www.tensorflow.org/tutorials/structured_data/imbalanced_data
- https://colab.research.google.com/drive/1xL2jSdY-MGlN60gGuSH_L30P7kxxwUfM
- https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html
- https://www.dlology.com/blog/multi-class-classification-with-focal-loss-for-imbalanced-datasets/
- https://www.kaggle.com/tboyle10/methods-for-dealing-with-imbalanced-data
- https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/
- https://arxiv.org/pdf/1909.00169.pdf