CS145: INTRODUCTION TO DATA MINING

08: Classification Evaluation and Practical Issues

Instructor: Yizhou Sun

yzsun@cs.ucla.edu

February 4, 2019

Learnt Prediction and Classification Methods

	Vector Data	Set Data	Sequence Data	Text Data
Classification	**Logistic Regression; Decision Tree; KNN SVM; NN**			Naïve Bayes for Text
Clustering	K-means; hierarchical clustering; DBSCAN; Mixture Models			PLSA
Prediction	**Linear Regression** GLM*			
Frequent Pattern Mining		Apriori; FP growth	GSP; PrefixSpan	
Similarity Search			DTW	

Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Other issues
- Summary

Model Evaluation and Selection

- Evaluation metrics: How can we measure accuracy? Other metrics to consider?

- Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy

- Methods for estimating a classifier's accuracy:

  - Holdout method, random subsampling

  - Cross-validation

Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

- **Cross-validation** (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - Leave-one-out: *k* folds where *k* = # of tuples, for small sized data
  - *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the whole data

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

- Given m classes, an entry, $\textbf{\textit{CM}}_{i,j}$ in a **confusion matrix** indicates # of tuples in class i that were labeled by the classifier as class j
- May have extra rows/columns to provide totals

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Classifier Accuracy,** or recognition rate: percentage of test set tuples that are correctly classified

$$\text{Accuracy} = (\text{TP} + \text{TN})/\text{All}$$

- **Error rate:** *1 – accuracy*, or

$$\text{Error rate} = (\text{FP} + \text{FN})/\text{All}$$

- **Class Imbalance Problem**:
  - One class may be *rare*, e.g. fraud, or HIV-positive
  - Significant *majority of the negative class* and minority of the positive class
  - **Sensitivity**: True Positive recognition rate
    - **Sensitivity = TP/P**
  - **Specificity**: True Negative recognition rate
    - **Specificity = TN/N**

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive

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

- **Recall:** completeness – what % of positive tuples did the classifier label as positive?

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

- Perfect score is 1.0
- Inverse relationship between precision & recall
- *F measure (F$_1$ or F-score)*: harmonic mean of precision and recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

- *F$_\beta$:* weighted measure of precision and recall
  - assigns ß times as much weight to recall as to precision

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

Classifier Evaluation Metrics: Example

| Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity*)
cancer = no	**140**	**9560**	9700	98.56 (*specificity*)
Total	230	9770	10000	96.50 (*accuracy*)

- *Precision = 90/230 = 39.13%*   *Recall = 90/300 = 30.00%*

Classifier Evaluation Metrics: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- Area under the curve: the closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model
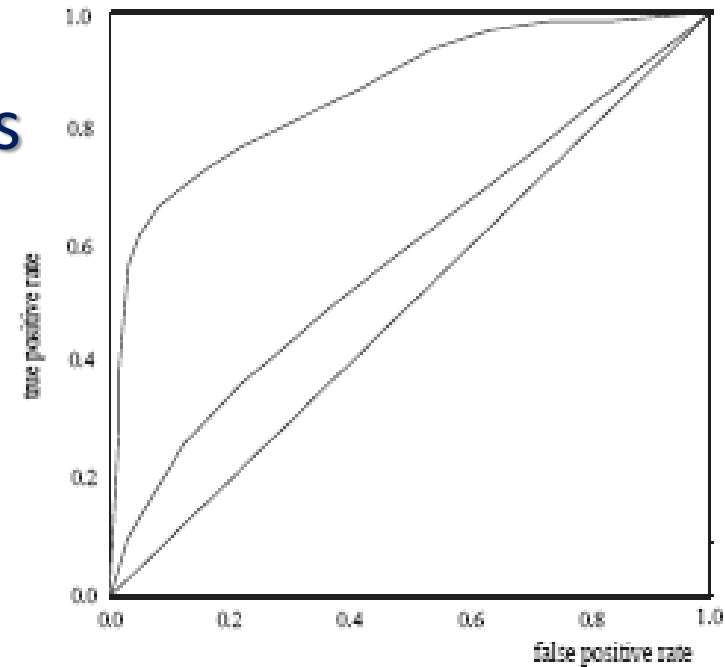
- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

Plotting an ROC Curve

- True positive rate: $TPR = TP/P$ (sensitivity)
- False positive rate: $FPR = FP/N$ (1-specificity)

- Rank tuples according to how likely they will be a positive tuple
  - Idea: when we include more tuples in, we are more likely to make mistakes, that is the trade-off!
  - Nice property: not threshold (cut-off) need to be specified, only rank matters

Example:

- #P = 5
- #N = 5

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	p	0.9	1	0	5	4	0.2	0
2	p	0.8	2	0	5	3	0.4	0
3	n	0.7	2	1	4	3	0.4	0.2
4	p	0.6	3	1	4	2	0.6	0.2
5	p	0.55	4	1	4	1	0.8	0.2
6	n	0.54	4	2	3	1	0.8	0.4
7	n	0.53	4	3	2	1	0.8	0.6
8	n	0.51	4	4	1	1	0.8	0.8
9	p	0.50	5	4	0	1	1.0	0.8
10	n	0.4	5	5	0	0	1.0	1.0
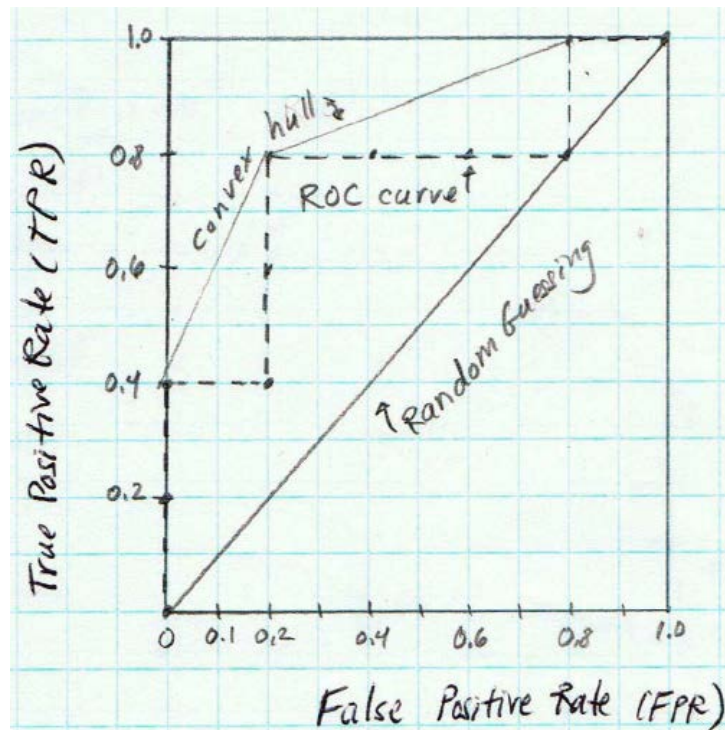
12

Evaluation and Other Practical Issues

- Model Evaluation and Selection

- Other issues ⬅

- Summary

Multiclass Classification

- Multiclass classification
  - Classification involving more than two classes (i.e., > 2 Classes)
  - Each data point can only belong to one class

- Multilabel classification
  - Classification involving more than two classes (i.e., > 2 Classes)
  - Each data point can belong to multiple classes
  - Can be considered as a set of binary classification problem

Solutions

- Method 1. **One-vs.-all** (OVA): Learn a classifier one at a time
  - Given m classes, train m classifiers: one for each class
  - Classifier j: treat tuples in class j as *positive* & all others as *negative*
  - To classify a tuple **X**, choose the classifier with maximum value
- Method 2. **All-vs.-all** (AVA): Learn a classifier for each pair of classes
  - Given m classes, construct m(m-1)/2 binary classifiers
  - A classifier is trained using tuples of the two classes
  - To classify a tuple **X**, each classifier votes. **X** is assigned to the class with maximal vote
- Comparison
  - All-vs.-all tends to be superior to one-vs.-all

Illustration of One-vs-All

One-vs-all (one-vs-rest):
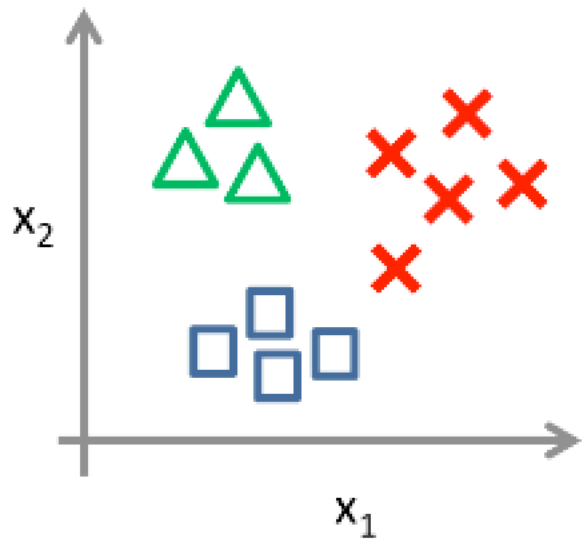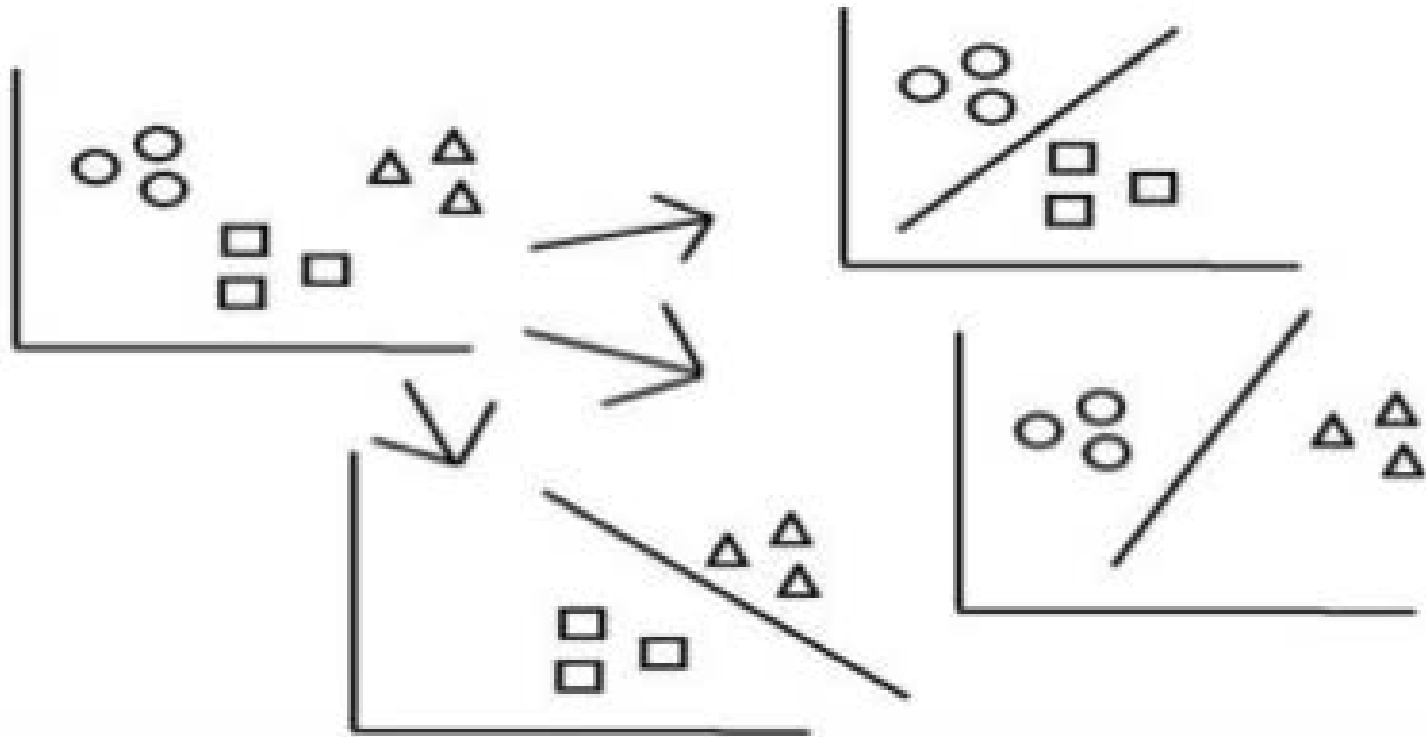

Class 1: △
Class 2: □
Class 3: ✖
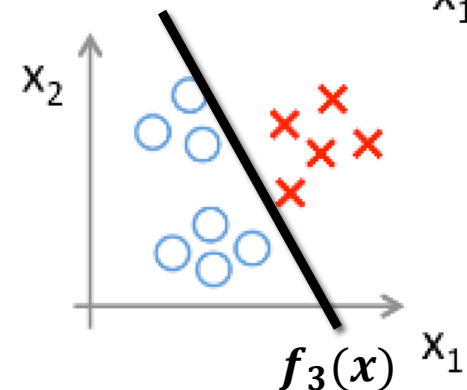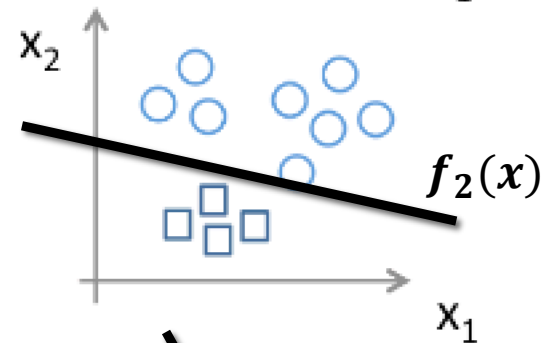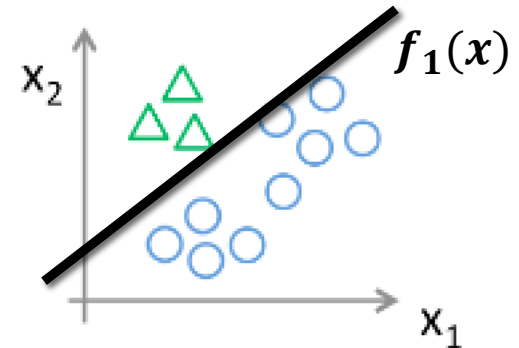
Classify x according to: $f(x) = argmax_i f_i(x)$

Illustration of All-vs-All

Classify x according to majority voting

Extending to Multiclass Classification Directly

- Very straightforward for
  - Logistic Regression
  - Decision Tree
  - Neural Network
  - KNN

Classification of Class-Imbalanced Data Sets

- Class-imbalance problem
  - Rare positive example but numerous negative ones, e.g., medical diagnosis, fraud, oil-spill, fault, etc.
- Traditional methods
  - Assume a balanced distribution of classes and equal error costs: not suitable for class-imbalanced data

Balanced dataset

Imbalanced dataset

How about predicting every data point as blue class?
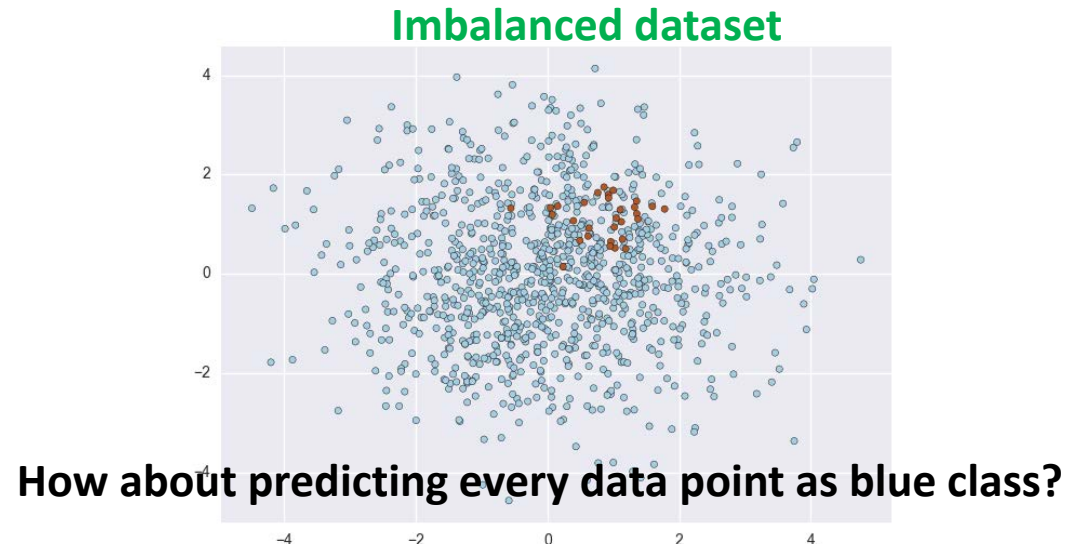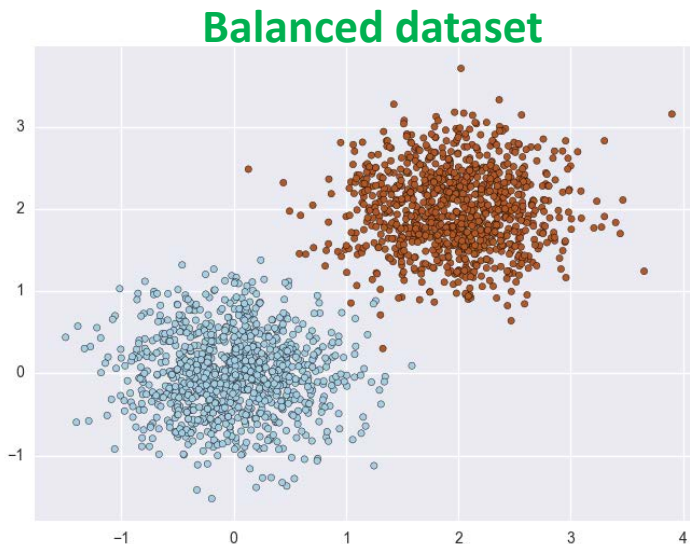
Solutions

- Pick the right evaluation metric
  - E.g., ROC is better than accuracy
- Typical methods for imbalance data in 2-class classification (training data):
  - **Oversampling**: re-sampling of data from positive class
  - **Under-sampling**: randomly eliminate tuples from negative class
  - **Synthesizing new data points** for minority class
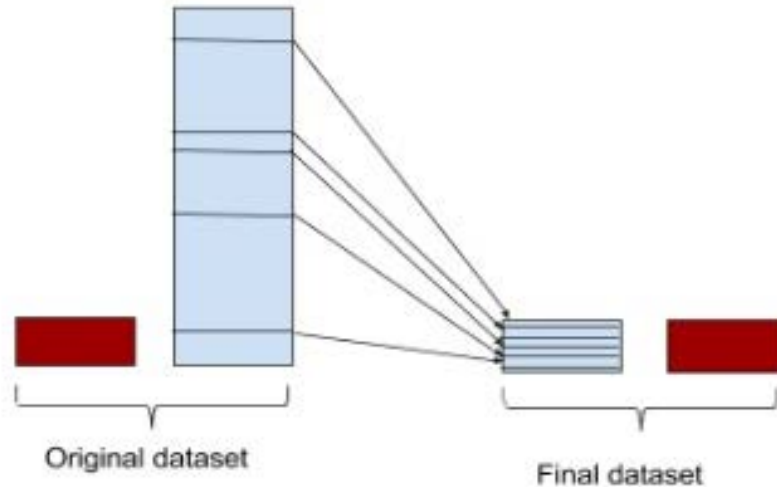- Still difficult for class imbalance problem on multiclass tasks

https://svds.com/learning-imbalanced-classes/

Illustration of Oversampling and Undersampling

Oversampling minority class

Original dataset

Final dataset
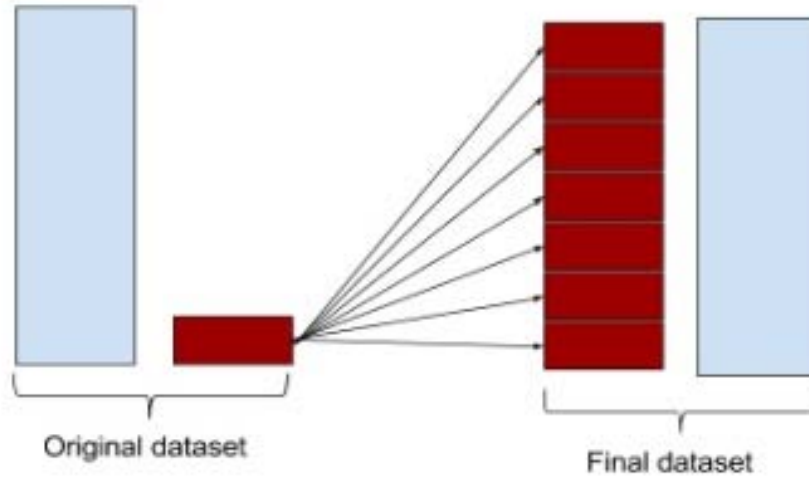
Undersampling majority class

Original dataset

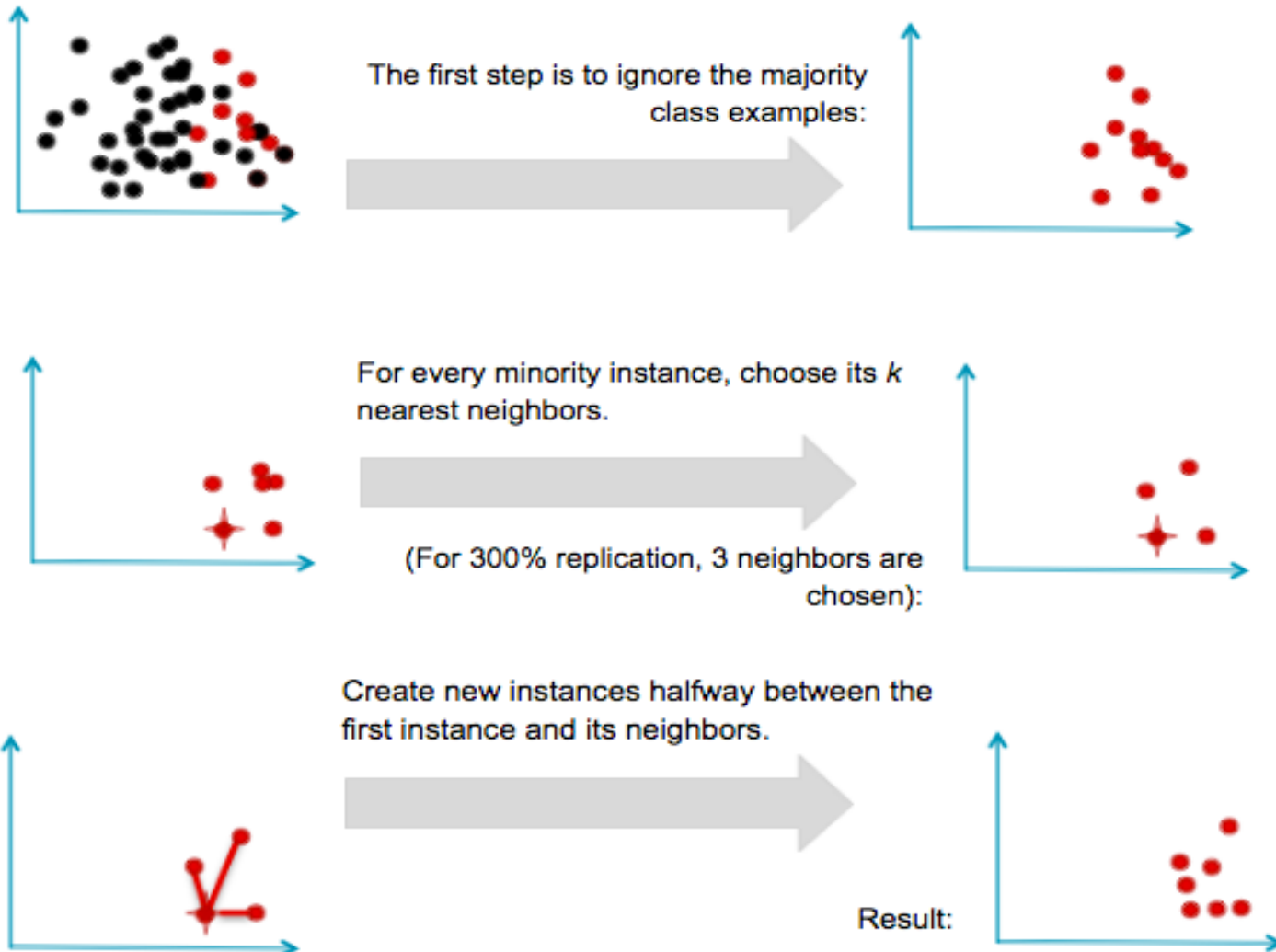Final dataset

Illustration of Synthesizing New Data Points

- SMOTE: Synthetic Minority Oversampling Technique (Chawla et. al)

The first step is to ignore the majority class examples:

For every minority instance, choose its k nearest neighbors.

(For 300% replication, 3 neighbors are chosen):

Create new instances halfway between the first instance and its neighbors.

Result:

Evaluation and Other Practical Issues

- Model Evaluation and Selection

- Other issues

- Summary ⬅

Summary

- Model evaluation and selection
  - Evaluation metric and cross-validation
- Other issues
  - Multi-class classification
  - Imbalanced classes