Class Imbalance

Problem Motivation

- Classification datasets can be "imbalanced"
    - i.e. many observations of one class, few of another
- Costs of false positive different from cost of false negative

    - e.g. missing fraud can be more costly than screening legitimate activity
- Accuracy-driven models will over-predict the majority class

Solutions

- Cost-sensitive learning
  - thresholding (aka "profit curves")
  - modified objective functions
- Sampling
  - Oversampling
  - Undersampling
  - SMOTE - Synthetic Minority Oversampling TEchnique

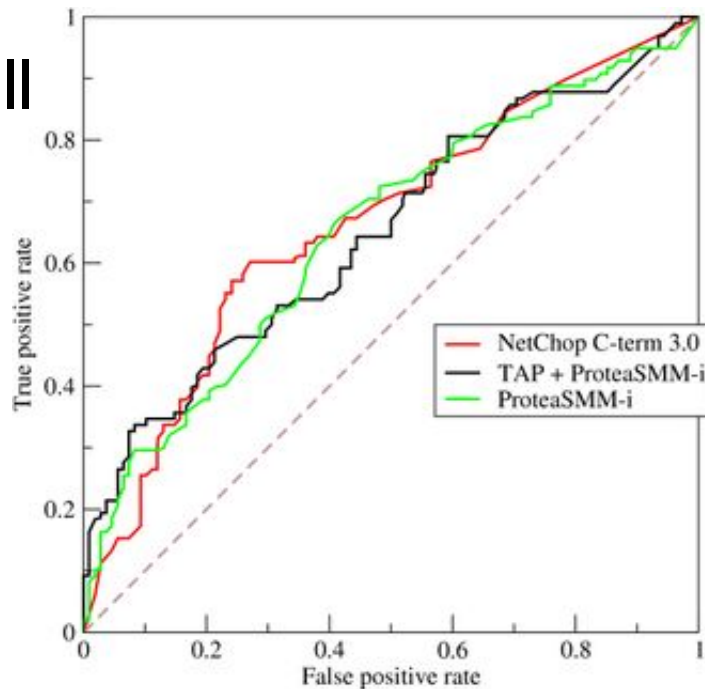# Cost-sensitive Learning - Thresholding

Cost-sensitive Learning - Thresholding

Recall the ROC Curve:

- ROC shows precision vs recall
- doesn't give preference to one over the other

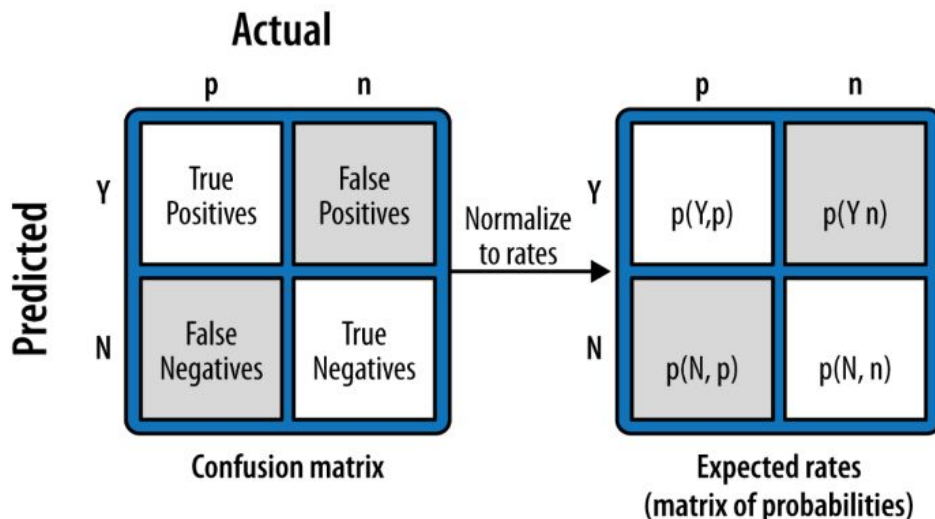**Q:** How to handle unequal error costs?

A: Plot expected profit

Cost-sensitive Learning - Thresholding

Computing Expected Profit

Step 1 - Compute error probabilities

Computing Expected Profit

Step 2 - Determine error costs and benefits

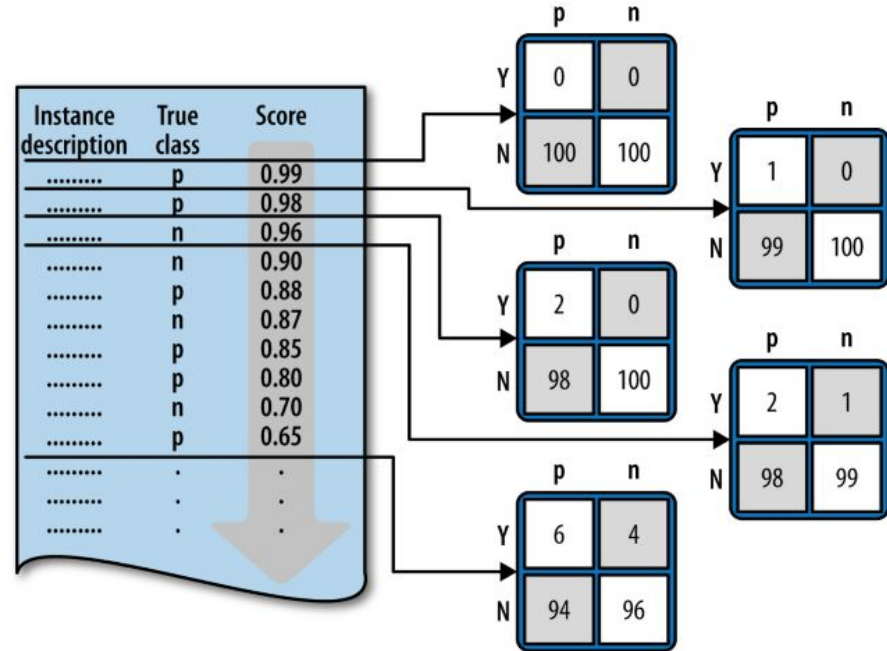# Cost-sensitive Learning - Thresholding

Computing Expected Profit

Step 3 - Combine probabilities and payoffs

$$
\begin{aligned}
E[Profit] \;=\; & P(Y,p) \cdot b(Y,p) + P(Y,n) \cdot c(Y,n) + \\
& P(N,p) \cdot c(N,p) + P(N,n) \cdot b(N,n) \\[6pt]
\;=\; & P(Y|p) \cdot P(p) \cdot b(Y,p) + P(Y|n) \cdot P(n) \cdot c(Y,n) + \\
& P(N|p) \cdot P(p) \cdot c(N,p) + P(N|n) \cdot P(n) \cdot b(N,n) \\[6pt]
\;=\; & P(p) \cdot [P(Y|p) \cdot b(Y,p) + P(N|p) \cdot c(N,p)] + \\
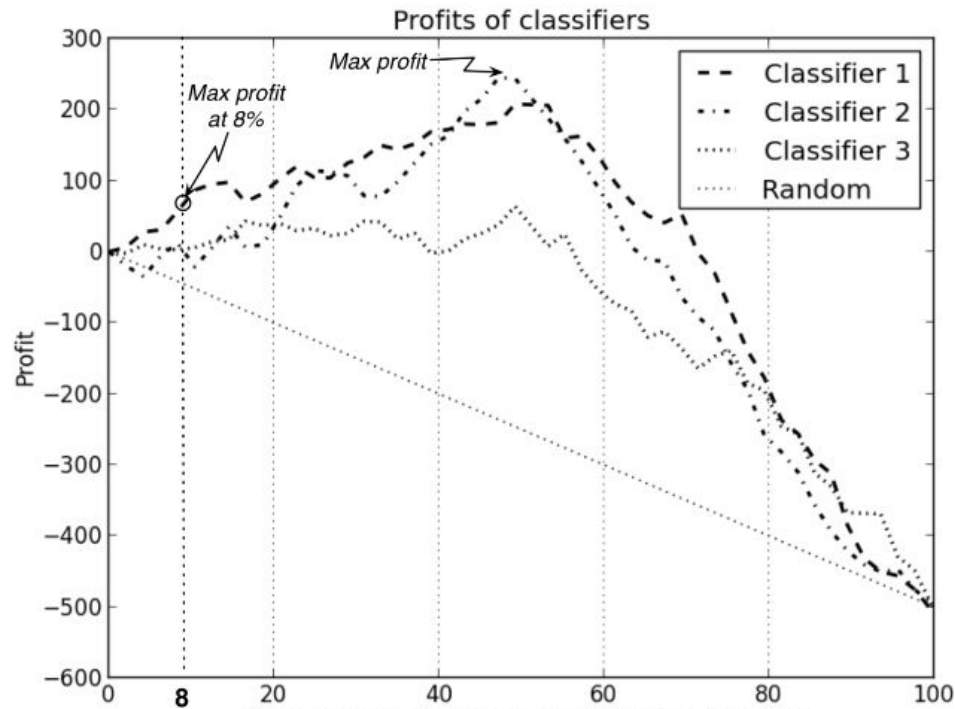& P(n) \cdot [P(Y|n) \cdot c(Y,n) + P(N|n) \cdot b(N,n)]
\end{aligned}
$$

Cost-sensitive Learning - Thresholding

Find the profit-maximizing threshold

- For each possible threshold, compute expected profit
- Then select threshold with highest expected profit

Cost-sensitive Learning - Thresholding

Profits of classifiers

Cost-sensitive Learning
Modified Objective Functions

- models with explicit cost function can be modified to incorporate classification cost e.g. logistic regression
- can affect optimization
  - e.g. cost-sensitive logistic regression is not convex
- not all models have a cost-sensitive implementation

Sampling Techniques

Sampling Techniques - Undersampling

- Undersampling randomly discards majority class observations to balance training sample
- PRO:
  Reduces runtime on very large datasets
- CON:
  Discards potentially important observations

Sampling Techniques - Oversampling

- Oversampling replicates observations from minority class to balance training sample
- PRO:
  Doesn't discard information
- CON:
  Likely to overfit
- Often better to use SMOTE

Sampling Techniques - SMOTE

- SMOTE - <u>S</u>ynthetic <u>M</u>inority <u>O</u>versampling <u>TE</u>chnique
- Generates new observations from minority class

Sampling Techniques - SMOTE

- For each minority class observation and for each feature, randomly generate between it and any/all of its k-nearest neighbors
- Can be combined with undersampling and other techniques
- See also SMOTEBoosting and SMOTEBagging

Sampling Techniques - SMOTE

SMOTE pseudocode

```python
synthetic_observations = []
while len(synthetic_observations) + len(minority_observations) < target:
    obs = random.choice(minority_observations):
    neighbor = random.choice(kNN(obs, k)) # randomly selected neighbor
    new_observation = {}
    for feature in obs:
        weight = random() # random float between 0 and 1
        new_feature_value = weight*obs[feature] \
                            + (1-weight)*neighbor[feature]
        new_observation[feature] = new_feature_value
    synthetic_observations.append(new_observation)
```

See also psuedo-code in original SMOTE paper: https://www.jair.org/media/953/live-953-2037-jair.pdf

Sampling Techniques - Distribution

What's the right amount of over-/under-sampling?

- If you know the cost matrix:
  - If CTP = CTN = 0,
    set target proportion = P(N) / P(P) = C(FP) / C(FN)
  - Can maximize profit curve over target proportion
- If you don't know the cost matrix:
  - No clear answer
  - ROC plot's AUC may be more useful

Cost Sensitivity vs Sampling

- Neither is strictly superior
- Oversampling tends to work better than undersampling on small datasets
- Some algorithms don't have an obvious cost-sensitive adaptation, requiring sampling

See also "Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?" http://storm.cis.fordham.edu/gweiss/papers/dmin07-weiss.pdf

Evaluation Metrics Review

Evaluating a Binary Classifier

Confusion Matrix

	Predicted Positive	Predicted Negative
Actually Positive	True Positives	False Negatives
Actually Negative	False Positives	True Negatives

Classifier Metrics

Accuracy

$$\frac{TP + TN}{n}$$

True Positive Rate (Sensitivity/Recall)

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

True Negative Rate (Specificity)

$$\frac{TN}{N} = \frac{TN}{TN + FP}$$

Precision

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

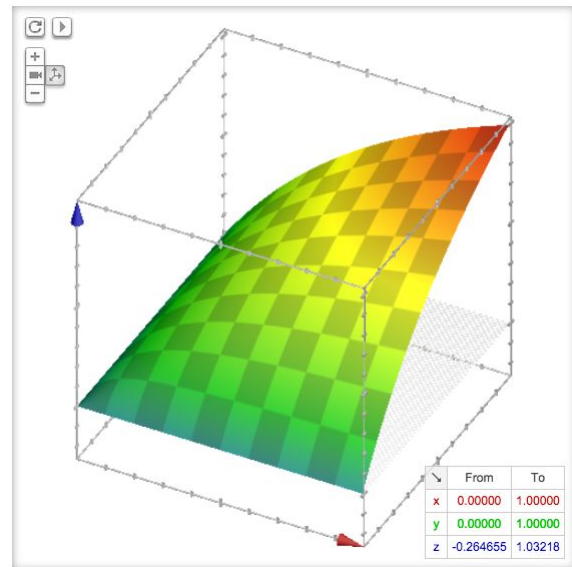# Evaluating a Binary Classifier

F1 Score

- harmonic mean of precision and recall

$$F_1 = \frac{2 * precision * recall}{precision + recall} = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

F$_\beta$ Score

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

	From	To
x	0.00000	1.00000
y	0.00000	1.00000
z	-0.264655	1.03218

Evaluating a Binary Classifier

ROC Plot

- If classifier A's ROC curve is strictly greater than classifier B's, then classifier A is always preferred
- If two classifier's ROC curves intersect, then the choice depends on relative importance of sensitivity and specificity
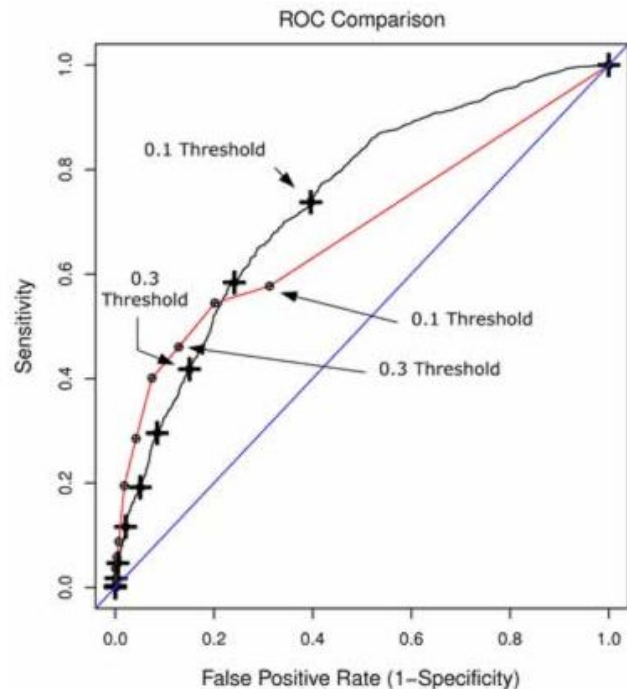
Evaluating a Binary Classifier

ROC - Area Under Curve (AUC)

- equals the probability that the model will rank a randomly chosen positive observation higher than a randomly chosen negative observation
- useful for comparing different classes of models in general setting