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 is more costly than screening legitimate activity
- Accuracy-driven models will over-predict the majority class

2 Basic Solutions

- Sampling
  - Oversampling
  - Undersampling
  - SMOTE - Synthetic Minority Oversampling TEchnique
- Cost-sensitive learning
  - modify cost function to account for different costs of false positives and false negatives

Sampling Techniques - Undersampling

- Undersampling randomly selects subset of majority class to balance training sample
- Can train multiple classifiers across many samples and ensemble their output
- Reduces runtime on very large datasets
- Discards potentially important observations

Sampling Techniques - Oversampling

- Oversampling replicates observations from minority class to balance training sample
- Can cause overfitting
- Doesn’t discard information
- Often better to use SMOTE

Sampling Techniques - SMOTE

- SMOTE - Synthetic Minority Oversampling TEchnique
- Generates new observations from minority class

Sampling Techniques - SMOTE

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

Original 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 target goal is cost minimization, set ratio = CFP / CFN
- Can cross-validate to optimize metric of choice (e.g. Fbeta score)

Cost-sensitive Learning

- Can modify cost function to represent real world costs/benefits
- Use cost matrix representing costs of True/False Positive/Negative
- Makes business goal explicit in ML logic

Cost-sensitive Learning - Thresholding

- Typical threshold for positive classification is $h = .5$
- Instead select threshold to minimize expected cost
- For trees, this is done at each node
- Can plot “profit curve” which shows expected profit/loss over classification threshold
- See earlier lecture on this topic
- (Think about the equivalency with sampling!)

Cost-sensitive Logistic Regression

- Logistic regression's usual objective function:

$$\ln p(\vec{y}|X;\theta) = \sum_{i=1}^{n} \left(y_i \ln h_\theta(x_i) + (1 - y_i)\ln(1 - h_\theta(x_i))\right)$$

- New objective function, representing expected cost:

$$J^c(\theta) = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i(h_\theta(X_i)C_{TP_i} + (1 - h_\theta(X_i))C_{FN_i})$$
$$+(1 - y_i)(h_\theta(X_i)C_{FP_i} + (1 - h_\theta(X_i))C_{TN_i})\Big).$$

Note that the cost-sensitive objective function is not convex!

Cost-sensitive Logistic Regression

- Logistic regression goal: accurately estimate parameter of Bernoulli random variables
- Cost-sensitive logistic regression goal: minimize misclassification cost
- Both assume that observations are Bernoulli

Cost Sensitivity vs Sampling

- Neither is strictly superior
- Oversampling tends to work well on small datasets
- Some algorithms don’t have an obvious cost-sensitive adaptation, requiring sampling
- Methods can be combined (e.g. thresholding and SMOTE)

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

Cost-sensitivity vs Sampling

- Equivalency between cost-sensitivity and sampling based approaches
    - e.g. 2x oversampling is same as doubling misclassification cost using thresholding method

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

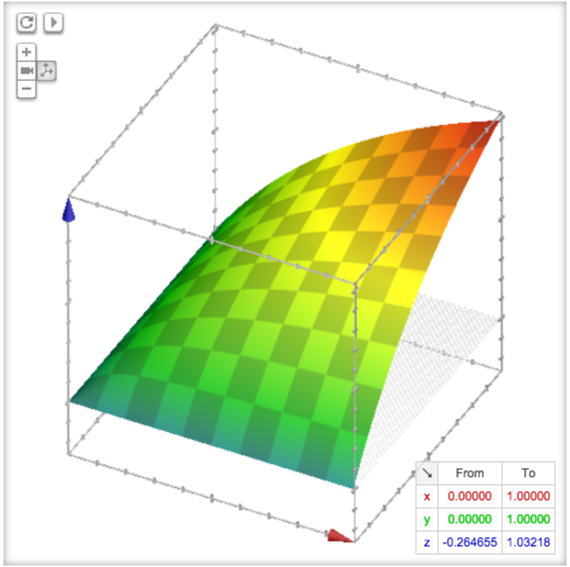
F1 Score

- harmonic mean of precision and recall

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

F_β Score

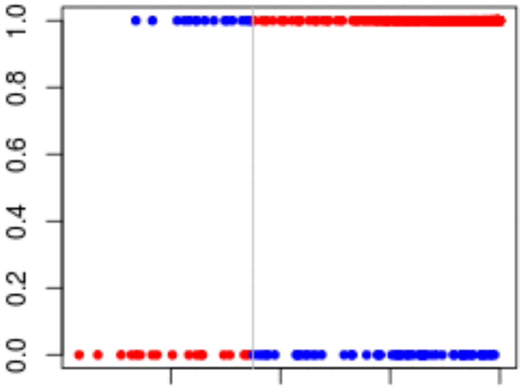
$$F_\beta = (1+\beta^2)\frac{precision * recall}{\beta^2 precision + recall}$$

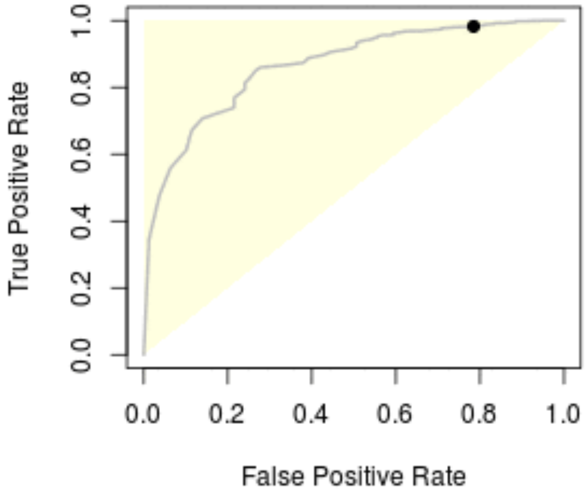


Evaluating a Binary Classifier

ROC Plot

- Shows how true and false positive rates vary as the decision boundary is moved (animation)

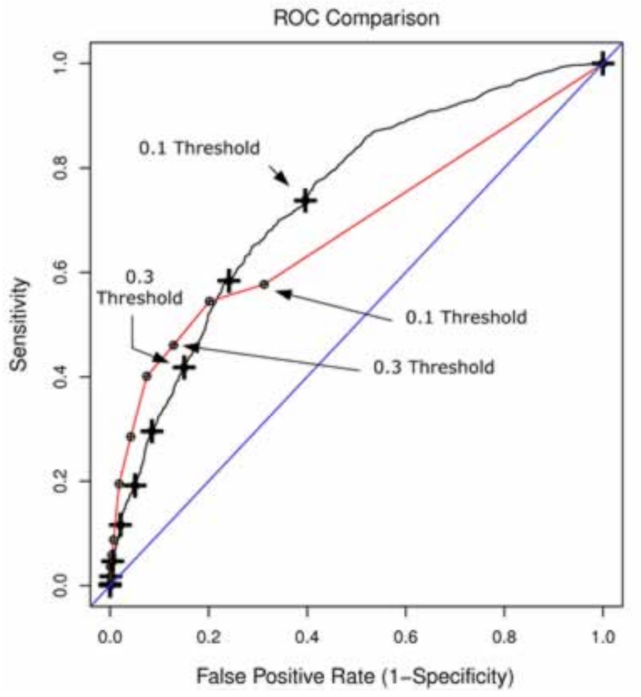




Evaluating a Binary Classifier

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