CS249: ADVANCED DATA MINING

Classification Evaluation and Practical Issues

Instructor: Yizhou Sun

yzsun@cs.ucla.edu

April 24, 2017

Announcements

- Homework 2 out
  - Due May 3rd (11:59pm)

- Course project proposal
  - Due May 8th (11:59pm)

Learnt Prediction and Classification Methods

	Vector Data	Text Data	Recommender System	Graph & Network
Classification	**Decision Tree**; **Naïve Bayes**; **Logistic Regression** **SVM; NN**			Label Propagation
Clustering	K-means; hierarchical clustering; DBSCAN; Mixture Models; kernel k-means	PLSA; LDA	Matrix Factorization	SCAN; Spectral Clustering
Prediction	**Linear Regression** **GLM**		Collaborative Filtering	
Ranking				PageRank
Feature Representation		Word embedding		Network embedding

Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Bias-Variance Trade-off
- 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

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, $CM_{i,j}$ 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

  Accuracy = (TP + TN)/All

- **Error rate:** *1 – accuracy*, or

  Error rate = (FP + FN)/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_β***: 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%

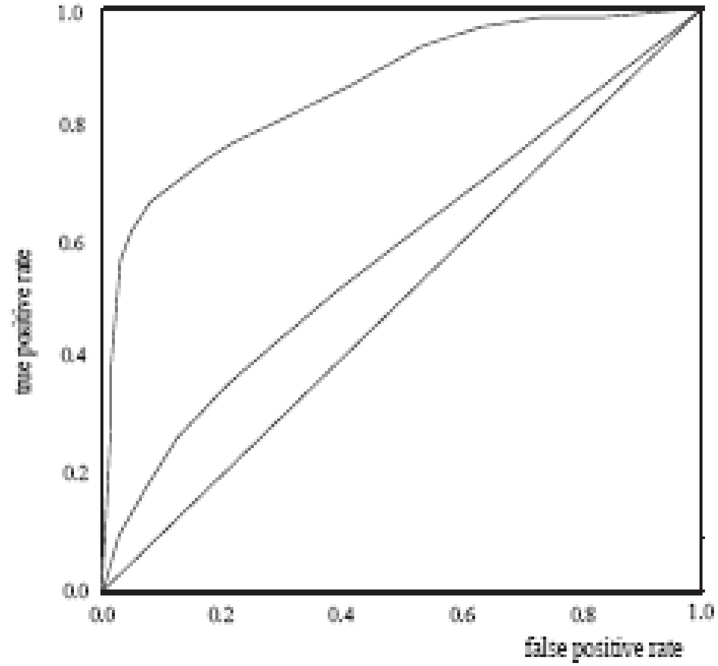
Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **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 initial data

Model Selection: 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

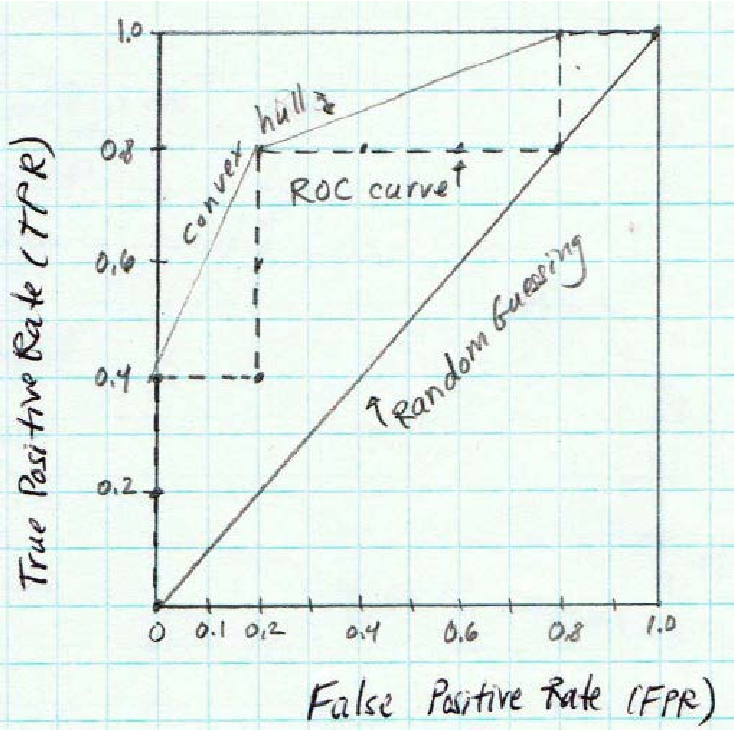
Plotting an ROC Curve

- 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

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

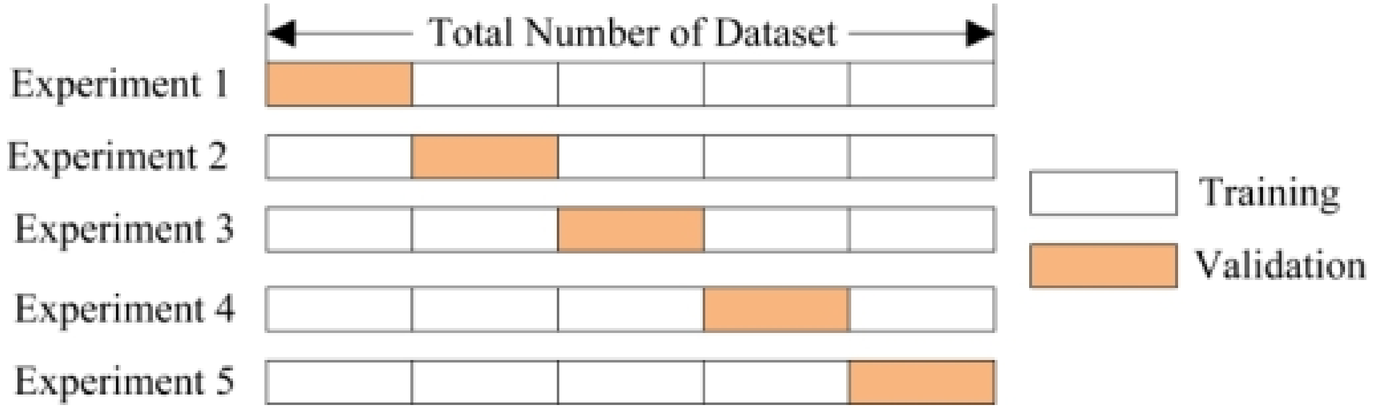
Example



- Similar for prediction tasks

Cross-Validation

- Partition the data into K folds
  - Use K-1 fold as training, and 1 fold as testing
  - Calculate the average accuracy best on K training-testing pairs
    - Accuracy on validation/test dataset!
      - **Mean square error** can again be used: $\sum_i \left(\boldsymbol{x}_i^T \widehat{\boldsymbol{\beta}} - y_i\right)^2 / n$



AIC & BIC

- AIC and BIC can be used to test the quality of statistical models
  - AIC (Akaike information criterion)
    - $AIC = 2k - 2\ln(\hat{L})$,
    - where k is the number of parameters in the model and $\hat{L}$ is the likelihood under the estimated parameter
  - BIC (Bayesian Information criterion)
    - $BIC = k\ln(n) - 2\ln(\hat{L})$,
    - Where n is the number of objects

Stepwise Feature Selection

- Avoid brute-force selection
  - 2^p
- Forward selection
  - Starting with the best single feature
  - Always add the feature that improves the performance best
  - Stop if no feature will further improve the performance
- Backward elimination
  - Start with the full model
  - Always remove the feature that results in the best performance enhancement
  - Stop if removing any feature will get worse performance

Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Bias-Variance Trade-off
- Other issues
- Summary

Model Selection Problem

- Basic problem:
  - how to choose between competing linear regression models

- Model too simple:
  - “underfit” the data; poor predictions; high bias; low variance
- Model too complex:
  - “overfit” the data; poor predictions; low bias; high variance
- Model just right:
  - balance bias and variance to get good predictions

Bias and Variance

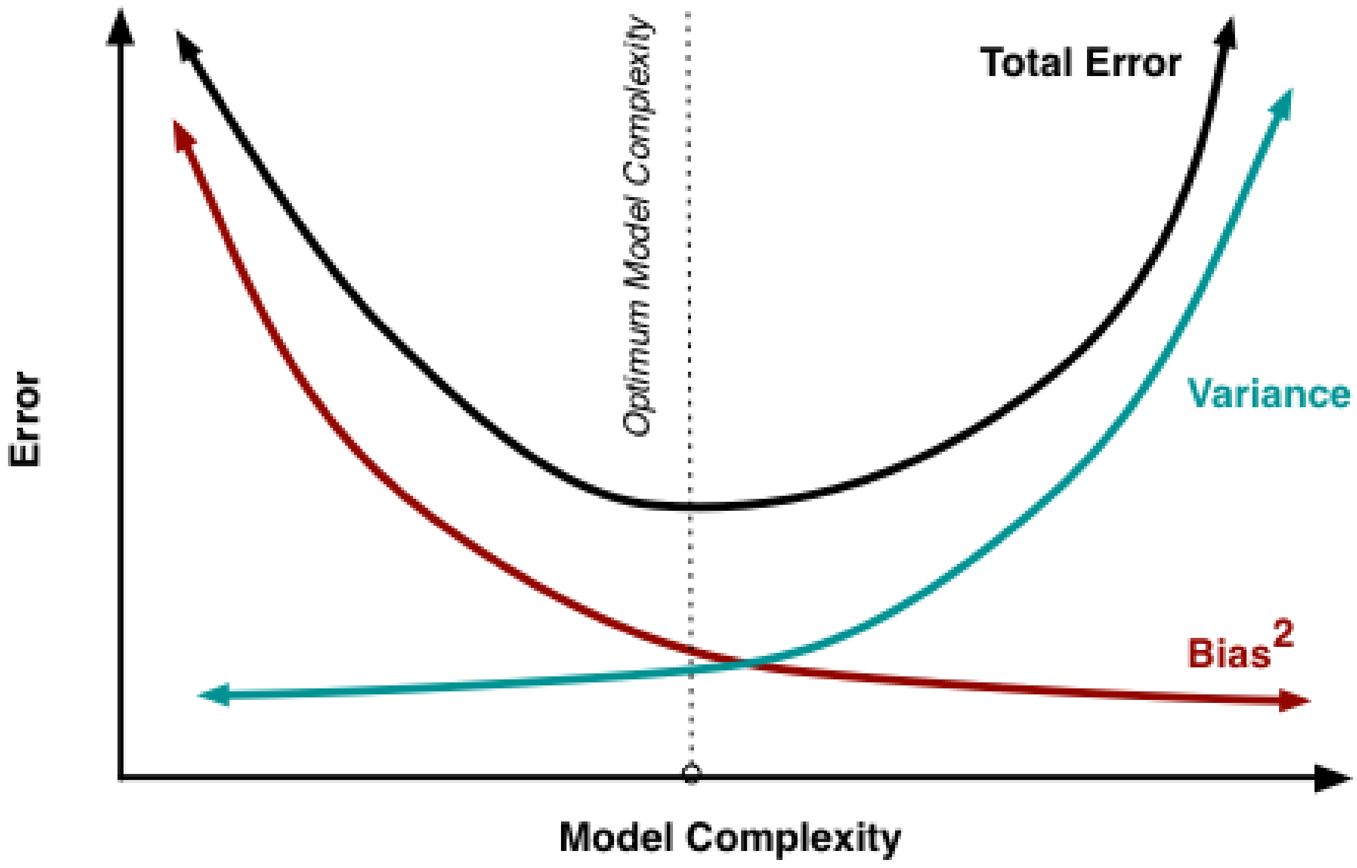
True predictor $f(x)$: $x^T\boldsymbol{\beta}$

Estimated predictor $\hat{f}(x)$: $x^T\widehat{\boldsymbol{\beta}}$

- Bias: $E(\hat{f}(x)) - f(x)$
  - How far away is the expectation of the estimator to the true value? The smaller the better.
- Variance: $Var\left(\hat{f}(x)\right) = E[\left(\hat{f}(x) - E\left(\hat{f}(x)\right)\right)^2]$
  - How variant is the estimator? The smaller the better.
- Reconsider mean square error
  - $J(\widehat{\boldsymbol{\beta}})/n = \frac{1}{2}\sum_i \left(\boldsymbol{x}_i^T\widehat{\boldsymbol{\beta}} - y_i\right)^2/n$
  - Can be considered as
    - $E[\left(\hat{f}(x) - f(x) - \varepsilon\right)^2] = bias^2 + variance + noise$

Note $E(\varepsilon) = 0, Var(\varepsilon) = \sigma^2$

Bias-Variance Trade-off

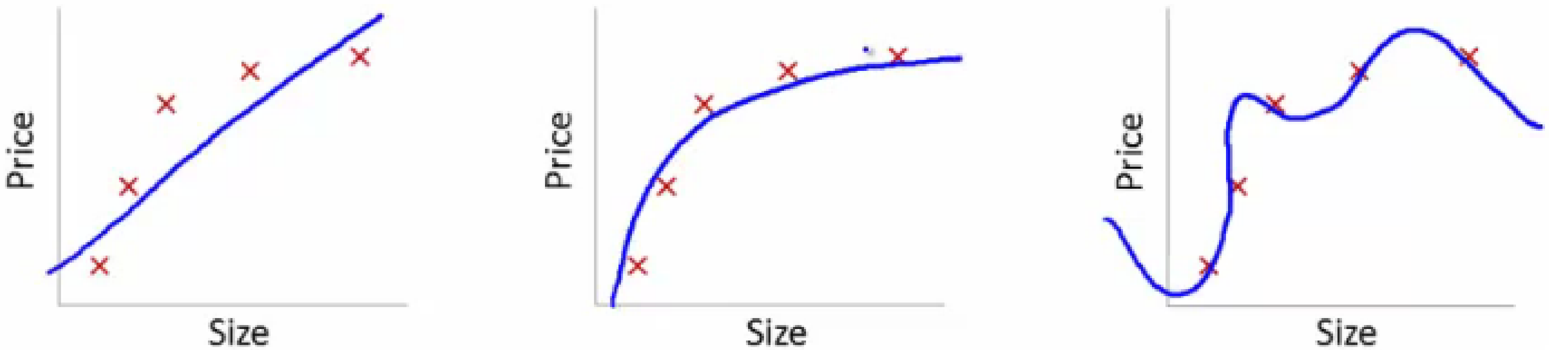


Example: degree d in regression

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$



$\theta_0 + \theta_1 x$	$\theta_0 + \theta_1 x + \theta_2 x^2$	$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$
High bias (underfit)	"Just right"	High variance (overfit)

http://www.holehouse.org/mlclass/10_Advice_for_applying_machine_learning.html

Example: regularization term in regression

Linear regression with regularization

Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

Large λ
High bias (underfit)
$\lambda = 10000.\ \theta_1 \approx 0, \theta_2 \approx 0, \ldots$
$h_\theta(x) \approx \theta_0$

Intermediate λ
"Just right"

Small λ
High variance (overfit)
$\lambda \approx 0$

Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Bias-Variance Trade-off
- Other issues
- Summary

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
- Typical methods for imbalance data in 2-class classification:
  - **Oversampling**: re-sampling of data from positive class
  - **Under-sampling**: randomly eliminate tuples from negative class
  - **Threshold-moving**: moves the decision threshold, t, so that the rare class tuples are easier to classify, and hence, less chance of costly false negative errors
  - Ensemble techniques: Ensemble multiple classifiers introduced above
- Still difficult for class imbalance problem on multiclass tasks

Multiclass Classification

- Classification involving more than two classes (i.e., > 2 Classes)
- 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**, the set of classifiers vote as an ensemble
- 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
  - Problem: Binary classifier is sensitive to errors, and errors affect vote count

Evaluation and Other Practical Issues

- Model Evaluation and Selection
- Bias-Variance Trade-off
- Other issues
- Summary

Summary

- Model evaluation and selection
  - Evaluation metric and cross-validation
- Bias-Variance Trade-off
  - Error = bias^2 + variance + noise
- Other issues
  - Imbalanced classes
  - Multi-class classification