IS422P - DATA MINING CLASSIFICATION

(PART III)

AMIRA REZK

INFORMATION SYSTEM DEPARTMENT

AGENDA

The Basics
- What is Classification?
- General Approach

Decision Tree Induction
- The Algorithm
- Attribute Selection Measures
- Tree Pruning
- Extracting Rules from Decision Trees

Bayes Classification
- Bayes' Theorem
- Naïve Bayesian Classification

Lazy Learners
- K-Nearest Neighbor Classifiers

Regression analysis
- Linear regression

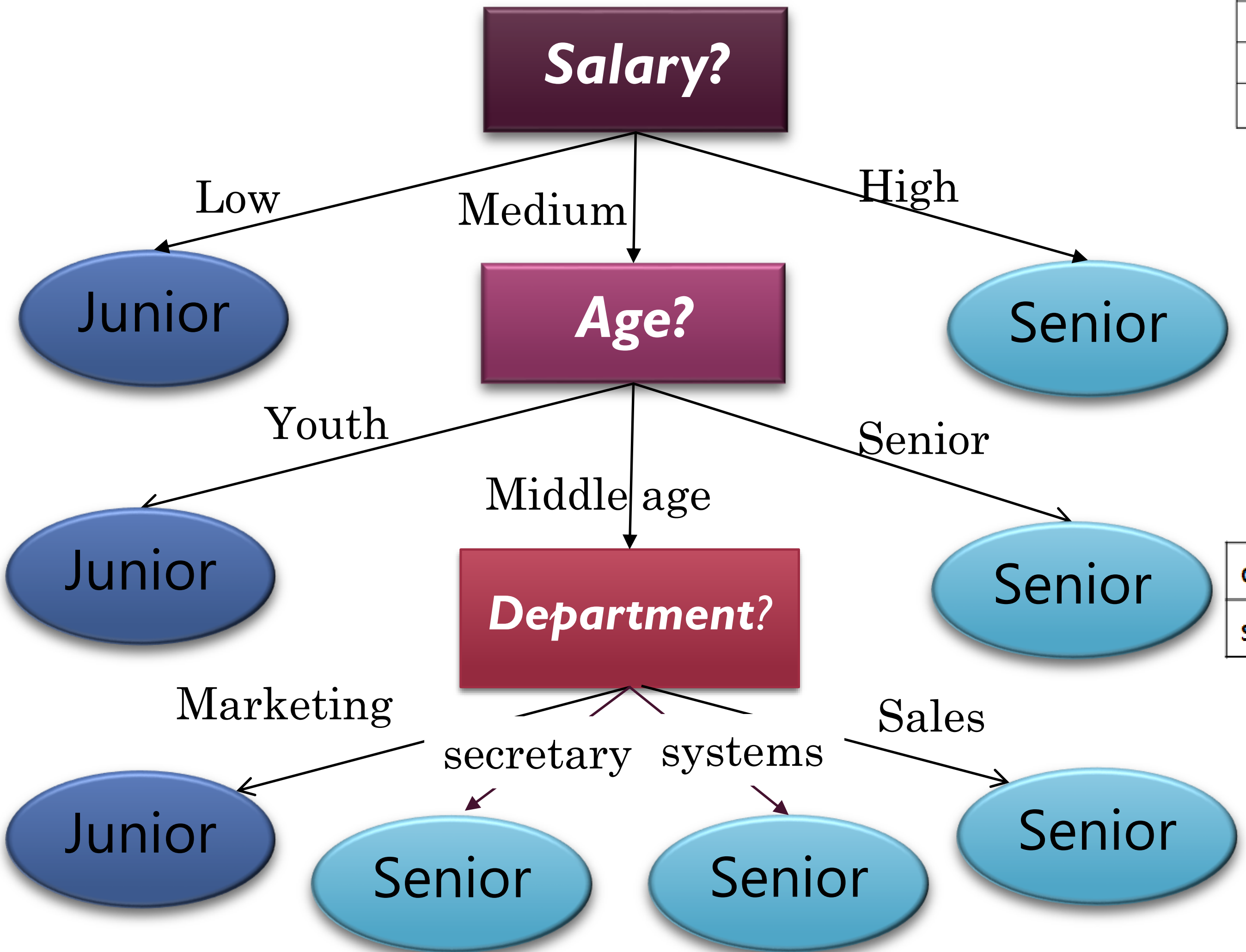
Model Evaluation
- Metrics for Evaluating Classifiers Performance
- Cross-Validation
- Bootstrap

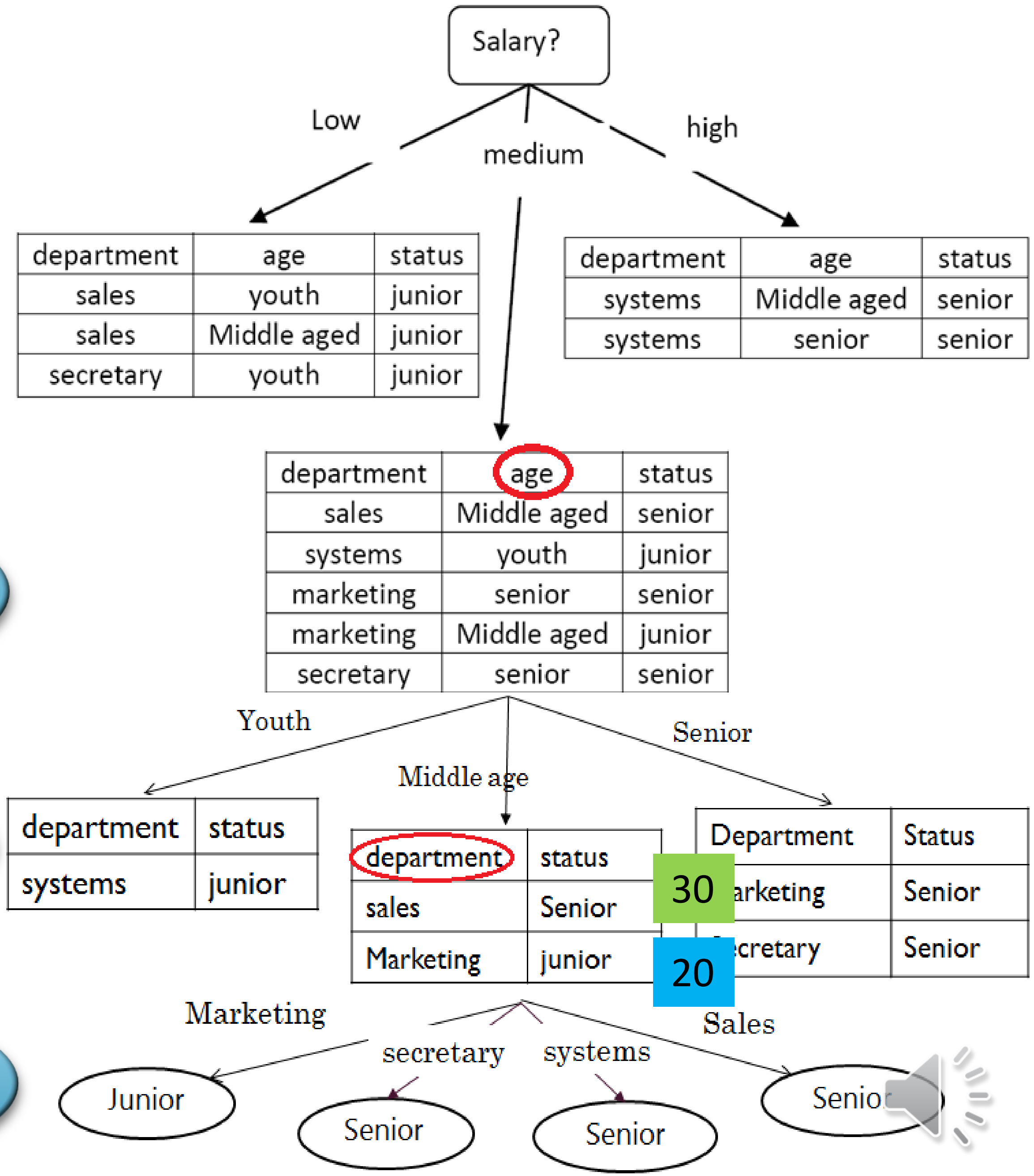
Improving Classification Accuracy
- Bagging
- Boost and AdaBoost

DECISION TREE INDUCTION
THE ALGORITHM

- **Terminating conditions**
  - All the tuples in D (represented at node N) belong to the same class
  - There are no remaining attributes on which the tuples may be further partitioned
    - majority voting is employed → convert node into a leaf and label it with the most common class in data partition
  - There are no tuples for a given branch
    - a leaf is created with the majority class in data partition

DECISION TREE INDUCTION
ATTRIBUTE SELECTION MEASURE





department	age	status
sales	youth	junior
sales	Middle aged	junior
secretary	youth	junior

department	age	status
systems	Middle aged	senior
systems	senior	senior

department	age	status
sales	Middle aged	senior
systems	youth	junior
marketing	senior	senior
marketing	Middle aged	junior
secretary	senior	senior

department	status
systems	junior

department	status
sales	Senior
Marketing	junior

Department	Status
arketing	Senior
cretary	Senior

MODEL EVALUATION
METRICS FOR EVALUATING CLASSIFIER PERFORMANCE

Measure	Formula
accuracy, recognition rate	$\frac{TP+TN}{P+N}$
error rate, misclassification rate	$\frac{FP+FN}{P+N}$
sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
precision	$\frac{TP}{TP+FP}$

Positives →
tuples representing <u>class of interest</u>
Negatives →
tuples representing other class(es)
True Positives →
positive tuples <u>*correctly*</u> labeled
False Positives →
negative tuples <u>*incorrectly*</u> labeled
True Negatives →
negative tuples <u>*correctly*</u> labeled
False Negatives →
positive tuples <u>*incorrectly*</u> labeled

MODEL EVALUATION
METRICS FOR EVALUATING CLASSIFIER PERFORMANCE

Predicted

Actual

	Yes	No	Total
Yes	**TP**	**FN**	P
No	**FP**	**TN**	N
Total	$\hat{P}$	$\hat{N}$	**P + N**

Confusion Matrix

MODEL EVALUATION
METRICS FOR EVALUATING CLASSIFIER PERFORMANCE

Balanced Classes

Predicted

Actual

	Yes	No	Total
Yes	**6954**	**46**	7000
No	**412**	**2588**	3000
Total	7366	2634	**10000**

Recognition(%)
$TP/P = 99.34$
$TN/N = 86.27$
$\frac{TP+TN}{P+N} = 95.42$

Example Buys_Computer Confusion Matrix

Use *sensitivity* (TPs or *recall*) and *specificity*

MODEL EVALUATION
METRICS FOR EVALUATING CLASSIFIER PERFORMANCE

Imbalanced Classes

Predicted

Actual

	Yes	No	Total	Accuracy (%)
Yes	90	210	300	30
No	140	9560	9700	98.56
Total	230	9770	10000	96.4

Low

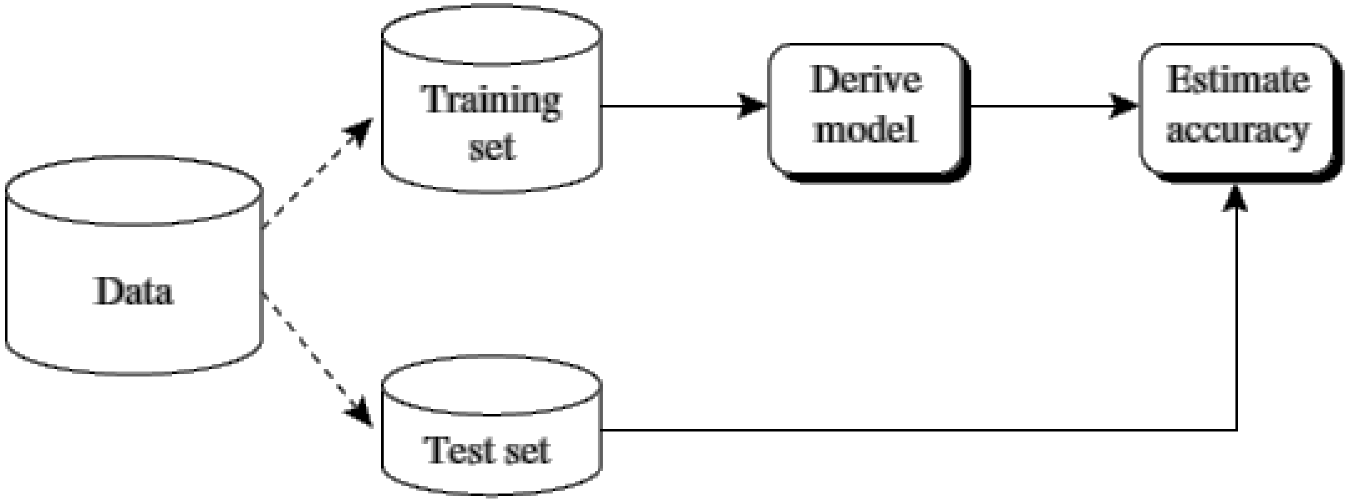
High

Example Cancer Confusion Matrix

Use *sensitivity* (TPs or *recall*) and *specificity*

MODEL EVALUATION
HOLDOUT AND RANDOM SUBSAMPLING

- **Holdout** → RANDOMLY allocate 2/3 of data for training and remaining 1/3 for testing
- **Random Subsampling** → Repeat holdout *k* times and take average accuracy

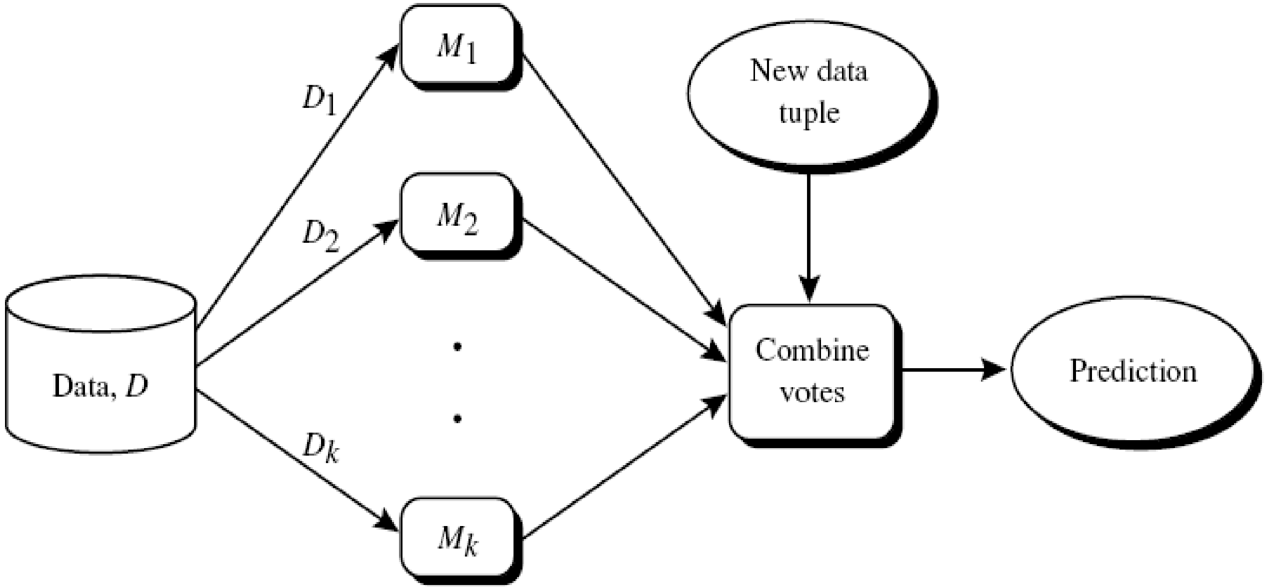


MODEL EVALUATION
CROSS-VALIDATION

- ***k-fold* cross-validation** → randomly partition dataset into k mutually exclusive ***folds*** of approximately equal size
- In iteration i, $fold_i$ is test set and all other folds are training set
- Accuracy $= \dfrac{\sum correct\ classifications\ for\ all\ k\ iterations}{dataset\ size}$
- **Stratified k-fold cross-validation** → class distribution in each fold is approximately the same as in initial dataset
  - **Stratified 10-fold cross-validation** is recommended

IMPROVING CLASSIFICATION ACCURACY ENSEMBLE METHODS

- **Ensemble** → a set of classifiers, each with a vote for a class label
  - Each base classifier is produced from a different partition of the dataset
  - Majority voting is used to compose an **aggregate classification**



IMPROVING CLASSIFICATION ACCURACY
ENSEMBLE METHODS – BAGGING / BOOTSTRAP

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D, a set of d training tuples;
- k, the number of models in the ensemble;
- a classification learning scheme (decision tree algorithm, naïve Bayesian, etc.).

Output: The ensemble—a composite model, $M*$.

Method:

(1) **for** $i = 1$ **to** k **do** // create k models:
(2) create bootstrap sample, D_i, by sampling D with replacement;
(3) use D_i and the learning scheme to derive a model, M_i;
(4) **endfor**

To use the ensemble to classify a tuple, X:

let each of the k models classify X and return the majority vote;

Bootstrap → same size as dataset, sampling with replacement

3
5
3
7
4
3
7
9
6
10

QUESTION?

NEXT …….