CENG646
Data Mining

Ch. 8: Classification: Basic Concepts (Part 2)
Ch. 9: Classification: Advanced

[Adapted from DATA MINING Concepts and Techniques. Third Edition. J. Han, M. Kamber and J. Pei]

Chapter 8. Classification: Basic Concepts

- Bayes Classification Methods
- Model Evaluation and Selection
- Summary

Bayesian Classification: Why?

- Bayesian classifier: Bayesian classifiers are statistical classifiers. They can predict class membership probabilities such as the probability that a given tuple belongs to a particular class
- Foundation: Based on Bayes' Theorem
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data

Bayesian Theorem: Basics

- Let **X** be a data sample ("*evidence*"): class label is unknown
  - Example: X = (age: between 31..40, income: medium)
  - There are two classes: C1: will buy computer, C2: will not buy computer
- Let H be a *hypothesis* that X belongs to class C1
- Classification is to determine P(H|**X**) (*posteriori probability):* the probability that the hypothesis holds given the observed data sample **X**
  - What is the probability of buying a computer conditional on the observation X = (age: between 31..40, income: medium)

Bayesian Theorem: Basics

- P(H) (*prior probability*): the initial probability
  - E.g., **X** will buy computer, regardless of age, income, …
- P(**X**) (evidence): probability that sample data is observed
- P(**X**|H) (likelyhood): the probability of observing the sample **X**, given that the hypothesis holds
  - E.g., Given that **X** will buy computer, the probability that X is 31..40, medium income

Bayesian Theorem

- Bayes theorem states that:

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H)/P(\mathbf{X})$$

- Informally, this can be written as

  posteriori = likelihood x prior/evidence

Bayesian Theorem

- Bayesian classification works as follows:
  - Suppose that there are m classes C_1 ,C_2, … , C_m
  - Given a tuple **X:**
  - The classifier predicts that **X** belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|X)$ for all the m classes (k=1, … , m)
  - Example: P(will buy computer | **X**) = 0.8 and P(will not buy computer | **X**) = 0.2 -> Classify as buys computer
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Bayesian Theorem

- The posterior probability is computed using Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(**X**) does not depend on the class C_i, we only need to find the class that maximizes

$$P(\mathbf{X}|C_i)P(C_i)$$

- How the probabilities P(**X**|C_i) and P(C_i) are determined?
  - They are learned from the training data

Towards Naïve Bayesian Classifier

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n dimensional attribute vector **X** = (x_1, x_2, …, x_n)
  - n is the number of attributes (features)
- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

- This greatly reduces the computation cost: Only counts the class distribution

Naïve Bayesian Classifier: An Example

Classes:

C1:buys_computer = ‘yes’

C2:buys_computer = ‘no’

Data sample

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

Training data:

age	income	student	credit_rating	comp
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Naïve Bayesian Classifier: An Example

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X is classified as belonging to class ("buys_computer = yes")

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional probability to be **non-zero**. Otherwise, the estimated probability will be zero

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)
- Use **Laplacian correction** (or Laplacian estimator)
  - *Adding 1 to each case*

    Prob(income = low) = 1/1003

    Prob(income = medium) = 991/1003

    Prob(income = high) = 11/1003
  - The "corrected" probability estimates are close to their "uncorrected" counterparts

Naïve Bayesian Classifier: Comments

- Advantages
  - Easy to implement
  - Good results obtained in most of the cases
- Disadvantages
  - Assumption: class conditional independence, therefore loss of accuracy
  - Practically, dependencies exist among variables
    - E.g., hospitals: patients: Profile: age, family history, etc.

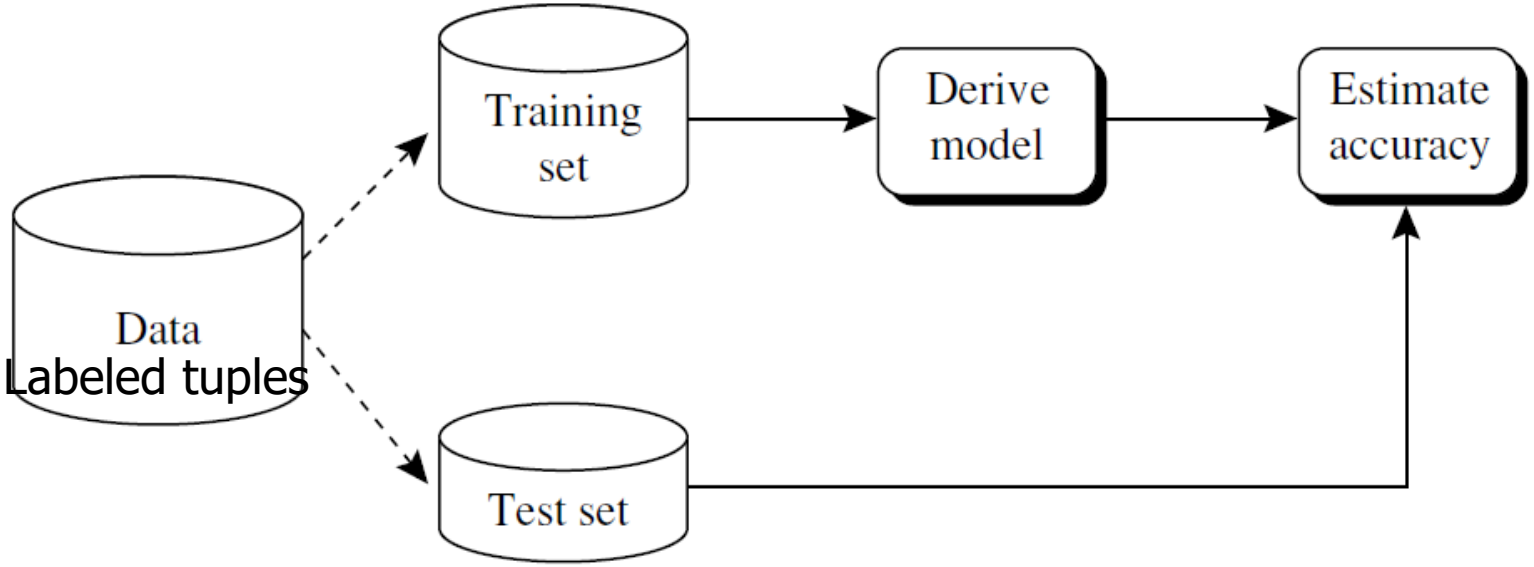
      Symptoms: fever, cough etc., Disease: lung cancer, diabetes, etc.
    - Dependencies among these cannot be modeled by Naïve Bayesian Classifier
- How to deal with these dependencies? Bayesian Belief Networks (Chapter 9)

Chapter 8. Classification: Basic Concepts

- Bayes Classification Methods
- Model Evaluation and Selection
- Summary

Model Evaluation and Selection

- Evaluation of a classifer:
  - How "accurate" a classifier is at predicting the class label of tuples? How can we measure accuracy?
- Use **test set** of class-labeled tuples instead of training set when assessing accuracy



Model Evaluation and Selection

- Evaluating a model comes down to dividing the available labeled data into three sets: training, validation, and test.

Training set	Validation set	Test set

- You train on the training data and evaluate your model on the validation data.
  - Once your model is ready, you evaluate it one final time on the test data.

- The purpose of evaluation on the validation set is to **select** which model is the best when comparing several models (e.g., naïve Baysian vs. decision tree), or for **tuning** a model configuration (e.g., number of trees and maximum tree depth in a random forest)

Model Evaluation and Selection

- Choosing or tuning a model is a form of *learning*: a search for a good configuration in some parameter space.
- Why not have only two sets: a training set and a test set?
  - You'd train on the training data and evaluate on the test data. Much simpler!
- Evaluating on the test data can quickly result in *overfitting to the test set*, even though your model is never directly trained on it.
  - There are *information leaks*.
  - Every time the model is tuned based on the model's performance on the test set, some information about the test data leaks into the model.
- At the end of the day, you'll end up with a model that performs artificially well on the test data.
- You care about performance on completely new data: the model shouldn't have had access to *any* information about the test set, even indirectly.

Model Evaluation and Selection

- A common approach to split a labeled dataset:
  - Training Set: 70-80% of the dataset
  - Validation Set: 10-15% of the dataset
  - Test Set: 10-15% of the dataset

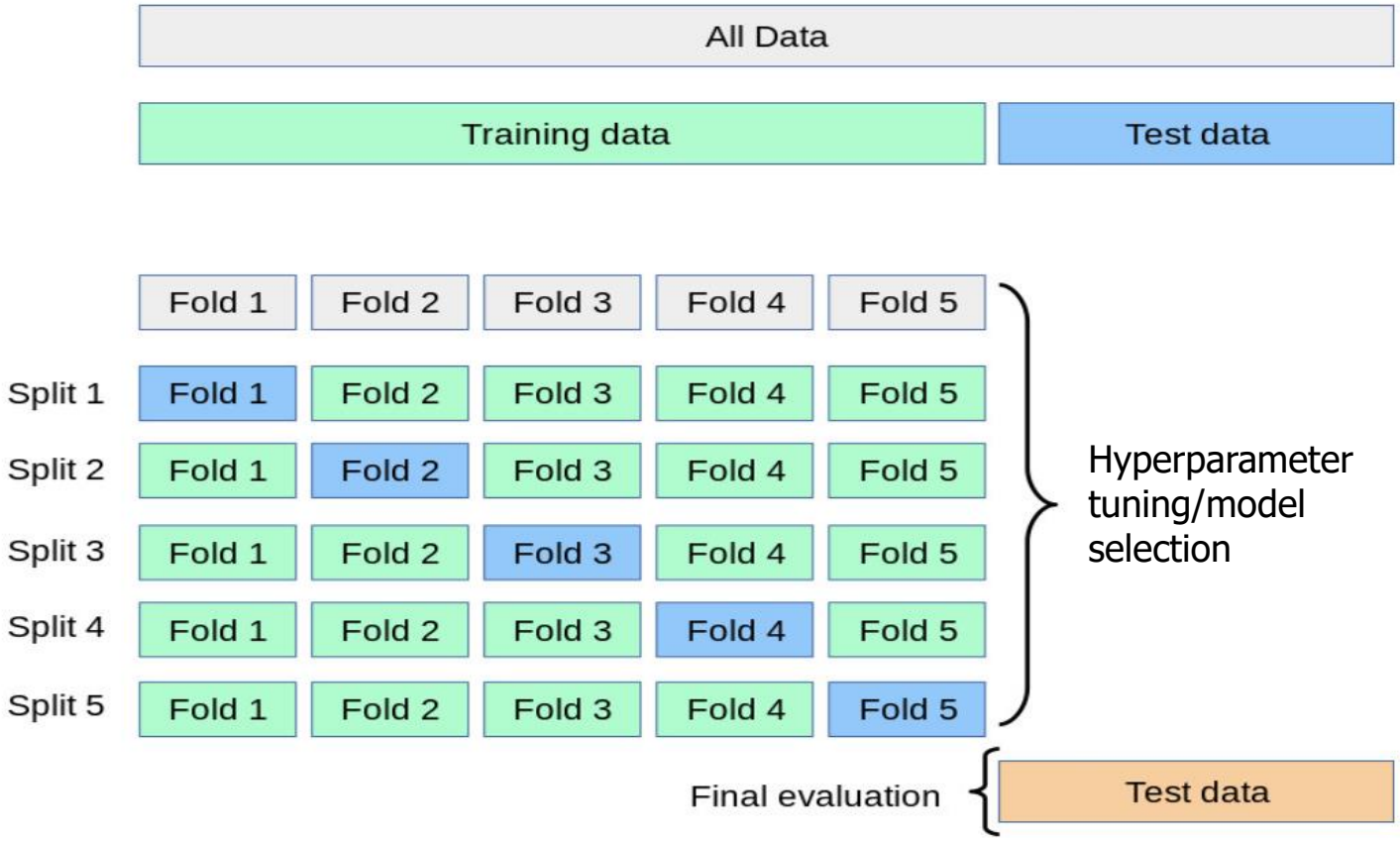
- Small dataset: Use cross-validation
- Cross-validation is a resampling technique that involves partitioning a dataset into multiple subsets, training and evaluating the model on different combinations of these subsets
- Types of cross-validation: k-fold cross-validation, stratified k-fold cross-validation, and leave-one-out cross-validation

Model Evaluation and Selection

- k-fold cross-validation:
  - The **training set** is split into k parts of same size, usually after data shuffling
  - Training happens k times, each time leaving out a different part of the training set. Each training generates a model
  - Typically, the error of these k-models is averaged

Model Evaluation and Selection

- Example: for k=5,
  - The model is trained 5 times
  - Each time, use 4 folds for training and 1 fold for evaluation
  - The 5 evaluation metric values can be averaged to obtain one value
- Try several models or differed hyperparameter values, and select the model with best average value
- After hyperparameter tuning/model selection:
  - The selected model is trained on all training data and evaluated on test data



Classifier Evaluation Metrics: Confusion Matrix

- Consider a binary classification problem with two classes:
  - C_1 : positive tuples (tuples of main class of interest)
  - $\neg C_1$: negative tuples (all other tuples)
- Confusion matrix of classifier:

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)**

- TP and TN: the classifier is getting things right
- FP and FN: the classifier is getting things wrong (i.e., mislabeling)

Classifier Evaluation Metrics: Confusion Matrix

Example of Confusion Matrix for buying computer

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

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

Actual\Predicted	C	¬C	Total
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
Total	**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

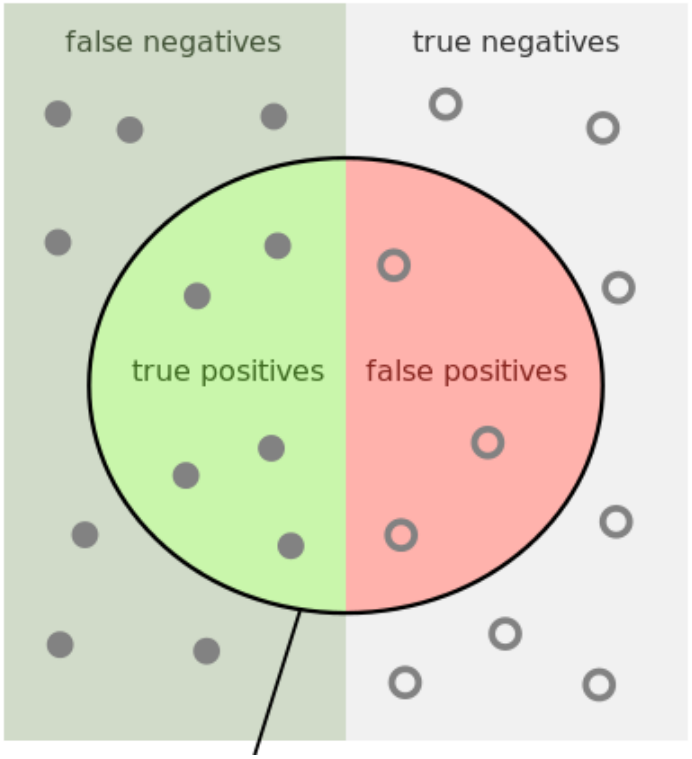
$$precision = \frac{TP}{TP + FP}$$

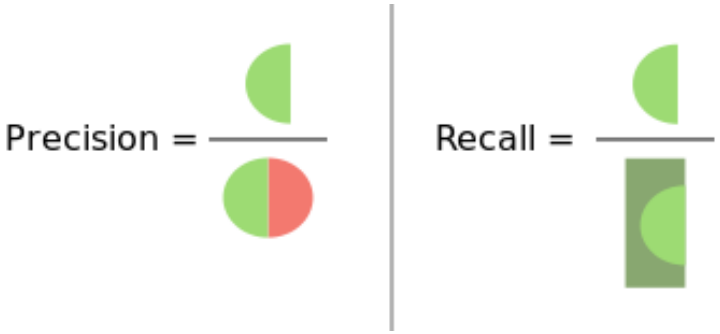
Recall and sensitivity are the same

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

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

- Perfect scores are 1.0
- Inverse relationship between precision & recall: It is possible to increase one at the cost of reducing the other





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

- ***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

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

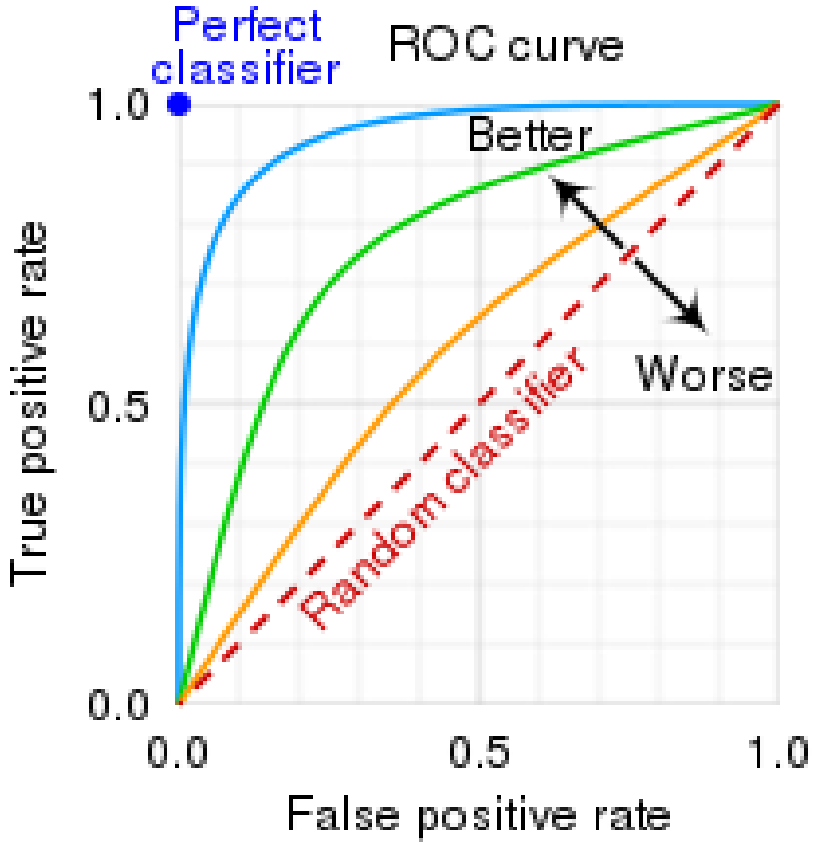
Classifier Evaluation Metrics: Example

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	96.40 (*accuracy*)

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

Model Selection: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- ROC curve is a performance measurement for classification problem at various thresholds settings
- 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
- 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 (TP/P) (=recall)
- Horizontal axis rep. the false positive rate (FP/N = 1-TN/N)
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

Model Selection: ROC Curves

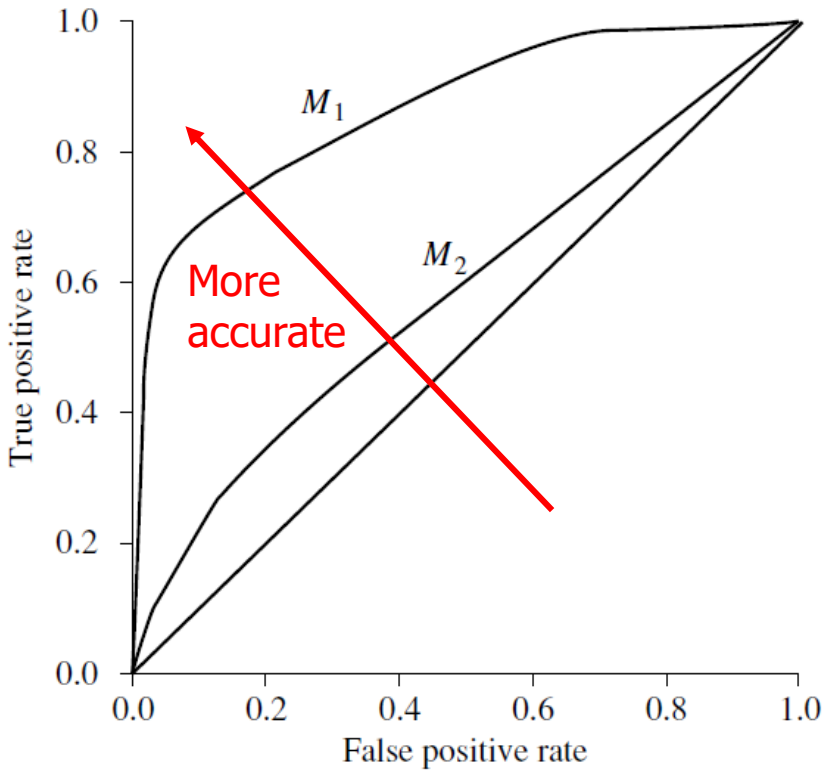
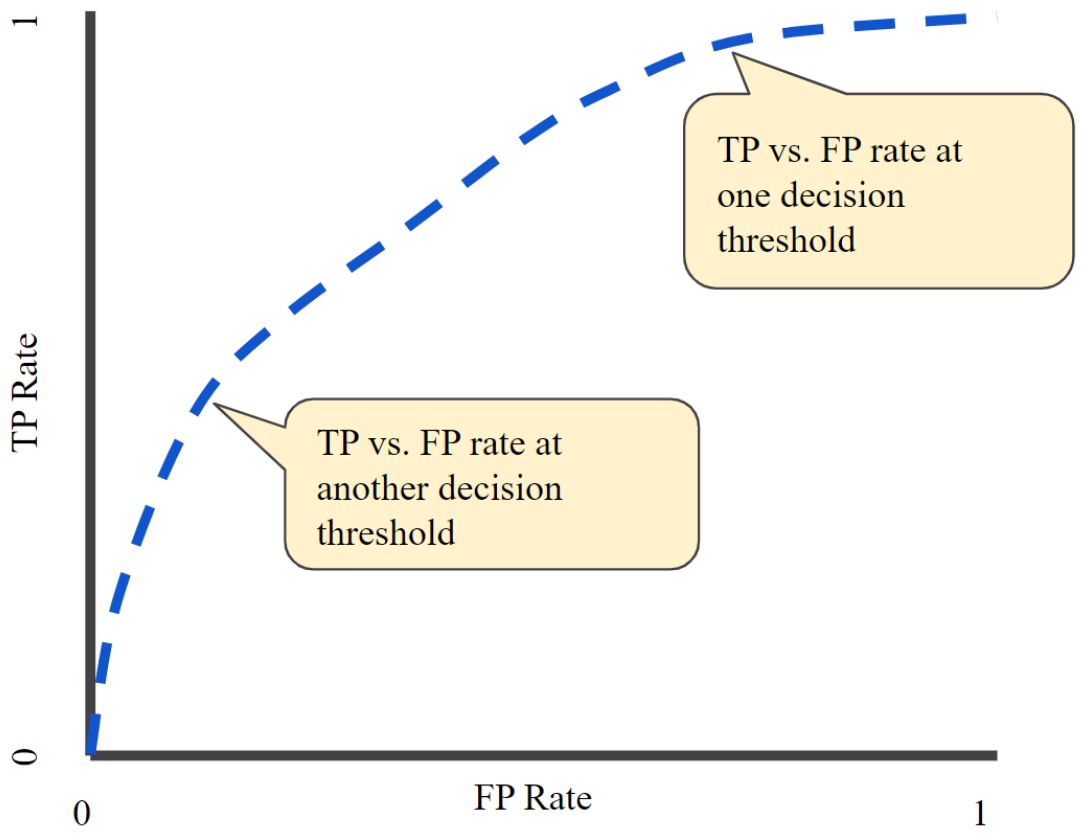


Figure 8.20 ROC curves of two classification models, M_1 and M_2. The diagonal shows where, for every true positive, we are equally likely to encounter a false positive. The closer an ROC curve is to the diagonal line, the less accurate the model is. Thus, $M1$ is more accurate here.

Model Selection: ROC Curves

- An ROC curve plots TPR vs. FPR at different classification thresholds.
- Example: Bayesian classifier:
  - For an observation **x**, compute the probability v = P(class=Positive|**x**)
  - Fix a threshold value v_{th}
  - If $v > v_{th}$, then classify as positive,
  - otherwise classify as negative

- Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.



AUC: Area Under the ROC Curve

- **AUC** stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve.
- AUC provides an aggregate measure of performance across all possible classification thresholds.
- AUC ranges in value from 0 to 1.
- The higher the AUC value, the better the classifier.

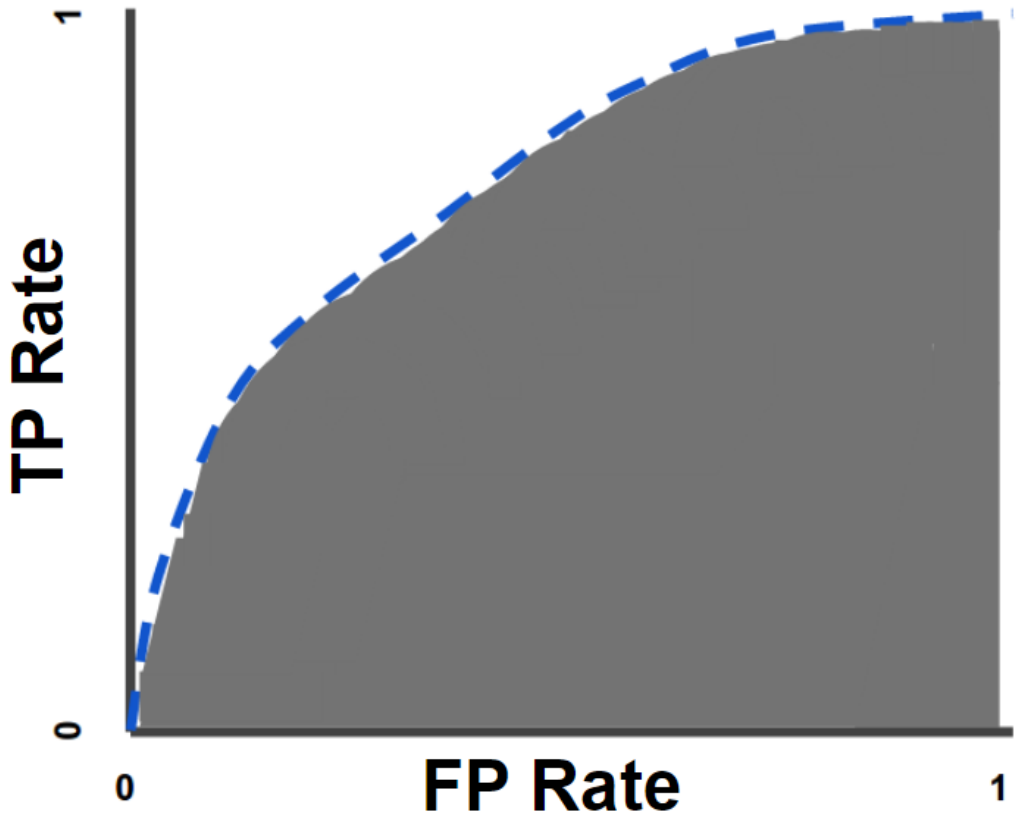


Figure 5. AUC (Area under the ROC Curve).

Issues Affecting Model Selection

- **Accuracy**
  - classifier accuracy: predicting class label
- **Speed**
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- **Robustness**
  - handling noise and missing values
- **Interpretability**
  - understanding and insight provided by the model

Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Model Evaluation and Selection
- Summary

Summary

- Classification is a form of data analysis that extracts models describing important data classes.
- Effective methods have been developed for decision tree, random forest, and naive Bayesian classification.
- Evaluation metrics include: accuracy, sensitivity, specificity, precision, recall, F measure, and F_{β} measure.
- ROC curves are useful for model selection.