Data Mining

Ensemble Techniques

Introduction to Data Mining, 2^{nd} Edition

by

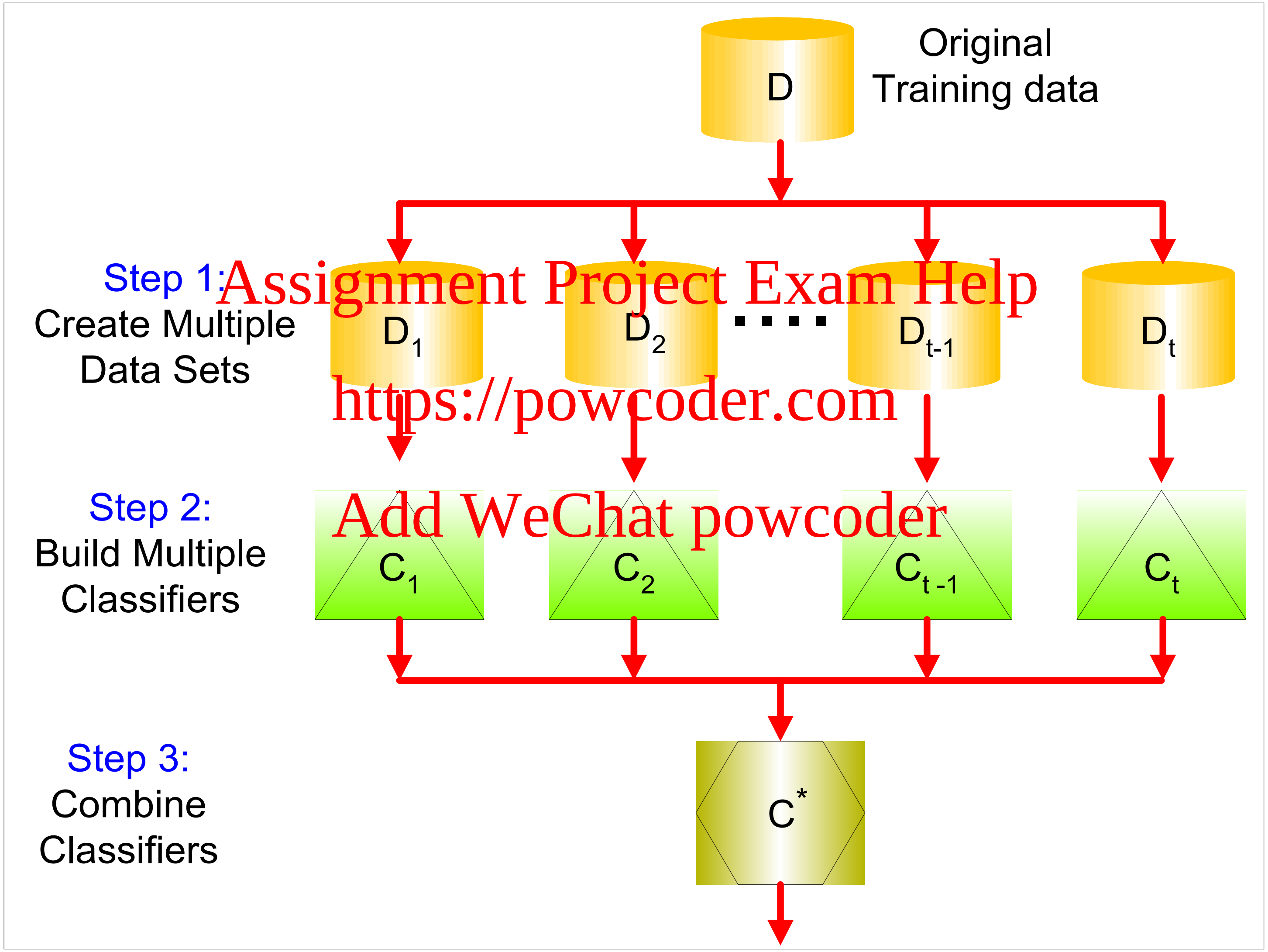
Tan, Steinbach, Karpatne, Kumar

Ensemble Methods

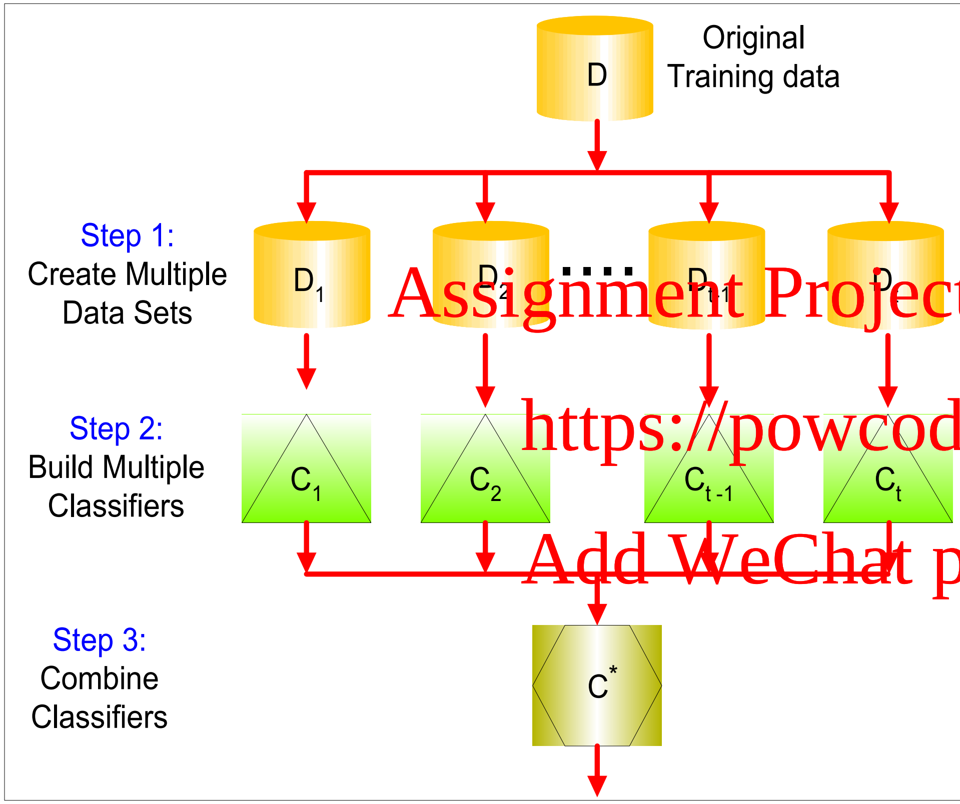
- Construct a set of classifiers from the training data

- Predict class label of test records by combining the predictions made by multiple classifiers
  - E.g. by majority vote

General Approach



Estimate "combined" error rate



Example:
There are 10 base classifiers (C1 to C10).
Each has error rate 0.3.
For each input testing sample, we apply C1 to C10. The final prediction is the consensus of C1 to C10's output.

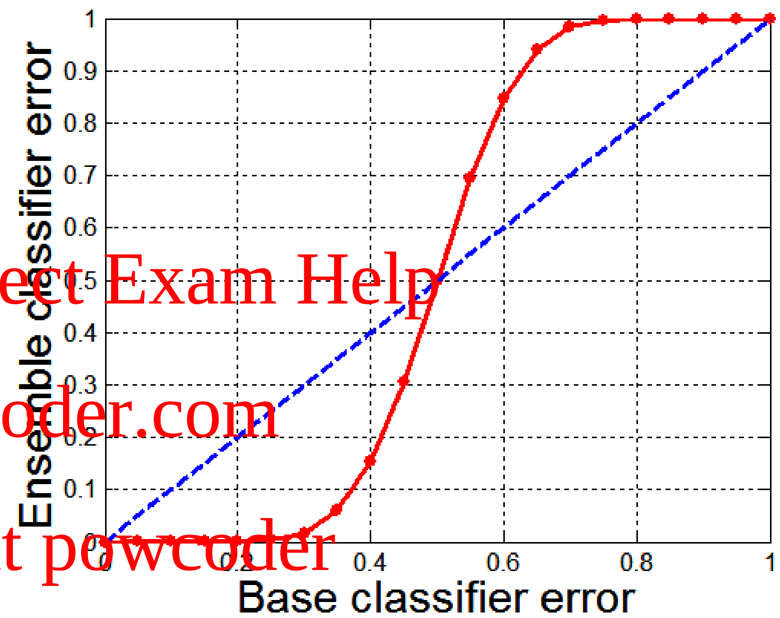
In-class exercise:

Assuming that all the base classifiers are independent of each other, estimate the prediction error rate of the ensemble classifier?

C*: take the consensus prediction

Why Ensemble Methods work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume errors made by classifiers are uncorrelated
  - Probability that the ensemble classifier makes a wrong prediction:



Voting using the predictions of base classifiers

$$P(X \quad 13) = \sum_{i=13}^{25} \binom{25}{i} \varepsilon^{i} (1-\varepsilon)^{25-i} = 0.06$$

X: number of base classifiers with wrong prediction

Re-examine our assumption

Record	X1	X2	X3	Class	C1	C2	C3
1	0	2	1	+	+	+	-
2	1	1	0	-	-	+	-
3	1	2	1	-	-	-	+
4	0	2	0	+	-	+	+
5	1	1	1	-	-	+	+
6	1	0	1	-	+	-	-
7	0	2	0	+	+	-	-
8	1	1	0	+	-	+	+
9	0	1	1	-	-	+	-
10	0	0	0	-	-	+	-

Record	X1	X2	X3	Class	C1	C2	C3
1	0	2	1	+	+	+	-
2	1	1	0	-	+	+	+
3	1	2	1	-	-	-	-
4	0	2	0	+	+	+	+
5	1	1	1	-	-	-	+
6	1	0	1	-	-	-	-
7	0	2	0	+	-	-	-
8	1	1	0	+	+	+	+
9	0	1	1	-	+	+	+
10	0	0	0	-	-	-	-

- Which ensemble classifier will work (i.e. can improve the classification performance)?

Types of Ensemble Methods

- Manipulate data distribution (our focus)
  - Example: bagging, boosting
- Manipulate input features
  - Example: random forests (construct multiple trees using a subset of features. Works well for data sets with redundant features)

Bagging

- Sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Build classifier on each bootstrap sample

- Each sample has probability $\boxed{1-(1-1/n)^n}$ of being selected

n is the number of samples in the original data. Think about the probability that a sample is not chosen in n trials.

Bagging Algorithm

Algorithm 5.6 Bagging Algorithm

```
Let k be the number of bootstrap samples.
for i = 1 to k do
    Create a bootstrap sample of size n, D_i.
    Train a base classifier C_i on the bootstrap sample D_i.
end for
```

$C^*(x) = \arg\max_y \sum_i \delta(C_i(x) = y)$. $\{\delta(\cdot) = 1$ if its argument is true, and 0 otherwise.$\}$

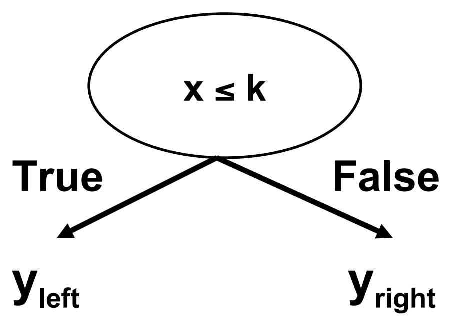
Bagging Example

- Consider 1-dimensional data set:

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

- Classifier is a decision stump
  - Decision rule: $x \le k$ versus $x > k$
  - Split point k is chosen based on entropy



Bagging Example

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Example

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.5	0.9	1	1	1
y	1	1	1	-1	-1	-1	1	1	1	1

x <= 0.7 ➔ y = 1
x > 0.7 ➔ y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ➔ y = 1
x > 0.3 ➔ y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Example

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 → y = -1
x > 0.75 → y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 → y = -1
x > 0.75 → y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 → y = -1
x > 0.75 → y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 → y = -1
x > 0.75 → y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 → y = 1
x > 0.05 → y = 1

Bagging Example

- Summary of Training sets:

Round	Split Point	Left Class	Right Class
1	0.35	1	-1
2	0.7	1	1
3	0.35	1	-1
4	0.3	1	-1
5	0.35	1	-1
6	0.75	-1	1
7	0.75	-1	1
8	0.75	-1	1
9	0.75	-1	1
10	0.05	1	1

Bagging Example

- Assume test set is the same as the original data
- Use majority vote to determine class of ensemble classifier

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1

Predicted Class

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of each boosting round

Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

AdaBoost

- Base classifiers: C_1, C_2, …, C_T

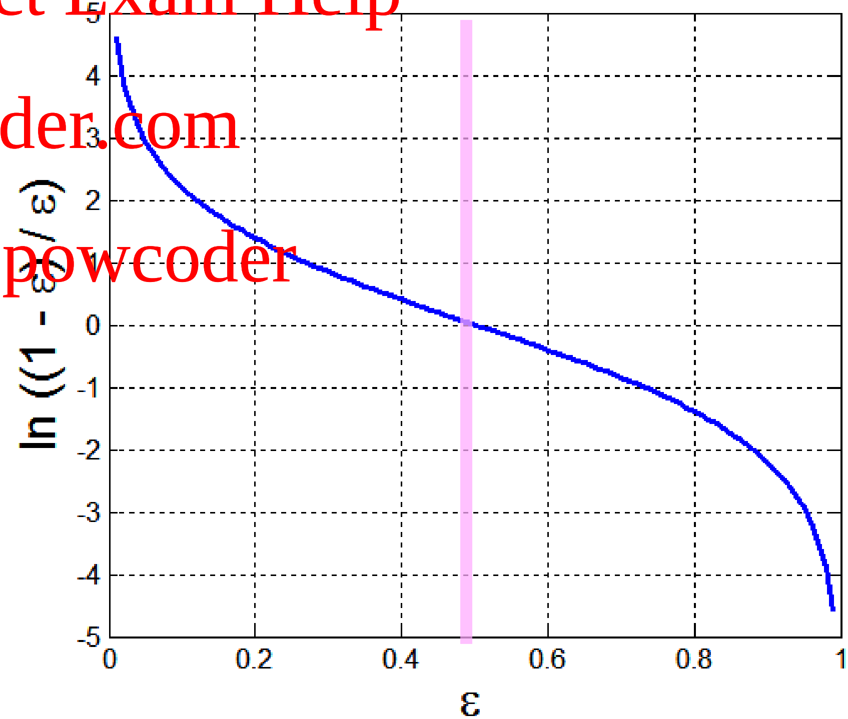
- Error rate of each C_i:

For every sample j=1 to N, w_j is the weight, δ is the identity function (δ(x)=1 if x is true

$$\varepsilon_i = \frac{1}{N}\sum_{j=1}^{N} w_j \delta\left(C_i(x_j) \neq y_j\right)$$

- Importance of a classifier:

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$



AdaBoost Algorithm

- Weight update:

$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \begin{cases} \exp^{-\alpha_j} & \text{if } C_j(x_i) = y_i \\ \exp^{\alpha_j} & \text{if } C_j(x_i) \neq y_i \end{cases}$$

where Z_j is the normalization factor

- If any intermediate rounds produce error rate higher than 50%, the weights are reverted back to 1/n and the resampling procedure is repeated

- Classification:

$$C^*(x) = \arg\max_y \sum_{j=1}^{T} \alpha_j \delta\left(C_j(x) = y\right)$$

AdaBoost Algorithm

Algorithm 5.7 AdaBoost Algorithm

```
1:  w = {w_j = 1/n | j = 1, 2, ..., n}.   {Initialize the weights for all n instances.}
2:  Let k be the number of boosting rounds.
3:  for i = 1 to k do
4:      Create training set D_i by sampling (with replacement) from D according to w.
5:      Train a base classifier C_i on D_i.
6:      Apply C_i to all instances in the original training set, D.
7:      ε_i = (1/n) [ Σ_j w_j δ(C_i(x_j) ≠ y_j) ]   {Calculate the weighted error}
8:      if ε_i > 0.5 then
9:          w = {w_j = 1/n | j = 1, 2, ..., n}.   {Reset the weights for all n instances.}
10:         Go back to Step 4.
11:     end if
12:     α_i = (1/2) ln ((1 − ε_i) / ε_i).
13:     Update the weight of each instance according to equation (5.88).
14: end for
15: C*(x) = argmax_y Σ_{j=1}^{T} α_j δ(C_j(x) = y)).
```

Step 7: $\epsilon_i = \frac{1}{n}\left[\sum_j w_j\, \delta\big(C_i(x_j) \neq y_j\big)\right]$

Step 12: $\alpha_i = \frac{1}{2}\ln\frac{1-\epsilon_i}{\epsilon_i}$

Step 15: $C^*(\mathrm{x}) = \arg\max_y \sum_{j=1}^{T} \alpha_j \delta(C_j(\mathrm{x}) = y))$.

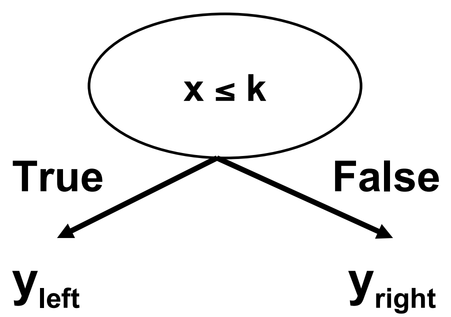
AdaBoost Example

- Consider 1-dimensional data set:

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

- Classifier is a decision stump
  - Decision rule: $x \le k$ versus $x > k$
  - Split point k is chosen based on entropy



AdaBoost Example

- Training sets for the first 3 boosting rounds:

Boosting Round 1:

x	0.1	0.4	0.5	0.6	0.6	0.7	0.7	0.7	0.8	1
y	1	-1	-1	-1	-1	-1	-1	-1	1	1

Boosting Round 2:

x	0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3
y	1	1	1	1	1	1	1	1	1	1

Boosting Round 3:

x	0.2	0.2	0.4	0.4	0.4	0.4	0.5	0.6	0.6	0.7
y	1	1	-1	-1	-1	-1	-1	-1	-1	-1

- Summary:

Round	Split Point	Left Class	Right Class	alpha
1	0.75	-1	1	1.738
2	0.05	1	1	2.7784
3	0.3	1	-1	4.1195

AdaBoost Example

- Weights

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	0.311	0.311	0.311	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.029	0.029	0.029	0.228	0.228	0.228	0.228	0.009	0.009	0.009

- Classification

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	-1	-1	-1	-1	-1	-1	-1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
Sum	5.16	5.16	5.16	-3.08	-3.08	-3.08	-3.08	0.397	0.397	0.397
Sign	1	1	1	-1	-1	-1	-1	1	1	1

Predicted Class for x=0.2:

C1: -1, C2: 1, C3=1

-1: 1.738

1: 2.77+4.11. Thus, the prediction is 1 (correct)

Round	Split Point	Left Class	Right Class	alpha
1	0.75	-1	1	1.738
2	0.05	1	1	2.7784
3	0.3	1	-1	4.1195

Exercise: what is the prediction for x=0.4?

Data Mining
Classification: Alternative Techniques

Imbalanced Class Problem

Introduction to Data Mining, 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

Class Imbalance Problem

- Lots of classification problems where the classes are skewed (more records from one class than another)
  - Credit card fraud
  - Intrusion detection
  - Defective products in manufacturing assembly line

Challenges

- Evaluation measures such as accuracy is not well-suited for imbalanced class

- Detecting the rare class is like finding needle in a haystack

Confusion Matrix

- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Accuracy

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Problem with Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

Record	X1	X2	X3	Class	C1
1	0	2	1	+	-
2	1	1	0	-	-
3	1	2	1	-	-
4	0	2	0	+	-
5	1	1	1	-	-
6	1	0	1	-	-
7	0	2	0	-	-
8	1	1	0	-	-
9	0	1	1	-	-
10	0	0	0	-	-

In the left example, what is the accuracy of the classifier C1?

How is the performance of C1 on predicting + class?

Problem with Accuracy

- Consider a 2-class problem
  - Number of Class NO examples = 990
  - Number of Class YES examples = 10

- If a model predicts everything to be class NO, accuracy is 990/1000 = 99 %

  - This is misleading because the model does not detect any class YES example
  - Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)

Alternative Measures

	PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F - measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

Alternative Measures

	PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	10	0
	Class=No	10	980

$$\text{Precision (p)} = \frac{10}{10+10} = 0.5$$

$$\text{Recall (r)} = \frac{10}{10+0} = 1$$

$$\text{F-measure (F)} = \frac{2*1*0.5}{1+0.5} = 0.62$$

$$\text{Accuracy} = \frac{990}{1000} = 0.99$$

Alternative Measures

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	10	0
Class=No	10	980

$$\text{Precision (p)} = \frac{10}{10+10} = 0.5$$

$$\text{Recall (r)} = \frac{10}{10+0} = 1$$

$$\text{F-measure (F)} = \frac{2*1*0.5}{1+0.5} = 0.62$$

$$\text{Accuracy} = \frac{990}{1000} = 0.99$$

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	1	9
Class=No	0	990

$$\text{Precision (p)} = \frac{1}{1+0} = 1$$

$$\text{Recall (r)} = \frac{1}{1+9} = 0.1$$

$$\text{F-measure (F)} = \frac{2*0.1*1}{1+0.1} = 0.18$$

$$\text{Accuracy} = \frac{991}{1000} = 0.991$$

Alternative Measures

	PREDICTED CLASS	
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	10	40

$$\text{Precision (p)} = 0.8$$
$$\text{Recall (r)} = 0.8$$
$$\text{F - measure (F)} = 0.8$$
$$\text{Accuracy} = 0.8$$

Alternative Measures (balanced vs unbalanced)

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	40	10
Class=No	10	40

$$\text{Precision (p)} = 0.8$$
$$\text{Recall (r)} = 0.8$$
$$\text{F - measure (F)} = 0.8$$
$$\text{Accuracy} = 0.8$$

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	40	10
Class=No	1000	4000

$$\text{Precision (p)} =\sim 0.04$$
$$\text{Recall (r)} = 0.8$$
$$\text{F - measure (F)} =\sim 0.08$$
$$\text{Accuracy} =\sim 0.8$$

Measures of Classification Performance

	PREDICTED CLASS	
ACTUAL CLASS	Yes	No
Yes	TP	FN
No	FP	TN

α is the probability that we reject the null hypothesis when it is true. This is a Type I error or a false positive (FP).

β is the probability that we accept the null hypothesis when it is false. This is a Type II error or a false negative (FN).

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

$$ErrorRate = 1 - accuracy$$

$$Precision = Positive\ Predictive\ Value = \frac{TP}{TP + FP}$$

$$Recall = Sensitivity = TP\ Rate = \frac{TP}{TP + FN}$$

$$Specificity = TN\ Rate = \frac{TN}{TN + FP}$$

$$FP\ Rate = \alpha = \frac{FP}{TN + FP} = 1 - specificity$$

$$FN\ Rate = \beta = \frac{FN}{FN + TP} = 1 - sensitivity$$

$$Power = sensitivity = 1 - \beta$$

Alternative Measures

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	10	40

$$\text{Precision (p)} = 0.8$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.2$$
$$\text{F - measure (F)} = 0.8$$
$$\text{Accuracy} = 0.8$$

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	1000	4000

$$\text{Precision (p)} =\sim 0.04$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.2$$
$$\text{F - measure (F)} =\sim 0.08$$
$$\text{Accuracy} =\sim 0.8$$

Alternative Measures

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	10	40
	Class=No	10	40

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.2$$
$$\text{FPR} = 0.2$$

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	25	25
	Class=No	25	25

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.5$$

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	40	10
	Class=No	40	10

Exercise: for the left table, what is precision, TRP, and FPR?

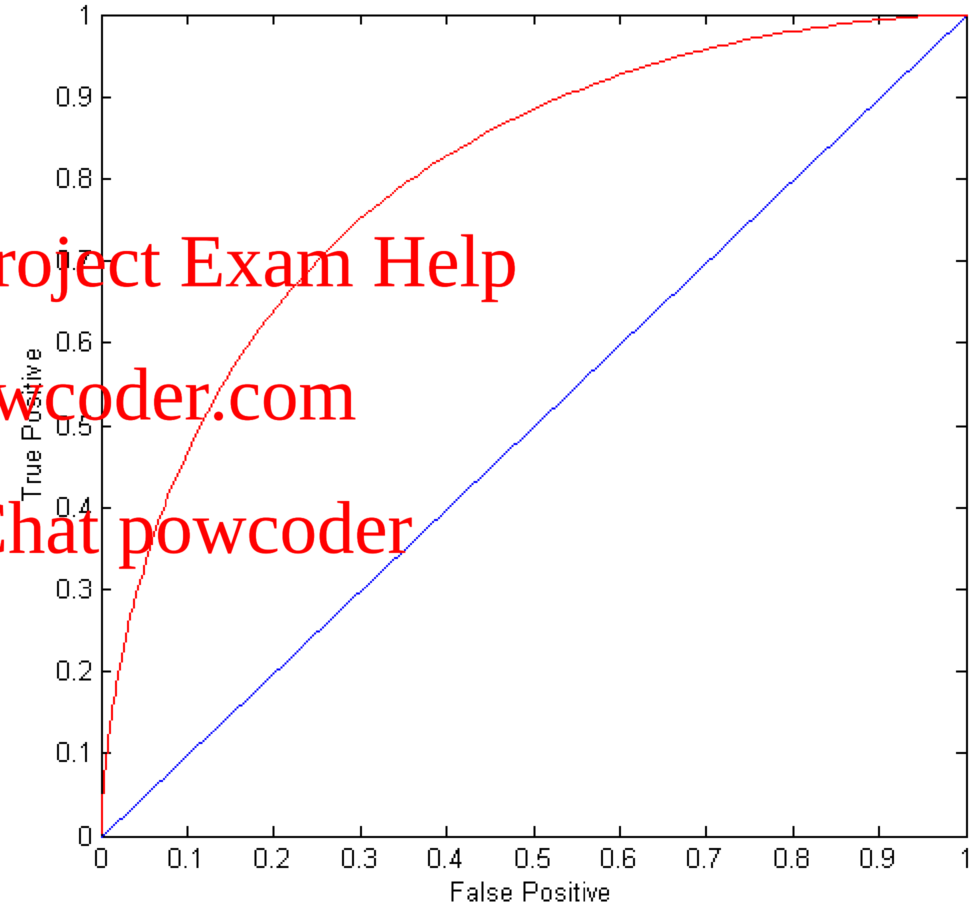
ROC (Receiver Operating Characteristic)

- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - Performance of a model represented as a point in an ROC curve
  - Changing the threshold parameter of classifier changes the location of the point

ROC Curve

(TPR,FPR):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal



- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

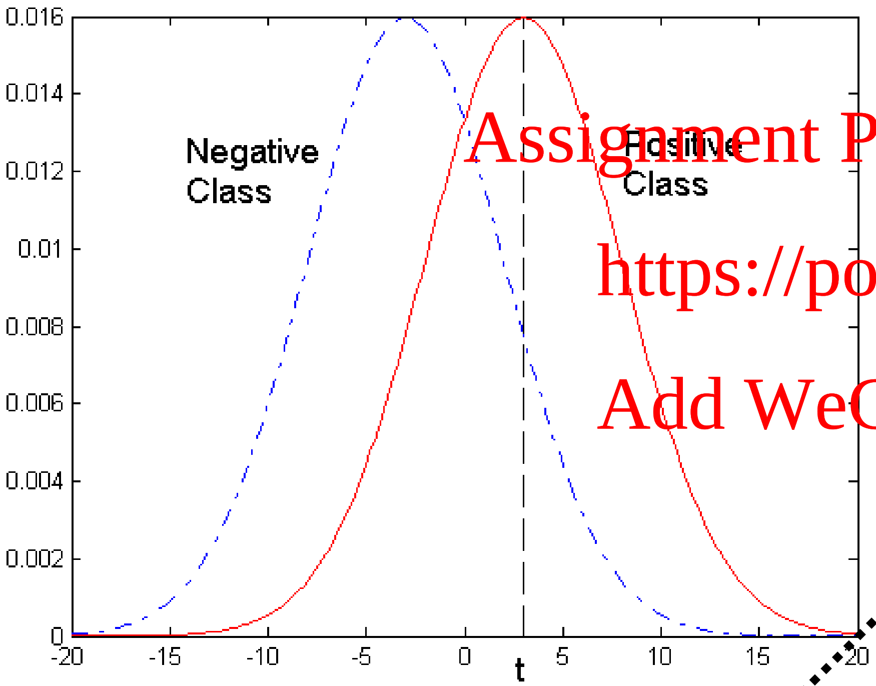
ROC (Receiver Operating Characteristic)

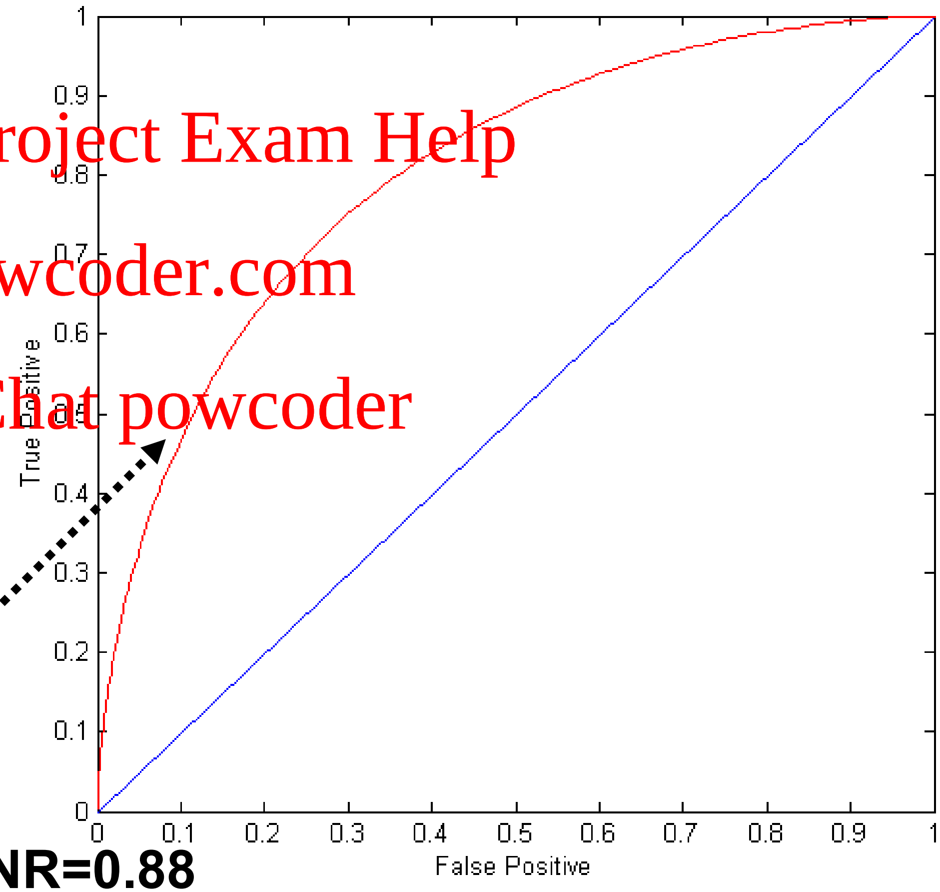
- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record

- Many classifiers produce only discrete outputs (i.e., predicted class)
  - How to get continuous-valued outputs?
    - Decision trees, rule-based classifiers, neural networks, Bayesian classifiers, k-nearest neighbors, SVM

ROC Curve Example

- 1-dimensional data set containing 2 classes (positive and negative)
- Any points located at x > t is classified as positive

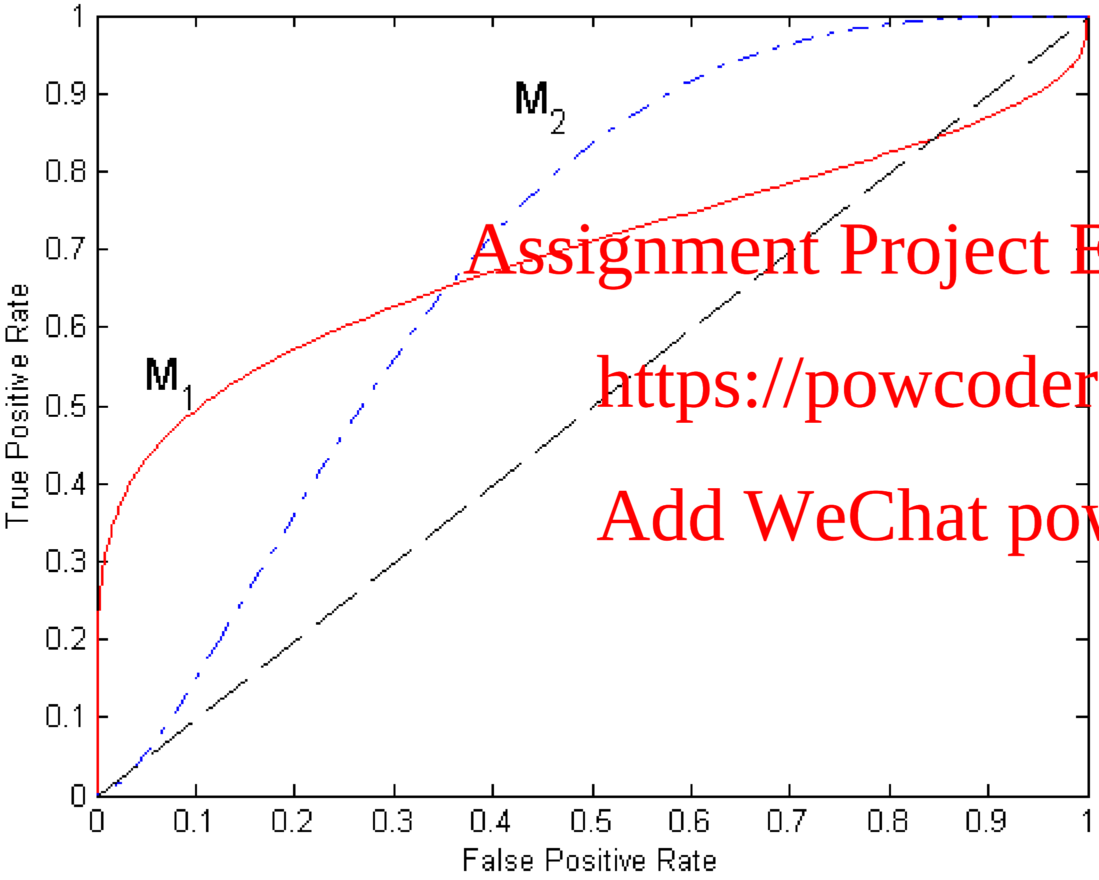




At threshold t:

TPR=0.5, FNR=0.5, FPR=0.12, TNR=0.88

Using ROC for Model Comparison



- No model consistently outperform the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

How to Construct an ROC curve

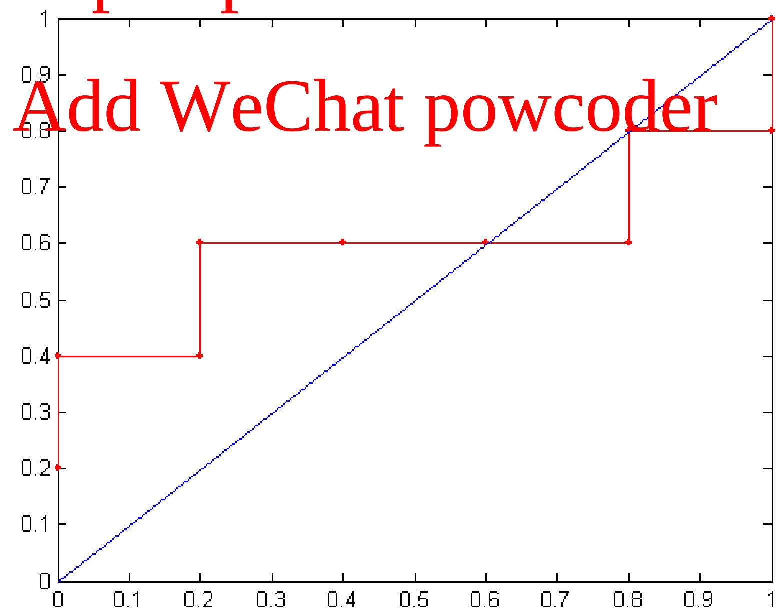
Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use a classifier that produces a continuous-valued score for each instance
  - The more likely it is for the instance to be in the + class, the higher the score
- Sort the instances in decreasing order according to the score
- Apply a threshold at each unique value of the score
- Count the number of TP, FP, TN, FN at each threshold
  - TPR = TP/(TP+FN)
  - FPR = FP/(FP + TN)

How to construct an ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Handling Class Imbalanced Problem

- Class-based ordering (e.g. RIPPER)
  - Rules for rare class have higher priority

- Cost-sensitive classification
  - Misclassifying rare class as majority class is more expensive than misclassifying majority as rare class

- Sampling-based approaches

Sampling-based Approaches

- Modify the distribution of training data so that rare class is well-represented in training set
  - Undersample the majority class
  - Oversample the rare class