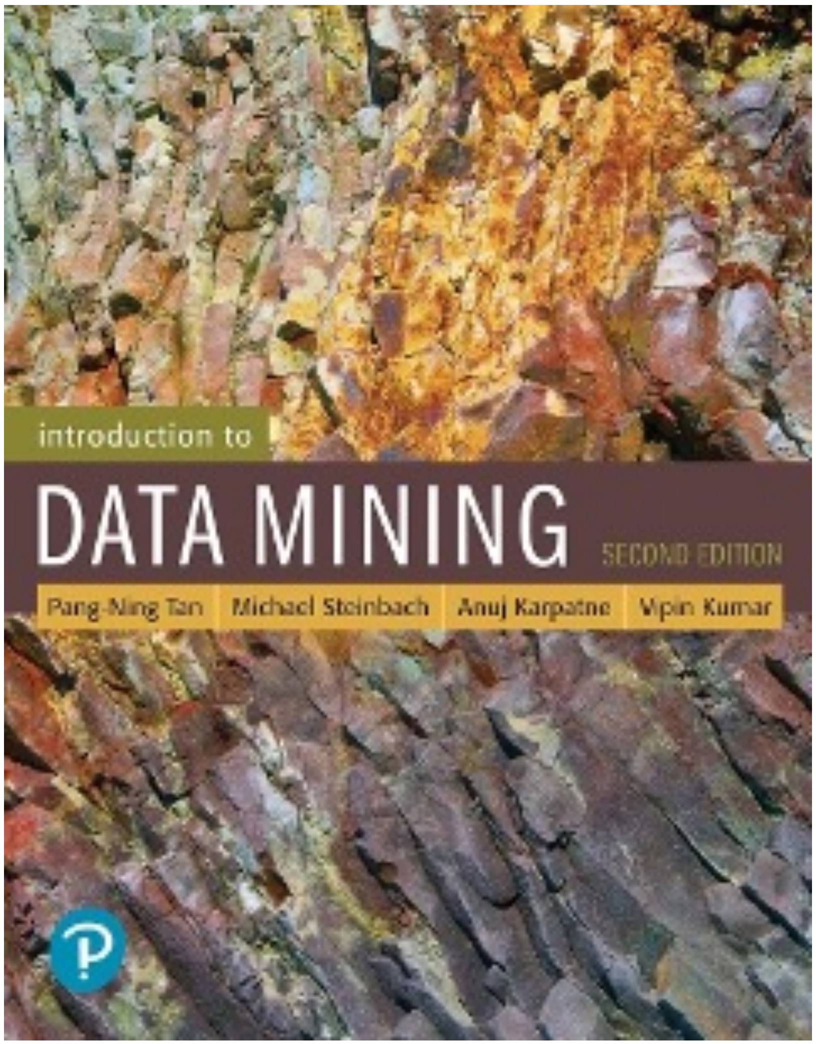
# CSCE 5380/4380 – Data Mining



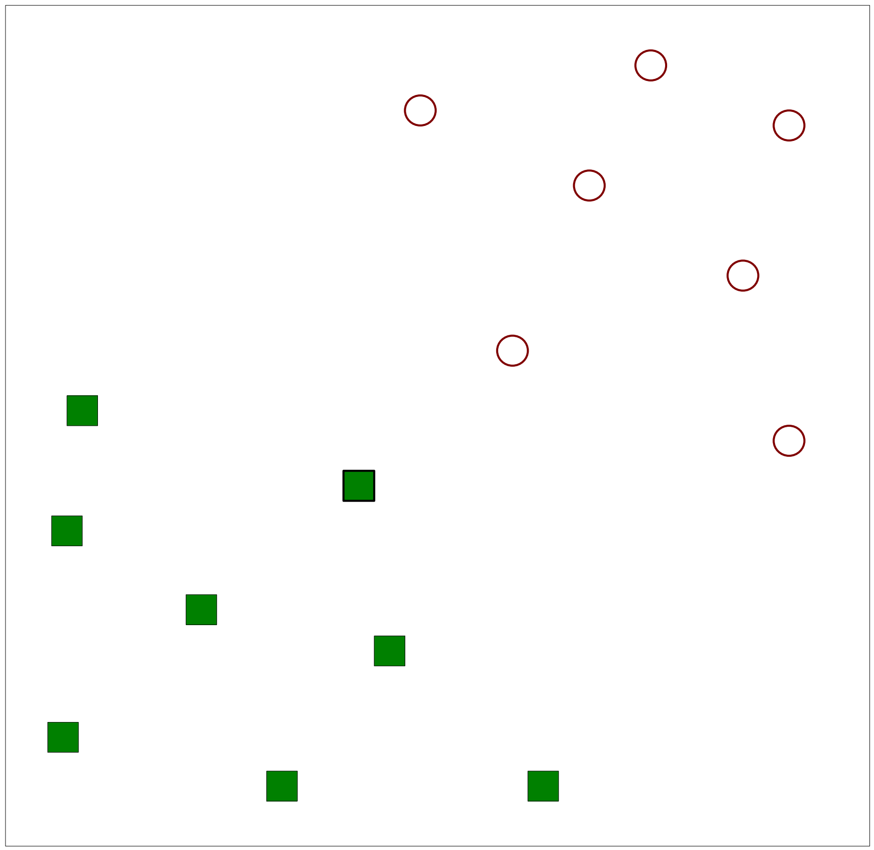

Chapter Four:

**Support Vector Machines & Imbalanced Classes**

# Outline

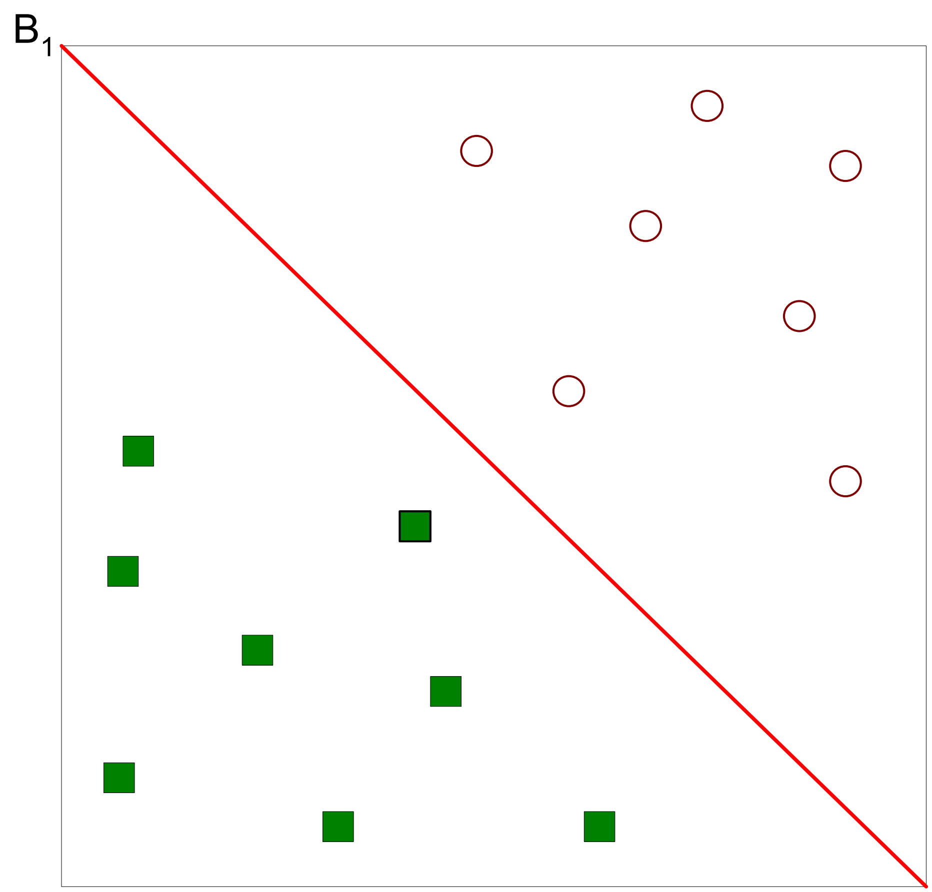
- Support Vector Machines
- Nonlinear Support Vector Machines
- Characteristics of SVM
- Class Imbalance Problem
- Measures of Classification Performance & Imbalanced Classes
- ROC (Receiver Operating Characteristic)
- Building Classifiers with Imbalanced Training Set

# Support Vector Machines



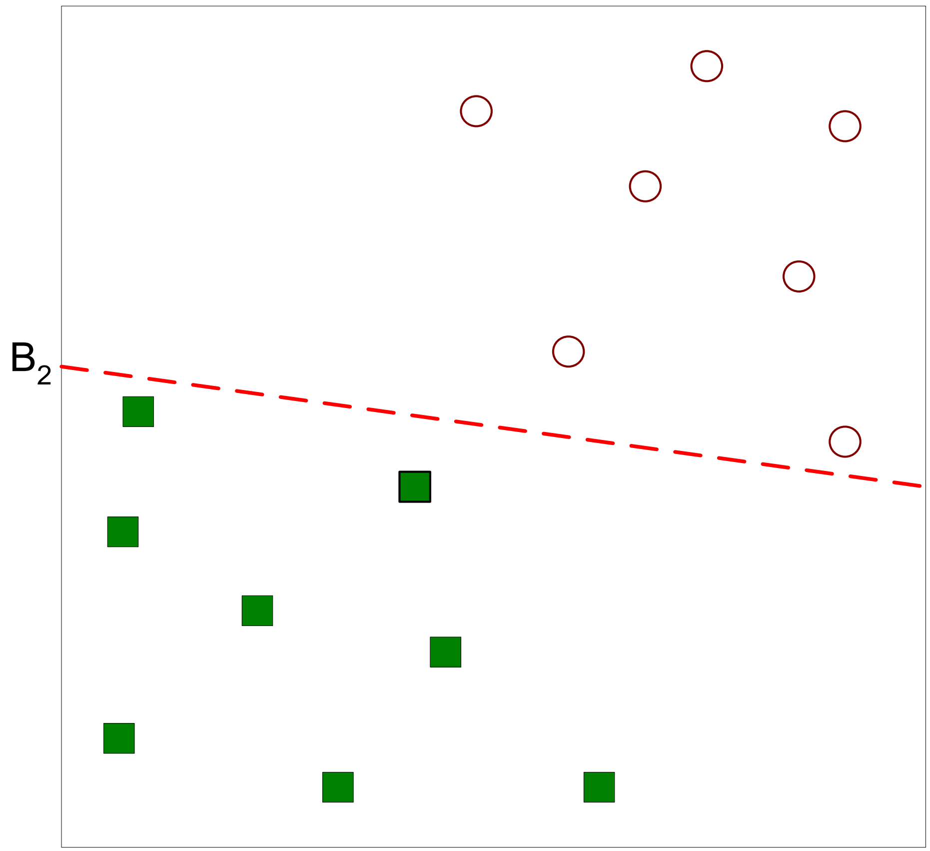
- Find a linear hyperplane (decision boundary) that will separate the data

# Support Vector Machines



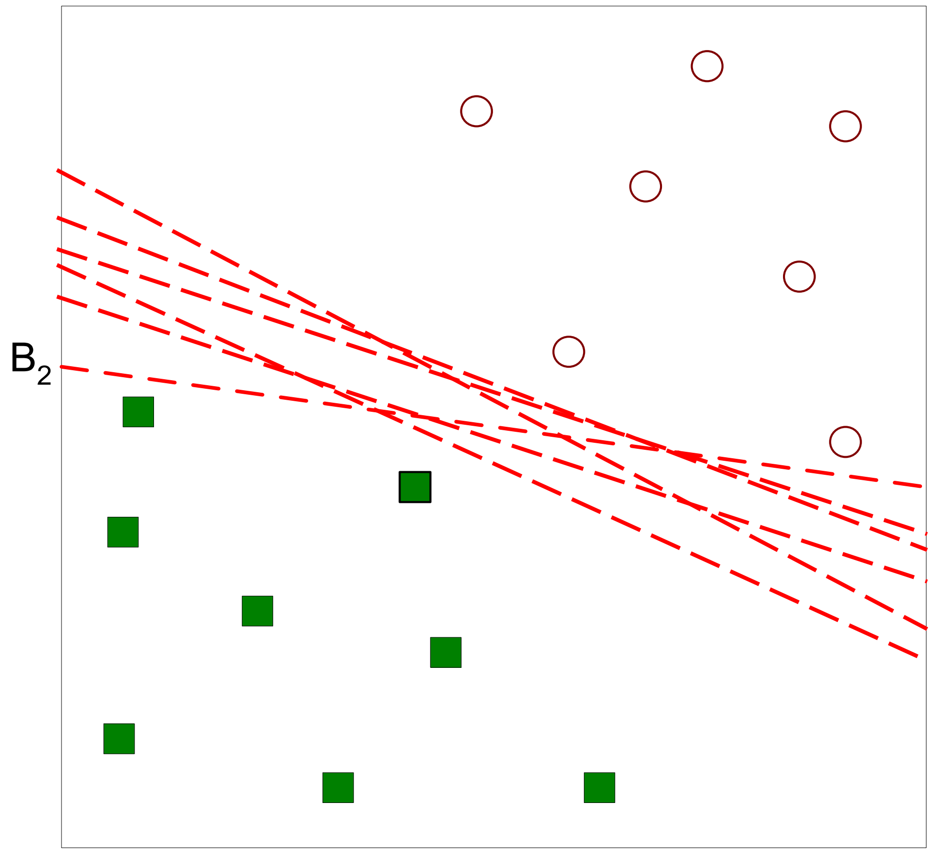

- One Possible Solution

# Support Vector Machines



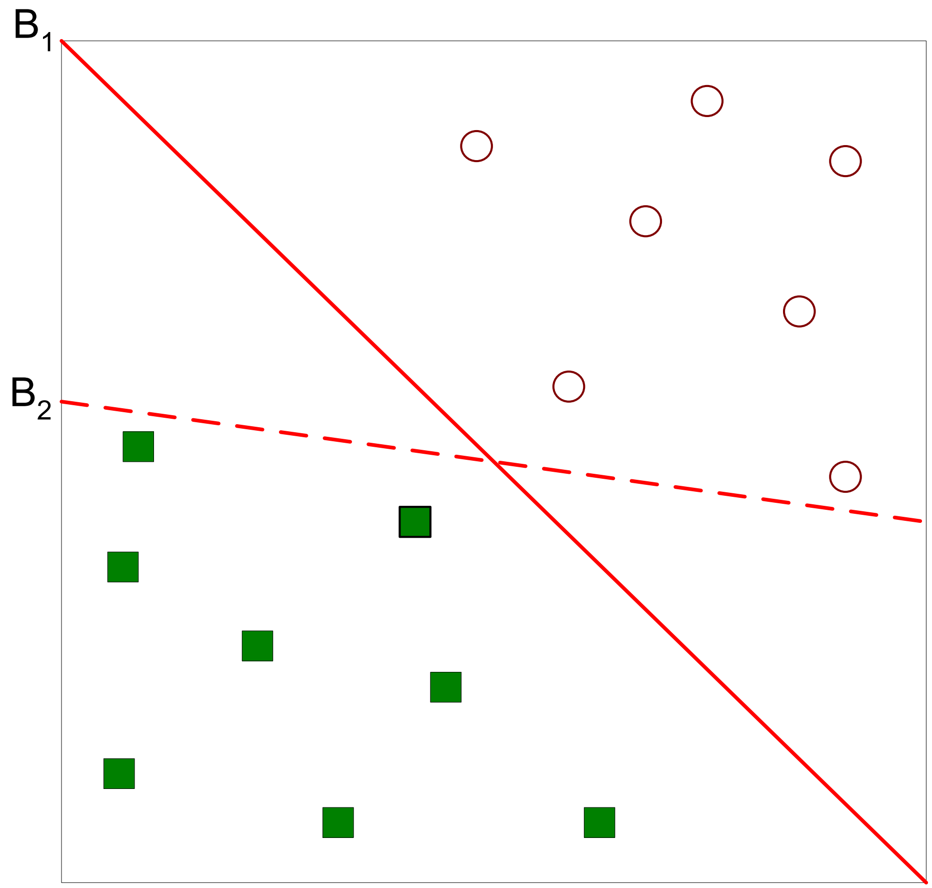

- Another possible solution

# Support Vector Machines



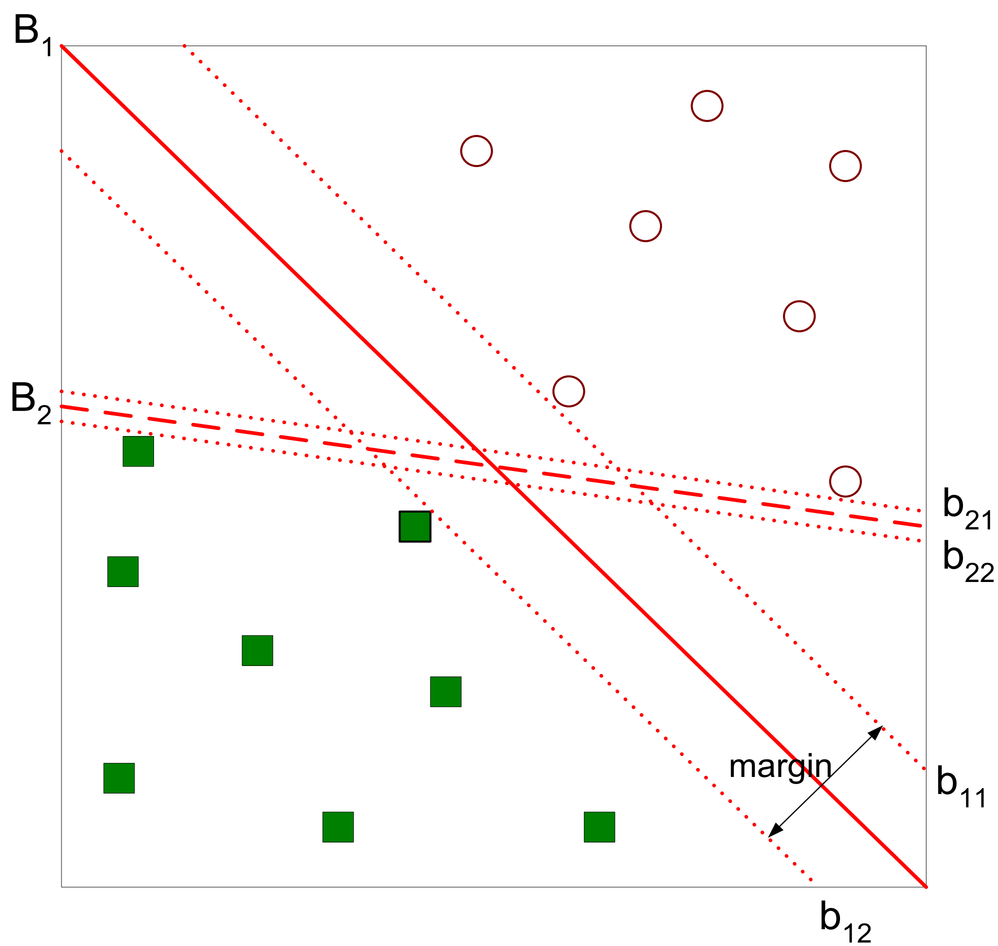

- Other possible solutions

# Support Vector Machines




- Which one is better? B1 or B2?
- How do you define better?

# Support Vector Machines

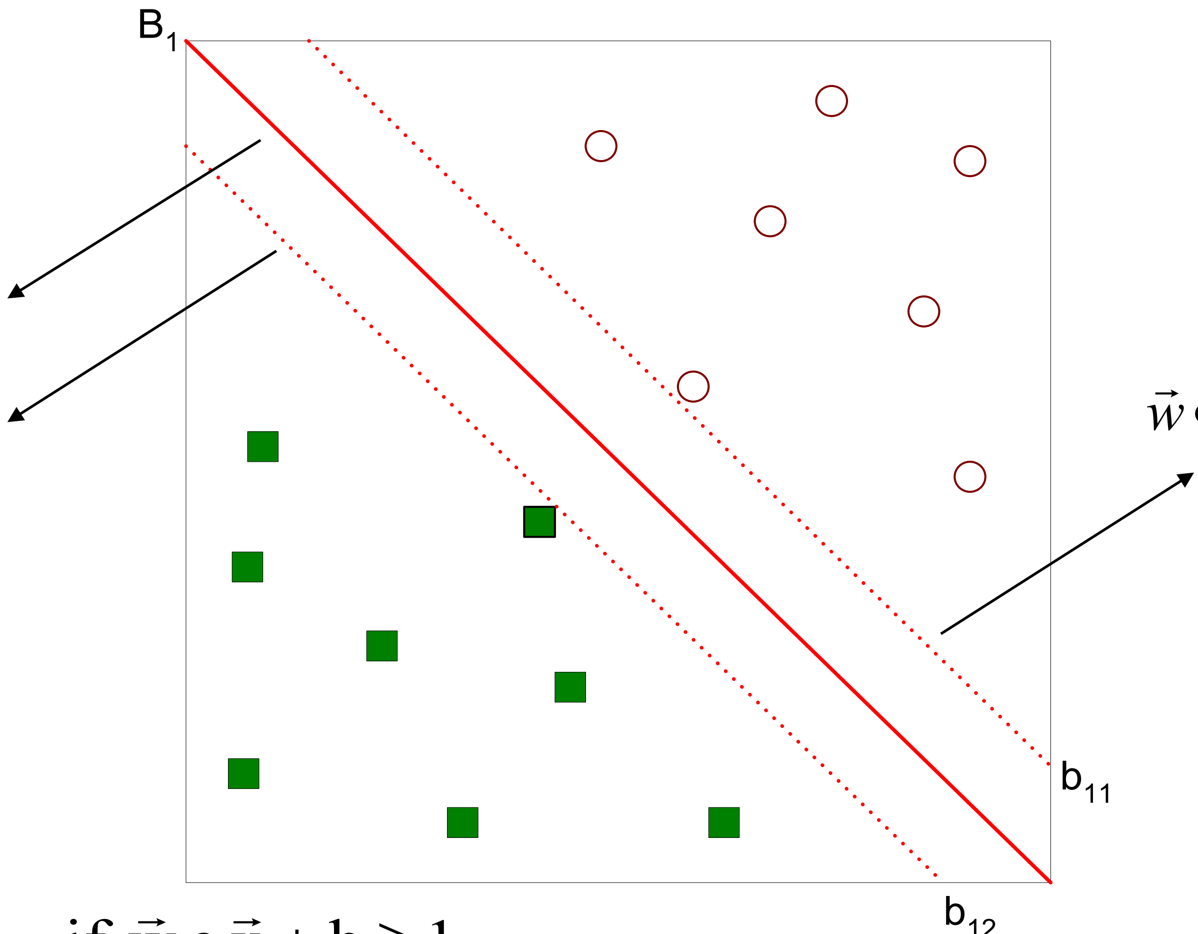



- Find hyperplane maximizes the margin => B1 is better than B2

# Support Vector Machines

$$\vec{w} \bullet \vec{x} + b = 0$$

$$\vec{w} \bullet \vec{x} + b = -1$$




$$\vec{w} \bullet \vec{x} + b = +1$$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\| \vec{w} \|}$$

# Linear SVM

- Linear model:

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}} + \mathrm{b} \leq -1 \end{cases}$$

- Learning the model is equivalent to determining the values of $\vec{w}$ and $b$
  - How to find $\vec{w}$ and $b$ from training data?

# Learning Linear SVM

- Objective is to maximize: $\text{Margin} = \frac{2}{\|\vec{w}\|}$
  - Which is equivalent to minimizing: $L(\vec{w}) = \frac{\|\vec{w}\|^2}{2}$
  - Subject to the following constraints:

$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \leq -1 \end{cases}$$

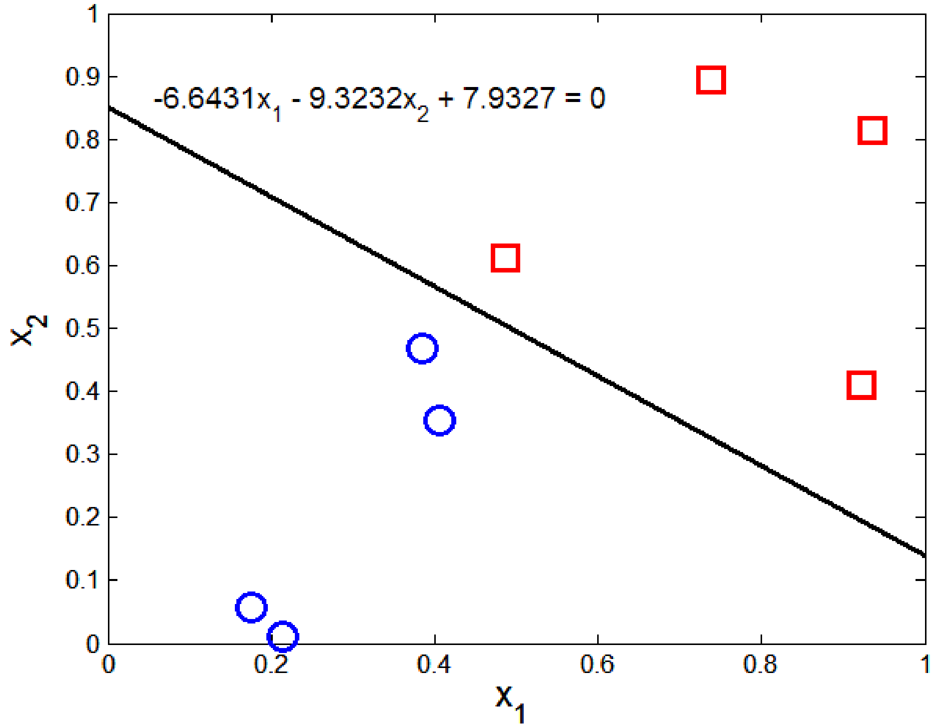
or

$$y_i(\mathrm{w} \bullet \mathrm{x}_i + b) \geq 1, \qquad i = 1,2,\ldots,N$$

◆ This is a constrained optimization problem

– Solve it using Lagrange multiplier method

# Example of Linear SVM




Support vectors

| x1 | x2 | y | λ |
|---|---|---|---|
| 0.3858 | 0.4687 | 1 | 65.5261 |
| 0.4871 | 0.611 | -1 | 65.5261 |
| 0.9218 | 0.4103 | -1 | 0 |
| 0.7382 | 0.8936 | -1 | 0 |
| 0.1763 | 0.0579 | 1 | 0 |
| 0.4057 | 0.3529 | 1 | 0 |
| 0.9355 | 0.8132 | -1 | 0 |
| 0.2146 | 0.0099 | 1 | 0 |

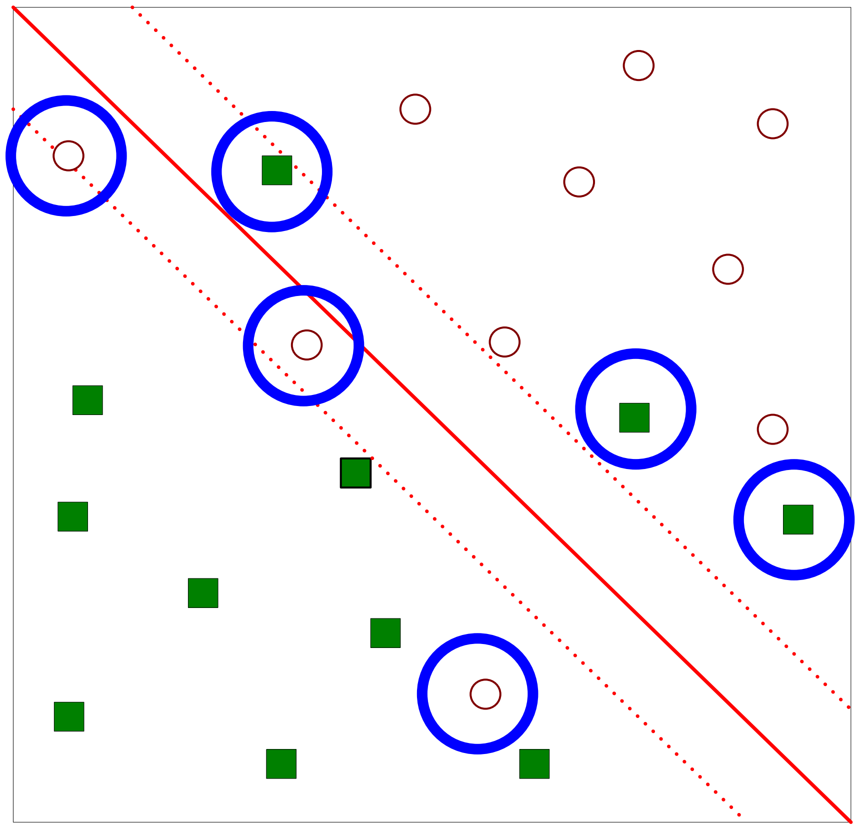
# Learning Linear SVM

- Decision boundary depends only on support vectors
  - If you have data set with same support vectors, decision boundary will not change
  - How to classify using SVM once **w** and *b* are found? Given a test record, $x_i$

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

# Support Vector Machines

- What if the problem is not linearly separable?



# Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

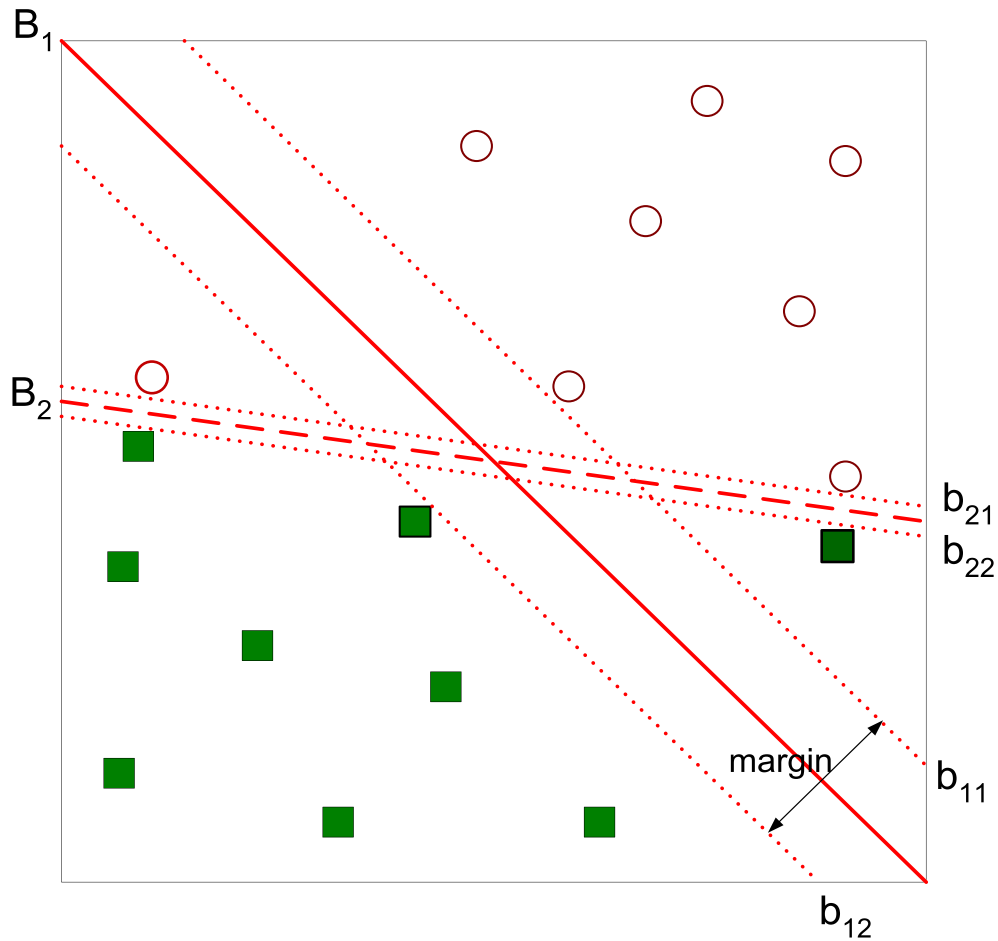
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k\right)$$

    - Subject to:

$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 - \xi_{\mathrm{i}} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 + \xi_{\mathrm{i}} \end{cases}$$

    - If k is 1 or 2, this leads to similar objective function as linear SVM but with different constraints (see textbook)

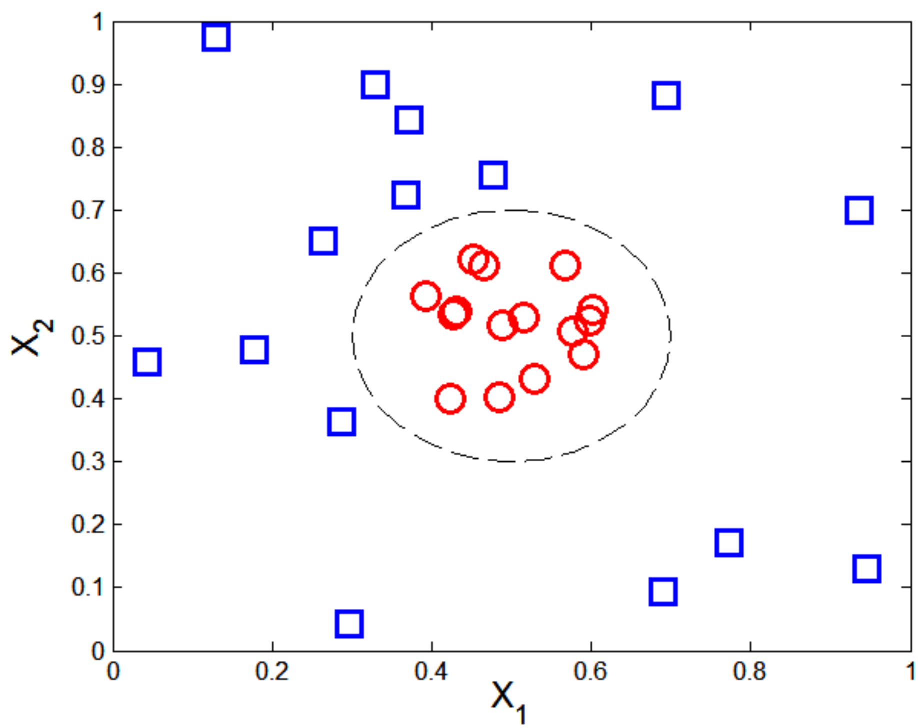
# Support Vector Machines




- Find the hyperplane that optimizes both factors

# Nonlinear Support Vector Machines

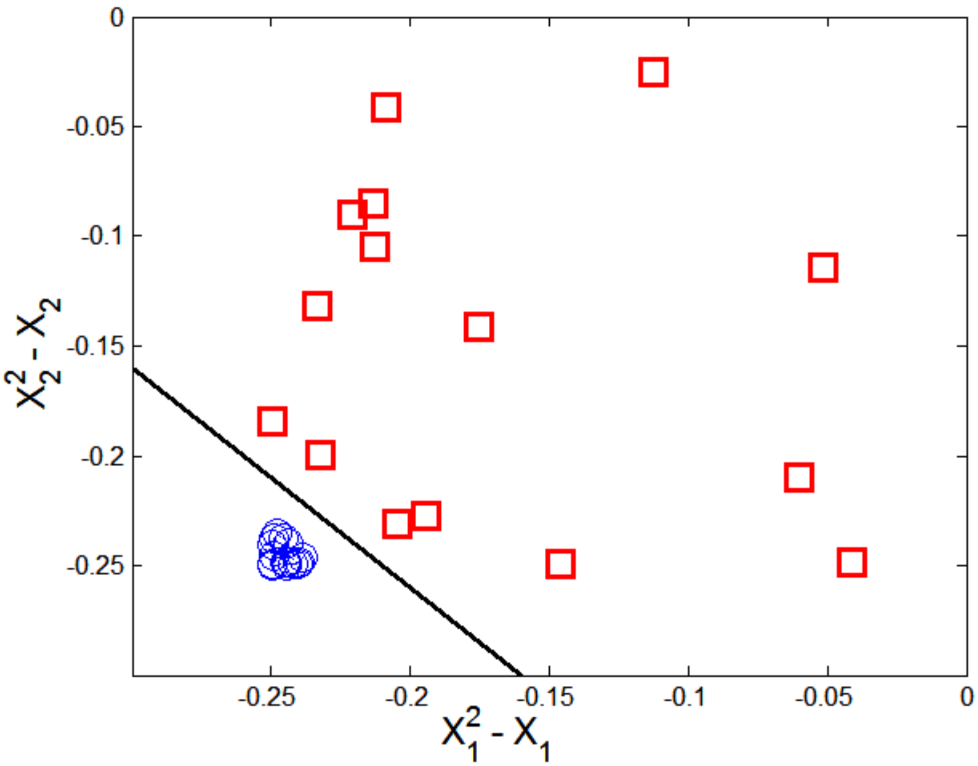
- What if decision boundary is not linear?




$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & \text{otherwise} \end{cases}$$

# Nonlinear Support Vector Machines

- Transform data into higher dimensional space




$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46.$$

$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1).$$

$$w_4 x_1^2 + w_3 x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

**Decision boundary:**

$$\vec{w} \bullet \Phi(\vec{x}) + b = 0$$

# Learning Nonlinear SVM

- Optimization problem:

$$\min_{w} \frac{\|\mathbf{w}\|^2}{2}$$

$$\textit{subject to} \quad y_i(w \cdot \Phi(x_i) + b) \geq 1, \ \forall\{(x_i, y_i)\}$$

- Which leads to the same set of equations (but involve $\Phi$(x) instead of x)

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

$$\mathbf{w} = \sum_i \lambda_i y_i \Phi(\mathbf{x}_i)$$

$$\lambda_i \{y_i (\sum_j \lambda_j y_j \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_i) + b) - 1\} = 0,$$

$$f(\mathbf{z}) = sign(\mathbf{w} \cdot \Phi(\mathbf{z}) + b) = sign(\sum_{i=1}^{n} \lambda_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{z}) + b).$$

# Learning NonLinear SVM

- Issues:
  - What type of mapping function $\Phi$ should be used?
  - How to do the computation in high dimensional space?
    - Most computations involve dot product $\Phi(x_i) \bullet \Phi(x_j)$
    - Curse of dimensionality?

# Learning Nonlinear SVM

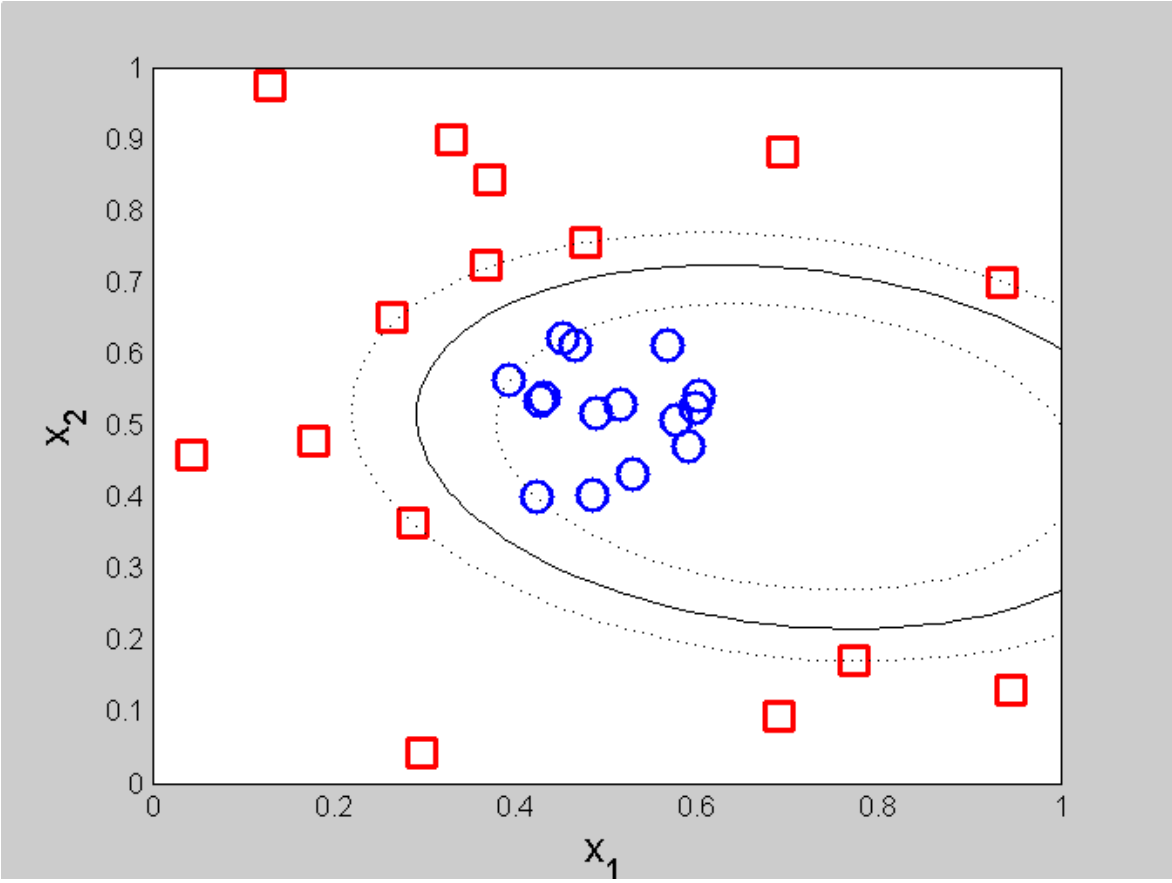
- Kernel Trick:
  - $\Phi(x_i) \bullet \Phi(x_j) = K(x_i, x_j)$
  - $K(x_i, x_j)$ is a kernel function (expressed in terms of the coordinates in the original space)
    - Examples:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$$

$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/(2\sigma^2)}$$

$$K(\mathbf{x}, \mathbf{y}) = \tanh(k\mathbf{x} \cdot \mathbf{y} - \delta)$$

# Example of Nonlinear SVM




**SVM with polynomial degree 2 kernel**

# Learning Nonlinear SVM

- Advantages of using kernel:
  – Don't have to know the mapping function $\Phi$
  – Computing dot product $\Phi(x_i) \bullet \Phi(x_j)$ in the original space avoids curse of dimensionality

- Not all functions can be kernels
  – Must make sure there is a corresponding $\Phi$ in some high-dimensional space
  – Mercer's theorem (see textbook)

# Characteristics of SVM

- The learning problem is formulated as a convex optimization problem
  - Efficient algorithms are available to find the global minima
  - Many of the other methods use greedy approaches and find locally optimal solutions
  - High computational complexity for building the model

- Robust to noise
- Overfitting is handled by maximizing the margin of the decision boundary,
- SVM can handle irrelevant and redundant attributes better than many other techniques
- The user needs to provide the type of kernel function and cost function
- Difficult to handle missing values

- What about categorical variables?

# 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
  - COVID-19 test results on a random sample

- **Key Challenge**:
  - Evaluation measures such as accuracy are not well-suited for imbalanced class

# Measures of Classification Performance

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

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

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

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

$$ErrorRate = 1 - accuracy$$

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

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

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

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

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

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

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

| | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 0 | 10 |
| | Class=No | 0 | 990 |

# Which model is better?

A

| | PREDICTED | |
|---|---|---|
| ACTUAL | Class=Yes | Class=No |
| Class=Yes | 0 | 10 |
| Class=No | 0 | 990 |

**Accuracy: 99%**

B

| | PREDICTED | |
|---|---|---|
| ACTUAL | Class=Yes | Class=No |
| Class=Yes | 10 | 0 |
| Class=No | 500 | 490 |

**Accuracy: 50%**

# 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 |
| ACTUAL CLASS 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 | |
|---|---|---|
| | Class=Yes | Class=No |
| ACTUAL CLASS Class=Yes | 1 | 9 |
| ACTUAL CLASS Class=No | 0 | 990 |

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

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

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

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

# Which of these classifiers is better?

**A**

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

$\text{Precision (p)} = 0.8$

$\text{Recall (r)} = 0.8$

$\text{F-measure (F)} = 0.8$

$\text{Accuracy} = 0.8$

**B**

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

$\text{Precision (p)} =\sim 0.04$

$\text{Recall (r)} = 0.8$

$\text{F-measure (F)} =\sim 0.08$

$\text{Accuracy} =\sim 0.8$

# Alternative Measures

| A | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 40 | 10 |
| | Class=No | 10 | 40 |

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

$$\frac{\text{TPR}}{\text{FPR}} = 4$$

| B | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 40 | 10 |
| | Class=No | 1000 | 4000 |

$$\text{Precision (p)} = 0.038$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.2$$
$$\text{F−measure (F)} = 0.07$$
$$\text{Accuracy} = 0.8$$

$$\frac{\text{TPR}}{\text{FPR}} = 4$$

# ROC (Receiver Operating Characteristic)

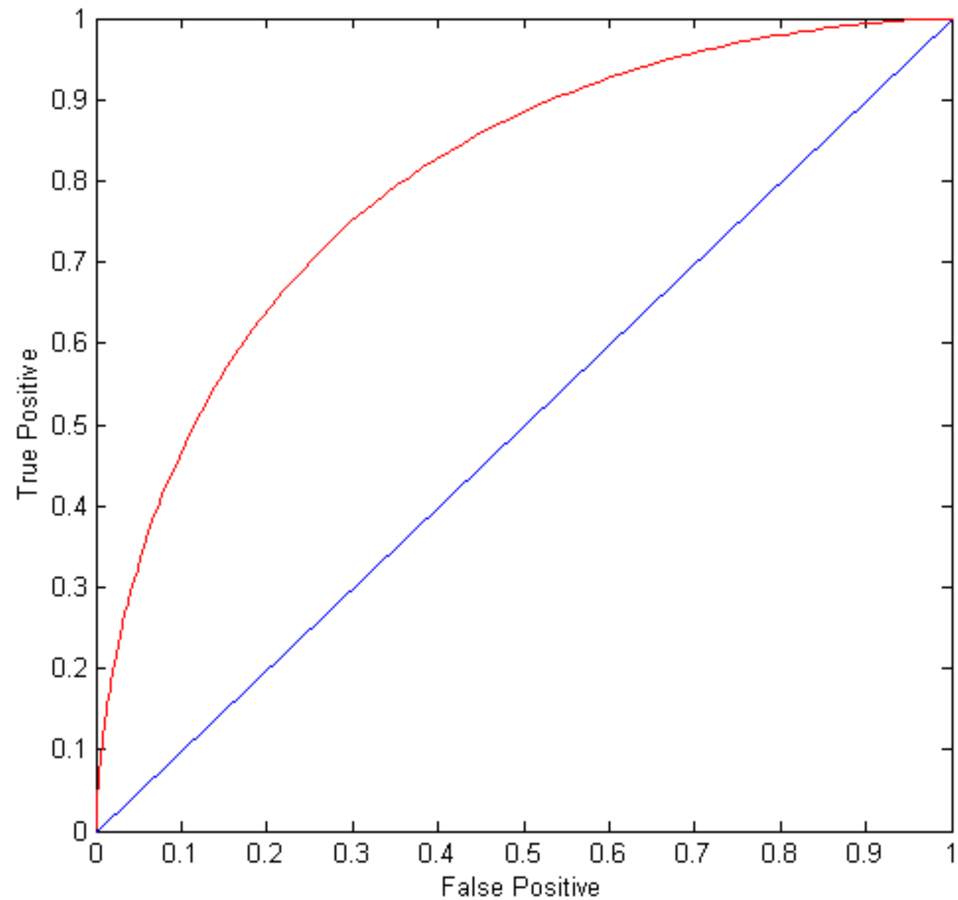
- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - Performance of a model represented as a point in an ROC curve

# 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

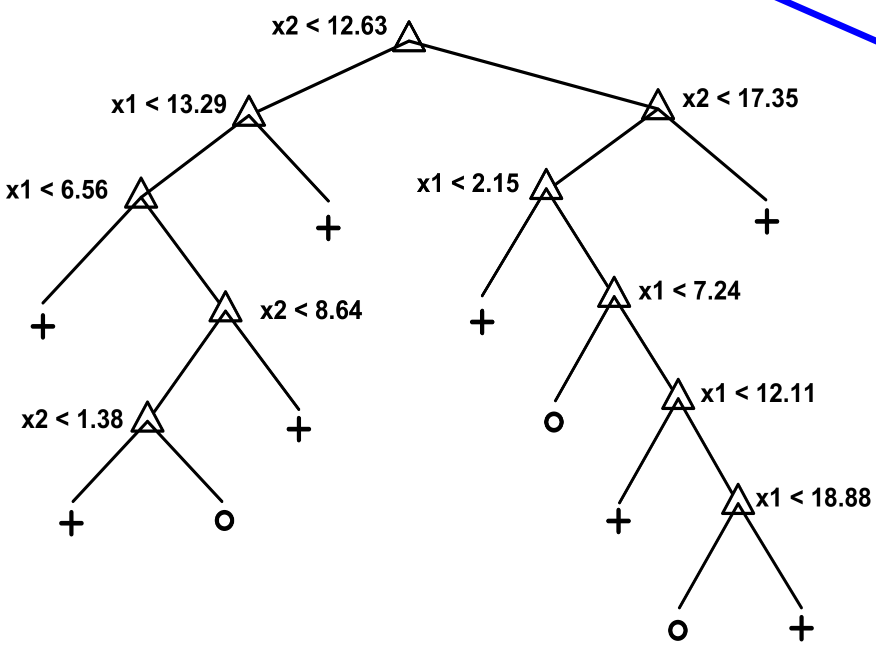


# ROC (Receiver Operating Characteristic)

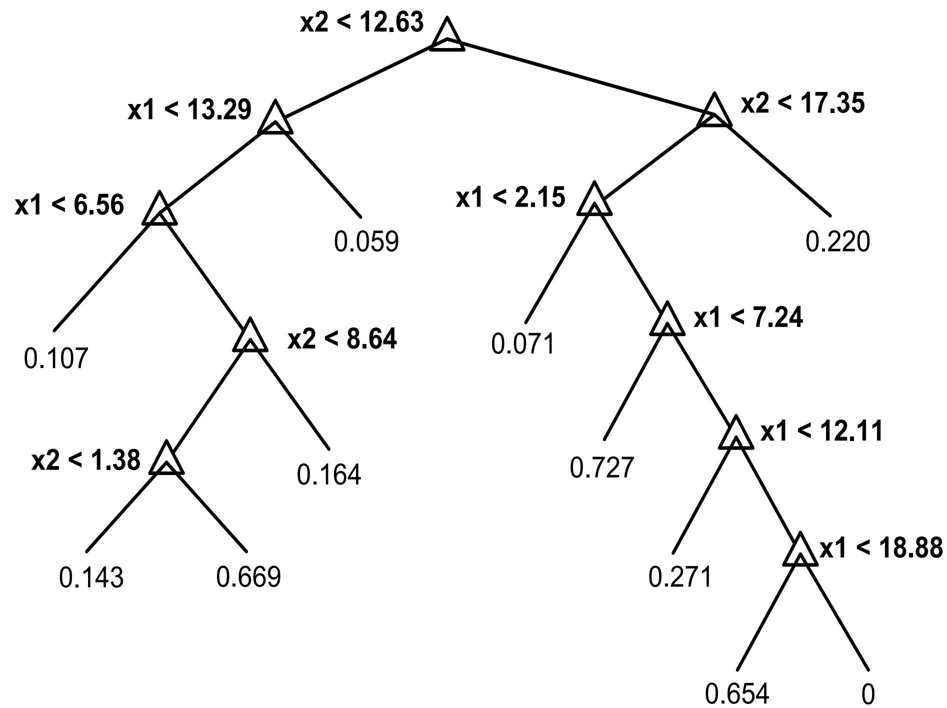
- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record
  - By using different thresholds on this value, we can create different variations of the classifier with TPR/FPR tradeoffs
- 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

# Example: Decision Trees

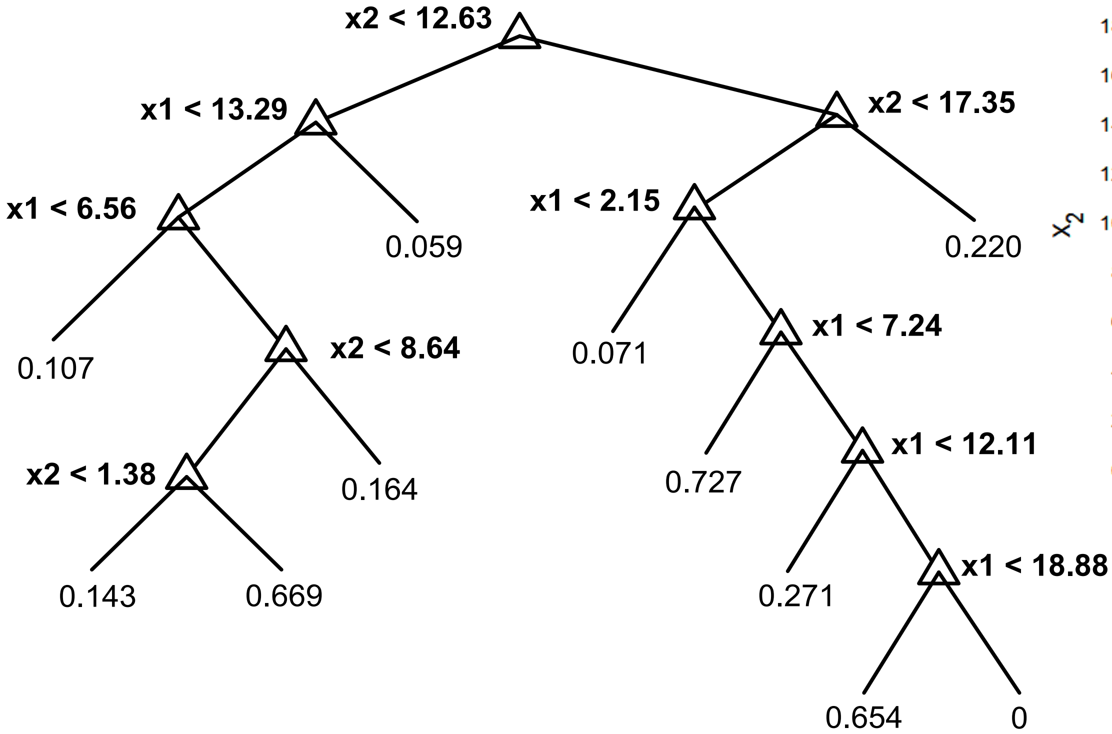
Decision Tree

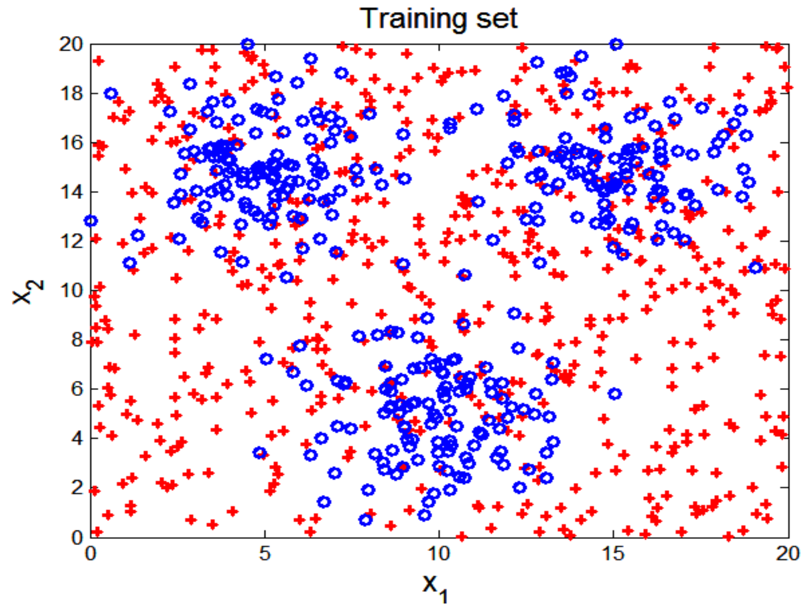



Continuous-valued outputs



# ROC Curve Example



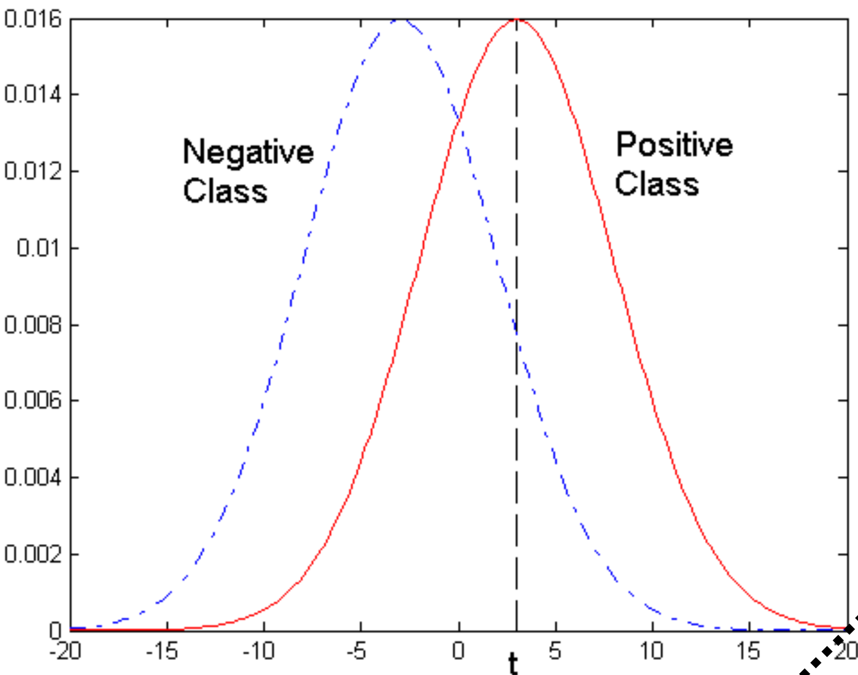



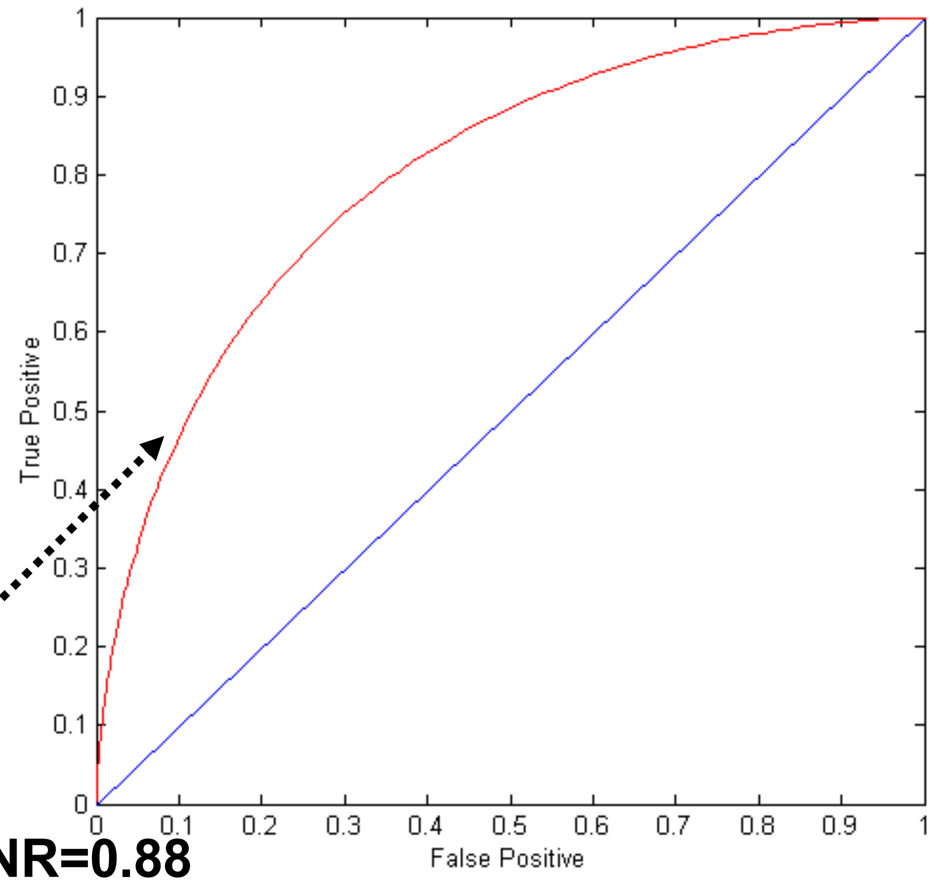

| $\alpha = 0.3$ | | Predicted Class | |
|---|---|---|---|
| | | Class o | Class + |
| Actual Class | Class o | 645 | 209 |
| | Class + | 298 | 948 |

| $\alpha = 0.7$ | | Predicted Class | |
|---|---|---|---|
| | | Class o | Class + |
| Actual Class | Class o | 181 | 673 |
| | Class + | 78 | 1168 |

# 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**

# How to Construct an ROC curve

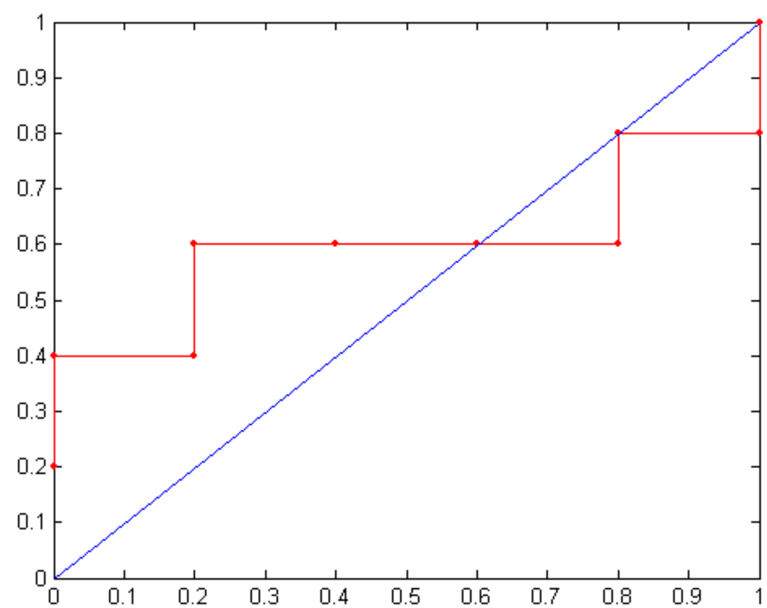
| Instance | Score | True Class |
|---|---|---|
| 1 | 0.95 | + |
| 2 | 0.93 | + |
| 3 | 0.87 | - |
| 4 | 0.85 | - |
| 5 | 0.85 | - |
| 6 | 0.85 | + |
| 7 | 0.76 | - |
| 8 | 0.53 | + |
| 9 | 0.43 | - |
| 10 | 0.25 | + |

- 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)

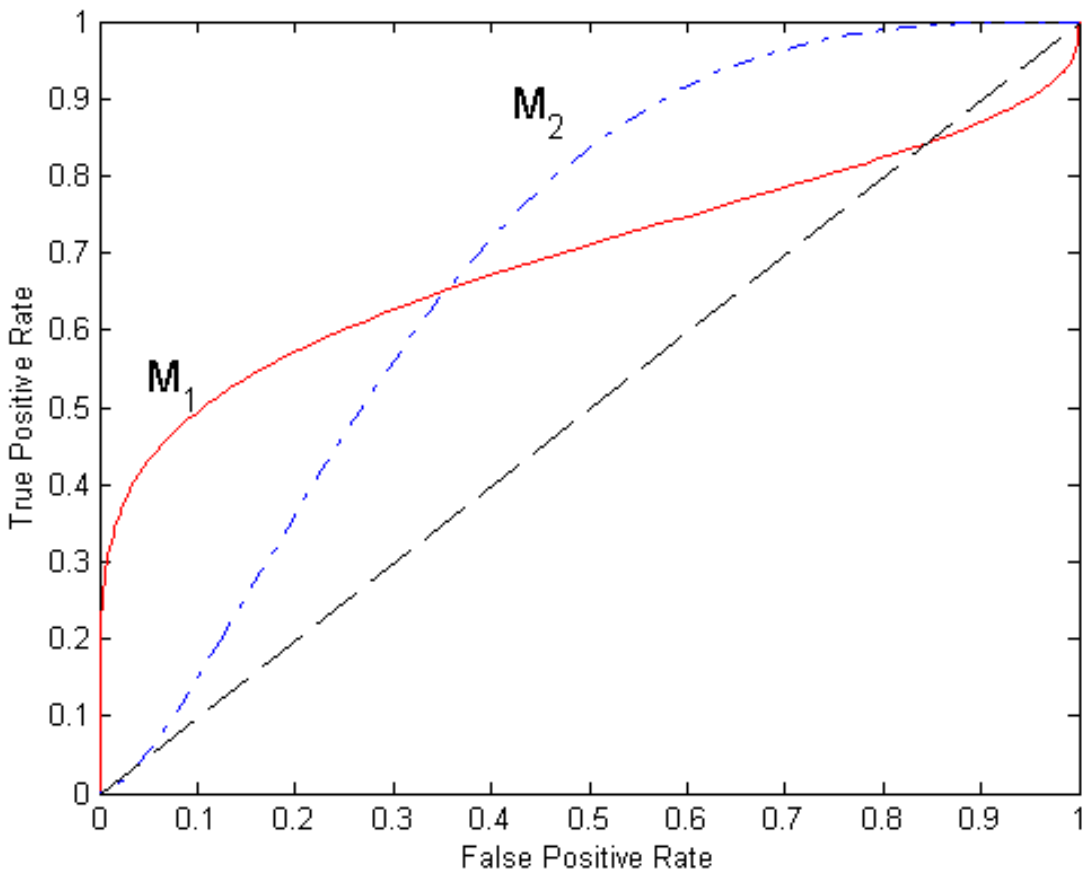
# How to construct an ROC curve

| 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:



# Using ROC for Model Comparison




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

# Dealing with Imbalanced Classes - Summary

- Many measures exists, but none of them may be ideal in all situations
  - Random classifiers can have high value for many of these measures
  - TPR/FPR provides important information but may not be sufficient by itself in many practical scenarios
  - Given two classifiers, sometimes you can tell that one of them is strictly better than the other
    - C1 is strictly better than C2 if C1 has strictly better TPR and FPR relative to C2 (or same TPR and better FPR, and vice versa)
  - Even if C1 is strictly better than C2, C1's F-value can be worse than C2's if they are evaluated on data sets with different imbalances
  - Classifier C1 can be better or worse than C2 depending on the scenario at hand (class imbalance, importance of TP vs FP, cost/time tradeoffs)

# Which Classifier is better? Low Skew case

| T1 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 50 | 50 |
| | Class=No | 1 | 99 |

$$\text{Precision (p)} = 0.98$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F} - \text{measure} = 0.66$$

| T2 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 10 | 90 |

$$\text{Precision (p)} = 0.9$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F} - \text{measure} = 0.94$$

| T3 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 1 | 99 |

$$\text{Precision (p)} = 0.99$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F} - \text{measure} = 0.99$$

# Which Classifier is better? Medium Skew case

| T1 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 50 | 50 |
| | Class=No | 10 | 990 |

$$\text{Precision (p)} = 0.83$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F} - \text{measure} = 0.62$$

| T2 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 100 | 900 |

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F} - \text{measure} = 0.66$$

| T3 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 10 | 990 |

$$\text{Precision (p)} = 0.9$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F} - \text{measure} = 0.94$$

# Which Classifier is better? High Skew case

| T1 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 50 | 50 |
| | Class=No | 100 | 9900 |

$$\text{Precision (p)} = 0.3$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 50$$
$$\text{F} - \text{measure} = 0.375$$

| T2 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 1000 | 9000 |

$$\text{Precision (p)} = 0.09$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.1$$
$$\text{TPR/FPR} = 9.9$$
$$\text{F} - \text{measure} = 0.165$$

| T3 | PREDICTED CLASS | | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | 99 | 1 |
| | Class=No | 100 | 9900 |

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.99$$
$$\text{FPR} = 0.01$$
$$\text{TPR/FPR} = 99$$
$$\text{F} - \text{measure} = 0.66$$

## Building Classifiers with Imbalanced Training Set

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