# Machine Learning: Chenhao Tan

University of Colorado Boulder

LECTURE 12

Slides adapted from Chris Ketelsen

## Logistics

- HW2 available on Github, due in 2 days

## Learning objectives

- Understand the ROC curve and AUC
- Understand inherent multi-class classifiers
- Understand techniques to convert binary classifiers to multi-class classifiers
- A deep dive into regularization (bonus)

## Outline

ROC, AUC

Inherent multi-class classifiers

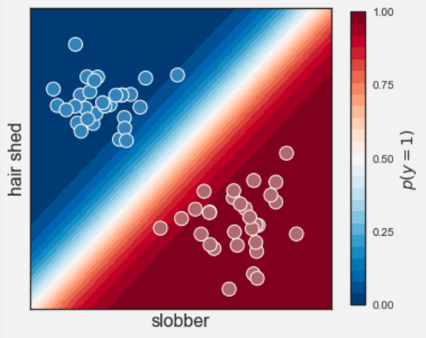
From binary classifiers to multi-class classifiers

Regularization (bonus)

## Prediction score

We have so far assumed all predictions are binary.
We can differentiate the “confidence” of a prediction with its predicted score.
For example, in logistic regression,

$$P(y = 1 \mid \boldsymbol{x}) = \sigma(\beta^T \boldsymbol{x})$$

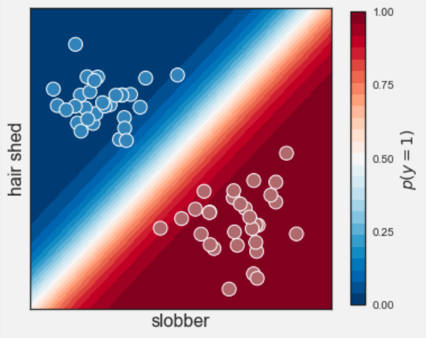


## Prediction score

We have so far assumed all predictions are binary.
We can differentiate the “confidence” of a prediction with its predicted score.
For example, in logistic regression,

$$P(y = 1 \mid \boldsymbol{x}) = \sigma(\beta^T \boldsymbol{x})$$

We have always used 0.5 as a threshold to generate a binary prediction, but choosing the threshold can be tricky for imbalanced classes.



## TPR and FPR

| | | predicted labels | |
|---|---|---|---|
| | | positive (1) | negative (0) |
| true labels | positive (1) | true positive ($TP$) | false negative ($FN$) |
| | negative (0) | false positive ($FP$) | true negative ($TN$) |

- True positive rate, $TPR = \frac{TP}{TP+FN}$
- False positive rate, $FPR = \frac{FP}{FP+TN}$

## TPR and FPR

Example: Suppose you build a logistic regression classifier to predict credit card fraud from recent transactions. Customers would rather be warned even when things are OK than let actual fraud be missed.
This means we're willing to accept a high _____ in order to secure a high _____ by choosing a _____ threshold.

A. TPR, FPR, high

B. FPR, TPR, low

## TPR and FPR

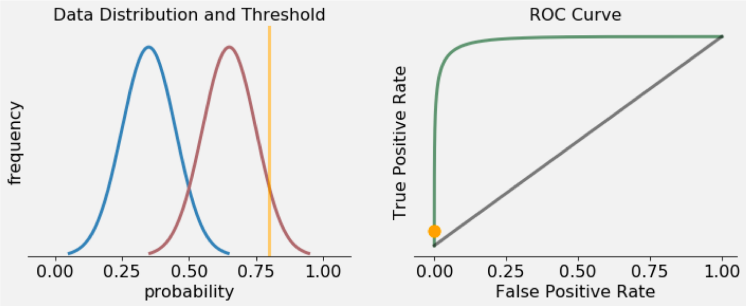
Example: Suppose you build a logistic regression classifier to predict credit card fraud from recent transactions. Customers would rather be warned even when things are OK than let actual fraud be missed.
This means we're willing to accept a high _____ in order to secure a high _____ by choosing a _____ threshold.

A. TPR, FPR, high

B. FPR, TPR, low

The answer is B. A ROC Curve gives us a visual way to evaluate suitable thresholds to fit our needs.

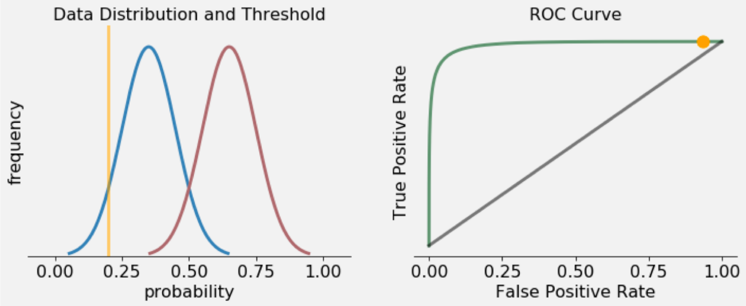
## The ROC curve




A ROC Curve is a plot of FPR (horizontal) vs. TPR (vertical) for all possible threshold values.

Convenient to see how a model would perform at all thresholds simultaneously, rather than looking at misclassification rate for each threshold individually.

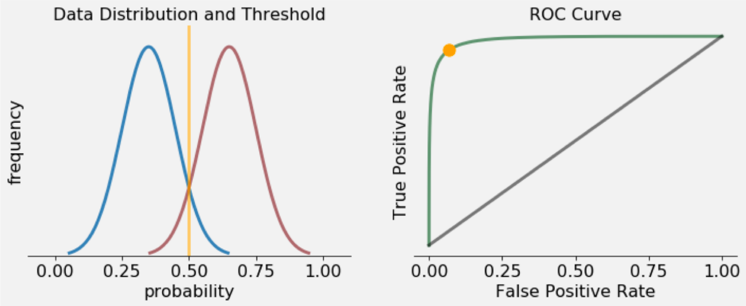
## The ROC curve




A ROC Curve is a plot of FPR (horizontal) vs. TPR (vertical) for all possible threshold values.

Convenient to see how a model would perform at all thresholds simultaneously, rather than looking at misclassification rate for each threshold individually.

## The ROC curve

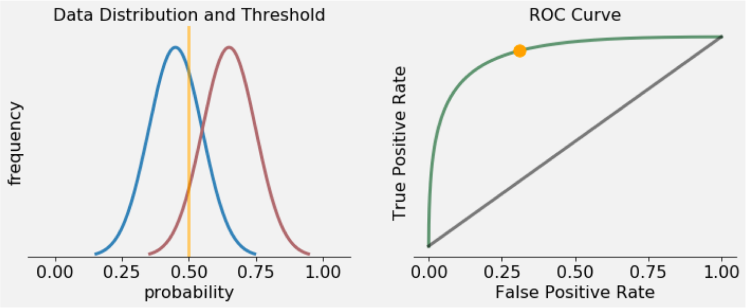



The threshold gives the parameterization of the ROC curve (i.e., it moves the dot).

When the threshold separates the two classes fairly well, the curve is far away from the diagonal.

What happens if we can't separate the classes very well?

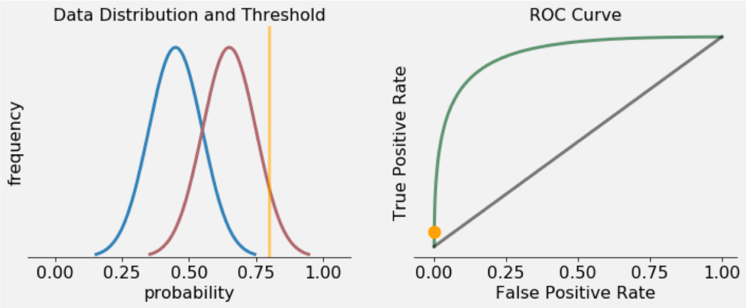
## The ROC curve




Now we're not doing so well at separating the classes.

The ROC curve starts bending towards the center.

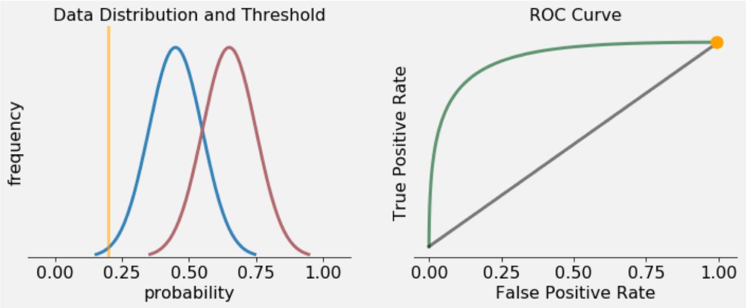
## The ROC curve




Now we're not doing so well at separating the classes.

The ROC curve starts bending towards the center.

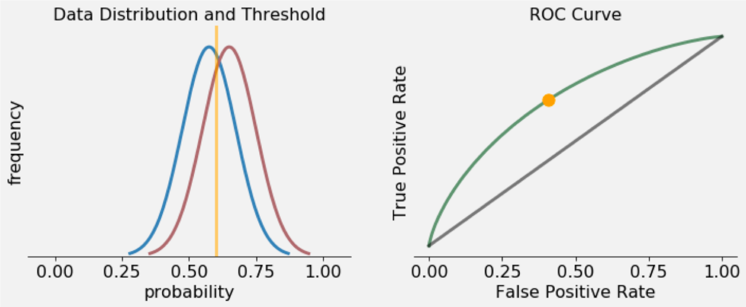
## The ROC curve




Now we're not doing so well at separating the classes.

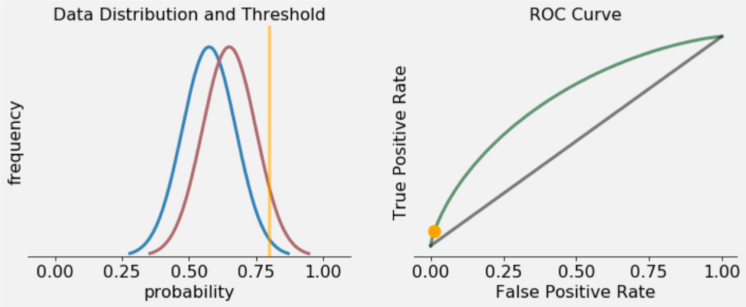
The ROC curve starts bending towards the center.

## The ROC curve



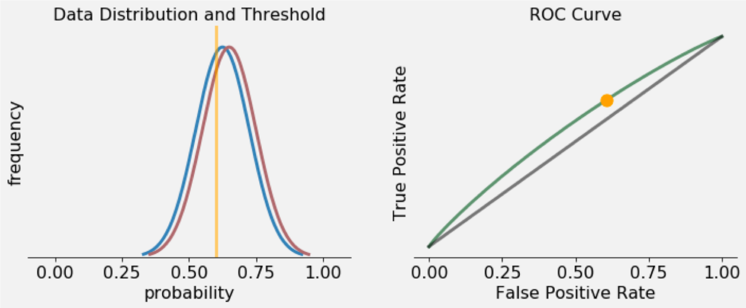

And as we do a poorer job of separating the classes, the curve continues to bend.

## The ROC curve



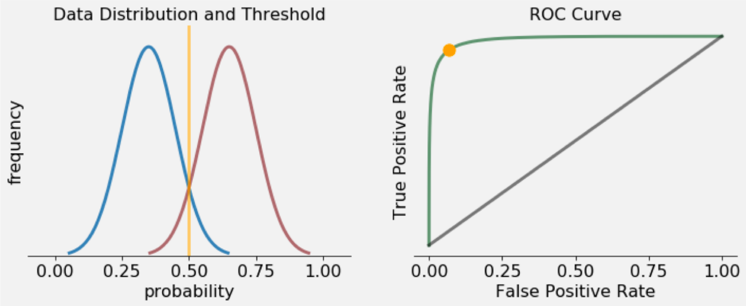

And as we do a poorer job of separating the classes, the curve continues to bend.

## The ROC curve



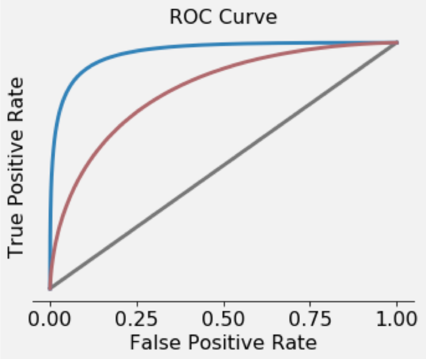

And if we do a terrible job, the curve approaches the random chance line, indicating that our classifier is not much better than a random guess.

## The ROC curve




The ROC curve addresses the cases when we're worried about FPs and TPs simultaneously.
But, if you want a single number, evaluating how the model will do in all cases
You can compute the AUC (Area under the ROC curve).

## ROC-AUC comparisons




To compare two models, plot their ROC curves on the same axes.
If one encloses the other, then it's better on both ends of the spectrum, and has higher AUC.

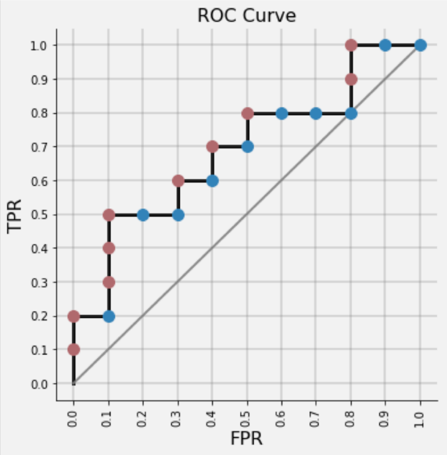
## Constructing a ROC curve

You need a classifier that is able to rank examples by predicted score.

- Order all examples by prediction confidence
- Move threshold to each point, one at a time
- If point is true positive, move vertically (1/NP)
- If point is true negative, move horizontally (1/NN)

| # | $c$ | $\hat{p}$ | # | $c$ | $\hat{p}$ |
|---|---|---|---|---|---|
| 1 | $P$ | 0.90 | 11 | $P$ | 0.40 |
| 2 | $P$ | 0.80 | 12 | $N$ | 0.39 |
| 3 | $N$ | 0.70 | 13 | $P$ | 0.38 |
| 4 | $P$ | 0.60 | 14 | $N$ | 0.37 |
| 5 | $P$ | 0.55 | 15 | $N$ | 0.36 |
| 6 | $P$ | 0.54 | 16 | $N$ | 0.35 |
| 7 | $N$ | 0.53 | 17 | $P$ | 0.34 |
| 8 | $N$ | 0.52 | 18 | $P$ | 0.33 |
| 9 | $P$ | 0.51 | 19 | $N$ | 0.30 |
| 10 | $N$ | 0.50 | 20 | $N$ | 0.10 |

## Constructing a ROC curve




| # | $c$ | $\hat{p}$ | # | $c$ | $\hat{p}$ |
|---|---|---|---|---|---|
| 1 | $P$ | 0.90 | 11 | $P$ | 0.40 |
| 2 | $P$ | 0.80 | 12 | $N$ | 0.39 |
| 3 | $N$ | 0.70 | 13 | $P$ | 0.38 |
| 4 | $P$ | 0.60 | 14 | $N$ | 0.37 |
| 5 | $P$ | 0.55 | 15 | $N$ | 0.36 |
| 6 | $P$ | 0.54 | 16 | $N$ | 0.35 |
| 7 | $N$ | 0.53 | 17 | $P$ | 0.34 |
| 8 | $N$ | 0.52 | 18 | $P$ | 0.33 |
| 9 | $P$ | 0.51 | 19 | $N$ | 0.30 |
| 10 | $N$ | 0.50 | 20 | $N$ | 0.10 |

## ROC curve

ROC cares both about TPR and FPR, so it values both positive examples and negative examples.
If only positive examples are important, one can plot precision and recall curve.

## Outline

## Multi-class classification

- Binary examples
  - Spam classification
  - Sentiment classification

## Multi-class classification

- Binary examples
  - Spam classification
  - Sentiment classification
- Multi-class examples
  - Star-ratings classification
  - Part-of-speech tagging
  - Image classification

## Binary vs. multi-class classification

Given: $S_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ training examples, $\boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$
Goal: Find hypothesis function $h : X \rightarrow Y$

## Binary vs. multi-class classification

Given: $S_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ training examples, $\boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$
Goal: Find hypothesis function $h : X \to Y$

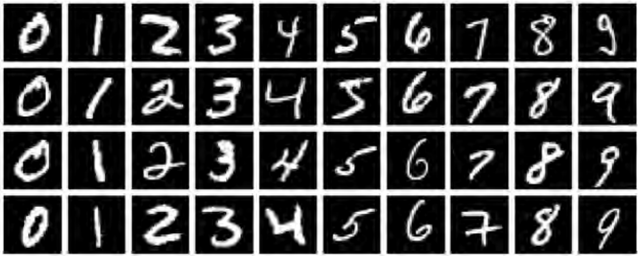
Given: $S_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ training examples, $\boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{0, 1, \ldots, C-1\}, C > 2$
Goal: Find hypothesis function $h : X \to Y$

## What we have learned so far

- Decision tree
- K-nearest neighbor
- Perceptron
- Logistic regression

## Inherent multi-class classifiers

K-nearest neighbor: Find the K-nearest neighbors of $x$ in training data and predict the majority label of those K points.



## Inherent multi-class classifiers

Logistic regression:

$$P(y = 1 \mid \boldsymbol{x}) = \sigma(\beta_0 + \sum_j \beta_j \boldsymbol{x}_j)$$

$$P(y = 0 \mid \boldsymbol{x}) = 1 - \sigma(\beta_0 + \sum_j \beta_j \boldsymbol{x}_j),$$

where $\sigma(z) = \frac{1}{1+exp[-z]}$

## Inherent multi-class classifiers

Logistic regression:

$$P(y=1 \mid \boldsymbol{x}) = \sigma(\beta_0 + \sum_j \beta_j \boldsymbol{x}_j)$$

$$P(y=0 \mid \boldsymbol{x}) = 1 - \sigma(\beta_0 + \sum_j \beta_j \boldsymbol{x}_j),$$

where $\sigma(z) = \frac{1}{1+exp[-z]}$
In the odds view,

$$\beta_0 + \sum_j \beta_j \boldsymbol{x}_j = \log \frac{P(y=1|\boldsymbol{x})}{P(y=0 \mid \boldsymbol{x})}$$

## Inherent multi-class classifiers

Logistic regression with more than two classes:

$$\beta_{10} + \sum_j \beta_{1j} \boldsymbol{x}_j = \log \frac{P(y = 1|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

$$\beta_{20} + \sum_j \beta_{2j} \boldsymbol{x}_j = \log \frac{P(y = 2|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

$$\vdots$$

$$\beta_{C-1,0} + \sum_j \beta_{C-1,j} \boldsymbol{x}_j = \log \frac{P(y = C-1|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

## Inherent multi-class classifiers

Logistic regression with more than two classes:

$$\beta_{10} + \sum_j \beta_{1j} \boldsymbol{x}_j = \log \frac{P(y = 1|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

$$\beta_{20} + \sum_j \beta_{2j} \boldsymbol{x}_j = \log \frac{P(y = 2|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

$$\vdots$$

$$\beta_{C-1,0} + \sum_j \beta_{C-1,j} \boldsymbol{x}_j = \log \frac{P(y = C-1|\boldsymbol{x})}{P(y = C \mid \boldsymbol{x})}$$

$$P(y = c \mid \boldsymbol{x}) = \frac{\exp(\beta_c^T \boldsymbol{x})}{1 + \sum_{c'=1}^{C-1} \exp(\beta_{c'}^T \boldsymbol{x})}, P(y = C \mid \boldsymbol{x}) = \frac{1}{1 + \sum_{c'=1}^{C-1} \exp(\beta_{c'}^T \boldsymbol{x})}$$

## Outline

ROC, AUC

Inherent multi-class classifiers

From binary classifiers to multi-class classifiers

Regularization (bonus)

## Classifiers

Now we are left with classifiers that are basically binary

- Perceptron
- SVM

Is there anything that we can do?

## Reduction

### Multiclass Data

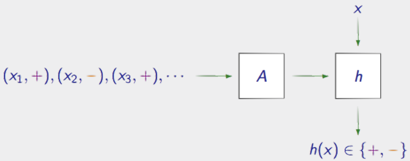
```
⟨name=Cindy , age=5 , sex=F⟩,
⟨name=Marcia, age=15, sex=F⟩,
⟨name=Bobby , age=6 , sex=M⟩,
⟨name=Jan   , age=12, sex=F⟩,
⟨name=Peter , age=13, sex=M⟩,
```

## Reduction

### Multiclass Data

$\langle$name=Cindy , age=5 , sex=F$\rangle$,
$\langle$name=Marcia, age=15, sex=F$\rangle$,
$\langle$name=Bobby , age=6 , sex=M$\rangle$,
$\langle$name=Jan , age=12, sex=F$\rangle$,
$\langle$name=Peter , age=13, sex=M$\rangle$,

### Binary Classifier

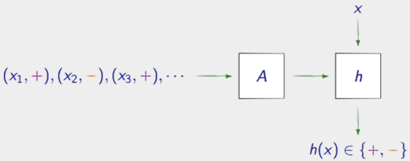


## Reduction

### Multiclass Data

⟨name=Cindy , age=5 , sex=F⟩,
⟨name=Marcia, age=15, sex=F⟩,
⟨name=Bobby , age=6 , sex=M⟩,
⟨name=Jan , age=12, sex=F⟩,
⟨name=Peter , age=13, sex=M⟩,

### Binary Classifier



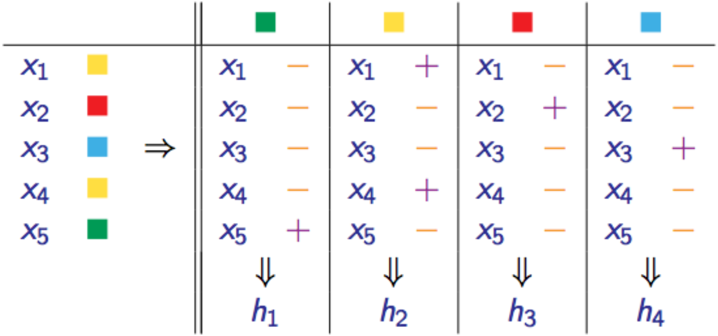

Goal: Multiclass Classifier

## Reduction

Two strategies

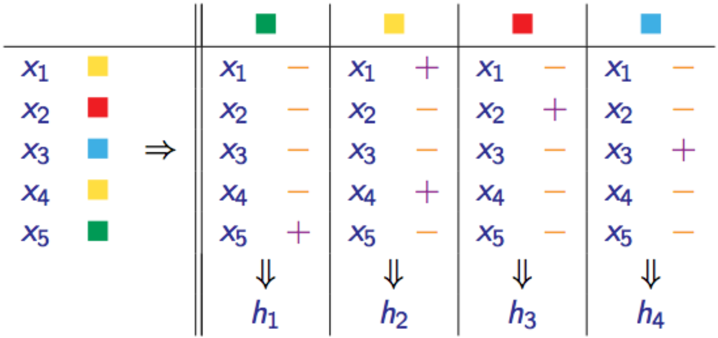
- One against all
- All pairs

## One against all




- Break $k$-class problem into $k$ binary problems and solve separately
- Combine predictions: evaluate all $h$'s, hope exactly one is $+$ (otherwise, take highest confidence)

## One against all




$$h(x) = \arg\max_{c \in C} h_c(\boldsymbol{x})$$

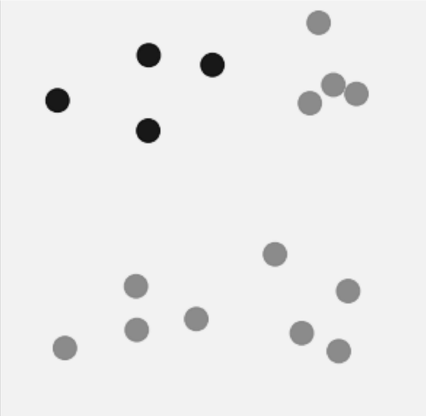
## One against all

Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

## One against all

Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

Black vs. not black



## One against all

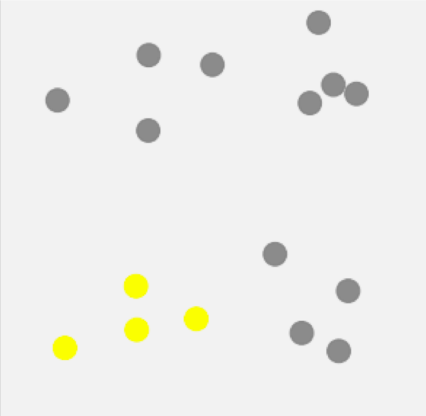
Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

Red vs. not red

## One against all

Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

Yellow vs. not yellow



## One against all

Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

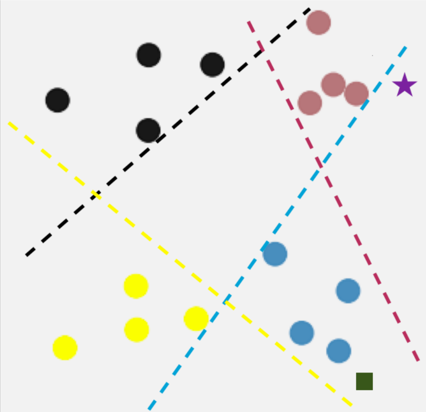
Blue vs. not blue

## One against all

Build $C$ binary classifiers of the form Class $c$ vs Class $\neg c$

Predict class with highest confidence

- Predict green square
- Predict purple star



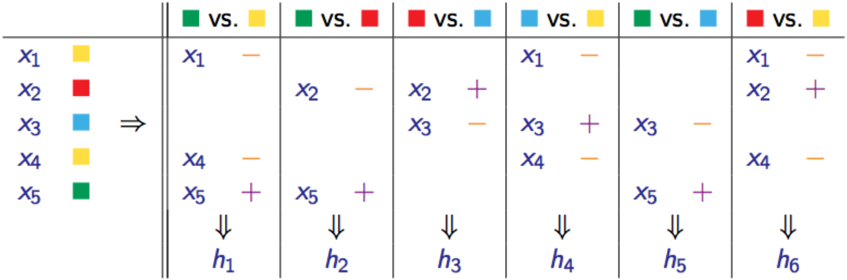
## One against all

Can you see any pitfalls of the one-against-all method?

## One against all

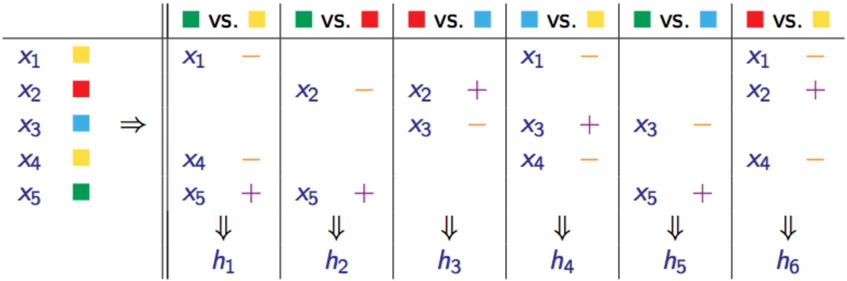
Can you see any pitfalls of the one-against-all method?
A big one is that if you start with a balanced training data, you immediately create imbalanced data.

## All pairs




- Break $k$-class problem into $k(k-1)/2$ binary problems and solve separately
- Combine predictions: evaluate all $h$'s, take the one with highest sum confidence

## All pairs




$$h(x) = \arg\max_{c \in C} \sum_{c' \neq c} h_{c'c}(\boldsymbol{x})$$

## Time Comparison

Assume training time is $\mathcal{O}(m^{\alpha})$ and test time is $\mathcal{O}(c_t)$

## Time Comparison

Assume training time is $\mathcal{O}(m^\alpha)$ and test time is $\mathcal{O}(c_t)$

| | Training | Testing |
|---|---|---|
| One-against-all | $\mathcal{O}(Cm^\alpha)$ | $\mathcal{O}(Cc_t)$ |
| All-pairs | $\mathcal{O}\left(C^2\left(\frac{m}{C}\right)^\alpha\right)$ | $\mathcal{O}\left(C^2 c_t\right)$ |

## Time Comparison

Assume training time is $\mathcal{O}(m^\alpha)$ and test time is $\mathcal{O}(c_t)$

| | Training | Testing |
|---|---|---|
| One-against-all | $\mathcal{O}(Cm^\alpha)$ | $\mathcal{O}(Cc_t)$ |
| All-pairs | $\mathcal{O}\left(C^2\left(\frac{m}{C}\right)^\alpha\right)$ | $\mathcal{O}(C^2 c_t)$ |

- One-against-all better for testing time
- All-pairs better for training
- All-pairs usually better for performance

## Outline

ROC, AUC

Inherent multi-class classifiers

From binary classifiers to multi-class classifiers

Regularization (bonus)

## Ridge vs. Lasso

Ridge Regression or $\ell_2$-Regularization:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \sum_{k=1}^{D} w_k^2$$

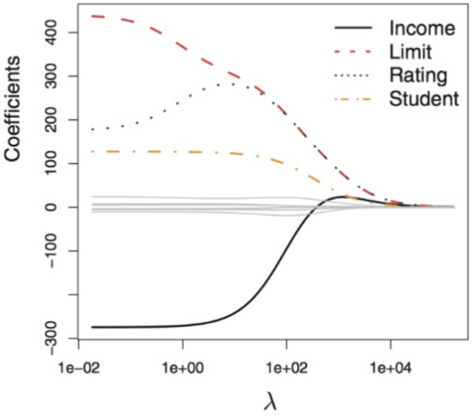
Lasso Regression or $\ell_1$-Regularization:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \sum_{k=1}^{D} |w_k|$$

Different penalty terms lead to different character of models

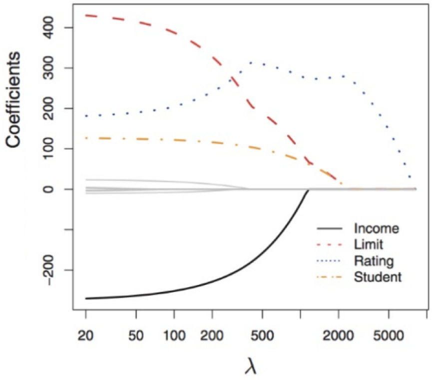
## Ridge

Coefficients shrink to zero uniformly smoothly



## Lasso

Some coefficients shrink to zero very fast



## The constrained optimization explanation

Consider the minimizer of

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \sum_{k=1}^{D} w_k^2 \quad \text{or} \quad \min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \sum_{k=1}^{D} |w_k|$$

For each objective function, can show that for a given $\lambda$ there is an equivalent $s$ such that the usual solution also solves

$$\text{Ridge:} \quad \min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \quad \text{s.t.} \quad \sum_{k=1}^{D} w_k^2 \leq s$$

$$\text{Lasso:} \quad \min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \quad \text{s.t.} \quad \sum_{k=1}^{D} |w_k| \leq s$$

## The constrained optimization explanation

Think of the constraint as a budget on the size of the parameters

For a given budget $s$ (corresponding to a given $\lambda$), find the $\mathbf{w}$ that minimizes the residual sum of squares (RSS) while staying inside the constrained region

Lasso Region for Two Features: Diamond

$$|w_1| + |w_2| \leq s$$

Ridge Region for Two Features: Circle

$$w_1^2 + w^2 \leq s$$

## The constrained optimization explanation

Minimum is more likely to be at point of diamond with Lasso, causing some feature weights to be set to zero.