

Machine learning - HT 2016

4. Basis Expansion, Regularization, Validation

Varun Kanade

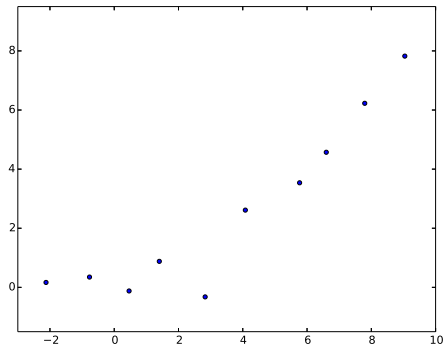
University of Oxford

February 03, 2016

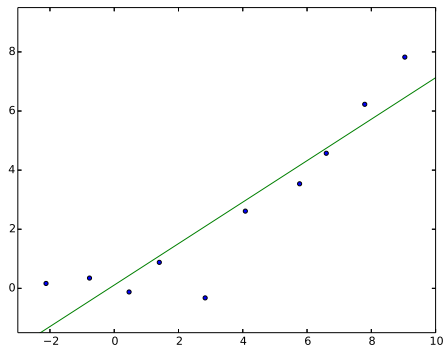
Outline

- ▶ Introduce **basis function** to go beyond linear regression
- ▶ Understanding the tradeoff between **bias** and **variance**
- ▶ Overfitting: What happens when we make models too complex
- ▶ Regularization as a means to control model complexity
- ▶ Cross-validation to perform model selection

Basis Expansion



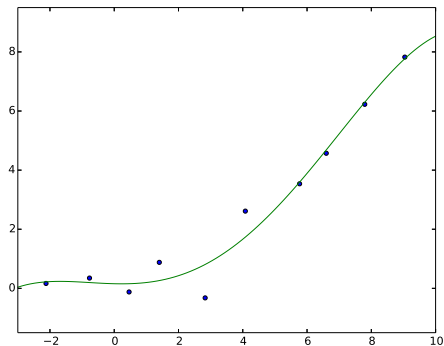
Basis Expansion



Basis Expansion

$$\phi(x) = [1, x, x^2, x^3, x^4]$$

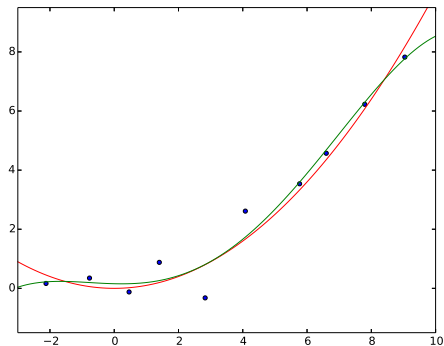
$$w_1 + w_2x + w_3x^2 + w_4x^3 + w_5x^4 = \phi(x) \cdot [w_1, w_2, w_3, w_4, w_5]$$



Basis Expansion

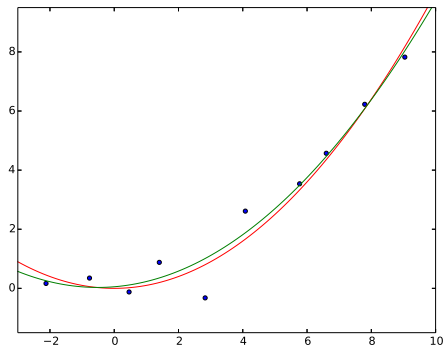
$$\phi(x) = [1, x, x^2, x^3, x^4]$$

$$w_1 + w_2x + w_3x^2 + w_4x^3 + w_5x^4 = \phi(x) \cdot [w_1, w_2, w_3, w_4, w_5]$$



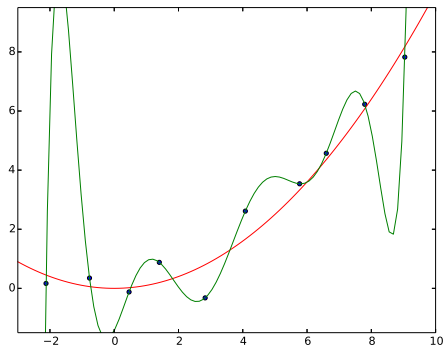
Basis Expansion

$$\phi(x) = [1, x, x^2]$$



Basis Expansion

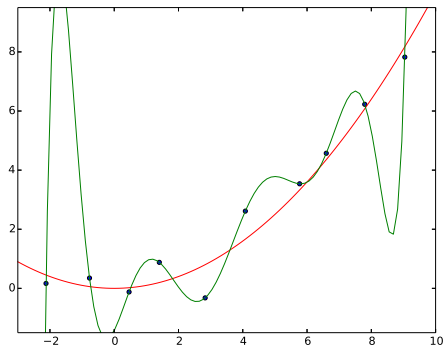
$$\phi(x) = [1, x, x^2, x^3, \dots, x^9]$$



Basis Expansion

$$\phi(x) = [1, x, x^2, x^3, \dots, x^d]$$

How can we avoid overfitting?

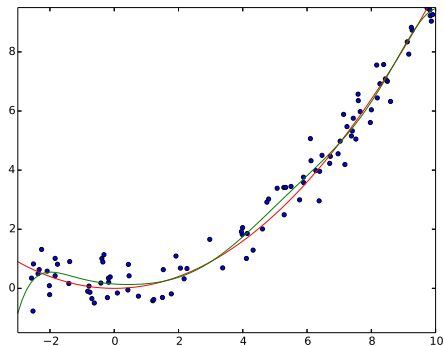


Basis Expansion

$$\phi(x) = [1, x, x^2, x^3, \dots, x^d]$$

How can we avoid overfitting?

Does more data help?

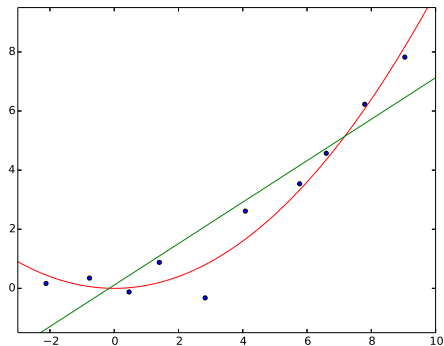


Basis Expansion

$$\phi(x) = [1, x, x^2, x^3, \dots, x^d]$$

How can we avoid overfitting?

Does more data help?

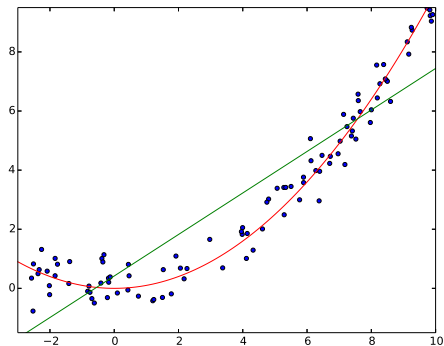


Basis Expansion

$$\phi(x) = [1, x, x^2, x^3, \dots, x^d]$$

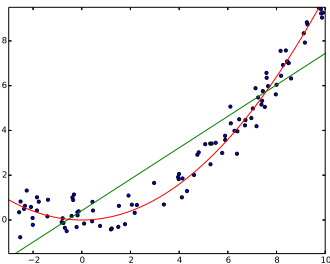
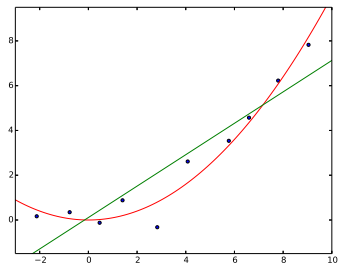
How can we avoid overfitting?

Does more data help?



Bias Variance Tradeoff

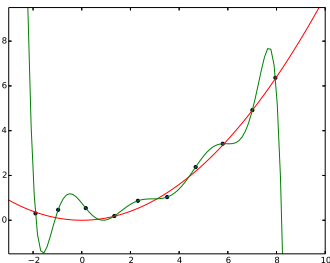
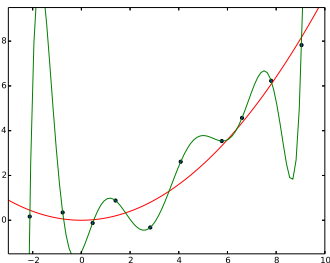
- ▶ For linear model, more data would make little difference



- ▶ **Bias** results from model being simpler than the “truth”
- ▶ High bias results in underfitting

Bias Variance Tradeoff

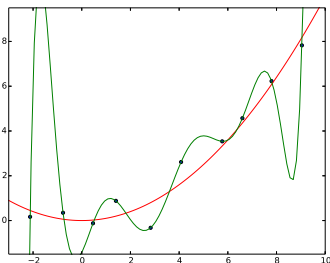
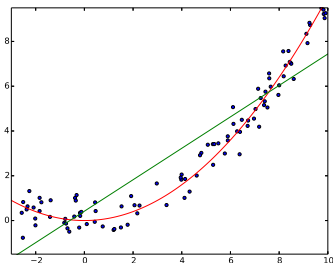
- ▶ What happens when we fit model on different (randomly drawn) training datasets?



- ▶ **Variance** arises when the (complex) model is sensitive to fluctuations in training dataset
- ▶ Variance results in overfitting

Bias Variance Tradeoff

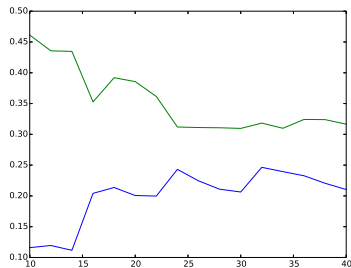
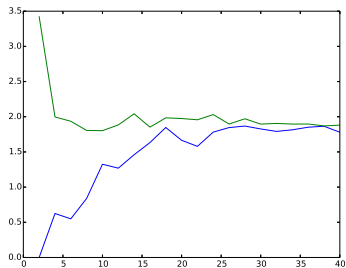
- ▶ When does more data help?
- ▶ $\text{Error} = \text{Bias}^2 + \text{Variance} + \text{Noise}$ (Exercise for linear regression)



- ▶ For more complex models, difficult to visually overfitting and underfitting
- ▶ Keep aside some points as “test set”

Learning Curves

- ▶ Suppose we have a training set and test set
- ▶ Train on increasing sizes of the training set, and plot the errors



- ▶ Once training and test error approach each other, then more data won't help!

Basis Expansion Using Kernels

We can use **kernels** as features, *e.g.*, radial basis functions (RBFs)

Feature expansion:

$$\phi(\mathbf{x}) = [1, \kappa(\mathbf{x}, \boldsymbol{\mu}_1, \sigma), \dots, \kappa(\mathbf{x}, \boldsymbol{\mu}_d, \sigma)], \quad \text{e.g., } \kappa(\mathbf{x}, \boldsymbol{\mu}_i, \sigma) = e^{-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|^2}{2\sigma^2}}$$

Model: $y = \phi(\mathbf{x})^T \mathbf{w} + \text{noise}$

Basis Expansion Using Kernels

As in the case of polynomials, the **width** σ can cause overfitting or underfitting

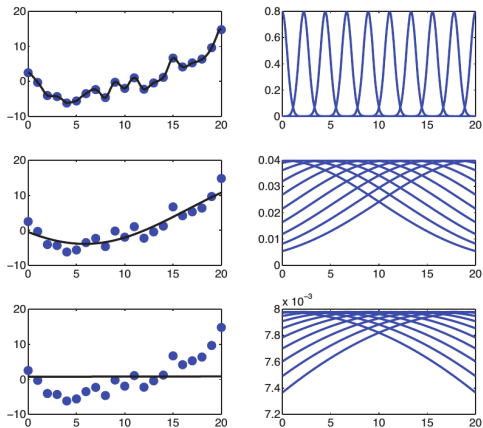


Image Source: K. Murphy (2012)

Polynomial Basis Expansion in Higher Dimensions

We are basically fitting linear models (at the cost of increasing dimensions)

$$y = \phi(\mathbf{x}) + \text{noise}$$

Linear Model: $\phi(\mathbf{x}) = [1, x_1, x_2]$

Quadratic Model: $\phi(\mathbf{x}) = [1, x_1, x_2, x_1^2, x_2^2, x_1x_2]$

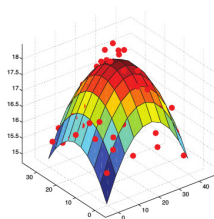
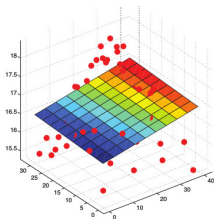


Image Source: K. Murphy (2012)

How many dimensions do you get for degree d polynomials over n variables?

Overfitting!

In high dimensions, we can have many many parameters!

With 100 variables and degree 10 we have $\sim 10^{20}$ parameters!

Enrico Fermi to Freeman Dyson

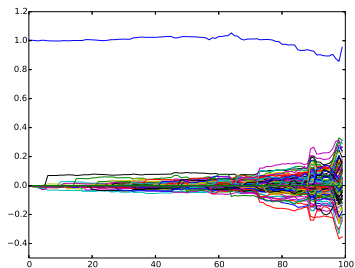
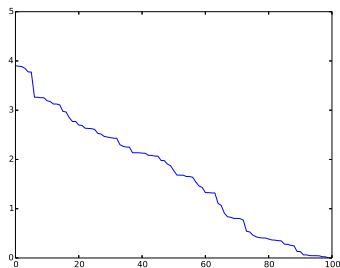
"I remember my friend Johnny von Neumann used to say, with four parameters I can fit an elephant, and with five I can make him wiggle his trunk." [\[video\]](#)

How do we prevent overfitting?

How does overfitting occur?

Suppose $\mathbf{X}_{100 \times 100}$ with every entry $\mathcal{N}(0, 1)$

And let $y_i = x_{i,1} + \mathcal{N}(0, \sigma^2)$, for $\sigma = 0.2$



Ridge Regression

Say our data is $\langle (\mathbf{x}_i, y_i) \rangle_{i=1}^m$, where $\mathbf{x} \in \mathbb{R}^N$ where N is really really large!

We used the squared loss

$$L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y})$$

and obtained the estimate

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Suppose we want \mathbf{w} to be “small” (**weight decay**)

$$L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w} - \lambda w_1^2$$

We will not regularize the “bias” (or constant) term

Exercise: If all x_i (except x_1) have mean 0 and y has mean 0, $w_1 = 0$

Deriving Estimate for Ridge Regression

$$L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w}$$

Ridge Regression

Objective/Loss: $L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w}^T\mathbf{w}$

Estimate: $\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$

Estimate depends on the scaling of the inputs

Common practice: Normalise all input dimensions

Ridge Regression and MAP Estimation

Objective/Loss: $L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w}^T\mathbf{w}$

Ridge Regression and MAP Estimation

Objective/Loss: $L(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w}$

$$\exp\left(\frac{-1}{2\sigma^2} L(\mathbf{w})\right) = \exp\left(-\frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T \Sigma^{-1}(\mathbf{y} - \mathbf{X}\mathbf{w})\right) \cdot \exp\left(-\frac{1}{2}\mathbf{w}^T \Lambda^{-1}\mathbf{w}\right)$$

Bayesian View of Machine Learning

General Formulation

Prior $p(\mathbf{w})$ on \mathbf{w}

Model $p(y | \mathbf{x}, \mathbf{w})$

Compute posterior given data $\mathcal{D} = \langle (\mathbf{x}_i, y_i) \rangle_{i=1}^m$

$$p(\mathbf{w} | \mathcal{D}) \propto$$

Linear Regression

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \mathbf{0}, \tau^2 \mathbf{I}_n)$$

$$p(y | \mathbf{x}, \mathbf{w}) = \mathcal{N}(y | \mathbf{x}^T \mathbf{w}, \sigma^2)$$

Bayesian View of Regression

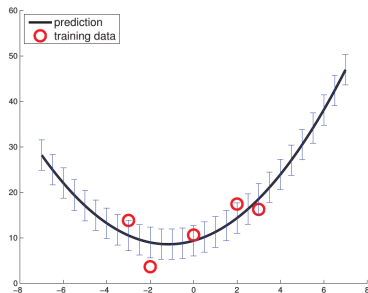
Making a prediction on a new point \mathbf{x}_{new}

$$p(y | \mathcal{D}, \mathbf{x}_{\text{new}}) = \int_{\mathbf{w}} p(y | \mathbf{w}, \mathbf{x}_{\text{new}}) p(\mathbf{w} | \mathcal{D}) d\mathbf{w}$$

Prediction MAP vs Fully Bayesian

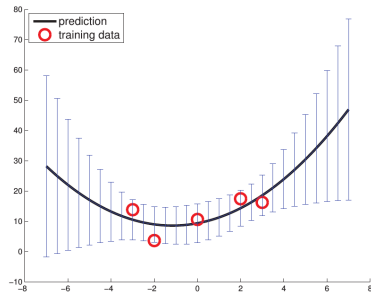
MAP Approach

$$y_{\text{new}} \sim \mathcal{N}(\mathbf{x}_{\text{new}}^T \mathbf{w}_{\text{map}}, \sigma^2)$$



Bayesian Approach

$$y_{\text{new}} \sim \mathcal{N}(\mathbf{x}_{\text{new}}^T \mathbf{w}_{\text{map}}, \sigma_X^2)$$
$$\sigma_X^2 = \sigma^2 \left(1 + \mathbf{x}_{\text{new}}^T (\mathbf{X}^T \mathbf{X} + \frac{\sigma^2}{\tau^2} \mathbf{I})^{-1} \mathbf{x}_{\text{new}} \right)$$



Ridge Regression

Minimize:

$$(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w}^T\mathbf{w}$$

Minimize $(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y})$

such that $\mathbf{w}^T\mathbf{w} \leq R$

Ridge Regression

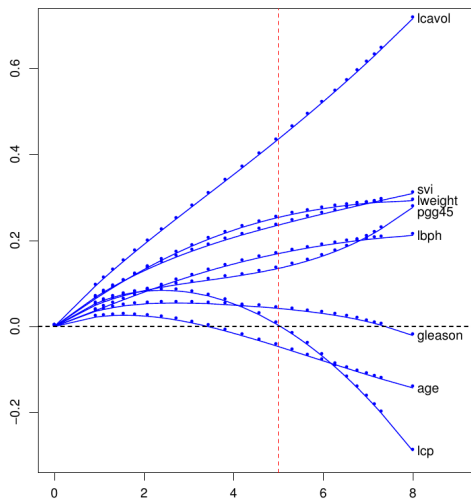


Image Source: Hastie, Tibshirani, Friedman (2013)

Lasso Regression

Minimize:

$$(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \sum_{i=1}^N |w_i|$$

Minimize $(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y})$

such that $\sum_{i=1}^N |w_i| \leq R$

Lasso Regression

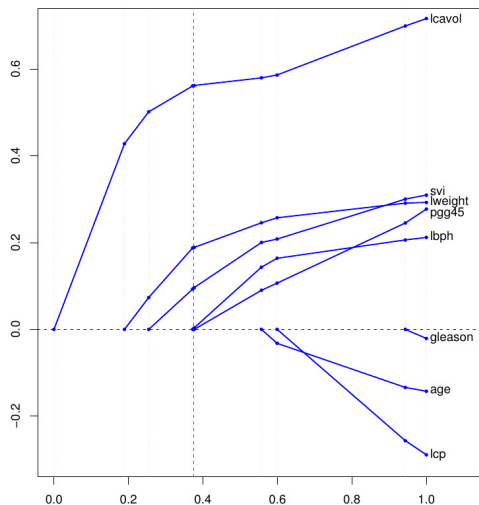
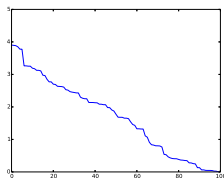


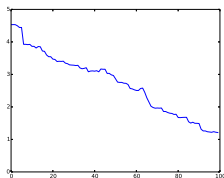
Image Source: Hastie, Tibshirani, Friedman (2013)

Regularization: Ridge or Lasso

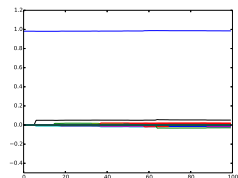
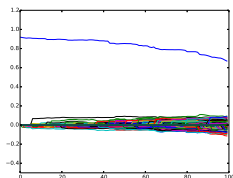
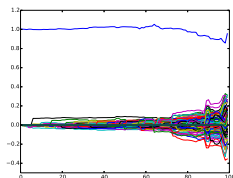
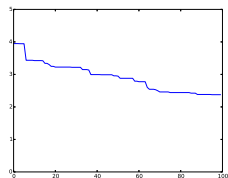
No regularization



Ridge



Lasso



Choosing Parameters

Before we were just trying to find w

Now, we have to worry about how to choose λ for ridge or Lasso

If we use kernels, we also have to pick the width σ

If we use higher degree polynomials, we have to pick d

Cross Validation

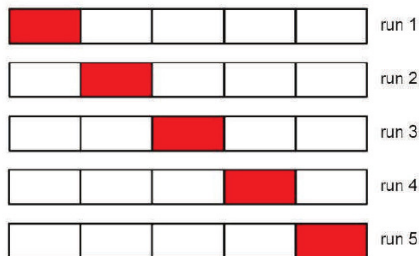
- ▶ Keep a part of data as “validation” or “test” set
- ▶ Look for error on “training” and “validation” sets

λ	Train Error(%)	Test Error(%)
0.01	25	27
0.1	10	8
1	2	12
10	0	43
100	0	89

k -Fold Cross Validation

What do we do when data is scarce?

- ▶ Divide data into k parts
- ▶ Use $k - 1$ parts for training and 1 part as validation
- ▶ When $k = m$ (the number of datapoints), we get LOOCV (Leave one out cross validation)



Overfitting on the Validation Set

Suppose you do all the right things

- ▶ Train on the training set
- ▶ Choose hyperparameters using proper validation
- ▶ Test on the test set, and your error is high!

What would you do?

Winning Kaggle without reading the data!

Suppose the task is to predict m
binary labels

Winning Kaggle without reading the data!

Suppose the task is to predict m binary labels

Algorithm (Wacky Boosting):

1. Choose $\mathbf{y}^1, \dots, \mathbf{y}^k \in \{0, 1\}^m$ uniformly
2. Set $I = \{i \mid \text{accuracy}(\mathbf{y}^i) > 51\%\}$
3. Output $\hat{y}_j = \text{majority}\{y_j^i \mid i \in I\}$

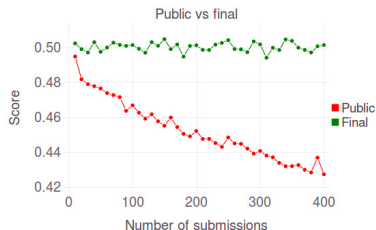
Source: blog.mrtz.org

Winning Kaggle without reading the data!

Suppose the task is to predict m binary labels

Algorithm (Wacky Boosting):

1. Choose $\mathbf{y}^1, \dots, \mathbf{y}^k \in \{0, 1\}^m$ uniformly
2. Set $I = \{i \mid \text{accuracy}(\mathbf{y}^i) > 51\%\}$
3. Output $\hat{y}_j = \text{majority}\{y_j^i \mid i \in I\}$



Source: blog.mrtz.org

Next Time

- ▶ Optimization Methods
- ▶ Read up on gradients, multivariate calculus