李宏毅–机器学习 (2017FALL)–个人笔记

Hanjun Liu
liuhanjun369@gmail.com

College of Computer Science and Engineering, Shandong
University of Science and Technology

October 26, 2018

前言

突然很想把自己的笔记整理成 PDF 然后打印出来，顺便学学 Latex, 个人水平较差如果发现笔记中有错误, 可以通过下面的方式与我联系。

liuhanjun369@gmail.com
Github
Telegram

Contents

Part I

Supervised Learning

Chapter 1

Regression

Chapter 2

Classification

Chapter 3

Ensemble

3.1 Introduction

就是好几个模型一起上，在分类或是回归问题上能够得到一个更好的结果。

3.2 Framework of Ensemble

假设你在做一个分类任务，首先你要有一大堆不同的的 classifiers，它们要一起完成一个分类任务，就好像是玩游戏时各自的分工是不一样的，我们需要找到一个方法替这些 classifiers 找到一个好的分工从而提高分类的准确率。

3.3 Ensemble: Bagging

注意: Bagging 适用于很复杂的模型

3.3.1 Review: Bias v.s. Variance

如果是很简单的模型就会有很小的 Variance 和很大的 Bias，如果模型很复杂就会有很大的 Varience 和很小的 Bias。我们将这些很复杂的模型平均起来就可以的减小 Variance 从而提高系统的性能。

3.3.2 Bagging

怎么做 Bagging 呢，假设我们有 N 笔 Training example 每次从中随机抽取 $N^{'}$ 个 examples 来训练一个模型。如下图一共抽取了四次训练了四个不同模型。

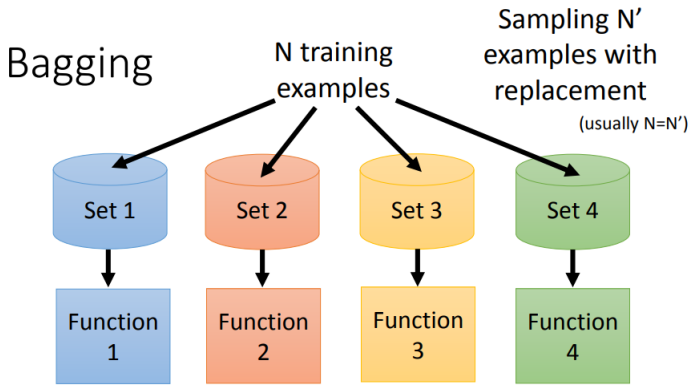

Figure 3.1: Training Bagging

Bagging 适用于很复杂很容易过拟合的模型。在测试阶段如果是回归问题那么就求这四个输出的平均值作为结果，如果是分类问题那么我们使用 voting（选择出现次数最多的那个类最为最终输出）。

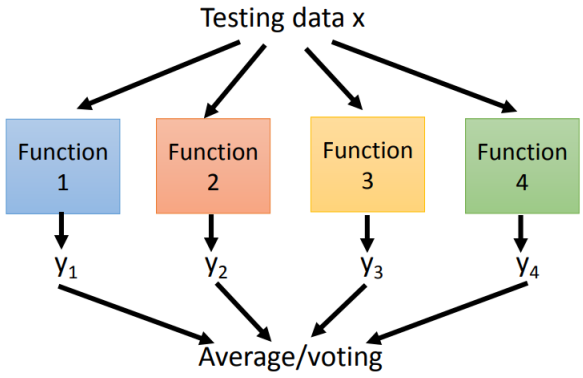

Figure 3.2: Training Bagging

3.3.3 Decision Tree

决策树非常容易过拟合，只要树够深就能 100% 拟合训练集，这里是做了一个实验让决策树判断像素点是不是属于初音。

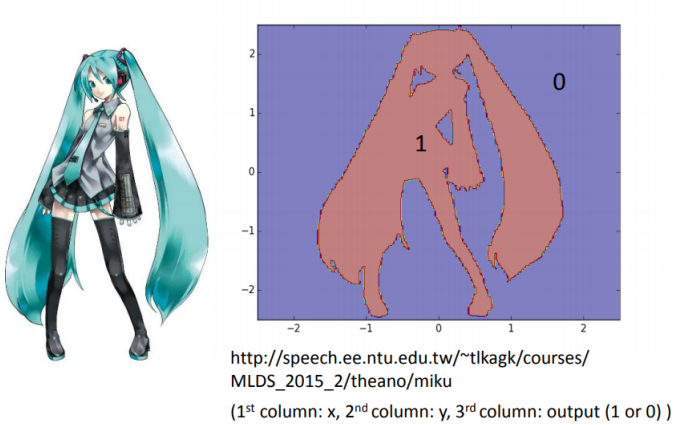

Figure 3.3: Experiment: Function of Miku

下图是我们使用单个决策树的情况，可以看出来当输的深度达到 20 时决策树就可以完美的拟合啦

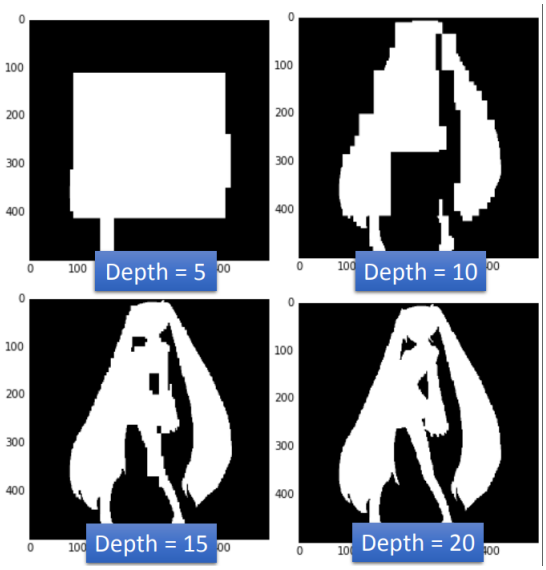

Figure 3.4: Experiment: Function of Miku(Decision Tree)

3.3.4 Random Forest

Random Forest 就是 Bagging of decision tree, 随机森林在对每棵树随机抽取数据训练外还限制了一些 features 这个就需要具体看一下随机森林的算法了。用 Out-of-bag（OOB）对 Bagging 算法进行验证。

给出数据分配图一共 4 笔数据，有四个决策树，每个决策树只能看到两笔数据那么就可以拿另外两笔数据来验证模型的效果，如图 3.6。

train	f_1	f_2	f_3	f_4
x^1	O	X	O	X
x^2	O	X	X	O
x^3	X	O	O	X
x^4	X	O	X	O

Figure 3.5: 数据分配

如 f_2 没看过 x^1, f_4 也没看 x^1, 那么就可以用 x^1 验证 f_2 和 f_4 Bagging 后的效果。

- Using RF = f_2+f_4 to test x^1
- Using RF = f_2+f_3 to test x^2
- Using RF = f_1+f_4 to test x^3
- Using RF = f_1+f_3 to test x^4

Figure 3.6: Out-of-bag

决策树得到的结果比单个决策树要平滑一些。

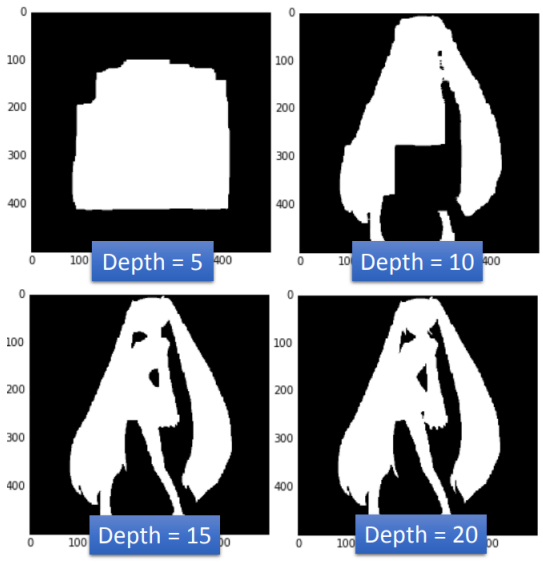

Figure 3.7: Experiment: Function of Miku（Random Forest）

3.4 Ensemble: Boosting

注意：Boosting 是用于弱分类器

3.4.1 Boosting

使用 Boosting 有一个很强的保证只要你有一个正确率大于 50% 的分类器，经过 Boosting 之后你会得到一个错误率是 0% 的分类器。

3.4.2 Framework of boosting

首先你要有一个分类器 $f_1(x)$, 然后找到另一个分类器 $f_2(x)$ 来帮助 $f_1(x)$ ，但是 $f_2(x)$ 和 $f_1(x)$ 是互补的，$f_2(x)$ 要做 $f_1(x)$ 做不到的事情，如果二者特别相似帮助也不会很大，得到 $f_2(x)$ 之后再用 $f_3(x)$ 来帮助 $f_2(x)$..... 重复下去。(Boosting 训练是有顺序的)
假设我们有一个数据集$(x^1, \hat{y}^1), (x^2, \hat{y}^2), ..., (x^n, \hat{y}^n)$ 其中 $\hat{y}^i = \pm 1$ 是一个二分类问题，那么我们该怎么得到不同的分类器呢?
我们可以用不同的数据来训练分类器，有两种方法来生成不同的数据。

- 1. 随机从训练集中抽取数据进行训练
- 2. 给训练集一个权重

随机抽取数据不必多说，给数据权重就是每一笔数据给一个 u^i 作为权重然后我们的数据集就成了$(x^1, \hat{y}^1\ u^1), (x^2, \hat{y}^2\ u^2), ..., (x^n, \hat{y}^n, u^n)$ 然后我们要改变我们的 Loss Function:$L(f) = \sum_{i=0}^{n} u^i l(f(x^i), \hat{y}^i)$，这样如果某笔 data 的权重很大造成 loss 很大，就会受到更多的关注。(初始权重为 1)

3.4.3 Idea of Adaboost

Adaboost 的思想就是先训练 $f_1(x)$(准确度必然大于 50%), 然后更改权重 u，使 $f_1(x)$ 在新的数据上得到准确度等于 50%, 再用这个数据训练 $f_2(x)$ 这样就让 $f_2(x)$ 和 $f_1(x)$ 互补啦，因为二者学习的是不同的数据。给出更改权重的公式:

ϵ_1 是 $f_1(x)$ 的错误率

$$Z_1 = \sum_{i=1}^{n} u_1^i$$

$$\epsilon_1 = \frac{\sum_{i=0}^{n} u_1^i \delta(f(x^i) \neq \hat{y}^i)}{Z_1}, \epsilon_1 < 0.5$$

其中 $\delta(f(x^i) \neq \hat{y}^i) = 0$, 如果 $f(x^i) \neq \hat{y}^i$,$\delta(f(x^i) \neq \hat{y}^i) = 1$, 如果 $f(x^i) = \hat{y}^i$

要找到一个 u_2 替换掉上面的 u_1 使

$$\frac{\sum_{i=0}^{n} u_2^i \delta(f(x^i) \neq \hat{y}^i)}{Z_2} = 0.5$$

下面给一个 re-weighting 的例子, $\epsilon_1 = 0.25$, 然后我们将分类正确的权重除以 $\sqrt{3}$ 降低影响, 将分类错误的权重乘以 $\sqrt{3}$ 增大影响（至于为什么是 $\sqrt{3}$ 下面会给出详细的解答）。然后 $\frac{\sum_{i=0}^{n} u_2^i \delta(f(x^i) \neq \hat{y}^i)}{Z_2} = 0.5$ 再重新训练一个 $f_2(x)$ 使 $\epsilon_2 < 0.5$。

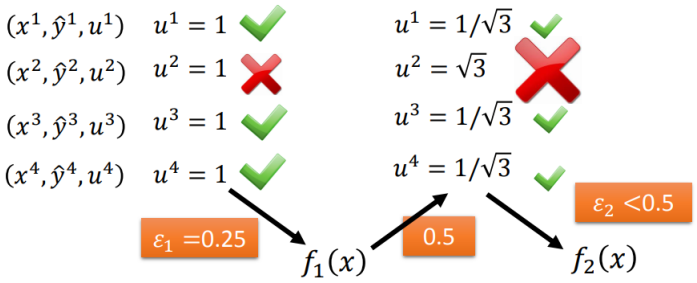

Figure 3.8: Re-weighting example

如果一个数据被分类正确那么就把它的权重除以一个数 $d_1(d_1 > 1)$，如果分类错误就把对应权重乘以 $d_1(d_1 > 1)$, 那么怎么求 d_1，下面给出推倒。

$$Z_1 = \sum_{i=1}^{m} u_1^i$$

$$\epsilon_1 = \frac{\sum_{i=0}^{n} u_1^i \delta(f(x^i) \neq \hat{y}^i)}{Z_1}, \epsilon_1 < 0.5$$

$$\frac{\sum_{i=0}^{n} u_2^i \delta(f(x^i) \neq \hat{y}^i)}{Z_2} = 0.5$$

$$\sum_{i=0}^{n} u_2^i \delta(f(x^i) \neq \hat{y}^i) = \sum_{f(x_i) \neq \hat{y}_i i=1}^{n} u_1^i d_1$$

$$Z_2 = \sum_{f(x_i) \neq \hat{y}_i i=1}^{n} u_2^i + \sum_{f(x_i) = \hat{y}_i i=1}^{n} u_2^i$$

$$Z_2 = \sum_{f(x_i) \neq \hat{y}_i i=1}^{n} u_1^i d_1 + \sum_{f(x_i) = \hat{y}_i i=1}^{n} \frac{u_1^i}{d_1}$$

将方程代入原式

$$\frac{\sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i d_1}{\sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i d_1 + \sum_{f(x_i)=\hat{y}_i i=1}^{n} \frac{u_1^i}{d_1}} = 0.5$$

将上面的式子化简得到

$$\sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i d_1 = \sum_{f(x_i)=\hat{y}_i i=1}^{n} \frac{u_1^i}{d_1}$$

将 d_1 拿出来得到

$$d_1 \sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i = \frac{1}{d_1} \sum_{f(x_i)=\hat{y}_i i=1}^{n} u_1^i$$

$$\frac{\sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i}{Z_1} = \epsilon_1 >>> \sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i = Z_1\epsilon_1$$

$$\frac{\sum_{f(x_i)=\hat{y}_i i=1}^{n} u_1^i}{Z_1} = 1-\epsilon_1 >>> \sum_{f(x_i)=\hat{y}_i i=1}^{n} u_1^i = Z_1(1-\epsilon_1)$$

将上述结果带入到 $d_1 \sum_{f(x_i)\neq\hat{y}_i i=1}^{n} u_1^i = \frac{1}{d_1} \sum_{f(x_i)=\hat{y}_i i=1}^{n} u_1^i$ 得到 $d_1 = \sqrt{\frac{(1-\epsilon_1)}{\epsilon_1}} > 1$

3.4.4 Algorithm for AdaBoost

由 d_t 换成 α_t 简化了式子，最终括号一应该填 $-y_t f_t(x)$

- Giving training data
  $\{(x^1, \hat{y}^1, u_1^1), \cdots, (x^n, \hat{y}^n, u_1^n), \cdots, (x^N, \hat{y}^N, u_1^N)\}$
  - $\hat{y} = \pm 1$ (Binary classification), $u_1^n = 1$ (equal weights)
- For t = 1, …, T:
  - Training weak classifier $f_t(x)$ with weights $\{u_t^1, \cdots, u_t^N\}$
  - ε_t is the error rate of $f_t(x)$ with weights $\{u_t^1, \cdots, u_t^N\}$
  - For n = 1, …, N:
    - If x^n is misclassified classified by $f_t(x)$: $\hat{y}^n \neq f_t(x^n)$
    - $u_{t+1}^n = u_t^n \times d_t = u_t^n \times \exp(\alpha_t)$ $\quad d_t = \sqrt{(1-\varepsilon_t)/\varepsilon_t}$
    - Else:
    - $u_{t+1}^n = u_t^n / d_t = u_t^n \times \exp(-\alpha_t)$ $\quad \alpha_t = ln\sqrt{(1-\varepsilon_t)/\varepsilon_t}$

$$u_{t+1}^n \leftarrow u_t^n \times exp(\qquad \alpha_t)$$

Figure 3.9: Algorithm for AdaBoost 1

在算最终结果时也要乘上 α_t, 这样做能让 ϵ 高的分类器得到更高的分数

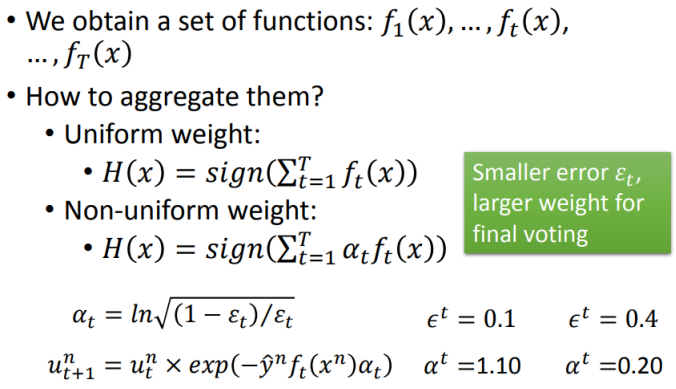

Figure 3.10: Algorithm for AdaBoost 2

3.4.5 Toy Example

用 3 个 weak classifier = decision stump 随便切一刀

- Step 1:

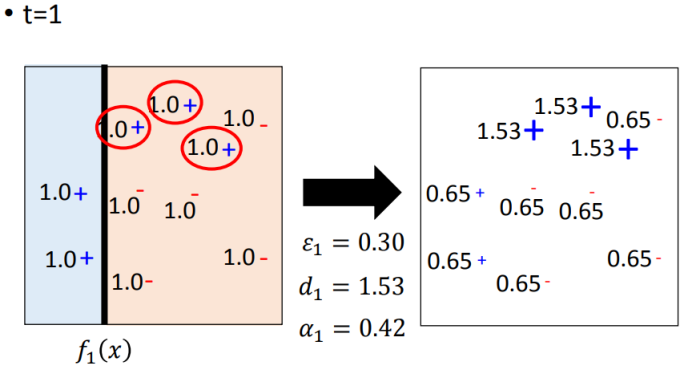

Figure 3.11: Toy Example Step 1

- Step 2:

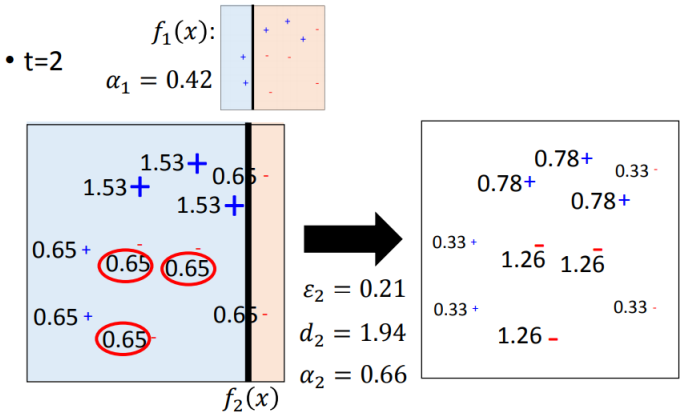

Figure 3.12: Toy Example Step 2

- Step 3:

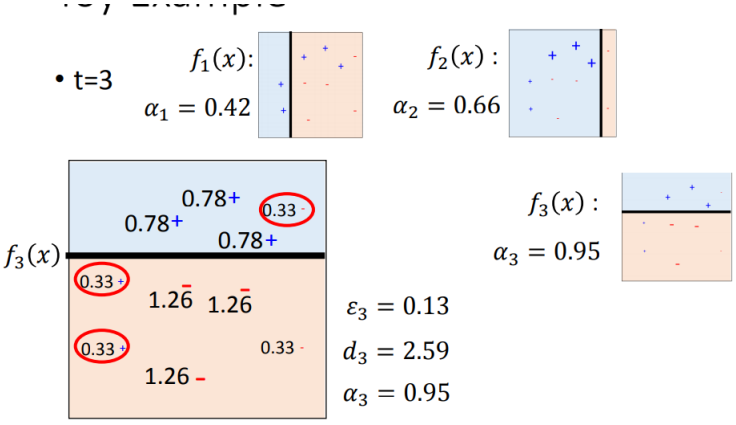

Figure 3.13: Toy Example Step 3

- Step 4:

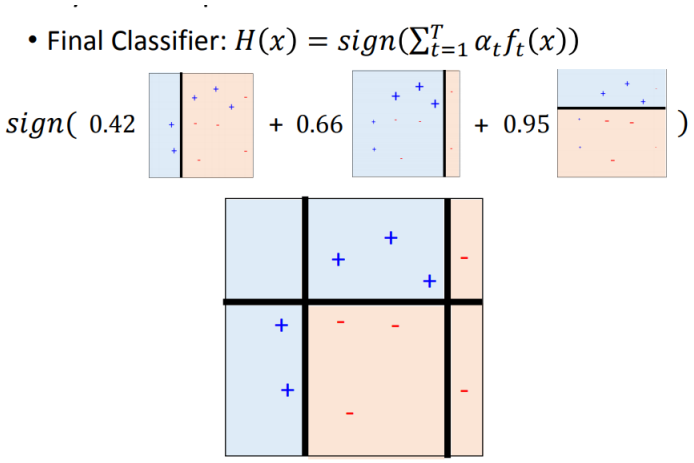

Figure 3.14: Toy Example Step 4

3.4.6 More Detial in Adaboost

最终的分类 $H(x)$ 的方程是:

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t f_t(x))$$

$$g(x) = \sum_{t=1}^{T} \alpha_t f_t(x)$$

训练数据上的错误率为（有 N 笔数据）:

$$ErrorRate = \frac{1}{N} \sum_{n=1}^{N} \delta(H(x^n) \neq \hat{y}^n) = \frac{1}{N} \sum_{n=1}^{N} \delta(\hat{y}^n g(x^n) < 0)$$

ErrorRate 存在一个上限:

$$ErrorRate \leq \frac{1}{N} \sum_{n=1}^{N} \exp(-\hat{y}^n g(x^n))$$

图中绿线代表 $\delta(\hat{y}^n g(x^n) < 0)$ 蓝线代表 $\exp(-\hat{y}^n g(x^n))$，横坐标是 $\hat{y}^n g(x^n)$

Figure 3.15: 上限

我们要证明上限越来越小

$$Z_t = \sum_{i=1}^{n} u_t^i$$

$$Z_{t+1} = \sum_{i=1}^{n} u_{t+1}^i$$

$$\left.\begin{aligned} u_1^n &= 1 \\ u_{t+1}^n &= u_t^n \times exp(-\hat{y}^n f_t(x^n)\alpha_t) \end{aligned}\right\} \quad u_{T+1}^n = \prod_{t=1}^{T} exp(-\hat{y}^n f_t(x^n)\alpha_t)$$

$$\begin{aligned} Z_{T+1} &= \sum_n \prod_{t=1}^{T} exp(-\hat{y}^n f_t(x^n)\alpha_t) \\ &= \sum_n exp\left(-\hat{y}^n \sum_{t=1}^{T} g(x) f_t(x^n)\alpha_t\right) \end{aligned}$$

看懂上面的推导之后我们发现 $\frac{1}{N} Z_{t+1} = \frac{1}{N} \sum_{n=1}^{N} \exp(-\hat{y}^n g(x^n))$ 所以能证明 Z 越来越小就 ok 了。

Training Data Error Rate

$$\leq \frac{1}{N}\sum_{n} exp(-\hat{y}^n g(x^n)) = \frac{1}{N} Z_{T+1}$$

$$g(x) = \sum_{t=1}^{T} \alpha_t f_t(x)$$

$$\alpha_t = ln\sqrt{(1-\varepsilon_t)/\varepsilon_t}$$

$Z_1 = N$ (equal weights)

$$Z_t = Z_{t-1}\varepsilon_t exp(\alpha_t) + Z_{t-1}(1-\varepsilon_t)exp(-\alpha_t)$$

Misclassified portion in Z_{t-1} Correctly classified portion in Z_{t-1}

$$= Z_{t-1}\varepsilon_t\sqrt{(1-\varepsilon_t)/\varepsilon_t} + Z_{t-1}(1-\varepsilon_t)\sqrt{\varepsilon_t/(1-\varepsilon_t)}$$

$$= Z_{t-1} \times 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$$

$$Z_{T+1} = N\prod_{t=1}^{T} 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$$

Training Data Error Rate $\leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)}$ (<1)

Smaller and smaller

Figure 3.16: 证明 Z_t 越来越小

Adaboost 有很神奇的现象，当在 training 上的 error 达到零时，在 test 上的 error 还可以下降原因是权重可以增加 *Margin* 其实原理很简单, AdaBoost 的 error 会无限接近于零但不是零。

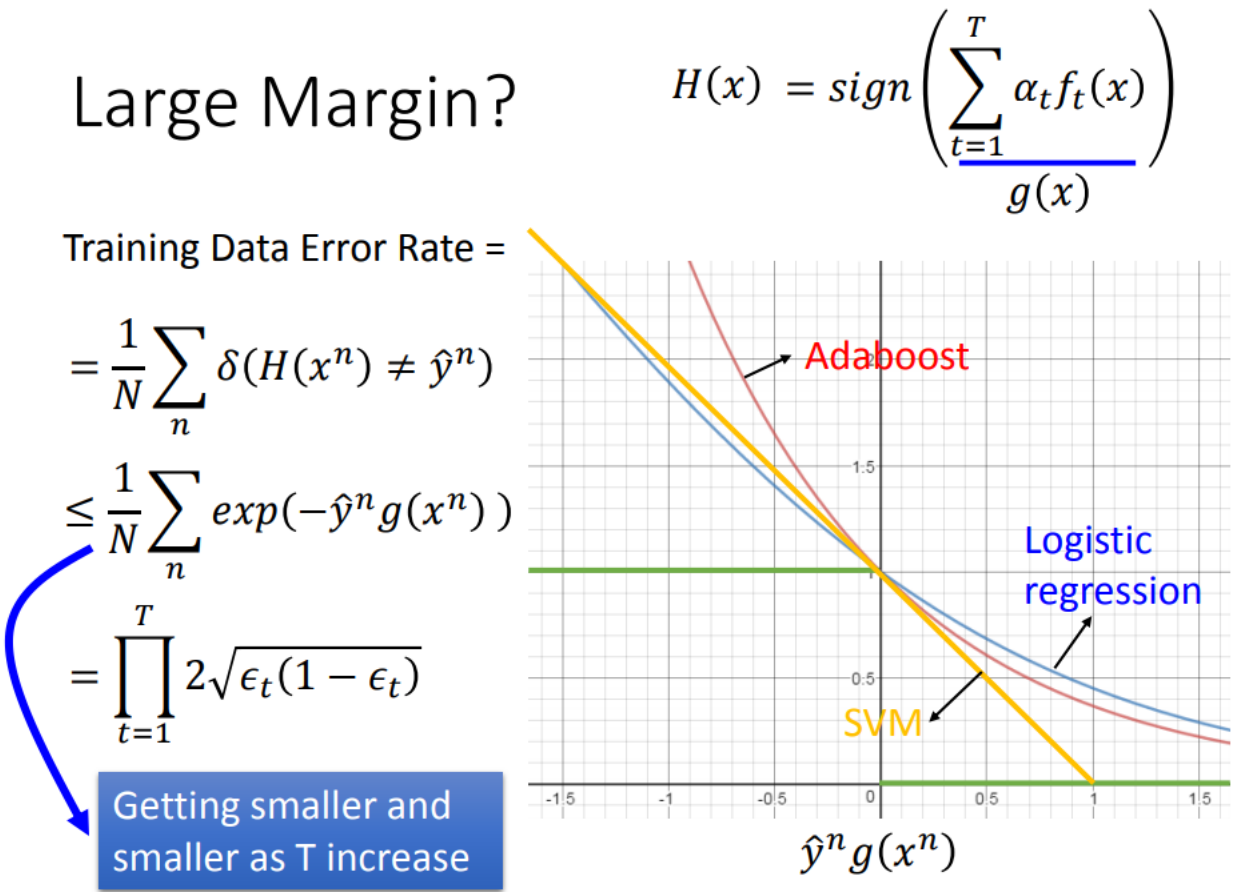

Figure 3.17: 证明 Test 上的 error rate 会变小

我们用深度是 5 的决策树进行 Adaboost 效果要比在单棵树上好太多。

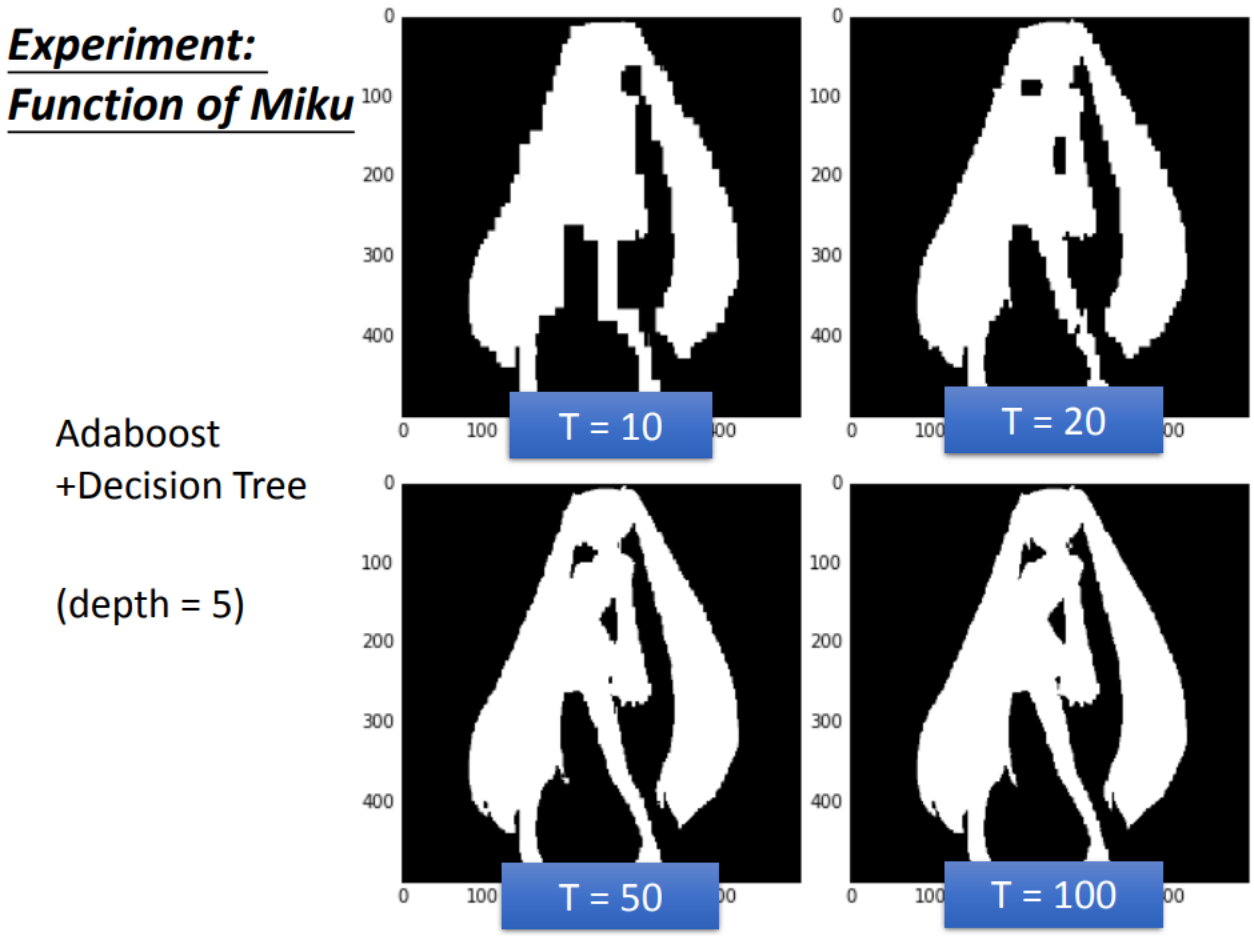

Figure 3.18: AdaBoost Experienment

3.4.7 Gradient Boosting

使用梯度下降的角度来阐释 Boosting, 只不过使我们的梯度和 LearningRate 可以直接求出来，我们可以修改 Boosting 的 object function 来达到不同目的。

- Initial function $g_0(x) = 0$
- For t = 1 to T:
  - Find a function $f_t(x)$ and α_t to improve $g_{t-1}(x)$
    - $g_{t-1}(x) = \sum_{i=1}^{t-1} \alpha_i f_i(x)$
  - $g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$
- Output: $H(x) = sign\big(g_T(x)\big)$

What is the learning target of $g(x)$?

Minimize $L(g) = \sum_n l\big(\hat{y}^n, g(x^n)\big) = \sum_n exp(-\hat{y}^n g(x^n))$

- Find $g(x)$, minimize $L(g) = \sum_n exp(-\hat{y}^n g(x^n))$
  - If we already have $g(x) = g_{t-1}(x)$, how to update $g(x)$?

Gradient Descent:

$$g_t(x) = g_{t-1}(x) - \eta \left.\frac{\partial L(g)}{\partial g(x)}\right|_{g(x)=g_{t-1}(x)}$$

$$\sum_n exp(-\hat{y}^n g_{t-1}(x^n))(\ \hat{y}^n)$$

Same direction

$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$$

$f_t(x)$ ⟺ $\sum_n exp(-\hat{y}^n g_t(x^n))(\hat{y}^n)$

Same direction

We want to find $f_t(x)$ maximizing

Minimize Error

$$\sum_n \underbrace{exp(-\hat{y}^n g_{t-1}(x^n))}_{\text{example weight } u_t^n} \underbrace{(\hat{y}^n) f_t(x^n)}_{\text{Same sign}}$$

$$u_t^n = exp\left(-\hat{y}^n g_{t-1}(x^n)\right) = exp\left(-\hat{y}^n \sum_{i=1}^{t-1} \alpha_i f_i(x^n)\right)$$

$$= \prod_{i=1}^{t-1} exp\left(-\hat{y}^n \alpha_i f_i(x^n)\right)$$

Exactly the weights we obtain in Adaboost

- Find g(x), minimize $L(g) = \sum_n exp(-\hat{y}^n g(x^n))$

$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$$

α_t is something like learning rate

Find α_t minimzing $L(g_{t+1})$

$$L(g) = \sum_n exp\left(-\hat{y}^n\left(g_{t-1}(x) + \alpha_t f_t(x)\right)\right)$$

$$= \sum_n exp\left(-\hat{y}^n g_{t-1}(x)\right) exp\left(-\hat{y}^n \alpha_t f_t(x)\right)$$

$$= \sum_{\hat{y}^n \neq f_t(x)} exp\left(-\hat{y}^n g_{t-1}(x^n)\right) exp(\alpha_t) + \sum_{\hat{y}^n = f_t(x)} exp\left(-\hat{y}^n g_{t-1}(x^n)\right) exp(-\alpha_t)$$

Find α_t such that

$$\frac{\partial L(g)}{\partial \alpha_t} = 0$$

$$\alpha_t = ln\sqrt{(1-\varepsilon_t)/\varepsilon_t}$$

Adaboost!

3.5 Voting

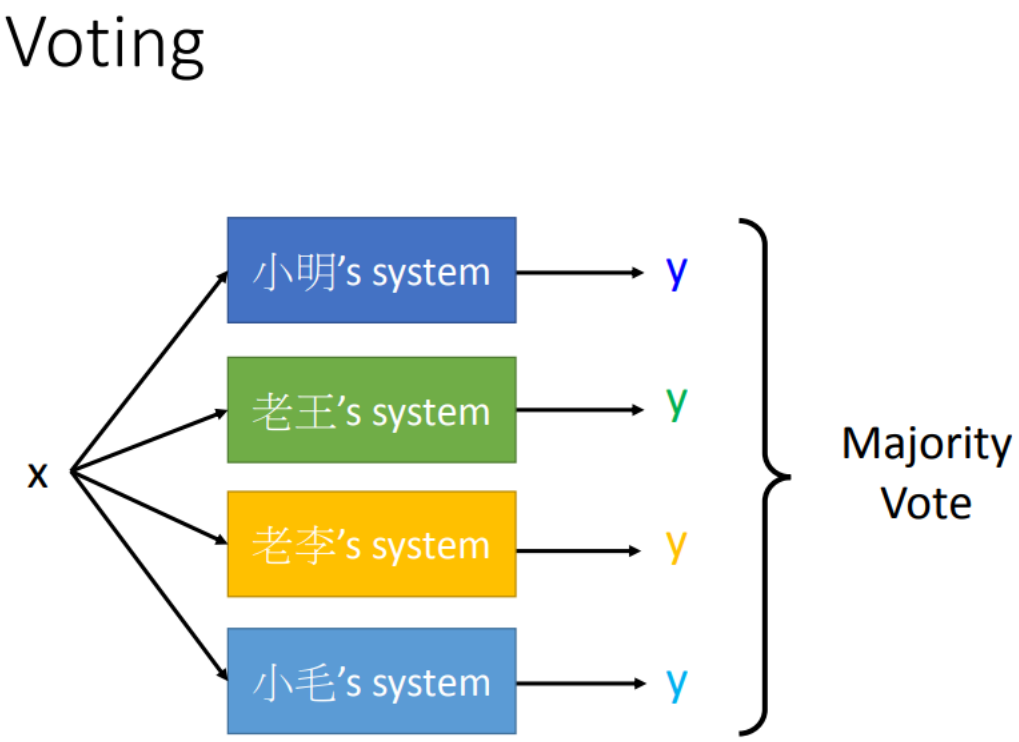

Figure 3.19: AdaBoost Experienment

3.6 Stacking

做 Stacking 时你要将 training data 分成两笔，一笔训练单模型一笔训练 final classifier，目的是要避免过拟合的单模型获得大的权重。

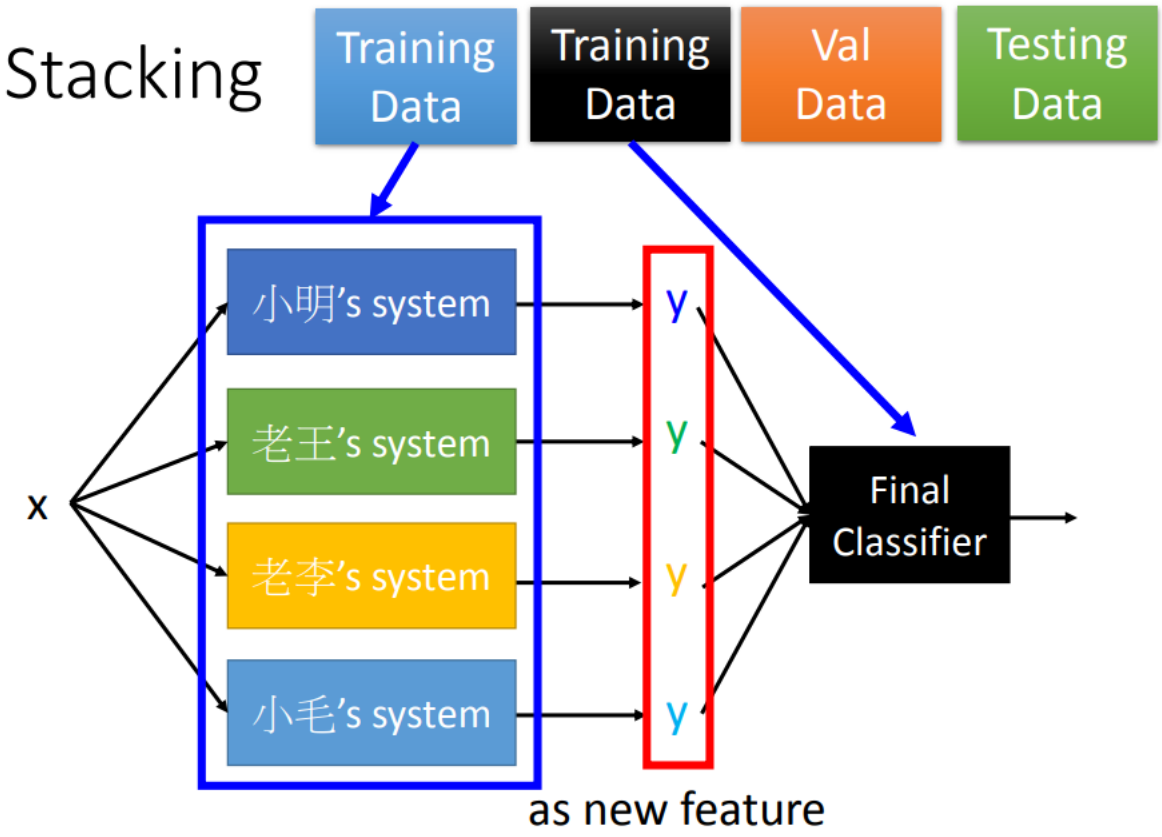

Figure 3.20: AdaBoost Experienment