Machine Learning

上次为 Anomaly detection 异常检测 (无监督学习)

为什么 Problem motivation 问题动机

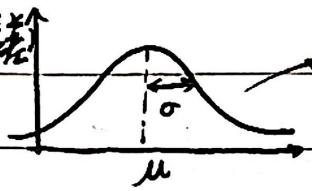
· 对于给定无异常 Dataset: $\{x^{(1)}, x^{(2)}, \cdots, x^{(m)}\}$, → Is x_{test} anomalous?

⇒ 建立 x 的概率模型, 获取阈值 ε → $\begin{cases} P(x_{test}) < \varepsilon \rightarrow \text{Anomaly} \\ P(x_{test}) \geq \varepsilon \rightarrow \text{OK} \end{cases}$

· 常见应用:

欺诈检测, 工程(材料检测), Monitoring computers in a data center.

$x \sim N(\mu, \sigma^2)$ (均值)(方差)/标准差



$p(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

3-2 Gaussian distribution

$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$

$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} (x^{(i)} - \mu)^2$

$p(x) = p(x_1;\mu_1,\sigma_1^2)\,p(x_2;\mu_2,\sigma_2^2)\cdots p(x_n;\mu_n,\sigma_n^2)$

3-3 Algorithm

- 假设 Training set: $\{x^{(1)}, \dots, x^{(m)}\}$ / Each example is $x \in \mathbb{R}^n$.

每个特征 $x_j \sim N(\mu_j, \sigma_j^2)$. → [独立分布] (实际上在算法中，不独立表现也较好)

- Anomaly detection algorithm

1. Choose feature 等 x_i that you think might be indicative of anomalous examples
2. Fit parameters $\mu_1, \dots, \mu_n, \sigma_1^2, \dots, \sigma_n^2$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \quad ; \quad \sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m} (x_j^{(i)} - \mu_j)^2$$

3. Given new example x, compute $p(x)$

$$p(x) = \prod_{j=1}^{n} p(x_j;\mu_j,\sigma_j^2)$$

Anomaly if $p(x) < \varepsilon$.

3-4 Developing and evaluating an anomly detection system

- [Real-number evaluation] (假设有一个正常/异常混合带标签的样本集)

Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ ← 全为正常样本, $y=0$

Cross validation set: $(x_{cv}^{(1)}, y_{cv}^{(1)}), \dots, (x_{cv}^{(m)}, y_{cv}^{(m)})$

Test set: $(x_{test}^{(1)}, y_{test}^{(1)}), \dots, (x_{test}^{(m)}, y_{test}^{(m)})$

} 其中有一些异常样本, $y=1$

↳ [Algorithm evaluation]

- Fit model $p(x)$ on training set $\{x^{(1)}, \dots, x^{(m)}\}$
- On a cross validation/test example x, predict $y = \begin{cases} 1, & \text{if } p(x) < \varepsilon \text{ (Anomaly)} \\ 0, & \text{if } p(x) \geq \varepsilon \text{ (Normal)} \end{cases}$

→ Possible evaluation metrics:

- TP, FP, FN, TN
- Precision / Recall
- F_1-score

Can also use cross validation set to choose parameter ε.

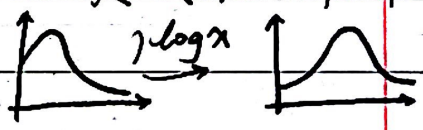
12-5 Anomaly detection vs. supervised learning

- for Anomaly, 适用于：
  - Very small number of positive examples ($y=1$) and large number of negative ($y=0$).
  - 会出不同类型的 anomalies. 很难去学习/预测新异常.
- for supervised,
  - large number of positive and negative examples
  - future positive examples likely to be similar to ones in training set.

12-6 Choosing what features to use.

- For Non-gaussian feature. 可以通过处理让其接近高斯分布

e.g. $x \to \log(x+c)$ (适合处理"拖尾"式：



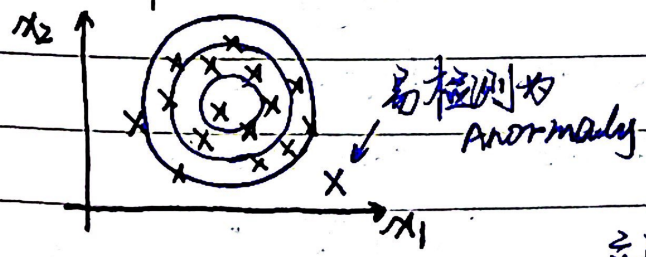
$x \to x^a$ (like $x^{\frac{1}{2}}, x^{\frac{1}{3}} \cdots$)

- Error analysis. 常见问题是 $p(x)$ is comparable (both large) for normal and anormaly examples $\Rightarrow$ 尝试建立新特征

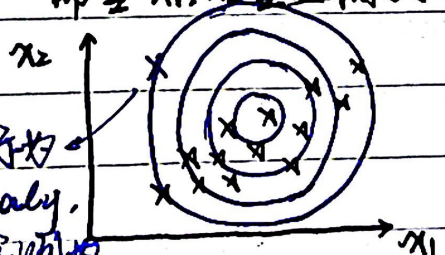
12-7 Multivariate Gaussian distribution 多元高斯分布.

- Motivation: 当特征不独立(e.g 呈正相关)时，预测出错.

E.g. 正常假设下 x_1 和 x_2 是相互独立的特征，则



而当 x_1, x_2 呈正相关时，



为解决这一问题，开发改良版异常检测算法，即多元高斯分布

· Multivariate Gaussian (Normal) distribution

$x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \dots$, etc. separately

Model $p(x)$ all in one go. Parameters: $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

$$\Rightarrow p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

↓ $\det(\Sigma)$

e.g. 1. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}1&0\\0&1\end{bmatrix}$ 2. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}0.6&0\\0&0.6\end{bmatrix}$ ⇒ "变集中"

x_2↑ ⇒ → μ决定中心位置 3. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}2&0\\0&2\end{bmatrix}$ ⇒ "变扩散"

x_1→

4. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}0.6&0\\0&1\end{bmatrix}$ ⇒ x_2↑

5. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}2&0\\0&1\end{bmatrix}$ ⇒ x_2↑

6. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}1&0.5\\0.5&1\end{bmatrix}$ ⇒ x_1→

7. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}1&0.8\\0.8&1\end{bmatrix}$ ⇒ "变狭窄"

8. $\mu=\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma=\begin{bmatrix}1&-0.5\\-0.5&1\end{bmatrix}$ ⇒

13-8 Anomaly detection using the multivariate Gaussian distribution

· 流程: 1. Fit model $p(x)$ by setting $\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$, $\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)}-\mu)(x^{(i)}-\mu)^T$

2. Given a new example x, compute

$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

flag an anomaly if $p(x) < \varepsilon$

· 对比

Original model (用得更多)

$(p(x) = p(x_1;\mu_1,\sigma_1^2)\cdots p(x_n;\mu_n,\sigma_n^2))$

实际上是 $\Sigma=\begin{bmatrix}\sigma_1^2&&0\\&\ddots&\\0&&\sigma_n^2\end{bmatrix}$ 的特殊情况

Multivariate Gaussian

→ Automatically captures correlation between features.

缺点: · computationally more expensive · Must have $m > n$

Δ 如果要解决 "中间题" → manually create features to capture anomalies where x_1, x_2 take unusual combinations of values (e.g. 构建 $x_3 = \frac{x_1}{x_2} = \frac{\text{CPU load}}{\text{memory}}$)

优点: · Computationally cheaper · OK even if m (training set size) is small.

1.14 Recommender Systems 推荐系统

14-1 Problem formulation 问题规划

对于一些问题，算法能够自动学习一些特征。

推荐系统 e.g. 用户给电影打分。

14-2 Content-based recommendations 基于内容的推荐算法

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs karate	0	0	5	?	0	0.9

- $r(i,j)=1$ if user j has rated movie i (0 otherwise)
- $y(i,j)$ = rating by user j on movie i (if defined)
- $\theta^{(j)}$ = ~~parameter~~ vector for movie i) / parameter vector for user j
- $x^{(i)}$ = feature

For user j, movie i, predicted rating: $(\theta^{(j)})^T(x^{(i)})$

e.g. for Alice, $x^{(3)}=\begin{bmatrix}x_0\\x_1\\x_2\end{bmatrix}=\begin{bmatrix}1\\0.99\\0\end{bmatrix}$. 假设已经得到 $\theta^{(1)}=\begin{bmatrix}0\\5\\0\end{bmatrix}$.

则 Alice 对于 "cute puppies of love" 评分为: $(\theta^{(1)})^T x^{(3)}=4.95$

- $m^{(j)}$ = no. of movies rated by user j. To learn $\theta^{(j)}$.

→ Optimization objective

To learn $\theta^{(j)}$ → $\min_{\theta^{(j)}} \frac{1}{2}\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)}-y^{(i,j)})^2+\frac{\lambda}{2}\sum_{k=1}^{n}(\theta_k^{(j)})^2$

To learn $\theta^{(1)}\dots\theta^{(n_u)}$ (n_u user)

→ $\min_{\theta^{(1)},\dots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)}-y^{(i,j)})^2+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$

→ Gradient descent update:

$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)}-y^{(i,j)})x_k^{(i)}$ (for $k=0$) → θ_0 不需正则化项

$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\left(\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)}-y^{(i,j)})x_k^{(i)}+\lambda\theta_k^{(j)}\right)$ (for $k\neq 0$)

14-3 Collaborative filtering 协同过滤

- Optimization algorithm

Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(i)}$:

$$\min_{x^{(i)}} \frac{1}{2} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (x_k^{(i)})^2$$

Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(1)}, \dots, x^{(n_m)}$:

$$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^{(i)})^2$$

→ Collaborative filtering → (and movie ratings),

Given $x^{(1)}, \dots, x^{(n_m)}$, can estimate $\theta^{(1)}, \dots, \theta^{(n_u)}$.

Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, can estimate $x^{(1)}, \dots, x^{(n_m)}$.

可以随机猜取一些 θ 值 → x → θ → x → ……

14-4 Collaborative filtering Algorithm

- Minimizing $x^{(1)}, \dots, x^{(n_m)}$ and $\theta^{(1)}, \dots, \theta^{(n_u)}$ simultaneously (将被上文设置)

1. Initialize $x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}$ to small random values

2. Minimize $J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)})$

→ $\min_{\substack{x^{(1)} \dots x^{(n_m)} \\ \theta^{(1)} \dots \theta^{(n_u)}}} J(\dots) = \min_{\dots} \frac{1}{2} \sum_{\substack{i:j: \\ r(i,j)=1}} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^{(i)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^{(j)}$

⇒ Using gradient descent (or other algorithms) e.g. for every $\begin{matrix} i = 1, \dots, n_m \\ j = 1, \dots, n_u \end{matrix}$

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right) \theta_k^{(j)} + \lambda \theta_k^{(j)} \right)$$

3. For a user with parameters θ and a movie with (learned) features x, predict a rating of $\theta^T x$.

(注: $x \in \mathbb{R}^n$, $\theta \in \mathbb{R}^n$, 无 x_0, 因为需要算法自动学习特征)

14-5 Vectorization: Low rank matrix factorization 矢量化：低秩矩阵分解

· Predicted rating:

$$\begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \cdots & (\theta^{(n_u)})^T(x^{(1)}) \\ \vdots & \vdots & \vdots & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \cdots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix} \rightarrow X\theta^T$$

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix} \Rightarrow X = \begin{bmatrix} -(x^{(1)})^T- \\ -(x^{(2)})^T- \\ \vdots \\ -(x^{(n_m)})^T- \end{bmatrix} \quad \theta = \begin{bmatrix} -(\theta^{(1)})^T- \\ \vdots \\ -(\theta^{(n_u)})^T- \end{bmatrix}$$

· Finding related movies

For product i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$

Learn → 5 most similar movies to movie i:

⇒ Find the 5 movies j with the smallest $\|x^{(i)} - x^{(j)}\|$

14-6 Implementational detail: Mean normalization 均值规范化

假设用户 Eve(5) 对前文5个电影均未评分，→ 预测 Eve(5) 对电影评分.

→ 求 $\theta^{(5)}$，因为对 Eve 而言，$r(i,j)$ 均为 0，$\min J(\dots)$ 只有 $\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$ 一项起作用，→ 使 $\theta^{(5)}$ 尽可能小，i.e. $\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

⇒ Eve(5) 对评分均为 0，没有意义

· Mean Normalization:

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \quad \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix} \rightarrow Y = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

For user j, on movie i predict:

$$(\theta^{(j)})^T(x^{(i)}) + \mu_i$$

User 5 (Eve):

$$(\theta^{(5)})^T(x^{(i)}) + \mu_i = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix}$$

15 Large scale machine learning 大规模机器学习

15-1 Learning with large datasets

* 初步检查：假设 train set $m = 100,000,000$，需要考虑选择其中1000个样本训练结果会不会相似（参见 8-6）

15-2 Stochastic gradient descent 随机梯度下降

对于训练样本过大的情况，例如 $m = 300,000,000$

$$J_{train}(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(\theta, (x^{(i)}, y^{(i)}))$$

Batch gradient descent (批量…)

$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

→ Repeat {

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right]$$

(for every $j = 0, \cdots, n$)

$$\frac{\partial}{\partial \theta_j} J_{train}(\theta)$$

每一次迭代都要求 m 样本

}

Stochastic gd

$$Cost(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

1. Randomly shuffle (reorder) training example

2. Repeat {

for $i := 1, \cdots, m$ {

$$\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

(for every $j = 0, \cdots, n$)

一次迭代针对单个样本

} }

15-3 Mini-batch gradient descent.

- Batch gd: Use all m examples in each iteration
- Stochastic gd: Use 1 example in each iteration
- Mini-batch gd: Use b example in each iteration

E.g. Say $b = 10$, $m = 1000$

Repeat { for $i = 1, 11, 21, 31, \cdots 991$

$$\theta_j := \theta_j - \alpha\frac{1}{10}\sum_{k=i}^{i+9}(h_\theta(x^{(k)}) - y^{(k)})x_j^{(k)}$$

for every $j = 0, \cdots, n$. } }

15-4 Stochastic gradient descent convergence 收敛 ← 如何保证学习率α

· Cost $(\theta, (x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$ ← 可以用来检查算法表现

→ During learning, compute Cost $(\theta, (x^{(i)}, y^{(i)}))$ before updating θ using $(x^{(i)}, y^{(i)})$

→ (一般) Every 1000 iterations, plot Cost $(\theta, (x^{(i)}, y^{(i)}))$ averaged over the last 1000 examples processed by algorithm.

· 一般地，学习率α总设定为用不变常数。但，为了更好收敛到全局最小值，可以让学习率α随时间逐渐减小，E.g. $\alpha = \frac{\text{Costant-1}}{\text{no.of iteration} + \text{Costant-2}}$

(注：Costant 1 和 2 是需要额外设置的参数) 迭代次数

15-5 Online learning

E.g. Product Search (learning to search)

User searches for "Android phone"

Have 100 phone in store. Will return 10 results.

x = features of phone, how many words in user query match name of phone, how many words in query match description of phone, etc.

y = 1 if users click on link, y=0, otherwise. → learn $p(y=1|x;\theta)$ (predicted CTR) ← 点击率预测学习问题

Use to show user the 10 phones they're most likely to click on.

实际上每个用户访问网站，就得到十个样本数据，从每一部手机得到一个特征矢量x，以及对应y值 ⇒ 如何运行该网站？

→ 不停地给用户展示你对他们可能会喜欢的十个手机的预测，那么每次一个用户访问，你将会得到十个样本-(x,y)数据对，然后利用一个在线学习算法来更新你的参数，i.e. $\theta_j := \theta_j - \alpha(h_\theta(x) - y) \cdot x_j$ 据

对这十个样本利用十步梯度下降法来更新参数，→ 然后可以丢弃这些数

(因为具有大量用户流

不必要使用重复数据)

↗ 数据过于庞大以至于无法使用单个设备

15-6 Map-reduce and data parallelism 减少映射与数据并行

E.g. 假设 $m = 400$

Training Set → 均匀分成4份

Use $(x^{(1)}, y^{(1)}), \dots, (x^{(100)}, y^{(100)})$.

Computer 1: $temp_j^{(1)} = \sum_{i=1}^{100} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Computer 2 ⋮

3 ⋮

Use $(x^{(201)}, y^{(201)}), \dots (x^{(300)}, y^{(300)})$

4 ⋮

$temp_j^{(4)} = \sum_{i=301}^{400} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$

Combine results

$\theta_j := \theta_j - \alpha \frac{1}{400} (temp_j^{(1)} + \dots + temp_j^{(4)})$

$(j = 0, \dots, n)$

16 Application Example: Photo OCR 应用实例：图片文字识别

16-1 Problem description and pipeline 问题描述和流程图

- Photo OCR pipeline:

Image → Text detection (文字检测) → Character segmentation (字符分割) → Character recognition (字符检测)

16-2 Sliding window 滑动窗口

- Text detection: 利用一系列包含 { 正样本 [y=1] (对应区域有文字); 负样本 [y=0] (一无文字) } 的训练集 (监督学习)

→ 训练一个固定比例的滑动窗口分类器.

→ 应用到新图片上 ⇒ 得到一张通过像素值大小反映某处有文字的概率 → 进行一定的形态学处理 → 将含有文字的区域进行分离.

- Character segmentation: 再次使用监督学习算法. { 正样本 y=1 ⇒ 存在"文字分割"的部分; 负 y=0 }

↓

E.g. 中 "W" 和 "A"

16-3 Getting lots of data: Artificial data synthesis 获取大量数据和人工数据合成

· 一个可靠的得到高性能机器学习系统的方法:
使用一个低偏差机器学习算法，并且使用庞大的训练集去训练它.

→ 如何获得大量数据集? 对于特定问题可以采用人工数据合成.

一般两种形式 { I. 重新创造新数据
II. 已经有小的标签训练集，以某种方式扩充训练集

I. E.g. for Photo OCR, 可利用免费字体库，将不同字体和不同背景图结合来创建一个接近真实数据的人工合成训练集.

II. 引入失真 (introducing distortions)

E.g. 语音识别: Audio: Background noise, bad cellphone connection.

→ Distortion introduced should be representation of the type of noise/distortions in the test set.

Usually does not help to add purely random/meaningless noise to your data

(上限分析)

16-4 Ceiling analysis: What part of the pipeline to work on next.

· Image → Text detection → Character segmentation → Character recognition.

→ What part of the pipeline should you spend the most time trying to improve?

component	Accuracy	← 整个系统
Overall system	72%	
Text ... 将之调为 100% accuracy	89%	↑ 17%
Ch... seg...	90%	↑ 1%
Ch... rec...	100%	↑ 10%

2,3 均为 100% accuracy

依次将流程 2,3,4 项通过人工控制调试为 100% accuracy 观察整个系统的 accuracy 增益