Prof. Andrew Ng — Winter 2003 — 7/14/19
Fall 2019 — 9/23/19

Machine Learning — Stanford Eng, Everywhere — CS221 : L1

The Motivations & Applications of Machine Learning

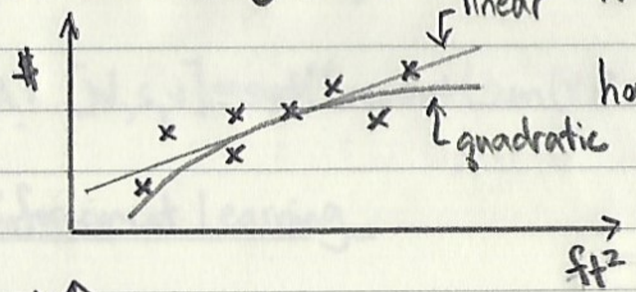
optical character recognition - US post zipcode
flying helicopter
mining databases - medical records
identifying fraudulent transactions
personalized recommendations
understanding human genome

Machine Learning defition:

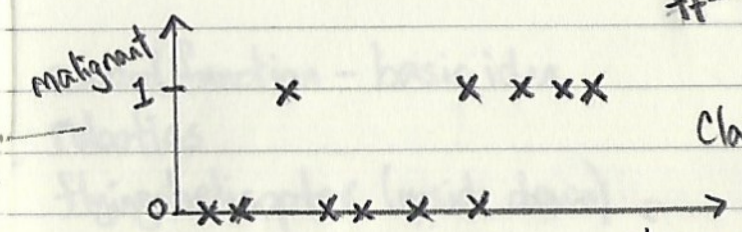
Arthur Samuel (1959) - gives computers the ability to learn without being explicitly programmed. Checkers self-play

Tom Mitchell (1998) - learns from experience E with respect to task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

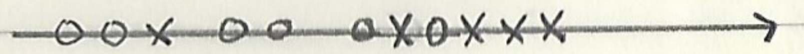
#1 Supervised Learning: learn x to y mappings — #2 Deep Learing



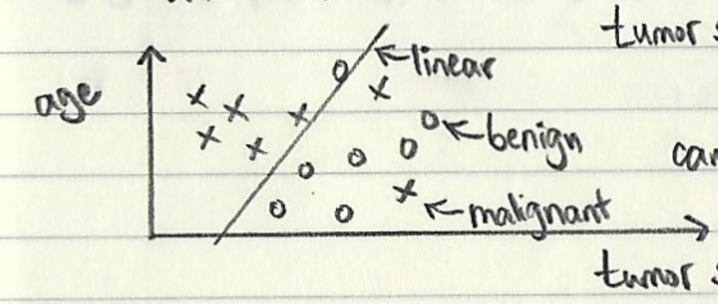
housing prices - regression: continuous value prediction



Classification : discrete value predicti



o - benign
x - malignant



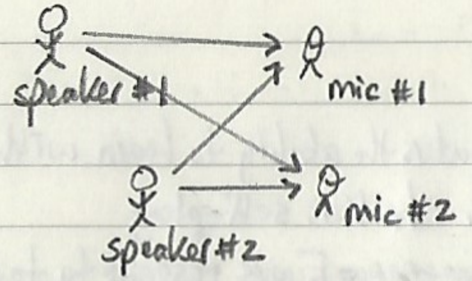
can be many dimensions up to ∞

$x - \begin{bmatrix} age \\ size \end{bmatrix} \mapsto y - \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

7/14/19

CS221: L1

Cocktail party problem:

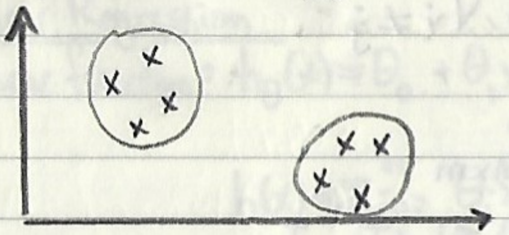


independant component analysis (ICA)

#3 Learning Theory - Practical ML advice

When can you guarentee that a learning algorithm will work?
What algorithms will approximate which fuctions?
How much data is needed to train?

#4 Unsupervised Learning - only input X



clustering -
discover structure in the data

gene analysis
organize computing clusters
image clustering - grouping pixels together
social network analysis
grouping news articles
market segmentation
astronomical data analysis

```
ICA: [W,s,v]=svd((repmat(sum(x.*x,1),size(x,1),1).*x)*x');
```

#5 Reinforcement Learning

reward function - basic idea
robotics
flying helicopter (upside down)

9/28/19

CS229:S1

Linear Algebra

vector $v \in \mathbb{R}^n$ — n-dim real space

matrix $A \in \mathbb{R}^{m \times n}$ $\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$ m row vectors, n col vectors

identity I^n $\begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix}$

diagonal D^n $\begin{bmatrix} * & & 0 \\ & \ddots & \\ 0 & & * \end{bmatrix}$ $D_{ij} = 0 \; \forall i \neq j$

transpose $A \in \mathbb{R}^{m \times n}$, $A^T \in \mathbb{R}^{n \times m}$

$A_{ij} = (A^T)_{ji}$ $(A+B)^T = A^T + B^T$

symetric $A = A^T$ $(AB)^T = B^T A$

inner product: $v \in \mathbb{R}^n$ $u \in \mathbb{R}^n$

$v^T u = \langle v; u \rangle = \sum_{i=1}^{n} v_i u_i \in \mathbb{R}$

$\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$

outer product: $v \in \mathbb{R}^m$ $u \in \mathbb{R}^n$

$v \cdot u^T \in \mathbb{R}^{m \times n}$ $(v \cdot u^T)_{ij} = v_i u_j$

7/14/19

CS229:N1

Supervised Learning

$(x^{(i)}, y^{(i)})$ - training example $\{(x^{(i)}, y^{(i)}); i = 1 \ldots m\}$ - training set

input, target

$X = Y = \mathbb{R}$

learn function $h: X \mapsto Y$ so $h(x)$ is a good predictor

hypothesis

Linear Regression

linear function: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$, θ_i - parameters, weights

features

$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$, $x_0 = 1$ - intercept term

$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ - cost function

ordinary least squares

choose θ to minimize $J(\theta)$

num features

gradient descent: $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$, $j = 0 \ldots n$

learning rate

LMS update: $\theta_j := \theta_j - \alpha (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$ - single training example

repeat until convergence {

$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$, for every j }

batch gradient descent

J is a convex quadratic function - gradient descent always converges

loop {

for i=1 to m {

$\theta_j := \theta_j + \alpha (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$, for every j }}

stocastic gradient descent

may never converge

9/28/19

CS229:S1

matrix · vector : $A \in \mathbb{R}^{m \times n}$ $x \in \mathbb{R}^n$

1) $A = [a_1, a_2 \dots a_n]$ $a_i \in \mathbb{R}^m$

$y = Ax = a_1 x_1 + a_2 x_2 + \dots + a_n x_n$

linear combination of columns of A

2) $A = \begin{bmatrix} -a_1^T- \\ \vdots \\ -a_m^T- \end{bmatrix}$ $a_i = \mathbb{R}^n$

$$y = Ax = \begin{bmatrix} a_1^T x \\ \vdots \\ a_m^T x \end{bmatrix} \in \mathbb{R}^{m \times 1}$$

matrix · matrix : $A \in \mathbb{R}^{m \times k}$ $B \in \mathbb{R}^{k \times n}$

$A \cdot B \in \mathbb{R}^{m \times n}$

$$A = \begin{array}{cc} \begin{bmatrix} — \\ — \\ — \end{bmatrix} & m \\ k & \end{array} \quad B = \begin{array}{cc} \begin{bmatrix} | & | & | \end{bmatrix} & k \\ n & \end{array}$$

operation & properties:

trace – $tr(A) = \Sigma_{i=1}^{n} A_{ii}$, $\mathbb{R}^{n \times n} \to \mathbb{R}$ only square matrix

range & null space: $A \in \mathbb{R}^{m \times n}$

range of A = $R(A) = \{ v \in \mathbb{R}^m : v = A \cdot x, x \in \mathbb{R}^n \}$

projection: projection of vector $y \in \mathbb{R}^m$ onto span of $\{x_1, \dots, x_n\}$ is

$v = Proj(y; x)$ vector v s.t. v is as close as possible to y

$= \underset{v \in R(x)}{\arg\min} \|v - y\|_2$

$= x(x^T x)^{-1} x^T y$

null space $= \{ x \in \mathbb{R}^n, Ax = 0 \}$

7/14/19

CS229 = N1

The Normal Equations

for function $f : \mathbb{R}^{m \times d} \mapsto \mathbb{R}$, $\nabla_A f(A) = $ (m×n matrix)

$$\nabla_A f(A) = \begin{array}{cc} \begin{bmatrix} \partial f / \partial A_{11} & \cdots & \partial f / \partial A_{1d} \\ \vdots & \ddots & \vdots \\ \partial f / \partial A_{n1} & \cdots & \partial f / \partial A_{nd} \end{bmatrix} & n \\ d & \end{array}$$

trace operator: $trA = \Sigma_{i=1}^{d} A_{ii}$ sum of diagonal entries of n-by-n A

$trAB = trBA$ $trABC = trCAB = trBCA$

cyclic permutation property

$trA = trA^T$

$tr(A+B) = trA + trB$

$tr\, aA = a\, trA$

$\nabla_A trAB = B^T$ B fixed $\nabla_A tr AA^T C = CA + C^T A$

$\nabla_{A^T} f(A) = (\nabla_A f(A))^T$

$\nabla_A trABA^T C = CAB + C^T AB^T$

$\nabla_A |A| = |A| (A^{-1})^T$ – non-singular square A, |A| is determinant

$$X = \begin{array}{cc} \begin{bmatrix} —(x^{(1)})^T— \\ \vdots \\ —(x^{(d)})^T— \end{bmatrix} & n \\ d & \end{array}, \quad \vec{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(d)} \end{bmatrix}, \quad X\theta - \vec{y} = \begin{bmatrix} h_\theta(x^{(1)}) - y^{(1)} \\ \vdots \\ h_\theta(x^{(d)}) - y^{(d)} \end{bmatrix}$$

$$\tfrac{1}{2}(X\theta - \vec{y})^T (X\theta - \vec{y}) = \tfrac{1}{2} \Sigma_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2 = J(\theta)$$

recall $z^T z = \Sigma_i z_i^2$

$\nabla_\theta J(\theta) = \nabla_\theta \tfrac{1}{2} (X\theta - \vec{y})^T (X\theta - \vec{y})$

$= X^T X \theta - X^T \vec{y}$

$= 0$

$\therefore X^T X \theta = X^T \vec{y}$

$\theta = (X^T X)^{-1} X^T \vec{y}$ – closed form

9/28/19

CS221: S1

determinant : $A \in \mathbb{R}^{n \times n}$
$\det(A) \to \mathbb{R}$

$|A| = |A^T|$
$|A \cdot B| = |A| \cdot |B|$
$|A| = 0$, A is not full rank
$|A^{-1}| = 1/|A|$ if A is invertible

rank(A) : size of largest subset of columns of A that are linerly independent

$A \in \mathbb{R}^{n \times n}$
eigenvalue of A is $\lambda \in \mathbb{C}$ (complex number)
u is eigenvector of A if $Au = \lambda u$ $(u \neq 0)$
$$Au - \lambda u = 0 \iff (A - \lambda I)u = 0$$
$$\Updownarrow$$
$$\det(A - \lambda I) = 0$$

1) $\forall A \in \mathbb{R}^{n \times n}$, n eigenvalues (not necessarily unique)
$\det(I - \lambda I) = (\lambda - 1)^n$
2) $\forall u$ eigenvector of A
$\exists \lambda$ s.t. $Au = \lambda u$
$\forall c \neq 0$, cu also an eigenvector $A(cu) = \lambda(cu)$
λ eigenvalue u_1, u_2 eigenvectors, $Au_1 = \lambda u_1$, $Au_2 = \lambda u_2$
$\forall c_1, c_2 - c_1u_1 + c_2u_2$ is still eigenvector of λ
3) $tr(A) = \sum_{i=1}^{n} \lambda_i$
$\det(A) = \prod_{i=1}^{n} \lambda_i$
$\text{rank}(A) = \text{nnz}(\{\lambda_i\}_{i=1}^{n})$

7/14/19

CS229: N1

Probabalistic Interpretation

$y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}$, $\varepsilon^{(i)}$ is error term, IID (independent, identically distrib

$$\varepsilon^{(i)} \sim \mathcal{N}(0, \sigma^2), \; p(\varepsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\varepsilon^{(i)})^2}{2\sigma^2}\right)$$
density of $\varepsilon^{(i)}$

$$p(y^{(i)} | x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$
parameterized by θ, not a random variable

distribution of $y^{(i)} = y^{(i)} | x^{(i)}; \theta \sim \mathcal{N}(\theta^T x^{(i)}, \sigma^2)$

liklihood function $L(\theta) = L(\theta; X, \vec{y}) = p(\vec{y} | X; \theta)$
$= \prod_{i=1}^{m} p(y^{(i)} | x^{(i)}; \theta)$
$= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$

log likihood $\ell(\theta) = \log L(\theta) = m \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$
maximizing $\ell(\theta)$ means minimizing $\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$
independent of σ — same as $J(\theta)$ least-squares

Locally Weighted Linear Regression (LWR)

1. fit θ to minimize $\sum_i w^{(i)} (y^{(i)} - \theta^T x^{(i)})^2$
2. output $\theta^T x^{(i)}$

non-negative weights

standard choice for $w^{(i)} = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)$, τ is bandwidth paramete

non-parametric algorithm - need entire training set to make predictions

8/5/19

CS229: N1

Classification and Logistic Regression

values of y take on small number of discrete values

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}, \quad g(z) = \frac{1}{1+e^{-z}}, \quad g'(z) = g(z)(1-g(z))$$

↑ sigmoid

assume $P(y=1 \mid x;\theta) = h_\theta(x)$, $P(y=0 \mid x;\theta) = 1 - h_\theta(x)$ $\Big\}$ $p(y \mid x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y}$

↑ binary classification

likelihood of parameters $L(\theta) = p(\vec{y} \mid X;\theta) = \prod_{i=1}^m (h_\theta(x^{(i)}))^{y^{(i)}} (1-h_\theta(x^{(i)}))^{1-y^{(i)}}$

$\ell(\theta) = \log L(\theta) = \sum_{i=1}^m y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)}))$ ← maximize

gradient ascent $\theta := \theta + \alpha \nabla_\theta \ell(\theta)$

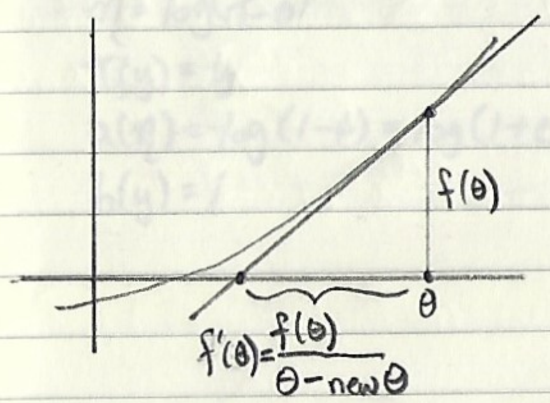
↑ + because maximizing log likelihood

$\frac{\partial}{\partial \theta_j} \ell(\theta) = (y - h_\theta(x)) x_j$

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

Newton's method: to find θ such that $f(\theta) = 0$, use update rule

$$\theta := \theta - \frac{f(\theta)}{f'(\theta)}, \quad \theta \in \mathbb{R}$$



To maximize $\ell(\theta)$, find $\ell'(\theta) = 0$ using Newton's method:

$$\theta := \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$$

CS229: N1

Gaussian distribution $-\ p(y;\mu)=\frac{1}{\sqrt{2\pi}}\exp(-\frac{1}{2}(y-\mu)^2)$

$$=\frac{1}{\sqrt{2\pi}}\exp(-\tfrac{1}{2}y^2)\cdot\exp(\mu y-\tfrac{1}{2}\mu^2)$$

$\sigma^2=1$ since σ^2 has no effect on θ or $h_\theta(x)$

$\eta=\mu$, $T(y)=y$, $a(\eta)=\mu^2/2=\eta^2/2$, $b(y)=\frac{1}{\sqrt{2\pi}}\exp(-\frac{y^2}{2})$

Constructing GLMs - predict random y as a function of x

three assuptions for GLM:

1. $y|x;\theta\sim \text{ExponentialFamily}(\eta)$
2. given x, predict expected value of $T(y)$ given x
   if $T(y)=y$, $h(x)=E[y|x]$
3. $\eta=\theta^T x$ - η and inputs are related linearly

ordinary least squares - $h(x)=E[y|x;\theta]=\mu=\eta=\theta^T x$
  $y|x;\theta\sim\mathcal{N}(\mu,\sigma^2)$ - Gaussian

logistic regression - $h(x)=E[y|x;\theta]=\phi=1/(1+e^{-\eta})=1/(1+e^{-\theta^T x})$
  $y|x;\theta\sim\text{Bernoulli}(\phi)$

canonical response function - $g(\eta)=E[T(y);\eta]$
canonical link function - g^{-1}

softmax regression - $y\in\{1,2,\dots,K\}$, K classes
  parameterize multinomial with $\phi_1,\dots,\phi_{K-1}$
    $\phi_i=p(y=i;\phi)$ and $p(y=K;\phi)=1-\Sigma_{i=1}^{K-1}\phi_i$
    $\phi_K=1-\Sigma_{i=1}^{K-1}\phi_i$ - not a parameter since fully specified by $\phi_1,\dots,\phi_{K-1}$

8/18/19

CS229: N1

$T(y) \in \mathbb{R}^{k-1}$ - one-hot vectors

$$T(1) = \begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, T(2) = \begin{bmatrix}0\\1\\0\\\vdots\\0\end{bmatrix}, T(k-1) = \begin{bmatrix}0\\0\\\vdots\\0\\1\end{bmatrix}, T(k) = \begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix}$$

$(T(y))_i = i^{th}$ element of vector $T(y)$

indicator function - $1\{\cdot\} = 1$ if true, 0 if false $\therefore (T(y))_i = 1\{y=i\}$

$$E[(T(y))_i] = P(y=i) = \phi_i$$

$$\begin{aligned}
p(y;\phi) &= \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \dots \phi_k^{1\{y=k\}} \\
&= \phi_1^{(T(y))_1} \phi_2^{(T(y))_2} \dots \phi_k^{1-\sum_{i=1}^{k-1} (T(y))_i} \\
&= \exp((T(y))_1 \log \phi_1 + (T(y))_2 \log \phi_2 + (1 - \textstyle\sum_{i=1}^{k-1} (T(y))_i) \log \phi_k) \\
&= \exp((T(y))_1 \log \phi_1/\phi_k + (T(y))_2 \log \phi_2/\phi_k + (T(y))_{k-1} \log \phi_{k-1}/\phi_k + \log \phi_k)
\end{aligned}$$

$$\eta = \begin{bmatrix}\log \phi_1/\phi_k \\ \log \phi_2/\phi_k \\ \vdots \\ \log \phi_{k-1}/\phi_k\end{bmatrix}, \ a(\eta) = -\log \phi_k, \ b(y) = 1$$

link function (for $i = 1, \dots, k$) - $\eta_i = \log \phi_i/\phi_k$, $\eta_k = \log \phi_k/\phi_k = 0$

response function - $e^{\eta_i} = \phi_i/\phi_k$

$$\phi_k e^{\eta_i} = \phi_i$$

substitution

$$\phi_k \sum_{i=1}^{k} e^{\eta_i} = \sum_{i=1}^{k} \phi_i = 1 \therefore \phi_k = 1/\sum_{i=1}^{k} e^{\eta_i}$$

$$\phi_i = e^{\eta_i}/\sum_{j=1}^{k} e^{\eta_j}$$ ← softmax function

$$p(y=i|x;\theta) = \phi_i = e^{\eta_i}/\sum_{j=1}^{k} e^{\eta_j} = e^{\theta_i^T x}/\sum_{j=1}^{k} e^{\theta_j^T x}$$

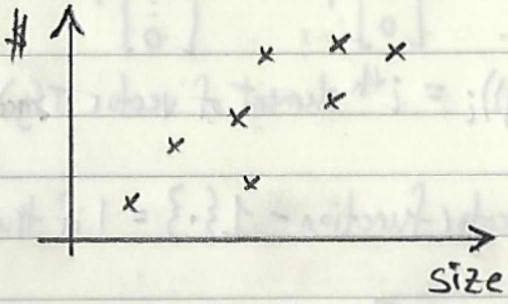
$$h_\theta(x) = E[T(y)|x;\theta] = E\left[\begin{matrix}1\{y=1\}\\1\{y=2\}\\\vdots\\1\{y=k-1\}\end{matrix}\middle| x;\theta\right] = \begin{bmatrix}\phi_1\\\phi_2\\\vdots\\\phi_{k-1}\end{bmatrix}$$

Supervised learning : $x \to y$
regressions vs. classification

Housing data : training set

		size	#bedrooms	price ('000)
$x^{(1)}$	1	2104	4	\$400
$x^{(2)}$	1	1416	3	\$232
	⋮	⋮	⋮	⋮

↓ n rows



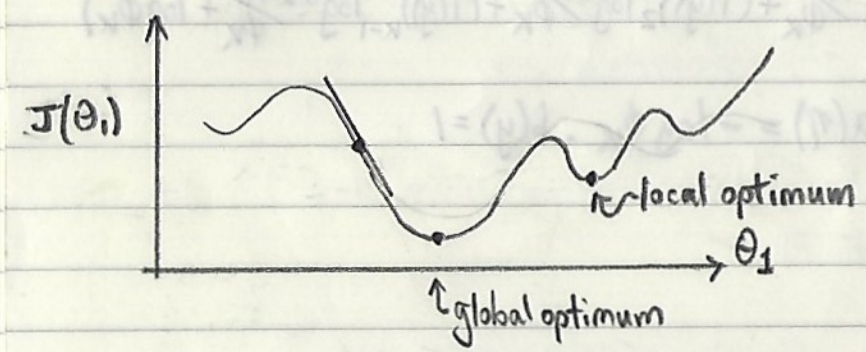
$h(x) = \theta_0 + \theta_1 x$ ← affine function

x_1 : size

x_2 : # bedrooms

$x_1^{(2)} = 1416$

$\vec{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$ $\vec{X} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$, $x_0 = 1$



Prof. Andrew Ng | Winter 2003 | 8/19/19
Prof. Moses Charikar | Fall 2019 | 9/24/19

Machine Learning | CS229: L2

Linear Regression, Gradient Descent, Normal Equations

continuous value prediction - regression problem

n - # training examples, (x, y) | d - # features

x - input variables / features | ↑ x, θ are $(d+1)$ dimensional

y - output variable / target

$(x^{(i)}, y^{(i)})$ - i^{th} training example

training examples
↓
learning algorithm | "hypothesis"
↓
input → $\boxed{h}$ → estimate

linear
↓
$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$
$= \sum_{j=0}^{2} \theta_j x_j = \theta^T x$
↑ $x_0 = 1$
$= \sum_{j=0}^{d} \theta_j x_j$

predicted value

θ - parameters

$\min_\theta \frac{1}{2} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)})^2 = \min_\theta J(\theta)$ ↑ cost function

start with $\theta = \vec{0}$ (init to zeros)

Keep changing θ to reduce $J(\theta)$

learning rate

<u>gradient descent</u> : $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) := \theta_j - \alpha (h_\theta(x) - y) x_j$

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2}(h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2} (h_\theta(x) - y) \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) \\
&= (h_\theta(x) - y) \frac{\partial}{\partial \theta_j} (\theta_0 x_0 + \dots + \theta_d x_d - y) \\
&= (h_\theta(x) - y) x_j
\end{aligned}$$
← single training example

repeat until convergence : batch gradient descent (works for small n)

$\theta_j := \theta_j - \frac{\alpha}{n} \sum_{i=1}^{n} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)} := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$

↑ n training examples

ordinary least squares - one global min.
with linear h

$\nabla_\theta \operatorname{tr} \theta\theta^T X^T X = \nabla_\theta \operatorname{tr} \underset{A}{\theta}\underset{B}{I}\underset{A}{\theta^T}\underbrace{X^T X}_{C} = X^T X\theta I + X^T X \theta I$

$= X^T X \theta + X^T X \theta$

$\nabla_\theta \operatorname{tr} \underbrace{y^T}_{B} \underset{A}{X\theta} = \underset{B^T}{X^T y}$

$X^T X \theta = X^T \vec{y}$

$\theta = (X^T X)^{-1} X^T \vec{y}$

not invertible if dependent features — pseudo inverse

8/19/19

CS229: L2

repeat until convergence: <u>stochastic</u> <u>gradient</u> <u>descent</u>

for $i = 1$ to n:

$\theta_j := \theta_j - \alpha (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$ (for all $j = 0, \dots, d$)

won't converge to global minimum — need to "time out"

<u>Normal equations</u>

$$\nabla_\theta J(\theta) = \begin{bmatrix} \partial J / \partial \theta_0 \\ \vdots \\ \partial J / \partial \theta_d \end{bmatrix} \in \mathbb{R}^{d+1}$$

gradient descent: $\theta := \theta - \alpha \nabla_\theta J$

$\uparrow \mathbb{R}^{d+1}$

$f: \mathbb{R}^{n \times d} \mapsto \mathbb{R}$, $f(A)$ where $A \in \mathbb{R}^{n \times d}$

$$\nabla_A f(A) = \begin{bmatrix} \partial f / \partial A_{11} & \cdots & \partial f / \partial A_{1d} \\ \vdots & \ddots & \vdots \\ \partial f / \partial A_{n1} & \cdots & \partial f / \partial A_{nd} \end{bmatrix}, \quad \operatorname{tr} A = \textstyle\sum_{i=1}^d A_{ii} \text{ - sum diag. elements}$$

$$X = \begin{bmatrix} — (x^{(1)})^T — \\ \vdots \\ — (x^{(n)})^T — \end{bmatrix}, \quad X\theta = \begin{bmatrix} — x^{(1)T}\theta — \\ \vdots \\ — x^{(n)T}\theta — \end{bmatrix} = \begin{bmatrix} h_\theta(x^{(1)}) \\ \vdots \\ h_\theta(x^{(n)}) \end{bmatrix}$$

$$\vec{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}, \quad X\theta - \vec{y} = \begin{bmatrix} h_\theta(x^{(1)}) - y^{(1)} \\ \vdots \\ h_\theta(x^{(n)}) - y^{(n)} \end{bmatrix} \updownarrow n$$

recall $z^T z = \sum_i z_i^2$

$\uparrow$ vector

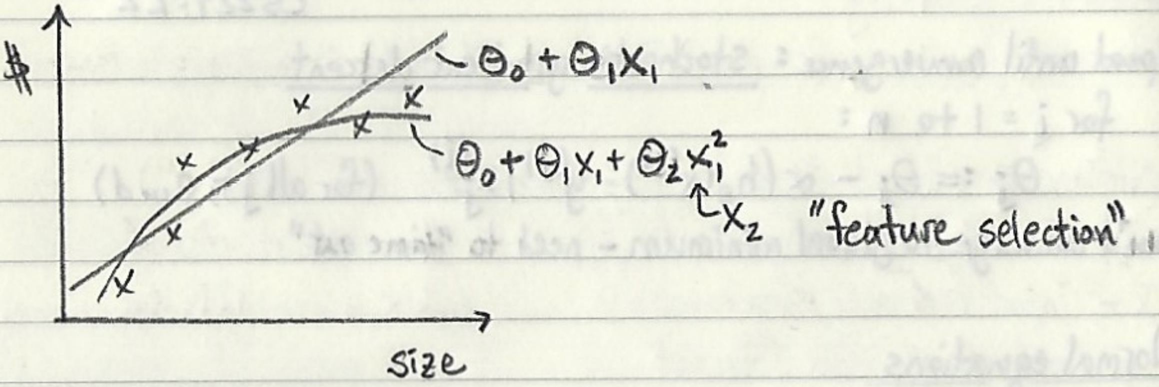
$\frac{1}{2}(X\theta - \vec{y})^T (X\theta - \vec{y}) = \frac{1}{2} \sum_{i=1}^n (h_\theta(x^{(i)}) - y^{(i)})^2 = J(\theta)$

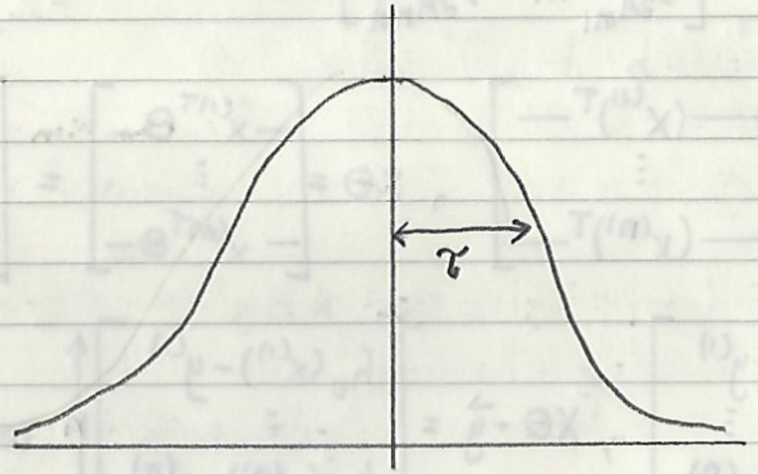
$\nabla_\theta J(\theta) = \vec{0} = \nabla_\theta \frac{1}{2} (X\theta - \vec{y})^T (X\theta - \vec{y})$

$= \frac{1}{2} \nabla_\theta \operatorname{tr} (\theta^T X^T X \theta - \theta^T X^T \vec{y} - \vec{y}^T X \theta + \underbrace{\vec{y}^T \vec{y}}_{\text{does not depend on } \theta})$

$= \frac{1}{2} [\nabla_\theta \operatorname{tr} \theta\theta^T X^T X - \nabla_\theta \operatorname{tr} \vec{y}^T X\theta - \nabla_\theta \operatorname{tr} y^T X \theta]$

$= \frac{1}{2} [X^T X \theta + X^T X \theta - X^T \vec{y} - X^T \vec{y}] = X^T X \theta - X^T \vec{y} = 0$





$w^{(i)}$ = weighting function

Prof. Andrew Ng — Winter 2003 — 8/21/19
Prof. Moses Charikar — Fall 2019 — 9/30/19

Machine Learning — CS229: L3

Locally Weighted Regression, Probabalistic Interpretation, Logistic Regression

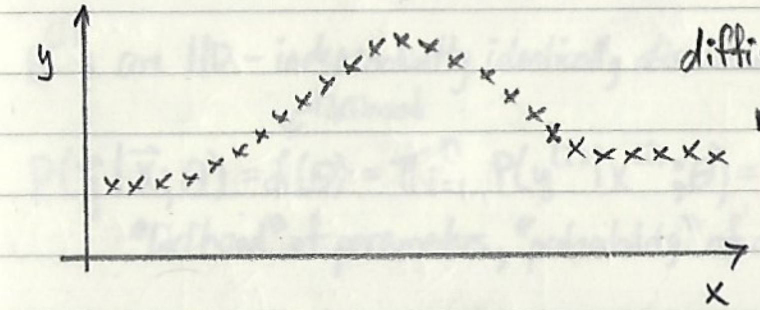
$h_\theta(x) = \sum_{j=0}^{d} \theta_j x_j = \theta^T x$ where $x_0 = 1$ — $(x^{(i)}, y^{(i)})$ i^{th} example

$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$ – quadratic cost — $x^{(i)} \in \mathbb{R}^{d+1}$, $x_0 = 1$

$\theta = (X^T X)^{-1} X^T y$ – closed form — $\theta \in \mathbb{R}^{d+1}$

"Parametric" learning algorithm – θs fixed set of parameters fit to data
↑ linear regression

"Non-parametric" learning algorithm – # of parameters grows (linearly) with n
↑ locally weighted regression (loess)



difficult to define features
not quadratic

to evaluate h at a certain x

LR: fit θ to min. $\sum_i (y^{(i)} - \theta^T x^{(i)})^2$ then return $\theta^T x$

LWR: apply LR to data points only in vicinity of x
- fit θ to min. $\sum_i w^{(i)} (y^{(i)} - \theta^T x^{(i)})^2$ where $w^{(i)} = \exp\left(\frac{-(x^{(i)} - x)^2}{2\tau^2}\right)$
  - ↑ other weight algorithms are possible
  - if $|x^{(i)} - x|$ is small then $w^{(i)} \approx 1$
  - if $|x^{(i)} - x|$ is large then $w^{(i)} \approx 0$
  - τ is "bandwidth" – controls width of bell curve
- need to run new fitting for every prediction – not "building model"
  - ↑ expensive
- can still overfit or underfit

8/21/19

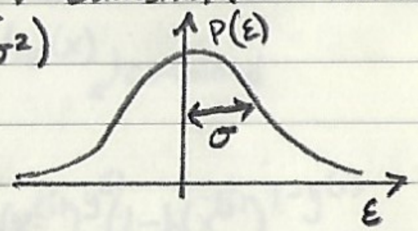
CS229: L3

Probabalistic Interpretation of Least Squares

assume $y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}$, $\varepsilon^{(i)}$ - error, unmodeled effects + noise

$\varepsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$ - Gaussian (central limit theorem)

density $P(\varepsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(\varepsilon^{(i)})^2}{2\sigma^2}\right)$



not random variable

$P(y^{(i)} | x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$

$y^{(i)} | x^{(i)}; \theta \sim \mathcal{N}(\theta^T x, \sigma^2)$

$\varepsilon^{(i)}$s are IID - independently identically distributed (assumption)

likelihood

$P(\vec{y} | \vec{x}; \theta) = \mathcal{L}(\theta) = \prod_{i=1}^n P(y^{(i)} | x^{(i)}; \theta) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$

"liklihood" of parameters, "probability" of data

maximum liklihood: choose θ to max. $\mathcal{L}(\theta) = P(\vec{y} | \vec{x}; \theta)$

log liklihood $\ell(\theta) = \log \mathcal{L}(\theta) = \log \prod \frac{1}{\sigma\sqrt{2\pi}} \exp(\ldots)$

$= \sum_{i=1}^n \log\left[\frac{1}{\sigma\sqrt{2\pi}} \exp(\ldots)\right]$

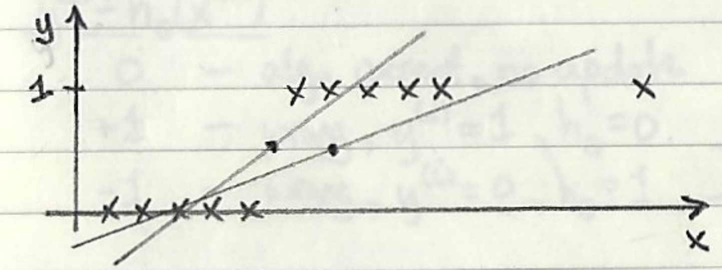
$= n \log \frac{1}{\sigma\sqrt{2\pi}} - \sum_{i=1}^n \frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}$

max. $\ell(\theta)$ same as min. $\frac{1}{2} \sum_{i=1}^n (y^{(i)} - \theta^T x^{(i)})^2 = J(\theta)$

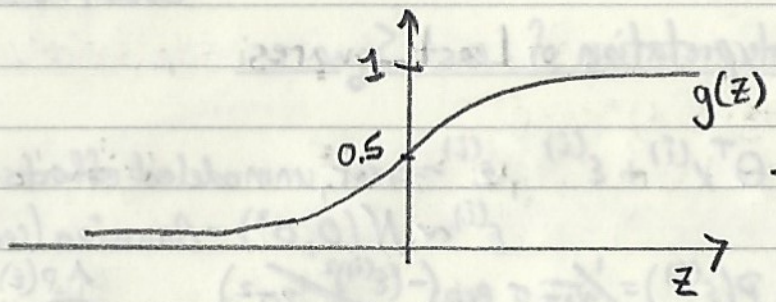
independent of σ^2

Classification

$y \in \{0, 1\}$ - binary



LR generally does not work, although it can sometimes



8/21/19

CS229: L3

Logistic Regression

$y \in \{0,1\}$, $h_\theta(x) \in [0,1]$ ← bound predictions to between 0 and 1

choose: $h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$

↑ sigmoid, $g(z) = \frac{1}{1+e^{-z}}$ - logistic function

$P(y=1|x;\theta) = h_\theta(x)$, $p(y=0|x;\theta) = 1 - h_\theta(x)$ combined

$P(y|x;\theta) = h_\theta(x)^y (1-h_\theta(x))^{1-y}$

$\mathcal{L}(\theta) = P(\vec{y}|\vec{x};\theta) = \prod_i p(y^{(i)}|x^{(i)};\theta) = \prod_i h(x^{(i)})^{y^{(i)}} (1-h(x^{(i)}))^{1-y^{(i)}}$

$\ell(\theta) = \log \mathcal{L}(\theta) = \sum_{i=1}^n y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)}))$

gradient ascent - $\theta := \theta + \alpha \nabla_\theta \ell(\theta)$

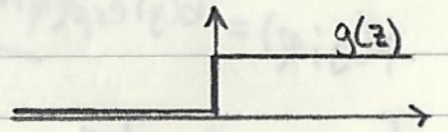
$\frac{\partial}{\partial\theta_j} \ell(\theta) = \sum_{i=1}^n (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$

$\theta_j := \theta_j + \alpha \frac{\partial}{\partial\theta_j} \ell(\theta)$

$\quad := \theta_j + \alpha \sum_{i=1}^n (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$

Perceptron

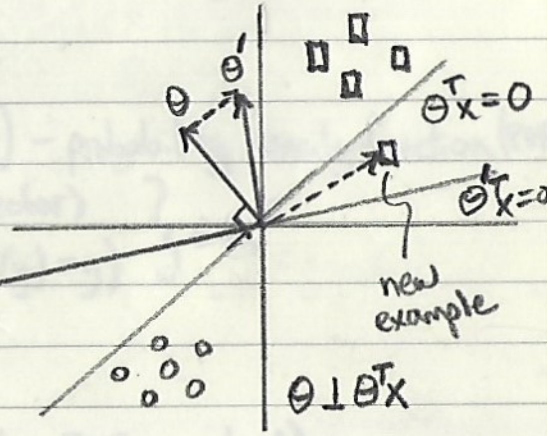
$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$ - step function



$h(x) = g(\theta^T x) = \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise} \end{cases}$

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

$y^{(i)} - h_\theta(x^{(i)})$	
0	- algo correct, no update
+1	- wrong, $y^{(i)} = 1$, $h_\theta = 0$
-1	- wrong, $y^{(i)} = 0$, $h_\theta = 1$



$$p(y;\eta) = \frac{b(y)\exp(\eta^T T(y))}{\exp(a(\eta))}$$

$$\int_y p(y;\eta) = \frac{1}{\exp(a(\eta))}\int_y b(y)\exp(\eta^T T(y))\,dy = 1$$

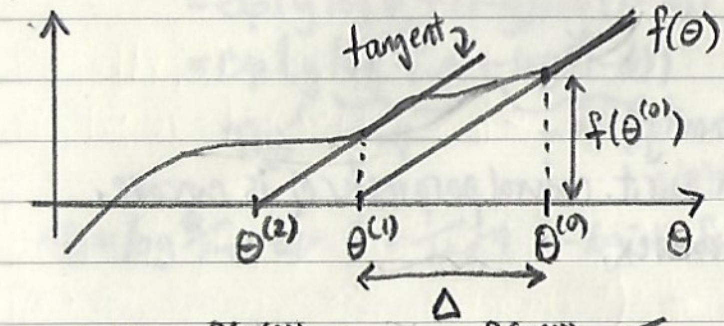
$$\exp(a(\eta)) = \int_y b(y)\exp(\eta^T T(y))\,dy$$

Machine Learning

CS229: L4

Newton's Method, Exponential Families, Generalized Linear Models

Newton's method runs faster than gradient ascent for logistic regression. ← quadratic convergence - very fast



find θ s.t. $f(\theta)=0$

can init $\theta^{(0)}$ to 0

$$f'(\theta^{(0)}) = \frac{f(\theta^{(0)})}{\Delta},\quad \Delta = \frac{f(\theta^{(0)})}{f'(\theta^{(0)})},\quad \theta^{(1)} = \theta^{(0)} - \frac{f(\theta^{(0)})}{f'(\theta^{(0)})}$$

$$\theta^{(t+1)} := \theta^{(t)} - \frac{f(\theta^{(t)})}{f'(\theta^{(t)})}$$ - iterate

$\ell(\theta)$ want θ s.t. $\ell'(\theta)=0 \therefore \theta^{(t+1)} = \theta^{(t)} - \frac{\ell'(\theta^{(t)})}{\ell''(\theta^{(t)})}$

$$\theta^{(t+1)} := \theta^{(t)} - H^{-1}\nabla_\theta \ell(\theta)$$ - θ is a vector

↑ Hessian - $H_{ij} = \frac{\partial^2 \ell(\theta)}{\partial\theta_i \partial\theta_j}$ $\in \mathbb{R}^{(d+1)\times(d+1)}$

H^{-1} expensive to compute with large d

$P(y|x;\theta)$

- $y \in \mathbb{R}$: Gaussian → least squares
- $y \in \{0,1\}$: Bernoulli → logistic regression

Bernoulli(ϕ) - $P(y=1;\phi)=\phi$ } Exponential Family Distributions
$\mathcal{N}(\mu,\sigma^2)$

Exponential Family

$p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))$ - probability density function (PDF)

- η - natural parameter (scalar/vector) } $\in \mathbb{R}^k$
- $T(y)$ - sufficient statistic (usually $T(y)=y$)
- y - data (scalar)
- $b(y)$ - base measure (scalar)
- $a(\eta)$ - log-partition function (makes PDF sum to 1)

$$-\log\left(1-\frac{1}{e^{-\eta}}\right) = -\log\left(\frac{e^{-\eta}}{1+e^{-\eta}}\right)$$
$$= -\log\left(\frac{1}{1+e^{\eta}}\right) = \log(1+e^{\eta})$$

Properties of Exponential Family:

a) max. liklihood estimation w.r.t. natural parameter η is concave.
negative log liklihood is convex.

b) $E[y;\eta] = \frac{\partial}{\partial \eta} a(\eta)$

c) $Var[y;\eta] = \frac{\partial^2}{\partial^2 \eta} a(\eta)$

Real : Gaussian
Binary : Bernoulli
Count : Poisson
$\mathbb{R}^+$: Gamma, Exponential
Dist^n : Beta, Dirichlet

$$x \longrightarrow \underbrace{\boxed{\theta^T x}}_{\text{linear model}} \xrightarrow{\eta} \boxed{\begin{matrix}\text{Exp. Family}\\ b: \\ a: \\ T: \end{matrix}} \longrightarrow E[y;\eta]$$
$$= E[y;\theta^T x] = h_\theta(x)$$

test time

training: $\max_\theta \log p(y^{(i)}; \theta^T x^{(i)})$

learning: $\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$
↖ plug in appropriate $h_\theta(x)$

Bernoulli(ϕ) – $p(y=1;\phi)=\phi$, $p(y=0;\phi)=1-\phi$

$P(y;\phi) = \phi^y(1-\phi)^{1-y}$
$= \exp(\log(\phi^y(1-\phi)^{1-y}))$ – exp and log cancel each other
$= \exp(y\log\phi + (1-y)\log(1-\phi))$
$= \exp(\underbrace{y}_{T(y)} \underbrace{\log\frac{\phi}{1-\phi}}_{\eta} + \underbrace{\log(1-\phi)}_{-a(\eta)})$ $b(y)=1$

$a(\eta) = -\log(1-\phi) = -\log(1-\frac{1}{1+e^{-\eta}}) = \log(1+e^{\eta})$

$\eta = \log\frac{\phi}{1-\phi} \Rightarrow \phi = \frac{1}{1+e^{-\eta}}$ ← logistic function

Gaussian: $N(\mu,\sigma^2)$, set $\sigma^2=1$ since no impact on max. likilihood estimate

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}(y-\mu)^2\right) = p(y;\mu) = \underbrace{\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}y^2\right)}_{b(y)} \exp(\underbrace{\mu}_{\eta} \underbrace{y}_{T(y)} - \underbrace{\tfrac{1}{2}\mu^2}_{-a(\eta)})$$

$a(\eta) = \frac{1}{2}\mu^2$

<u>Generalized Linear Model (GLM)</u>

assume:

1) $y|x;\theta \sim ExpFamily(\eta)$
2) given x, goal is to output $E[y|x;\theta] = h_\theta(x)$
want $h(x) = E[T(y)|x]$
3) $\eta = \theta^T x$ – linear relationship between η and x (design choice)
↖ $\eta_i = \theta_i^T x$ if $\eta \in \mathbb{R}^k$ $\theta \in \mathbb{R}^d$, $x \in \mathbb{R}^d$

Bernoulli: $y|x;\theta \sim ExpFamily(\eta)$
for fixed x, θ, algorithm output $h_\theta(x) = E[y|x;\theta] = p(y=1|x;\theta) = \phi$
$= \frac{1}{1+e^{-\eta}} = \frac{1}{1+e^{-\theta^T x}}$

$g(\eta) = E[y;\eta] = \frac{1}{1+e^{-\eta}}$ – canonical response function
g^{-1} – canonical link function

$\mu = E[y;\eta] = g(\eta) = \frac{\partial}{\partial\eta} a(\eta)$
$\eta = g^{-1}(\mu)$

3 parameters:

1) model 2) natural 3) canonical

$\theta \xrightarrow[\theta^T x]{} \eta \underset{g^{-1}}{\overset{g}{\rightleftarrows}}$ ϕ : Bernoulli, μ : Gaussian, λ : Poisson

logistic regression:

$h_\theta(x) = E[y|x;\theta] = \phi = \frac{1}{1+e^{-\eta}} = \frac{1}{1+e^{-\theta^T x}}$

8/23/19

CS 229: L4

Multinomial: $y \in \{1, \dots, k\}$

parameters: $\phi_1, \phi_2, \dots, \phi_{k-1}, \phi_k = 1-(\phi_1 + \dots + \phi_{k-1})$

$P(y=i) = \phi_i$ ↑redundant parameter

$T(1) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, T(2) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \dots \in \mathbb{R}^{k-1}, T(k-1) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{bmatrix}, T(k) = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$

$1\{\text{True}\} = 1$, $1\{\text{False}\} = 0$ - indicator function

$T(y)_i = 1\{y=i\}$

$P(y) = \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \dots \phi_k^{1\{y=k\}} = \phi_1^{T(y)_1} \phi_2^{T(y)_2} \dots \phi_k^{1-\sum_{j=1}^{k-1} T(y)_j}$

$= b(y) \exp(\eta^T T(y) - a(\eta))$

$\eta = \begin{bmatrix} \log(\phi_1/\phi_k) \\ \vdots \\ \log(\phi_{k-1}/\phi_k) \end{bmatrix} \in \mathbb{R}^{k-1}$, $a(\eta) = -\log(\phi_k)$, $b(y) = 1$

$\rightarrow \phi_i = \frac{e^{\eta_i}}{1+\sum_{j=1}^{k-1} e^{\eta_j}}$ $(i = 1, \dots, k-1)$

$= \frac{e^{\theta_i^T x}}{1+\sum_{j=1}^{k-1} e^{\theta_j^T x}}$ (using $\eta_i = \theta_i^T x$)

$h_\theta(x) = E[T(y)|x;\theta] = E\left[\begin{matrix} 1\{y=1\} \\ \vdots \\ 1\{y=k-1\} \end{matrix} \middle| x;\theta \right] = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_{k-1} \end{bmatrix}$

$= \begin{bmatrix} \frac{e^{\theta_1^T x}}{1+\sum_{j=1}^{k-1} e^{\theta_j^T x}} \\ \vdots \\ \frac{e^{\theta_{k-1}^T x}}{1+\sum_{j=1}^{k-1} e^{\theta_j^T x}} \end{bmatrix}$

Softmax regression (multiclass classification)

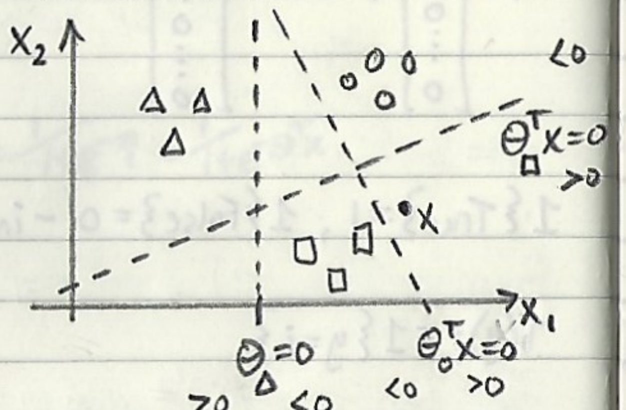
Cross Entropy Minimization

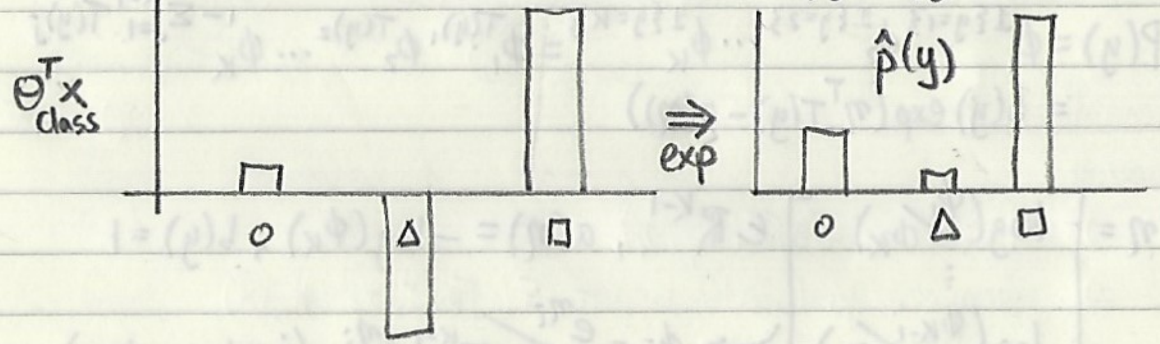
$x^{(i)} \in \mathbb{R}^d$

$y \in \{0,1\}^K$ eg. $(0,0,1,0)$ - one-hot vector

$\theta_{class} \in \mathbb{R}^d$ (K such)

$$\begin{bmatrix} -\theta_1- \\ \vdots \\ -\theta_K- \end{bmatrix}$$





$$\text{cross entropy}(p, \hat{p}) = \sum_{y \in \{o,\Delta,\square\}} p(y) \log \hat{p}(y)$$

$$= -\log \hat{p}(y_o)$$

$$= -\log\left(\frac{e^{\theta_o^T x}}{\sum_{class \in \{o,\Delta,\square\}} e^{\theta_{class}^T x}}\right)$$

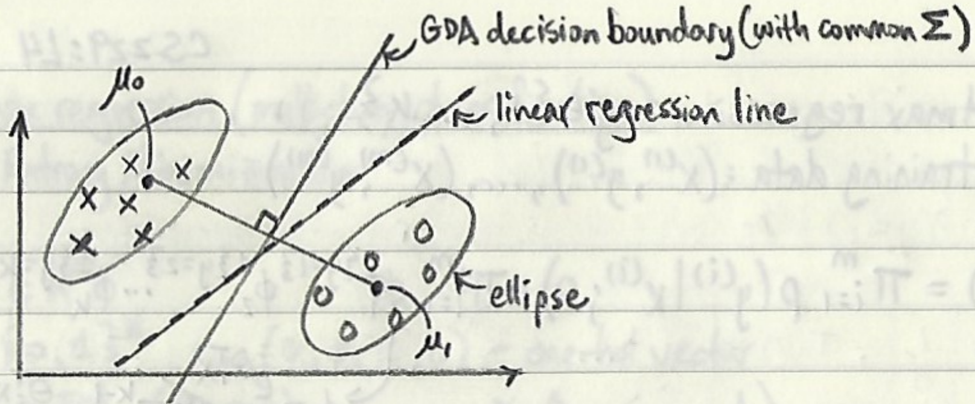
Softmax regression: $y \in \{1, \dots, k\}$

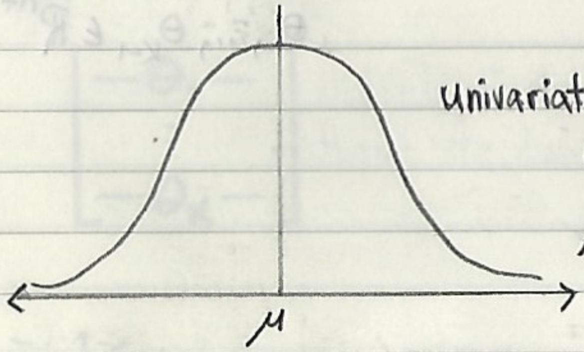
training data: $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$

$$L(\theta) = \prod_{i=1}^m p(y^{(i)} | x^{(i)}; \theta) = \prod_{i=1}^m \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \cdots \phi_k^{1\{y=k\}}$$

$$\phi_1 = \frac{e^{\theta_1^T x}}{1 + \sum_{j=1}^{k-1} e^{\theta_j^T x}}$$

$\theta_1, \dots, \theta_{k-1} \in \mathbb{R}^{n+1}$





univariate Gaussian

$\sigma^2 = E[(X-\mu)^2] = E[X^2] - \mu^2$

$\mu = E[X]$

$E[z] = \vec{\mu}$

$\Sigma = E[zz^T] - (E[z])(E[z])^T$

parameters: $\mu_0, \mu_1, \Sigma, \phi$

$\mu_0, \mu_1 \in \mathbb{R}^d$

$\Sigma \in \mathbb{R}^{d \times d}$

$\phi \in [0,1]$

$\mathcal{L}(\phi, \mu_0, \mu_1, \Sigma) = \prod_{i=1}^n p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma)$

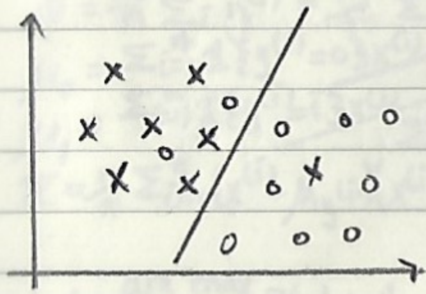
$= \prod_{i=1}^n p(x^{(i)} | y^{(i)}) p(y^{(i)})$

$\mathcal{L}(\Theta) = \prod_{i=1}^n p(y^{(i)} | x^{(i)}; \Theta)$ ← log. reg.

Prof. Andrew Ng — Winter 2003 — 8/24/19
Prof. Moses Charikar — Fall 2019 — 10/7/19

Machine Learning — CS229:L5

Generative Learning Algorithms, Gaussian Discriminative Analysis (GDA)



logistic regression - find straight line to separate two classes

sigmoid $g(\theta^T x)$

discriminative - learns $p(y|x)$ or learns $h_\theta(x) \in \{0,1\}$ directly

generative - $p(x|y)$ (& $p(y)$ - class prior)

features, class label

using Bayes' Rule: $p(y=1|x) = \frac{p(x|y=1)p(y=1)}{p(x)}$

$p(x) = p(x|y=1)p(y=1) + p(x|y=0)p(y=0)$

↑ not necessary since $\arg\max_y p(y|x) = \arg\max_y \frac{p(x|y)p(y)}{p(x)}$

$= \arg\max_y p(x|y)p(y)$

assume $x \in \mathbb{R}^d$, continuous valued

Gaussian Discriminant Analysis: $p(x|y)$ is Gaussian ← assume

multivariate Gaussian: $z \sim \mathcal{N}(\vec{\mu}, \Sigma)$, $\mu \in \mathbb{R}^d$, $\Sigma \in \mathbb{R}^{d \times d}$, $z \in \mathbb{R}^d$

mean, covariance matrix $\Sigma = E[(z-\mu)(z-\mu)^T]$

$\Sigma_{ij} = E[(z_i - \mu_i)(z_j - \mu_j)]$

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

↑ determinant

$P(y) = \phi^y (1-\phi)^{1-y}$ ~ Bernoulli(ϕ) $p(y=1) = \phi$

$P(x|y=0) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_0)^T \Sigma^{-1} (x-\mu_0)\right) \sim \mathcal{N}(\mu_0, \Sigma)$

$p(x|y=1) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)\right) \sim \mathcal{N}(\mu_1, \Sigma)$

← joint likelihood

$\ell(\phi, \mu_0, \mu_1, \Sigma) = \log \prod_{i=1}^n p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma)$

$= \log \prod_{i=1}^n p(x^{(i)} | y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi)$

$\ell(\theta) = \log \prod_{i=1}^n p(y^{(i)} | x^{(i)}; \theta)$ ← conditional likelihood, logistic regression

↑ for comparison

8/25/19

CS229: L5

maximize ℓ w.r.t. $\phi, \mu_0, \mu_1, \Sigma$: no iteration

$\phi = \frac{1}{n} \sum_i y^{(i)} = \frac{1}{n} \sum_i \mathbb{1}\{y^{(i)}=1\}$ ← proportion of examples with $y^{(i)}=1$

$\mu_0 = \frac{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=0\} x^{(i)}}{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=0\}}$ ← # examples with label 0

$\mu_1 = \frac{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\} x^{(i)}}{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\}}$ μ_0 = avg. of all training examples with label 0

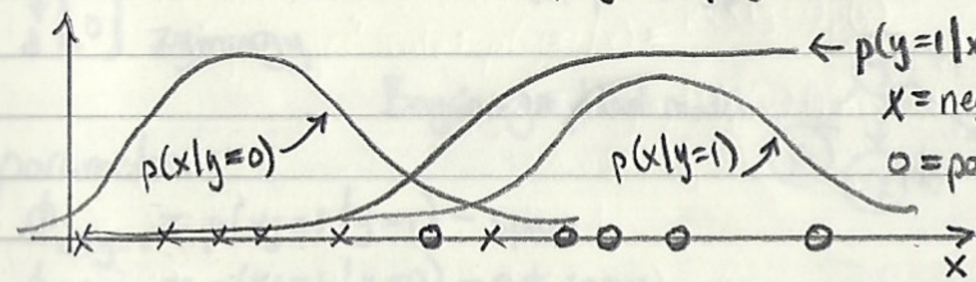
$\Sigma = \frac{1}{n} \sum_{i=1}^{n} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T$

predict: $\arg\max_y P(y|x; \phi, \mu_0, \mu_1, \Sigma) = \arg\max_y \frac{P(x|y)P(y)}{P(x)}$

$= \arg\max_y P(x|y)P(y)$

if $P(y)$ is uniform $= \arg\max_y P(x|y)$

↖ $p(y=0) = p(y=1)$



← $p(y=1|x)$ is sigmoid

x = neg. example

o = pos. example

$p(y=1|x) = \frac{P(x|y=1)p(y=1)}{p(x)}$, $p(y=1) = \phi$

, $p(x) = p(x|y=0)p(y=0) + p(x|y=1)p(y=1)$

↙ stronger assumption

$x|y \sim$ Gaussian $\overset{\Rightarrow}{\not\Leftarrow}$ logistic posterior for $p(y=1|x)$

Logistic Regression assumes ←

$p(y=1|x) = \frac{1}{1+e^{-\theta^T x}}$ $(x_0 = 1)$

$\left.\begin{array}{l} x|y=1 \sim \text{Poisson}(\lambda_1) \\ x|y=0 \sim \text{Poisson}(\lambda_0) \end{array}\right\} \overset{\Rightarrow}{\not\Leftarrow}$ $p(y=1|x)$ is logistic $\overset{\Leftarrow}{\not\Rightarrow} \left\{\begin{array}{l} x|y=1 \sim \text{ExpFamily}(\eta_1) \\ x|y=0 \sim \text{ExpFamily}(\eta_0) \end{array}\right.$

Gaussian Discriminate Analysis makes a stonger assumption (e.g. $x|y \sim$ Gaussian) than logistic regression and so can work with less training data.

Logistic Regression very robust for different distributions

$p(y=1|x; \phi, \mu_0, \mu_1, \Sigma) = \frac{1}{1+\exp(-\theta^T x)}$, θ approx. function of $\phi, \mu_1, \mu_2, \Sigma$

8/25/19

CS229: L5

Naive Bayes: $y \in \{0,1\}$ ← spam email?

$$
\begin{array}{ll}
 & \text{vocabulary} \\
x = \left[\begin{array}{c} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{array}\right] &
\begin{array}{l} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \text{buy} \\ \vdots \\ \text{cs229} \\ \vdots \\ \text{zymurgy} \end{array}
\end{array}
$$

feature vector

d

$x \in \{0,1\}^d$, $d = 50{,}000+$, 2^{50000} possible values of x

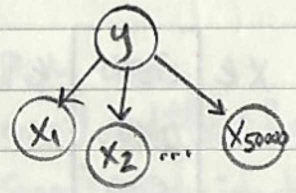
$x_i \in \{0,1\}$

naive assumption: x_i's are conditionally independent given y

↑ extremely strong

$$
\begin{aligned}
p(x_1, x_2, \dots, x_{50000} \mid y) &= p(x_1 \mid y)\, p(x_2 \mid y, x_1)\, p(x_3 \mid y, x_1, x_2) \cdots \\
&= p(x_1 \mid y)\, p(x_2 \mid y)\, p(x_3 \mid y) \cdots p(x_{50000} \mid y) \\
&= \prod_{i=1}^{d} p(x_i \mid y)
\end{aligned}
$$

Baysian graphical model



parameters:

$\phi_{j|y=1} = p(x_j = 1 \mid y=1)$ - spam

$\phi_{j|y=0} = p(x_j = 1 \mid y=0)$ - not spam

$\phi_y = p(y=1)$

training set $\{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$

joint likelihood $\mathcal{L}(\phi_y, \phi_{j|y=1}, \phi_{j|y=0}) = \prod_{i=1}^{n} p(x^{(i)}, y^{(i)})$

$$\phi_{j|y=1} = \frac{\sum_{i=1}^{n} \mathbb{1}\{x_j^{(i)}=1, y^{(i)}=1\} + 1}{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\} + 2}$$

← Laplace smoothing

← fraction of spam emails where word j appears

+2 because binary: spam or not spam

$$\phi_y = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\}$$

prediction time → $p(y=1 \mid x) = \dfrac{p(x \mid y=1)\, p(y=1)}{p(x \mid y=1)\, p(y=1) + p(x \mid y=0)\, p(y=0)}$

$p(x \mid y=1) = \prod_{i=1}^{50000} p(x_j \mid y=1)$ ← zero if any word unseen in $y=1$

(e.g. $p(x_{30000} \mid y=1) = 0$) ← $\phi_{30000|y=1}$

Laplace smoothing:

if $y \in \{1, \dots, k\}$, $P(y=j) = \dfrac{\sum_{j=1}^{n} \mathbb{1}\{y^{(i)}=j\} + 1}{n+k}$

↑ possible values of y

size	<400	400-800	800-1200	1200+
x	1	2	3	4

$$x \in \begin{bmatrix} 3500 \\ 100 \\ \vdots \\ 5000 \\ 9765 \end{bmatrix} \in \mathbb{R}^{d_i}$$

↑ very short email

d_i = length of email

$x_j \in \{1, \dots, 50000\}$

$p(x|y) \overset{\text{assume}}{=} \prod_{j=1}^{d_i} p(x_j|y)$

order of words does not matter (no "j")

parameters:

$\phi_y = P(y=1)$

$\phi_{k|y=0} = P(x_j=k|y=0)$

$\phi_{k|y=1} = P(x_j=k|y=1)$

MLE:

$$\phi_{k|y=0} = \frac{\sum_{i=1}^{n} \left(\underbrace{\mathbb{1}\{y^{(i)}=0\}}_{\text{non-spam}} \sum_{j=1}^{d_i} \underbrace{\mathbb{1}\{x_j^{(i)}=k\}}_{\text{\# times word k appears}}\right)+1}{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=0\} \cdot d_i + \underbrace{50000}_{\text{size of dictionary}}}$$

Prof. Andrew Ng — winter 2003 — 9/1/19

Prof. Moses Charikar — fall 2019 — 10/9/19

Machine Learning

CS229: L6

Naive Bayes, Neural Networks, Support Vector Machines

$$x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix} \begin{matrix} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \\ \\ \end{matrix}$$

n ← vocabulary size

$x_i \in \{0,1\}$ ← multivariate Bernoulli event model

Generative learning algo: $P(x|y) = \prod_{i=1}^{n} P(x_i|y)$ } model

$P(y)$

$\arg\max_y P(y|x) = \arg\max_y P(x|y)P(y)$

if $x_i \in \{1, \dots, k\}$, $P(x|y) = \prod_{i=1}^{n} P(x_i|y)$

↑ discretize continuous variable ↑ multinomial

multinomial event model: some words appear multiple times

$(x_1^{(i)}, x_2^{(i)}, \dots, x_{d_i}^{(i)})$ where d_i = # words in email ← different for every training example

$x_j \in \{1, 2, \dots, 50000\}$ ← index into dictionary

$P(x,y) = \left(\prod_{i=1}^{n} P(x_i|y)\right)P(y)$, d is length of email, $y \in \{0,1\}$

parameters: $\phi_{k|y=1} = P(x_j=k|y=1) = \frac{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\} \sum_{j=1}^{d_i} \mathbb{1}\{x_j^{(i)}=k\} \overset{\text{smoothing}}{+1}}{\sum_{i=1}^{n} \mathbb{1}\{y^{(i)}=1\} d_i + 50000}$

↑ of all spam emails, fraction of words that were word k

$\phi_{k|y=0} = P(x_j=k|y=0)$

$\phi_y = P(y=1)$

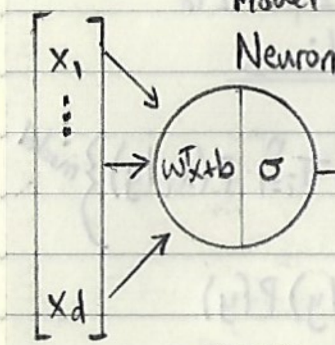
$\ell(\phi_{k|y=1}, \phi_{k|y=0}, \phi_y) = \log \prod_{i=1}^{n} P(x^{(i)}, y^{(i)}; \phi_{k|y=1}, \phi_{k|y=0}, \phi_y)$

$= \log \prod_{i=1}^{n} \prod_{j=1}^{d_i} P(x^{(i)}|y^{(i)}; \phi_{k|y=1}, \phi_{k|y=0}) P(y^{(i)}; \phi_y)$

performs better for text classification because takes into account number of times a word appears in a document

Model ≡ architecture + parameters

Neuron ≡ linear + activation



$\hat{y} = \sigma(\theta^T x)$ ← scalar

$= \sigma(w x + b)$

1×12,288 ↑ ↑ 12,288×1

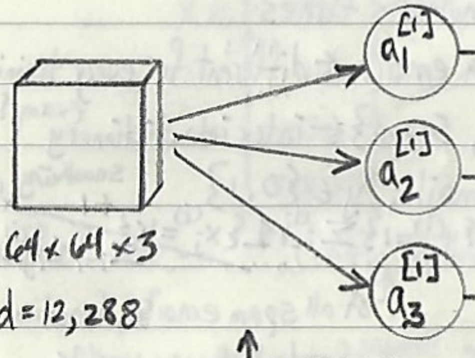
12,289 parameters ↑ (+1 for bias)

↑ 64×64×3 image

1) initialize w, b
2) find optimum w, b
3) use $\hat{y} = \sigma(wx+b)$ to predict

$$\mathcal{L} = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$$ ← log likelihood

$$w = w - \alpha \frac{\partial \mathcal{L}}{\partial w}$$
$$b = b - \alpha \frac{\partial \mathcal{L}}{\partial b}$$



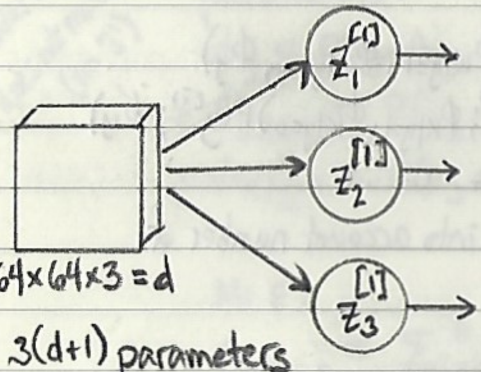
$$\hat{y}_1 = a_1^{[1]} = \sigma(\underbrace{w_1^{[1]} x + b_1^{[1]}}_{z_1^{[1]}})$$
$$\hat{y}_2 = a_2^{[1]} = \sigma(\underbrace{w_2^{[1]} x + b_2^{[1]}}_{z_2^{[1]}})$$
$$\hat{y}_3 = a_3^{[1]} = \sigma(\underbrace{w_3^{[1]} x + b_3^{[1]}}_{z_3^{[1]}})$$

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{array}{l} \text{cat} \\ \text{lion} \\ \text{iguana} \end{array}$$

↑ label

$d = 12{,}288$

↑ 12,289×3 parameters



64×64×3 = d

3(d+1) parameters

$$\frac{e^{z_1^{[1]}}}{\sum_{k=1}^{3} e^{z_k^{[1]}}}$$ softmax

$$\mathcal{L}_{CE} = -\sum_{k=1}^{3} y_k \log \hat{y}_k$$ ← cross-entropy

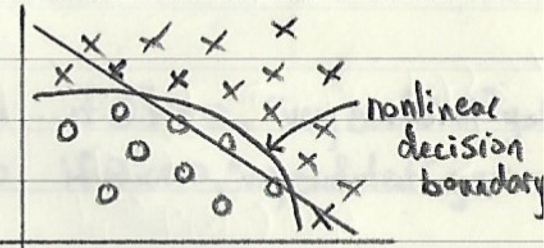
9/1/19

CS229: L6

Nonlinear classifiers

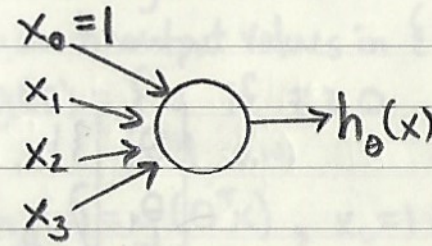
logistic regression: $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

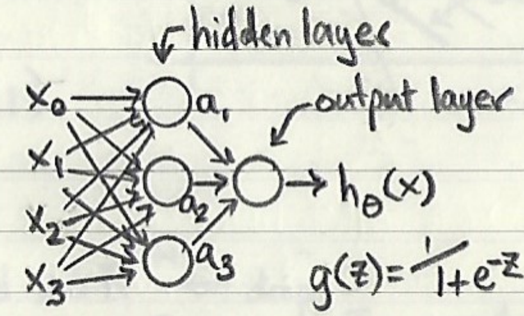
$x|y=1 \sim \text{Expfamily}(\mu_1)$
$x|y=0 \sim \text{Expfamily}(\mu_0)$ } ⇒ logistic posterior



Naive Bayes falls into ExpFamily ∴ using linear classifier

neural network:





$g(z) = \frac{1}{1+e^{-z}}$

$$a_1 = g(x^T\theta^{(1)}),\ a_2 = g(x^T\theta^{(2)}),\ a_3 = g(x^T\theta^{(3)})$$
$$h_\theta(x) = g(\vec{a}^T\theta^{(4)}),\quad \vec{a} = \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

↑ $\theta_1, \dots, \theta_4$

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - h_\theta(x^{(i)}))^2$$ ← quadratic cost function

neural nets not guarenteed to converge to global optimum.

↑ non-convex optimization problem

$$\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \quad \begin{matrix} \} b \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} w \end{matrix}$$

$w^T x + b = \sum_{i=1}^d w_i x_i + b$

want $\hat{\gamma} \gg 0$

Optimal margin classifier: choose w, b to maximize γ

max. margin classifier: $\max_{\gamma, w, b} \gamma$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq \gamma$
$\|w\| = 1$

9/1/19

CS221: L6

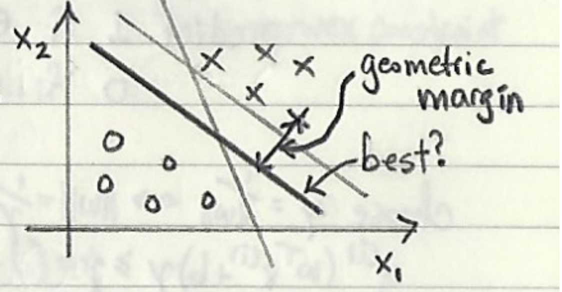
Support Vector Machines (SVM)

$p(y=1|x;\theta)$ and $h_\theta(x) = \theta^T x$

compute $\theta^T x$, predicts 1 iff $\theta^T x \geq 0$ if $\theta^T x \gg 0$, "very confident" $y=1$
0 iff $\theta^T x < 0$ if $\theta^T x \ll 0$, "very confident" $y=0$

nice if $\forall i$ s.t. $y^{(i)} = 1$, have $\theta^T x \gg 0$
$\forall i$ s.t. $y^{(i)} = 0$, have $\theta^T x \ll 0$

assume training set is linearly seperable



notation: $y \in \{-1, 1\}$
have h output values in $\{-1, 1\}$

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{o.w.} \end{cases}$$

$h_\theta(x) = g(\theta^T x)$, $x_0 = 1$ and $x \in \mathbb{R}^{d+1}$ ← drop

$h_{w,b}(x) = g(w^T x + b)$, $b = \theta_0$, $w = [\theta_1 \ldots \theta_n]^T$, $w \in \mathbb{R}^d$, $x \in \mathbb{R}^d$, $b \in \mathbb{R}$

functional margin of hyperplane (w, b) w.r.t. $(x^{(i)}, y^{(i)})$ is

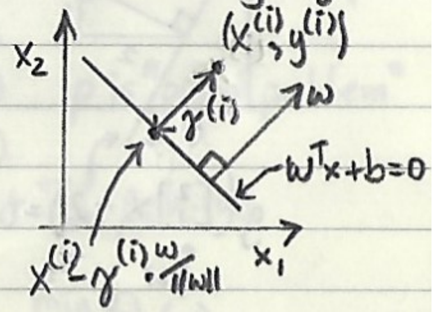
→ $\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b)$ if $y^{(i)} = 1$ want $w^T x^{(i)} + b \gg 0$
if $y^{(i)} = -1$ want $w^T x^{(i)} + b \ll 0$

if $y^{(i)}(w^T x^{(i)} + b) > 0$ then classified $(x^{(i)}, y^{(i)})$ correctly

$\hat{\gamma} = \min_i \hat{\gamma}^{(i)}$ – functional margin of training set is worst case

normalization condition, like $\|w\| = 1$, so $w^T x + b$ is not arbitrarily large

geometric margin – $\gamma^{(i)}$ (no hat)



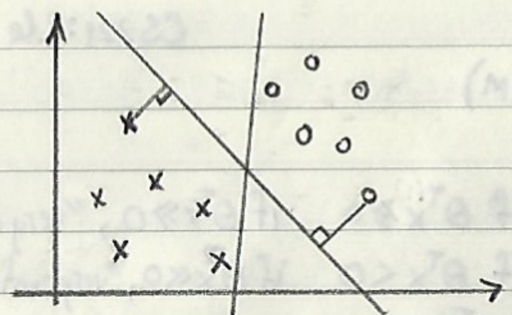
$\frac{w}{\|w\|}$ is unit vector normal to $w^T x + b$

$w^T(x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|}) + b = 0$

$w^T x^{(i)} + b = \gamma^{(i)} \frac{w^T w}{\|w\|} = \gamma^{(i)} \|w\|$

$\gamma^{(i)} = \left[\left(\frac{w}{\|w\|} \right)^T x^{(i)} + \frac{b}{\|w\|} \right] y^{(i)}$ $\gamma^{(i)} = \frac{\hat{\gamma}^{(i)}}{\|w\|}$, $\gamma = \min_i \gamma^{(i)}$

↑ more general to make sign correct



← prefer because points far from boundary

LR - $h_\theta(x) = g(\theta^T x)$

predict 1 if $\theta^T x \geq 0$ want $\theta^T x \gg 0$

0 if $\theta^T x < 0$ want $\theta^T x \ll 0$

choose $\gamma = \frac{1}{\|w\|} \Leftrightarrow \|w\| = \frac{1}{\gamma}$

$y^{(i)}(w^T x^{(i)} + b)\gamma \geq \gamma$

$\Rightarrow y^{(i)}(w^T x^{(i)} + b) \geq 1, \forall i = 1 \dots n$

suppose: $w = \sum_{i=1}^n \alpha_i x^{(i)}$ — Representer Theorem

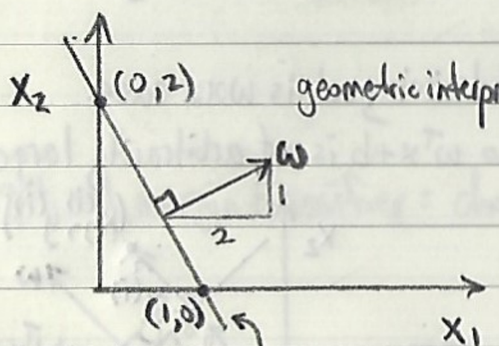
$= \sum_{i=1}^n \alpha_i y^{(i)} x^{(i)}$ ↑ $\{0,1\}$

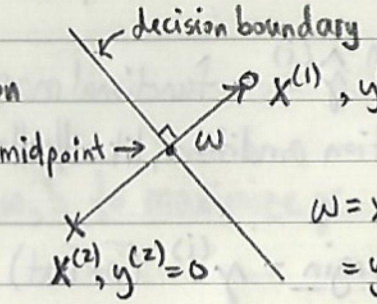
w is a linear combination of examples and lies in "span" of examples

LR - $\theta_0 = 0$

SGD - $\theta_i = \theta - \alpha(h_\theta(x^{(i)} - y^{(i)}))\cdot x^{(i)}$

batch - $\theta_i = \theta - \alpha \sum_{i=1}^n (h_\theta(x^{(i)} - y^{(i)}))x^{(i)}$





$w = x^{(1)} - x^{(2)}$

$= y^{(1)}x^{(1)} + y^{(2)}x^{(2)}$

$g\left(\begin{bmatrix}2\\1\end{bmatrix}x - 2\right) = 0$ decision boundary

↑w ↑b

Prof. Andrew Ng winter 2003 9/5/19

Prof. Moses Charikar fall 2019 10/14/19

Machine Learning CS229: L7

Optimal Margin Classifiers, KKT Conditions, SVM Duals

impose constraint $\|w\| = 1$ or $|w_1| = 1$ or $w_1^2 + |w_2| = 17$

↑ can only choose one and then scale to fit $w^T x + b$

#1 $\max_{\gamma, w, b} \gamma$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq \gamma, \forall i = 1 \dots n$, $\|w\| = 1$ ← constraint

↑ non-convex constraint

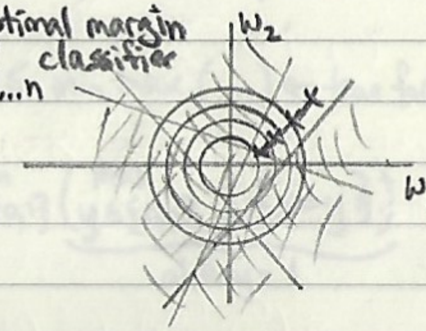
#2 $\max_{\hat\gamma, w, b} \frac{\hat\gamma}{\|w\|}$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq \hat\gamma, \forall i = 1 \dots n$

↑ non-convex objective

impose constraint $\hat\gamma = 1$, $\min_i y^{(i)}(w^T x^{(i)} + b) = 1$

#3 $\min_{w,b} \frac{1}{2}\|w\|^2$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq 1$, $i = 1 \dots n$ ← optimal margin classifier

↑ same as $\max_{w,b} \frac{1}{\|w\|}$



Lagrange multiplier:

$\min_w f(w)$ s.t. $h_i(w) = 0$, $i = 1 \dots l \to h(w) = \begin{bmatrix} h_1(w) \\ \vdots \\ h_l(w) \end{bmatrix} = \vec{0}$

Lagrangian: $\mathcal{L}(w, \beta) = f(w) + \sum_i \beta_i h_i(w)$

↑ Lagrange multipliers

$\frac{\partial \mathcal{L}}{\partial w} \overset{set}{=} 0$, $\frac{\partial \mathcal{L}}{\partial \beta} \overset{set}{=} 0$

for w^* to be a solution, necessary that $\exists \beta^*$ s.t. $\frac{\partial \mathcal{L}(w^*, \beta^*)}{\partial w} = 0$, $\frac{\partial \mathcal{L}(w^*, \beta^*)}{\partial \beta} = 0$

↑ there exists

$\min f(w)$ s.t. $g_i(w) \leq 0$, $i = 1 \dots k$ $(g(w) \leq \vec{0})$ p is "primal problem"

$h_i(w) = 0$, $i = 1 \dots l$ $(h(w) = \vec{0})$

Lagrangian: $\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w)$

define: $\theta_p(w) = \max_{\alpha, \beta} \mathcal{L}(w, \alpha, \beta)$ s.t. $\alpha_i > 0$

consider: $p^* = \min_w \max_{\alpha, \beta} \mathcal{L}(w, \alpha, \beta)$ s.t. $\alpha_i > 0 = \min_w \theta_p(w)$

↑ value of primal problem

$$\min \frac{1}{2}\|w\|^2 = \frac{1}{2}\left(\sum_{i=1}^n \alpha_i y^{(i)} x^{(i)}\right)^T \left(\sum_{j=1}^n \alpha_j y^{(j)} x^{(j)}\right)$$
$$= \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y^{(i)} y^{(j)} \underbrace{x^{(i)T} x^{(j)}}_{\langle x^{(i)}, x^{(j)}\rangle}$$

Constraint:

$$y^{(i)}\left(\Big(\underbrace{\sum_{j=1}^n \alpha_j y^{(j)} x^{(j)}}_{w}\Big) x^{(i)} + b\right) \geq 1$$

$$y^{(i)}\left(\Big(\sum_{j=1}^n \alpha_j y^{(j)} \underbrace{\langle x^{(i)}, x^{(j)}\rangle}_{\text{"Kernel trick"}}\Big) + b\right) \geq 1$$

9/5/19

CS 229: L7

$\Theta_p(\omega)$ if $g_i(\omega) > 0$

then $\Theta_p(\omega) = \infty$

if $h_i(\omega) \neq 0$ then $\Theta_p(\omega) = \infty$

otherwise $\Theta_p(\omega) = f(\omega)$

thus $\Theta_p(\omega) = \begin{cases} f(\omega) & \text{if constraints } g_i, h \text{ are satisfied} \\ +\infty & \text{otherwise} \end{cases}$

so $\min_\omega \Theta_p(\omega) =$ original problem

Dual problem: $\Theta_D(\alpha, \beta) = \min_\omega \mathcal{L}(\omega, \alpha, \beta)$

$$d^* = \max_{\alpha \geq 0, \beta} \min_\omega \mathcal{L}(\omega, \alpha, \beta) = \max_{\alpha \geq 0, \beta} \Theta_D(\alpha, \beta)$$

$d^* \leq p^*$, $\max \min (\ldots) \leq \min \max (\ldots)$ ← true for any function

$$\max_{y \in \{0,1\}} \underbrace{\left(\min_{x \in \{0,1\}} \mathbb{1}\{x = y\}\right)}_{\text{always } 0} \leq \min_{x \in \{0,1\}} \underbrace{\left(\max_{y \in \{0,1\}} \mathbb{1}\{x = y\}\right)}_{\text{always } 1}$$

Sometimes $d^* = p^*$

let f be convex (Hessian is positive semi-definite, $H \geq 0$)

suppose h_i is affine $[h_i(\omega) = a_i^T \omega + b_i]$ (similar to linear)

and constraint g_i are (strictly) feasible $[\exists \omega \text{ s.t. } \forall i \; g_i(\omega) < 0]$

then $\exists\, \omega^*, \alpha^*, \beta^*$ s.t. ω^* solves primal

α^*, β^* solve dual

and $p^* = d^* = \mathcal{L}(\omega^*, \alpha^*, \beta^*)$

$\frac{\partial}{\partial \omega} \mathcal{L}(\omega^*, \alpha^*, \beta^*) = 0$, $\frac{\partial}{\partial \beta} \mathcal{L}(\omega^*, \alpha^*, \beta^*) = 0$

$\alpha_i^* g_i(\omega) = 0$ ← KKT complimentarity condition

$g_i(\omega^*) \leq 0$, $\alpha_i^* \geq 0$

9/5/19

CS 229: L7

Karish-Kuhn-Tucker complementarity condition:

if $\alpha_i > 0 \Rightarrow g_i(w^*) = 0$

usually $\alpha_i^* \neq 0 \Leftrightarrow g_i(w^*) = 0$

$g_i(w)$ is a "active" constraint

Lagrange multipliers: $\alpha_i, \beta_i \rightarrow \alpha_i$

parameters: $w \rightarrow w, b$

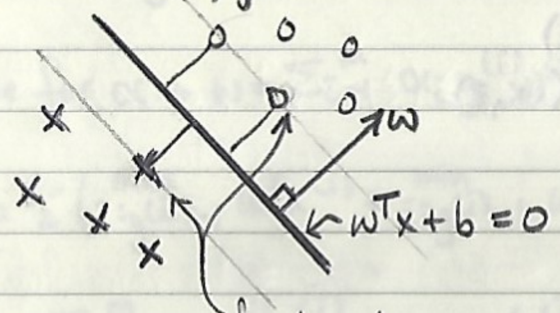
<u>Optimal Margin Classifier:</u>

$\min \frac{1}{2}\|w\|^2$ s.t. $y^{(i)}(w^T x^{(i)} + b) \geq 1$, $i = 1 \dots m$

$\hookrightarrow g(w,b) = -y^{(i)}(w^T x^{(i)} + b) + 1 \leq 0$

$\alpha_i \geq 0 \Rightarrow g_i(w,b) = 0$ "active" constraint

$\Leftrightarrow (x^{(i)}, y^{(i)})$ has functional margin $= 1$



functional margin $= 1$ (usually $\alpha_i \neq 0$)

points called Support Vectors (relatively few)

$\alpha_i = 0$ for non-Support Vectors

$\mathcal{L}(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^m \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1)$ ← no β_i because only inequality constraints

Dual problem: $\Theta_D(\alpha) = \min_{w,b} \mathcal{L}(w,b,\alpha)$

$\nabla_w \mathcal{L}(w,b,\alpha) = w - \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} \stackrel{set}{=} 0 \Rightarrow w = \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}$

$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^m \alpha_i y^{(i)} \stackrel{set}{=} 0$

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_i \sum_j y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \quad \text{s.t. } \alpha_i \geq 0$$
$$\sum y^{(i)} \alpha_i = 0$$

solve for α_i, b

prediction: $h_{\omega,b}(x) = g(\omega^T x + b)$
$$= g((\textstyle\sum_i \alpha_i y^{(i)} x^{(i)})^T x + b)$$
$$= g((\textstyle\sum_i \alpha_i y^{(i)} \langle x^{(i)}, x \rangle) + b)$$

Kernel trick

1) write algo in terms of $\langle x^{(i)}, x^{(j)} \rangle$
   $\uparrow x \quad \uparrow z \quad \langle x, z \rangle$

2) mapping $x \mapsto \phi(x)$
   $\uparrow$ very high dimensional

3) compute $K(x,z) = \phi(x)^T \phi(z)$

4) replace $\langle x, z \rangle$ in algo by $K(x,z)$

9/5/19

CS229: L7

$$\mathcal{L} = \frac{1}{2} \omega^T \omega - \sum_{i=1}^{m} \alpha_i (y^{(i)} (\omega^T x^{(i)} + b) - 1)$$

$$\omega^T \omega = (\textstyle\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)})^T (\sum_{j=1}^{m} \alpha_j y^{(j)} x^{(j)}) \leftarrow \text{from previous}$$

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle - \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle + \sum_{i=1}^{m} \alpha_i$$

$\uparrow$ inner product $- (x^{(i)})^T x^{(j)}$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle = W(\alpha)$$

Dual problem: $\max W(\alpha)$ s.t. $\alpha_i \geq 0$, $\sum_i y_i \alpha_i = 0$.

if $\sum_i y_i \alpha_i \neq 0$ then $\Theta_D(\alpha) = -\infty$
if $\sum_i y_i \alpha_i = 0$ then $\Theta_D(\alpha) = W(\alpha)$

solve for $\alpha \rightarrow \omega = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}$

$$b = \frac{1}{2} \left(\max_{i: y^{(i)} = -1} \omega^T x^{(i)} + \min_{i: y^{(i)} = 1} \omega^T x^{(i)} \right)$$

$\omega = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}$

$h_{\omega,b}(x) = g(\omega^T x + b)$ $\quad (x^{(i)})^T x$

$\omega^T x + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b \leftarrow$ only need to compute for support vectors ($\alpha_i \neq 0$)

Kernels $-$ $x^{(i)}$ very high dimensional ($x^{(i)} \in \mathbb{R}^{\infty}$)
$\langle x^{(i)}, x^{(j)} \rangle$ efficiently computed

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \mathbb{R}^d \quad \phi(x) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \end{bmatrix} \in \mathbb{R}^{d^2} \quad \phi(z) = \begin{bmatrix} z_1z_1 \\ z_1z_2 \\ z_1z_3 \\ z_2z_1 \\ z_2z_2 \\ z_2z_3 \\ z_3z_1 \\ z_3z_2 \\ z_3z_3 \end{bmatrix} \in \mathbb{R}^{d^2}$$

Prof. Andrew Ng — winter 2003 — 9/10/19

Prof. Moses Charikar — fall 2019 — 10/16/19

Machine Learning — CS229: L8

Support Vector Machines, Soft Margin Optimization, Kernels

Convex optimization problem: $\min \frac{1}{2}\|w\|^2$

s.t. $y^{(i)}(w^T x^{(i)} + b) \geq 1,\ i = 1 \ldots m$

assumes data linearly seperable

dual problem: $\max \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y^{(i)} y^{(j)} \alpha_i \alpha_j \underbrace{\langle x^{(i)}, x^{(j)} \rangle}_{(x^{(i)})^T x^{(j)}}$

s.t. $\alpha_i \geq 0,\ \sum_i y_i \alpha_i = 0$

$w = \sum_i \alpha_i y^{(i)} x^{(i)}$, $h_{w,b}(x) = g(w^T x + b)$ ← make prediction

$= g(\sum_i \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b)$

have $x \in \mathbb{R}$ - living area of house

$$x \xrightarrow{\phi} \begin{bmatrix} x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \rightarrow \phi(x)$$, replace $\langle x^{(i)}, x^{(j)} \rangle$ with $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$

↑ high dimensional, possibly infinite

Kernel funtion: $K(x^{(i)}, x^{(j)}) = \langle \phi(x^{(i)}, x^{(j)}) \rangle$ ← replace all inner products

$x, z \in \mathbb{R}^n$

$K(x,z) = (x^T z)^2 = (\sum_{i=1}^d x_i z_i)(\sum_{j=1}^d x_j z_j) = \sum_{i=1}^d \sum_{j=1}^d (x_i x_j)(z_i z_j)$

$= (\phi(x))^T(\phi(z))$

$$\phi(x) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \\ \sqrt{2c}\,x_1 \\ \sqrt{2c}\,x_2 \\ \sqrt{2c}\,x_3 \\ c \end{bmatrix}$$

$O(d^2)$ to compute $\phi(x)$

$O(d)$ to compute $K(x,z)$

$n=3 \rightarrow$ n^2

$\leftarrow K(x,z) = (x^T z + c)^2$

if x, z are similar, $K(x,z) = \phi(x)^T \phi(z)$ is "large"
" dissimilar " " "small"

$$K(x,z) = \exp\left(\frac{-(x-z)^2}{2\sigma^2}\right)$$

$$K(x,x) = \phi(x)^T \phi(x) \geq 0$$

more generally $x^{(1)}, \dots, x^{(t)}$

$$\left(\sum_{i=1}^t z_i x^{(i)}\right)^T \left(\sum_{j=1}^t z_j x^{(j)}\right) \geq 0$$

$$= \sum_{i=1}^t \sum_{j=1}^t z_i z_j K(x^{(i)}, x^{(j)}) \geq 0$$

$$= z^T K z \geq 0 \quad \forall z \in \mathbb{R}^t$$

9/10/19

CS229: L8

$K(x,z) = (x^T z + c)^t \to \binom{n+t}{t}$ features of all monomials up to degree t

$\uparrow$ $n+t$ choose t $\approx (d+t)^t$

$x \mapsto \phi(x)$, $z \mapsto \phi(z)$

$\uparrow$ attributes $\uparrow$ feature vector

$\langle \phi(x), \phi(z) \rangle$

$K(x,z)$ - large if x, z similar

small if x, z dissimilar

$K(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$ - Gaussian Kernel

is K a valid Kernel: $\exists \phi$ s.t. $K(x,z) = \langle \phi(x), \phi(z) \rangle$?

Suppose K is a kernel, let $\{x^{(1)}, \dots, x^{(n)}\}$ be given

let $K \in \mathbb{R}^{n \times d}$, $K_{ij} = K(x^{(i)}, x^{(j)})$

$\uparrow$ kernel matrix

$a^T b = \sum_k a_k b_k$

then for any vector $z \in \mathbb{R}^n$, $z^T K z = \sum_i \sum_j z_i K_{ij} z_j$

$$= \sum_i \sum_j z_i \phi(x^{(i)})^T \phi(x^{(j)}) z_j$$

$$= \sum_i \sum_j z_i \sum_k (\phi(x^{(i)}))_k (\phi(x^{(j)}))_k z_j$$

$$= \sum_k \sum_i \sum_j z_i (\phi(x^{(i)}))_k (\phi(x^{(j)}))_k z_j$$

$$= \sum_k \left(\sum_i z_i \phi(x^{(i)})_k\right)^2 \geq 0$$

Kernel positive semidefinite $\uparrow$

valid

theorem (Mercer): let $K(x,z)$ be given, then K is a valid (Mercer) Kernel (i.e. $\exists \phi$ s.t. $K(x,z) = \phi(x)^T \phi(z)$) if and only if for all $\{x^{(1)}, \dots, x^{(t)}\}$ $(t < \infty)$ the Kernel matrix $K \in \mathbb{R}^{t \times t}$ is symmetric positive semidefinite.

$K(x,x) = -1 \neq \phi(x)^T \phi(x)$ - Inner product of vector with itself is ≥ 0

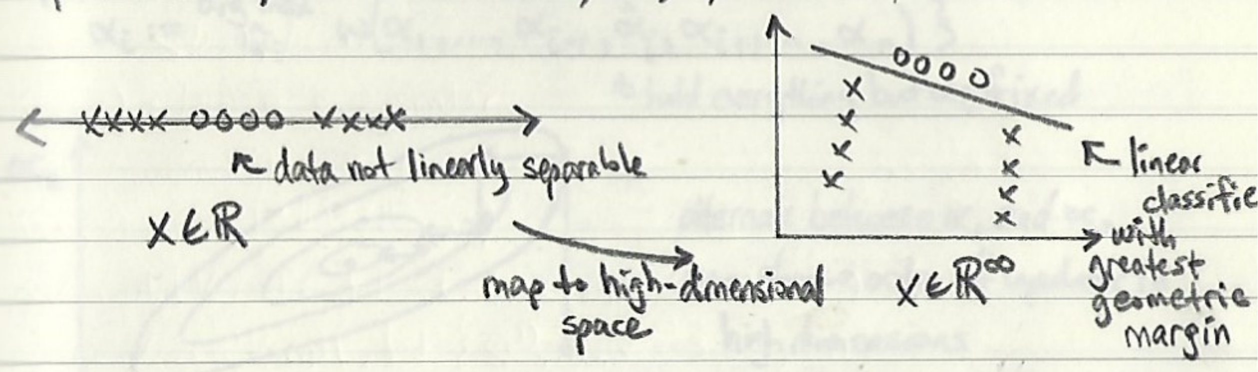
9/10/19

CS229: L8

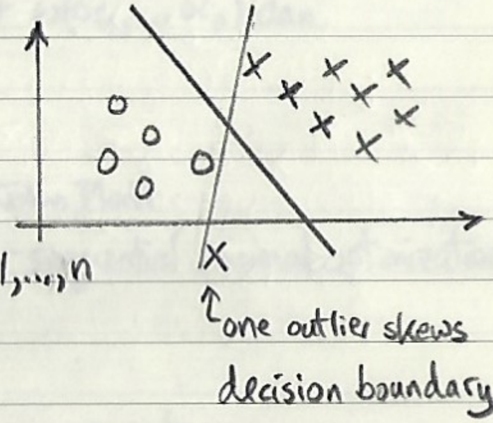
choose $K(x,z) = \exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$ Gausian - ∞ dimensional feature space

or $(x^Tz+c)^d$

replace $\langle x^{(i)}, x^{(j)}\rangle$ with $K(x^{(i)}, x^{(j)})$, $x^{(i)} \to \phi(x^{(i)})$



Kernals much broader than SVMs - most linear algorithms can be rewritten using $\langle x^{(i)}, x^{(j)}\rangle \rightsquigarrow K(x^{(i)}, x^{(j)})$



L_1 norm soft margin SVM:

$$\min \frac{1}{2}\|w\|^2 + \overbrace{C\sum_{i=1}^n \xi_i}^{\text{penalty term}} \quad \leftarrow \text{violations}$$

$$\text{s.t. } y^{(i)}(w^Tx^{(i)}+b) \geq 1 - \xi_i, \ \xi_i \geq 0 \ \forall i=1,\dots,n$$

↑ functional margin

if $y^{(i)}(w^Tx^{(i)}+b) > 0 \Rightarrow$ classified correctly

$$\mathcal{L}(w,b,\xi,\alpha,r) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i\left(y^{(i)}(w^Tx^{(i)}+b) - 1 + \xi_i\right) - \sum_{i=1}^n r_i\xi_i$$

↑ Lagrangian ↑ hyperparameter

$$\max W(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_i\sum_j y^{(i)}y^{(j)}\alpha_i\alpha_j\langle x^{(i)}, x^{(j)}\rangle$$

$$\text{s.t. } \sum_{i=1}^n y^{(i)}\alpha_i = 0, \quad 0 \leq \alpha_i \leq C \ \forall i=1,\dots,n$$

$\alpha_i = 0 \Rightarrow y^{(i)}(w^Tx^{(i)}+b) \geq 1$

$\alpha_i = C \Rightarrow y^{(i)}(w^Tx^{(i)}+b) \leq 1$

$0 < \alpha_i < C \Rightarrow y^{(i)}(w^Tx^{(i)}+b) = 1$

10/20/19

CS229: L8

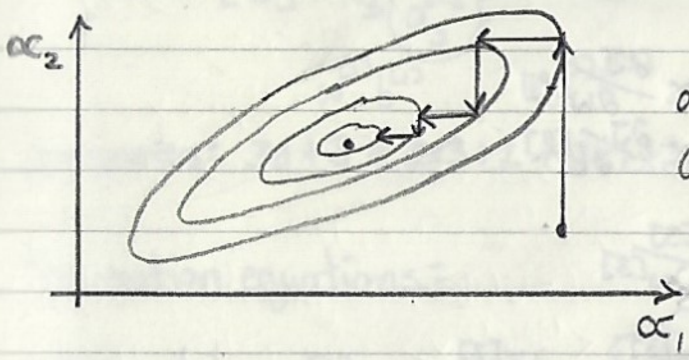
Coordinate Ascent

$\max(\alpha_1, \dots, \alpha_n)$ - no constraints on α_i's

repeat { for $i = 1$ to n

$$\alpha_i := \arg\max_{\hat{\alpha}_i} w(\alpha_1, \dots, \alpha_{i-1}, \hat{\alpha}_i, \alpha_{i+1}, \dots, \alpha_n) \}$$

↑ hold everything but α_i fixed



alternate between α_1 and α_2

can choose order of update in high dimensions

takes more steps than Newton's method, but $w(\alpha_1, \dots \alpha_n)$ can be inexpensive

$\sum_{i=1}^{n} y^{(i)} \alpha_i = 0$ - constraint

chang 2 α_i's at a time

← John Platt

SMO - sequential minimal optimization

select α_i, α_j

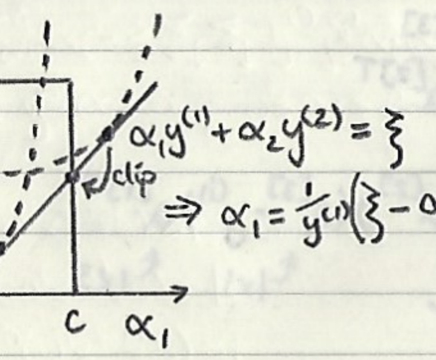
hold all α_k fixed except α_i, α_j

optimize $w(\alpha)$ w.r.t. α_i, α_j s.t. constraints

update: α_1 and α_2 (select 2 α_i on every iteration)

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = -\sum_{i=2}^{n} \alpha_i y^{(i)} = \xi \quad \text{- constraint}$$

$0 \leq \alpha_i \leq C$ - box constraint



$$\Rightarrow \alpha_1 = \frac{1}{y^{(1)}}(\xi - \alpha_2 y^{(2)})$$

$w(\alpha_1, \dots, \alpha_n) =$

$w(\frac{1}{y^{(1)}}(\xi - \alpha_2 y^{(2)}), \alpha_2, \dots) =$

$a\alpha_2^2 + b\alpha_2 + c$

Loss function:

$$J(\hat{y}, y) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}^{(i)}$$

$$\mathcal{L}^{(i)} = -\left[y^{(i)} \log \hat{y}^{(i)} + (1-y^{(i)}) \log (1-\hat{y}^{(i)}) \right]$$

Backward propogation:

$$\forall l = 1, \dots 3 \quad w^{[l]} := w^{[l]} - \alpha \frac{\partial J}{\partial w^{[l]}}$$
$$b^{[l]} := b^{[l]} - \alpha \frac{\partial J}{\partial b^{[l]}}$$

$$\frac{\partial J}{\partial w^{[3]}} = \underbrace{\frac{\partial J}{\partial a^{[3]}} \cdot \frac{\partial a^{[3]}}{\partial z^{[3]}}} \cdot \frac{\partial z^{[3]}}{\partial w^{[3]}}$$

$$\frac{\partial J}{\partial w^{[2]}} = \underbrace{\frac{\partial J}{\partial z^{[3]}} \cdot \frac{\partial z^{[3]}}{\partial a^{[2]}} \cdot \frac{\partial a^{[2]}}{\partial z^{[2]}}} \cdot \frac{\partial z^{[2]}}{\partial w^{[2]}}$$

$$\frac{\partial J}{\partial w^{[1]}} = \frac{\partial J}{\partial z^{[2]}} \cdot \frac{\partial z^{[2]}}{\partial a^{[1]}} \cdot \frac{\partial a^{[1]}}{\partial z^{[1]}} \cdot \frac{\partial z^{[1]}}{\partial w^{[1]}}$$

$$\begin{aligned}
\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[3]}} &= -\left[y^{(i)} \frac{\partial}{\partial w^{[3]}} \log \overbrace{\sigma(w^{[3]} a^{[2]} + b^{[3]})}^{a^{[3]}} + (1+y^{(i)}) \frac{\partial}{\partial w^{[3]}} \log \left(1 - \overbrace{\sigma(w^{[3]} a^{[2]} + b^{[3]})}^{a^{[3]}}\right) \right] \\
&= -\left[y^{(i)} \cdot \frac{1}{a^{[3]}} a^{[3]} (1-a^{[3]}) a^{[2]T} + (1+y^{(i)}) \frac{1}{1-a^{[3]}} (-1) a^{[3]} (1-a^{[3]}) a^{[2]T} \right] \\
&= -\left[y^{(i)} (1-a^{[3]}) a^{[2]T} - (1-y^{(i)}) a^{[3]} a^{[2]T} \right] \\
&= -\left[y^{(i)} a^{[2]T} - a^{[3]} a^{[2]T} \right] \\
&= -(y^{(i)} - a^{[3]}) a^{[2]T}
\end{aligned}$$

$$\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[2]}} = \underbrace{\frac{\partial \mathcal{L}}{\partial a^{[3]}} \cdot \frac{\partial a^{[3]}}{\partial z^{[3]}}}_{\frac{\partial \mathcal{L}}{\partial z^{[3]}}} \cdot \underbrace{\frac{\partial z^{[3]}}{\partial a^{[2]}}}_{w^{[3]T}} \cdot \underbrace{\frac{\partial a^{[2]}}{\partial z^{[2]}}}_{a^{[2]}(1-a^{[2]})} \cdot \underbrace{\frac{\partial z^{[2]}}{\partial w^{[2]}}}_{a^{[1]T}}$$

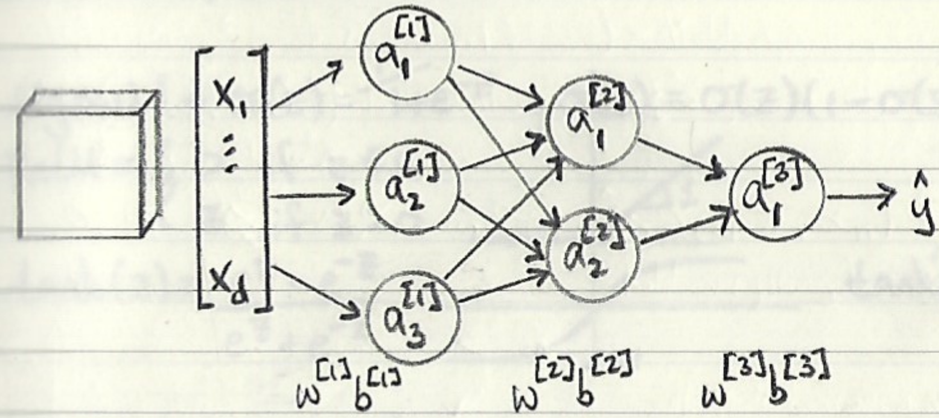
$$\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[3]}} = \underbrace{\frac{\partial \mathcal{L}}{\partial z^{[3]}}}_{-(y^{(i)} - a^{[3]})} \cdot \underbrace{\frac{\partial z^{[3]}}{\partial w^{[3]}}}_{a^{[2]T}}$$

$$\underbrace{\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[2]}}}_{2\times3} = \underbrace{\underbrace{w^{[3]T}}_{2\times1} * \underbrace{a^{[2]}(1-a^{[2]})}_{2\times1}}_{2\times1} \underbrace{(a^{[3]} - y^{(i)})}_{1\times1} \underbrace{a^{[1]T}}_{1\times3}$$

Prof. Moses Charikar 10/21/19

Machine Learning CS229: L9

Neural Networks



parameters: $3d + 3 + 2\times3 + 2 + 2\times1 + 1$

Propogation equations:

$$\begin{aligned}
\underbrace{z^{[1]}}_{3\times1} &= \underbrace{w^{[1]}}_{3\times d} \underbrace{x}_{d\times1} + \underbrace{b^{[1]}}_{3\times1} \\
a^{[1]} &= \sigma(z^{[1]}) \\
z^{[2]} &= w^{[2]} a^{[1]} + b^{[2]} \\
a^{[2]} &= \sigma(z^{[2]}) \\
z^{[3]} &= w^{[3]} a^{[2]} + b^{[3]} \\
\underbrace{a^{[3]}}_{1\times1} &= \sigma(\underbrace{z^{[3]}}_{1\times1})
\end{aligned}$$

examples

$$X = \underbrace{\begin{bmatrix} | & & | \\ x^{(1)} & \cdots & x^{(n)} \\ | & & | \end{bmatrix}}_{} \; d$$

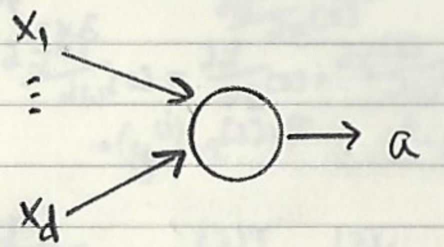
$$z^{[1]} = \begin{bmatrix} | & & | \\ z^{[1](1)} & \cdots & z^{[1](n)} \\ | & & | \end{bmatrix} \; 3 = \#\text{nodes in } [1]$$

$$\underbrace{z^{[1]}}_{3\times n} = \underbrace{w^{[1]}}_{3\times d} \underbrace{x}_{d\times n} + \underbrace{b^{[1]}}_{3\times1}$$

broadcast to $\underbrace{\tilde{b}^{[1]}}_{3\times n}$

$W \approx \sqrt{\frac{1}{d^{[l]}}}$ – works well for sigmoid

$W \approx \sqrt{\frac{2}{d^{[l]}}}$ – " ReLU (He initialization)



$a = \sigma(z)$

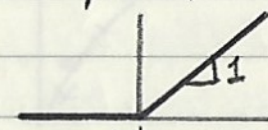
$z = \Sigma\, w_i x_i$

initialize $w_i \approx \frac{1}{d}$

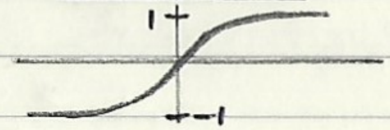
Activation functions:

Sigmoid $= \sigma(z) = \frac{1}{1+e^{-z}}$, $\sigma'(z) = \sigma(z)(1-\sigma(z))$

ReLU $= \begin{cases} 0 & \text{if } z \le 0 \\ z & \text{if } z > 0 \end{cases}$



$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$



$\tanh'(z) = (1-\tanh(z))^2$

Sigmoid and tanh suffer from vanishing gradient problem (not ReLU)

with no activation neural network reduces to linear regression

typically use same activation for each layer

Initialization:

$\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix} = \frac{1}{n} \Sigma_{i=1}^{n} x^{(i)}$ ← center data around origin

$x = x - \mu$

$\Sigma = \frac{1}{n} \Sigma_{i=1}^{n} x^{(i)2}$ ← normalize

$x = \Sigma^{-\frac{1}{2}} x$

compute Σ and μ once for training data and reuse for test data

exploding gradient:

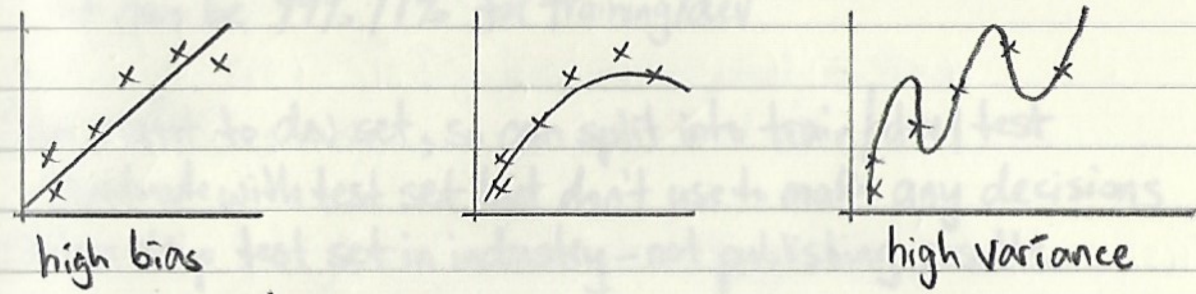
$\hat{y} = w^{[L]} w^{[L-1]} \ldots w^{[1]} \Rightarrow w^{[l]} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}, \hat{y} = \begin{bmatrix} 1.5^L & 0 \\ 0 & 1.5^L \end{bmatrix}$

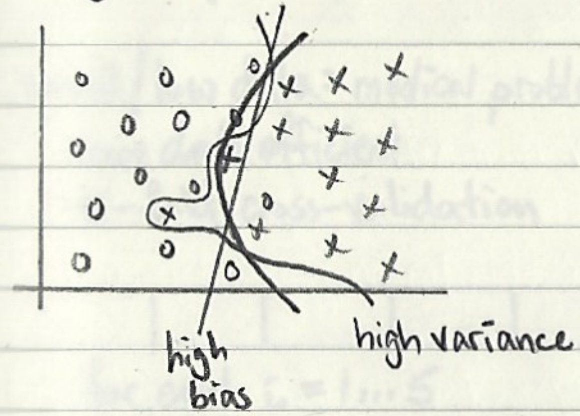
$\hat{y} = \begin{bmatrix} 0.5^L & 0 \\ 0 & 0.5^L \end{bmatrix}$

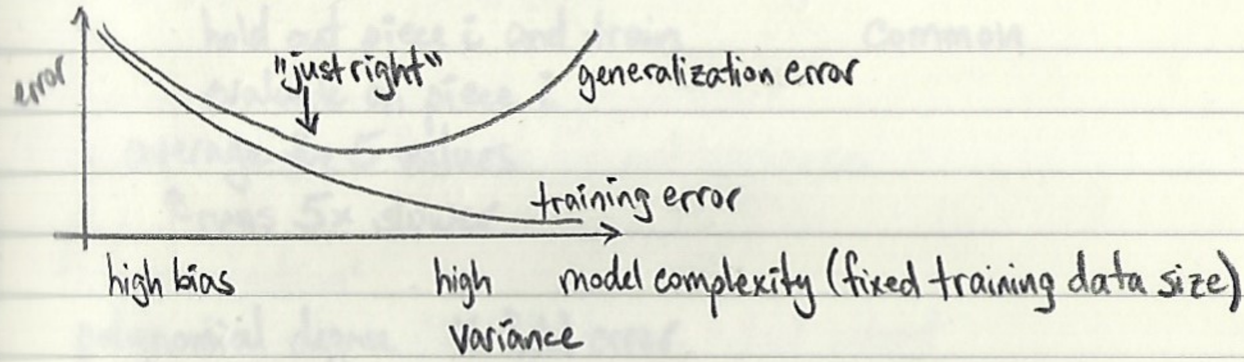
Prof. Andrew Ng 10/23/19

Machine Learning CS229: L10

Bias-Variance, Cross Validation, Model Selection, Regularization







model selection:

$\theta_0 + \theta_1 x$ linear

$\theta_0 + \theta_1 x + \theta_2 x^2$ quadratic

$\theta_0 + \dots + \theta_3 x^3$ cubic

Hold-out cross validation: split data

70%	30%
training	hold-out CV development

- train on training data
- evaluate on dev set
- pick best model on dev set
- retrain on 100% of data (optional)

10/23/19

CS229: L10

With very large data sets, dev % is much smaller
↳ can be 99% / 1% for training/dev

Can overfit to dev set, so can split into train / dev / test
evaluate with test set but don't use to make any decisions
Often skip test set in industry – not publishing results

small / low data: medical problems
more data efficient
K-fold cross-validation

5-fold CV
↑ 10-fold CV is most common

for each $i = 1, \dots, 5$
hold out piece i and train
evaluate on piece i
average the 5 values
↑ runs 5x slower

polynomial degree	K-fold error	
1	3.5	
2	2.7	← best
3	3.8	
4	5.0	

leave-one-out CV: $K = n$
use for tiny (< 100) examples

can use K-fold on training while reserving test for evaluation

CS229: L10

target ~1% error

train 10% error
CV 10.5% error } high bias – does poorly on train and CV

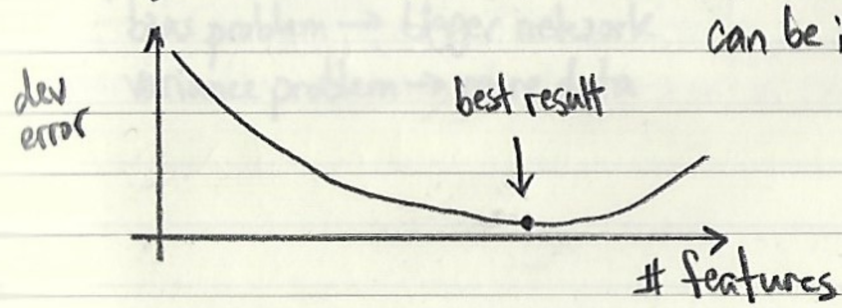
train 1%
CV 10% } high variance – overfitting training data

train 0.5%
CV 0.6% } success!

train 10%
CV 22% } high bias and variance – can happen in very high-dimensional spaces

more data only helps with high variance
more features can fix bias but not variance

Feature selection:
very high-dimensional features – select subset
forward search: start with $\mathcal{F} = \{\}$ (no features)
repeat { try adding each feature to $\mathcal{F}$
train model for each addition
add best choice to $\mathcal{F}$ as measured by CV }
difficult to find features that interact
choose set of features that works best on dev set
can be inefficient



10/23/19

CS229: L10

Backward search: start with all features and subtract
"Wrapper" feature selection
↑ learning algo called as subroutine

"Filter" feature selection:
for each feature i compute a score that tells us how informative x_i is about y
pick top K features
sort features based on $|\mathrm{corr}(x_i, y)|$
mutual information – $p(x_i, y)$
word ↑ ↑ label

	$x_i = 0$	$x_i = 1$
$y = 0$	0.1	0.3
$y = 1$	0.1	0.5

$$MI(x_i, y) = \sum_{x_i \in \{0,1\}} \sum_{y \in \{0,1\}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

$$= \text{difference}\left(p(x,y),\, p(x)p(y)\right)$$
↑ KL divergence

pick top K features – pick "k" based on dev set performance

Regularization: reduces variance

$$\min_\theta \frac{1}{2} \sum_{i=1}^n \|y^{(i)} - \theta^T x^{(i)}\|^2 + \lambda\|\theta\|^2 \quad \text{linear regression}$$

$$\max_\theta \sum_{i=1}^n \log p(y^{(i)} | x^{(i)}; \theta) \underbrace{- \lambda\|\theta\|^2}_{\text{regularization}} \quad \text{logistic regression}$$

penalize large values of θ – smooths out function

deep neural networks diminish bias-variance trade off
bias problem → bigger network
variance problem → more data

Prof. Christopher Ré 10/28/19

Machine Learning CS229: L11

Practical Advice for ML Projects

7 steps overview:

1) acquire data
2) look at data - after every step
3) Create train/dev/test splits
4) Create refine specification
5) build model - simplest that works
6) measurement
7) repeat

cold-start: feature doesn't exist yet
legal/ethical issues to look at data } already incorporating bias

data artifacts are hard - models may pick up unintended signal

"Become one with the data." - Karpathy
have the right people look at the data

train/dev/test - randomly sampled
test is proxy for real world
leakage - (nearly) same example in train and dev
performance overstated

dev for tuning parameters

a good specification has little ambiguity
specification must be embodied is a set of examples - test set!

10/28/19

CS229: L11

inner annotator agreement - how often do they agree?
meaningless accuracy problem
train annotators

subtle problem: consistency in test sets
" : spec creep - needs to be consistent

avoid getting bogged down in models
build simplier models even after fancy models
how accurate is the baseline

Ablation studies - remove one feature at a time from top model

measure end-to-end quality metrics
measure simple things

Slice-based monitoring - overall performance might be
less important than given "slice"
record and scoreboard on slices
monitoring should support fine-grained reporting

avoid unknown mistakes - popularity shift/cold start
- input shift

retrain models periodically
labels and input drift over time

10/28/19

CS229 = L11

a well-running ML system is a rewritten poorly running system

errors in specification:
- class schema issue - two distinguishable classes merged
  - one class split into two
- unknowable class - information not available to model
- unrealized structure
- test set label variance
- change between test set versions

class confusion matrix

"ground truth" is constructed - fix the specification
fix the data

measure error

select more labels - uniform random sampling
importance-based sampling

error bucketing is still critical
add labels to drive model to predict the right class

model diagnostics: try simple models first
linear or logistic regression with simple features

10/28/19

CS229: L11

train vs. test error:
  if error too high, model needs more capacity
  model needs more data or less complexity
  train and test should be close

variance diagnostic: sample data set
  k-fold cross validation

calibration plots: bump means there's a lurking class

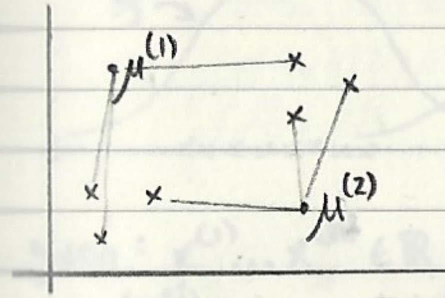
Prof. Christopher Ré 10/30/19

Machine Learning CS229: L12

K-means, GMM, Expectation Maximization

unsupervised learning - no labels
- clustering

given: $x^{(1)} \dots x^{(n)}$, $x^{(i)} \in \mathbb{R}^d$, K = # of clusters
find assignment of $x^{(i)} \rightarrow$ clusters



1) assign points to clusters
2) compute centroids of clusters
$\mu^{(i)} = \arg\min \Sigma \| \mu^{(i)} - x^{(i)} \|$
3) repeat until no changes

randomly select $\mu^{(1)} \dots \mu^{(K)} \in \mathbb{R}^d$

for $i = 1$ to n: ← assign clusters

$$C^{(i)} = \arg\min_{j \in 1 \dots k} \| x^{(i)} - \mu_j \|^2$$

for each $j = 1$ to k: ← compute centroid

$$\mu^{(j)} = \frac{1}{|\Omega_j|} \sum_{i \in \Omega_j} x^{(i)}, \quad \Omega_j = \{ i : c^{(i)} = j \}$$

$$J(C, \mu) = \Sigma_{i=1}^{n} \| x^{(i)} - \mu_{c^{(i)}} \|^2$$

may not find minimum - μ gets stuck on wrong side of cluster
minimization is NP-hard

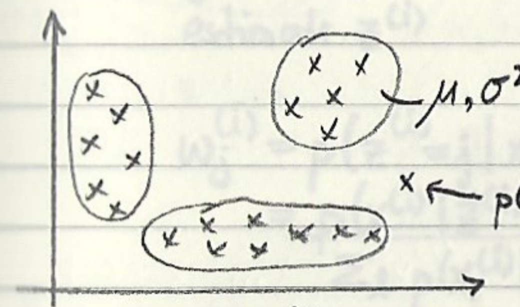
K-means++ (2007) - μ initialization

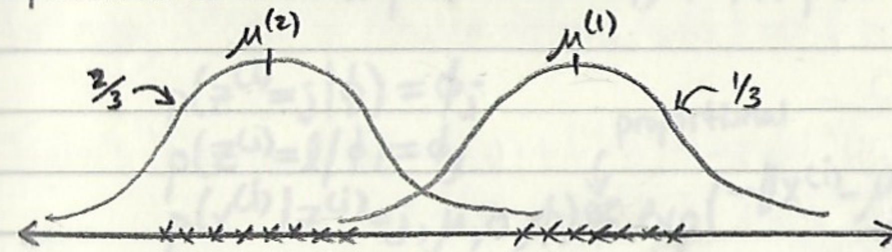
need to understand data to pick K

10/30/19

CS229: L12

Mixture of Gaussians:





given: $x^{(1)} \dots x^{(n)} \in \mathbb{R}$, $K > 0$

$p(z^{(i)} = j)$ - probability $x^{(i)}$ belongs to cluster j

Gaussian Mixture Model (GMM):

$p(x^{(i)}, z^{(i)}) = p(x^{(i)} | z^{(i)})\, p(z^{(i)})$

$z^{(i)} \sim \text{multinomial}(\phi)$ $\sum_{j=1}^{K} \phi_j = 1$, $\phi_j \geq 0$

$x^{(i)} | z^{(i)} = j \sim \mathcal{N}(\mu_j, \sigma_j^2)$

estimate ϕ, μ_j, σ_j^2 , $z^{(i)}$ is hidden / latent - postulate

(e-step) 1. guess values of $z^{(i)}$ for each point

(m-step) 2. update μ_j, σ_j, ϕ parameters

10/30/19

CS229: L12

e-step: given $x^{(1)} \dots x^{(n)}$, ϕ, μ, σ
estimate $z^{(i)}$

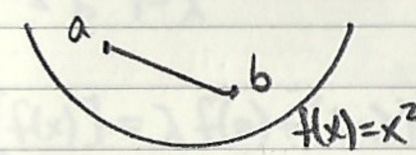
$$w_j^{(i)} = p(z^{(i)}=j \mid x^{(i)}, \mu, \sigma, \phi)$$
$$= \frac{p(x^{(i)} \mid z^{(i)}=j, \mu, \sigma, \phi)\, p(z^{(i)}=j \mid \phi)}{\Sigma_l\, p(x^{(i)} \mid z^{(i)}=l, \mu, \sigma, \phi)\, p(z^{(i)}=l \mid \phi)}$$

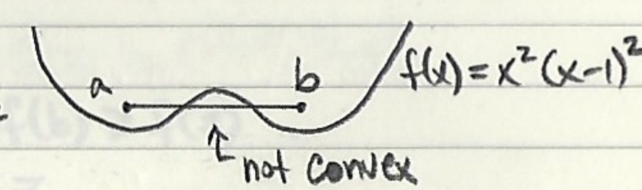
$$p(z^{(i)}=j \mid \phi) = \phi_j$$
$$p(z^{(i)}=l \mid \phi) = \phi_l$$
$$p(x^{(i)} \mid z^{(i)}=j, \mu, \sigma, \phi) \underset{\text{proportional}}{\propto} \exp\left(-\frac{\|x^{(i)}-\mu_j\|^2}{\sigma^2}\right)$$

m-step: given $w_j^{(i)}$, $i=1 \dots n$, $j=1 \dots k$

$$\phi_j = \frac{1}{n} \Sigma_{i=1}^{n} w_j^{(i)} \qquad \mu_j = \frac{\Sigma_{i=1}^{n} w_j^{(i)} x^{(i)}}{\Sigma_{i=1}^{n} w_j^{(i)}}$$

convex analysis: a set Ω is convex if $a, b \in \Omega$ the line between a, b is in Ω.





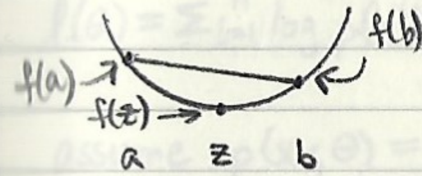
$$\lambda \in [0,1],\ a, b \in \mathbb{R} \Rightarrow \lambda a + (1-\lambda) b \in \Omega$$

given function f, graph of f, $G_f = \{(x,y) : y \geq f(x)\}$

function is convex if its graph is a convex set

10/30/19

CS229: L12



$\lambda(a, f(a)) + (1-\lambda)(b, f(b)) \in \Omega$

$z = \lambda a + (1-\lambda) b$

$\lambda f(a) + (1-\lambda) f(b) \geq f(z)$

for every $f''(x) > 0$ (twice differentiable) $\Rightarrow$ f is convex

Taylor's theorem: $f(a) = f(z) + f'(z)(a-z) + \overbrace{\tfrac{1}{2} f''(z_a)(a-z)^2}^{C_a}$

$f(b) = f(z) + f'(z)(b-z) + \underbrace{\tfrac{1}{2} f''(z_b)(b-z)^2}_{C_b}$

$\lambda f(a) + (1-\lambda) f(b) = f(z) + f'(z)\underbrace{(\lambda a + (1-\lambda) b - z)}_{=0} + \underbrace{C_a + C_b}_{\text{both} > 0}$

$\therefore f(z) < \lambda f(a) + (1-\lambda) f(b)$

Jensen's inequality: $\mathbb{E}[f(x)] \geq f(\mathbb{E}[x])$, x is random, f is convex

$x \to a \quad \lambda$

$\searrow b \quad 1-\lambda$

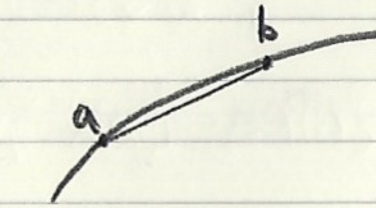
$\mathbb{E}[f(x)] = \lambda f(a) + (1-\lambda) f(b) \geq f(z)$

$\mathbb{E}[x] = \lambda a + (1-\lambda) b = z$

f is concave if $-f$ is convex

$f = \log x$

$f'' = -x^{-2}$



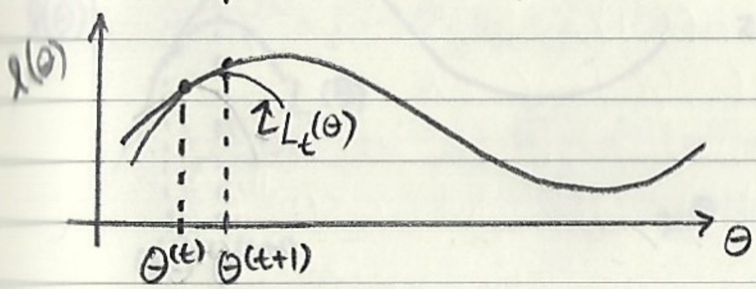
$\mathbb{E}[g(x)] \leq g(\mathbb{E}[x])$

10/30/19

CS229: L12

$\ell(\theta) = \sum_{i=1}^{n} \log p(x^{(i)}; \theta)$

assume $p(x;\theta) = \sum_z p(x, z; \theta)$



maximum of $L_t(\theta)$

$L_t(\theta) \leq \ell(\theta)$
$L_t(\theta^{(t)}) = \ell(\theta^{(t)})$

e-step: find $L_t(\theta^{(t)})$ given $\theta^{(t)}$
m-step: solve $\theta^{(t+1)} = \arg\max_{\theta} L_t(\theta^{(t)})$

$$\log \sum_z p(x, z; \theta) = \log \sum \frac{Q(z)\, p(x, z; \theta)}{Q(z)} \quad \text{s.t. } Q(z) \geq 0,\ \sum_z Q(z) = 1$$

$$= \log \mathbb{E}_Q\left[\frac{p(x, z; \theta)}{Q(z)}\right] \leftarrow \text{concave}$$

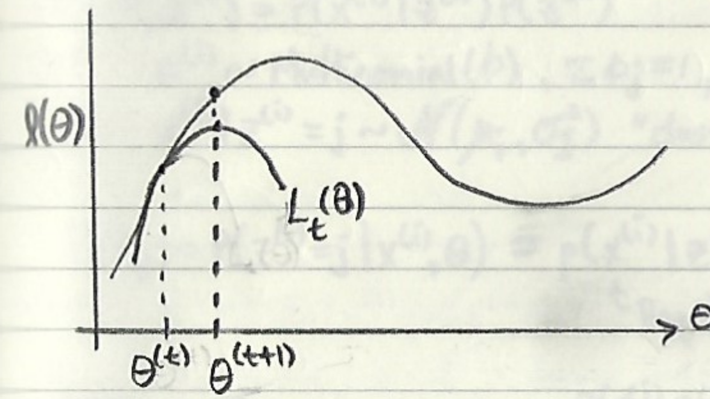
$$= \mathbb{E}_Q\left[\log \frac{p(x, z; \theta)}{Q(z)}\right]$$

$$= \sum_z Q(z) \log \frac{p(x, z; \theta)}{Q(z)}$$

Prof. Christopher Re 11/4/19

Machine Learning CS229: L13

Expectation Maximization, Factor Analysis



1. $L_t(\theta) \leq \ell(\theta)$ lowerbound
2. $L_t(\theta^{(t)}) = \ell(\theta^{(t)})$ tight

$\ell(\theta) = \sum_{i=1}^n \log \sum_z Q_i(z) \frac{P(x^{(i)}, z; \theta)}{Q_i(z)}$ $\quad \sum_z Q_i(z) = 1, \; Q_i(z) > 0$

$L_t(\theta) = \sum_{i=1}^n \sum_z Q_i(z) \log \frac{P(x^{(i)}, z; \theta)}{Q_i(z)}$ ← less than $\ell(\theta) \; \forall \theta$

choose $Q_i(z)$ – may depend on $x^{(i)}, \theta$ (do not want to depend on z)

$\frac{P(x^{(i)}, z; \theta)}{Q_i(z)} = c \Rightarrow \ell(\theta) = \sum_{i=1}^n \log \sum_z Q_i(z) \cdot c = \sum_{i=1}^n \log c$

$L_t(\theta) = \sum_{i=1}^n \sum_z Q_i(z) \log c = \sum_{i=1}^n \log c$

guarentees

$Q_i(z) = P(z \mid x^{(i)}, \theta)$

e-step: for $i = 1, \dots, n$

set $Q_i(z) = P(z \mid x^{(i)}, \theta^{(t)})$

m-step: $\theta^{(t+1)} = \arg\max L_t(\theta)$

terminate?: $\ell(\theta^{(t+1)}) \geq \max_\theta L_t(\theta) \geq L_t(\theta^{(t)}) = \ell(\theta^{(t)})$ – climbing locally not globally optimal

11/4/19

CS229: L13

Mixture of Gaussians:

$P(x^{(i)}, z^{(i)}) = P(x^{(i)} | z^{(i)}) P(z^{(i)})$

$z^{(i)} \sim \text{Multinomial}(\phi)$, $\Sigma \phi_j = 1$, $\phi_j > 0$ $z^{(i)}$ "hidden"

$x^{(i)} | z^{(i)} = j \sim \mathcal{N}(\mu_j, \sigma_j^2)$ "cluster means"

$Q_i(j) = P(z^{(i)} = j | x^{(i)}, \theta) = p(x^{(i)} | z^{(i)} = j, \theta)$ e-step

Bayes's Rule

$\max_{\phi, \mu, \Sigma} \Sigma_{i=1}^n \Sigma_{z^{(i)}} Q_i(z^{(i)}) \log \frac{P(x^{(i)}, z^{(i)}; \theta)}{Q_i(z)}$ m-step

$\Sigma_{i,j} w_j^{(i)} \log \left[\frac{1}{\sqrt{2\pi |\Sigma_j|}} \exp\left(\frac{-1}{2} (x^{(i)} - \mu_j)^T \Sigma_j^{-1} (x^{(i)} - \mu_j) \right) \phi_j / w_j^{(i)} \right]$

$\doteq \Sigma_{i,j} w_j^{(i)} \log \frac{1}{\sqrt{2\pi |\Sigma_j|}} \frac{-1}{2} (x^{(i)} - \mu_j)^T \Sigma_j^{-1} (x^{(i)} - \mu_j) + \log \phi_j - \log w_j^{(i)}$

$\nabla_{\mu_j} L_t = \Sigma_i w_j^{(i)} \Sigma_j^{-1} (x^{(i)} - \mu_j) = \Sigma_j^{-1} (\Sigma_i w_j^{(i)} (x^{(i)} - \mu_j)) = 0$

full rank because can be inverted

$\mu_j = \frac{\Sigma_i w_j^{(i)} x^{(i)}}{\Sigma_i w_j^{(i)}}$

11/4/19

CS 229: L13

Factor Analysis

$n \ll d$ - many fewer points than dimensions

ex: tempeture sensors, $d = 10{,}000$, $n \sim 30$

GMM will not work well

$$\mu = \frac{1}{n} \Sigma x^{(i)} \quad \text{- OK}$$

$$\Sigma = \frac{1}{n} \Sigma (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

$\uparrow$ rank $= d \leq n < d$ - cannot invert Σ and $|\Sigma| = 0$

building block 1: $\Sigma = \sigma^2 I$ - constrain Σ, $\Sigma \in \mathbb{R}^{d \times d}$ $\therefore$ $|\Sigma| = \sigma^{2d}$

$$\max_{\mu} \Sigma_{i=1}^{n} - \log 2\pi |\Sigma|^{1/2} + (x^{(i)} - \mu)^T \Sigma^{-1} (x^{(i)} - \mu)$$

$$= \max_{z} \Sigma_{i=1}^{n} - d \log z + z^{-1} (x^{(i)} - \mu)^T (x^{(i)} - \mu), \quad z = \sigma^2$$

$$z = \frac{1}{nd} \Sigma_{i=1}^{n} (x^{(i)} - \mu)^T (x^{(i)} - \mu)$$

building block 2: $\Sigma = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_d^2 \end{bmatrix}$ - breaks down into d 1-dimensional problems

$$z_j = \frac{1}{n} \Sigma_{i=1}^{n} (x_j^{(i)} - \mu_j)^2$$

model: $P(x, z) = P(x|z) p(z)$, $z \sim \mathcal{N}(0, I) \in \mathbb{R}^s$, $s \ll d$

$$x = \underbrace{\mu}_{\in \mathbb{R}^d} + \overbrace{\Lambda}^{\in \mathbb{R}^{d \times s}} z + \underbrace{\epsilon}_{\mathcal{N}(0, \Phi)}$$

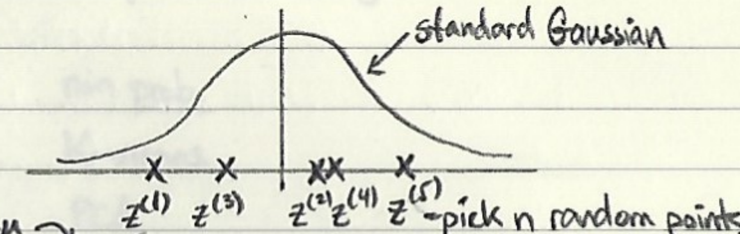
Φ: diagonal

11/4/19

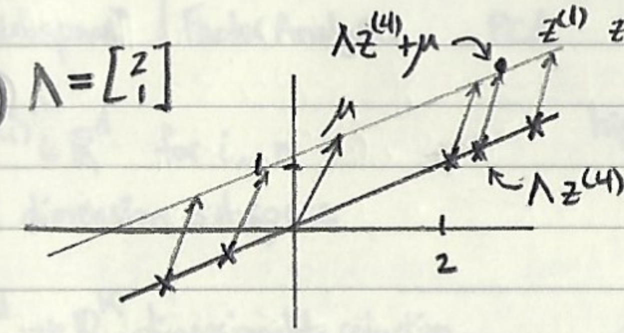
CS:229:L13

$d=2,\ s=1,\ n=5$

1) generate $z^{(1)} \dots z^{(5)}$



2) $\Lambda = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$



$$X = \begin{bmatrix} X_1 \\ \hline X_2 \end{bmatrix} \begin{matrix} \updownarrow d_1 \\ \updownarrow d_2 \end{matrix} \qquad X_1 \in \mathbb{R}^{d_1},\ X_2 \in \mathbb{R}^{d_2},\ X \in \mathbb{R}^{d_1+d_2}$$

$$\Sigma = \begin{array}{c} \overset{d_1}{\longleftrightarrow} \overset{d_2}{\longleftrightarrow} \\ \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \end{array} \begin{matrix} \updownarrow d_1 \\ \updownarrow d_2 \end{matrix} \qquad \Sigma_{ij} \in \mathbb{R}^{d_i \times d_j}$$

$X \sim \mathcal{N}(\mu, \Sigma)$

fact 1: $p(x_1) = \int_{x_2} p(x_1, x_2) = \mathcal{N}(\mu_{11}, \Sigma_{11})$

fact 2: $p(x_1 | x_2) \sim \mathcal{N}(\mu_{1|2}, \Sigma_{1|2})$

$$\mu_{1|2} = \mu_1 + \Sigma_{12}^{-1} \Sigma_{22} (x_2 - \mu_2)$$
$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21}$$

$\begin{pmatrix} z \\ x \end{pmatrix} = \mathcal{N}\left(\begin{pmatrix} 0 \\ \mu \end{pmatrix}, \Sigma\right),\quad \Sigma_{11} = [zz^T] = I$

$$\Sigma_{12} = \mathbb{E}[z(x-\mu)^T] = \mathbb{E}[z(\Lambda z + \epsilon)^T]$$
$$= \mathbb{E}[zz^T]\Lambda^T + \mathbb{E}[z\epsilon^T] = \Lambda^T$$

$$\Sigma_{22} = \mathbb{E}[(x-\mu)(x-\mu)^T] = \mathbb{E}[(\Lambda z + \epsilon)(\Lambda z + \epsilon)^T]$$
$$= \mathbb{E}[\Lambda z z^T \Lambda] + \mathbb{E}[\epsilon \epsilon^T] = \Lambda\Lambda^T + \Phi$$

$$\Sigma = \begin{bmatrix} I & \Lambda^T \\ \Lambda & \Lambda\Lambda^T + \Phi \end{bmatrix}$$

Prof. Christopher Ré 11/6/19

Machine Learning CS229: L14

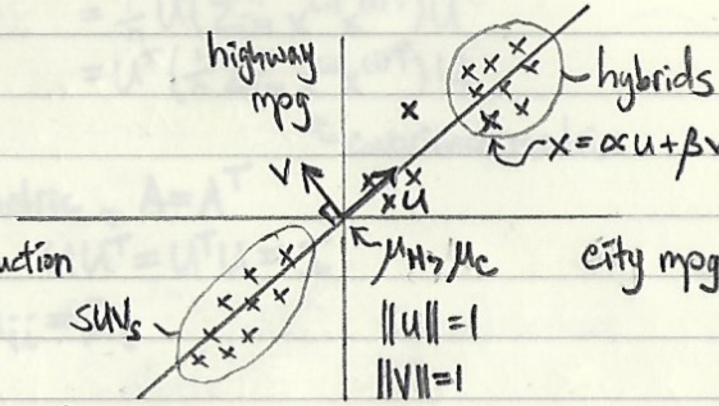
Principal and Independent Component Analysis

structure	probabalistic	non prob.
"cluster"	GMM	K-means
"subspace"	Factor Analysis	PCA

$x^{(i)} \in \mathbb{R}^d$ for $i \ldots n$

K dimension subspace

$\mathbb{R}^d \to \mathbb{R}^K$ dimensionality reduction

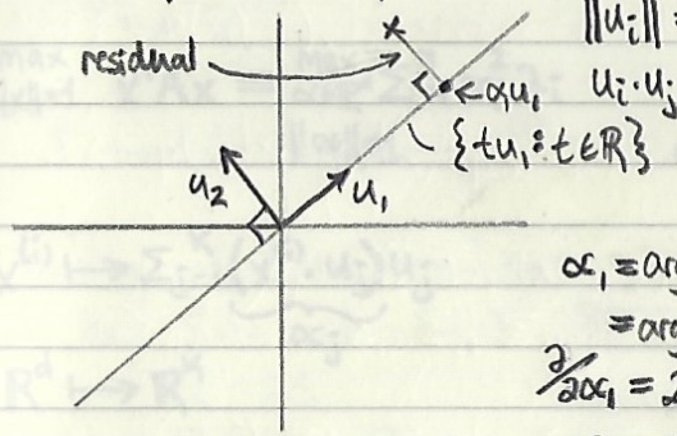


preprocessing - $\mu = \frac{1}{n} \sum_i x^{(i)}$ "centering"

$x^{(i)} \mapsto x^{(i)} - \mu$

data should also be of "same scale"

PCA as optimization problem:



$\|u_i\| = 1$ $u_1, \ldots u_K \in \mathbb{R}^d$

$u_i \cdot u_j = \delta_{ij}$

0 if different, 1 is same

$\alpha_1 = \arg\min_{\alpha} \|x - \alpha u_1\|^2$

$= \arg\min_{\alpha} \alpha \|u_1\|^2 - 2\alpha \langle u_1, x \rangle + \|x\|^2$

$\partial / \partial \alpha_1 = 2\alpha_1 - 2\langle u_1, x \rangle = 0$

$\alpha_1 = u_1 \cdot x$

$$\min_{\alpha \in \mathbb{R}^K} \|x - \sum_{j=1}^K \alpha_j u_j\|^2 = \min_{\alpha \in \mathbb{R}^K} \left[\sum_{j=1}^K \alpha_j^2 - 2\sum_{j=1}^K \alpha_j \langle u_j, x \rangle + \|x\|^2\right]$$

$\alpha_j = u_j \cdot x$

We can find PCA
1) maximize amount in subspace
2) minimize residual

$$\max_{\substack{u \in \mathbb{R}^d \\ \|u\|=1}} \frac{1}{n}\sum_{i=1}^{n} (u \cdot x^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n} (u \cdot x^{(i)})(x^{(i)} \cdot u)$$
$$= \frac{1}{n} u^T \left(\sum_{i=1}^{n} x^{(i)} x^{(i)T}\right) u$$
$$= u^T \left(\frac{1}{n}\sum_{i=1}^{n} x^{(i)} x^{(i)T}\right) u$$

covariance matrix

Let $A \in \mathbb{R}^{d \times d}$ be symmetric, $A = A^T$

$A = U \Lambda U^T$ $\qquad UU^T = U^T U = I$

diagonal $\Lambda_{ii} = \lambda_i$

$$Ax = U\Lambda U^T x \qquad x = \sum \alpha_i u_i$$
$$= U\Lambda \sum_{i=1}^{d} \alpha_i e_i$$

standard unit vectors

$$= U \sum \alpha_i \lambda_i e_i$$
$$= \sum_{i=1}^{d} \alpha_i \lambda_i u_i$$

$$\max_{\|x\|=1} x^T A x \equiv \max_{\substack{\alpha \in \mathbb{R}^d \\ \|\alpha\|=1}} \sum_{i=1}^{n} \alpha_i^2 \lambda_i \qquad \lambda_1 \geq \lambda_2 \ldots \geq \lambda_n \geq 0$$

$\alpha_1 = 1$, o.w. 0

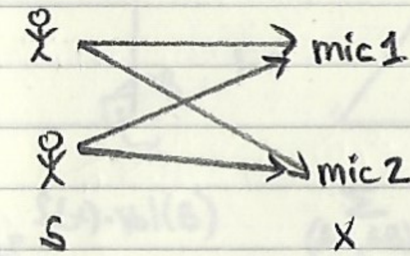
$$x^{(i)} \longmapsto \sum_{j=1}^{K} \underbrace{(x^{(i)} \cdot u_j)}_{\alpha_j} u_j$$

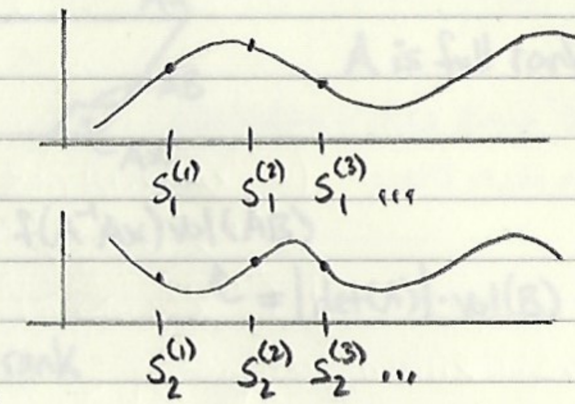
$$\mathbb{R}^d \longmapsto \mathbb{R}^K$$

11/6/19

CS229 : L14

Cocktail party problem:





observed: $x_j^{(t)} = a_{j1} s_1^{(t)} + a_{j2} s_2^{(t)}$

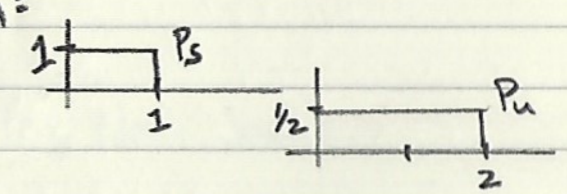
$X^{(t)} = AS^{(t)}$, $A \in \mathbb{R}^{d \times d}$, $S \in \mathbb{R}^d$ ← matrix form
↑ static

given: $X^{(1)} \ldots X^{(n)} \in \mathbb{R}^d$ (d mics)
find: $S^{(1)} \ldots S^{(d)} \in \mathbb{R}^d$ (d speakers)
$A \in \mathbb{R}^{d \times d}$ – mixing matrix, $W = A^{-1}$ – unmixing matrix

$S^{(t)} = A^{-1} X^{(t)}$ $\qquad W = \begin{bmatrix} w_1^T \\ \vdots \\ w_d^T \end{bmatrix}$ $\quad S_j^{(t)} = w_j \cdot X^{(t)}$

S cannot be Gaussian – $S^{(i)} \sim \mathcal{N}(0, I) \therefore X^{(i)} \sim \mathcal{N}(0, AA^T)$
$AA^T = AIA^T = AUU^TA^T$

density under linear transform:
ex: $S \sim \text{uniform}(0,1)$
$u = 2S$
$P_u(x/2) \neq P_S(x)$

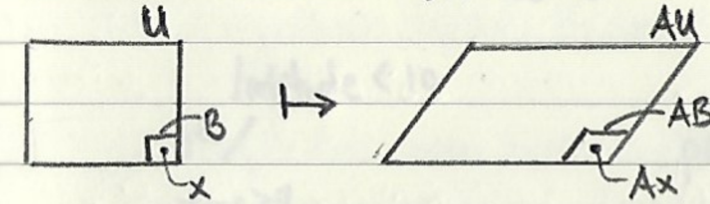


$P_u(x) = P_S(A^{-1}x) \cdot |\det(A^{-1})|$

11/6/19

CS229 = L14

$\int_U f(x)\,dx$ $\qquad$ $\int_{AU} f(A^{-1}y)\,dy = |\det(A)| \cdot \int_U f(x)\,dx$



A is full rank

$\sum_{(x,B)\in P} f(x)\cdot \mathrm{vol}(B)$ $\qquad$ $\sum_{(Ax,AB)\in \hat{P}} f(A^{-1}Ax)\,\mathrm{vol}(AB)$

$= |\det(A)| \cdot \mathrm{vol}(B)$

$\det(A^{-1}) = \frac{1}{\det(A)}$ with A full rank

$P_S(S) = \prod_{i=1}^{d} P_s(S_j)$

$p(x) = \prod_{j=1}^{d} P_s(w_j \cdot x)\,|\det(W)|$

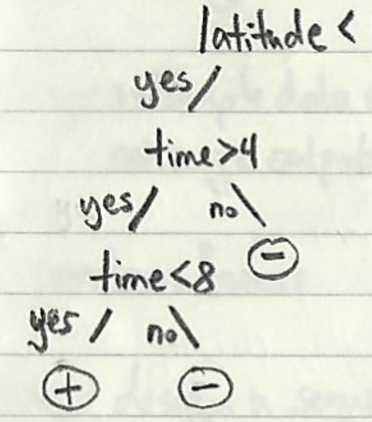
$P_s(x) \propto g'(x)$ for $g(x) = (1+e^{-x})^{-1}$

$\max_{\theta} \log P(x,\theta)$

TAs: Leo Mehr, Atharva Parulekar 11/8/19

Machine Learning CS229: S7

Decision Trees, Boosting, Bagging



prediction – drop down tree until reach node

impurity as a loss function:

- misclassification loss – # examples misclassified with majority prediction

$$L_{misclass}(\hat{p}) = 1 - \max(\hat{p}, 1-\hat{p})$$

- cross-entropy loss

$$L_{cross}(\hat{p}) = -\hat{p}\log\hat{p} - (1-\hat{p})\log(1-\hat{p})$$

when to split leaf node? – decrease average cardinality-weighted loss in children

$$\frac{|R_1| L(R_1) + |R_2| L(R_2)}{|R_1| + |R_2|}$$

handles categorical variables (e.g. North, South)

regularization: max leaf size
max depth

advantages: interpretable
easily to visualize
categorical data

disadvantages: high variance
optimal NP-complete

bias – classifier not expressive enough

variance – final hypothesis depends on sampling of training data

bagging – resample data many times to reduce variance
average outputs of trained models

e.g. random forest

boosting – design a sequential process where weak learners try solving problem. used with low-bias models.

additive modeling –

input: training set $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$ and distribution $p^{(1)}, \ldots, p^{(m)}$
with $\Sigma_{i=1}^{m} p^{(i)} = 1$ and $p^{(i)} \geq 0$

weak classifier: $\phi_j : \mathbb{R}^n \mapsto \{-1, 1\}$ s.t. $\Sigma_{i=1}^{m} p^{(i)} \mathbb{1}\{y^{(i)} \neq \phi_j(x^{(i)})\} \leq \frac{1}{2} - \gamma$

$J(\Theta) = \frac{1}{m} \Sigma_{i=1}^{m} \exp(-y^{(i)} \Theta^T \phi(x^{(i)}))$

$\frac{\partial \mathcal{L}}{\partial \Theta^*} = \Sigma w^+ e^{-\Theta^K} + \Sigma w^- e^{\Theta^K}$

Adaboost – additive modeling with exponential loss

Gradient boost – use gradients to create weak classifiers

Prof. Christopher RÉ 11/11/19

Machine Learning CS229: L15

Weak Supervision

active learning: select points to label more intelligently
semi-supervised learning: use unlabeled data as well
transfer learning: transfer from one dataset to another
weak supervision: label data in cheaper, higher-level ways
↳ programatic supervision – heuristics to label training data
↳ heuristic supervision
distant supervision
data augmentation
other models

ML Applications – model + data + hardware
↑ comes from dirty, messy processes

label quality and quantity much more important than model choice

manual labels – slow, expensive, static (relabel if schema changes)
programmatic labels – fast, cheap, dynamic ← lower quality

Snorkel: formalizing programmatic labeling
↑ replace ad hoc weak supervision

problem: noisy soures conflict and are correlated
↳ model and combine noisy labels into probabilities
↳ use probabalistic labels to train a model

Key idea: observe overlapping judgements on many points

11/11/19

CS229: L15

given: $x^{(1)} \dots x^{(n)} \in \mathbb{R}^d$

$\lambda_1 \dots \lambda_m$, $\lambda_i : \mathbb{R}^d \to \{-1, 1\}$ ← labeling functions

← conditionally independent

find: $p(y^{(i)} = 1 \mid \bar{\lambda}, x^{(i)})$

↑ all labeling functions

model: each labeler has accuracy α_i (hidden)

with probability α_i, $\lambda_i(x) = y$ ← correct

" " $1 - \alpha_i$, $\lambda_i(x) = -y$ ← incorrect

$$\mathbb{E}[\lambda_i y] = \underbrace{\alpha_i 1}_{\text{agree}} + \underbrace{(1-\alpha_i)(-1)}_{\text{disagree}} = 2\alpha_i - 1 = a_i$$

$$\mathbb{E}[\lambda_i \lambda_j] = \mathbb{E}[\lambda_i Y \lambda_j Y] = \begin{cases} a_i a_j & i \neq j \\ 1 & \text{o.w.} \end{cases}$$

$Y^2 = 1$

$$M_{ij} = \mathbb{E}[\lambda_i \lambda_j] \leftarrow \text{observe}$$

$$\hat{M}_{ij} = \Sigma_{i=1}^{n} \lambda_j(x^{(i)}) \lambda_k(x^{(i)})$$

i, j, k distinct indexes – $\dfrac{M_{ij} M_{jk}}{M_{ik}} = \dfrac{(a_i a_j)(a_j a_k)}{a_i a_k} = a_j^2$

↑ 3 distinct labelers "vote"

Knowing sign of a_1 enabels knowing sign for all a_j

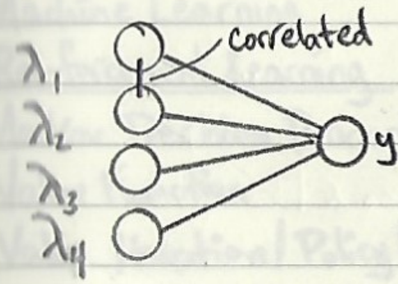
$a_i = 0$ means accuracy is ½ ← random

gradient descent:

$$\min_a \left\| M - \left(I + \begin{bmatrix} a_1^2 & & \\ & \ddots & \\ & & a_n^2 \end{bmatrix} + a a^T\right) \right\|_F^2$$

11/11/19

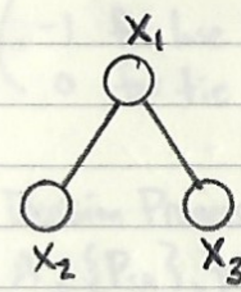
CS229: L15



$$\mathbb{E}[\lambda_i\lambda_j|Y] = \mathbb{E}[\lambda_i|Y]\,\mathbb{E}[\lambda_j|Y] \qquad (i,j)\notin E$$

← Graphical model

use of inverse co-variance:



$x_1 \sim \mathcal{N}(0,1)$ $\quad \mathbb{E}[x_1]=0$

$x_2 \sim \mathcal{N}(x_1,1)$ $\quad x_2 = x_1 + \epsilon_2$

$x_3 \sim \mathcal{N}(x_1,1)$ $\quad x_3 = x_1 + \epsilon_3 \to \mathcal{N}(0,1)$

$\mathbb{E}[x_1^2] = 1$

$\mathbb{E}[x_2^2] = \mathbb{E}[(x_1+\epsilon_2)^2] = 2$

$\mathbb{E}[x_1x_2] = \mathbb{E}[x_1^2 + \cancel{x_1\epsilon_2}^{\,0}] = 1$

$\mathbb{E}[x_2x_3] = \mathbb{E}[(x_1+\epsilon_2)(x_1+\epsilon_3)] = 1$

$$\Sigma = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \quad \Sigma^{-1} = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

zeros where no edges present ↑

We say a $p: \mathbb{R}^d \to [0,1]$ agrees with graph $G = (\{1\ldots d\}, E)$ if $p(x) = \prod p_{u,v}(x_u,x_v) \cdot \prod p_v(x_v)$

↑ edges

Prof. Moses Charikar 11/13/19

Machine Learning CS229: L16

Reinforcement Learning

Markov Decision Processes - MDPs

Value Function

Value Iteration / Policy Iteration

$$R(s) = \begin{cases} +1 & \text{for win} \\ -1 & \text{for lose} \\ 0 & \text{for tie} \end{cases}$$ Reward as function of state

Markov Decision Process:

$(S, A, \{P_{SA}\}, \gamma, R)$

S: set of states

A: set of actions

P_{SA}: state transition probability function, $\sum_{s'} P_{sa}(s') = 1$

γ: discount factor $\in [0,1)$

R: rewards

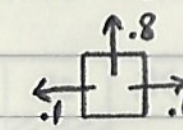
terminal states

→	→	→	+1
↑		↑	-1
↑	←	←	←

optimal policy

11 states

Actions: $\{N, S, E, W\}$



$R((4,3)) = +1$

$R((4,2)) = -1$

$R(s) = -0.02$ for all other s

incentive to get to target

$P_{(3,1)N} = ((3,2)) = 0.8$

$P_{(3,1)N} = ((4,1)) = 0.1$

$P_{(3,1)N} = ((2,1)) = 0.1$

start s_0
choose action a_0
get to $s_1 \sim P_{s_0 a_0}(s_1)$
choose action a_1
get to $s_2 \sim P_{s_1 a_1}(s_2)$

total payoff:
$R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots$ discounting future rewards
↳ required for convergence

goal: choose actions over time to maximize reward
$\mathbb{E}[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots]$

policy $\pi : S \mapsto A$
↑ controller

define $V^\pi : S \mapsto \mathbb{R}$ for policy π
$V^\pi(s)$ is expected total payoff for starting in s and executing policy π – value function
$V^\pi(s) = \mathbb{E}[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots \mid \pi, s_0 = s]$

π

→	→	→	+1
↓	▨	→	-1
→	→	↑	↑

V^π

.52	.73	.77	+1
-.90	▨	-.82	-1
-.88	-.87	-.85	-1.00

Bellman's Equations:

$$V^{\pi}(s) = R(s) + \gamma \underbrace{\sum_{s'} P_{s,\pi(s)}(s') V^{\pi}(s')}_{\text{future reward}}$$

$$\begin{aligned} V^{\pi}(s) &= \mathbb{E}\left[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \dots \mid \pi, s_0 = s\right] \\ &= R(s) + \gamma\, \mathbb{E}\left[R(s_1) + \gamma R(s_2) + \dots \mid \pi, s_0 = s\right] \\ &= R(s) + \gamma\, \mathbb{E}\left[V^{\pi}(s_1) \mid \pi, s_0 = s\right] \\ &= R(s) + \gamma \sum_{s'} P_{s,\pi(s)}(s') V^{\pi}(s') \end{aligned}$$

$$V^{\pi}(s) = R(s) + \gamma \sum_{s'} P_{s,\pi(s)}(s') V^{\pi}(s')$$

$$V^{\pi}((3,1)) = R((3,1)) + \gamma \big(0.8\, V^{\pi}((3,2)) + 0.1\, V^{\pi}((4,1)) + 0.1\, V^{\pi}((2,1))\big)$$

$$\begin{bmatrix} V^{\pi}((1,1)) \\ V^{\pi}((2,1)) \\ \vdots \\ V^{\pi}((4,3)) \end{bmatrix} \in \mathbb{R}^{11}$$

V^*: optimum value function

$$V^*(s) = \max_{\pi} V^{\pi}(s)$$

Bellman's equation:

$$V^*(s) = R(s) + \max_{a \in A} \gamma \sum_{s'} P_{sa}(s') V^*(s')$$

11/13/19

CS221: L16

$\pi^*(s) = \arg\max_{a \in A} \sum_{s'} P_{sa}(s') V^*(s')$: optimum policy function

$V^*(s) = V^{\pi^*}(s) \geq V^{\pi}(s)$

Value iteration: V converges to V^* (gets arbitrarily close)

initialize $V(s) = 0 \ \forall s$ (for all s)

$\forall s$, update:

$V(s) := R(s) + \max_a \gamma \sum_{s'} P_{sa}(s') V(s')$

Synchronous update: update all values of V simultaneously

asynchronous update: update one V at a time

Bellman backup operator: $V := B(V)$

.86	.90	.93	+1
.81		.69	-1
.78	.75	.71	.49

Policy iteration: converges to π^* (get to π^* but requires more computation of Bellman's equation)

initialize π randomly

repeat

set $V = V^{\pi}$ (solve Bellman's equations)

set $\pi(s) = \arg\max_{a \in A} \sum_{s'} P_{sa}(s') V(s')$

Linear Algebra Review

$C = AB \in \mathbb{R}^{m\times p}$, $C_{ij} = \sum_{k=1}^n A_{ik}B_{kj}$

$x^Ty \in \mathbb{R} = \sum_{i=1}^n x_i y_i$

$(AB)C = A(BC)$, $A(B+C) = AB + AC$

$AI = A = IA$

$(A^T)^T = A$, $(AB)^T = B^TA^T$, $(A+B)^T = A^TB^T$

symmetric: $A = A^T$, $A \in \mathbb{R}^{n\times n}$

for any $A \in \mathbb{R}^{n\times n}$, $A + A^T$ is symmetric

$\hookrightarrow A = \frac{1}{2}(A+A^T) + \frac{1}{2}(A - A^T)$

$tr(A) = \sum_{i=1}^n A_{ii}$

$A \in \mathbb{R}^{n\times n}$, $trA = trA^T$

$A, B \in \mathbb{R}^{n\times n}$, $tr(A+B) = trA + trB$

if AB is square, $trAB = trBA$

if ABC is square, $tr\,ABC = tr\,BCA = tr\,CAB$

$\nabla_{A^T} f(A) = (\nabla_A f(A))^T$

$\nabla_A tr AB = B^T$

$\nabla_A tr\,ABA^TC = CAB + C^TAB^T$

$\nabla_A |A| = |A|(A^{-1})^T$ – non-singular square A

$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$, $\|x\|_2^2 = x^Tx$, $\|x\|_1 = \sum_{i=1}^n |x_i|$

$A \in \mathbb{R}^{m\times n}$ – col rank = row rank = rank

rank$(A) \le \min(m,n)$, rank$(A) = \min(m,n)$ full rank

rank(A) = rank(A^T)

rank(AB) = min(rank(A), rank(B))

$A^{-1}A = I = AA^{-1}$, $A \in \mathbb{R}^{n\times n}$

$(A^{-1})^{-1} = A$, $(AB)^{-1} = B^{-1}A^{-1}$, $(A^{-1})^T = (A^T)^{-1} = A^{-T}$

$\llcorner x^Ty = 0$ – orthogonal

$U \in \mathbb{R}^{n\times n}$ – orthogonal if all columns orthogonal and normalized

$U^TU = I = UU^T$, $\|Ux\|_2 = \|x\|_2$ $x \in \mathbb{R}^n$

span of $\{x_1 .. x_n\}$ = set of all vectors that are liner comb.

range(A), $A \in \mathbb{R}^{m\times n}$ = span of columns of A

proj$(y; A) = A(A^TA)^{-1}A^Ty$

$|I| = 1$, $A \in \mathbb{R}^{n\times n}$ mult. single row by t = $t|A|$

switch rows = $-|A|$

$|A| = |A^T|$, $|AB| = |A||B|$, $|A| = 0$ iff A singular

$|A^{-1}| = 1/|A|$

PD – symmetric and $x^TAx > 0$ – full rank, invertible

PSD – symmetric and $x^TAx \ge 0$

$Ax = \lambda x$, $x \ne 0$ $\lambda \in \mathbb{C}$ eigenvalue, $x \in \mathbb{C}^n$ eigenvector

$(\lambda I - A)x = 0$, $|(\lambda I - A)| = 0$

$trA = \sum_{i=1}^n \lambda_i$, $|A| = \prod_{i=1}^n \lambda_i$

Hessian $\nabla_x^2 f(x) \in \mathbb{R}^{n\times n} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots \\ \vdots & & \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & & \end{bmatrix}$

↑ always symmetric

Probability Theory

$\mathbb{E}[g(x)] \triangleq \sum_x g(x) p_x(x)$

$\mathbb{E}[a] = a$ for constant a

$\mathbb{E}[af(x)] = a\mathbb{E}[f(x)]$

$\mathbb{E}[f(x) + g(x)] = \mathbb{E}[f(x)] + \mathbb{E}[g(x)]$

$\mathbb{E}[\mathbb{1}\{x = k\}] = p(X = k)$

$Var[x] \triangleq \mathbb{E}[(x - \mathbb{E}[x])^2] \ge 0$

$= \mathbb{E}[x^2] - \mathbb{E}[x]^2$

$Var(x_1 + \dots + x_n) = Var(x_1) + \dots + Var(x_n)$

↑ independent

$Var[a] = 0$, $Var[af(x)] = a^2 Var[f(x)]$

	mean	var
Bernoulli	p	$p(1-p)$
Gaussian	μ	σ^2
Poisson	λ	λ

$Cov[X, Y] \triangleq \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$

$= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$

$Var[X + Y] = Var[X] + Var[Y] + 2Cov[X, Y]$

$Var[AX] = A\Sigma A^T$, $A \in \mathbb{R}^{m\times n}$ $Cov[X, X] = Var[X]$

$Var[a^TX] = a^T\Sigma a$, $a \in \mathbb{R}^n$

LMS Algo

$J(\theta) = \frac{1}{2}\sum_{i=1}^n (h_\theta(x^{(i)}) - y^{(i)})^2$

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$

$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2}(h_\theta(x) - y)^2$

$= (h(x) - y)\frac{\partial}{\partial \theta_j}(h(x) - y)$

$= (h(x) - y)x_j$

$J(\theta) = \frac{1}{2}(X\theta - y)^T(X\theta - y)$

$\nabla_\theta J(\theta) = X^TX\theta - X^Ty = 0$

$\theta = (X^TX)^{-1}X^Ty$

Probabalistic Interpretation

$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$, $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$

$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y^{(i)} - \theta^Tx^{(i)})^2}{2\sigma^2}\right)$

$L(\theta) = \prod_{i=1}^n p(y^{(i)}|x^{(i)};\theta)$

$\ell(\theta) = \log L(\theta) = n\log\frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{\sigma^2}\overbrace{\frac{1}{2}\sum_{i=1}^n (y^{(i)} - \theta^Tx^{(i)})^2}^{\text{maximize}}$

Locally-weighted linear regression

fit θ to $\sum_i w^{(i)}(y^{(i)} - \theta^Tx^{(i)})^2$

$w^{(i)} = \exp\left(\frac{-1}{2\tau^2}(x^{(i)} - x)^2\right)$

$\min_x f(x) = \|Ax - b\|_2^2 = \langle Ax - b, Ax - b\rangle$

$= x^TA^TAx - 2b^TAx + \|b\|_2^2$

$\nabla_x f(x) = 2A^TAx - 2A^Tb + 0 = 0$

$A^TAx = A^Tb$

$x = (A^TA)^{-1}A^Tb$

$\nabla_x^2 f(x) = 2A^TA \ge 0$

$\nabla_x x^TAx = 2Ax$

$\nabla_x b^Tx = b$

$\nabla_x^2 x^TAx = 2A$

$\nabla_A |A| = |A|A^{-T}$

$\nabla_A \log|A| = A^{-1}$

$\nabla_x x^Tb = b$

Logistic Regression

$h_\theta(x) = g(\theta^Tx) = \frac{1}{1 + e^{-\theta^Tx}}$

$g(z) = \frac{1}{1+e^{-z}}$, $g'(z) = g(z)(1 - g(z))$

$p(y=1|x;\theta) = h_\theta(x)$

$p(y=0|x;\theta) = 1 - h_\theta(x)$

$p(y|x;\theta) = (h_\theta(x))^y(1 - h_\theta(x))^{1-y}$

$\ell(\theta) = \sum_{i=1}^n y^{(i)}\log h(x^{(i)}) + (1 - y^{(i)})\log(1 - h(x^{(i)}))$

$\frac{\partial}{\partial\theta_j}\ell(\theta) = (y - h(x))x_j$

Newton's Method

$\theta := \theta - \frac{f(\theta)}{f'(\theta)} := \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$

$:= \theta - H^{-1}\nabla_\theta \ell(\theta)$, $H_{ij} = \frac{\partial^2 \ell(\theta)}{\partial\theta_i \partial\theta_j}$

Generalized Linear Models

$p(y;\eta) = b(y)\exp(\eta^TT(y) - a(\eta))$

Bernoulli: $p(y;\phi) = \phi^y(1-\phi)^{1-y}$

$= \exp(y\log\phi + (1-y)\log(1-\phi))$

$= \exp((\log\frac{\phi}{1-\phi})y + \log(1-\phi))$

$T(y) = y$, $a(\eta) = -\log(1-\phi)$, $b(y) = 1$

$\eta = \log\frac{\phi}{1-\phi}$ $= \log(1 + e^\eta)$, $\phi = \frac{1}{1+e^{-\eta}}$

Gaussian: $\sigma^2 = 1$

$p(y;\mu) = \frac{1}{\sqrt{2\pi}}\exp(\frac{-1}{2}(y - \mu)^2)$

$= \frac{1}{\sqrt{2\pi}}\exp(\frac{-1}{2}y^2)\cdot\exp(\mu y - \frac{1}{2}\mu^2)$

$\eta = \mu$, $T(y) = y$, $a(\eta) = \frac{\mu^2}{2} = \frac{\eta^2}{2}$

$b(y) = \frac{1}{\sqrt{2\pi}}\exp(\frac{-1}{2}y^2)$

Assumptions: 1) $y|x;\theta \sim ExpFamily(\eta)$

2) predict $\mathbb{E}[T(y)]$ given x

if $T(y) = y$, $h(x) = \mathbb{E}[y|x]$

3) $\eta = \theta^Tx$

ordinary least squares:

$h(x) = \mathbb{E}[y|x] = \mu = \eta = \theta^Tx$

$y|x;\theta \sim \mathcal{N}(\mu, \sigma^2)$

logistic regression:

$h(x) = \mathbb{E}[y|x;\theta] = \phi = \frac{1}{1+e^{-\eta}} = \frac{1}{1+e^{-\theta^Tx}}$

$y|x;\theta \sim Bernoulli(\phi)$

Canonical response $g(\eta) = \mathbb{E}[T(y);\eta]$

canonical link g^{-1}

Properties: max likelihood w.r.t. η concave

$\mathbb{E}[y;\eta] = \frac{\partial}{\partial\eta}a(\eta)$

$Var[y;\eta] = \frac{\partial^2}{\partial\eta^2}a(\eta)$

	η	$a(\eta)$	$b(y)$
Poisson:	$\log\lambda$	e^η	$\frac{1}{y!}$
Geometric:	$\log(1-\phi)$	$\log(\frac{e^\eta}{1-e^\eta})$	1

Softmax regression with GLM:

$(T(y))_i = \mathbb{1}\{y=i\}$ – one hot $T(y)\in\mathbb{R}^{k-1}$, $T(k)=\vec{0}$

$\mathbb{E}[(T(y))_i] = P(y=i) = \phi_i$

$p(y;\phi) = \phi_1^{\mathbb{1}\{y=1\}}\phi_2^{\mathbb{1}\{y=2\}}\cdots\phi_k^{\mathbb{1}\{y=k\}}$

$= \phi_1^{(T(y))_1}\phi_2^{(T(y))_2}\cdots\phi_k^{1-\sum_{i=1}^{k-1}(T(y))_i}$

$= \exp((T(y))_1\log\phi_1 + (T(y))_2\log\phi_2 + \dots + (1-\sum_{i=1}^{k-1}(T(y))_i)\log\phi_k)$

$= \exp((T(y))_1\log\frac{\phi_1}{\phi_k} + (T(y))_2\log\frac{\phi_2}{\phi_k} + \dots + (T(y))_{k-1}\log\frac{\phi_{k-1}}{\phi_k} + \log\phi_k)$

$\eta = \begin{bmatrix}\log \phi_1/\phi_k \\ \vdots \\ \log\frac{\phi_{k-1}}{\phi_k}\end{bmatrix}$, $a(\eta) = -\log(\phi_k)$, $g(\eta_i) = \eta_i = \log\frac{\phi_i}{\phi_k}$

$b(y) = 1$ $\qquad \phi_i = \frac{e^{\eta_i}}{\sum_{j=1}^k e^{\eta_j}}$

$\eta_i = \theta_i^T x$, $p(y=i|x;\theta) = \phi_i = \frac{e^{\theta_i^T x}}{\sum_{j=1}^k e^{\theta_j^T x}}$

$h_\theta(x) = \mathbb{E}[T(y)|x;\theta] = \mathbb{E}\left[\begin{matrix}\mathbb{1}\{y=1\} \\ \vdots \\ \mathbb{1}\{y=k-1\}\end{matrix}\middle| x;\theta\right] = \begin{bmatrix}\phi_1 \\ \vdots \\ \phi_{k-1}\end{bmatrix}$, $p(y=k|x;\theta) = 1-\sum_{i=1}^{k-1}\phi_i$

$\ell(\theta) = \sum_{i=1}^n \log p(y^{(i)}|x^{(i)};\theta) = \sum_{i=1}^n \log\prod_{l=1}^k\left(\frac{e^{\theta_l^T x^{(i)}}}{\sum_{j=1}^k e^{\theta_j^T x^{(i)}}}\right)$

Multivariate Gaussian

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

Σ is PD, so Σ^{-1} is PD $\qquad \Sigma_{ij} = Cov[X_i, X_j]$

$\Sigma = \mathbb{E}[(X-\mu)(X-\mu)^T] = \mathbb{E}[XX^T] - \mu\mu^T = Cov[X_i, X_j]$

Σ is PSD for any random X

univariate $\sigma^2 = \mathbb{E}[(X-\mu)^2] = \mathbb{E}[X^2]-\mu^2$, $\mu = \mathbb{E}[X]$

Generative Learning Algorithms

discriminative – learn $p(y|x)$ or $h_\theta(x)$ directly

generative – learn $p(x|y)$ & $p(y)$

$\arg\max_y p(y|x) = \arg\max_y p(x|y)p(y)$

$X\sim\mathcal{N}(\mu,\Sigma)$, $\mathbb{E}[X] = \int_x x\,p(x;\mu,\Sigma)dx = \mu$

$Cov(Z) = \mathbb{E}[ZZ^T] - \mathbb{E}[Z](\mathbb{E}[Z])^T$

$Cov(X) = \Sigma$

$y\sim Bernoulli(\phi)$, $x|y=0\sim\mathcal{N}(\mu_0,\Sigma)$
$x|y=1\sim\mathcal{N}(\mu_1,\Sigma)$

$p(y) = \phi^y(1-\phi)^{1-y}$

$p(x|y=0) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}}\exp\left(\frac{-1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\right)$

$\ell(\phi,\mu_0,\mu_1,\Sigma) = \log\prod_{i=1}^n p(x^{(i)}|y^{(i)};\mu_0,\mu_1,\Sigma)p(y^{(i)};\phi)$

maximize ℓ: $\phi = \frac{1}{n}\sum_{i=1}^n\mathbb{1}\{y^{(i)}=1\}$

$\mu_0 = \frac{\sum_{i=1}^n\mathbb{1}\{y^{(i)}=0\}x^{(i)}}{\sum_{i=1}^n\mathbb{1}\{y^{(i)}=0\}}$

$\Sigma = \frac{1}{2}\sum_{i=1}^n(x^{(i)}-\mu_{y^{(i)}})(x^{(i)}-\mu_{y^{(i)}})^T$

decision boundary with common Σ orthogonal to line connecting μ's

$p(y=1|x;\phi,\Sigma,\mu_0,\mu_1) = \frac{1}{1+\exp(-\theta^T x)}$ $(x_0=1)$

θ approximate function of ϕ,μ_0,μ_1,Σ

Naive Bayes

$x\in\{0,1\}^d$, $d=50{,}000$, $x_i\in\{0,1\}$

$p(x_1,\dots,x_d|y) = \prod_{i=1}^d p(x_i|y)$

$\mathcal{L}(\phi_y,\phi_{j|y=0},\phi_{j|y=1}) = \prod_{i=1}^n p(x^{(i)}|y^{(i)})$

maximize: $\phi_{j|y=0} = \frac{\sum_{i=1}^n\mathbb{1}\{x_j^{(i)}=1\wedge y^{(i)}=1\}+1}{\sum_{i=1}^n\mathbb{1}\{y^{(i)}=0\}+2}$ ↑possible values of y

$\phi_y = \frac{1}{n}\sum_{i=1}^n\mathbb{1}\{y^{(i)}=1\}$

multinomal $= \{(x^{(i)},y^{(i)}); i=1\dots n\}$ where $x^{(i)} = (x_1^{(i)},x_2^{(i)},\dots,x_{d_i}^{(i)})$, $d=$ # words in $x^{(i)}$

$\mathcal{L}(\phi_y,\phi_{k|y=0},\phi_{k|y=1}) = \prod_{i=1}^n p(x^{(i)},y^{(i)})$

$= \prod_{i=1}^n\left(\prod_{j=1}^{d_i}p(x_j^{(i)}|y;\phi_{k|y=0},\phi_{k|y=1})\right)p(y^{(i)};\phi_y)$

maximize: $\phi_{k|y=0} = \frac{\sum_{i=1}^n\sum_{j=1}^{d_i}\mathbb{1}\{x_j^{(i)}=k\wedge y^{(i)}=0\}+1}{\sum_{i=1}^n\mathbb{1}\{y^{(i)}=0\}d_i+|V|}$

Kernel Methods

$\phi:\mathbb{R}^d\mapsto\mathbb{R}^p$ may be high or infinite dimension

$\theta := \theta+\alpha\sum_{i=1}^n(y^{(i)}-\theta^T\phi(x^{(i)}))\phi(x^{(i)})$

$\theta = \sum_{i=1}^n\beta_i\phi(x^{(i)})$, $\theta\in\mathbb{R}^p$, $\beta\in\mathbb{R}^n$

$\theta := \sum_{i=1}^n(\beta_i+\alpha(y^{(i)}-\theta^T\phi(x^{(i)})))\phi(x^{(i)})$

$\beta_i := \beta_i+\alpha(y^{(i)}-\theta^T\phi(x^{(i)}))$

$:= \beta_i+\alpha(y^{(i)}-\sum_{j=1}^n\beta_j\phi(x^{(j)})^T\phi(x^{(i)}))$, $\forall i=1\dots n$

$K_{ij} = \phi(x^{(i)})^T\phi(x^{(j)}) = \phi(x^{(j)})^T\phi(x^{(i)}) = K_{ji}$

PSD

$\phi(x^{(j)})^T\phi(x^{(i)}) = \langle\phi(x^{(j)}),\phi(x^{(i)})\rangle \triangleq K(x^{(j)},x^{(i)}) = K(x,z)\in\mathbb{R}^{n\times n}$

1) pre-compute $\langle\phi(x^{(i)}),\phi(x^{(j)})\rangle$ for all i,j pairs

2) can often be computed efficiently

$\langle\phi(x),\phi(z)\rangle = 1+\sum_{i=1}^d x_iz_i+\sum_{i,j=1}^d x_ix_jz_iz_j+\sum_{i,j,k=1}^d x_ix_jx_kz_iz_jz_k$

$= 1+\langle x,z\rangle+\langle x,z\rangle^2+\langle x,z\rangle^3$

$\beta := \beta+\alpha(\vec{y}-K\beta)$, $\beta_i := \beta_i+\alpha(y^{(i)}-\sum_{j=1}^n\beta_jK(x^{(i)},x^{(j)}))$

prediction: $\theta^T\phi(x) = \sum_{i=1}^n\beta_i\phi(x^{(i)})^T\phi(x) = \sum_{i=1}^n\beta_iK(x^{(i)},x)$

$K(x,z) = (x^Tz)^2 = \sum_{i=1}^d x_iz_i\sum_{j=1}^d x_jz_j = \sum_i\sum_j(x_ix_j)(z_iz_j)$

$K(x,z) = (x^Tz+c)^2 = \sum_{i,j=1}^d(x_ix_j)(z_iz_j)+\sum_{i=1}^d(\sqrt{2c}\,x_i)(\sqrt{2c}\,z_i)+c^2$

Gaussian $K(x,z) = \exp\left(\frac{-1}{2\sigma^2}\|x-z\|^2\right)$

$z^TKz = \sum_i\sum_j z_iKz_j = \sum_i\sum_j z_i\phi(x^{(i)})^T\phi(x^{(j)})z_j$

$= \sum_i\sum_j z_i\sum_k\phi_k(x^{(i)})\phi_k(x^{(j)})z_j$

$= \sum_k\sum_i\sum_j z_i\phi_k(x^{(i)})\phi_k(x^{(j)})z_j$

$= \sum_k\left(\sum_i z_i\phi_k(x^{(i)})\right)^2 \geq 0$

Support Vector machines

$y\in\{-1,1\}$, $h_{w,b}(x) = g(w^Tx+b)$, $g(z)=1$ if $z\geq 0$ and -1 o.w.

functional margin $\hat{\gamma}^{(i)} = y^{(i)}(w^Tx^{(i)}+b)$, $\hat{\gamma} = \min_i\hat{\gamma}^{(i)}$, normalize $\|w\|=1$

geometric margin $\gamma^{(i)} = \frac{1}{\|w\|}(w^Tx^{(i)}+b)y^{(i)}$, $\gamma = \min_i\gamma^{(i)}$, w,b to max γ

optimal margin classifier:

#1 max γ s.t. $y^{(i)}(w^Tx^{(i)}+b)\geq\gamma$, $i=1\dots n$ and $\|w\|=1$

#2 max $\hat{\gamma}/\|w\|$ s.t. $y^{(i)}(w^Tx^{(i)}+b)\geq\hat{\gamma}$, $i=1\dots n$

#3 min $\frac{1}{2}\|w\|^2$ s.t. $y^{(i)}(w^Tx^{(i)}+b)\geq 1$, $i=1\dots n$

$\min\frac{1}{2}\|w\|^2 = \frac{1}{2}\left(\sum_{i=1}^n\alpha_iy^{(i)}x^{(i)}\right)^T\left(\sum_{j=1}^n\alpha_jy^{(j)}x^{(j)}\right)$

$= \frac{1}{2}\sum_i\sum_j\alpha_i\alpha_jy^{(i)}y^{(j)}\underbrace{x^{(i)T}x^{(j)}}_{\langle x^{(i)},x^{(j)}\rangle}$

Constraint:

$y^{(i)}((\sum_{j=1}^n\alpha_jy^{(j)}x^{(j)})x^{(i)}+b)$
≥ 1

$y^{(i)}((\sum_j\alpha_jy^{(j)}\langle x^{(i)},x^{(j)}\rangle+$
≥ 1

Neural Network

$$X = \begin{bmatrix} | & & | \\ x^{(1)} & \cdots & x^{(n)} \\ | & & | \end{bmatrix} \updownarrow d$$

$$Z = \begin{bmatrix} | & & | \\ z^{[1](1)} & \cdots & z^{[1](n)} \\ | & & | \end{bmatrix} \updownarrow \#\text{ nodes}$$

forward: $z^{[1]} = W^{[1]}x + b^{[1]}$

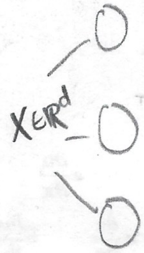
$a^{[1]} = \sigma(z^{[1]})$ $\quad \sigma(z) = \frac{1}{1+e^{-z}}$

$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$

$a^{[2]} = \sigma(z^{[2]})$

$z^{[3]} = W^{[3]}a^{[2]} + b^{[3]}$

$\underbrace{a^{[3]}}_{1\times1} = \sigma(z^{[3]}) = \underbrace{\hat{y}}_{1\times 1}$



$W^{[1]} \in \mathbb{R}^{3\times d}$

$X \in \mathbb{R}^{d\times 1}$

$W^{[1]}X \in \mathbb{R}^{3\times 1}$

$b^{[1]} \in \mathbb{R}^{3\times 1}$

real-valued regression: $\mathcal{L}(y,\hat{y}) = \frac{1}{2}(\hat{y}-y)^2$

logistic regression: $\mathcal{L}(y,\hat{y}) = -(y\log\hat{y} + (1-y)\log(1-\hat{y}))$

softmax over k: $\mathcal{L}(y,\hat{y}) = -\sum_{j=1}^{K} \mathbb{1}\{y=j\}\log\hat{y}_j$

Jensen's inequality:
f is convex and X is random,
$\mathbb{E}[f(x)] \geq f(\mathbb{E}[x])$

Accuracy $= \frac{TP+TN}{all}$

Precision $= \frac{TP}{TP+FP}$

recall sensitivity $= \frac{TP}{TP+FN}$

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

F1 $= \frac{2TP}{2TP+FP+FN}$

true pos. rate $= \frac{TP}{TP+FN}$

false pos. rate $= \frac{FP}{TN+FP}$

$\mathcal{L}^{(i)} = -[y^{(i)}\log\hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})]$

$J(\hat{y},y) = \frac{1}{n}\sum_i \mathcal{L}^{(i)}$

backward: $\frac{\partial J}{\partial w^{[3]}} = \underbrace{\frac{\partial J}{\partial a^{[3]}} \cdot \frac{\partial a^{[3]}}{\partial z^{[3]}}} \cdot \frac{\partial z^{[3]}}{\partial w^{[3]}}$

$\frac{\partial J}{\partial w^{[2]}} = \underbrace{\frac{\partial J}{\partial z^{[3]}} \cdot \frac{\partial z^{[3]}}{\partial a^{[2]}} \cdot \frac{\partial a^{[2]}}{\partial z^{[2]}}} \cdot \frac{\partial z^{[2]}}{\partial w^{[2]}}$

$\frac{\partial J}{\partial w^{[1]}} = \frac{\partial J}{\partial z^{[2]}} \cdot \frac{\partial z^{[2]}}{\partial a^{[1]}} \cdot \frac{\partial a^{[1]}}{\partial z^{[1]}} \cdot \frac{\partial z^{[1]}}{\partial w^{[1]}}$

$\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[3]}} = -\left[y^{(i)}\frac{\partial}{\partial w^{[3]}}\log\overbrace{\sigma(w^{[3]}a^{[2]}+b^{[3]})}^{a^{[3]}} + (1-y^{(i)})\frac{\partial}{\partial w^{[3]}}\log(1-\sigma(w^{[3]}a^{[2]}+b^{[3]}))\right]$

$= -(y^{(i)} - a^{[3]})a^{[2]T}$

$\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[2]}} = \underbrace{\frac{\partial \mathcal{L}}{\partial a^{[3]}} \cdot \frac{\partial a^{[3]}}{\partial z^{[3]}}} \cdot \underbrace{\frac{\partial z^{[3]}}{\partial a^{[2]}}}_{w^{[3]T}} \cdot \underbrace{\frac{\partial a^{[2]}}{\partial z^{[2]}}}_{a^{[2]}(1-a^{[2]})} \cdot \underbrace{\frac{\partial z^{[2]}}{\partial w^{[2]}}}_{a^{[1]T}}$

$\frac{\partial \mathcal{L}}{\partial w^{[3]}} = \underbrace{\frac{\partial \mathcal{L}}{\partial z^{[3]}}}_{-(y^{(i)}-a^{[3]})} \cdot \underbrace{\frac{\partial z^{[3]}}{\partial w^{[3]}}}_{a^{[2]T}}$

$\frac{\partial \mathcal{L}^{(i)}}{\partial w^{[2]}} = w^{[3]T} * a^{[2]}(1-a^{[2]})(a^{[3]}-y^{(i)})a^{[1]T}$

K-means

given: $x^{(1)} \ldots x^{(n)} \in \mathbb{R}^d$, $k = \#$ clusters

1) randomly select $\mu^{(1)}, \ldots, \mu^{(k)} \in \mathbb{R}^d$

2) for $i = 1 \ldots n$: $\quad \leftarrow$ assign

$c^{(i)} = \arg\min_{j \in 1\ldots k} \|x^{(i)} - \mu_j\|^2$

3) for $j = 1 \ldots k$: $\quad \leftarrow$ compute μ

$\mu^{(j)} = \frac{1}{|\Omega_j|}\sum_{i\in\Omega_j} x^{(i)}$, $\Omega_j = \{i : c^{(i)} = j\}$

$J(c,\mu) = \sum_i \|x^{(i)} - \mu_{c^{(i)}}\|^2$

$\binom{n}{k} = \frac{n!}{k!(n-k)!}$

↑ n choose k

$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}.$$

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} & \frac{\partial y_2}{\partial x} & \cdots & \frac{\partial y_m}{\partial x} \end{bmatrix}.$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}.$$

$$\frac{\partial y}{\partial \mathbf{X}} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{12}} & \cdots & \frac{\partial y}{\partial x_{1q}} \\ \frac{\partial y}{\partial x_{21}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{2q}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{p1}} & \frac{\partial y}{\partial x_{p2}} & \cdots & \frac{\partial y}{\partial x_{pq}} \end{bmatrix}$$

Identities: vector-by-vector $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$

Condition	Expression	Numerator layout, i.e. by $\mathbf{y}$ and $\mathbf{x}^T$	Denominator layout, i.e. by $\mathbf{y}^T$ and $\mathbf{x}$
$\mathbf{a}$ is not a function of $\mathbf{x}$	$\frac{\partial \mathbf{a}}{\partial \mathbf{x}} =$	$\mathbf{0}$	
	$\frac{\partial \mathbf{x}}{\partial \mathbf{x}} =$	$\mathbf{I}$	
$\mathbf{A}$ is not a function of $\mathbf{x}$	$\frac{\partial \mathbf{A}\mathbf{x}}{\partial \mathbf{x}} =$	$\mathbf{A}$	$\mathbf{A}^T$
$\mathbf{A}$ is not a function of $\mathbf{x}$	$\frac{\partial \mathbf{x}^T\mathbf{A}}{\partial \mathbf{x}} =$	$\mathbf{A}^T$	$\mathbf{A}$
a is not a function of $\mathbf{x}$, $\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial a\mathbf{u}}{\partial \mathbf{x}} =$	$a\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	
$v = v(\mathbf{x})$, $\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial v\mathbf{u}}{\partial \mathbf{x}} =$	$v\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \mathbf{u}\frac{\partial v}{\partial \mathbf{x}}$	$v\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial v}{\partial \mathbf{x}}\mathbf{u}^T$
$\mathbf{A}$ is not a function of $\mathbf{x}$, $\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial \mathbf{A}\mathbf{u}}{\partial \mathbf{x}} =$	$\mathbf{A}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\mathbf{A}^T$
$\mathbf{u} = \mathbf{u}(\mathbf{x})$, $\mathbf{v} = \mathbf{v}(\mathbf{x})$	$\frac{\partial (\mathbf{u}+\mathbf{v})}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$	
$\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}$
$\mathbf{u} = \mathbf{u}(\mathbf{x})$	$\frac{\partial \mathbf{f}(\mathbf{g}(\mathbf{u}))}{\partial \mathbf{x}} =$	$\frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{u}}{\partial \mathbf{x}}\frac{\partial \mathbf{g}(\mathbf{u})}{\partial \mathbf{u}}\frac{\partial \mathbf{f}(\mathbf{g})}{\partial \mathbf{g}}$

$b, x \in \mathbb{R}^n$

$f(x) = \sum_{i=1}^n b_i x_i,\ f(x) = b^T x$

$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^n b_i x_i = b_k$

$\nabla_x b^T x = b$

$f(x) = x^T A x,\ A \in \mathbb{S}^n$

$= \sum_{i=1}^n \sum_{j=1}^n A_{ij} x_i x_j$

$\frac{\partial f(x)}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_{i=1}^n \sum_{j=1}^n A_{ij} x_i x_j$

$= \frac{\partial}{\partial x_k}\Big[\sum_{i\neq k}\sum_{j\neq k} A_{ij} x_i x_j + \sum_{i\neq k} A_{ik} x_i x_k + \sum_{j\neq k} A_{kj} x_k x_j + A_{kk} x_k^2\Big]$

$= \sum_{i\neq k} A_{ik} x_i + \sum_{j\neq k} A_{kj} x_j + 2A_{kk} x_k$

$= \sum_{i=1}^n A_{ik} x_i + \sum_{j=1}^n A_{kj} x_j$

$= 2\sum_{i=1}^n A_{ki} x_i$

$\nabla_x x^T A x = 2Ax,\ A \in \mathbb{S}^n$

$\frac{\partial^2 f(x)}{\partial x_k \partial x_\ell} = \frac{\partial}{\partial x_\ell} 2\sum_{i=1}^n A_{ki} x_i = 2A_{k\ell} = 2A_{\ell k}$

$\nabla_x^2 x^T A x = 2A$

maximum likelihood estimate $L = \prod_{i=1}^n p(x|y)$

Lagrangian $\mathcal{L}(\omega, b) = f(\omega) + \sum_{i=1}^{l} \beta_i h_i(\omega)$

SVM (cont.)

Hinge loss $L(z,y) = \max(0, 1 - zy)$

Lagrangian $\mathcal{L}(\omega, b, \alpha) = \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^n \alpha_i \left[y^{(i)}(\omega^T x^{(i)} + b) - 1\right]$

$\nabla_\omega \mathcal{L}(\omega, b, \alpha) = \omega - \sum_i \alpha_i y^{(i)} x^{(i)} = 0$

$\frac{\partial}{\partial b} \mathcal{L} = \sum_i \alpha_i y^{(i)} = 0$

$\mathcal{L}(\omega, b, \alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j=1}^n y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$

$\max_\alpha W(\alpha) = \mathcal{L}(\omega, b, \alpha)$ s.t. $\alpha_i \geq 0,\ \sum_{i=1}^n \alpha_i y^{(i)} = 0$

solve for α_i, b

$\omega^* = \sum_{i=1}^n \alpha_i y^{(i)} x^{(i)}$

$b^* = \frac{1}{2}\left(\max_{i:y^{(i)}=-1} \omega^{*T} x^{(i)} + \min_{i:y^{(i)}=1} \omega^{*T} x^{(i)}\right)$

prediction $= h_{\omega,b}(x) = g(\omega^T x + b)$

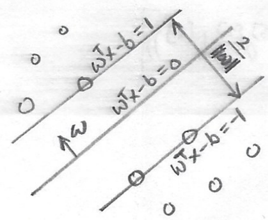
$= g\left(\left(\sum_i \alpha_i y^{(i)} x^{(i)}\right) x + b\right)$

$= g\left(\sum_i \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b\right)$

Representer theorem: ω is linear combo of examples

$\omega = \sum_i \alpha_i x^{(i)} = \sum_i \alpha_i y^{(i)} x^{(i)}$

$\uparrow \{0,1\}$



$\omega^T x + b = \left(\sum_i \alpha_i y^{(i)} x^{(i)}\right)^T x + b$

$= \sum_i \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b$

zero except for support vectors

Loss functions:

Least square error: $\frac{1}{2}(y - z)^2$

Logistic: $\log(1 + \exp(-yz))$

Hinge: $\max(0, 1 - yz)$

Cross-entropy: $-[y\log(z) + (1-y)\log(1-z)]$

$J(\theta) = \sum_{i=1}^n L(h(x^{(i)}), y^{(i)})$

$\theta \leftarrow \theta - \alpha \nabla J(\theta)$

likelihood $L(\theta)$, $\theta^{opt} = \arg\max_\theta L(\theta)$

regularization: penalty C - hyperparameter, violations ξ

$\min \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^n \xi_i$ s.t. $y^{(i)}(\omega^T x^{(i)} + b) \geq 1 - \xi_i,\ \xi_i \geq 0$

if $y^{(i)}(\omega^T x^{(i)} + b) > 0 \Rightarrow$ classified correctly

$\mathcal{L}(\omega, b, \xi, \alpha, r) = \frac{1}{2}\omega^T\omega + C\sum_i \xi_i - \sum_i \alpha_i \left[y^{(i)}(x^T\omega + b) - 1 - \xi_i\right] - \sum_i r_i \xi_i$

$\max_\alpha W(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j=1}^n y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$

s.t. $0 \leq \alpha_i \leq C,\ \sum_i \alpha_i = 0,\ i = 1 \ldots n$

$\alpha_i = 0 \Rightarrow y^{(i)}(\omega^T x^{(i)} + b) \geq 1$

$\alpha_i = C \Rightarrow y^{(i)}(\omega^T x^{(i)} + b) \leq 1$

$0 < \alpha_i < C \Rightarrow y^{(i)}(\omega^T x^{(i)} + b) = 1$