① Regularization

Recall logistic regression

Basic/unregularized objective: $L(w) = \sum_{i=1}^{n} -\log \sigma(\underbrace{y^{(i)} \cdot w^T x^{(i)}}_{\text{margin}})$

margin larger better

Goal: maximize margin on all training examples

trade-off

$\boxed{\text{prefer simple function}}$

L2 Regularization

add $\lambda \|w\|^2$ to objective

constant hyperparam

Goal: have smaller entries in w

L1 Regularization

add $\lambda \|w\|_1$ to objective

$\sum_{j=1}^{d} |w_j|$

Goal: have some entries of $w = 0$

Maximum a Posterior Inference (MAP) ← provides justification for L2, L1 reg

parameters w, dataset D

view both as random variables (Bayesian)

(In contrast, in MLE w is not a random variable) (Frequencist)

natural goal:

prior over w

← same as MLE

maximize $P(w|D) = \dfrac{P(w)\,P(D|w)}{P(D)}$

← normalizing constant doesn't depend on w

Suppose $P(w)$ is that each $w_j \sim N(0, \sigma^2)$ independently

$P(w) = \prod_{j=1}^{d} \frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{w_j^2}{\sigma^2}\right)$

$\log P(w) = \sum_{j=1}^{d} \log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \frac{w_j^2}{\sigma^2} = \text{constant} - \sum_{j=1}^{d} \frac{w_j^2}{\sigma^2}$

constant

$= \text{constant} - \frac{1}{\sigma^2}\|w\|^2$

maximizing $\log p(w) \Leftrightarrow$ minimizing $\|w\|^2$

if $p(w)$ Gaussian, equivalent to L_2 regularization

Kernals: a way to modify existing algorithms

- ☐ kernalized linear regression
- ☐ kernalized logistic regression
- ...

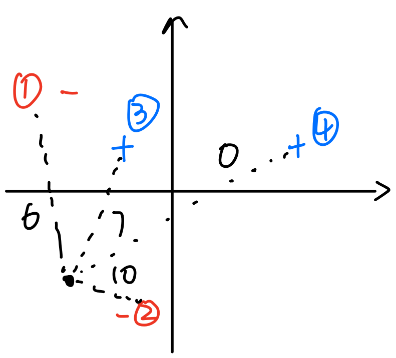
Infinite-dimensional features

(by preprocessing data, we can add new features)

price	Area	Area^2	...

Kernals systematically add many features (possible infinite), but also give a way to work with big feature vectors efficiently.

Make predictions by measuring similarity of test example to each training example. (similar to K-NN)



example weights:

$a_1 = -1$

$a_2 = -1$

$a_3 = 1$

$a_4 = 1$

predict: $-1 \cdot 6 - 1 \cdot 10 + 1 \cdot 0 + 1 \cdot 7$

$= -9 \Rightarrow -1$

In kernal logistic regression:

prediction $f(x) = \underbrace{\sum_{i=1}^{n}}_{\text{sum over training examples}} \underbrace{a_i}_{\text{weights learned}} \underbrace{k(x, x^{(i)})}$ ↙ kernal function measures how similar two inputs are if $f(x) > 0$, predict 1

if $f(x) < 0$, predict -1

Consider $k(x, z) = x^T z$

Captures some notion of similarity, if $x = z$, $k(x,z) \geq 0$

if point in opposite direction, $k(x,z) < 0$

Logistic regression algorithm can be written in terms of kernels only

(any x's only show up in the kernel function)

Logistic regression via GD

$w^{(0)} \leftarrow 0 \in \mathbb{R}^d$

for $t = 1, \cdots, T$:

$$w^{(t)} \leftarrow w^{(t-1)} - \eta \sum_{i=1}^{n} \underbrace{\sigma(-y^{(i)} \cdot w^{(t-1)T} \cdot x^{(i)}) \cdot y^{(i)}}_{\text{scaler}} \cdot \underbrace{x^{(i)}}_{\text{vector}}$$

$w \leftarrow w^{(T)}$

Given x, compute $w^T x$ for prediction

① w is a linear combination of $x^{(i)}$'s

"reparametrize" algorithm to update coefficients of this linear combination

Kernel logistic regression

$a^{(0)} \leftarrow 0 \in \mathbb{R}^n$ define $w^{(t)} = \sum_{i=1}^{n} a_i^{(t)} \cdot x^{(i)}$

for $t = 1, \cdots, T$:

for $i = 1, \cdots, n$:

$$a_i^{(t)} \leftarrow a_i^{(t-1)} + \eta \cdot \sigma(-y^{(i)} \cdot \underbrace{w^{(t-1)T}}\, x^{(i)}) \cdot y^{(i)}$$

return $a = a^{(T)}$

$$= \left(\sum_{j=1}^{n} a_j^{(t-1)} x^{(j)}\right)^T x^{(i)}$$

$$= \sum_{j=1}^{n} a_j^{(t-1)} x^{(j)T} x^{(i)}$$

$$= \sum_{j=1}^{n} a_j^{(t-1)} K(x^{(j)}, x^{(i)})$$

prediction: $W^T x$ for test input x

$$= \sum_{j=1}^{n} a_j K(x^{(j)}, x)$$

If we can compute kernal between any 2 x's, we don't have to store x's themselves.