Joint probability:

① two independent event A and B

probability of A and B occuring Both, $P(A \cap B) = P(A) * P(B)$

② two dependent even A and B

probability of A and B occuring, $P(A \cap B) = P(A) * P(B|A)$

Conditional probability:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Total probability:

$$P(A) = \sum_{i}^{n} P(A|B_i) * P(B_i|B)$$

Bayes Rule:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \leftarrow \text{prior}$$

Gaussian (Normal) Distribution:

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

② modeling small random noise errors.

Maximum Likelyhood Estimation:

distribution parameter

$$P(D|\theta) = \theta^3(1-\theta)^2$$

H, T, T, H, H

θ = Bernoulli Distribution

iid

$$L(\theta|D) = \log P(D|\theta)$$

$$\max_\theta = \theta^3(1-\theta)^2$$

$$\log(L) = 3\log\theta + 2\log(1-\theta)$$

$$\frac{d}{d\theta}\log L = \frac{3}{\theta} + \frac{2}{(1-\theta)}(-1)$$

$$\Rightarrow \frac{3}{\theta} - \frac{2}{(1-\theta)} = 0$$

$$\theta^* = \frac{3}{5} \quad \| \quad \hat{\theta}_{MLE} = \frac{n_H}{n_H + n_T}$$

parameter values that maximize the Likelyhood data

MLE for Mean of Gaussian; $\hat{\mu}_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i$

MLE for variance of Gaussian. $\hat{\sigma}^2_{MLE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_{MLE})^2$

MAP:

① $P(x|\theta) = \theta^m(1-\theta)^{n-m}$ | $D = \{x_1, x_2 \dots x_n\}$

① prion $= P(\theta) = Beta(\alpha, \beta)$ | $B(\alpha, \beta) = \alpha - 1, \beta - 1$

② Posterior $P(\theta|x) \propto P(x|\theta) \times P(\theta)$

$$= \theta^m(1-\theta)^{n-m} \cdot \theta^{\alpha-1}(1-\theta)^{\beta-1}$$

$$= \theta^{m+\alpha-1}(1-\theta)^{n-m+\beta-1}$$

for bigger sample, MAP $\approx$ MLE,

$$\hat{\theta}_{MAP} = \frac{n_H + \alpha - 1}{n_H + \alpha - 1 + n_T + \beta - 1}$$

Linear Regression:

$$y(x, w) = w_0 + w_1 x_1 + w_2 x_2 + \dots + w_d x_d$$

$$= w_0 + \sum_{j=1}^{d} w_j x_j$$

Notion convenience:

$$y(\vec{x}, \vec{w}) = \sum_{j=0}^{d} w_j x_j$$

$$= \vec{w}^T \vec{x}$$

Squared Loss:

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} \underbrace{\left(x_n^T w\right)}_{\text{Prediction}} - \underbrace{(t_n)}_{\text{Actual}}{}^2 =$$

$$\frac{1}{2} (xw - t)^T (xw - t)$$

Optimization:

minimize $E(w)$. find w.

$$\hat{w} = \left(X^T X\right)^{-1} X^T t$$

Probability density function: (Gaussian) Normal

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$$

μ = mean

σ = variance

Likelyhood of Linear Regression:

$$(2\pi\sigma)^{-\frac{N}{2}} e^{-\frac{1}{2\sigma^2}(t - xw)^T (t + xw)}$$

Ridge regression:

$$w = (\lambda I + X^T X)^{-1} X^T t$$

$$E(w) = (Xw - t)^T (Xw - t) + \lambda \vec{w}^T w$$

if $\lambda \to 0$ unregularized

$\lambda > 0$ (should be for regularization)

if $\lambda \Rightarrow \infty$ $\|w\| \Rightarrow 0$

underfitting

Polynomial Curve fitting:

$x \in R$, $y(x,w) = w_0 + w_1 x + w_2 x^2 + \dots + w_m x^m = \sum_{j=0}^{m} w_j x^j$

x = Non-linear

w = linear

General case: (Linear Basis function)

$$y(x,w) = w_0 + w_1 \phi(x) + w_2 \phi(x) + \dots + w_{m-1} \phi_{m-1}(x)$$

$$\phi(x) = x^j$$

$$y(x,w) = \sum w^T \phi(x)$$

Least square:

$$w_{ML} = (\phi^T \phi)^{-1} \phi^T t$$

- Generalization meaning in the context of machine learning:

  Good performance on any new, unseen data.

- Underfitting = high bias
- Overfitting = high variance

Logistic function (Sigmoid Function):

$$f(x) = \frac{1}{1+e^{-x}}$$

Logistic regression probabilistic function:

$$P(Y=0|x) = \frac{1}{1+\exp\left(w_0 + \sum w_i x_i\right)}$$

$$P(Y=1|x) = \frac{\exp\left(w_0+\sum w_i x_i\right)}{1+\exp\left(w_0+\sum_{i} w_i x_i\right)}$$

odds ratio:

$$\frac{P(Y=1|x)}{P(Y=0|x)} = \exp\left(w_0 + \sum w_i x_i\right)$$

$$\log\left(\frac{P(Y=1|x)}{P(Y=0|x)}\right) = \underbrace{w_0 + \sum_{i} w_i x_i}_{\text{Linear fuction}}$$

For more than two class:

K = total number of class

d = total feature

$j=1$ = index for all class

$$P(Y=y_k|x) = \frac{\exp\left(w_{k0} + \sum_{i=1} w_{ki} x_i\right)}{1+\sum_{j=1}^{k-1} \exp\left(w_{j0}+\sum_{i=1}^{d} w_j x_i\right)}$$

$$k=K \quad P(Y=y_k|x) = \frac{1}{1+\sum_{j=1}^{k-1} \exp\left(w_{j0}+\sum_{i=1} w_j x_i\right)}$$

Training logistic Regression:

$$l(w) = \ln \prod P(Y^j|X^j, w)$$

$$= \sum_{j} \left[y^j\left(w_0 + \sum_{i}^{d} w_i x_i^j\right) - \ln\left(1+\exp\left(w_0+\sum_{i}^{d} w_i x_i^j\right)\right)\right]$$

- $l(w)$ is a concave function.

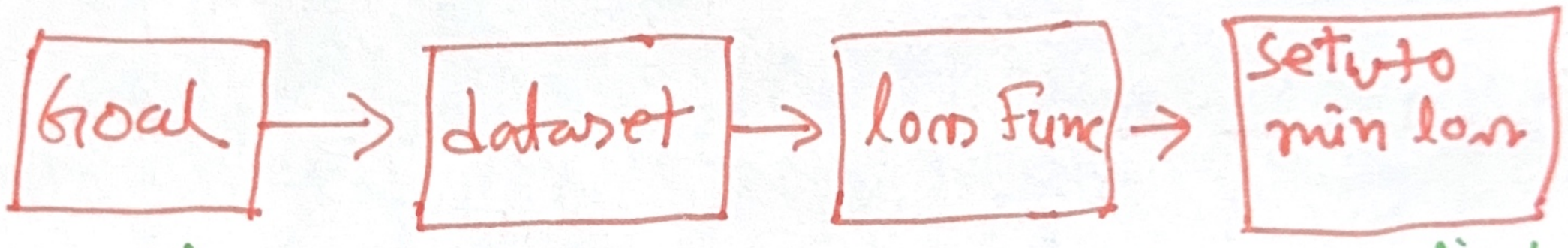
for example:

$$y = -9 - x^2 - y^2$$

cost of x^2 is negative.

- No-closed form solution.
- easy of optimized (unique maximum)

- Gradient discent:



$$Argmin_x f(x), \quad x_{t+1} = x_t - \lambda f'(x_t)$$

or,

$$w \leftarrow w - \eta \nabla E_D(w)$$

- Batch gradient:

$$\nabla E_D(w) = \left[\frac{dE_D(w)}{dw_0} \cdots \frac{dE_D(w)}{dw_n} \right]$$

$$w \leftarrow w - \eta \nabla E_D(w)$$

- Stochastic gradient: (Faster)

$$\nabla E_d(w) = \left[\frac{dE_d(w)}{dw_0} \cdots \frac{dE_d(w)}{dw_n} \right]$$

$$w \leftarrow w - \eta \nabla E_d(w)$$

Gradient ascent for logistic regression:

$$\nabla l(w) = \sum_j x_i^j \left(y^j - p(y^j = 1 \mid x^j, w) \right)$$

$$= \sum_j x_i^j \left(y^j - \frac{\exp\left(w_0 + \sum_i w_i x_i^j\right)}{1 + \exp\left(w_0 + \sum_i w_i x_i^j\right)} \right)$$

y^j = Ground truth

$p(y^j = 1 \mid x^j, w)$ = prediction

Effect of Step Size:

$$w \leftarrow w - \eta \nabla E_p(w)$$

η = too large ⇒ oscillating

η = too small ⇒ slow convergence

Regularization:

$$w^* = \arg\max \ln \sum_{j=1}^{N} p(y^j \mid x^j, w)$$

normal

$$w_i^{t+1} \leftarrow w_i^t + \eta \sum_j x_i^j \left[y^j - p(y^j = 1 \mid x^j, w) \right]$$

regularized

$$w^* = \arg\max \ln \sum_{j=1}^{N} p(y^j \mid x^j, w) - \frac{\lambda}{2} \|w\|^2 \quad \left(\|w\|^2 = \sum_{i=1}^{k} w_i^2\right)$$

$$w_i^{(t+1)} \leftarrow w_i^t + \eta \left[-\lambda w_i^t + \sum_j x_i^j \left[y^j - p(y = 1 \mid x, w) \right] \right]$$

1. Gradient ascent to maximize the cost function $l(w)$. [coz it's concave]

2. purpose of regularization in logistic Regression is

Soft margin

- $\frac{1}{2}\|w\| + C\sum \xi_i$

  if margin $\gg 1$: Don't care
  margin < 1: pay linear penalty

  slack, $(wx+b)y_i \geq 1-\xi_i$

  $\rightarrow \xi = 1 - y(w \cdot x + b) \Rightarrow$ Hinge loss

  $\rightarrow C$ = slack penalty $C > 0$;

- if $C = \infty$,
  have to separate the data, overfitting

  if $C = 0$,
  ignore data.

  C = small, smooth and simple
  decision boundary.
  $\xi > 1$ if misclassified.

- multiple class.

$$(w_k, b_k)_k = 1, 2, 3$$

$$y = \arg\max_k (w_k x + b_k)$$

C = Chosen by cross-validation

$wx+b=0$ → decision boundary

$wx+b=+1$

$wx+b=-1$

Goal: $\min w, b$ to $\max \gamma$ = margin

For all data.

$$(wx+b)y_j \geq \gamma$$

projection on plane.

$$\underline{x_j} = \bar{x_j} + \underline{\lambda}\left[\frac{w}{\|w\|}\right] \text{ unit vector}$$

projection ↓ Distance

For class:

$$x^+ = x^- + 2\lambda \frac{w}{\|w\|}$$

Unnormalize problem:

Maximize γ, w, b

$$(wx_j+b)y_j \geq \gamma$$

Normalized problem:

$$\gamma = \frac{\|w\|}{w.w} = \frac{1}{\sqrt{w.w}} = \frac{1}{2}\|w\|^2$$

↑ minimize

$$(w.x_j+b)y_j \geq 1 \quad \forall j \in \text{Dataset.}$$

min

Dual SVM - linearly separable case

$$L(w,b,\alpha) = \frac{1}{2} w \cdot w - \sum_j \alpha_j \left[(w \cdot x_j + b) y_j - 1 \right]$$

$$\max_{\alpha \geq 0} \min_{w,b} \qquad \alpha_j \geq 0$$

$$\frac{dL}{dw} = 0 \Rightarrow w = \sum_j \alpha_j y_j x_j \quad -(1)$$

$$\frac{dL}{db} = 0 \Rightarrow \sum_j \alpha_j y_j = 0 \quad -(11)$$

Sub (11) and (1).

$$\sum \alpha_j - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j \underline{(x_i x_j)}$$

$$\underline{\sum_i \alpha_i y_i = 0} \quad \text{train sol gives } \alpha_j\text{'s}$$

$$\alpha_i \geq 0 \qquad w =$$

Dual SVM - Sparsity.

Support vector if $\alpha_j > 0$.

Kernel trick:

$$k(x_i, x_j) = \varphi(x_i)^T \times \varphi(x_j)$$

Dual SVM - Non-separable

① Same as Linear.

$$C \geq \alpha_j \geq 0$$

Efficient dot product of poly:

$$K(u,v) = \varphi(u) \cdot \varphi(v) = (u \cdot v)^d$$

Definition of kernel:

$$k(x,x') = \varphi(x)^T \varphi(x')$$

$k_{ij} = k(x_i, x_j)$ is positive semi-definite

SVM with kernel:

1. Choose feature and kernel
2. Solve dual SVM to get α_j
3. Compute w, b.
4. Classify as $(w \cdot \varphi(x) + b)$

Avoid Overfitting by:

→ Setting C

→ Better kernel

→ varying parameter of kernel

- Boosting works well even if error ≤ 0.5
- Highly sensitive to outlier. → overfit
- weight update rule:

$$D_1(i) = \frac{1}{m}$$

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x))}{Z_t}$$

$$Z_t = \sum_{i=1}^{m} D(t)\exp(-\alpha_t y_i h_t(x_i))$$

- weight $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
- training error,

$$\epsilon_t = \sum_{i=1}^{m} D_t \delta\left(h_t(x_i) \neq y_i\right)$$

$$= \frac{1}{m}\sum_{i=1}^{m} \delta\left(H(x_i) \neq y_i\right) \leq \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i H(x_i))$$

$$h(x) = \sum_t \alpha_t h_t(x) \quad \text{[convex]}$$

- Boosting the margin

$$r(x_i) = y_i \frac{\alpha_1 h_i(x_1) + \dots \alpha_T H_T(x_1)}{\alpha_1 + \alpha_2 + \dots + \alpha_T}$$

- Handles overfit By throwing all the classifiers have low variance, high bias.

k-NN:

k too high → overfitting

k too low → underfitting

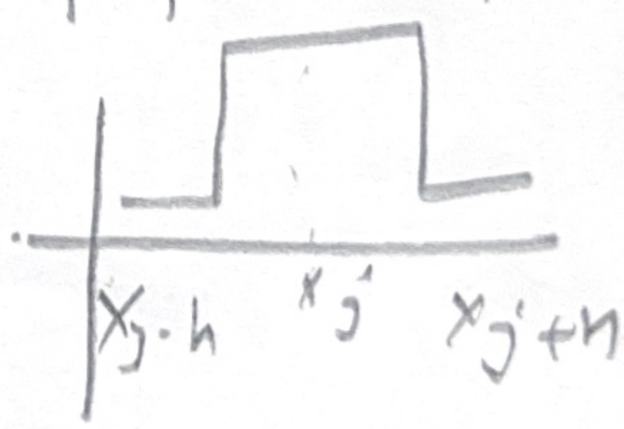
for a new data (x_1, x_2)

- Find the distance from all data points
- Define k, and find all the nearest neighbour.
- Select by vote.

Kernel Regression:

boxcar $k(x) = \frac{1}{2} I(x)$

$k\left(\frac{x_j - x}{h}\right)$



h is low → overfitting

h is high → underfitting

Density Estimation:

$$p_i = \frac{n_i}{N\Delta_i}$$

Δ 0.04 overfit

Δ 0.25 underfit

Δ 0.08 perfect.

more data = complex model

number of parameter ≃ training data

k-means clustering:

- unsupervised,
- gets Density estimation
- groups on cluster.
- Sensitive to seed choice. (some convergence).

So, do running the multiple starting point result at another result.

Algorithm:

① Random centeroid. $u^{(0)} = u_1^{(0)} \dots u_k^{(0)}$

② Classify each point,

$$c^{(t)}(j) \leftarrow \arg\min_{i=1\dots k} \|u_i^{(t)} - x_j\|$$

③ Recenter centeroid.

$$u_i^{(t+1)} \leftarrow \arg\min_{u} \sum_{j:c^{(t)}(j)=i} \|u - x_j\|$$

u_1 = avg points / mean,

cost function:

$$F(u,c) = \sum_{j=1}^{m} \|u_{c(j)} - x_j\|$$

j = datapoints
x_j = data
$c(j)$ = Centeroid of that data

So, min the distance
So here, fix u, optimize c

$$F(u,c) = \sum_{i=1}^{k} \sum_{c(j)=i} \|u_i - x_j\|$$

i = 1 … k is the number of centers.

fix c, optimize u

Cons:

- → Sensitive to outliers.
- → only for vector data
- → No method to choose number of cluster from cost function.

k-medoids:

⊙ recenter

$$c_i^{(t+1)} \leftarrow \underset{c(j)=k}{\operatorname{argmin}} \sum \|x_j - x_i\|^2$$

properties:

higher value of cost.

$$F(M, C) = \sum_{j=1}$$

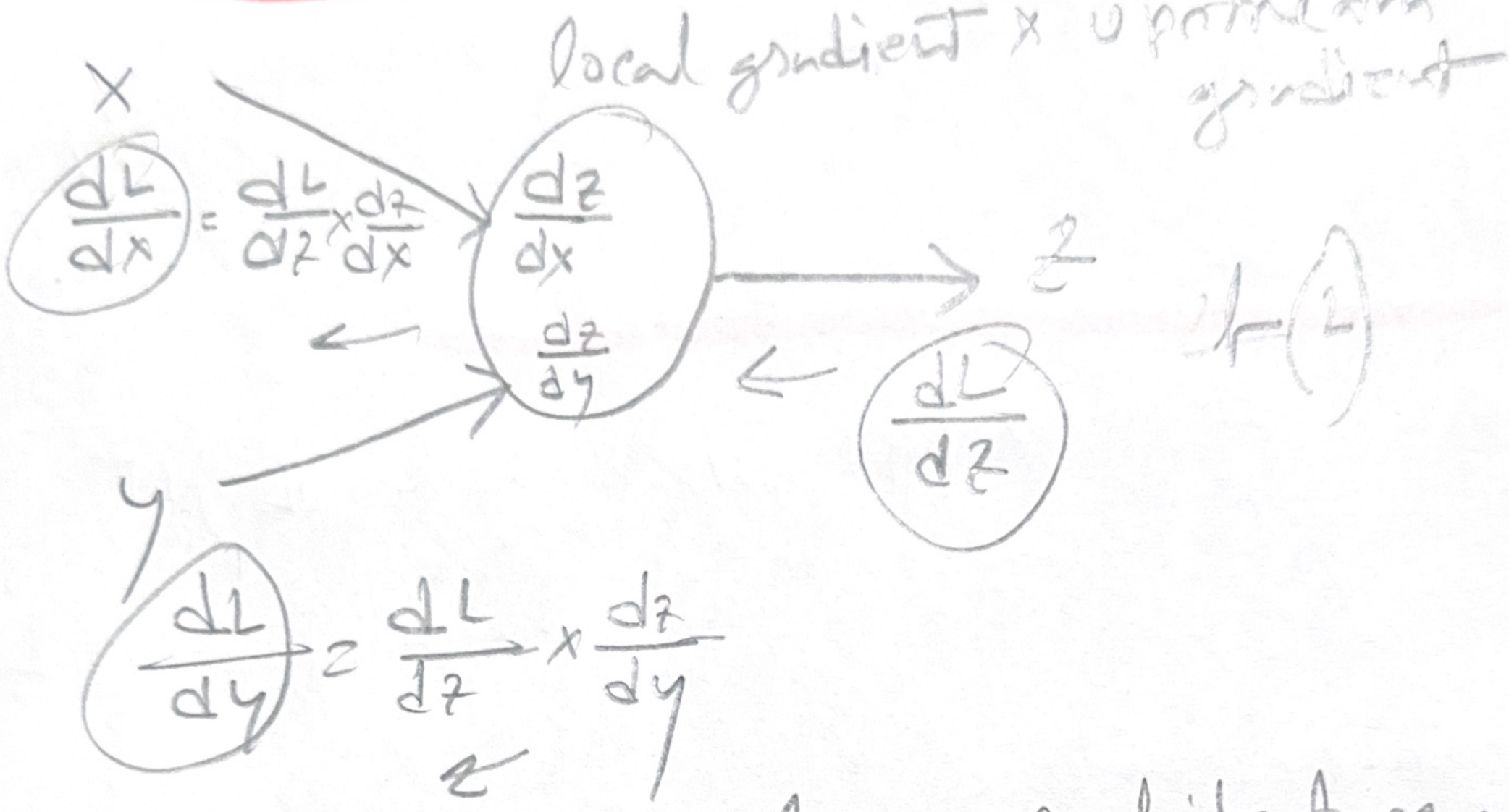
Soft k-means:

probability of point belongs to cluster

$$p(x, c_i) = \frac{\exp\{-d(x, c_i)\}}{\sum_j \exp\{-d(x, c_i)\}}$$

$$c_i = \frac{\sum_x x\, p(x, c_i)}{\sum_x p(x, c_i)}$$ weighted center.

Neural Network:



Activation for layer Architecture:

$$t = W_1 x$$
$$t_2 = W_2 \max(0, W_1 x)$$

Activation function:

Sigmo. $\sigma(x) = \frac{1}{1+e^{-x}}$, $\frac{d\sigma(x)}{dx} = \sigma(x)(1-\sigma(x))$

ReLU $t(x) = \max(0, x)$, $\begin{cases} t'(x) = 0 & x \leq 0 \\ t'(x) = 1 & x > 0 \end{cases}$

LRelu $t(x) = \max(0.01x, x)$, $\begin{cases} t'(x) = 0.01x & x \leq 0 \\ t'(x) = 1 & x > 0 \end{cases}$

ELU $= t(x) = \begin{cases} x, & x > 0 \\ \alpha(\exp(x) - 1) & x \leq 0 \end{cases}$ $t'(x) = 1 \quad x > 0$

Maxout $t(x) = \max(w_1^T b_1 + b_1 + w_2^T x_2 + b_2)$

output volume size: (conv)

$$W_2 = \frac{W_1 - F + 2P}{S} + 1 \times \text{Number of kernel/filter}$$

Number of parameter: (conv)

size $F \times F \times (\text{channel} + 1b) \times K$

common K = power of 2

torch.nn.conv2d

common $F=2, S=2$

conv,

$F=3, S=2$

Input,

$W_1 \times H_1 \times D_1$

output,

$$W_2 = \frac{W_1 - F + 2P}{S} + 1$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1$$

$D_2 = K$ = Number of filter

for pooling

Input, $W_1 \times H_1 \times D_1$

$$W_2 = \frac{W_1 - F}{S} + 1$$

$$H_2 = \frac{H_1 - F}{S} + 1$$

$D_2 = D_1$