CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Mathematical Foundations

Table of contents

Basic Linear Algebra

- Given a vector **x** of m dimensions, the transpose $\mathbf{x}^t$ is -

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 & . & . & . & . & . & . & x_m \end{bmatrix} \quad \mathbf{x}^t = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ . \\ x_m \end{bmatrix}$$

Basic Linear Algebra - Determinant

- Given a 2x2 matrix **A**, the determinant **|A|** is defined as -

$$\mathrm{A} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} = a_{00}a_{11} - a_{01}a_{10}$$

Basic Linear Algebra - Determinant

- Given a 2x2 matrix **A**, the determinant **|A|** is defined as -

$$\mathrm{A} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} = a_{00}a_{11} - a_{01}a_{10}$$

- Given a 3x3 matrix **A**, the determinant **|A|** is defined as -

$$\mathrm{A} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} = a_{00}(a_{11}a_{22} - a_{12}a_{21}) - a_{01}(a_{10}a_{22} - a_{20}a_{12}) + a_{02}(a_{10}a_{21} - a_{11}a_{20})$$

Determinant - Examples

- What is the determinant of A = $\begin{bmatrix} 1 & 7 \\ 2 & -4 \end{bmatrix}$

Determinant - Examples

- What is the determinant of A = $\begin{bmatrix} 1 & 7 \\ 2 & -4 \end{bmatrix}$

$$|A| = 1 \times -4 - 7 \times 2 = -18$$

Determinant - Examples

- What is the determinant of A = $\begin{bmatrix} 1 & 7 \\ 2 & -4 \end{bmatrix}$

$$|A| = 1 \times -4 - 7 \times 2 = -18$$

- What is the determinant of A = $\begin{bmatrix} 2 & 3 & 7 \\ -3 & 4 & 0 \\ 1 & -1 & 6 \end{bmatrix}$

Determinant - Examples

- What is the determinant of A = $\begin{bmatrix} 1 & 7 \\ 2 & -4 \end{bmatrix}$

$$|A| = 1 \times -4 - 7 \times 2 = -18$$

- What is the determinant of A = $\begin{bmatrix} 2 & 3 & 7 \\ -3 & 4 & 0 \\ 1 & -1 & 6 \end{bmatrix}$

$$|A| = 2(4 \times 6 - 0 \times -1) - 3(-3 \times 6 - 1 \times 0) + 7(-3 \times -1 - 4 \times 1) = 95$$

Basic Linear algebra - Inverse of a matrix

- For a matrix A, inverse is denoted by A^{-1}

Basic Linear algebra - Inverse of a matrix

- For a matrix A, inverse is denoted by A^{-1}
- $AA^{-1} = A^{-1}A = I$

Basic Linear algebra - Inverse of a matrix

- For a matrix A, inverse is denoted by A^{-1}
- $AA^{-1} = A^{-1}A = I$
- Why is an inverse of a matrix needed?
    - Because matrices cannot be divided!

Inverse of a matrix - Example 1

Q. Calculate the inverse of P = $\begin{bmatrix} 1 & 2 \\ 3 & -5 \end{bmatrix}$

Inverse of a matrix - Example 1

Q. Calculate the inverse of P = $\begin{bmatrix} 1 & 2 \\ 3 & -5 \end{bmatrix}$

Step 1: Calculate the determinant. Here, |P| = -11

Inverse of a matrix - Example 1

Q. Calculate the inverse of P = $\begin{bmatrix} 1 & 2 \\ 3 & -5 \end{bmatrix}$

Step 1: Calculate the determinant. Here, |P| = -11

Step 2: Swap diagonal elements and add negative sign to off-diagonal elements.

$$\begin{bmatrix} -5 & -2 \\ -3 & 1 \end{bmatrix}$$

Inverse of a matrix - Example 1

Q. Calculate the inverse of P = $\begin{bmatrix} 1 & 2 \\ 3 & -5 \end{bmatrix}$

Step 1: Calculate the determinant. Here, |P| = -11

Step 2: Swap diagonal elements and add negative sign to off-diagonal elements. $\begin{bmatrix} -5 & -2 \\ -3 & 1 \end{bmatrix}$

Step 3: Divide the matrix by the determinant.

$$\begin{bmatrix} \frac{5}{11} & \frac{2}{11} \\ \frac{3}{11} & \frac{-1}{11} \end{bmatrix}$$

Inverse of a matrix - Example 2

Q. Calculate the inverse of $\begin{bmatrix} 3 & -9 \\ 2 & -6 \end{bmatrix}$

Inverse of a matrix - Example 2

Q. Calculate the inverse of $\begin{bmatrix} 3 & -9 \\ 2 & -6 \end{bmatrix}$

Solution. Here, |A| = 0. Therefore, the inverse does not exist!

Such a matrix is called **singular matrix**!

Probability

- A probability space is a triplet (Ω, $\mathcal{B}$, P) that is used to model a process or an experiment with random outcomes.

Probability

- A probability space is a triplet (Ω, $\mathcal{B}$, P) that is used to model a process or an experiment with random outcomes.
  - Ω - Sample space

Probability

- A probability space is a triplet (Ω, $\mathcal{B}$, P) that is used to model a process or an experiment with random outcomes.
  - Ω - Sample space
  - $\mathcal{B}$ - Collection of subsets of Ω

Probability

- A probability space is a triplet (Ω, $\mathcal{B}$, P) that is used to model a process or an experiment with random outcomes.
  - Ω - Sample space
  - $\mathcal{B}$ - Collection of subsets of Ω
  - P - probability

Conditional Probability

- Let $(\Omega, \mathcal{B}, P)$ be a probability space and let $H \epsilon \mathcal{B}$ with $P(H)>0$. For any $B \epsilon \mathcal{B}$, $P(B|H)$ is defined as-

$$P(B|H) = P(BH) / P(H)$$

and call $P(B|H)$ the conditional probability of B given H.

Total Probability Rule

- Let $(\Omega, \mathcal{B}, P)$ be a probability space, and let $\{H_j\}$ be pairwise disjoint events in $\mathcal{B}$ (i.e. $H_j H_k = \phi$, $\forall j \neq k$) and $\cup_{[j=1,\ldots,\infty]} H_j = \Omega$.

  Suppose $P(H_j) > 0$, $\forall j$, then,

$$P(B) = \sum_{j=1,\ldots,\infty} P(H_j) P(B|H_j)$$

Bayes Theorem

- Let $(\Omega, \mathcal{B}, P)$ be a probability space, and let $\{H_j\}$ be pairwise disjoint events in $\mathcal{B}$ (i.e. $H_j H_k = \phi$, $\forall j \neq k$) and $\cup_{[j=1,\ldots,\infty]} H_j = \Omega$. and $P(H_j)>0$, $\forall j$. We have, $\forall B \in \mathcal{B}$ and $P(B)>0$,

Bayes Theorem

- Let $(\Omega, \mathcal{B}, P)$ be a probability space, and let $\{H_j\}$ be pairwise disjoint events in $\mathcal{B}$ (i.e. $H_j H_k = \phi$, $\forall j \neq k$) and $\cup_{[j=1,\ldots,\infty]} H_j = \Omega$. and $P(H_j)>0$, $\forall j$. We have, $\forall B \in \mathcal{B}$ and $P(B)>0$,

$$P(H_j|B) = \frac{P(H_j)\, P(B|H_j)}{\sum_{i=1,\ldots,\infty} P(H_i)P(B|H_i)}, \quad \forall j$$

Bayes Theorem

- Let $(\Omega, \mathcal{B}, P)$ be a probability space, and let $\{H_j\}$ be pairwise disjoint events in $\mathcal{B}$ (i.e. $H_j H_k = \phi$, $\forall j \neq k$) and $\cup_{[j=1,\ldots,\infty]} H_j = \Omega$. and $P(H_j)>0$, $\forall j$. We have, $\forall B \in \mathcal{B}$ and $P(B)>0$,

Prior → $P(H_j)$ ← Likelihood $P(B|H_j)$

Posterior → $P(H_j|B) = \dfrac{P(H_j)\,P(B|H_j)}{\sum_{i=1,\ldots,\infty} P(H_i)P(B|H_i)}, \quad \forall j$

Evidence ← $\sum_{i=1,\ldots,\infty} P(H_i)P(B|H_i)$

Bayes Theorem

Consider two events A and B, then the joint probability is-

$$P(AB) = P(B|A)P(A)$$

➡ $$P(AB) = P(A|B)P(B)$$

➡ $$P(B|A)P(A) = P(A|B)P(B)$$

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

Bayes Theorem - Example 1

Q. A test is developed to detect a disease that 0.1% of the population have. The test is 99% effective in detecting an infected person. However, it gives a false positive result for 0.5% of cases. Find the probability that a person actually has the disease if the person tests positive?

Sol.: Let X be the event that a person has the disease & Y be the event that the test result is true. P(X) = 0.001, P(Y|X) = 0.99, P(Y|~X) = 0.005, P(~X)=1-P(X)=0.991. We need to find P(X|Y). Using Bayes theorem and total probability rule, we have

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y|X)P(X)+P(Y|\sim X)P(\sim X)}$$

Substituting the above values, we get P(X|Y) = 0.165

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Supervised Learning

Table of contents

Project Part 1 - Density Estimation and Supervised Learning

- **Due on: 17 Feb, 11:59 PM MST.**
- Binary classification using Fashion-MNIST dataset.
- Two classes - T-shirt (label 0) and Trouser (label 1)
- 28x28 grayscale images
- Training set: "Tshirt": 6000; "Trouser": 6000.
- Testing set: "Tshirt": 6000; "Trouser": 1000.

More information on Fashion-MNIST, refer https://github.com/zalandoresearch/fashion-mnist

Project Part 1 - Tasks

- Calculate two features of each image - average and standard deviation.
- Estimate parameters for 2D Gaussian distribution for each class.
- Implement Naive Bayes to perform classification
- Train Logistic regression using gradient ascent to perform classification

Project Part 1 - Deliverables

- Code:
  - Acceptable file types are .py/.m or .zip.
  - Well-commented code. Be sure to read through the directions carefully to ensure you have included all necessary parts in your code.
- Report:
  - Acceptable File types: .pdf
  - Length: 2-5 A4 pages
  - Content: Refer Canvas

Project Part 1 - Languages/Software

- What can you use?
  - Python/MATLAB to code.
  - Libraries or in-built functions to manipulate the data i.e. NumPy, SciPy libraries etc.
- What cannot be used?
  - Packages like scikit-learn which have ready-to-use algorithms.

What is supervised learning?

- Data - <sample, label> pairs
- Objective - Learn from the given data such that label can be predicted for a new data sample.
- Based on labels - → Regression
  → Classification

Density Estimation

- Estimating underlying probability density function, based on the training data
- Parametric: each class of images (the feature vectors) may be modeled by a density function $p_\theta(x)$ with parameter θ.
- Non-parametric: makes no assumption about the distribution of data or modeling the data without any parameters

Maximum Likelihood Estimation

- Given some training data and assuming a parametric model p(x|θ); what specific θ will fit/explain the data best?
- To consider all the samples denoted by D={x_1, x_2, ……, x_n}, assume that all the samples are i.i.d - independent and identically distributed.
- So, data likelihood represented by L(θ) is -

$$L(\boldsymbol{\theta}) = P(D|\boldsymbol{\theta}) = \prod_i P(x_i|\theta)$$

Maximum Likelihood Estimation

1. Likelihood function
2. Take derivative
3. Set to 0.

- Maximum Likelihood Estimation (MLE): Finding the parameter that maximizes the likelihood function

$$\hat{\theta} = \text{argmax}_{\theta} p(\mathbf{x}|\theta)$$

$$\hat{\theta} = \underset{\theta}{\text{argmax}} \prod_{i}^{n} P(x_i|\theta)$$

$$= \underset{\theta}{\text{argmax}} \log \prod_{i}^{n} P(x_i|\theta) = \underset{\theta}{\text{argmax}} \sum_{i} \log P(x_i|\theta)$$

MLE - Example 1

$\theta = \{\mu, \sigma^2\}$

Given *n* i.i.d. samples $\{x_i\}$ from the 1-D normal distribution $N(\mu, \sigma^2)$, find the MLE for μ and σ^2.

1. Likelihood func

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

$$p(D|\theta) = \prod_{i=1}^{n} p(x_i|\theta)$$

$$= \prod_{i=1}^{n}$$

$$\log p(D|\theta) = \log \prod_{i=1}^{n} p(x_i|\theta) = \log\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \sum_{i=1}^{n} \frac{(x_i-\mu)^2}{2\sigma^2}$$

$$-n \log \sigma\sqrt{2\pi}$$

(μ, σ^2)

MLE - Example 1

(2) Take derivative w.r.t μ

$$\frac{\partial}{\partial \mu} \log P(D|\theta) = 0 - \sum_{i=1}^{n} \frac{-2(x_i - \mu)}{2\sigma^2}$$

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

(3) Equate derivative to 0.

$$\sum_{i=1}^{n} \frac{x_i - \mu}{\sigma^2} = 0$$

$$\sum_{i=1}^{n} (x_i - \mu) = 0$$

$$\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \mu = 0$$

$$\sum_{i=1}^{n} x_i - n\mu = 0$$

$$\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n} \Rightarrow \text{Sample mean}$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{\mu})^2$$

Generative vs Discriminative Models

- Generative models -
  - Learn P(y) and P(x|y).
  - Ex: Bayesian classifier, Naive Bayes.
- Discriminative models -
  - Directly learn P(y|x)
  - Ex: Logistic Regression

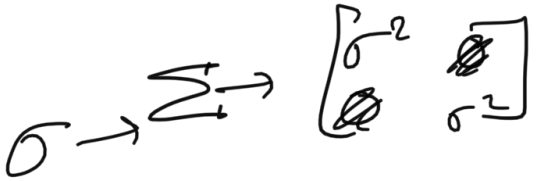
Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Supervised Learning

Table of contents

MLE - Example 1 (Continued)



Given *n* i.i.d. samples $\{x_i\}$ from the 1-D normal distribution $N(\mu, \sigma^2)$, find the MLE for μ and σ^2.

$$\log P(D|\mu,\sigma) = \underbrace{-n\log(\sigma\sqrt{2\pi})}_{-\frac{n}{2}\log(2\pi\sigma^2)} - \sum_{i=1}^{n}\frac{(x_i-\mu)^2}{2\sigma^2}$$

$$\frac{\partial}{\partial\sigma}\log P(D|\mu,\sigma) = -\frac{n}{\sigma} + \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{\sigma^3}$$

Equate it 0,

$$-\frac{n}{\sigma} + \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{\sigma^3} = 0$$

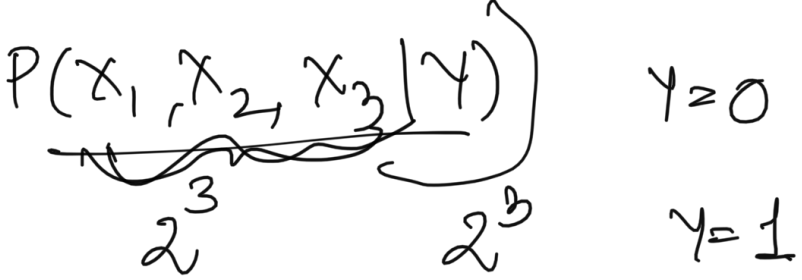
MLE - Example 1 (Continued)

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

$$\hat{\sigma}^2_{MLE} = \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{n}$$ — Biased

$$\text{Bias} = E(\hat{\sigma}^2_{MLE}) - \sigma^2 = 0$$ — Unbiased

Naive Bayes



- The "naive" conditional independence assumption: each feature is (conditionally) independent of every other feature, given the label, i.e., $p(x_i | \{x_j \text{ for any } j \neq i\}, y) = p(x_i|y)$
- How does this simplify?
  - Consider the previous example again: d-dimensional binary features, and y is also binary.

No. of probabilities $P(X_1, X_2, X_3 | Y) = 2^{3+1}$

Bayesian ⇒ d-dimensional X → $2^{d+1} = 4d$

$P(X_1, X_2, X_3 | Y) = P(X_1|Y) \cdot P(X_2|Y) \cdot P(X_3|Y)$

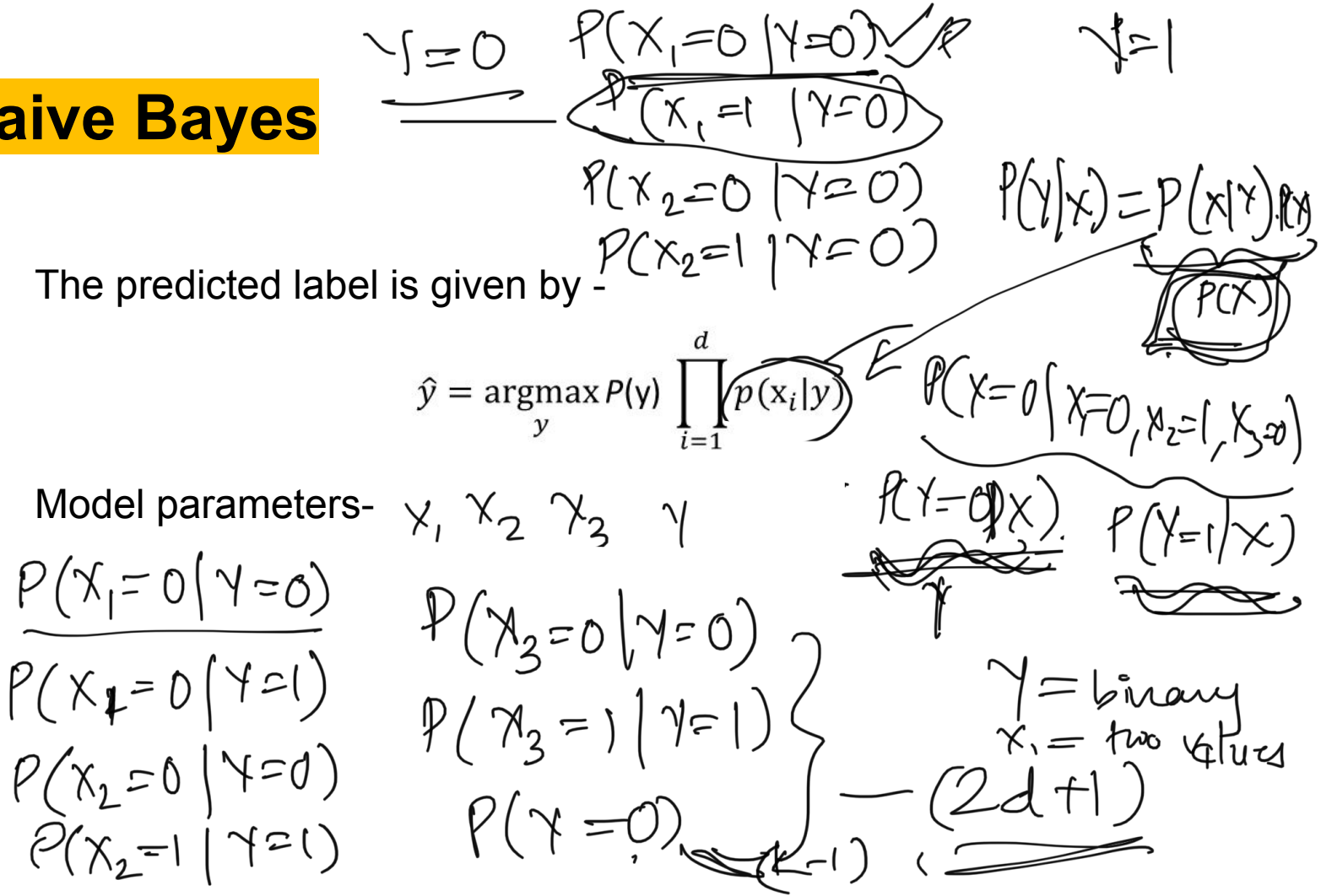
$Y=0 \rightarrow P(X_1=0|Y=0) \cdot P(X_1=1|Y=0)$ 6+6 = 12

Naive Bayes

- The predicted label is given by -

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y) \prod_{i=1}^{d} p(x_i|y)$$

- Model parameters-



Naive Bayes - Example

X						Y
x_1	x_2	x_3	x_4	x_5	x_6	
Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

$P(Y = Yes) = \frac{3}{4}$

$P(X_1 = Sunny \mid Y = Yes) = \frac{3}{3}$

$(2d+1) = 13$ independent parameter

$P(X|Y)$

Logistic Regression

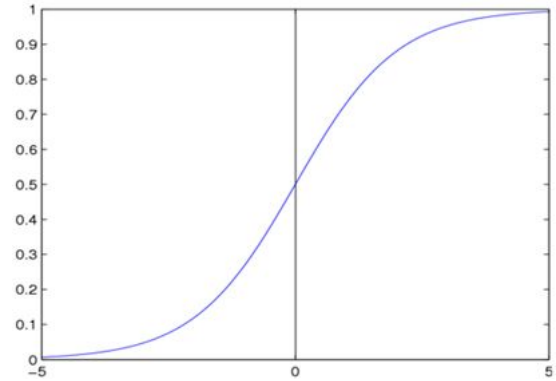
$w^T x \geq 0$

- Training set: n labelled samples <x(i), y(i)>
- Use the logistic function for modeling P(y|x), considering only the case of $y \in \{0,1\}$

$$P(y=0|\mathbf{x}) = \frac{1}{1+\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)} = 1 - \sigma(w^T x)$$

$w^T x$

$$P(y=1|\mathbf{x}) = \frac{\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)}{1+\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)} = \sigma(w^T x)$$



Sigmoid $\sigma(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t}$

Logistic Regression

$w \rightarrow$ model parameter

- Model parameters - $P(Y|X) = \left[1 - \sigma(w^T x^{(i)})\right]^{1-y} \left[\sigma(w^T x)\right]^{y}$

$$P(y^{(1)}, y^{(2)}, \ldots y^{(n)} | x^{(1)} x^{(2)} \ldots x^{(n)}) = \prod_{i=1}^{n} \left[1 - \sigma(w^T x^{(i)})\right]^{(1-y^{(i)})} \left[\sigma(w^T x^{(i)})\right]^{y^{(i)}}$$

$$\text{argmax}\ l(w) = \sum_{i=1}^{n} \left[1 - y^{(i)}\right] \log\left[1 - \sigma(w^T (x^{(i)}))\right] + y^{(i)} \log\left(\sigma(w^T x^{(i)})\right)$$

Gradient Ascent

Update Equation —

$$W_{t+1} = W_t + \eta \nabla_w L(w)$$

- W_{t+1}: Updated weights
- W_t: Current weight
- η: Learning rate
- $\nabla_w L(w)$: Gradient of the Loss function

Descent —

Gradient Ascent

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Supervised Learning

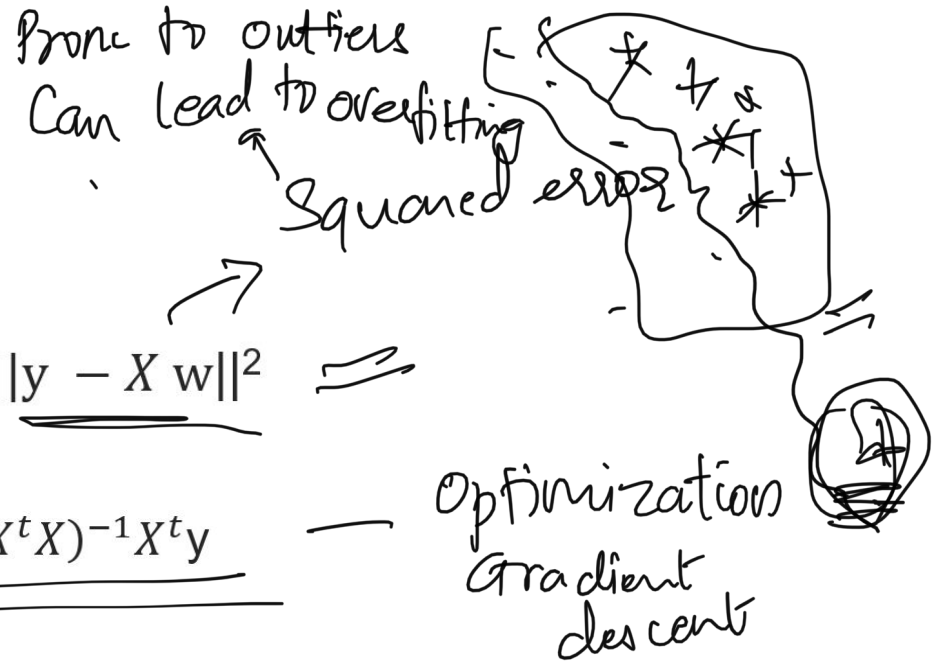
Table of contents

Linear Regression

- Regression - A training set of n samples $<x^{(i)}, y^{(i)}>$ where $y^{(i)}$ is a continuous “label” (or target value) for $x^{(i)}$
- Linear regression - modeling the relation between y and x via a linear function

$$y \approx w_0 + w_1 x_1 + \ldots + w_d x_d = \mathbf{w}^t \mathbf{x}$$

Linear Regression



- The error is given as -

$$\text{argmin} \, ||e||^2 = ||y - Xw||^2$$

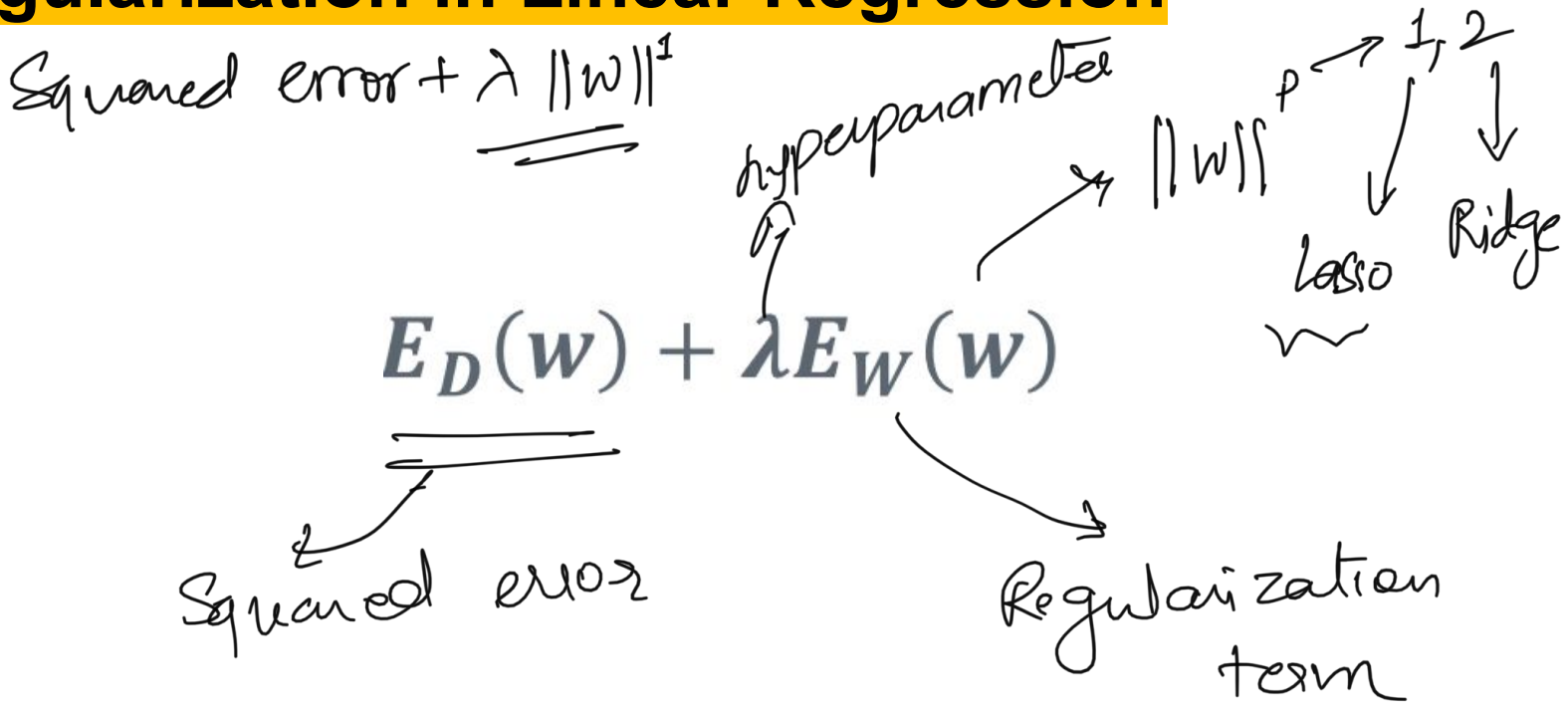
- Weights can be found using -

$$\hat{w} = (X^t X)^{-1} X^t y$$

- How to generalize?

$$y = w_0 + w_1 \phi_1(\mathbf{x}) + \ldots + w_{M-1} \phi_{M-1}(\mathbf{x})$$

Regularization in Linear Regression



Why regularize and how it works?

- Used to avoid overfitting of the network

Covariance M

[ ]

- Regularization terms shrinks the w estimates.

100 X 2

	Mean	SD
100		

Test Accuracy	Logistic	Naive
0	· %	%
1		

Questions?

Project Part-1

After feature extraction – 6000×2 (Class 0)

Training size – 12000 × 2 ≈ 784

Test data size = 2000 × 2

① Estimation parameters – np.cov

One class: $\mu = [\mu_1, \mu_2]$, $\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$

Class 1 – $\mu = [\mu_1, \mu_2]$, Σ

Class 0 – $\mu = [\mu_1, \mu_2]$, Σ

Hint: The features are independent. $\begin{bmatrix} \sigma_{11} & 0 \\ 0 & \sigma_{22} \end{bmatrix}$

Class 0 – Avg, standard deviation

Avg → 1-D Gaussian; Std → 1-D Gaussian

Class 1 – Avg → 1-D; Std → 1-D Gaussian

784 → 1

② Naive Bayes –

$P(Y|X) \sim P(X|Y) \cdot P(Y)$ (From the data)

$P(X_1|Y) \cdot P(X_2|Y) \cdot P(Y)$

$P(Y=0|x) > P(Y=1|x) \rightarrow$ Class 0

otherwise Class = 1

③ Logistic Regression – weights w $P(Y|X)$

Step 1: Initialize w randomly. $[w_0, w_1, w_2]$

Step 2: $z = \sigma(w^T x)$ $\sigma(t) = \frac{1}{1+e^{-t}}$

$w_0 + w_1 x_1 + w_2 x_2$

Step 3: Calculate the cost function.

Log-likelihood func^n –

$$L(w) = \sum_{i=1}^{m} y^{(i)} \log z^{(i)} + (1-y^{(i)}) \log(1-z^{(i)})$$

$\leftarrow e^{-8}$

Step 4: Gradient-ascent np.random

$$w_{t+1}^j = w_t^j + \eta \frac{\partial L(w)}{\partial w^j}$$

iteration

$$\frac{\partial L(w)}{\partial w^j} = \frac{\partial L(w)}{\partial z} \cdot \frac{\partial z}{\partial p} \cdot \frac{\partial p}{\partial w^j} \Rightarrow \text{Using chain rule.}$$

$$\frac{\partial L(w)}{\partial z} = \frac{y}{z} - \frac{(1-y)}{1-z} \quad -(1)$$

$$\frac{\partial z}{\partial p} = z(1-z) \quad -(2)$$

$$\frac{\partial p}{\partial w^j} = x^j \quad -(3)$$

$$\frac{\partial L(w)}{\partial w^j} = \frac{y(1-z) - z(1-y)}{z(1-z)} x^j$$

$$= (y-z)\, x^j$$

Prediction $z = \sigma(w^T x)$

$z > 0.5 \rightarrow$ Class 1

otherwise $\rightarrow$ Class 0

Covariance matrix – x_1 x_2

6000 × 2 ⇒ Class 0

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} = \begin{bmatrix} \sigma_{11} & 0 \\ 0 & \sigma_{22} \end{bmatrix}$$

12000 × 2 (x_1 x_2) – Training data

6000 × 2 Class 0 → μ_1, μ_2, Σ

6000 × 2 Class 1 → μ_1, μ_2, Σ

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Linear Machines & SVM

Table of contents

Quick Recap - Logistic Regression

- Data - X (features) and Y (labels)
- Learn P(Y|X) assuming a logistic function — Discriminative model.
- $g(x) = w^T x$ is called the linear discriminant function.

Linear Discriminant functions

- Two types of notations-

$$g(x) = w^T x \quad \text{or} \quad g(x) = w^T x + w_0$$

w : weights, w_0 : bias / threshold

- For 2-class problem, learn **w** such that -

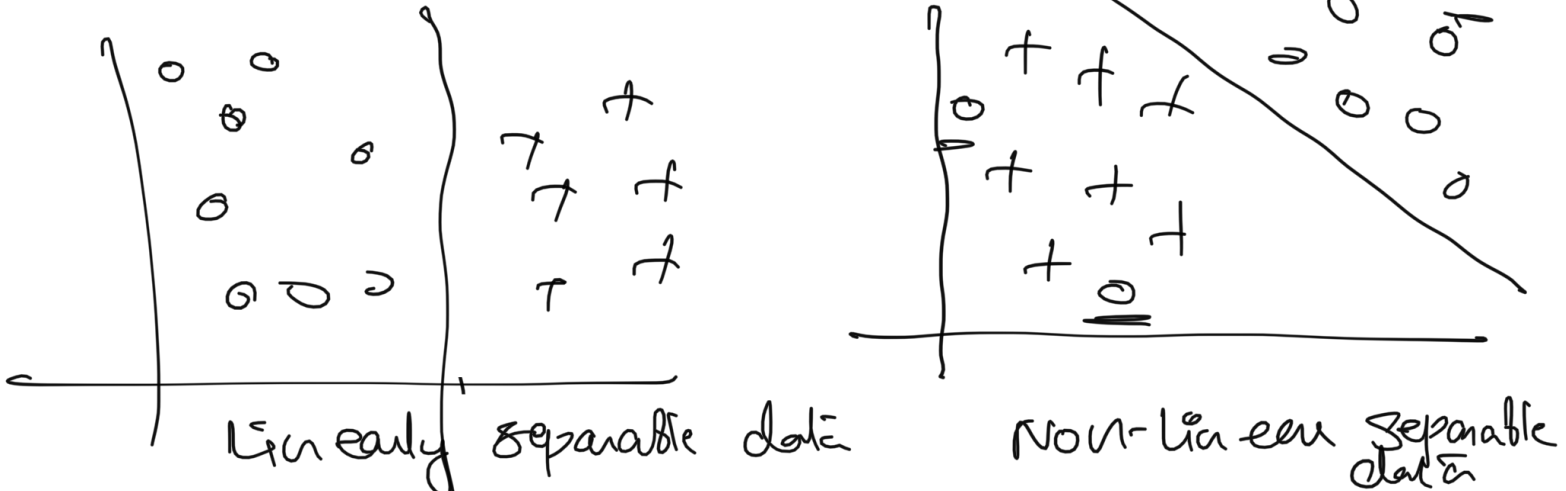
$w^T x \geq 0 \rightarrow$ Class 1

$w^T x \leq 0 \rightarrow$ Class 0.

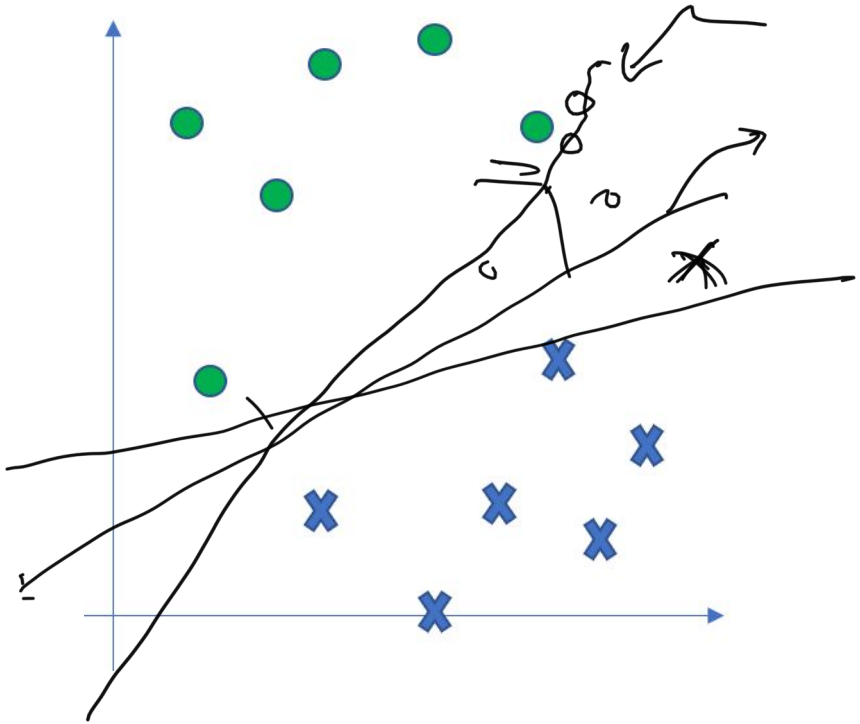
What is Linear Separability?

$w^T k$

- If there is atleast one solution of **w** such that g(x) classifies all the samples in the training data, then the data is said to be linearly separable.



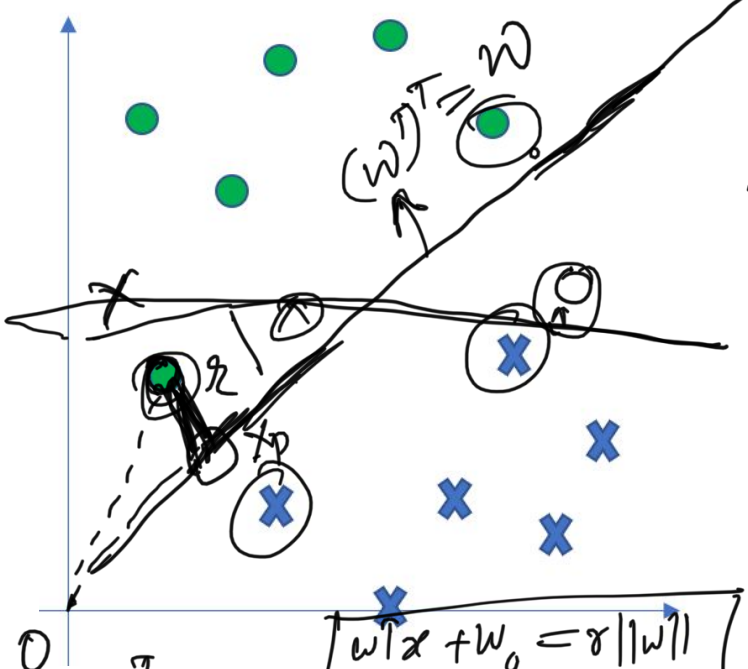
Is the solution unique?



Concept of Margin

$g(x) = w^T x + w_0 = 0$

$z = \frac{g(x)}{\|w\|}$



$\frac{w}{\|w\|}$

$x = x_p + r\frac{w}{\|w\|}$

$x_p = x - r\frac{w}{\|w\|}$ — (1)

$g(x_p) = 0$

$w^T x_p + w_0 = 0$

$w^T\left(x - r\frac{w}{\|w\|}\right) + w_0 = 0$

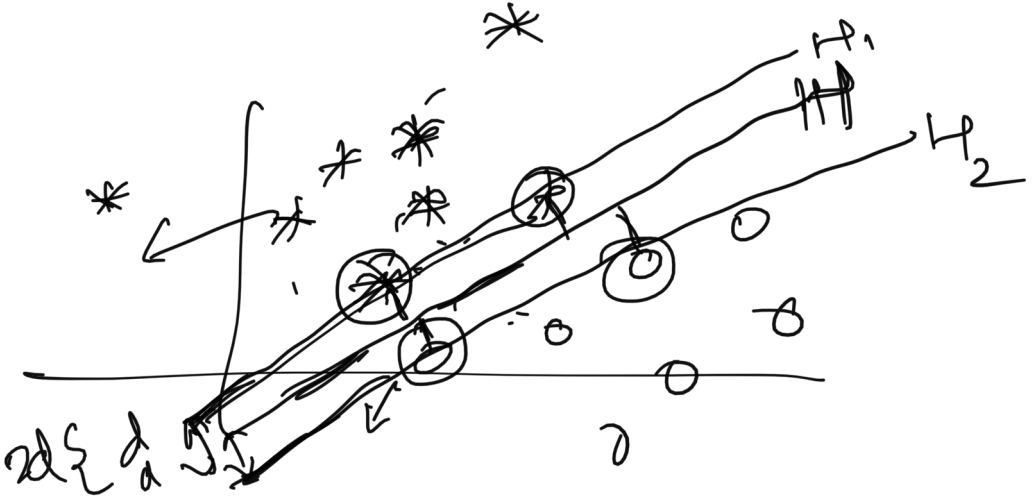
$w^T x - r\frac{w^T w}{\|w\|} + w_0 = 0$

$w^T x + w_0 = r\|w\|$

$g(x) = r\|w\|$

Concept of Margin

- Margin of a sample x (w.r.t. the decision plane) is defined as the distance from x to the plane.
- For a classifier, the margin should be as large as possible for better performance.

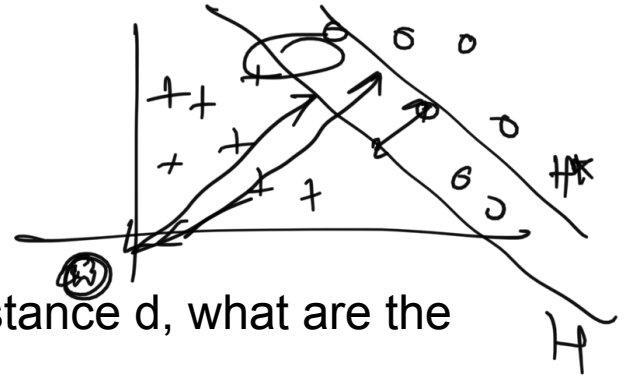


Support Vector Machines (SVM)

⇒ maximum margin classifier.

- Key idea - To find the decision boundary such that the margin is maximized.
- Data - $\langle x^{(i)}, y^{(i)} \rangle$, $y^{(i)} \in \{-1,1\}$, $x^{(i)} \in \boldsymbol{R}^d$, for all i=1,…,n
- Assume that the data is linearly separable

SVM - Problem Formulation



- Given separating plane H: $w^t x + b = 0$ and distance d, what are the equations for H_1 and H_2?
- Consider H* plane with equation - $H^* = w^T x + b = \|w\| d$

$$d(\text{origin}, H) = \frac{g(0)}{\|w\|} = \frac{b}{\|w\|}$$

$$d(\text{origin}, H^*) = \frac{g^*(0)}{\|w\|} = \frac{b - \|w\| d}{\|w\|}$$

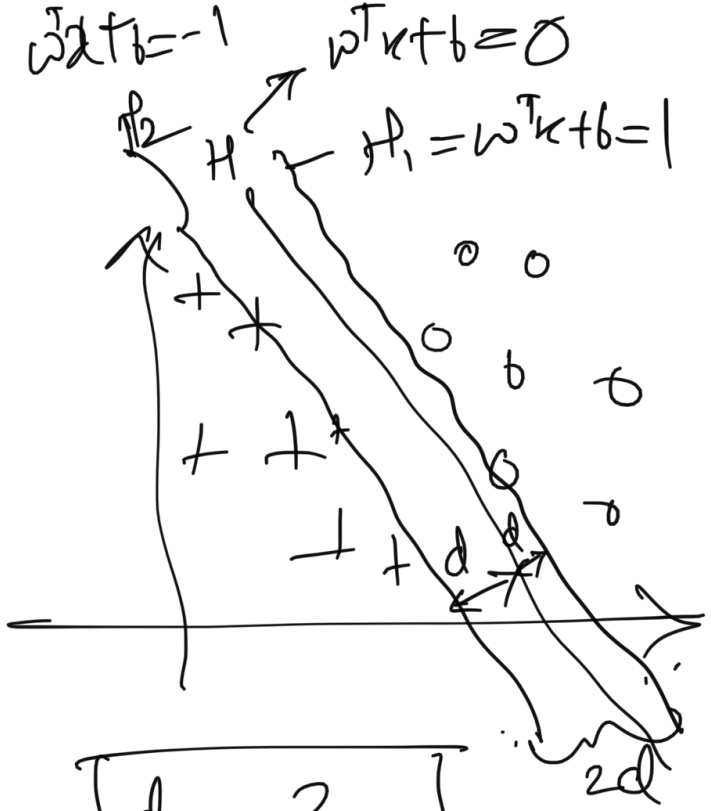
$$d(H, H^*) = d$$

SVM - Problem Formulation

$$H_1 = w^T x + b = \|w\| d$$

$$H_2 = w^T x + b = -\|w\| d$$

$$H = w^T x + b = 0$$



$$\begin{cases} H_1 = w^T x + b = 1 \\ H_2 = w^T x + b = -1 \\ H = w^T x + b = 0 \end{cases}$$

$$\boxed{d = \frac{2}{\|w\|}}$$

SVM - Problem Formulation

- Plane equations-

H_1

H_2

H

- Margin -

$d = \frac{2}{\|w\|}$ $\quad d = \frac{2}{\|w\|}$

SVM - Problem Formulation

$$\{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \|\mathbf{w}\| \quad or \quad \{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{w}\|^2$$

Subject to

$$\mathbf{w}^t\mathbf{x}^{(i)} + b \geq 1 \quad \text{for } y^{(i)} = +1$$

$$\mathbf{w}^t\mathbf{x}^{(i)} + b \leq -1 \quad \text{for } y^{(i)} = -1$$

The constraints can be combined into:

$$y^{(i)}(\mathbf{w}^t\mathbf{x}^{(i)} + b) - 1 \geq 0 \quad \forall i$$

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Linear Machines & SVM

Table of contents

SVM - Problem Formulation

$$d = \frac{2}{\|w\|}$$

$$\{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \|\mathbf{w}\| \quad or \quad \{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{w}\|^2$$

Subject to

$$\mathbf{w}^t\mathbf{x}^{(i)} + b \geq 1 \quad \text{for } y^{(i)} = +1$$

$$\mathbf{w}^t\mathbf{x}^{(i)} + b \leq -1 \quad \text{for } y^{(i)} = -1$$

The constraints can be combined into:

$$y^{(i)}(\mathbf{w}^t\mathbf{x}^{(i)} + b) - 1 \geq 0 \quad \forall i$$

SVM Lagrangian Dual Formulation

$$L(w, b, \alpha) = \frac{1}{2}\underbrace{\|w\|^2}_{w^T w} - \sum_{i=1}^{n} \alpha_i \left[y^{(i)} (w^T x^{(i)} + b) - 1 \right] \quad \text{(1)}$$

↖ Lagrangian multipliers

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} \Rightarrow w = \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} \quad \text{(2)}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0 \quad \text{(3)}$$

Plugging (2) in (1),

$$L(w, b, \alpha) = \frac{1}{2} \sum_{i=1}^{n} \alpha_i y^{(i)} (x^{(i)})^T \sum_{j=1}^{n} \alpha_j y^{(j)} (x^{(j)})$$

For more on Lagrangian Formulation, refer (from page 11): http://cs229.stanford.edu/notes2020fall/notes2020fall/cs229-notes3.pdf

SVM Lagrangian Dual Formulation

$$-\sum_{i=1}^{n}\alpha_i y^{(i)}(x^{(i)})^T \sum_{j=1}^{n}\alpha_j y^{(j)} x^{(j)} - \sum_{i=1}^{n}\alpha_i y^{(i)} b = 0$$

$$+\sum_{i=1}^{n}\alpha_i$$

$$= -\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^T x^{(j)} + \sum_{i=1}^{n}\alpha_i .$$

$$L(w,b,\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y^{(i)}y^{(j)}\langle x^{(i)}, x^{(j)}\rangle$$

For more on Lagrangian Formulation, refer (from page 11): http://cs229.stanford.edu/notes2020fall/notes2020fall/cs229-notes3.pdf

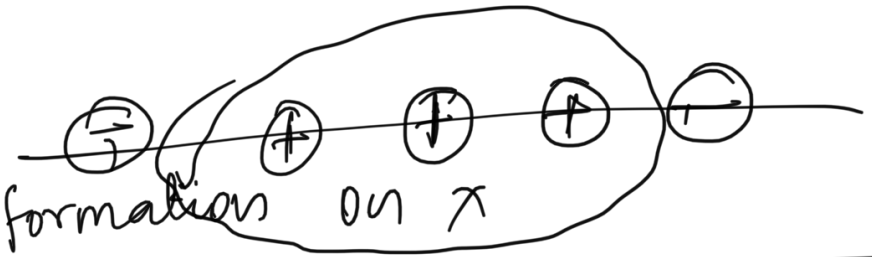
$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = x_1 y_1 + x_2 y_2 + x_3 y_3 = \bar{x} \cdot \bar{y} \cdot \cos\theta$$

What if data is not linearly separable?

$2 \rightarrow 4$

$4 \rightarrow 16$

$8 \rightarrow 64$



$\phi(x)$ — Transformation on x

$$L(w, b, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} \langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$$

$K(x^{(i)}, x^{(j)})$ — similarity between data points

What if data is not linearly separable?

- Kernel trick!

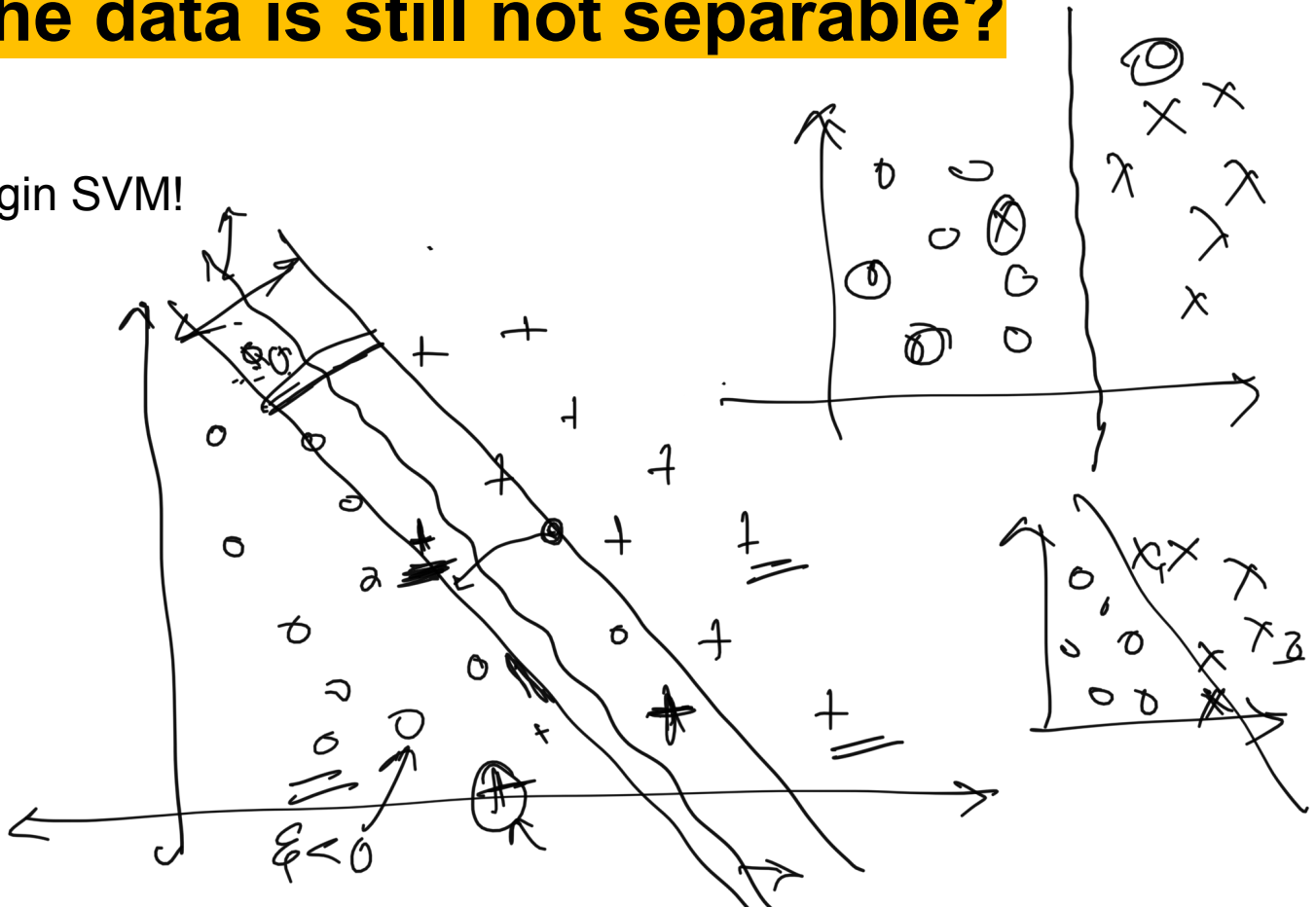
$$L(w, b, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)})$$

↳ Reduces computation

↳ Don't have to find the transformation ϕ

What if the data is still not separable?

- Soft-margin SVM!



Soft-Margin SVM Formulation

$$\{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2 + C(\textstyle\sum_i \xi_i)$$

subject to

$$w^t x^{(i)} + b \geq 1 - \xi_i \quad \text{for } y^{(i)} = +1$$

$$w^t x^{(i)} + b \leq -1 + \xi_i \quad \text{for } y^{(i)} = -1$$

$$\xi_i \geq 0, \forall i$$

Effect of C parameter

C ⇒ Infinity ⇒ Hard-margin SVM

C = 0 ⇒  Decision boundary with very high misclassification.

Will SVM overfit?

↳ Robust to overfitting.

How to overcome overfitting?

- Tune C parameter.
- Tune the parameters used in kernels.
- Change the kernels

Difference between SVM and Logistic Regression

$$y = \frac{1}{1 + e^{wx}}$$

0-1

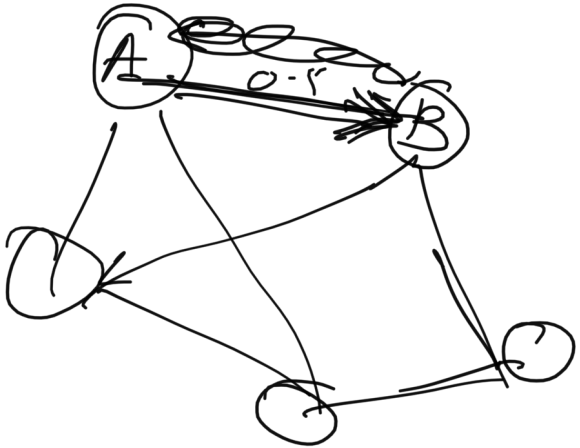
Support Vector Machines	Logistic Regression
(1) Margin, no probabilities	(1) Probabilities
(2) Sparse solution	(2) Not sparse
(3) Robust to overfitting	(3) Can overfit
(4) Hinge loss	(4) Log likelihood loss

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

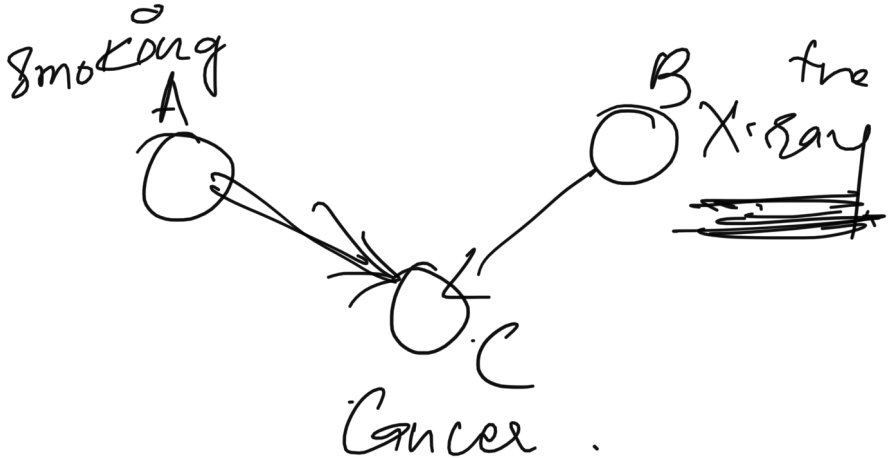
Graphical Models

Table of contents

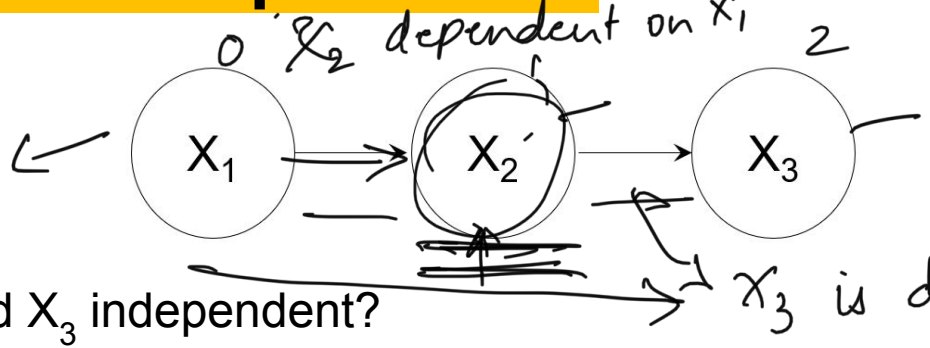


Bayesian Networks

- A probabilistic graphical model representing a set of variables and their conditional dependencies via a directed acyclic graph.



Conditional Independence



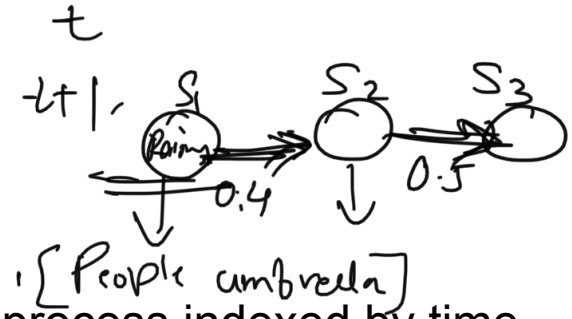
- Are X_1 and X_3 independent?

— No.

- Are they conditionally independent given X_2?

— Yes.

Hidden Markov Model (HMM)

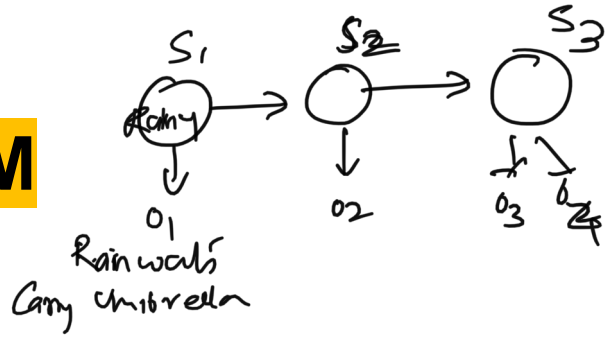


- Dynamic Bayesian network - modeling the process indexed by time.
- Two assumptions -
  - First-order Markov chain - $P(s^t=S_j \mid s^{t-1}=S_i)$
  - $a_{ij} = P(s^t=S_j \mid s^{t-1}=S_i)$, $1 \le i,j \le N$, for any t

$P(S_3 \mid S_2) = 0.5$

Stationary.

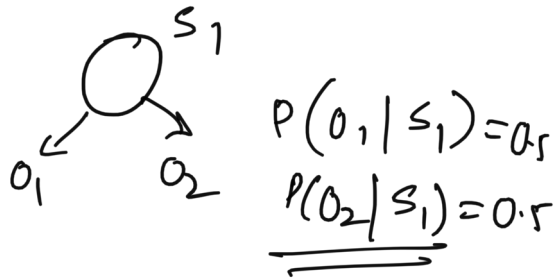
Specifying HMM



- Θ - set of hidden states $= \{S_1, S_2, S_3\}$
- Ω - set of output observations $= \{O_1, O_2, O_3, O_4\}$
- π - initial state distribution
  - $\pi = \{\pi_i\}$, $\pi_i = P(s^1 = S_i)$ $= S_2$

$\{0, 1, 0\}$ $P(S_0 = S_2) = 1$

Specifying HMM



- State transition matrix - A — Shape of A = $n \times n$
  - $A=\{a_{ij}\}$ where $a_{ij} = P(s^t=S_j \mid s^{t-1}=S_i)$ for $1 \le i,j \le N$
- Observation/emission probability matrix B
  - $B = \{b_{jk}\} = P(o^t=v_k \text{ at } t | s^t=S_j)$ where v_k is the k^{th} symbol in Ω

$n \rightarrow$ states K – observations

	o_1	o_2	K
s_1	0.5	0.5	
n			

Problems in HMM

- For a given HMM $\Lambda=\{\Theta,\Omega,A,B,\pi\}$
  - Estimation of model parameters
  - Given an observation sequence **O**= $\{o^1,o^2, \dots ,o^k\}$, what is the most likely state sequence S = $\{s^1, s^2, \dots ,s^k\}$ that has produced O?
  - How likely is an observation O?

HMM Parameter Estimation

$S_1(O_1) - S_2(O_2)$

- Given labeled data - state and observation

$S_1(O_1) \rightarrow S_2(O_2) \rightarrow S_3(O_1)$ —

$$a(S_i|S_j) = \frac{\text{number of } (s^t = S_i, s^{t-1} = S_j)}{\text{number of } S_j} =$$

transition probability

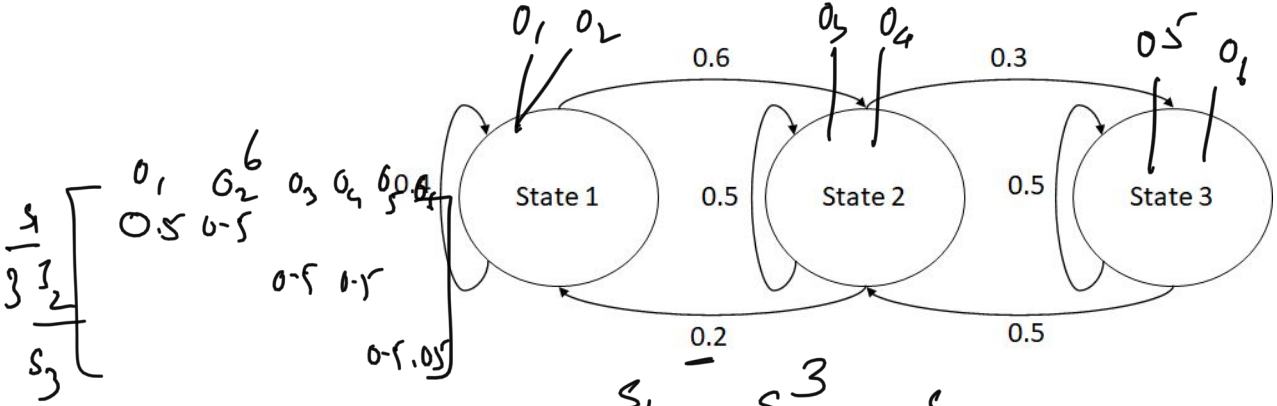
$$b(o_r|S_j) = \frac{\text{number of } (o^t = o_r, s^t = S_j)}{\text{number of } S_j}$$

emission probability

Sta. Given data – sequence of observations.

– Forward-Backward Algorithm.

HMM Parameter Estimation Example



$$A = \begin{array}{c} \\ S_1 \\ S_2 \\ S_3 \end{array} \begin{array}{c} \begin{array}{ccc} S_1 & S_2 & S_3 \end{array} \\ \begin{bmatrix} 0.4 & 0.6 & 0 \\ 0.2 & 0.5 & 0.3 \\ 0 & 0.5 & 0.5 \end{bmatrix} \end{array} = 1$$

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Graphical Models

Table of contents

Problems in HMM

- For a given HMM $\Lambda=\{\Theta,\Omega,A,B,\pi\}$
  - Estimation of model parameters
  - Given an observation sequence **O**= $\{o^1,o^2, \dots ,o^k\}$, what is the most likely state sequence S = $\{s^1, s^2, \dots ,s^k\}$ that has produced O?
  - How likely is an observation O?

HMM Parameter Estimation

- Given labeled data - state and observation

A

$$t(S_i|S_j) = \frac{\text{number of } (s^t = S_i, s^{t-1} = S_j)}{\text{number of } S_j}$$

B

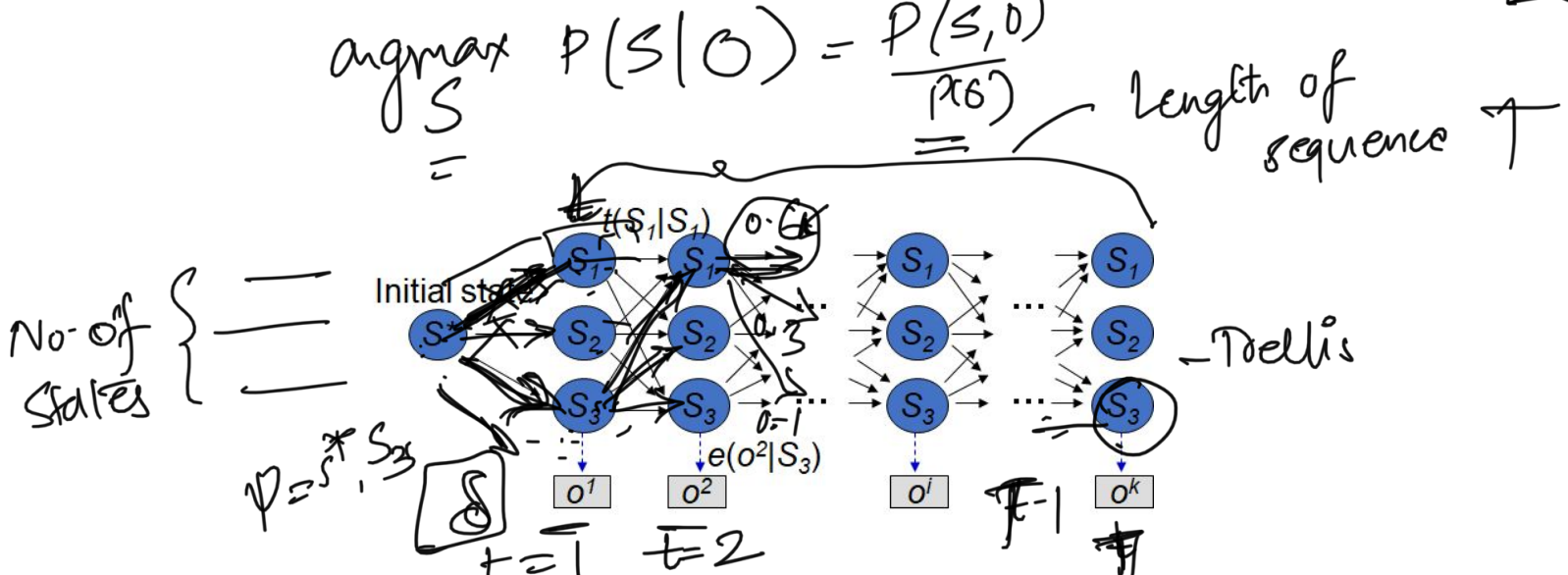
$$e(o_r|S_j) = \frac{\text{number of } (o^t = o_r, s^t = S_j)}{\text{number of } S_j}$$

- Given observation sequences -

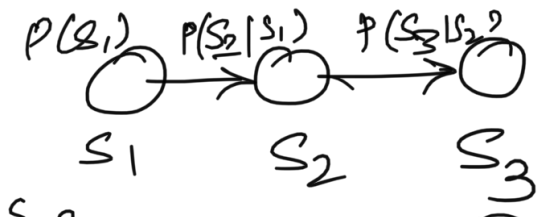
Forward-Backward Algorithm - Expectation-Maximization Approach.

HMM State Estimation

- Given an observation (sequence) O= {o_1, o_2, … ,o_k}, what is the most likely state sequence S={s_1, s_2, … ,s_k} that has produced O?



HMM State Estimation



$$P(S,O) = P(O|S) \cdot P(S)$$

$$= P(O^1 \dots O^k | S_1 \dots S^k) \prod_{j=1}^{k} P(S^j | S^1 \dots S^k)$$

$$= \prod_{i=1}^{k} P(O^i | O^1 \dots O^k, S^1 \dots S^k) \prod_{j=1}^{k} P(S^j | S^1 \dots S^k)$$

$$= \prod_{i=1}^{k} P(O^i | S^i) \prod_{j=1}^{k} P(S^j | S^{j-1})$$

Emission probability

$$P(S,O) = \prod_{i=1}^{k} P(O^i | S^i) \prod_{j=1}^{k} P(S^j | S^{j-1})$$

Transition probability

HMM State Estimation - Viterbi Algorithm

Initialization

$$\delta_{S_i}(1) = t(S_i|s^*)e(o^1|S_i), \quad \forall S_i \in \Theta$$

Induction:

for 2≤*t*≤*k*, do

$$\delta_{S_i}(t) = \max_{S_j} t(S_i|S_j)e(o^t|S_i)\delta_{S_j}(t-1)$$

$$\psi_{S_i}(t) = \operatorname{argmax}_{S_j} t(S_i|S_j)e(o^t|S_i)\delta_{S_j}(t-1)$$

Termination:

- The probability of the best state sequence: $\max_{S_j}\delta_{S_j}(k)$
- The best last state: $\hat{s}^k = \operatorname{argmax}_{S_j} \delta_{S_j}(k)$
- Back trace to get other states:

$$\hat{s}^t = \psi_{\hat{s}^{t+1}}(t), \text{ for } t = k-1, \ldots, 1.$$

HMM - Evaluate P(O)

- How likely is an observation O? $O = \{O_1 \ldots O_K\}$

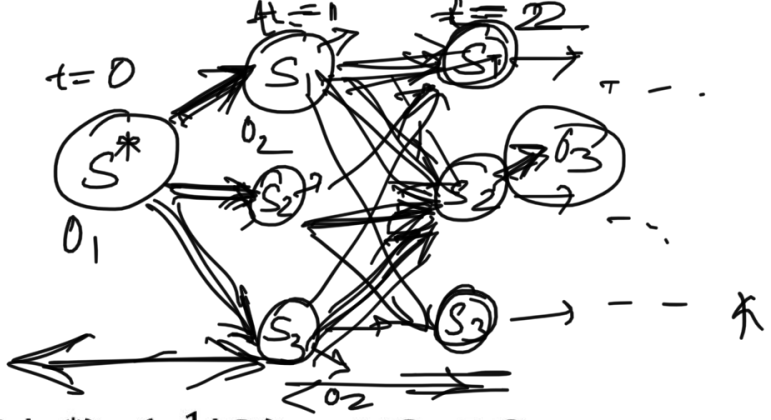
$$P(O) = \sum_S P(S, O)$$

— forward Algorithm — α $(t-1)$ at t

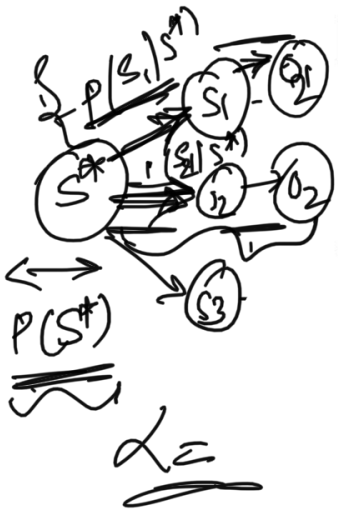
— Backward Algorithm — β $(t+1)$ to T at t^{th} step

HMM - Evaluate P(O)

- Forward Algorithm



Initialization: $\alpha_{S_i}(1) = t(S_i|s^*)e(o^1|S_i), \quad \forall S_i \in \Theta$



Induction:
for $2 \le t \le k$, do

$$\alpha_{S_i}(t) = \sum_{S_j} t(S_i|S_j)e(o^t|S_i)\alpha_{S_j}(t-1)$$

Termination:

$$P(\boldsymbol{O}) = \sum_{S_j} \alpha_{S_j}(k)$$

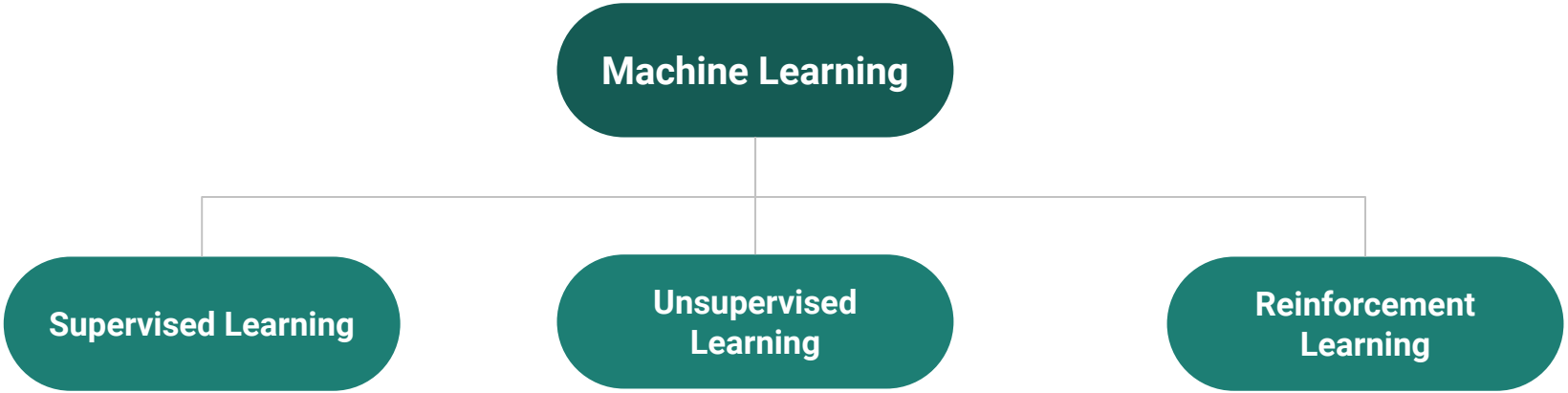
Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Midterm Review

Table of contents

Types of Machine Learning



Bias – $E[\hat{\theta}] - \theta = 0$ $E[\hat{\theta}] = \theta$

estimated parameter is unbiased

Maximum Likelihood Estimation

Ex – $E[\hat{\mu}] = \mu \Rightarrow$ Unbiased

$\neq \mu \Rightarrow$ biased.

$E[x] = \int x\, p(x)\, dx$

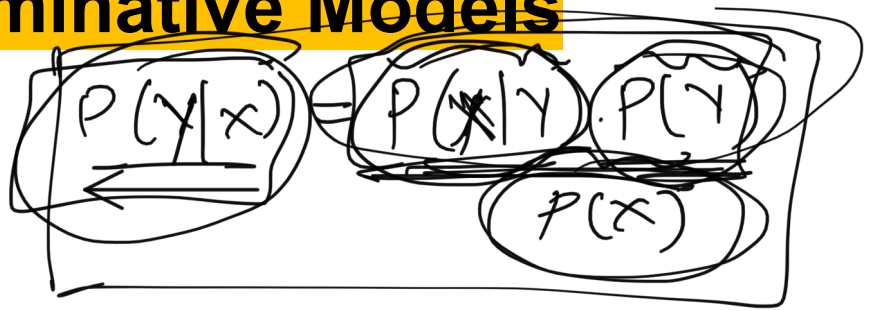
- Given some training data and assuming a parametric model p(x|θ); what specific θ will fit/explain the data best?
- To consider all the samples denoted by D={x_1, x_2, ……, x_n}, assume that all the samples are i.i.d - independent and identically distributed.
- So, data likelihood represented by L(θ) is -

$$L(\boldsymbol{\theta}) = P(D|\boldsymbol{\theta}) = \prod_i P(x_i|\theta)$$

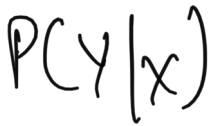
$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}}\, p(\mathbf{x}|\boldsymbol{\theta})$$

$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i$

Generative vs Discriminative Models



- Generative models -
  - Learn P(y) and P(x|y).
  - Ex: Bayesian classifier, Naive Bayes.
- Discriminative models -
  - Directly learn P(y|x)
  - Ex: Logistic Regression



Naive Bayes — Supervised

- The "naive" conditional independence assumption: each feature is (conditionally) independent of every other feature, given the label, i.e., $p(x_i \mid \{x_j \text{ for any } j \neq i\}, y) = p(x_i \mid y)$
- The predicted label is given by -

$$\hat{y} = \underset{y}{\operatorname{argmax}} P(y) \prod_{i=1}^{d} p(x_i \mid y)$$

— Predicted label.

Linear Regression

- Regression - A training set of n samples $<x^{(i)}, y^{(i)}>$ where $y^{(i)}$ is a continuous “label” (or target value) for $x^{(i)}$
- Linear regression - modeling the relation between y and x via a linear function $y \approx w_0 + w_1 x_1 + \ldots + w_d x_d = \mathbf{w}^t \mathbf{x}$
- The error is given as - $||e||^2 = ||y - Xw||^2$

Logistic Regression

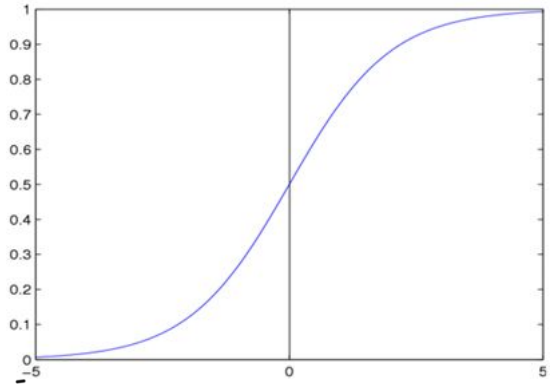
$P(Y|X) = \text{logistic function}(w^T x)$

$w^T x = 0$

- Training set: n labelled samples <x(i), y(i)>
- Use the logistic function for modeling P(y|x), considering only the case of $y \in \{0,1\}$

$$P(y=0|\mathbf{x}) = \frac{1}{1+\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)}$$

$$P(y=1|\mathbf{x}) = \frac{\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)}{1+\exp\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)}$$



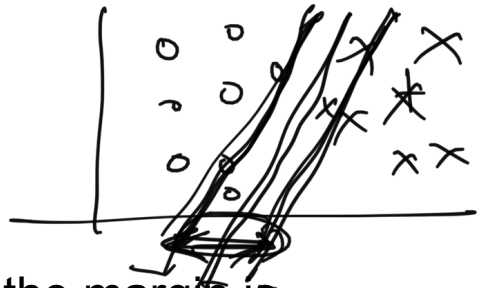
$$\sigma(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t}$$

→ Gradient ascent

$$w_{t+1} = w_t + \alpha \frac{\partial \bar{L}(w)}{\partial w}$$

(updated, previous, loss)

Support Vector Machines (SVM)



- Key idea - To find the decision boundary such that the margin is maximized.
- Data - $\langle x^{(i)}, y^{(i)} \rangle$, $y^{(i)} \in \{-1,1\}$, $x^{(i)} \in \boldsymbol{R}^d$, for all i=1,…,n
- Plane equations-

$$\left.\begin{aligned} w^T x + b &= 1 \\ w^T x + b &= -1 \end{aligned}\right\} \text{ – Margin hyperplanes.}$$

$$w^T x + b = 0 \quad \text{— Decision boundary.}$$

- Margin -

$$d = \frac{2}{\|w\|}$$

SVM - Problem Formulation

$$\{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \|\mathbf{w}\| \quad or \quad \{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{w}\|^2$$

Subject to

$$\begin{cases} \mathbf{w}^t\mathbf{x}^{(i)} + b \geq 1 & \text{for } y^{(i)} = +1 \\ \mathbf{w}^t\mathbf{x}^{(i)} + b \leq -1 & \text{for } y^{(i)} = -1 \end{cases}$$

The constraints can be combined into:

$$y^{(i)}(\mathbf{w}^t\mathbf{x}^{(i)} + b) - 1 \geq 0 \quad \forall i$$

Soft-Margin SVM Formulation

$$\{\mathbf{w}^*, b^*\} = \underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_i \xi_i\right)$$

Parameter controls the penalty.

Slack variable

Very high values → Hard margin SVM

Very low values → Larger misclassifications

subject to

$$w^t x^{(i)} + b \geq 1 - \xi_i \quad \text{for } y^{(i)} = +1$$

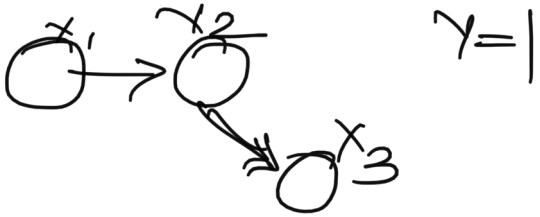
$$w^t x^{(i)} + b \leq -1 + \xi_i \quad \text{for } y^{(i)} = -1$$

$$\xi_i \geq 0, \forall i$$

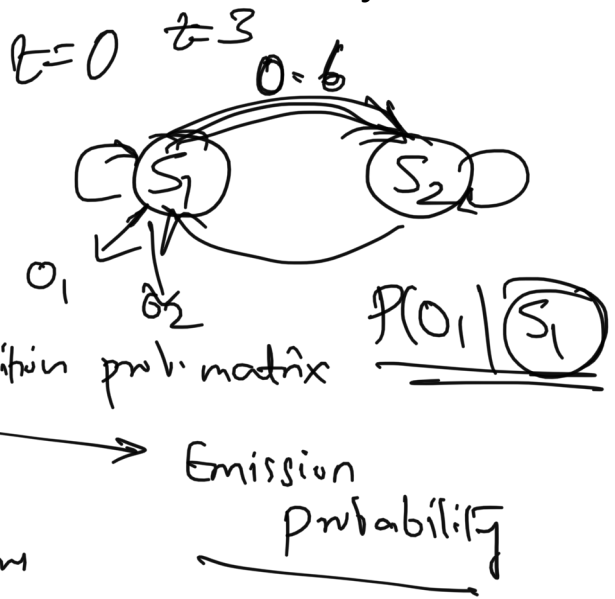
ϕ

$\langle x_i, x_j \rangle$

Hidden Markov Model (HMM)



- Dynamic Bayesian network - modeling the process indexed by time.
- Two assumptions -
  - First-order Markov chain - $P(s^t=S_j \mid s^{t-1}=S_i)$
  - $a_{ij} = P(s^t=S_j \mid s^{t-1}=S_i)$, $1 \le i,j \le N$, for any t
- Specifying HMM - Θ, $\mathbf{\Omega}$, π, A, B



Problems in HMM

- For a given HMM $\Lambda=\{\Theta,\Omega,A,B,\pi\}$
  - Estimation of model parameters
  - Given an observation sequence **O**= $\{o^1,o^2, \dots ,o^k\}$, what is the most likely state sequence S = $\{s^1, s^2, \dots ,s^k\}$ that has produced O?
  - How likely is an observation O?

Trained HMMs

Decoding

P(O)

HMM Parameter Estimation

- Given labeled data - state and observation

A — $$t(S_i|S_j) = \frac{\text{number of } (s^t = S_i, s^{t-1} = S_j)}{\text{number of } S_j}$$

B — $$e(o_r|S_j) = \frac{\text{number of } (o^t = o_r, s^t = S_j)}{\text{number of } S_j}$$

- Observation sequence — Forward-Backward Algo.

HMM State Estimation

- Given an observation (sequence) O= {o_1, o_2, … ,o_k}, what is the most likely state sequence S={s_1, s_2, … ,s_k} that has produced O?

Vitubi

Initialization $\delta_{S_i}(1) = t(S_i|s^*)e(o^1|S_i), \quad \forall S_i \in \Theta$

Induction:

for 2≤*t*≤*k*, do

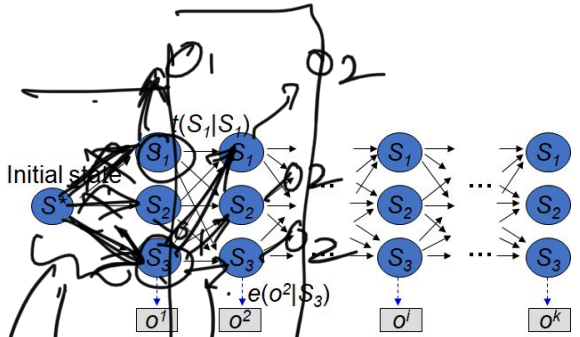
$$\delta_{S_i}(t) = \max_{S_j} t(S_i|S_j)e(o^t|S_i)\delta_{S_j}(t-1)$$

$$\psi_{S_i}(t) = \operatorname{argmax}_{S_j} t(S_i|S_j)e(o^t|S_i)\delta_{S_j}(t-1)$$

Termination:

- The probability of the best state sequence: $\max_{S_j} \delta_{S_j}(k)$
- The best last state: $\hat{s}^k = \operatorname{argmax}_{S_j} \delta_{S_j}(k)$
- Back trace to get other states:

$$\hat{s}^t = \psi_{\hat{s}^{t+1}}(t), \text{ for } t = k-1, \ldots, 1.$$



S^4, S_3, S_1

HMM - Evaluate P(O)

- How likely is an observation O?

$$P(O) = \sum_{S} P(S, O)$$

- Forward algorithm-

Initialization: $\alpha_{S_i}(1) = t(S_i|s^*)e(o^1|S_i), \quad \forall S_i \in \Theta$

Induction:
for $2 \leq t \leq k$, do $\alpha_{S_i}(t) = \sum_{S_j} t(S_i|S_j)e(o^t|S_i)\alpha_{S_j}(t-1)$

Termination: $P(\boldsymbol{O}) = \sum_{S_j} \alpha_{S_j}(k)$

Sample Questions

Problem 1:

The following data is used for training Naive Bayes binary classifier. The last column is the binary class label; Each of the first 4 columns is a binary feature, and each row is a training example.

X1	X2	X3	X4	Y
1	0	0	0	1
0	1	1	0	1
1	0	0	1	0
0	1	1	1	0
1	1	1	0	1

Problem 2:

State true/false along with justification:

1. Logistic Regression may give us a non-linear classifier, depending on how the training examples are distributed in the feature space.

Ans – False because logistic regression is a linear classifier, producing a linear decision boundary.

Problem 3:

Multiple Choice Questions:

Which of the following statements is true about HMM?

a. Given some initial state and transition matrix, it is possible that there exists a state that you can never achieve.
b. The elements of a column of the state transition matrix always sum to 1.
c. The sum of a row of the transition matrix may not be 1.

Ans - (a)

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Unsupervised Learning

Table of contents

What is unsupervised learning?

- Given a training set of n unlabeled samples - x(i)
- What can we learn from the samples?
  - Estimate the overall distribution of the data without knowing their label. — EM algorithm
  - Figure out the groupings of the samples (if any). — Clustering (K-means)
  - Identify some features that may be more important than others. — Feature selection

Finding clusters

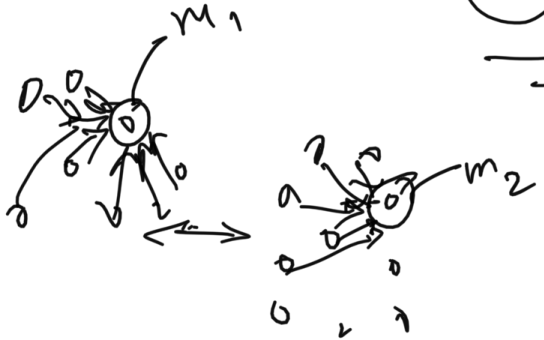
- How to represent the clusters?
  - Centroid (mean of all samples)
- Which cluster a sample should be assigned to (e.g., membership)?
  - Some similarity measure like Euclidean dist.
- What similarity measure to use?
  - Different measures – Euclidean dist.
    - Cosine similarity.

Clustering Objective function

- The sum-of-squared-error criterion/cost-

$$J_e = \sum_{i=1}^{C} \sum_{\mathbf{x} \in D_i} \|\mathbf{x} - \mathbf{m}_i\|^2$$

(centroid of i^{th} cluster.)



$$\mathbf{m}_i = \frac{1}{n_i} \sum_{\mathbf{x} \in D_i} \mathbf{x}$$

Reduce the intra-cluster distance
Increase the inter-cluster distance.

K-Means Clustering

Given: n samples, a number k. → No. of clusters.

Begin

initialize μ_1, μ_2, …, μ_k (randomly selected)

do classify n samples according to nearest μ_i ⟹ Membership Assignment

recompute μ_i

until no change in μ_i

return μ_1, μ_2, …, μ_k

End

K-Means Clustering - Some scenarios

- What happens if the distance of a point is same from more than one centroid?

  Randomly assigned one of the clusters.

  - Hard assignment

- What happens in case of outliers?

  Sensitive to outliers.

K-Means Clustering - Some scenarios

- Different initializations?

  ¬ Might lead to sub-optimal results.

- Different number of clusters than k?

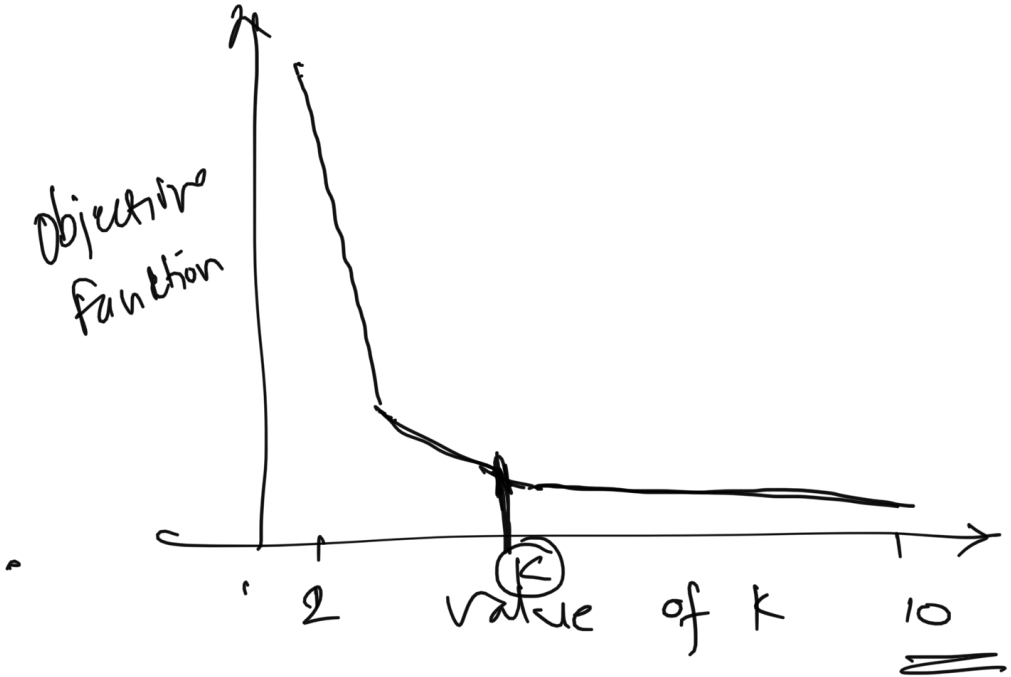
How to choose the k?

- Elbow method.

10,000 $k=1$

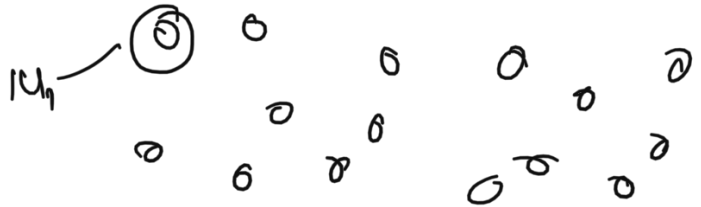
1-9999 $k=n$

$1 < k < n$



How to deal with sensitivity to initialization problem?

1. Run the algo for few epochs & take the average result
2. first centroid randomly. – Sensitive to outliers.
3. Kmeans++ ⇒ Smart initialization



For detailed explanation of how K-Means++ works, refer https://www.youtube.com/watch?v=HatwtJSsj5Q ⇒

K-Means Clustering - Pros and Cons

Pros

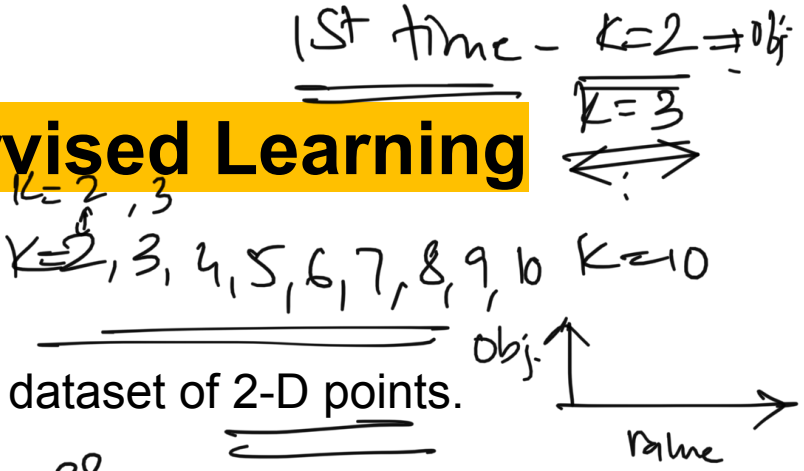
1. Easy to implement
2. fast.

Cons

1. Sensitive to initialization
2. " " outliers.
3. Hard Assignment of data points.
4. Choose no. of clusters manually.

Project Part 2 - Unsupervised Learning (K-means)

- Apply k-means algorithm the given dataset of 2-D points.
- Initialization strategies:
  - Strategy 1: randomly pick the initial centers from the given samples.
  - Strategy 2: pick the first center randomly; for the i-th center (i>1), choose a sample (among all possible samples) such that the average distance of this chosen one to all previous (i-1) centers is maximal.



Project Part 2 - Unsupervised Learning (K-means)

- Objective function-

$$\sum_{i=1}^{k} \sum_{\mathbf{x} \in D_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \longrightarrow SSE$$

- Number of clusters (k) - from 2-10
- Plot the objective function value vs. the number of clusters k.

Project Part 2 - Deliverables

— Due on 26 March (Arizona time) 11:59 PM

- Well-commented code — python / MATLAB / Jupyter notebook (.ipynb)
- A report that summarizes the results and includes all the plots. — .pdf format (typed, not handwritten)
- Please submit the code and the report as separate files on Canvas. **Do not zip them.**

.py
.py
.pdf

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Unsupervised Learning

Table of contents

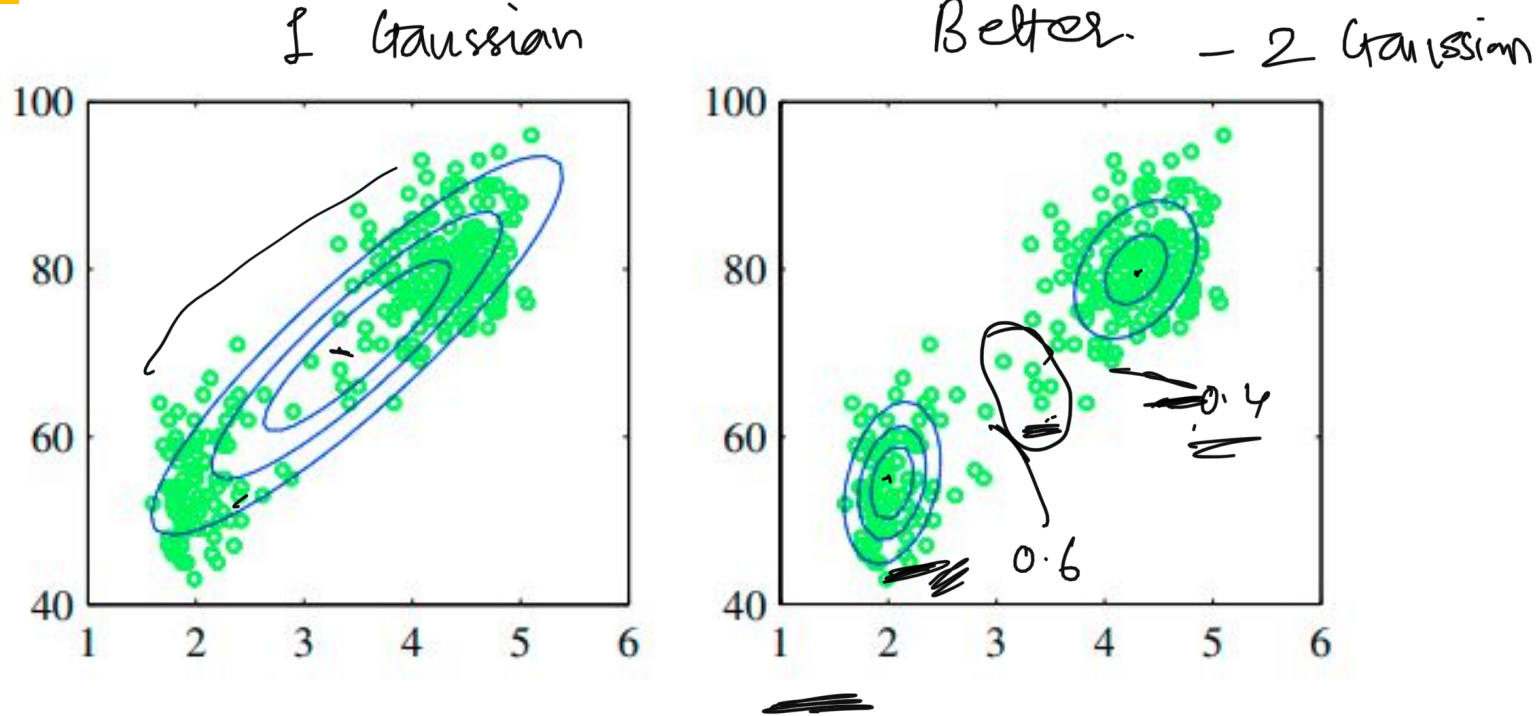
Project - strategy 2 $K=3$ $\sum \|x - \mu_i\|^2$

1st center - randomly initialize

2nd center - max. dist from 1st center.

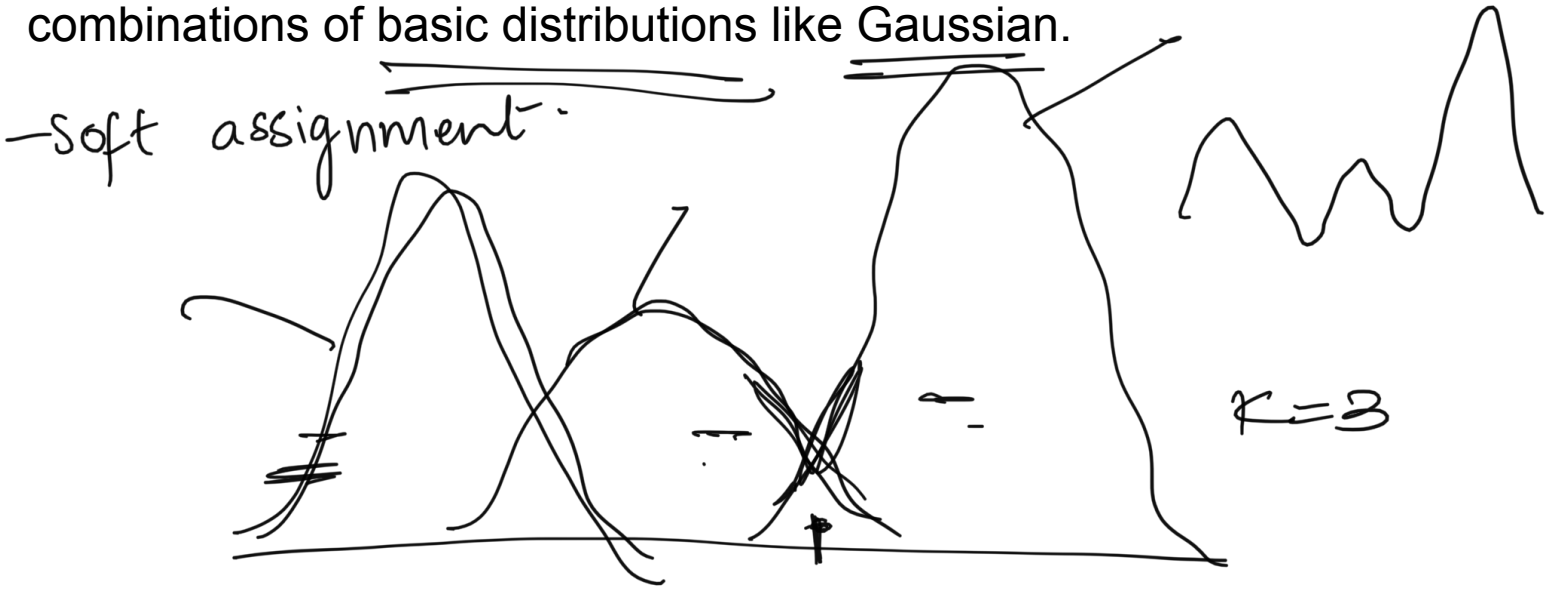
3rd center - max. avg. dist from 1st & 2nd center

Which distribution better represents the given data?



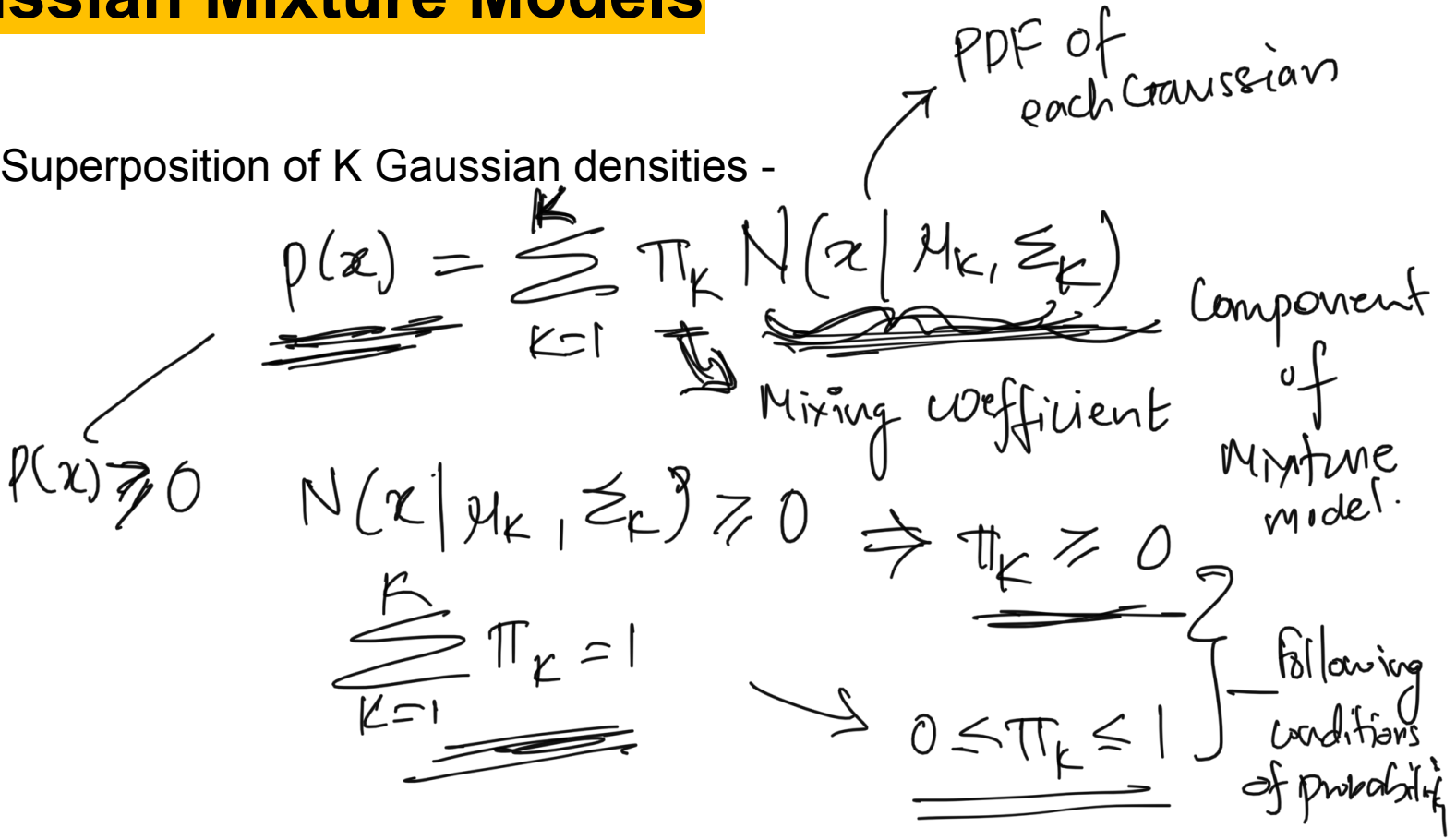
Mixture Models

- Mixture distributions are probabilistic models formed by linear combinations of basic distributions like Gaussian.



Gaussian Mixture Models

- Superposition of K Gaussian densities -



log-likelihood — $\ln p(X|\mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln\left(\sum_{k} \pi_k N(\quad)\right)$

Gaussian Mixture Models

$$p(x) = \sum_{k} p(k) \cdot p(x|k)$$

$p(k|x)$ · — π_k — Prior — $N(x|\mu_k|\Sigma_k)$ conditional probability

$$\gamma_k(x) = \frac{p(k) \cdot p(x|k)}{\sum_{k} p(k) \cdot p(x|k)} \cdot \}\text{ - From Bayes theorem.}$$

(responsibilities

$$= \frac{\pi_k N(x|\mu_k, \Sigma_k)}{\sum_{k} \pi_k N(x|\mu_k, \Sigma_k)}$$

$k \Rightarrow$ No. of components in the mixture model

EM

$K=4$

EM algorithm for GMM

k-means initialization

k-means centroids

sample covariance

No. of samples in each cluster / Total samples

1. Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k, and evaluate the initial value of the log likelihood.
2. **E step**. Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{9.23}$$

EM algorithm for GMM

3. **M step**. Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\mathbf{x}_n \tag{9.24}$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^{\mathrm{T}} \tag{9.25}$$

$$\pi_k^{\text{new}} = \frac{N_k}{N} \tag{9.26}$$

where

$$N_k = \sum_{n=1}^{N}\gamma(z_{nk}). \tag{9.27}$$

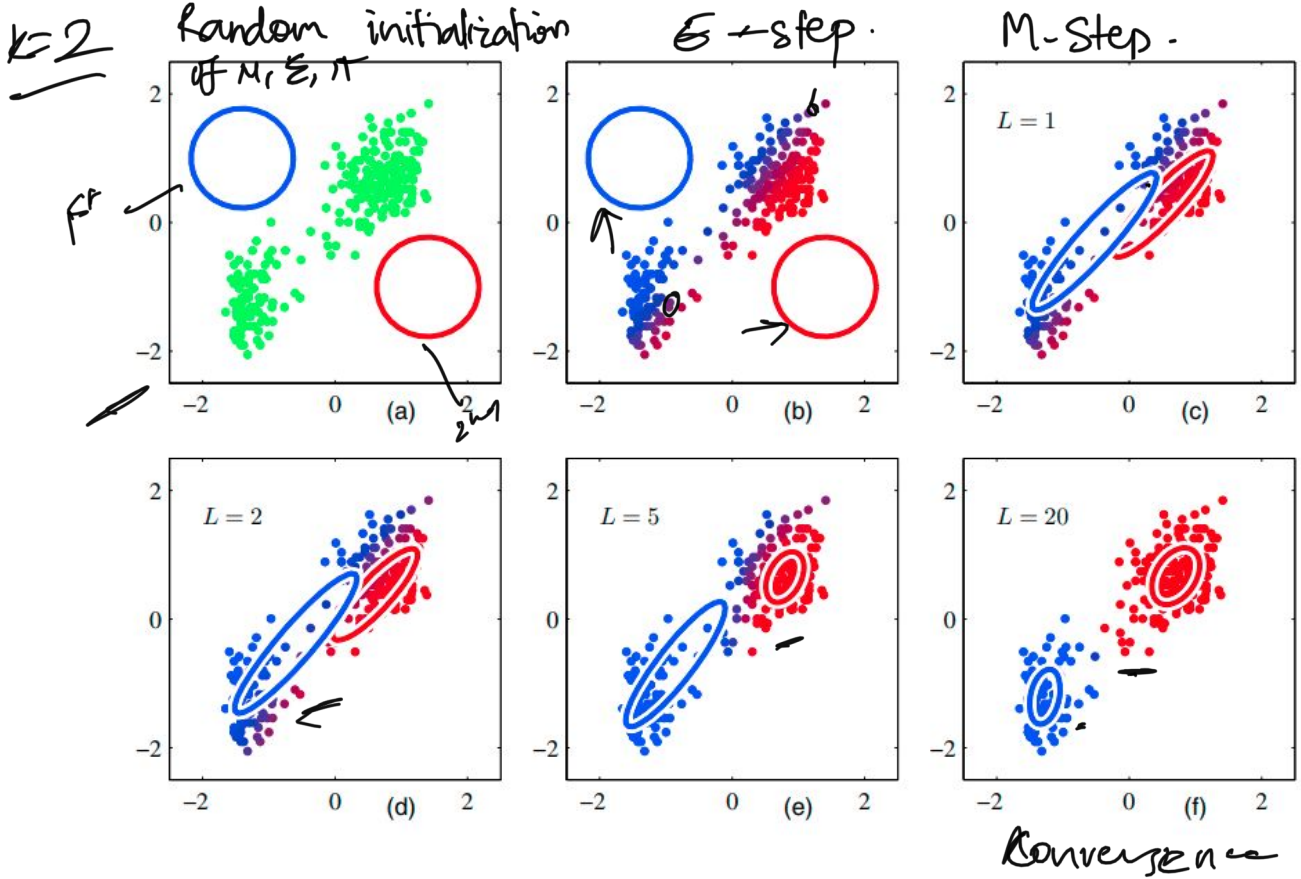
4. Evaluate the log likelihood

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N}\ln\left\{\sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right\} \tag{9.28}$$

and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.

MLE

Illustration of EM for GMM



Questions?

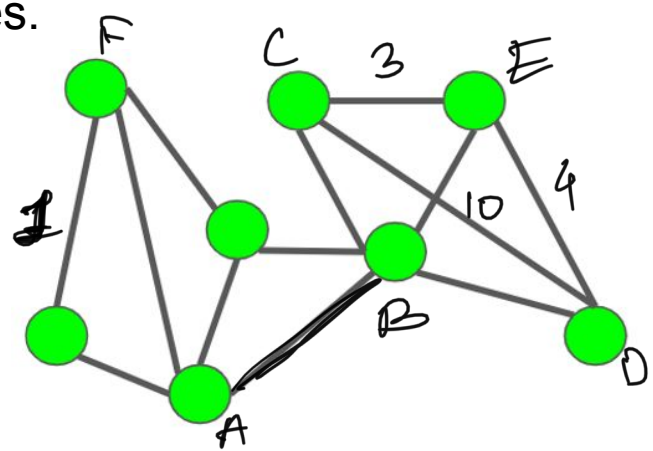
CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Spectral Clustering

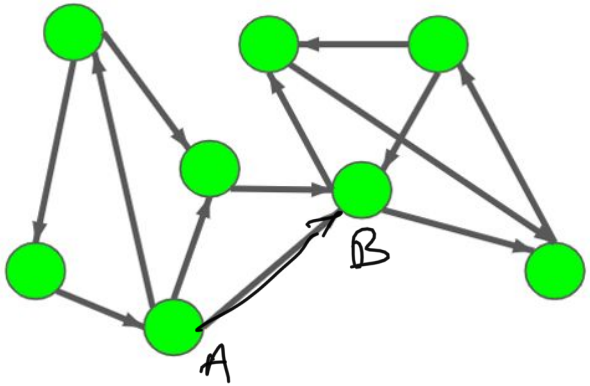
Table of contents

What is a graph?

- A graph G = (V, E) is defined by V, a set of N vertices, and E, a set of edges.



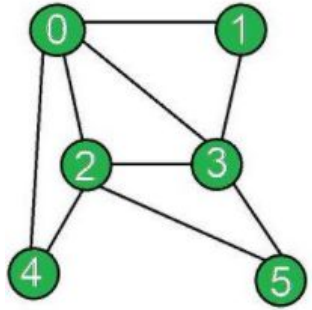
Undirected graph

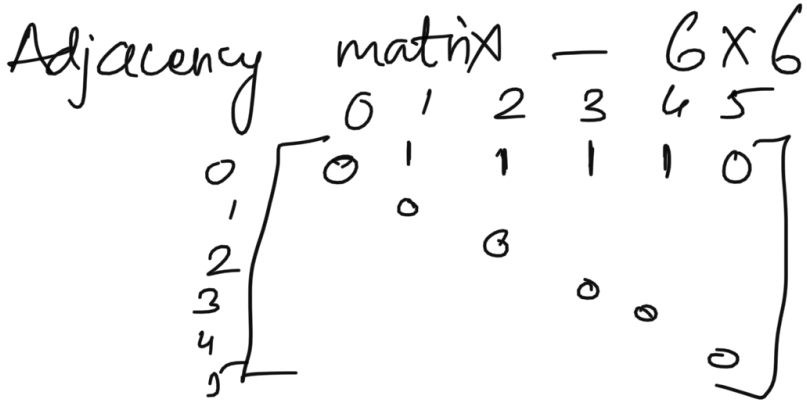


Directed graph

Graph Representation - Adjacency Matrix W

- For an undirected graph - N vertices



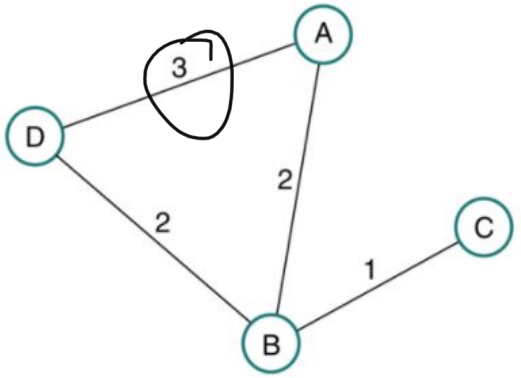


Graph Representation - Weighted Adjacency Matrix

NXN matrix

4X4 matrix

	A	B	C	D
A	0	2	0	3
B		0		
C			0	
D				0



Graph Representation - Degree Matrix D

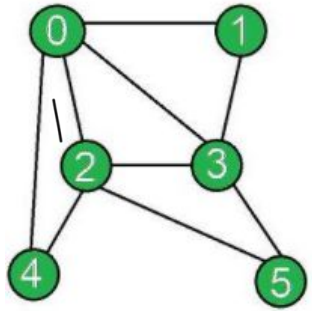
- Degree of a node is the number of edges incident to the node.

- NXN matrix for N nodes.

- Diagonal matrix

	0	1	2	3	4	5
0	4	0	0	0		
1	0	2				
2			4			
3	0			4		
4	0				2	
5	0					2

— Undirected graph.



Graph Representation - Graph Laplacian L

- **L = D - W**
- Properties of **L** -
  - **L** is symmetric and positive semi-definite
  - L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$
  - The smallest eigenvalue is 0, the corresponding eigenvector is the 1-vector (all elements being 1)

Refer for all more details on the properties: https://arxiv.org/pdf/0711.0189.pdf and https://sites.cs.ucsb.edu/~veronika/SpectralClustering.pdf

Different ways to build similarity graphs

- ε-neighborhood graph $dist(A, B) < \varepsilon \Rightarrow$ Edge betn A & B
  - unweighted graph.
- k-nearest neighbor graph –

  mutual k-nearest neighbor - property is satisfied both ways
- fully connected graph

Clustering – Increase inter-cluster distance
Decrease intra-cluster distance.

Clustering as a Graph Partition

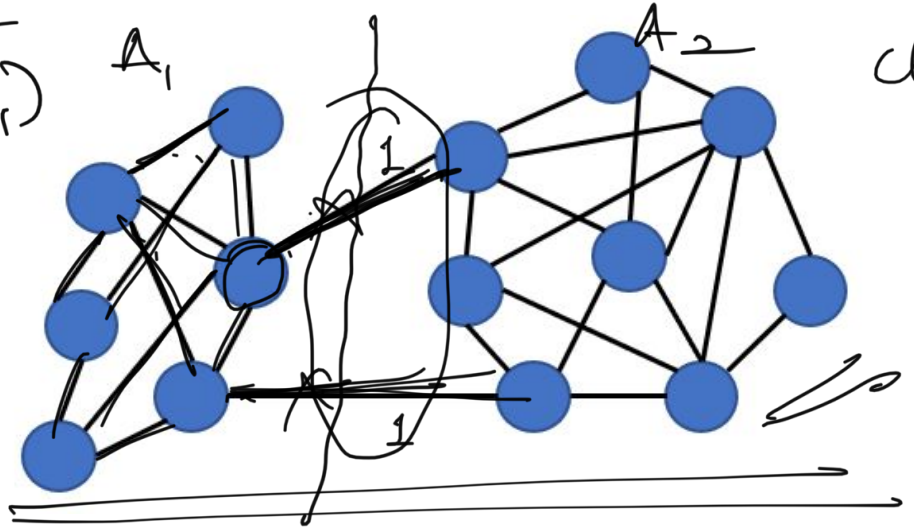
- Find a partition of a graph such that the edges between different groups have a very low weight while the edges within a group have high weight.

$W(A_1, A_1)$

$|A_1| = 6$

$|A_2| = 8$

$vol(A_1) =$



Clusters = 2

MinCut Problem

- Given a similarity graph with adjacency matrix W, the simplest and most direct way to construct a partition of the graph is to solve the mincut problem.

minimize $\mathrm{cut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A}_i)$

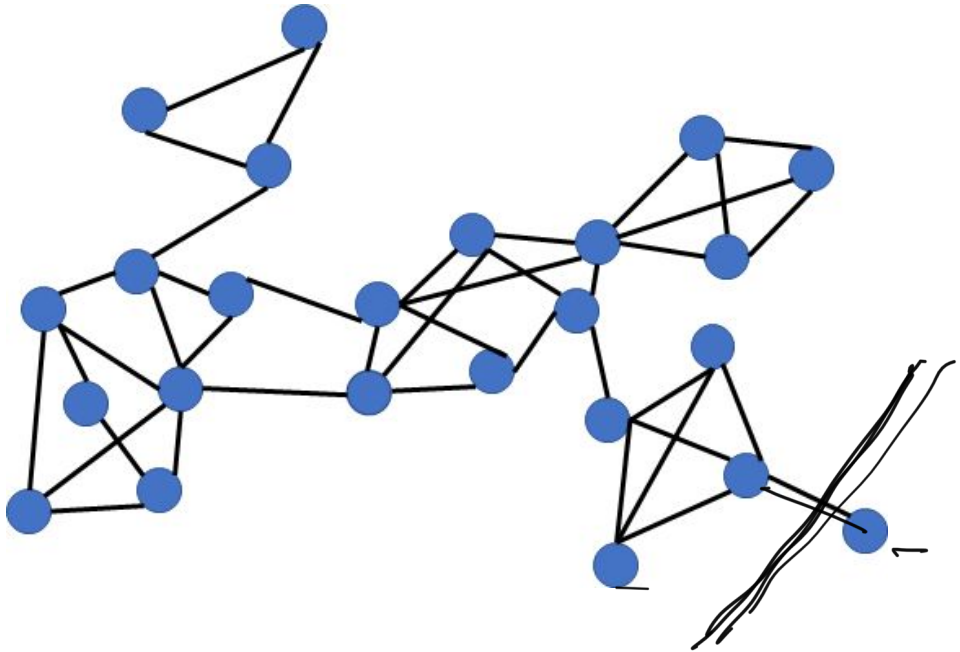
$\overline{A}_i$: complement of A_i

$k = 2$

$\mathrm{cut}(A_1, A_2) = \frac{1}{2} \sum_{i=1}^{2} W(A_i, \overline{A}_i)$

MinCut Problem - Major Drawback

- It does not balance the size of the partitions



$|A_i| \Rightarrow$ No. of vertices/nodes in a cluster A_i.

Other types of cuts

- To overcome the drawback of MinCut, one obvious solution is to specify that the partitions are reasonably large.

$$\mathrm{RatioCut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{|A_i|} = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, \overline{A}_i)}{|A_i|}$$

Normalized cut. $\leftarrow$

$$\mathrm{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{\mathrm{vol}(A_i)} = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, \overline{A}_i)}{\mathrm{vol}(A_i)}.$$

$$\mathrm{MinMaxCut}(A_1, \ldots, A_k) := \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, \overline{A}_i)}{W(A_i, A_i)}.$$

Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Spectral Clustering

Table of contents

Other types of cuts

- To overcome the drawback of MinCut, one obvious solution is to specify that the partitions are reasonably large.

$$\text{RatioCut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{|A_i|} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{|A_i|}$$

$$\text{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{\text{vol}(A_i)} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{\text{vol}(A_i)}.$$

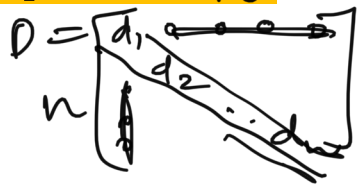
$$\text{MinMaxCut}(A_1, \ldots, A_k) := \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{W(A_i, \overline{A}_i)}.$$

No. of vertices in a cluster A_i

Page No. 25 of the doc in Recommended Resources

Properties of Graph Laplacian

- **L = D - W** → Adjacency Matrix (Degree matrix)



- Properties of **L** -
  - *For every vector $f \in \mathbb{R}^n$ we have*

$$f'Lf \geq 0$$

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

+ve +ve

f' f

Proof - By the definition of d_i,

$$f'Lf = f'Df - f'Wf = \sum_{i=1}^{n} d_i f_i^2 - \sum_{i,j=1}^{n} f_i f_j w_{ij}$$

Element of D

Element in W

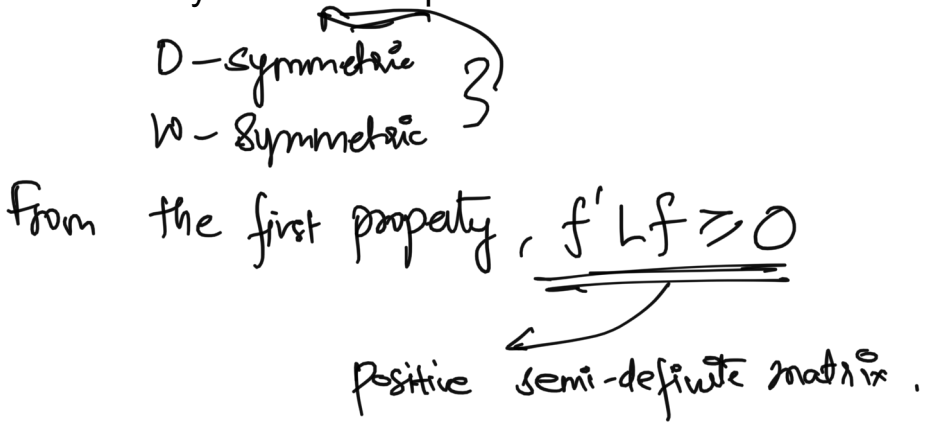
$$= \frac{1}{2}\left(\sum_{i=1}^{n} d_i f_i^2 - 2\sum_{i,j=1}^{n} f_i f_j w_{ij} + \sum_{j=1}^{n} d_j f_j^2\right) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

$(a-b)^2 = a^2 - 2ab + b^2$

Reference: https://arxiv.org/pdf/0711.0189.pdf and https://sites.cs.ucsb.edu/~veronika/SpectralClustering.pdf

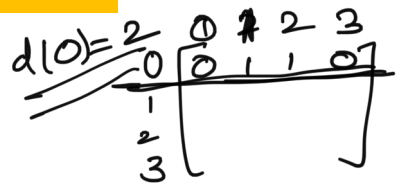
Properties of Graph Laplacian

- Properties of **L** -
  - **L** is symmetric and positive semi-definite



Reference: https://arxiv.org/pdf/0711.0189.pdf and https://sites.cs.ucsb.edu/~veronika/SpectralClustering.pdf

Properties of Graph Laplacian



- Properties of **L** -
  - The smallest eigenvalue is 0, the corresponding eigenvector is the 1-vector (all elements being 1)

Proof –

$$L = D - W = \begin{pmatrix} d_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_n \end{pmatrix} - \begin{pmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{pmatrix} = \begin{pmatrix} d_1 - w_{11} & -w_{12} & \cdots \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{j=1}^{n} w_{1j} - w_{11} & \cdots & -w_{1n} \\ \vdots & \ddots & \vdots \\ -w_{n1} & \cdots & \sum_{j=1}^{n} w_{nj} - w_{nn} \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} \sum_{j=1}^{n} w_{1j} - w_{11} & \cdots & -w_{1n} \\ \vdots & \ddots & \vdots \\ -w_{n1} & \cdots & \sum_{j=1}^{n} w_{nj} - w_{nn} \end{pmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

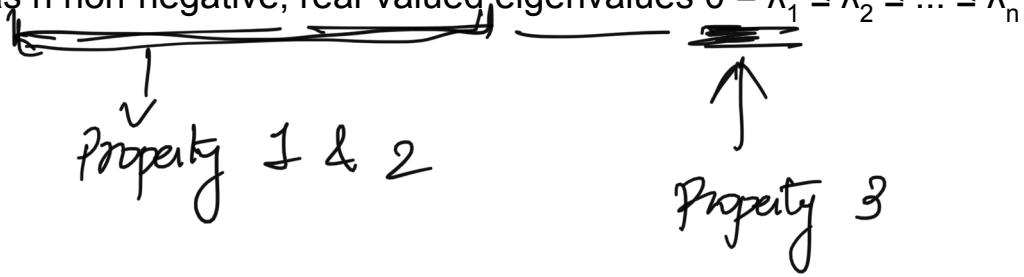
$$\begin{pmatrix} \sum_{j=1}^{n} w_{1j} - w_{11} - \sum_{j=2}^{n} w_{1j} \\ \sum_{j=1}^{n-1}(-w_{nj}) - \sum_{j=1}^{n} w_{nj} - w_{nn} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

$Lx = \lambda x$

Reference: https://arxiv.org/pdf/0711.0189.pdf and https://sites.cs.ucsb.edu/~veronika/SpectralClustering.pdf

Properties of Graph Laplacian

- Properties of **L** -
  - L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$



Refer for all more details on the properties: https://arxiv.org/pdf/0711.0189.pdf and https://sites.cs.ucsb.edu/~veronika/SpectralClustering.pdf

MinCut Problem

- Given a similarity graph with adjacency matrix W, the simplest and most direct way to construct a partition of the graph is to solve the mincut problem.

Minimize $\operatorname{cut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A}_i)$

MinCut for k=2

- Given **W** and a cluster membership vector **q,**

$$q_i = \begin{cases} 1 & i \in Cluster\ A \\ -1 & i \in Cluster\ B \end{cases}$$

$$\mathbf{q} = \underset{\mathbf{q} \in [-1,\ 1]^n}{\operatorname{argmin}} \frac{1}{4} \sum_{i,j} (q_i - q_j)^2 w_{i,j}$$

$n \rightarrow$ no. of samples

$2^n \Rightarrow$ exponential Complexity.

Cut size

Relaxation Approach

$$\mathbf{q} = \underset{\mathbf{q}}{\operatorname{argmin}}\, J = \underset{\mathbf{q}}{\operatorname{argmin}}\, \mathbf{q}^T(\mathbf{D}-\mathbf{W})\mathbf{q}$$

$$\text{subject to } \sum_{i=1}^{n} q_i^2 = n$$

n → no. of samples

$q^T L q$

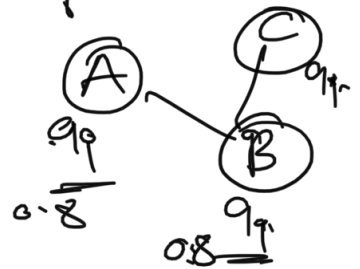
eigenvector corresponding to second smallest eigenvalue of L is the soln

$q_i \Rightarrow$ any real value betn -1, 1

$q = [0.8 \quad -0.2 \quad 0.1]$

$k=2 \quad \{q>0 \Rightarrow 1 \quad q<0 \Rightarrow -1\}$

$q_i = 0.8$



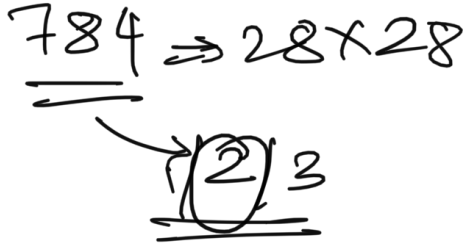
Questions?

CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Dimensionality Reduction

Table of contents

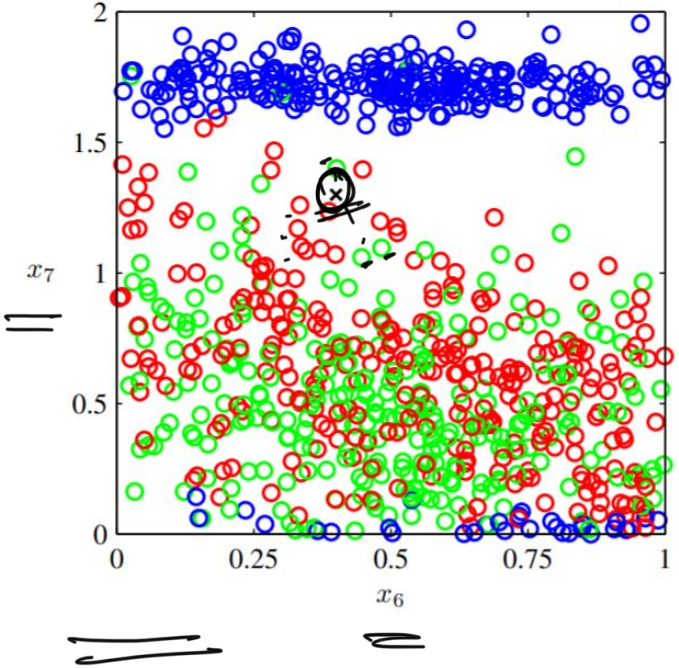
Dimensionality reduction

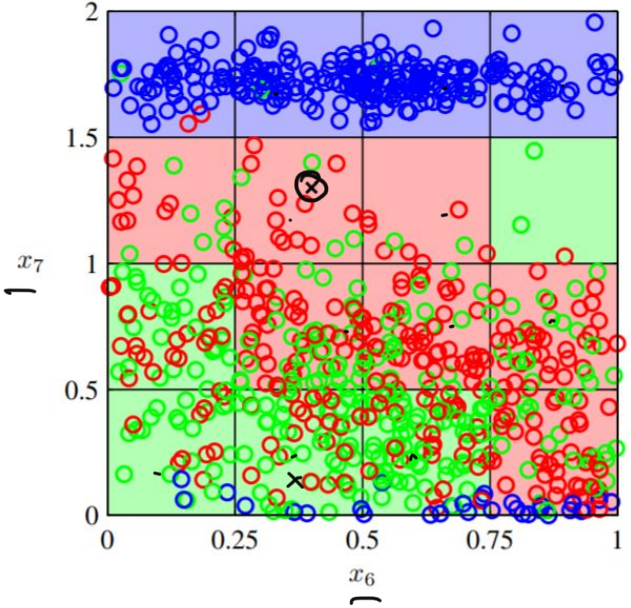


- Given N data points in a high-dimensional space (in the order of tens of thousands of dimensions), project them into some low-dimensional space.
- Why project them into some low-dimensional space?

– Curse of dimensionality.

Curse of dimensionality



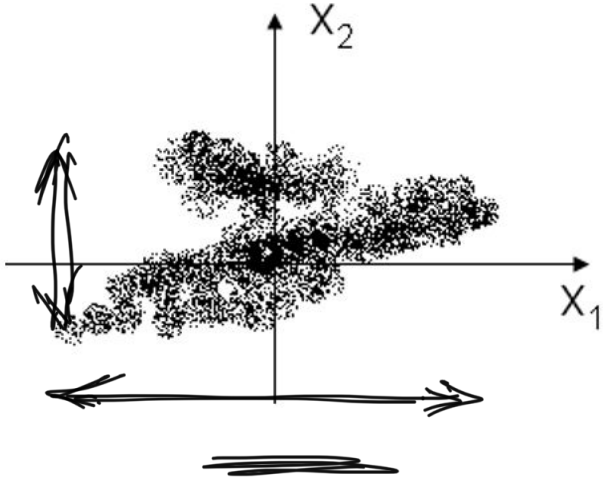


From 'Pattern Recognition and Machine Learning', by Bishop book. (Eq no. 33, Section 1.4)

Principal Component Analysis (PCA)

- Basic idea -

2D to 1D



Different ways -

(1) Random

(2) Discard the less descriptive

(3) Project onto a space such that the max. variance is obtained.

PCA Problem Formulation

d dimensions

Problem

Given n samples $D=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ in d-dimensional space, find a direction $\mathbf{e}_1$, such that the projection of D onto $\mathbf{e}_1$ gives the largest variance (compared with any other direction).

$e_1 \Rightarrow$ unit vector, $\|e_1\| = 1$

Projections – $y_i = x_i \cdot e_1$ → Inner product.

Mean of projections – $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{1}{n}\sum_{i=1}^{n} x_i \cdot e_1 = \bar{x} \cdot e_1$

PCA Problem Formulation

Variance of projections –

$$\sigma^2(e_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 = \frac{1}{n}\sum_{i=1}^{n}\left[(x_i - \bar{x})e_1\right]^2$$

$$= \sum_{j=1}^{d}\sum_{k=1}^{d}\left[\frac{1}{n}\sum_{i=1}^{n}(x_{i,j} - \bar{x}_{i,j})(x_{i,k} - \bar{x}_{i,k})\right]e_j e_k$$

(the bracketed term is the covariance)

$$= \sum_{j=1}^{d}\sum_{k=1}^{d} e_j e_k C_{jk}$$ where $C \rightarrow$ covariance matrix.

$$\sigma^2(e_1) = e^t C e \Rightarrow \text{Matrix form.}$$

PCA Problem Formulation

$$e_1 = \underset{e}{\operatorname{argmax}}\ \sigma^2(e) \quad \text{subject to } \|e\| = 1$$

$$F(e) = \underset{e}{\operatorname{argmax}}\ \sigma^2(e) - \lambda(e^t e - 1)$$

(λ: Lagrange's multiplier)

$$\sigma^2(e) \downarrow e^t C e$$

$$\frac{\partial F(e)}{\partial e} = 2Ce - 2\lambda e = 0 \Rightarrow \boxed{Ce = \lambda e}$$

$e \rightarrow$ eigenvector of covariance matrix C

$\lambda \rightarrow$ eigenvalue corresponding to eigenvector e.

PCA - Principal Components

- What are principal components?

  — e_i vectors.

  — First principal component – eigenvector with the largest eigenvalue.

- How many principal components to keep?

Total variance – Sum of variance in all the projections.

$$= \sum_{j=1}^{d} \lambda_j$$

$d' \Rightarrow$ new no. of dimensions where $d' << d$

$$\frac{\sum_{j=1}^{d'} \lambda_j}{\sum_{j=1}^{d} \lambda_j} \sim$$

$D = [n \times d]$ $n \times d'$

PCA Algorithm

$d \rightarrow$ total no. of dimensions of data.

1. Compute the $d \times d$ sample covariance matrix C
2. Find the eigenvalues and corresponding eigenvectors of C
3. Project the original data onto the space spanned by the eigenvectors
   - The projection may be done onto a d'-dimensional subspace spanned by the first d' eigenvectors (ordered by the eigenvalue in descending order)
     - d' is determined by the desired accuracy

Questions?

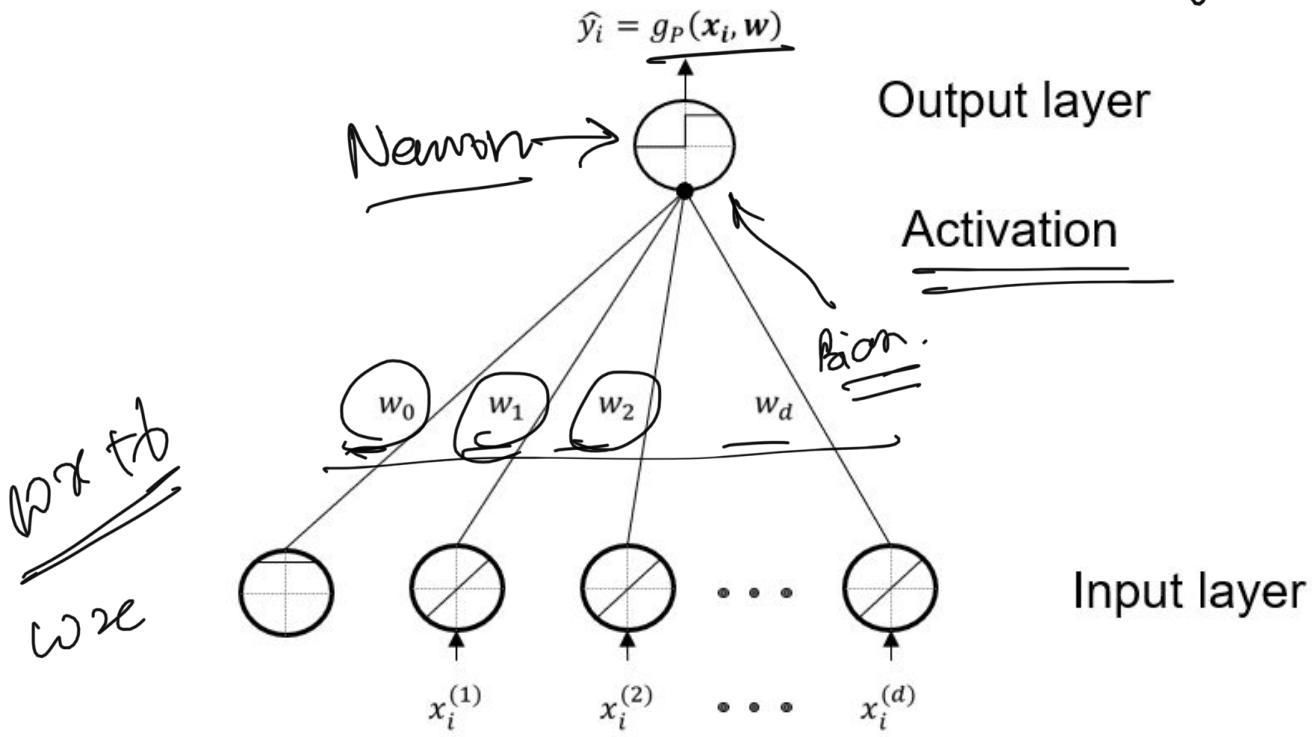
CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Neural Networks & Deep Learning

Table of contents

What is a Perceptron?

linear binary classifier.



Learning in Perceptron

Iterate for t until a stop criterion is met

{

for each sample x_i with label y_i:

{

compute the output ($\hat{y}_i$) of the network

estimate the error of the network $e(w(t))= y_i - \hat{y}_i$

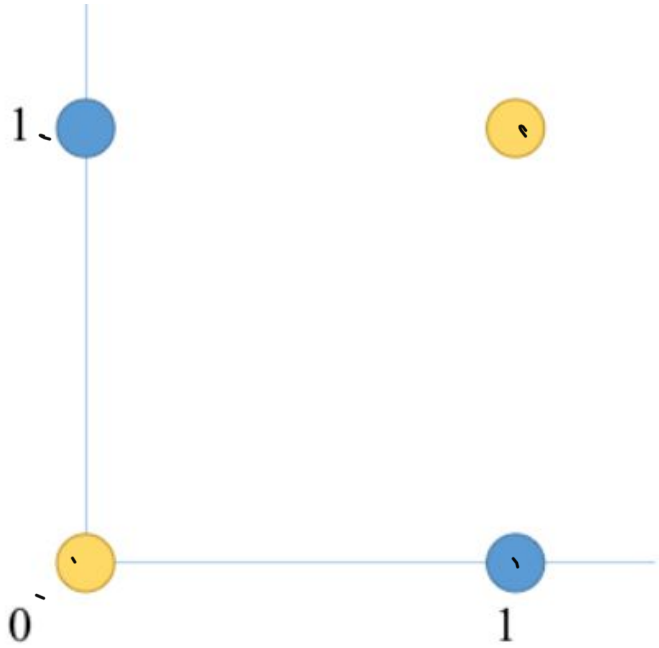
update the weight $w(t+1)=w(t)+e(w(t))x_i$

}

t++

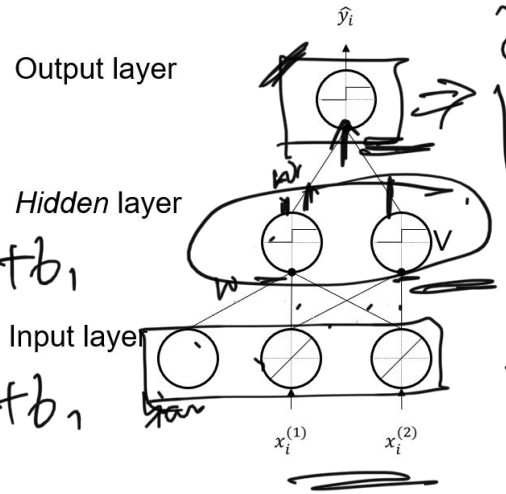
}

XOR problem



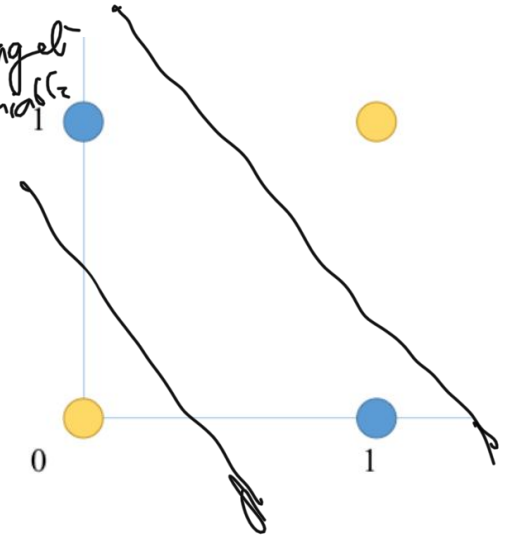
XOR problem - Using MLP (Multi-layer perceptron).

$w_{t+1} = w_t - \eta \nabla_w L$



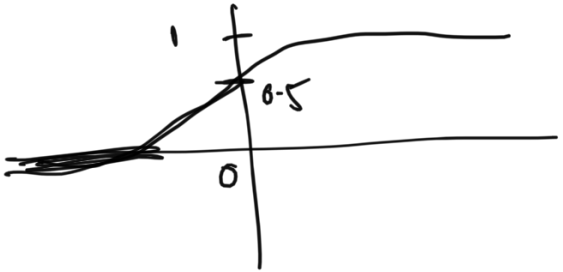
$z_1 = w_1 x + b_1$

$z_2 = w_2 x + b_1$



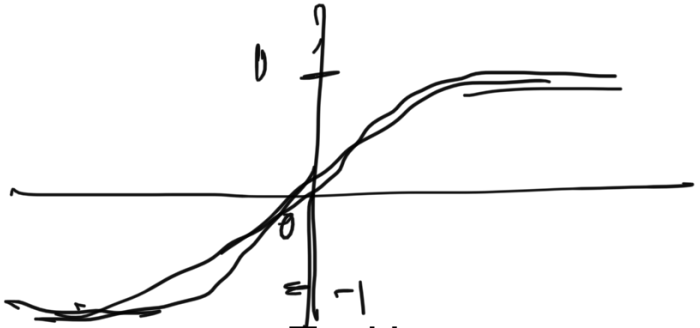
- How do we learn weights? Can perceptron learning be applied?
  - Gradient decent
  - Differentiable.
  - Backpropagation.

Activation Functions



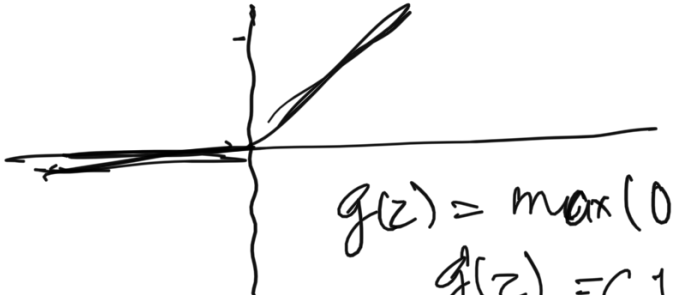
Sigmoid

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



TanH

$g(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$ $g(z) = 1-(g(z))^2$

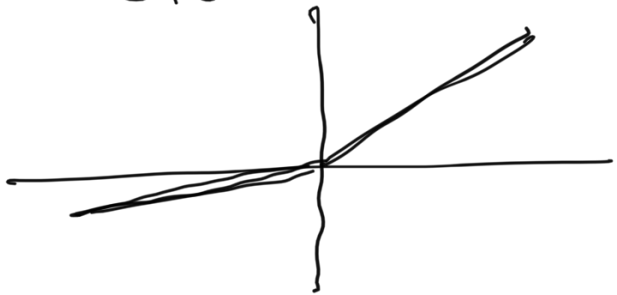


ReLU

(Rectified Linear Unit)

$g(z) = \max(0, z)$

$g(z) = \begin{cases} 1 & z \geq 0 \\ 0 & z < 0 \end{cases}$



Leaky ReLU

$g(z) = \max(\epsilon z, z)$

Why non-linear activation functions are important?

$w_1 x + b_1 \rightarrow w_2(w_1 x + b_1) + b_2$

$w_2(\sigma(w_1 x + b_1))$

w'

- Bring non-linearity to the network.

[0.05 0.05 0.9]

Handling Multiple Classes

- No. of output neurons - No. of classes.
- Activation function - Softmax activation — $\frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$ ⇒ Normalization Constant

  C → no. of classes.

  — Bring the values betn 0 & 1.

  — argmax of softmax probabilities
- Error Function- Cross-entropy (binary cross entropy)

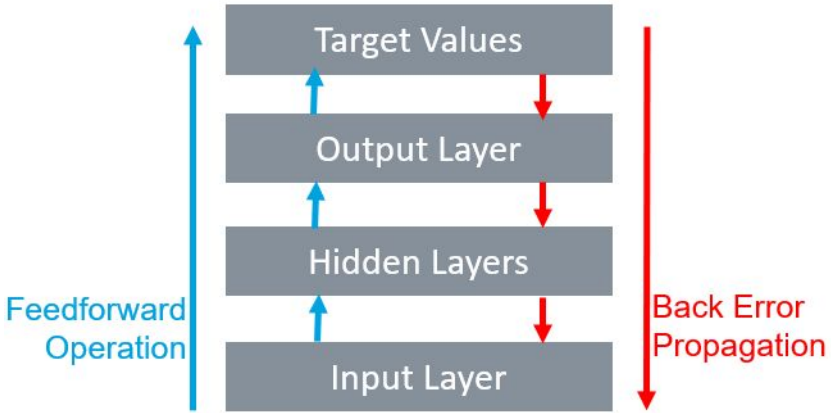
$$L = -\sum_{i=1}^{n} \sum_{j=1}^{} \Pi_j(y_i) \log P(y_i = j | x_i)$$

$\Pi_j(y_i)$: $\begin{cases} 1 \\ 0 \end{cases}$ $y_i = j$ otherwise

$P(y_i = j | x_i)$ → softmax probability.

MLP Learning - Back Propagation

- Key idea - Properly distribute error computed from output layer back to earlier layers to allow their weights to be updated in a way that reduce the error.



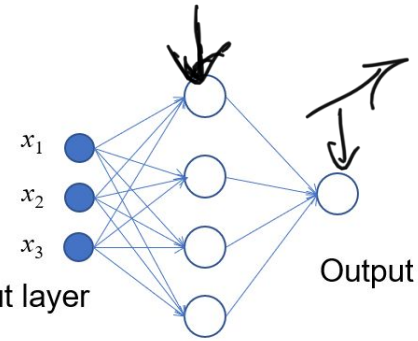
4x3 (4x1) 4x3
Broadcasting

MLP Learning - Back Propagation

Feed-Forward Phase-

m → total no. of samples.

activation fun^n → sigmoid.



3-dimension input
4 hidden nodes
1 output nodes
(binary classification)

$A_0=X$ (3xm) → $Z_1=W_1A_0+b_1$ → Z_1 (4xm) → $A_1=f(Z_1)$ → A_1 (4xm) → $Z_2=W_2A_1+b_2$ → Z_2 (1xm) → $A_2=f(Z_2)$ → A_2 (1xm) → $L(A_2,Y)$

W_1: 4x3
b_1: 4x1

W_2: 1x4
b_2: 1x1

Loss function

$Z_1 = W_1 A_0 + b_1$

(4xm) (4x3) (3xm) (4x1)

$Z_2 = W_2 A_1 + b_2$

(1xm) (1x4) (4xm) (1x1)

MLP Learning - Back Propagation

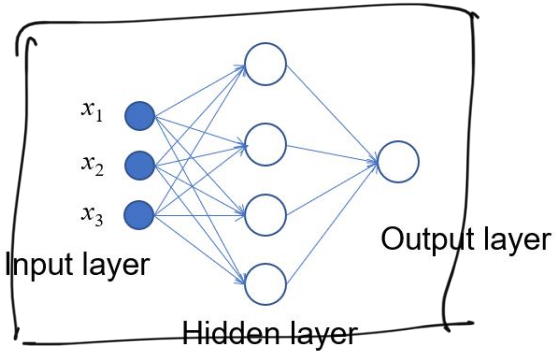
Back propagation phase-

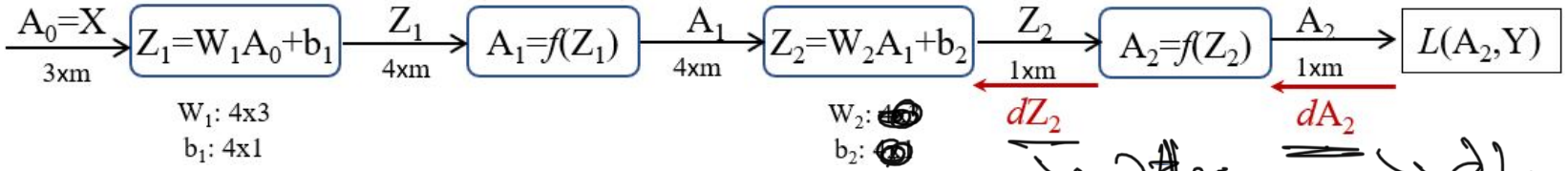
$$w_{t+1} = w_t - \eta \nabla_w L$$

Parameters -

W_1, b_1, W_2, b_2

L = Loss function

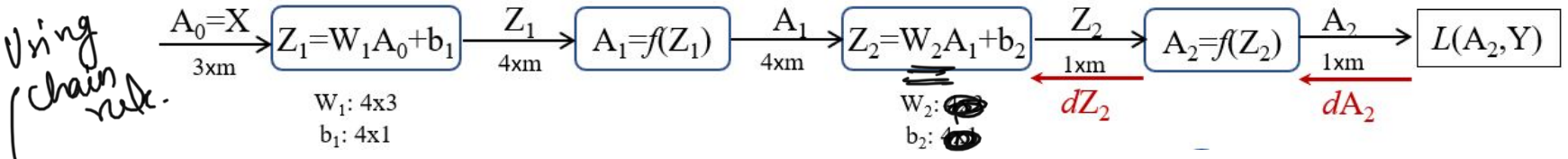




$\frac{\partial L}{\partial W_1}, \frac{\partial L}{\partial b_1}, \frac{\partial L}{\partial W_2}, \frac{\partial L}{\partial b_2}$ → chain rule. $\frac{\partial L}{\partial Z_2}$ $\frac{\partial L}{\partial A_2}$

Activation → Sigmoid, Output → Binary loss → BCE

MLP Learning - Back Propagation



$$\frac{\partial L}{\partial W_2}=\frac{\partial L}{\partial A_2}\cdot\frac{\partial A_2}{\partial Z_2}\cdot\frac{\partial Z_2}{\partial W_2},\quad \frac{\partial L}{\partial b_2}=\frac{\partial L}{\partial A_2}\cdot\frac{\partial A_2}{\partial Z_2}\cdot\frac{\partial Z_2}{\partial b_2},\quad \frac{\partial L}{\partial W_1}=\frac{\partial L}{\partial A_2}\cdot\frac{\partial A_2}{\partial Z_2}\cdot\frac{\partial Z_2}{\partial A_1}\cdot\frac{\partial A_1}{\partial Z_1}\cdot\frac{\partial Z_1}{\partial W_1}$$

$$\frac{\partial L}{\partial b_1}=\frac{\partial L}{\partial A_2}\cdot\frac{\partial A_2}{\partial Z_2}\cdot\frac{\partial Z_2}{\partial A_1}\cdot\frac{\partial A_1}{\partial Z_1}\cdot\frac{\partial Z_1}{\partial b_1}$$

$$L=-\left(y\log A_2+(1-y)\log(1-A_2)\right)$$

$$\frac{\partial L}{\partial A_2}=\frac{A_2-Y}{A_2(1-A_2)},\quad \frac{\partial Z_2}{\partial W_2}=A_1^T$$

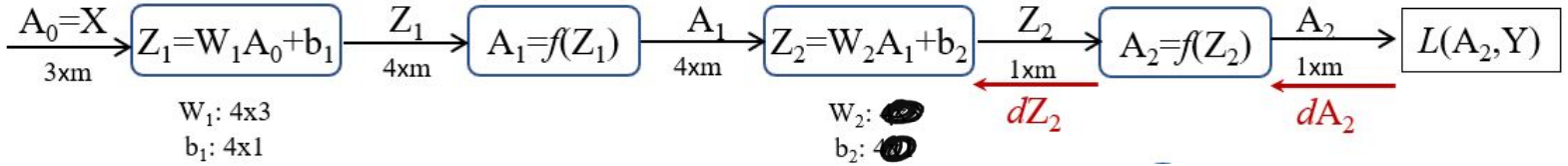
$$\frac{\partial A_2}{\partial Z_2}=A_2(1-A_2),\quad \frac{\partial Z_2}{\partial b_2}=[1,1\ldots 1]^T$$

$$\frac{\partial Z_2}{\partial A_1}=W_2^T,\quad \frac{\partial A_1}{\partial Z_1}=A_1(1-A_1)$$

$$\frac{\partial Z_1}{\partial W_1}=A_0^T,\quad \frac{\partial Z_1}{\partial b_1}=[1,\ldots 1]^T$$

1xm 4x1 4xm 4x1

MLP Learning - Back Propagation



$$\frac{\partial L}{\partial W_2} = \underbrace{(A_2-Y)}_{1\times m} \cdot \underbrace{A_2(1-A_2)}_{1\times m} \cdot A_1^T \quad (1\times 4,\ m\times 4)$$

$$\frac{\partial L}{\partial b_2} = (A_2-Y) \cdot A_2(1-A_2) \cdot [1 \dots 1]^T \quad (1\times 1,\ m\times 1)$$

$$\frac{\partial L}{\partial W_1} = (A_2-Y) \cdot A_2(1-A_2) \cdot (W_2)^T \cdot A_1(1-A_1) \cdot A_0^T$$

$$\frac{\partial L}{\partial W_1} = \underbrace{(W_2)^T}_{4\times 1}\ \underbrace{(A_2-Y)}_{1\times m}\ \underbrace{A_1(1-A_1)}_{4\times m}\ \underbrace{A_0^T}_{m\times 3} \quad (4\times 3)$$

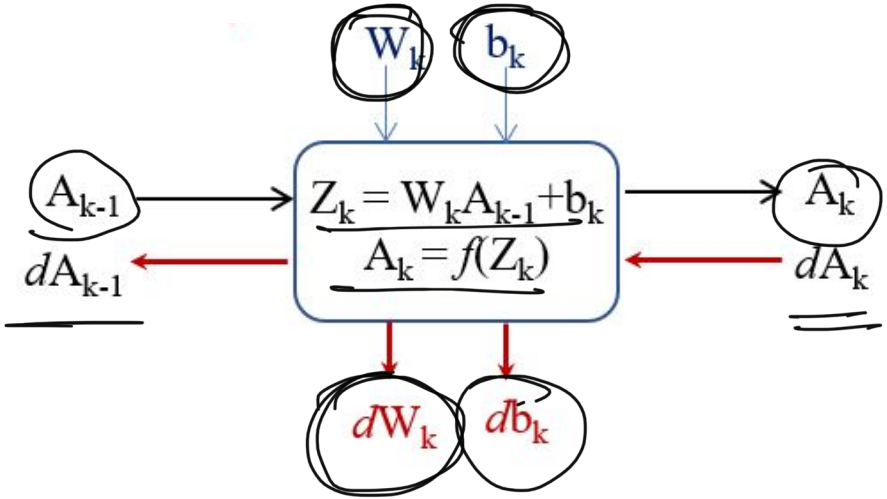
Element-wise multiplication

$$\frac{\partial L}{\partial b_1} = (A_2-Y) \cdot A_2(1-A_2) \cdot W_2^T A_1(1-A_1) \cdot [1 \dots 1]^T$$

$$\frac{\partial L}{\partial b_1} = \underbrace{W_2^T}_{4\times 1} \cdot \underbrace{(A_2-Y)}_{1\times m}\ \underbrace{A_1(1-A_1)}_{4\times m}\ \underbrace{[1 \dots 1]^T}_{m\times 1} \quad (4\times 1)$$

Element-wise multiplication

MLP Learning - Back Propagation



Questions?

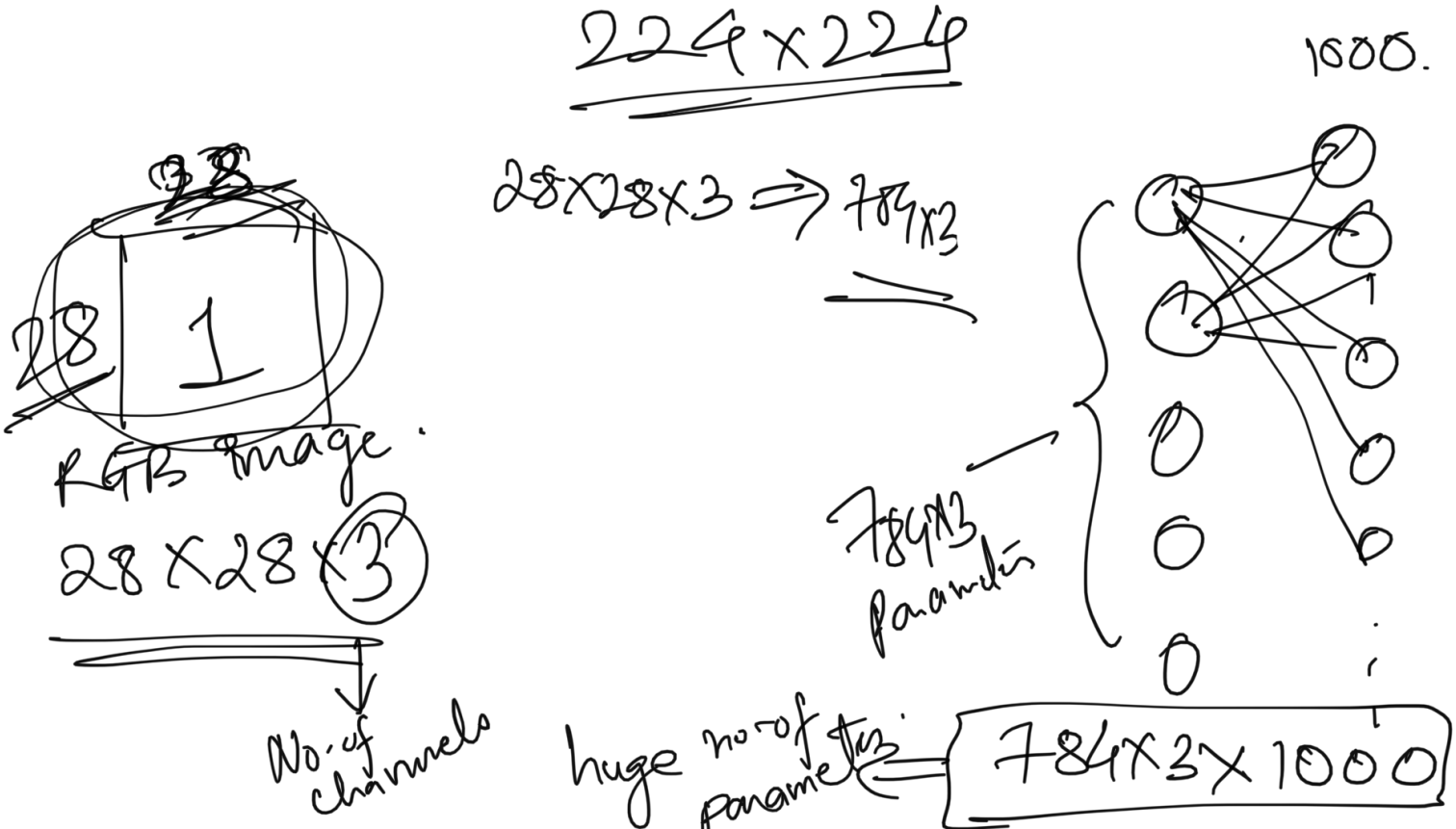
CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Neural Networks & Deep Learning

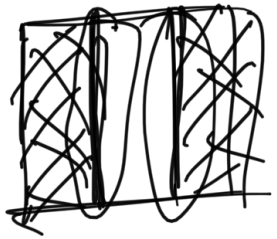
Table of contents

FC (fully connected).

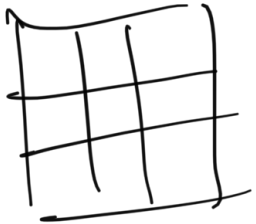
Number of parameters in dense layers



Convolutional Neural Network (CNN)

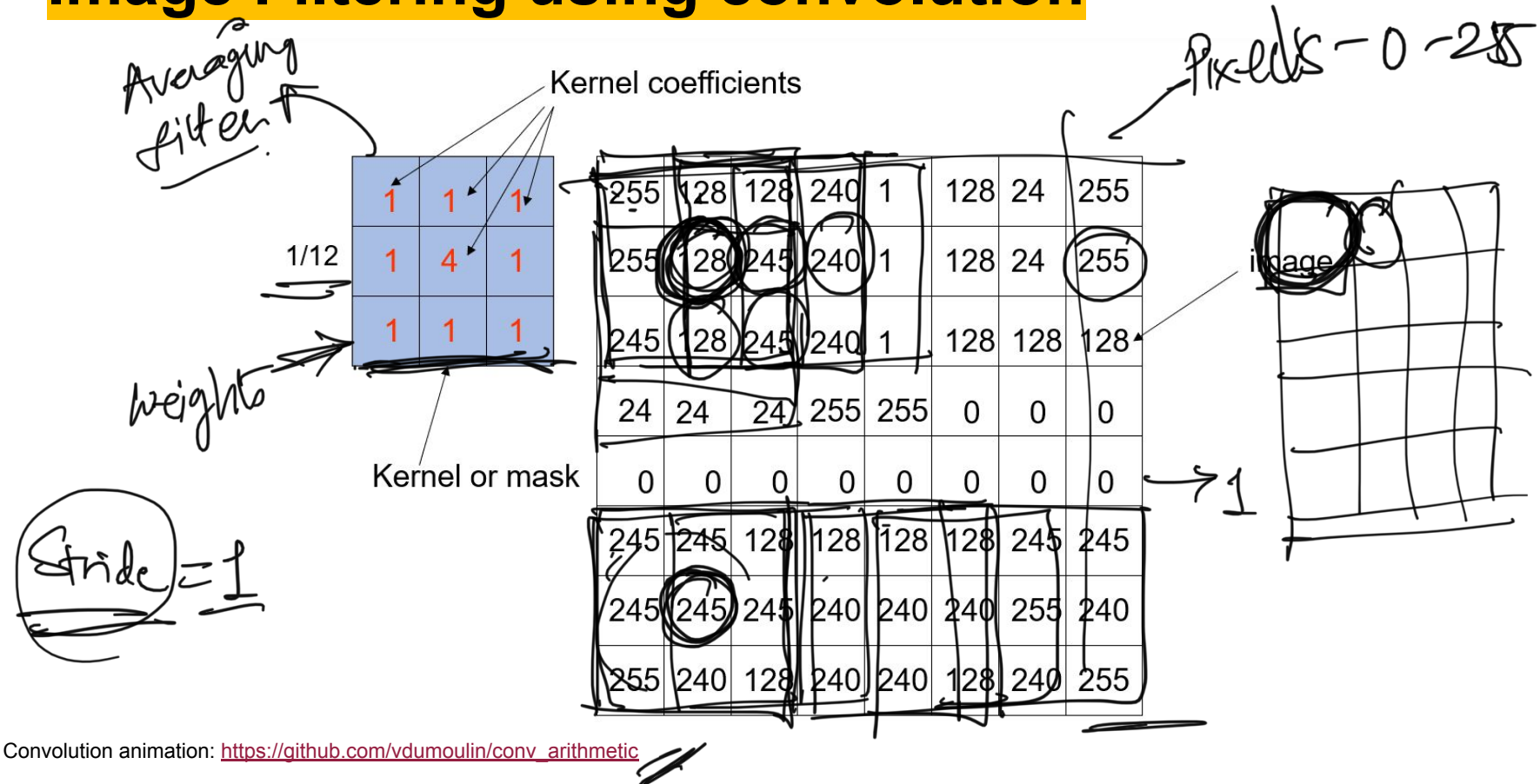


- Basic building block is the convolution for image processing
- Weight sharing - reduces parameters
- Learn the kernels based on the task.



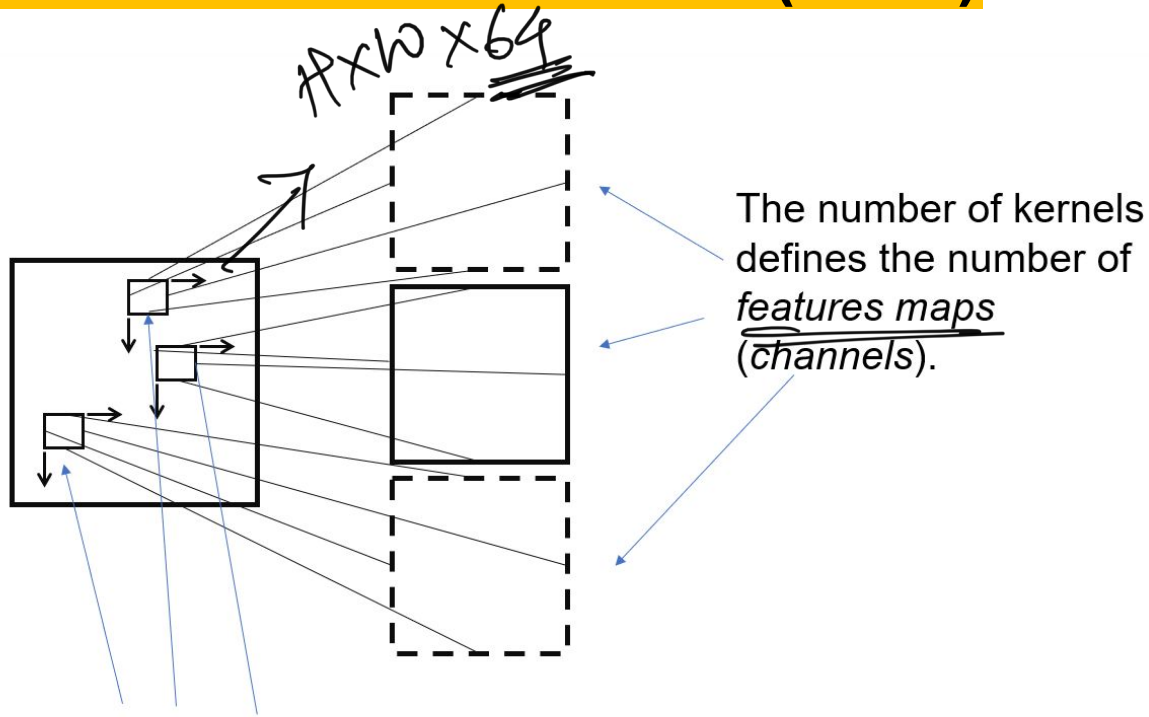
Edge (Vertical).

Image Filtering using convolution



Convolution animation: https://github.com/vdumoulin/conv_arithmetic

Convolutional Neural Network (CNN)



The sizes of the kernels define the *receptive fields*.

Summary of CNNs : https://cs231n.github.io/convolutional-networks/

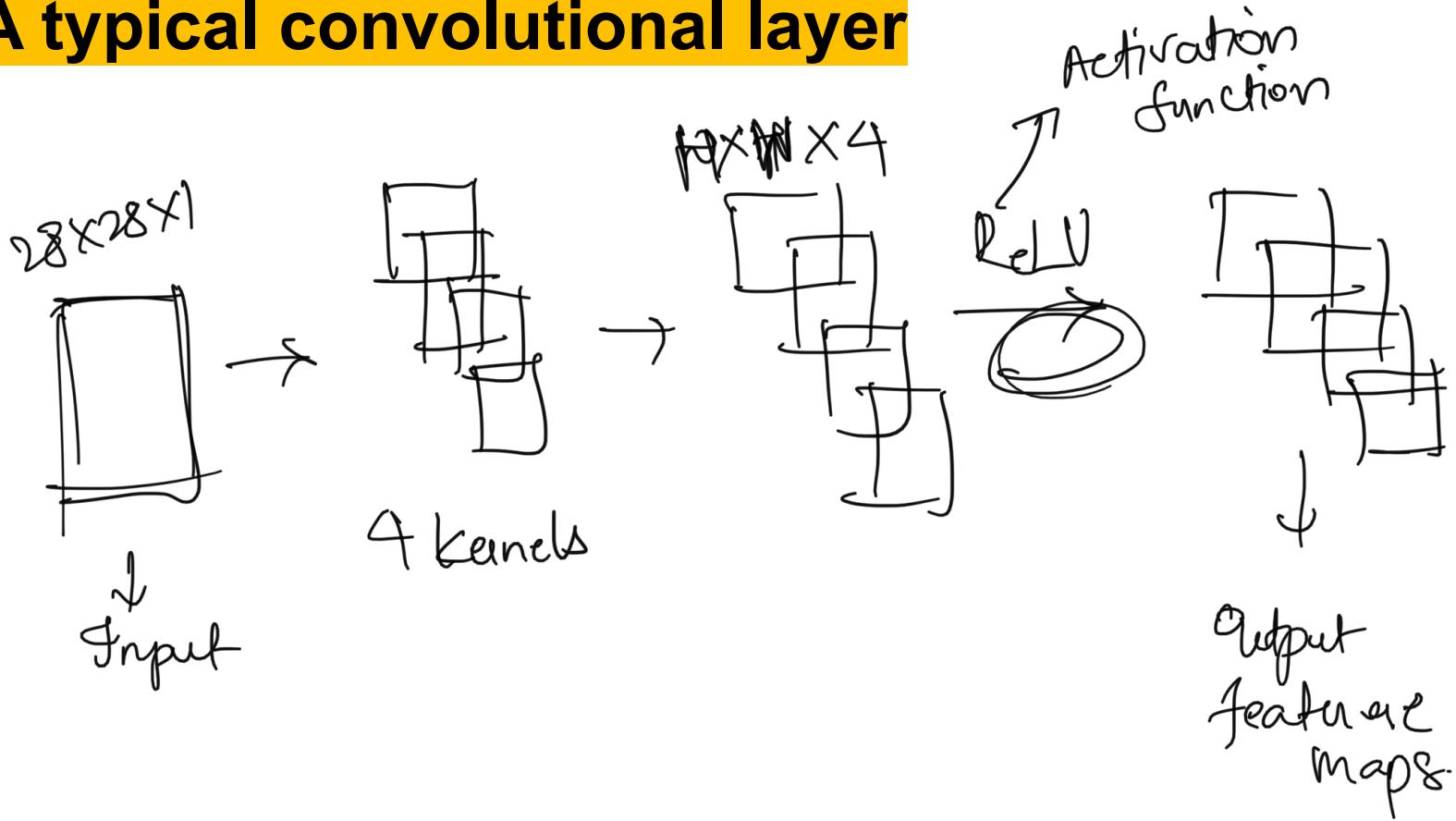
Padding in convolutional layer

- Zero-padding
  - Equally value the boundary pixels.
- Same / less reduced size of the output.

255	128	128	240	1	128	24	255
255	128	245	240	1	128	24	255
245	128	245	240	1	128	128	128
24	24	24	255	255	0	0	0
0	0	0	0	0	0	0	0
245	245	128	128	128	128	245	245
245	245	245	240	240	240	255	240
255	240	128	240	240	128	240	255

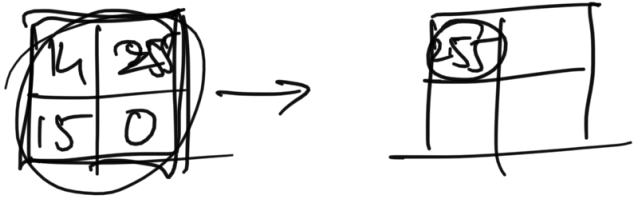
Summary of CNNs : https://cs231n.github.io/convolutional-networks/

A typical convolutional layer



Summary of CNNs : https://cs231n.github.io/convolutional-networks/

Pooling in CNN

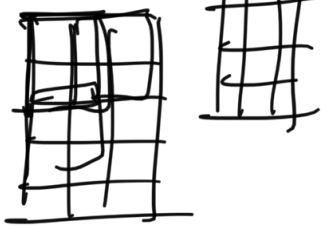


- Used to reduce the size of the output, thereby reducing the number of parameters and preventing overfitting.
- Done on every channel along the height and width of the output. So, the number of channels remain the same.

— Max Pooling

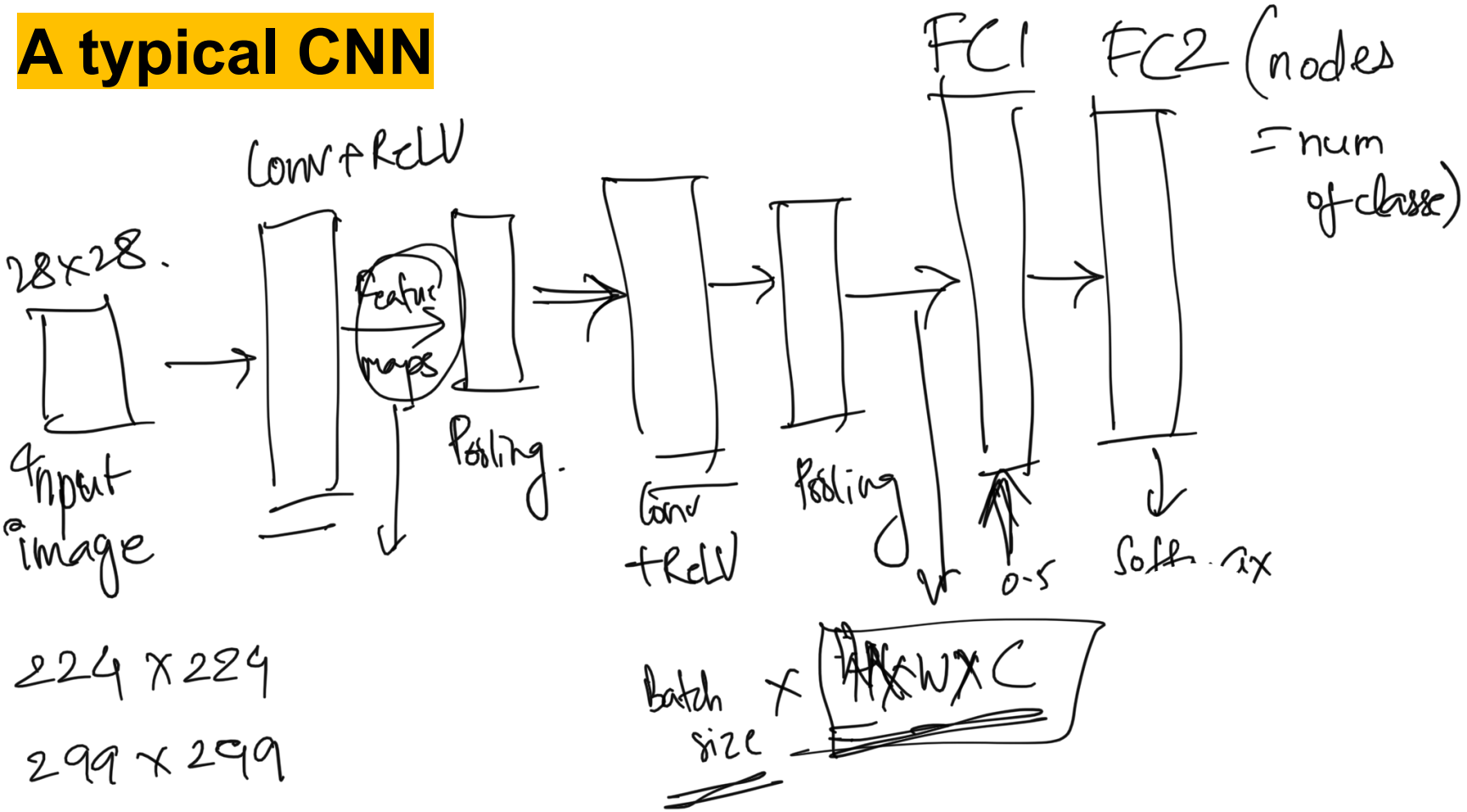
— Average Pooling.

— Kernel size, type of pooling, stride.



28x28 ⇒ 784

A typical CNN



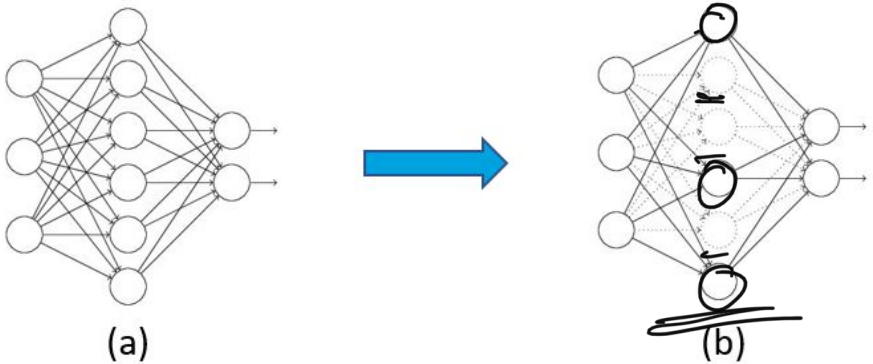
Why is regularization necessary?

- Huge parameter space - poor solutions
- Convergence to local minimum which leads to good performance on training data only.
- To overcome -
  - L2 norm of weights
  - Dropout
  - Batch normalization.

Dropout

- Obtain (b) by randomly deactivate some hidden nodes in (a).
- Reducing co-adaptation of neurons

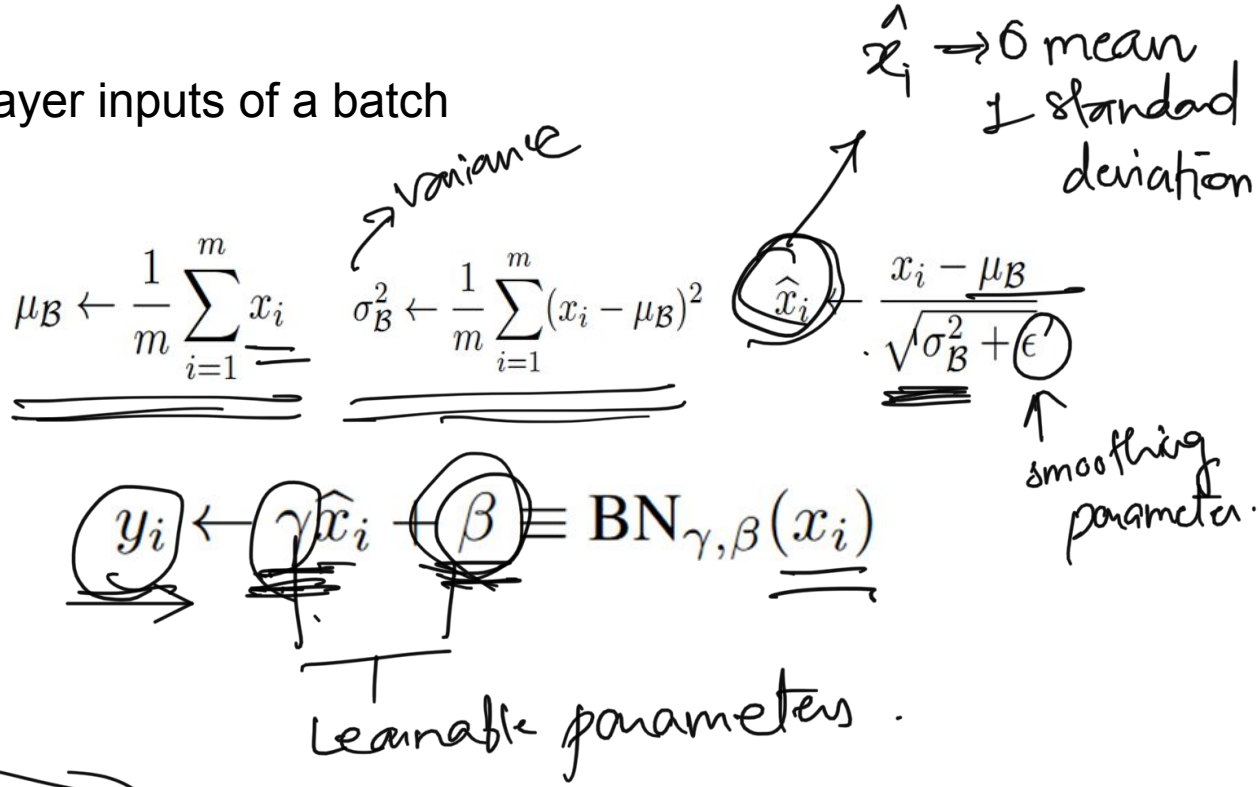
Probability of dropping a neuron. (0 - 1) - (p)



Refer for details on Dropout : https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Batch Normalization

- Normalizes layer inputs of a batch

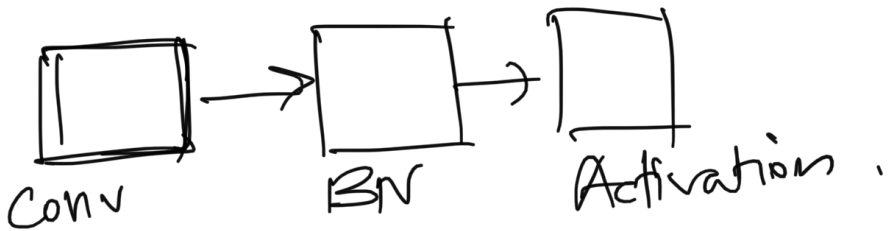


Paper about BN: https://arxiv.org/pdf/1502.03167.pdf

Batch Normalization - Advantages

- Use of higher learning rates
- Covariate shift is reduced
- Avoids vanishing gradients

- Makes the network less sensitive to weight initialization.



Paper about BN: https://arxiv.org/pdf/1502.03167.pdf

Questions?

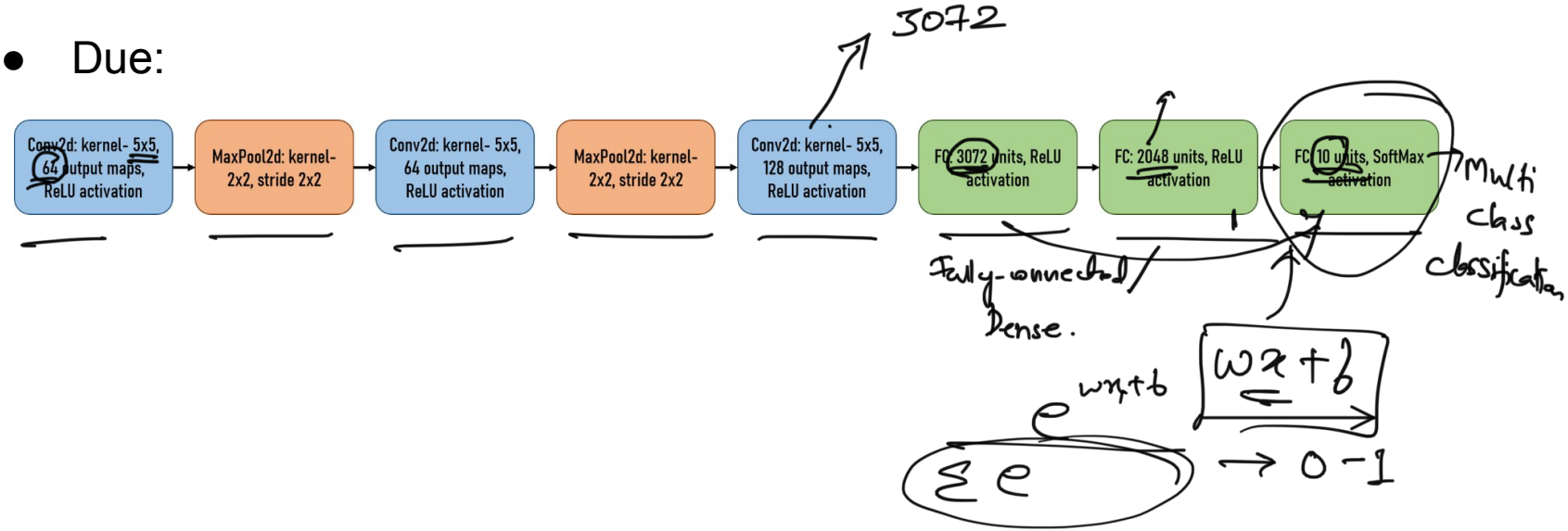
CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Neural Networks & Deep Learning

Table of contents

Project Part 3

- Goal - train a small convolutional neural network for classification of SVHN dataset.
- Due:

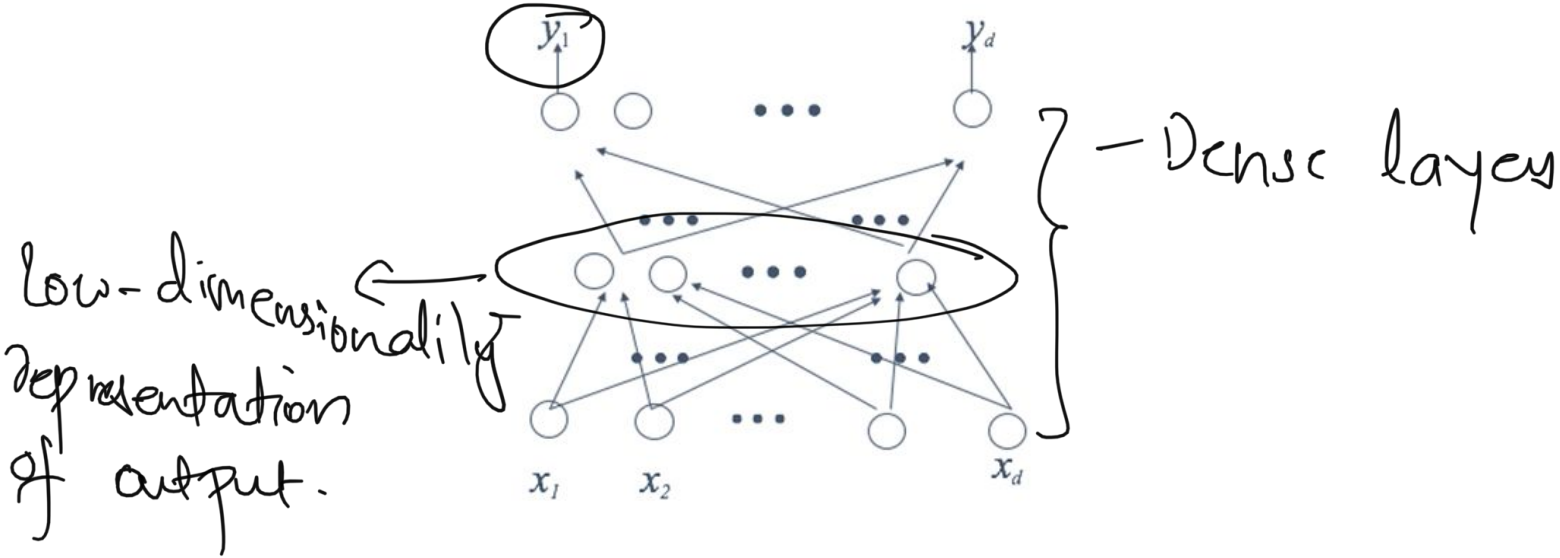


Project Part 3 - Software & Libraries

- Language - Python
- Deep learning libraries - Keras, Tensorflow, PyTorch
- Deliverables-
    - Code with proper comments.
    - A report including the plots for the learning/testing errors and the final classification accuracy.
    - Please submit the code and the report as separate files on Canvas. **Do not zip them.**

Autoencoder

- Train a network without supervision
- y_i being an approximation of x_i

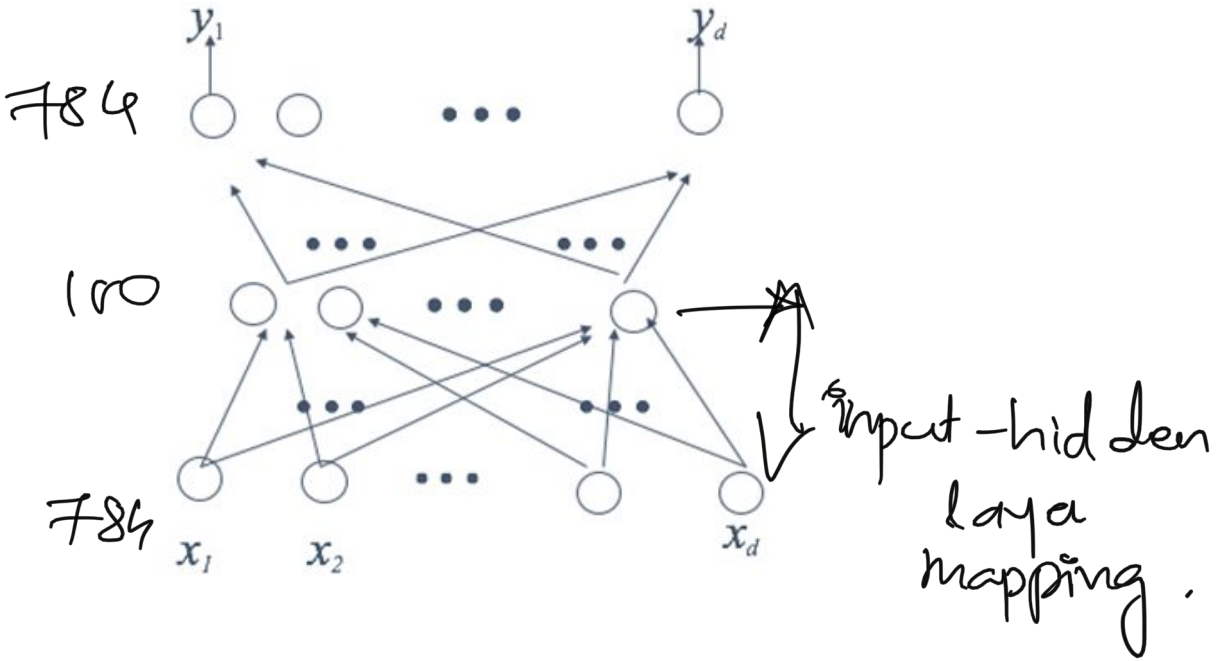


Autoencoder

- Perfect Autoencoder - Reconstructed output will be exactly same as input.
- Practical Scenario -
  - There is error in reconstruction.
  - Learning objective - Learn such that the reconstruction error is minimized.
    - MSE loss.

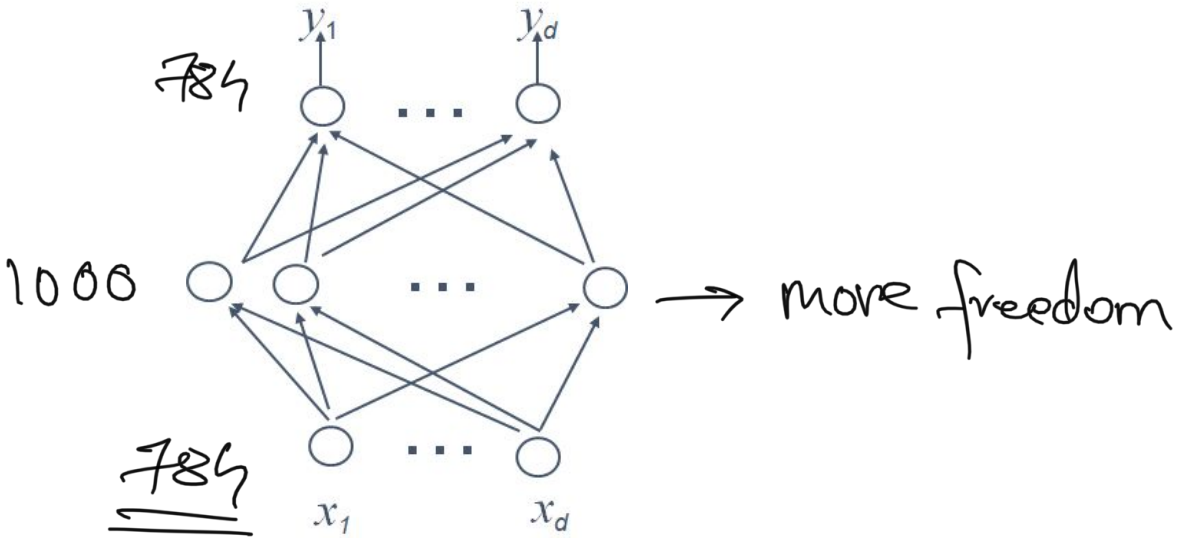
Autoencoder - 2 cases

- Case 1- Much fewer hidden nodes than input nodes



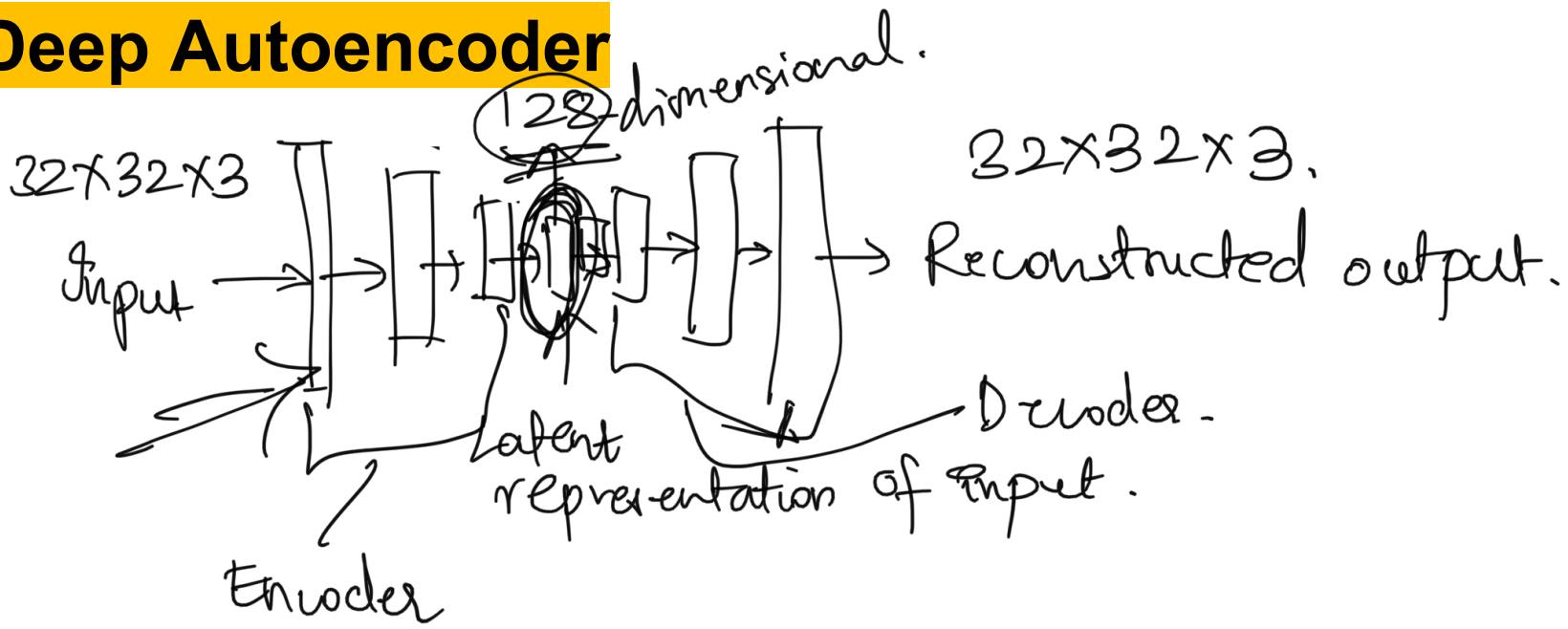
Autoencoder - 2 cases

- Case 1- Allow more hidden nodes than input



- Regularization.

Deep Autoencoder



- Encoder ⇒ Convolution
- Decoder ⇒ Deconvolution.

Deep Autoencoder

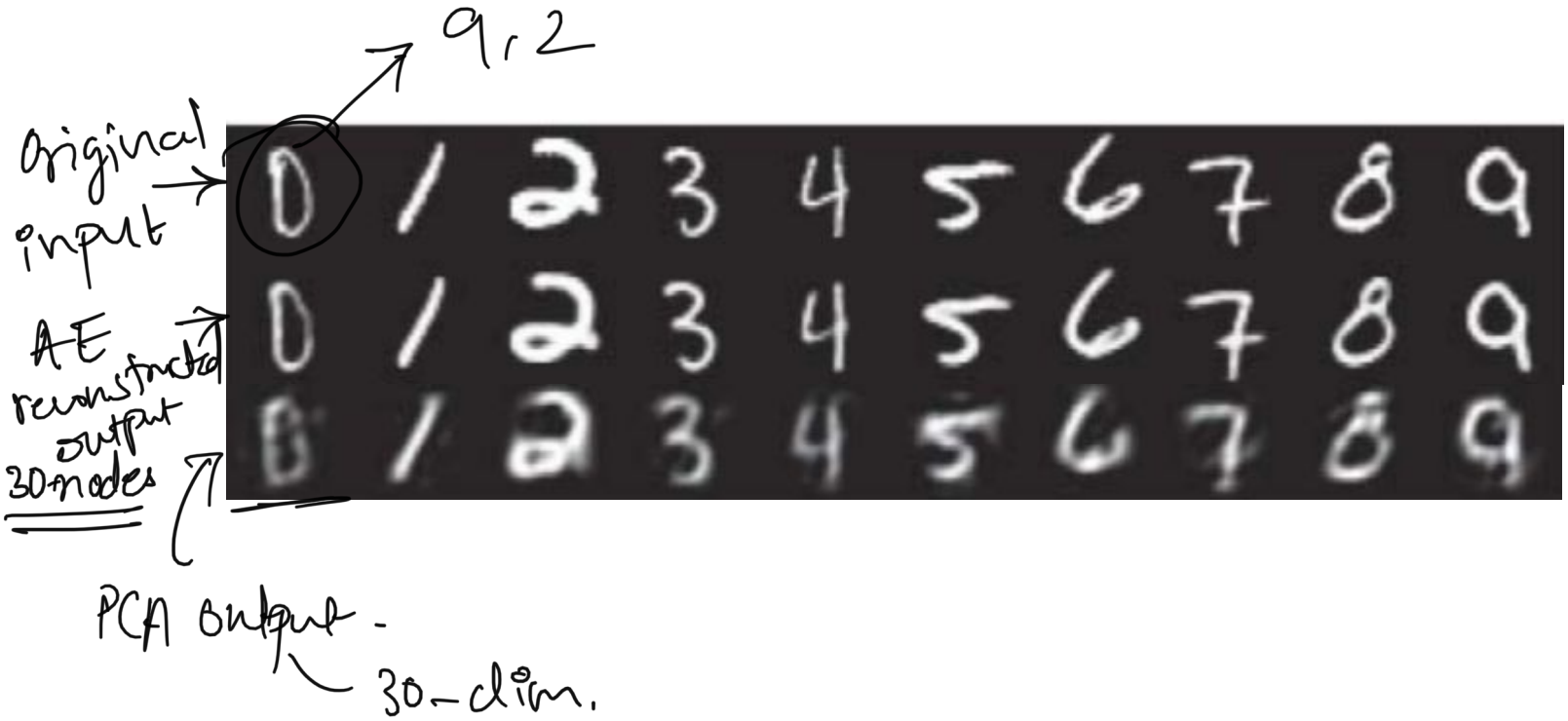
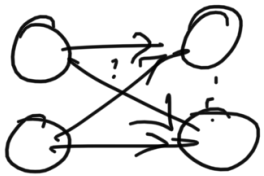
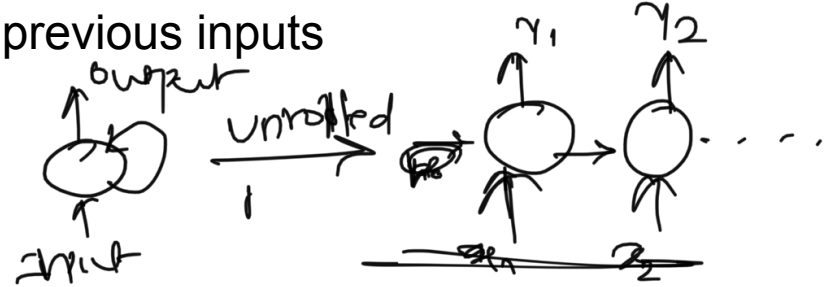


Image source: https://science.sciencemag.org/content/sci/313/5786/504.full.pdf?casa_token=rjwqf-A86OkAAAAA:9c2l-BOLY6kVFX3lnupw60aH6CkhKqGWjQlxtK3VGwaU_jOeFFjsV8u7SLt_bxtX-w1GP9Gql1OUtM8

Recurrent Neural Networks

- Allow directed cycles in connections between neurons
- Such networks could naturally model variable-length sequential data
- Influence of current as well as previous inputs





— very popular in NLP (Natural Language Processing)

— Vision — to process videos.

LSTMs, GRU

More about RNN: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

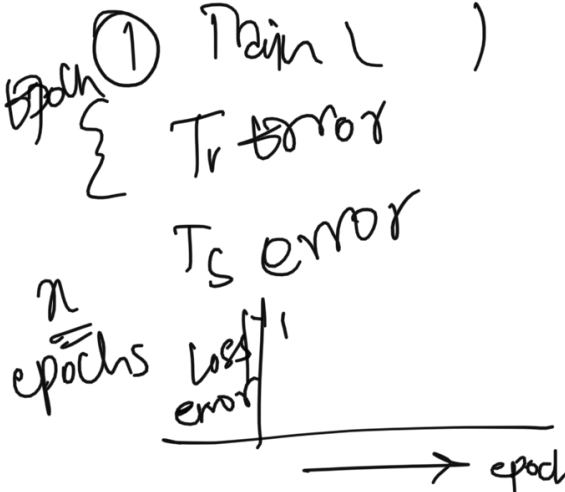
Questions?

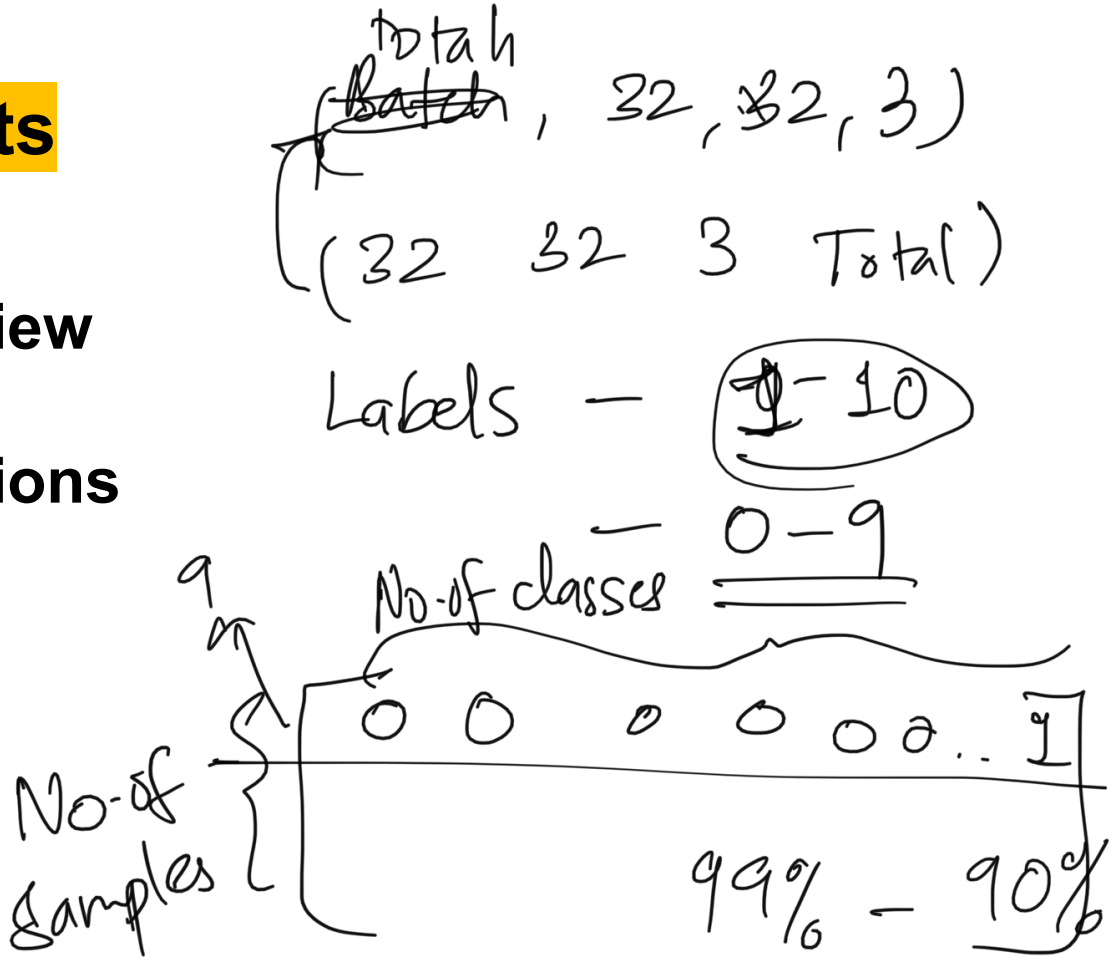
CSE 575: Statistical Machine Learning (Spring 2021)
Instructor: Nupur Thakur

Final Exam Review

Table of contents

1. Concepts Review
2. Sample Questions





Maximum Likelihood Estimation

- Given some training data and assuming a parametric model p(x|θ); what specific θ will fit/explain the data best?
- To consider all the samples denoted by D={x_1, x_2, ……, x_n}, assume that all the samples are i.i.d - independent and identically distributed.
- So, data likelihood represented by L(θ) is -

$$L(\boldsymbol{\theta}) = P(D|\boldsymbol{\theta}) = \prod_i P(x_i|\theta)$$

$$\widehat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta})$$

Naive Bayes

- The "naive" conditional independence assumption: each feature is (conditionally) independent of every other feature, given the label, i.e., $p(x_i \mid \{x_j \text{ for any } j \neq i\}, y) = p(x_i|y)$
- The predicted label is given by -

$$\hat{y} = \underset{y}{\operatorname{argmax}} P(y) \prod_{i=1}^{d} p(x_i|y)$$

Linear Regression

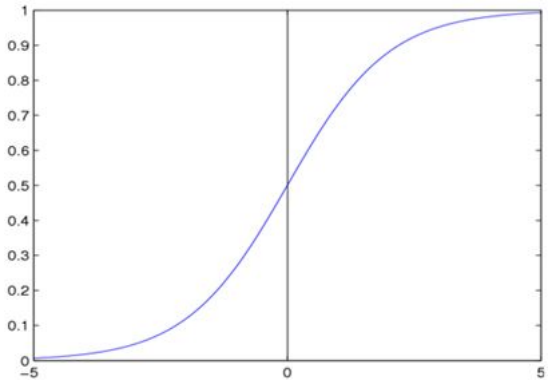
- Regression - A training set of n samples $<x^{(i)}, y^{(i)}>$ where $y^{(i)}$ is a continuous “label” (or target value) for $x^{(i)}$
- Linear regression - modeling the relation between y and x via a linear function $y \approx w_0 + w_1 x_1 + \ldots + w_d x_d = \mathbf{w}^t\mathbf{x}$
- The error is given as - $||e||^2 = ||y - X\,w||^2$

Logistic Regression

- Training set: n labelled samples <x(i), y(i)>
- Use the logistic function for modeling P(y|x), considering only the case of y∈{0,1}

$$P(y=0|\mathbf{x}) = \frac{1}{1+\exp\left(w_0+\sum_{i=1}^{d} w_i x_i\right)}$$

$$P(y=1|\mathbf{x}) = \frac{\exp\left(w_0+\sum_{i=1}^{d} w_i x_i\right)}{1+\exp\left(w_0+\sum_{i=1}^{d} w_i x_i\right)}$$



$$\sigma(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t}$$

Support Vector Machines (SVM)

- Key idea - To find the decision boundary such that the margin is maximized.
- Data - $\langle x^{(i)}, y^{(i)} \rangle$, $y^{(i)} \in \{-1,1\}$, $x^{(i)} \in \boldsymbol{R}^d$, for all i=1,…,n
- Plane equations-

- Margin -

Hidden Markov Model (HMM)

- Dynamic Bayesian network - modeling the process indexed by time.
- Two assumptions -
  - First-order Markov chain - $P(s^t=S_j \mid s^{t-1}=S_i)$
  - $a_{ij} = P(s^t=S_j \mid s^{t-1}=S_i)$, $1 \le i,j \le N$, for any t
- Specifying HMM - Θ, $\mathbf{\Omega}$, π, A, B

Problems in HMM

- For a given HMM $\Lambda=\{\Theta,\Omega,A,B,\pi\}$
  - Estimation of model parameters
  - Given an observation sequence **O**= $\{o^1,o^2, \dots ,o^k\}$, what is the most likely state sequence S = $\{s^1, s^2, \dots ,s^k\}$ that has produced O?
  - How likely is an observation O?

K-Means Clustering

Given: n samples, a number k.

Begin

initialize $\mu_1, \mu_2, \ldots, \mu_k$ (randomly selected)

do classify n samples according to nearest μ_i

recompute μ_i

until no change in μ_i

return $\mu_1, \mu_2, \ldots, \mu_k$

End

Clusters

EM algorithm for GMM

Gaussian Mixture Models

Expectation Maximization

1. Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k, and evaluate the initial value of the log likelihood.
2. **E step**. Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{9.23}$$

3. **M step**. Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{9.24}$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^{\mathrm{T}} \tag{9.25}$$

$$\pi_k^{\text{new}} = \frac{N_k}{N} \tag{9.26}$$

= Bishop's book.

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{9.27}$$

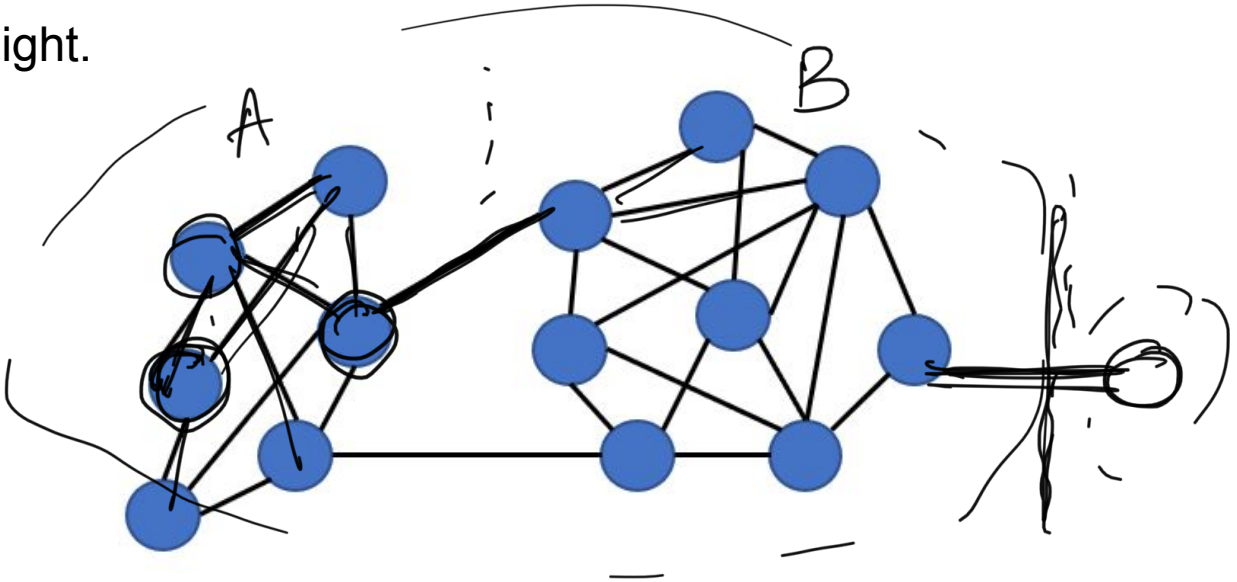
4. Evaluate the log likelihood

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{9.28}$$

and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.

Clustering as a Graph Partition

- Find a partition of a graph such that the edges between different groups have a very low weight while the edges within a group have high weight.



Types of cuts

$$\text{Min cut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} W(A_i, \overline{A}_i)$$

$$\text{RatioCut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{|A_i|} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{|A_i|}$$

$|A_i|$ ← No. of nodes in cluster

$$\text{Ncut}(A_1, \ldots, A_k) := \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A}_i)}{\text{vol}(A_i)} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{\text{vol}(A_i)}.$$

$$\text{MinMaxCut}(A_1, \ldots, A_k) := \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A}_i)}{W(A_i, A_i)}.$$

PCA Algorithm

— Dimensionality Reduction.

d dimensions.

1. Compute the $d \times d$ sample covariance matrix C

2. Find the eigenvalues and corresponding eigenvectors of C

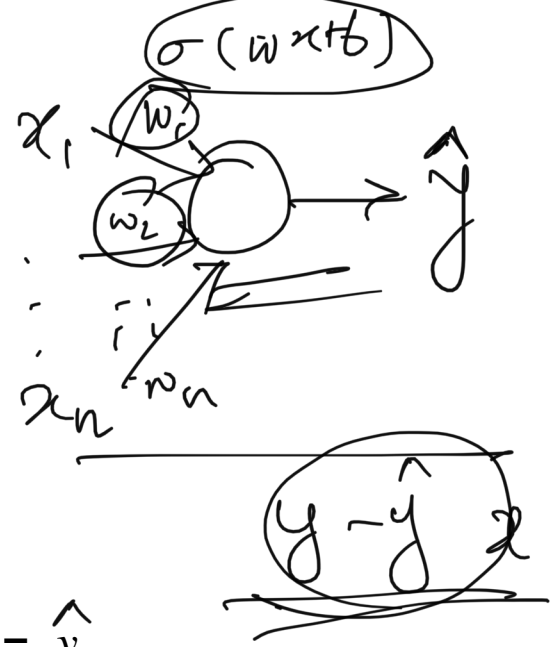
3. Project the original data onto the space spanned by the eigenvectors

   $d' \leq d$

   – The projection may be done onto a d'-dimensional subspace spanned by the first d' eigenvectors (ordered by the eigenvalue in descending order)

     • d' is determined by the desired accuracy

Learning in Perceptron



Iterate for t until a stop criterion is met

{

for each sample x_i with label y_i:

{

compute the output (y_i) of the network

estimate the error of the network $e(w(t))= y_i - \hat{y}_i$

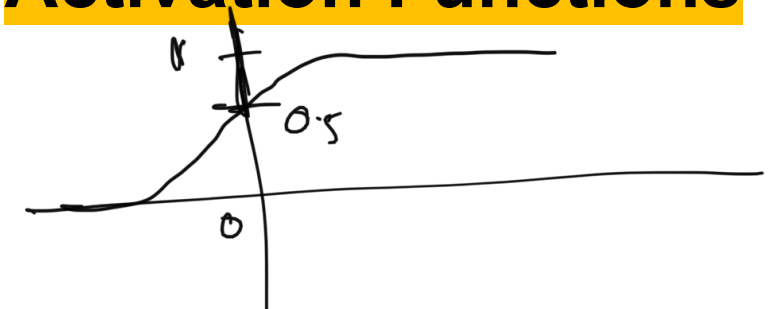
update the weight $w(t+1)=w(t)+e(w(t))x_i$

}

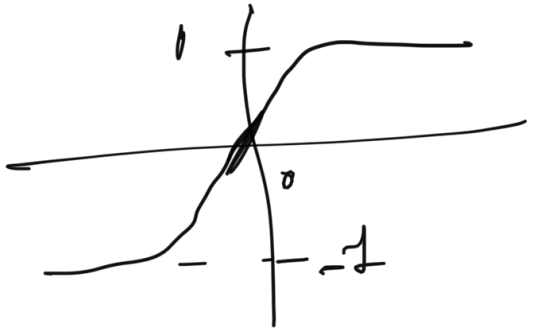
t++

}

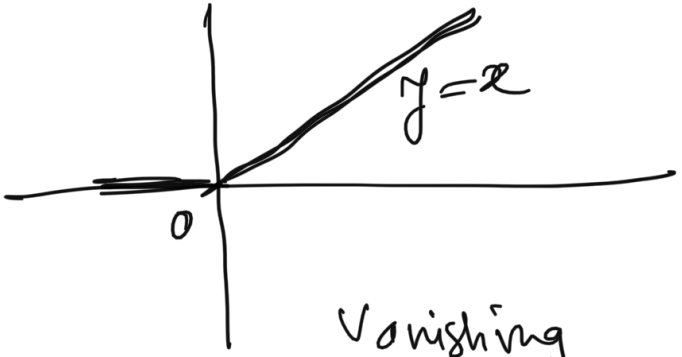
Activation Functions



Sigmoid

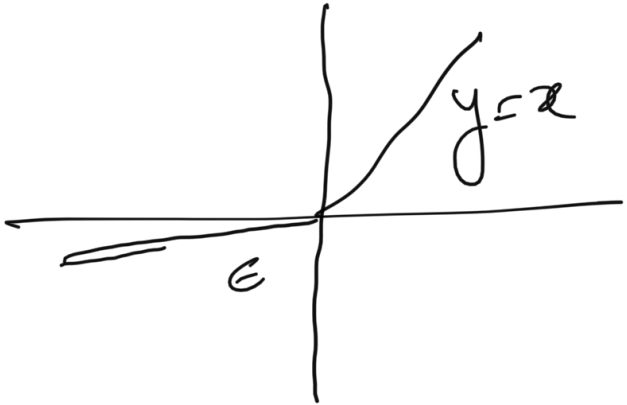


TanH



ReLU

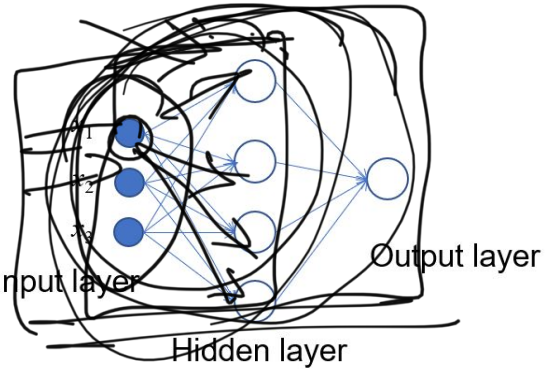
Vanishing gradients



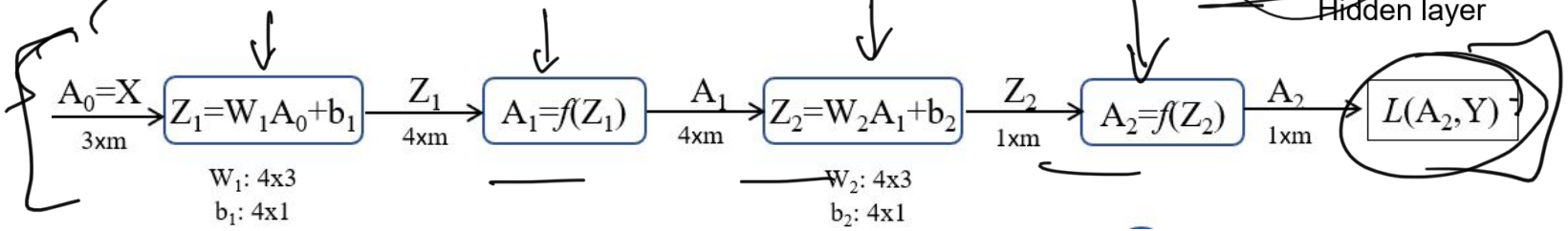
Leaky ReLU

Multi-layer perceptron

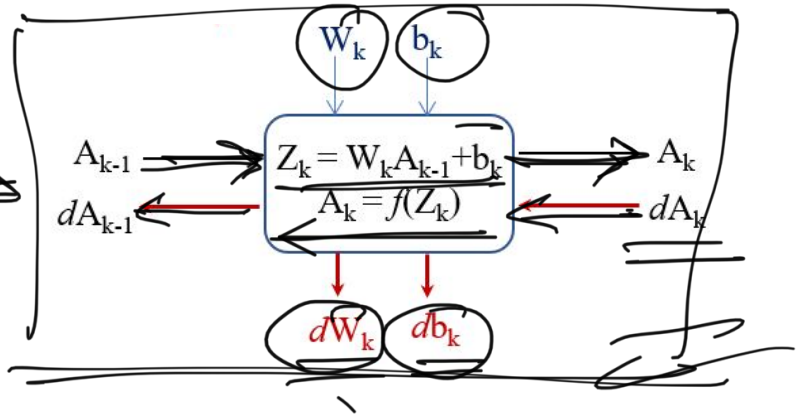
MLP Learning



Forward pass



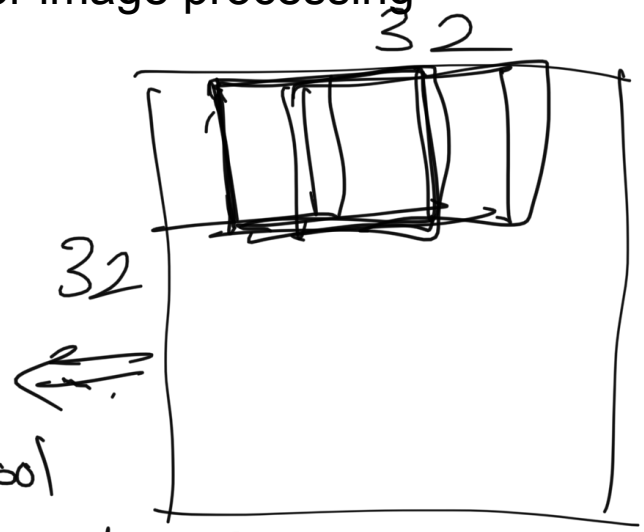
Backpropagation phase.



Convolutional Neural Network (CNN)

3X3X4

- Basic building block is the convolution for image processing
- Weight sharing - reduces parameters
- Learn the kernels based on the task.
- Layers -



convolutional layers.

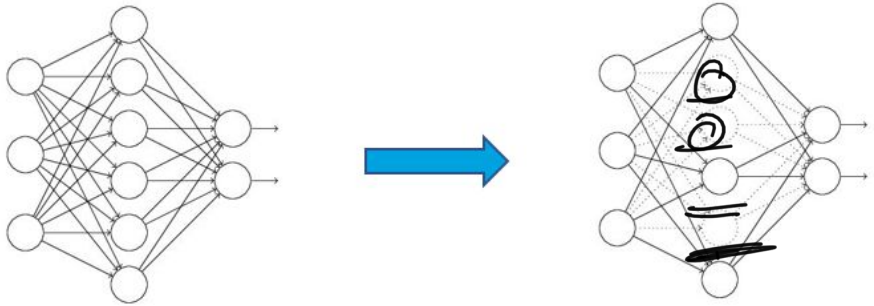
Pooling layer. — MaxPool

Activation layers - ReLU, Leaky ReLU.

Fully-connected — End of the network

Dropout

- Obtain (b) by randomly deactivate some hidden nodes in (a).
- Reducing co-adaptation of neurons



Batch Normalization

- Normalizes layer inputs of a batch

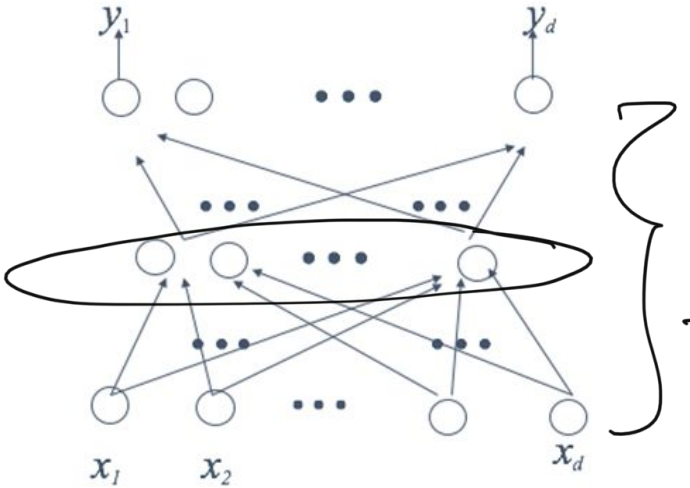
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i)$$

Paper about BN: https://arxiv.org/pdf/1502.03167.pdf

Autoencoder — Unsupervised.

- Train a network without supervision
- y_i being an approximation of x_i



— Encoder / Decoder Structure.

Questions?