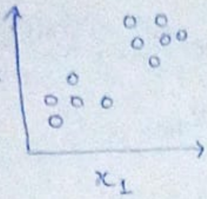
Week_3 Machine learning

Unsupervised Learning

· clustering:

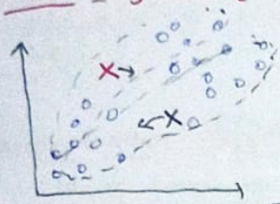


labels are not present.

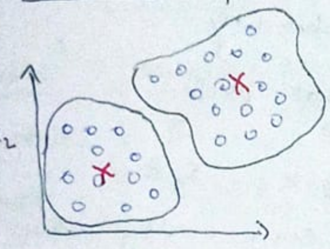
- grouping similar news
- DNA analysis
- Astronomical data analysis

Training set: $\{ x^{(1)}, x^{(2)} \dots \}$

K-means Algorithm



- Take a random guess to where are center of cluster **cluster centroid**
- Recompute the centroid
  It will compute the average location



⇒ Randomly initialize K cluster centroid $u_1, u_2 \dots u_K$

Repeat {
  # Assign points to cluster centroid
  for $i = 1$ to m
    $c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$
    → $\min_k \|x^{(i)} - \mu_k\|^2$
  # Move cluster centroid
  for $k = 1$ to K
    u_k := average (mean) of points assigned to cluster k
}

· optimization:

Cost function: minimizing it.

$$J(c^{(1)}, \dots c^{(m)}, u_1 \dots u_K) = \frac{1}{m} \sum_{i=1}^{m} \| x^{(i)} - \mu_{c^{(i)}} \|^2$$

· choosing the value of K

Evaluate K-means based on how well it perform on the later purpose

- **Anomaly detection**

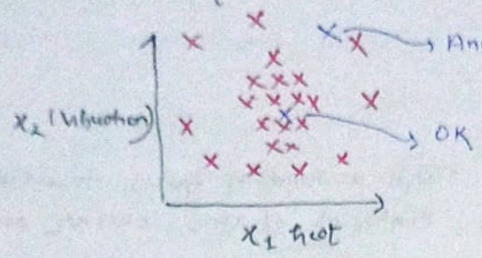
Eg. Aircraft engine features:

x_1 = heat generated

x_2 : vibration intensity

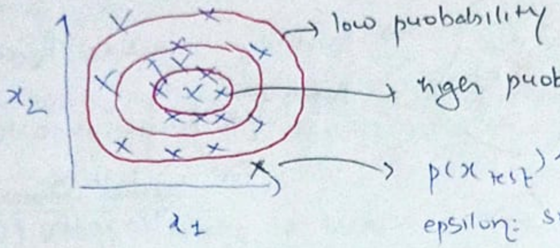
Dataset: $\{x^{(1)}, x^{(2)} \ldots x^{(m)}\}$

New engine: x_{test}



⇒ Density estimation:

→ probability of x being seen in dataset

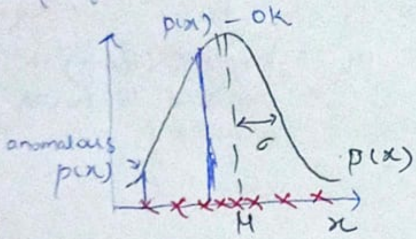


epsilon: small number

- **Gaussian (Normal) distribution.**

say x is a number

Probability of x is determined by a Gaussian with mean μ . variance σ^2



σ standard deviation

σ^2 variance

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

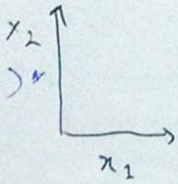
$$\boxed{\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \mu)^2}$$

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

Algorithm:

Training set: $\{\vec{x}^{(1)}, \vec{x}^{(2)} \ldots \vec{x}^{(m)}\}$

Each example $\vec{x}^{(i)}$ has n features



$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$p(\vec{x}) = p(x_1; \mu_1, \sigma_1^2) * p(x_2; \mu_2, \sigma_2^2) * p(x_3 \cdots) * \cdots p(x_n \quad)$

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

Anomaly detection algorithm

1. Choose n features x_i that you think might be indicative of anomalous ex.

2. Fit parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$

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

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

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

Anomaly if $p(x) < \varepsilon$

* **Recommender System:**

• Movie Recommendation

features $\begin{cases} n_u = 4 \\ n_m = 5 \\ n = 2 \end{cases}$

Movies	Alice(1)	Bob(2)	Carol(3)	Dave(4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance ...	5	?	?	0	1.0	0.01
Cute puppies ...	?	4	0	?	0.99	0
Nonstop car ...	0	0	5	4	0.1	1.0
Sword vs karate	0	0	5	?	0	0.9

(rating: Bob, Carol, Dave columns)

$x^{(1)} = \begin{bmatrix} 0.9 \\ 0 \end{bmatrix}$

for user 1: Predict rating for movie i as

$w \cdot x^{(i)} + b \leftarrow$ just linear regression

$x^{(3)} = \begin{bmatrix} 0.99 \\ 0 \end{bmatrix}$

$w^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad b^{(1)} = 0 \quad x^{3} = \begin{bmatrix} 0.99 \\ 0 \end{bmatrix}$

$w^{(1)} \cdot x^{(3)} + b^{(1)} = 4.95$ } appropr prediction

Cost function.

$$\min J(w^{(j)}, b^{(j)}) \Rightarrow \frac{1}{2m^{(j)}} \sum_{i:r(i,j)=1} (w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)})^2 + \underbrace{\frac{\lambda}{2m^{(j)}} \sum_{k=1}^{n} (w_k^{(j)})^2}_{\text{regularization term}}$$

to avoid overfitting

- Collaborative filtering Vs Content-based filtering

Recommend items to you based on rating of users who gave similar rating as you

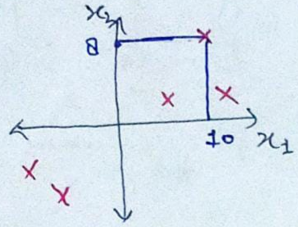
Recommend items to you based on features of user and item to find good match

- PCA algorithm:

coordinates $x_1 = 10$

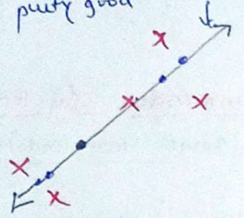
$x_2 = 8$

Can we choose a different axis?

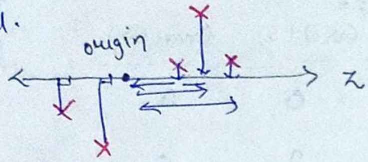


preprocess features

- Its pretty good



- Its pretty not good choice as our data is more spread.



Reinforcement learning

position of helicopter $\longrightarrow$ How to move control sticks

state s $\longrightarrow$ action a

$x \longrightarrow y$

reward function

↳ positive reward: helicopter flying well +1

negative reward: helicopter flying poorly −1000

Application:

- controlling robots
- factory optimization
- financial (stock) trading
- Playing game

ex

$100			★		$40
	0	0	0	0	
1	2	3	4	5	6

- Return: $\Rightarrow 0 + (0.9)0 + (0.9)^2\, 0 + (0.9)^3\, 100 = 72.9$

$\Rightarrow R_1 + \gamma R_2 + \gamma^2 R_3$ ---- until terminal state

Discount factor :- $\gamma = 0.9 \quad 0.99 \quad 0.999$

$\gamma = 0.5$

Return: $0 + (0.5)0 + (0.5)^2\, 0 + (0.5)^3\, 100 = 12.5$

Example of return

100	50 ←	25 ←	12.5 ←	6.25 ←	40
100	0	0	0	0	40

$\gamma = 0.5$

The return depends on the action you take

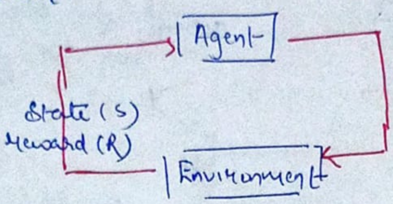
100	2.5 →	5 →	10 →	20 →	40
100	0	0	0	0	40

state (s) $\xrightarrow[\pi]{\text{policy}}$ action (a)

100	50 ←	25 ←	12.5 ←	20 →	40
100	0	0	0	0	40

Markov Decision Process (MDP)

This state future depends on where you are present now. Not how you reach.



Bellman Equation.

$Q(s,a) \Rightarrow$ Return if you
- start in state s.
- take action a (once)
- then behave optimally after that.

$R(1)=100 \quad R(2)=0 \quad \cdots \quad R(6)=40$

s: current state
a = current action
s': state you get to after taking action a
a': action that you take in state s'

$R(s)$ = reward of current state

$$\boxed{Q(s,a) = R(s) + \underbrace{\gamma}_{\text{discount factor}} \max_{a'} Q(s',a')}$$

$$\left\{ Q(s,a) = R_1 + \gamma [R_2 + \gamma R_3 + \gamma^2 R_4 + \cdots \right\}$$