Week 9

Problem Motivation

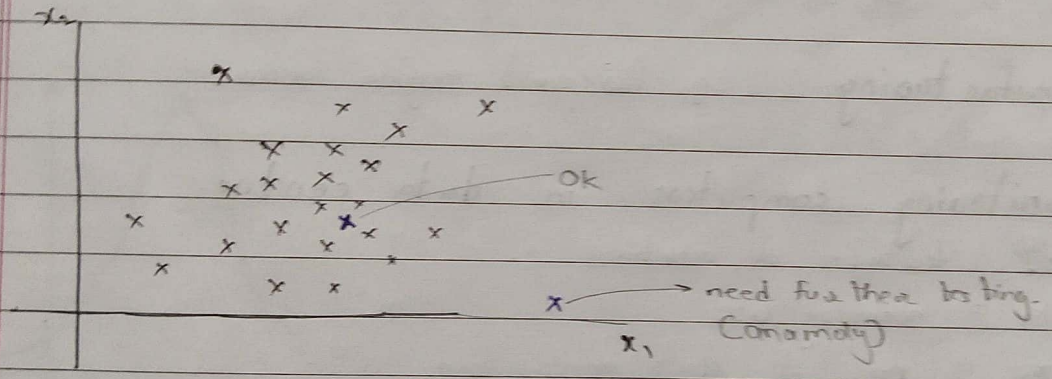
Anomaly detection

Anomaly detection problem.

x_1 = heat generated
x_2 = vibration intensity.

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

New engine : x_{test}

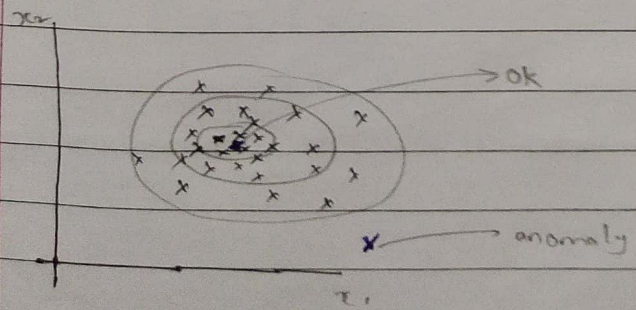


Density estimation

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

Is x_{test} anomalous?

We build a model → $p(x)$



$P(x_{test}) < \Sigma$ → flag anomaly.

$P(x_{test}) \geq \Sigma$ → ok

Anamoly detection example

- Fraud detection.
  $x^{(i)}$ = features of user i's activities
  Model = $p(x)$ from data.
  Identify unusual users by checking which have $p(x) < \varepsilon$

  $x_1 \rightarrow$ frequency of login
  $x_2 \rightarrow$ what pages visited.
  $x_3 \rightarrow$ no of posts on forum.
  $x_4 \rightarrow$ typing speed.

- Manufacturing. e.g. Aircraft engine anomaly

- Monitoring computers in data centre.
  $x^{(i)}$ features of machine i.
  x_1 = memory use
  x_2 = no of disk accesses/sec.
  x_3 = CPU load
  x_4 = CPU load/network traffic.

Q) $P(x) \leq \varepsilon$, suppose system is flagging too many things anomalous but actually are not [false positive].
✓ Try decreasing ϵ

-- @ Gaussian Distribution — — — —

Gaussian Distribution / Normal distribution.

★ Gaussian (Normal) Distribution.

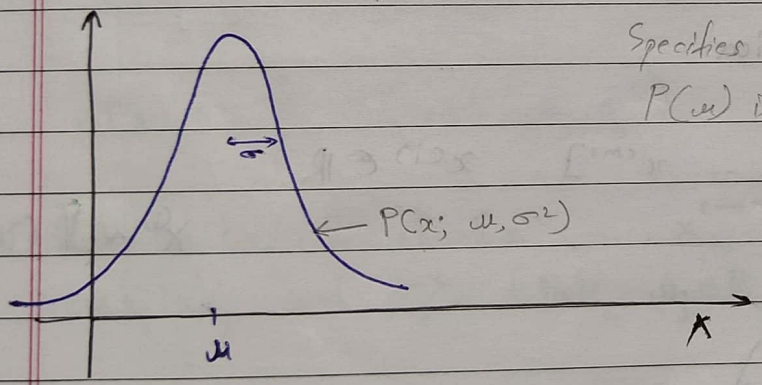
Say $x \in \mathbb{R}$. If x is a distributed Gaussian with mean μ, variance σ^2.

$$X \sim N(\mu, \sigma^2)$$

Parameters.

★ Gaussian Distribution / Gaussian Probablity density.

Curve is parameterized by μ, σ.



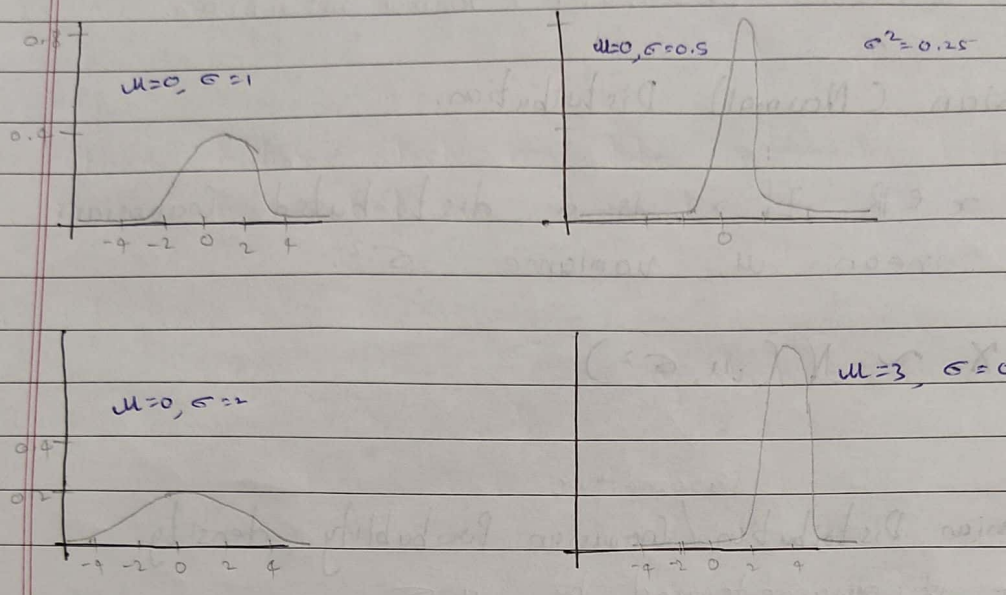
Specifies probablity of x taking different values, $P(\mu)$ is high

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

σ : standard deviation.

σ^2 : variance.

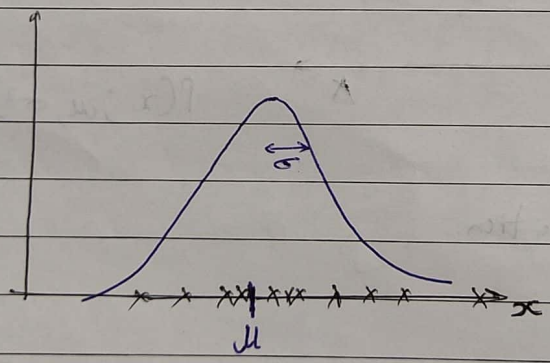
Gaussian distribution example.



Parameter estimation

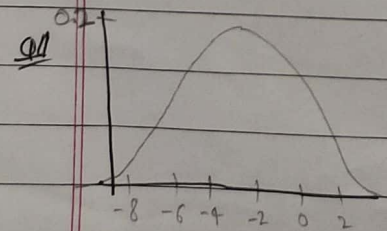
Dataset: $\{x^{(1)}, x^{(2)}, \ldots x^{(m)}\}$ $x^{(i)} \in \mathbb{R}$.

$x^{(i)} \sim \mathcal{N}(\mu, \sigma^2)$



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

μ, σ^2 are maximum likelihood estimates → $\frac{1}{m-1}$ is also used by some, $\frac{1}{m}$ is good practice



$$p(x) = \frac{1}{\sqrt{2\pi}\times 2} \exp\left(\frac{-(x+3)^2}{2\times 4}\right)$$

— @ Algorithm — — —
estimate $p(x)$

★ Density estimation.

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

Each example is $x \in \mathbb{R}^n$.

$x_1 \sim N(\mu_1, \sigma_1^2)$
$x_2 \sim N(\mu_2, \sigma_2^2)$
$x_3 \sim N(\mu_3, \sigma_3^2)$

$$p(x) = p(x_1; \mu_1, \sigma_1^2)\ p(x_2; \mu_2, \sigma_2^2)\ p(x_3; \mu_3, \sigma_3^2) \dots p(x_n; \mu_n, \sigma_n^2)$$

→ This eqⁿ assumes independent assumption on $x_1 \dots x_n$, But this works fine whether/not features are anywhere close to independent. If works fine.

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

Q Given a training set $\{x^{(1)}, \dots, x^{(m)}\}$, how would you estimate each μ_j and σ_j^2 (Note $\mu_j \in R$, $\sigma_j^2 \in R$)

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

☆ Anamaly detection algorithm.

1) Choose features x_i that you think might be indicative of anomalous ~~ema~~ example.
Choose features which may take unusually high/low value for what an anomalous example might look like.

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

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \quad \text{--- } \mu_j \text{ stands for mean of } i^{th} \text{ feature.}$$

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

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{bmatrix} = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

vectorized.

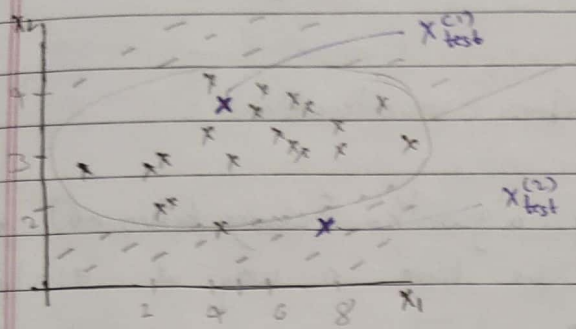
3) Given new example x, compute $p(x)$:

$$p(x) = \prod_{j=1}^{n} p(x_j; \mu_j, \sigma_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\,\sigma_j} \exp\left(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}\right)$$

Anomaly if $p(x) < \varepsilon$

Anomaly detection ~~proble~~ example.

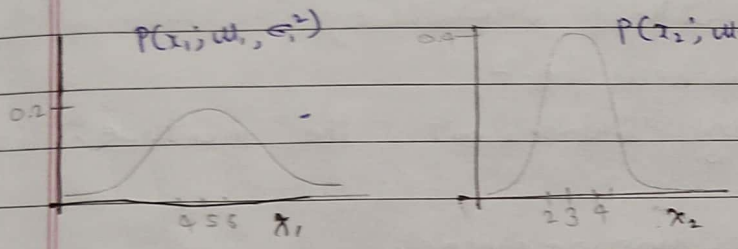
example lying outside are anomaly.



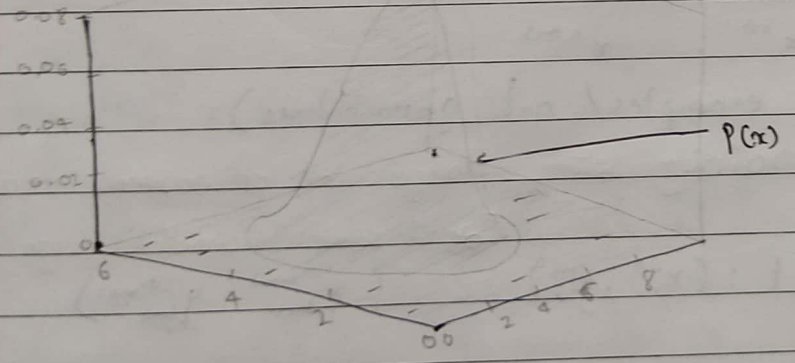
$\mu_1 = 5, \quad \sigma_1 = 2$

$\mu_2 = 3, \quad \sigma_2 = 1$

$\sigma_1^2 = 4, \quad \sigma_2^2 = 1$



$$p(x) = p(x_1; \mu_1, \sigma_1^2)\, p(x_2; \mu_2, \sigma_2^2)$$



All examples having hight above surface corresponds to non-anomalous examples as normal example.

-- (a) Developing and Evaluating an Anomaly detection system.

how to build model.

The importance of real-number evaluation.

When developing a learning algorithm (choosing features, etc), making decisions is much easier if we have a way of evaluating our learning algorithm.

Run a algo with & without a feature,
If we get a number back,
it is easy to say include / not the feature.

Anomaly detection is unsupervised.

Assume we have some labelled data of anomalous and non-anomalous example
($y=0$ if normal, $y=1$ if anomalous)

This brings us our problem closer to supervised.

Training set : $x^{(1)}, \ldots x^{(m)}$
(assume normal examples / not anomalous)
Unlabelled training set $\Rightarrow$ large set of normal examples.
(ok if some anomalous slips in training set)

To evaluate anomaly detection algo

Cross Validation set : $(x_{cv}^{(1)}, y_{cv}^{(1)}), \ldots (x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$

Test set : $(x_{test}^{(1)}, y_{test}^{(1)}), \ldots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

→ Will contain few anomalous examples, which are known.

Aircraft engines motivating examples.

10,000 good (normal) engines.
20 flawed engines (anomalous)

That's how we split the data

Training set : 6,000 good engines ($y=0$) $p(x) = p(x_1; \mu_1, \sigma_1^2) \dots p(x_n; \mu_n, \sigma_n^2)$
CV : 2600 good engines ($y=0$), 10 anomalous ($y=1$)
Test : 2000 good engines ($y=0$), 10 anomalous ($y=1$)

Alternative

Training set : 6000 good engines
CV : 4,000 good engines ($y=0$), 10 anomalous ($y=1$)
Test : 4,000 good engines ($y=0$), 10 anomalous ($y=1$)

→ This is not recommended using same set of $y=0$ in both CV & Test.
Even same set of $y=1$, 20, 20 can be done for CV, test
but this is not recommended.

Algorithm evaluation.

- Fit model $p(x)$ on training set $\{x^{(1)}, \dots, x^{(m)}\}$
- On a cross validation/test example x, predict y

$$y = \begin{cases} 1 & \text{if } p(x) < \varepsilon \text{ (anomaly)} \\ 0 & \text{if } p(x) \geq \varepsilon \text{ (normal)} \end{cases}$$

$(x_{test}^{(i)}, y_{test}^{(i)})$

Now anomaly problem is predicting y, just like supervised.

Possible evaluation metrics:
- ~~True +ve~~ Confusion matrix (TP, FP, TN, FN)
- Precision / recall
- F1 - score → single number

Can also use CV set to choose parameter ε.
Choose ε that maximizes F1 score. ← Try many ε

Q) Suppose you have fit a model $p(x)$. When evaluating on the cross validation set or test set, your algorithm predicts $y = \begin{cases} 1 & p(x) \leq \varepsilon \\ 0 & p(x) > \varepsilon \end{cases}$ | Is classification accuracy a good way to measure algo's performance.

→ No, because of skewed classes (so an algo that always predicts $y=0$ will have high accuracy)

→ classification accuracy.

So use, FP, FP, FN, TP, TN; Precision/recall; F_1-score.

--④ Anomaly detection vs Supervised learning.

Why didn't we use logistic regn to solve anomaly detection, if we had labelled data.
When to use logistic / anomaly DP 3

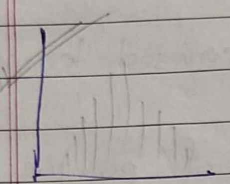
Anomaly detection	vs	Supervised learning
Very small number of positive examples ($y=1$) (0-20 is common)		large no of positive & negative examples.
Large number of negative ($y=0$) examples. → fit $p(x)$		
Many different "types" of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like. • Many way to go wrong aircraft engine. • Difficult to learn how anomaly look like from small set of positive examples. • Future anomaly may look nothing like previous anomaly. • So model is train on negative examples rather than hard positive eg.		Enough positive examples for algorithm to get a sense of what positive examples are like, future positive examples likely to be similar to ones in training set. e.g. • email spam classifier. • Weather prediction • Cancer classification.
Future anomaly may look nothing like any of the anomaly e.g we've seen so far. e.g. • Fraud detection • Manufacturing (engine) • Monitoring machines in data centers.		If we have enough data for positive e.g's we can shift from anomaly to supervised learning. It was skipped. easy.

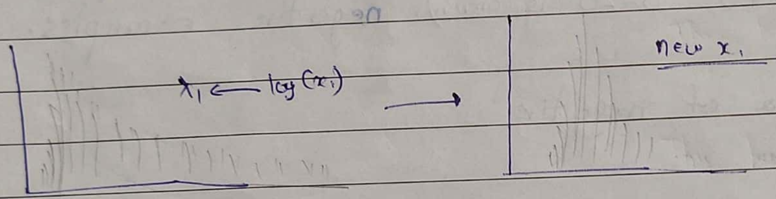
-- @ Choosing what feature to use (optional)

Features matters very much.

Make sure if data is gaussian / normally distributed.
If not play with it to make gaussian.

$x_1 \leftarrow \log(x_1)$
$x_2 \leftarrow \log(x_2+1)$
$\log(x_2+c)$
$x_3 \leftarrow \sqrt{x_3}$
$x_4 \leftarrow x^{\frac{1}{4}}$
That's how we play.





Error analysis for supervised

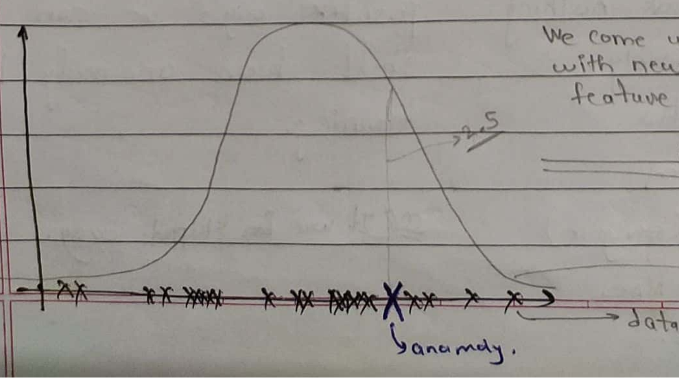
Train model, test on CV can we get extra features that we got it wrong in CV to make model better.
Same here.

(similar)

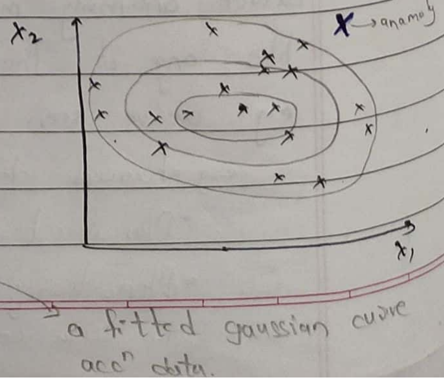
☆ Error analysis for anamoly detection — Procedure

We want { $p(x)$ large for normal examples x.
$p(x)$ small for anomalous examples x.

Most common problem.
→ $P(x)$ is comparable (say, both large) for normal & anomalous eg.



We come up with new feature x_2 ⟹



If existing feature is failing to flag anomalous as anomaly try to find new features.

Choose features that will take very large / very low value, in event of an anomaly.

* Monitoring Computer in data centre
  - x_1 = memory use of PC.
  - x_2 = number of disk access/sec
  - x_3 = CPU load
  - x_4 = network traffic.

new features to capture anomaly be like.

$$x_5 = \frac{\text{network load}}{\text{network traffic}} \quad \frac{(x_1)^2}{n \, t}$$

OR

Q If model outputs a large value of $p(x)$ for many value of $p(x)$ for many normal examples and for many anomalous examples in your cross validation dataset.
Which of the following changes to your algo. what is most likely to help

- ✓ Try comming up with more features b/w normal & anomalous e.g.
- Try using fewer features.

-- (8) Multivariate Gaussian Distribution (optional)

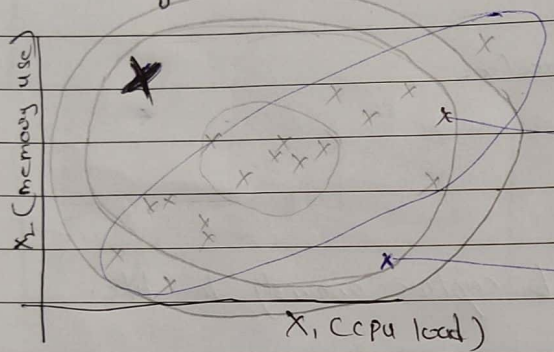
extension of anomaly detection

→ uses multivariate GD

Advantages

Disadvantages

motivating example: Monitoring machines in data centre.



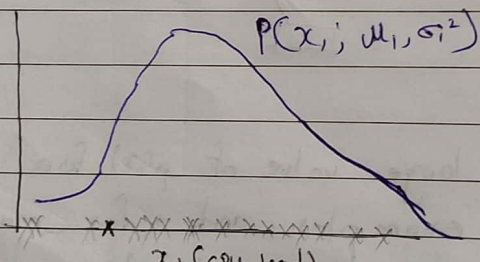
Somewhat linearly.

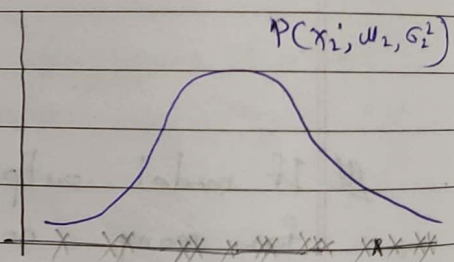
This has high probability.

The circles doesnot make us realize this

This has low probability.

We now modify to get rid of this using multivariate



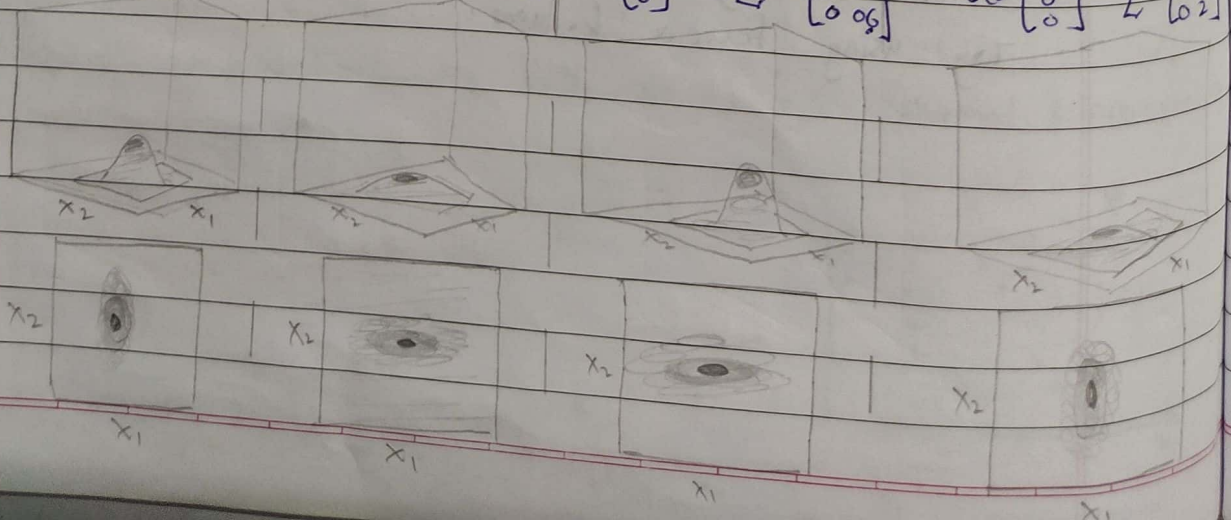


x_1 vs x_2 example looks anamolous

But in x_1, x_2 seperately, e.g. not look like an anomalous.

skip this →

$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 0.6 & 0 \\ 0 & 1 \end{bmatrix}$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix}$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$



★ Multivariate Gaussian (Normal) distribution.

$x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \ldots$, etc separately

Model $p(x)$ all in one go.

Parameters: $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

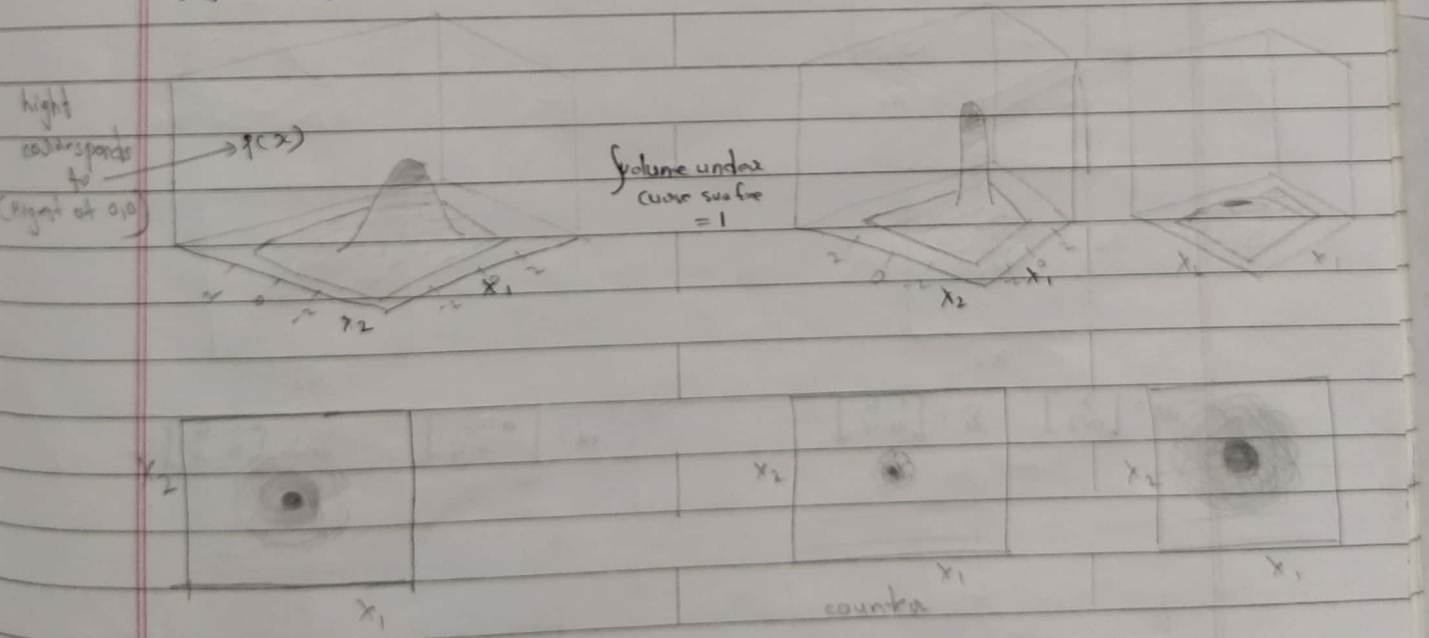
→ same form P.S.A.

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

$|\Sigma|$ = determinant of Σ | det(Sigma)

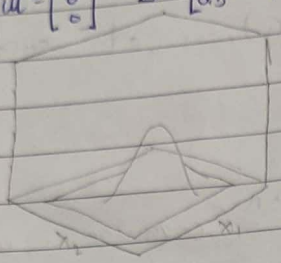
Multivariate Gaussian (Normal) examples.

$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $\quad$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma = \begin{bmatrix} 0.6 & 0 \\ 0 & 0.6 \end{bmatrix}$ $\quad$ $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$

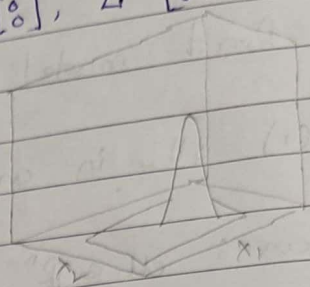


Go To Previous Page. and Read skip This →

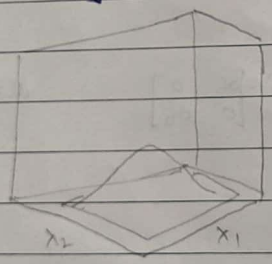
$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$

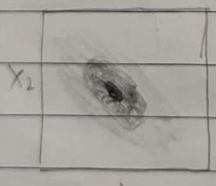


$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$

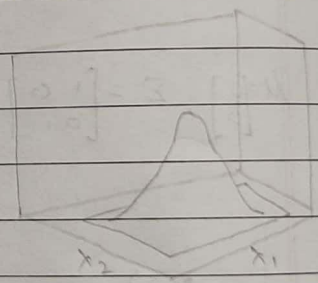


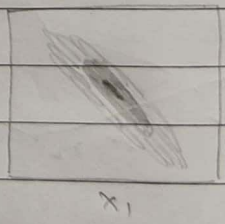
$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$



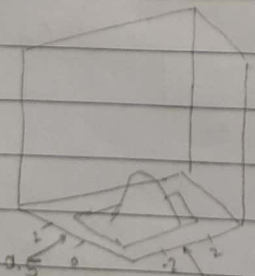


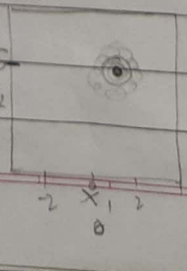
$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$



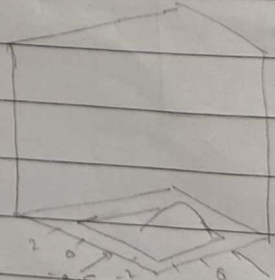


$\mu = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

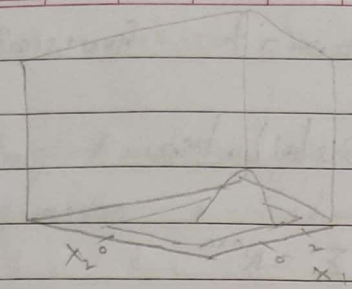


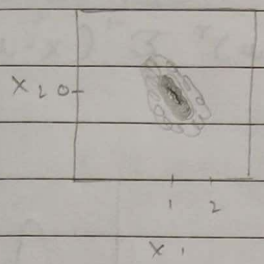


$\mu = \begin{bmatrix} 1.5 \\ -0.5 \end{bmatrix}$ $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$



4





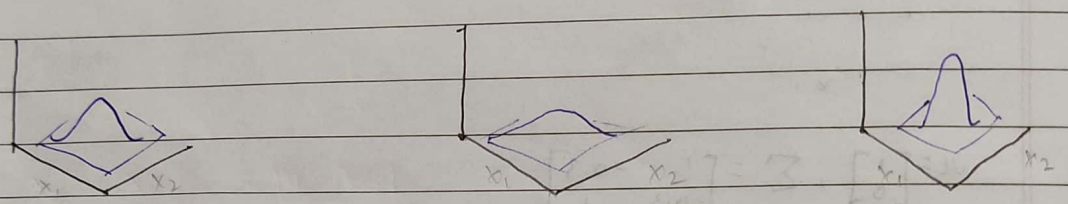
$$\mu = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & -0.3 \\ -0.3 & 1 \end{bmatrix}$$

--(8) Anomaly Detection using Multivariate Gaussian Distribution.

Multivariate Gaussian (Normal) Distribution.

Parameters μ, Σ $\quad \mu \in \mathbb{R}^n, \ \Sigma \in \mathbb{R}^{n \times n}$

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



Parameter fitting:

Given training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\} \leftarrow x \in \mathbb{R}^n$

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

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

Anamoly detection using multivariate.

1) Fit model $p(x)$ by setting.

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

2) Given a new example x, compute.

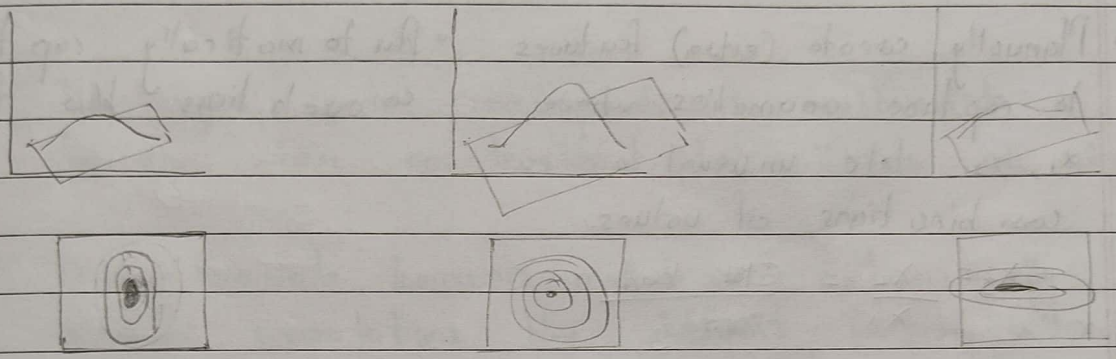
$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

Flag anamoly if $p(x) < \varepsilon$

☆ Relationship to original model.

Original model is special case of multivariate with counters of gaussian constrained to be always axis alligned.

Original model : $p(x) = p(x_1; \mu_1, \sigma_1^2) * p(x_2; \mu_2, \sigma_2^2) * \cdots * p(x_n; \mu_n, \sigma_n^2)$



$\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$

This will give diagonal scoreatch.

Corresponds to multivariate Gaussian

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

where $\Sigma = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \sigma_2^2 & \\ & & \ddots \\ 0 & & \sigma_n^2 \end{bmatrix}$

Page No: 170

Date: / /

Original model vs Multivariate Gaussian.

$p(x_1; \mu_1, \sigma_1^2) \times \cdots \times p(x_n; \mu_n, \sigma_n^2)$

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

used more often

used somewhat, but it can capture correlations b/w features

- Manually create (extra) features to capture anomalies where x_1, x_2 take unusual combinations of values.

  $x_3 = \frac{x_2}{x_1} = \frac{\text{CPU load}}{\text{memory}}$

  If we got time to manually create extra feature, original works.

- Flu to matically cap tures co rrela tions b/w features

$\Sigma \in \mathbb{R}^{n \times n}$ Σ^{-1} is tough

- Computationally cheaper (alternatively, scales better to large n). n=10,000 / 100,000

- Computationally more expensive.

- Ok even if m (training set size) is small.

- Must have $m > n$ or else Σ is non-invertible

  requirement

Recommended

$m \geq 10n$

$\Sigma \sim \frac{n^2}{2}$ no. of parameters

In multivariate if Σ is singular/non-invertable
→ ~~Fails~~ fails $m > n$ cond^n
2) We got redundant features $x_1 = x_2$ | $x_3 = x_4 + x_5$

Q) consider anamoly detection using training set $\{x^{(1)}, \ldots x^{(m)}\}$ where $x^{(i)} \in \mathbb{R}^n$.

✓ The original model corresponds to multivariate Gaussian where countors of $p(x; \mu, \Sigma)$ are axis alligned.

✓ The multivariate Gaussian model can automatically capture correlations b/w different features in x.

✓ The original model can be more computationally efficient than the multivariate Gaussian model, and thus might scale better to very large value of n (no of features)

(Recommender system)

(a) Problem formulation

Why? 1) Important.
2) Big idea → Algorithm, just to learn what features to use.

algo, just to learn what feature to use, ~~and~~ by itself recommender system is one of them.

Eg. Predicting movie ratings.

(simpler)

Users rates movies using 0 to 5 stars

movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)
LOL	5	5	0	0
RF	5	?	?	0
CPOL	?	4	0	?
NCC	0	0	5	4
Svsk	0	0	5	?

n_u = no of users.
n_m = no of movies.
$r(i,j) = 1$ if user j has rated movie i
$y(i,j)$ = rating given by user j to movie i (defined only if $r(i,j) = 1$)

$n_u = 4$, $n_m = 5$

Try to predict (?)

Q) $n_m = 2$, $n_u = 3$

	User 1	User 2	User 3
movie 1	0	1	?
movie 2	?	5	5

→ ✓ $r(2,1) = 0$, $y^{(2,3)}$ = undefined ← Recommender system goal is to find this.

Content based recommandations

Content based recommander systems.

	Movie	Alice (1) $\theta^{(1)}$	Bob (2) $\theta^{(2)}$	Carol (3) $\theta^{(3)}$	Dave (4) $\theta^{(4)}$	x_1 (Romance)	x_2 (Action)
$x^{(1)}$ →	Love at last 1	5	5	0	0	0.9	
$x^{(2)}$ →	Romance forever 2	5	2	2	0	1.0	0.1
$x^{(3)}$ →	Cute puppies of love 3	? 4.95	4	6	2	0.99	0
$x^{(4)}$ →	Nonstop car chases 4	0	0	5	4	0.1	1.0
$x^{(5)}$ →	Sword vs karate 5	0	0	5	?	0	0.9

$x_0 = 1$ $\quad x^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$ $\quad n = 2$ excluding $x_0 = 1$

$x_1^{(1)} = \begin{bmatrix} x_1^{(1)} = 1 \\ x_2^{(1)} = 0.9 \\ x_3^{(1)} = 0 \end{bmatrix}$ $\quad n_u = 4, \; n_m = 5$

For each user j, learn a parameter $\theta^{(j)} \in \mathbb{R}^3$. ($\theta^{(j)} \in \mathbb{R}^{n+1}$)

Predict user j as rating movie i with $(\theta^{(j)})^T x^{(i)}$ stars.

$$x^{(3)} = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix} \longleftrightarrow \theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$$

$$(\theta^{(1)})^T x^{(3)} = 5 \times 0.99 = 4.95 \quad \therefore \text{Alice, CPOL} = 4.95$$

We are applying a different copy of this linear Page for each user, and we're saying what it does is A has parameter vector $\theta^{(1)}$ that he uses, that we use to predict his ratings as a fn of how romantic & action packed a movie is.

B, C, D each has diff LR fn of x_1, x_2 thats how we are going to predict their ratings.

Q|| Consider

	A	B	C	D	romance x_1	action x_2
-	5	5	0	0	0.9	0
-	5	2	2	0	1.0	0.4
-	2	4	0	2	0.99	0
-	0	0	5	4	0.1	1.0
-	0	0	5	7	0	0.9

romance by $\theta^{(3)} = 2$

- $\begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$ - $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ - $\begin{bmatrix} 1 \\ 0 \\ 4 \end{bmatrix}$ ✓ $\begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$ → because we want θ That to be small.

☆ Problem formulation.

$r(i,j) = 1$ if user j has rated movie i (0 otherwise)

$y^{(i,j)}$ = rating by user j on movie i (if defined)

$\theta^{(j)}$ = parameter vector for user j.

$x^{(i)}$ = feature vector for movie i.

For user j, movie i, predicted rating: $(\theta^{(j)})^T (x^{(i)})$

$m^{(j)}$ = no of movies rated by user j.

To learn $\theta^{(j)}$:

- Basically a linear regression problem.
- choose $\theta^{(j)}$ as predicted value are close as possible to values that we observed in training sets & the values we observed in our data.

$\theta^{(j)} \in \mathbb{R}^{n+1}$

$$\min_{\theta^{(j)}} \frac{1}{2m^{(j)}} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)} \right)^2 + \frac{\lambda}{2m^{(j)}} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

→ just to make look simpler.

★ optimization objective.

To learn $\theta^{(j)}$ (parameter for user j): (only for specific user)

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

To learn $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$: (For all users)

$$\min_{\theta^{(1)}, \dots \theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

★ Optimization algorithm:

$$\min_{\theta^{(1)}, \dots, \theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u}$$

Gradient descent update.

user

movie

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right) x_k^{(i)} \quad (\text{for } k=0)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right) x_k^{(i)} + \lambda \theta_k^{(j)} \right) \quad (\text{for } k \neq 0)$$

Content based Recommendation.

Someone else giving us all of these features for all of the movie in our data set.

-- @ Collaborative filtering — — — —

property

Feature learning → algo that can learn for itself what features to use

Previously we had assume for each movie someone had come and told us how romantic that movie was and how much action it was.
We had data about how the movie is.
But we want to learn more features than just two

Problem motivation.

$x_0 = 1$

	Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)	x_1	x_2
		$\theta^{(1)}$	$\theta^{(2)}$	$\theta^{(3)}$	$\theta^{(4)}$	romance	action
$x^{(1)}$	LaL	5	5	0	0	?	?
	RF	5	?	?	0	?	?
	CPOL	?	4	0	?	?	?
	NVC	0	0	5	4	?	?
	Sus	0	0	5	?	?	?

Now let make a diff assumption

assumption → We've gone to each of users & each of our users has told us how much they like romantic movie and how much they like action..

she told $\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$, $\theta^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$, $\theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$, $\theta^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$

$\theta^{(1)}$: likes romantic, no like action
$\theta^{(2)}$: like romantic, no like action
$\theta^{(3)}$: like action, no like romantic
$\theta^{(4)}$: ...

Each user tells us value of $\theta^{(j)}$
↳ specifies how much they like diff types of movies.

$\theta^T x = y$

	θ^T	x	$=$	y
previously	$\begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}$ ← to find	given		given
content based	Parabol θ was given, we found whole by $\theta^T x = y$ (LR)	Features were given x_1, x_2		Dataset had ratings.
Train	Given	To find		Given

Now
Call collaborative filtering.

For $x^{(1)}$: $(\theta^{(1)})^T x^{(1)} \approx 5$
$(\theta^{(2)})^T x^{(1)} \approx 5$
$(\theta^{(3)})^T x^{(1)} \approx 0$
$(\theta^{(4)})^T x^{(1)} \approx 0$,

So $x^{(1)}$ might be $\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$

Q1 Consider following movie ratings:

	user1	user2	user3	~~user4~~ (romance)
movie 1	0	1.5	2.5	?

only one feature x_1, suppose $\theta^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\theta^{(2)} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}$, $\theta^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$, $x_1 = ?$

→ 0.5 , $\theta^T x$ should be $\cong y$ → (dataset recordings)

☆ Optimization algorithm.

- Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(i)}$:

$$\min_{x^{(i)}} \frac{1}{2} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

- Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(1)}, \dots, x^{(n_m)}$:

$$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

Q) Suppose you use gradient descent to minimize, then update for $i \neq 0$

$$\rightarrow x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} \left((\theta^{(j)})^T (x^{(i)}) - y^{(i,j)}\right) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

☆ Collaborative filtering

1) content based recommendation: Given $x^{(1)} \dots x^{(n_m)}$ (and movie rating) can estimate $\theta^{(1)}, \dots, \theta^{(n_u)}$

2) Collaborative filtering: Given $\theta^{(1)}, \dots, \theta^{(n_u)}$ can estimate $x^{(1)}, \dots, x^{(n_m)}$

$\theta \xrightarrow{2)} x \xrightarrow{1)} \theta \rightarrow x \rightarrow \theta \rightarrow x \dots$

first Guess randomly.

Every new entry make algo stronger → "collaborative"

-- ② Collaborative filter algorithm. — — — —

Given $x^{(1)}, \dots, x^{(n_m)}$, estimate $\theta^{(1)}, \dots, \theta^{(n_u)}$:

1) $$\min_{\theta^{(1)}, \dots \theta^{(n_u)}} \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

sum over all users — sum over all movies rated by that user — summing over all pairs i,j that corresponds to movie rated by user

Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, estimate $x^{(1)}, \dots, x^{(n_m)}$:

2) $$\min_{x^{(1)}, \dots, x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

for a movie i sum over all users j that have rated movie.

Minimizing $x^{(1)}, \dots x^{(n_m)}$ and $\theta^{(1)}, \dots, \theta^{(n_u)}$ simultaneously:

Simultaneously 1) & 2)

$$J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}) = \frac{1}{2} \sum_{(i,j):r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

$$\min_{\substack{x^{(1)}, \dots, x^{(n_m)} \\ \theta^{(1)}, \dots, \theta^{(n_u)}}} J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)})$$

same as ⇒ ① → ② → ① → ② → ① → ② → ... but cheaper

~~$x_0 = 1$~~ $x \in \mathbb{R}^n$ ~~θ_0~~ $\theta \in \mathbb{R}^n$ ~~$x \in \mathbb{R}^{n+1}$~~

If algo wants a feature ~~always~~ always = 1 It will choose to learn itself e.g. $x^{(1)} = 1$

This has got some properties like.

- If we hold x's constant && just minimize with respect to thetas → 1) — because either of regularization term is constant
- If we hold theta constant && just minimize J() with respect to x's → 2)

Collaborative filtering algo.

1) Initialize $x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)}$ to small random values.

2) Minimize $J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$ using gradient descent (or an advanced optimization algo).

e.g. for every $j = 1, \ldots, n_u$, $i = 1, \ldots, n_m$:

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j : r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i : r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

3) For a user with parameters θ and a movie with (learned) features x, predict a star rating of $\theta^T x$.

$\rightarrow (\theta^{(j)})^T x^{(i)}$

Q) Why we initialized $x^{(1)}, \ldots x^{(n_m)}$ & $\theta^{(1)}, \ldots \theta^{(n_u)}$ to small random values.

→ This serves a symmetry breaking (similar to random initialization of a neural network's parameters) and ensures the algo learns features $x^{(1)}, \ldots, x^{(n_m)}$ that are different from each other.

Collaborative filtering

-- (8) Vectorization : Low Rank matrix Factorization.

vectorization of collaborative filtering.

applications :-

- Given one product can you find other product that are related to this

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)
LoL	5	5	0	0
RF	5	?	?	0
CPOL	?	4	0	?
NCC	0	0	5	4
Snsk	0	0	5	?

$n_m = 5$

$n_u = 4$

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix}$$

$y^{(i,j)}$

* Predicted ratings :

$$\begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \cdots & (\theta^{(n_u)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \cdots & (\theta^{(n_u)})^T(x^{(2)}) \\ \vdots & \vdots & & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \cdots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix}$$

user2, movie1

movie1

user1, movie last

$(i,j) \rightarrow (\theta^{(j)})^T(x^{(i)})$

(matrix way)

$X\Theta^T$

(vectorized way)

Vectorization

$$X = \begin{bmatrix} -(x^{(1)})^T- \\ -(x^{(2)})^T- \\ \vdots \\ -(x^{(n_m)})^T- \end{bmatrix} \qquad \Theta = \begin{bmatrix} —(\theta^{(1)})^T— \\ —(\theta^{(2)})^T— \\ \vdots \\ —(\theta^{(n_u)})^T— \end{bmatrix}$$

- $X\Theta^T$ has property name low rank matrix
- So algo is called ——— factorization.

Collaborative filtering algo here, you can use the learned features in order to find related movies.

★ Finding related movies.

- For each product i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$,
  x_1 = romance, x_2 = action, x_3 = comedy, x_4 = ---
  It can hard to figure out what these features are, But it will learn features that are very meaningful for capturing whatever are most important properties of movie.

- How to find movies j related to movie i ?
→ small $\|x^{(i)} - x^{(j)}\|$ ⟶ movie j and i are "similar".

5 most similar movies to movie i :
Find the 5 movies j with the smallest $\|x^{(i)} - x^{(j)}\|$

--@ Implementational Detail: Mean normalization. — —

★ Users who have not rated any movie.

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)	Eve(5)
LoL	5	5	0	0	?
RF	5	?	?	0	?
CPOL	?	4	0	?	?
NCC	0	0	5	4	?
Svk	0	0	5	?	?

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix}$$

$$\min_{x^{(1)},\dots,x^{(n_m)},\ \theta^{(1)},\dots,\theta^{(n_u)}} \frac{1}{2}\sum_{(i,j):r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n} (x_k^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n} (\theta_k^{(j)})^2$$

This term doesn't matter no $r(i,j)=1$

only this term matter in estimation $\theta^{(5)}$

$n=2$, (∵ We want to learn two features)

$$\frac{\lambda}{2}\left[(\theta_1^{(5)})^2 + (\theta_2^{(5)})^2\right]$$

minimizing this gives

$$\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Rating given by Eve $\Rightarrow (\theta^{(5)})^T x = 0$ --- (So Eve will rate all movies with zero stars)

→ This doesn't feel useful., We cannot recommand her any movie.

Mean normalization will fix this problem.

☆ Mean normalization. (Mean normalization as a sort of preprocessing step for collaborative filtering)

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & ? & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \qquad \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix} \longrightarrow Y = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & -1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

compute average rating that each movie obtained.

$\mu_i = \frac{\text{sum of observation}}{\text{no of observation}}$ (i.e. who has rated the movie)

Subtract off mean rating. Normalizing each movie to have average rating of zero.

(? stays ?)

Now each movie has average rating of 0.

Use this as dataset. learn $\theta^{(j)}, x^{(i)}$

For user j on movie i predict:

$\longrightarrow (\theta^{(j)})^T (x^{(i)}) + \mu_i$

User 5 (Eve):

$\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\underbrace{(\theta^{(5)})^T (x^{(i)})}_{=0} + \mu_i$

So, we are predicting rating as μ_i for user Eve. movie1 = 2.5, 2→2.5, 3→2, 4→2.25

(This makes sense)

If we have movie with no rating.---

analogous to

~~movie with no~~
user who hasn't rated anything.

Normalize different columns (instead of normalizing rows)

Maybe shouldn't recommend that movie to anyone, anyway.
If you have a movie with no rating.

We talked about mean normalization. However, unlike some other applications of feature scaling, we did not scale the movie ratings by dividing by the range (max-min) value. This is because..-

→ ✓ All movie ratings are already comparable (0-5 stars), so they are already on similar scale.