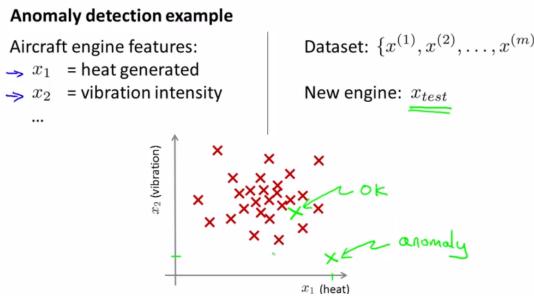
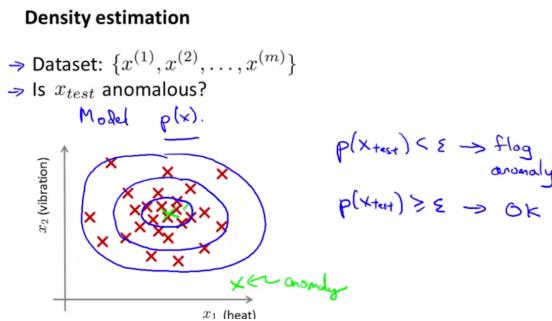


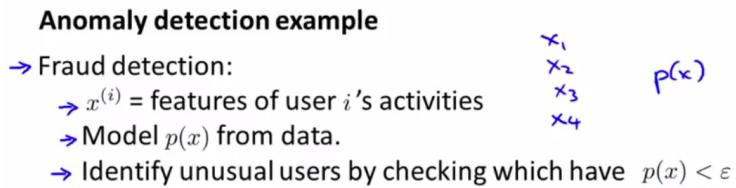
- Density Estimation
- Problem Motivation
- Anomaly detection example



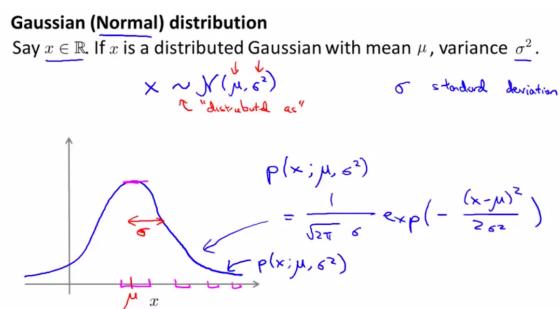
- Density estimation



- Anomaly detection example

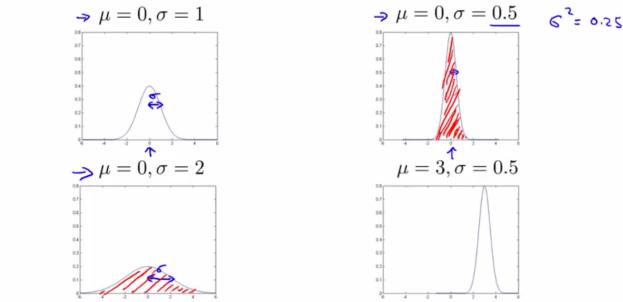


- Manufacturing
- Monitoring computers in a data center
 - compute the features of machines
- Gaussian distribution or normal distribution



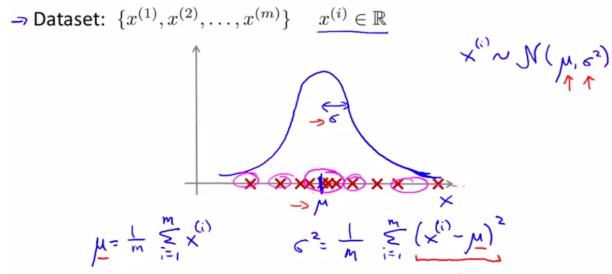
- Gaussian distribution example

Gaussian distribution example



- Parameter estimation

Parameter estimation



- Algorithm

- Density estimation

Density estimation

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

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

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

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

$\sum_{i=1}^n i = 1+2+3+\dots+n$
 $\prod_{i=1}^n i = 1 \times 2 \times 3 \times \dots \times n$

- Anomaly detection algorithm

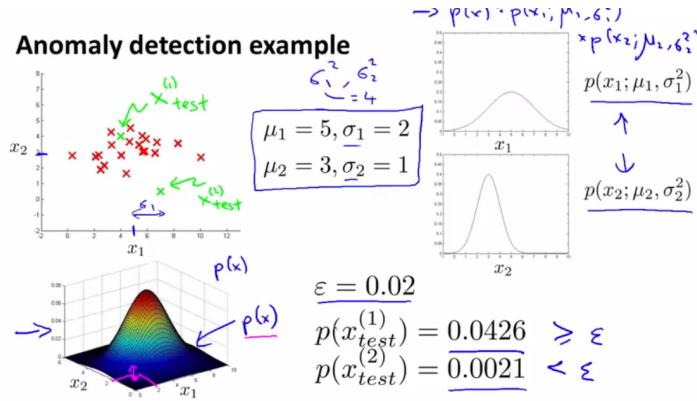
Anomaly detection algorithm

- Choose features x_i that you think might be indicative of anomalous examples. $\{x^{(1)}, \dots, x^{(n)}\}$
 - Fit parameters $\mu_1, \dots, \mu_n, \sigma_1^2, \dots, \sigma_n^2$

$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$$

$$\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2$$
 - 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 $p(x) < \epsilon$

- Anomaly detection example



- Developing and Evaluating an Anomaly Detection System

- 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 learning algorithm
 - Assume we have some labeled data, of anomalous and non anomalous examples. ($y=0$ if normal, $y=1$ if anomalous)
 - Training set: (assume normal examples/not anomalous)
 - Cross validation set:

\rightarrow Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (assume normal examples/not anomalous)
 \rightarrow Cross validation set: $(x_{cv}^{(1)}, y_{cv}^{(1)}), \dots, (x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$
 \rightarrow Test set: $(x_{test}^{(1)}, y_{test}^{(1)}), \dots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$
 $y=1$

- Aircraft engines motivating example

Aircraft engines motivating example

\rightarrow 10000 good (normal) engines
 \rightarrow 20 flawed engines (anomalous) $y=1$
 \rightarrow Training set: 6000 good engines ($y=0$) $p(x) = p(x_1; \mu_1, \sigma_1^2) \dots p(x_n; \mu_n, \sigma_n^2)$
 CV: 2000 good engines ($y=0$), 10 anomalous ($y=1$)
 Test: 2000 good engines ($y=0$), 10 anomalous ($y=1$)

- Algorithm evaluation

Algorithm evaluation

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

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

Possible evaluation metrics:

- True positive, false positive, false negative, true negative
- Precision/Recall
- F₁-score

Can also use cross validation set to choose parameter ε

- Anomaly Detection vs. Supervised Learning

Anomaly detection	vs.	Supervised learning
<ul style="list-style-type: none"> → Very small number of positive examples ($y = 1$). (<u>0-20</u> is common). → Large number of negative ($y = 0$) examples. (<u>$p(x)$</u>) → Many different "types" of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like; → future anomalies may look nothing like any of the anomalous examples we've seen so far. 		<p>Large number of positive and negative examples.</p> <p>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.</p>

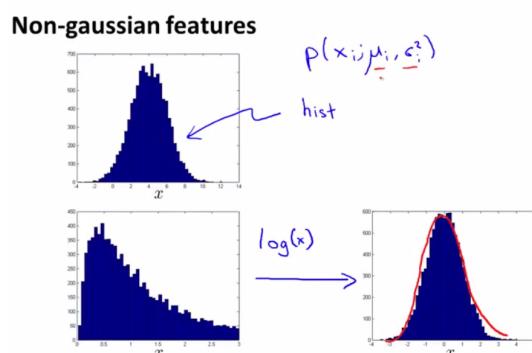
- Anomaly Detection vs. Supervised Learning

Anomaly detection	vs.	Supervised learning
<ul style="list-style-type: none"> → <u>Fraud detection</u> (<u>$y=1$</u>) • Manufacturing (e.g. aircraft engines) • Monitoring machines in a data center <p style="text-align: center;">⋮</p>		<ul style="list-style-type: none"> • Email spam classification • Weather prediction (sunny/rainy/etc). • Cancer classification <p style="text-align: center;">⋮</p>

- Choosing What features to Use

- Non-gaussian features

- if data is not gaussian take transformations of the data to obtain a gaussian distribution



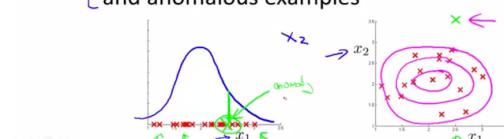
- Error analysis for anomaly detection

- Error analysis for anomaly detection

Want $p(x)$ large for normal examples x .
 $p(x)$ small for anomalous examples x .

Most common problem:

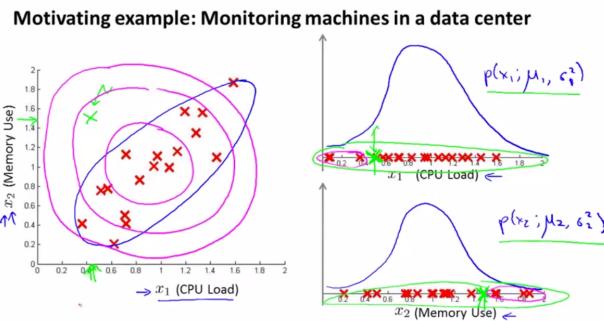
$p(x)$ is comparable (say, both large) for normal and anomalous examples



- Monitoring computers in a datacenter
 - Choose features that might take on unusually large or small values in the event of an anomaly.
 - memory use of computer
 - number of disk accesses/sec
 - cpu load
 - network traffic
 - creating new features
 - cpu load/ network traffic
 - this type of feature would help to detect anomalies

Multivariate Gaussian Distribution

- Motivating example: Monitoring machines in a data center



- thinks everything on the pink lines have the same probability which is wrong
- Multivariate Gaussian (normal) distribution

Multivariate Gaussian (Normal) distribution

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

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

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

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

| Σ | = determinant of Σ | det(Sigma)

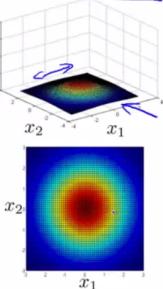
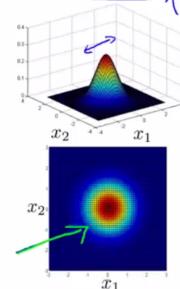
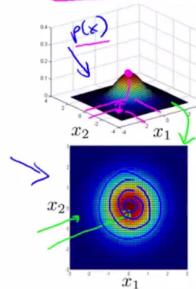
- Multivariate Gaussian (Normal) examples

Multivariate Gaussian (Normal) examples

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

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

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



Andrew 1

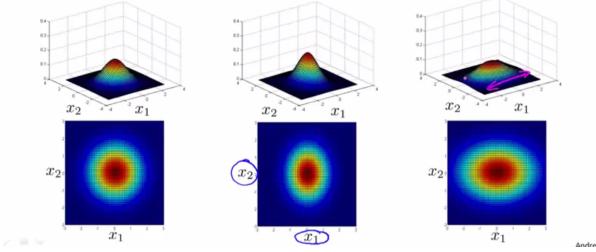
- Multivariate Gaussian (Normal) examples

Multivariate Gaussian (Normal) examples

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

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

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 0.2 & 0 \\ 0 & 1 \end{bmatrix}$$

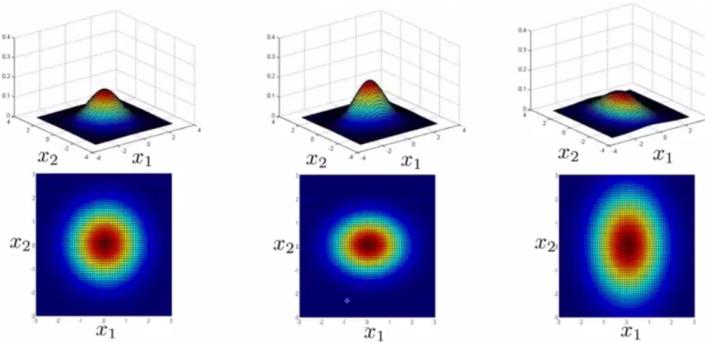


Multivariate Gaussian (Normal) examples

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

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

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



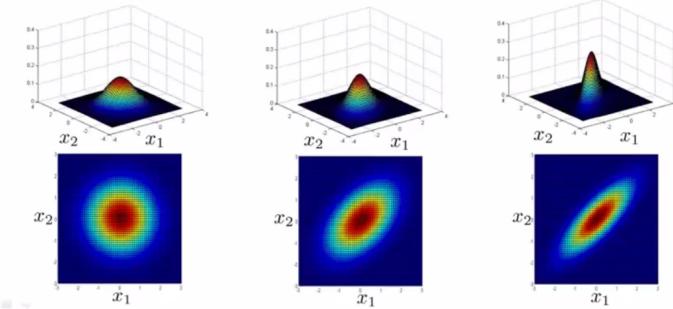
Andrew N

Multivariate Gaussian (Normal) examples

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

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

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



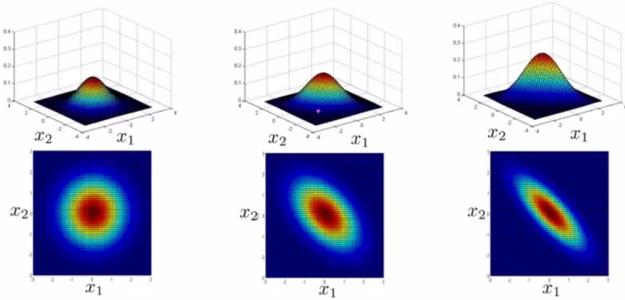
Andrew N

Multivariate Gaussian (Normal) examples

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

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

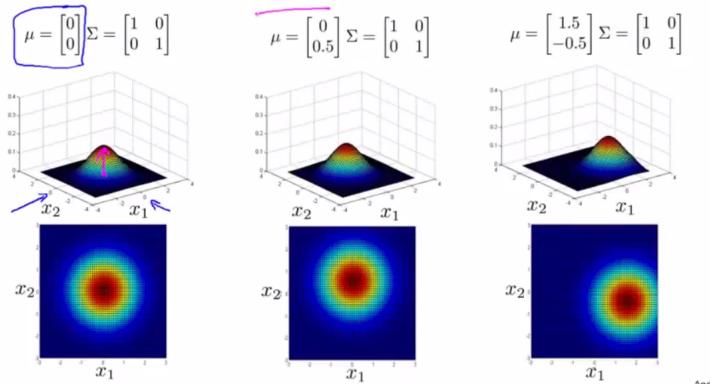
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$$



Andre

- captures the negative correlation between x_1 and x_2
- Multivariate Gaussian (Normal) examples

Multivariate Gaussian (Normal) examples

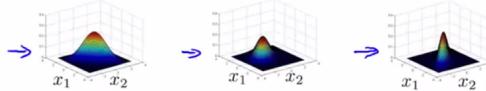


- Anomaly Detection using the Multivariate Gaussian Distribution
- Multivariate Gaussian (Normal) distribution

Multivariate Gaussian (Normal) distribution

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

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



Parameter fitting:

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

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

- Anomaly detection with the multivariate Gaussian

Anomaly detection with the multivariate Gaussian

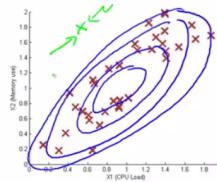
1. Fit model $p(x)$ by setting

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

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 an anomaly if $p(x) < \varepsilon$

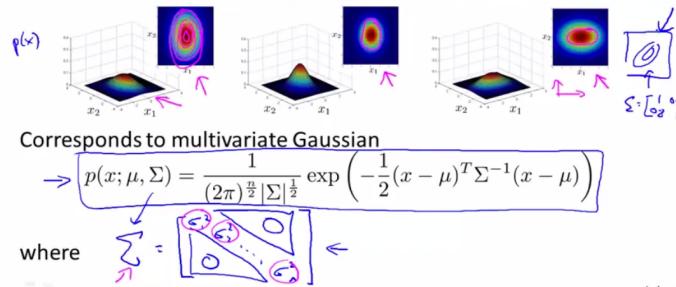


- Relationship to original model

- the same as a multivariate Gaussian with the constraint that sigma on the diagonal and every element in the matrix has to be zero (the covariance matrix)
- the model has to be axis aligned

Relationship to original model

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



Andrew I

- Original model vs. Multivariate Gaussian model

→ Original model	vs.	→ Multivariate Gaussian
$p(x_1; \mu_1, \sigma_1^2) \times \dots \times p(x_n; \mu_n, \sigma_n^2)$		$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} \Sigma ^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$
Manually create features to capture anomalies where x_1, x_2 take unusual combinations of values. $\rightarrow X_3 = \frac{x_1}{x_2} = \frac{\text{CPU load}}{\text{memory}}$		Automatically captures correlations between features $\Sigma \in \mathbb{R}^{n \times n}$ Σ^{-1}
→ Computationally cheaper (alternatively, scales better to large n) $n=10,000, m=100,000$		Computationally more expensive
OK even if m (training set size) is small		Must have $m > n$, or else Σ is non-invertible.

Andrew N

- Predicting Movie Ratings
- Problem Formulation
- Prediction Movie Ratings

Example: Predicting movie ratings

>User rates movies using one to five stars

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	
Love at last	5	5	0	0	→
Romance forever	5	?	0	0	→
Cute puppies of love	?	4	0	?	→
Nonstop car chases	0	0	5	4	→
Swords vs. karate	0	0	5	?	→

$n_u = 4$ $n_m = 5$

$n_u = \text{no. users}$
 $n_m = \text{no. movies}$
 $r(i, j) = 1$ if user j has rated movie i
 $y^{(i,j)} = \text{rating given by user } j \text{ to movie } i$ (defined only if $r(i, j) = 1$)
 $0, \dots, 5$

- given these movie ratings what should the values of ? be

- Content Based Recommendations
- Content based recommender systems
 - each movie has a set of features

Movie	Content-based recommender systems				$n_u = 4, n_m = 5$	$x^{(i)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$
	Alice (1)	Bob (2)	Carol (3)	Dave (4)		
Love at last	5	5	0	0	$\theta^{(1)} = \begin{bmatrix} 0.9 \\ 0 \\ 0 \end{bmatrix}$	$x_1 \rightarrow 0.9, x_2 \rightarrow 0$
Romance forever	5	?	?	0	$\theta^{(2)} = \begin{bmatrix} 1.0 \\ 0.01 \\ 0 \end{bmatrix}$	$x_1 \rightarrow 1.0, x_2 \rightarrow 0.01$
Cute puppies of love	?	4	0	?	$\theta^{(3)} = \begin{bmatrix} 0.99 \\ 0 \\ 0 \end{bmatrix}$	$x_1 \rightarrow 0.99, x_2 \rightarrow 0$
Nonstop car chases	0	0	5	4	$\theta^{(4)} = \begin{bmatrix} 0.1 \\ 1.0 \\ 0 \end{bmatrix}$	$x_1 \rightarrow 0.1, x_2 \rightarrow 1.0$
Swords vs. karate	0	0	5	?	$\theta^{(5)} = \begin{bmatrix} 0 \\ 0.9 \\ 0 \end{bmatrix}$	$x_1 \rightarrow 0, x_2 \rightarrow 0.9$

For each user j , learn a parameter $\theta^{(j)} \in \mathbb{R}^3$. Predict user j as rating movie i with $(\theta^{(j)})^T x^{(i)}$ stars.

$$\theta^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix} \Leftrightarrow \theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad (\theta^{(1)})^T x^{(3)} = 5 \times 0.99 = 4.95$$

- Problem formulation

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: $\underline{(\theta^{(j)})^T(x^{(i)})}$
- $m^{(j)}$ = no. of movies rated by user j
To learn $\underline{\theta^{(j)}}$:

- To learn the parameter vector j $\theta^{(j)}$

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

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

- Optimization objective

Optimization objective:

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

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

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

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

- add the additional summation to learn all the users
- Optimization algorithm
 - Optimization objective j or the cost function
 - Gradient descent update:

Optimization algorithm:

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

\downarrow

$J(\theta^{(1)}, \dots, \theta^{(n_u)})$

Gradient descent update:

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

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

$\frac{\partial}{\partial \theta_k^{(j)}} J(\theta^{(1)}, \dots, \theta^{(n_u)})$

Andrew T

- Collaborative Filtering

- Problem motivation

- have a dataset where we do not know the values of the features for the movies
- theta vector for each user the first entry is for the bias $x_0 = 1$. The second value of the vector is for the romance movie and the third value of the vector is for the action movie.
- given these values it is possible to infer the values of the features for each movie

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)	$x_0 = 1$
Love at last	5	5	0	0	1.0	0.0	
Romance forever	5	?	?	0	?	?	$x^{(1)} = \begin{bmatrix} 1 \\ 1.0 \\ 0 \end{bmatrix}$
Cute puppies of love	?	4	0	?	?	?	
Nonstop car chases	0	0	5	4	?	?	
Swords vs. karate	0	0	5	?	?	?	

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

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

- Optimization algorithm

- double summation to learn all the features for all the movies

Optimization algorithm

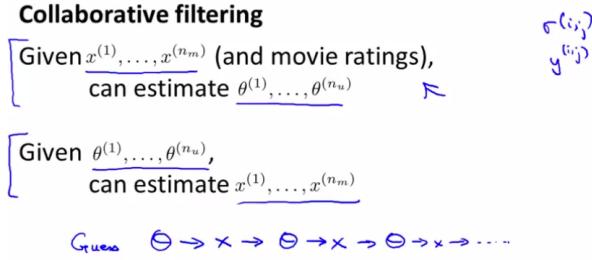
Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(i)}$:

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

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

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

- Collaborative filtering



- **Collaborative Filtering System**
- Collaborative filtering optimization objective

Collaborative filtering optimization objective $(i, j) : r(i, j) \in \mathbb{R}$

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

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

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

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

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

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

Andrew Ng

- Collaborative filtering algorithm

Collaborative filtering algorithm

$\cancel{x \in \mathbb{C}}$ $\cancel{x \in \mathbb{R}^n}, \theta \in \mathbb{R}$

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

2. Minimize $J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)})$ using gradient descent (or an advanced optimization algorithm). E.g. for every $j = 1, \dots, n_u, i = 1, \dots, n_m$:

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

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

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

$$(\theta^{(1)})^T (x^{(1)})$$

- **Vectorization: Low Rank Matrix Factorization**
- Collaborative Filtering

Collaborative filtering

$X \Theta^T \leftarrow (\Theta^{(1)})^T (x^{(1)})$

Predicted ratings: $(\Theta^{(1)})^T (x^{(1)})$

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

$\underbrace{(\theta^{(1)})^T (x^{(1)})}_{(\theta^{(1)})^T (x^{(2)})} \quad \underbrace{(\theta^{(2)})^T (x^{(1)})}_{(\theta^{(2)})^T (x^{(2)})} \quad \dots \quad (\theta^{(n_u)})^T (x^{(1)})$

$\underbrace{(\theta^{(1)})^T (x^{(n_m)})}_{(\theta^{(1)})^T (x^{(n_m)})} \quad (\theta^{(2)})^T (x^{(n_m)}) \quad \dots \quad (\theta^{(n_u)})^T (x^{(n_m)})$

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

Low rank matrix factorization

Andrew Ng

- Finding related movies
 - for each product i , we learn a feature vector $x(i)$ element R^n
 - how to find movies j related to movie i ?

How to find $\underset{\uparrow}{\text{movies } j}$ related to $\underset{\wedge}{\text{movie } i}$?

small $\|x^{(i)} - x^{(j)}\| \rightarrow$ 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 movies

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	Eve (5)
Love at last	5	5	0	0	?
Romance forever	5	?	?	0	?
Cute puppies of love	?	4	0	?	?
Nonstop car chases	0	0	5	4	?
Swords vs. karate	0	0	5	?	?

Andrew Ng

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

$$n=2 \quad \Theta^{(s)} \in \mathbb{R}^2 \quad \Theta^{(s)} = \begin{bmatrix} ? \\ ? \end{bmatrix} \quad \frac{\lambda}{2} [(\Theta_1^{(s)})^2 + (\Theta_2^{(s)})^2] < 0$$

$$(\Theta^{(s)})^T x^{(s)} = 0$$

- Mean Normalization

Mean Normalization:

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

For user j , on movie i predict:

$$\rightarrow (\Theta^{(s)})^T (x^{(s)}) + \mu_i$$

learn $\underline{\Theta^{(s)}}, \underline{x^{(s)}}$

User 5 (Eve):

$$\underline{\Theta^{(s)}} = \begin{bmatrix} ? \\ ? \end{bmatrix}$$

$$\underbrace{(\Theta^{(s)})^T (x^{(s)})}_{\approx 0} + \boxed{\mu_i}$$