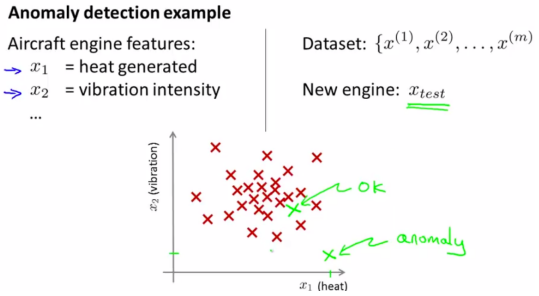
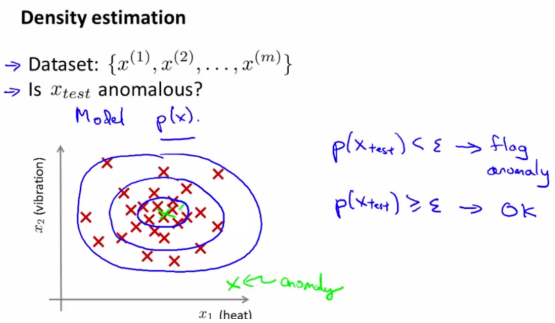
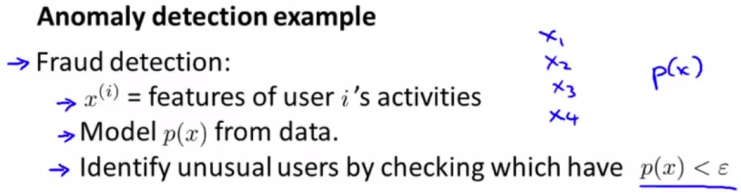
- **Density Estimation**
- **Problem Motivation**
- Anomaly detection example



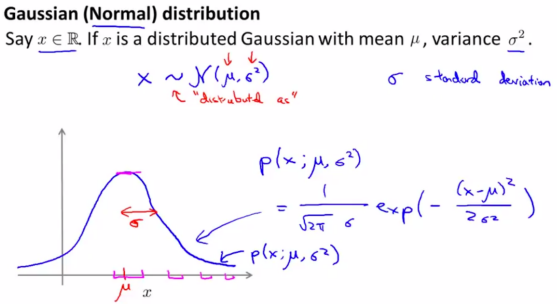
- Density estimation



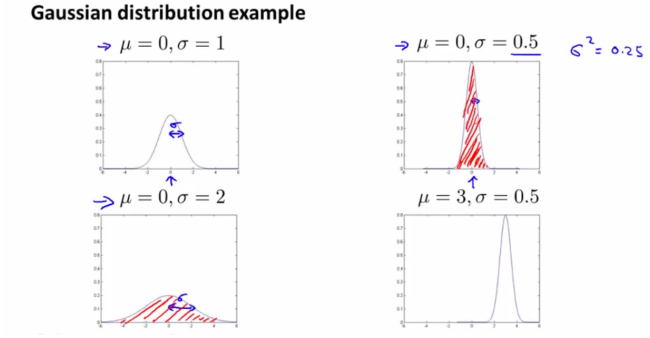
- Anomaly detection example



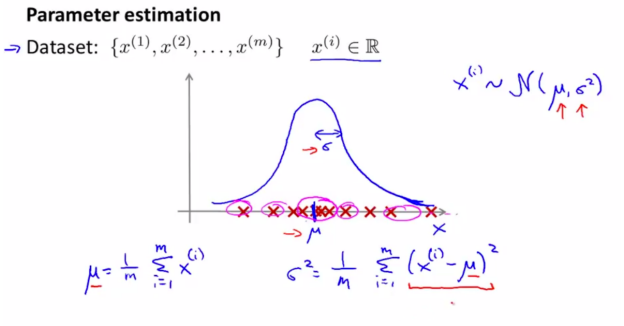
  - Manufacturing
  - Monitoring computers in a data center
    - compute the features of machines
- **Gaussian distribution or normal distribution**



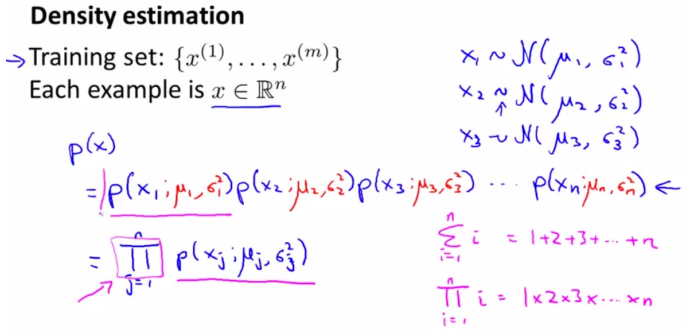
- Gaussian distribution example



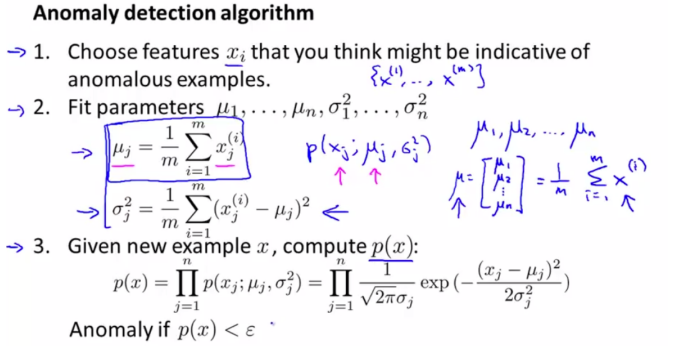
- Parameter estimation



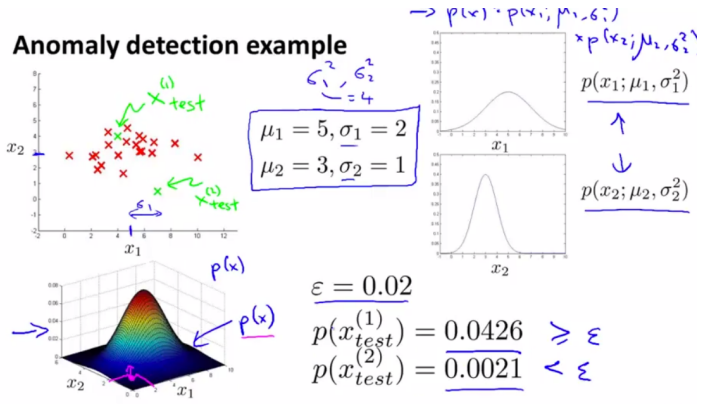
- **Algorithm**
- Density estimation



- Anomaly detection algorithm



- 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 is 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:

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

- Aircraft engines motivating example

Aircraft engines motivating example
→ 10000 good (normal) engines
→ 20 flawed engines (anomalous)
→ Training set: 6000 good engines ($y = 0$)
CV: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)
Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

- Algorithm evaluation

Algorithm evaluation
→ Fit model $p(x)$ on training set $\{x^{(1)}, \ldots, x^{(m)}\}$
→ 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
- F_1-score

Can also use cross validation set to choose parameter ε

- Anomaly Detection vs. Supervised Learning

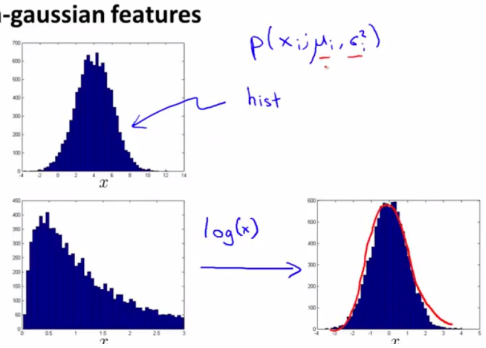
Anomaly detection	vs.	Supervised learning
Very small number of positive examples ($y = 1$). (0-20 is common).		Large number of positive and negative examples.
Large number of negative ($y = 0$) examples. $p(x)$		
Many different "types" of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like; future anomalies may look nothing like any of the anomalous examples we've seen so far.		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.

- Anomaly Detection vs. Supervised Learning

Anomaly detection	vs.	Supervised learning
• Fraud detection $y=1$		• Email spam classification
• Manufacturing (e.g. aircraft engines)		• Weather prediction (sunny/rainy/etc).
• Monitoring machines in a data center		• Cancer classification
⋮		⋮

- **Choosing What features to Use**
- Non-gaussian features
  - if data is not gaussian take transformations of the data to obtain a gaussian distribution

Non-gaussian features

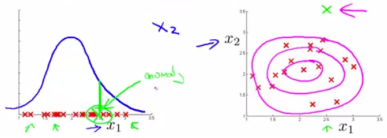


- Error analysis for anomaly detection

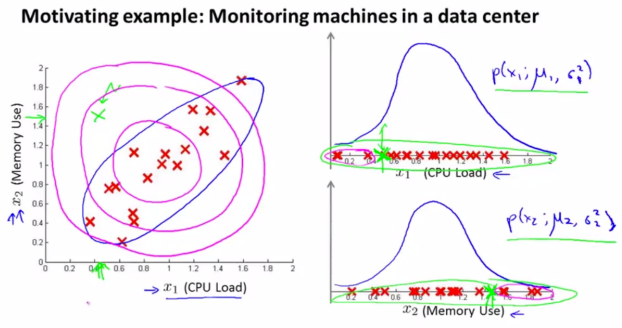
Error analysis for anomaly detection

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

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

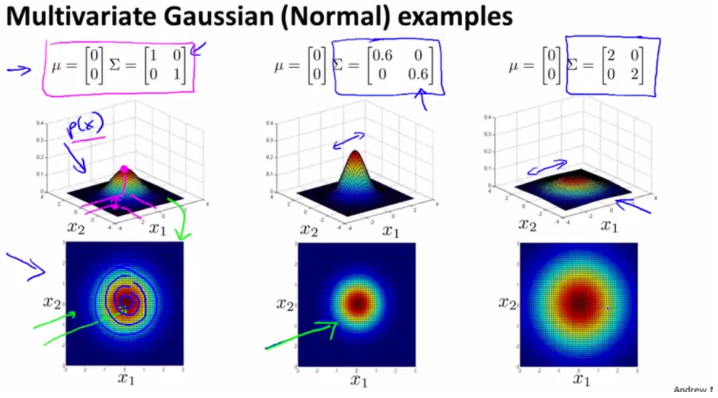
Model $p(x)$ all in one go.

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

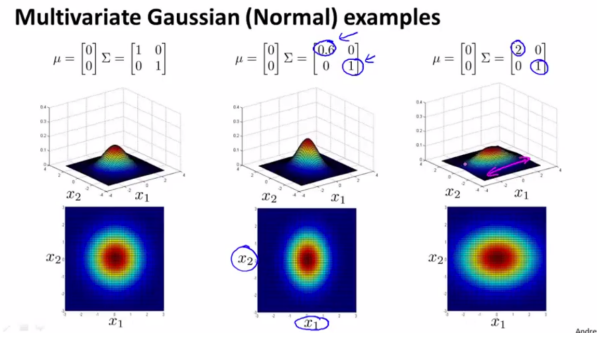
$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

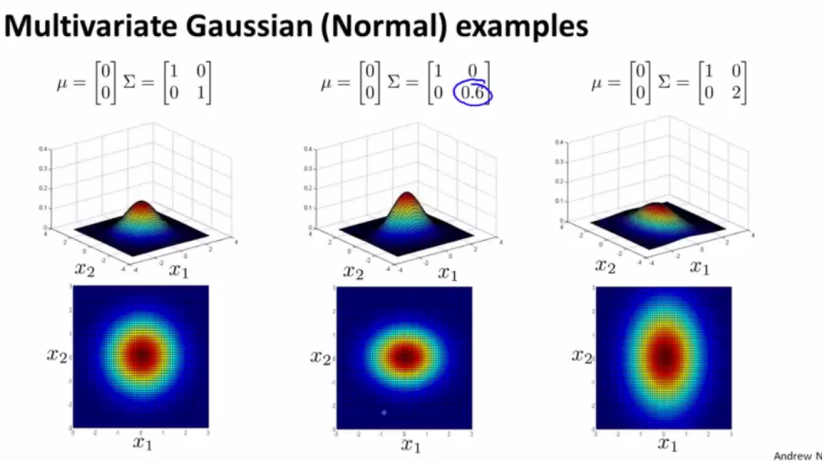
$|\Sigma|$ = determinant of Σ | det (Sigma)

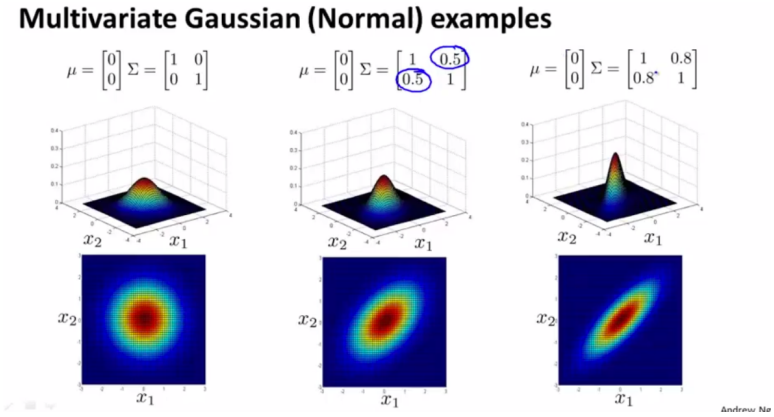
- Multivariate Gaussian (Normal) examples

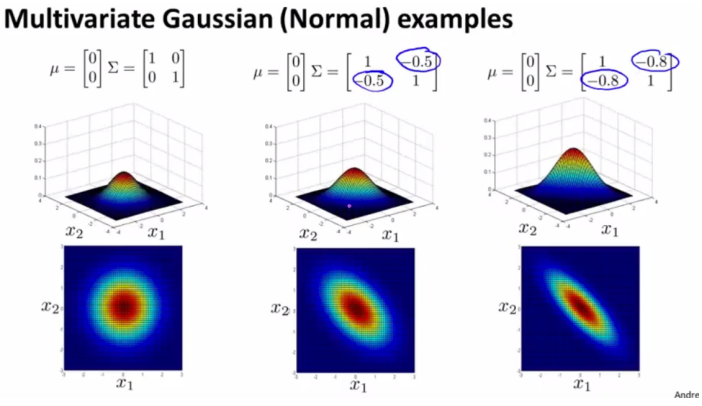


- Multivariate Gaussian (Normal) examples

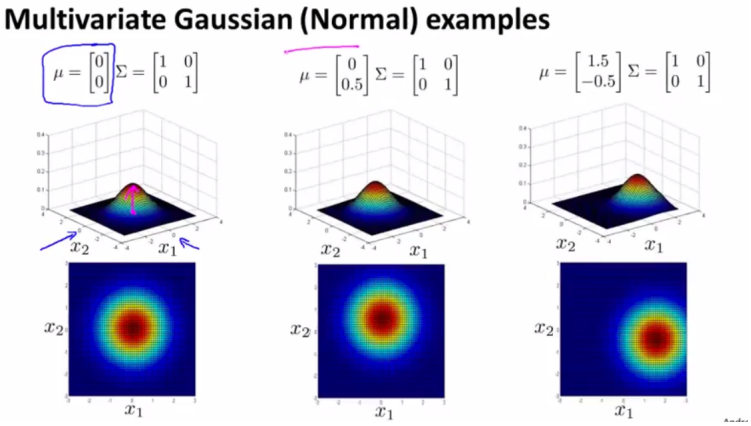




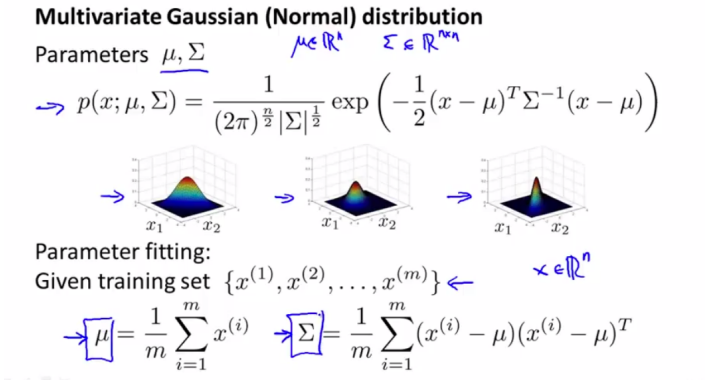




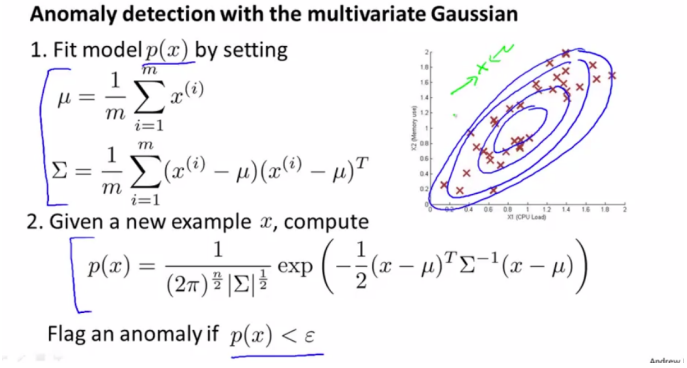
  - captures the negative correlation between x1 and x2
- Multivariate Gaussian (Normal) examples



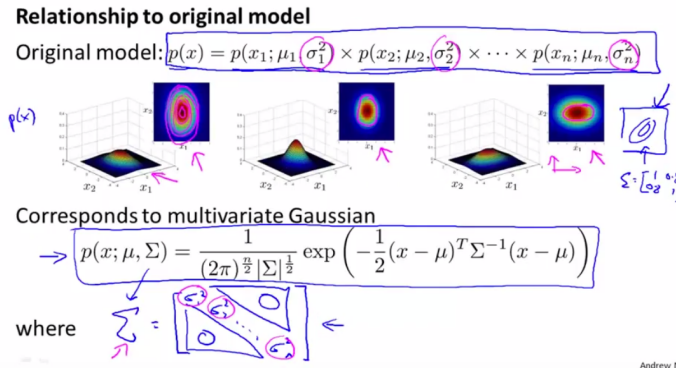
- **Anomaly Detection using the Multivariate Gaussian Distribution**
- Multivariate Gaussian (Normal) distribution



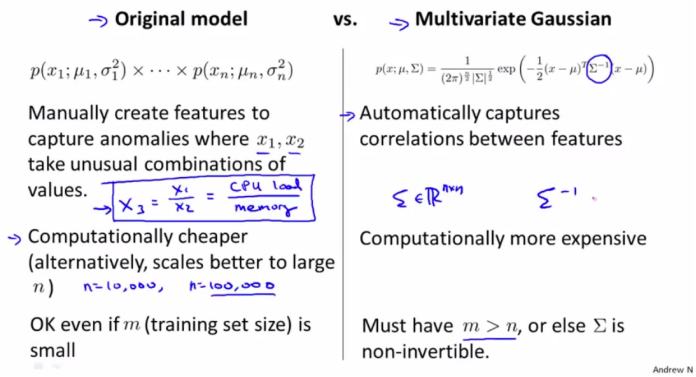
- Anomaly detection with the multivariate Gaussian



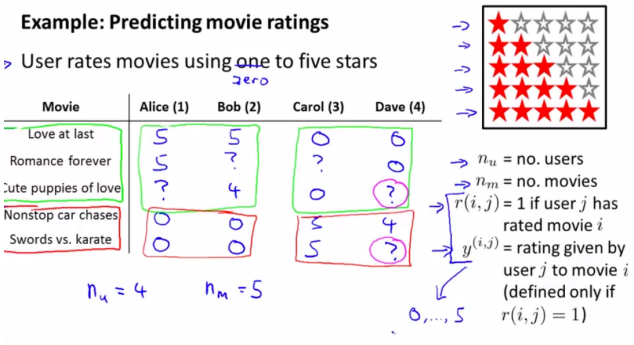
- Relationship to original model
  - the same as a multivariate Gaussian with the constraint that sigma on the diagonal and every element in the matrix has to be zero (the covariance matrix)
  - the model has to be axis aligned



- Original model vs. Multivariate Gaussian model



- **Predicting Movie Ratings**
- **Problem Formulation**
- Prediction Movie Ratings



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

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

Content-based recommender systems

Movie	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 forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

$n_u = 4$, $n_m = 5$, $x_0 = 1$, $n = 2$, $x^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$

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

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

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

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

  - To learn the parameter vector j theta(j)

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

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

- Optimization objective

Optimization objective:

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

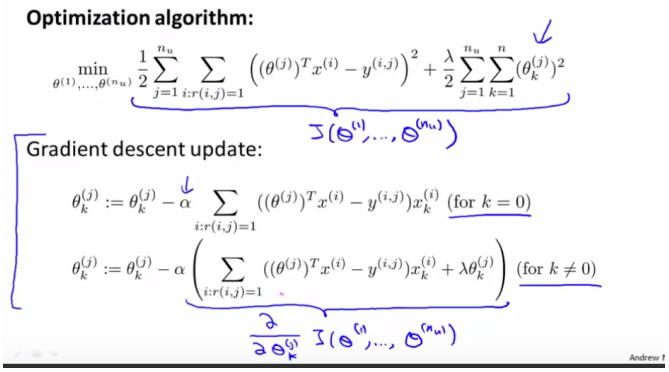
$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

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

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

- add the additional summation to learn all the users

- Optimization algorithm
  - Optimization objective j or the cost function
  - Gradient descent update:



- **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 x0 = 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) $\theta^{(1)}$	Bob (2) $\theta^{(2)}$	Carol (3) $\theta^{(3)}$	Dave (4) $\theta^{(4)}$	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	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	?	?	?

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

$x^{(1)} = \begin{bmatrix}1\\1.0\\0.0\end{bmatrix}$

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

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

Optimization algorithm

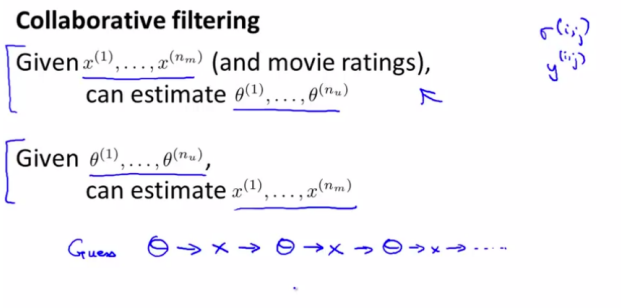
Given $\theta^{(1)},\dots,\theta^{(n_u)}$, to learn $x^{(i)}$:

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

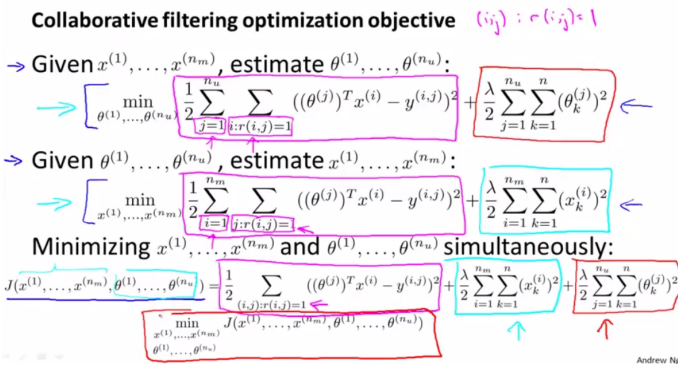
Given $\theta^{(1)},\dots,\theta^{(n_u)}$, to learn $x^{(1)},\dots,x^{(n_m)}$:

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

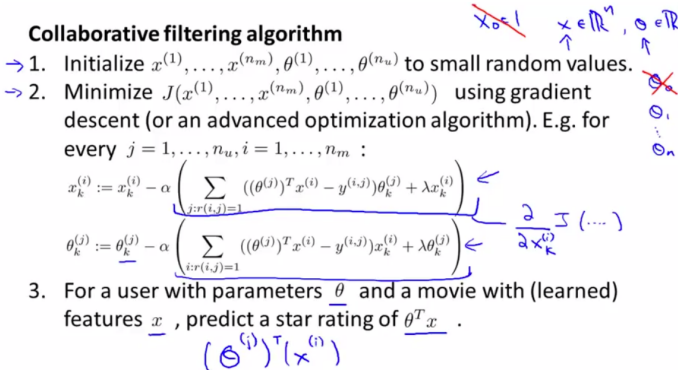
- Collaborative filtering



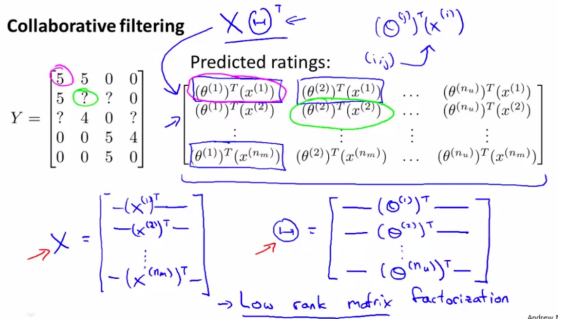
- **Collaborative Filtering System**
- Collaborative filtering optimization objective



- Collaborative filtering algorithm



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



- 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 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 movies

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	?	?

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

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

$n = 2$ $\quad \theta^{(5)} \in \mathbb{R}^2$ $\quad \theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\quad \frac{\lambda}{2}\left[(\theta_1^{(5)})^2 + (\theta_2^{(5)})^2\right]$

$(\theta^{(5)})^T x^{(i)} = 0$

Andrew Ng

- Mean Normalization

Mean Normalization:

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

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

For user j, on movie i predict:

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

User 5 (Eve):

$\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\quad (\theta^{(5)})^T (x^{(i)}) + \mu_i$ $\quad \approx 0$

-