Machine Learning
Week 9
Section 1
Anomaly Detection

This week will cover anomaly detection and recommender systems. Anomaly detection usually takes a large number of data points and figures out which vary significantly from the average. Recommender systems look at patterns of activities and from different sources and products and produce recommendations. Some recommender system algorithms include the collaborative filtering algorithm and low-rank matrix factorization.

Density Estimation

Problem Motivation

Anomaly Detection is a common ML problem type. It is an unsupervised problem, but there are some aspects of it that are similar to supervised learning problems as well.

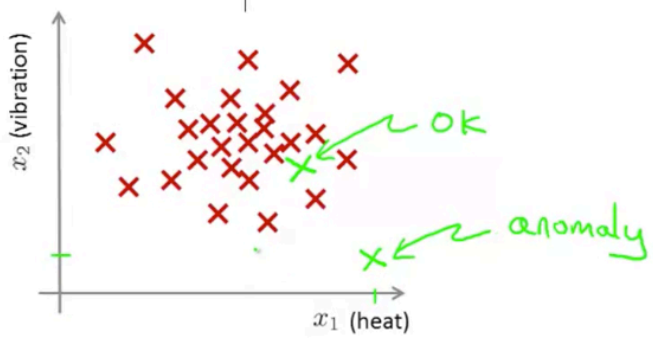
Ex. Anomaly Detection in Aircraft Engine Manufacturing.

Anomaly detection example

Aircraft engine features:
- x_1 = heat generated
- x_2 = vibration intensity
- ...

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

New engine: x_{test}



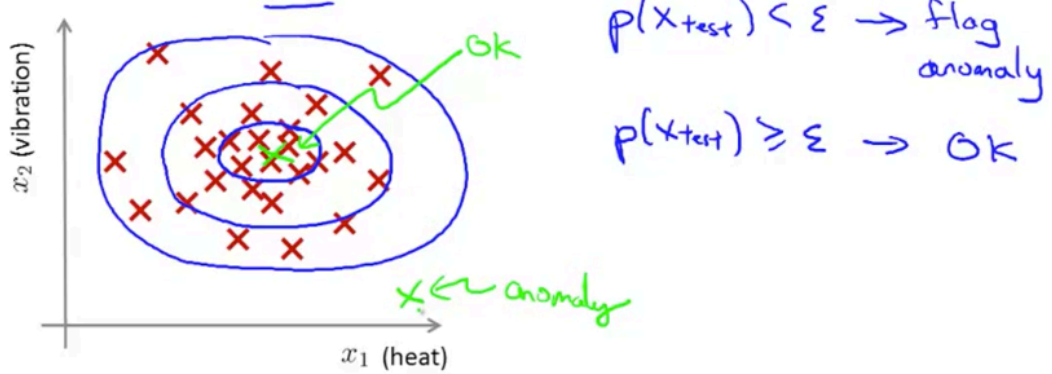
More formally, we are given a dataset and assume these examples are normal / non-anomalous.

Given this unlabeled training set, we need to create a Model p(x) to predict if new cases are normal or anomalous.

Density estimation

- Dataset: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$
- Is x_{test} anomalous?

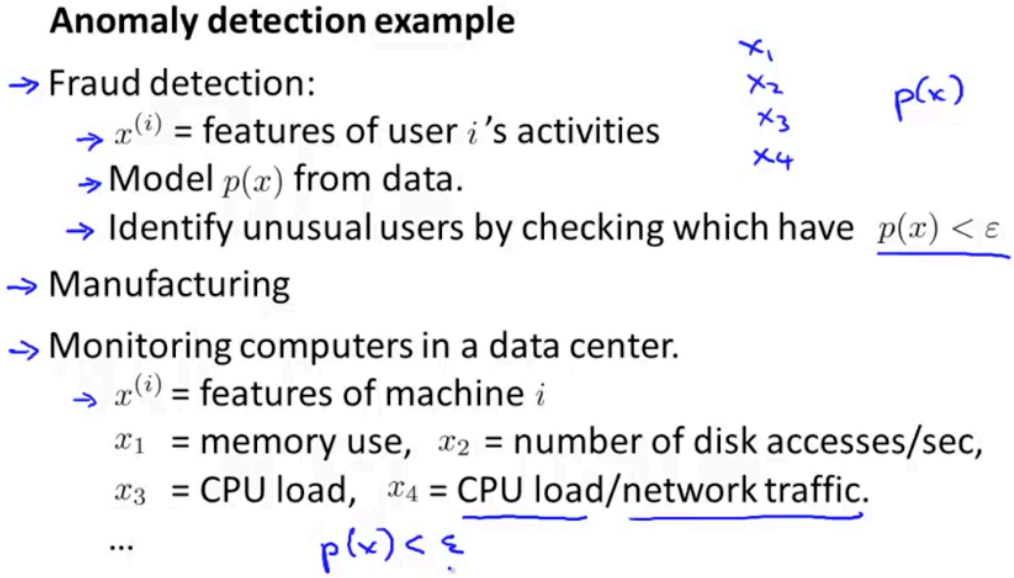
Model $p(x)$.



Examples of Anomaly Detection Applications:

Here is an anomaly detection example of fraud detection.

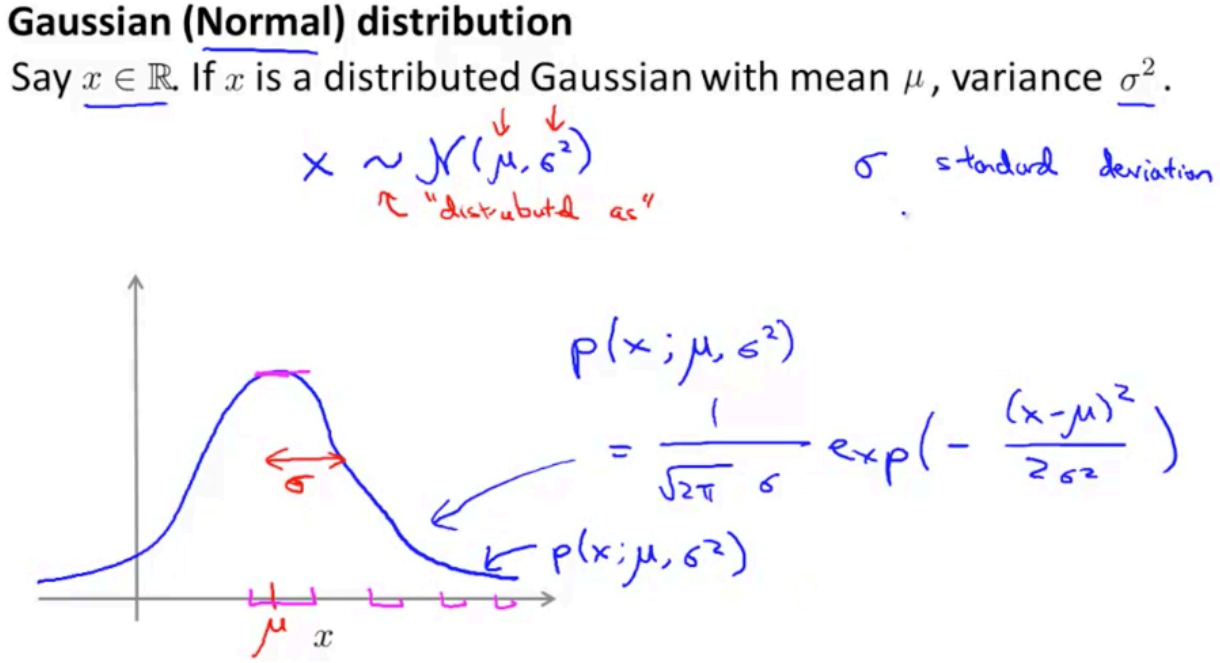
This type of model may flag users who are acting unusually, not just fraudulently.



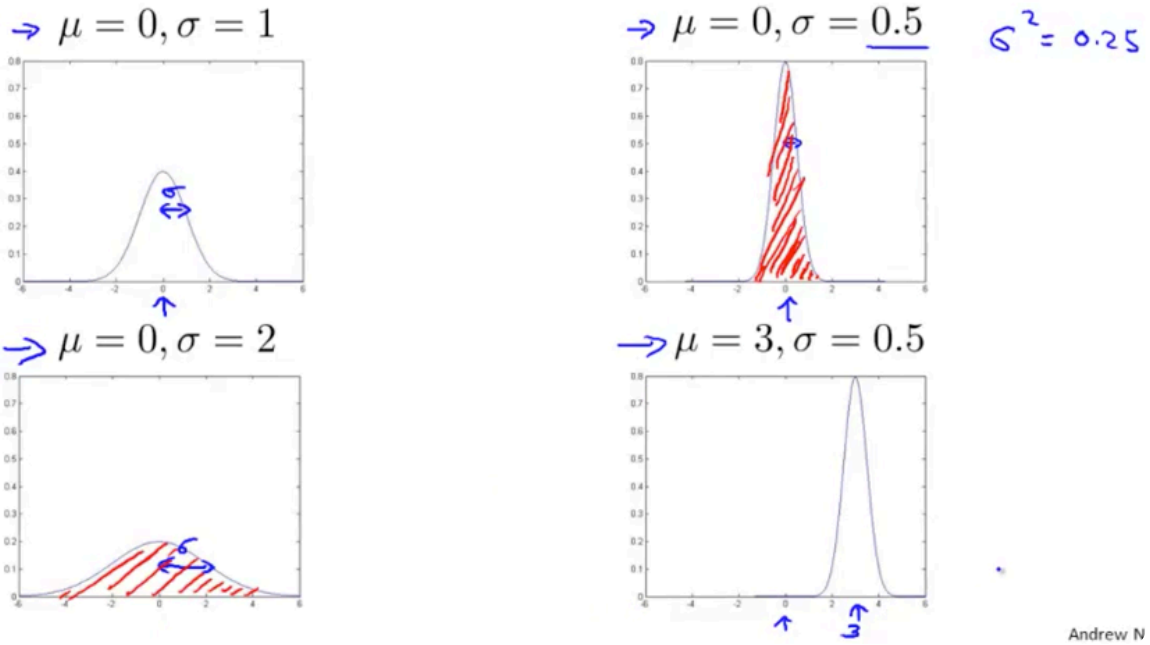
Similar to the classification problems we saw before, if there are too many false positives or true negatives being predicted by the model, you can try increasing or decreasing epsilon to tune the sensitivity.

Gaussian Distribution

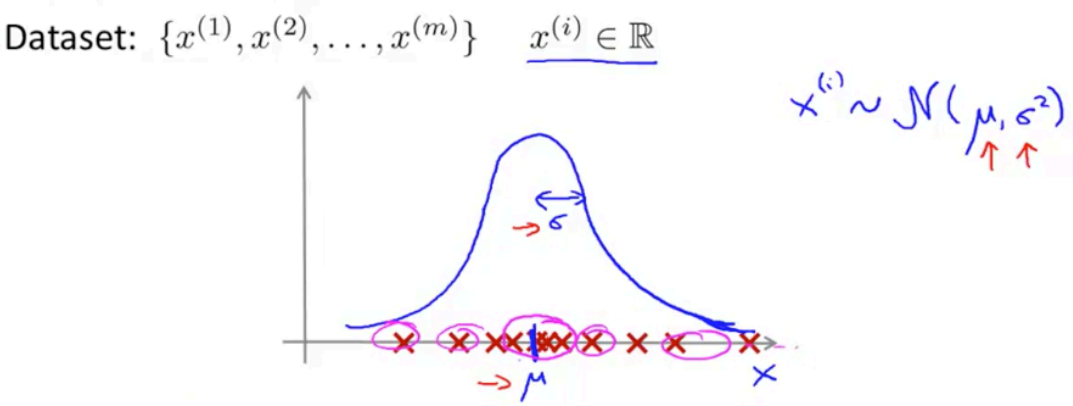
In this video we will discuss the Gaussian distribution / Normal distribution.



Some examples of the gaussian for different values of the mean (mu) and std (sigma)

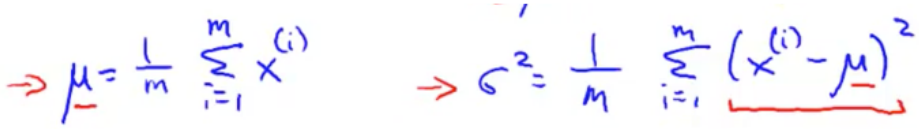


Next lets talk about the parameter estimate problem:



If we suspect that our examples come from a Gaussian distribution w/ unknown mu and std. How do we estimate what those parameters could be?

The standard formulas for these estimates are as follows:



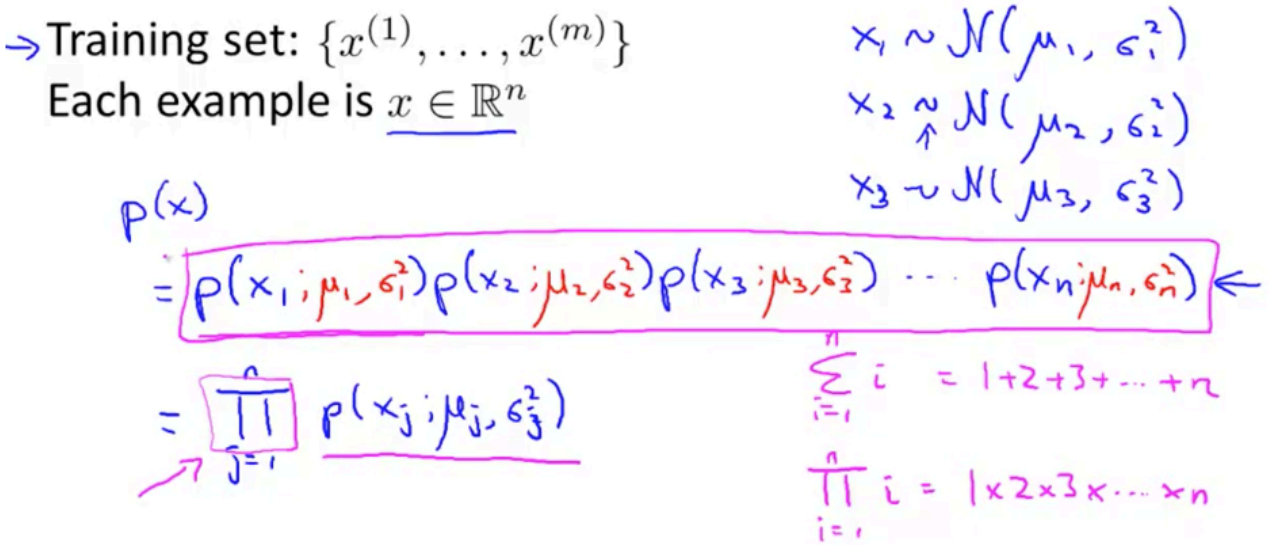
These are the 'maximum likelihood estimates' from statistics.

**Note in some cases for the variances formula, people use 1 / (m-1) instead of 1 / m, but typically for ML it is 1 / m
The two versions have slightly different mathematical properties but for our cases 1 / m is fine.

Anomaly Detection Algorithm

In this video we will apply the Gaussian distribution to develop an anomaly detection algorithm.

The problem of density estimation, we need to compute the gaussian for each feature on each training example.



In statistics this equation corresponds to an independent assumption on the values of the features x1 through xn.

Putting everything together we have the following algorithm:

Anomaly detection algorithm

1. Choose features x_i that you think might be indicative of anomalous examples.
2. Fit parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$

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

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

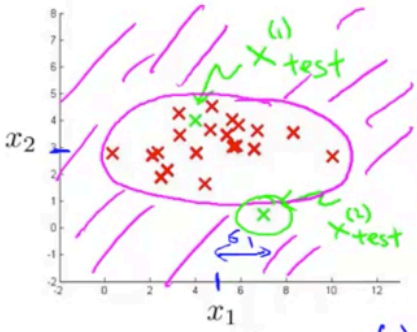
3. Given new example x, compute $p(x)$:

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

Anomaly if $p(x) < \varepsilon$

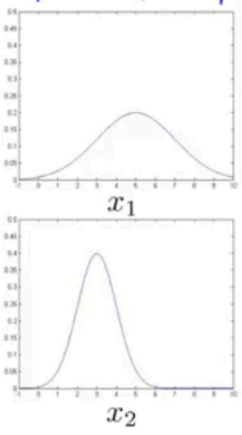
Example of an application of this method:

Anomaly detection example



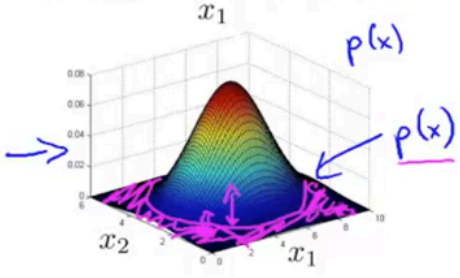
$\mu_1 = 5, \sigma_1 = 2$

$\mu_2 = 3, \sigma_2 = 1$



$p(x_1; \mu_1, \sigma_1^2)$

$p(x_2; \mu_2, \sigma_2^2)$



$\varepsilon = 0.02$

$p(x_{test}^{(1)}) = 0.0426 \geq \varepsilon$

$p(x_{test}^{(2)}) = 0.0021 < \varepsilon$

Building An Anomaly Detection System

Developing and Evaluating an Anomaly Detection System

In the last video we talked about the process of developing a specific application of anomaly detection to a particular problem. This this video we will focus on the process of evaluating an anomaly detection algorithm.

Recall from previous evaluation processes, it is easier to make decisions about these parameters and quality of your models by having a real number metric you can use.

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.

- Assume we have some labeled data, of anomalous and non-anomalous examples. ($y = 0$ if normal, $y = 1$ if anomalous).
- 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: $(x_{test}^{(1)}, y_{test}^{(1)}), \ldots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

**It's okay if a negligible amount of anomalous data is part of the assumed normal examples in the training data set.

Aircraft engines motivating example

- 10000 good (normal) engines
- 20 flawed engines (anomalous) 2-50 $y=1$
- Training set: 6000 good engines ($y = 0$) $\mu_1, \sigma_1^2, \ldots, \mu_n, \sigma_n^2$. $p(x) = p(x_1; \mu_1, \sigma_1^2) \cdots p(x_n; \mu_n, \sigma_n^2)$
- CV: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)
- Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Use the training set (normal examples, non-anomalous, but unlabeled) to fit p(x)

Given the training set, cross validation set, and test set, here is how we develop an evaluation algorithm:

Algorithm evaluation

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

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

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

As with the classification algorithm and skewed datasets, we use the confusion matrix and calculate the F1 score as our metric, and continually tune our parameters on the cross validation set.

Anomaly Detection vs. Supervised Learning

For anomaly detection algorithms, we saw in the previous video that we used some labelled data for evaluating performance, even though this is an unsupervised learning algorithm.

Why don't we just use a supervised learning algorithm to perform anomaly detection?

Let's look at some guidelines for deciding when to use which algorithm:

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.

** Negative examples -> Non-anomalous

** Note that spam is a supervised learning problem because we have such a large collection of spam and non spam emails even though there are so many different types of spam.

There are some cases where anomaly detection could be a supervised problem but it depends on the size and ratio of the dataset.

Example of different applications:

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

One of the biggest factors on performance for anomaly detection algorithms are the features we use/choose to give to the anomaly detection algorithm.

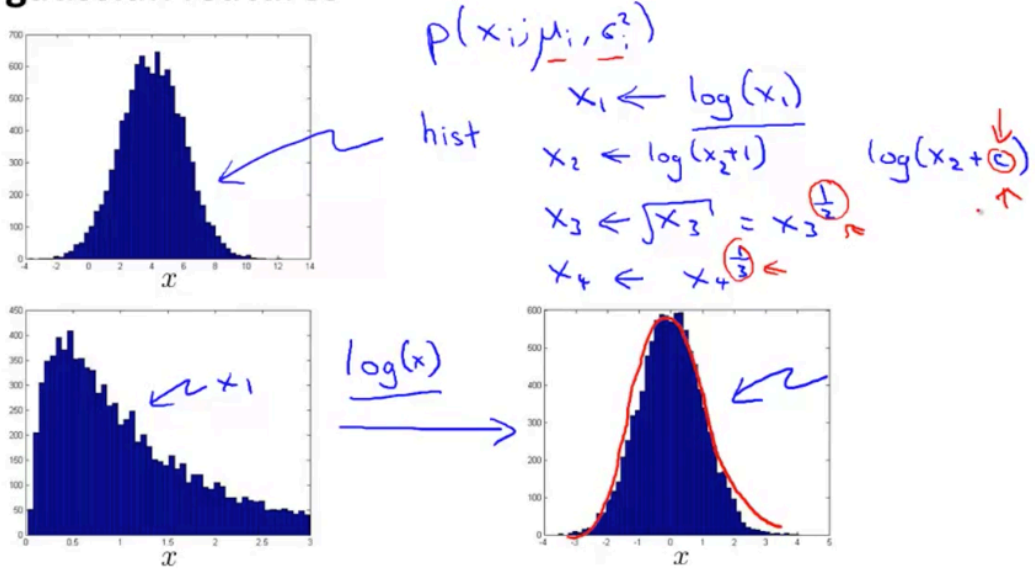
What are the best practices when choosing features to give to an anomaly detection algorithm?

Ideally we want to use gaussian features.

If we have non-gaussian data, it's not the end of the world it can still work.

- We can try playing around with different transformations on the data to make it look more gaussian.
    - There are many different transforms we can try out and play with.

Non-gaussian features



We can come up with a choice of features by using an error analysis procedure (similar to the error analysis procedure we used for supervised learning):

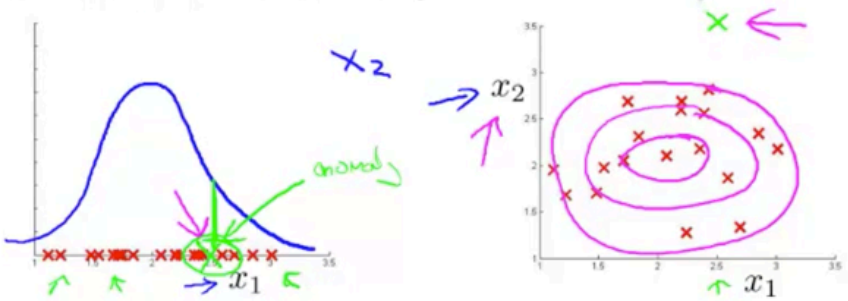
What if the green example is incorrectly classified? We should explore that feature manually and see if there is a problem, ex. x2 is abnormally high. Maybe we can create a new feature that would work better.

→ **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



What are some other good guidelines for picking features?

Ex.

CPU load and Network traffic should grow linearly, so if one is high and one is low that may indicate an anomaly.

Monitoring computers in a data center

Choose features that might take on unusually large or small values in the event of an anomaly.

- x_1 = memory use of computer
- x_2 = number of disk accesses/sec
- x_3 = CPU load
- x_4 = network traffic

$$x_5 = \frac{\text{CPU load}}{\text{network traffic}} \qquad x_6 = \frac{(\text{CPU load})^2}{\text{network traffic}}$$

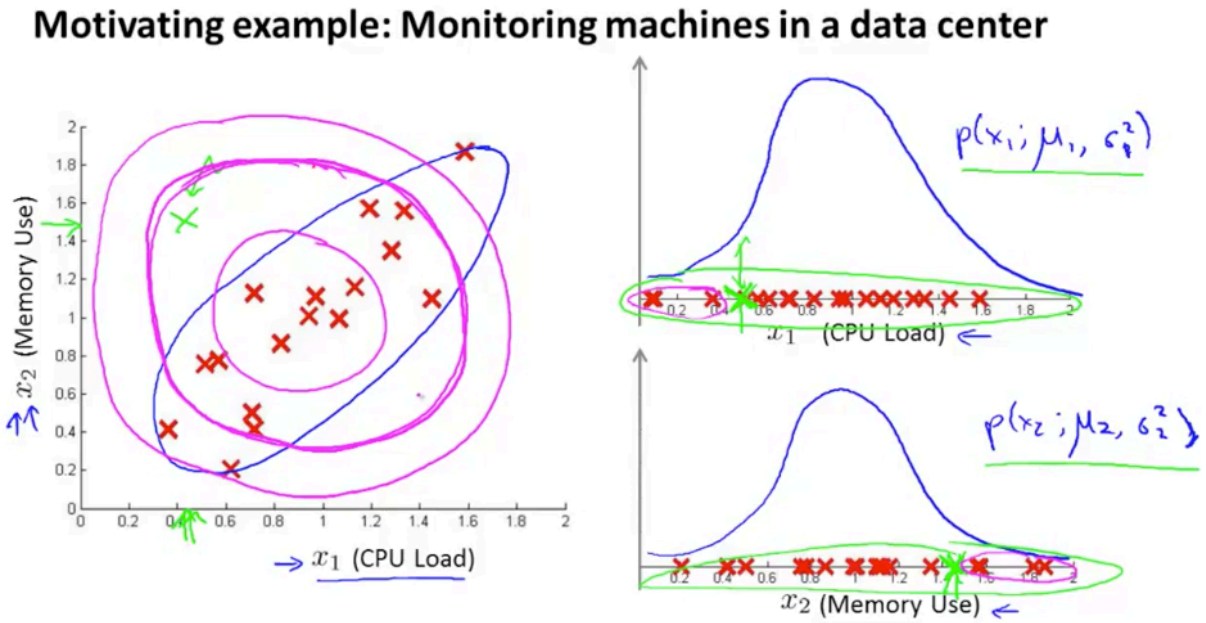
Multivariate Gaussian Distribution

Multivariate Gaussian Distribution

In this video we will discuss an extension to the Anomaly Detection Algorithm we've developed so far.

This extension uses the Multivariate Gaussian Distribution and has some advantages and disadvantages for Anomaly Detection over the previous algorithm.

Ex. Here we can see if the CPU Load and Memory Use grow linearly, but if we look at the individual values of x1 and x2 for the green example, both still lie within the Gaussian distribution (not too far out).



The algorithm doesn't understand that outside of the blue ellipse, the probability of an anomaly is much higher.

As the circles increase, there is higher probability of an anomaly.

As such, we need to create a modified version of the anomaly detection algorithm using the multivariate Gaussian (normal) distribution.

Multivariate Gaussian (Normal) distribution

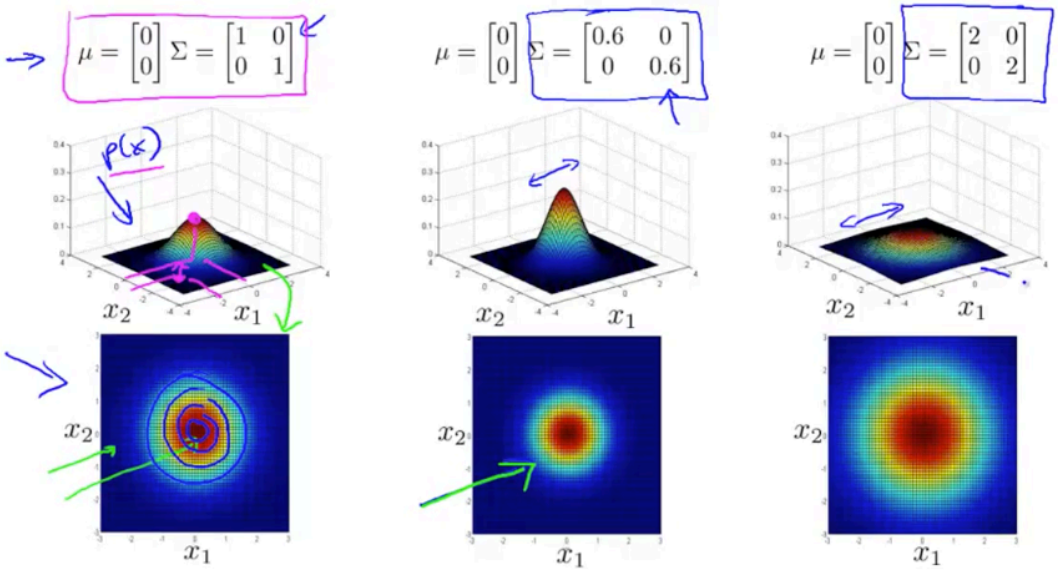
→ $x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \ldots$, etc. separately.
Model $p(x)$ all in one go.
Parameters: $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

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

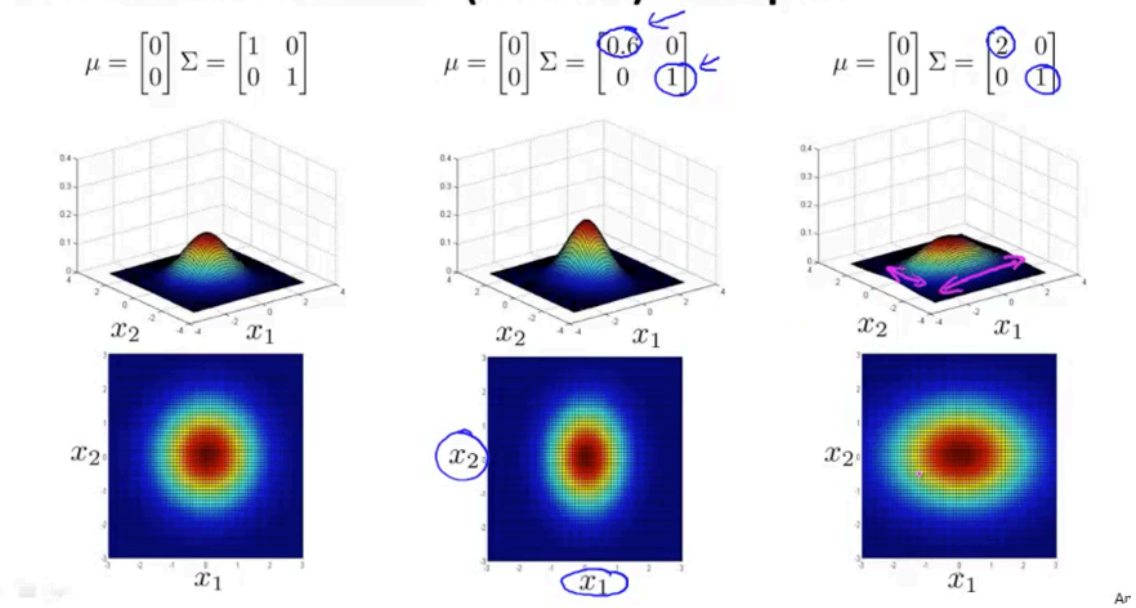
$|\Sigma|$ = determinant of Σ | det (Sigma)

What does this 'new/modified' p(x) look like? Let us look at a 2D example.

Multivariate Gaussian (Normal) examples

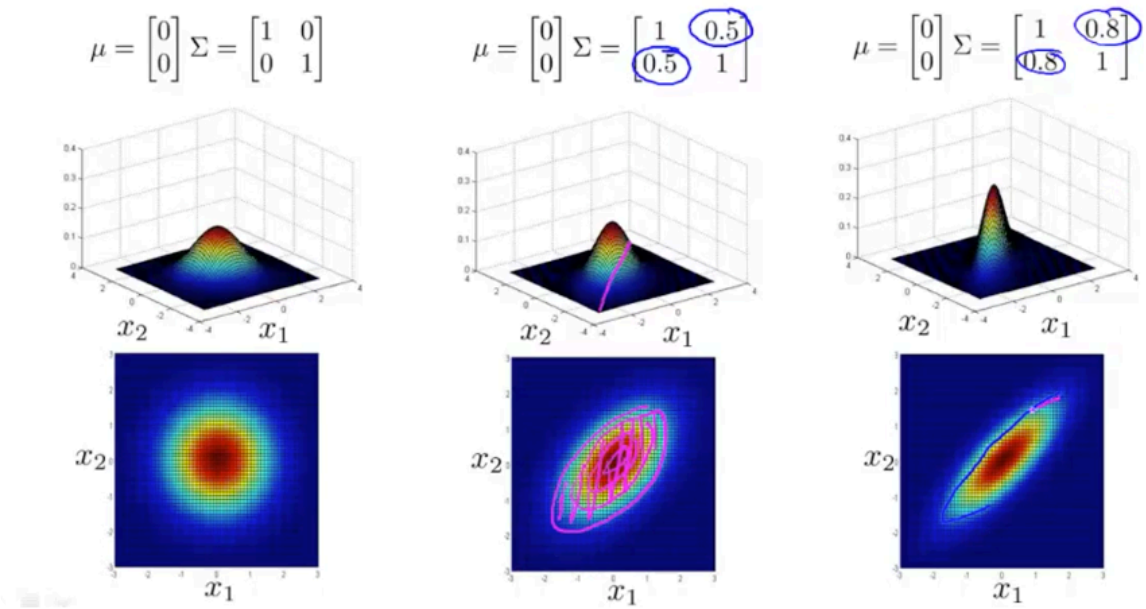


Multivariate Gaussian (Normal) examples

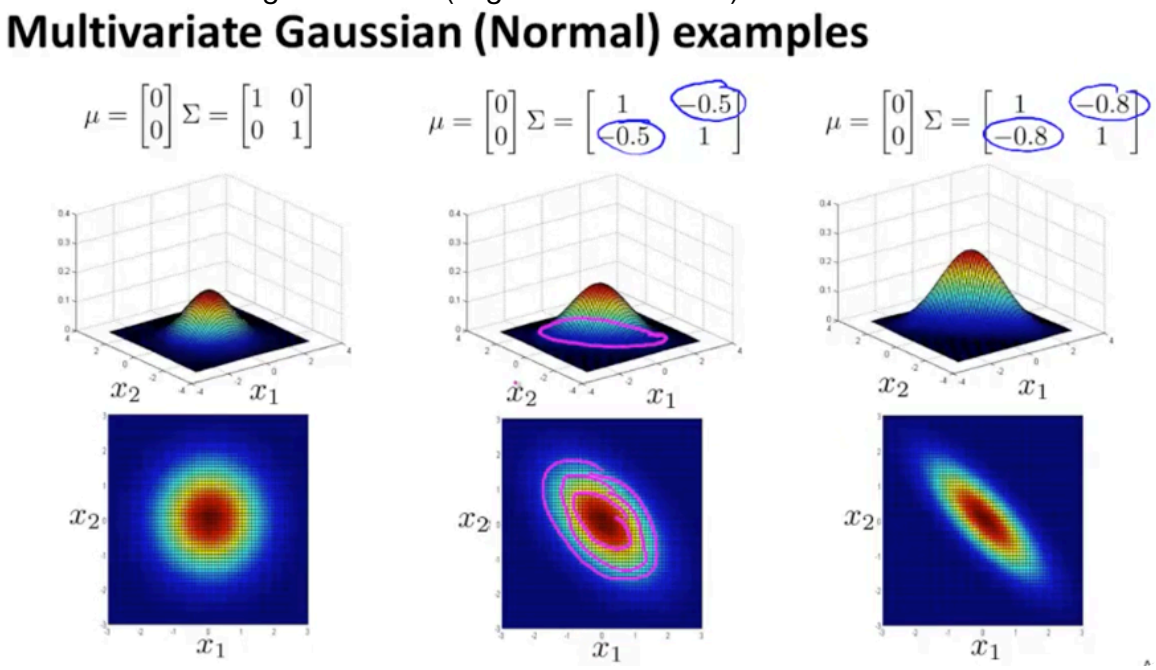


We can also use this to model correlations:

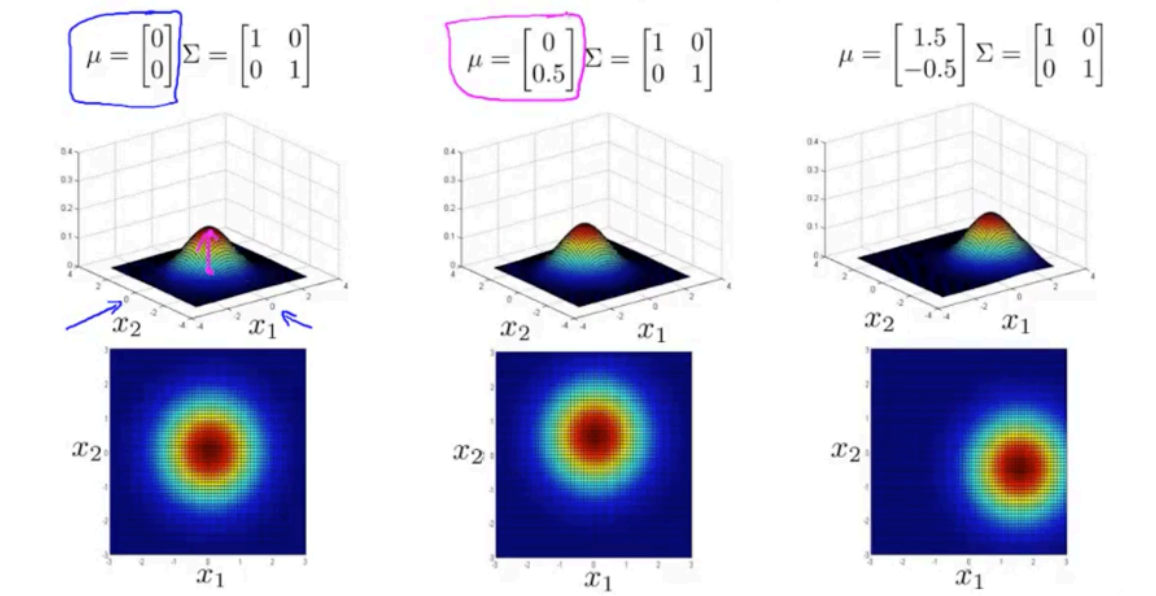
Multivariate Gaussian (Normal) examples



We can also use negative values (negative correlations):



Above we are only changing the covariance matrix, but we can also change the values for mu:



As such with this approach we can capture when we expect two different features to be positively or negatively correlated.

Anomaly Detection using the Multivariate Gaussian distribution

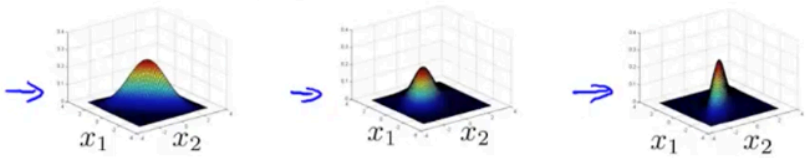
In this video we will apply the ideas we learn from the previous video to developing a different Anomaly Detection Algorithm.

Recall we have two parameters, mu and sigma (the covariance matrix)

Lets look at the parameter fitting / estimation problem.

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

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



Parameter fitting:

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

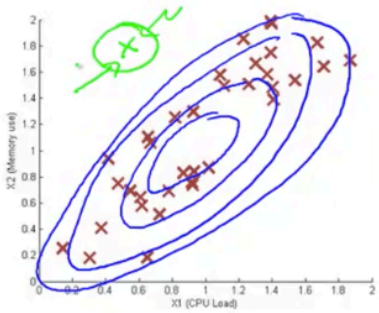
$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} \qquad \Sigma = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

Anomaly detection with the multivariate Gaussian

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

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

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



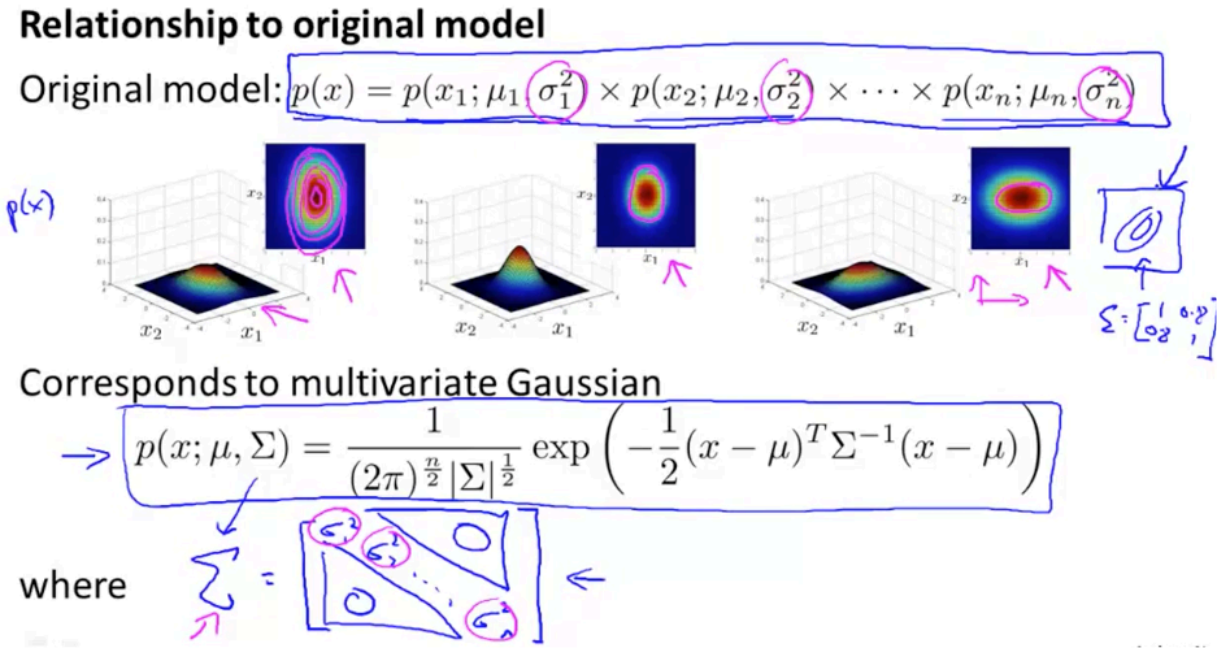
2. Given a new example x, compute

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

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

What is the relationship between the multivariate Gaussian distribution model and the original model?

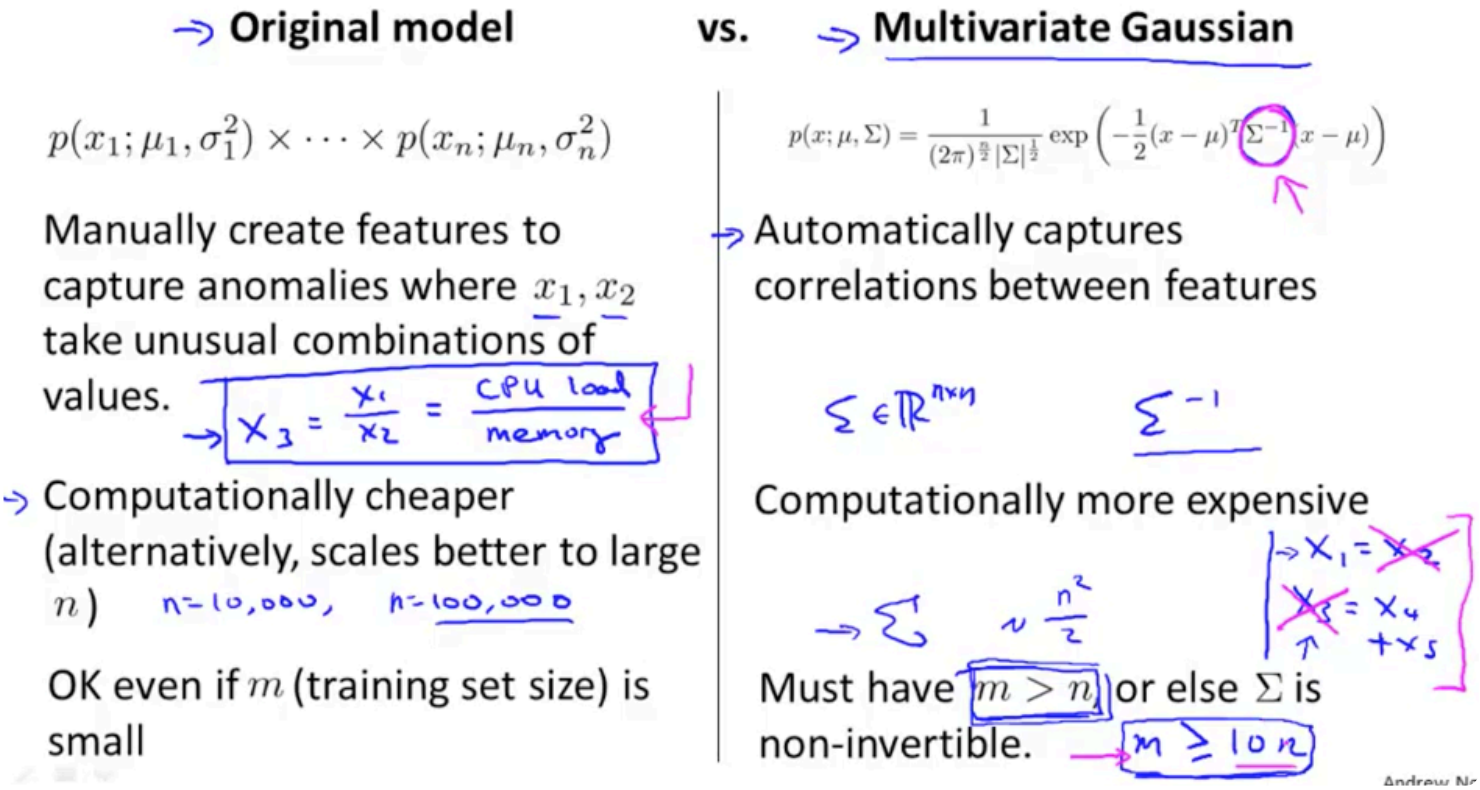
The original model turns out to be a special case of the multivariate Gaussian distribution model where the distribution of p(x) is constrained so that the contours of the probability distribution functions are 'axis aligned'.



I.e. horizontal and vertical, not diagonal.

Mathematically this means the covariance matrix sigma is constrained as shown.

So when would you use either of these two models?



The original model is probably used more often, whereas the multivariate model is used less often but is able to automatically capture the correlations between features.

** Note redundant features == linearly dependent.