15. Anomaly Detection

≫ PROBLEM MOTIVATION:

Inputs:

Anomaly detection example

Aircraft engine features:

- x_1 = heat generated
- x_2 = vibration intensity

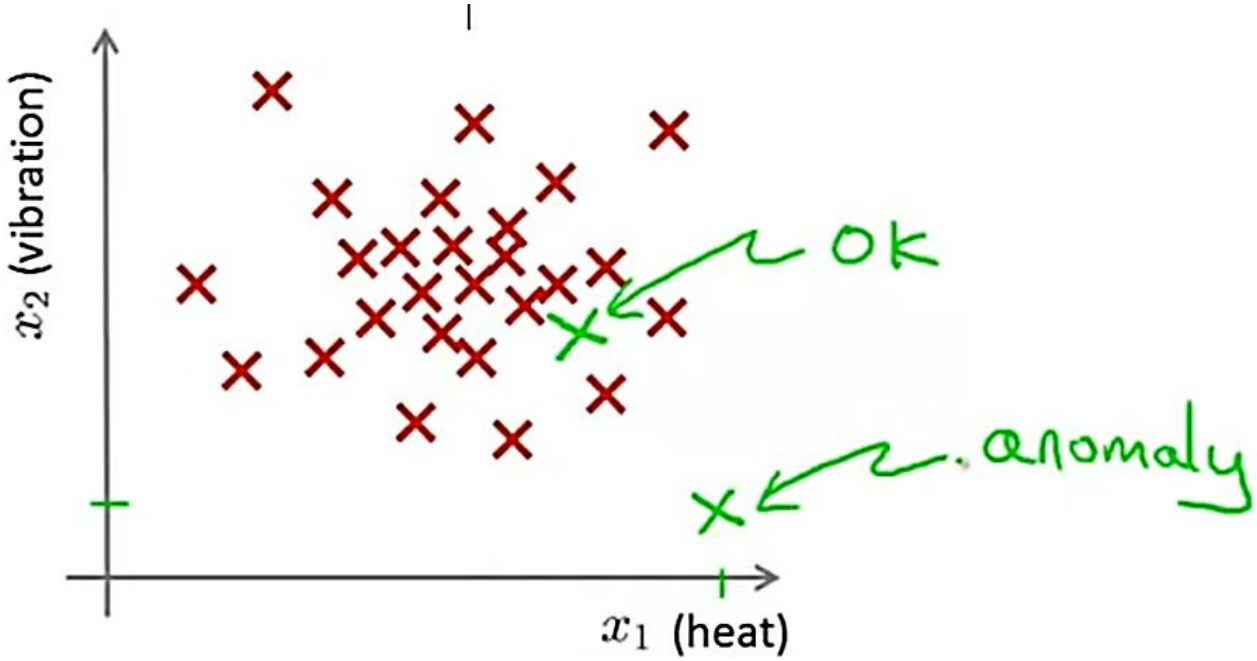
...

Test Data:

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

New engine: x_{test}

Plot:



Problem: we are given some training examples. We assume them to be NON-ANOMALOUS. So, we need to find if a new example is anomalous or not.

We train a PROBABILITY MODEL:

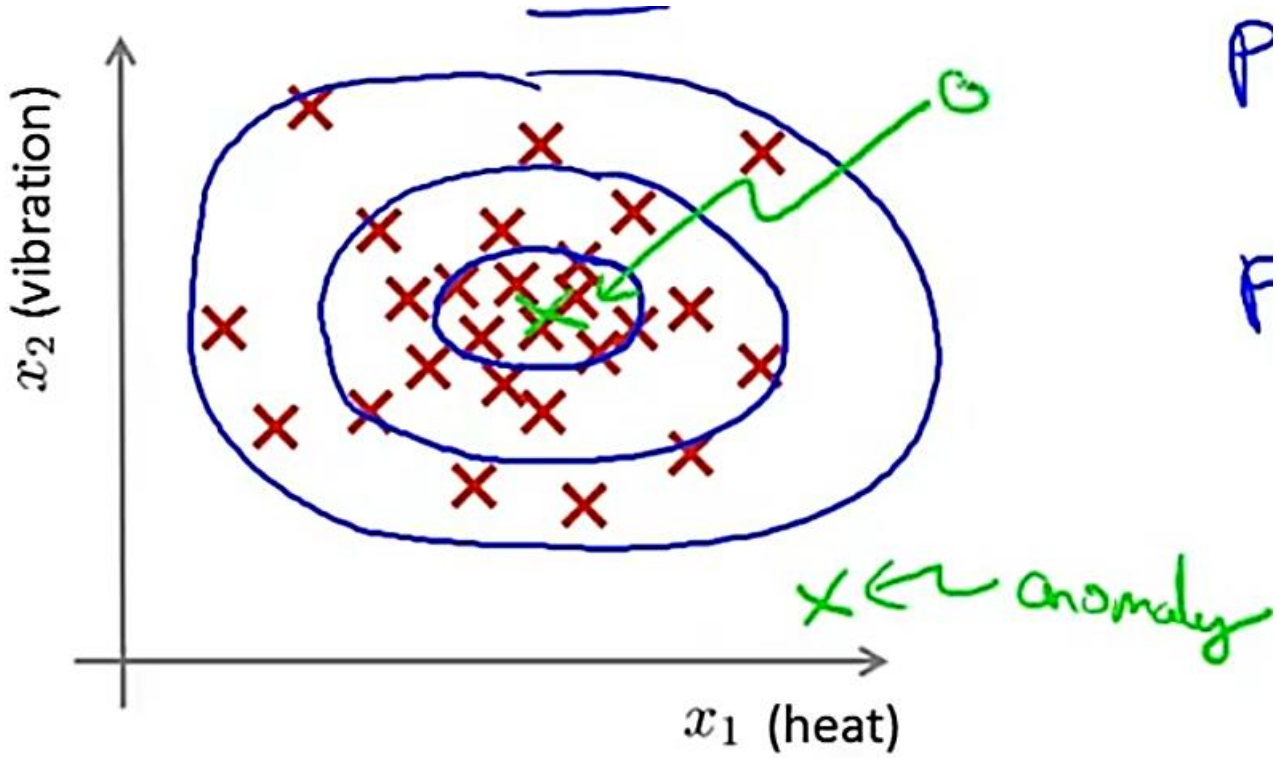
$p(x_{test}) < \varepsilon \rightarrow$ flag anomaly

$p(x_{test}) \geq \varepsilon \rightarrow$ Ok

It divides the plot into various regions, such that each region corresponds to a level of provability.

Centre → highest probability

Outside → lowest P



Example:

Anomaly detection example

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

  x_1 = memory use, x_2 = number of disk accesses/sec,

  x_3 = CPU load, x_4 = CPU load/network traffic.

  ...

≫ GAUSSIAN DISTRIBUTION:

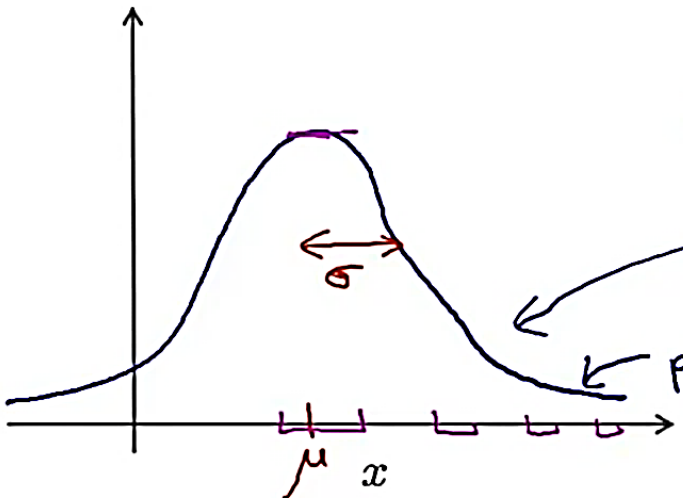
Gaussian (Normal) distribution

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

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

"distributed as"

σ = standard deviation



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

Examples:

σ → controls the width and height of curve

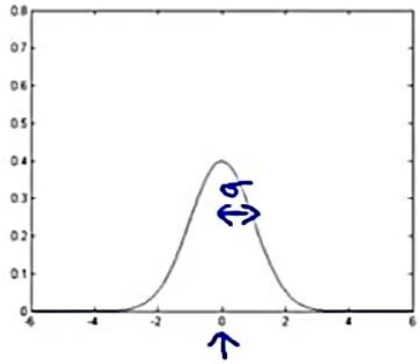
μ → mean of all values of x: controls the center..

Area below the curve is always = 1

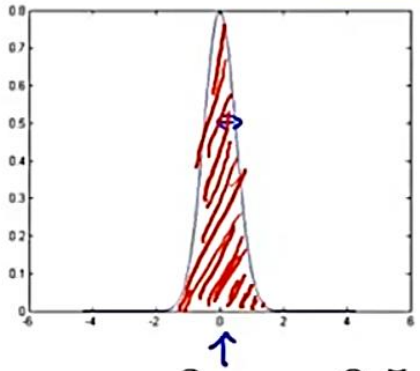
So, the curve is either taller or wider.

Gaussian distribution example

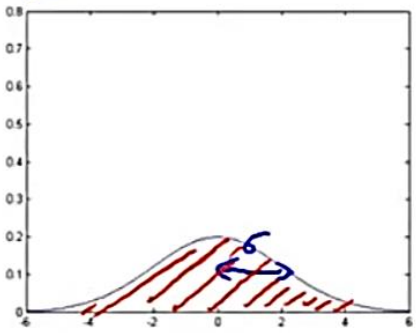
$\mu = 0, \sigma = 1$



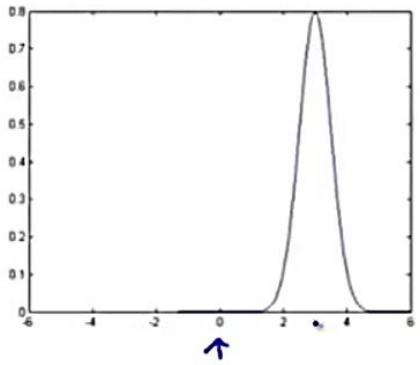
$\mu = 0, \sigma = 0.5$



$\mu = 0, \sigma = 2$



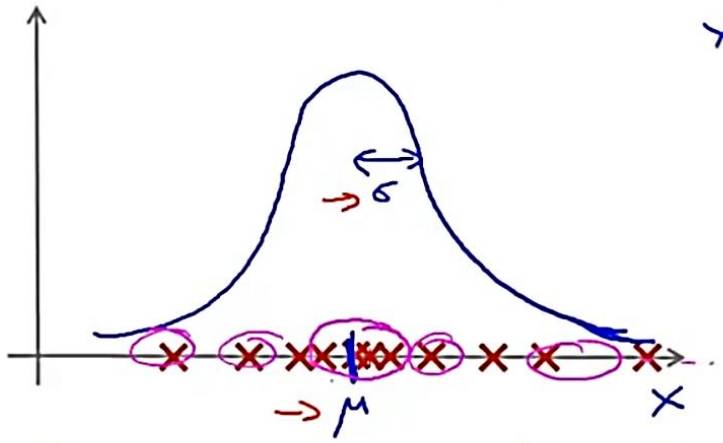
$\mu = 3, \sigma = 0.5$



Estimation of μ and σ :

Parameter estimation

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



≫ ALGORITHM FOR ANOMALY DETECTION:

Density estimation

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

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

Each feature can be individually distributred using Normal(Gaussian) distribution:

$x_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$

$x_2 \overset{\sim}{\uparrow} \mathcal{N}(\mu_2, \sigma_2^2)$

P(x) = product of probabilities of individual features

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

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

Note:

$$\sum_{i=1}^{n} i = 1+2+3+\cdots+n$$

$$\prod_{i=1}^{n} i = 1\times 2\times 3\times\cdots\times n$$

Steps of algorithm:

Anomaly detection algorithm

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

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

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

$p(x_j;\mu_j,\sigma_j^2)$ $\quad \mu_1,\mu_2,\ldots,\mu_n$

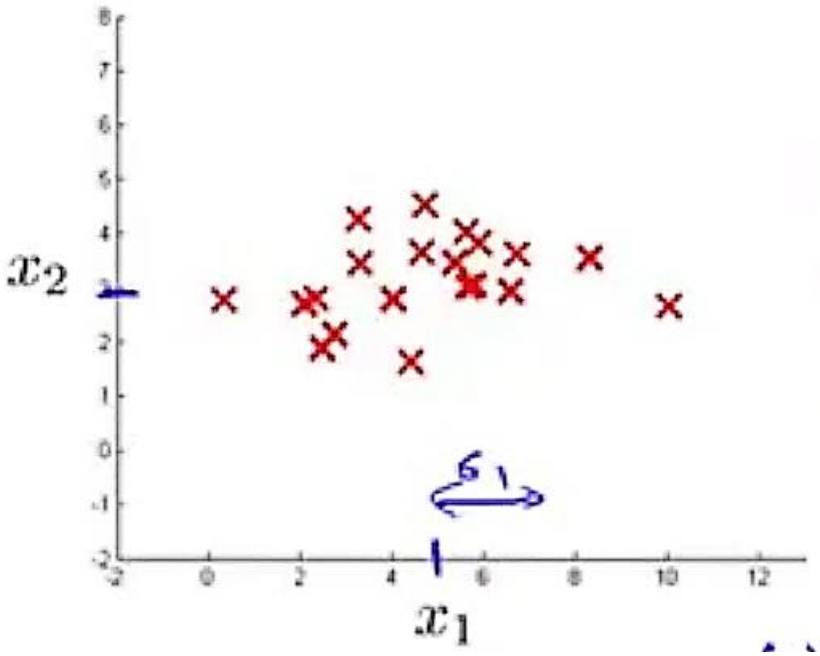
$$\mu = \begin{bmatrix}\mu_1\\ \mu_2\\ \vdots\\ \mu_n\end{bmatrix} = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

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

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

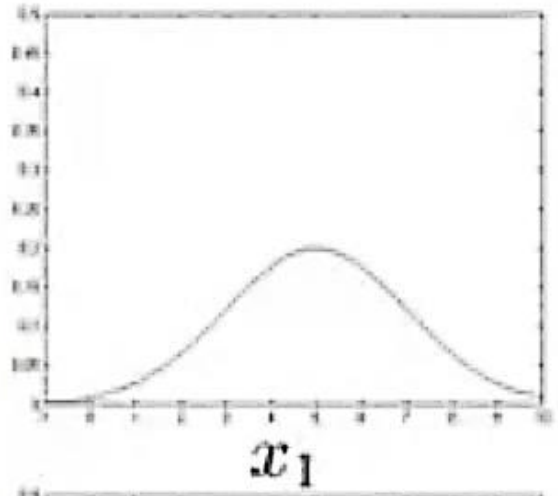
Anomaly if $p(x) < \varepsilon$

Example:

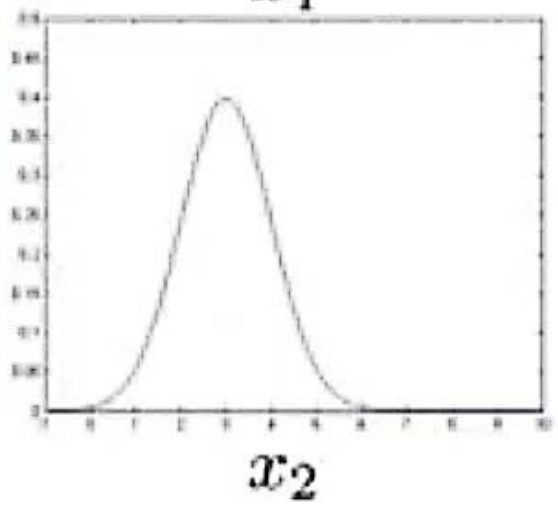


$$\mu_1 = 5, \sigma_1 = 2$$
$$\mu_2 = 3, \sigma_2 = 1$$

Gaussian Curves of both features:

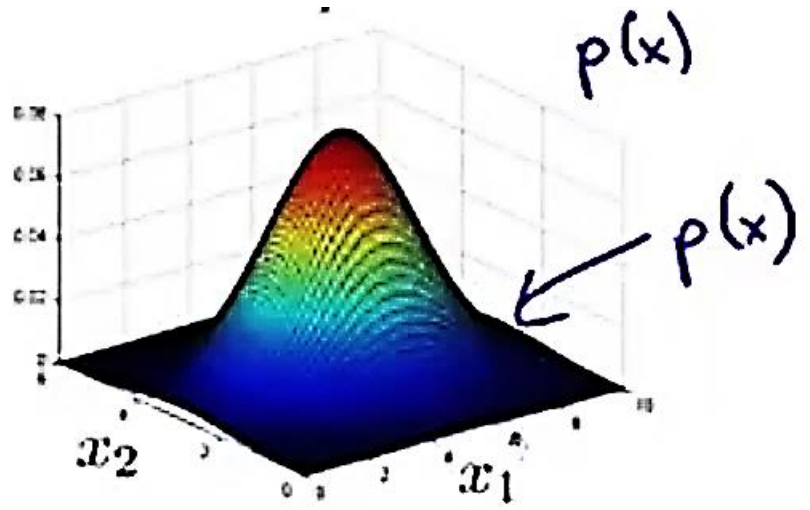


$$p(x_1; \mu_1, \sigma_1^2)$$

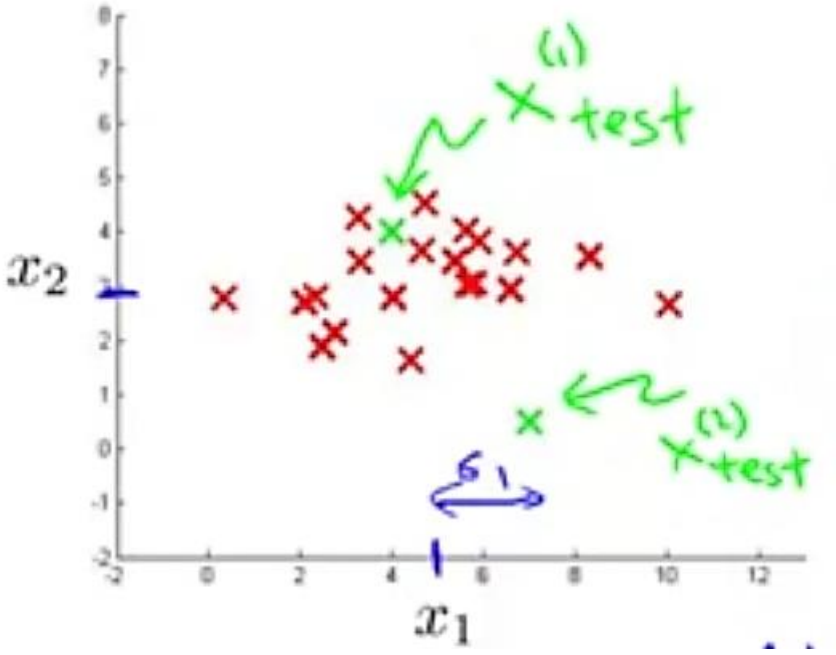


$$p(x_2; \mu_2, \sigma_2^2)$$

P(x) → height of curve == **$P(x_1 ; \mu_1 , \sigma^2_2)$ x $P(x_2 ; \mu_2 , \sigma^2_2)$**



For a new example:



$$\varepsilon = 0.02$$

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

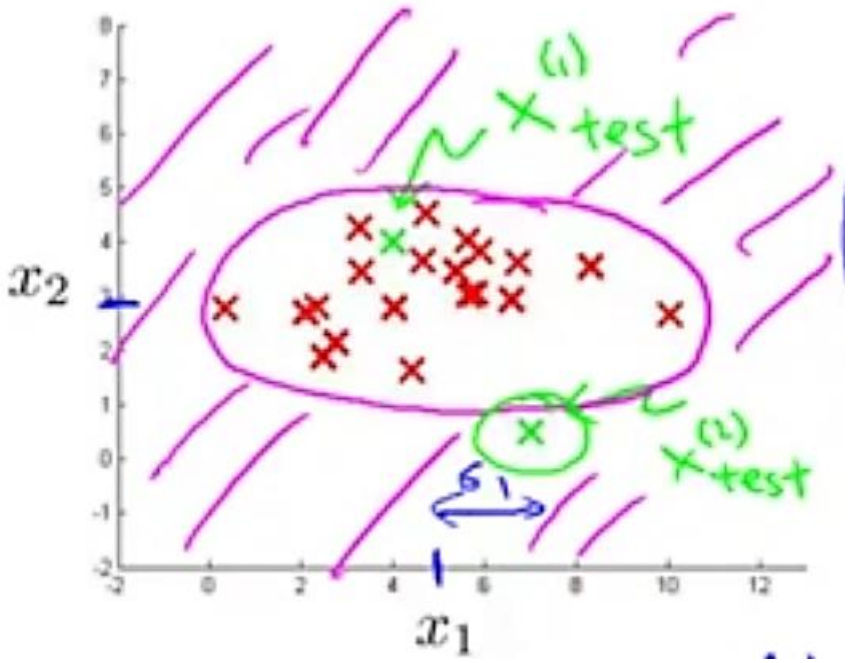
$$p(x^{(2)}_{test}) = 0.0021 < \varepsilon$$

This means that anything below a particular height in the plot, given **by ϵ :** is an anomaly

OR

Anything outside the learned region is an anomaly:

i.e., everything outside the magenta curve:



≫ DEVELOPING ANOMALY DETECTION SYSTEM:

Anomaly detection system problem can be converted to a likes of Supervised Learning problem.

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.

Therefore:

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

Thus we can take:

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ (assume normal examples/not anomalous)

Meaning all the training set examples are non-anomalous

In Cross validation set and test set, we can have some examples of **anomalous (y=1)** and some of **non-anomalous (y=0)** type:

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

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

Example:

Aircraft engines motivating example

10000 good (normal) engines

20 flawed engines (anomalous) 2-50 $y=1$

Training set: 6000 good engines ($y=0$) $\rightarrow \mu_1, \sigma_1^2, \ldots, \mu_n, \sigma_n^2$. $p(x) = p(x_1; \mu_1, \sigma_1^2) \ldots p(x_n; \mu_n, \sigma_n^2)$

CV: 2000 good engines ($y=0$), 10 anomalous ($y=1$)

Test: 2000 good engines ($y=0$), 10 anomalous ($y=1$)

Algorithm evaluation

→ Fit model $p(x)$ on training set $\{x^{(1)}, \ldots, x^{(m)}\}$ $(x_{test}^{(i)}, y_{test}^{(i)})$

→ On a cross validation/test example x, predict

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

$y=0$

≫ How to evaluate the algorithm:

Possible evaluation metrics:

- → - True positive, false positive, false negative, true negative
- → - Precision/Recall
- → - F_1-score

≫ How to choose ϵ :

Can also use cross validation set to choose parameter ε

We can choose the value of ϵ which gives the best value of F_1 score.

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.

In anomaly detection, it's better to model anomalies based on negative examples, rather than positive examples… as future anomalies may be totally different.

Applications of Anomaly detection vs Supervised Learning:

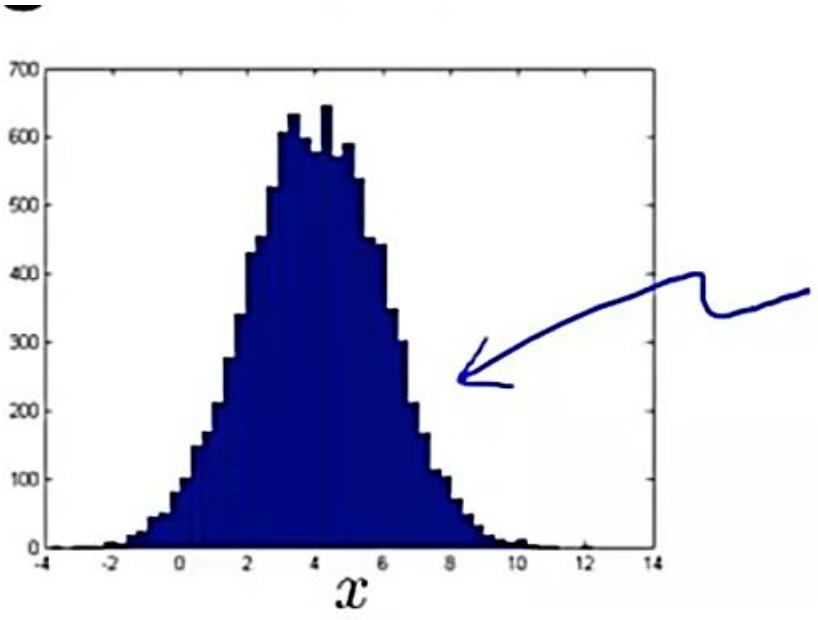
Anomaly detection	vs.	Supervised learning
• Fraud detection $y=1$		• Email spam classification
• Manufacturing (e.g. aircraft engines)		• Weather prediction (sunny/rainy/etc).
• Monitoring machines in a data center		• Cancer classification
⋮		⋮

Fraud detection can be a supervised learning application but only if there are a lot of people on the website who are doing fraudulent activity, i.e., most of the examples are positive, otherwise, it's an anomaly detection problem only

---

≫ Choosing what features to use:

We plot the data and the histogram looks like :

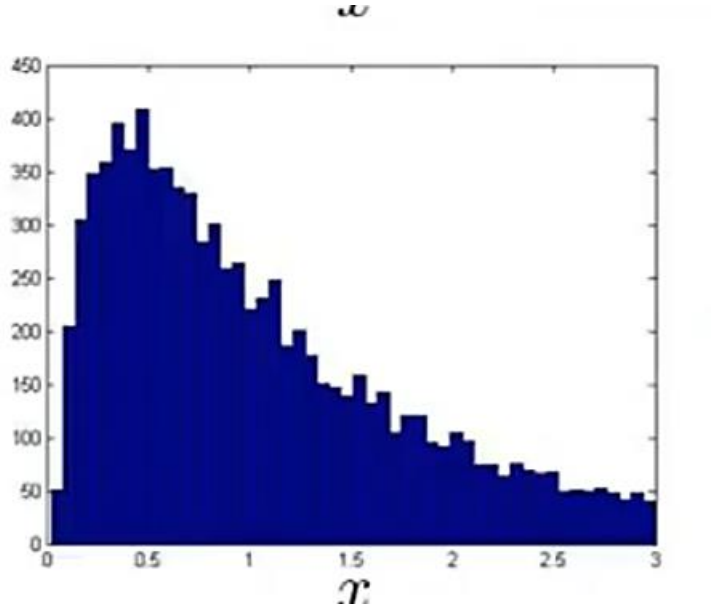


$p(x_i; \mu_i, \sigma_i^2)$

hist

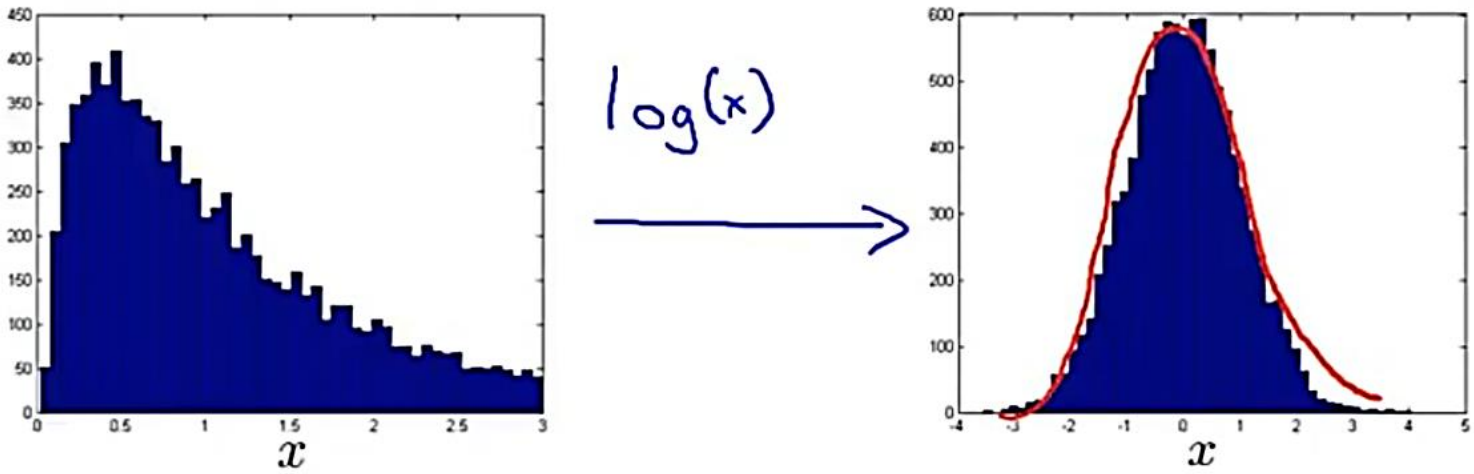
We'd be happy to see this as this means that the feature x is a gaussian feature

But: if the histogram looks like:



This is a non gaussian distribution

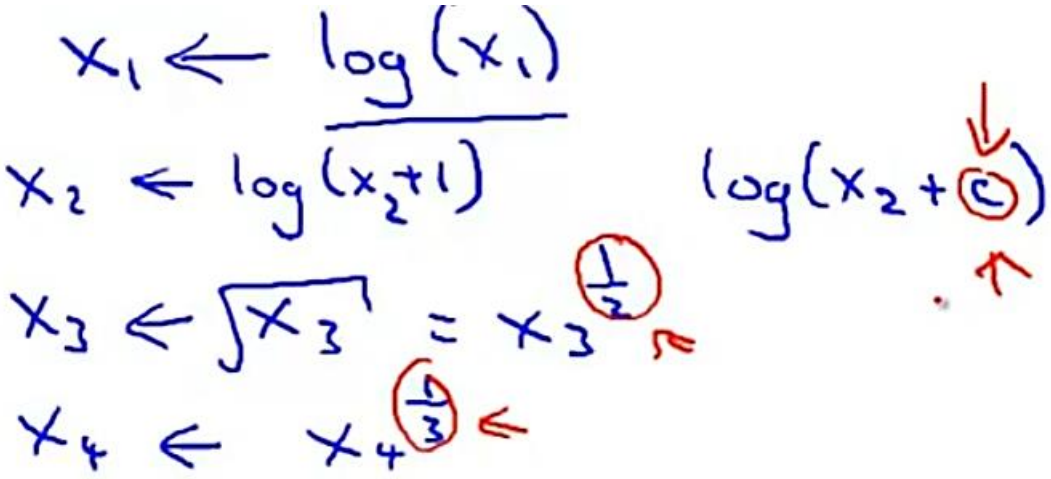
So, we use different transforms on our data to make it as close as possible to a gaussian distribution:



➔ These are feature v/s P(x)

We can use different transforms for different features to make them gaussian features:

We may have to try out different transforms for the same feature to find the best (which gives the best gaussian look to the data).



These parameters in the red are parameters we can vary to make the data look more and more Gaussian.

≫ Coming up with features:

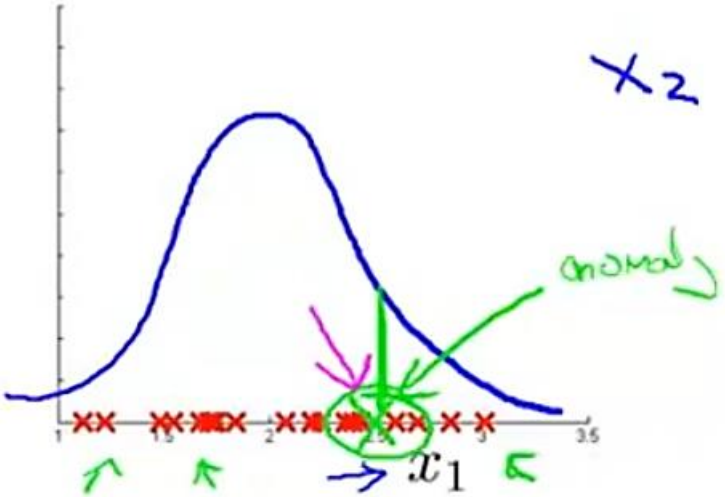
Error analysis method:

⇒ Error analysis for anomaly detection

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

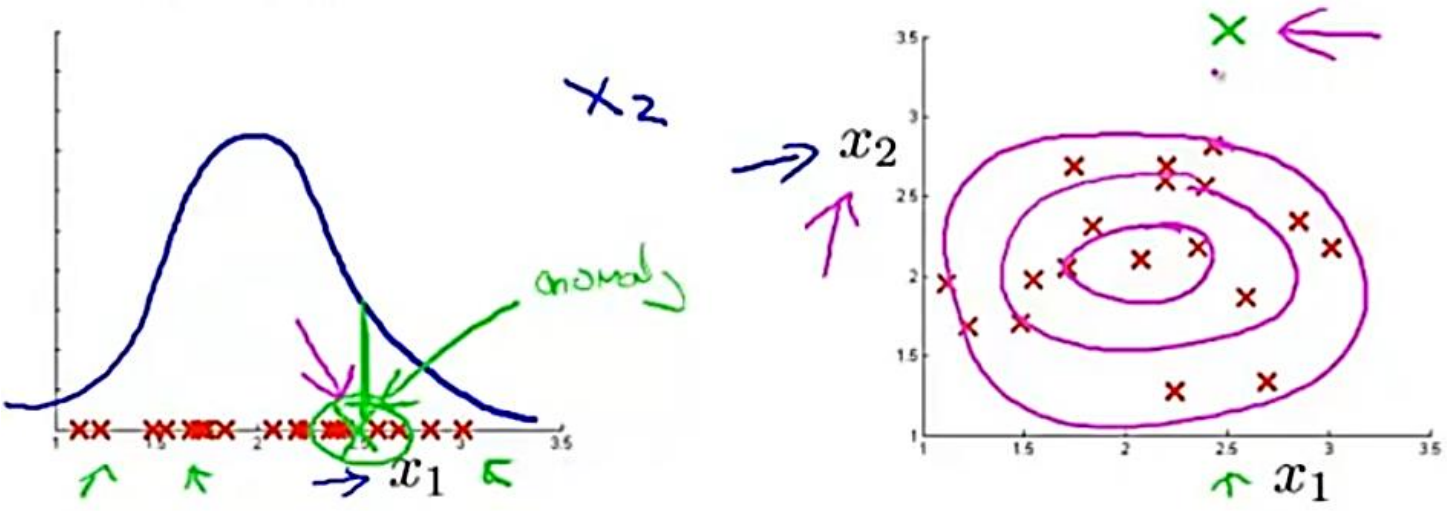
Most common problem:

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



If there is an anomalous example in middle of some non-anomalous examples, then the algo will fail.

- So, we can look at that particular example and try to come up with a new feature that can tell what went wrong with that example



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

Example:

Monitoring computers in a data center

x_1 = memory use of computer

x_2 = number of disk accesses/sec

x_3 = CPU load

x_4 = network traffic

It may occur that if one of the computers is stuck in an infinite loop, the CPU load grows but the network traffic doesn't:

Then we can come up with a new feature:

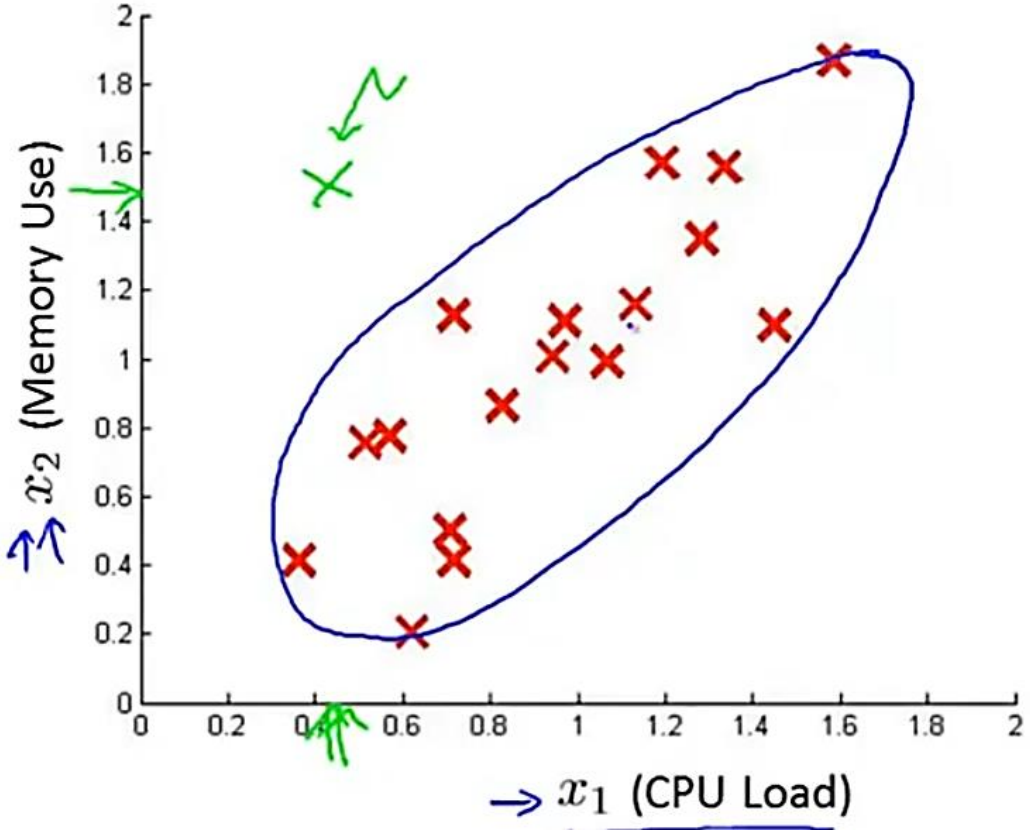
$$x_5 = \frac{\text{CPU load}}{\text{network traffic}}$$

OR

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

≫ MULTIVARIATE GAUSSIAN DISTRIBUTION:

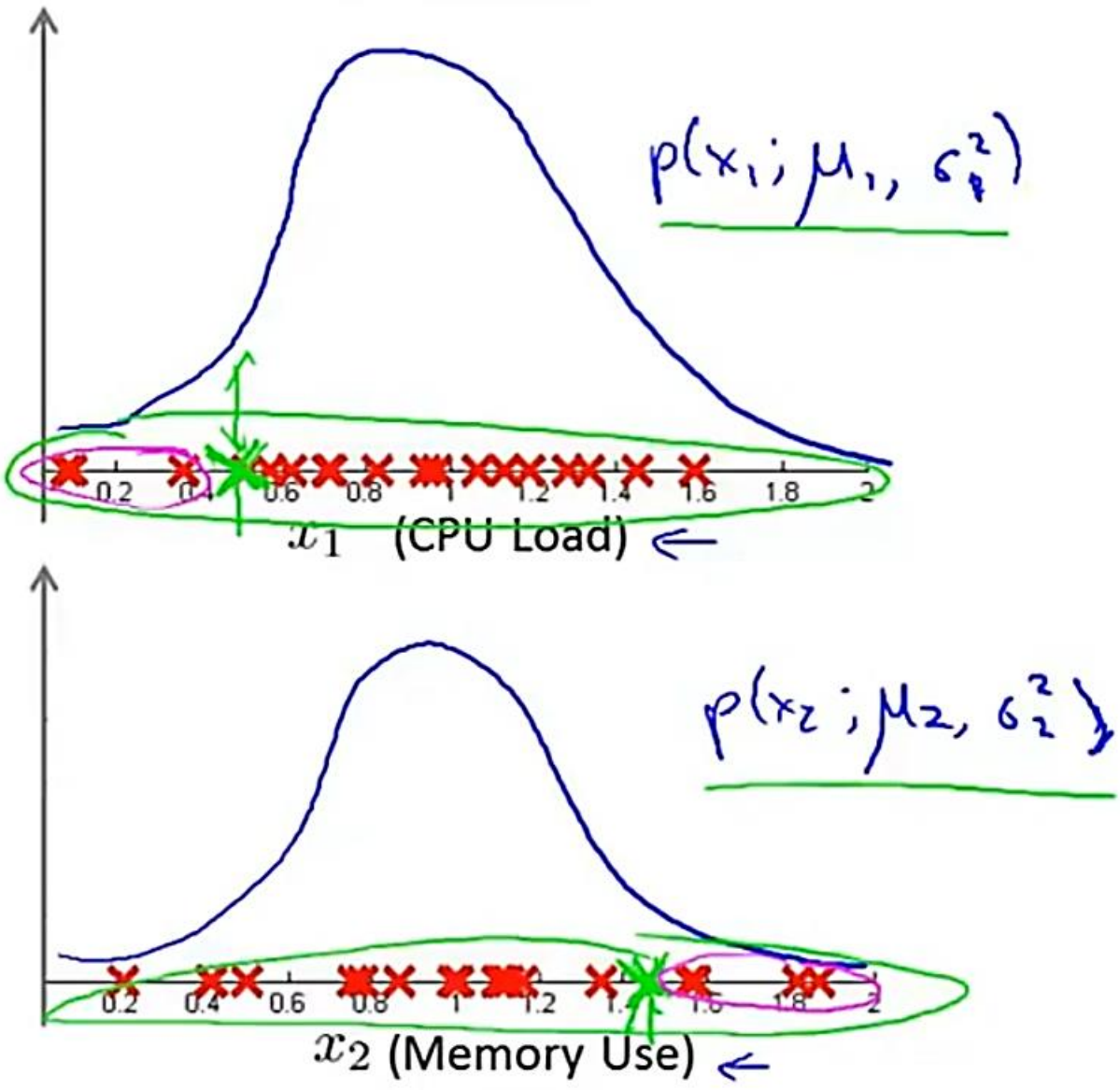
Motivating example: Monitoring machines in a data center



Here, red points are training data

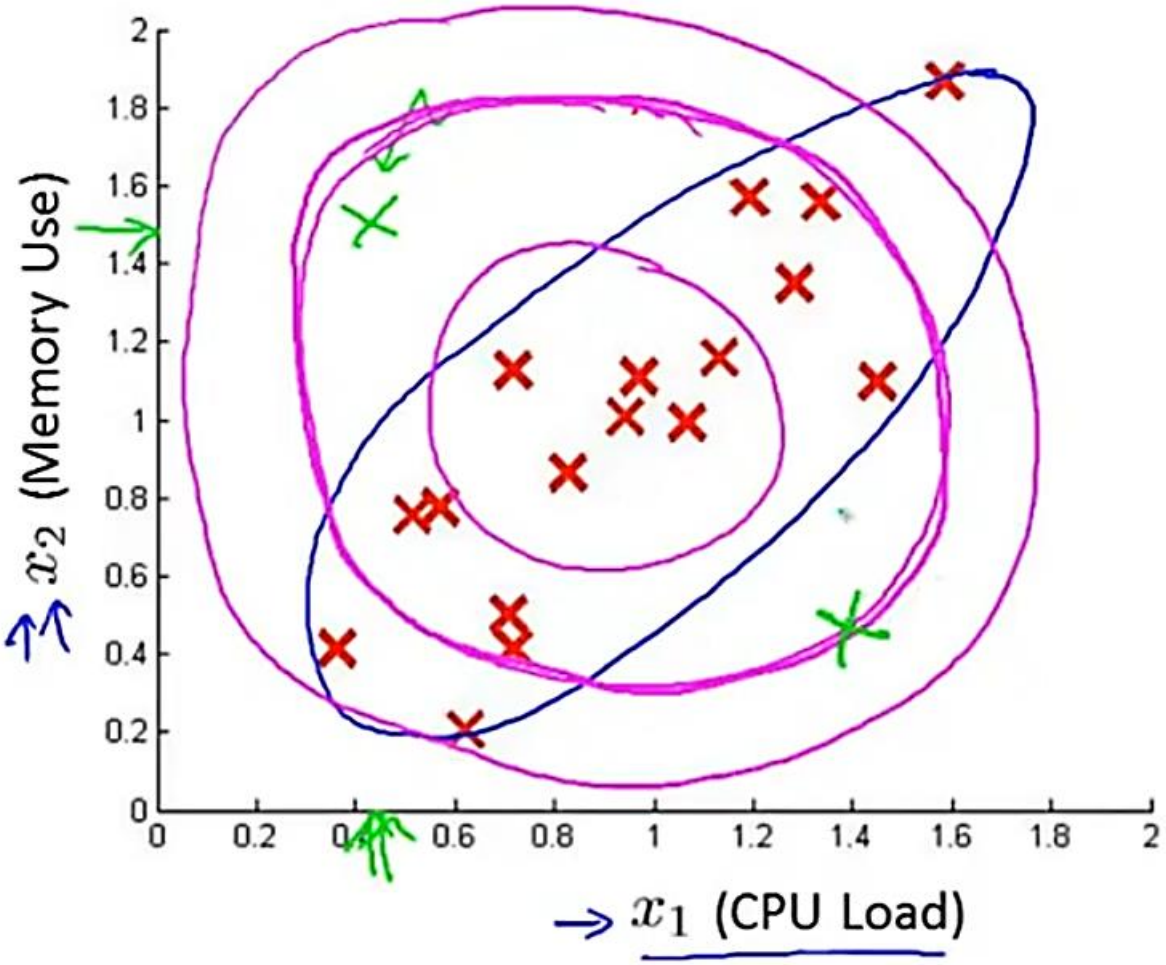
Green point is our test data

- Lets look at both features individually:



The algo will not predict the right o/p … since the i/p data is distributed on the whole axis, so all the points have some probability of being correct.

Probability Contours:



To solve this:

Multivariate Gaussian (Normal) distribution

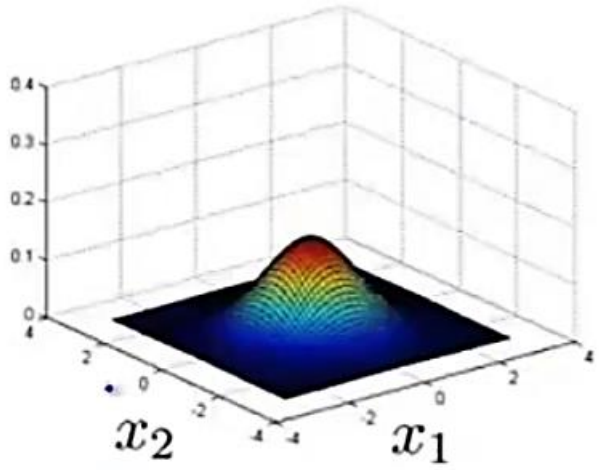
- $x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \ldots$, etc. separately. Model $p(x)$ all in one go.
  Parameters: $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

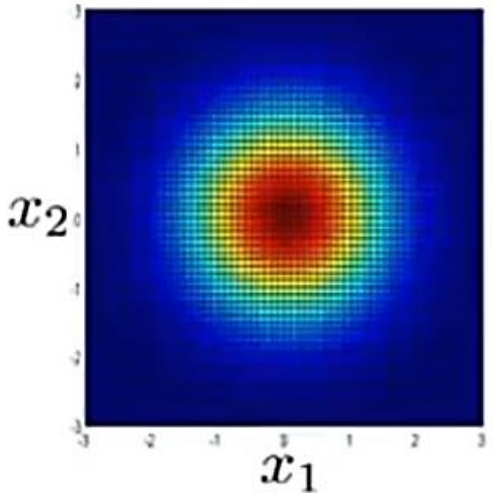
$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

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

Examples:

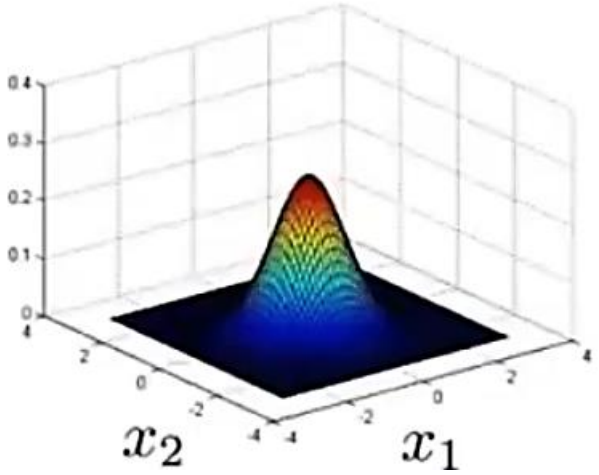
$$\mu = \begin{bmatrix}0\\0\end{bmatrix} \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$$

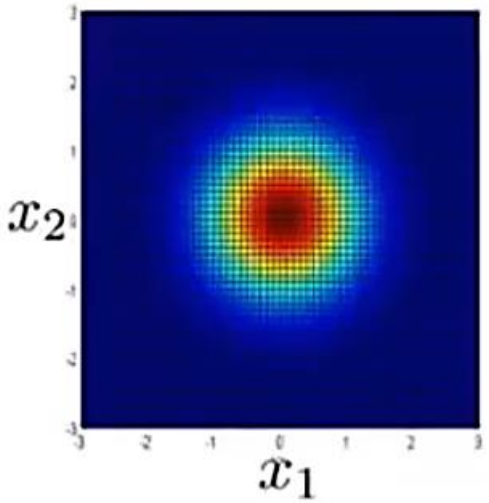




If we decr Σ :

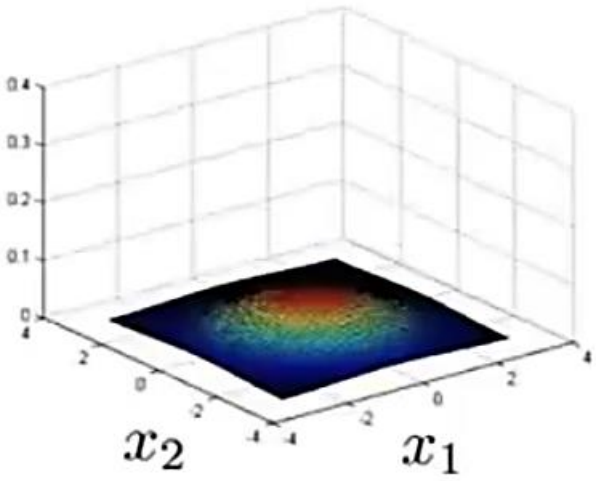
$$\mu = \begin{bmatrix}0\\0\end{bmatrix} \Sigma = \begin{bmatrix}0.6 & 0\\0 & 0.6\end{bmatrix}$$

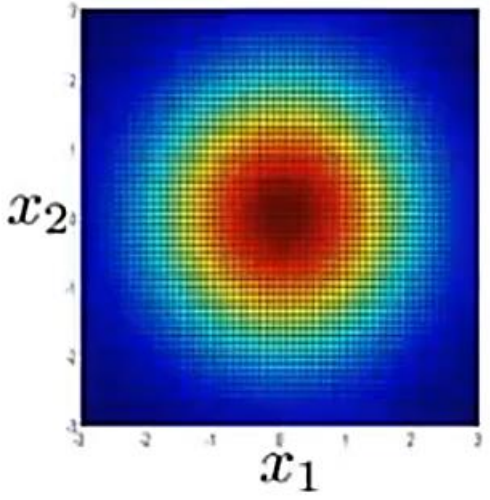




If we incr Σ :

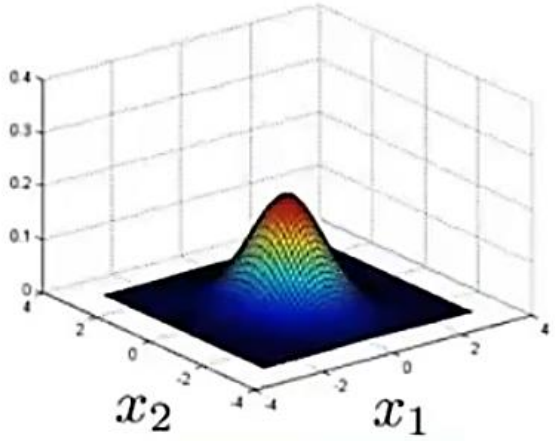
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

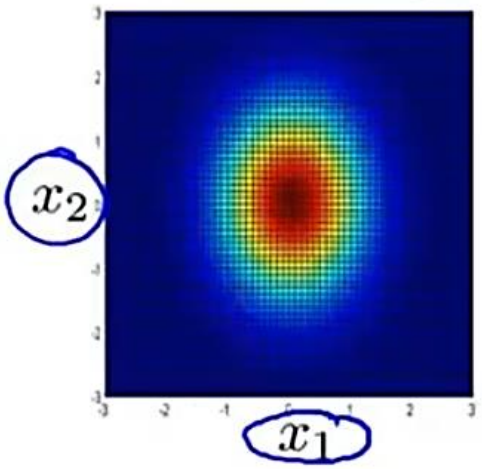




If we decr the variance of only one of the features:

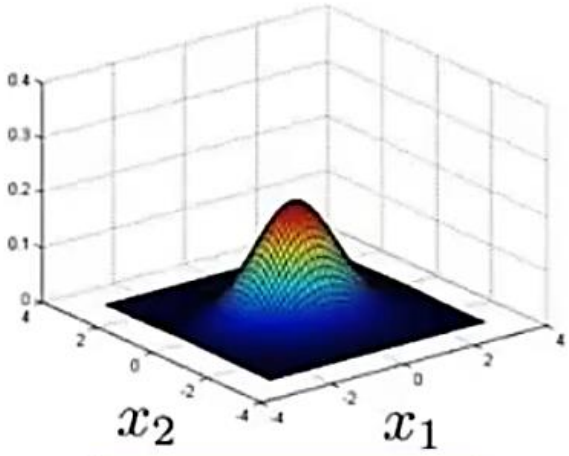
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 0.6 & 0 \\ 0 & 1 \end{bmatrix}$$

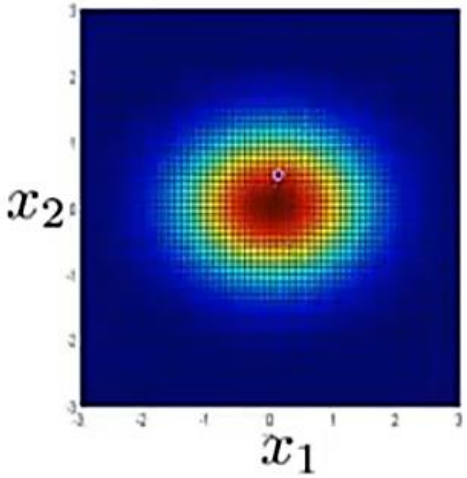




OR

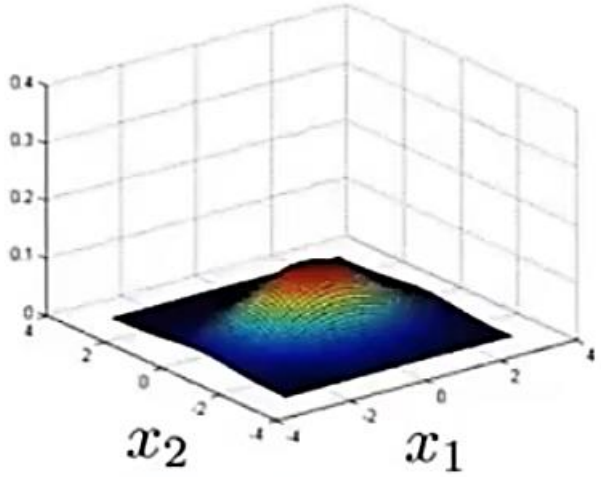
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 0.6 \end{bmatrix}$$

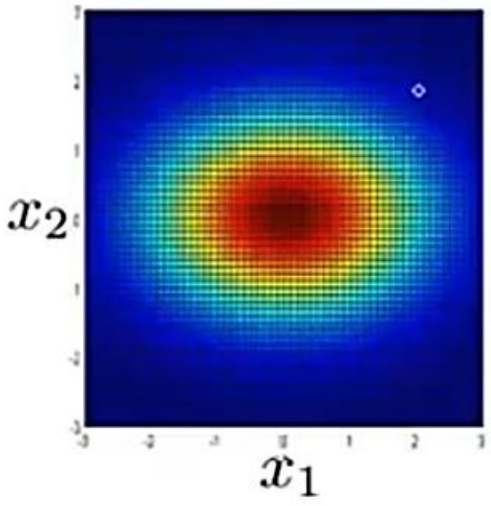




If we incr the variance of only one of the features:

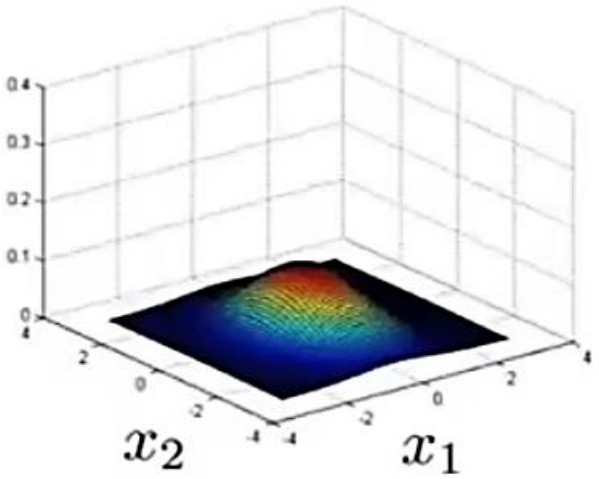
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

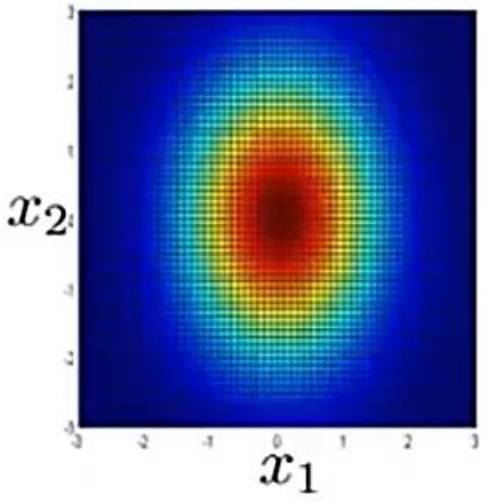




OR

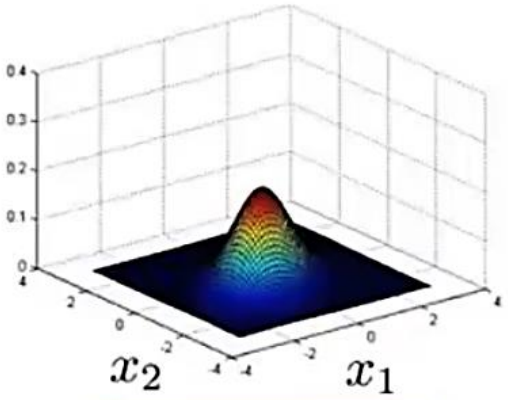
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

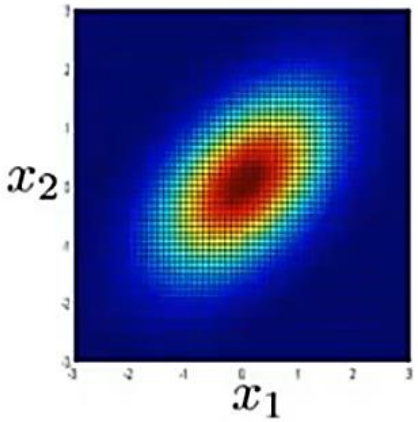




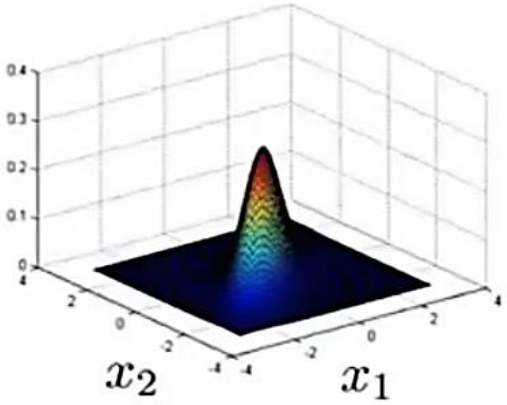
Other variations in Σ :

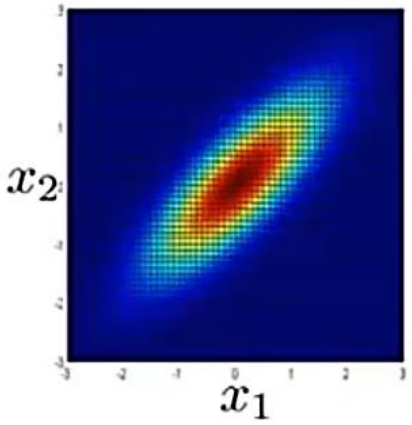
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$



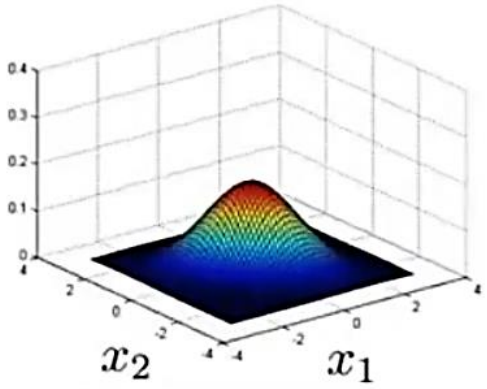


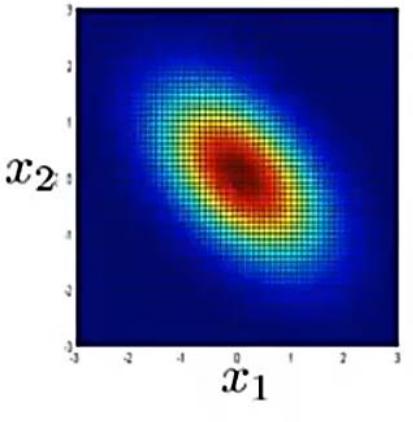
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$



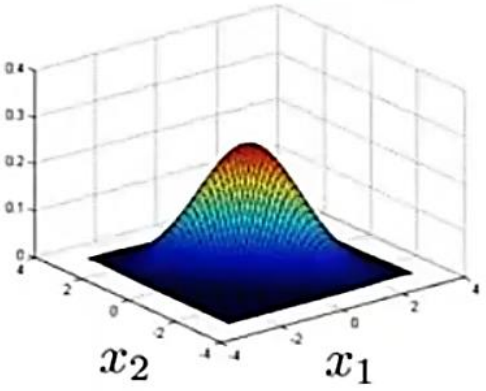


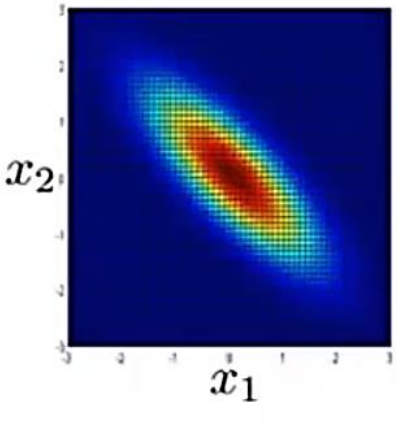
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$$





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

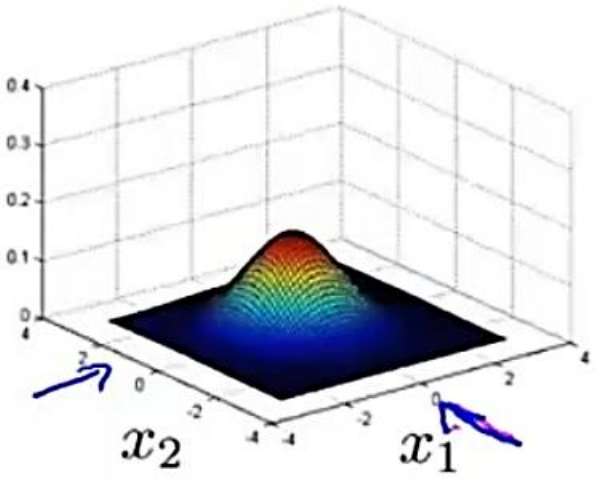


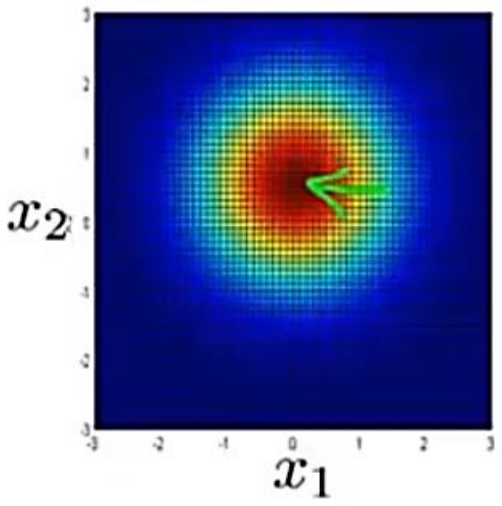


An

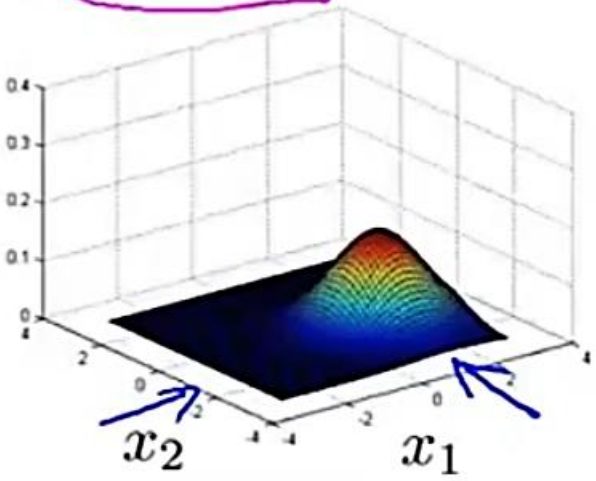
Varying the mean (μ): it moves the centre of contours, where the probability (P(x)) is highest.

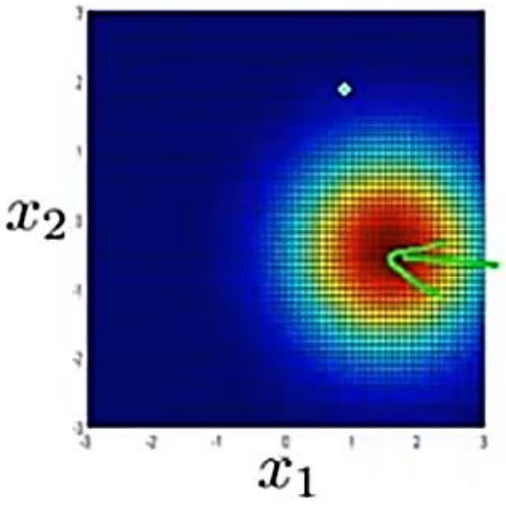
$$\mu = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$





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



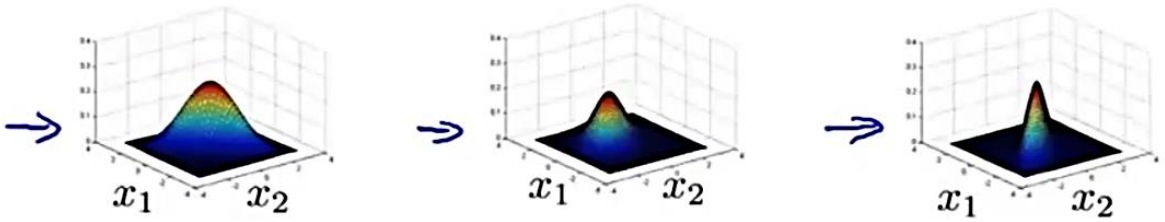


≫ Multivariate Gaussian Distribution Algorithm:

Multivariate Gaussian (Normal) distribution

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

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

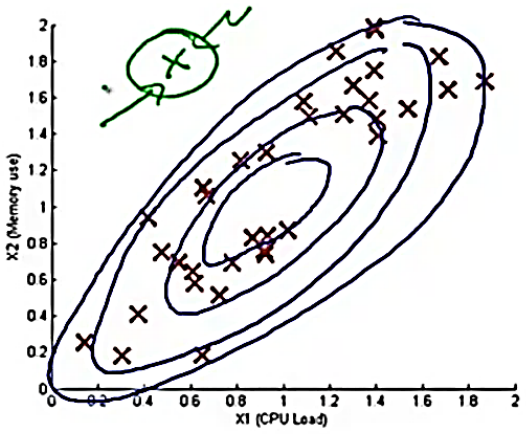
Algorithm:

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

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

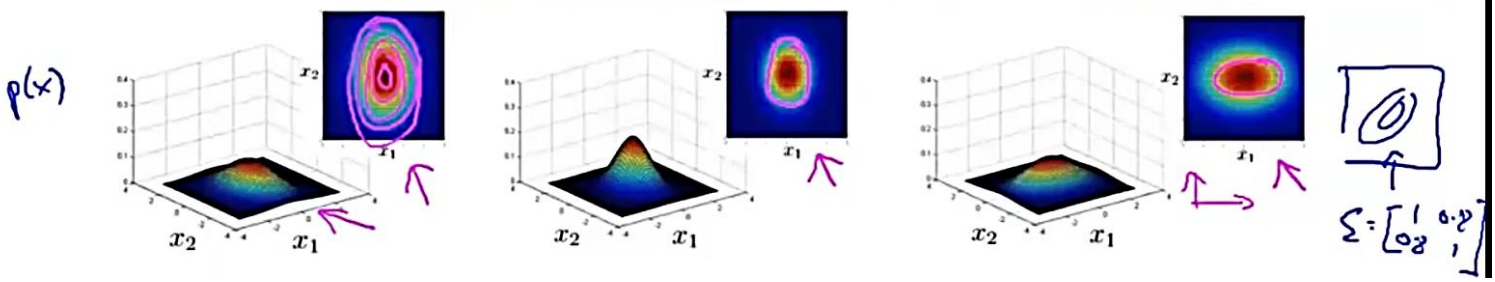
Flag an anomaly if $p(x) < \varepsilon$

≫ Relationship of multivariate Gaussian Model with Original Gaussian Model:

Original gaussian model is actually a special case of multivariate model, in which, the contours have their axes aligned with the features axes, i.e., the contours are not at nay angles:

Relationship to original model

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



Corresponds to multivariate Gaussian

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

Original model is mathematically the multivariate model with a constraint, that is:

≫ WHEN TO USE ORIGINAL MODEL vs MULTIVARIATE MODEL:

→ Original model	vs.	→ Multivariate Gaussian
$p(x_1;\mu_1,\sigma_1^2)\times\cdots\times p(x_n;\mu_n,\sigma_n^2)$		$p(x;\mu,\Sigma)=\frac{1}{(2\pi)^{\frac{n}{2}}\lvert\Sigma\rvert^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$
Manually create features to capture anomalies where x_1, x_2 take unusual combinations of values. → $x_3 = \frac{x_1}{x_2} = \frac{\text{CPU load}}{\text{memory}}$		→ Automatically captures correlations between features
→ Computationally cheaper (alternatively, scales better to large n) n=10,000, n=100,000		Computationally more expensive → $\Sigma \sim \frac{n^2}{2}$
OK even if m (training set size) is small		Must have $m > n$ or else Σ is non-invertible. → $m \geq 10n$

Andrew

- In some cases, in original model, we may require to manually create extra features, so that the model can work fine.
- In case of multivariate model, its important to get rid of redundant features, o/w the algo is very expensive, and Σ may even be non-invertible

.....

.....