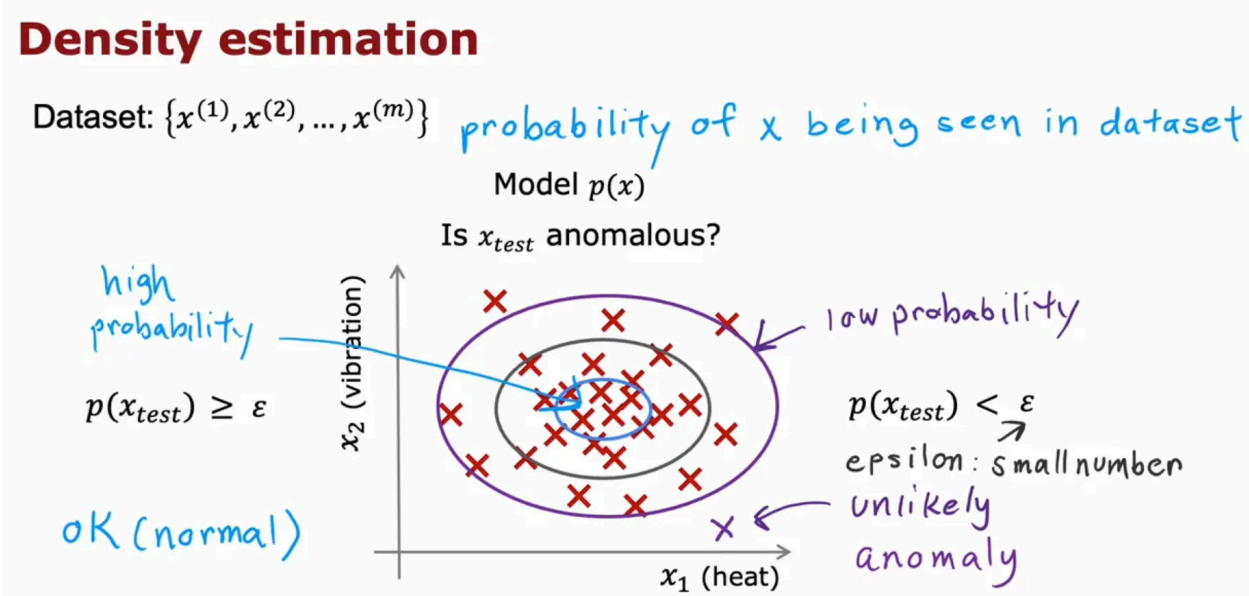
Unsupervised Learning, Recommenders, Reinforcement Learning

Anomaly Detection

Anomaly detection is a type of unsupervised learning. It identifies unusual data points in a dataset that consists mostly of normal examples. The algorithm learns the pattern of normal data and flags any example that may be an anomaly.



Anomaly detection example

how often log in?
how many web pages visited?
transactions?
posts? typing speed?

Fraud detection:

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

perform additional checks to identify real fraud vs. false alarms

Manufacturing:
$x^{(i)}$= features of product i

airplane engine
circuit board
smartphone

ratios

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

For anomaly detection, we use the Gaussian distribution, which is defined by two parameters: the mean (μ) and the variance (σ²). The mean determines the center of the distribution, while the standard deviation (σ) controls the width of the curve.

If a data point has a high probability density under this curve, it is considered normal; otherwise, it is considered an anomaly.

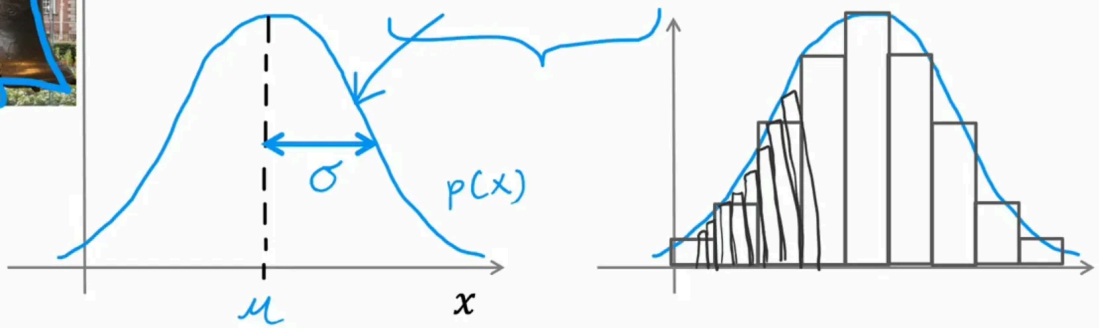
Gaussian (Normal) distribution

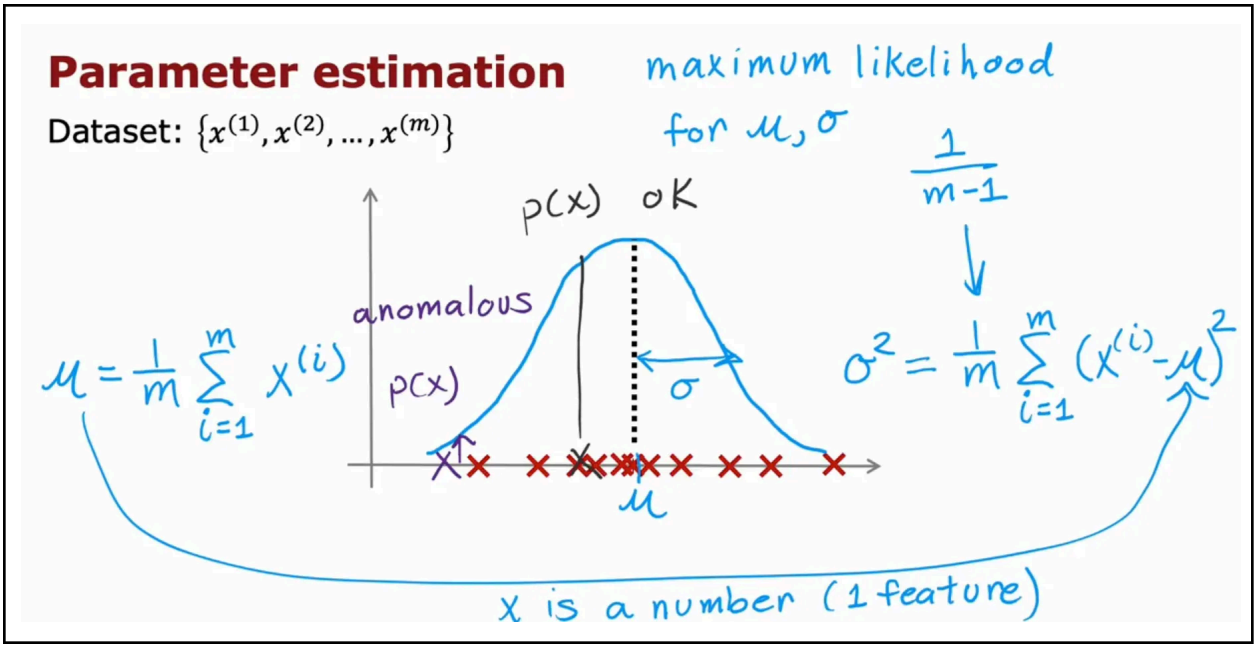
σ standard deviation
σ^2 variance

Say x is a number.
Probability of x is determined by a Gaussian with mean μ, variance σ^2.

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

$\pi = 3.14$

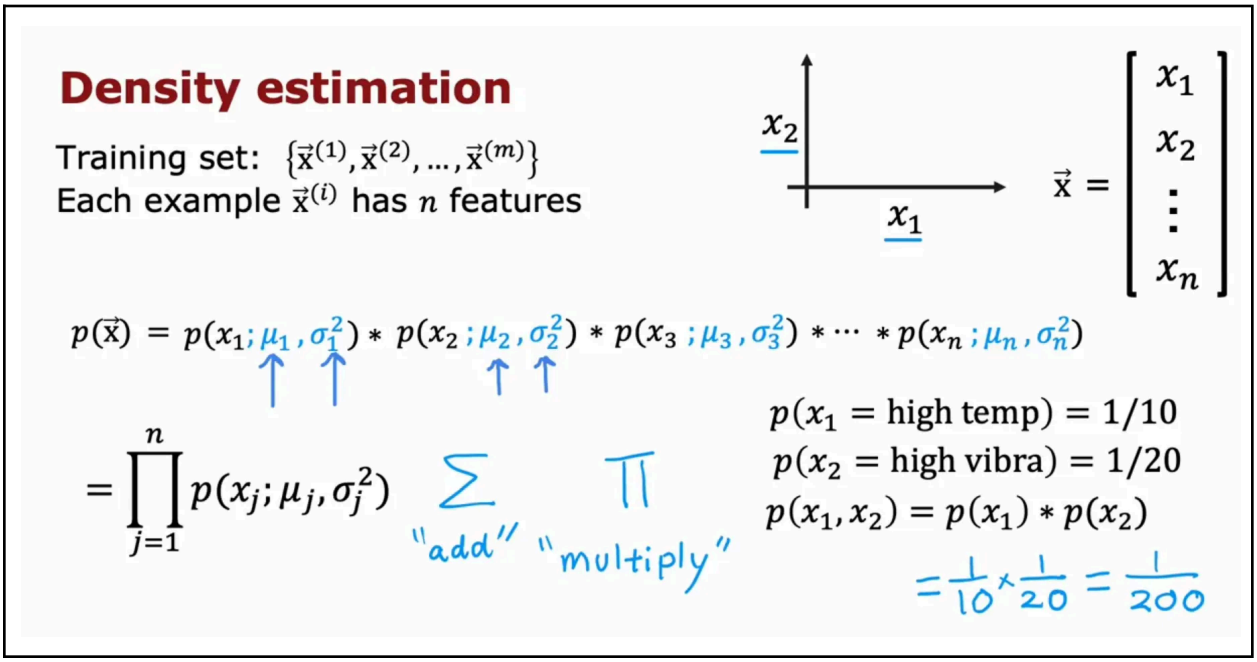




Now, how do we apply the Gaussian distribution to datasets with multiple features?

For each feature, we calculate its mean (μ) and variance (σ²). The total probability of an example x, denoted as p(x), is computed as the product of the individual probabilities of each feature. If this total probability falls below a threshold ε, the example is flagged as an anomaly.

Since the total probability is a product of individual probabilities, even a single feature with a low probability can significantly reduce p(x), allowing the model to detect anomalies effectively.



Anomaly detection algorithm

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

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

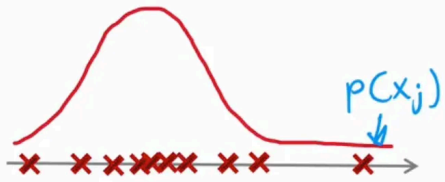
Vectorized formula

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

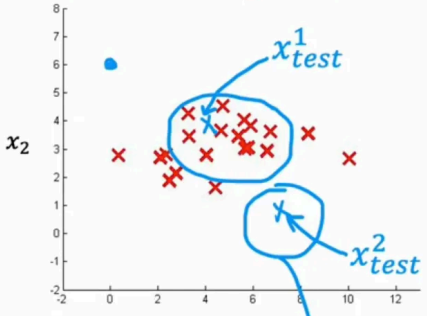
3. Given new example x, compute $p(x)$:

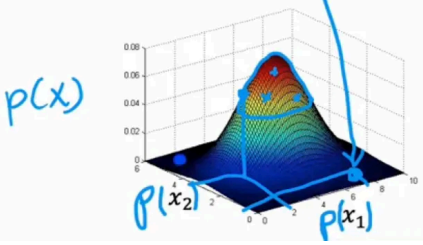
$$p(x) = \prod_{j=1}^{n} p(x_j; \mu_j, \sigma_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2})$$

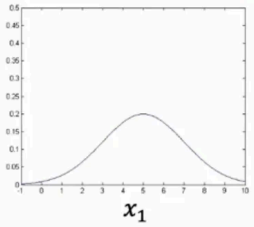
Anomaly if $p(x) < \varepsilon$

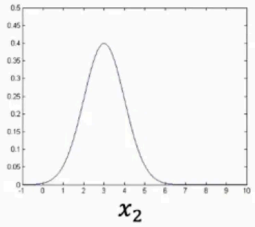


Anomaly detection example









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

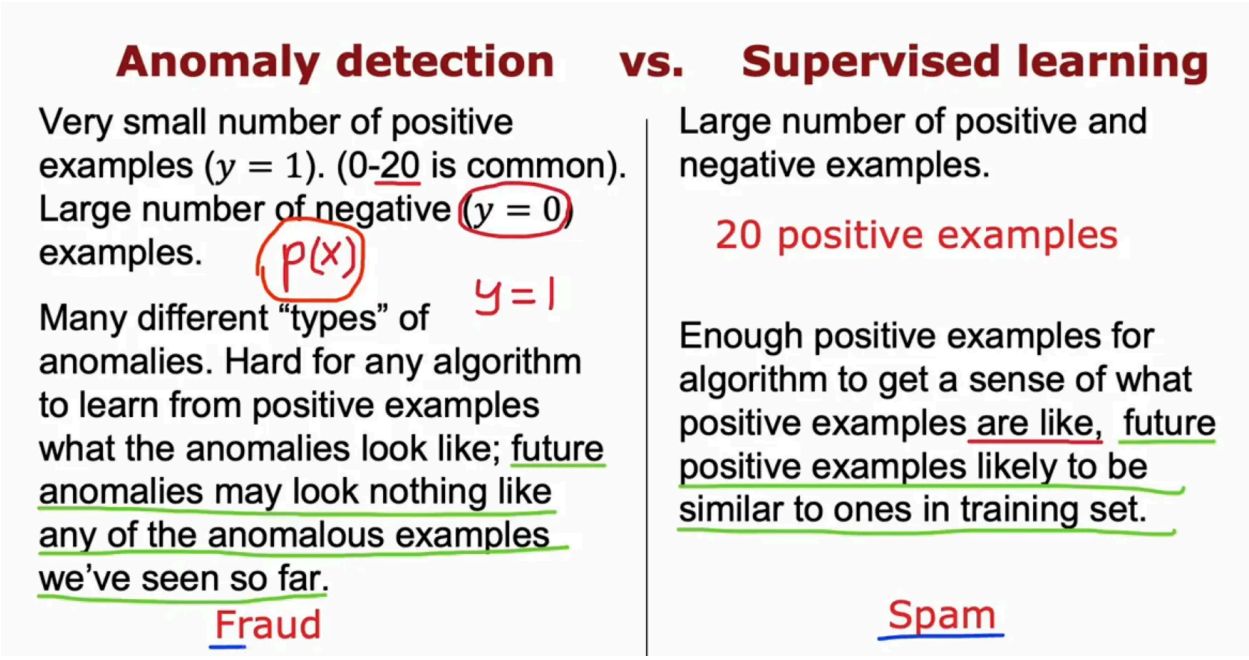
$p(x_1; \mu_1, \sigma_1^2)$ $\qquad$ $p(x_2; \mu_2, \sigma_2^2)$

$\varepsilon = 0.02$

$p\left(x_{test}^{(1)}\right) = 0.0426 \longrightarrow$ "ok"

$p\left(x_{test}^{(2)}\right) = 0.0021 \longrightarrow$ anomaly

Not only for anomaly detection algorithms but for any model development, it's important and helpful to evaluate the algorithm during development. This allows you to notice the impact of adjustments to features and parameters.



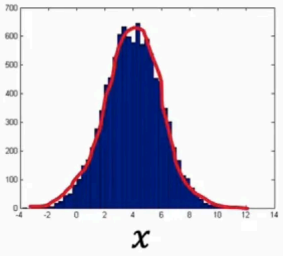
Feature selection is very important when building an anomaly detection model. This is because anomaly detection relies on unlabeled data, and unhelpful features make it harder to detect anomalies.

Anomaly detection algorithms often assume that features are normally (Gaussian) distributed, so it's important to transform non-Gaussian features into Gaussian ones. This can be done using transformations like logarithms, square roots, or exponents.

Another helpful approach is to create new features, such as combinations or ratios of existing ones, to better capture anomalous patterns.

Overall, feature engineering and error analysis are essential steps for preparing high-quality data that improves the performance of anomaly detection systems.

Non-gaussian features



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

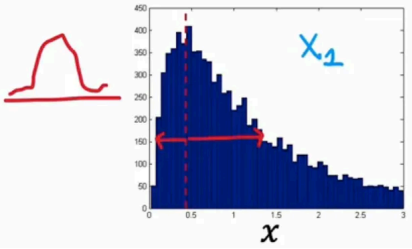
`plt.hist(x)`

$x_1 \leftarrow \log(x_1)$

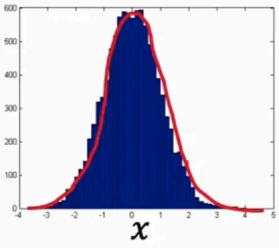
$x_2 \leftarrow \log(x_2 + 1)$ $\log(x_2 + c)$

$x_3 \leftarrow \sqrt{x_3} = x_3^{1/2}$

$x_4 \leftarrow x_4^{1/3}$



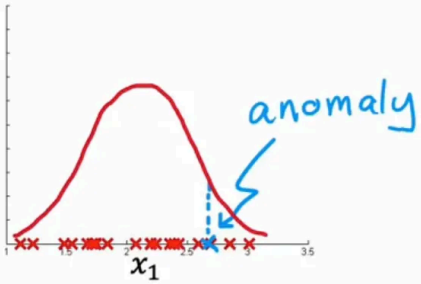
`np.log(x)`

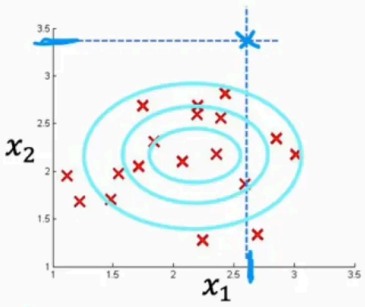


Error analysis for anomaly detection

Want $p(x) \geq \varepsilon$ large for normal examples x.
$p(x) < \varepsilon$ small for anomalous examples x.

Most common problem:
$p(x)$ is comparable for normal and anomalous examples.
($p(x)$ is large for both)





x_1 num transactions x_2 typing speed

Recommender Systems — Collaborative Filtering