⇒ Week-9

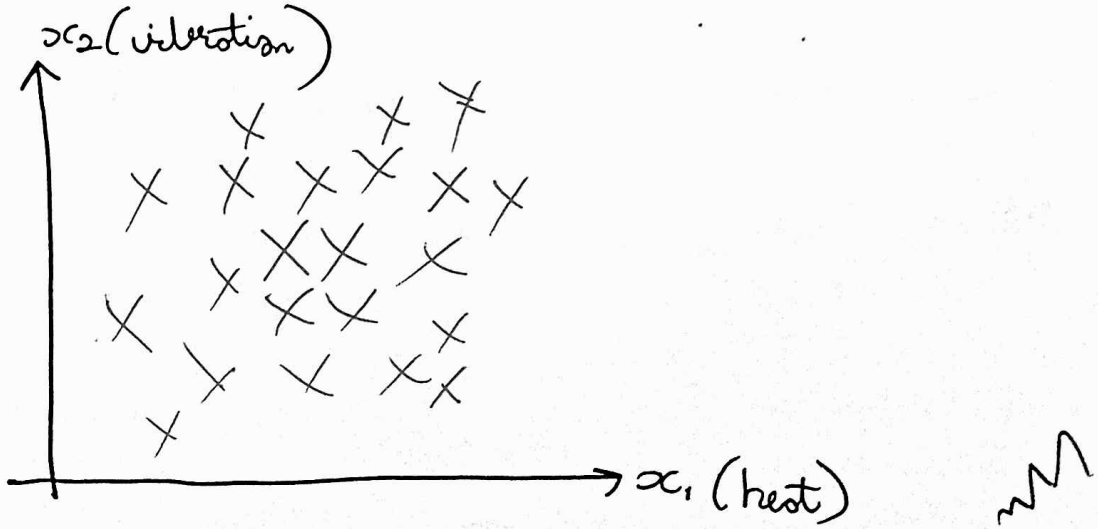
⇒ Anomaly detection : Problem motivation

⇒ What is anomaly detection ?

Anomaly detection refers to identification of items or events that do not conform to an expected pattern or to other items in a dataset.

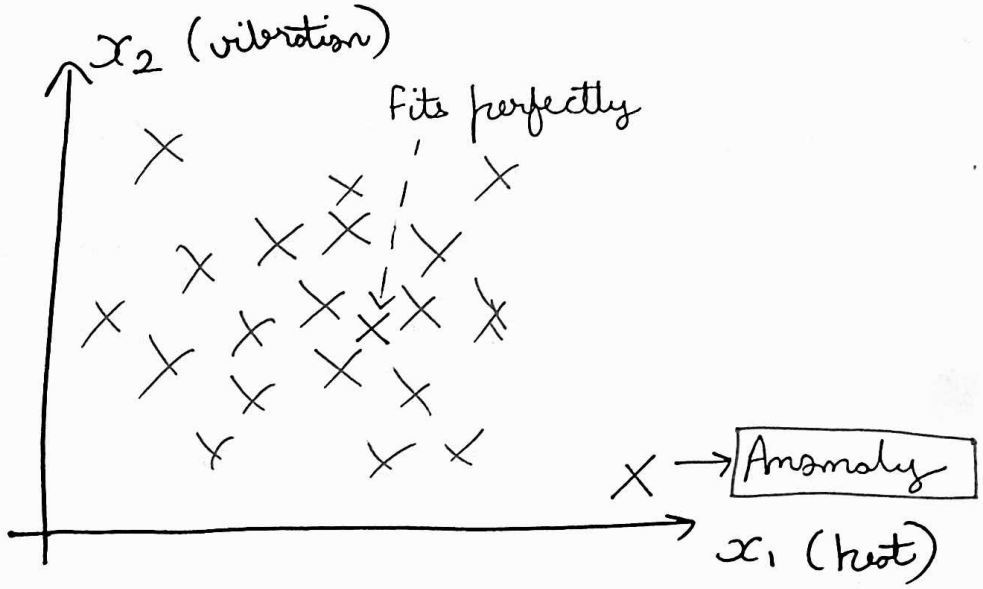
eg : Suppn. we're an aircraft manufacturing company & have a dataset ($x^{(1)}$ to $x^{(m)}$) consisting of 2 features & examples: x_1 (heat) & x_2 (vibration)

Suppn. our dataset looks like this :



→ Next day you have 2 new aircraft engines to test. When plotted the with the full dataset looks as follows:

(Note the new entries have been marked with green)



→ The anomaly detection method will see if the new engine is anomalous or not (when compared to the prev. engines)

For the above example, it'll detect the anomalous aircraft & send it for futher testing.

⇒ Density estimation

→ In probability & statistics, ~~density~~ density estimation is the construction of an estimate, based on observed data, of an unobservable underlying probability density function (PDF).

The unobservable PDF is thought of as the density according to which a large population is distributed.

→ Density estimation is also frequently used in anomaly detection or novelty ~~detect~~ detection.

If an observation lies in a very low-density region, it is likely to be an anomaly or a novelty.

→ Given a dataset : $\{x^1, x^2, \ldots, x^m\}$ s.t. the dataset has non-anomalous data.

→ Given a new example x^{test}. We check if its anomalous or not.

→ How do we do it?

Using our training dataset, we build a model. We can access this model using $p(x)$.

* This $p(x)$ is not a probability but rather the normalized probability density function as parametrized by the feature vector x

♫ Determination of the actually probability can be done by integrating $p(x)$ over some range.

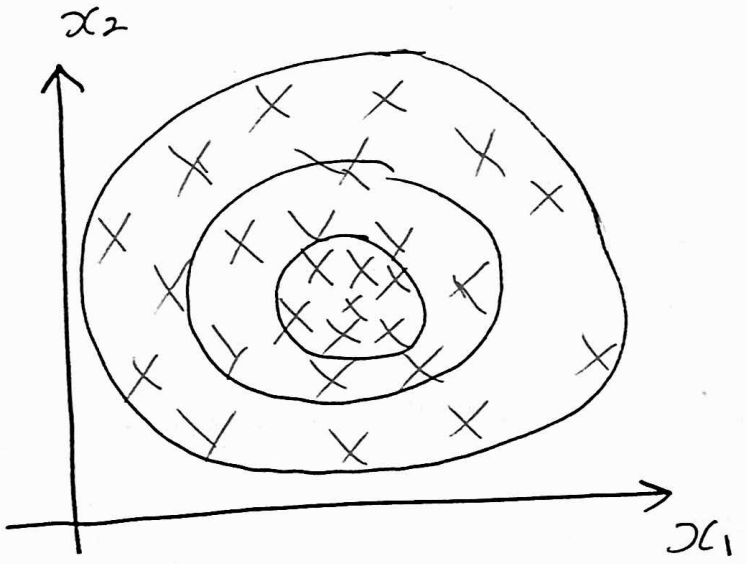
* $p(x)$ indirectly asks "What is the probability that the example x is non anomalous?"

→ Once model is built,

* if $p(x^{test}) < \epsilon$ } flag this as an anomaly
* if $p(x^{test}) \geq \epsilon$ } flag this as not anomalous

⇒ ϵ is a threshold probability value which we define depending on how sure we need / want to be

→ For our aircraft example, we want our model to (graphically) look like this:



(The inner circles have more probability of getting characterized as anomalous.)

⇒ Applications of anomaly detection

→ Fraud detection

* x^i = features of 'i'-th user's activities

eg of features : Typing speed, No. of webpage/transactions
Still time

* Model $p(x)$ from data

* Identify unusual users by checking which have $p(x) < \epsilon$.

→ Manufacturing : Discused in aircraft example

→ Monitoring computers in data center

* x^i → Memory use, no. of diskaccesses/sec, CPU load, Network traffic

* Identify crashing system by checking $p(x) < \epsilon$

→ If our anomaly detector is flagging too many anomalous examples, then we need to decrease our threshold ϵ.

⇒ Gaussian distribution (or Normal distribution)

→ The Gaussian distribution is a familiar bell shaped curve that be described by the function $\mathcal{N}(\mu, \sigma^2)$

→ Formally,

Say $X \in \mathbb{R}^m$ ($x^i \in \mathbb{R}$). If X → dataset is a distributed Gaussian with mean μ, variance σ^2, then:

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

* The little $\sim$ or 'tilde' can be read as "distributed as".

* The Gaussian distribution is completely parametrized by a mean (μ) → describes the center of the curve & standard deviation(σ) → describes the width of the curve:

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

→ Why standard deviation represents width?

→ Standard deviation (σ) is a measure of variability: How spread out are the values?

→ While a measure of central tendency (like mean, median) describe the "typical" value of the dataset, measures of variability define how far away the data points tend to fall from the center.

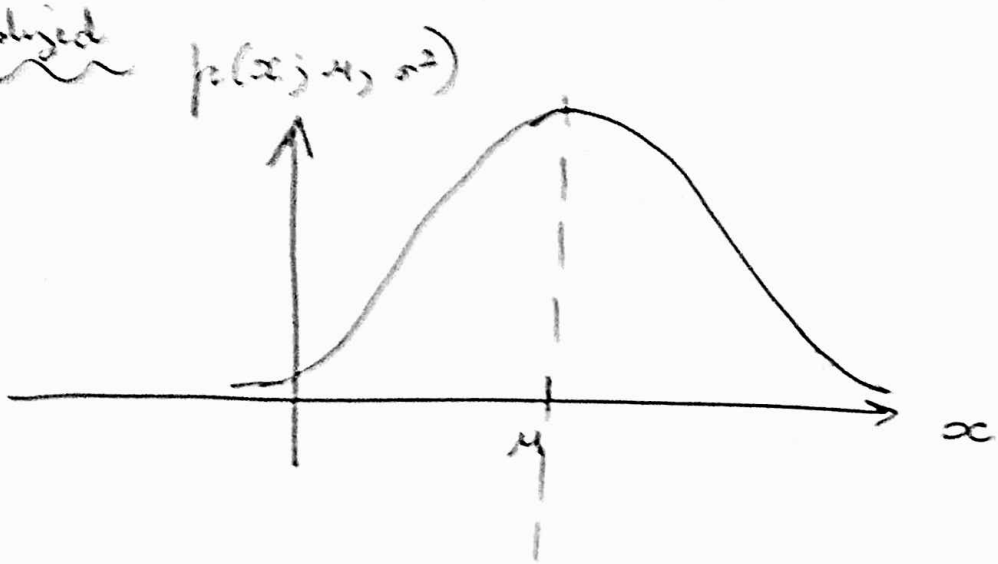
→ A low value of σ denotes data points being clustered tightly around the center (at mean μ in our case). A high value signifies that they tend to fall further away.

So,

→ We can see how low σ causes less width of the distribution and a high σ causes the distribution to be more wide.

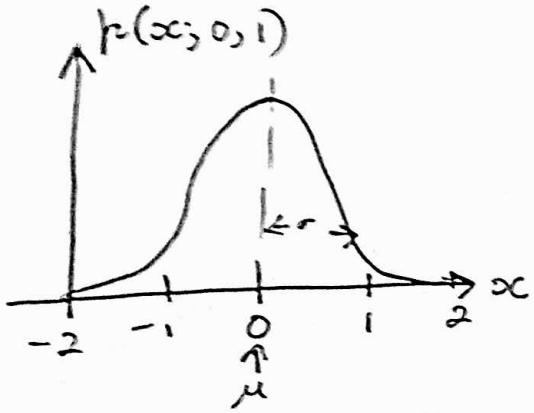
⇒ Gaussian distribution examples

→ Generalized

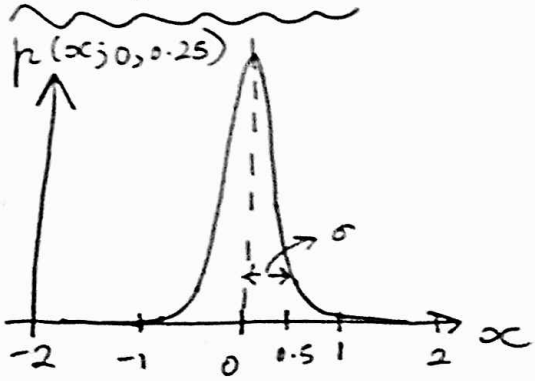


* Area is always equal to 1 but width & crest can be changed by playing with values of σ & μ respectively.

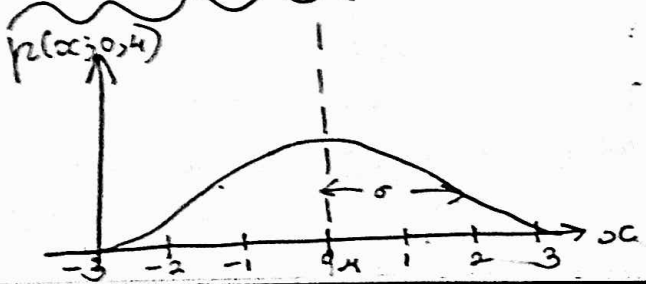
→ μ = 0, σ = 1



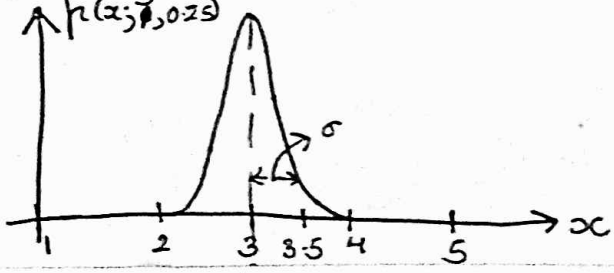
→ μ = 0, σ = 0.5



→ μ = 0, σ = 2



→ μ = 3, σ = 0.5



⇒ Parameter estimation

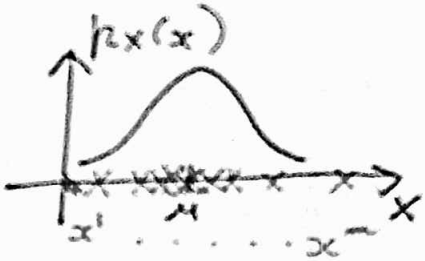
* Modelling our dataset in terms of a probabilistic model can help us in calculating probabilities for each example.

* For anomaly detection, using Gaussian distribution makes sense, intuitively. because of its inherent structure & properties. eg: You can visually tell the points which are closer to the "typical" or average value

→ Problem: Given a dataset $\{x^1, x^2, \ldots, x^m\}$ $x^i \in \mathbb{R}$

Can you estimate the distribution

→ Could be something like this:



* Seems like a reasonable fit - data seems likes a higher probability of being in the central region, lower probability of being further away

→ The Gaussian distribution can be completely described by μ & σ^2 & they can be calculated as follows:

$$\mu \equiv \text{Average of examples} = \frac{1}{m} \sum_{i=1}^{m} x^i$$

$$\sigma^2 \equiv \left(\text{Standard deviation}\right)^2 = \text{Variance} = \frac{1}{m} \sum_{i=1}^{m} (x^i - \mu)^2$$

~~* These values μ & σ^2 are~~

* These values are the maximum likelihood estimations (MLE) for μ & σ^2.

* We can also do $1/(m-1)$ instead of $1/m$ which is used in statistics community, but in practice it makes hardly any difference.

⇒ Density estimation algorithm

→ Before formally stating the anomaly detection algorithm, we need to estimate the probability density of our dataset.

→ Given a training set X : $\{x^1, \ldots, x^m\}$; where $x^i \in \mathbb{R}^n$

* We model each of the features by assuming each feature is distributed according to a Gaussian distribution; s.t.

$$X_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$$
$$X_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$$
$$\vdots$$
$$X_n \sim \mathcal{N}(\mu_n, \sigma_n^2)$$

s.t. $p_{X_i}(x_i; \mu_i, \sigma_i^2)$ → PDF of feature x_i, given μ_i & σ_i^2 using a Gaussian distribution.

→ We can assume that each feature is statistically independent from each other. (If if they aren't, just use a dimensionality reduction technique like PCA to get independent features)

→ Since our features are independent, we can formulate the joint PDF of these features as the product of the individual PDFs.

$$p_{X_1,\ldots,X_m}(x_1,\ldots,x_m) = p_{X_1}(x_1;\mu_1,\sigma_1^2)\cdot\ldots\cdot p_{X_m}(x_m;\mu_m,\sigma_m^2)$$

→ ~~Rewrite formula above~~

$$\therefore\ p_{X_1,\ldots X_m}(x_1,\ldots,x_m) = \prod_{j=1}^{m} p_{X_j}(x_j;\mu_j,\sigma_j^2)$$

⇒ Anomaly detection algorithms

1) Choose features x_i that you think might be indicative of anomalous examples.

2) Fit (or calculate) parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$

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

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

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

3) Given new example x, calculate $p(x)$

$$p(x) \equiv p_{x_1, \ldots, x_n}(x_1, \ldots, x_n) = \prod_{j=1}^{n} p_{x_j}(x_j ; \mu_j, \sigma_j^2)$$

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

Declare anomaly if $p(x) < \epsilon$

⇒ Developing & evaluating an anomaly detection system

→ When developing a learning algo, making decisions (like choosing which ^features^ to include, etc) becomes ~~easier~~ easier if we have a way of evaluating our learning algorithm.

→ In case of anomaly detection, we take some labelled data, categorized into ~~non~~ anomalous & non anomalous examples ($y=0$ if ⊗ non-anomalous, $y=1$ if anomalous)

* Among that, take a large proportion of good, non-anomalous data for the training set on which to train $p(x)$.

Training set : $x^1, x^2, \ldots, x^m$ (assume non anomalous)

→ Then take a smaller proportion of missed anomalous & non-anomalous examples for your cross validation & test sets.

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

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

→ eg: Aircraft engines example

Given data: { 10 000 good (non-anomalous) engines ($y=0$); ≈ 20 flawed (anomalous) engines ($y=1$) }

Training set	CV set	Test set
6000 good engines ($y=0$)	2000 good engines ($y=0$) 10 anomalous ($y=1$)	2000 good engines ($y=0$) 10 anomalous ($y=1$)

→ In Summary:

* Split the data 60 : 20 : 20 :: Training set : CV set : test set
* Split the anomalous examples 50 : 50 :: CV set : test set

⇒ Algorithm evaluation workflow

1) Fit model $p_x(x)$ on training set $X = \{x^1, \ldots, x^m\}$

2) On a cross validation/test example x, predict

$$y = \begin{cases} 1, & p_x(x) < \epsilon \Rightarrow \text{anomaly} \\ 0, & p_x(x) \geq \epsilon \Rightarrow \text{non-anomalous} \end{cases}$$

3) Possible evaluation metrics: Since the no. of non-anomalous examples will always be much higher than the anomalous ones (or the classes are skewed), the alg that predicts $y = 0$ will always have higher accuracy if classification accuracy is used. The foll. metrics will give better results:
   * True +ve, False +ve, True -ve, False -ve
   * Precision / Recall
   * F_1 score

→ For taking important decisions like choosing ϵ, which features to drop, use the cross validation set for that is choose the ~~configaw~~ configuration which minimizes the cross validation error.

⇒ Anomaly detection vs Supervised learning

* ~~*~~ Since, we ~~be~~ have ~~a~~ labelled data with ourselves, with examples that we know are either anomalous or non-anomalous, why don't we use a supervised learning technique like SVM, logistic regression or neural nets to directly learn from our data in predicting whether $Y=0$ or $Y=1$?

Anomaly detection

→ Use this when you have a very small no. of +ve examples ($y=1$) (0-20 is common)

→ Reason: There can exist "different" many different "types" of anomalies. Our low no. of labels for anomalous examples in our dataset doesn't help either!

It'd be hard for a supervised algo to predict a label for some anomaly because a future anomaly might look completely diff. than our example anomalies

Supervised learning

→ Use this when you have large no. of both +ve & -ve examples

→ Reason: We have enough examples +ve (or anomaly) examples for our algo to generalize for the every type of +ve examples thrown at it in the future.

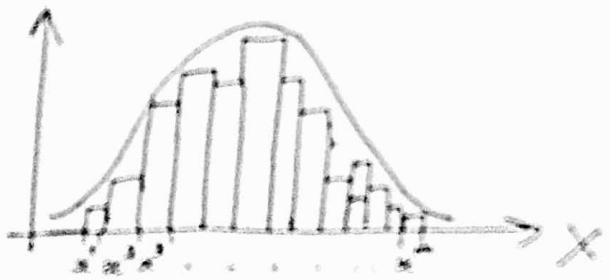
→ Usecases of supervised learning: Email Spam classification, Weather prediction (sunny/rainy/etc), Cancer classification

→ Usecases of anomaly detection: Fraud detection, Manufacturing (eg: Aircraft engines), Monitoring machines

⇒ Choosing what features to use

→ The features you choose to use / include in your model will greatly impact how well your anomaly detection algorithm works.

→* We can check whether our features are "Gaussian" by plotting a histogram of our data & check for the bell shaped curve.



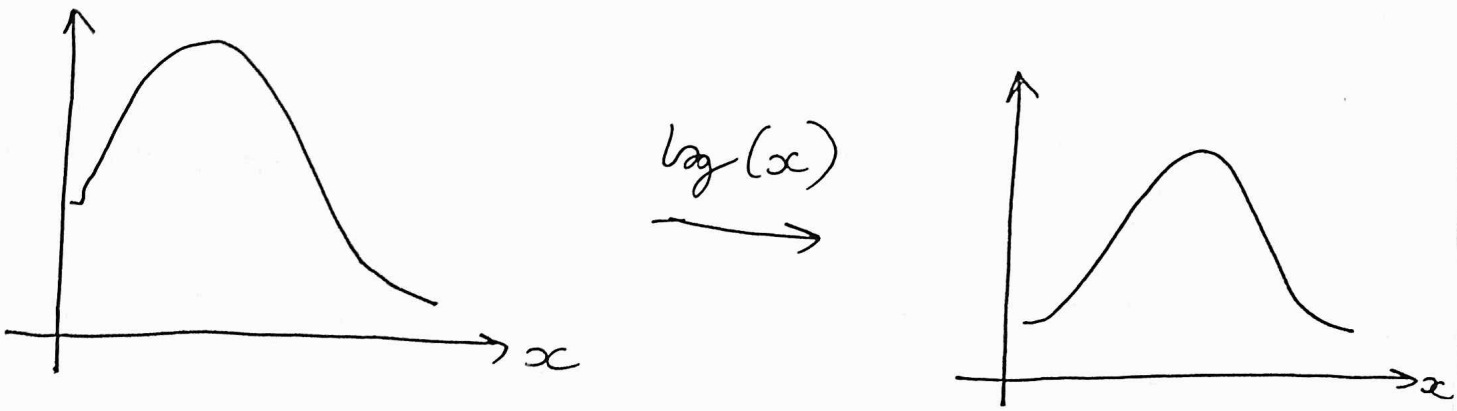
* You can decreasing the no. of bins for a more "finer" visualization.

=/ Non Gaussian features

⇒ We can apply some "transforms" on a example feature x that does not have the bell shaped curve:

* $\log(x + c)$; $c \rightarrow$ constant
* $\sqrt{x}$
* $x^{1/3}$

eg: For a left skewed distribution:



⇒ Error analysis for anomaly detection

→ Requirement:

* Want $p(x)$ to be large for non-anomalous examples x

* " " " " small " anomalous examples x

→ A common problem:

After training $p(x)$ might come out to be ~~comparable~~ relatively similar both both anomalous & non-anomalous examples alike. ~~Error~~ (Very similar to what happens in underfitting!)

* In this case, ~~choose~~ examine the anomalous examples ~~&~~ that are giving a "higher" probability & try to figure out new features that will better distinguish the data.

→ eg: Monitoring computers in a data center

Suppl. our features are as follows:

x_1 =	memory use
x_2 =	no. of (disc accesses)/sec
x_3 =	CPU load
x_4 =	network traffic

Say, our server in stuck in an infinite loop, so our CPU load will shoot up but our network traffic is still low.

* Therefore $p(x_3)$ → denoting CPU load decreases, but $p(x_4)$ → denoting network traffic is at a higher value.

* As for calculating the final probability we'll multiply each probability ($p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3)p(x_4)$)

So the higher val. of $p(x_4)$ brings down the lower val. of $p(x_3)$ thus misclassifying the example as non-anomalous!

→ There exists 2 sol^n's for this :

1) Choose a feature that highlights the unusually high CPU load & addresses the low network traffic at the same time

eg : $x_5 = \frac{\text{CPU load}}{\text{Network traffic}}$; $x_6 = \frac{(\text{CPU load})^2}{\text{Network traffic}}$

2) Use a multivariate Gaussian distribution [Coming up!]

→ In summary, choose / create features that might take on unusually large OR small values in the event of an anomaly.