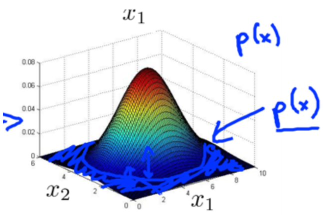
Anomaly Detection:
Anomaly detection is a derivation learning algorithm from the Gaussian Distribution.

The nature of this algorithm is to detect datasets that deviate from the normal/average case of a dataset, in exceptional ways, or in rare ways that are unlikely to repeat in a predictable pattern.

The model: represented by P(x)



In this image above the variable x above is assumed to have n = 2 features.
Assume $x \in \mathbb{R}\ if\ x\ is\ a\ distributed\ Gaussian\ with\ mean\ \mu\ and\ var\ \sigma^2$

$$More\ formally: x \sim N(\mu, \sigma^2)$$

The Gaussian Distribution model goes as follows:

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}, \sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)} - \mu\right)^2$$

Then to calculate the probability frequency of a new example x_{test}:

$$P(x_{test}; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

This calculates the probability of that particular value x with one feature as compared to the distribution over the entire set of the model. The threshold value ε then determines whether the example is to be considered anomalous or not.

$$P(x_{test}) < \varepsilon \rightarrow Flag\ anomaly$$
$$P(x_{test}) \geq \varepsilon \rightarrow normal$$

Now, we want to extend this to a model with multiple features $n \in \mathbb{R}^n$
Assuming independence of each feature with respect to each other, we have:
For 2 variables:

$$P(x) = P(x_1; \mu_1, \sigma_1^2) * P(x_2; \mu_2, \sigma_2^2)$$

For n variables:

$$P(x_{test}) = \prod_{j=1}^{n} P\left(x_j; \mu_j, \sigma_j^2\right) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{\left(x_j-\mu_j\right)^2}{2\sigma_j^2}}$$

Implementation notes:
When training a anomaly detection model, we assume that the overwhelming majority of data in the train dataset is an average and normal piece of data. (One or 2 anomalies in the dataset will not impact the detection significantly.)

Now, to choose a $\varepsilon\ value$:
Iterate through a range of possible ε values
Train a model P(x)
On a labeled set: (x_{cv}, y_{cv}) which has a number of known anomalies. Apply a prediction
Then apply the evaluation of F score to select the optimal model.

Anomaly Detection	Supervised Learning
➔ Very small number of positive examples ➔ Large number of negative examples	➔ Large number of positive and negative examples
➔ Many different unique or very rare kinds of anomalies. Hard for any algorithm to learn from positive examples ➔ Future Examples may look nothing like anomalies seen so far.	➔ Enough positive examples for algorithm to pattern recognize what positive examples are like. ➔ Future Positive Examples are likely to be similar to the train set
Applications:	
Fraud Detection (If enough positive examples can be put into supervised) Manufacturing Monitoring Machines in a Data Center	Email Spam Classification Weather Prediction Cancer Classification

Choosing Features for anomaly detection:
Plot the raw dataset's features
Ideally it should follow a Gaussian Distribution.
Perform Data transformation if data is not very gaussian

- ➔ Log Transformation
- ➔ Squareroot of the data
- ➔ x^n

Error Analysis:
Often, current features may be such that P(x) is comparable for anomalous and normal examples.
These instances should be evaluated and new features can be made that detect that kind of anomaly.
Eg:
Computers in a datacenter:

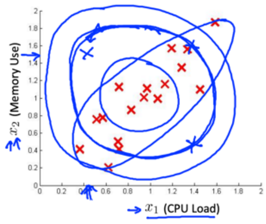
$$x_1 = memory\ usage\ of\ computer$$
$$x_2 = number\ of\ disk\ accesses$$
$$x_3 = CPU\ Load$$
$$x_4 = network\ traffic$$

We can use these features to comeup with new features:

$$x_5 = \frac{CPU\ Load}{Network\ Traffic}$$
$$x_6 = \frac{(CPU\ Load)^2}{Network\ Traffic}$$

Such that if one value is abnormally high the value here would be abnormally low/high.

Extension to Multi-Variate Gaussian Distribution:
The model above assumes that data is not highly corelated.



However we may end up with a scenario like this where the anomaly model views the blue crosses as non-anomalous when in reality it is anomalous.
This can be resolved manually by creating features that address this correlation.
Alternatively, this can be resolved using Multi-Variate Gaussian Distribution.

- ⇨ Model p(x) all at once.
- ⇨ Parameters: $\Sigma \in \mathbb{R}^{n*n} (Covariance\ matrix), \mu \in \mathbb{R}^{n}$
- ⇨ P(x; $\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}, |\Sigma| = determinant\ of\ \Sigma$
- ⇨ However must be such that: m>n so that Matrix is invertible.

However:
Computationally more expensive than Original Model!