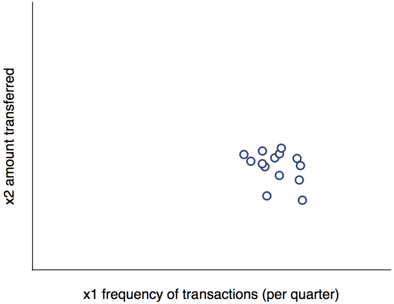
Anomaly Detection

Lecture 12 - DAMLF | ML1

Frankfurt School

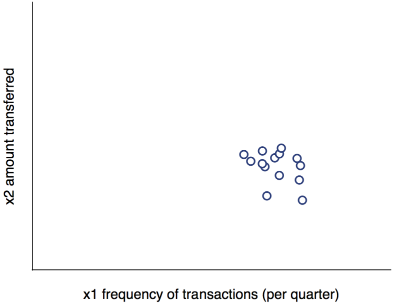
Fraud Detection

Anomaly Detection



Example: Money Laundering

Anomaly Detection



Example: Money Laundering

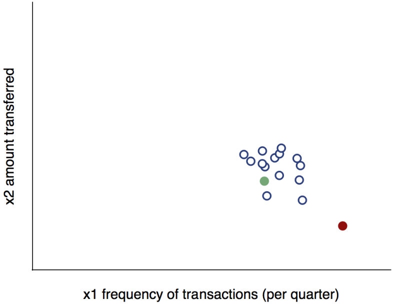
Data set: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Account features:

- x_1: Frequency of transactions (per quarter)
- x_2 Amount transferred

$\vdots$

Anomaly Detection



Example: Money Laundering

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

Account features:

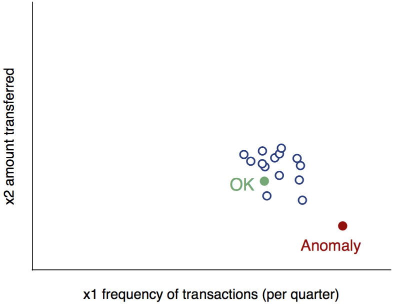
- x_1: Frequency of transactions (per quarter)
- x_2 Amount transferred

$\vdots$

A new account: x_{new}

- Is x_{new} **OK** or an **Anomaly**?

Anomaly Detection



Example: Money Laundering

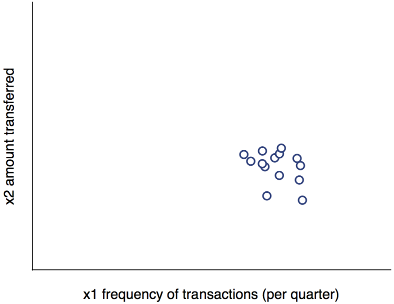
Data set: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Account features:
- x_1: Frequency of transactions (per quarter)
- x_2 Amount transferred

$\vdots$

A new account: x_{new}
- Is x_{new} **OK** or an **Anomaly**?

Density Estimation

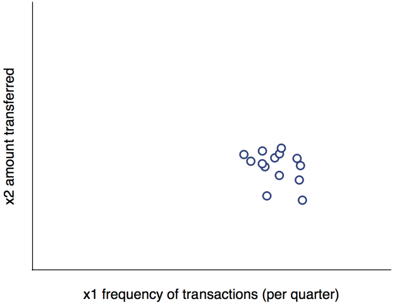


Problem Formulation

Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ of *normal* transactions.

Goal: Determine whether x_{new} is **anomalous**.

Density Estimation



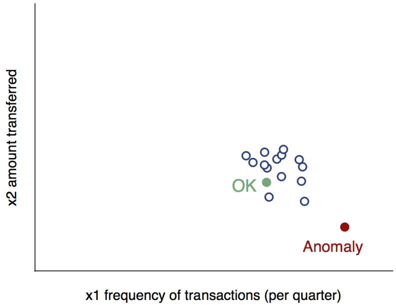
Problem Formulation

Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ of *normal* transactions.

Idea: Build a model for the probability of x

$$p(x)$$

Density Estimation



Problem Formulation

Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ of *normal* transactions.

Idea: Build a model for the probability of x

$$p(x)$$

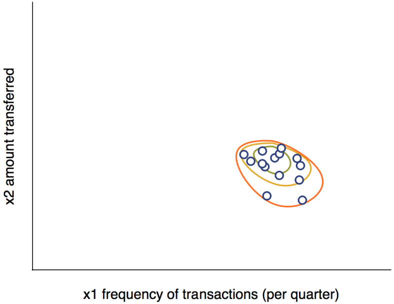
Then, for some threshold ϵ and $x_{\text{new}} \notin$ **Data**:

$$p(x_{\text{new}}) < \epsilon \;\Rightarrow\; \text{mark as an Anomaly}$$

otherwise,

$$p(x_{\text{new}}) \geqslant \epsilon \;\Rightarrow\; \text{mark as an OK}$$

Density Estimation



Problem Formulation

Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ of *normal* transactions.

Idea: Build a model for the probability of x

$$p(x)$$

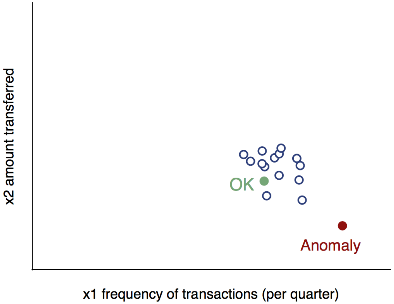
Then, for some threshold ϵ and $x_{\text{new}} \notin$ **Data**:

$$p(x_{\text{new}}) < \epsilon \;\Rightarrow\; \text{mark as an Anomaly}$$

otherwise,

$$p(x_{\text{new}}) \geqslant \epsilon \;\Rightarrow\; \text{mark as an OK}$$

Anomaly Detection

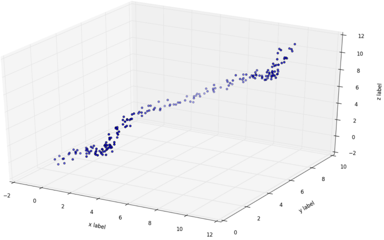


Examples

Fraud detection:

- $x^{(i)}$ = feature vector of user i's activities
- Model $p(x)$ from data
- Identify unusual activity by evaluating $p(x_{\text{new}}) < \epsilon$

Anomaly Detection



Examples

Fraud detection:

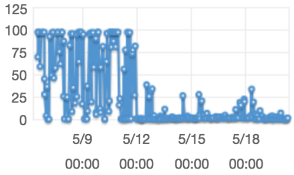
- $x^{(i)}$ = feature vector of user i's activities
- Model $p(x)$ from data
- Identify unusual activity by evaluating $p(x_{\text{new}}) < \epsilon$

Manufacturing

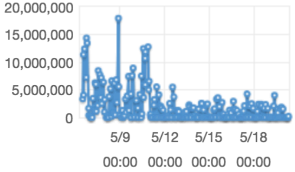
- $x^{(i)}$ = vector of measurements of widget i

Anomaly Detection

CPU Utilization (Percent)



Network Out (Bytes)



Examples

Fraud detection:

- $x^{(i)}$ = feature vector of user i's activities
- Model $p(x)$ from data
- Identify unusual activity by evaluating $p(x_{\text{new}}) < \epsilon$

Manufacturing

- $x^{(i)}$ = vector of measurements of widget i

Monitoring computers in data center

- $x^{(i)}$ = features of machine:

memory use, CPU load, disk accesses / sec, etc.

Gaussian Distribution

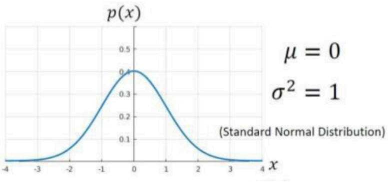
Gaussian Distribution

Gaussian (Normal) Distribution

Suppose $x \in \mathbb{R}$.

Let x be a distributed **Gaussian** with mean μ and variance σ^2.

Gaussian Distribution



Gaussian (Normal) Distribution

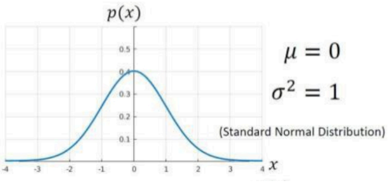
Suppose $x \in \mathbb{R}$.

Let x be a distributed **Gaussian** with mean μ and variance σ^2.

Notation:

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

Gaussian Distribution



Gaussian (Normal) Distribution

Suppose $x \in \mathbb{R}$.

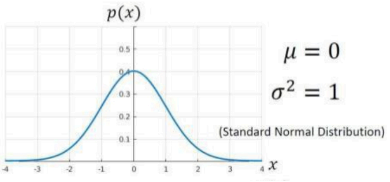
Let x be a distributed **Gaussian** with mean μ and variance σ^2.

Notation:

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

This distribution specifies the probability of x taking on different values.

Gaussian Distribution



Gaussian (Normal) Distribution

Suppose $x \in \mathbb{R}$.
Let x be a distributed **Gaussian** with mean μ and variance σ^2.
Notation:

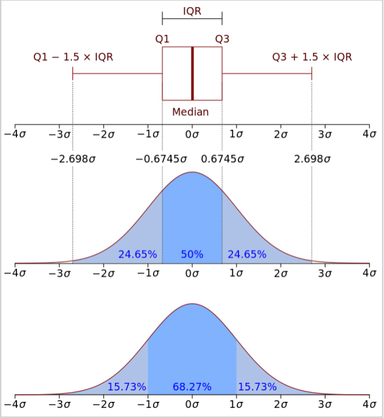
$$x \sim \mathcal{N}(\mu, \sigma^2)$$

This distribution specifies the probability of x taking on different values.

We can write the probability of x parameterized by μ and σ^2 from the normal distribution as

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

Gaussian Distribution



Boxplot and probability density function of a normal distribution $N(0, \sigma^2)$.

Source: Wikipeadia

Gaussian (Normal) Distribution

Suppose $x \in \mathbb{R}$.

Let x be a distributed **Gaussian** with mean μ and variance σ^2.

Notation:

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

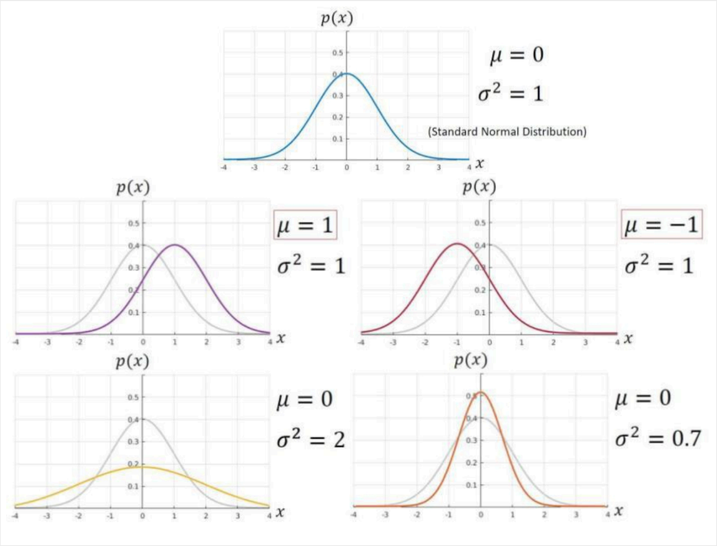
This distribution specifies the probability of x taking on different values.

We can write the probability of x parameterized by μ and σ^2 from the normal distribution as

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

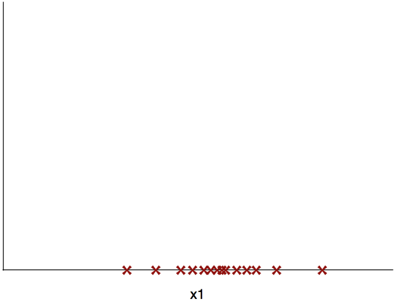
The probability that x will fall within **one standard deviation** σ from the mean μ is approximately .6827.

Gaussian Distribution



Above are several examples of 1D Gaussian Distribution illustrating how the adjusting the mean and variance will shift and stretch/compress the distribution. Credit: Dan Lee

Parameter Estimation

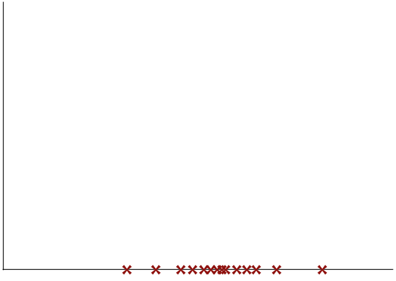


Parameter Estimation Problem

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

Imagine that you suspect that your m data points came from a Gaussian distribution.

Parameter Estimation



Parameter Estimation Problem

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

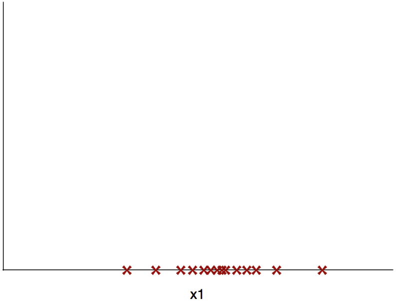
Imagine that you suspect that your m data points came from a Gaussian distribution.

That is, you suspect that (for i: 1 to m)

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

for **some** μ and σ^2.

Parameter Estimation



Parameter Estimation Problem

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

Imagine that you suspect that your m data points came from a Gaussian distribution.

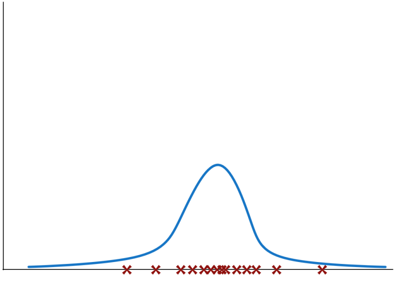
That is, you suspect that (for i : 1 to m)

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

for **some** μ and σ^2.

The **problem of parameter estimation** is that, given a data set and the *the assumption* that it is distributed according to some distribution (e.g., Gaussian), what are the parameters of the *specific* distribution that the data came from?

Parameter Estimation



Parameter Estimation Problem

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

Imagine that you suspect that your m data points came from a Gaussian distribution.

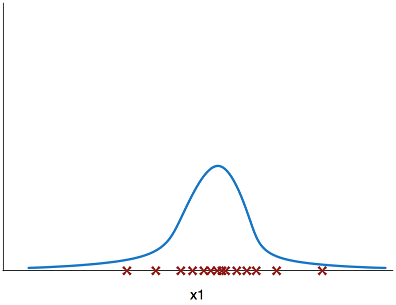
That is, you suspect that (for i : 1 to m)

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

for **some** μ and σ^2.

The **problem of parameter estimation** is that, given a data set and the *the assumption* that it is distributed according to some distribution (e.g., Gaussian), what are the parameters of the *specific* distribution that the data came from?

Parameter Estimation



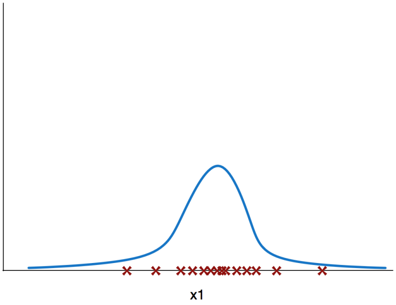
Parameter Estimation Problem

Formulas for estimating μ and σ^2

Estimating μ by $\hat{\mu}$:

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

Parameter Estimation



Parameter Estimation Problem

Formulas for estimating μ and σ^2

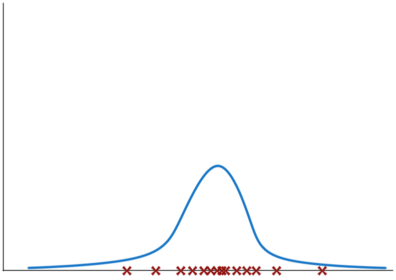
Estimating μ by $\hat{\mu}$:

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

Estimating σ^2 by $\hat{\sigma}^2$:

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

Parameter Estimation



Parameter Estimation Problem

Formulas for estimating μ and σ^2

Estimating μ by $\hat{\mu}$:

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

Estimating σ^2 by $\hat{\sigma}^2$:

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

Notes: $\hat{\mu}$ and $\hat{\sigma}^2$ are the **maximum likelihood estimates** of μ and σ^2, respectively.

In statistics textbooks you will usually see $\frac{1}{m-1}$ rather than the $\frac{1}{m}$ convention in machine learning; $\frac{1}{m}$ is a **biased** estimator, but is nevertheless a common convention in ML.

Returning to Anomaly Detection

Next, let's apply the machinery of Gaussian distributions to develop an anomaly detection algorithm.

Anomaly Detection Algorithm

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1 \qquad) p(x_2 \qquad) p(x_3 \qquad) \cdots p(x_n \qquad)$$

Where:

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1 \qquad)p(x_2 \qquad)p(x_3 \qquad) \cdots p(x_n \qquad)$$

Where:

$$x_1 \sim \mathcal{N}(\hat{\mu}_1, \hat{\sigma}_1^2)$$

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1; \hat{\mu}_1, \hat{\sigma}_1^2)p(x_2 \qquad\quad)p(x_3 \qquad\quad) \cdots p(x_n \qquad\quad)$$

Where:

$$x_1 \sim \mathcal{N}(\hat{\mu}_1, \hat{\sigma}_1^2)$$

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1; \hat{\mu}_1, \hat{\sigma}_1^2)p(x_2; \hat{\mu}_2, \hat{\sigma}_2^2)p(x_3 \qquad) \cdots p(x_n \qquad)$$

Where:

$$x_1 \sim \mathcal{N}(\hat{\mu}_1, \hat{\sigma}_1^2)$$
$$x_2 \sim \mathcal{N}(\hat{\mu}_2, \hat{\sigma}_2^2)$$

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1; \hat{\mu}_1, \hat{\sigma}_1^2) p(x_2; \hat{\mu}_2, \hat{\sigma}_2^2) p(x_3; \hat{\mu}_3, \hat{\sigma}_3^2) \cdots p(x_n \qquad)$$

Where:

$$x_1 \sim \mathcal{N}(\hat{\mu}_1, \hat{\sigma}_1^2)$$
$$x_2 \sim \mathcal{N}(\hat{\mu}_2, \hat{\sigma}_2^2)$$
$$x_3 \sim \mathcal{N}(\hat{\mu}_3, \hat{\sigma}_3^2)$$

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x; \boldsymbol{\theta}) = p(x_1; \hat{\mu}_1, \hat{\sigma}_1^2)p(x_2; \hat{\mu}_2, \hat{\sigma}_2^2)p(x_3; \hat{\mu}_3, \hat{\sigma}_3^2) \cdots p(x_n; \hat{\mu}_n, \hat{\sigma}_n^2)$$

Where:

$$\begin{aligned}
x_1 &\sim \mathcal{N}(\hat{\mu}_1, \hat{\sigma}_1^2) \\
x_2 &\sim \mathcal{N}(\hat{\mu}_2, \hat{\sigma}_2^2) \\
x_3 &\sim \mathcal{N}(\hat{\mu}_3, \hat{\sigma}_3^2) \\
&\vdots \\
x_n &\sim \mathcal{N}(\hat{\mu}_n, \hat{\sigma}_n^2)
\end{aligned}$$

Density Estimation

Problem Formulation

Training Data: $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x \in \mathbb{R}^n$

Model: Let $p(x; \boldsymbol{\theta})$ be:

$$p(x;\ \hat{\mu}, \hat{\sigma}^2) = p(x_1; \hat{\mu}_1, \hat{\sigma}_1^2)p(x_2; \hat{\mu}_2, \hat{\sigma}_2^2)p(x_3; \hat{\mu}_3, \hat{\sigma}_3^2) \cdots p(x_n; \hat{\mu}_n, \hat{\sigma}_n^2)$$

Compact Notation:

$$p(x; \hat{\mu},\ \hat{\sigma}^2) = \prod_{j=1}^{n} p(x_j; \hat{\mu}_j, \hat{\sigma}_j^2)$$

Notation: $p(x; \boldsymbol{\theta})$ refers to the probability density of the random variable X at the point x, with θ being the parameter of the distribution. This is a **frequentist** convention.

When θ is a random variable, we may write $p(x \mid \theta)$. This is a **Bayesian** convention.

See (Wheeler 2018, Section 4) for a brief discussion touching on the difference between frequentist and Bayesian statistics.

Anomaly Detection Algorithm

1. Choose features x_i you think might be indicative of anomalous examples.
2. Fit parameters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2$, using the *maximum likelihood estimators* for each each feature:

$$\hat{\mu}_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)} \qquad \text{for } j = 1 \text{ to } n$$

$\hat{\mu}_j$ is the average value of the jth feature

$$\hat{\sigma}_j^2 = \frac{1}{m} \sum_{i=1}^{m} \left(x_j^{(i)} - \hat{\mu}_j \right)^2 \qquad \text{for } j = 1 \text{ to } n$$

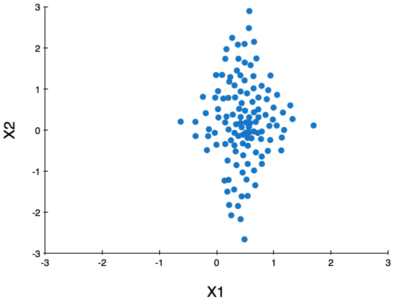
3. Given a new example x_{new}, compute $p(x_{\text{new}})$:

$$p(x_{\text{new}};\ \hat{\mu}, \hat{\sigma}^2) = \prod_{j=1}^{n} p(x_j; \hat{\mu}_j, \hat{\sigma}_j^2)$$

$$= \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi\hat{\sigma}^2}} \cdot \exp\left(-\frac{(x - \hat{\mu})^2}{2\hat{\sigma}^2} \right)$$

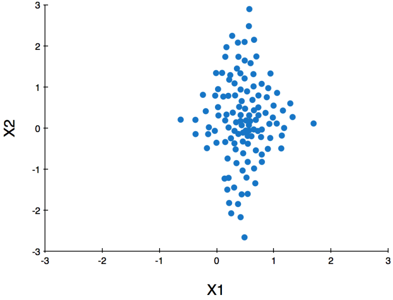
If $p(x_{\text{new}};\ \hat{\mu}, \hat{\sigma}^2) < \epsilon$, then x_{new} is marked as an anomaly.

Anomaly Detection Estimation



$\mu_1 = 0.5, \sigma_1^2 = 1$ and
$\mu_2 = 0, \sigma_2^2 = 2$

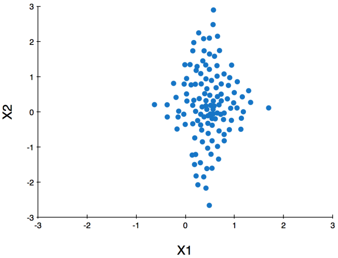
Anomaly Detection Estimation

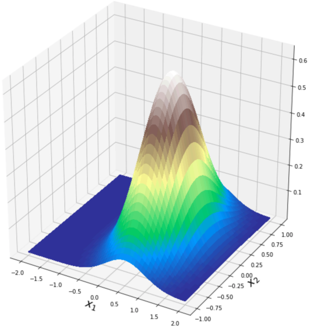


$\mu_1 = 0.5, \sigma_1^2 = 1$ and
$\mu_2 = 0, \sigma_2^2 = 2$

Plot: $p(x_1; \mu_1, \sigma_1^2) \times p(x_2; \mu_2, \sigma_2^2)$

Anomaly Detection Estimation

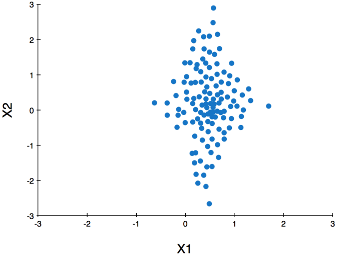


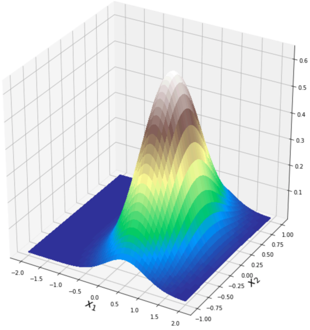


$\mu_1 = 0.5, \sigma_1^2 = 1$ and

$\mu_2 = 0, \sigma_2^2 = 2$

Anomaly Detection Estimation

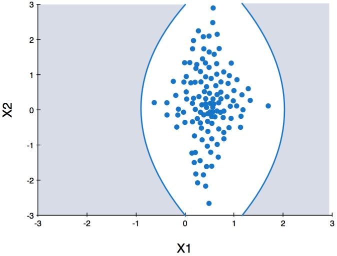


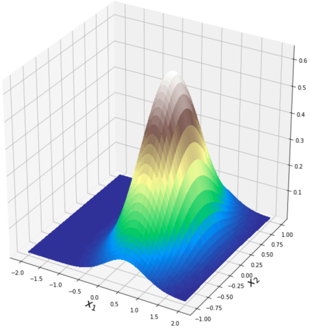


$\mu_1 = 0.5, \sigma_1^2 = 1$ and
$\mu_2 = 0, \sigma_2^2 = 2$

Plot: $p(x_1; \mu_1, \sigma_1^2) \times p(x_2; \mu_2, \sigma_2^2)$

Anomaly Detection Estimation





$\mu_1 = 0.5, \sigma_1^2 = 1$ and

$\mu_2 = 0, \sigma_2^2 = 2$

Plot: $p(x_1; \mu_1, \sigma_1^2) \times p(x_2; \mu_2, \sigma_2^2)$

Suppose $\epsilon = 0.02$

If $p(x_{\text{test}}^{(1)};\ \hat{\mu}, \hat{\sigma}^2) = 0.034$, then **OK**

If $p(x_{\text{test}}^{(2)};\ \hat{\mu}, \hat{\sigma}^2) = 0.004$, then **Anomaly**

How to evaluate an anomaly detection system

Evaluation

Importance of real-number evaluation

When you are developing a machine learning algorithm, deciding what to do next is easier if you have some way of evaluating your learning algorithm.

Evaluation

Importance of real-number evaluation

When you are developing a machine learning algorithm, deciding what to do next is easier if you have some way of evaluating your learning algorithm.

To evaluate an **anomaly detection system**, it is helpful to have some labeled data specifying anomalous ($y = 1$) and non-anomalous ($y = 0$) examples.

Money Laundering Example

Evaluation

Suppose you have some **labeled data** of normal transactions:

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

assume all are non-anomalous

Money Laundering Example

Evaluation

Suppose you have some **labeled data** of normal transactions:

$$\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\} \qquad \textit{assume all are non-anomalous}$$

Next, define a **cross-validation** set:

$$\{(x_{\text{cv}}^{(1)}, y_{\text{cv}}^{(1)}), (x_{\text{cv}}^{(2)}, y_{\text{cv}}^{(2)}), \ldots, (x_{\text{cv}}^{(m_{\text{cv}})}, y_{\text{cv}}^{(m_{\text{cv}})})\}$$

and **test** set:

$$\{(x_{\text{test}}^{(1)}, y_{\text{test}}^{(1)}), (x_{\text{test}}^{(2)}, y_{\text{test}}^{(2)}), \ldots, (x_{\text{test}}^{(m_{\text{test}})}, y_{\text{test}}^{(m_{\text{test}})})\}$$

each containing **known** anomalous examples.

Money Laundering Example

Evaluation

100,000 normal transactions
50 illegal (anomalous) transactions

Money Laundering Example

Evaluation

100,000 normal transactions
50 illegal (anomalous) transactions

Training: 60,000 normal transactions ($y = 0$)
used to fit $p(x) = p(x_1; \mu_1, \sigma_1^2), \ldots, p(x_n; \mu_n, \sigma_n^2)$
i.e., we estimate $\mu_1, \sigma_1^2, \mu_2, \sigma_2^2, \ldots, \mu_n, \sigma_n^2$

Money Laundering Example

Evaluation

100,000 normal transactions
50 illegal (anomalous) transactions

Training: 60,000 normal transactions ($y = 0$)
used to fit $p(x) = p(x_1; \mu_1, \sigma_1^2), \ldots, p(x_n; \mu_n, \sigma_n^2)$
i.e., we estimate $\mu_1, \sigma_1^2, \mu_2, \sigma_2^2, \ldots, \mu_n, \sigma_n^2$

CV: 20,000 normal transactions ($y = 0$), 25 anomalous ($y = 1$)
Test: 20,000 normal transactions ($y = 0$), 25 anomalous ($y = 1$)

Algorithm Evaluation

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

On a cross validation/test example x, predict

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

Algorithm Evaluation

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

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

For each of our **test** examples $(x_{\text{test}}^{(i)}, y_{\text{test}}^{(i)})$, we evaluate the predictions of $(y = 1)$.

Note that the classes are **skewed**.

Algorithm Evaluation

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

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

Possible evaluation metrics:

- Confusion matrix
- Precision/Recall
- F_1-score

Algorithm Evaluation

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

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

Possible evaluation metrics:

- Confusion matrix
- Precision/Recall
- F_1-score

Can also use cross-validation set to choose parameter ϵ

Anomaly Detection & Supervised Learning

Using labeled data

Since we are using labeled data, one might ask:

Why don't we simply use a supervised learning algorithm?

Anomaly Detection vs Supervised Learning

Anomaly Detection

Very small number of positive examples:
 0-50 ($y = 1$)

Very large number of negative examples:
 thousands ($y = 0$)

Supervised Learning

Large number of examples:
 balanced ($y = 0$) and ($y = 1$)

Anomaly Detection vs Supervised Learning

Anomaly Detection

Very small number of positive examples:
  0-50 ($y = 1$)

Very large number of negative examples:
  thousands ($y = 0$)

Fit $\mathbf{p(x)}$ with *only* negative examples.

Supervised Learning

Large number of examples:
  balanced ($y = 0$) and ($y = 1$)

Anomaly Detection vs Supervised Learning

Anomaly Detection

Very small number of positive examples:
 0-50 ($y = 1$)

Very large number of negative examples:
 thousands ($y = 0$)

Fit $\mathbf{p(x)}$ with *only* negative examples.

Many different "types" of anomalies.
It is difficult for any algorithm to learn directly from positive examples what the anomalies look like.

Future anomalies may look nothing like past anomalies

Supervised Learning

Large number of examples:
 balanced ($y = 0$) and ($y = 1$)

Enough positive ($y = 1$) examples for the algorithm to learn *directly* what positive examples are like.

Future anomalies will likely resemble past anomalies

Applications

Anomaly Detection

Fraud detection

Manufacturing defects

Monitoring a data center

Supervised Learning

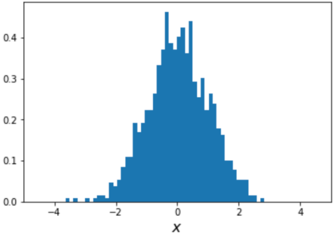
Weather prediction

Cancer classification

Email spam classification

Feature Selection I

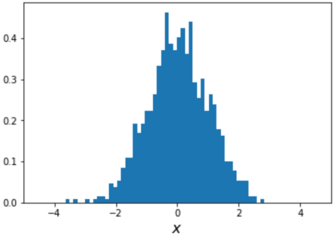
Non-Gaussian Features



For anomaly detection, we modeled each of the features x by a **Gaussian** distribution:

$$p(x; \mu, \sigma^2)$$

Non-Gaussian Features



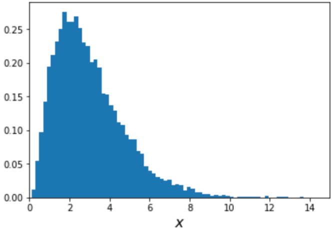
For anomaly detection, we modeled each of the features x by a **Gaussian** distribution:

$$p(x; \mu, \sigma^2)$$

Before applying this model, we can plot a **histogram** of the data to see whether this is a reasonable modeling assumption.

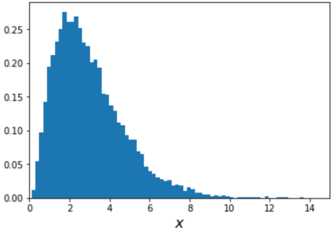
```
plt.figure()
plt.hist(data, bins = 50, normed = True)
plt.xlim(-5,5)
plt.xlabel(r'$x$', fontsize=16)
plt.show()
```

Non-Gaussian Features



Although the algorithm will usually work for not-exactly-Gaussian features, you might run into very non-Guassian data.

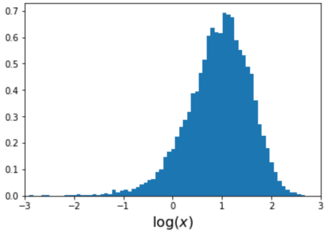
Non-Gaussian Features



Although the algorithm will usually work for not-exactly-Gaussian features, you might run into very non-Guassian data.

One thing you can do is look at different **transformations** of your data to better approximate a Gaussian distribution

Non-Gaussian Features



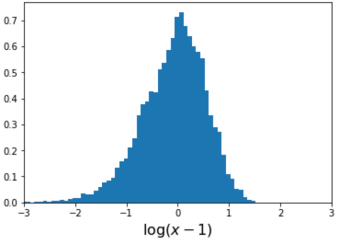
Although the algorithm will usually work for not-exactly-Gaussian features, you might run into very non-Guassian data.

One thing you can do is look at different **transformations** of your data to better approximate a Gaussian distribution

For example, you might take the **log** transformation of your data:

$$\log(x)$$

Non-Gaussian Features



Although the algorithm will usually work for not-exactly-Gaussian features, you might run into very non-Guassian data.

One thing you can do is look at different **transformations** of your data to better approximate a Gaussian distribution

For example, you might take the **log** transformation of your data:

$$\log(x)$$

More generally, you can try

$$\log(x + c) \qquad \text{for some constant } c$$

Error Analysis for Anomaly Detection

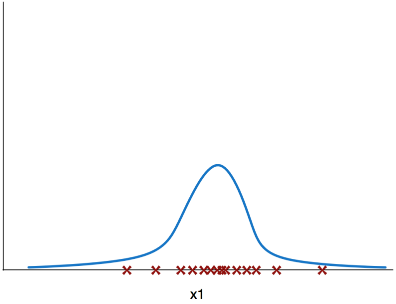
How to find features?

Goal:

$p(x)$ **large** for normal examples x

$p(x)$ **small** for anomalous examples x

Error Analysis for Anomaly Detection



How to find features?

Goal:

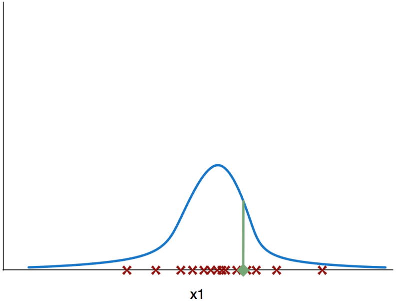
$p(x)$ **large** for normal examples x

$p(x)$ **small** for anomalous examples x

Common Problem:
$p(x)$ is comparable (e.g., both large) for normal and anomalous examples, thus the probability model $p(x)$ **fails to discriminate**.

Inspect the anomaly, and try to introduce a **new feature** x_2 to distinguish this anomaly from normal examples.

Error Analysis for Anomaly Detection



How to find features?

Goal:

$p(x)$ **large** for normal examples x

$p(x)$ **small** for anomalous examples x

Common Problem:
$p(x)$ is comparable (e.g., both large) for normal and anomalous examples, thus the probability model $p(x)$ **fails to discriminate**.

Inspect the anomaly, and try to introduce a **new feature** x_2 to distinguish this anomaly from normal examples.

Error Analysis for Anomaly Detection

How to find features?

The aim is to pick features that take on unusually large or small values in the event of an anomaly.

Example: Computer cluster

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

Error Analysis for Anomaly Detection

How to find features?

The aim is to pick features that take on unusually large or small values in the event of an anomaly.

Example: Computer cluster

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

New feature:

$x_5 = x_3/x_4$

Feature Selection II

Dimensionality

The **Curse of Dimensionality**, a phrase coined by Richard Bellman, refers to problems unique to data analysis in high-dimensional spaces.

Dimensionality

The **Curse of Dimensionality**, a phrase coined by Richard Bellman, refers to problems unique to data analysis in high-dimensional spaces.

> **Combinatorics**: for n binary variables, the possibility combinations is exponential in n; $O(2^n)$.
>
> **Optimization**: Backward induction is computed for each combination of values

Feature Selection for Supervised Learning

Goal: Either

(a) Given a **fixed** generalization error rate (**e**), select the **smallest** subset $w \subseteq x$ of features whose error is no greater than **e**;

Feature Selection for Supervised Learning

Goal: Either

(a) Given a **fixed** generalization error rate (**e**), select the **smallest** subset $w \subseteq x$ of features whose error is no greater than **e**;

(b) For subsets of features w of x of **size** k (i.e., $|w| = k$), find the subset of features that **minimizes e**.

Feature Selection for Supervised Learning

Two Main Approaches:

Wrapper Method: Evaluate subsets of the feature vector, but the search space **grows exponentially** in the size of the feature vector.

Feature Selection for Supervised Learning

Two Main Approaches:

Wrapper Method: Evaluate subsets of the feature vector, but the search space **grows exponentially** in the size of the feature vector.

- slow to intractable
- generally good performance (classification rates on test sets)
- prone to overfitting

Feature Selection for Supervised Learning

Two Main Approaches:

Wrapper Method: Evaluate subsets of the feature vector, but the search space **grows exponentially** in the size of the feature vector.

- slow to intractable
- generally good performance (classification rates on test sets)
- prone to overfitting

Filter Method: Model-independent selection criteria, evaluated by the **relevance** of the feature to the target variable you wish to predict.

Feature Selection for Supervised Learning

Two Main Approaches:

Wrapper Method: Evaluate subsets of the feature vector, but the search space **grows exponentially** in the size of the feature vector.

- slow to intractable
- generally good performance (classification rates on test sets)
- prone to overfitting

Filter Method: Model-independent selection criteria, evaluated by the **relevance** of the feature to the target variable you wish to predict.

- fast
- sometimes fails to select the best feature subset
- more robust against overfitting

Feature Selection for Supervised Learning

Filter Methods often use **information theory** to assess the relevance of a feature subset to the predictor variable. Specifically:

Feature Selection for Supervised Learning

Filter Methods often use **information theory** to assess the relevance of a feature subset to the predictor variable. Specifically:

- Entropy
- Divergence
- Mutual Information

Entropy

For a discrete random variables X with k possible values, suppose a probability distribution

$$p(x_i) := p(X = x_i) \qquad \text{for } i = 1, 2, \ldots, k.$$

Entropy

For a discrete random variables X with k possible values, suppose a probability distribution

$$p(x_i) := p(X = x_i) \qquad \text{for } i = 1, 2, \ldots, k.$$

Entropy (H) is a measure of uncertainty of a random variable (X):

$$H(X) \; = \; -\sum_{i=1}^{k} p(x_i) \cdot \log p(x_i)$$

Entropy

For a discrete random variables X with k possible values, suppose a probability distribution

$$p(x_i) := p(X = x_i) \qquad \text{for } i = 1, 2, \ldots, k.$$

Entropy (H) is a measure of uncertainty of a random variable (X):

$$H(X) \;=\; -\sum_{i=1}^{k} p(x_i) \cdot \log p(x_i)$$

If X is a continuous random variable with probability density $f(x)$, then **Differential Entropy** is defined

$$H(X) \;=\; -\int_{-\infty}^{\infty} f(x) \cdot \log f(x)\, dx$$

Joint Entropy

A **joint probability mass** function for random variables X and Y is written $p(x_i, y_j)$.

Joint Entropy

A **joint probability mass** function for random variables X and Y is written $p(x_i, y_j)$.

Joint Entropy is defined

$$H(X, Y) = -\sum_{i=1}^{k} \sum_{j=1}^{h} p(x_i, y_j) \cdot \log p(x_i, y_j)$$

Joint Entropy

A **joint probability mass** function for random variables X and Y is written $p(x_i, y_j)$.

Joint Entropy is defined

$$H(X, Y) \;=\; -\sum_{i=1}^{k}\sum_{j=1}^{h} p(x_i, y_j) \cdot \log p(x_i, y_j)$$

and takes values in the range

$$\max\left(H(X), H(Y)\right) \;\leqslant\; H(X, Y) \;\leqslant\; H(X) + H(Y)$$

Joint Entropy

$$H(X, Y) = -\sum_{i=1}\sum_{j=1} p(x_i, y_j) \cdot \log p(x_i, y_j)$$

and takes values in the range

$$\max(H(X), H(Y)) \leqslant H(X, Y) \leqslant H(X) + H(Y)$$

where

the **maximum** value occurs when X and Y are **independent** ($X \perp\!\!\!\perp Y$)

the **minimum** value occurs when X is completely **dependent** on Y

Conditional Entropy

Conditional Entropy is

$$H(X \mid Y) = H(X, Y) - H(Y)$$

where the **minimal** value is 0, which occurs when there is no uncertainty in X if the value of Y is known, and

Conditional Entropy

Conditional Entropy is

$$H(X \mid Y) = H(X, Y) - H(Y)$$

where the **minimal** value is 0, which occurs when there is no uncertainty in X if the value of Y is known, and

the **maximal** value occurs when knowing the value of Y does not reduce the uncertainty in X.

Mutual Information

Mutual Information measures "how much information" one random variable has about another random variable.

Mutual Information

Mutual Information measures "how much information" one random variable has about another random variable.

In the context of **Feature Selection**, mutual information is used to quantify the relevance of a feature subset to the target variable Y (which in `Python` we treat as a rank-one array (vector), `y`).

Mutual Information

The **Mutual Information** (MI) of X and Y, $I(X, Y)$, is defined

$$I(X;Y) = \sum_{i=1}\sum_{j=1} p(x_i, y_j) \cdot \log\left(\frac{p(x_i, y_j)}{p(x_i)p(y_j)}\right),$$

where:

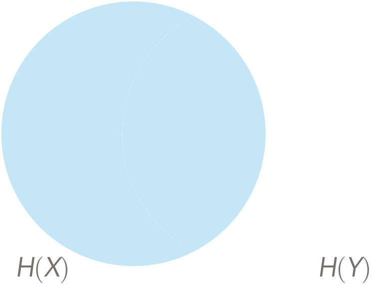
$I(X;Y) = 0$ if X is **independent** of Y ($X \perp\!\!\!\perp Y$), since $\log(1) = 0$,

$I(X;Y) > 0$ if X and Y are **positively correlated**,

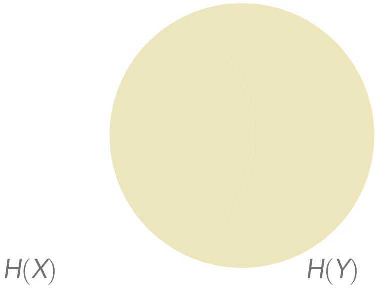
$I(X;Y) < 0$ if X and Y are **negatively correlated**.

and $I(X;Y)$ is **symmetric**:

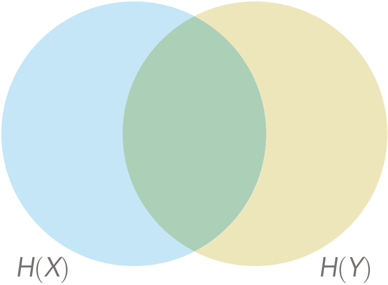
$$I(X;Y) = \begin{cases} H(X) - H(X \mid Y) \\ H(Y) - H(Y \mid X) \\ H(X) + H(Y) - H(X,Y) \end{cases}$$



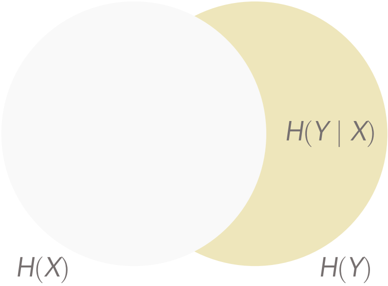
Individual Entropy $H(X)$



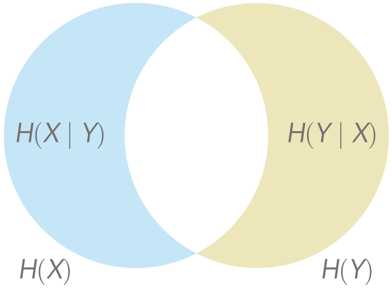
Individual Entropy $H(Y)$



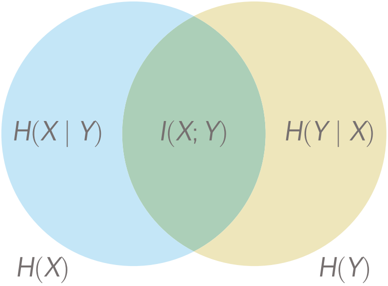
Joint Entropy $H(X; Y)$



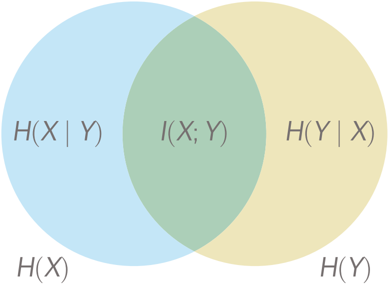
Conditional Entropy $H(Y \mid X)$



Conditional Entropy and $H(X \mid Y)$



Mutual Information $I(X; Y)$



Mutual Information

$$I(X;Y) = \begin{cases} H(X) - H(X \mid Y) \\ H(Y) - H(Y \mid X) \\ H(X) + H(Y) - H(X,Y) \end{cases}$$

References

Wheeler, G. (2018).
Bounded rationality.
In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Winter 2018 ed.). Metaphysics Research Lab, Stanford University.