Performance Evaluation and Hypothesis Testing
Motivation

Evaluating the performance of learning systems is important because:

➢ Learning systems are usually designed to predict the class/value of “future” unlabeled data points

➢ In some cases, evaluating alternative models (that we call «hypotheses») is an integral part of the learning process

➢ For example, when pruning a decision tree, alternative pruned trees represent different “hypotheses” on how to interpret our data; in neural networks, different network architectures – with different numbers of hidden layers – also represent alternative hypotheses.

Which one is the best predictor of reality?
Issues

1. Which performance measure we should use?
2. How well can a classifier be expected to perform on “novel” data, not used for training? Since a performance measure is an estimate on a sample, how accurate is our estimate?
3. How to compare performances of different hypotheses or those of different classifiers?
Which performance should we use?

Performance can be objectively evaluated only if we have a «ground truth» (either test data not used during training, or human experts able to validate predictions).

Performance measures are a function of the errors made by the current model.

Adopted performance measures depend on whether we are learning a classifier or a regressor.

For classifiers, e.g., decision trees, the error function is binary: either the learned model is correct (it predicts the right class) or it is wrong.

For regressors, we must take into account the «distance» between the predicted value and the ground-truth.
The «real» Function and hypothesis

Whether our algorithm must learn a discrete $c(x)$ or continuous $f(x)$ function, the problem is that we are given the «true» values of the function ONLY for the points (examples) of the training set.

Learning a model amounts to learning a function $h(x)$ – named an hypothesis – that approximates the unknown function at best (note we now use $h(x)$ rather than $f(x)$ or $c(x)$, to highlight the fact that ML systems learn approximate solutions of a given problem!)

Perfect learning is not possible in the majority of real-life cases.

During the learning process, ML algorithms try to «fit» at best $h(x)$ (usually, in an iterative manner) so as to minimize errors on the learning set points.

Once an hypothesis is learned, we must evaluate its quality.
Classifier and (linear) regressor errors

\[ c(x_{1i}, x_{2i}) = Y \]

\[ e = \begin{cases} \text{red} & \text{if } Y = 0 \\ \text{blue} & \text{if } Y = 1 \end{cases} \]

\[ f(x_i) = y_i = mx_i + q \]

\[ y = f(x_i) \]

h(x): \( y_i = mx_i + q \)
Performance measures for classifiers
Classifier error (measured on the test set)

\[ c(x_{1i}, x_{2i}) = Y \]

\[ x_i: (x_{1i}, x_{12i}) \]

\[ \text{error}(h(x)) = \sum_{i=1}^{n} \delta(c(x_i), h(x_i)) \]

\[ \delta(x, y) = 0 \text{ if } x = y, \text{ else } \delta(x, y) = 1 \]

\[ \delta \text{ It is called the Kronecker function} \]

\[ y = \{ \text{red, blue} \} \]
Performances of classifiers

● For classifiers, often it matters to distinguish the **types** of errors: is the system misclassifying the «reds» or the «blues»??
● Performances are usually reported in the form of a **confusion matrix** (also called **contingency table**)
● The table has four cells (in case of binary classifiers):

  ➢ **True Positive** (TP): number of positive (=blue, =1..) instances classified as positive by the system
  ➢ **False Positive** (FP): number of negative instances classified as positive by the system
  ➢ **True Negative** (TN): number of negative (=red, =0,..) instances classified as negative by the system
  ➢ **False Negative** (FN): number of positive instances classified as negative by the system
Contingency Table

<table>
<thead>
<tr>
<th>PREDICTED CLASS</th>
<th>ACTUAL CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSITIVE</td>
<td>TRUE POSITIVE (TP)</td>
</tr>
<tr>
<td>NEGATIVE</td>
<td>FALSE NEGATIVE (FN)</td>
</tr>
</tbody>
</table>
Learned separating line

Total instances 43
Total positive 24
Total negative 19

<table>
<thead>
<tr>
<th></th>
<th>TP: 22</th>
<th>FP: 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN: 2</td>
<td>TN: 17</td>
<td></td>
</tr>
</tbody>
</table>
Performances measures of classifiers (1)

Precision = \frac{TP}{TP + FP}

Recall = \frac{TP}{TP + FN}

F - Score = 2 \frac{Precision \cdot Recall}{Precision + Recall}

ACCURACY = \frac{TP + TN}{TP + TN + FP + FN}

Error rate = Classification error = \frac{FP + FN}{TP + TN + FP + FN} = 1 - Accuracy

➢ Recommended reading [link]
Other measures

**Specificity** (or True Negative Rate) \(\frac{TN}{TN+FP}\) detected negative over all negative

**False Positive Rate** \(\frac{FP}{FP+TN}\) misclassified negative over all negative

**False Negative Rate** (Miss rate) \(\frac{FN}{FN+TP}\) misclassified positive over all positive

Note that \(FP+TN = \) total number of negative in test set

\(FN+TP = \) total number of positive in test set
Total instances 43
Total positive 24
Total negative 19

<table>
<thead>
<tr>
<th>TP</th>
<th>FP</th>
<th>FN</th>
<th>TN</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>2</td>
<td>2</td>
<td>17</td>
</tr>
</tbody>
</table>

Accuracy: \( \frac{22 + 17}{43} = 0.907 \)

Precision: \( \frac{22}{24} = 0.91 \)

Recall (TPR): \( \frac{22}{24} = 0.91 \)

F-Score: \( 0.91 \)

FPR: \( \frac{2}{19} = 0.10 \)
Performances measures of classifiers (2)

- **Receiver Operating Characteristic curve** (or ROC curve.) is a **graphical plot** that illustrates the performance of a **binary classifier** systems.

- The curves are created by plotting the **recall** (True Positive rate; TPR) against the **false positive rate** (FPR) at various system settings (e.g., hyperparameters, the dimension of training set, etc). One **would** aim at **high recall** and **low FPR**.

- FPR=\(\frac{\text{False positives}}{\text{All negatives}}\)

Note that «random» is a bisector if data have an equal probability of being positive or negative.
Performance measures of classifiers (3): AUROC/AUC

- The **Area Under the ROC (AUROC or AUC)** ROC is a probability curve and AUC represents a measure of «separability». It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.

Here, A and B are two different models (e.g., with different values of some hyperparameter) at different settings (e.g., increased dimension of training set)
Why is AUROC useful?

It may help understand what is the «uncertainty» zone of your predictor, and output a classification only if outside this zone.

Example: we predict if a paper will be accepted at a conference, based on features like length of the paper, number of authors... Say red points are accepted papers, blue are rejected.

Let’s say our predictor output a probability, or confidence value (on the x axis), that a paper is accepted or not. The y axis is the count of observations in the test set (say we have 250 accepted, 250 rejected in our test set). For example, there are 50 papers for which \( p=0.7 \), and they are indeed all positive. 20 papers have \( p=0.5 \), and 10 are positive, 10 are negative.

As shown in the figure, when the system output a probability between 0.4 and 0.6, it has a 50% chance of being wrong! So we should not rely on system’s predictions for these output values.
AUC detects the «uncertainty area»

For $p>0.6$ the system reliably classifies positive, for $p<0.4$ it reliably classifies negative, in between the system is unable to correctly separate positive from negatives.
To summarise, green and red curves represent the probability that a given model classifies an instance as positive or negative given the values of its features. AUROC (the rightmost curves) tells us how good the model is at separating.

This diagram is not representative of most real-world problems:

1. Most problems don’t have balanced classes
2. Predicted probabilities are unlikely to have a "smooth" distribution

↩ Here the model is reciprocating the classes!
Performances measures of classifiers (4)

**Precision-Recall curve** (or PR curve) is a **graphical plot** that illustrates the performance of a **binary classifier** system. The curve is created by plotting the **recall** (True Positive rate; TPR) against the **precision** at various system settings (parameters, e.g., A and B, the dimension of training set, etc).

- The **Area Under the Precision-Recall curve (AUPR)** has an intuitive meaning just like **AUROC**. However:
  - **AUROC** is better for a **binary balanced** problem.
  - **AUPR** is better for a **binary imbalanced** problem (we discussed about imbalanced classes under the topic feature engineering). See [link](#)
Why not ROC/AUROC with imbalanced classes?

- Suppose we have imbalanced data, e.g., in credit risk prediction, the vast majority of instances in the dataset are negative (not fraudulent users) and only a minority is positive. **We really care about capturing positive instances.**
- ROC curve is not a good visual illustration for highly imbalanced data, because the False Positive Rate (False Positives / Total Real Negatives) **does not drop drastically when the total number of real negatives is huge.**
- Whereas Precision (True Positives / (True Positives + False Positives)) **is highly sensitive to False Positives** and is not impacted by a large total real negative denominator.
Performance measures of classifiers (5): Learning Curves

- Plots accuracy vs. **size of the training set**
- **Questions:**
  - Has maximum accuracy nearly been reached or will more examples help?
  - Is one system better when training data is limited?

Most learners eventually converge to optimal given sufficient training examples.
Performance of regressors
Performance measures of Regressors

**Error**

\[
\text{error}(h(x_i)) = f(x_i) - h(x_i)
\]

**MAE (mean absolute error)**

\[
\frac{1}{n} \sum_{i=1}^{n} |f(x_i) - h(x_i)|
\]

**RMSE (Root Mean Squared Error)**

\[
\sqrt{\frac{1}{n} \left( \sum_{i=1}^{n} (f(x_i) - h(x_i))^2 \right)}
\]

**RSS (Residual Sum of Squares)**

\[
\left( \sum_{i=1}^{n} (f(x_i) - h(x_i))^2 \right)
\]

.. plus others algorithm-dependent error functions, that we will see later
Issues

1. Which performance measure we should use?

2. How well can a classifier be expected to perform on “novel” data, not used for training? Since a performance measure is an estimate on a sample, how accurate is our estimate?

3. How to compare performances of different hypotheses or those of different classifiers?
Before we try to answer the issue 2..

WE NEED TO UNDERSTAND WHAT ARE THE CAUSES OF AN ERROR (IN CLASSIFIERS AND REGRESSORS)
Why errors, in the first place?

- The task of a ML learning algorithm is to find an hypothesis model $h(x)$ such that it approximates at best the real function $y=f(x)$ both on the points $x_1..x_n$ of our dataset $D$, and on all other unseen examples.
- So $h(x)$ must GENERALIZE on unseen examples.
- However, perfectly fitting $f(x)$ is impossible in most cases, as we said.
- The errors (the difference between the real and learned functions) is made up of different components.
The error components

Bias-variance decomposition

\[
\text{Total Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}
\]

The bias is an error originating from erroneous assumptions in the learning algorithm (e.g., that data are linearly separable but they are not).

The variance is an error originating from sensitivity to fluctuations in the data (e.g., if feature values are very sparse, and the predictor is not sufficiently robust, performance may vary greatly according to a specific choice of the learning/test sets).

Irreducible error is the error that can’t be reduced by creating good models (it is related to inherent complexity of a problem)

A good model has to minimize the error (low bias and variance!)

Will explain later this formula
Bias: definition

Bias is average of difference between predicted and actual result. High Bias means that we are getting low performance. Bias originates from wrong assumptions (e.g. that data are linearly separable, while they are not)

$$\text{Bias}^2 = (E[h(x)] - f(x))^2$$

$E[h(x)]$ expected value (mean) of $h(x)$

In this example (a regressor), the (unknown) $f(x)$ is not linear, but the ML algorithm has a linear bias. The three models $h_1(x)$, $h_2(x)$ and $h_3(x)$ are learned on 3 different training sets. The mean $E[h(x)]$ is computed averaging the three different linear predictors.
Warning: Reducing bias may lead to overfitting, e.g. paying too much attention to the data
What causes bias in data?

4 types of bias

1. **Sample bias** (collected data are not representative)  
   Ex: your goal is to create a model for face recognition, but you train the model only with white males images.

2. **Exclusion bias** (we are excluding some relevant feature)  
   Ex: In the famous Titanic problem, say we want to predict who survived and who didn’t. One might disregard the passenger id of the travelers as it may seem completely irrelevant to whether the passenger survived or not. Little did they know that Titanic passengers were assigned rooms according to their passenger id.
Types of bias

3. **Prejudice bias** Ex. Goal is to detect people at work. Your model has been fed to thousands of training data where men are coding and women are cooking. The algorithm is likely to learn that coders are men and women are chefs.

4. **Measurement bias**. Ex. Shooting images data with a camera that increases the brightness. This causes example data to be systematically altered wrt reality.

    There are no clear-cut methods to prevent bias.
    It is a «hot» research topic [link](#).
Variance (the intuition)

High variance

Low variance
Variance (for any discrete distribution)

Variance is defined as the mean of squared differences between values of \( N \) individual outcomes \( x_i \) and the mean (\( \bar{x} \)), i.e. the dispersion around the mean.

\[
\sigma^2 = \text{var}(x) = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N}
\]

- For infinite values (continuous distributions), the sum becomes an integral.

In our case, the variance is the «dispersion» of the data instances values \( x_i \) around the mean.

\[
\sum_{i=1}^{n} (h(x_i) - E(h(x)))^2
\]

NOTE: In classifiers, the source of variance is not the class value, but the feature values.
Ex of variance in data: stock prices
How to reduce the variance?

Variance cannot be reduced if inherent of our data. We can adopt two techniques (or a combination of the two):

- Train different models (each takes care of specific features of the data, we will discuss this technique with ensemble methods)
- Use a validation technique that reduces the possibility of «being unlucky» when randomly selecting a test set: **K-Fold Cross-Validation**:

  K-FCV: perform several independent splits on learning and test set and then average the performance over these different splits.
K-Fold Cross-Validation of a hypothesis (model)

Partition all the available labeled data in \( k \) equally sized random samples.
K-Fold Cross-Validation of a hypothesis (model)

At each step $i$, learn from $\text{Learn}_i$ and test on $\text{Test}_i$, then compute the error ($e_i$) on $\text{Test}_i$.
Why K-Fold Cross Validation reduces the variance?

- Intuitively, it reduces the probability of “being lucky”, or unlucky, in selecting the test-set

- To understand the issue more in detail, we need to introduce the next topic:  
  ➢ testing the accuracy of an error estimate
Back to the initial issue:

1. Which performance measure we should use?
2. How well can a classifier be expected to perform on “novel” data, not used for training? Since a performance measure is an estimate on a sample, how accurate is our estimate?
3. How to compare performances of different hypotheses or those of different classifiers?
Warning

Performance evaluation is about estimating errors

Error estimates are governed by **probabilities**

The language of next coming slides is a «probability» language

- The **test set** is a SAMPLE selected from a **POPULATION** (the instances of our data)
- The **value** of an instance, or a feature, in a sample is called an **OBSERVATION**
- the **error** of a machine learning model on a test set is a **RANDOM VARIABLE**, our task is to **ESTIMATE** this error
Evaluation: What is an Estimator?

An Estimator is any function on a sample of the data that is used to estimate some useful qualities of the original data from which the sample is drawn. Formally, an estimator is a function on a sample $S$:

$$\hat{\theta}_S = g(S), S = (x(1), \ldots, x(m)),$$

where $x(i)$ is a random variable drawn from a distribution $D$, i.e. $x(i) \sim D$.

- We would like to use the sample $S$ to estimate some useful qualities of the original data.
- For example, the mean is an estimator.
- In general, an estimator is any random variable used to estimate some parameter of the underlying population from which the sample is drawn.
- An obvious question to ask about any estimator is whether «on average» it gives the right estimate.
Questions to be considered in estimating the error of a model

Let \( h(x) \) be a model learned by a specific ML algorithm \( L \) with some specific hyper-parameters. We denote \( h \) as a “hypothesis”, and the objective is to estimate its prediction accuracy. The following are relevant questions:

1. **Q1:** Given the observed accuracy (or any other performance measure) of \( h \) over a limited sample of test data \( S \), how well does this estimate its accuracy over additional (unseen) instances?

2. **Q2:** Given that one hypothesis \( h_1 \) outperforms another, \( h_2 \), over some sample data \( S \), how probable is it that this hypothesis is more accurate in general (= over the full instance space)?

Note: we analyse the problem for classifiers, extending to regressors is straightforward
Estimating Hypothesis Accuracy

A better formulation of Q1:

➢ Given a hypothesis $h$ and a data sample containing $n$ instances drawn at random according to distribution $D$, what is the best estimate of the accuracy of $h$ over future instances drawn from the same distribution? Need to consider: sample error vs. true error

➢ What is the «probable error» in this accuracy estimate?

Need to consider: confidence intervals (ranges in which the «true value» of the error may lie)

In other terms, if we measure an error rate (on a sample S) of, say, 20%, the true error rate of $h$ on any sample is not guaranteed to be exactly 20%. Let’s say that it is $20\% \pm \Delta$. Can we estimate the value of $\Delta$?
Confidence intervals

Estimating the interval around the estimated error, such that the true (unknown) error lies within these bounds with some confidence. (See later)
Sample Error and True Error

- **Definition Sample Error** (i.e., error\(_s\)(h), error rate): The **sample error** of hypothesis \(h(x)\) for the target function \(c(x)\) (the ground-truth classification value \(c(x)\) of instances \(x\) in \(S\)), on a data sample \(S\) is:

\[
\text{error}(h(x)) = \frac{\sum_{i=1}^{n} \delta(c(x_i), h(x_i))}{n}
\]

where:

- \(n\) is the number of instances in sample \(S\)
- \(r\) is the number of misclassified instances
- \(h(x)\) is the classification produced by our current model \(h\)
- \(\delta( c(x) \neq h(x) ) = 1\) if \(c(x) \neq h(x)\), and \(0\) otherwise.
Sample error and True error

- **Definition True Error** (i.e., $\text{error}_D(h)$, $p$):
The true error of hypothesis $h$ for the target (unknown) classification function $c(x)$ and distribution $D$ of instances, is the probability that $h$ will misclassify any instance $x$ drawn at random according to $D$

$$\text{error}_D(h) = \Pr_{x \in D}[c(x) \neq h(x)]$$

$\text{error}_s(h)$ is an estimator of $\text{error}_D(h)$, which is a probability

So, how good is this estimator?
Estimate, probability and random variables

We are given a sample $S$ of $n$ instances, we classify $S$ with $h(x)$ and we measure $r$ errors, we then estimate the error probability of $h(x)$:

$$\text{error}_S(h) = P(r \text{ errors in } n \text{ instances}) = \frac{r}{n} = 1 - \text{accuracy}_S(h)$$

● Note: We call $S$ “sample” since it can be any subset $X'$ of the set of instances $X$ sampled according to a distribution $D$.

● However, $r$ (or $\frac{r}{n}$) is a random variable, governed by chance. If we choose another sample $S'$ of $n$ different instances, we may get a different number $r'$ and a different estimate. In general $\text{error}_S(h) \neq \text{error}_{S'}(h)$

A Random Variable can be viewed as the name of an experiment with a probabilistic outcome. Its value is the outcome of the experiment.
Estimate, probability and random variables

- **A simple experiment for a Random Variable:**
  Make $k$ different sets of trials, in each trial, toss a coin 10 times and measure the number of “head”. Although, as the number of experiments $k$ increases, the average number of “head” occurrences tend to $\frac{1}{2}$, in every single trial you will likely obtain different numbers.

- In coin tossing, we know that the “real” head rate (the expected value for the fraction of head tosses) is 50%, but in hypothesis testing, **we don’t know what is the real error rate**. So, how can we get an idea of $\text{error}_D(h)$ on the entire population $X$, distributed according to $D$?
Our question is: Is the Sample Error a good estimator for the True Error?

We do not know the “true” error probability however we know that $\text{error}_s(h)$ is a random variable that follows a binomial distribution with mean $p$ (the unknown true error rate)

Why? What is this “binomial”?
Sample Error & True Error:

Why? What is this “binomial”?

- Say \( p \) is the (unknown) “true” error probability of \( h(x) \) on \( X \). If we have a sample \( S \) of \( n \) instances (test set), what is the probability that, given instances \( x \) in \( S \), \( f(x) \neq h(x) \) for \( r \) times??

- Even if we do not know the true error rate \( p \), each instance \( x \) in \( S \) has probability \( p \) of being misclassified by \( h(x) \) and \( (1-p) \) of being classified correctly.

- The probability of observing \( r \) misclassified examples in \( n \) instances is then:

\[
P(X = r) = \binom{n}{r} p^r (1 - p)^{n-r} = \frac{n!}{r!(n-r)!} p^r (1 - p)^{n-r}
\]

This is called a binomial
Example:

\( \rho \) is the probability of rain in January

The abscissa is the value \( r \), e.g. there is a 20% probability that there will be 5 rainy days out of 20 observations, 6% probability of 8 rainy days out of 20, etc.

What is the probability of \( x \) rainy days out of 20 days?
How do we compute these probabilities?

- Say we know that $p(\text{rain}) = 25\%$ (on January)

- However, if we watch the weather in 4 consecutive days, **we are not sure** we will get “rain” 1 time and “not rain” 3 times. The number of observed “rainy days” in each trial of 4 consecutive days **is governed by chance**.

- What is the probability of getting, instead, 2 rainy days in 4 days?

  \[
  P(\text{2 "rain" in 4 observed days}) = \binom{4}{2} (0.25)^2 (1 - 0.25)^2 = \\
  = \frac{4!}{2!(4-2)!} (0.25)^2 (1 - 0.25)^2 = \frac{1 \cdot 2 \cdot 3 \cdot 4}{1 \cdot 2(1 \cdot 2)} 0.065(0.5625) = 0.21
  \]

**Same formula** to estimate the probability of 2 errors over 4 instances, given we know that the true error rate is 25\%
Example: *p is the probability that our ML system misclassifies an instance x drawn at random from the entire population of instances.*

What is the probability of x errors when classifying 20 instances?

- The abscissa is the value *r*, e.g. there is a 20% probability that there will be 5 errors out of 20 classifications, 6% probability of 8 out of 20, etc.
We can normalize and plot \( \frac{r}{n} \)

Now \( x \) is the \% of wrongly classified instances

Even if we do not know \( p \), we know that if we perform several experiments on different samples \( S \) (test sets) we will observe a bell shape distribution of the error rate \( \frac{r}{n} \)!!!!
Properties of Binomial distribution

\[ P(X = r) = \frac{n!}{r!(n-r)!} p^r (1 - p)^{n-r} \]

- **Expected Value of** \( r \) **over** \( n \) **trials:** \( E(X) = np \)
- **Variance:** \( Var(X) = np(1 - p) \)
- **Standard Deviation (STD, SD):** \( \sigma(X) = \sqrt{np(1 - p)} \)

\[ Var(X) = \frac{\sum_{i=1}^{n} (o_i - p)^2}{n} = \frac{1}{n} (np(1 - p)^2 + n(1 - p)(1 - p)^2) = np(1 - p) \]

\( o_i \) is the outcome of a correctness test on instance \( x_i \). It is 1 if \( c(x) \neq h(x) \) and 0 if \( c(x) = h(x) \).
Now, we know that the random variable \( X = r \) (number of errors observed in \( n \) independent tests) follows a binomial distribution with unknown mean \( np \). If we compute the error rate on a sample of \( n \) observations \( S \), we obtain a value \( r/n \) which is our current estimate \( \text{error}_S(h) \) of \( \text{error}_D(h) \).

- **Note** that the “estimator” \( \text{error}_S(h) \) is also a random variable! If we perform many experiments on different samples \( S_i \) we could get different values.

- However, for large enough dimension of the sample \( S \), the expected value of \( \text{error}_S(h) \) (i.e. \( \mathbb{E}[\text{error}_S(h)] \)) is the same as for \( \text{error}_D(h) \)!

*Why? Because of the Central Limit Theorem*
Central Limit Theorem

**General Formulation:**
The theorem states that the **arithmetic mean** of a sufficiently large number of experiments of **independent random variables**, each with a well-defined **expected value** and well-defined **variance**, will be approximately **normally distributed**.

- This will hold **regardless of whether the source population is normal or skewed (biased)**, provided the samples size is sufficiently large (usually \( n > 30 \))

- Furthermore, **the mean of all such experiments will (tend to) be the same as the “real” population mean**

- **A Normal distribution (or Gaussian Distribution):**
  A family of continuous probability distributions such that the probability density function is the normal (or Gaussian) function
Putting it all together:
error$_S$(h) and error$_D$(h) both follow a gaussian law, and
E(error$_S$(h)) $\rightarrow p$

In our case we know:

- a) **Experiments are accuracy tests** on data samples $S_i$
- b) The involved **random variables are the error rates** $r_i/n_i$
  observed on these samples $S_i$. These random variables are
  statistically **independent** of each other, and **follow a**
  **binomial** distribution, as we have seen
- c) For a **sufficiently large number of experiments**, the
  observed **values** $r_i/n_i$ will be approximately **normally**
  **distributed**, according to the central limit theorem
- d) Their **mean value** will tend to the “real” (the unknown true
  value) expected error $p$ over the entire set of instances $X$
The average of many observed values of the random variable $\text{error}_S(h)$, generated by repeated random experiments, converges toward $p$, the expected value of the TRUE error rate over the entire distribution $D$ of instances.

$p$ is the unknown expected (true error) error rate.

Yellow bars are the results of different experiments on different samples $S_i$.

The average of these results tends to $p$ as the number of experiments grows.
What is a Gaussian Distribution?

The curve parameters are the mean $\mu$ (e.g., $p$ in our specific case of error rate) and the standard deviation $\sigma$.

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$
Interesting properties of gaussian distributions

Gaussian curves allow to establish a **fixed relationship** between standard deviation and portions of the area under the Gaussian curve. These areas are interpreted as **probability mass**.

In a gaussian curve, **for any $\mu$ and $\sigma$**, it holds that:

- $99.7\%$ of the probability mass lies in the area below the mean value $\mu \pm 3\sigma$
- $95.4\%$ of the probability mass lies in the area below $\mu \pm 2\sigma$
- $68.3\%$ of the probability mass lies in the area below $\mu \pm \sigma$
Interesting properties of Gaussian distributions

In our case, the mean $\mu$ is $p$ (i.e., $\text{error}_D(h)$).
but..when shall we get to our problem of estimating the quality of an error estimator ???
Stay tuned pls..
Consequences of applicability central limit theorem to the random variable r/n

Result 1:
If the random variable \( X = \frac{r}{n} \) follows a Gaussian distribution, then error\(_S\)(h(x))=\( \frac{r}{n} \) is an unbiased estimator of the real error rate since:

\[
\text{Bias}^2 = (E[h(x)] - f(x))^2
\]

In our case:

\[
\text{Bias(error}\_S(h))^2 = (E[\text{error}\_S(h)] - \text{error}_D(h))^2 = (p - p)^2 = 0
\]
Consequences of applicability central limit theorem to the random variable $r/n$
Consequences of applicability central limit theorem to the random variable $r/n$

**Result 2:** we can approximate the standard deviation of error$_D(h(x))$

The Standard deviation of a sample $S$ of $n$ instances is defined as:

$$\sigma_S = \frac{o_r}{n} = \frac{1}{n} \sqrt{np \cdot (1-p)} = \sqrt{\frac{p(1-p)}{n}} \simeq \sqrt{\frac{r/n}{n}} = \sqrt{\frac{error_s(h) \cdot (1-error_s(h))}{n}}$$

➢ **Note** that for $n \to \infty$ (very large samples), then $\sigma_S \to 0$

(since $r/n \to p$ i.e., the observed error will converge to the real error rate)

We replace the (unknown) $p$ with our computed mean value $r/n$. This is an **estimate** since we assume that $r/n$ is a good approximation of the real error rate $p$, which holds approximately true for large enough $n$, according to CLT!
Why is this approximation acceptable (and replacing p with r/n is not)?

- Why we can set $p(1-p) = \frac{r}{n}(1-\frac{r}{n})$??
- Say $p=0.6$ and $r/n=0.7$ (difference is 0.1)
- However, $p(1-p)=0.24$ $r/n(1-r/n)=0.21$ (difference is 0.03)

- Although approximating the real error with the estimated error can lead to a significant over or under-estimate, **approximating the real SD with the estimated SD is much less critical**

- In general, if $n$ is sufficiently large, the probability that our estimate is very far from real sigma is sufficiently low
Consequences of applicability central limit theorem to the random variable r/n

Result 3:

Normal (gaussian) distributions have important properties (e.g., 99.7% of the probability mass lies in the area below the mean value $\mu \pm 3\sigma$, 95.4% of the probability mass lies in the area below $\mu \pm 2\sigma$, 68.3% of the probability mass lies in the area below $\mu \pm \sigma$ ...)

This property allows easy calculation of confidence intervals!!
Confidence interval for an error estimate

- The confidence interval represents the statistical significance (Margin Error; ME) of the expected distance $\Delta$ between the real value (in our case, $p$) and the observed estimate (in our case, $r/n$).
- **Definition:** An N% confidence interval for some parameter $p$ is an interval $[LB, UB]$ that is expected with probability N% to contain $p$. *(with probability N% we have $LB \leq p \leq UB$)*
- The confidence interval is a way to show what the uncertainty is with a certain statistic. The margin of error ME tells you **how many percentages points your results will differ** from the real population value.

NOTE: With smaller samples ($n < 30$) the Central Limit Theorem does not apply, and another distribution called the **t distribution** must be used.
Confidence interval for an error estimate

- **Confidence interval (CI)**

\[
\Delta = |error_D(h) - error_s(h)| \leq ME = |p - \frac{\bar{z}}{n}| \leq ME
\]

\[
= |p - \frac{\bar{z}}{n}| \leq z\sigma_s = LB = \frac{\bar{z}}{n} - z\sigma_s \leq p \leq z\sigma_s + \frac{\bar{z}}{n} = UB
\]

- **ME (Margin Error)** = (Critical Value) x (Standard Deviation for the population)

  The critical values are also called z-values.

- **Δ** is called **Absolute Error** of the estimate

We are expressing the interval in terms of «how many» standard deviations. z is the «how many»
Confidence intervals

The good news is that the error follows a gaussian distribution, a regular and symmetric distribution that facilitates the computation of such intervals.
Confidence intervals computation with Gaussian Distributions

\[ \Delta = | \text{error}_D(h(x)) - \text{error}_S(h(x)) | \]

For any gaussian, we can say “with a probability of 68% (95%, 99.7%, \( N\% \)) any value \( x \) we measure for \( \text{error}_S \) will lie in the interval \( \pm 1\sigma (\pm 2\sigma, \pm 3\sigma, \pm z\sigma) \) around the mean \( p \)”. More in general, with an \( N\% \) probability it will lie in the \( \pm z\sigma \) interval.

One reason that we prefer to work with the gaussian distribution is that we have tables specifying the size of the interval around the mean that contains \( N\% \) of the probability mass under the Normal distribution. This is precisely the information (the critical values, z-values) needed to calculate our \( N\% \) confidence interval.

Since \( \text{error}_S(h(x)) \) follows a gaussian, we can use this property!
How do we compute confidence intervals in practice?

- We must fix either N% (the confidence) or z (the length of the interval, in terms of «how many» standard deviations)
- Clearly, the higher is the confidence we need, the larger is the interval we will find
- We must set as our target **either** the confidence, **or** the length of the interval
- For gaussian curves, tables are provided to determine one variable when the other is given, e.g.:

<table>
<thead>
<tr>
<th>N%</th>
<th>50</th>
<th>68</th>
<th>80</th>
<th>90</th>
<th>95</th>
<th>98</th>
<th>99</th>
</tr>
</thead>
<tbody>
<tr>
<td>z_N</td>
<td>0.67</td>
<td>1.00</td>
<td>1.28</td>
<td>1.64</td>
<td>1.96</td>
<td>2.33</td>
<td>2.58</td>
</tr>
</tbody>
</table>

- Tables are provided to compute z for any N and viceversa
- To compute confidence intervals from z tables, see [here](#)
The Z-table: Gaussian Distribution

z is **HALF** of the length of the interval around the mean $p$ that includes $N\%$ of the total probability mass. A **z-value** tells you "how many standard deviations" ($z \times \sigma$) from the mean $p$ your result $r/n$ is.

$z = (1.4 + 0.05) = 1.45$

Highlighted cell says that $N = 0.4265 \times 2$ (because you are considering only half) = 0.853 = 85.3% of the probability mass lies $\pm z = (1.4 + 0.05) = 1.45$ times the standard deviations around the mean
The Z-table: Gaussian Distribution

The red line is your z-value (e.g. 1.45)

The z-value in the table gives you this area (for a mean =0)

Or viceversa if the input is N: Dividing by 2 the probability mass (say, N= 85.3% /2 = 0.4265), we obtain the z (1.45) value from the table, to calculate the interval
How to : Finding the N% confidence interval

- We know the formula to compute the interval, given the estimated error rate:

\[ [LB, UB] = \left[ \frac{r}{n} - z \sqrt{\frac{r/n(1 - \frac{r}{n})}{n}}, \frac{r}{n} + z \sqrt{\frac{r/n(1 - \frac{r}{n})}{n}} \right] \]

- In this formula, \( z \) is unknown. But we fixed \( N \), so we look in the table and we obtain \( z \) for the desired \( N \), and compute the interval.
We have a classifier which produced a hypothesis model $h(x)$, and a test set $S$ of 100 instances.

We apply $h(x)$ on the sample test set $S$ and compute 13% (0.13) error rate ($r/n$).

Since $n > 30$, we assume that the error distribution follows a Gaussian distribution with mean 0.13 and standard deviation $\sigma_S$:

$$\sqrt{\frac{0.13(1 - 0.13)}{100}}$$

To compute the N=90% confidence interval, on the table we find $Z=1.64$.
Example 1: Calculating the N% Confidence Interval

- We then have: $Z = 1.64$ and $\sigma_s \approx \sqrt{0.13(1-0.13)/100}$
- The 90% confidence interval is estimated using the previous formula is:

$\left[0.13 - 1.64 \sqrt{\frac{0.13(1-0.13)}{100}}, 0.13 + 1.64 \sqrt{\frac{0.13(1-0.13)}{100}}\right] = [0.075, 0.19]$
Example 2:
Finding 95% CI on a face recognition task

Given the following extract from a scientific paper on multimodal emotion recognition:

We trained the classifiers with 156 samples and tested with 50 samples from three subjects.

Table 3. Emotion recognition results for 3 subjects using 156 training and 50 testing samples.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Number of Classes</th>
<th>Classifier</th>
<th>Correctly classified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face*</td>
<td>67</td>
<td>C4.5</td>
<td>78 %</td>
</tr>
<tr>
<td>Body*</td>
<td>140</td>
<td>BayesNet</td>
<td>90 %</td>
</tr>
</tbody>
</table>

For the Face modality, what is $n$? What is $\text{error}_s(h)$?

<table>
<thead>
<tr>
<th>$N%$</th>
<th>$Z_N$</th>
<th>0.67</th>
<th>1.00</th>
<th>1.28</th>
<th>1.64</th>
<th>1.96</th>
<th>2.33</th>
<th>2.58</th>
</tr>
</thead>
</table>
Example 2:

Accuracy is 0.78, hence **error rate** is 0.22; the test set has 50 instances, hence \( n=50 \).

Choose, e.g., to compute the N% confidence interval with \( N=0.95 \).

Given that \( error_S(h)=0.22 \) and \( n=50 \), and \( z_N=1.96 \) for \( N=95 \), we can now say that with 95% probability \( error_D(h) \) will lie in the interval:

\[
\left[ 0.22 - 1.96 \sqrt{\frac{0.22(1-0.22)}{50}}, 0.22 + 1.96 \sqrt{\frac{0.22(1-0.22)}{50}} \right] = \\
\left[ 0.11, 0.34 \right]
\]
One side bound (Tailed-test)

We might be interested in computing the probability that the error of our ML system is “at most” a given value, rather than within a given range like before.

Which amounts to computing the blue area

Now $N\%$ is the area for which $\text{error}_S \leq z\sigma$
Gaussian is symmetric!

One-sided / Two-sided bounds: The Gaussian distribution is symmetric and its total area is 100% (of the probability mass).
Example: One/Two-Sided bounds

In the previous emotion recognition example, we said that with 95% probability (the confidence), the true error (i.e. $\text{error}_D$) lies in the [0.11, 0.34] interval.

- There is a 5% ($100-95=5$) area outside this interval, of which, 2.5% to the left and 2.5% to the right (due to symmetry)
- Therefore, we can also say that there is a 2.5% probability that $\text{error}_D > 0.34$ (the upper bound $\text{UB}$) and 2.5% probability that $\text{error}_D < 0.11$ (the lower bound $\text{LB}$)

- There is a 97.5% (95+2.5=97.5) probability that: $\text{error}_D < 0.34$
- There is a 97.5% (95+2.5=97.5) probability that: $\text{error}_D > 0.11$
Issues

1. Which performance measure we should use?
2. How well can a classifier be expected to perform on “novel” data, not used for training?
3. Since a performance measure is an estimate on a sample, how accurate is our estimate?

4. How to compare performances of different hypotheses or those of different classifiers?

What follows is NOT part of the 2021-2022 program!!
Comparing Two Learned Hypotheses

When evaluating two hypotheses (e.g. using different hyper-parameters on the same ML algorithm), their observed ordering concerning accuracy may or may not reflect the ordering of their true accuracies.

- Assume $h_1$ is tested on the test-set $S_1$ of size $n_1$
- Assume $h_2$ is tested on the test-set $S_2$ of size $n_2$

Observe $h_1$ more accurate than $h_2$
Comparing Two Learned Hypotheses

When evaluating two hypotheses (e.g. using different hyper-parameters on the same ML algorithm), their observed ordering concerning accuracy may or may not reflect the ordering of their true accuracies.

- Assume $h_1$ is tested on the test-set $S_3$ of size $n_1$
- Assume $h_2$ is tested on the test-set $S_4$ of size $n_2$

Observe $h_1$ less accurate than $h_2$
Alternative Hypothesis Tests

When we wish to understand how much we can rely on a statistical finding (e.g., that an ML model $h_2$ is more precise than $h_1$ on a sample dataset), we need to list the alternatives (e.g. $h_2$ in not more precise than $h_1$ on the entire population).

One of these alternatives is called the Null Hypothesis $H_0$.

Usually, the null hypothesis disconfirms our findings.
Alternative Hypothesis Tests

Suppose we measured the error rate of h1 and h2 finding that 
\[ d = \text{error}_{S1}(h1) - \text{error}_{S2}(h2) \neq 0; \text{ we can perform 3 different tests:} \]

1. **Two-Tailed Test:** We formulate and tst two alternatives
   - **H0 (null hypothesis):** data do not support that \( h1 \neq h2 \)
     (hence \( \text{error}_D(h1) - \text{error}_D(h2) \) could actually be 0)
   - **H1:** data support that \( h1 \neq h2 \)
     (\( d \) is either positive or negative; with high confidence our finding is true; like \( N=95\%) \)
Alternative Hypothesis Tests

2. One-tailed right-test (d>0)
   ➢ **H0 (null hypothesis):** data do **not support** that h₂>h₁
   ➢ **H¹:** data support h₂>h₁ (error of h₁ is significantly lower)

3. One-tailed left-test (d<0)
   ➢ **H0 (null hypothesis):** data do **not support** that h₂<h₁
   ➢ **H¹:** data support h₁>h₂ (error of h₁ is significantly higher)
How to do: Two-Tailed Test

1. Assumes:
   ➢ $h_1$ is tested on the test-set $S_1$ of size $n_1$
   ➢ $h_2$ is tested on the test-set $S_2$ of size $n_2$
   ➢ $n_1 > 30$, $n_2 > 30$ to hold the Central Limit Theorem
   ➢ Note: for CLT, binomial is approximated by Gaussian, so both $error_{s_1}(h_1)$ and $error_{s_2}(h_2)$ approximately follow a gaussian distribution.

2. Suppose we wish to estimate the difference $d_D$ between the (unknown) true errors of these two hypotheses:
   \[ d_D = error_D(h_1) - error_D(h_2) \]

3. As usual, define $d_S$ the estimator of $d_D$ with the known sample errors:
   \[ d_S = error_{s_1}(h_1) - error_{s_2}(h_2) \]
   ➢ $d_S$ is an unbiased estimator of $d_D$. We will not give the proof.
How to do: Two-Tailed Test

4. To obtain the confidence interval we need to compute the $\sigma_{ds}$:

**Note:** What is the probability distribution governing the random variable? For CLT, we know that both $\text{error}_{s_1}(h1)$ and $\text{error}_{s_2}(h2)$ follow distributions that are approximately Gaussian. Because the difference of two Normal distributions (Gaussian) is also a **Normal distribution**, $d_S$ will also follow an approximately Normal distribution, with mean $d_D$ and variance:

$$\text{VAR}(d_S) \approx \frac{\text{error}_{s_1}(h1) \cdot (1-\text{error}_{s_1}(h1))}{n_1} + \frac{\text{error}_{s_2}(h2) \cdot (1-\text{error}_{s_2}(h2))}{n_2}$$

**Note:** It can also be shown that the **variance of this distribution is the sum of the variances of** $\text{error}_{s_1}(h1)$ and $\text{error}_{s_2}(h2)$

$$\sigma_{d_S} = \sqrt{\text{VAR}(d_S)} \approx \sqrt{\frac{\text{error}_{s_1}(h1) \cdot (1-\text{error}_{s_1}(h1))}{n_1} + \frac{\text{error}_{s_2}(h2) \cdot (1-\text{error}_{s_2}(h2))}{n_2}}$$
How to do: Two-Tailed Test

If H0 (null hypothesis) holds true, then we must have:

\[ \text{error}_D(h1) = \text{error}_D(h2) \Rightarrow d_D = 0 \]

Which means: although in our experiments we observe that \( \text{error}_S(h1) \neq \text{error}_S(h2) \) (e.g., \( \text{error}_S(h1) < \text{error}_S(h2) \)), the “true” expected value of error differences of \( h1 \) and \( h2 \) on \( D \) is zero.

- To test the likelihood of H0, we have to consider:

\[
    d_S = \text{error}_s(h1) - \text{error}_s(h2) \\
    d_D = \text{error}_D(h1) - \text{error}_D(h2) = 0 \\
    |d_s - d_D| = |d_s - 0| = |d_s| \leq z \sigma_s = -z \sigma_s \leq d_s \leq z \sigma_s \\
    z = \frac{d_s}{\sigma_s}
\]

Error estimates on the samples
\( d_D \) must be zero if H0 holds true
Error bounds in estimating \( d_s \)

We know both \( d_s \) and \( \sigma_{ds} \) so we compute \( z \) and look on a z-table, to see “how many times” our result \( d_s \) is far from the expected mean difference (which is zero according to H0)
How to do: Two-Tailed Test

If the area lies within the non-critical region (i.e. $N \leq 95\%$), the Null Hypothesis $H_0$ is accepted (= there is no significant difference between the two hypotheses)

$|d_s-0|=z\sigma$, $z=|d_s|/\sigma$

Given $z$, using the table we can compute $N$ (the confidence area).

The “common wisdom” is that the acceptance region for $H_0$ is within $-2\sigma$ and $+2\sigma$ ($N \leq 95\%$)

We estimate $d_s$ on the sample

If $H_0$ holds, $d_D=0$
How to do: Two-Tailed Test

- In other terms: the farther our measured distance $d_S$ is from the “expected” distance ($d_D=0$ in case the null hypothesis H0 holds), the less confident we should be in H0.

- For any measured value of $d_S$, the y-axis gives us the probability of observing that value.

- If $d_S$ is farther than $\pm 2\sigma$ from $d_D$, then we may conclude that the probability of having observed the value $d_S$ in case $d_D=0$ is too small. And hence we reject H0 as being very unlikely.
How to do:
Example of Two-Tailed Test

Assume:
- \( d_S = 0.15 \)
- \( \sigma_S = 0.05 \)
- \( z = \frac{d_S}{\sigma} = 3 \)

Then \( \Rightarrow N = 99.87\% \)
How to do:
Example of Two-Tailed Test

Our z-test says that, if $d_D = 0$, the probability to obtain the value $d_S = 0.15$ or more, is less than $0.13\%$ (100-99.87)!! So $H_0$ is very unlikely.

We should reject $H_0$!!
p-value

The p-value is the "probability value" of observing our estimate, *given that H0 holds.*

- **The common wisdom** is to reject the null hypothesis if \( p<0.05 \) (5% the area under the curve of the tails) (same as saying that the estimated value lies outside the \( \pm 2\sigma \) interval, or outside the 95% probability mass around the mean)
- In the previous example, we obtained \( p < 0.0013 \) (0.13%)
How to do: One-Tailed Test

For the 1-tailed test the p-value = 0.05 (5%)

Then, the right area is 0.95 (95%) of the total area

This left area shaded dark blue is .05 of the total area under the curve

In a one-tailed test we test either $h_1 > h_2$ or $h_1 < h_2$

In case $h_1 > h_2$, we state the null hypothesis as follows:

- **H0**: not support that $h_1 > h_2$ (hence $h_1 \leq h_2$)
- **H1**: support $h_1 > h_2$ (in this case we should get an estimate of $d$)
How to do:
Example of One-Right tailed test

Right-tailed test (h2 > h1 (d>0)):

- $\text{error}_{s1}(h1) = x1 = 17.5\%, \quad \text{error}_{s2}(h2) = x2 = 12.4\%, \quad d = 5.1\% (0.51)$
- $n_1 = 50, \quad n_2 = 50$

$$\sigma_{d_s} \approx \sqrt{\frac{\text{error}_{s1}(h1) (1-\text{error}_{s1}(h1))}{n_1} + \frac{\text{error}_{s2}(h2) (1-\text{error}_{s2}(h2))}{n_2}}$$

$$= \sqrt{\frac{0.175 \cdot (1-0.175)}{50} + \frac{0.124 \cdot (1-0.124)}{50}} = \sqrt{0.005}$$

$$z = \frac{(x1 - x2) + (\text{error}_{d}(h1) - \text{error}_{d}(h2))}{0.07} = \frac{(0.051 - 0)}{0.07} = 0.73$$

$$N = 0.2327 = 23.27\% \Rightarrow p > 0.05$$

The null hypothesis is accepted:
- The difference is not large enough to support $h1 < h2$ (p is not lower than 0.05)
Summary: Two-Tailed Test
Comparing Two Learning Algorithms

- Comparing the average accuracy of hypotheses produced by two different ML algorithms is more difficult. Ideally, we want to measure:

\[ E_{S \subset D} (\text{error}_D(L_A(S)) - \text{error}_D(L_B(S))) \]

➢ where \( L_X(S) \) represents the hypothesis learned by learning algorithm \( L_X \) from training data \( S \).

- To accurately estimate this, we need to average over multiple, independent training and test sets.

- However, since labeled data is limited, generally must average over multiple splits of the overall data set into training and test sets (K-Fold Cross Validation, see the beginning of this lesson).
How to use:
K-Fold Cross Validation to evaluate different learning algorithms

Randomly partition dataset $D$ into $k$ disjoint equal-sized ($N$) subsets $P_1...P_k$

For $i$ from 1 to $k$ do:

Use $P_i$ for the test set and remaining data for training

$D_i = (D - P_i)$
$h_A = L_A(D_i)$
$h_B = L_B(D_i)$  (learn models on $D_i$ )

$\delta_i = \text{error}_{P_i}(h_A) - \text{error}_{P_i}(h_B)$  (test models on $P_i$ and compute difference)

Return the average difference in error:

$$\bar{\delta} = \frac{1}{k} \sum_{i=1}^{k} \delta_i$$

Error bound is computed as:

$$\sigma_\delta = \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^{k} (\delta_i - \bar{\delta})^2}$$

$$\bar{\delta} \pm Z \cdot \sigma_\delta$$
Is $L_A$ better than $L_B$?

- K-fold cross-validation improves confidence in our estimate of $\delta$ since we are performing many experiments and computing $\delta$ as the average of $\delta_i$.
- As $K$ grows this average tends to the true mean difference (however we cannot make $K$ too big since individual samples should be large enough for the CLT to apply)
- We can, in any case, apply hypothesis testing as before
Example:
Sample Experimental Results

Which experiment provides better evidence that SystemA is better than SystemB?

<table>
<thead>
<tr>
<th>Experiment 1</th>
<th></th>
<th>Experiment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SystemA</td>
<td>SystemB</td>
</tr>
<tr>
<td>Trial 1</td>
<td>87%</td>
<td>82%</td>
</tr>
<tr>
<td>Trial 2</td>
<td>83%</td>
<td>78%</td>
</tr>
<tr>
<td>Trial 3</td>
<td>88%</td>
<td>83%</td>
</tr>
<tr>
<td>Trial 4</td>
<td>82%</td>
<td>77%</td>
</tr>
<tr>
<td>Trial 5</td>
<td>85%</td>
<td>80%</td>
</tr>
<tr>
<td>Average</td>
<td>85%</td>
<td>80%</td>
</tr>
</tbody>
</table>

Experiment 1 mean δ has σ=0, therefore we have perfect confidence in the estimate of δ.
Experimental Evaluation: Conclusions

- Good experimental methodology is important for evaluating learning methods.
- Important to test on a variety of domains to demonstrate generality for a variety of problems. Testing on 10+ data sets is common.
- Variety of freely available data sources
  - UCI Machine Learning Repository (link)
  - KDD Cup (large data sets for data mining) (link)
  - CoNLL Shared Task (natural language problems) (link)
- Data for real problems is preferable to artificial problems to demonstrate usefulness in real contexts.
- Many available datasets have been subjected to significant feature engineering to make them learnable.
Related links

- Metrics: [link], [link], [link], [link], [link]
- Basic statistics: [link]
- Bias, Variance, and Error: [link], [link], [link]
- Estimator: [link], [link]
- Estimating the accuracy of a hypothesis/Confidence Interval:
  - Text: [link], [link], [link], [link], [link], [link]
  - Video: [link]
  - Hypothesis testing: [link], [link], [link]
  - Z-table: [link]