# Automated Machine Learning
## (AutoML)

M. Lindauer F. Hutter

University of Freiburg

# Lecture 3:
# Evaluation and Visualization

# Where are we? The big picture

- Introduction
- → Background
  - Design spaces in ML
  - → Evaluation and visualization
- Hyperparameter optimization (HPO)
  - Bayesian optimization
  - Other black-box techniques
  - Speeding up HPO with multi-fidelity optimization
- Pentecost (Holiday) – no lecture
- Architecture search I + II
- Meta-Learning
- Learning to learn & optimize
- Beyond AutoML: algorithm configuration and control
- Project announcement and closing

# Learning Goals

After this lecture, you will be able to . . .

- explain the role of outliers in CS/ML
- compare and visualize the performance of different configurations
- compare and visualize the performance of AutoML systems
- explain and correctly apply statistical hypothesis tests

# How CS differs from other empirical sciences

- We have a complete and precise mathematical description of the object under study
- We have complete and precise control of the object under study (and to some degree also the experimental environment)
  - as a result, experiments can be reproduced perfectly

# How CS differs from other empirical sciences

- We have a complete and precise mathematical description of the object under study
- We have complete and precise control of the object under study (and to some degree also the experimental environment)
  - as a result, experiments can be reproduced perfectly
  - what do we need for experiments to be reproducible?

# How CS differs from other empirical sciences

- We have a complete and precise mathematical description of the object under study
- We have complete and precise control of the object under study (and to some degree also the experimental environment)
  - as a result, experiments can be reproduced perfectly
  - what do we need for experiments to be reproducible?
    - Code & dependencies, inputs, environment ($\rightarrow$ VM & cloud)

# How CS differs from other empirical sciences

- We have a complete and precise mathematical description of the object under study
- We have complete and precise control of the object under study (and to some degree also the experimental environment)
  - as a result, experiments can be reproduced perfectly
  - what do we need for experiments to be reproducible?
    - Code & dependencies, inputs, environment ($\rightarrow$ VM & cloud)
- We often have quite cheap experiments
  - price for computers is monotonically decreasing
  - often maximal runtimes of 1h; exception: deep learning (up to a week)
  - compare e.g., experimental physics: 1 week of beam time per year

# How CS differs from other empirical sciences

- We have a complete and precise mathematical description of the object under study
- We have complete and precise control of the object under study (and to some degree also the experimental environment)
  - as a result, experiments can be reproduced perfectly
  - what do we need for experiments to be reproducible?
    - Code & dependencies, inputs, environment ($\rightarrow$ VM & cloud)
- We often have quite cheap experiments
  - price for computers is monotonically decreasing
  - often maximal runtimes of 1h; exception: deep learning (up to a week)
  - compare e.g., experimental physics: 1 week of beam time per year
- We can conduct and analyze experiments fully automatically
  - we can gather large amounts of data quickly (e.g., 100 repetitions)
  - but: don't confuse statistical significance and relevance

# Outliers are quite different in computer experiments

Is the following statement correct?

"As usual in other empirical sciences, we (in CS) should take care to detect and remove outliers before further analysis."

# Outliers are quite different in computer experiments

Is the following statement correct?

"As usual in other empirical sciences, we (in CS) should take care to detect and remove outliers before further analysis."

## Outliers in CS/ML

- outliers should be investigated closely – why is there an outlier?
- outliers are (hopefully) reproducible – narrow down their reason!

# Outliers are quite different in computer experiments

Is the following statement correct?

"As usual in other empirical sciences, we (in CS) should take care to detect and remove outliers before further analysis."

## Outliers in CS/ML

- outliers should be investigated closely – why is there an outlier?
- outliers are (hopefully) reproducible – narrow down their reason!
    - **Algorithm:** When characterizing performance of an optimization algorithm, outliers with poor values can indicate (deep) local optima

# Outliers are quite different in computer experiments

Is the following statement correct?

"As usual in other empirical sciences, we (in CS) should take care to detect and remove outliers before further analysis."

## Outliers in CS/ML

- outliers should be investigated closely – why is there an outlier?
- outliers are (hopefully) reproducible – narrow down their reason!
    - **Algorithm:** When characterizing performance of an optimization algorithm, outliers with poor values can indicate (deep) local optima
    - **Environment:** Outliers can indicate a problem with the environment (e.g., file system or network issues)

# Outliers are quite different in computer experiments

Is the following statement correct?

"As usual in other empirical sciences, we (in CS) should take care to detect and remove outliers before further analysis."

## Outliers in CS/ML

- outliers should be investigated closely – why is there an outlier?
- outliers are (hopefully) reproducible – narrow down their reason!
  - **Algorithm:** When characterizing performance of an optimization algorithm, outliers with poor values can indicate (deep) local optima
  - **Environment:** Outliers can indicate a problem with the environment (e.g., file system or network issues)
  - **Datasets:** When characterizing cost across a distribution of datasets, outliers with small values can indicate trivial datasets.

# Lecture Overview

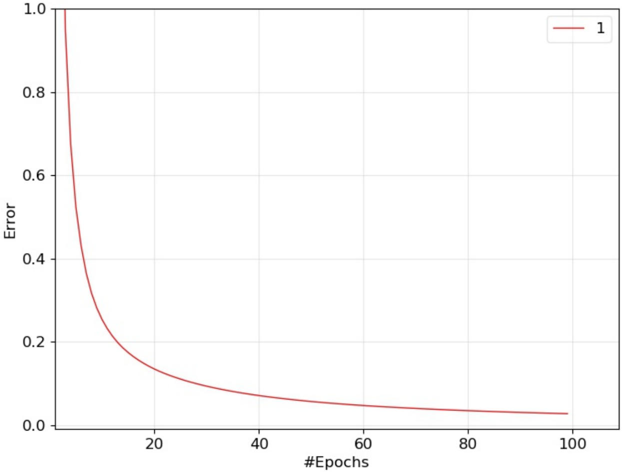
1. Visualization of Configuration Performance
2. Visualization of AutoML Performance
3. Statistical Hypothesis Testing

# Setup

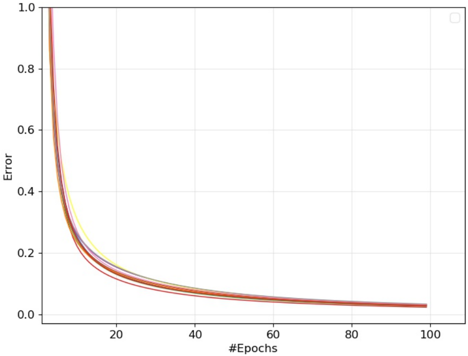
For the following slides, we have used the following setup:

- Model: simple MLP (from sklearn)
  with 2 layers with 128 neurons and 64 neurons, resp.
- Dataset: Digits
- Setting 1: learning rate of $0.001$
- Setting 2: learning rate of $0.01$

# A Single Learning Curve (Setting 1)

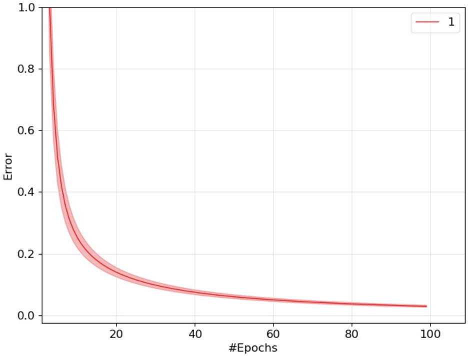


# 10 Learning Curves (Setting 1)



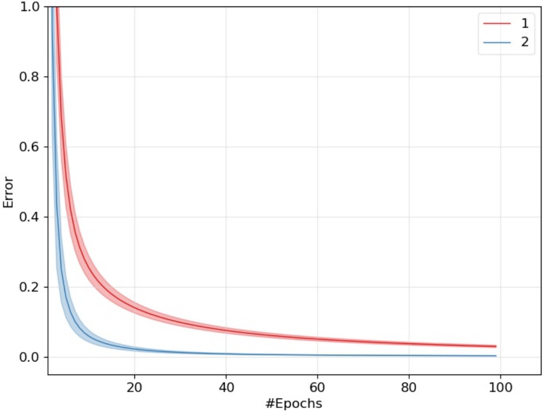

- ⇝ Deep learning (and most other ML algorithms) are non-deterministc
- ⇝ Measure performance more than once and estimate noise-level

# Aggregated Learning Curves (Setting 1)




- Plot mean and stdev (shaded area) across $n$ (here $100$) random seeds
- Only use mean and stdevarcSepA=[0pt] if noise is somehow normal distributed
- alternatives are the mean+standard error or median+25/75-percentiles

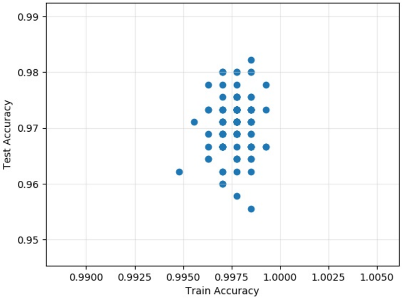
# Comparing Learning Curves




- If uncertainties (shaded area) overlap, results might not be statistically significant but due to noise

# Scatter Plot: Train vs. Test

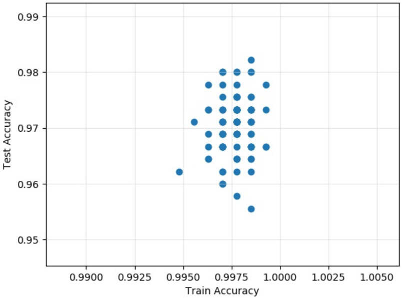
Setting 1




- perfect would be if train and test score are correlated
- here at least not anti-correlated

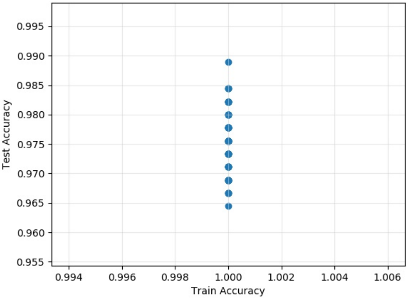
# Scatter Plot: Train vs. Test

Setting 1



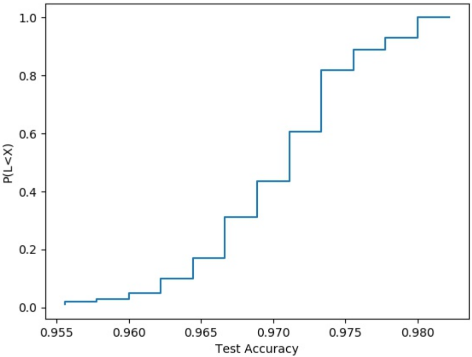

- perfect would be if train and test score are correlated
- here at least not anti-correlated

Setting 2




- already perfect training score
- generalization nevertheless noisy

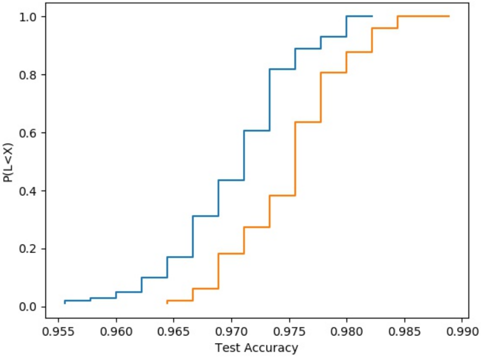
# eCDF: Distribution of Performance (Setting 1)




How to compute an empirical CDF:

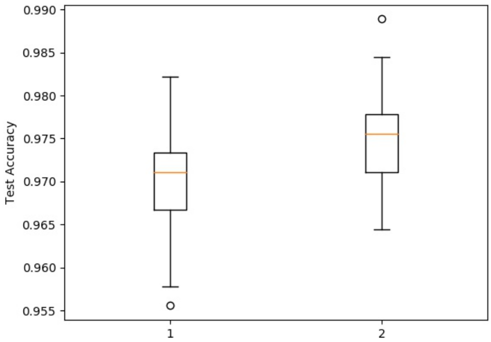
1. X: sorted error scores
2. Y: $[1 \dots \#\text{points}]/\#\text{points}$
3. step_function(X,Y)

# eCDF: Distribution of Performance



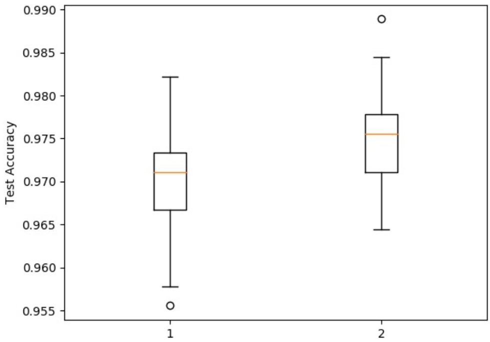

Which curve corresponds to which setting?

# Box Plot: Comparing Two Configurations




- alternative: violin plots

# Box Plot: Comparing Two Configurations




- alternative: violin plots
- if you have paired populations (e.g., configurations evaluated on different datasets), you should generate boxplots with $\mathcal{L}/\mathcal{L}_{\text{baseline}}$

$\rightsquigarrow$ insight on how many datasets one of the two performed better

# Lecture Overview

1. Visualization of Configuration Performance
2. Visualization of AutoML Performance
3. Statistical Hypothesis Testing

# AutoML Systems over Time

- Don't only measure the performance of AutoML systems for a fixed budget because:
  1. A priori, it is unknown how long a user will run an AutoML system
  2. Some systems perform well for small budgets and some others need more time before performing well
     - BO-based systems often perform as good as random search in the beginning, but perform very well later on

# AutoML Systems over Time

- Don't only measure the performance of AutoML systems for a fixed budget because:
  1. A priori, it is unknown how long a user will run an AutoML system
  2. Some systems perform well for small budgets and some others need more time before performing well
     - BO-based systems often perform as good as random search in the beginning, but perform very well later on

$\rightsquigarrow$ AutoML systems should have a good anytime performance

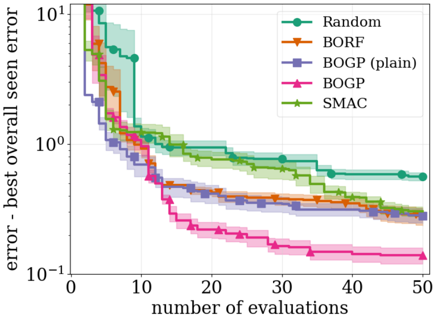
# AutoML Systems over Time

- Don't only measure the performance of AutoML systems for a fixed budget because:
  1. A priori, it is unknown how long a user will run an AutoML system
  2. Some systems perform well for small budgets and some others need more time before performing well
     - BO-based systems often perform as good as random search in the beginning, but perform very well later on

$\rightsquigarrow$ AutoML systems should have a good anytime performance

- Recommendation: plot performance (e.g., error) over time
  - Similar to learning curves of DNNs

# Aggregated AutoML Systems over Time




- Different HPO systems minimizing the error of an MLP on MNIST
- How to do it:
  - Run each systems several times and always log its current incumbent
  - Plot mean and stdev for each system after each incumbent update
  - Use step functions, because linear interpolation would be too optimistic

# Aggregated AutoML Systems over Time and Datasets

- An AutoML system should not only perform well on a single dataset but on many datasets
- To summarize and compare the performance across a set of datasets, we can plot the performance over time and across datasets

# Aggregated AutoML Systems over Time and Datasets

- An AutoML system should not only perform well on a single dataset but on many datasets
- To summarize and compare the performance across a set of datasets, we can plot the performance over time and across datasets
- Ways to do it:
  1. Average loss metric (e.g., error) across datasets
     - Problem: different scales of errors on different datasets

# Aggregated AutoML Systems over Time and Datasets

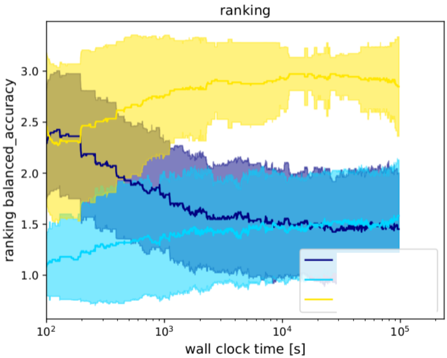
- An AutoML system should not only perform well on a single dataset but on many datasets
- To summarize and compare the performance across a set of datasets, we can plot the performance over time and across datasets
- Ways to do it:
  1. Average loss metric (e.g., error) across datasets
     - Problem: different scales of errors on different datasets
  2. Normalize loss metric across datasets,
     e.g., best seen loss relates to a zero cost
     - Problem: Minor improvements ($\leq 0.1\%$) might look substantial

# Aggregated AutoML Systems over Time and Datasets

- An AutoML system should not only perform well on a single dataset but on many datasets
- To summarize and compare the performance across a set of datasets, we can plot the performance over time and across datasets
- Ways to do it:
  1. Average loss metric (e.g., error) across datasets
     - Problem: different scales of errors on different datasets
  2. Normalize loss metric across datasets,
     e.g., best seen loss relates to a zero cost
     - Problem: Minor improvements ($\leq 0.1\%$) might look substantial

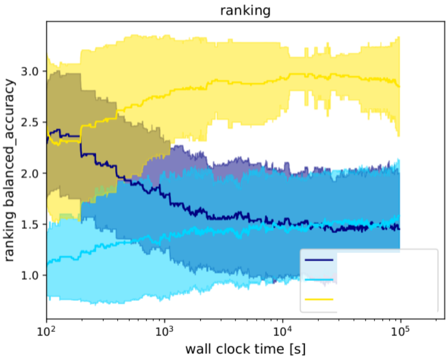
$\rightsquigarrow$ There is no way around that you have some kind of information loss

# Ranked AutoML Systems over Time and Datasets




- Remark: x-axis should be on log-scale

# Ranked AutoML Systems over Time and Datasets




- Remark: x-axis should be on log-scale
- Ranks avoid the problem of different scales
- Problem: we don't know whether a better ranking relates to a substantial improvement of the actual cost metric

# Uncertainties in Ranking Plots

Given:

- $k$ systems you want compare
- $n$ runs of each system
- $d$ datasets

# Uncertainties in Ranking Plots

Given:

- $k$ systems you want compare
- $n$ runs of each system
- $d$ datasets

## Across Datasets ($d > 1$)

1. for each system and dataset, average performance across the $n$ runs
2. compute ranking on each datasets
3. compute mean and stdev across rankings

# Uncertainties in Ranking Plots

Given:

- $k$ systems you want compare
- $n$ runs of each system
- $d$ datasets

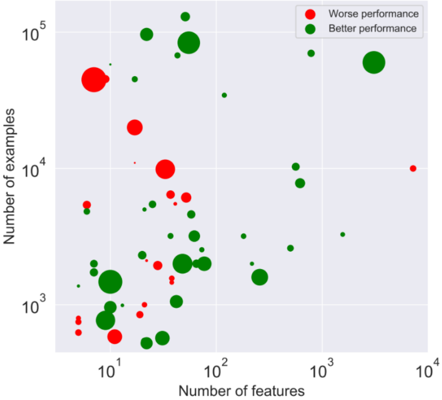
## Across Datasets ($d > 1$)

1. for each system and dataset, average performance across the $n$ runs
2. compute ranking on each datasets
3. compute mean and stdev across rankings

## On a single dataset ($d = 1$)

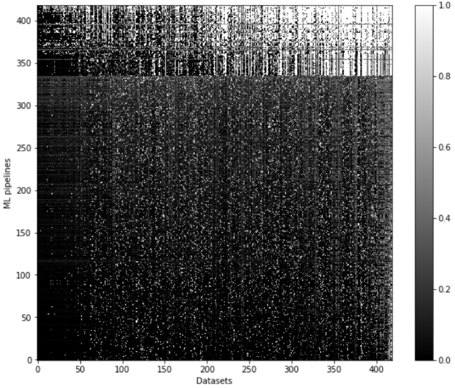
1. for each system, sample a single run out of the $n$ runs
2. compute ranking on these samples
3. repeat 1. and 2. at least $10\,000$ times to obtain many rankings
4. compute mean and stdev across rankings

# Scatter Plot: Meta Features




- Compare two systems (or 1 vs rest)
- each dot is a dataset
- color encoding better or worse
- size indicates performance difference
- meta-features (e.g., #features, #samples) on axes
  - using PCA on a larger set of meta-features leads to algorithm footprints [Smith-Miles et al. 2012]

# Heatmaps: Comparing several Configurations




- compare many configurations on many datasets
- can reveal:
  - homogeneity if smooth transition from hard to easy
  - heterogeneity if stripes exist (here the case)
- Remark: datasets and configurations should be sorted according to their average cost

# Lecture Overview

# Background: statistical hypothesis tests

- When we have a lot of data, we need to summarize it
  - But we already saw that summarization hides a lot of data
  - Ideally, we want to draw high-level conclusions
    (e.g., "A outperforms B on datasets of type X")

# Background: statistical hypothesis tests

- When we have a lot of data, we need to summarize it
  - But we already saw that summarization hides a lot of data
  - Ideally, we want to draw high-level conclusions (e.g., "A outperforms B on datasets of type X")
- Problem: we only have a finite number of observations
  - Can we attribute observed performance differences to chance?
  - Are we reasonably sure that a claim we make is reproducible?
  - $\rightsquigarrow$ Statistical tests can help

# Statistical hypothesis testing

1. Define initial research hypothesis

# Statistical hypothesis testing

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
    - Alternative hypothesis should be your research hypothesis

# First example: Courtroom Tiral

- A prosecutor tries to prove the guilt of the defendant
- $H_0$: 
- $H_1$: 

# First example: Courtroom Tiral

- A prosecutor tries to prove the guilt of the defendant
- $H_0$: The defendant is not guilty
  - Accepted for the moment
    ("not guilty as long as their guilt is not proven")
- $H_1$: The defendant is guilty
  - prosecutor hopes to support that

# First example: Courtroom Tiral

- A prosecutor tries to prove the guilt of the defendant
- $H_0$: The defendant is not guilty
  - Accepted for the moment ("not guilty as long as their guilt is not proven")
- $H_1$: The defendant is guilty
  - prosecutor hopes to support that

| | Truly not guilty | Truly guilty |
|---|---|---|
| Found not guilty | Acquittal | Type II Error |
| Found guilty | Type I Error | Conviction |

$\rightsquigarrow$ We want to minimize Type I error!

# Statistical hypothesis testing (cont'd)

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
   - Alternative hypothesis should be your research hypothesis
3. Consider statistical assumptions
   - E.g., is your data Gaussian distributed?

# Statistical hypothesis testing (cont'd)

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
   - Alternative hypothesis should be your research hypothesis
3. Consider statistical assumptions
   - E.g., is your data Gaussian distributed?
4. Decide test and test statistic $T$
   - The correct test depends on your statistical assumptions.
   - Typically: if you use more assumptions, the test is more powerful (i.e., less Type-I error)

# Statistical hypothesis testing (cont'd)

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
   - Alternative hypothesis should be your research hypothesis
3. Consider statistical assumptions
   - E.g., is your data Gaussian distributed?
4. Decide test and test statistic $T$
   - The correct test depends on your statistical assumptions.
   - Typically: if you use more assumptions, the test is more powerful (i.e., less Type-I error)
5. Decide significance level $\alpha$
   (i.e., acceptable Type-I error to reject null hypothesis)

# Statistical hypothesis testing (cont'd)

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
   - Alternative hypothesis should be your research hypothesis
3. Consider statistical assumptions
   - E.g., is your data Gaussian distributed?
4. Decide test and test statistic $T$
   - The correct test depends on your statistical assumptions.
   - Typically: if you use more assumptions, the test is more powerful (i.e., less Type-I error)
5. Decide significance level $\alpha$ (i.e., acceptable Type-I error to reject null hypothesis)
6. Compute observed $t_{obs}$ of test statistic $T$
7. Calculate $p$-value given $t_{obs}$
   - i.e., probability under the null hypothesis of sampling a test statistic as extreme as observed (probability of Type-I error)

# Statistical hypothesis testing (cont'd)

1. Define initial research hypothesis
2. Derive null $H_0$ and alternative $H_1$ hypothesis
   - Alternative hypothesis should be your research hypothesis
3. Consider statistical assumptions
   - E.g., is your data Gaussian distributed?
4. Decide test and test statistic $T$
   - The correct test depends on your statistical assumptions.
   - Typically: if you use more assumptions, the test is more powerful (i.e., less Type-I error)
5. Decide significance level $\alpha$ (i.e., acceptable Type-I error to reject null hypothesis)
6. Compute observed $t_{obs}$ of test statistic $T$
7. Calculate $p$-value given $t_{obs}$
   - i.e., probability under the null hypothesis of sampling a test statistic as extreme as observed (probability of Type-I error)
8. If $p < \alpha$, reject null hypothesis in favor of alternative hypothesis
   - If $p > \alpha$, it doesn't tell you anything about the null hypothesis!

# Second example for a statistical test

- Claim: “the students in this class are more intelligent than average”

# Second example for a statistical test

- Claim: "the students in this class are more intelligent than average"

- Null Hypothesis $H_0$: $\mu = 100$ ($\mu$ is the population mean of this class)
- Alternative Hypothesis $H_1$: $\mu > 100$ (one-sided hypothesis)

# Second example for a statistical test

- Claim: "the students in this class are more intelligent than average"

- Null Hypothesis $H_0$: $\mu = 100$ ($\mu$ is the population mean of this class)
- Alternative Hypothesis $H_1$: $\mu > 100$ (one-sided hypothesis)

- IQ values are known to be normally distributed with $X \sim \mathcal{N}(100, 15)$
  - $\rightarrow$ statistical assumption

# Second example for a statistical test

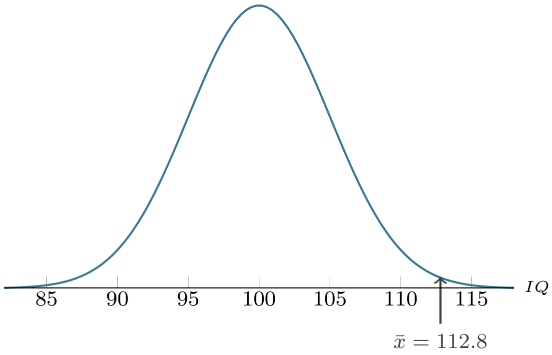
- Claim: "the students in this class are more intelligent than average"
- Null Hypothesis $H_0$: $\mu = 100$ ($\mu$ is the population mean of this class)
- Alternative Hypothesis $H_1$: $\mu > 100$ (one-sided hypothesis)
- IQ values are known to be normally distributed with $X \sim \mathcal{N}(100, 15)$
  - $\rightarrow$ statistical assumption
- Let's say we observed IQ values $x_i$ of 9 students in the class:
  - $\{x_1, \dots, x_9\} = \{116, 128, 125, 119, 89, 99, 105, 116, 118\}$.
  - The sample mean is $\bar{x} = 112.8$
  - Does this data support the claim?

# Example continued

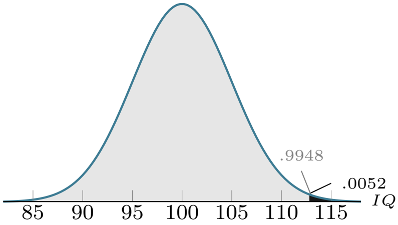
- Distribution of the test statistic
  - Under $H_0$, we know that each $x_i \sim \mathcal{N}(100, 15)$
  - The test statistic that we measure is the sample mean $\bar{x} = \frac{1}{9}\sum_{i=1}^{9} x_i$

# Example continued

- Distribution of the test statistic
  - Under $H_0$, we know that each $x_i \sim \mathcal{N}(100, 15)$
  - The test statistic that we measure is the sample mean $\bar{x} = \frac{1}{9}\sum_{i=1}^{9} x_i$
  - Under $H_0$, the distribution of $\bar{x}$ is $\mathcal{N}(100, 15/\sqrt{9})$
    - Our observation $\bar{x} = 112.8$ is quite extreme under that distribution

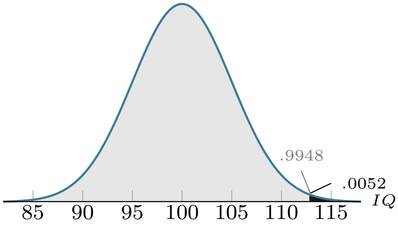


# General principle



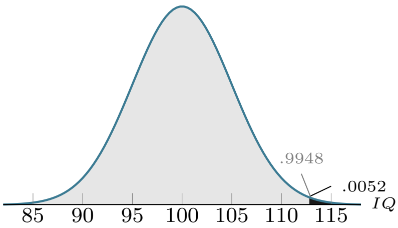

- Compare the test statistic (here: $\bar{x}$) to its sampling distribution under $H_0$

# General principle



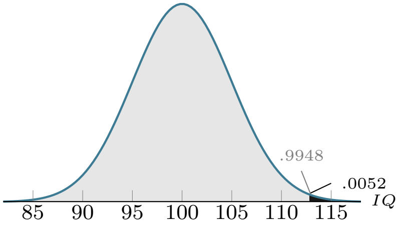

- Compare the test statistic (here: $\bar{x}$) to its sampling distribution under $H_0$
- P-value: probability $p$ of observing values at least as extreme as $\bar{x}$

# General principle




- Compare the test statistic (here: $\bar{x}$) to its sampling distribution under $H_0$
- P-value: probability $p$ of observing values at least as extreme as $\bar{x}$
- Compare $p$ to pre-defined confidence level $\alpha$ (usually $\alpha = 0.05$); if $p < \alpha$, reject $H_0$

# General principle




- Compare the test statistic (here: $\bar{x}$) to its sampling distribution under $H_0$
- P-value: probability $p$ of observing values at least as extreme as $\bar{x}$
- Compare $p$ to pre-defined confidence level $\alpha$ (usually $\alpha = 0.05$); if $p < \alpha$, reject $H_0$
- With $\alpha = 0.01$, would we reject $H_0$ in this case?

# Summary of example

- We just used a so-called $Z$-test
- $H_0$: $\mu = \mu_0$, $H_1$: $\mu > \mu_0$
- Assumptions: $X \sim \mathcal{N}(\mu, \sigma^2)$ , with known $\mu$ and $\sigma^2$

# Summary of example

- We just used a so-called $Z$-test
- $H_0$: $\mu = \mu_0$, $H_1$: $\mu > \mu_0$
- Assumptions: $X \sim \mathcal{N}(\mu, \sigma^2)$ , with known $\mu$ and $\sigma^2$
- Test statistic: sample mean $\bar{x}$; evaluate under $\mathcal{N}(\mu = \mu_0, s = \sigma^2/\sqrt{n})$

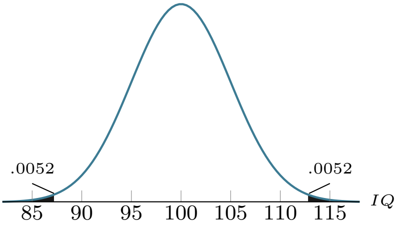
# Summary of example

- We just used a so-called $Z$-test
- $H_0$: $\mu = \mu_0$, $H_1$: $\mu > \mu_0$
- Assumptions: $X \sim \mathcal{N}(\mu, \sigma^2)$ , with known $\mu$ and $\sigma^2$
- Test statistic: sample mean $\bar{x}$; evaluate under $\mathcal{N}(\mu = \mu_0, s = \sigma^2/\sqrt{n})$
- Equivalent: compute the Z-statistic: $Z = (\bar{x} - \mu_0)/s$ and evaluate cumulative distribution $\Phi(Z)$ of $Z$ under $\mathcal{N}(0, 1)$

# Summary of example

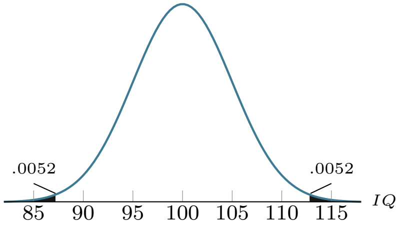
- We just used a so-called $Z$-test
- $H_0$: $\mu = \mu_0$, $H_1$: $\mu > \mu_0$
- Assumptions: $X \sim \mathcal{N}(\mu, \sigma^2)$ , with known $\mu$ and $\sigma^2$
- Test statistic: sample mean $\bar{x}$; evaluate under $\mathcal{N}(\mu = \mu_0, s = \sigma^2/\sqrt{n})$
- Equivalent: compute the Z-statistic: $Z = (\bar{x} - \mu_0)/s$ and evaluate cumulative distribution $\Phi(Z)$ of $Z$ under $\mathcal{N}(0, 1)$
  - There are standard tables to look up $\Phi(Z)$ for different values of $Z$
  - Nowadays, there are standard libraries to compute $\Phi(Z)$

# Two-sided tests



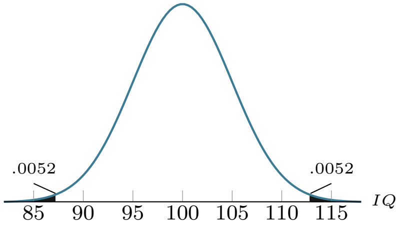

- Similar to one-sided tests, but testing for extreme values in both tails
- Example Z-test: two-sided alternative hypothesis $H_1$: $\mu \neq \mu_0$

# Two-sided tests




- Similar to one-sided tests, but testing for extreme values in both tails
- Example Z-test: two-sided alternative hypothesis $H_1$: $\mu \neq \mu_0$
- Compute $Z = (\bar{x} - \mu_0)/s$ as before
- Compute $p$-value as $p = 2\Phi(Z)$, to account for both tails

# Two-sided tests




- Similar to one-sided tests, but testing for extreme values in both tails
- Example Z-test: two-sided alternative hypothesis $H_1$: $\mu \neq \mu_0$
- Compute $Z = (\bar{x} - \mu_0)/s$ as before
- Compute $p$-value as $p = 2\Phi(Z)$, to account for both tails
- With $\alpha = 0.01$, would we reject $H_0$ in this case?

# General points about statistical hypothesis tests

- What if $p > \alpha$?
  - Failure to reject $H_0$
  - This does not mean that we accept $H_0$

# General points about statistical hypothesis tests

- What if $p > \alpha$?
  - Failure to reject $H_0$
  - This does not mean that we accept $H_0$

- Beware (i): most tests make some assumptions
  - E.g., $Z$-test and popular $t$-test assume normality
  - Our data is often far from normally-distributed
    - ⇝ E.g., exponential runtime distributions of local search optimizers
    - ⇝ E.g., distribution of fitting a neural network with different random seeds is not well studied

# General points about statistical hypothesis tests

- What if $p > \alpha$?
  - Failure to reject $H_0$
  - This does not mean that we accept $H_0$

- Beware (i): most tests make some assumptions
  - E.g., $Z$-test and popular $t$-test assume normality
  - Our data is often far from normally-distributed
    - $\rightsquigarrow$ E.g., exponential runtime distributions of local search optimizers
    - $\rightsquigarrow$ E.g., distribution of fitting a neural network with different random seeds is not well studied

- Beware (ii): if you use cross-validation, observations are not independent (you cannot apply statistical tests that assume independence)

# The Wilcoxon rank-sum test for non-normal data

- Compare the distributions of random variables $X$ and $Y$
  - based on samples $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$
- Assumptions
  - All of $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are independent from each other
  - Responses are ordinal (we can compare them)

# The Wilcoxon rank-sum test for non-normal data

- Compare the distributions of random variables $X$ and $Y$
  - based on samples $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$
- Assumptions
  - All of $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are independent from each other
  - Responses are ordinal (we can compare them)
- $H_0$: $P(X > Y) = P(Y > X)$

# The Wilcoxon rank-sum test for non-normal data

- Compare the distributions of random variables $X$ and $Y$
  - based on samples $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$
- Assumptions
  - All of $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are independent from each other
  - Responses are ordinal (we can compare them)
- $H_0$: $P(X > Y) = P(Y > X)$
- $H_1$: $P(X > Y) > P(Y > X)$ (one-sided)

# The Wilcoxon rank-sum test for non-normal data

- Compare the distributions of random variables $X$ and $Y$
  - based on samples $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$
- Assumptions
  - All of $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are independent from each other
  - Responses are ordinal (we can compare them)
- $H_0$: $P(X > Y) = P(Y > X)$
- $H_1$: $P(X > Y) > P(Y > X)$ (one-sided)
- Test statistic
  - Order all elements $x_i$ and $y_j$ and give them ranks (1 for smallest)
  - Compute the sum of ranks of $x_i$ and of $y_j$

# The Wilcoxon rank-sum test for non-normal data

- Compare the distributions of random variables $X$ and $Y$
  - based on samples $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$
- Assumptions
  - All of $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$ are independent from each other
  - Responses are ordinal (we can compare them)
- $H_0$: $P(X > Y) = P(Y > X)$
- $H_1$: $P(X > Y) > P(Y > X)$ (one-sided)
- Test statistic
  - Order all elements $x_i$ and $y_j$ and give them ranks (1 for smallest)
  - Compute the sum of ranks of $x_i$ and of $y_j$
- Under $H_0$, the distribution of that test statistic is known
  $\rightsquigarrow$ can evaluate how extreme the observed test statistic is

# The permutation test: another test for non-normal data

- Framework for testing several types of claims
- E.g., $H_0$: $X$ and $Y$ have equal means
- Test statistic: $t = \frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{m}\sum_{j=1}^{m} y_j$

# The permutation test: another test for non-normal data

- Framework for testing several types of claims
- E.g., $H_0$: $X$ and $Y$ have equal means
- Test statistic: $t = \frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{m}\sum_{j=1}^{m} y_j$
- The sampling distribution to compare $t$ against:
  - Put $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_m\}$ into a single pool
  - $S = []$; repeat, e.g., 10 000 times
    - draw a random permutation & permute pool with it
    - add test statistic over permuted pool to $S$

# The permutation test: another test for non-normal data

- Framework for testing several types of claims
- E.g., $H_0$: $X$ and $Y$ have equal means
- Test statistic: $t = \frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{m}\sum_{j=1}^{m} y_j$
- The sampling distribution to compare $t$ against:
  - Put $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_m\}$ into a single pool
  - $S = []$; repeat, e.g., 10 000 times
    - draw a random permutation & permute pool with it
    - add test statistic over permuted pool to $S$
- $p$-value: percentile of $s$ in $S$:
  fraction of samples $s$ in $S$ with $s < t$

# Paired vs. unpaired tests

- if you have two unsorted populations (e.g., repeated measurements with different random seeds), you use an unpaired test (as discussed above)
- if you can attribute a measurement to concrete objects (e.g., measurements on different datasets), you use a paired test
  - paired test are more powerful
    (i.e., higher confidence for the same amount of data)
- Examples for paired permutation tests
  - Wilcoxon signed rank test
  - paired permutation test
    - permutation of measurement pairs (e.g., same dataset)

# Paired Permutation Test

Given:

- ordered observations $[x_1, ...x_n]$ in $X$ (resp. in $Y$) where each observation is attributed to concrete objects

# Paired Permutation Test

Given:

- ordered observations $[x_1, ...x_n]$ in $X$ (resp. in $Y$) where each observation is attributed to concrete objects
- S = []; repeat, e.g., 10 000 times
  1. $X' = []$ and $Y' = []$
  2. for each pair of observations $\langle x_i, y_i \rangle$ sample whether (i) you put $x_i$ into $X'$ and $y_i$ into $Y'$ or (ii) $x_i$ into $Y'$ and $y_i$ into $X'$.
  3. add test statistic based on $X'$ and $Y'$ to $S$

# Multiple Testing Correction

- To compare many systems, we apply statistical tests several time (once for each pair of systems)
- Each test induces some error (bounded by $\alpha$)

$\leadsto$ the error probability for $k$ tests is bounded by $1 - (1 - \alpha)^k$

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $1 - (1 - 0.05)^k$ | 0.05 | 0.10 | 0.14 | 0.19 | 0.23 | 0.27 | 0.30 |

# Multiple Testing Correction

- To compare many systems, we apply statistical tests several time (once for each pair of systems)
- Each test induces some error (bounded by $\alpha$)

$\rightsquigarrow$ the error probability for $k$ tests is bounded by $1-(1-\alpha)^k$

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $1-(1-0.05)^k$ | 0.05 | 0.10 | 0.14 | 0.19 | 0.23 | 0.27 | 0.30 |

- Better would be if the error probability would not increase with $k$

$\rightsquigarrow$ multiple testing correction

# Multiple Testing Correction

- To compare many systems, we apply statistical tests several time (once for each pair of systems)
- Each test induces some error (bounded by $\alpha$)

$\leadsto$ the error probability for $k$ tests is bounded by $1 - (1 - \alpha)^k$

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $1 - (1 - 0.05)^k$ | 0.05 | 0.10 | 0.14 | 0.19 | 0.23 | 0.27 | 0.30 |

- Better would be if the error probability would not increase with $k$

$\leadsto$ multiple testing correction

- Bonferronie testing correction: $\alpha_{\text{local}} = \alpha_{\text{global}}/k$

# Multiple Testing Correction

- To compare many systems, we apply statistical tests several time (once for each pair of systems)
- Each test induces some error (bounded by $\alpha$)

$\leadsto$ the error probability for $k$ tests is bounded by $1-(1-\alpha)^k$

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $1-(1-0.05)^k$ | 0.05 | 0.10 | 0.14 | 0.19 | 0.23 | 0.27 | 0.30 |

- Better would be if the error probability would not increase with $k$

$\leadsto$ multiple testing correction

- Bonferronie testing correction: $\alpha_{\text{local}} = \alpha_{\text{global}}/k$
  - very conservative approach
  - there exist other, less conservative approaches

# Checklist for good scientific practices

Incomplete list of good scientific practices (specifically for students):

1. keep track of your code and design decisions (on all levels)
2. measure performance of randomized algorithms multiple times and show uncertainty of results
3. apply suitable statistical tests to check for significance
4. choose a metric that is relevant for the application
5. always add legends, axis labels and so on in plots
6. be aware of other research results
7. avoid peeking at your test data
    - no cherry-picking!

# Learning Goals

Now, you should be able to . . .

- explain the role of outliers in CS/ML
- compare and visualize the performance of different configurations
- compare and visualize the performance of AutoML systems
- explain and correctly apply statistical hypothesis tests

# Literature [These are links]

- [P. Langley 1988. Machine Learning as an Experimental Science]
- [C. Drummod 2006. Machine Learning as an Experimental Science (Revisited)]
- [J. Demsar 2006. Statistical Comparisons of Classifiers over Multiple Data Sets]
- [Wilson et al. 2014. Best Practices for Scientific Computing]
- [Wilson et al. 2017. Good enough practices in scientific computing]