Best Practice in Machine Learning for Computer Vision

Christoph H. Lampert
Institute of Science and Technology Austria (ISTA)

Vision and Sports Summer School 2022
July 25-29, 2022 — Prague, Czech Republic

I S T AUSTRIA
Institute of Science and Technology

Slides and exercise data: `https://cvml.ist.ac.at/`

About ISTA (Institute of Science and Technology Austria, near Vienna)

Research topics in ISTA's ELLIS Unit:

- transfer/trustworthy learning, computer vision (me)
- scalable deep learning (Alistarh group)
- theory of deep learning (Mondelli group)
- Bayesian methods in genomics (Robinson group)
- machine learning for material science (Cheng group)
- discrete optimization (Kolmogorov group)

ISTA Graduate School

- two-phase,English language PhD program
- full salary positions

Other possibilities: internships, postdocs, faculty, ...

- visit: `cvml.ist.ac.at` or ask me during a break

Schedule

Computer Vision

Machine Learning for Computer Vision

Some Background

Best Practice and How to Avoid Pitfalls

Computer Vision

Playground Prediction Competition

CIFAR-10 - Object Recognition in Images

Identify the subject of 60,000 labeled images

Kaggle · 231 teams · 8 years ago

Overview Data Code Discussion Leaderboard Rules

Join Competition

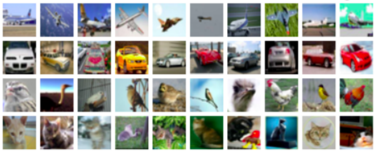
Overview

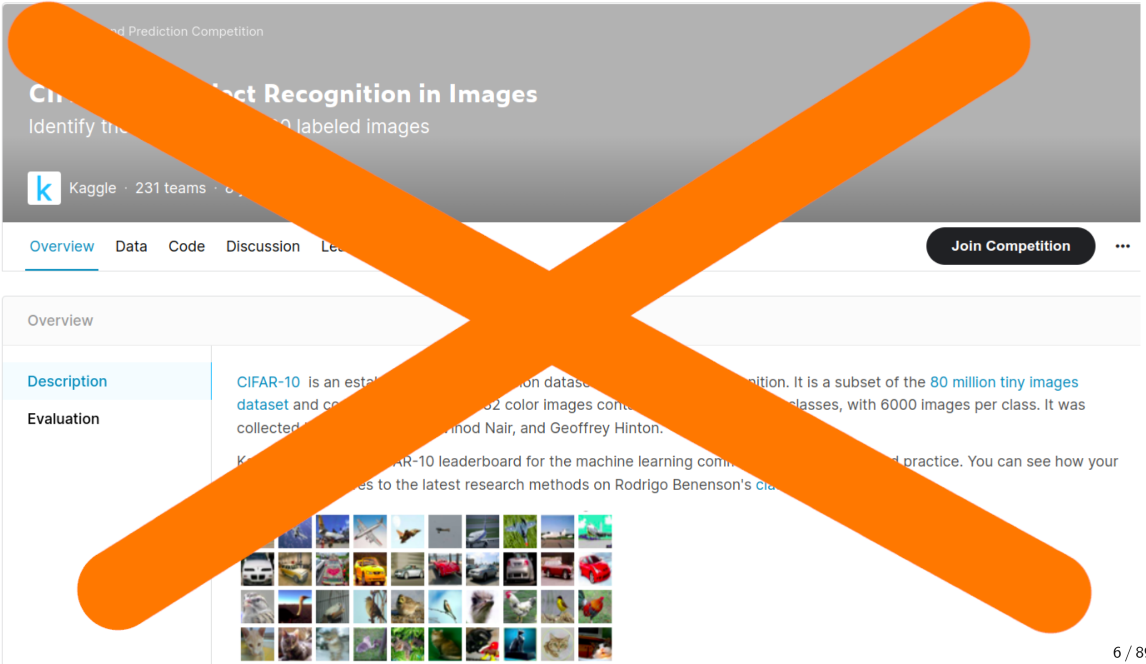
Description

Evaluation

CIFAR-10 is an established computer-vision dataset used for object recognition. It is a subset of the 80 million tiny images dataset and consists of 60,000 32×32 color images containing one of 10 object classes, with 6000 images per class. It was collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton.

Kaggle is hosting a CIFAR-10 leaderboard for the machine learning community to use for fun and practice. You can see how your approach compares to the latest research methods on Rodrigo Benenson's classification results page.




Prediction Competition
Recognition in Images
labeled images
Kaggle · 231 teams
Overview
Data
Code
Discussion
Join Competition
Overview
Description
Evaluation
CIFAR-10 is an esta
ition. It is a subset of the 80 million tiny images
dataset and c
classes, with 6000 images per class. It was
collected
Vinod Nair, and Geoffrey Hinton.
AR-10 leaderboard for the machine learning com
practice. You can see how your
es to the latest research methods on Rodrigo Benenson's

Computer Vision with a purpose

Robotics (e.g. *autonomous cars*)

Healthcare (e.g. *visual aids*)

Commerce (e.g. *Amazon Go*))

Consumer Electronics
(e.g. *human computer interaction*)

Augmented Reality
(e.g. *HoloLens, Pokemon Go*)

Security
(e.g. *Face Unlock*)

Amazon Go Store image: amazon.com

Computer vision systems should

- perform interesting/relevant tasks, e.g. drive a car

For this, they need to

- work in the real world,
- interact with non-experts,
- do what they are supposed to do without failures.

Computer vision systems should

- perform interesting/relevant tasks, e.g. drive a car

For this, they need to

- work in the real world,
- interact with non-experts,
- do what they are supposed to do without failures.

Machine learning is not particularly good at those...

- we have to be extra careful to know what we're going!

[Image: https://xkcd.com/1838/]

Machine Learning for Computer Vision

Example: Solving a Computer Vision Task using Machine Learning

Step 0) Sanity checks...

Step 1) Decide what exactly you want

Step 2) Collect and annotate data

Step 3) Model training

Step 4) Model evaluation

Example: Solving a Computer Vision Task using Machine Learning

Step 0) Sanity checks...

Step 1) Decide what exactly you want

Step 2) Collect and annotate data

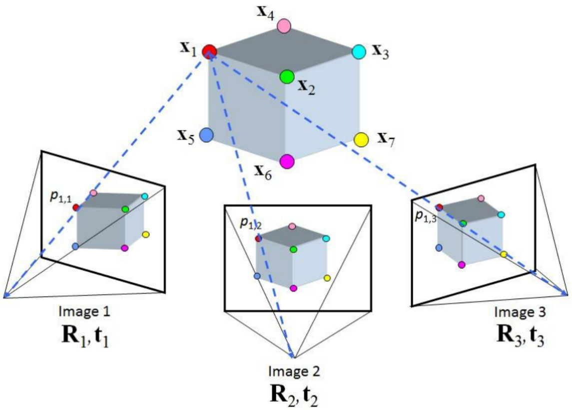
Step 3) Model training

Step 4) Model evaluation

This lecture:

- **Question 1: why do we do it like this?**
- **Question 2: what can go wrong, and how to avoid that?**

Create a 3D model from many 2D correspondences



Sanity Check: Is Machine Learning the right way to solve it?

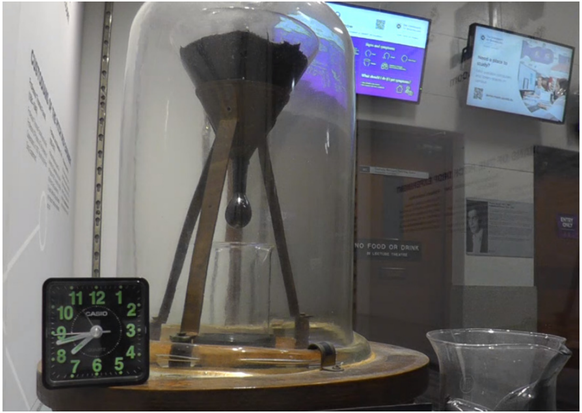
- Can you think of an algorithmic way to solve the problem?

  Yes: optimization with geometric constraints!

→ not a strong case for using machine learning. ✗

images: `http://vision.in.tum.de/research/image-based_3d_reconstruction/multiviewreconstruction`

Example Task: Identify Pitch Drop Events from a Video Feed

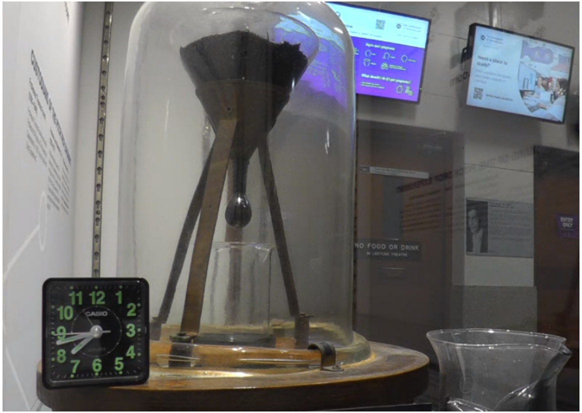


Sanity Check: Is Machine Learning the right way to solve it?

- Can you think of an algorithmic way to solve the problem? **No.**

image: `http://www.thetenthwatch.com/feed/`

Example Task: Identify Pitch Drop Events from a Video Feed



Sanity Check: Is Machine Learning the right way to solve it?

- Can you think of an algorithmic way to solve the problem? **No.**
- Can you get sufficient data? **No.**

→ not a strong case for using machine learning. ✗

image: http://www.thetenthwatch.com/feed/

Sanity Check: Is Machine Learning the right way to solve it?

- Can you think of an algorithmic way to solve the problem? **No.**

Image: RobeSafe Driver Monitoring Video Dataset (RS-DMV), http://www.robesafe.com/personal/jnuevo/Datasets.html

Example Task: Detecting Driver Fatigue

Sanity Check: Is Machine Learning the right way to solve it?

- Can you think of an algorithmic way to solve the problem? **No.**
- It is possible to get data for the problem? **Yes.**

$\rightarrow$ machine learning sounds worth trying. ✓

Image: RobeSafe Driver Monitoring Video Dataset (RS-DMV), http://www.robesafe.com/personal/jnuevo/Datasets.html

Step 1) Decide what exactly you want

- **input** x: images (driver of a car)
- **output** $y \in [0, 1]$: e.g. *"how tired does the driver look?"*
  $y = 0$: totally awake, $y = 1$: sound asleep
- **quality measure**: e.g. $\ell(\, y, f(x)\,) = (y - f(x))^2$
- **model** f_θ: e.g. ConvNet with certain topology, e.g. *ResNet50*
  - input layer: fixed size image (scaled version of input x)
  - output layer: single output, value $f_\theta(x) \in [0, 1]$
  - parameters: θ (all weights of all layers)
- **goal**: find parameters θ^* such that model makes good predictions

Step 1) Decide what exactly you want

- **input** x: E.g. images of the driver of a car
- **output** $y \in [0,1]$: E.g. *"how tired does the driver look?"*
  $y = 0$: totally awake, $y = 1$: sound asleep
- **quality measure**: $\ell(\, y, f(x)\,) = (y - f(x))^2$
- **model** f_θ: convolution network with certain topology
- **goal**: find parameters θ^* such that f_{θ^*} makes good predictions

Take care that

- the outputs make sense for the inputs
  - e.g., what to output for images that don't show a person at all?

Step 1) Decide what exactly you want

- **input** x: E.g. images of the driver of a car
- **output** $y \in [0,1]$: E.g. *"how tired does the driver look?"*
  $y = 0$: totally awake, $y = 1$: sound asleep
- **quality measure**: $\ell(\, y, f(x)\,) = (y - f(x))^2$
- **model** f_θ: convolution network with certain topology
- **goal**: find parameters θ^* such that f_{θ^*} makes good predictions

Take care that

- the outputs make sense for the inputs
  – e.g., what to output for images that don't show a person at all?
- the inputs are informative about the output you're after
  – e.g., frontal pictures? or profile? or back of the head?

Step 1) Decide what exactly you want

- **input** x: E.g. images of the driver of a car
- **output** $y \in [0,1]$: E.g. *"how tired does the driver look?"*
  $y = 0$: totally awake, $y = 1$: sound asleep
- **quality measure**: $\ell(\, y, f(x)\,) = (y - f(x))^2$
- **model** f_θ: convolution network with certain topology
- **goal**: find parameters θ^* such that f_{θ^*} makes good predictions

Take care that

- the outputs make sense for the inputs
  - e.g., what to output for images that don't show a person at all?
- the inputs are informative about the output you're after
  - e.g., frontal pictures? or profile? or back of the head?
- the quality measure makes sense for the task
  - 'fatigue' (real-valued): regression, e.g. $\ell(y, f(x)) = (y - f(x))^2$
  - 'driver identification': classification, e.g. $\ell(y, f(x)) = [\![y \neq f(x)]\!]$
  - 'failure probability', e.g. $\ell(y, f(x)) = y \log f(x) + (1-y)\log(1-f(x))$

Step 1) Decide what exactly you want

- **input** x: E.g. images of the driver of a car
- **output** $y \in [0,1]$: E.g. *"how tired does the driver look?"*
  $y = 0$: totally awake, $y = 1$: sound asleep
- **quality measure**: $\ell(\,y, f(x)\,) = (y - f(x))^2$
- **model** f_θ: convolution network with certain topology
- **goal**: find parameters θ^* such that f_{θ^*} makes good predictions

Take care that

- the outputs make sense for the inputs
  - e.g., what to output for images that don't show a person at all?
- the inputs are informative about the output you're after
  - e.g., frontal pictures? or profile? or back of the head?
- the quality measure makes sense for the task
  - 'fatigue' (real-valued): regression, e.g. $\ell(y, f(x)) = (y - f(x))^2$
  - 'driver identification': classification, e.g. $\ell(y, f(x)) = [\![y \neq f(x)]\!]$
  - 'failure probability', e.g. $\ell(y, f(x)) = y \log f(x) + (1-y)\log(1 - f(x))$
- the model class makes sense (later...)

Step 2) Collect and annotate data

- **collect data**: images $x_1, x_2, \ldots$
- **annotate data**: have an expert assign 'correct' outputs: $y_1, y_2, \ldots$

In data collection, take care that

- the data reflects the situation of interest well
  - same conditions (resolution, perspective, lighting) as from car camera, ...
- the data covers all situations of interest
  - drivers of all genders and ethnicities
  - diverse set of car models
- you have enough data
  - the more the better, but quantity is not a replacement for quality

For real-world projects, often the majority of time is invested in this step.

Step 2) Collect and annotate data

- **collect data**: images $x_1, x_2, \ldots$
- **annotate data**: have an expert assign 'correct' outputs: $y_1, y_2, \ldots$

For data annotation, take care that

- the annotation is of high quality (i.e. exactly what you want)
  - perceived 'fatigue' might be subjective
  - how tired is '0.7' compared to '0.6' ?
  - if multiple annotators are involved, common standards are required
  - beware of lazy or tired annotators,
  - beware of conventions, conversion or data entering errors
- you have enough annotated data
  - the more the better, but quantity is not a replacement for quality

For real-world projects, this step can incur the majority of cost.

Step 3) Model training

- **take a training set**, $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
- **solve the following optimization problem** to find θ^*

$$\min_\theta J(\theta) \quad \text{with} \quad J(\theta) = \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$$

Step 3) Model training

- **take a training set**, $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
- **solve the following optimization problem** to find θ^*

$$\min_{\theta} J(\theta) \quad \text{with} \quad J(\theta) = \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$$

What to take care of? That's a much longer story... we'll need:

- Embrace probabilities
- Empirical risk minimization
- Overfitting / underfitting
- Regularization
- Model selection

Embrace probabilities

Embrace probabilities

Most quantities in computer vision are not fully deterministic.

- true randomness of events
  - a photon reaches a camera's CCD chip, is it detected or not?
    it depends on quantum effects, which are stochastic.
- measurement error
  - depth sensor may only be accurate $\pm 5\%$
- incomplete knowledge
  - what will be the next picture I take with my smartphone?
- insufficient representation
  - from what material is that green object made?
    hardly possible to tell from RGB, maybe easier in hyperspectral image,
    but ultimately image information is not enough

In practice, there is no difference between these!

Probability theory allows us to deal with this.

Empirical Risk Minimization

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

What do we want?

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$

What do we want?

A model f_{θ^*} that works well (=has small loss) when applied to the data in S.

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

What do we want?

A model f_{θ^*} that works well (=has small loss) when applied to the data in S.

Why? If you are interested in the label of some x_i, just look it up in S.

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$

What do we want?

A model f_{θ^*} that works well (=has small loss) when applied to the data in S.

Why? If you are interested in the label of some x_i, just look it up in S.

What do we (really) want?

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$

What do we want?

A model f_{θ^*} that works well (=has small loss) when applied to the data in S.

Why? If you are interested in the label of some x_i, just look it up in S.

What do we (really) want?

Let $S' = \{(x'_1, y'_1), \dots, (x'_n, y'_n)\}$ be a test set. We want a model θ^* that f_{θ^*} that works well (=has small loss) when applied to the data in S'.

What do we want?

We have chosen (or we are given):

- general setup: inputs $\mathcal{X}$, outputs $\mathcal{Y}$
- set of models: f_θ for $\theta \in \Theta$
- loss function to judge model quality: ℓ
- a training set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$

What do we want?

A model f_{θ^*} that works well (=has small loss) when applied to the data in S.

Why? If you are interested in the label of some x_i, just look it up in S.

What do we (really) want?

Let $S' = \{(x'_1, y'_1), \dots, (x'_n, y'_n)\}$ be a test set. We want a model θ^* that f_{θ^*} that works well (=has small loss) when applied to the data in S'.

Still, why? If you are interested in the label of some x'_i, just look it up in S'.

What do we (really, really) want?

A model f_θ that works well (=has small loss) when we apply it to future data.

What do we (really, really) want?

A model f_θ that works well (=has small loss) when we apply it to future data.

Problem: we don't know what inputs the future will bring!

Probabilities to the rescue:

- we are **uncertain** about future input data $\to$ use **random variable** X
- $\mathcal{X}$: all possible images, $p(x)$ probability to see any $x \in \mathcal{X}$

What do we (really, really) want?

A model f_θ that works well (=has small loss) when we apply it to future data.

Problem: we don't know what inputs the future will bring!

Probabilities to the rescue:

- we are **uncertain** about future input data $\rightarrow$ use **random variable** X
- $\mathcal{X}$: all possible images, $p(x)$ probability to see any $x \in \mathcal{X}$

Problem: we don't know what the right outputs are for the inputs.

Probabilities to the rescue:

- we are **uncertain** about the outputs $\rightarrow$ use **random variable** Y
- $\mathcal{Y}$: all possible outputs, $p(y|x)$ probability that $y \in \mathcal{Y}$ is correct for some $x \in \mathcal{X}$

Note: we don't pretend that we know $p(x)$ or $p(y|x)$, we just assume they exist.

What do we want formally?

A model f_θ with small expected loss (aka "risk" or "test error")

$$\mathcal{R} = \mathbb{E}_{x \sim p(x)} \mathbb{E}_{y \sim p(y|x)} [\, \ell(\, y, f_\theta(x)) \,] = \mathbb{E}_{(x,y) \sim p(x,y)} [\, \ell(\, y, f_\theta(x)) \,]$$

New problem: we can't compute $\mathcal{R}$, because we don't know $p(x, y)$ (nor $p(x)$, nor $p(y|x)$)

What do we want formally?

A model f_θ with small expected loss (aka "risk" or "test error")

$$\mathcal{R} = \mathbb{E}_{x\sim p(x)}\mathbb{E}_{y\sim p(y|x)}[\,\ell(\,y, f_\theta(x))\,] = \mathbb{E}_{(x,y)\sim p(x,y)}[\,\ell(\,y, f_\theta(x))\,]$$

New problem: we can't compute $\mathcal{R}$, because we don't know $p(x,y)$ (nor $p(x)$, nor $p(y|x)$)

Good news: We can estimate it!

Empirical risk

Let $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$ be a set of input-output pairs. Then

$$\hat{\mathcal{R}} = \frac{1}{n}\sum_{i=1}^{n} \ell(\,y_i, f_\theta(x_i))$$

is called the empirical risk of $\mathcal{R}$ w.r.t. S. (short: training error)

We can easily compute $\hat{\mathcal{R}}$. Can we use it as a drop-in replacement for $\mathcal{R}$?

Is $\hat{\mathcal{R}} = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$ a good estimator of $\mathcal{R}$?

Is $\hat{\mathcal{R}} = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$ a good estimator of $\mathcal{R}$? That depends on data we use.

Independent and identically distributed (i.i.d.) training data

If the samples $x_1, \ldots, x_n$ are sampled independently from $p(x)$ and the outputs $y_1, \ldots, y_n$ are sampled independently from $p(y|x)$, then $\hat{\mathcal{R}}$ is an unbiased and consistent estimator of $\mathcal{R}$:

$$\mathbb{E}_{(x_1,y_1),\ldots,(x_n,y_n)}[\hat{\mathcal{R}}] = \mathcal{R} \qquad \text{and} \qquad \mathrm{Var}(\hat{\mathcal{R}}) \to 0 \ \text{ with speed } O(\tfrac{1}{n})$$

(essentially, $\hat{\mathcal{R}}$ is a noisy version of $\mathcal{R}$, and the more data we have, the smaller the noise)

Is $\hat{\mathcal{R}} = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$ a good estimator of $\mathcal{R}$? That depends on data we use.

Independent and identically distributed (i.i.d.) training data

If the samples $x_1, \dots, x_n$ are sampled independently from $p(x)$ and the outputs $y_1, \dots, y_n$ are sampled independently from $p(y|x)$, then $\hat{\mathcal{R}}$ is an unbiased and consistent estimator of $\mathcal{R}$:

$$\mathbb{E}_{(x_1,y_1),\dots,(x_n,y_n)}[\hat{\mathcal{R}}] = \mathcal{R} \qquad \text{and} \qquad \mathrm{Var}(\hat{\mathcal{R}}) \to 0 \ \text{ with speed } O(\tfrac{1}{n})$$

(essentially, $\hat{\mathcal{R}}$ is a noisy version of $\mathcal{R}$, and the more data we have, the smaller the noise)

What if the data $(x_1, y_1), \dots, (x_n, y_n)$ are not independent (e.g. video frames)?

- $\hat{\mathcal{R}}$ is unbiased, but Var$[\hat{\mathcal{R}}]$ will converge slower to 0 (potentially very very slowly)

What if the distribution of the samples $(x_1, y_1), \dots, (x_n, y_n)$ is different from p (biased inputs, biased outputs or both)?

- $\hat{\mathcal{R}}$ can differ a lot from $\mathcal{R}$ ($\to$ "domain adaptation", "out-of-distribution", "fairness")

Step 2) Collect and annotate data

- collect examples: $x_1, x_2, \ldots$
- have an expert annotate them with 'correct' outputs: $y_1, y_2, \ldots$

Take care that:

- data comes from the data distribution we actually care about ("prediction time")
- examples are independent
- output annotation is a good as possible

Step 2) Collect and annotate data

- collect examples: $x_1, x_2, \ldots$
- have an expert annotate them with 'correct' outputs: $y_1, y_2, \ldots$

Take care that:

- data comes from the data distribution we actually care about ("prediction time")
- examples are independent
- output annotation is a good as possible

Step 3) Model training

Take a training set, $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, and find θ^* by minimizing the loss on the training set:

$$\min_\theta J(\theta) \quad \text{with} \quad J(\theta) = \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$$

Observation: $J(\theta)$ is simply the *empirical risk* $\hat{\mathcal{R}}$ (up to an irrelevant factor $\frac{1}{n}$)

Learning by **Empirical Risk Minimization**

Learning is about finding a model that works on future data.

The true quantity of interest is the expected error on future data:

$$\mathcal{R} = \mathbb{E}_{(x,y)\sim p(x,y)}\ell(\,y, f(x)) \qquad \text{(test error)}$$

We cannot compute $\mathcal{R}$, but we can estimate it.

For a training set, $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$, we can compute the average error:

$$\hat{\mathcal{R}} = \sum_{i=1}^{n} \ell(\,y_i, f(x_i)) \qquad \text{(training error)}$$

Training data should ideally be i.i.d. from the right distribution

If the training data are sampled independently and all from the distribution $p(x, y)$ then $\hat{\mathcal{R}}$ is a good (unbiased and consistent) estimator of $\mathcal{R}$.

Learning $\equiv$ empirical risk minimization

The core of most learning methods is to minimize the training error.

Overfitting / underfitting

We found a model f_{θ^*} by minimizing the training error $\hat{\mathcal{R}}$.

Q: Will it work well on future data, i.e. have small test error, $\mathcal{R}$?

A: **Unfortunately, that is not guaranteed.**

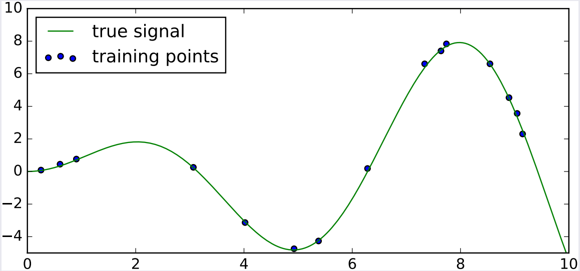
We found a model f_{θ^*} by minimizing the training error $\hat{\mathcal{R}}$.

Q: Will it work well on future data, i.e. have small test error, $\mathcal{R}$?

A: **Unfortunately, that is not guaranteed.**

Relation between training error and test error

Example: 1D curve fitting



training points

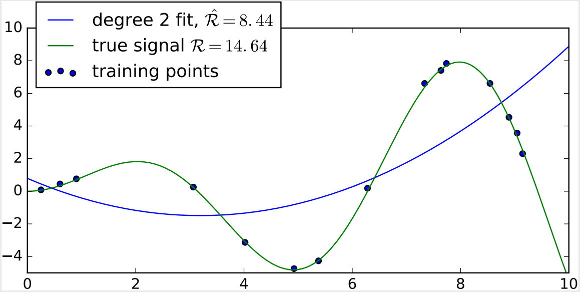
We found a model f_{θ^*} by minimizing the training error $\hat{\mathcal{R}}$.

Q: Will it work well on future data, i.e. have small test error, $\mathcal{R}$?

A: **Unfortunately, that is not guaranteed.**

Relation between training error and test error

Example: 1D curve fitting



best learned polynomial of degree 2: large $\hat{\mathcal{R}}$, large $\mathcal{R}$

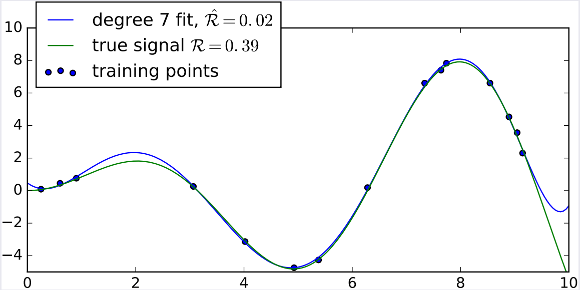
We found a model f_{θ^*} by minimizing the training error $\hat{\mathcal{R}}$.

Q: Will it work well on future data, i.e. have small test error, $\mathcal{R}$?

A: **Unfortunately, that is not guaranteed.**

Relation between training error and test error

Example: 1D curve fitting



best learned polynomial of degree 7: small $\hat{\mathcal{R}}$, small $\mathcal{R}$

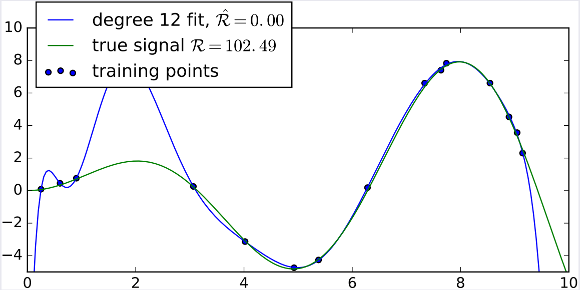
$\color{orange}{\text{We found a model } f_{\theta^*} \text{ by minimizing the training error } \hat{\mathcal{R}}.}$

Q: Will it work well on future data, i.e. have small test error, $\mathcal{R}$?

A: **Unfortunately, that is not guaranteed.**

Relation between training error and test error

Example: 1D curve fitting



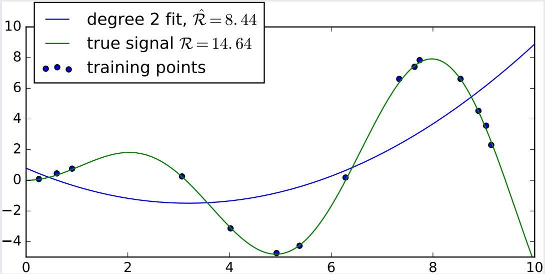
best learned polynomial of degree 12: small $\hat{\mathcal{R}}$, large $\mathcal{R}$

We found a model f_{θ^*} by minimizing the training error $\hat{\mathcal{R}}$.

Q: Will its test error, $\mathcal{R}$, be small?

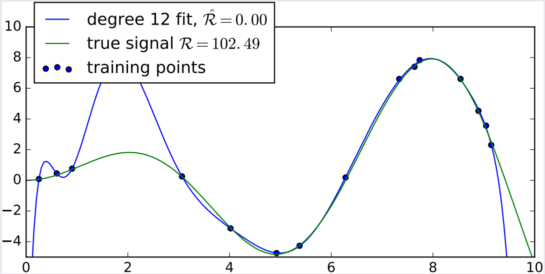
A: **Unfortunately, that is not guaranteed.**

Underfitting/Overfitting



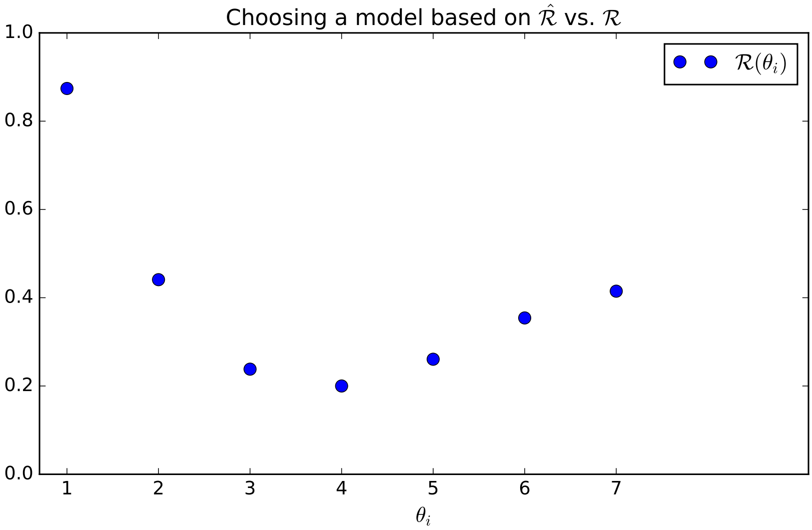
Underfitting

(to some extent) detectable from $\hat{\mathcal{R}}$

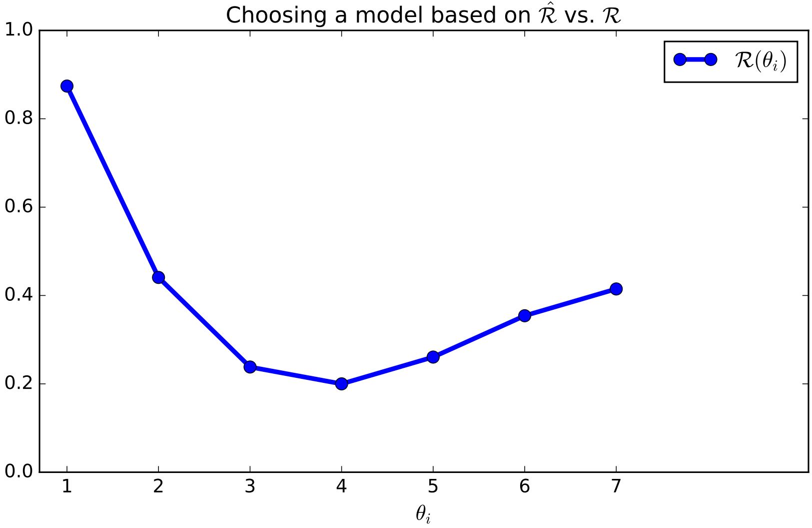


Overfitting

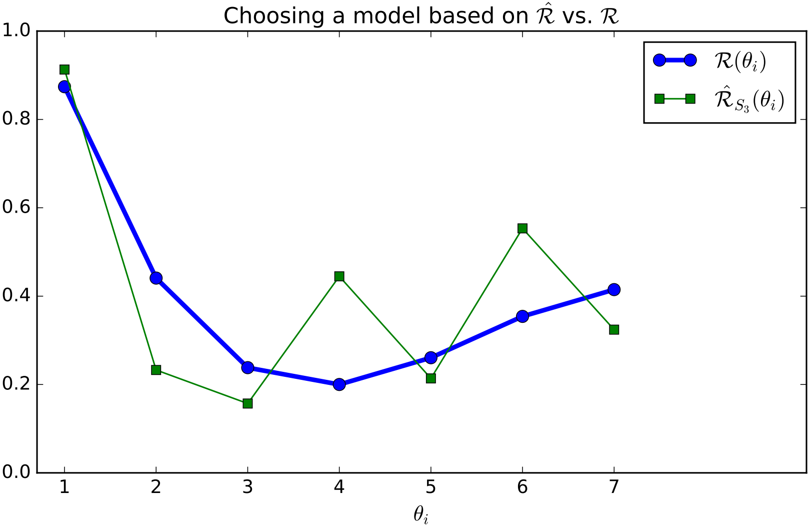
not detectable from $\hat{\mathcal{R}}$!



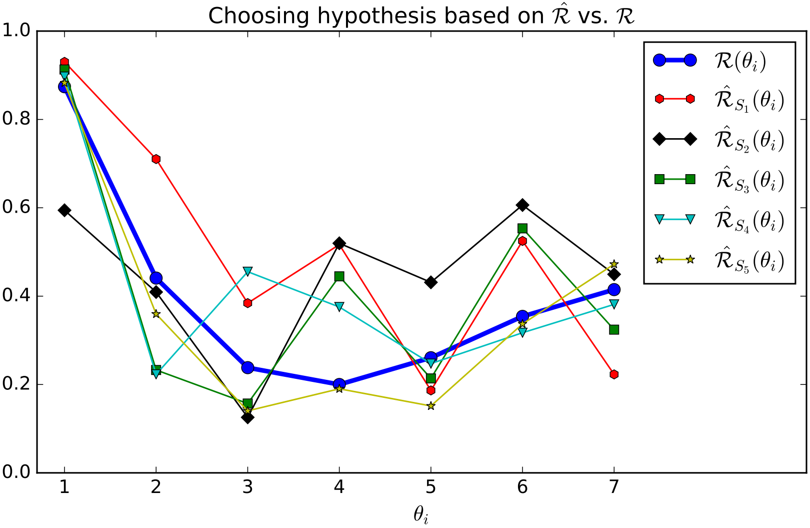
test error $\mathcal{R}$ for 7 different models/parameter choices



test error $\mathcal{R}$ for 7 different models/parameter choices

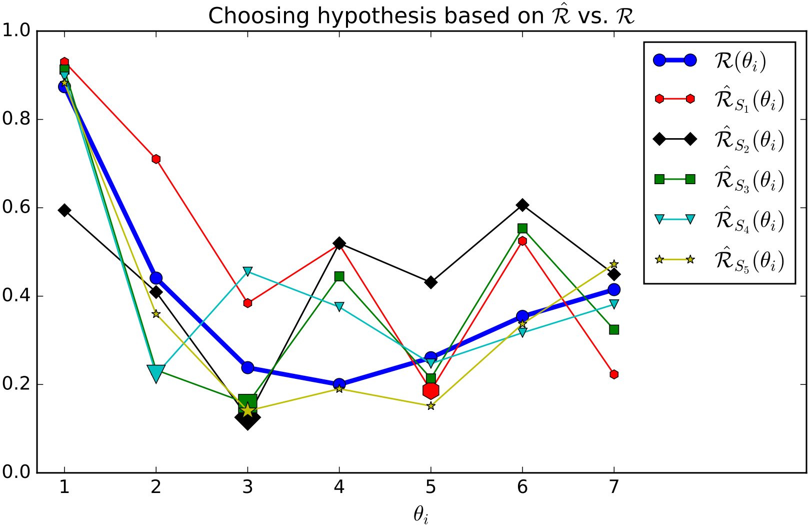


training error $\hat{\mathcal{R}}$ for a training set, S



training errors $\hat{\mathcal{R}}$ for 5 possible training sets

Where does overfitting come from?



model with smallest training error can have high test error

The training error does not tell us how good a model really is. So, what does?

The training error does not tell us how good a model really is. So, what does?

Step 4) Model evaluation

Take new data, $S' = \{(x'_1, y'_1), \ldots, (x'_m, y'_m)\}$, and compute the model performance as

$$\hat{\mathcal{R}}_{\text{tst}} = \frac{1}{m} \sum_{i=1}^{m} \ell(\, y'_i, f_{\theta^*}(x'_i)\,)$$

Take care that:

- data and annotation for evaluation comes from the real distribution
- examples are independent from each other
- data is independent of the chosen model,
  in particular: model architecture and parameter depend in no way on S'

If all of these are fulfilled:

- $\hat{\mathcal{R}}_{\text{tst}}$ is an unbiased estimate of $\mathcal{R}$, and we can expect $\hat{\mathcal{R}}_{\text{tst}} = \mathcal{R} + O(\frac{1}{\sqrt{m}})$

The training error does not tell us how good a model really is. So, what does?

Step 4) Model evaluation

Take new data, $S' = \{(x'_1, y'_1), \ldots, (x'_m, y'_m)\}$, and compute the model performance as

$$\hat{\mathcal{R}}_{\text{tst}} = \frac{1}{m} \sum_{i=1}^{m} \ell(\, y'_i, f_{\theta^*}(x'_i)\,)$$

Take care that:

- data and annotation for evaluation comes from the real distribution
- examples are independent from each other
- data is independent of the chosen model,
  in particular: model architecture and parameter depend in no way on S'

If all of these are fulfilled:

- $\hat{\mathcal{R}}_{\text{tst}}$ is an unbiased estimate of $\mathcal{R}$, and we can expect $\hat{\mathcal{R}}_{\text{tst}} = \mathcal{R} + O(\frac{1}{\sqrt{m}})$

The training error does not tell us how good a model really is. So, what does?

Step 4) Model evaluation

Take new data, $S' = \{(x'_1, y'_1), \ldots, (x'_m, y'_m)\}$, and compute the model performance as

$$\hat{\mathcal{R}}_{\text{tst}} = \frac{1}{m}\sum_{i=1}^{m} \ell(\, y'_i, f_{\theta^*}(x'_i)\,)$$

Take care that:

- data and annotation for evaluation comes from the real distribution
- examples are independent from each other
- data is independent of the chosen model,
  in particular: model architecture and parameter depend in no way on S'

If all of these are fulfilled:

- $\hat{\mathcal{R}}_{\text{tst}}$ is an unbiased estimate of $\mathcal{R}$, and we can expect $\hat{\mathcal{R}}_{\text{tst}} = \mathcal{R} + O(\frac{1}{\sqrt{m}})$

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓
- download videos recorded in winter whereas the training set was recorded in summer

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓
- download videos recorded in winter whereas the training set was recorded in summer ✗

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓
- download videos recorded in winter whereas the training set was recorded in summer ✗
- download videos from a different video platform than the training set

Quiz: test sets

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓
- download videos recorded in winter whereas the training set was recorded in summer ✗
- download videos from a different video platform than the training set ✗

Task: **Driver fatigue**. Available training data are images from many driving videos.

Which of these are proper **test sets**?

- download more videos of the same drivers ✗
- download more videos of the same drivers, but make sure that drivers who were asleep in the training set only appear as awake in the test set and vice versa ✗
- download more videos of different drivers ✓
- download videos recorded in winter whereas the training set was recorded in summer ✗
- download videos from a different video platform than the training set ✗

Summary: It's not always obvious how to obtain a good test set.

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model

You are given a **test set**. What are you allowed to use it for?

- train the model ✗

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore)

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate)

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate) ✗

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate) ✗
- get upset, trash your model, and switch to a completely different network architecture

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate) ✗
- get upset, trash your model, and switch to a completely different network architecture ✗

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate) ✗
- get upset, trash your model, and switch to a completely different network architecture ✗
- evaluate your final model's accuracy in order to compare it to the literature

"overfitting to the test set"

Quiz: overfitting to the test set

You are given a **test set**. What are you allowed to use it for?

- train the model ✗
- decide when to stop training (e.g. when test error does not change anymore) ✗
- adjust hyper-parameters (e.g. learning rate) ✗
- get upset, trash your model, and switch to a completely different network architecture ✗

"overfitting to the test set"

- evaluate your final model's accuracy in order to compare it to the literature ✓

Avoid overfitting to the test set! It

- invalidates your results (you'll think your model is better than it is),
- undermines the credibility of your experimental evaluation.

Test sets are precious! They can only be used once!

Trap 1: additional hyper-parameters

Imagine you propose a new term, $\Gamma(\theta)$ for a loss function

- previous work:

$$\min_{\theta} \quad F(\theta)$$

- proposed method:

$$\min_{\theta} \quad F(\theta) + \lambda\Gamma(\theta)$$

- hyperparameters: $\lambda \in \{0, 0.1, \ldots, 0.9, 1\}$

Common traps: don't fall for these!

Trap 1: additional hyper-parameters

Imagine you propose a new term, $\Gamma(\theta)$ for a loss function

- previous work:
$$\min_\theta \quad F(\theta)$$
- proposed method:
$$\min_\theta \quad F(\theta) + \lambda\Gamma(\theta)$$
- hyperparameters: $\lambda \in \{0, 0.1, \ldots, 0.9, 1\}$

If you select λ using the test set, results for the new method will always be at least as good as for the old one, regardless what Γ was. $\rightarrow$ experimental results prove nothing!

Overfitting to the test set can be avoided by not making the test data public.

Example: ChaLearn Connectomics Challenge

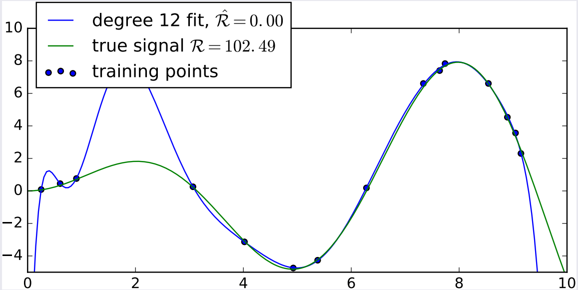
- We are given labeled training data.
- We build/train a model.
- We upload the model in executable form to a server.
- The server applies the model to test data and evaluates the results.

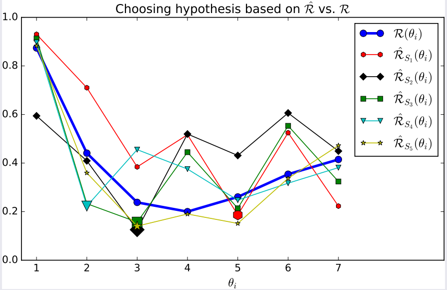
Example: ImageNet ILSVR Challenge

- We are given labeled training data and unlabeled test data.
- We build/train a model.
- We apply the model to the test data.
- We upload the model predictions to a server.
- The server evaluates the results.

Regularization

Reminder: Overfitting

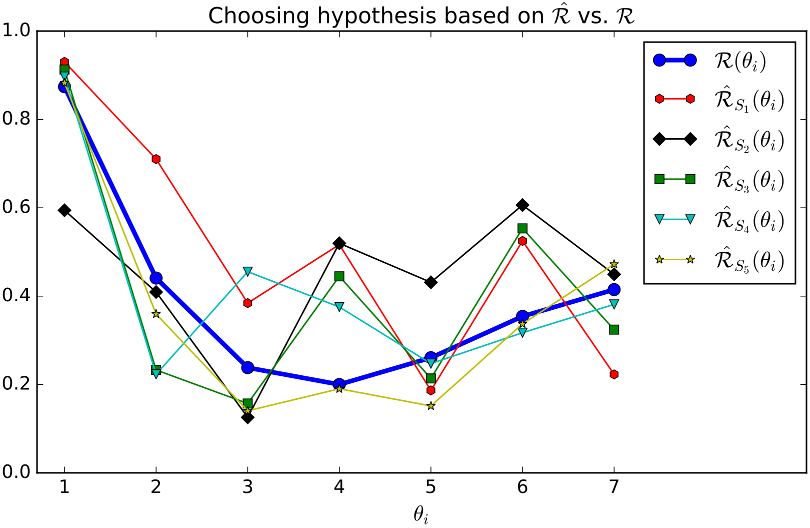




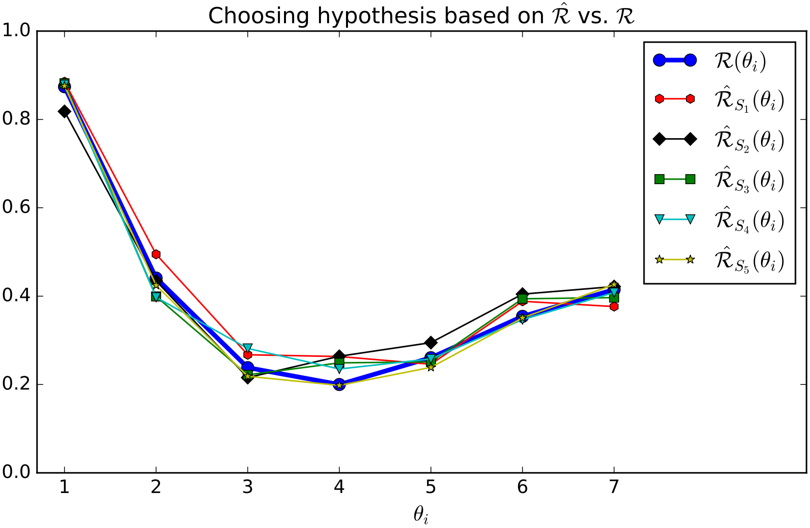
overfitting

$\hat{\mathcal{R}}$ **vs.** $\mathcal{R}$

How can we prevent overfitting when learning a model?

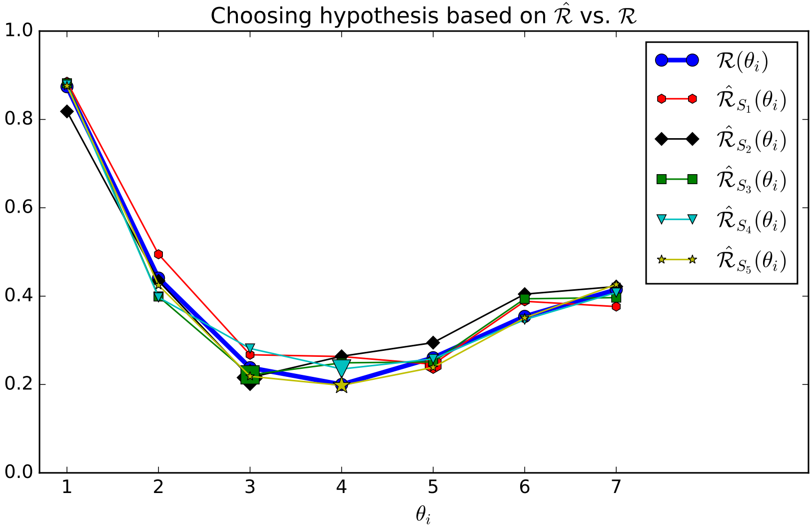

Choosing hypothesis based on $\hat{\mathcal{R}}$ vs. $\mathcal{R}$
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
7
θ_i
$\mathcal{R}(\theta_i)$
$\hat{\mathcal{R}}_{S_1}(\theta_i)$
$\hat{\mathcal{R}}_{S_2}(\theta_i)$
$\hat{\mathcal{R}}_{S_3}(\theta_i)$
$\hat{\mathcal{R}}_{S_4}(\theta_i)$
$\hat{\mathcal{R}}_{S_5}(\theta_i)$

Preventing overfitting 1) larger training set



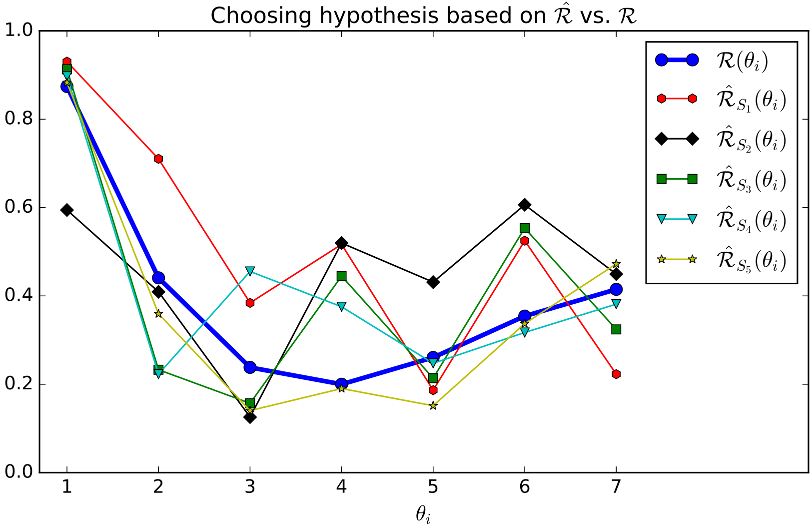
larger training set $\rightarrow$ smaller variance of $\hat{\mathcal{R}}$

Preventing overfitting 1) larger training set

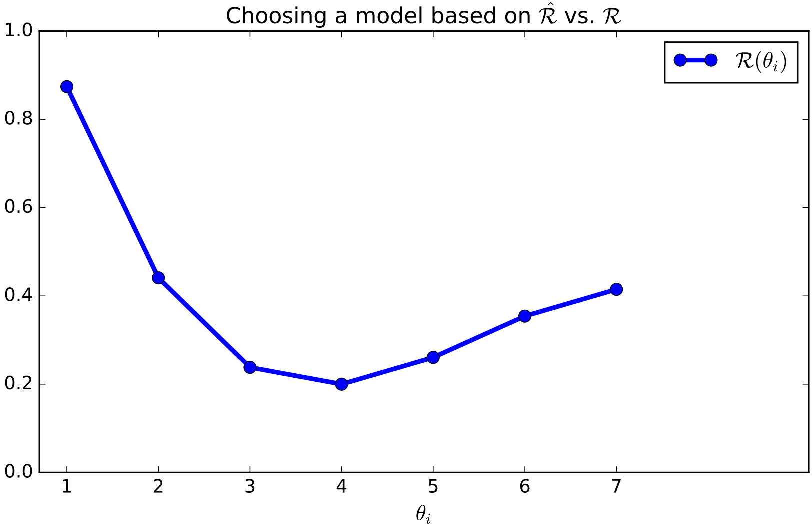


lower probability that $\hat{\mathcal{R}}$ differs strongly from $\mathcal{R}$ $\rightarrow$ less overfitting

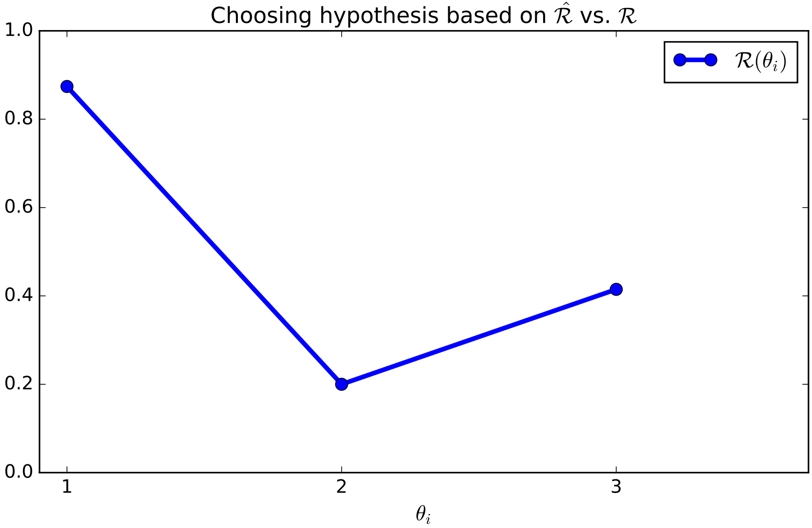
Preventing overfitting 2) reduce the number of models to choose from



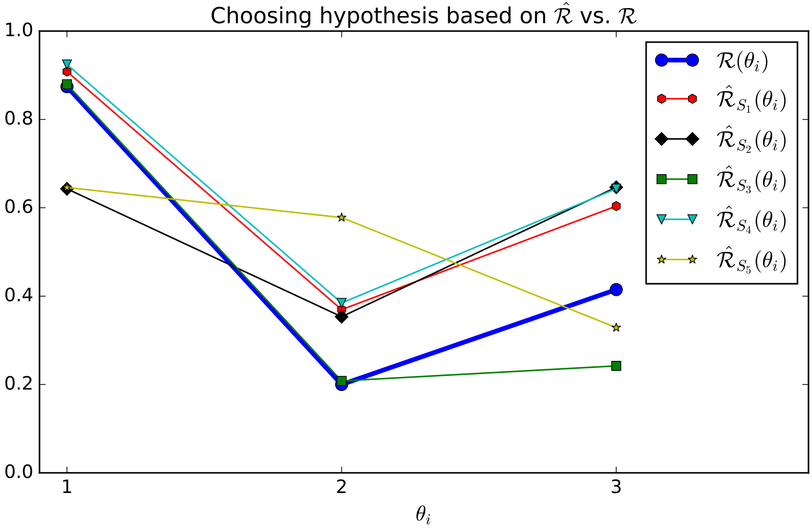
Preventing overfitting 2) reduce the number of models to choose from



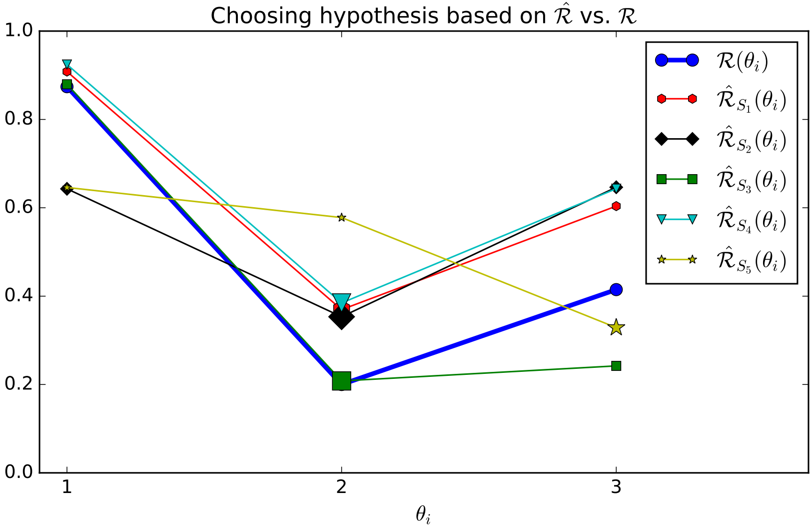
Preventing overfitting 2) reduce the number of models to choose from



Preventing overfitting 2) reduce the number of models to choose from



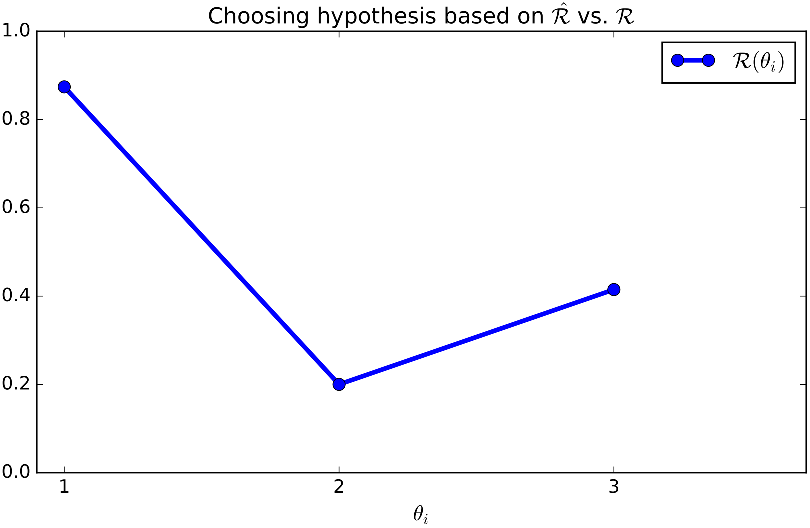
fewer models $\rightarrow$ lower probability of a model with small $\hat{\mathcal{R}}$s but high $\mathcal{R}$



fewer models $\rightarrow$ lower probability of a model with small $\hat{\mathcal{R}}$s but high $\mathcal{R}$

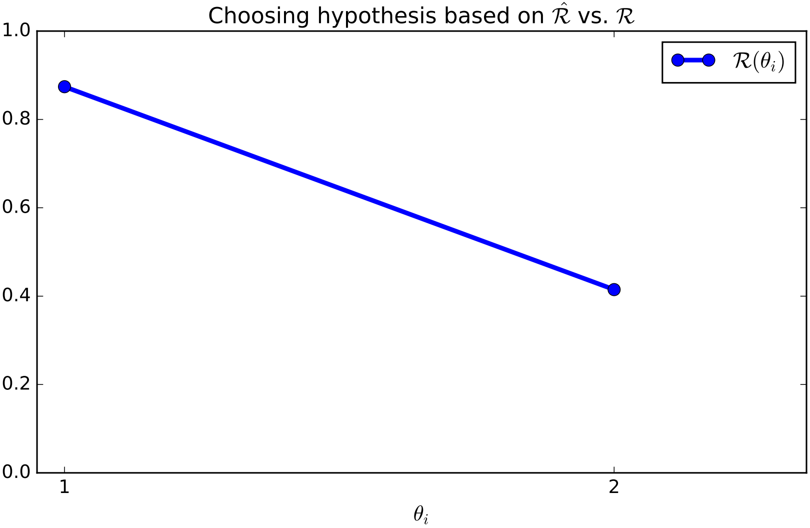
But: danger of underfitting

But: danger of underfitting



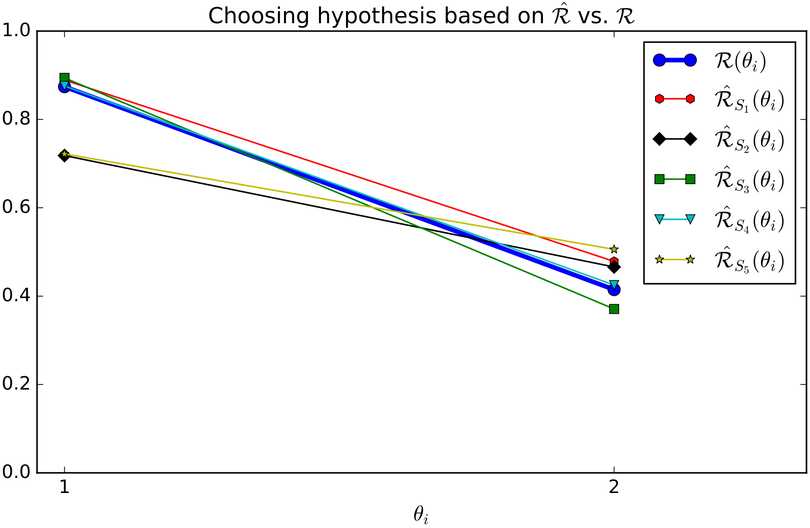
to few models select to from $\rightarrow$ danger that no model with low $\mathcal{R}$ is left!

But: danger of underfitting

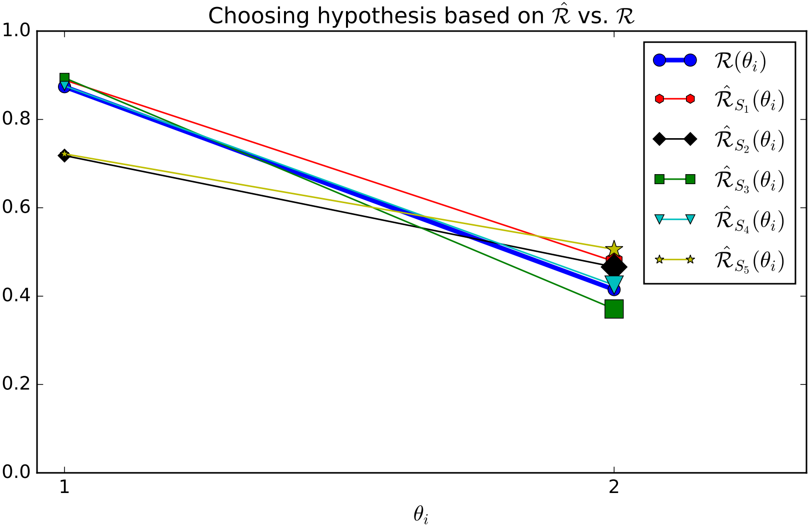


to few models select to from $\rightarrow$ danger that no model with low $\mathcal{R}$ is left!

But: danger of underfitting



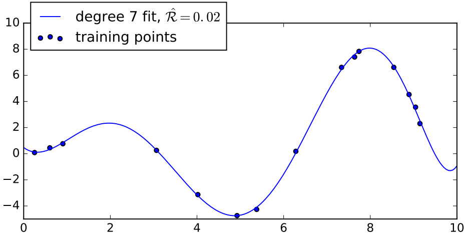
Underfitting!



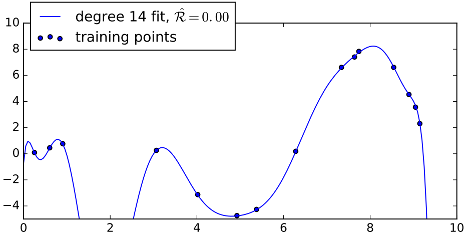
Underfitting!

Models with big difference between training error and test error are typically **extreme cases**:

- a large number of model parameters
- model parameters take large values
- learned function is very non-smooth



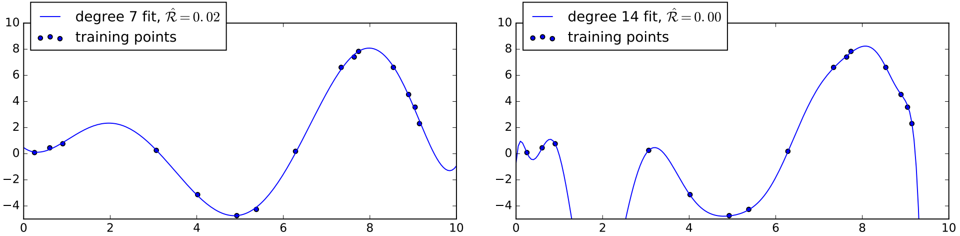
coeffs: $\theta_i \in [-2.4, 4.6]$



coeffs: $\theta_i \in [-1312.5, 1136.6]$

Models with big difference between training error and test error are typically **extreme cases**:

- a large number of model parameters
- model parameters take large values
- learned function is very non-smooth



coeffs: $\theta_i \in [-2.4, 4.6]$ coeffs: $\theta_i \in [-1312.5, 1136.6]$

Regularization: avoid overfitting by preventing extremes to occur

- explicit regularization (changing the objective function)
- implicit regularization (modifying the optimization procedure)

Add a regularization term (=regularizer) to the empirical risk that penalizes extreme parameter values.

Regularized risk minimization

Take a training set, $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, find θ^* by solving,

$$\min_\theta J_\lambda(\theta) \quad \text{with} \quad J_\lambda(\theta) = \underbrace{\sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))}_{\text{empirical risk}} + \underbrace{\lambda\Omega(\theta)}_{\text{regularizer}}$$

$$\text{e.g. with} \quad \Omega(\theta) = \|\theta\|_{L^2}^2 = \sum\nolimits_j \theta_j^2 \qquad \text{or} \qquad \Omega(\theta) = \|\theta\|_{L^1} = \sum\nolimits_j |\theta_j|$$

Explicit regularization

Add a regularization term (=regularizer) to the empirical risk that penalizes extreme parameter values.

Regularized risk minimization

Take a training set, $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$, find θ^* by solving,

$$\min_\theta J_\lambda(\theta) \quad \text{with} \quad J_\lambda(\theta) = \underbrace{\sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))}_{\text{empirical risk}} + \underbrace{\lambda\Omega(\theta)}_{\text{regularizer}}$$

e.g. with $\quad \Omega(\theta) = \|\theta\|_{L^2}^2 = \sum_j \theta_j^2 \qquad$ or $\qquad \Omega(\theta) = \|\theta\|_{L^1} = \sum_j |\theta_j|$

Optimization searches model with small training error <u>and</u> small parameter values.

Regularization (hyper)parameter $\lambda \geq 0$: trade-off between both.

- $\lambda = 0$: empirical risk minimization (risk of overfitting)
- $\lambda \to \infty$: all parameters 0 (risk of underfitting)

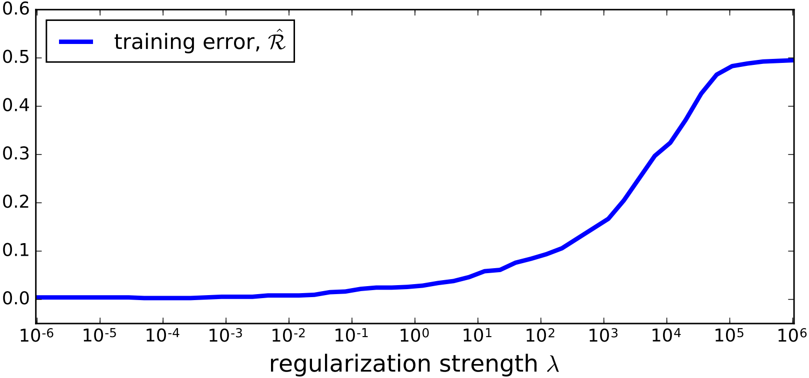
Example: linear classification with least-squared loss (LS-SVM)

$$\min_{w} J_\lambda(w) \quad \text{for} \quad J_\lambda(w) = \sum_{i=1}^{n} (w^\top x_i - y_i)^2 + \lambda \|w\|^2$$

Example: linear classification with least-squared loss (LS-SVM)

$$\min_w J_\lambda(w) \quad \text{for} \quad J_\lambda(w) = \sum_{i=1}^{n} (w^\top x_i - y_i)^2 + \lambda \|w\|^2$$

Train/val error for minimizer of J_λ with varying amounts of regularization:

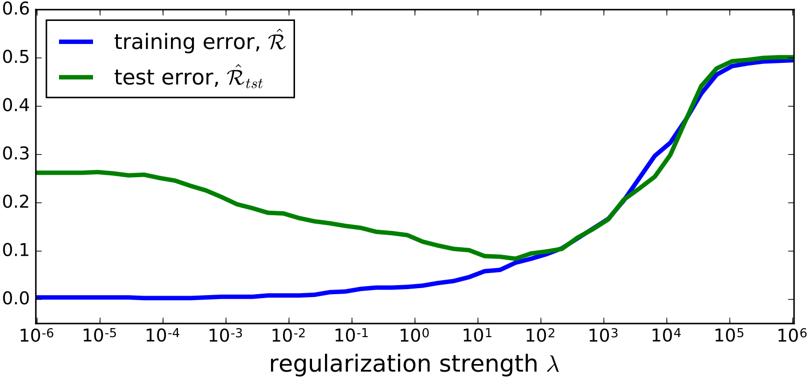


eye dataset: 737 examples for training, 736 examples for evaluation

Example: linear classification with least-squared loss (LS-SVM)

$$\min_{w} J_\lambda(w) \quad \text{for} \quad J_\lambda(w) = \sum_{i=1}^{n} (w^\top x_i - y_i)^2 + \lambda \|w\|^2$$

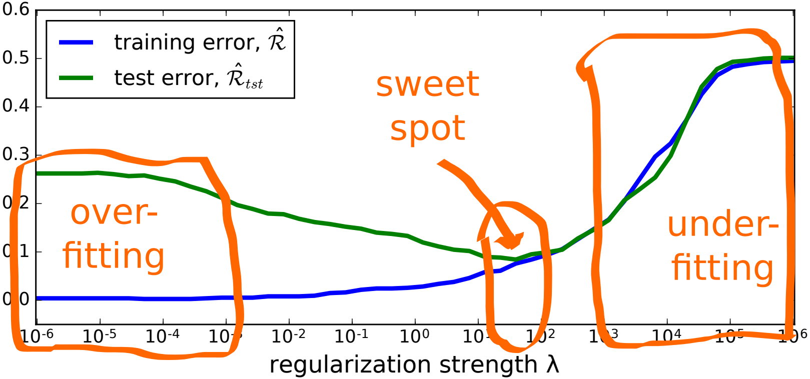
Train/val error for minimizer of J_λ with varying amounts of regularization:



eye dataset: 737 examples for training, 736 examples for evaluation

$$\min_{w} J_\lambda(w) \quad \text{for} \quad J_\lambda(w) = \sum_{i=1}^{n} (w^\top x_i - y_i)^2 + \lambda \|w\|^2$$

Train/val error for minimizer of J_λ with varying amounts of regularization:



eye dataset: 737 examples for training, 736 examples for evaluation

Training error, $\hat{\mathcal{R}}$, is a noisy estimate of the test error, $\mathcal{R}$

- original risk $\hat{\mathcal{R}}$ is unbiased, but variance can be huge
- regularization introduces a bias, but reduces variance
- for $\lambda \to \infty$, the variance goes to 0, but the bias gets very big

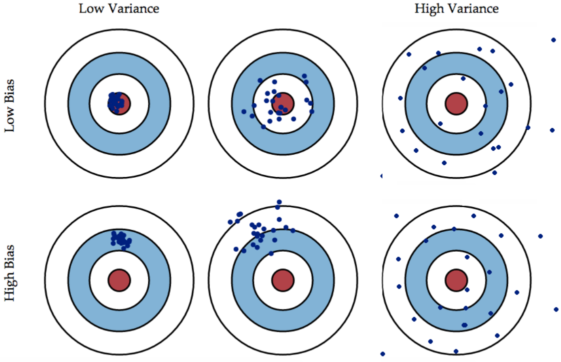


Image: adapted from http://scott.fortmann-roe.com/docs/BiasVariance.html

Regularization as Trading Off Bias and Variance

Training error, $\hat{\mathcal{R}}$, is a noisy estimate of the test error, $\mathcal{R}$

- original risk $\hat{\mathcal{R}}$ is unbiased, but variance can be huge
- regularization introduces a bias, but reduces variance
- for $\lambda \to \infty$, the variance goes to 0, but the bias gets very big

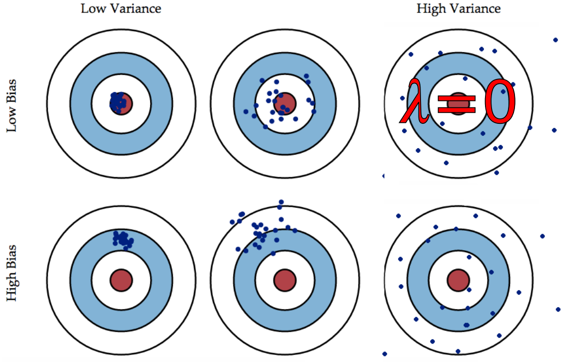


Image: adapted from http://scott.fortmann-roe.com/docs/BiasVariance.html

Training error, $\hat{\mathcal{R}}$, is a noisy estimate of the test error, $\mathcal{R}$

- original risk $\hat{\mathcal{R}}$ is unbiased, but variance can be huge
- regularization introduces a bias, but reduces variance
- for $\lambda \to \infty$, the variance goes to 0, but the bias gets very big

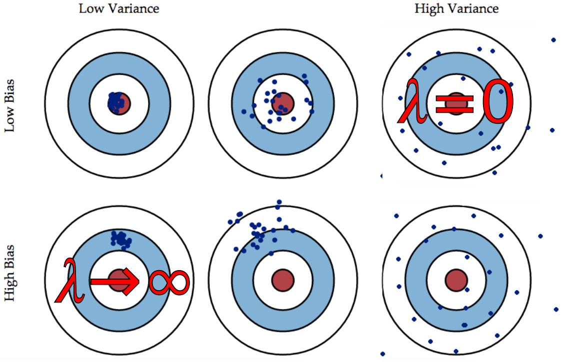


Image: adapted from http://scott.fortmann-roe.com/docs/BiasVariance.html

Training error, $\hat{\mathcal{R}}$, is a noisy estimate of the test error, $\mathcal{R}$

- original risk $\hat{\mathcal{R}}$ is unbiased, but variance can be huge
- regularization introduces a bias, but reduces variance
- for $\lambda \to \infty$, the variance goes to 0, but the bias gets very big

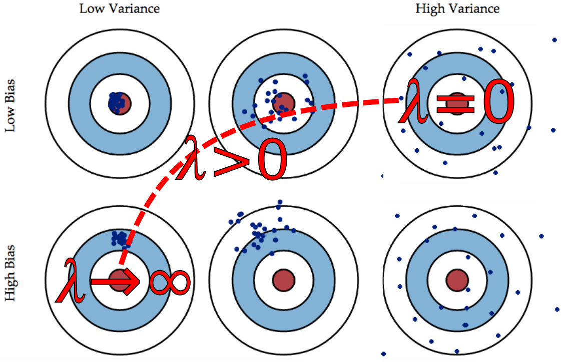


Image: adapted from http://scott.fortmann-roe.com/docs/BiasVariance.html

Numerical optimization is performed iteratively, e.g. gradient descent

Gradient descent optimization

- initialize $\theta^{(0)}$
- **for** $t = 1, 2, \ldots$
- $\theta^{(t)} \leftarrow \theta^{(t-1)} - \eta_t \nabla_\theta J(\theta^{(t-1)})$ ($\eta_t \in \mathbb{R}$ is some stepsize rule)
- **until convergence**

Implicit regularization prevent overfitting by modifying these steps, e.g.

- early stopping
- weight decay
- data augmentation
- dropout
- inductive bias $\longrightarrow$ **Intro to Deep Learning**

Model Selection

Which of model class?

- linear classifier, gradient boosted decision tree, neural network?
- ResNet? DenseNet? EfficientNet? Transformer? MLPMixer? GAN? Diffusion Network?
- which layers? how many? how wide?

Which data representation?

- raw pixels? manual preprocessing? pretrained features?

Which hyperparameters?

- batchsize? learning rate? number of epochs? which optimizer?
- regularization or not? which one? how much?

Which of model class?

- linear classifier, gradient boosted decision tree, neural network?
- ResNet? DenseNet? EfficientNet? Transformer? MLPMixer? GAN? Diffusion Network?
- which layers? how many? how wide?

Which data representation?

- raw pixels? manual preprocessing? pretrained features?

Which hyperparameters?

- batchsize? learning rate? number of epochs? which optimizer?
- regularization or not? which one? how much?

Model Selection

Which of model class?

- linear classifier, gradient boosted decision tree, neural network?
- ResNet? DenseNet? EfficientNet? Transformer? MLPMixer? GAN? Diffusion Network?
- which layers? how many? how wide?

Which data representation?

- raw pixels? manual preprocessing? pretrained features?

Which hyperparameters?

- batchsize? learning rate? number of epochs? which optimizer?
- regularization or not? which one? how much?

Model Selection

Model selection is a difficult (and often underestimated) problem:

- we can't decide based on training error: we won't catch overfitting
- we can't decide based on test error: if we use test data to select (hyper)parameters, we're overfitting to the test set and the result is not a good estimate of true risk anymore

Problem: We want to evaluate a model on different data than the data it was trained on, but not the test set.

Solution: Emulate the train/test split from only the data available for training.

- full data ("**trainval**") $\longrightarrow$ **train** and **validation** sets

- aim for train/validation sets to have same relation as train/test sets:
  - same data distribution
  - trained model independent of **validation** data
  - both sets as big as possible

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets...

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? $\rightarrow$ what about repeated drivers?

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? $\rightarrow$ what about repeated drivers?
- split the drivers randomly into two subsets. . .

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets... ? $\rightarrow$ what about repeated drivers?
- split the drivers randomly into two subsets... ? $\rightarrow$ repeated car models?

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets... ? → what about repeated drivers?
- split the drivers randomly into two subsets... ? → repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp...

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets... ? $\rightarrow$ what about repeated drivers?
- split the drivers randomly into two subsets... ? $\rightarrow$ repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp... ✗

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? → what about repeated drivers?
- split the drivers randomly into two subsets. . . ? → repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp. . . ✗
- split the drivers by gender. . .

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? → what about repeated drivers?
- split the drivers randomly into two subsets. . . ? → repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp. . . ✗
- split the drivers by gender. . . ✗

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? $\rightarrow$ what about repeated drivers?
- split the drivers randomly into two subsets. . . ? $\rightarrow$ repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp. . . ✗
- split the drivers by gender. . . ✗
- split the drivers by last name (`A-L` vs `M-Z`)

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets. . . ? → what about repeated drivers?
- split the drivers randomly into two subsets. . . ? → repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp. . . ✗
- split the drivers by gender. . . ✗
- split the drivers by last name (`A-L` vs `M-Z`) ? → can introduce bias

Quiz: train/validation sets

Driver fatigue: Available data are images from many driving videos of some group of people

Which of these are good **train/validation splits**?

- split the images randomly into two subsets, one for training and the other for validation ✗
- split the videos randomly into two subsets... ? $\rightarrow$ what about repeated drivers?
- split the drivers randomly into two subsets... ? $\rightarrow$ repeated car models?

Random splits are hard to make reproducible. How about deterministic ones?

- split the videos by time stamp... ✗
- split the drivers by gender... ✗
- split the drivers by last name (`A-L` vs `M-Z`) ? $\rightarrow$ can introduce bias

Summary: It's not always obvious how to form a validation set (especially for time series/videos).

Model selection with a validation set

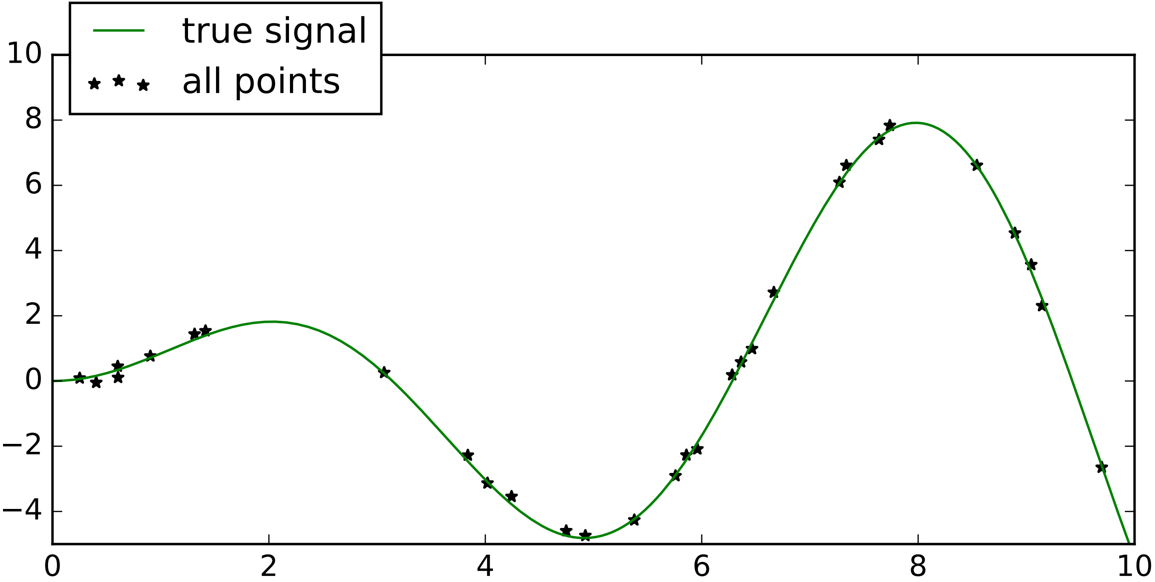
Given: training set $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$, possible hyperparameters $A = \{\eta_1, \dots, \eta_m\}$ (including, e.g., which model class to use).

- Split available training data in to disjoint *real train* and *validation* set

$$S = S_{\text{trn}} \mathbin{\dot\cup} S_{\text{val}}$$

- for all $\eta \in A$:
- $\quad f^\eta \leftarrow$ train a model with hyperparameters η using data S_{trn}
- $\quad E^\eta_{\text{val}} \leftarrow$ evaluate model f^η on set S_{val}
- $\eta^* = \operatorname{argmin}_{\eta \in A} E^\eta_{\text{val}}$ (select most promising hyperparameters)
- optionally: retrain model with hyperparameters η^* on complete S

Illustration: learning polynomials (of different degrees)


true signal
all points
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

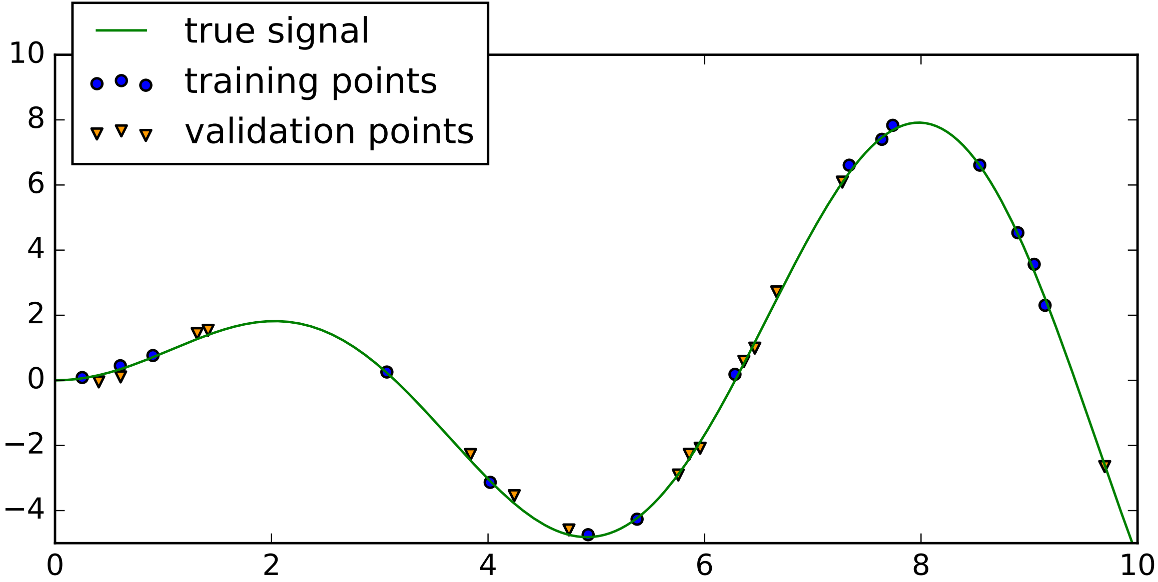

true signal
training points
validation points
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

Illustration: learning polynomials (of different degrees)

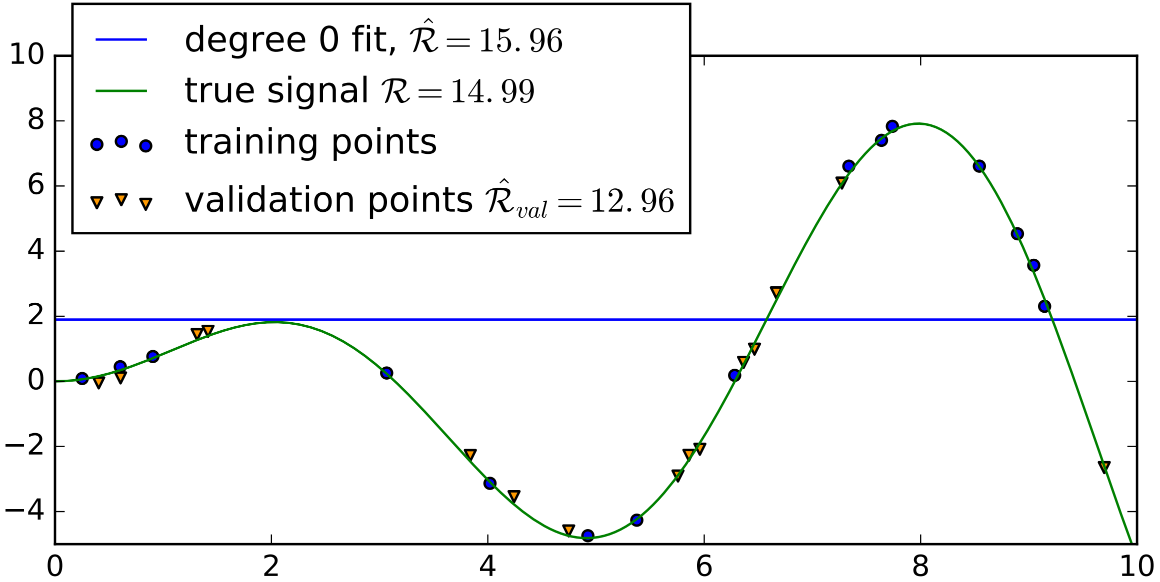


Illustration: learning polynomials (of different degrees)

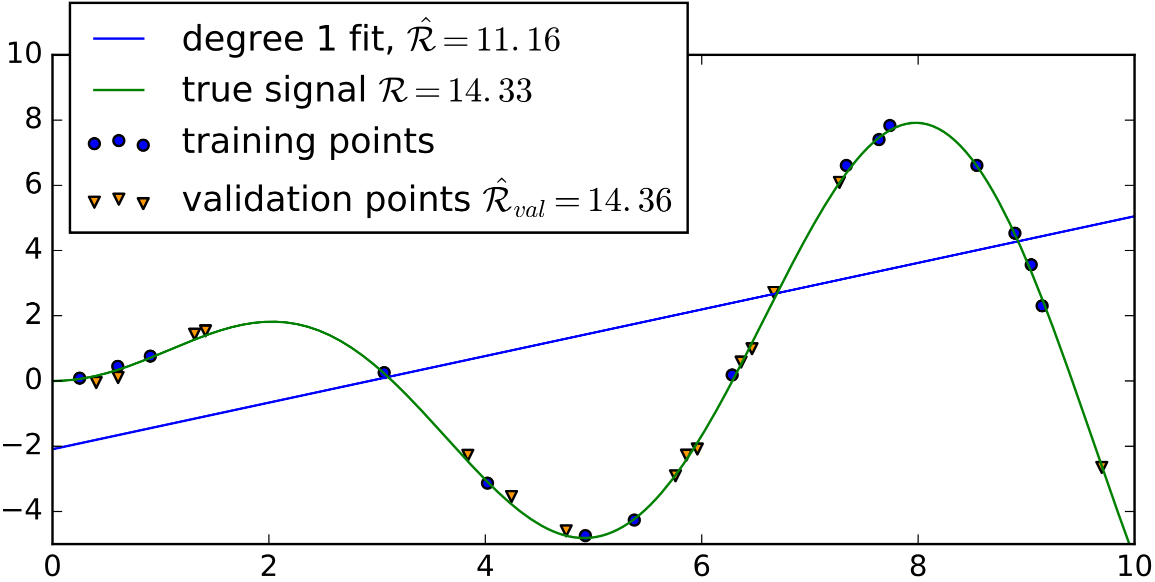
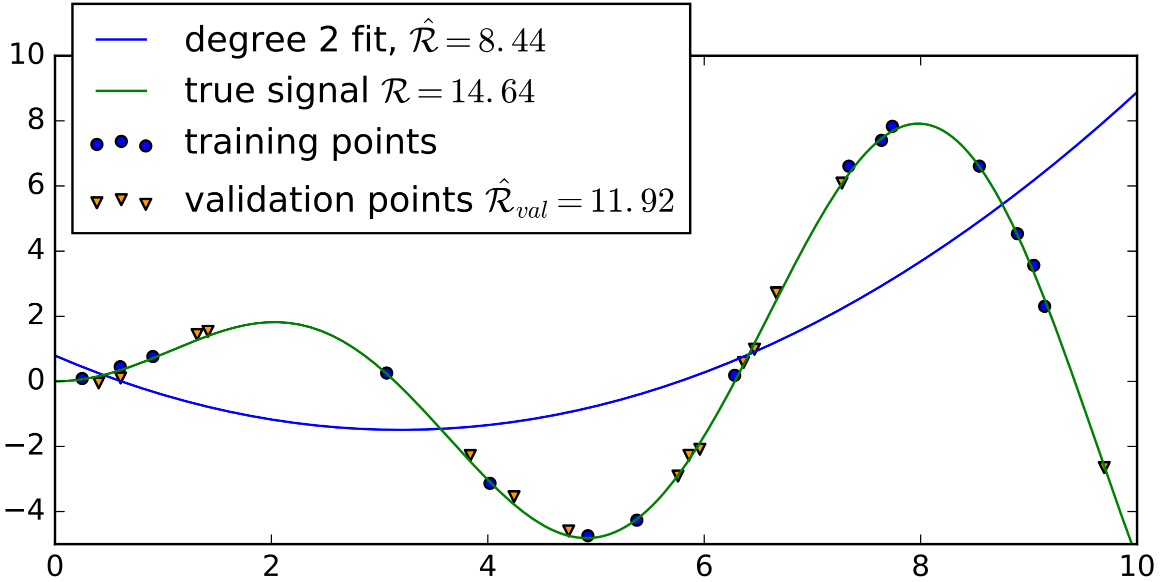
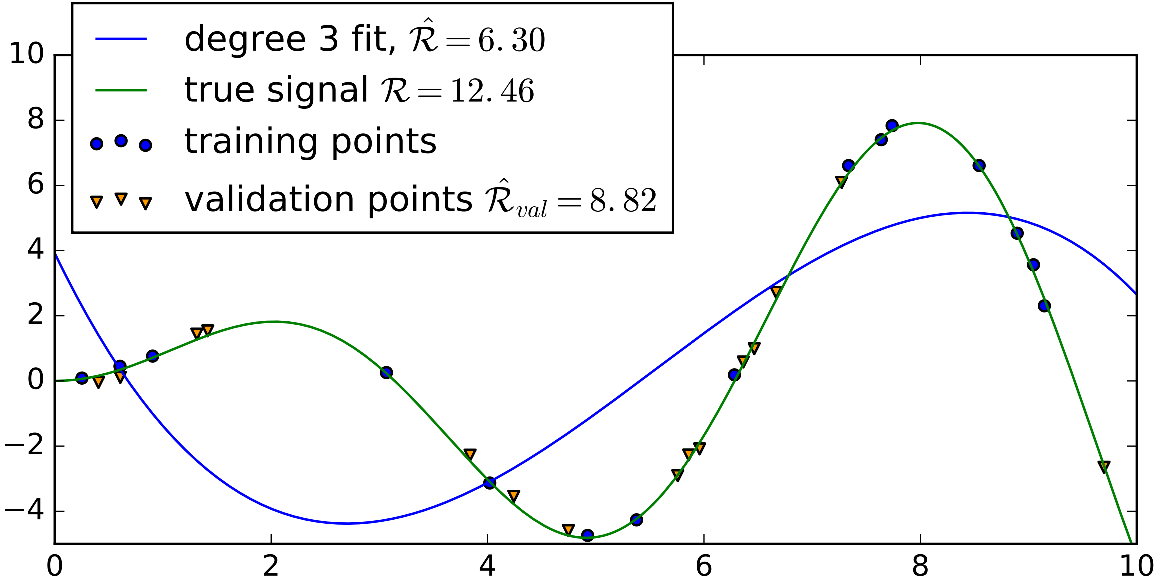


Illustration: learning polynomials (of different degrees)




degree 3 fit, $\hat{\mathcal{R}} = 6.30$
true signal $\mathcal{R} = 12.46$
training points
validation points $\hat{\mathcal{R}}_{val} = 8.82$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

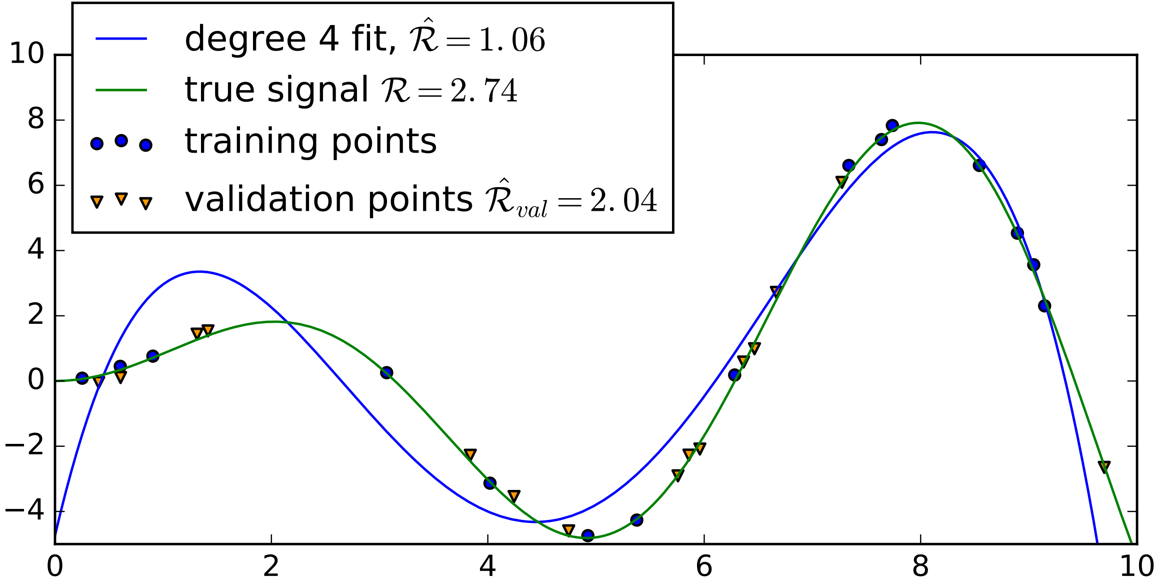
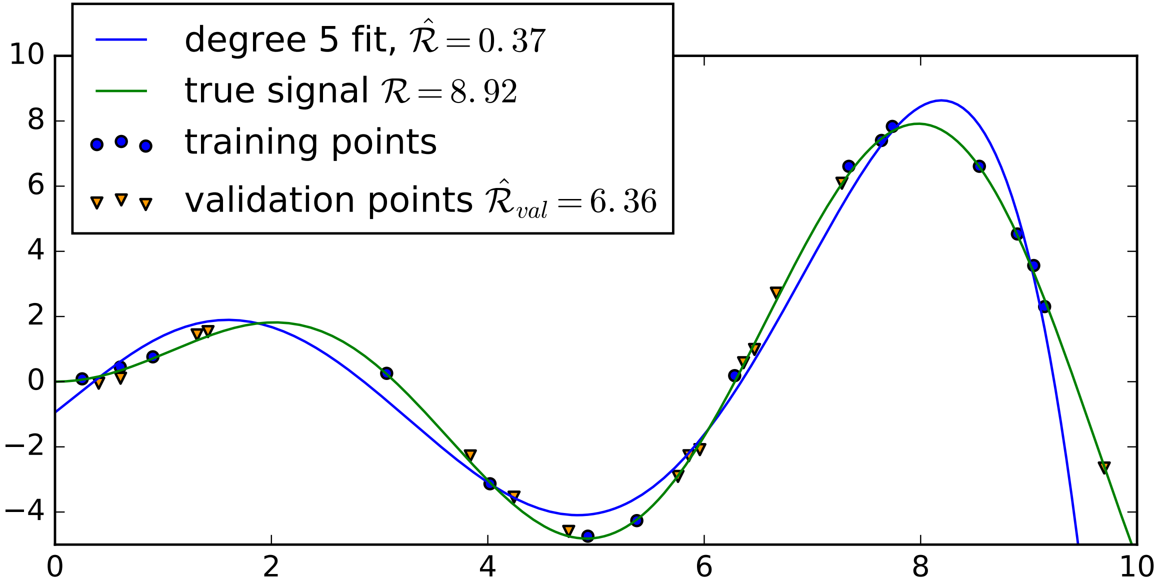
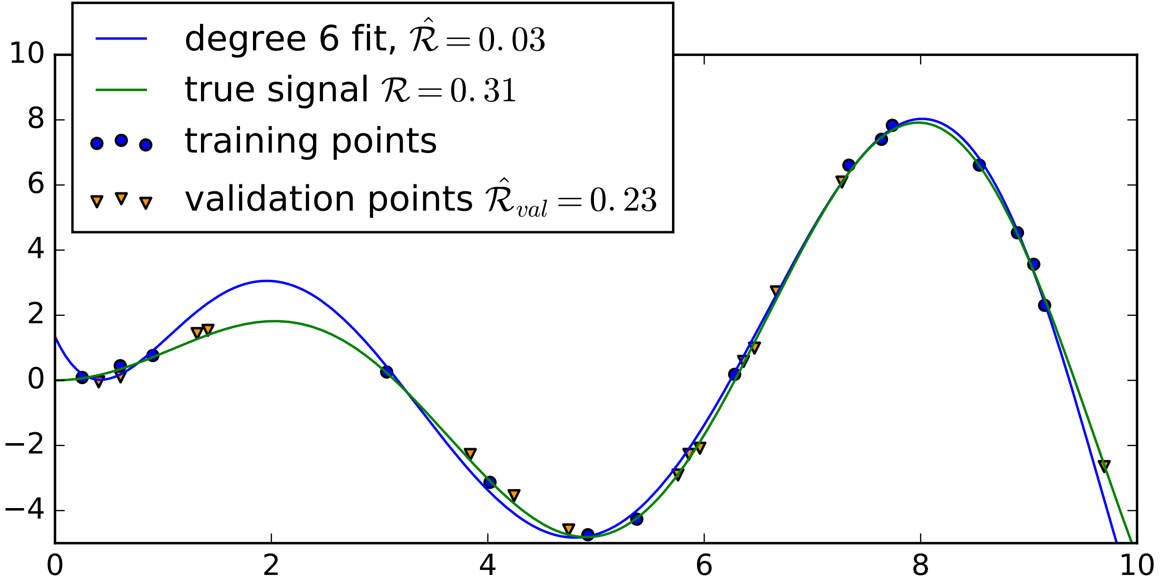
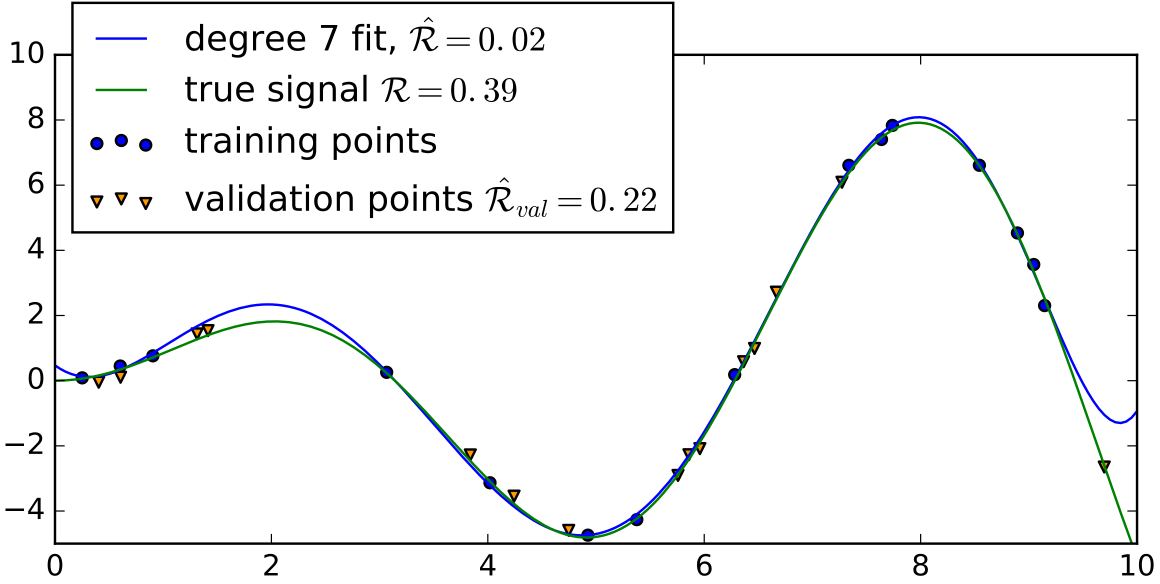
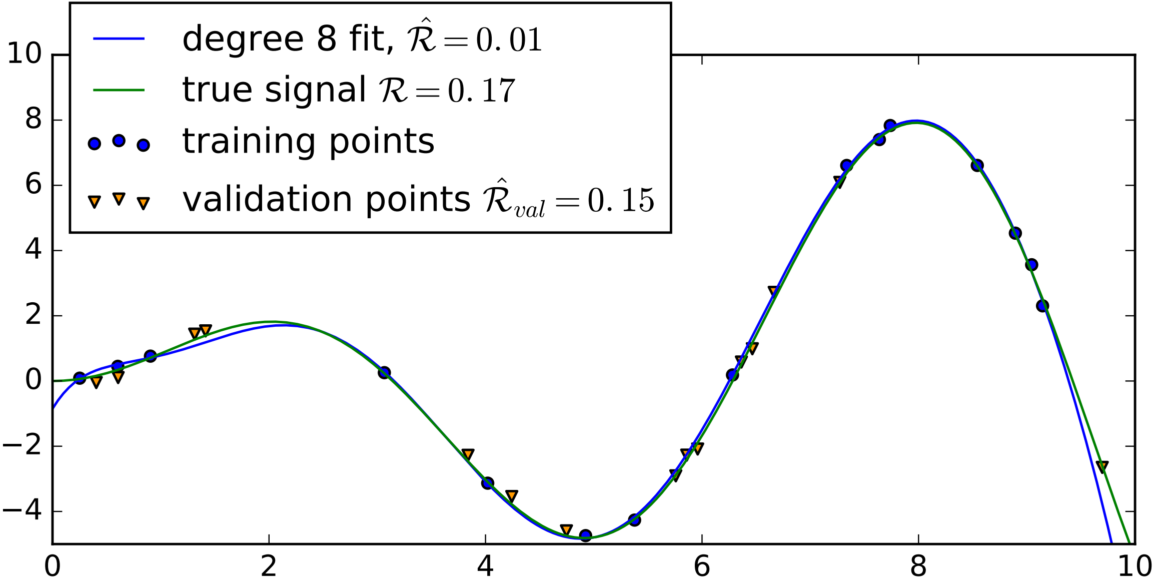

degree 4 fit, $\hat{\mathcal{R}} = 1.06$
true signal $\mathcal{R} = 2.74$
training points
validation points $\hat{\mathcal{R}}_{val} = 2.04$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

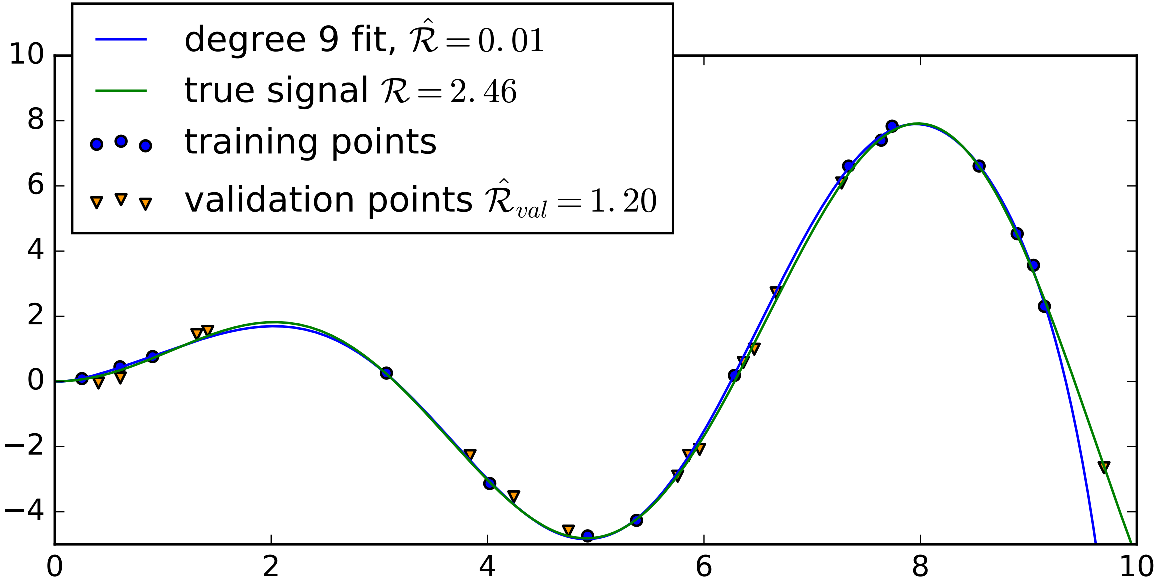
Illustration: learning polynomials (of different degrees)




degree 6 fit, $\hat{\mathcal{R}} = 0.03$
true signal $\mathcal{R} = 0.31$
training points
validation points $\hat{\mathcal{R}}_{val} = 0.23$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10


degree 7 fit, $\hat{\mathcal{R}} = 0.02$
true signal $\mathcal{R} = 0.39$
training points
validation points $\hat{\mathcal{R}}_{val} = 0.22$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10


degree 8 fit, $\hat{\mathcal{R}} = 0.01$
true signal $\mathcal{R} = 0.17$
training points
validation points $\hat{\mathcal{R}}_{val} = 0.15$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10


degree 9 fit, $\hat{\mathcal{R}} = 0.01$
true signal $\mathcal{R} = 2.46$
training points
validation points $\hat{\mathcal{R}}_{val} = 1.20$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

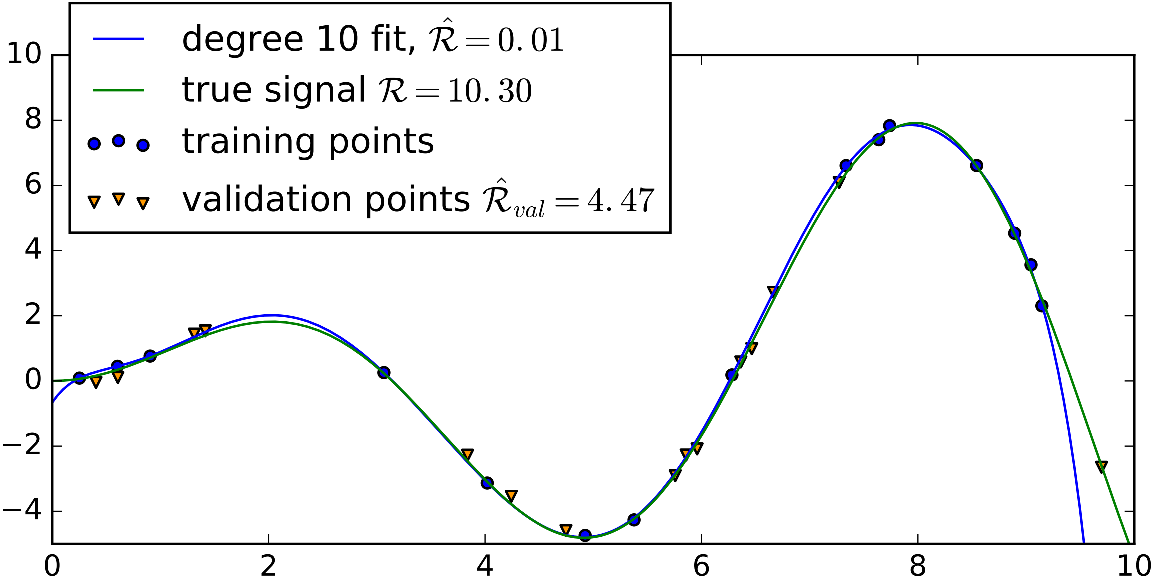

degree 10 fit, $\hat{\mathcal{R}} = 0.01$
true signal $\mathcal{R} = 10.30$
training points
validation points $\hat{\mathcal{R}}_{val} = 4.47$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

Illustration: learning polynomials (of different degrees)

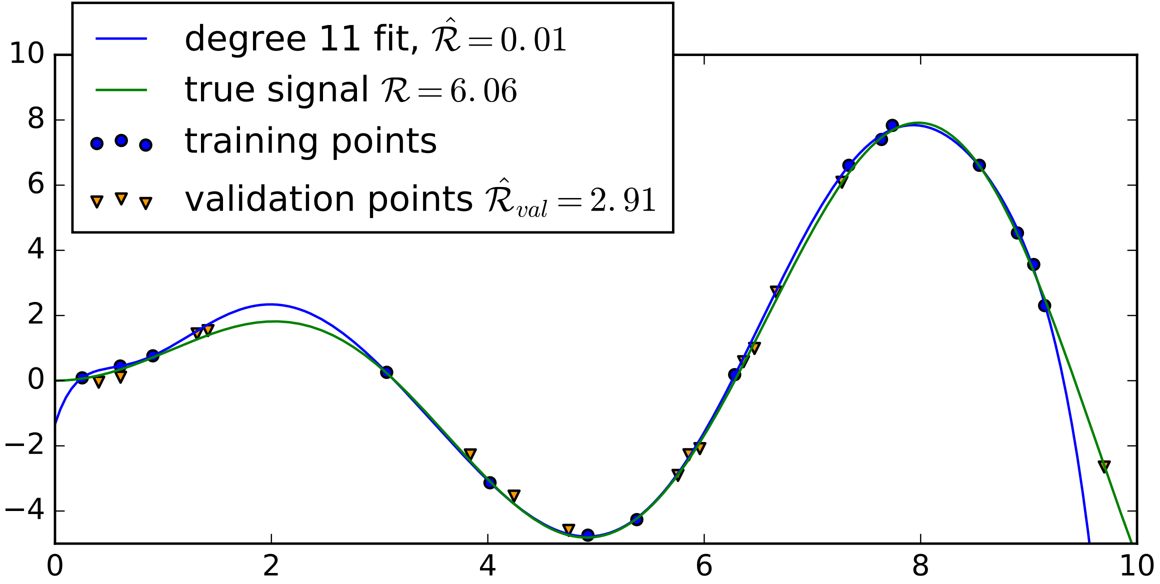
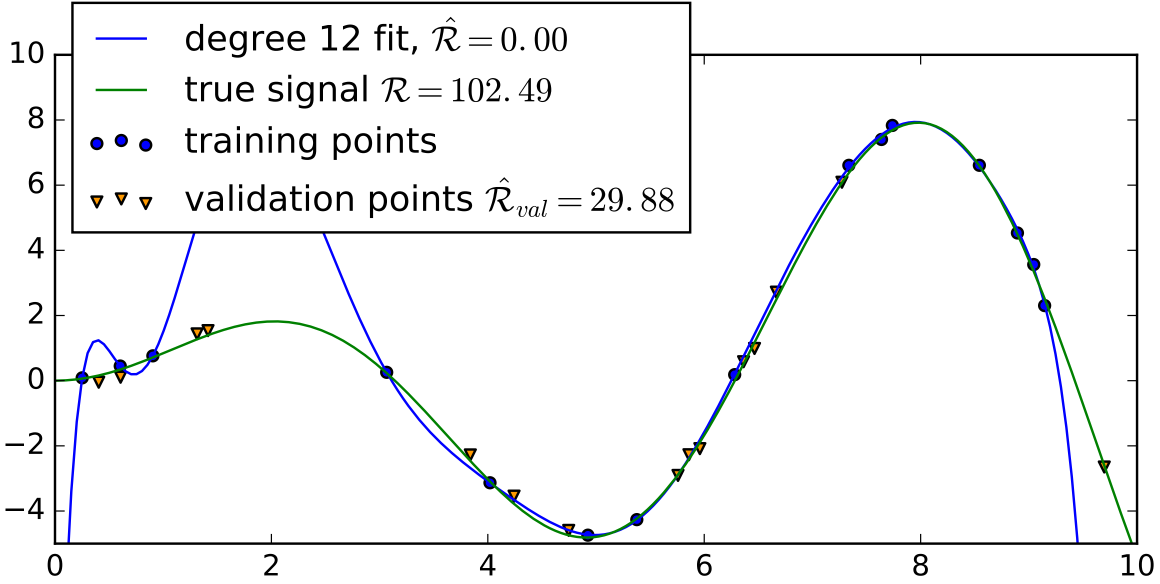
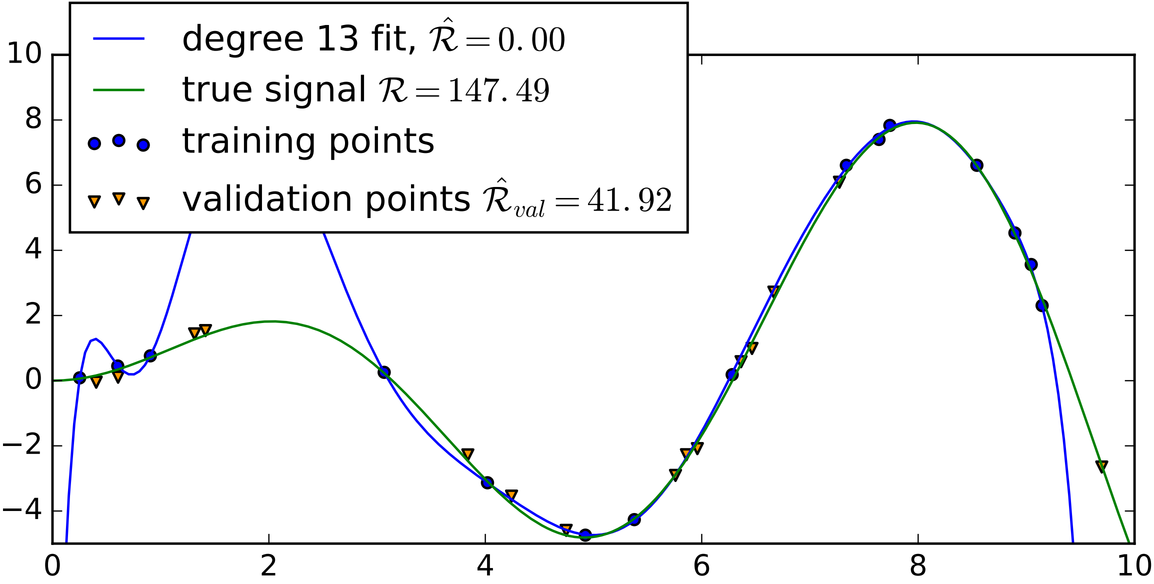
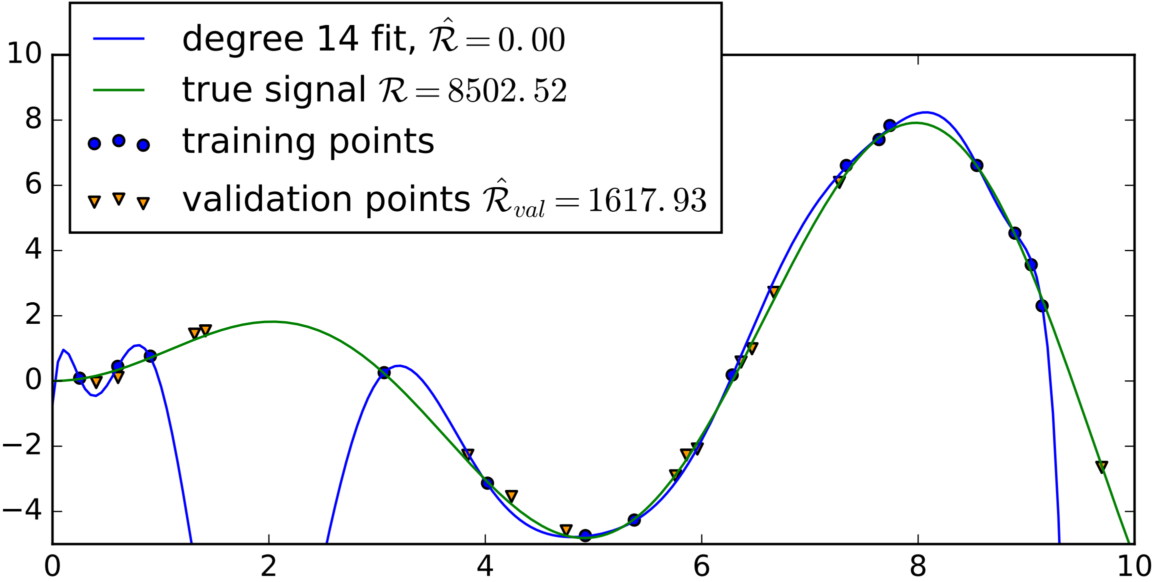


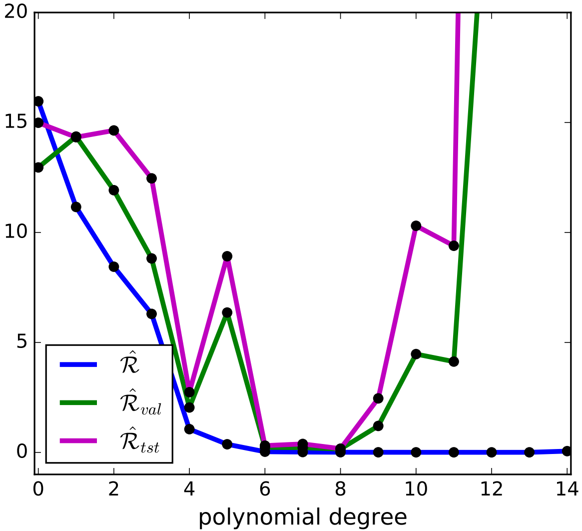
Illustration: learning polynomials (of different degrees)



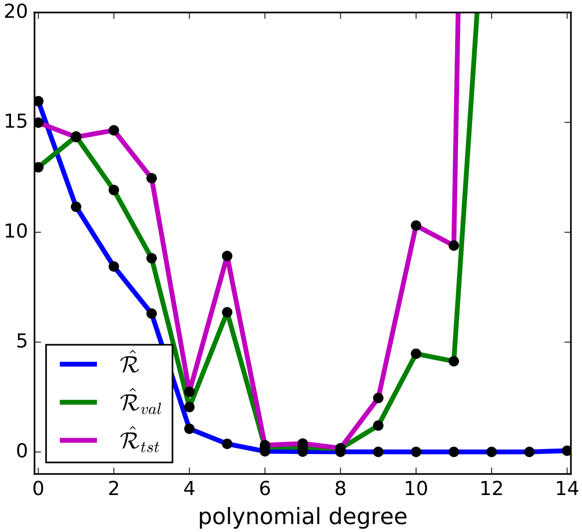

degree 13 fit, $\hat{\mathcal{R}} = 0.00$
true signal $\mathcal{R} = 147.49$
training points
validation points $\hat{\mathcal{R}}_{val} = 41.92$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10


degree 14 fit, $\hat{\mathcal{R}} = 0.00$
true signal $\mathcal{R} = 8502.52$
training points
validation points $\hat{\mathcal{R}}_{val} = 1617.93$
10
8
6
4
2
0
−2
−4
0
2
4
6
8
10

degree	$\hat{\mathcal{R}}$	$\hat{\mathcal{R}}_{val}$	$\mathcal{R}$
0	15.96	12.96	14.99
1	11.16	14.36	14.33
2	8.44	11.92	14.64
3	6.30	8.82	12.46
4	1.06	2.04	2.74
5	0.37	6.36	8.92
6	0.03	0.23	0.31
7	0.02	0.22	0.39
8	0.01	0.15	0.17
9	0.01	1.20	2.46
10	0.01	4.48	10.31
11	0.01	2.91	6.06
12	0.00	30.34	104.11
13	0.00	40.87	142.73
14	0.00	1622.37	8494.42



degree	$\hat{\mathcal{R}}$	$\hat{\mathcal{R}}_{val}$	$\mathcal{R}$
0	15.96	12.96	14.99
1	11.16	14.36	14.33
2	8.44	11.92	14.64
3	6.30	8.82	12.46
4	1.06	2.04	2.74
5	0.37	6.36	8.92
6	0.03	0.23	0.31
7	0.02	0.22	0.39
8	**0.01**	**0.15**	**0.17**
9	0.01	1.20	2.46
10	0.01	4.48	10.31
11	0.01	2.91	6.06
12	0.00	30.34	104.11
13	0.00	40.87	142.73
14	0.00	1622.37	8494.42



From infinite to finite hyperparameter set

Typically, finitely many choices for model classes, optimizers

- we can/have to try them all

For hyperparameters typically too many or even infinitely many choices:

- neural network depth and widths
- learning rate(s)
- regularization strength
- number of training iterations

Subsample or discretize in a reasonable way, e.g.

- regularization: exponentially, e.g. $\lambda \in \{2^{-20}, 2^{-19}, \dots, 2^{20}\}$
- learning rate: exponential scale in known range, e.g. $\eta \in \{10^{-6}, 3 \cdot 10^{-6}, \dots, 10^{-1}\}$
- iterations/epochs: linearly, e.g. $T \in \{1, 5, 10, 50, 100, \dots\}$ (or adaptive using $\mathcal{R}_{\mathsf{val}}$)

If model selection picks value at boundary of range, choose a larger range.

Selecting multiple hyperparameters: $\eta_1 \in A_1, \eta_2 \in A_2, \ldots, \eta_J \in A_J$

Selecting multiple hyperparameters: $\eta_1 \in A_1, \eta_2 \in A_2, \ldots, \eta_J \in A_J$

Grid search (good, but rarely practical for more than $J > 2$)

- $\eta_{\text{total}} = (\eta_1, \ldots, \eta_K)$, $A_{\text{total}} = A_1 \times A_2 \times \cdots \times A_J$
- try all $|A_1| \times |A_2| \times \cdots \times |A_J|$ combinations

Selecting multiple hyperparameters: $\eta_1 \in A_1, \eta_2 \in A_2, \ldots, \eta_J \in A_J$

Grid search (good, but rarely practical for more than $J > 2$)

- $\eta_{\text{total}} = (\eta_1, \ldots, \eta_K)$, $A_{\text{total}} = A_1 \times A_2 \times \cdots \times A_J$
- try all $|A_1| \times |A_2| \times \cdots \times |A_J|$ combinations

Greedy search (sometimes practical, but suboptimal)

- for $j = 1, \ldots, J$:
- $\quad \eta_j^* \leftarrow$ pick $\eta_j \in A_j$, with η_k for $k > j$ fixed at a reasonable default

Problem: what are 'reasonable defaults'?

Selecting multiple hyperparameters: $\eta_1 \in A_1, \eta_2 \in A_2, \dots, \eta_J \in A_J$

Grid search (good, but rarely practical for more than $J > 2$)

- $\eta_{\text{total}} = (\eta_1, \dots, \eta_K)$, $A_{\text{total}} = A_1 \times A_2 \times \cdots \times A_J$
- try all $|A_1| \times |A_2| \times \cdots \times |A_J|$ combinations

Greedy search (sometimes practical, but suboptimal)

- for $j = 1, \dots, J$:
- $\quad \eta_j^* \leftarrow$ pick $\eta_j \in A_j$, with η_k for $k > j$ fixed at a reasonable default

Problem: what are 'reasonable defaults'?

Trust in a higher authority (avoid this!)

- set hyperparameters to values from the literature, e.g. $\lambda = 1$

Problem: this can fail horribly if the situation isn't completely identical.

Be sceptical of models with many (> 2) hyperparameters!

Trap 2: domain adaptation

Imagine you develop a method for *domain adaptation (DA)*:

- given: labeled data from some data distribution ("source")
- goal: create a classifier that works for a different data distribution ("target"), from which only unlabeled data is given

How to do model-selection?

Trap 2: domain adaptation

Imagine you develop a method for *domain adaptation (DA)*:

- given: labeled data from some data distribution ("source")
- goal: create a classifier that works for a different data distribution ("target"), from which only unlabeled data is given

How to do model-selection?

Reverse validation:

- train DA classifier on labeled source, use it to predict labels for unlabeled target data
- train DA classifier on (now labeled) target data, predict on source
- compare predicted labels and ground truth labels on source

Tedious and not perfect, but better than nothing...

[E. Zhong, W. Fan, Q. Yang, O. Verscheure, J. Ren. "Cross validation framework to choose amongst models and datasets for transfer learning." ECML/PKDD 2010], [L. Bruzzone, M. Marconcini, "Domain adaptation problems: A DASVM classification technique and a circular validation strategy", TPAMI 2010].

Trap 3: zero-shot learning

Imagine: at prediction time, *classes will occur that are not in the training set*, e.g.

- training data: images of 40 animal classes
- prediction time: images of 10 other animals classes

How to do model-selection?

Trap 3: zero-shot learning

Imagine: at prediction time, *classes will occur that are not in the training set*, e.g.

- training data: images of 40 animal classes
- prediction time: images of 10 other animals classes

How to do model-selection?

Simulate prediction situation:

- split training data into actual train and val data with disjoint classes (e.g. 32-8)
- train zero-shot classifier on actual train data to predict validation classes
- evaluate on val
- potential repeat multiplie times with different splits and average

Tedious to implement, and needs sufficiently many classes, but does the trick...

[Y. Xian, CHL, B. Schiele, Z. Akata, "Zero-shot learning-A comprehensive evaluation of the good, the bad and the ugly", TPAMI 2018]

Trap 4: outlier detection / out-of-distribution prediction

Imagine you want a system that knows what data it is useful for

- goal: classification of 10 object classes
- task: if other data occurs, refuse to classify

How to do model-selection?

Trap 4: outlier detection / out-of-distribution prediction

Imagine you want a system that knows what data it is useful for

- goal: classification of 10 object classes
- task: if other data occurs, refuse to classify

How to do model-selection?

Unclear.

- option 1: make up "out-of-distribution" data, e.g. random noise
  Problem: system might only detect noise, not other out-of-distribution data
- option 2: leave out some classes, use those as 'outliers' (like zero-shot)
  Problem: system might only detect new object classes, not e.g. pure noise
- option 3: use optimization to create "difficult" data, like adversarial examples
  Problem: very hard optimization problem, might still not work

Same problem for actual evaluation, because no test set exists.

Trap 5: weakly-supervised learning

Imagine you learn from *weakly annotated data*, e.g.

- goal: object detection, i.e. predicting object bounding boxes
- given: training image with per-image labels

How to do model-selection?

Trap 5: weakly-supervised learning

Imagine you learn from *weakly annotated data*, e.g.

- goal: object detection, i.e. predicting object bounding boxes
- given: training image with per-image labels

How to do model-selection?

I have no clue.

- option 1: choose model and hyperparameters heuristically, hope for the best
- option 2: choose model and hyperparameters on another dataset that has bounding box annotation, hope for the best
- option 3: admit that you will need some images with bounding box annotation for model selection $\rightarrow$ but: why not use (part of) this for training?

If you can come up with a better way, please let me know.

Summary: Machine Learning for Computer Vision

Step 1) Decide what exactly you want: the computer can't read your thoughts (yet)

- **inputs** x, **outputs** y, **loss** ℓ, **model class** f_θ (with hyperparameters)

Step 2) Collect and annotate data: a good model with bad data is useless

- **collect and annotate data**, ideally i.i.d. from the prediction-time data distribution

Step 3) Model training $\leftarrow$ often repeatedly

- **model training** on a training data, **model selection** on validation subset

Step 4) Model evaluation

- **evaluate model** on test set (that is disjoint from training/validation set)

The Big Picture

What's the purpose of (academic) science/research?

What's the purpose of (academic) science/research?

- getting rich and famous?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?
- pleasing your supervisor?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?
- pleasing your supervisor? ?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?
- pleasing your supervisor? ?
- advancing human knowledge and understanding?

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?
- pleasing your supervisor? ?
- advancing human knowledge and understanding? ✓

Two often overlooked aspects:

- You're doing research **for others**, not just for yourself.
  - do something good, and then tell others about it
- You have to **advance** something for this to make sense.
  - it's not actually about doing something good, it's about doing something better.

What's the purpose of (academic) science/research?

- getting rich and famous? ✗
- having fun while being paid for it? ?
- pleasing your supervisor? ?
- advancing human knowledge and understanding? ✓

Two often overlooked aspects:

- You're doing research **for others**, not just for yourself.
  - do something good, and then tell others about it ... and they have to believe you!
- You have to **advance** something for this to make sense.
  - it's not actually about doing something good, it's about doing something better.

One of the cornerstones of science is that results must be reproducible:

Any person knowledgeable in the field should be able to repeat your experiments and get the same results.

Reproducibility

- reproducing the experiments requires access to the data
  - use public data and provide information where you got it from
  - if you have to use your own data, make it publicly available
- reproducing the experiments requires knowledge of all components
  - clearly describe all components used, not just the new ones
  - list all implementation details (preferably release the source code)
  - list all (hyper)parameter values and how they were obtained

Your goal should not be to just to get papers accepted, but to create new knowledge.

Be your own method's harshest critic: you know best what's going on under the hood and what could have gone wrong in the process.

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, is the comparison *fair*?

Just that you invested a lot of work into a project doesn't mean it deserves to be published. A project must be able to *fail*, otherwise it's not research.

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

Best practice: scientific scrutiny

Machine learning can be used for many purposes. Not all are desirable.

DOI: 10.1037/pspa0000098 • Corpus ID: 1379347

Deep Neural Networks Are More Accurate Than Humans at Detecting Sexual Orientation From Facial Images

Yilun Wang, M. Kosinski • Published 1 February 2018 • Computer Science • Journal of Personality and Social Psychology

We show that faces contain much more information about sexual orientation than can be perceived or interpreted by the human brain. We used deep neural networks to extract features from 35,326 facial images. These features were entered into a logistic regression aimed at classifying sexual orientation. Given a single facial image, a classifier could correctly distinguish between gay and heterosexual men in 81% of cases, and in 71% of cases for women. Human judges achieved much lower accuracy: 61… Expand

Best practice: scientific scrutiny

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

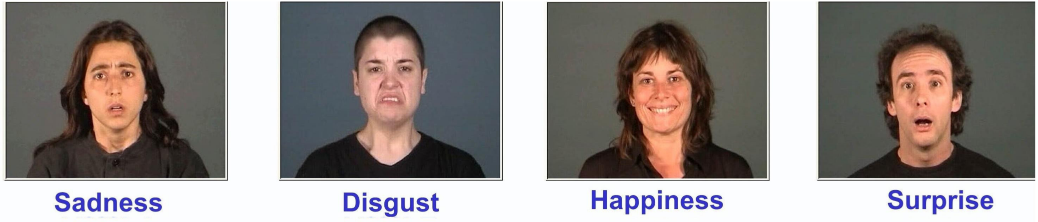
Is the data representative for the actual problem?

Example task: emotion recognition for autonomous driving

- e.g. 99% accuracy on DAFEX database for facial expressions

Example task: emotion recognition for autonomous driving

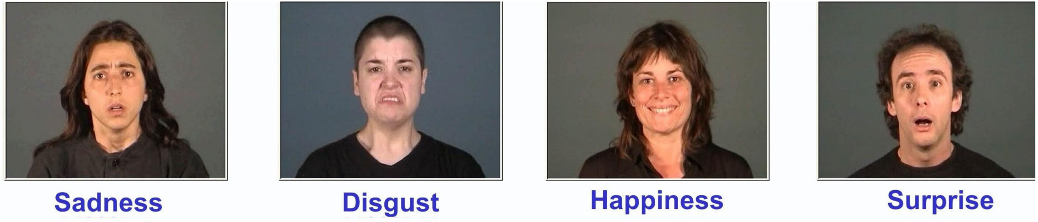
- e.g. 99% accuracy on DAFEX database for facial expressions



Is the data representative for the actual problem?

Example task: emotion recognition for autonomous driving

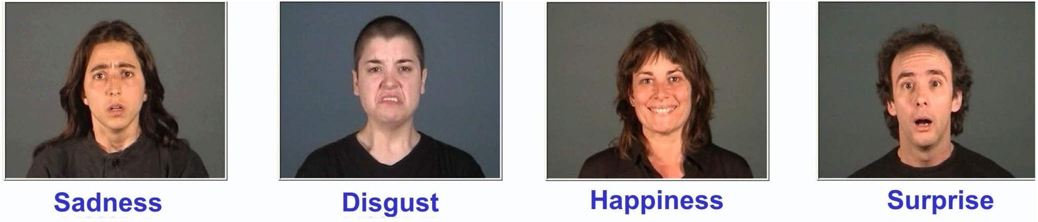
- e.g. 99% accuracy on DAFEX database for facial expressions



- frontal faces at fixed distance. Will that be the case in the car?

Example task: emotion recognition for autonomous driving

- e.g. 99% accuracy on DAFEX database for facial expressions

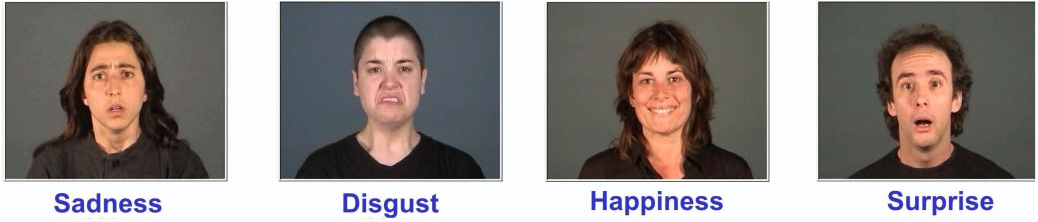


- frontal faces at fixed distance. Will that be the case in the car?
- perfect lighting conditions. Will that be the case in the car?

Is the data representative for the actual problem?

Example task: emotion recognition for autonomous driving

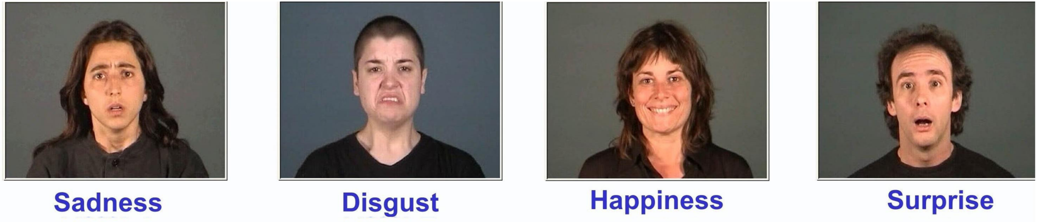
- e.g. 99% accuracy on DAFEX database for facial expressions



- frontal faces at fixed distance. Will that be the case in the car?
- perfect lighting conditions. Will that be the case in the car?
- homogeneous gray background. Will that be the case in the car?

Example task: emotion recognition for autonomous driving

- e.g. 99% accuracy on DAFEX database for facial expressions

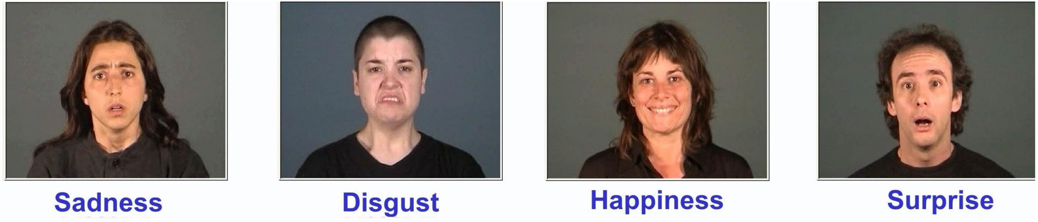


- frontal faces at fixed distance. Will that be the case in the car?
- perfect lighting conditions. Will that be the case in the car?
- homogeneous gray background. Will that be the case in the car?
- all faces are Italian actors. Will all drivers be Italian actors?

Is the data representative for the actual problem?

Example task: emotion recognition for autonomous driving

- e.g. 99% accuracy on DAFEX database for facial expressions

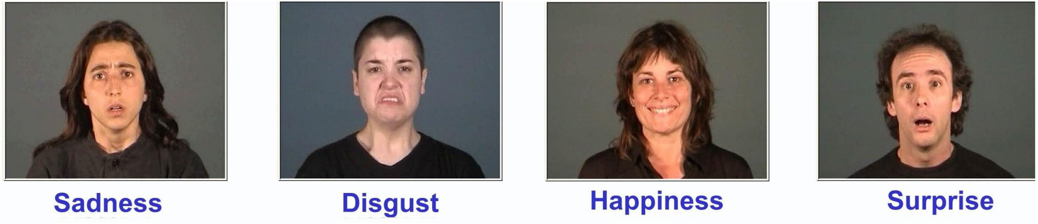


- frontal faces at fixed distance. Will that be the case in the car?
- perfect lighting conditions. Will that be the case in the car?
- homogeneous gray background. Will that be the case in the car?
- all faces are Italian actors. Will all drivers be Italian actors?
- the emotions are not real, but (over)acted!

Is the data representative for the actual problem?

Example task: emotion recognition for autonomous driving

- e.g. 99% accuracy on DAFEX database for facial expressions



- frontal faces at fixed distance. Will that be the case in the car?
- perfect lighting conditions. Will that be the case in the car?
- homogeneous gray background. Will that be the case in the car?
- all faces are Italian actors. Will all drivers be Italian actors?
- the emotions are not real, but (over)acted!

Little reason to believe that the system will work in practice $\rightarrow$ don't overclaim!

Images: https://i3.fbk.eu/resources/dafex-database-kinetic-facial-expressions

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

In the natural sciences, to show that an effect was not just random chance one uses repeated experiments.

In computer science, running the same code multiple times should give identical results, so on top we must use a form of explicit randomization.

- different random splits of the data, e.g. for training / evaluation
- different random initialization, e.g. for neural network training

Which of these make sense depends on the situation

- for benchmark datasets, test set is usually fixed
- for convex optimization methods, initialization plays no role, etc.

Repeats

We can think of accuracy on a validation/test set also as repeated experiments:

- we apply a fixed model many times, each time to a single example
- outcomes: $r_1, r_2, \dots, r_m$

Having done repeated experiments, one reports either

- the mean of outcomes and standard deviations

$$\bar{r} \pm \sigma \qquad \text{for} \qquad \bar{r} = \frac{1}{m}\sum_j r_t, \qquad \sigma = \sqrt{\frac{1}{m}\sum_j (r_j - \bar{r})^2}$$

"If you try this once, where can you expect the result to lie?"

or

- the mean and standard error of the mean

$$\bar{r} \pm \mathsf{SE} \qquad \text{for} \qquad \bar{r} = \frac{1}{m}\sum_j r_t, \quad \mathsf{SE} = \frac{\sigma}{\sqrt{m}}$$

"If you try this many times, where can you expect the mean to lie?"

Illustration as figure with error bars:

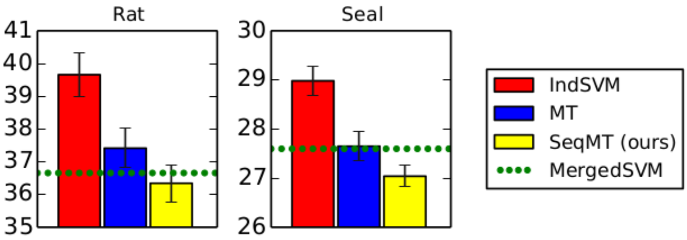
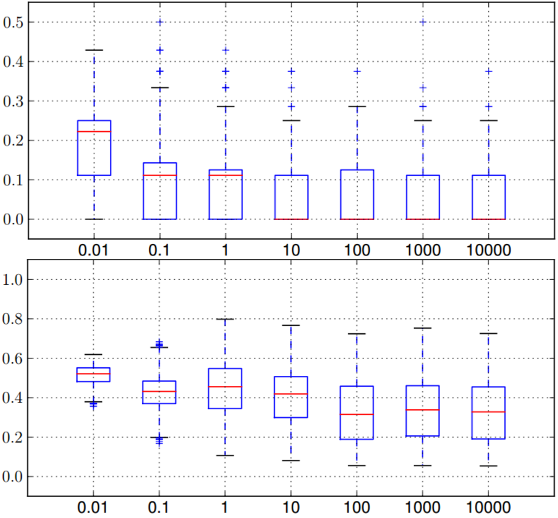


Illustration as table with error intervals:

error rate	Chimpanzee	Giant panda
component 1	25.47 ± 0.42	21.02 ± 0.58
component 2	23.94 ± 0.32	19.44 ± 0.50
combination	23.66 ± 0.33	19.23 ± 0.52

Quick (and dirty) check if results could be due to random chance:
do mean±error bars overlap?

More informative: **box and whisker plots** (short: box plot)



- red line: data median
- blue body: 25%–75% quantile
- whiskers: value range
- blue markers: outliers

sometimes additional symbols for mean or uncertainty of the median

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random $(> 14\%)$

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random $(> 14\%)$

- A: 40% accuracy on 1000 examples
- B: 60% accuracy on 1000 examples

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random $(> 14\%)$

- A: 40% accuracy on 1000 examples
- B: 60% accuracy on 1000 examples

small chance that effect is random $(< 10^{-10})$

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random ($> 14\%$)

- A: 40% accuracy on 1000 examples
- B: 60% accuracy on 1000 examples

small chance that effect is random ($< 10^{-10}$)

- A: 99.77% accuracy on 10000 ex.s
- B: 99.79% accuracy on 10000 ex.s

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear: 5/7 vs 6/7 or 710/1000 vs 850/1000?
always report test set size!

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random ($> 14\%$)

- A: 40% accuracy on 1000 examples
- B: 60% accuracy on 1000 examples

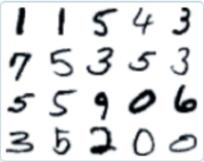
small chance that effect is random ($< 10^{-10}$)

- A: 99.77% accuracy on 10000 ex.s
- B: 99.79% accuracy on 10000 ex.s

high chance that effect is random ($> 20\%$)

MNIST

who is the best in MNIST ?



MNIST 50 results collected

Units: error %

> Classify handwriten digits. Some additional results are available on the original dataset page.

Result	Method	Venue	Details
0.21%	Regularization of Neural Networks using DropConnect	ICML 2013	
0.23%	Multi-column Deep Neural Networks for Image Classification	CVPR 2012	
0.23%	APAC: Augmented PAttern Classification with Neural Networks	arXiv 2015	
0.24%	Batch-normalized Maxout Network in Network	arXiv 2015	Details

Quiz: Randomness in Repeated Experiments

We can think of accuracy on a test set as repeated experiments:
apply a fixed model many times, each time to a single example.

Which of these differences are real and which are random fluctuations?

- method A: 71% accuracy
- method B: 85% accuracy

unclear. 5/7 vs 6/7 or 710/1000 vs 850/1000?

- A: 40% accuracy on 10 examples
- B: 60% accuracy on 10 examples

high chance that effect is random ($> 14\%$)

- A: 40% accuracy on 1000 examples
- B: 60% accuracy on 1000 examples

small chance that effect is random ($< 10^{-10}$)

- A: 99.77% accuracy on 10000 ex.s
- B: 99.79% accuracy on 10000 ex.s

high chance that effect is random ($> 20\%$)

Actual procedure: **statistical significance test!**

Generalizability

In the natural sciences, to show that an effect is generally relevant one shows it in multiple model organisms.

In machine learning, this corresponds to multiple datasets and/or multiple models/architectures.

To show that a newly proposed model is better than earlier ones:

- show its superiority on multiple datasets
- idealy, show it for multiple different tasks (classification, segmentation, detection, ...)

To show that a new method or a new component is beneficial:

- show its superiority for multiple models
- idealy, show it for multiple different tasks (classification, segmentation, detection, ...)

Generalizability

In the natural sciences, to show that an effect is generally relevant one shows it in multiple model organisms.

In machine learning, this corresponds to multiple datasets and/or multiple models/architectures.

To show that a newly proposed model is better than earlier ones:

- show its superiority on multiple datasets
- idealy, show it for multiple different tasks (classification, segmentation, detection, ...)

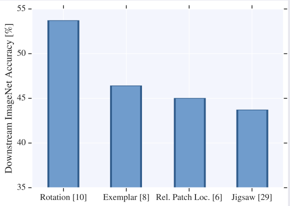
To show that a new method or a new component is beneficial:

- show its superiority for multiple models
- idealy, show it for multiple different tasks (classification, segmentation, detection, ...)

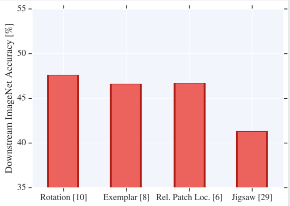
Datasets are proxies for real-world situations. Nobody should <u>actually</u> care about the results on the dataset.

Observations: comparing on a single model architecture and/or dataset can be misleading.

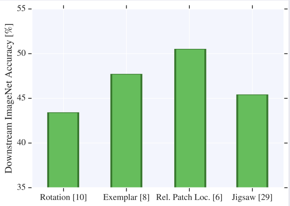
Example: Unsupervised Pretraining for Image Classification



ResNet50



ResNet50v1



ResNet50v2

Method [10] looks great for original ResNet50, good for ResNet50v1, bad for ResNet50v2.

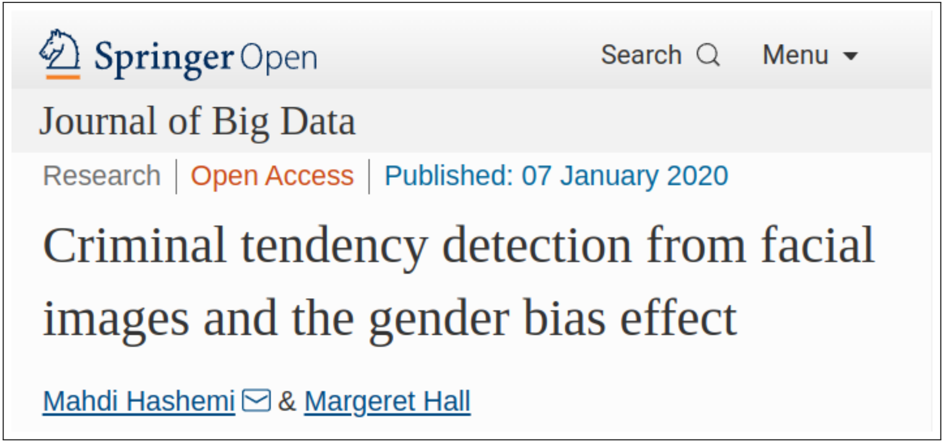
Designing convincing experiments is harder than it seems.

Source: [A. Kolesnikov, X. Zhai, L. Beyer: "Revisiting Self-Supervised Visual Representation Learning", CVPR2019]

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

Best practice: scientific scrutiny



SpringerOpen Search Menu

Journal of Big Data

Research | Open Access | Published: 07 January 2020

Criminal tendency detection from facial images and the gender bias effect

Mahdi Hashemi & Margeret Hall

Paper reports 97% accuracy on 10.000 images (10-fold cross-validation).

Source: [M. Hashemi, M. Hall. "Criminal tendency detection from facial images and the gender bias effect", Journal of Big Data (7)2, 2020]

Best practice: scientific scrutiny

Reported effect was explainable purely by a bias in the dataset collection.

criminals: *mugshots*

non-criminals: *normal photos from the web*

Note: original paper was withdrawn because the authors had not asked for ethics approval.

more information [M. Hashemi, M. Hall. "Criminal tendency detection from facial images and the gender bias effect", Journal of Big Data (7)2, 2020]
Images: https://huggingface.co/spaces/dalle-mini/dalle-mini (`"mugshot of a criminal"` vs `"face of a normal person from a dating website"`)

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

Always check your results for plausibility.

Sometimes, good results are the effect of bugs or (test) data leakage.

If it looks too good to be true, it's probably not true.

arXiv > cs > arXiv:2204.07953v1

Search...

Help | Advanced

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 17 Apr 2022 (this version), **latest version 19 May 2022** *(v3)]*

Learning with Signatures

J. de Curtò, I. de Zarzà, Carlos T. Calafate, Hong Yan

In this work we investigate the use of the Signature Transform in the context of Learning. Under this assumption, we advance a supervised framework that provides state-of-the-art classification accuracy with the use of very few labels without the need of credit assignment and with minimal or no overfitting. We leverage tools from harmonic analysis by the use of the signature and log-signature and use as a score function RMSE and MAE Signature and log-signature. We develop a closed-form equation to compute probably good optimal scale factors. Classification is performed at the CPU level orders of magnitude faster than other methods. We report results on AFHQ dataset, Four Shapes, MNIST and CIFAR10 achieving 100% accuracy on all tasks.

Best practice: scientific scrutiny

Always check your results for plausibility.

Sometimes, good results are the effect of bugs or (test) data leakage.

If it looks too good to be true, it's probably not true.

arXiv > cs > arXiv:2204.07953v1

Search...

Help | Advanced

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 17 Apr 2022 (this version), **latest version 19 May 2022** *(v3)]*

Learning with Signatures

J. de Curtò, I. de Zarzà, Carlos T. Calafate, Hong Yan

In this work we investigate the use of the Signature Transform in the context of Learning. Under this assumption, we advance a supervised framework that provides state-of-the-art classification accuracy with the use of very few labels without the need of credit assignment and with minimal or no overfitting. We leverage tools from harmonic analysis by the use of the signature and log-signature and use as a score function RMSE and MAE Signature and log-signature. We develop a closed-form equation to compute probably good optimal scale factors. Classification is performed at the CPU level orders of magnitude faster than other methods. We report results on AFHQ dataset, Four Shapes, MNIST and CIFAR10 achieving 100% accuracy on all tasks.

Always check your results for plausibility.

Sometimes, good results are the effect of bugs or (test) data leakage.

If it looks too good to be true, it's probably not true.

arXiv > cs > arXiv:2204.07953v1

Search...

Help | Advanced

Computer Science > Computer Vision and Pattern Recognition

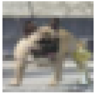
*[Submitted on 17 Apr 2022 (this version), **latest version 19 May 2022** (v3)]*

Learning with Signatures

J. de Curtò, I. de Zarzà, Carlos T. Calafate, Hong Yan

In this work we investigate the use of the Signature Transform in the context of Learning. Under this assumption, we advance a supervised framework that provides state-of-the-art classification accuracy with the use of very few labels without the need of credit assignment and with minimal or no overfitting. We leverage tools from harmonic analysis by the use of the signature and log-signature and use as a score function RMSE and MAE Signature and log-signature. We develop a closed-form equation to compute probably good optimal scale factors. Classification is performed at the CPU level orders of magnitude faster than other methods. We report results on AFHQ dataset, Four Shapes, MNIST and CIFAR10 achieving 100% accuracy on all tasks.

cat?



cat?

CIFAR test set has annotation errors, 100% should be impossible.

Image: `https://labelerrors.com/`

Story: see `https://www.reddit.com/r/MachineLearning/comments/u7ouxh/r_authors_claim_to_have_solved_mnist_and_cifar/`

Scientific scrutiny

- are you solving a relevant problem?
- is the data representative for the actual problem?
- are the results explainable just by random chance/noise?
- does the system make use of artifacts in the data?
- can you rule out bugs in the code or data leakage?
- if you compare to baselines, are they truly comparable?

If you compare to baselines, are they truly comparable?

If you compare to baselines, are they truly comparable?

Not always easy to answer:

Do the methods have solve the same problem?

- previous work: object classification
- proposed work: object detection

If you compare to baselines, are they truly comparable?

Not always easy to answer:

Do the methods have solve the same problem?

- previous work: object classification
- proposed work: object detection

Do they evaluate their performance on the same data?

- previous work: PASCAL VOC 2010
- proposed work: PASCAL VOC 2012

If you compare to baselines, are they truly comparable?

Not always easy to answer:

Do the methods have solve the same problem?

- previous work: object classification
- proposed work: object detection

Do they evaluate their performance on the same data?

- previous work: PASCAL VOC 2010
- proposed work: PASCAL VOC 2012

Do they have access to the same data for training?

- previous work: ImageNet
- proposed work: ImageNet, pretrained on JFT-300M

Do they have access to the same annotation?

- previous work: MSCOCO with bounding-box annotation
- proposed work: MSCOCO with segmentation annotation

Do they have access to the same annotation?

- previous work: MSCOCO with bounding-box annotation
- proposed work: MSCOCO with segmentation annotation

Do they use the same data augmentation?

- previous work: CIFAR-10 images
- proposed work: CIFAR-10 images with translation/rotation/crop augmentation

Do they have access to the same annotation?

- previous work: MSCOCO with bounding-box annotation
- proposed work: MSCOCO with segmentation annotation

Do they use the same data augmentation?

- previous work: CIFAR-10 images
- proposed work: CIFAR-10 images with translation/rotation/crop augmentation

for runtime comparisons: Do you use comparable hardware?

- previous work: laptop with 3.4 GHz quad-core CPU
- proposed work: workstation with 8 A100 GPUs

Do they have access to the same annotation?

- previous work: MSCOCO with bounding-box annotation
- proposed work: MSCOCO with segmentation annotation

Do they use the same data augmentation?

- previous work: CIFAR-10 images
- proposed work: CIFAR-10 images with translation/rotation/crop augmentation

for runtime comparisons: Do you use comparable hardware?

- previous work: laptop with 3.4 GHz quad-core CPU
- proposed work: workstation with 8 A100 GPUs

for runtime comparisons: Do you use comparable languages/implementations?

- previous work: method implemented in pytorch
- proposed work: method implemented in tensorflow

Is/was model selection for the baselines done thoroughly?

- baseline: hyperparameter set arbitrarily ad hoc
- proposed method: exhaustive hyperparameter selection

Now, is the new method better, or just the baseline worse than it could be?

Is/was model selection for the baselines done thoroughly?

- baseline: hyperparameter set arbitrarily ad hoc
- proposed method: exhaustive hyperparameter selection

Now, is the new method better, or just the baseline worse than it could be?

Positive exception:

Nowozin, Bakır. "Discriminative Subsequence Mining for Action Classification", ICCV 2007

Method	**KTH accuracy**
Niebles et al. [11], LOO, pLSA	81.50
Dollár et al. [4], LOO, SVM RBF	80.66
Schuldt et al. [17], splits, SVM match	71.71
Ke et al. [7], splits, forward feat.-sel.	62.94
Subsequence Boosting, $B = 12$, splits	84.72

Is/was model selection for the baselines done thoroughly?

- baseline: hyperparameter set arbitrarily ad hoc
- proposed method: exhaustive hyperparameter selection

Now, is the new method better, or just the baseline worse than it could be?

Positive exception:

Nowozin, Bakır. "Discriminative Subsequence Mining for Action Classification", ICCV 2007

Method	**KTH accuracy**
Niebles et al. [11], LOO, pLSA	81.50
Dollár et al. [4], LOO, SVM RBF	80.66
Schuldt et al. [17], splits, SVM match	71.71
Ke et al. [7], splits, forward feat.-sel.	62.94
baseline SVM linear bin, 1-vs-1	83.33
baseline SVM RBF bin, 1-vs-1	85.19
baseline SVM χ^2 bin, 1-vs-1	87.04
Subsequence Boosting, $B = 12$, splits	84.72

Take home messages

Don't use machine learning unless you have good reasons.

Keep in mind: all data is random and all numbers are uncertain.

Learning is about finding a model that works on future data.

Proper model-selection is hard. Avoid free (hyper)parameters!

A test set is for evaluating a single final ultimate model, nothing else.

You don't do research for yourself, but for others (users, readers, society, ...).

Results must be convincing: honest, reproducible, not overclaimed.