Statistical Machine Learning

https://cvml.ist.ac.at/courses/SML_W18

Christoph Lampert

IST AUSTRIA

Institute of Science and Technology

Spring Semester 2018/2019

Lecture 5

Overview (tentative)

Date		no.	Topic
Oct 08	Mon	1	A Hands-On Introduction
Oct 10	Wed	–	self-study (Christoph traveling)
Oct 15	Mon	2	Bayesian Decision Theory
			Generative Probabilistic Models
Oct 17	Wed	3	Discriminative Probabilistic Models
			Maximum Margin Classifiers
Oct 22	Mon	4	Generalized Linear Classifiers, Optimization
Oct 24	Wed	5	Evaluating Predictors; Model Selection
Oct 29	Mon	–	self-study (Christoph traveling)
Oct 31	Wed	6	Overfitting/Underfitting, Regularization
Nov 05	Mon	7	Learning Theory I: classical/Rademacher bounds
Nov 07	Wed	8	Learning Theory II: miscellaneous
Nov 12	Mon	9	Probabilistic Graphical Models I
Nov 14	Wed	10	Probabilistic Graphical Models II
Nov 19	Mon	11	Probabilistic Graphical Models III
Nov 21	Wed	12	Probabilistic Graphical Models IV
until Nov 25			final project

Evaluating Predictors

So, you've trained a predictor, $f : \mathcal{X} \to \mathcal{Y}$. How good is it really?

- The loss on the training set, $\mathcal{D} = \{ (x^1, y^1), \dots, (x^n, y^n) \}$,

$$\hat{\mathcal{R}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(y^i, f(x^i))$$

  tells us little about the quality of a learned predictor. Reporting it would be misleading as best.

- Really, we would care about the expected loss (=generalization loss),

$$\mathcal{R}(f) = \mathop{\mathbb{E}}_{(x,y) \sim p(x,y)} \ell(y, f(x)).$$

  Unfornately, we cannot compute it, because $p(x, y)$ is unknown.

- In practice, we use a a test set,

$$\mathcal{D}_{\mathsf{tst}} = \{ (x^1, y^1), \dots, (x^m, y^m) \},$$

  i.e. examples that were not used for training, and compute

$$\hat{\mathcal{R}}_{\mathsf{tst}}(f) = \frac{1}{m} \sum_{i=1}^{m} \ell(y^i, f(x^i))$$

  Why?

So, you've trained a predictor, $f : \mathcal{X} \to \mathcal{Y}$. How good is it really?

- The loss on the training set, $\mathcal{D} = \{ (x^1, y^1), \dots, (x^n, y^n) \}$,

$$\hat{\mathcal{R}}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(y^i, f(x^i))$$

  tells us little about the quality of a learned predictor. Reporting it would be misleading as best.

- Really, we would care about the expected loss (=generalization loss),

$$\mathcal{R}(f) = \mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \ell(y, f(x)).$$

  Unfornately, we cannot compute it, because $p(x, y)$ is unknown.

- In practice, we use a a test set,

$$\mathcal{D}_{\text{tst}} = \{ (x^1, y^1), \dots, (x^m, y^m) \},$$

  i.e. examples that were not used for training, and compute

$$\hat{\mathcal{R}}_{\text{tst}}(f) = \frac{1}{m} \sum_{i=1}^{m} \ell(y^i, f(x^i))$$

  Why? Let's look at $\hat{\mathcal{R}}_{\text{tst}}(f)$ as an **estimator** of $\mathcal{R}(f)$.

Excurse: Estimators

Estimators

An estimator is a rule for calculating an estimate, $\hat{E}(S)$, of a quantity E based on observed data, S. If S is random, then $\hat{E}(S)$ is also random.

Properties of estimators: bias

Let $\hat{E}$ be an estimator of E. We can compute the expected value of the estimate, $\mathbb{E}_S[\hat{E}(S)]$, and define:

$$\mathrm{bias}(\hat{E}) = \mathbb{E}_S[\hat{E}(S)] - E$$

Properties of estimators: unbiasedness

If $\hat{E}$ is an estimator of E, we call it unbiased, if

$$\mathrm{bias}(\hat{E}) = 0 \qquad \text{(i.e. } \mathop{\mathbb{E}}_S[\hat{E}(S)] = E)$$

If $\hat{E}$ is unbiased, we can think of it as a noisy version of E.

Example: Estimating the mean of a Gaussian

Examples: let $S = \{z^1, z^2, \ldots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$. We look at different estimators for μ:

- $\hat{E}(S) = 1$ has bias $1 - \mu$. bias$(\hat{E}) = \mathbb{E}_S \, \hat{E}(S) - \mu = 1 - \mu$

Example: Estimating the mean of a Gaussian

Examples: let $S = \{z^1, z^2, \ldots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$. We look at different estimators for μ:

- $\hat{E}(S) = 1$ has bias $1 - \mu$. bias$(\hat{E}) = \mathbb{E}_S \, \hat{E}(S) - \mu = 1 - \mu$
- $\hat{E}(S) = \frac{1}{n} \sum_{i=1}^{n} z^i$ is unbiased.

Example: Estimating the mean of a Gaussian

Examples: let $S = \{z^1, z^2, \dots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$. We look at different estimators for μ:

- $\hat{E}(S) = 1$ has bias $1 - \mu$. bias$(\hat{E}) = \mathbb{E}_S\, \hat{E}(S) - \mu = 1 - \mu$
- $\hat{E}(S) = \frac{1}{n}\sum_{i=1}^{n} z^i$ is unbiased.

$$\mathbb{E}_S[\hat{E}(S)] = \mathbb{E}_S[\tfrac{1}{n}\textstyle\sum_i z^i] = \tfrac{1}{n}\textstyle\sum_i \mathbb{E}_S[z^i] = \tfrac{1}{n}\textstyle\sum_i \mu = \mu$$

Example: Estimating the mean of a Gaussian

Examples: let $S = \{z^1, z^2, \dots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$. We look at different estimators for μ:

- $\hat{E}(S) = 1$ has bias $1 - \mu$. bias$(\hat{E}) = \mathbb{E}_S\, \hat{E}(S) - \mu = 1 - \mu$
- $\hat{E}(S) = \frac{1}{n}\sum_{i=1}^{n} z^i$ is unbiased.

$$\mathbb{E}_S[\hat{E}(S)] = \mathbb{E}_S[\tfrac{1}{n}\textstyle\sum_i z^i] = \tfrac{1}{n}\textstyle\sum_i \mathbb{E}_S[z^i] = \tfrac{1}{n}\textstyle\sum_i \mu = \mu$$

- $\hat{E}(S) = z^1$ is unbiased: $\mathbb{E}_S[\hat{E}(S)] = \mathbb{E}_S[z^1] = \mu$

Example: Estimating the mean of a Gaussian

Examples: let $S = \{z^1, z^2, \dots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$. We look at different estimators for μ:

- $\hat{E}(S) = 1$ has bias $1 - \mu$. bias$(\hat{E}) = \mathbb{E}_S \hat{E}(S) - \mu = 1 - \mu$
- $\hat{E}(S) = \frac{1}{n}\sum_{i=1}^n z^i$ is unbiased.
$$\mathbb{E}_S[\hat{E}(S)] = \mathbb{E}_S[\tfrac{1}{n}\sum_i z^i] = \tfrac{1}{n}\sum_i \mathbb{E}_S[z^i] = \tfrac{1}{n}\sum_i \mu = \mu$$
- $\hat{E}(S) = z^1$ is unbiased: $\mathbb{E}_S[\hat{E}(S)] = \mathbb{E}_S[z^1] = \mu$
- $\hat{E}(S) = \frac{1}{n} + \frac{1}{n}\sum_{i=1}^n z^i$ has bias $\frac{1}{n}$

Example: Stochastic Gradient Descent

Reminder: we wanted to optimize

$$f(\theta) = \sum_{j=1}^{n} f_j(\theta)$$

Instead of

$$v := \nabla f(\theta)$$

we use

$$\hat{v} := n \nabla f_i(\theta) \quad \text{with} \quad i \overset{\text{uniformly}}{\sim} \{1, \ldots, n\}$$

Claim: $\hat{v}$ is an unbiased estimator for v.

Example: Stochastic Gradient Descent

Reminder: we wanted to optimize

$$f(\theta) = \sum_{j=1}^{n} f_j(\theta)$$

Instead of

$$v := \nabla f(\theta)$$

we use

$$\hat{v} := n \nabla f_i(\theta) \quad \text{with} \quad i \overset{\text{uniformly}}{\sim} \{1, \dots, n\}$$

Claim: $\hat{v}$ is an unbiased estimator for v.

$$\mathop{\mathbb{E}}_{i}[\hat{v}] = \sum_{i=1}^{n} p(i)\hat{v}[i] = \sum_{i=1}^{n} \frac{1}{n} n \nabla f_i(\theta) = \sum_{i=1}^{n} \nabla f_i(\theta) = \nabla f(\theta)$$

How far is one estimate, $\hat{E}(S)$, from its expected value, $\mathbb{E}_S[\hat{E}(S)]$?

Properties of estimators: variance

$$\mathrm{Var}(\hat{E}) = \mathop{\mathbb{E}}_{S}\left[\left(\hat{E}(S) - \mathop{\mathbb{E}}_{S}[\hat{E}(S)]\right)^2\right]$$

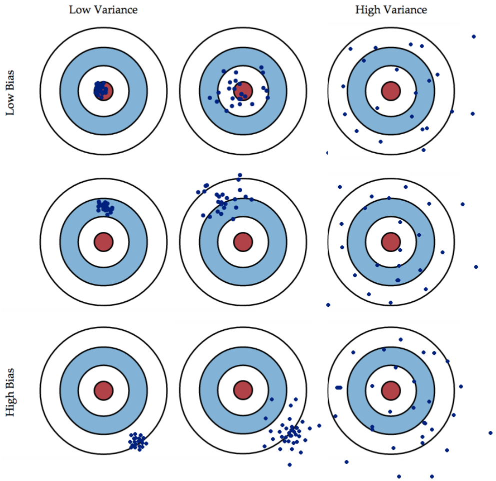
If $\mathrm{Var}(\hat{E})$ is large, then the estimate for different S differ a lot.

Examples:
Let $S = \{z^1, z^2, \dots, z^n\}$ be independent samples from $\mathcal{N}(x; \mu, \sigma^2)$.
We look at different estimators for μ:

- $\hat{E}(S) = 1$ has variance 0.
- $\hat{E}(S) = \frac{1}{n}\sum_{i=1}^{n} z_i$ has variance $\frac{\sigma^2}{n}$ *(exercise)*
- $\hat{E}(S) = z_1$ has variance σ^2
- $\hat{E}(S) = \frac{1}{n-1}\sum_{i=1}^{n} z_i$ has variance $?$ *(exercise)*

Bias-Variance Trade-Off

It's good to have small or no bias, and it's good to have small variance.



If you can't have both at the same time, look for a reasonable trade-off.

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

What if we get more and more data, $S_n = \{z_1, \dots, z_n\}$ for $n \to \infty$?

Properties of estimators: consistency

An estimator $\hat{E}$ is called consistent, if

$$\hat{E}(S_n) \to E \qquad \text{for} \quad n \to \infty.$$

Convergence is "in probability", i.e. it means,

$$\lim_{n\to\infty} \Pr\{\ |\hat{E}(S_n) - E| > \epsilon\ \} = 0.$$

Any estimator $\hat{E}$ with bias$(\hat{E}) \overset{n\to\infty}{\to} 0$ and Var$(\hat{E}) \overset{n\to\infty}{\to} 0$ is consistent.

Proof... follows from later observations

Back to learning...

Is

$$\hat{\mathcal{R}}_{\mathsf{tst}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(\, y_i, f(x_i))$$

a good estimator of

$$\mathcal{R}(f) = \mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \ell(\, y, f(x)\,)$$

Yes, if we use the right data:

Test error as an unbiased estimator

If $\mathcal{D}_{\mathsf{tst}} = \{\,(x^1, y^1), \dots, (x^m, y^m)\,\}$ are sampled independently from the distribution $p(x,y)$, and f was chosen independently of them.
Then $\hat{\mathcal{R}}_{\mathsf{tst}}$ is an unbiased and consistent estimator of $\mathcal{R}$:

Otherwise? Things might go wrong...

Proof: unbiased

- $\mathcal{D}$ is a set of random variables, $(X^1, Y^1), \dots, (X^m, Y^m) \in \mathcal{X} \times \mathcal{Y}$.
- All $(X^1, Y^1), \dots, (X^m, Y^m)$ are independent with distribution p.
- For fixed functions f, ℓ, chosen independently of $\mathcal{D}$

$$\ell(Y^1, f(X^1)),\ \dots\ ,\ \ell(Y^m, f(X^m))$$

are independent (real-valued) random variables.

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathcal{D}\sim p^{\otimes m}} \hat{\mathcal{R}}_{\mathsf{tst}}(\mathcal{D}) &= \mathop{\mathbb{E}}_{(X^1,Y^1),\dots,(X^m,Y^m)\sim p} \frac{1}{m}\sum_{i=1}^m \ell(Y^i, f(X^i)) \\
&= \frac{1}{m}\sum_{i=1}^m \mathop{\mathbb{E}}_{(X^1,Y^1),\dots,(X^m,Y^m)\sim p} \ell(Y^i, f(X^i)) \\
&= \frac{1}{m}\sum_{i=1}^m \mathop{\mathbb{E}}_{(X^i,Y^i)\sim p} \ell(Y^i, f(X^i)) \\
&= \frac{1}{m}\sum_{i=1}^m \mathop{\mathbb{E}}_{(X,Y)\sim p} \ell(Y, f(X)) \\
&= \mathop{\mathbb{E}}_{(X,Y)\sim p} \ell(Y, f(X)) = \mathcal{R}(f) \qquad \square
\end{aligned}$$

Excurse: Concentration of Measure I

Concentration of Measure Inequalities

- Z random variables, taking values $z \in \mathcal{Z} \subseteq \mathbb{R}$.
- $p(Z = z)$ probability distribution
  - $\mu = \mathbb{E}[\,Z\,]$ mean
  - $\text{Var}[z] = \mathbb{E}[\,(Z-\mu)^2\,]$ variance

Lemma (Law of Large Numbers)

Let $Z_1, Z_2, \dots,$ be i.i.d. random variables with mean $\mathbb{E}[Z] < \infty$, then

$$\frac{1}{m}\sum_{i=1}^{m} Z_i \quad \overset{m\to\infty}{\longrightarrow} \quad \mathbb{E}[Z] \qquad \textit{with probability } 1.$$

In machine learning, we have finite data, so $m \to \infty$ is less important. Concentration of measure inequalities quantify the deviation between average and expectation for finite m.

Assumption: $\mathcal{Z} \subseteq \mathbb{R}_+$, i.e. Z takes only non-negative values.

Lemma (Markov's inequality)

$$\forall a > 0 : \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}.$$

Proof. Step 1) We can write

$$\mathbb{E}[Z] = \int_{x=0}^{\infty} \mathbb{P}[Z \geq x] \, dx$$

Step 2) Since $\mathbb{P}[Z \geq x]$ is non-increasing in x, we have for any $a \geq 0$:

$$\mathbb{E}[Z] \geq \int_{x=0}^{a} \mathbb{P}[Z \geq x] \, \mathsf{dx} \geq \int_{x=0}^{a} \mathbb{P}[Z \geq a] \, \mathsf{dx} = a \, \mathbb{P}[Z \geq a]$$

Proof sketch of Step 1 inequality (ignoring questions of measurability and exchange of limit processes and writing the expression as if Z had a density $p(z)$)

$$\mathbb{P}[Z \geq x] = \int_{z=x}^{\infty} p(z) dz = \int_{z=0}^{\infty} [\![z \geq x]\!] \; p(z) \; dz$$

$$\begin{aligned}
\int_{x=0}^{\infty} \mathbb{P}[Z \geq x] \; dx &= \int_{x=0}^{\infty} \int_{z=0}^{\infty} [\![z \geq x]\!] \; p(z) dz \; dx \\
&= \int_{z=0}^{\infty} \underbrace{\int_{x=0}^{\infty} [\![z \geq x]\!] \; dx}_{=z} \; p(z) dz \\
&= \int_{z=0}^{\infty} \int_{x=0}^{z} dx \; p(z) dz \\
&= \int_{z=0}^{\infty} z \; p(z) dz \\
&= \mathbb{E}[Z]
\end{aligned}$$

Assumption: $\mathcal{Z} \subseteq \mathbb{R}_+$, i.e. Z takes only non-negative values.

Lemma (Markov's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}.$$

Example

Is it possible that more than half of the population have a salary more than twice the mean salary?

Assumption: $\mathcal{Z} \subseteq \mathbb{R}_+$, i.e. Z takes only non-negative values.

Lemma (Markov's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}.$$

Corollary

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a\,\mathbb{E}[Z]] \leq \frac{1}{a}.$$

Example

Is it possible that more than half of the population have a salary more than twice the mean salary? No, by corrolary with $a = 2$.

Assumption: $\mathcal{Z} \subseteq \mathbb{R}_+$, i.e. Z takes only non-negative values.

Lemma (Markov's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}.$$

Corollary

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a\,\mathbb{E}[Z]] \leq \frac{1}{a}.$$

Example

Is it possible that more than half of the population have a salary more than twice the mean salary? No, by corrolary with $a = 2$.

Example

Is it possible that more than 90% of the population have a salary less than one tenth of the mean?

Assumption: $\mathcal{Z} \subseteq \mathbb{R}_+$, i.e. Z takes only non-negative values.

Lemma (Markov's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a] \leq \frac{\mathbb{E}[Z]}{a}.$$

Corollary

$$\forall a \geq 0 : \quad \mathbb{P}[Z \geq a\,\mathbb{E}[Z]] \leq \frac{1}{a}.$$

Example

Is it possible that more than half of the population have a salary more than twice the mean salary? No, by corrolary with $a = 2$.

Example

Is it possible that more than 90% of the population have a salary less than one tenth of the mean? Easily: $p(\$1) = 0.99$, $p(\$100000) = 0.01$.

Lemma (Chebyshev's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[|Z - \mathbb{E}[Z]| \geq a] \leq \frac{\mathit{Var}[Z]}{a^2}$$

Proof. Apply Markov's Inequality to the random variable $(Z - \mathbb{E}[Z])^2$.

Lemma (Chebyshev's inequality)

$$\forall a \geq 0 : \quad \mathbb{P}[|Z - \mathbb{E}[Z]| \geq a] \leq \frac{\mathit{Var}[Z]}{a^2}$$

Proof. Apply Markov's Inequality to the random variable $(Z - \mathbb{E}[Z])^2$.

For any $a \geq 0$:

$$\mathbb{P}[\,|Z - \mathbb{E}[Z]| \geq a] = \mathbb{P}[(Z - \mathbb{E}[Z])^2 \geq a^2] \overset{\text{Markov}}{\leq} \frac{\mathbb{E}[\,(Z - \mathbb{E}[Z])^2\,]}{a^2} = \frac{\mathrm{Var}[Z]}{a^2}.$$

Lemma (Chebyshev's inequality)

$$\forall a \geq 0: \quad \mathbb{P}[|Z - \mathbb{E}[Z]| \geq a] \leq \frac{\mathit{Var}[Z]}{a^2}$$

Proof. Apply Markov's Inequality to the random variable $(Z - \mathbb{E}[Z])^2$.

For any $a \geq 0$:

$$\mathbb{P}[\,|Z - \mathbb{E}[Z]| \geq a] = \mathbb{P}[(Z - \mathbb{E}[Z])^2 \geq a^2] \overset{\text{Markov}}{\leq} \frac{\mathbb{E}[\,(Z - \mathbb{E}[Z])^2\,]}{a^2} = \frac{\mathrm{Var}[Z]}{a^2}.$$

Remark: Chebyshev ineq. has similar role as "σ-rules" for Gaussians:

- 68% of probability mass of a Gaussian lie within $\mu \pm \sigma$,
- 95% of probability mass of a Gaussian lie within $\mu \pm 2\sigma$,
- 99.7% of probability mass of a Gaussian lie within $\mu \pm 3\sigma$,

Chebyshev holds for arbitrary probability distributions, not just Gaussians.

Chebyshev's Inequality

Example (Soccer Match Statistics)

- $z = -1$ for loss, $z = 0$ for draw, $z = 1$ for win.
- $p(-1) = \frac{1}{10}$, $p(1) = \frac{1}{10}$, $p(0) = \frac{4}{5}$.
- $\mathbb{E}[Z] = 0$.
- $\text{Var}[Z] = \mathbb{E}[\,(Z)^2] = \frac{1}{10}(-1)^2 + \frac{4}{5}0^2 + \frac{1}{10}(1)^2 = \frac{1}{5}$

What if we pretended Z is Gaussian?

- $\mu = 0$, $\sigma = \sqrt{\frac{1}{5}} \approx 0.45$,
- we expect $\leq 5\%$ prob.mass outside of the 2σ-interval $[-0.9, 0.9]$
- but really, its 20%!

With Chebyshev:

- $\mathbb{P}[|Z| \geq 0.9] \leq \frac{1}{5}/(0.9)^2 \approx 0.247$, so bound is correct

Applying Chebyshev's Inequality

Lemma (Quantitative Version of the Law of Large Numbers)

Set $Z_1, \ldots, Z_m$ be i.i.d. random variables with $\mathbb{E}[Z_i] = \mu$ and Var$[Z_i] \leq C$. Then, for any $\delta \in (0,1)$, the following inequality holds with probability at least $1 - \delta$:

$$\left| \frac{1}{m} \sum_{i=1}^{m} Z_i - \mu \right| < \sqrt{\frac{C}{\delta m}}.$$

Equivalent formulations:

$$\Pr\left[\left| \frac{1}{m} \sum_{i=1}^{m} Z_i - \mu \right| < \sqrt{\frac{C}{\delta m}} \right] \geq 1 - \delta.$$

$$\Pr\left[\left| \frac{1}{m} \sum_{i=1}^{m} Z_i - \mu \right| \geq \sqrt{\frac{C}{\delta m}} \right] \leq \delta.$$

Applying Chebyshev's Inequality

Lemma (Quantitative Version of the Law of Large Numbers)

Set $Z_1, \ldots, Z_m$ *be i.i.d. random variables with* $\mathbb{E}[Z_i] = \mu$ *and* *Var*$[Z_i] \leq C$. *Then, for any* $\delta \in (0, 1)$,

$$\Pr\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \geq \sqrt{\frac{C}{\delta m}}\right] \leq \delta.$$

Applying Chebyshev's Inequality

Lemma (Quantitative Version of the Law of Large Numbers)

Set $Z_1, \dots, Z_m$ be i.i.d. random variables with $\mathbb{E}[Z_i] = \mu$ and Var$[Z_i] \leq C$. Then, for any $\delta \in (0,1)$,

$$\Pr\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \geq \sqrt{\frac{C}{\delta m}}\,\right] \leq \delta.$$

Proof. The Z_i are indep., so Var $\left[\frac{1}{m}\sum_{i=1}^{m} Z_i\right] = \frac{1}{m^2}\sum_{i=1}^{m}$ Var$[Z_i] \leq \frac{C}{m}$.

2) Chebyshev's inequality gives us for any $a \geq 0$:

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \geq a\right] \leq \frac{\text{Var}\left[\frac{1}{m}\sum_{i=1}^{m} Z_i\right]}{a^2} \leq \frac{C}{ma^2}.$$

Setting $\delta = \frac{C}{ma^2}$ and solving for a yields $a = \sqrt{\frac{C}{\delta m}}$.

Sanity check: How large should my test set be?

$$\boxed{\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \leq \sqrt{\frac{C}{\delta m}}\,\right] \geq 1-\delta.}$$

Setup: fixed classifier $g : \mathcal{X} \to \mathcal{Y}$, $0/1$-loss: $\ell(\bar{y}, y) = [\![\bar{y} \neq y]\!]$

- test set $\mathcal{D} = \{(x^1, y^1)\dots,(x^m, y^m)\} \overset{i.i.d.}{\sim} p(x, y)$,
- random variables $Z_i = [\![g(x^i) \neq y^i]\!] \in \{0, 1\}$,
- $\mathbb{E}[Z^i] = \mathbb{E}\{[\![g(x^i) \neq y^i]\!]\} = \mu$ (generalization error of g)
- $\mathrm{Var}[Z^i] = \mathbb{E}\{(Z^i - \mu)^2\} = \mu(1-\mu)^2 + (1-\mu)\mu^2 = \mu(1-\mu) \leq \frac{1}{4} =: C$

Setup: fixed confidence, e.g. $\delta = 0.1$, $\sqrt{\frac{C}{\delta m}} = \sqrt{\frac{0.25}{0.1m}} = \sqrt{\frac{2.5}{m}}$

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \leq \sqrt{\frac{2.5}{m}}\right] \geq 0.9$$

To be 90%-certain that the error is within ± 0.05, use $m \geq 1,000$.

Sanity check: How large should my test set be?

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \leq \sqrt{\frac{C}{\delta m}}\right] \geq 1-\delta.$$

Setup: fixed classifier $g : \mathcal{X} \to \mathcal{Y}$, 0/1-loss: $\ell(\bar{y}, y) = [\![\bar{y} \neq y]\!]$

- test set $\mathcal{D} = \{(x^1, y^1) \dots, (x^m, y^m)\} \overset{i.i.d.}{\sim} p(x, y)$,
- random variables $Z_i = [\![g(x^i) \neq y^i]\!] \in \{0, 1\}$,
- $\mathbb{E}[Z^i] = \mathbb{E}\{[\![g(x^i) \neq y^i]\!]\} = \mu$ (generalization error of g)
- $\mathrm{Var}[Z^i] = \mathbb{E}\{(Z^i-\mu)^2\} = \mu(1-\mu)^2 + (1-\mu)\mu^2 = \mu(1-\mu) \leq \frac{1}{4} =: C$

Setup: fixed confidence, e.g. $\delta = 0.1$, $\sqrt{\frac{C}{\delta m}} = \sqrt{\frac{0.25}{0.1m}} = \sqrt{\frac{2.5}{m}}$

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| \leq \sqrt{\frac{2.5}{m}}\right] \geq 0.9$$

To be 90%-certain that the error is within ± 0.05, use $m \geq 1,000$.
To be 99%-certain that the error is within ± 0.05, use $m \geq 10,000$.
To be 90%-certain that the error is within ± 0.005, use $m \geq 100,000$.

(for this case, tighter bounds are possible: later...)

Back to machine learning

Predictor Training (idealized)

```
**input** training data $\mathcal{D}_{\text{trn}}$
**input** learning procedure $A$
   $g \leftarrow A[\mathcal{D}]$    (apply $A$ with $\mathcal{D}$ as training set)
**output** resulting predictor $g : \mathcal{X} \to \mathcal{Y}$
```

Predictor Evaluation

```
**input** trained predictor $g : \mathcal{X} \to \mathcal{Y}$
**input** test data $\mathcal{D}_{\text{tst}}$
   apply $g$ to $\mathcal{D}_{\text{tst}}$ and measure performance $R_{\text{tst}}$
**output** performance estimate $R_{\text{tst}}$
```

Predictor Training (idealized)

```
input training data $\mathcal{D}_{\text{trn}}$
input learning procedure $A$
  $g \leftarrow A[\mathcal{D}]$    (apply $A$ with $\mathcal{D}$ as training set)
output resulting predictor $g : \mathcal{X} \rightarrow \mathcal{Y}$
```

Predictor Evaluation

```
input trained predictor $g : \mathcal{X} \rightarrow \mathcal{Y}$
input test data $\mathcal{D}_{\text{tst}}$
  apply $g$ to $\mathcal{D}_{\text{tst}}$ and measure performance $R_{\text{tst}}$
output performance estimate $R_{\text{tst}}$
```

Remark: In commercial applications, this is realistic:

- given some training set one builds a single system,
- one deploys it to the customers,
- the customers use it on their own data, and complain if disappointed

In research, one typically has no customer, but only a fixed amount of data to work with, so one *simulates* the above protocol.

Classifier Training and Evaluation

```
input data $\mathcal{D}$
input learning method $A$
    split $\mathcal{D} = \mathcal{D}_{\text{trn}} \,\dot\cup\, \mathcal{D}_{\text{tst}}$ disjointly
    set aside $\mathcal{D}_{\text{tst}}$ to a safe place        // do not look at it
    $g \leftarrow A[\mathcal{D}_{\text{trn}}]$                        // learn a predictor from $\mathcal{D}_{\text{trn}}$
    apply $g$ to $\mathcal{D}_{\text{tst}}$ and measure performance $R_{\text{tst}}$
output performance estimate $R_{\text{tst}}$
```

Classifier Training and Evaluation

```
input data $\mathcal{D}$
input learning method $A$
    split $\mathcal{D} = \mathcal{D}_{\mathsf{trn}} \dot\cup \mathcal{D}_{\mathsf{tst}}$ disjointly
    set aside $\mathcal{D}_{\mathsf{tst}}$ to a safe place        // do not look at it
    $g \leftarrow A[\mathcal{D}_{\mathsf{trn}}]$                              // learn a predictor from $\mathcal{D}_{\mathsf{trn}}$
    apply $g$ to $\mathcal{D}_{\mathsf{tst}}$ and measure performance $R_{\mathsf{tst}}$
output performance estimate $R_{\mathsf{tst}}$
```

Remark. $\mathcal{D}_{\mathsf{tst}}$ should be as small as possible, to keep $\mathcal{D}_{\mathsf{trn}}$ as big as possible, but large enough to be convincing.

- sometimes: 50%/50% for small datasets
- more often: 80% training data, 20% test data
- for large datasets: 90% training, 10% test data.

Classifier Training and Evaluation

```
input data $\mathcal{D}$
input learning method $A$
    split $\mathcal{D} = \mathcal{D}_{\text{trn}} \dot\cup \mathcal{D}_{\text{tst}}$ disjointly
    set aside $\mathcal{D}_{\text{tst}}$ to a safe place        // do not look at it
    $g \leftarrow A[\mathcal{D}_{\text{trn}}]$                        // learn a predictor from $\mathcal{D}_{\text{trn}}$
    apply $g$ to $\mathcal{D}_{\text{tst}}$ and measure performance $R_{\text{tst}}$
output performance estimate $R_{\text{tst}}$
```

Remark. $\mathcal{D}_{\text{tst}}$ should be as small as possible, to keep $\mathcal{D}_{\text{trn}}$ as big as possible, but large enough to be convincing.

- sometimes: 50%/50% for small datasets
- more often: 80% training data, 20% test data
- for large datasets: 90% training, 10% test data.

$\mathcal{D}_{\text{tst}}$ is "use once": it cannot be used for any decisions in building the predictor, only to evaluate it at the very end.

Classifier Training and Evaluation

```
input data $\mathcal{D}$
input learning method $A$
    split $\mathcal{D} = \mathcal{D}_{\mathsf{trn}} \dot\cup \mathcal{D}_{\mathsf{tst}}$ disjointly
    set aside $\mathcal{D}_{\mathsf{tst}}$ to a safe place        // do not look at it
    $g \leftarrow A[\mathcal{D}_{\mathsf{trn}}]$                         // learn a predictor from $\mathcal{D}_{\mathsf{trn}}$
    apply $g$ to $\mathcal{D}_{\mathsf{tst}}$ and measure performance $R_{\mathsf{tst}}$
output performance estimate $R_{\mathsf{tst}}$
```

In practice we often want more: not just train a classifier and evaluate it, but

- select the best algorithm out of multiple ones,
- select the best (hyper)parameters for a training algorithm.

We simulate the classifier evaluation step during the training procedure. This needs (at least) one additional data split:

Training and Selecting between Multiple Models

```
input data $\mathcal{D}$
input set of method $\mathcal{A} = \{A_1, \dots, A_K\}$
  split $\mathcal{D} = \mathcal{D}_{\text{trnval}} \dot\cup \mathcal{D}_{\text{tst}}$ disjointly
  set aside $\mathcal{D}_{\text{tst}}$ to a safe place (do not look at it)

  split $\mathcal{D}_{\text{trnval}} = \mathcal{D}_{\text{trn}} \dot\cup \mathcal{D}_{\text{val}}$ disjointly
  for all models $A_i \in \mathcal{A}$ do
    $g_i \leftarrow A_i[\mathcal{D}_{\text{trn}}]$
    apply $g_i$ to $\mathcal{D}_{\text{val}}$ and measure performance $E_{\text{val}}(A_i)$
  end for
  pick best performing $A_i$

  (optional) $g_i \leftarrow A_i[\mathcal{D}_{\text{trnval}}]$    // retrain best method on larger dataset
  apply $g_i$ to $\mathcal{D}_{\text{tst}}$ and measure performance $R_{\text{tst}}$
output performance estimate $R_{\text{tst}}$
```

How to split? For example $\frac{1}{3} : \frac{1}{3} : \frac{1}{3}$ or 70% : 10% : 20%.

Discussion.

- Each algorithm is trained on $\mathcal{D}_{\text{trn}}$ and evaluated on disjoint $\mathcal{D}_{\text{val}}$ ✓
- You select a predictor based on $\mathcal{R}_{\text{val}}$ (its performance on $\mathcal{D}_{\text{val}}$), only afterwards $\mathcal{D}_{\text{tst}}$ is used. ✓
- $\mathcal{D}_{\text{tst}}$ is used to evaluate the final predictor and nothing else. ✓

Discussion.

- Each algorithm is trained on $\mathcal{D}_{\text{trn}}$ and evaluated on disjoint $\mathcal{D}_{\text{val}}$ ✓
- You select a predictor based on $\mathcal{R}_{\text{val}}$ (its performance on $\mathcal{D}_{\text{val}}$), only afterwards $\mathcal{D}_{\text{tst}}$ is used. ✓
- $\mathcal{D}_{\text{tst}}$ is used to evaluate the final predictor and nothing else. ✓

Problems.

- small $\mathcal{D}_{\text{val}}$ is bad: $\mathcal{R}_{\text{val}}$ could be bad estimate of g_A's true performance, and we might pick a suboptimal method.
- large $\mathcal{D}_{\text{val}}$ is bad: $\mathcal{D}_{\text{trn}}$ is much smaller than $\mathcal{D}_{\text{trnval}}$, so the classifier learned on $\mathcal{D}_{\text{trn}}$ might be much worse than necessary.
- retraining the best model on $\mathcal{D}_{\text{trnval}}$ might overcome that, but it comes at a risk: just because a model worked well when trained on $\mathcal{D}_{\text{trn}}$, this does not mean it'll also work well when trained on $\mathcal{D}_{\text{trnval}}$.

Leave-one-out Evaluation (for a single model/algorithm)

```
input algorithm $A$
input loss function $\ell$
input data $\mathcal{D}$        (trnval part only: test part set aside earlier)
  for all $(x^i, y^i) \in \mathcal{D}$ do
    $g^{\neg i} \leftarrow A[\ \mathcal{D} \setminus \{(x^i, y^i)\}\ ]$    // $\mathcal{D}_{\text{trn}}$ is $\mathcal{D}$ with $i$-th example removed
    $r^i \leftarrow \ell(y^i, g^{\neg i}(x^i))$              // $\mathcal{D}_{\text{val}} = \{(x^i, y^i)\}$, disjoint to $\mathcal{D}_{\text{trn}}$
  end for
output $R_{\text{loo}} = \frac{1}{n}\sum_{i=1}^{n} r^i$    (average leave-one-out risk)
```

Properties.

- Each r^i is a unbiased (but high variance) estimate of the risk $\mathcal{R}(g^{\neg i})$
- $\mathcal{D} \setminus \{(x^i, y^i)\}$ is almost the same as $\mathcal{D}$, so we can hope that each $g^{\neg i}$ is almost the same as $g = A[\mathcal{D}]$.
- Therefore, R_{loo} can be expected a good estimate of $\mathcal{R}(g)$

Problem: slow, trains n times on $n-1$ examples instead of once on n

Compromise: use fixed number of small $\mathcal{D}_{\text{val}}$

K-fold Cross Validation (CV)

```
input algorithm $A$, loss function $\ell$, data $\mathcal{D}$ (trnval part)
  split $\mathcal{D} = \dot{\bigcup}_{k=1}^{K} \mathcal{D}_k$ into $K$ equal sized disjoint parts
  for $k = 1, \dots, K$ do
    $g^{\neg k} \leftarrow A[\, \mathcal{D} \setminus \mathcal{D}_k \,]$
    $r^k \leftarrow \frac{1}{|\mathcal{D}_k|} \sum_{(x,y) \in \mathcal{D}_k} \ell(y^i, g^{\neg k}(x))$
  end for
output $R_{K\text{-CV}} = \frac{1}{K} \sum_{k=1}^{n} r^k$    ($K$-fold cross-validation risk)
```

Observation.

- for $K = |\mathcal{D}|$ same as leave-one-out error.
- approximately k times increase in runtime.
- most common: $k = 10$ or $k = 5$.

Problem: training sets overlap, so the error estimates are correlated.
Exception: $K = 2$

5×2 Cross Validation (5×2-CV)

```
input algorithm $A$, loss function $\ell$, data $\mathcal{D}$ (trnval part)
  for $k = 1, \dots, 5$ do
    Split $\mathcal{D} = \mathcal{D}_1 \dot\cup \mathcal{D}_2$
    $g_1 \leftarrow A[\mathcal{D}_1]$,
    $r_1^k \leftarrow$ evaluate $g_1$ on $\mathcal{D}_2$
    $g_2 \leftarrow A[\mathcal{D}_2]$,
    $r_2^k \leftarrow$ evaluate $g_2$ on $\mathcal{D}_1$
    $r^k \leftarrow \frac{1}{2}(r_k^1 + r_k^2)$
  end for
output $\mathcal{R}_{5\times 2} = \frac{1}{5}\sum_{k=1}^{5} r^k$
```

Observation.

- 5×2-CV is really the average of 5 runs of 2-fold CV
- very easy to implement: shuffle the data and split into halfs
- within each run the training sets are disjoint and the classifiers g_1 and g_2 are independent

Problem: training sets are smaller than in 5- or 10-fold CV.

Classification with Imbalanced Classes

If classes are *imbalanced* accuracy might not tell us much:

- $p(y=-1)=0.99$, $p(y=+1)=0.01$ $\rightarrow$ "always no" is 99% correct
- there might not be a better non-constant classifier

Three "solutions":

- balancing
  - use only subset of the majority class to balance data (5:1, or 1:1)
- reweighting
  - multiple loss in optimization with class-dependent constant C_{y_i},

$$\frac{1}{|\mathcal{D}_+|}\sum_{(x_i,y_i)\in\mathcal{D}_+}^{n}\ell(\,y_i,f(x_i)\,)+\frac{1}{|\mathcal{D}_-|}\sum_{(x_i,y_i)\in\mathcal{D}_-}^{n}\ell(\,y_i,f(x_i)\,)+\Omega(f)$$

- treat as a retrieval problem instead of classification

Classifiers for Information Retrieval Tasks

Some classification tasks are really rather *retrieval* tasks, e.g.

- database lookup: is an entry x relevant $(y = 1)$ or not $(y = -1)$?

A typical property:

- prediction is performed on a fixed database
- we have access to all elements of the test set at the same time
- positives $(y = 1)$ are important, negative $(y = -1)$ are a nuisanse
- we don't need all decisions, a few correct positives is enough

For a classifier $g(x) = \operatorname{sign} f(x)$ with $f(x) : \mathcal{X} \to \mathbb{R}$ (e.g., $f(x) = \langle w, x \rangle$), we interpret $f(x)$ as its *confidence*.

To produce K positives we return the test samples of highest confidence.

Equivalently, we decide by $g_\theta(x) = \operatorname{sign}(\, f(x) - \theta \,)$, for the right θ.

Evaluating Retrieval Systems

Retrieval quality is often measure in terms of *precision* and *recall*:

Definition (Precision, Recall, F-Score)

For $\mathcal{Y} = \{\pm 1\}$, let $g : \mathcal{X} \to \mathcal{Y}$ a decision function and $\mathcal{D} = \{(x^1, y^1), \dots, (x^n, y^n)\} \subset \mathcal{X} \times \mathcal{Y}$ be a *database*.

Then we define

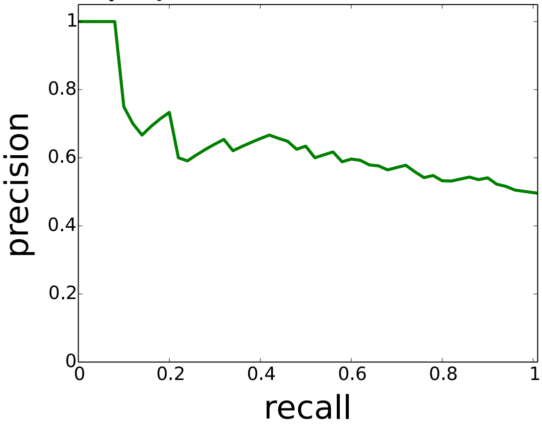
$$\textit{precision}(g) = \frac{\textit{number of test samples with } g(x^j) = 1 \textit{ and } y^j = 1}{\textit{number of test samples with } g(x^j) = 1}$$

$$\textit{recall}(g) = \frac{\textit{number of test samples with } g(x^j) = 1 \textit{ and } y^j = 1}{\textit{number of test samples with } y^j = 1}$$

$$F\textit{-score}(g) = 2\frac{\textit{precision}(g) \cdot \textit{recall}(g)}{\textit{precision}(g) + \textit{recall}(g)}$$

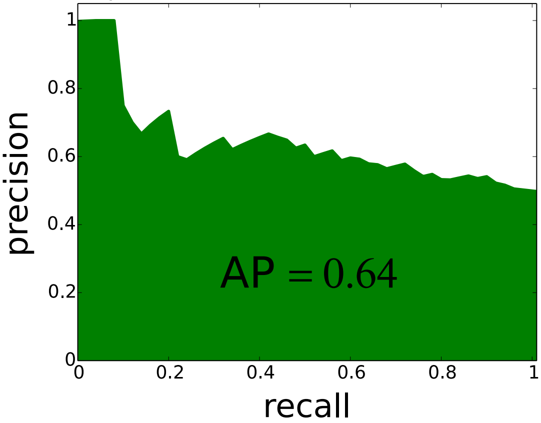
For different thresholds, θ, we obtain different precision and recall values.

They are summarized by a **precision-recall curve**:



For different thresholds, θ, we obtain different precision and recall values.

They are summarized by a **precision-recall curve**:



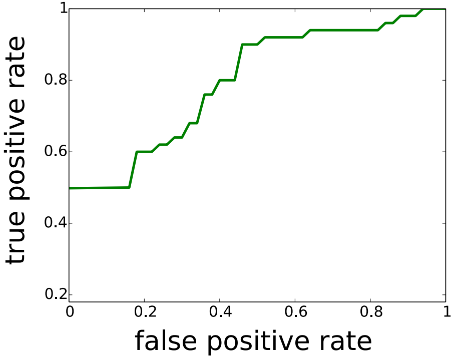
- If pressured, summarize into one number: **average precision**.
- Curve/value depends on class ratio: higher values for more positives

A similar role in different context:

Receiver Operating Characteristic (ROC) Curve

$$\textit{true-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = 1}{\textit{number of samples with } y^j = 1}$$

$$\textit{false-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = -1}{\textit{number of samples with } y^j = -1}$$

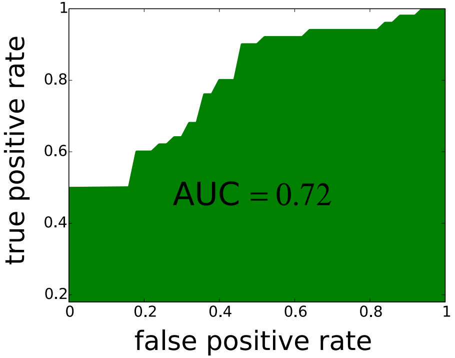


A similar role in different context:

Receiver Operating Characteristic (ROC) Curve

$$\textit{true-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = 1}{\textit{number of samples with } y^j = 1}$$

$$\textit{false-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = -1}{\textit{number of samples with } y^j = -1}$$



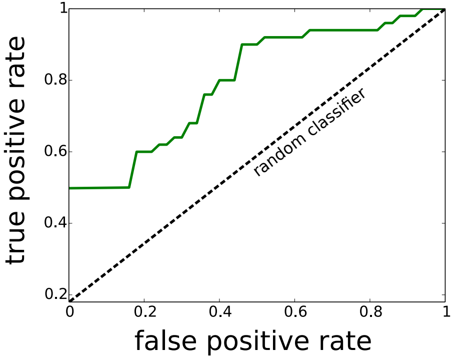
Summarize into: **area under ROC curve (AUC)**.

A similar role in different context:

Receiver Operating Characteristic (ROC) Curve

$$\textit{true-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = 1}{\textit{number of samples with } y^j = 1}$$

$$\textit{false-positive-rate}(g) = \frac{\textit{number of samples with } g(x^j) = 1 \textit{ and } y^j = -1}{\textit{number of samples with } y^j = -1}$$



Random classifier: $AUC = 0.5$, regardless of class proportions.