Machine Learning and Data Analysis
Lecture 7: Learning Theory

Filip Železný

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics
Intelligent Data Analysis lab
http://ida.felk.cvut.cz

November 8, 2011

Probability notation

P_X A probability distribution (density) on a countable (uncountable, respectively) set X.

$P_X(x)$ The value of P_X for value x.

$P_{X,Y}$ A joint probability distribution (density) on $X \times Y$.

$P_{X,Y}$(x,y) The value of $P_{X,Y}$ for values x and y.

$P_{X|Y}$ A conditional probability distribution, i.e. $P_{X|Y} = P_{X,Y}/P_Y$

$P_{X|Y}(x|y)$ The value of $P_{X|Y}$ for values x and y.

$\Pr$(expression) Probability of an event described by an expression such as $a = 1 \wedge b = 2$, then typically calculated using appropriate distributions such as $P_A(1)P_{B|A}(1|2)$.

Unsupervised learning

Assumed:

- Instance (observation) space X
  - real vectors, graphs, sequences, relational structures, ...
- Probability density P_X on X

Learner receives

- Finite *sample* ($m \in N$)

$$S = \{x_1, x_2, \ldots, x_m\}$$

drawn i.i.d. from P_X. S is a multiset, elements called *examples*.

Goals

- General: learn P_X: *density estimation task*, or
- Special: learn something about P_X: *manifold learning task*

Density estimation

Non-parametric

- No prior knowledge about P_X
- Unfeasible in general, unless P_X very simple and/or m very large

Parametric, e.g.

- Mixture of multivariate Gaussian distributions
  - $X = R^n$
  - Number of mixed Gaussians known
  - Learned parameters: means $\vec{\mu}$ and covariance matrices Σ
- Bayesian networks
  - Usually $X = \{0, 1\}^n$ (i.e., random events)
  - Independence structure (graph) known
  - Learned parameters: probabilities at vertices (CPT's)
- etc.

Manifold learning

Clustering

- Learns partitions with high P_X
- Represented explicitly (examples assigned to partitions)

Pattern learning

- Patterns define manifolds of X with unexpectedly high P_X
- Frequent itemsets, subgraphs, subsequences, ...
- *How* do patterns define manifolds?

Supervised learning

Assumed:

- Instance (observation) space X
  - real vectors, graphs, sequences, relational structures, ...
- State space Y
  - also various kinds, but usually subsets of R
- Probability density P_{XY} on $X \times Y$

Learner receives

- Finite *sample* ($m \in N$)

$$S = \{(x_1, y_1), (x_2, y_2) \ldots, (x_m, y_m)\}$$

drawn i.i.d. from P_{XY}. S is a multiset, elements called *examples*.

Goal?

Supervised learning: example

Age	Gender	Smoking	Cancer
56	Male	Yes	Yes
32	Female	No	No
48	Female	Yes	Yes
80	Female	No	Yes
80	Male	No	No
60	Male	Yes	Yes

↓

$$\textit{Smoking} \Rightarrow \textit{Cancer}$$

Supervised learning: goals

What should we learn from the sample? Consider options

- P_{XY}
  - The most general case
  - Principally same methods applicable as for learning P_X

Usually we want to learn to estimate the states y from observations x, so we really need to learn only

- $P_{Y|X}$
  - This is more special than learning P_{XY}. Why?

Estimates are single guesses, not distributions, so we need to learn only

- $f : X \rightarrow Y$ such that

$$f(x) = \arg\max_{y \in Y} P_{Y|X}(y|x)$$

  - This is more special than learning $P_{Y|X}$. Why?

Loss and risk

f will not always predict the correct state. Different errors may incur different losses. We capture this through a *loss function*

$$L : Y \times Y \rightarrow R$$

$L(y, y')$ is the amount of loss due to guessing y' when the actual state is y.

Example: $Y = \{\text{healthy}, \text{ill}\}$, $L(\text{healthy}, \text{ill}) = 1$, $L(\text{ill}, \text{healthy}) = 10$, $L(\text{healthy}, \text{healthy}) = 0$, $L(\text{ill}, \text{ill}) = 0$.

The mean (=*expected*) value of loss (w.r.t. P_{XY}) incurred by f is called the *risk* of f

$$R(f) = \sum_{x \in X} \sum_{y \in Y} L\left(y, f(x)\right) P_{XY}(x, y)$$

Replace $\sum$ by $\int$ for uncountable X and/or Y (also in what follows).

Risk minimization

Given a loss function, we want to learn a function f minimizing the risk

$$f = \arg\min_{f:X\to Y} R(f)$$

Pointwise solution (yields values for individual x)

$$f(x) = \arg\min_{y'\in Y} \sum_{y\in Y} L\left(y, y'\right) P_{Y|X}(y|x)$$

That does not help the learner who does not know $P_{Y|X}$. If he did, he would not have to learn!

Only shows the best thing he possibly could learn.

Classification

When Y is finite, the task is called *classification learning* (or simply *classification*). Elements of Y are called *classes* and f is a *classifier*.

If all errors have equal importance, the usual loss function is

$$L_{01}(y, y') = \begin{cases} 0 \text{ if } y = y' \\ 1 \text{ if } y \neq y' \end{cases}$$

with L_{01}, risk R becomes *classification error* e

$$e(f) = \sum_{x \in X} \sum_{y \in Y} L_{01}\left(y, f(x)\right) P_{XY}(x, y) = 1 - \underbrace{\sum_{x \in X} P_{XY}\left(x, f(x)\right)}_{\text{Prob. of correct classification}}$$

and the pointwise solution of $\arg\min_{f:X \to Y} e(f)$ corresponds to maximizing the posterior probability

$$f(x) = \arg\min_{y \in Y} \sum_{y \in Y} L_{01}\left(y, f(x)\right) P_{Y|X}(y|x) = \arg\max_{y \in Y} P_{Y|X}(y|x)$$

Regression

When $Y = R$, the task is called *regression learning* (or simply *regression*). Then usually $X = R^n$, and function f is called a *regressor*.

The usual choice of the loss function is then

$$L_{SQ}(y, y') = (y - y')^2$$

with L_{SQ}, risk R becomes *mean squared error* MSE

$$MSE(f) = \int_{x \in X} \int_{y \in Y} L_{SQ}\left(y, f(x)\right) \mathrm{d}P_{XY}(x, y)$$

and the pointwise solution of $\arg\min_{f:X \to Y} MSE(f)$ corresponds to yielding conditional expectations

$$f(x) = \int_{y \in Y} y \, \mathrm{d}P_{Y|X}(y|x) = \mathbf{E}_{Y|X}(x)$$

We won't be concerned too much with regression.

Concept learning

If

- $Y = \{0, 1\}$
- $P_{Y|X}(y|x) \in \{0, 1\}$ for all $x \in X, y \in Y$

then $P_{Y|X}$ represents a function $c : X \rightarrow \{0, 1\}$

$$c(x) = 1 \text{ iff } P_{Y|X}(1|x) = 1$$

and is called a *concept*. In *concept learning* we adopt loss function L_{01}.

A learned *classifier* f is then also called a *hypothesis*. Its classification error

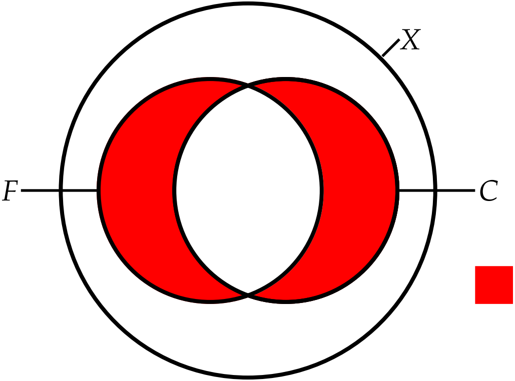
$$e(f) = \sum_{c(x) \neq f(x)} P_X(x)$$

Examples $(x, 1)$ are called *positive*, examples $(x, 0)$ are *negative*.

Concept learning: a set view

A concept c may be viewed as a subset $C = \{x \in X \mid c(x) = 1\}$.

Similarly for a hypothesis f, $F = \{x \in X \mid f(x) = 1\}$.



🟥 ...error ($f(x) \neq c(x)$)

Learnability considerations

Under what conditions can we learn successfully? What does it mean to learn successfully?

We need a few preliminary thoughts to define this.

Can we learn from any samples, as long as they are large enough ($m \to \infty$)?

Assumption of an **i.i.d. sample** is crucial. Otherwise, the sample could contain just m repetitions of a single example! Even if this is drawn according to P_{XY}, generalization would not be possible.

Without the i.i.d. assumption, learning would be possible if $m \to \infty$, examples do not repeat, P_{XY} is a concept and X is finite. (Think this through.) But that would be a very strong restriction!

Learnability considerations (cont'd)

If the i.i.d. sample is large enough, can we always learn a perfect (zero-risk) function?

Unless P_{XY} is a concept, even the best possible function $f = \arg\min_{f:X\rightarrow Y} R(f)$ will have a **non-zero risk**. In classification, the best possible f is called the *Bayes* classifier.

What if the i.i.d. sample is large enough and P_{XY} is a concept?

Not even concepts can be always learned given arbitrarily large i.i.d. samples. Any learned concept has to have a finite *representation* (rule, decision tree, functional form, program...), so there is a countable number of learnable concepts. But if X is infinite, there is an uncountable number $|2^X| = |R|$ of concepts.

Learnability Model

A learnability model defines what it means to learn successfuly. We first consider concept learning (simpler than general classification learning).

- We cannot require that *any* concept $c \in 2^X$ can be learned (follows from previous slide)
  - Rather, concepts will be chosen from some smaller *concept class* $\mathcal{C} \subset 2^X$.
  - Similarly, learner will search a hypothesis f in some *hypothesis class* $\mathcal{F} \subset 2^X$
  - it may or may not be that $\mathcal{C} = \mathcal{F}$
- We do not require zero error of learned hypotheses f
  - Rather bound the error with a small constant ϵ.
- We do not require that learning never fails ($=$ always produces a f with acceptable error)
  - Rather bound the probability of failure with a small constant δ.
- We ask how many examples are needed for given ϵ and δ

PAC Learnability

The PAC Learnability Model

An algorithm *PAC-learns* a concept class $\mathcal{C} \subset 2^X$ by a hypothesis class $\mathcal{F} \subset 2^X$ if for any $c \in \mathcal{C}$, P_X, $\epsilon > 0$, $\delta > 0$, using a **polynomial** number m of examples $(x_i, c(x_i))$ where x_i are drawn i.i.d. from P_X, outputs a hypothesis $f \in \mathcal{F}$ such that $e(f) \leq \epsilon$ with probability at least $1 - \delta$.

- 'PAC': Probably Approximately Correct
- 'polynomial': in $1/\epsilon$ and $1/\delta$, and the *size* of examples
  - size must be defined appropriately
  - if x are tuples (such as vectors), then usually the number of their components ("features")
- Instead of P_{XY}, the definiton considers P_X. Note that $P_{XY} = P_{Y|X} \cdot P_X$ and $P_{Y|X}$ is represented by c.
- Remind that $e(f) = \sum_{c(x) \neq f(x)} P_X(x)$

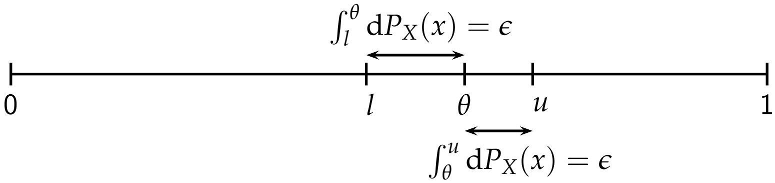
PAC Learning: example

$X = [0;1]$, $Y = \{0,1\}$, $\mathcal{C} = \{c_\theta : X \to Y \mid \theta \in [0;1]\}$, $\mathcal{F} = \mathcal{C}$

$$c_\theta(x) = \begin{cases} 0 \text{ if } x \leq \theta \\ 1 \text{ if } x > \theta \end{cases} \qquad f_{\hat{\theta}} = \begin{cases} 0 \text{ if } x \leq \hat{\theta} \\ 1 \text{ if } x > \hat{\theta} \end{cases}$$

Given sample S, learner sets $\hat{\theta}$ as the average of $\min\{x \mid (x,1) \in S\}$ and $\max\{x \mid (x,0) \in S\}$

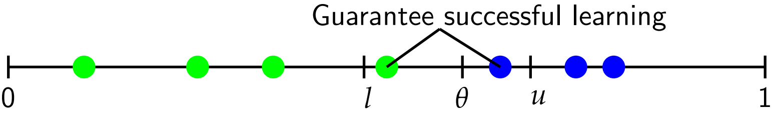
Define l and u as



$\int_l^\theta \mathrm{d}P_X(x) = \epsilon$
0
l
θ
u
1
$\int_\theta^u \mathrm{d}P_X(x) = \epsilon$

PAC Learning: example (cont'd)

If a sample point falls into $[l;\theta]$ and another into $[\theta;u]$, then learner succeeds, i.e. produces f with $e(f) \leq \epsilon$.



Given sample $S = \{(x_1, y_1), \ldots (x_m, y_m)\}$, what is the probability P of that *not* happening?

$$\begin{aligned} P &= \Pr\{(\forall i : x_i \notin [l,\theta]) \vee (\forall i : x_i \notin [\theta,u])\} \\ &\leq \Pr(\forall i : x_i \notin [l,\theta]) + \Pr(\forall i : x_i \notin [\theta,u]) \end{aligned}$$

$$\Pr(\forall i : x_i \notin [l,\theta]) = \Pi_1^m \Pr(x_i \notin [l,\theta]) = \Pi_1^m (1-\epsilon) = (1-\epsilon)^m$$

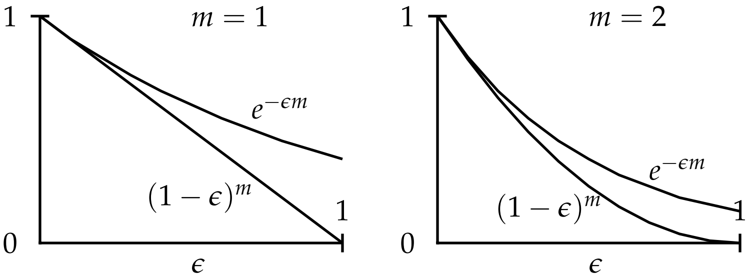
Similarly

$$\Pr(\forall i : x_i \notin [\theta,u]) = (1-\epsilon)^m$$

Exponential bound

For all $\epsilon \in [0,1]$, $m \geq 1$

$$(1-\epsilon)^m \leq e^{-\epsilon m}$$



PAC Learning: example (cont'd)

$$P \leq 2(1-\epsilon)^m \leq 2e^{-\epsilon m}$$

For PAC-learning, we must make sure that $P \leq \delta$. This holds if

$$2e^{-\epsilon m} \leq \delta$$

i.e. if the number of examples is

$$m \geq \frac{1}{\epsilon} \ln\left(\frac{2}{\delta}\right)$$

The algorithm PAC-learns $\mathcal{C}_\theta$ by $\mathcal{F}_\theta$ because it only needs a number of examples polynomial in $1/\delta$ and $1/\epsilon$. Note that m does not depend here on the *size* of examples, i.e. the number of digits used to (approximately) represent real numbers.

Empirical risk and training error

Given sample $S = \{(x_1, y_1) \ldots (x_m, y_m)\}$, the *empirical risk* of f (on S) is

$$R(S,f) = \frac{1}{m}\sum_{i=1}^{m} L\left(y_i, f(x_i)\right)$$

With L_{01} loss function, the empirical risk equals the empirical classification error

$$e(S,f) = \frac{1}{m}\sum_{i=1}^{m} L_{01}\left(y_i, f(x_i)\right)$$

which is the proportion of misclassified examples from S.

$e(S,f)$ is also called the *training error* of f (on S).

We say that f is *consistent* with S is $e(S,f) = 0$

Consistent PAC-learning

A *consistent* learner:

- on receiving any i.i.d. sample S of $c \in \mathcal{C}$
- returns a hypothesis $f \in \mathcal{F}$ consistent with S

How many examples does this learner need to PAC-learn $\mathcal{C}$ by $\mathcal{F}$?

Given an ϵ and $S = \{(x_1, y_1), \ldots (x_m, y_m)\}$, a hypothesis f such that $e(f) > \epsilon$ is called *bad*. Denote $\mathcal{F}_B = \{f \in \mathcal{F} \mid f \text{ is bad}\}$.

Probability that a bad hypothesis $f \in \mathcal{F}_B$ is consistent

$$\begin{aligned}\Pr(f \text{ is consistent}) &= \Pr\left(\wedge_{i=1}^{m} \{f(x_i) = c(x_i)\}\right) \\ &= \Pi_{i=1}^{m} \{1 - e(f)\} \\ &= \{1 - e(f)\}^m \\ &< (1 - \epsilon)^m \text{ (because } f \text{ is bad)}\end{aligned}$$

Consistent PAC-learning (cont'd)

Probability that $\mathcal{F}_B$ contains a consistent hypothesis

$$\begin{aligned}
\Pr(\text{some } f \in \mathcal{F}_B \text{ is consistent}) &\leq \sum_{f \in \mathcal{F}_B} \Pr(f \text{ is consistent}) \\
&< \sum_{f \in \mathcal{F}_B} (1-\epsilon)^m \\
&= |\mathcal{F}_B|(1-\epsilon)^m \\
&\leq |\mathcal{F}|(1-\epsilon)^m
\end{aligned}$$

If $\mathcal{F}_B$ does not contain any consistent hypothesis, then any consistent hypothesis is good ($e(f) \leq \epsilon$), i.e. the learner succeeds. We PAC-learn if this happens with probability at least $1-\delta$ so we set $|\mathcal{F}|(1-\epsilon)^m \leq \delta$.

Consistent PAC-learning (cont'd)

$$|\mathcal{F}|(1-\epsilon)^m \leq |\mathcal{F}|e^{-\epsilon m} \leq \delta$$

solving for m:

$$m \geq \frac{1}{\epsilon}\left(\ln|\mathcal{F}| + \ln\frac{1}{\delta}\right)$$

Therefore any consistent learner PAC-learns $\mathcal{C}$ by $\mathcal{F}$ unless $|\mathcal{F}|$ grows super-exponentially with the size of examples.

Remarks:

- $|\mathcal{F}|$ must be finite for the bound above to make sense!
- If $\mathcal{C} \not\subseteq \mathcal{F}$ then the learner cannot be consistent.

Consistent PAC-learning: Example

$\mathcal{F} = \mathcal{C} =$ monotone conjunctions $\equiv \mathcal{C}^{MC}$

- n propositional variables $p_1, p_2, \ldots, p_n$
- Monotone conjunction: a conjunction of some of the variables, negation not allowed. E.g. $p_1 \wedge p_4 \wedge p_5$
- $x =$ truth values of all $p's$, size of examples $= n$

Consistent algorithm: outputs a conjunction of all variables true in all positive examples. Define $f(x) = 1$ iff the conjunction is true.

$$\ln|\mathcal{F}| = \ln 2^n = n \ln 2$$

$$m \geq \frac{1}{\epsilon}\left(n \ln 2 + \ln\left(\frac{1}{\delta}\right)\right)$$

$\Rightarrow$ The algorithm PAC-learns $\mathcal{C}^{MC}$ by $\mathcal{C}^{MC}$.

Consistent PAC-learning: Example 2

$\mathcal{F} = \mathcal{C} =$ formulas in disjunctive normal form $\equiv \mathcal{C}^{DNF}$

- n propositional variables $p_1, p_2, \ldots, p_n$
- A DNF formula: a disjunction of conjunctions, only variables may be negated. E.g. $(p_1 \wedge \neg p_4) \vee (p_2 \wedge p_3 \wedge p_4)$
- $x =$ truth values of all $p's$, size of examples $= n$

How large is $|\mathcal{F}|$?

- 3^n possible conjunctions
- $|\mathcal{F}| = 2^{3^n}$ possible disjunctions of these conjunctions

$|\mathcal{F}|$ is super-exponential, $\ln|\mathcal{F}| = 3^n \ln 2$ is not polynomial. We cannot prove PAC-learnability (but neither refute).

PAC-Learnability of $\mathcal{C}^{DNF}$ is an open problem.

Bounds for classification learning

We now assume the general case of classification learning where $P_{Y|X}$ may not be a concept, i.e. may acquire vaues other than 0 or 1. We however continue assuming the L_{01} loss function.

Now we can get ‘contradictory’ examples such as $(x,0)$ and $(x,1)$ in the learning sample. Consistent learning is thus not generally possible.

Even if $P_{Y|X}$ happens to be a concept, consistent learning may not be possible if $\mathcal{C} \nsubseteq \mathcal{F}$. Then $\mathcal{F}$ may not contain a hypothesis consistent with the learning sample. The following analysis covers this situation.

Under such conditions we assume the learner selects the classifier with the least training error on the training sample S

$$\arg\min_{f \in \mathcal{F}} e(S,f)$$

Known as the ERM (empirical risk minimization) principle of learning.

Questions of interest

Relationship between classification and training errors of the learned classifier $f = \arg\min_{f \in \mathcal{F}} e(S, f)$:

- What is the difference between $e(f)$ and $e(S, f)$?
- How many examples are needed to make it small with high probability?

A generalized version of PAC-learning:

- Let f^* be the 'best classifier' in $\mathcal{F}$, i.e. $f^* = \arg\min_{f \in \mathcal{F}} e(f)$. How many examples are needed to make the difference $|e(f) - e(f^*)|$ small with high probability?

Note: Since we use the L_{01} loss, $f^*(x) = \arg\max_y P_{Y|X}(y|x)$ if this function is in $\mathcal{F}$.

Hoeffding inequality

Let P_Z be distribution on $\{0,1\}$ and $\{z_1, z_2, \dots z_m\}$ be an i.i.d. sample from P_Z. Then for the difference between the true and sample means it holds

$$\Pr\left(\left|P_Z(1) - \frac{1}{m}\sum_{i=1}^{m} z_i\right| > \epsilon\right) \leq 2e^{-2\epsilon^2 m}$$

Given a training sample $S = \{(x_1, y_1), \dots (x_m, y_m)\}$ and a classifier f, $e(f)$ and $e(S,f)$ are respectively the true and sample means of the random variable acquiring value 1 if an example is misclassified by f, and 0 otherwise (due to the L_{01} loss). Therefore:

$$\Pr\left(|e(f) - e(S,f)| > \epsilon\right) \leq 2e^{-2\epsilon^2 m}$$

Classification vs. training error

If $|e(f) - e(S,f)| \le \epsilon$ for all $f \in \mathcal{F}$ then it will be so also for the learned f. So we need to bound the probability that this is not true for some $f \in \mathcal{F}$.

$$
\begin{aligned}
&\Pr\left(\exists f \in \mathcal{F} : |e(f) - e(S,f)| > \epsilon\right)\\
\le &\sum_{f \in \mathcal{F}} \Pr\left(|e(f) - e(S,f)| > \epsilon\right)\\
= &|\mathcal{F}| \Pr\left(|e(f) - e(S,f)| > \epsilon\right)\\
\le &2|\mathcal{F}| e^{-2\epsilon^2 m}
\end{aligned}
$$

Classification vs. training error (cont'd)

If we set this probability to δ

$$2|\mathcal{F}|e^{-2\epsilon^2 m} = \delta$$

we get after arrangements

$$\epsilon = \sqrt{\frac{1}{2m} \ln \frac{2|\mathcal{F}|}{\delta}}$$

That is, with probability at least $1 - \delta$, the difference between training and classification error will be bounded as

$$|e(f) - e(S,f)| \leq \sqrt{\frac{1}{2m} \ln \frac{2|\mathcal{F}|}{\delta}}$$

for all $f \in \mathcal{F}$, including the learned classifier.

Classification vs. training error (cont'd)

If we instead solve

$$2|\mathcal{F}|e^{-2\epsilon^2 m} = \delta$$

for m, we get

$$m = \frac{1}{2\epsilon^2} \ln \frac{2|\mathcal{F}|}{\delta}$$

That is, this number of examples is enough to make $|e(f) - e(S,f)| \leq \epsilon$ with probability at least $1 - \delta$ for all $f \in \mathcal{F}$ including the learned classifier.

Deviation from the best possible classifier

Assume we have enough examples to make $|e(f) - e(S,f)| \leq \epsilon$ for all $f \in \mathcal{F}$ with probability at least $1-\delta$ (c.f. prev. slide).

Then with probability at least $1-\delta$

$$\begin{aligned} e(f) &\leq e(S,f) + \epsilon && \text{(by the assumption)} \\ &\leq e(S,f^*) + \epsilon && \text{(since } f \text{ minimizes } e(S,f) \text{ over all } f \in \mathcal{F}\text{)} \\ &\leq e(f^*) + 2\epsilon && \text{(since } e(S,f^*) \leq e(f^*) + \epsilon \text{ by the assumption)} \end{aligned}$$

Combining with the earlier derived formula for ϵ, we get that with probability at least $1-\delta$

$$e(f) \leq \left(\min_{f \in \mathcal{F}} e(f) \right) + 2\sqrt{\frac{1}{2m} \ln \frac{2|\mathcal{F}|}{\delta}}$$

The bias-variance trade-off

The smaller $\mathcal{F}$, the more *bias* and less *variance (flexibility)*.

With less bias (larger $\mathcal{F}$):

$$e(f) \leq \underbrace{\left(\min_{f \in \mathcal{F}} e(f) \right)}_{\text{may decrease}} + \underbrace{2\sqrt{\frac{1}{2m} \ln \frac{2|\mathcal{F}|}{\delta}}}_{\text{will increase}}$$

Too much bias: *underfitting*

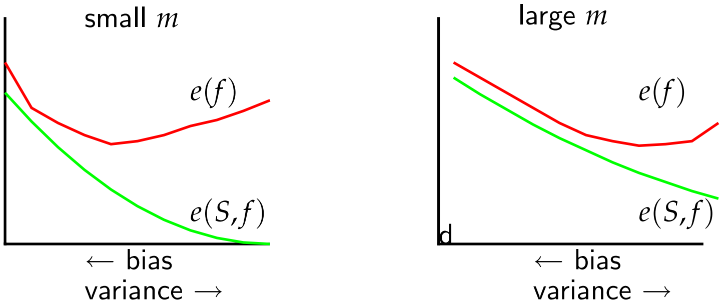
- cannot adapt well to data

Not enough bias: *overfitting*

- classification error too different from training error

The bias-variance trade-off

Typical behavior



To minimize $e(f)$ we must find a good trade-off between bias and variance.