CSC 480/580 Principles of Machine Learning

02 Limits of Learning

Jason Pacheco

Department of Computer Science

Motivation

- Supervised learning is a general & useful framework

- Understand when supervised learning will and will not work

Bayes optimal classifier and its error

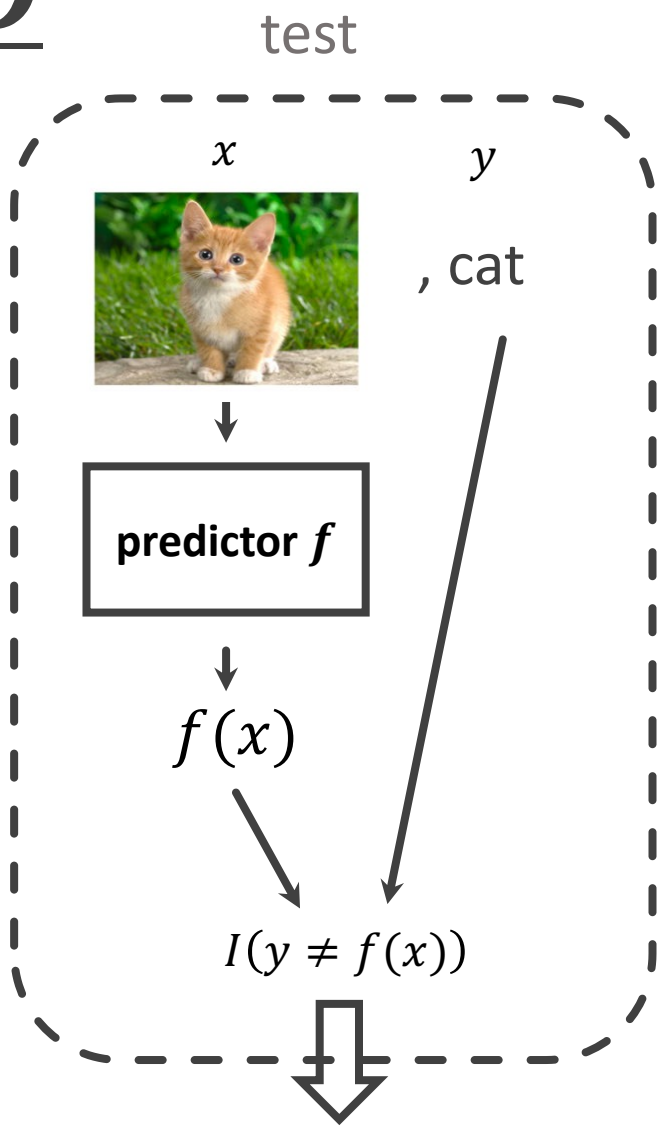
Optimal classification with known D

- Suppose: $I(A) = 1$ *if A happens, and = 0 otherwise*
  - Binary classification, 0-1 loss $\ell(y, \hat{y}) = I(y \neq \hat{y})$
  - D is *known:* for every (x, y), $P_D(x, y)$ is known to us

- What is the f that has the smallest *generalization error*

  $L_D(f) = \mathrm{E}_{(x,y)\sim D} I(y \neq f(x))$?

- Note also: $L_D(f) = P_{(x,y)\sim D}\,(y \neq f(x))$



Generalization error: $L_D(f) = \mathrm{E}_{(x,y)\sim D} I(y \neq f(x))$

Simple case: discrete domain $\mathcal{X}$

- Predicting whether the student will pass the course (y), given her project grade (x)

$P_D(x,y)$	$x=1$	$x=2$	$x=3$
$y=-1$	0.2	0.2	0.15
$y=+1$	0.1	0.3	0.05

Which classifier is better?

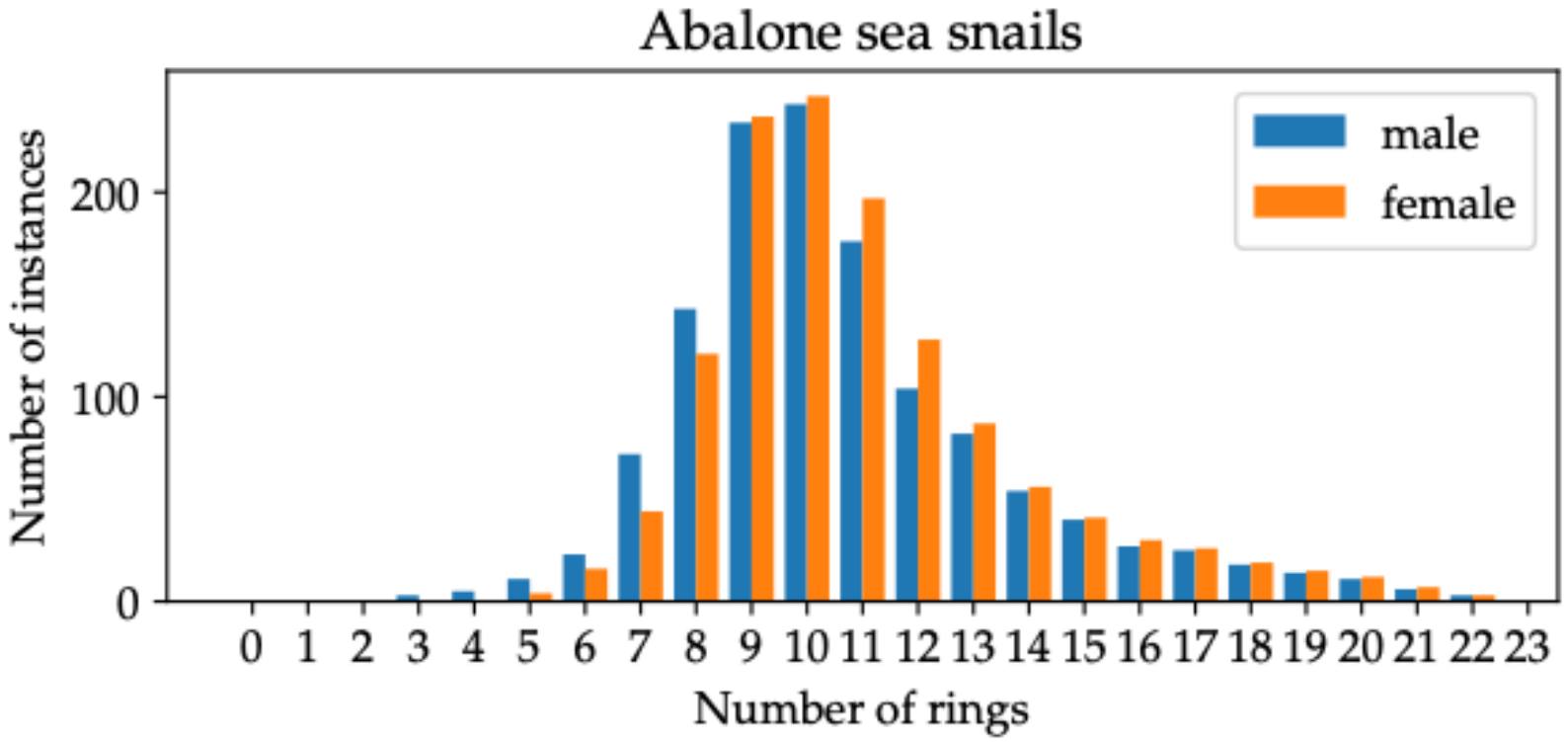
- $f_1(1)=-1, f_1(2)=-1, f_1(3)=-1 \quad \Rightarrow \quad L_D(f_1)=0.1+0.3+0.05$
- $f_2(1)=-1, f_2(2)=+1, f_2(3)=-1 \quad \Rightarrow \quad L_D(f_2)=0.1+0.2+0.05$

Is this the best classifier? Why?

- For any x, should predict y that has higher value of $P_D(x,y)$
- Intuition: predict the label that **better correlates** with the feature x
- $f^*(1)=-1, f^*(2)=+1, f^*(3)=-1$

Example: telling apart male vs female abalones

- If we see an abalone with 7 rings, should I guess it to be male or female? male
- What about 12 rings? female



Bayes optimal (BO) classifier

Fact f_{BO} achieves the smallest generalization error among all classifiers.

$$f_{BO}(x) = \arg\max_{y \in \mathcal{Y}} P_D(X = x, Y = y) = \arg\max_{y \in \mathcal{Y}} P_D(Y = y \,|X = x)\,, \forall x \in \mathcal{X}$$

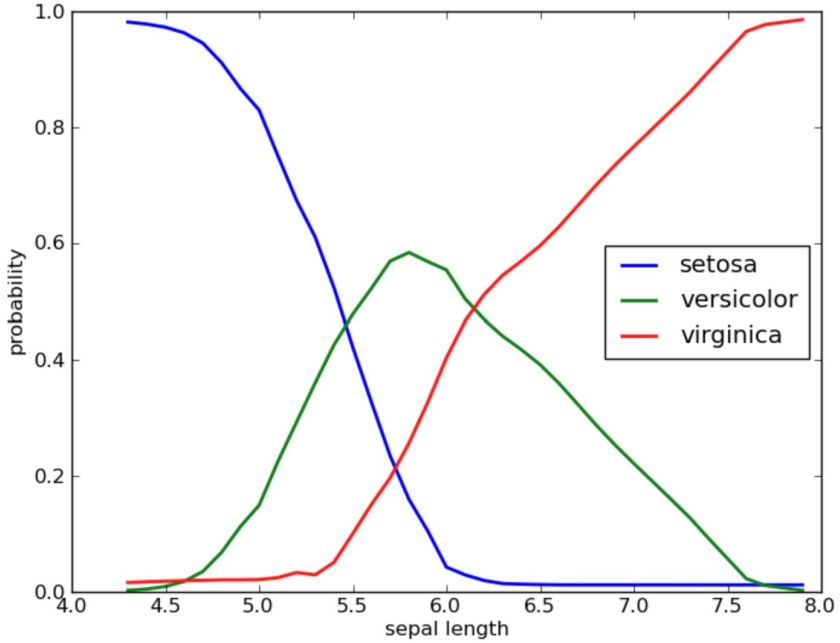
Example Iris dataset classification:

Iris Setosa

Iris Versicolor

Iris Virginica

$P_D(Y = y \,|X = x)$

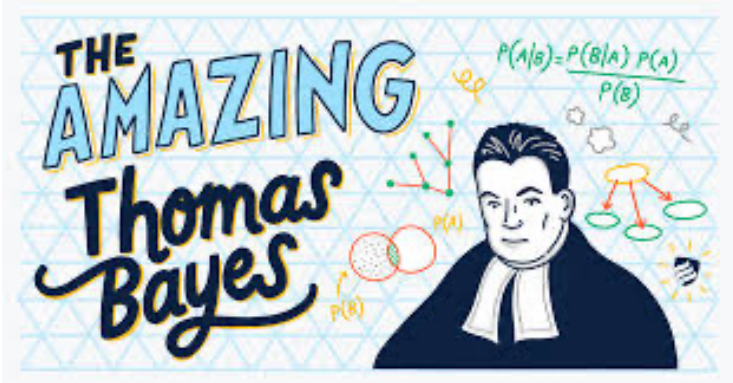


Where does “Bayes” come from?

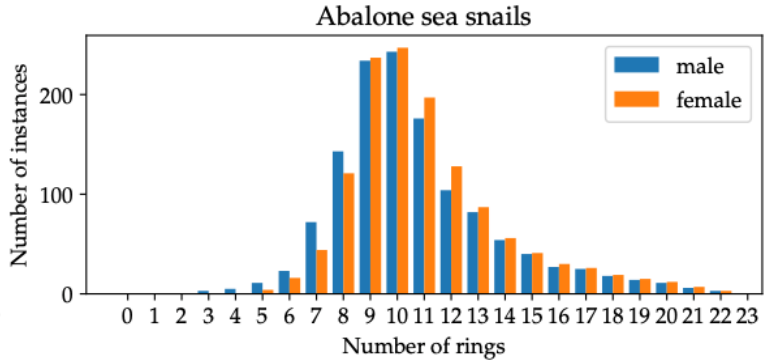
Bayes rule For events A, B,

$$P(A \mid B) = \frac{P(A) \cdot P(B \mid A)}{P(B)}$$

- Very easy to derive from the chain rule, so remember that first.
- Named after Thomas Bayes (1701-1761), English philosopher & pastor



Justification of the Fact



Step 1 consider accuracy,

- $A_D(f) = 1 - L_D(f) = P_D\big(Y = f(X)\big) = \sum_x P_D\big(X = x, Y = f(x)\big)$
- Suffices to show f_{BO} has the highest accuracy

Step 2 comparison,

$$A_D(f_{BO}) - A_D(f) = \sum_x P_D\big(X = x, Y = f_{BO}(x)\big) - P_D\big(X = x, Y = f(x)\big) \geq 0$$

$$f_{BO}(x) = \arg\max_{y \in \mathcal{Y}} P_D(X = x, Y = y)$$

Remarks

- Similar reasoning can be used to show the fact with continuous domain $\mathcal{X}$ (sum -> integral)

Bayes error rate: alternative form

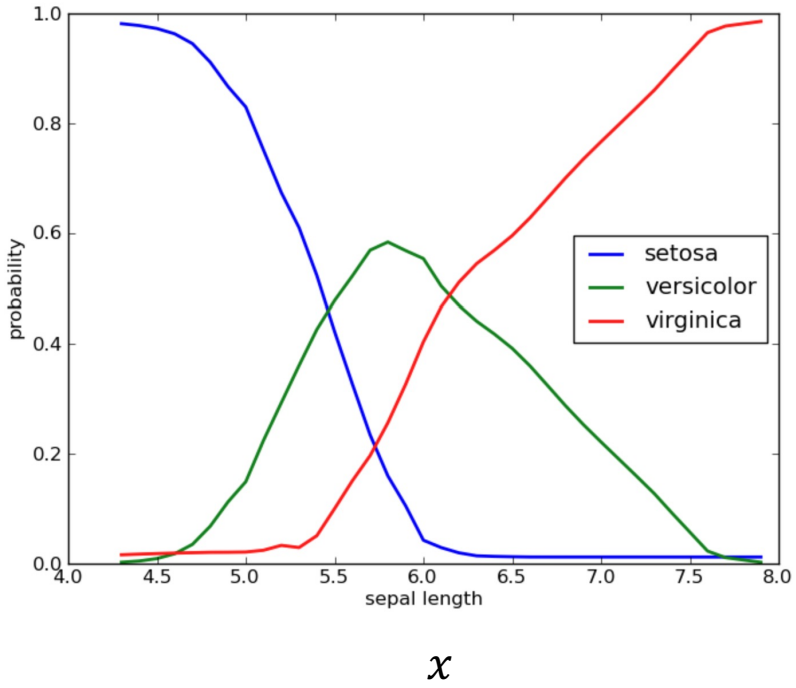
Fact

$$L_D(f_{BO}) = \mathrm{E}\left[1 - \max_y P_D(Y = y \mid X)\right]$$

$\max_y P_D(Y = y \mid X)$: upper envelope of the three colored curves

$L_D(f_{BO}) \approx$ area above the upper envelope

$P_D(Y = y \mid X = x)$



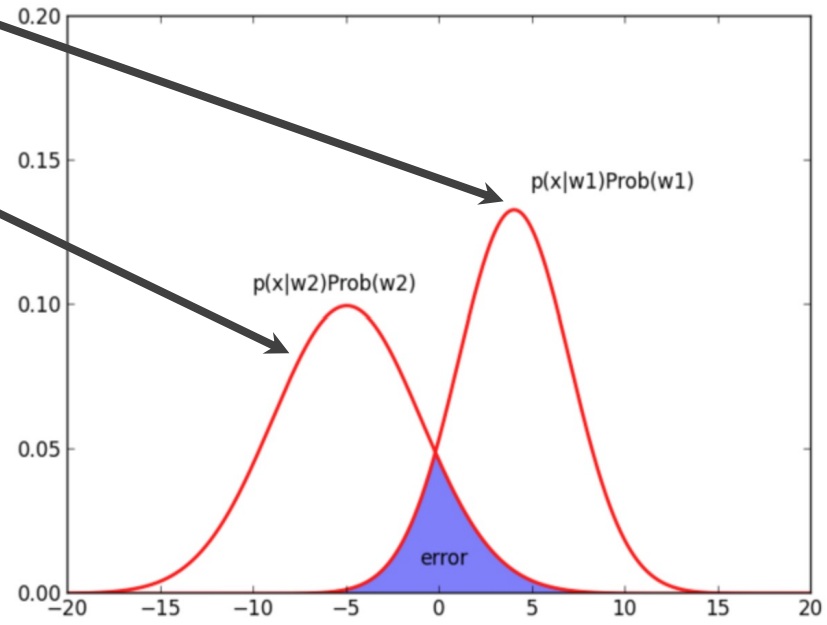
x

Bayes error rate: binary classification case

Fact when we have only two labels $-1, +1$,

$$L_D(f_{BD}) = \sum_x \min(\, P_D(Y = +1, X = x)\,, P_D(Y = -1, X = x))$$

$P_D(x, y)$	$x = 1$	$x = 2$	$x = 3$
$y = -1$	0.2	0.2	0.15
$y = +1$	0.1	0.3	0.05



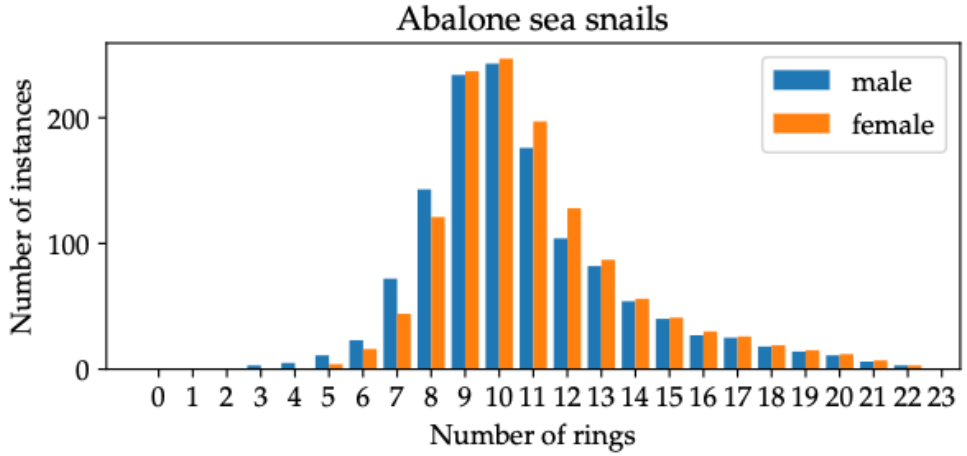
- Note: the Bayes error rate is a property of data distribution D
- Q: for a distribution D, when is its Bayes error rate zero?

Bayes error rate: more explicit formula

- When our prediction task has only binary labels:

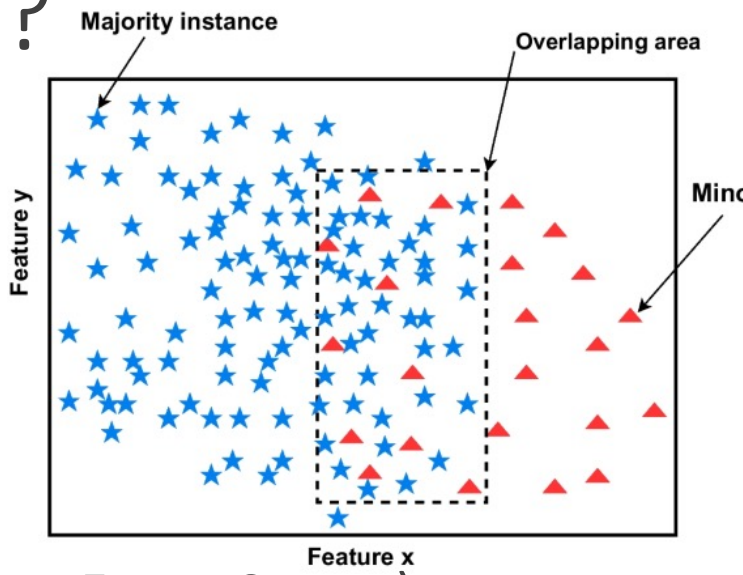
$$L_D(f_{BD}) = \sum_x \min(\, P_D(Y = +1, X = x)\,, P_D(Y = -1, X = x))$$

- When will the Bayes error rate be zero?



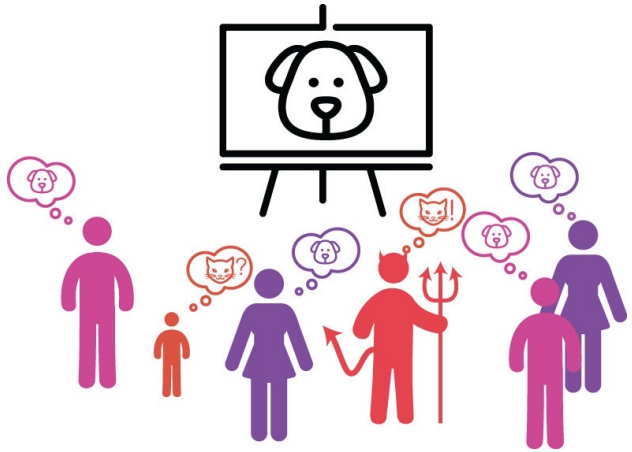
- For every x, one of $P_D(Y = +1, X = x)$, $P_D(Y = -1, X = x)$ must be zero
- That is, Y is *deterministic* given X

When is the Bayes error rate nonzero?



- $L_D(f_{BO}) \neq 0$ if $y \mid x$ is not deterministic for some x
- In other words, the two classes overlap in features

- $L_D(f_{BO}) \neq 0$ when we have:
- Limited feature representation (e.g. predicting Infection Status using Fever Status)
- Noise in the data
  - Feature noise – e.g. Sensor failure, Typo in reviews for sentiment classification
  - Label noise – e.g. typo transcribing reviews
- Labelers not having consensus about the label of an example

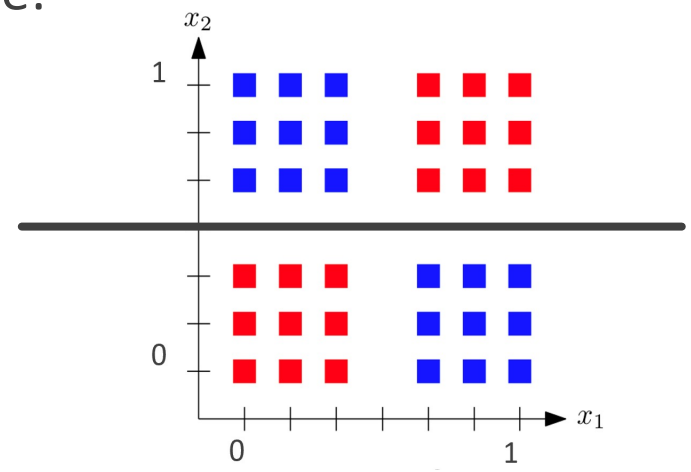


Overfitting and underfitting; model evaluation

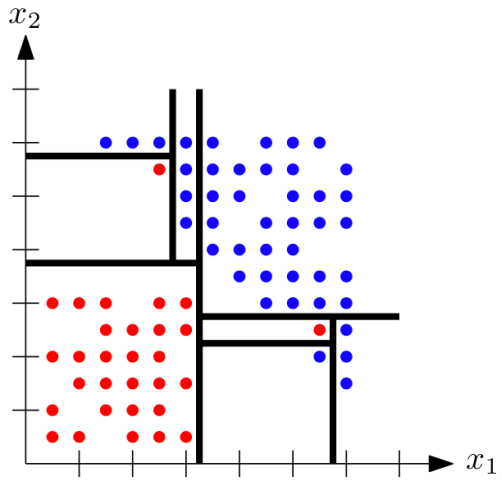
Overfitting vs Underfitting

- Q: should I learn a shallow or deep decision tree?

- Shallow tree:

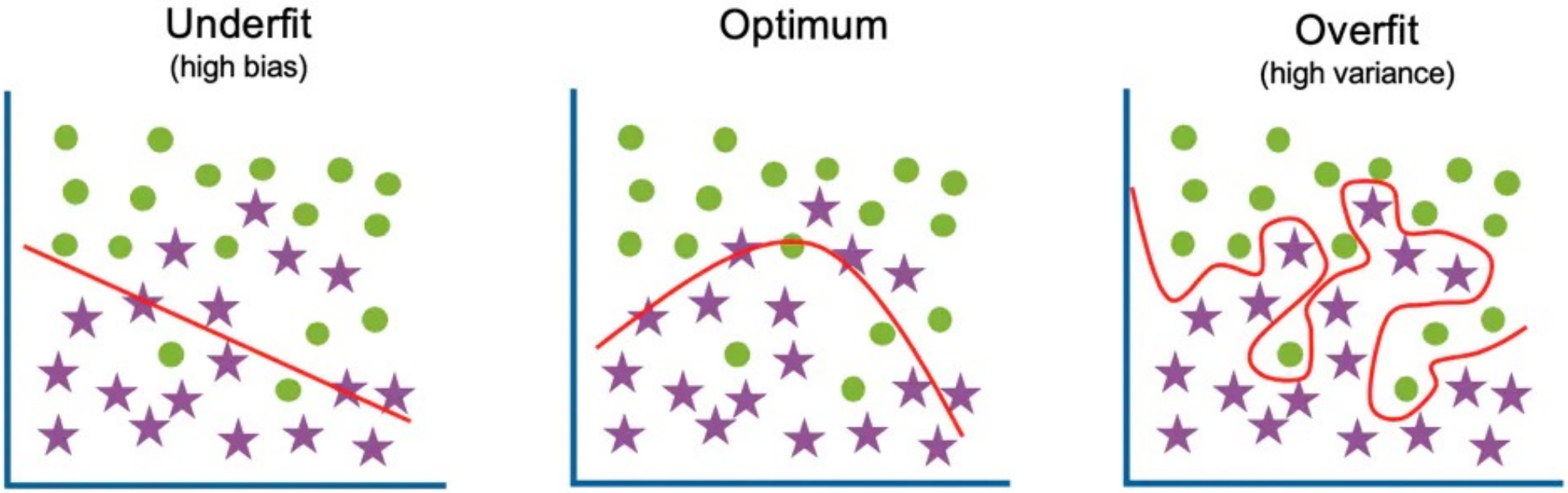


Deep tree:



- Underfitting: have the opportunity to learn something but didn't
- Overfitting: pay too much attention to idiosyncrasies to training data, and do not generalize well
- A model that neither overfits nor underfits is expected to do best

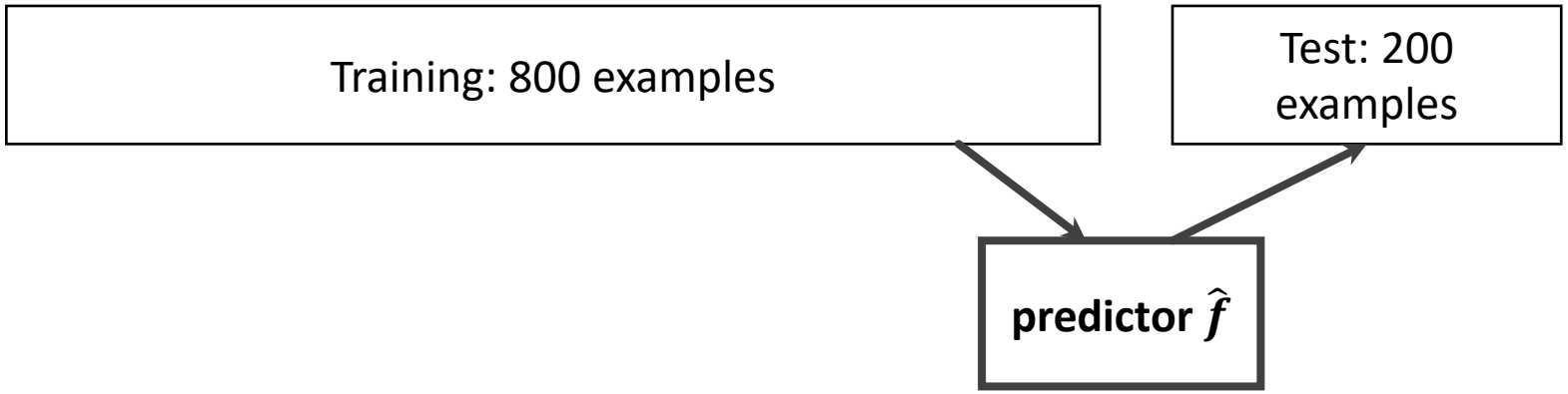
Overfitting vs Underfitting



Source: ibm.com

Unbiased model evaluation using test data

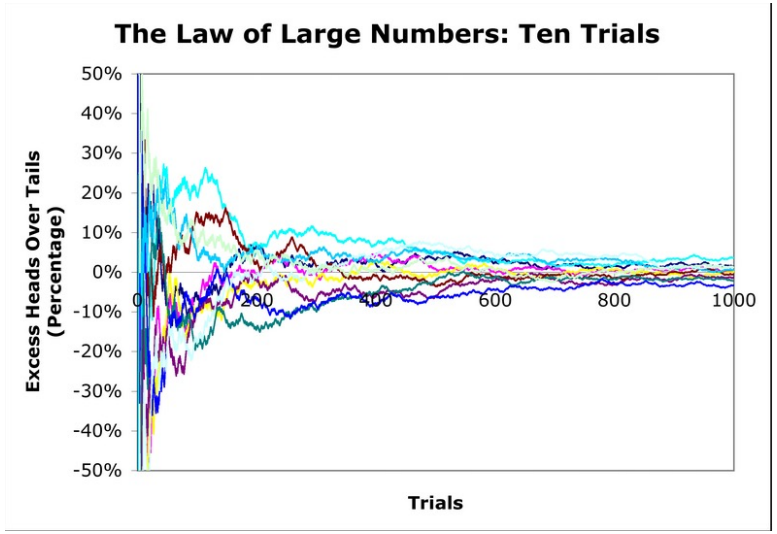
- Your boss says: I will allow your recommendation system to run on our website only if its error rate is <= 10%!
- How can we prove it to our boss?
- Idea: reserve some data as test data for evaluating predictors



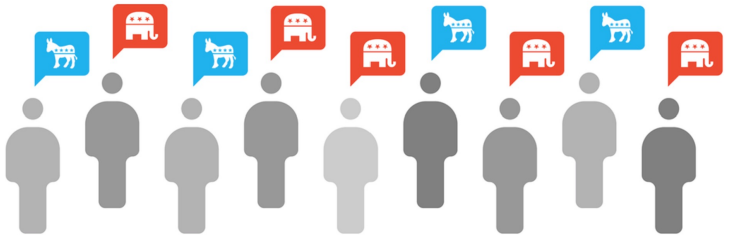
- Justification:
- $L_{\text{test}}(\hat{f}) = \frac{1}{|S_{\text{test}}|} \sum_{(x,y) \in S_{\text{test}}} I(y \neq \hat{f}(x))$
- Law of large numbers $\Rightarrow L_{\text{test}}(\hat{f}) \rightarrow L_D(\hat{f})$

Law of large numbers (LLN)

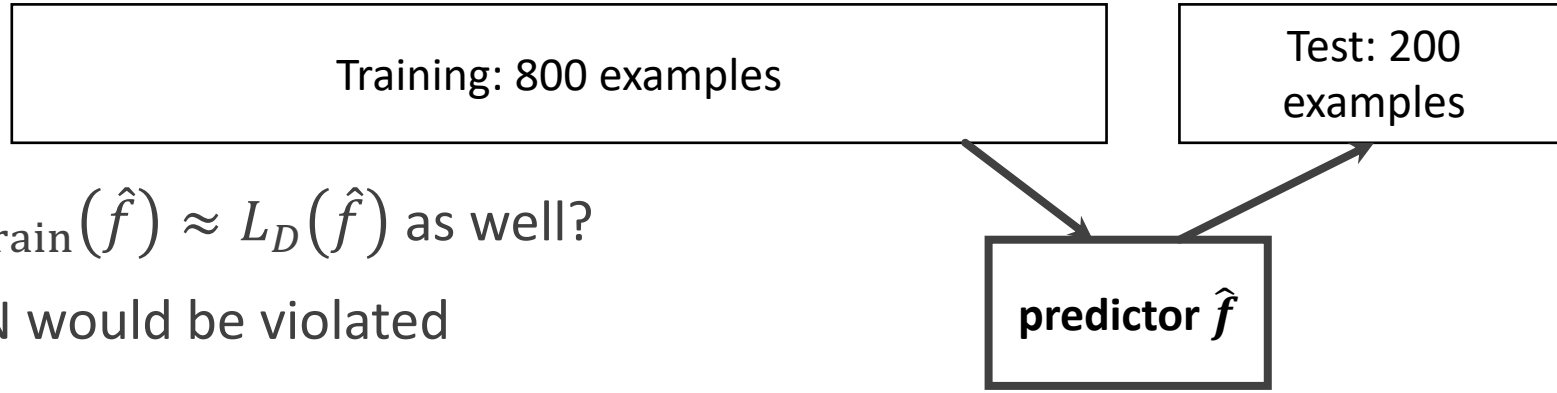
- Suppose $v_1, \ldots, v_n$ are IID (independent & identically distributed) random variables, the sample average $\bar{v} = \frac{1}{n}\sum_{i=1}^{n} v_i$ converges to $\mathrm{E}[v_1]$ as $n \to \infty$



- Useful in e.g. election poll
- Cornerstone of *Statistics*

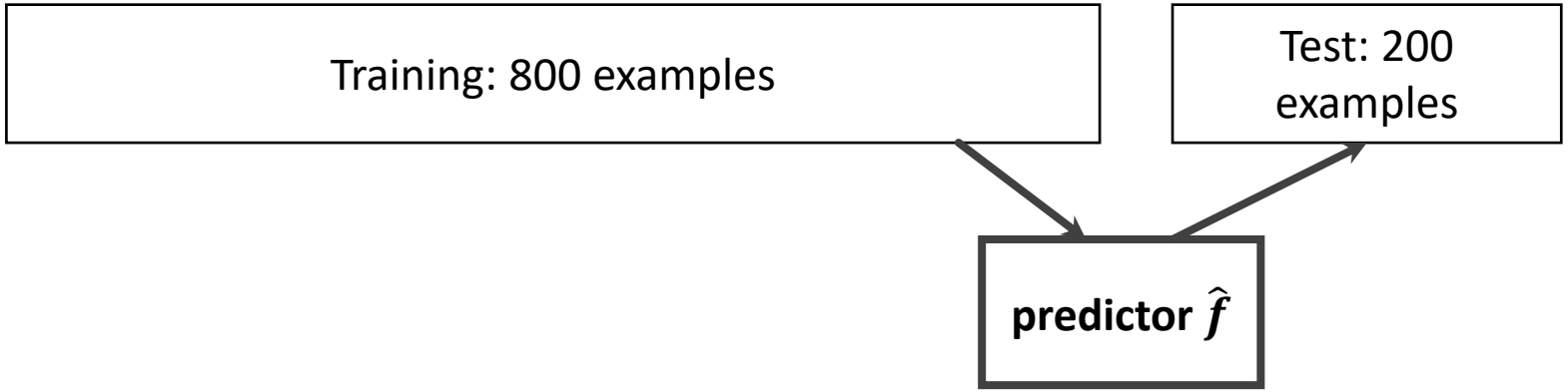


- LLN justifies that $L_{\text{test}}(\hat{f}) \approx L_D(\hat{f})$
- Can we apply LLN to conclude that $L_{\text{train}}(\hat{f}) \approx L_D(\hat{f})$ as well?
- **No!** The IID condition for applying LLN would be violated



Training can make the loss of $\hat{f}$ on each example *dependent* – e.g. $\hat{f}$ may be forced to predict correctly on exactly one of first two examples, making v_1 and v_2 dependent..

Never touch your test data (when training our model)!



- More precisely: test data should be used **only once** for **final evaluation**
- Otherwise, $\hat{f}$ depends on test examples, $L_{\text{test}}(\hat{f}) \approx L_D(\hat{f})$ may no longer be true
- Be mindful about indirect dependence as well:
  - adaptive data analysis – after seeing a previous algorithm doing badly on test data, develop a new learning algorithm that produces $\hat{f}$

Case Study: MNIST Dataset

All publications use standard train/test split

Hundreds of publications compare to each other

Type	Classifier	Distortion	Preprocessing	Error rate (%)
Linear classifier	Pairwise linear classifier	None	Deskewing	7.6[10]
Decision stream with Extremely randomized trees	Single model (depth > 400 levels)	None	None	2.7[28]
K-Nearest Neighbors	K-NN with rigid transformations	None	None	0.96[29]
K-Nearest Neighbors	K-NN with non-linear deformation (P2DHMDM)	None	Shiftable edges	0.52[30]
Boosted Stumps	Product of stumps on Haar features	None	Haar features	0.87[31]
Non-linear classifier	40 PCA + quadratic classifier	None	None	3.3[10]
Random Forest	Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)[32]	None	Simple statistical pixel importance	2.8[33]
Support-vector machine (SVM)	Virtual SVM, deg-9 poly, 2-pixel jittered	None	Deskewing	0.56[34]
Deep neural network (DNN)	2-layer 784-800-10	None	None	1.6[35]
Deep neural network	2-layer 784-800-10	Elastic distortions	None	0.7[35]
Deep neural network	6-layer 784-2500-2000-1500-1000-500-10	Elastic distortions	None	0.35[36]
Convolutional neural network (CNN)	6-layer 784-40-80-500-1000-2000-10	None	Expansion of the training data	0.31[37]
Convolutional neural network	6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.27[38]
Convolutional neural network (CNN)	13-layer 64-128(5x)-256(3x)-512-2048-256-256-10	None	None	0.25[22]
Convolutional neural network	Committee of 35 CNNs, 1-20-P-40-P-150-10	Elastic distortions	Width normalizations	0.23[17]
Convolutional neural network	Committee of 5 CNNs, 6-layer 784-50-100-500-1000-10-10	None	Expansion of the training data	0.21[24][25]
Random Multimodel Deep Learning (RMDL)	10 NN-10 RNN - 10 CNN	None	None	0.18[27]
Convolutional neural network	Committee of 20 CNNS with Squeeze-and-Excitation Networks[39]	None	Data augmentation	0.17[40]
Convolutional neural network	Ensemble of 3 CNNs with varying kernel sizes	None	Data augmentation consisting of rotation and translation	0.09[41]

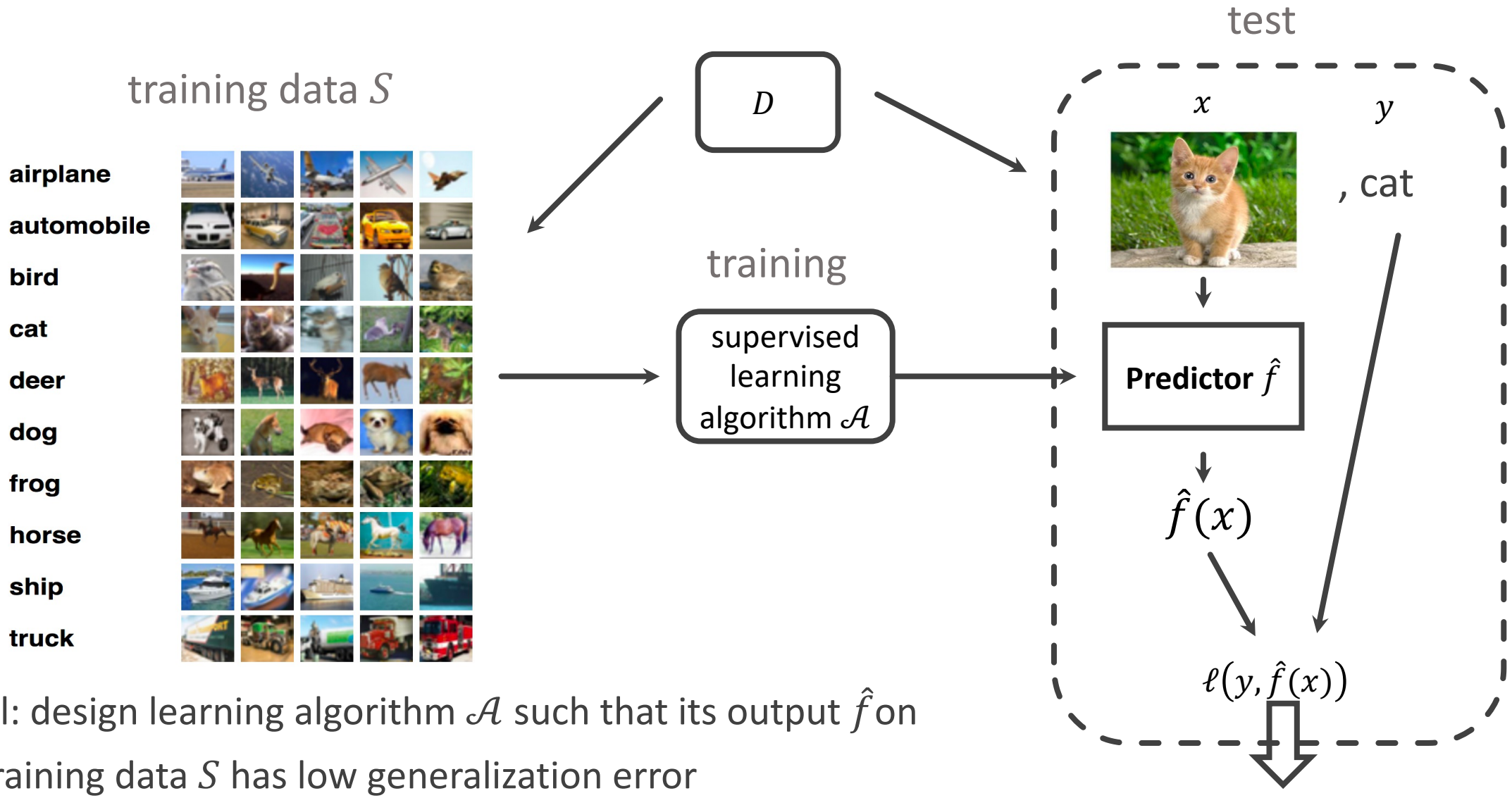
What's the problem with this?

The test dataset was publicly available at the very beginning

Perhaps some of the published works actually *overfit to the test dataset*

Combatting overfitting via hyperparameter tuning (aka model selection)

Supervised learning setup

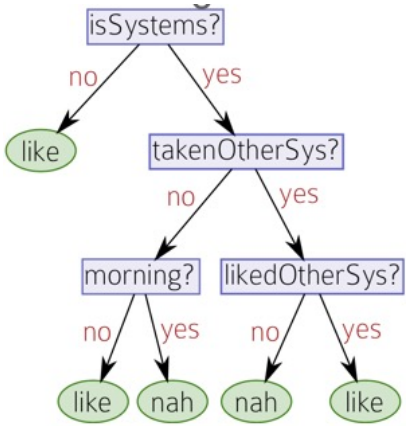


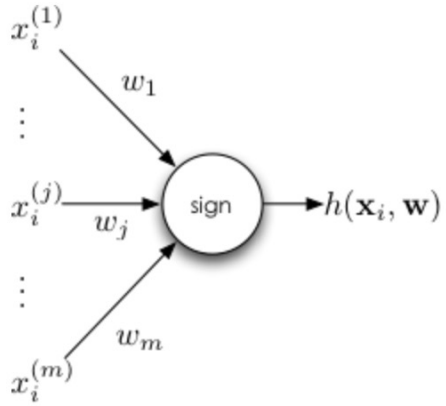
- Goal: design learning algorithm $\mathcal{A}$ such that its output $\hat{f}$ on iid training data S has low generalization error

Generalization error: $L_D(\hat{f}) = \mathrm{E}_{(x,y)\sim D}\, \ell(y, \hat{f}(x))$

Terminologies

- **Model:** the predictor $\hat{f}$
  - Often from a class $\mathcal{F}$ of models,
  - e.g. $\mathcal{F} = \{\text{depth} - 3 \text{ decision trees}\}, \{\text{linear classifiers}\}$





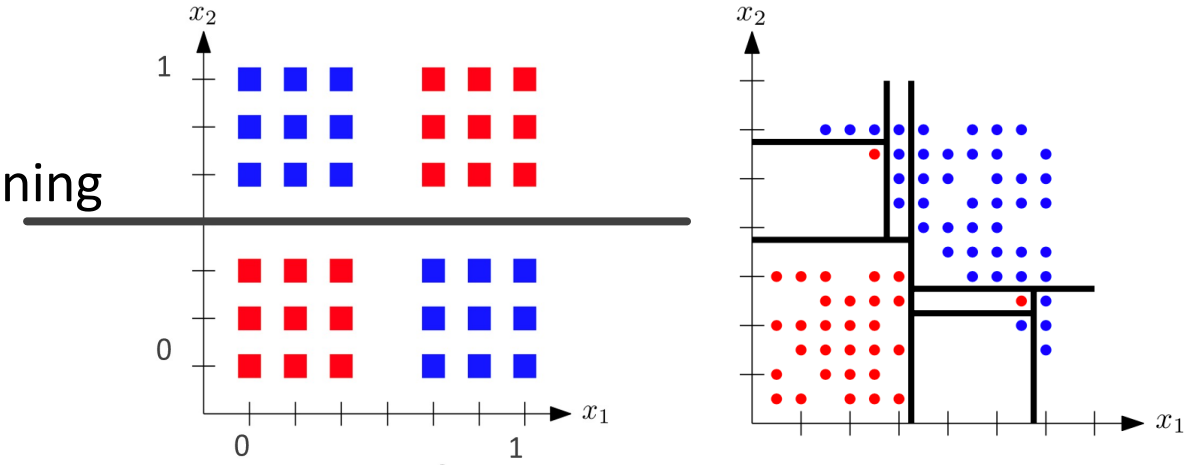
- **Parameter:** specifics of $\hat{f}$
  - E.g. for decision tree $\hat{f}$: tree structure, questions in nodes, labels in leaves
  - For linear classifier: linear coefficients

- **Hyperparameter:** specifics of learning algorithm $\mathcal{A}$
  - E.g. in DecisionTreeTrain, constrain to output trees of depth $\leq h$
  - Varying hyperparameters often results in {over, under}-fitting

Hyperparameter tuning using validation set

- E.g. in decision tree training, how to choose tree depth $h \in \{1, \dots, H\}$?

- For each hyperparameter $h \in \{1, \dots, H\}$:
  - Train Tree_h using DecisionTreeTrain by constraining the tree depth to be h
- Choose one from $\text{Tree}_1, \dots, \text{Tree}_H$



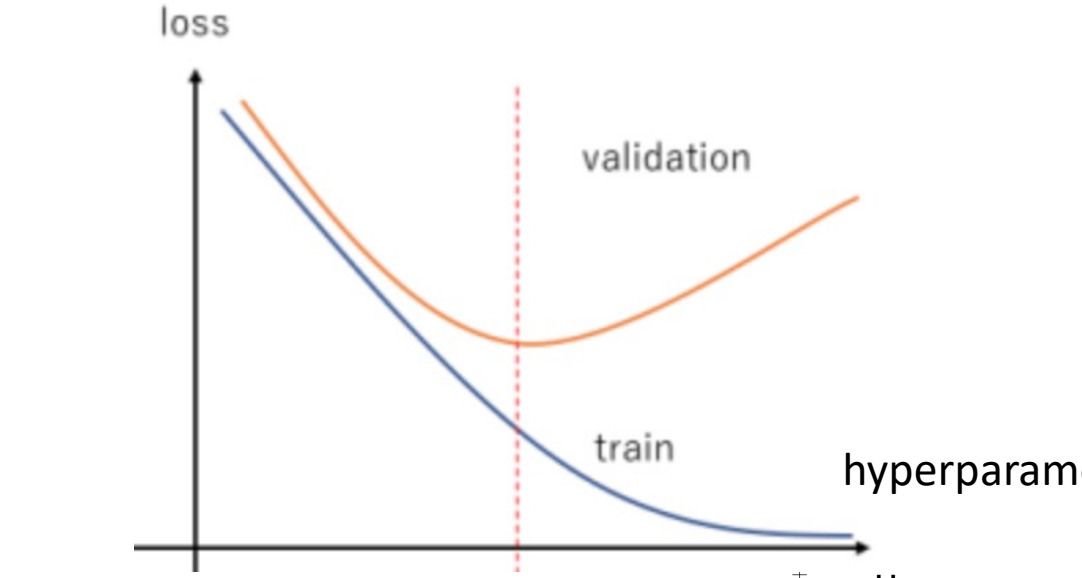
- Idea 1: choose Tree_h that minimizes training error ✗ Overfitting
- Idea 2: choose Tree_h that minimizes test error ✗ Never touch test set in training
- Idea 3: further split training set to model training set and validation set (development/hold-out set), (1) train Tree_h's using the (new) training set; (2) choose Tree_h that minimizes validation error

Model Training: 700 examples	Val:100 examples	Test: 200 examples

✓

Hyperparameter tuning using validation set

- E.g. in decision tree training, choose tree depth $h \in \{1, \ldots, H\}$ so that its trained model has the lowest validation loss

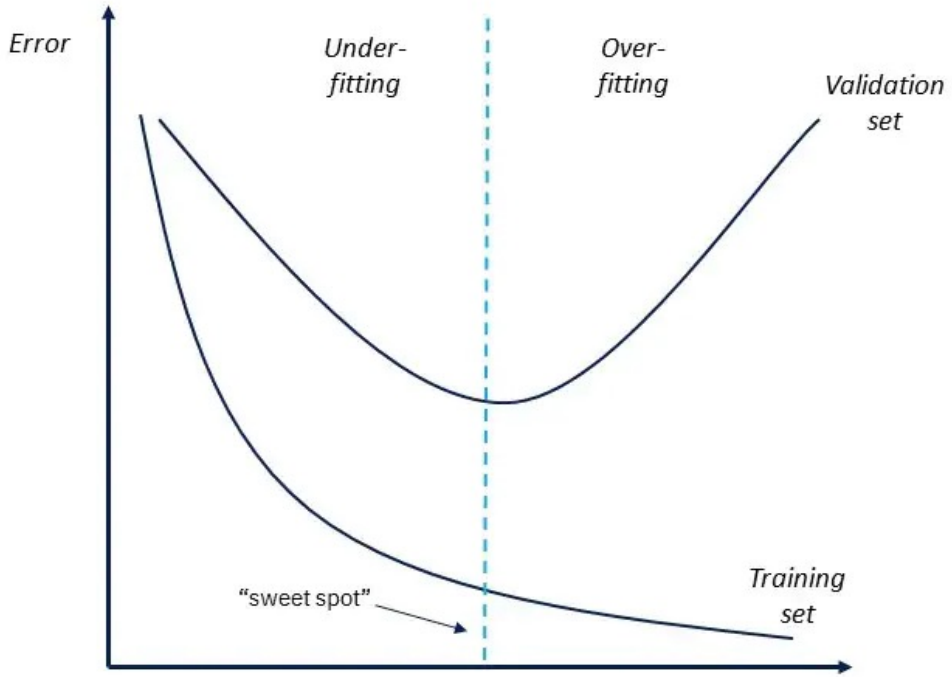


- Law of large numbers => Validation error closely approximates generalization error (& test error)

Overfitting vs Underfitting

Underfitting: performs poorly on *both* training and validation

Overfitting: performs well on training but not on validation

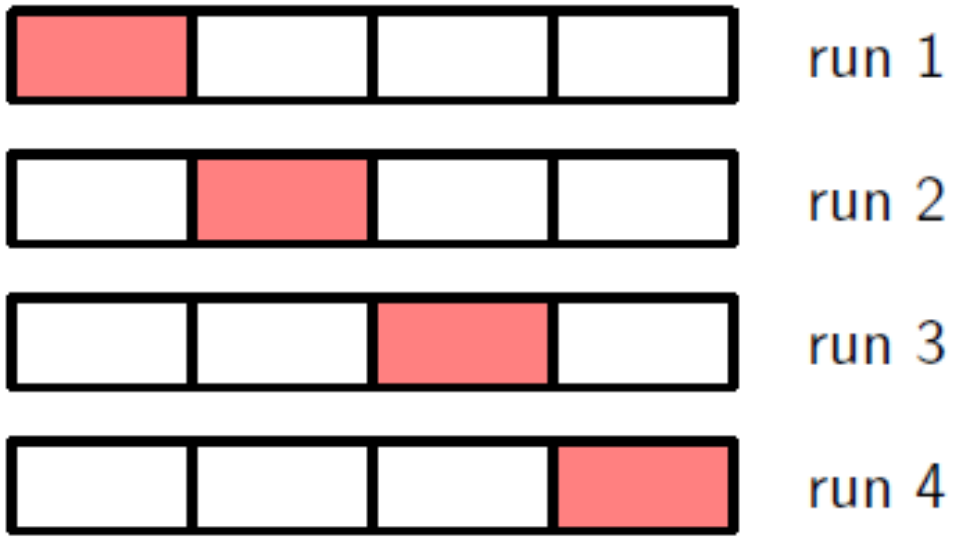


Source: ibm.com

- Note: this U-shaped validation error curve is typical, but may not always happen – "Benign overfitting" -- W-shaped error curve (Belkin et al, PNAS 2019)
- Nevertheless, choosing hyperparameters using validation error continues to be a good idea

Hyperparameter tuning: cross-validation

Main idea: improve data efficiency by splitting the training / validation data in multiple ways



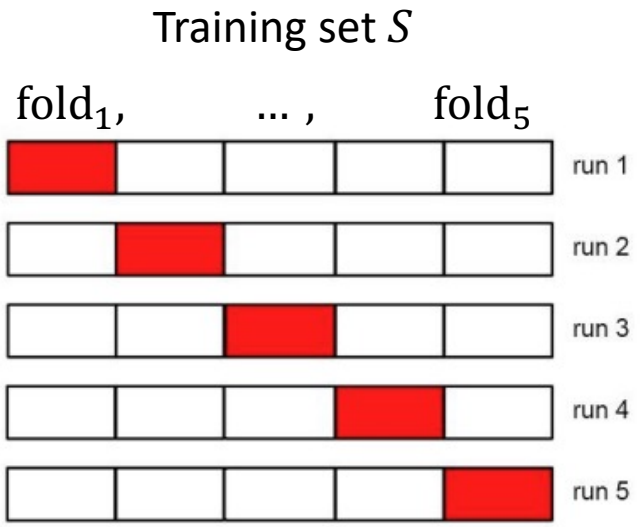
N-fold Cross Validation: Partition training data into N “chunks” and for each run select one chunk to be validation data

For each run, fit to training data (N-1 chunks) and measure accuracy on validation set. Average model error across all runs.

Source: Bishop, C. PRML

Cross-validation: formal description

- For hyperparameter $h \in \{1, \dots, H\}$
  - For $k \in \{1, \dots, K\}$
    - train f with $S \setminus \text{fold}_k$
    - measure error rate $e_{h,k}$ of f on fold_k
  - Compute the average error of the above: $E_h = \frac{1}{K}(e_{h,1} + \cdots + e_{h,K})$
- Choose $\hat{h} = \arg\min_h E_h$
- Train $\hat{f}$ using S (all training examples) with hyperparameter $\hat{h}$



- Typical K values: 5, 10
- Special case $K = |S|$: leave one out cross validation (LOOCV)

In-class exercise

- You have a dataset with 100 examples and you want to evaluate a decision tree learning algorithm's performance. For each of the following methods, how many decision trees do you need to train, and how many training examples will each decision tree be trained from?
  - Split data into 80% training and 20% validation.

    1 tree, 80

  - 10-fold cross validation.

    10 trees, 90

  - Leave-one-out.

    100 trees, 99

Miscellaneous concepts

Inductive bias

- What classification problem is class A vs. class B?
  - Birds vs. Non-birds
  - Flying animals vs. non-flying animals

- **Inductive bias**: if we don't have enough data to nail down the target concept, what type of predictors are we likely to prefer?

- What is the inductive bias of learning shallow decision trees?

Preferring decision trees that are shallow ("simple")

An example real-world machine learning pipeline

- Any step can go wrong
  - E.g. data collection, data representation

    e.g. collecting only mobile user's data

    need to consider position of ad as feature
- Debugging recommendation: run *oracle experiments*
  - Assuming all more basic tasks are perfectly done,

    is this step achieving what we want?

    e.g. assuming that data collection is perfect, is the model training step OK?
- General recommendation:
  - Build the stupidest thing that could possibly work
  - Then decide whether / where to fix it

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

Next lecture

- Geometric view of supervised learning; nearest neighbor methods

- Assigned reading: CIML Chap. 3 (Geometry and Nearest Neighbors)