Introduction to Machine Learning

Prof. Ziping Zhao

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

CS182: Introduction to Machine Learning (Fall 2021)
http://cs182.sist.shanghaitech.edu.cn

Outline

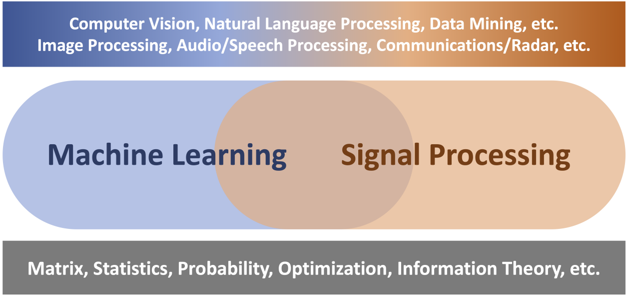
Outline

What is Machine Learning?

- **Machine Learning (ML)**
  is the science of
  making computer artifacts improve their performance
  with respect to a certain performance criterion
  using example data or past experience,
  without requiring humans to program their behavior explicitly.
- **Data Mining** (a.k.a. **knowledge discovery in databases (KDD)**)
  is the application of machine learning methods
  to large databases.

What is Machine Learning?

- **Machine Learning** and **Signal Processing (SP)** both rely on rigorous math foundations and share lots of similarities in theory and methods.



- These two subjects commonly belong to the discipline of MATH and also EECS.
- These two subjects facilitated the development of each other.
- Watch the video ML&SP: https://www.youtube.com/watch?v=2Wa245mSXrc

When is Machine Learning Needed?

- Human expertise is too expensive
  (e.g., intrusion detection, pathology)
- Human expertise does not exist
  (e.g., navigating on Mars)
- Humans cannot explain their expertise
  (e.g., speech recognition, visual perception)
- Problem (and hence solution) changes over time
  (e.g., network routing)
- ...

Some Characteristics of Machine Learning

- Data is (commonly) cheap and abundant ("Big Data"); knowledge is expensive and scarce.
- Details of the data generation process may be unknown, but the process is not completely random.
- Learning models from data by exploiting certain patterns or regularities in the data: inverting the data generation path.
- A model is often not an exact replica of the complete process, but is a good and useful approximation. (George Box: "All models are wrong, but some are useful.")
- A model may be descriptive to gain knowledge from data, or predictive to make predictions in the future, or both.
- Almost all of science is concerned with fitting models to data: inductive inference.

Roles of Statistics and Computer Science

- Machine learning makes extensive use of statistics in building mathematical models, because the core task is to make inference from a sample of observations.
- Role of computer science:
  - Developing efficient and accurate learning and inference algorithms
  - Common performance criteria:
    prediction accuracy, time complexity, space complexity.

Outline

Examples of Machine Learning Applications

- Learning associations
- Supervised learning (a.k.a. predictive learning):
  - Classification
  - Regression
- Unsupervised learning (a.k.a. descriptive learning or knowledge discovery)
- Reinforcement learning

- Other learning paradigms (e.g., semi-supervised learning, multi-task learning, multi-label learning) will not be considered in this introductory course.

Learning Associations

- Example: basket analysis
- In finding an association rule, we learn

$$P(Y \mid X)$$

which denotes the probability that somebody who buys product/service X also buys product/service Y .
- E.g., 70% of customers who buy beer also buy chips:

$$P(\text{chips} \mid \text{beer}) = 0.7$$

- To make a distinction among customers, we may instead learn

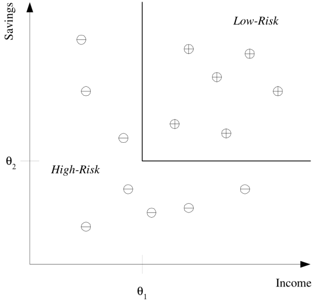
$$P(Y \mid X, D)$$

where D is the set of customer attributes, e.g., gender, age, etc.

Classification

- Example: credit scoring
- Differentiating between low-risk and high-risk customers from their income and savings.
- A classification rule (discriminant):

$$\text{IF income} > \theta_1 \text{ AND savings} > \theta_2 \text{ THEN low-risk ELSE high-risk}$$

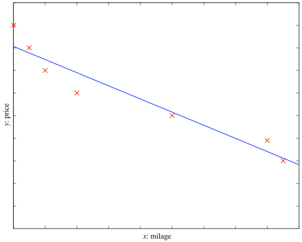


Classification: Other Applications

- Face detection and recognition
- Character recognition
- Speech recognition
- Object recognition and image classification
- Biometric authentication
- Multi-touch gesture classification
- Document classification
- Spam detection and filtering
- Intrusion detection
- Terrorism detection
- Medical diagnosis
- Weather forecasting
- ...

Regression

- Example: price prediction for used cars



- x: attributes of car
  y: price of car
- Model as regression function:

$$y = g(x \mid \theta)$$

  where $g(\cdot)$ is the model and θ denotes the parameters.

Regression: Other Applications

- Navigation of autonomous vehicle:
  angle of steering wheel
- Kinematics of robot arm:
  joint angles
- Recommender system:
  movie ratings
- Age estimation from facial image:
  age
- Chemical manufacturing process:
  yield
- Risk prediction from financial reports:
  risk
- ...

Supervised Learning

- Prediction of future cases:
  predict output for future input using the rule learned
- Knowledge extraction:
  rule is easier to understand
- Compression:
  rule is simpler than data it explains
- Outlier detection:
  exceptions not covered by rule, e.g., fraud

Unsupervised Learning

- Learning “what normally happens” and “interesting patterns” in the data
- More typical of human and animal learning than supervised learning
- No output given
- Much less well-defined than supervised learning with no obvious error measure
- Density estimation
- Dimensionality reduction/visualization: discovering latent factors
- Clustering (cluster discovery): grouping similar instances
- Examples of applications:
  - Customer segmentation in customer relationship management (CRM)
  - Image compression: color quantization
  - Bioinformatics: finding similar genes

Reinforcement Learning

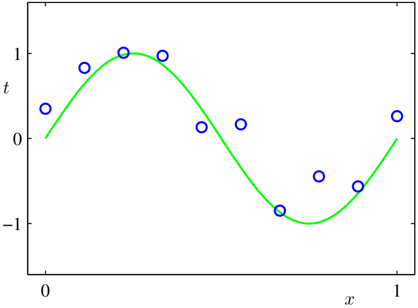
- Learning a policy (a sequence of actions) via a trial-and-error process (exploration vs. exploitation)
- No supervised output but delayed reward/penalty
- Credit assignment problem
- Examples of applications:
  - Game playing
  - Robot navigation in search of goal location
  - Individualized medical treatment for patients
  - Adaptive marketing campaign for maximizing long-term profits
- Some challenging issues:
  - Multiple agents
  - Partial observability of states

Outline

Polynomial Curve Fitting

- Regression problem:
  - Input variable: x
  - Target variable: t
- Data generation:

$$t = \sin(2\pi x) + \text{noise}$$



Polynomial Function as Linear Model

- Polynomial function for fitting data:

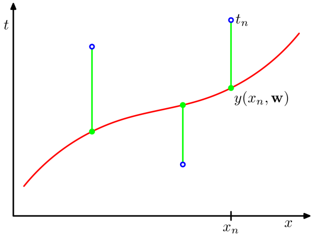
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

  where $\mathbf{w} = (w_0, \ldots, w_M)^T$ and M is the order of the polynomial.
- Linear model:
  The function $y(x, \mathbf{w})$ is nonlinear in x (if $M > 1$) but linear in $\mathbf{w}$.

Curve Fitting via Error Minimization

- Error function:

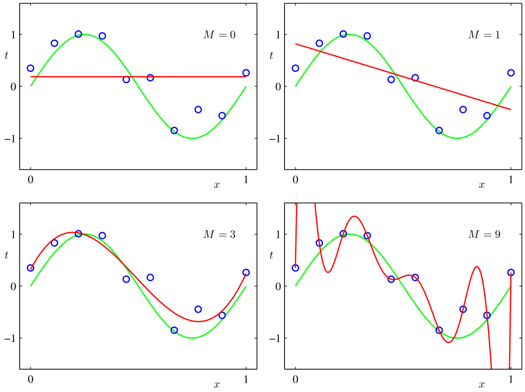
$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left[y(x_n, \mathbf{w}) - t_n \right]^2$$



- Optimal solution $\mathbf{w}^*$ that minimizes $E(\mathbf{w})$ can be found in closed form.

Order of Polynomial

- Choosing the order is an example of model selection.



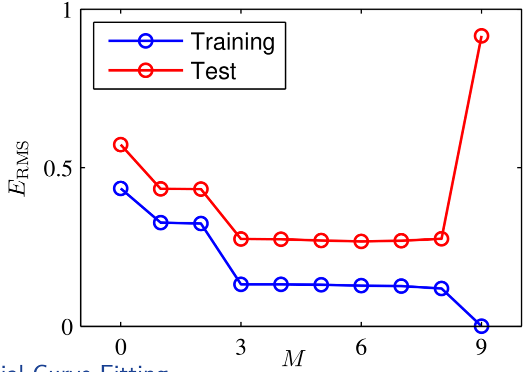
- For 10 points, it becomes the polynomial interpolation to find a 9st degree polynomial to predict the output for any future x.

Overfitting

- Root-mean-square (RMS) error:

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^*)/N}$$

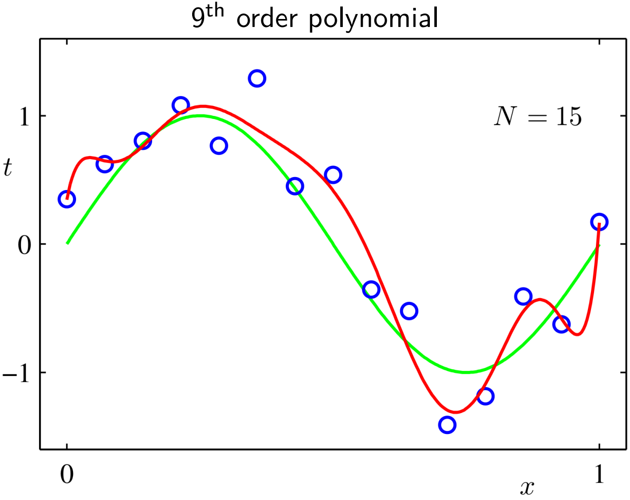
- Overfitting occurs at $M = 9$:



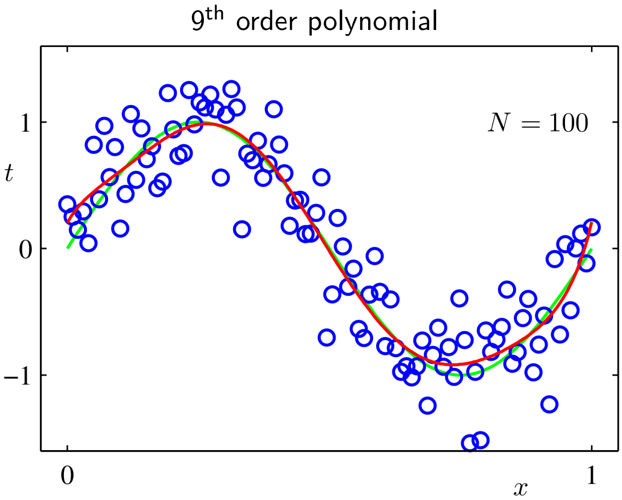
Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Sample Size $N = 15$



Sample Size $N = 100$



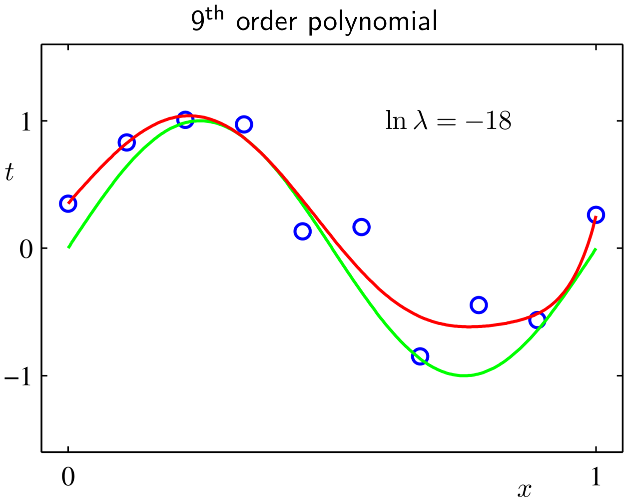
Regularization

- Regularized error function:

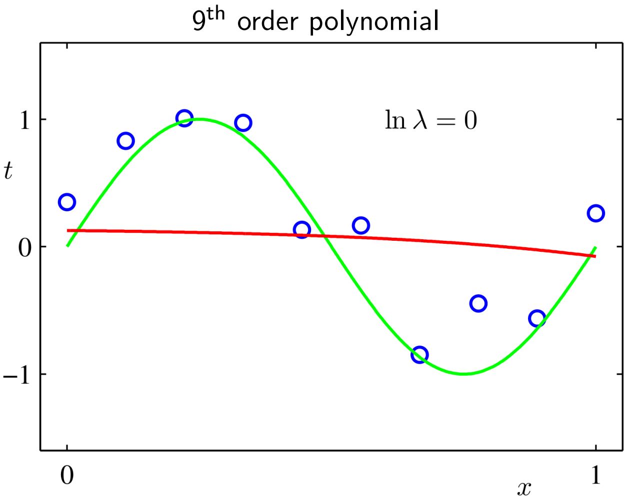
$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left[y(x_n,\mathbf{w}) - t_n\right]^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

where λ is a regularization parameter that governs the relative importance of the regularization term compared with the sum-of-squares error term.

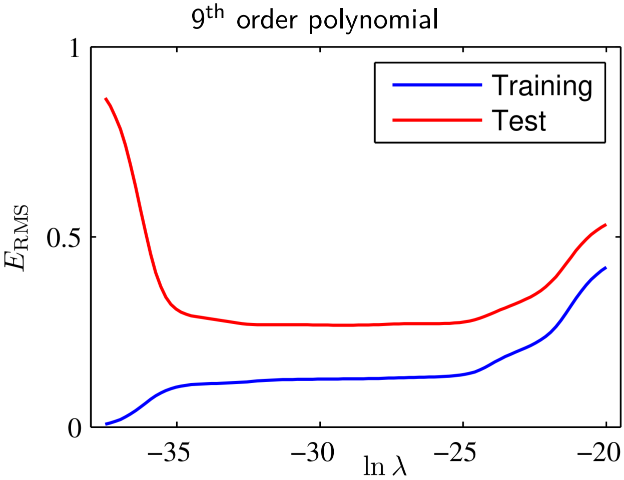
Regularization Parameter $\ln \lambda = -18$



Regularization Parameter $\ln \lambda = 0$



Regularization: E_{RMS} v.s. $\ln \lambda$



Polynomial Coefficients

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Outline

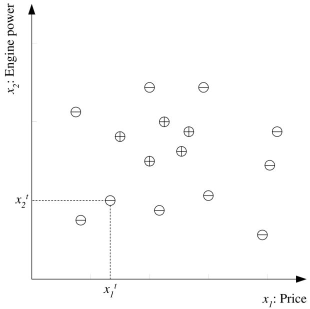
Learning a Class from Examples

- Task: learn the class C of family cars.
- Learning from examples:
  - Positive examples: family cars
  - Negative examples: other cars

  The training examples are labeled by humans.
- Class learning: find a description shared by all positive examples but no negative examples.
- Prediction: is the car x (not seen before in the training data) a family car?
- Input representation:
  - Identify features useful for discriminating family cars from other cars.
  - Represent each car x as a feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ formed by n features x_i, e.g., x_1 = price, x_2 = engine power.

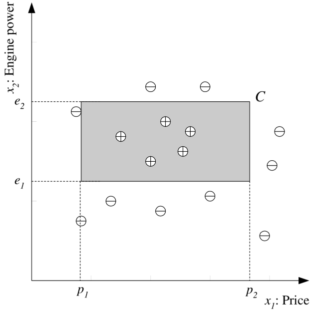
Training Set $\mathcal{X}$

- A set of N examples/instances: $\mathcal{X} = \{(\mathbf{x}^{(\ell)}, y^{(\ell)})\}_{\ell=1}^{N}$
- Each example, as an ordered pair $(\mathbf{x}^{(\ell)}, y^{(\ell)})$, corresponds to a car:
  - Feature vector: $\mathbf{x}^{(\ell)} = \left(x_1^{(\ell)}, x_2^{(\ell)}\right)^T$ (only 2 features for simplicity)
  - Class label: $y^{(\ell)} = \begin{cases} 1 & \text{if } \mathbf{x}^{(\ell)} \text{ is a positive example} \\ 0 & \text{if } \mathbf{x}^{(\ell)} \text{ is a negative example} \end{cases}$ (a Boolean value)



Class C

$$(p_1 \leq x_1 \leq p_2) \text{ AND } (e_1 \leq x_2 \leq e_2)$$



- The rectangle corresponds to a model (a.k.a. hypothesis) specified by 4 parameters.

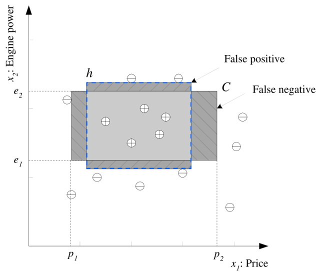
Hypothesis Class $\mathcal{H}$

- The learning algorithm finds a hypothesis $h \in \mathcal{H}$ to approximate the class C as closely as possible.

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } h \text{ classifies } \mathbf{x} \text{ as a positive example} \\ 0 & \text{if } h \text{ classifies } \mathbf{x} \text{ as a negative example} \end{cases}$$

- Each hypothesis is uniquely specified by a quadruple $(p_1^h, p_2^h, e_1^h, e_2^h)$ consisting of 4 parameters.
- We need to make sure that the hypothesis class/set $\mathcal{H}$ is flexible enough (or has enough capacity) to learn C, i.e., to find $h \in \mathcal{H}$ that is as similar as possible to C.

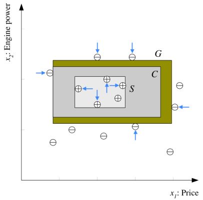
Empirical Error



- $C(\mathbf{x})$ is usually not known, so we cannot evaluate how well $h(\mathbf{x})$ matches $C(\mathbf{x})$.
- Empirical error (based on the training set $\mathcal{X}$):

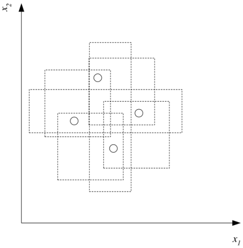
$$E(h \mid \mathcal{X}) = \sum_{\ell=1}^{N} \mathbf{1}_{[h(\mathbf{x}^{(\ell)}) \neq y^{(\ell)}]}$$

Version Space



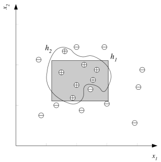
- Most specific hypothesis S: tightest rectangle in $\mathcal{H}$ that includes all positive examples but no negative examples.
- Most general hypothesis G: largest rectangle in $\mathcal{H}$ that includes all positive examples but no negative examples.
- Version space: the set of all $h \in \mathcal{H}$ between S and G (not all hypotheses in $\mathcal{H}$ are equally good in terms of generalization performance).
- The instances pointed are those that define (or support) the margin; other instances can be removed without affecting h.

Vapnik-Chervonenkis (VC) Dimension



- N points can be labeled in 2^N ways as $+/-$, one for a different learning problem.
- If for each of the 2^N learning problems, we can find a hypothesis $h \in \mathcal{H}$ that separates the positive and negative examples, then we say $\mathcal{H}$ shatters N points.
- VC dimension of $\mathcal{H}$, or VC($\mathcal{H}$): the maximum number of points that can be shattered by $\mathcal{H}$ (which measures the capacity of $\mathcal{H}$).
- VC($\{$axis-aligned rectangles$\}$) $= 4$; VC(a lookup table) $= +\infty$

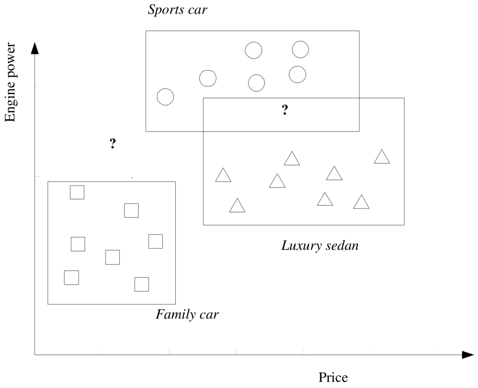
Noise and Model Complexity



- Sources of noise:
  - Measurement of attributes (input)
  - Labeling (output)
  - Hidden or latent attributes
- With a complex model, one can make a perfect fit and attain zero error.
- Using the simple rectangle (model) is preferred because:
  - Simpler to use with lower computational complexity
  - Easier to train with lower space complexity
  - Easier to explain with higher interpretability
  - Better generalization with lower variance (Occam's razor)

Multiple Classes

- In classification, we would like to learn the boundary separating the instances of one class from the instances of all other classes.
- A K-class classification problem can be regarded as K two-class problems.



Learning a Regression Function from Examples

- A set of N examples: $\mathcal{X} = \{(\mathbf{x}^{(\ell)}, y^{(\ell)})\}_{\ell=1}^{N}$
- Unlike classification problems, $y^{(\ell)} \in \mathbb{R}$ (a numeric value).
- Prediction via regression function:

$$y = f(\mathbf{x}) + \epsilon,$$

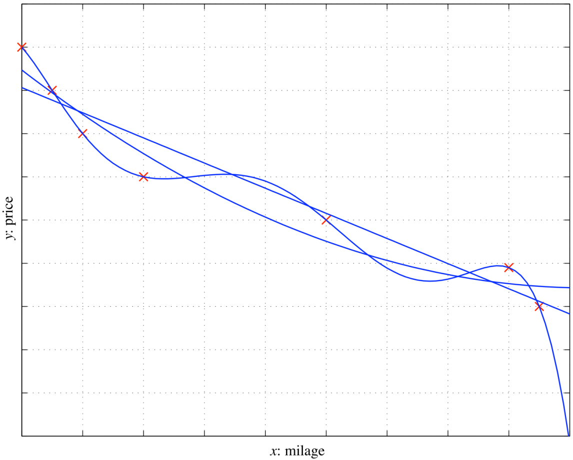
  where $f(\cdot)$ is the unknown regression function and ϵ is the random noise.
- Empirical error on $\mathcal{X}$:

$$E(f \mid \mathcal{X}) = \frac{1}{N} \sum_{\ell=1}^{N} \left[y^{(l)} - f(\mathbf{x}^{(l)}) \right]^2.$$

- The square of the difference is one error (loss) function that can be used; there are many others.
- Regression: find $f(\cdot)$ that minimizes the empirical error $E(f \mid \mathcal{X})$.

Polynomials

First-order (linear), second-order and sixth-order polynomials



Model Selection and Generalization

- Learning is an ill-posed problem; data alone is not sufficient to find a unique solution.
- Inductive bias (i.e., assumptions about $\mathcal{H}$) is needed.
  - In learning the class of family cars, assuming the shape of a rectangle is one inductive bias.
  - In linear regression, assuming a linear function is an inductive bias, and among all lines, choosing the one that minimizes squared error is another inductive bias.
- Generalization: how well a learned model performs on new data not seen before.
- Overfitting: $\mathcal{H}$ is more complex than C or f .
- Underfitting: $\mathcal{H}$ is less complex than C or f .
- Model selection: choosing the right inductive bias.

Triple Tradeoff

- Tradeoff between three factors:
  - Complexity of $\mathcal{H}$, $c(\mathcal{H})$ (capacity of the hypothesis class)
  - Training set size, N
  - Generalization error on new data, E
- $N \uparrow \quad \rightsquigarrow \quad E \downarrow$
- $c(\mathcal{H}) \uparrow \quad \rightsquigarrow \quad E \downarrow$ then $E \uparrow$

Cross-Validation

- To estimate the generalization error, we need data unseen during model training.
- Data splitting:
  - Training set (e.g., 50%)
  - Validation set (e.g., 25%)
  - Test set (e.g., 25%)
- Resampling is needed when data is limited.

Dimensions of a Supervised Learning Algorithm

- Three dimensions:
  - Model:
    $$g(\mathbf{x} \mid \theta)$$
  - Loss function:
    $$E(\theta \mid \mathcal{X}) = \sum_{\ell} L\big(y^{(l)}, g(\mathbf{x}^{(l)} \mid \theta)\big)$$
  - Optimization procedure/algorithm:
    $$\theta^* = \arg\min_{\theta} \quad E(\theta \mid \mathcal{X})$$
- No free lunch theorem:
  - There is no universally best model.
  - Different types of models have to be developed to suit the nature of the data in real applications.