Lecture 1: Theoretical foundations of ML

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Advanced Machine Learning
September–December 2021

Supervised learning

- Set of possible input instances X

Supervised learning

- Set of possible input instances X
- Categories C, say $\{0, 1\}$

Supervised learning

- Set of possible input instances X
- Categories C, say $\{0, 1\}$
- Build a classification model $M : X \to C$

Supervised learning

- Set of possible input instances X
- Categories C, say $\{0, 1\}$
- Build a classification model $M : X \to C$
- Restrict the types of models
  - Hypothesis space $\mathcal{H}$ — e.g., linear separators
  - Search for best $M \in \mathcal{H}$

Supervised learning

- Set of possible input instances X
- Categories C, say $\{0, 1\}$
- Build a classification model $M : X \to C$
- Restrict the types of models
  - Hypothesis space $\mathcal{H}$ — e.g., linear separators
  - Search for best $M \in \mathcal{H}$
- How do we find the best M?
  - Labelled training data
  - Choose M to minimize error (loss) with respect to this set
  - Why should M generalize well to arbitrary data?

No free lunch

- ML algorithms minimize training loss

No free lunch

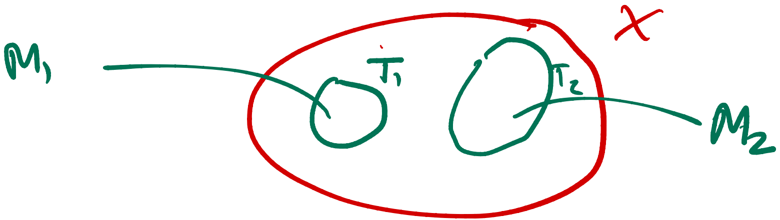
- ML algorithms minimize training loss
- Goal is to minimize generalization loss

No free lunch

- ML algorithms minimize training loss
- Goal is to minimize generalization loss

No Free Lunch Theorem [Wolpert, Macready 1997]

Averaged over all possible data distributions, every classification algorithm has the same error rate when classifying previously unobserved points.



No free lunch

- ML algorithms minimize training loss
- Goal is to minimize generalization loss

No Free Lunch Theorem [Wolpert, Macready 1997]

Averaged over all possible data distributions, every classification algorithm has the same error rate when classifying previously unobserved points.

- Is the situation hopeless?

No free lunch

- ML algorithms minimize training loss
- Goal is to minimize generalization loss

No Free Lunch Theorem [Wolpert, Macready 1997]

Averaged over all possible data distributions, every classification algorithm has the same error rate when classifying previously unobserved points.

- Is the situation hopeless?
- NFL theorem refers to prediction inputs coming from all possible distributions

No free lunch

- ML algorithms minimize training loss
- Goal is to minimize generalization loss

No Free Lunch Theorem [Wolpert, Macready 1997]

Averaged over all possible data distributions, every classification algorithm has the same error rate when classifying previously unobserved points.

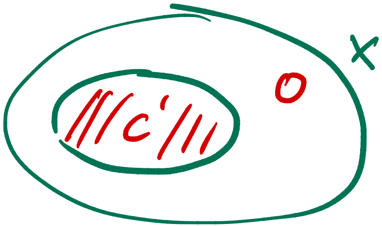
- Is the situation hopeless?
- NFL theorem refers to prediction inputs coming from all possible distributions
- ML assumes training set is “representative” of overall data
  - Prediction instances follow roughly the same distribution as training set

A theoretical framework for ML

- X is the space of input instances

A theoretical framework for ML

- X is the space of input instances
- $C \subseteq X$ is the target concept to be learned
  - e.g., X is all emails, C is the set of spam emails

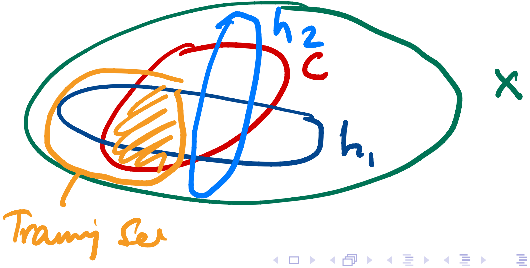


A theoretical framework for ML

- X is the space of input instances
- $C \subseteq X$ is the target concept to be learned
  - e.g., X is all emails, C is the set of spam emails
- X is equipped with a probability distribution D
  - Any random sample from X is drawn using D
  - In particular, training set and test set are such random samples

A theoretical framework for ML

- X is the space of input instances
- $C \subseteq X$ is the target concept to be learned
  - e.g., X is all emails, C is the set of spam emails
- X is equipped with a probability distribution D
  - Any random sample from X is drawn using D
  - In particular, training set and test set are such random samples
- $\mathcal{H}$ is a set of hypotheses
  - Each $h \in \mathcal{H}$ identifies a subset of X
  - Choose the best $h \in \mathcal{H}$ as model



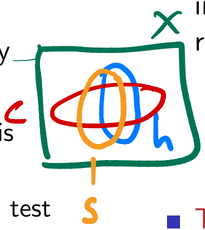
A theoretical framework for ML

- X is the space of input instances
- $C \subseteq X$ is the target concept to be learned
  - e.g., X is all emails, C is the set of spam emails
- X is equipped with a probability distribution D
  - Any random sample from X is drawn using D
  - In particular, training set and test set are such random samples
- $\mathcal{H}$ is a set of hypotheses
  - Each $h \in \mathcal{H}$ identifies a subset of X
  - Choose the best $h \in \mathcal{H}$ as model
- True error: Probability that h incorrectly classifies $x \in X$ drawn randomly according to D
  - $\text{err}_D(h) = \text{Prob}(h \Delta C)$
  - $h \Delta C = (h \setminus C) \cup (C \setminus h)$ is the symmetric difference

$h \subseteq X$
$C \subseteq X$

A theoretical framework for ML

- X is the space of input instances
- $C \subseteq X$ is the target concept to be learned
  - e.g., X is all emails, C is the set of spam emails
- X is equipped with a probability distribution D
  - Any random sample from X is drawn using D
  - In particular, training set and test set are such random samples



- $\mathcal{H}$ is a set of hypotheses
  - Each $h \in \mathcal{H}$ identifies a subset of X
  - Choose the best $h \in \mathcal{H}$ as model
- True error: Probability that h incorrectly classifies $x \in X$ drawn randomly according to D
  - $\text{err}_D(h) = \text{Prob}(h \Delta C)$
  - $h \Delta C = (h \setminus C) \cup (C \setminus h)$ is the symmetric difference
- Training error: Given a (finite) training sample $S \subseteq X$
  - $\text{err}_S(h) = |S \cap (h \Delta C)|/|S|$

A theoretical framework for ML

- X, inputs with distribution D
- $C \subseteq X$, target concept
- $h \in \mathcal{H}$, hypothesis (model) for C
- True error: $\mathrm{err}_D(h) = \mathrm{Prob}(h \Delta C)$
- Training error:
  $\mathrm{err}_S(h) = |S \cap (h \Delta C)|/|S|$

A theoretical framework for ML

- X, inputs with distribution D
- $C \subseteq X$, target concept
- $h \in \mathcal{H}$, hypothesis (model) for C
- True error: $\text{err}_D(h) = \text{Prob}(h \Delta C)$
- Training error:
  $\text{err}_S(h) = |S \cap (h \Delta C)| / |S|$

Goal

Minimizing training error should correspond to minimizing true error

A theoretical framework for ML

- X, inputs with distribution D
- $C \subseteq X$, target concept
- $h \in \mathcal{H}$, hypothesis (model) for C
- True error: $\text{err}_D(h) = \text{Prob}(h \Delta C)$
- Training error: $\text{err}_S(h) = |S \cap (h \Delta C)|/|S|$

Goal

Minimizing training error should correspond to minimizing true error

- Overfitting Low training error but high true error

A theoretical framework for ML

- X, inputs with distribution D
- $C \subseteq X$, target concept
- $h \in \mathcal{H}$, hypothesis (model) for C
- True error: $\text{err}_D(h) = \text{Prob}(h \Delta C)$
- Training error: $\text{err}_S(h) = |S \cap (h \Delta C)|/|S|$

Goal

Minimizing training error should correspond to minimizing true error

- Overfitting Low training error but high true error
- Underfitting Cannot achieve low training/true error

A theoretical framework for ML

- X, inputs with distribution D
- $C \subseteq X$, target concept
- $h \in \mathcal{H}$, hypothesis (model) for C
- True error: $\text{err}_D(h) = \text{Prob}(h \Delta C)$
- Training error: $\text{err}_S(h) = |S \cap (h \Delta C)|/|S|$

Goal

Minimizing training error should correspond to minimizing true error

- Overfitting Low training error but high true error
- Underfitting Cannot achieve low training/true error
- Related to the representational capacity of $\mathcal{H}$
  - How expressive is $\mathcal{H}$? How many different concepts can it capture?
  - Capacity too high — overfitting
  - Capacity too low — underfitting

Probably Approximately Correct (PAC) learning

- Assume $\mathcal{H}$ is finite — use $|\mathcal{H}|$ for capacity

Probably Approximately Correct (PAC) learning

- Assume $\mathcal{H}$ is finite — use $|\mathcal{H}|$ for capacity
- Probably Approximately Correct learning

  With high probability, the hypothesis h that fits the sample S also fits the concept approximately correctly

Probably Approximately Correct (PAC) learning

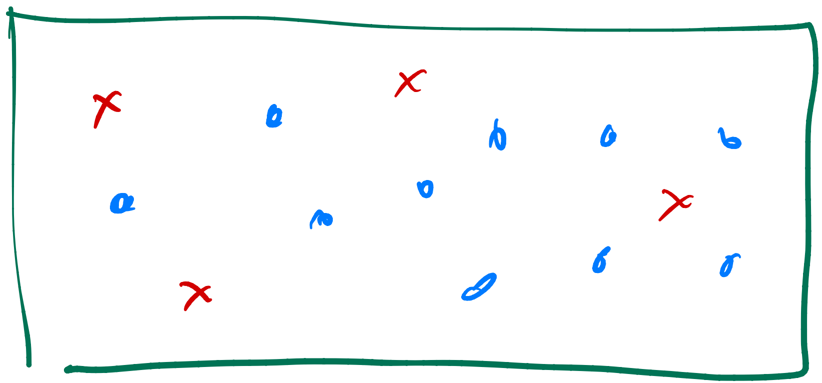
- Assume $\mathcal{H}$ is finite — use $|\mathcal{H}|$ for capacity
- Probably Approximately Correct learning

  With high probability, the hypothesis h that fits the sample S also fits the concept approximately correctly

Theorem (PAC learning guarantee)

Let $\delta, \epsilon > 0$. Let S be a training set of size $n \geq \frac{1}{\epsilon}(\ln|\mathcal{H}| + \ln(1/\delta))$ drawn using D. With probability $\geq 1 - \delta$, every $h \in \mathcal{H}$ with training error zero has true error $< \epsilon$.

- Size of the sample required for PAC guarantee determined by parameters δ, ϵ
  - Smaller δ means higher probability of find a good hypothesis
  - Smaller ϵ means better performance with respect to generalization



Small Samples

All red

All blue

Big samples

All red is hard

$|S| > 4$

Probably Approximately Correct (PAC) learning

Theorem (Uniform convergence)

Let $\delta, \epsilon > 0$. Let S be a training set of size $n \geq \frac{1}{2\epsilon^2}(\ln|\mathcal{H}| + \ln(2/\delta))$ drawn using D. With probability $\geq 1 - \delta$, every $h \in \mathcal{H}$ satisfies $|\text{err}_S(h) - \text{err}_D(h)| \leq \epsilon$.

- Stronger guarantee: even if we cannot achieve zero training error, the additional generalization error is bounded

Optimize within training set

Probably Approximately Correct (PAC) learning

Theorem (Uniform convergence)

Let $\delta, \epsilon > 0$. Let S be a training set of size $n \geq \frac{1}{2\epsilon^2}(\ln|\mathcal{H}| + \ln(2/\delta))$ drawn using D. With probability $\geq 1 - \delta$, every $h \in \mathcal{H}$ satisfies $|\text{err}_S(h) - \text{err}_D(h)| \leq \epsilon$.

- Stronger guarantee: even if we cannot achieve zero training error, the additional generalization error is bounded
- What if $\mathcal{H}$ is not finite?

Probably Approximately Correct (PAC) learning

Theorem (Uniform convergence)

Let $\delta, \epsilon > 0$. Let S be a training set of size $n \geq \frac{1}{2\epsilon^2}(\ln|\mathcal{H}| + \ln(2/\delta))$ drawn using D. With probability $\geq 1 - \delta$, every $h \in \mathcal{H}$ satisfies $|\mathrm{err}_S(h) - \mathrm{err}_D(h)| \leq \epsilon$.

- Stronger guarantee: even if we cannot achieve zero training error, the additional generalization error is bounded
- What if $\mathcal{H}$ is not finite?
- Other measures of capacity — e.g. VC-dimension

Probably Approximately Correct (PAC) learning

Theorem (Uniform convergence)

Let $\delta, \epsilon > 0$. Let S be a training set of size $n \geq \frac{1}{2\epsilon^2}(\ln|\mathcal{H}| + \ln(2/\delta))$ drawn using D. With probability $\geq 1-\delta$, every $h \in \mathcal{H}$ satisfies $|\mathrm{err}_S(h) - \mathrm{err}_D(h)| \leq \epsilon$.

- Stronger guarantee: even if we cannot achieve zero training error, the additional generalization error is bounded
- What if $\mathcal{H}$ is not finite?
- Other measures of capacity — e.g. VC-dimension
- Analogous convergence theorems in terms of VC-dimension

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting

Overfitting and underfitting

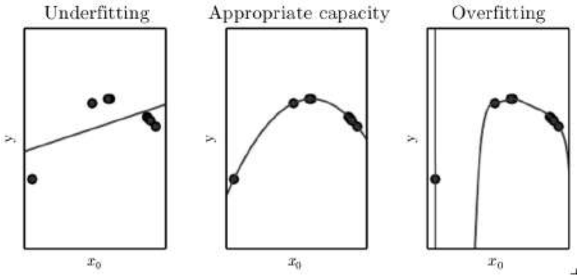
Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting
- Using too low a d results in underfitting

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting
- Using too low a d results in underfitting

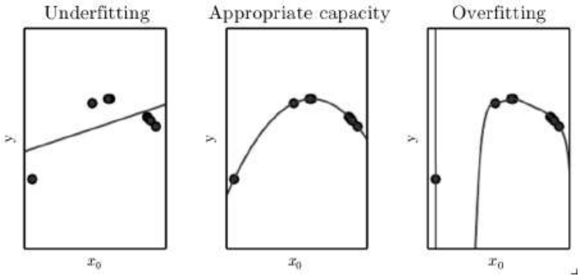


- Random points lying along a quadratic

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting
- Using too low a d results in underfitting

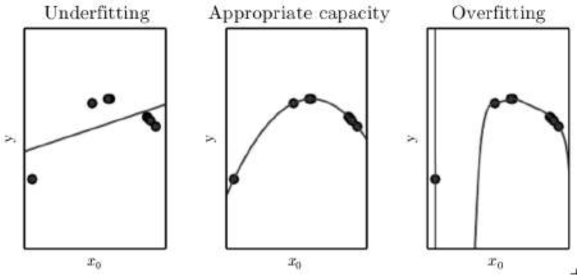


- Random points lying along a quadratic
- Linear function underfits

Overfitting and underfitting

Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting
- Using too low a d results in underfitting

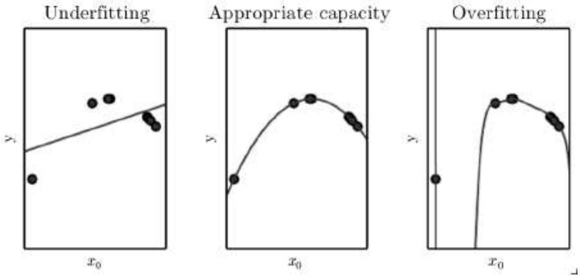


- Random points lying along a quadratic
- Linear function underfits
- Quadratic fits and generalizes well

Overfitting and underfitting

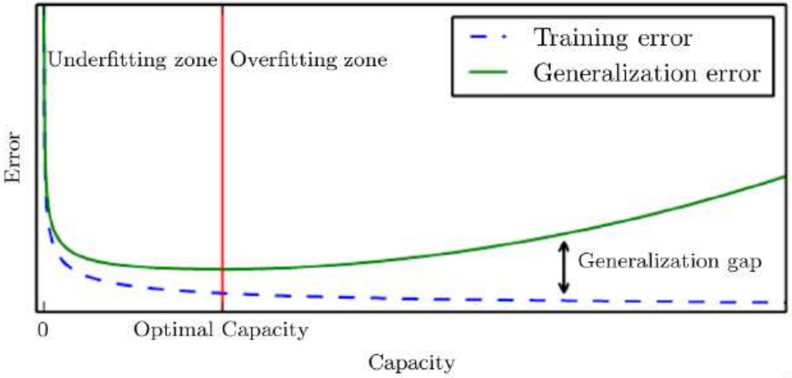
Example: Regression

- $\mathcal{H}_d$ is set of polynomials of degree d
- Increasing d increases expressiveness — higher representational capacity
- Using too high a d results in overfitting
- Using too low a d results in underfitting



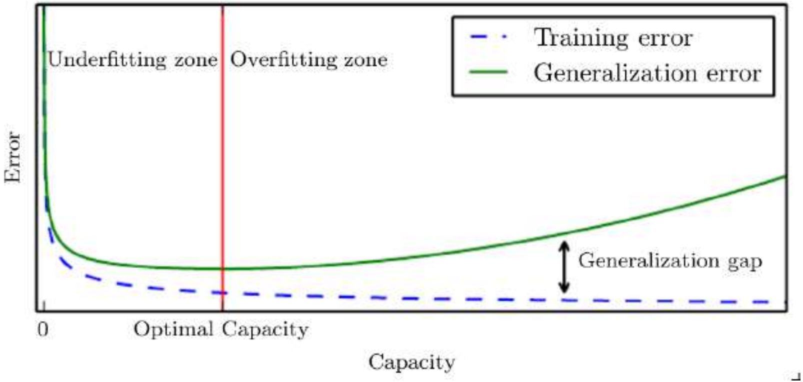
- Random points lying along a quadratic
- Linear function underfits
- Quadratic fits and generalizes well
- Degree 9 polynomial overfits

Capacity and error



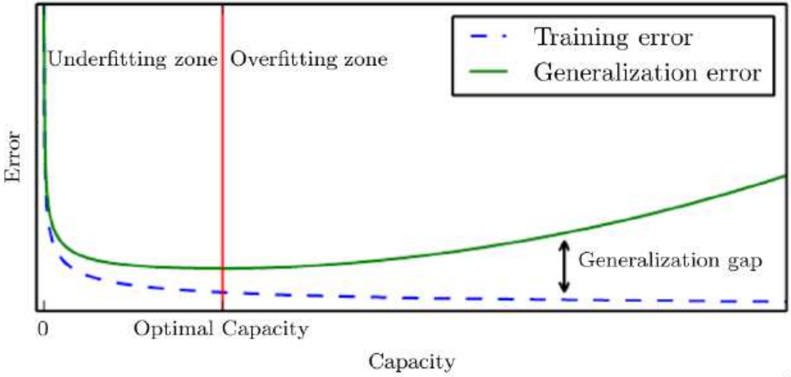
- As capacity increases, training error decreases

Capacity and error



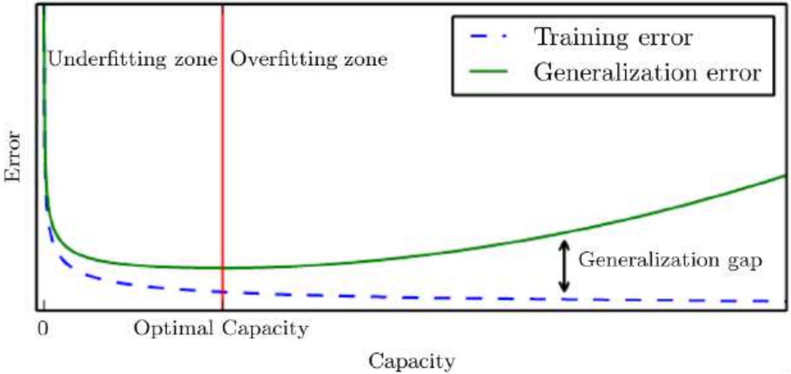
- As capacity increases, training error decreases
- Initially, generalization error also decreases

Capacity and error



- As capacity increases, training error decreases
- Initially, generalization error also decreases
- At some point, generalization error starts increasing

Capacity and error



- As capacity increases, training error decreases
- Initially, generalization error also decreases
- At some point, generalization error starts increasing
- Optimum capacity is not where training error is minimum

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly
- May not even be able to achieve true representational capacity

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly
- May not even be able to achieve true representational capacity
- Effective capacity limited by capabilities of parameter estimation algorithm (backpropagation with optimization)

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly
- May not even be able to achieve true representational capacity
- Effective capacity limited by capabilities of parameter estimation algorithm (backpropagation with optimization)
- Parameter estimation is a complex nonlinear optimization

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly
- May not even be able to achieve true representational capacity
- Effective capacity limited by capabilities of parameter estimation algorithm (backpropagation with optimization)
- Parameter estimation is a complex nonlinear optimization

Regularization

- Add a penalty for model complexity to the loss function
- Trade off lower training error against penalty

Theory and practice

- Deep learning models are too complex to compute representational capacity explicitly
- May not even be able to achieve true representational capacity
- Effective capacity limited by capabilities of parameter estimation algorithm (backpropagation with optimization)
- Parameter estimation is a complex nonlinear optimization

Regularization

- Add a penalty for model complexity to the loss function
- Trade off lower training error against penalty

Hyperparameters

- Settings that adjust the capacity — e.g., degree of polynomial
- Set externally, not learned
- Search hyperparameter combinations for optimal settings

Summary

- Supervised learning builds a model that minimize training error

Summary

- Supervised learning builds a model that minimize training error
- Real goal is to minimize generalization error

Summary

- Supervised learning builds a model that minimize training error
- Real goal is to minimize generalization error
- PAC learning provides a theoretical framework to justify this

Summary

- Supervised learning builds a model that minimize training error
- Real goal is to minimize generalization error
- PAC learning provides a theoretical framework to justify this
- Discrepancies in representational capacity of models can cause underfitting or overfitting

Summary

- Supervised learning builds a model that minimize training error
- Real goal is to minimize generalization error
- PAC learning provides a theoretical framework to justify this
- Discrepancies in representational capacity of models can cause underfitting or overfitting
- In practice, use regularization and hyperparameter search to identify optimum capacity