Machine Learning: Supervised Methods NOTES

Kristian Bonnici

October 4, 2021

Contents

Abstract

These notes cover some of the key concepts in supervised learning. However it's not intended to be a comprehensive introduction into supervised methods. To get most out of the notes, one should have some base knowledge on machine learning and basic algebra.

Part I
Theory

1 Introduction

1.1 Theoretical paradigms

Theoretical paradigms for machine learning **differ** mainly on what they assume about the process generating the data:

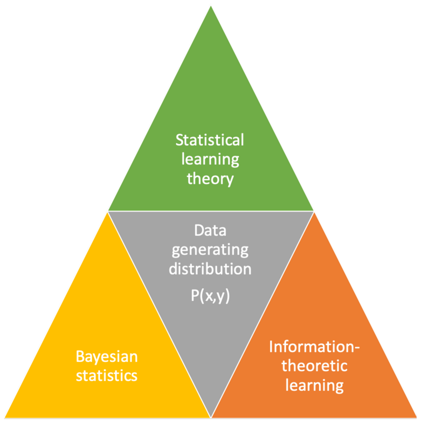
Statistical learning theory
Data generating distribution
P(x,y)
Bayesian statistics
Information-theoretic learning

Figure 1: Paradigms for data generation distributions

- **Statistical learning theory (focus on this course):** assumes data is i.i.d from an unknown distribution P(x), does not estimate the distribution (directly)
- **Bayesian Statistics:** assumes prior information on P(x), estimates posterior probabilities
- **Information theoretic learning:** (e.g.Minimum Description Length principle, MDL): estimates distributions, but does not assume a prior on P(x)

1.2 Dimensions of a supervised learning algorithm

1. **Training sample:** $S = \{(x_i, y_i)\}_{i=1}^{m}$ the training examples $(x, y) \in X \times Y$ independently drawn from a identical distribution $(i.i.d) D$ defined on $X \times Y, X$ is a space of inputs, Y is the space of outputs.

2. **Model or hypothesis:** $h : X \to Y$ that we use to predict outputs given the inputs x.
3. **Loss function:** $L : Y \times Y \to \mathbb{R}, L(...) \geq 0, L(y, y')$ is the loss incurred when predicting y' when y is true.
4. **Optimization** procedure to find the hypothesis h that minimize the loss on the training sample.

1.3 Classification (Task 1/3)

Problem: partitioning the data into pre-defined classes by a *decision boundary* or *decision surface.*

Multi-class classification: more than two classes

- **Multi-label Classification:** An example can belong to multiple classes at the same time
- **Extreme classification:** Learning with thousands to hundreds of thousands of classes (Prof. Rohit Babbar @ Aalto)

1.3.1 Version space

Version space: the set of all consistent hypotheses of the hypothesis class

- **Consistent hypothesis:** if correctly classifies all training examples
- **In version space:**
  - **Most general hypothesis** G**:** cannot be expanded without including negative training examples
  - **Most specific hypothesis** S**:** cannot be made smaller without excluding positive training points

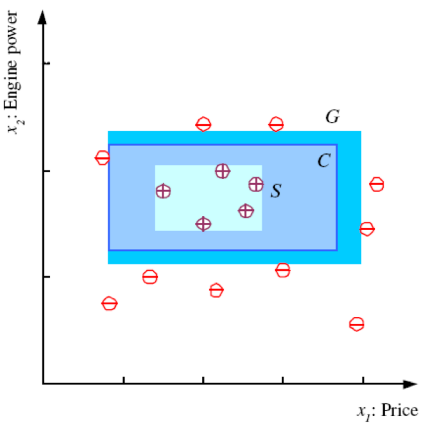

Figure 2: Illustration of a Version Space.

- Intuitively, the **"safest" hypothesis** to choose from the version space is the one that is furthers from the positive and negative training examples → maximum margin
  - Margin = minimum distance between the decision boundary and a training point

1.4 Regression (Task 2/3)

Problem: output variables which are numeric.

1.5 Ranking & preference learning (Task 3/3)

Problem: predict a ordered list of preferred objects.

Training data (typically): pairwise preferences.

- e.g. user x prefers movie y_i over movie y_j

Output: ranked list of elements.

1.6 Generalization

Aim: predict as well as possible the outputs of future examples, not only for training sample.

We would like to *minimize* the **generalization error**, or the **(true) risk**:

$$R(h) = E_{(x,y)\sim D}[L(h(x), y)] \tag{1}$$

Where:

D : Unknown distribution where from training and future examples are drawn from (i.i.d assumption)

What can we say about $R(h)$ based on training examples and the hypothesis class $\mathcal{H}$ alone? Two possibilities:

- Empirical evaluation by testing (Section 1.6.1)
- Statistical learning theory (Section 2)

1.6.1 Model evaluation by testing

What: estimate the model's ability to generalize on future data

How: approximating true risk by computing the empirical risk on a independent test sample:

$$R_{test}(h) = \sum_{(x_i,y_i)\in S_{test}}^{m} L(h(x_i), y_i)$$

- The expectation of $R_{test}(h)$ is the true risk $R(h)$

1.7 Hypothesis classes

There is a huge number of different hypothesis classes or model families in machine learning, **e.g:**

- **Linear models** such as logistic regression and perceptron
- **Neural networks:** compute non-linear input-output mappings through a network of simple computation units
- **Kernel methods:** implicitly compute non-linear mappings into high-dimensional feature spaces (e.g. SVMs)
- **Ensemble methods:** combine simpler models into powerful combined models (e.g. Random Forests)

Each have their different pros and cons in different dimensions (accuracy, efficiency, interpretability); No single best hypothesis class exists that would be superior to all others in all circumstances

---

2 Statistical Learning Theory

What: Statistical learning theory focuses in analyzing the generalization ability of learning algorithms. It's the theoretical background on machine learning.

Goal: Generalization (Section 1.6)

2.1 Probably Approximately Correct (PAC) learning

What: The most studied *theoretical framework* for analyzing the generalization performance of machine learning algorithms. It formalizes the notion of generalization in machine learning. In practice, it's asking for bounding the generaliation error (ϵ) with high probability ($1 - \delta$), with arbitrary level of error $\epsilon > 0$ and confidence $\delta > 0$.

Ingredients:

- **input space** X containing all possible **inputs** x
- set of possible **labels** Y (in binary classification $Y = \{0, 1\}$)
- **concept class** $\mathcal{C}$ contains **concepts** $C : X \to Y$ (to be learned), concept C gives a label $C(x)$ for each input x
  - underlying ground truth $\to$ to be learn in the ideal case
  - unknown in practice (or otherwise ML is not needed)
- Unknown (i.i.d) **probability distribution** D for the data
- **training sample** $S = (x_1, C(x_1)), ..., (x_m, C(x_m))$ drawn independently from D
- **hypothesis class** $\mathcal{H}$
  - in the **basic case** $\mathcal{H} = \mathcal{C}$ but this assumption can be relaxed

Goal: to learn a hypothesis with a low generalization error. For 0/1 loss:

$$R(h) = E_{x \sim D}[L_{0/1}(h(x), C(x))] = Pr_{x \sim D}(h(x) \neq C(x))$$

2.1.1 PAC learnability

What: Question, can we learn a concept class.

PAC-learnable concept class $\mathcal{C}$ if:

- if there exist an **algorithm** $\mathcal{A}$ that given a training sample S outputs a hypothesis $h_S \in \mathcal{H}$ that has generalization error satisfying

$$Pr(\underbrace{R(\overbrace{h_S}^{\text{chosen hypothesis from } \mathcal{H}}) \leq \epsilon}_{\text{"low generalization error" event}}) \geq \underbrace{1 - \delta}_{\text{success rate}} \tag{2}$$

$$P(GeneralizationError\ of\ h_S \leq GeneralizationError\ of\ interest) \geq Desired\ SuccessRate$$
$$P(selection\ hypothesis\ (from\ \mathcal{H})\ with\ GeneralizationError \leq \epsilon) \geq Desired\ SuccessRate$$
$$Probability\ of\ low\ GeneralizationError \geq Desired\ SuccessRate$$

Where:

h_S**:** output hypothesis

S**:** training sample

$m = |S|$**:** sample size that grows polynomially in $1/\epsilon$, $1/\delta$

ϵ**:** generalization error of interest (arbitrary)

$1 - \delta$**:** desired success rate / confidence (arbitrary)

Efficiently PAC-learnable concept class $\mathcal{C}$ if:

- **PAC-learnable**
- $\mathcal{A}$ runs in time polynomial in m, $1/\epsilon$, and $1/\delta$
  - We want the requirement for training data and running time **not to explode** when we make ϵ and δ stricter $\rightarrow$ requirement of polynomial growth

Interpretation:

- ϵ: sets the level of generalization error that is of interest to us
  - e.g. say we are satisfied with predicting incorrectly 10% of the new datapoints $\rightarrow \epsilon = 0.10$
- $1 - \delta$: sets a level of confidence that is of interest to us
  - e.g. say we are satisfied of the training algorithm to fail 5% of the time to provide a good hypothesis $\rightarrow \delta = 0.05$
- $\{R(h_S) \leq \epsilon\}$: The event "low generalization error"
  - considered as a *random variable* because we cannot know beforehand which hypothesis $h_S \in \mathcal{H}$ will be selected by the algorithm

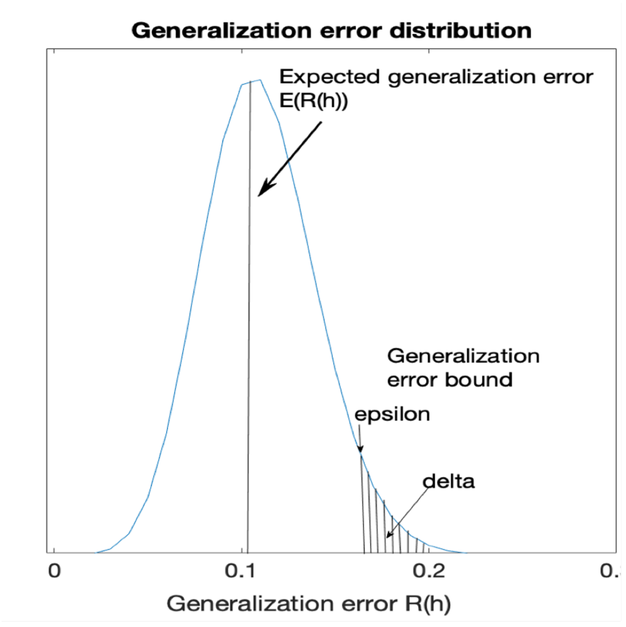

Figure 3: Generalization error distribution and error bound

Generalization error bound vs. expected test error:

- We bound the probability of being in the high error tail of the distribution (not the convergence to the mean or median generalization error) $\rightarrow$ thus **empirically estimated test errors** might be considerably lower than the bounds suggest.

2.2 Learning with finite hypothesis classes

What: Finite concept classes arise when:

- **Input variables** have finite domains or they are converted to such in preprocessing (e.g. discretizing real values)
- The representations of the hypotheses have finite size (e.g. the number of times a single variable can appear)

Finite hypothesis class - consistent case

- **Sample complexity bound** relying on the size of the hypothesis class (Mohri et al, 2018): $Pr(R(h_S) \leq \epsilon) \geq 1 - \delta$ if

$$m \geq \frac{1}{\epsilon}(log(|\mathcal{H}|) + log(\frac{1}{\delta}))$$

- An equivalent **generalization error bound**:

$$R(h) \leq \frac{1}{m}(log(|\mathcal{H}|) + log(\frac{1}{\delta}))$$

- **Holds for** any finite hypothesis class **assuming there is a** consistent hypothesis, one with zero empirical risk or close to it (relaxed).
- The more hypotheses there are in $\mathcal{H}$ $\rightarrow$ the more training examples are needed

2.2.1 Example: Boolean conjunctions

What: Example of a finity hypothesis class.

Example hypothesis class: Boolean conjunctions

- Dealing with subclasses of Boolean formulae, expressions **binary input variables** (literals) combined with **logical operators** AND & NOT.

Example case: Aldo likes to do sport only when the weather is suitable

- **Training data:** Also has given examples of suitable and not suitable weather

	x^t						$r(x^t)$
t	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	1
2	Sunny	Warm	High	Strong	Warm	Same	1
3	Rainy	Cold	High	Strong	Warm	Change	0
4	Sunny	Warm	High	Strong	Cool	Change	1

Table: Aldo's observed sport experiences in different weather conditions.

- **Model:** Let us build a classifier (boolean formulae containing AND, and NOT, but not OR operators) for Aldo to decide whether to do sports today
  - e.g. if (Sky=Sunny) AND NOT (Wind=Strong) then (EnjoySport=1)
- **Number of hypotheses $|\mathcal{H}|$:**
  - **Each variable:** "AND", "AND NOT", or can be excluded from the rule $\rightarrow$ 3 possibilities
  - **Total number of hypotheses** is thus 3^d, where d is the number of variables $\rightarrow |\mathcal{H}| = 3^6 = \underline{729}$
- **Plotting the bound for Aldo's problem using boolean conjunctions:**
  - **Left plot:** generalization bound ϵ is shown for different values of delta δ, using d = 6 variables.

– **Right plot:** generalization bound ϵ is shown for increasing number of input variables d, using delta $\delta = 0.05$

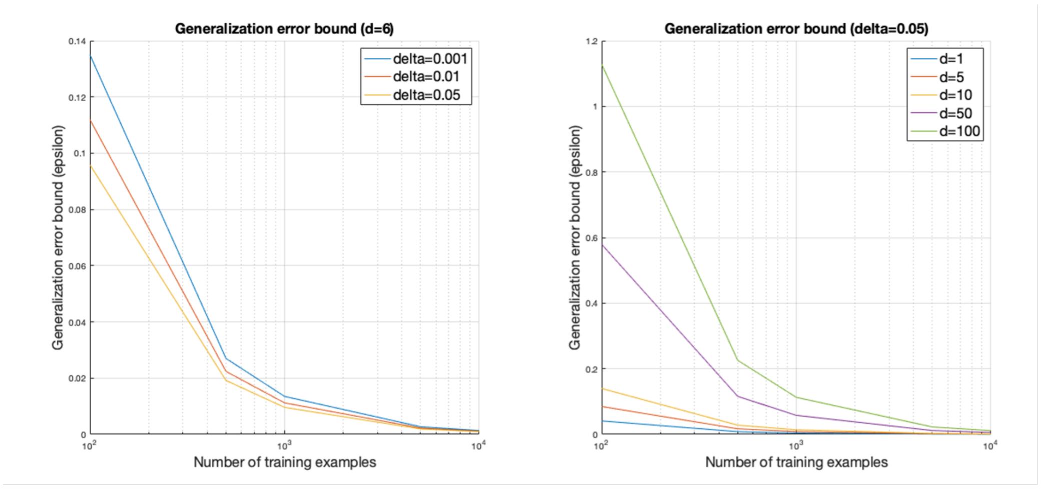

Figure 4: Boolean conjunctions: Generalization error bounds

- **Typical behaviour of ML learning algorithms is revealed:**
  - **increase of sample size decreases generalization error**
  - extra data gives **less and less additional benefit as the sample size grows** (law of diminishing returns)
  - requiring high level of confidence (small δ) for obtaining low error requires more data for the same level of error

2.3 Learning with infinite hypothesis classes

Most models used in practise rely on infinite hypothesis classes, e.g.

- $\mathcal{H}$ = hyperplanes in $\mathbb{R}^d$ (e.g. Support vector machines)
- $\mathcal{H}$ = neural networks with continuous input variables

2.3.1 Vapnik-Chervonenkis (VC) dimension

Purpose: Vapnik-Chervonenkis dimension lets us **study** learnability of infinite hypothesis classes **through** the concept of shattering

What: can be understood as measuring the capacity of a hypothesis class to adapt to different concepts

- $VCdim(\mathcal{H})$ = size of the **largest training set** that we can find a **consistent classifier for all labelings in** Y^m
- Intuitively:
  - low $VCdim \rightarrow$ easy to learn, low sample complexity
  - high $VCdim \rightarrow$ hard to learn, high sample complexity
  - infinite $VCdim \rightarrow$ cannot learn in PAC framework

How to show that $VCdim(\mathcal{H}) = d$ for a hypothesis class:

- We need to show two facts:
  1. There exists a set of inputs of size d that **can be shattered** by hypothesis in $\mathcal{H}$ (i.e. we can pick the set of inputs any way we like): $VCdim(\mathcal{H}) \geq d$
  2. There does not exist any set of inputs of size $d + 1$ that **can be shattered** (i.e. need to show a general property): $VCdim(\mathcal{H}) < d + 1$

Formally: (for binary labelings)

- Through **growth function**:

$$\Pi_{\mathcal{H}}(m) = \max_{\{x_1,\dots,x_m\}\subset X} |\{(h(x_1), ..., h(x_m)) : h \in \mathcal{H}\}|$$

  - The growth function gives the maximum number of unique labelings the hypothesis class $\mathcal{H}$ can provide for an arbitrary set of input points

  - The maximum of the growth function is 2^m for a set of m examples

- **Vapnik-Chervonenkis dimension** is then

$$VCdim(\mathcal{H}) = \max_{m}\{m| \sqcap_{\mathcal{H}}(m) = 2^m\}$$

2.3.1.1 Shattering

What: underlying concept in VC dimension

Given: a set of points $S = x_1, ..., x_m$ and a fixed class of functions $\mathcal{H}$

$\mathcal{H}$ is said to shatter S if: for any possible partition of S into positive S_+ and negative subset S_- we can find a hypothesis for which $h(x) = 1$ if and only if $x \in S_+$

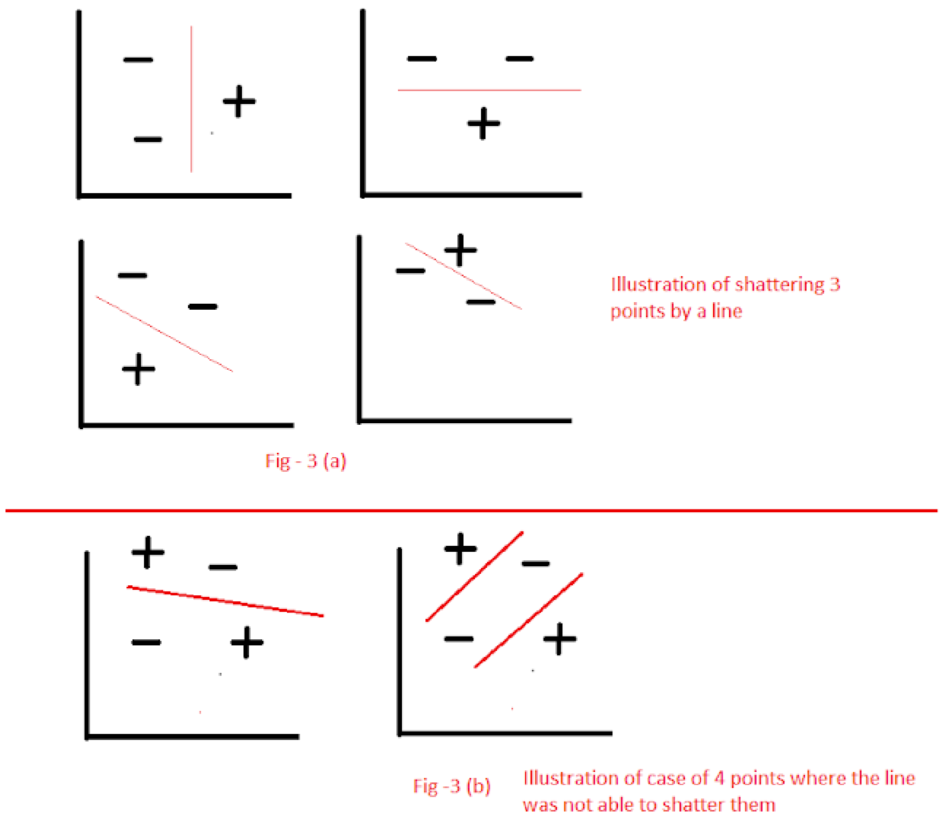
Illustration of shattering 3 points by a line
Fig - 3 (a)
Fig -3 (b)
Illustration of case of 4 points where the line was not able to shatter them

Figure 5: Illustration of shattering

2.3.1.2 Generalization bound with VC-dimension

What: Way of using VC-dimensions to analyze machine learning algorithms.

(Mohri, 2018) Let $\mathcal{H}$ be a family of functions taking values in $-1, +1$ with VC-dimension d . Then for any $\delta > 0$, with probability at least $1 - \delta$ the following holds for all $h \in \mathcal{H}$:

$$R(h) = \underbrace{\hat{R}(h)}_{\text{empirical risk}} + \sqrt{\frac{2\log(em/d)}{m/d}} + \sqrt{\frac{\log(1/\delta)}{2m}}$$

Where:
d**:** VC-dimension
m**:** amount of training data
e**:** Euler's number, $e \approx 2.71828$
δ**:** failure rate

- m/d shows that changing to a model with higher VC-dimension d makes the same amount of training samples m less effective. $\rightarrow$ the second term grows when VC-dimension grows.
  - For more complex $\mathcal{H}$ class, one needs more data m to guarantee similar kind of generalization.
- Manifestation of the **Occam's razor principle:** to justify an increase in the complexity, we need reciprocally more data

2.3.2 Rademacher complexity

What: Rademacher complexity defines complexity as the capacity of hypothesis class to fit random noise

Why: Rademacher complexity is a practical alternative to VC dimension, giving typically sharper bounds (but requires a lot of simulations to be run).

Experiment: how well does your hypothesis class fit noise?

- Consider a **set of training examples** $S_0 = \{(x_i, y_i)\}_{i=1}^m$
- **Generate** M **new datasets** $S_1, ..., S_M$ from S_0 by randomly drawing a new label $\sigma \in Y$ for each training example in S_0

$$S_k = \{(x_i, \sigma_{ik})\}_{i=1}^m$$

- Train a classifier h_k minimizing the empirical risk on training set S_k, record its empirical risk

$$\hat{R}(h_k) = \frac{1}{m} \sum_{i=1}^{m} 1_{h_k(x_i) \neq \sigma_{ik}}$$

- **Compute the average empirical risk over all datasets:**

$$\bar{\epsilon} = \frac{1}{M} \sum_{k=1}^{M} \hat{R}(h_k)$$

- **Observe the quantity:**

$$\hat{\mathcal{R}} = \frac{1}{2} - \bar{\epsilon}$$

  - When $(\hat{\mathcal{R}} = 0, \bar{\epsilon} = 0.5) \rightarrow$ predictions correspond to random coin flips (0.5 probability to predict either class)
  - When $(\hat{\mathcal{R}} = 0.5, \bar{\epsilon} = 0) \rightarrow$ all hypotheses $h_i, i = 1, ..., M$ have zero empirical error (perfect fit to noise, not good!)
  - **Ideally we want:**
    - to be able to **separate noise from signal**
      - Meaning large $\bar{\epsilon}$, so that the model is bad at classifying random noise. $\rightarrow$ low complexity $\hat{\mathcal{R}}$.
    - to have **low empirical error on real data** - otherwise impossible to obtain low generalization error

Rademacher complexity:

- For binary classification with labels $Y = \{-1, +1\}$ **empirical Rademacher complexity** can be defined as

$$\hat{\mathcal{R}}_S(\mathcal{H}) = \frac{1}{2} E_\sigma (\underbrace{\sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{t=1}^{m} \overbrace{\sigma^i}^{\text{random true label}} \overbrace{h(x_i)}^{\text{predicted random label}}}_{\text{hypothesis with highest correlation with random label}})$$

Where:
$\sigma_i \in \{-1, +1\}$**:** are Rademacher random variables,
drawn independently from uniform distribution
(i.e. $Pr\{\sigma = 1\} = 0.5$)

- We can also rewrite $\hat{\mathcal{R}}_S$ **in terms of empirical error**

$$\hat{\mathcal{R}}_S = \frac{1}{2} - E_\sigma \inf_{h \in \mathcal{H}} \hat{\epsilon}(h)$$

  - Now we have Rademacher complexity in terms of expected minimum error of classifying randomly labeled data

2.3.2.1 Generalization bound with Rademacher complexity

(Mohri et al. 2018): For any $\delta > 0$, with probability at least $1 - \delta$ over a sample drawn from an unknown distribution D, for any $h \in \mathcal{H}$ we have:

$$R(h) \leq \underbrace{\hat{R}_S(h)}_{\text{empirical risk}} + \underbrace{\hat{\mathcal{R}}_S(\mathcal{H})}_{\text{empirical Rademacher complexity}} + 3\sqrt{\frac{\log \frac{2}{\delta}}{2m}}$$

2.3.3 Vapnik-Chervonenkis dimension VS. Rademacher complexity

Note the differences between Rademacher complexity and VC dimension

- Dependency on training data

- **VC dimension:** independent $\rightarrow$ measures the worst-case where the data is generated in a bad way for the learner
- **Rademacher complexity:** depends on the training sample thus is dependent on the data generating distribution

- Focus
  - **VC dimension:** extreme case of realizing all labelings of the data
  - **Rademacher complexity:** measures smoothly the ability to realize random labelings

Example: Rademacher and VC bounds on a real dataset

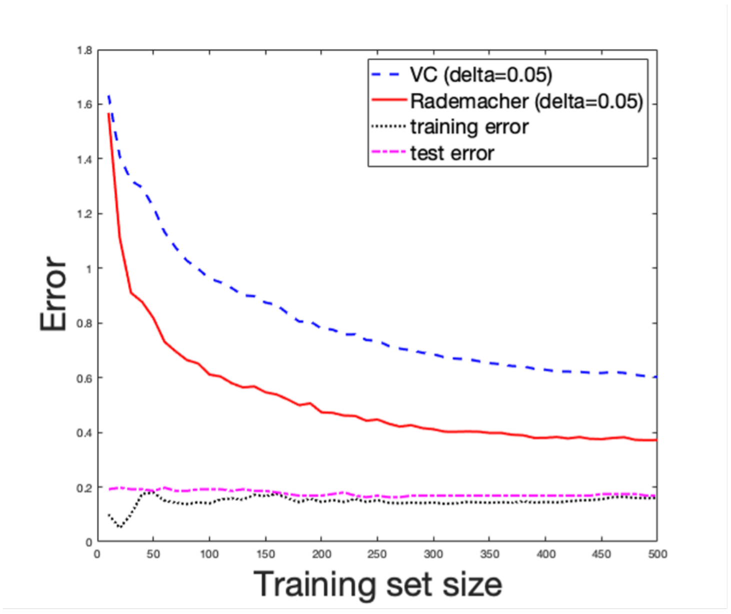

Figure 6: Rademacher and VC bounds on a real dataset

- Rademacher bound is sharper than the VC bound
- VC bound is not yet informative with 500 examples (> 0.5) using ($\delta = 0.05$)
- The gap between the mean of the error distribution ($\approx$ test error) and the 0.05 probability tail (VC and Rademacher bounds) is evident (and expected)