STAT 535

10/11/22

Lecture 5

Basic concepts
(Neural networks) ⟶ after this chapter

HW 1 posted
Q 1 - next Tue at 12:30
TB Posted - Kernel Regression Slides

Lecture II: Prediction – Basic concepts

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

October, 2022

Parametric vs non-parametric

Generative and discriminative models for classification

- Generative classifiers
- Discriminative classifiers
- Generative vs discriminative classifiers

Loss functions

- Bayes loss

Variance, bias and complexity

Reading HTF Ch.: 2.1–5,2.9, 7.1–4 bias-variance tradeoff, Murphy Ch.: 1., 8.6[1], Bach Ch.:

[1] Neither textbook is close to these notes except in a few places; take them as alternative perspectives or related reading

The "learning" problem

- **Given**
- a problem (e.g. recognize digits from $m \times m$ gray-scale images)
- a **sample** or (**training set**) of **labeled data**

$$\mathcal{D} = \{(x^1, y^1), (x^2, y^2), \ldots (x^n, y^n)\}$$

drawn i.i.d. from an unknown P_{XY}

- **model class** $\mathcal{F} = \{f\}$ = set of predictors to choose from

logistic regression
CART
linear regression
...

- **Wanted**
- a predictor $f \in \mathcal{F}$ that performs well on future samples from the same P_{XY}

  - "choose a predictor $f \in \mathcal{F}$" = training/learning
  - "performs well on future samples" (i.e. f **generalizes** well) – how do we measure this? how can we "guarantee" it?
  - choosing $\mathcal{F}$ is the **model selection problem** – about this later

A zoo of predictors

- Linear regression
- Logistic regression
- Linear Discriminant (LDA)
- Quadratic Discriminant (QDA)
- CART (Decision Trees)
- K-Nearest Neighbors
- Nadaraya-Watson (Kernel regression)
- Naive Bayes
- Neural networks/Deep learning
- Support Vector Machines
- Monotonic Regression

Parametric vs. non-parametric models

▶ CART with n leaves
↳ sample size

Example (Parametric and non-parametric predictors)

Parametric

- Linear, logistic regression $\beta \in \mathbb{R}^d$ $\beta_0 \in \mathbb{R}$
- Linear Discriminant Analysis (LDA)
- Neural networks
- Naive Bayes
- CART with L levels

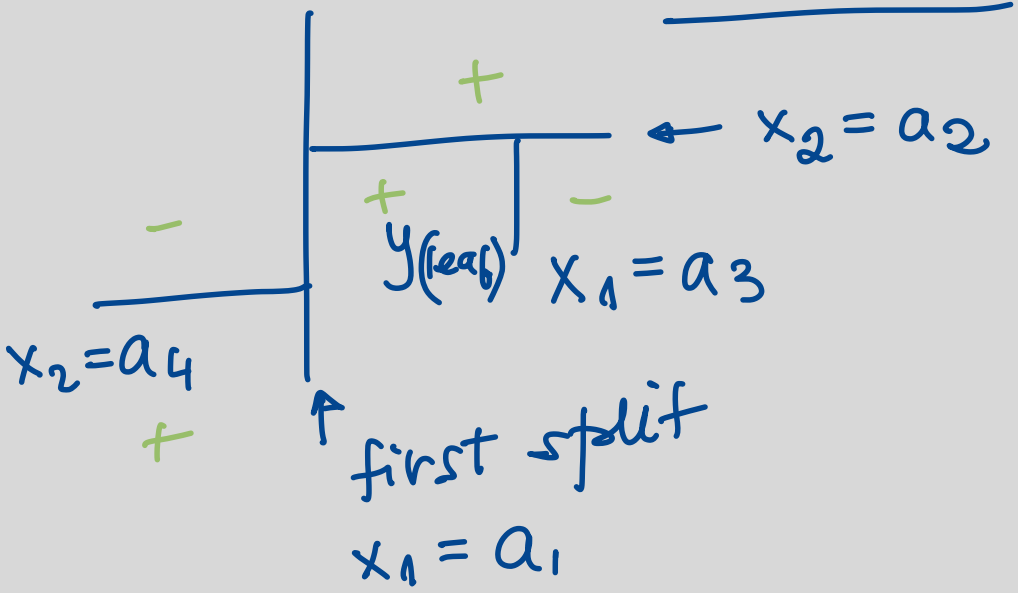
L leaves $\Leftrightarrow$ $L-1$ splits

Non-parametric

- Nearest-neighbor classifiers and regressors
- Nadaraya-Watson predictors
- Monotonic regression ← shape constrained estimation
- (Support Vector Machines)

Exercise Are Radial Basis Functions classifiers parametric or non-parametric?

Decision Tree



parameters:

#split:	coordinate, what leaf?	value, y
1	1, leaf	a_1, ..
2	2, ..	a_2, ..
3	1, ...	a_3, ..
⋮		
L	..	...

A mathematical definition

- A model class $\mathcal{F}$ is **parametric** if it is finite-dimensional, otherwise it is **non-parametric**

In other words

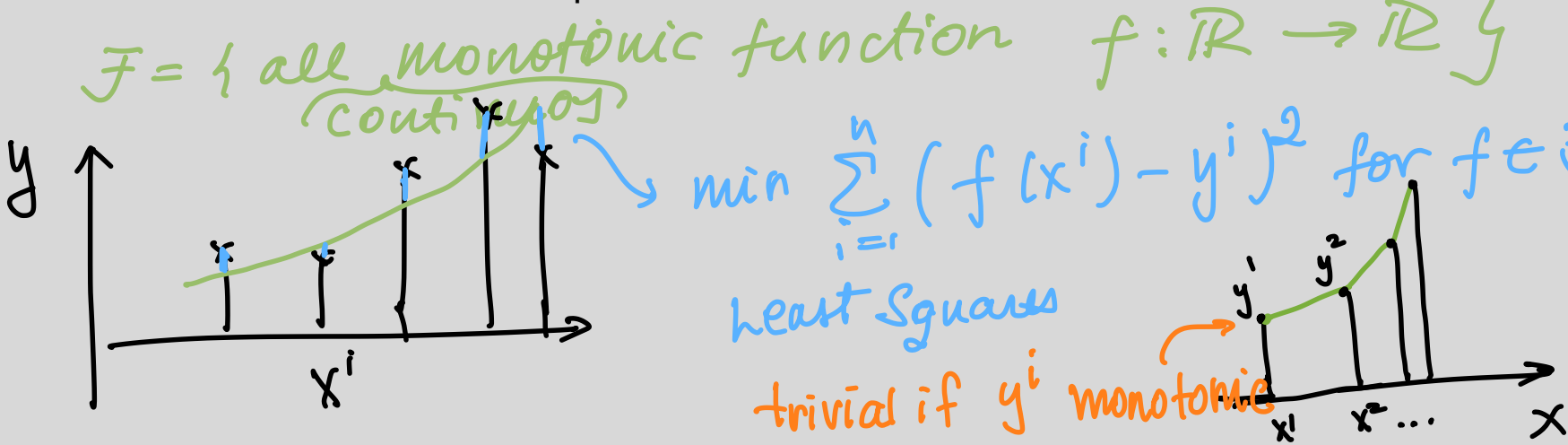
- When we estimate a parametric model from data, there is a fixed number of parameters, (you can think of them as one for each dimension, although this is not always true), that we need to estimate to obtain an estimate $\hat{f} \in \mathcal{F}$.
- The parameters are meaningful.
  E.g. the β_j in logistic regression has a precise meaning: the component of the normal to the decision boundary along coordinate j.
- The dimension of β does not change if the sample size N increases.

$= \dim \mathcal{F}$

- $\mathcal{F}$ infinite-dimensional iff there is no finite D so that $\mathcal{F} \cong \mathbb{R}^D$
  ↑ bijection

Non-parametric models – Some intuition

- When the model is non-parametric, the model class $\mathcal{F}$ is a function space.
- The $\hat{f}$ that we estimate will depend on some numerical values (and we could call them parameters), but these values have little meaning taken individually.
- The number of values needed to describe $\hat{f}$ generally grows with n.
  Examples In the Nearest neighbor and kernel predictors, we have to store all the data points, thus the number of values describing the predictor f grows (linearly) with the sample size. Exercise Does the number of values describing f always grow linearly with the sample size? Does it have to always grow to infinity? Does it have to always grow in the same way for a given $\mathcal{F}$?
- Non-parametric models often have a **smoothness parameter**.
  Examples of smoothness parameters K in K-nearest neighbor, h the kernel bandwidth in kernel regression.
  To make matters worse, a smoothness parameter is not a parameter! More precisely it is not a parameter of an $f \in \mathcal{F}$, because it is not estimated from the data, but a descriptor of the model class $\mathcal{F}$.
- We will return to smoothness parameters later in this lecture.



Generative classifiers

classifier < generative / discriminative

One way to define a classifier is to assume that each class is generated by a distribution $g_y(X) = P(X|Y = y)$. If we know the distributions g_y and the class probabilities $P(Y = y)$, we can derive the *posterior probability* distribution of Y for a given x. This is

$$P(Y = y|X) = \frac{P(Y = y)g_y(X)}{\sum_{y'} P(Y = y')g_{y'}(X)} = \frac{P(Y = y)g_y(X)}{P(X)} \quad (1)$$

The "best guess" for $Y(X)$ (i.e. the decision rule) is

$$f(X) = \text{argmax}_y P(Y = y|x) = \text{argmax}_y P(Y = y)g_y(x) \quad (2)$$

- (1) amounts to a likelihood ratio test for Y.
- The functions $g_y(x)$ are known as **generative models** for the classes y. Therefore, the resulting classifier is called a **generative classifier**. Examples: LDA, QDA, Naive Bayes.
- In contrast, a classifier defined directly in terms of $f(x)$ (or $P_{Y|X}$), like the linear, quadratic, decision tree is called a **discriminative classifier**.
- In practice, we may not know the functions $g_y(x)$, in which case we estimate them from the sample $\mathcal{D}$.

Generative classifiers $y \in \{\pm 1\}$

x : want $P_{y|x=x}$

$y=+1$ $P_{x|y=+1}$ distribution of examples from class $+1$

$y=-1$ $P_{x|y=-1}$ — " — -1

Bayes' Rule

$$P_{y=+1|x} = \frac{P_y[+1]\, P_{x|y=+1}(x)}{P_y[-1]\cdot P_{x|y=-1}(x) + P_y[+1]\, P_{x|y=+1}(x)}$$

step 3 of Algorithm

$$P_{y=-1|x} = \frac{P_y[-1]\cdot P_{x|y=-1}(x)}{+} = 1 - P_{y=+1|x=x}$$

$$f(x) = P_{y=+1|x=x} - \frac{1}{2}$$

$$\hat{y}(x) = \operatorname{sgn} f(x)$$

$$f(x) > 0 \iff \frac{P_{y=+1|x}}{P_{y=-1|x}} > 1$$

$$\frac{P_{x|y=+}}{P_{x|y=-}} > \frac{P_{y=-}}{P_{y=+}}$$

likelihood ratio — independent of x

Algorithm

1. choose model class for $P_{x|y=\pm 1}$
2. Estimate $P_{x|y=\pm 1}$ (e.g. by Max Likelihood)

   ↑ generative models for x given y
3. Estimate $P_{y=1} = \frac{n_1}{n}$

   n_1 = # examples with $y^i = +1$

Generative classifier and the likelihood ratio

$$P(Y=y|X) = \frac{P(Y=y)g_y(X)}{\sum_{y'} P(Y=y')g_{y'}(X)} = \frac{P(Y=y)g_y(X)}{P(X)}$$

$$f(x) = \text{argmax}_y P(Y=y|x) = \text{argmax}_y g_y(x)P(Y=y)$$

Likelihood Ratio test (for $y \in \{\pm 1\}$)

$$\frac{g_+(x)P(Y=+)}{g_-(x)P(Y=-)}$$

Example (Fisher's LDA in one dimension)

Assume $Y = \pm 1$, $g_y(x) = N(x, \pm\mu, \sigma^2 I)$, i.e each class is generated by a Normal distribution with the same spherical covariance matrix, but with a different mean. Let $P(Y = 1) = p \in (0, 1)$. Then, the posterior probability of Y is

$$P(Y = 1|x) \propto pe^{-||x-\mu||^2/(2\sigma^2)} \quad P(Y = -1|x) \propto (1-p)e^{-||x+\mu||^2/(2\sigma^2)} \tag{3}$$

and $f(x) = 1$ iff $\ln P(Y = 1|x)/P(Y = -1|x) \geq 0$, i.e iff

$$\ln \frac{p}{1-p} - \frac{1}{2\sigma^2}[||x^2|| - 2\mu^T x + ||\mu||^2 - ||x^2|| - (2\mu)^T x - ||\mu||^2] = \left(\frac{2\mu}{\sigma^2}\right)^T x + \ln \frac{p}{1-p} \geq 0 \tag{4}$$

Hence, the classifier $f(x)$ turns out to be a linear classifier. The decision boundary is perpendicular to the segment connecting the centers $\mu, -\mu$. This classifier is known as **Fisher's Linear Discriminant**. [**Exercises** Show that if the generative models are normal with different variances, then we obtain a quadratic classifier. What happens if the models g_y have the same variance, but it is a full covariance matrix Σ?]

Discriminative classifiers — NOT GENERATIVE

- Defined directly in terms of $f(x)$ or (almost) equivalently, in terms of the decision boundary $\{f(x) = 0\}$
- Can be classified by the shape of the decision boundary (if it's simple)
  - linear, polygonal, quadratic, cubic,...

The ambiguity of "linear classifier" ← can be either generative or discriminative

Does it mean $f(x) = \beta^T x$ OR $\{f(x) = 0\}$ is a hyperplane ?
If we talk about **classification** and the domain of x is $\mathbb{R}^d$, then "linear" refers to decision boundary. Otherwise it refers to the expression of $f(x)$. Exercise Find examples when the two definitions are not equivalent

- Can be grouped by model class (obviously)
  - Neural network, K-nearest neighbor, decision tree, ...
    Exercise Is logistic regression a generative or discriminative classifier?
- By method of training (together with model class) ←
  - For example, PERCEPTRON algorithm, Logistic Regression, (Linear) Support Vector Machine (see later), Decision Tree with 1 level are all linear classifiers, but usually produce different decision boundaries give a $\mathcal{D}$

A comparison of generative and discriminative classifiers

Advantages of generative classifiers

- Generative classifiers are statistically motivated
- Generative classifiers are *asymptotically optimal*

Theorem

If $Y \in \{\pm 1\}$, the model class G_y in which we are estimating g_y contains the true distributions $P(X|Y=y)$ for every y, and $g_y = P(X|Y), P(Y=y)$ are estimated by Maximum Likelihood then the expected loss[2] of the generative classifier f_g given by (2) tends to the Bayes loss when $n \to \infty$, i.e $\lim_{N\to\infty} L_{01}(f_g) \leq \min_{f\in\mathcal{F}} L_{01}(f)$. Here $\mathcal{F}$ is the class of likelihood ratio classifiers obtainable from g_y's in $\mathcal{G}_y$.

- The log-likelihood ratio $\ln \frac{P(Y=1|x)}{P(Y=-1|x)}$ is a natural confidence measure for the label at $f_g(x)$. The further away from 0 the likelihood ratio, the higher the confidence that the chosen y is correct.
- Generative classifiers extend naturally to more than two classes. If a new class appears, or the class distribution $P(Y)$ changes, updating the classifier is simple and computationally efficient.
- Often it is easier to pick a (parametric) model class for g_y than an f directly. Generative models are generally more intuitive, while often representing/visualizing decision boundaries between more than two classes is tedious.

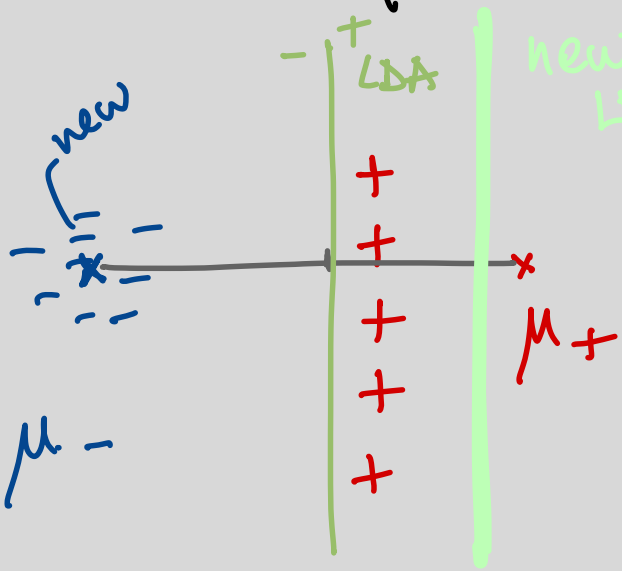
[2] Loss, Bayes loss, $L01$ are defined in the next section.

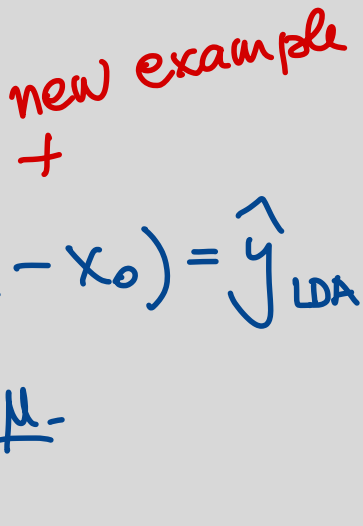
Advantages of discriminative classifiers

- Generative models offer no guarantees if the true g_y aren't in the chosen model class, whereas for many classes of discriminative models there are guarantees. ✔
- Many discriminative models have performance guarantees for any sample size n, while generative models are only guaranteed for large enough n ✔
- Discriminative classifiers offer many more choices (but one must know how to pick the right model) ✔
- Generative models do not use data optimally in the non-asymptotic regime (when $n \ll \infty$). This has been confirmed practically many times, as discriminative classifiers have been very successful for limited sample sizes

Exercise LDA vs Logistic regression: Experiment with LDA vs LR when data comes from 2 Normal distributions, with outliers. What outliers affect which method more? Experiment also on a toy data set like the one in the lecture notes.

Ex: LDA (fit $N(\mu_\pm, \sigma^2 I_d)$ to each class)





Loss functions

The **loss function** represents the cost of error in a prediction problem. We denote it by L, where

$$L(y, \hat{y}) = \text{the cost of predicting } \hat{y} \text{ when the actual outcome is } y$$

Note that sometimes the loss depends on x directly. Then we would write it as $L(y, \hat{y}, x)$.
As usually $\hat{y} = f(x)$ or $\text{sgn} f(x)$, we will typically abuse notation and write $L(y, f(x))$.

Loss for classification

$$L_{01}(y, \hat{y}) = \begin{cases} 1 & y \neq \hat{y} \quad \text{mistake} \\ 0 & y = \hat{y} \quad \text{correct} \end{cases}$$

$L_{\text{imbalanced}} =$

$\hat{y}$ \ y	$+$	$-$
$+$	0	L_{+-} ← False positive
$-$	L_{-+}	0

False negative

$L_{+-} \ll L_{-+}$

false alarm — lack of detection

Loss functions

The **loss function** represents the cost of error in a prediction problem. We denote it by L, where

$$L(y, \hat{y}) = \text{the cost of predicting } \hat{y} \text{ when the actual outcome is } y$$

Note that sometimes the loss depends on x directly. Then we would write it as $L(y, \hat{y}, x)$.
As usually $\hat{y} = f(x)$ or $\text{sgn} f(x)$, we will typically abuse notation and write $L(y, f(x))$.

$y \in \{1, 2, \ldots m\}$

Loss for multiway classification

Loss Matrix

$L =$

$\hat{y}$ \ y	1	2	..	m
1	0			
2		0		$L_{kk'}$
:			.	
m				0

$L_{kk'}$ = Loss when k predicted but truth is y=k'

Loss functions for classification

For classification, a natural loss function is the **misclassification error** (also called **0-1 loss**)

$$L_{01}(y, f(x)) = 1_{[y \neq f(x)]} = \begin{cases} 1 & \text{if } y \neq f(x) \\ 0 & \text{if } y = f(x) \end{cases} \quad (6)$$

Sometimes different errors have different costs. For instance, classifying a HIV+ patient as negative (**a false negative** error) incurs a much higher cost than classifying a normal patient as HIV+ (**false positive** error). This is expressed by **asymmetric misclassification costs**. For instance, assume that a false positive has cost one and a false negative has cost 100. We can express this in the matrix

$f(x):$	$+$	$-$
true $:+$	0	100
$-$	1	0

In general, when there are p classes, the matrix $L = [L_{kl}]$ defines the loss, with L_{kl} being the cost of misclassifying as l an example whose true class is k.