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

- **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

Example (Parametric and non-parametric predictors)

Parametric

- Linear, logistic regression
- Linear Discriminant Analysis (LDA)
- Neural networks
- Naive Bayes
- CART with L levels

Non-parametric

- Nearest-neighbor classifiers and regressors
- Nataraya-Watson predictors
- Monotonic regression
- (Support Vector Machines)

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

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.

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

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

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

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

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

- (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 classifier and the likelihood ratio

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

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

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

- 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"

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.

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 $\mathrm{sgn} f(x)$, we will typically abuse notation and write $L(y, f(x))$.

Least Squares (LS) loss

The **Least Squares (LS)** (or **quadratic**) loss function is given by

$$L_{LS}(y, f(x)) = (y - f(x))^2 \tag{5}$$

This loss is commonly associated with regression problems.
Example: L_{LS} is the log-likelihood of a regression problem (linear or not) with Gaussian noise.

Loss functions for classification

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

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

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.

Expected loss and empirical loss

- Objective of prediction = to minimize expected loss on future data, i.e.

$$\text{minimize}\, L(f) \;=\; E_{P(X,Y)}[L(Y, f(X)]\ \text{over}\ f \in \mathcal{F} \tag{7}$$

We call $L(f)$ above **expected loss**.

Example (Misclassification error $L_{01}(f)$)

$L_{01}(f)$ = probability of making an error on future data.

$$L_{01}(f) \;=\; P[Yf(X) < 0] \;=\; E_{P_{XY}}[1_{[Yf(X)<0]}] \tag{8}$$

Expected loss and empirical loss

- Objective of prediction = to minimize expected loss on future data, i.e.

$$\text{minimize}\, L(f) \;=\; E_{P(X,Y)}[L(Y, f(X)] \text{ over } f \in \mathcal{F} \tag{7}$$

  We call $L(f)$ above **expected loss**.
- $L(f)$ cannot be minimized or even computed directly, because we don't know the data distribution P_{XY}.
  Therefore, in training predictors, one uses the **empirical** data distribution given by the sample $\mathcal{D}$.
- The empirical loss (or **empirical error** or **training error**) is the average loss on $\mathcal{D}$

$$\hat{L}(f) \;=\; \frac{1}{n}\sum_{i=1}^{n} 1_{[y^i f(x^i)<0]} \tag{8}$$

- Finally, the value of the **optimal expected loss** for our model class (this is the loss value we are aiming for) is denoted by $L(\mathcal{F})$.

$$L(\mathcal{F}) \;=\; \min_{f\in\mathcal{F}} E_{P(X,Y)}[L(Y, f(X))] \tag{9}$$

  Note that of all the quantities above, we can only know $\hat{L}(f)$ for a finite number of f's in $\mathcal{F}$.

Bayes loss

- How small can the expected loss $L(f)$ be?
  It is clear that
  $$L(\mathcal{F}) = \min_{f \in \mathcal{F}} L(f) \geq \min_{f} L(f) = L^* \tag{10}$$
  where L^* is taken over all possible functions f that take values in $\mathcal{Y}$.
- L^* is the absolute minimum loss for the given P_{XY} and it is called the **Bayes loss**.
- The Bayes loss is usually not zero

Bayes loss for (binary) classification

- Fix x and assume $P_{Y|X}$ known. Then:
  - Label y will have probability $P_{Y|X}(y|x)$ at this x.
  - No deterministic guess $f(x)$ for y will make the classification error $E_{P_{Y|X=x}}[L_{01}(y, f(x))]$ (unless $P_{Y|X=x}$ is itself deterministic)
  - Best guess minimizes the probability of being wrong. This is achieved by chosing the most probable class
    $$y^*(x) \;=\; \operatorname*{argmax}_y P_{Y|X}(y|x) \tag{11}$$
  - The probability of being wrong if we choose $y^*(x)$ is $1 - p^*(x)$, where $p^*(x) = \max_y P_{Y|X}(y|x)$.
- The **Bayes classifier** is $y^*(x)$ as a function of x and its expected loss is the Bayes loss

$$L^*_{01} \;=\; E_{P_X}[1 - p^*(X)] \;=\; E_{P_X}[1 - \max_y P[Y|X]] \tag{12}$$

This shows that the Bayes loss is a property of the problem, via L and P_{XY}, and not of any model class or learning algorithm.

Example

In a classification problem where the class label depends deterministically of the input, the Bayes loss is 0. For example, classifying between written English and written Japanese has (probably) zero Bayes loss.

Example

Consider the least squares loss and the following data distribution: $P_{Y|X} \sim N(g(X), \sigma^2)$. In other words, the Y values are normally distributed around a deterministic function $g(X)$. In this case, optimal least squares predictor is the mean of Y given X, which is equal to $g(X)$. The Bayes loss is the expected squared error around the mean, which is σ^2. Exercise what is the expression of the Bayes loss if $P_{Y|X} \sim N(g(X), \sigma(X)^2)$?

Exercise What is the Bayes loss if (1) $P(Y|X) \sim N((\beta^*)^T X, \sigma^2 I)$ and the loss is L_{LS}; (2) $P(X|Y = \pm 1) \sim N(\mu_\pm, \sigma^2 I)$ and the loss is L_{01} (for simplicity, assume $X \in \mathbb{R}, \mu_{pm} = \pm 1, \sigma = 1$); (3) give a formula for the Bayes loss if we know $P(X|Y = \pm 1), P(Y)$, $Y \in \{\pm 1\}$ and the loss is L_{01}. (4) Give an example of a situation when the Bayes loss is 0.

Bias and variance: definitions (never to be used again)

Preliminaries

- What we have a data source P_{XY} and a class of predictors $\mathcal{F}$
- From P_{XY} we sample i.i.d. $\mathcal{D}_n$ of size n. Hence $\mathcal{D}_n \sim P^n_{XY}$. []

Bias and Variance as in Intro Stat Theory

- We want to estimate a parameter $\theta \in \Theta \subseteq \mathbb{R}$
- We use $\mathcal{D}_n$ to obtain estimator $\hat{\theta}_{\mathcal{D}_n}$ which is a function of $\mathcal{D}_n$.
- $\mathcal{D}_n$ is random, hence so is $\hat{\theta}_{\mathcal{D}_n}$.
- Bias$= (\hat{\theta}_{\mathcal{D}_n}) = E_{P^n}[\hat{\theta}_{\mathcal{D}_n}] - \theta$
- Variance$= Var_{P^n}(\hat{\theta}_{\mathcal{D}_n})$
  Both Bias and Variance are computed under the distribution from which we sampled $\mathcal{D}_n$, denoted by P^n.

Bias and Variance for us

- We use $\mathcal{D}_n$ to estimate $\hat{f}_n \in \mathcal{F}$

$$\hat{f}_{\mathcal{D}_n} = \operatorname*{argmin}_{f \in \mathcal{F}} \hat{L}(f, \mathcal{D}_n) \tag{15}$$

- $\mathcal{D}_n$ is random, hence so if $\hat{f}_n$.
- Main differences
  1. $\hat{f}$ is a function!
  2. We are interested in the predictions and not the parameters of $\hat{f}$.
- Several proposals to define bias and variance exist.
- Bias and variance are properties of $\mathcal{F}$.

What we need to know in this course is qualitative

Bias as model (mis)fit

The qualitative meaning of bias we will use has to do with the ability of the model class $\mathcal{F}$ to fit the data $\mathcal{D}_n$.

- We measure the misfit by the loss L associated with the task, i.e $\hat{L}(\hat{f}_{\mathcal{D}_n}, \mathcal{D}_n)$
- **Bias**$(\mathcal{F})= E_{P(X,Y)^n}[\hat{L}(\hat{f}_{\mathcal{D}_n}, \mathcal{D}_n)]$ (hence, bias is expected empirical loss).
- Richer model classes have less bias

$$\mathcal{F} \subset \mathcal{F}' \quad \text{then bias}(\mathcal{F}) \geq \text{bias}(\mathcal{F}')$$

- Larger data are harder to fit (hence more bias on average)[3]

[3] Not trivial, to find a reference.

Sampling variance

- Intuition: if we draw two different data sets $\mathcal{D}, \mathcal{D}' \sim P_{XY}$ (from the same distribution) we will obtain different predictors f, f'. Variance measures how different the predictions of f, f' can be on average.
- **Variance** at $x = Var_{P^n_{XY}}(\hat{f}_{\mathcal{D}_n}(x))$, where the randomness is over the sample $\mathcal{D}_n$
- **Variance** associated with predictor class $\mathcal{F}$ is the expectation over P_X of the variance at x, i.e $E_{P_X}[Var_{P^n_{XY}}(\hat{f}_{\mathcal{D}_n}(x))]$

- Variance depends on n, $\mathcal{F}$, and the data distribution P_{XY} Exercise If $P_{Y|X}$ is deterministic for all x, does it mean that the variance is 0?
- Richer model classes are subject to more variance

$$\mathcal{F} \subset \mathcal{F}' \quad \text{then}\, Var(\mathcal{F}) \leq Var(\mathcal{F}') \quad \text{for any}\, f^*$$

Variance, bias and model complexity

- Synonyms: rich class = complex model = flexible model = high modeling power = many degrees of freedom = many parameters

- Evaluating the model complexity[4]/number of free parameters of a model class $\mathcal{F}$ is usually a difficult problem!
  - Non-parametric models # parameters depends on P_{XY}, smoothing parameter and n
  - Parametric models # parameters NOT always equal to the number of parameters of f!

Example the classifier $f(x) = \mathrm{sgn}(\alpha x), x, \alpha \in \mathbb{R}$ depends on one parameter α but has ∞ degrees of freedom[5]!

Example the linear classifier and regressor on $\mathbb{R}^d$ has (no more than) $n+1$ degrees of freedom

Example the complexity of a two layer neural net with m fixed is not known (but there are approximation results); the number of weights in f is obviously $(m+1)(n+1)+1$

Example For K-NN, the variance increases when K decreases

Example For pruned Decision Tree, the variance increases whith the number of levels

- The variance of a predictor increases with the complexity of $\mathcal{F}$.
- But complexity is the opposite of bias, so bias decrease with the complexity of $\mathcal{F}$
- This is known as the **Bias-Variance tradeoff**

---

[4] There are several definitions of model complexity, but this holds for all definitions I know

[5] See VC-dimension later

The Bias-Variance tradeoff

Wanted property (for an $\mathcal{F}$)	unwanted consequence of $\mathcal{F}$ not satisfying this property	what to do
to fit $\mathcal{D}$ well	Bias	increase complexity
to be robust to sampling noise	Variance	decrease complexity

The **bias-variance tradeoff** is the observation that the better a predictor class $\mathcal{F}$ is able to fit any given sample, the more sensitive the selected f will be to sampling noise.
In this course we will learn some ways of balancing these desired properties (or these undesired consequences).

Examples, examples...

Example (K-nearest neighbor classifiers)

The 1-NN can fit any data set perfectly (every data point is it's own nearest neighbor). But for $K > 1$, the K-NN may not be able to reproduce any pattern of ± 1 in the labels. Hence its bias is larger than the bias of the 1-NN classifier. With the variance, the opposite happens: as K the number of neighbors increases, the decision regions of the K-NN classifier become more stable to the random sampling effects. Thus, the variance decreases with K.

Example (Linear vs quadratic vs cubic ... predictors)

The quadratic functions include all linear functions, the cubics include all quadratics, and so on. Linear classifiers will have more bias (less flexibility) than quadratic classifiers. On the other hand, the variance of the linear classifier will be lower than that of the quadratic. The case of regression is even more straightforward: if we fit the data with a higher degree polynomial, the fit will be more accurate, but the variation of the polynomial $f(x)$ for x values not in the training set will be higher too.

Example (Kernel regression)

Examples, examples... (2)

The bias-variance tradeoff can be observed on a continuous range for **kernel regression**. When the kernel width h is near 0, $f(x)$ from Lecture 1, equation (25) will fit the data in the training set exactly [Exercise: prove this], but will have high variance. When h is large, $f(x^i)$ will be smoothed between x^i and the other data points nearby, so it may be some distance from y^i. However, precisely because $f(x)$ is supported by a larger neighborhood, it will have low variance. [Exercise: find some intuitive explanations for why this is true] Hence, the smoothness parameter h controls the trade-off between bias and variance.

Example (Regularization)

The same can be observed if one considers equation (**??**). For $\lambda = 0$, one choses f that best fits the data (minimizes $\hat{L}$. For $\lambda \rightarrow \infty$, f is chosen to minimize the penalty J, disregarding the data completely. The latter case has 0 variance, but very large bias. Between these extreme cases, the parameter λ controls the amount in which we balance fitting the data (variance) with pulling f towards an a-priori "good" (bias).

Overfitting and Underfitting

- Bias and variance are properties of the model class $\mathcal{F}$ (sometimes toghether with the learning algorithm – more about this later). They are not properties of the parameters of f (e.g β), and not of a particular $f \in \mathcal{F}$.
- Variance decreases to 0 with n, but bias may not. This implies that for larger sample sizes n, the trade-off between variance and bias changes, and typically the "best" trade-off, aka the best model, will have larger complexity.
- **Overfitting**= is the situation of small bias and too much variance (i.e. $\mathcal{F}$ is too complex). In practice, if a learned predictor f has low $\hat{L}(f)$ but significantly higher $L(f)$, we say that the model has *overfit* the data $\mathcal{D}$. (Of course we cannot know $L(f)$ directly, and a significant amount of work in statistics is dedicated to predicting $L(f)$ for the purpose of chosing the best model.)
- **Underfitting**=bias is too high, or the model is too simple (a.k.a has too few degrees of freedom). [Exercise: what do you expect to see w.r.t. $\hat{L}(f)$ v.s. $L(f)$ for an underfitted model?]

Complexity, even though there are variations in its definition, and although it is not known exactly for most model classes, is at the core of learning theory, the part of statistical theory that gives provable results about the expected loss of a predictor.