STAT 535

10/4/22

ZOOM

Lecture 2

Predictors

Q∅ — ask Steve !!

Q1 $\geq$ next Tuesday

HW1 ~ T. B. posted

Supervised Learning

Lecture Notes I – Examples of Predictors

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

September 29, 2022

Prediction problems by the type of output

The "learning" paradigm and vocabulary

The Nearest-Neigbor and kernel predictors

Linear predictors
- Least squares regression
- Linear Discriminant Analysis (LDA)
- QDA (Quadratic Discriminant Analysis)
- Logistic Regression
- The PERCEPTRON algorithm

Classification and regression tree(s) (CART)

The Naive Bayes classifier

Reading HTF Ch.: 2.3.1 Linear regression, 2.3.2 Nearest neighbor, 4.1–4 Linear classification, 6.1–3. Kernel regression, 6.6.2 kernel classifiers, 6.6.3 Naive Bayes, 9.2 CART, 11.3 Neural networks, Murphy Ch.: 1.4.2 nearest neighbors, 1.4.4 linear regression, 1.4.5 logistic regression, 3.5 and 10.2.1 Naive Bayes,4.2.1–3 linear and quadratic discriminant, 14.7.3– kernel regression, locally weighted regression, 16.2.1–4 CART, (16.5 neural nets), Bach Ch.:

Prediction problems by the type of output

In supervised learning, the problem is *predicting* the value of an **output** (or **response** – typically in regression, or **label** – typically in classification) variable Y from the values of some observed variables called **inputs** (or **predictors, features, attributes**) $(X_1, X_2, \ldots X_d) = X$. Typically we will consider that the input $X \in \mathbb{R}^d$. Prediction problems are classified by the type of response $Y \in \mathcal{Y}$:

- *regression*: $Y \in \mathbb{R}$
- *binary classification*: $Y \in \{-1, +1\}$ – anomaly detection
- *multiway classification*: $Y \in \{y_1, \ldots y_m\}$ a finite set
- *ranking*: $Y \in \mathbb{S}_p$ the set of permutations of p objects
- *multilabel classification* $Y \subseteq \{y_1, \ldots y_m\}$ a finite set (i.e. each X can have several labels)
- *structured prediction* $Y \in \Omega_V$ the state space of a graphical model over a set of [discrete] variables V

Ex: - sports
- accidents OR history OR international

Example (**Multiway classification.**)

Handwritten digit classification: $Y \in \{0, 1, \ldots 9\}$ and X=black/white 64$\times$ 64 image of the digit. For example, ZIP codes are being now read automatically off envelopes.
OCR (Optical character recognition). The task is to recognized printed characters automatically. X is again a B/W digital image, $Y \in \{a-z, A-Z, 0-9, ".", ",", \ldots\}$, or another character set (e.g. Chinese).

Example (**Diagnosis**)

Diagnosis is multiway classification $+$ anomaly detection. $Y = 0$ means "normal/healthy", while $Y \in \{1, 2, \ldots\}$ enumerates failure modes/diseases.

Multiway classification

$y \in \{1, 2, 3, \ldots L\}$ L classes

Diagnosis $y = 0$ nominal state

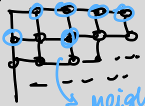
$y \in \{1, 2, \ldots L\}$ failure modes

Example (**Structured prediction.**)

Speech recognition. X is a segment of digitally recorded speech, Y is the word corresponding to it. Note that it is not trivial to *segment speech*, i.e to separate the speech segment that corresponds to a given word. These segments have different lengths too (and the length varies even when the same word is spoken).
The classification problem is to associate to each segment X of speech the corresponding word. But one notices that the words are not indepedent of other neighboring words. In fact, people speak in sentences, so it is natural to recognize each word in dependence from the others. Thus, one imposes a graphical model structure on the words corresponding to an utterance $X^1, X^2, \ldots X^m$. For instance, the labels $Y^{1:m}$ could form a *chain* $Y^1 - Y^2 - \ldots Y^m$. Other more complex graphical models structures can be used too.

$X^1 - X^2 - \ldots X^t - \ldots X^T$

"sky" pixels in image



neighbors also likely to have same class

X^1 X^2 X^3 X^5

Mary had a little lamb

$Y^1 - Y^2 - Y^3 - Y^4 - Y^5$

noun verb article adj noun

verb ??

dependencies btw labels

← "graphical model"

Ranking

input set $\{x^{1,1}, x^{1,2}, \ldots x^{1,m_1}\} = X^1 \longrightarrow y^1 = $ permutation of X^1

$\{x^{2,1} \quad \ldots \quad x^{2,m_2}\} = X^2$

Ex: Search engines

X = set of relevant pages

y = ordering, most relevant

- distances between rankings:

e.g. inversion distance

\# inversions between (A,B,C) and $(B,A,C) = 1$

$\hat{y} = (B, A, C)$

$y^{true} = (A, B, C)$

$\hat{\hat{y}} = (C, B, A)$

The "learning" paradigm and vocabulary

- **predictor** = a [deterministic] function that associates to an input x a corresponding $\hat{y} = f(x)$. ← our guess ↔ goal of learning
- A predictor is a kind of model (not yet a statistical model, though).
- **model class** $\mathcal{F}$ = the set of possible predictors for a problem

In reality $y(x)$ not deterministic

$P_{Y|X=x}$

"wreck a nice beach"

The "learning" paradigm and vocabulary

- **predictor** = a [deterministic] function that associates to an input x a corresponding $\hat{y} = f(x)$.
- A predictor is a kind of model (not yet a statistical model, though).
- **model class** $\mathcal{F}$ = the set of possible predictors for a problem

$\mathcal{F} = \{$regression trees$\}$

e.g. $\mathcal{F} = \{$linear classifiers$\}$

$\mathcal{F} = \{f$ predictors$\}$

- **Training** = inference, estimation
  - choose the "best" predictor in $\mathcal{F}$ (for a particular task)
  - based on a **sample** or (**training set**) of **labeled data**

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

  - n is the **sample size**.
  - (x^i, y^i) are **examples**
  - In binary classification labels are conventionally in $\{\pm\}$ (or $\{\pm 1\}$). We use the terms **negative**, respectively **positive** examples

$(x, -1)$

$(x, +1)$

$x \in \mathbb{R}^d$

$[0, 1]^d$

$\{0, 1\}^d$ binary

$x = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$ input

input feature attribute

y = label (in classification) output, response

The "learning" paradigm and vocabulary

- **predictor** = a [deterministic] function that associates to an input x a corresponding $\hat{y} = f(x)$.
- A predictor is a kind of model (not yet a statistical model, though).
- **model class** $\mathcal{F}$ = the set of possible predictors for a problem
- **Training** $\Rightarrow \hat{f} \in \mathcal{F}$
  - choose the "best" predictor in $\mathcal{F}$ (for a particular task)
  - based on a **sample** or (**training set**) of **labeled data**

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

  - n is the **sample size**.
  - (x^i, y^i) are **examples**
  - In binary classification labels are conventionally in $\{\pm\}$ (or $\{\pm 1\}$). We use the terms **negative**, respectively **positive** examples
- **Prediction** (also called **testing**) using the predictor $\hat{f}$
  - Given predictor f, and new input x, calculate

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

"test error" = $\Pr[\hat{y}(x) \text{ "wrong"}]$

Prediction – the workflow

Training phase

- Get labeled data $\mathcal{D} = \{(x^1, y^1), (x^2, y^2), \ldots (x^n, y^n)\}$
- Choose model class $\mathcal{F}$
- Learn/estimate/fit the model $f \in \mathcal{F}$ from data $\mathcal{D}$
  - Here the goal is to find f that predicts $y^{1,\ldots N}$ well
  - How to do it is the **learning algorithm** and depends on $\mathcal{F}$

[**Validation phase** How good is really this f?] — is f good?

Validation data set
new data
other features
e.g. residuals
in regression

"Testing" phase=Prediction

- now you have a predictor f, use it
- whenever new, unlabeled x comes in, output $\hat{y} = f(x)$

Prediction – the workflow

Preprocessing = selecting features x

Training phase

- Get labeled data $\mathcal{D} = \{(x^1, y^1), (x^2, y^2), \ldots (x^n, y^n)\}$ ← training set
- Choose model class $\mathcal{F}$
- Learn/estimate/fit the model $f \in \mathcal{F}$ from data $\mathcal{D}$
  - Here the goal is to find f that predicts $y^{1,\ldots N}$ well
  - How to do it is the **learning algorithm** and depends on $\mathcal{F}$

Model selection

[**Validation phase** How good is really this f?]

Learning Theory

- How to guarantee statistically that f predicts new $y(x)$ well

"Testing" phase=Prediction

- now you have a predictor f, use it
- whenever new, unlabeled x comes in, output $\hat{y} = f(x)$

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

for $\tilde{x}^{1:\tilde{n}}$

- compute $f(\tilde{x})$ ← prediction
- compare with $\tilde{y}$

how close?