<u>Classification</u>: $Y_i = k,\ k \in \{1, \dots, C\}$ (special case: $C=2$ two-class classification: $k \in \{0,1\}$, $k \in \{-1, 1\}$)

- deterministic classifier: one hard decision for each i: $\hat{Y}_i = f(X_i)$ with $\hat{Y}_i = Y_i^*$ ← truth

ideally: all are 0

collect outcomes in "confusion matrix"

$\hat{Y}$ \ Y^*	-1	$+1$
-1	true negative	false negative
$+1$	false positive	true positive

should be diagonal

quality metrics: ① false positive fraction $= \frac{\#FP}{N} \in [0,1]$

false negative fraction $= \frac{\#FN}{N} \in [0,1]$

② false positive rate $= \frac{\#FP}{\#FP + \#TN} \approx p(\hat{Y}=1 \mid Y^*=-1)$

false negative rate $= \frac{\#FN}{\#FN + \#TP} \approx p(\hat{Y}=-1 \mid Y^*=1)$

- probabilistic classifier: returns a probability for every label ≡ posterior distribution $\underbrace{p(Y=k \mid X)}_{\text{vector of probabilities } \mathbb{R}^C}$

"soft response"

more general than hard classification: can recover decision function by return the most probable label

$$p(Y=k \mid X) \Longrightarrow f(X) = \arg\max_k p(Y=k \mid X)$$

quality measured by "calibration": if $p(Y=k \mid X) = v$, then label k should be correct $v\%$ of the cases
↳ check with $k = Y^*$

if actual accuracy higher $v\%$: "under confident"
lower $v\%$: "over confident" ← dangerous, often in neural networks

<u>important</u>: calculate confusion matrix or calibration from <u>test set</u>

Threshold classifier: single feature: $X_i \in \mathbb{R}$ $\quad \hat{Y} = \text{sign}(X - T) = \begin{cases} 1 & \text{if } X > T \\ -1 & \text{if } X < T \end{cases}$

↑ threshold

ex: $Y = \begin{cases} 1: & \text{"obese"} \\ -1: & \text{"not obese"} \end{cases}$ $\quad X = \text{weight} \quad T: 90\,\text{kg}$

usually, a single feature is not enough to predict response

three classical solutions for multiple features: $\quad X_i = \{\text{weight}, \text{height}\}$ (kg, m)

1. design a formula to combine feature into single score ex. body-mass index $\text{BMI} = \text{weight} / \text{height}^2 \quad \hat{Y} = \text{sign}(\text{BMI} - 25)$
   but: hard & expensive for most problems
2. compute score as a linear combination and learn coefficients $\hat{S}_i = \sum_{j=1}^{D} \beta_j X_{ij} \Rightarrow \hat{Y}_i = \text{sign}(\hat{S}_i - T)$
   "linear classification"
3. classify test instance X_{test} according to most similar training instance $\hat{i} = \arg\min_i \text{dist}(X_i, X_{test})$
   "nearest neighbor classification" $\quad \hat{Y}_{test} = Y_{\hat{i}}^{*}$

problems of NN methods: - expensive when N is large: store entire TS and scan for every X_{test}
(faster search exists, but scales badly with D)
- very hard to define $\text{dist}(X_{test}, X_i)$ to reflect true semantic similarity
⇒ machine learning task "metric learning" figure out dist from training data

Bayes rule of conditional probabilities:

joint distribution of features & labels: $p(X, Y)$ can be decomposed by chain rule of probability in two ways

$$p(X,Y) = p(X)\, p(Y \mid X) \quad \text{first measure features, then resp.}$$

$$= p(Y)\, p(X \mid Y) \quad \text{first determine response then get compatible features}$$

posterior, likelihood, prior

$$\text{classification: } p(Y{=}k \mid X) = \frac{p(X \mid Y{=}k)\, p(Y{=}k)}{p(X)} \quad \text{Bayes' rule}$$

↑ marginal, evidence

$$\boxed{p(X) = \sum_{k=1}^{C} p(Y{=}k)\, p(X \mid Y{=}k)}$$

Bayes Rule: $p(Y=k|X) = \frac{p(X|Y=k)\,p(Y=k)}{p(X)}$

posterior — likelihood — evidence / marginal — prior

$$p(x) = \sum_{k=1}^{c} p(X|Y=k) \cdot p(Y=k)$$

why is this important?

- general: fundamental equation for probabilistic machine learning because it allows clean uncertainty handling
- defines two fundamental model types
- puts model accuracy into perspective: what is good or bad perf.?

fundamental model types

- discriminative models: learn $p(Y=k|X)$ (LHS of Bayes)
  - answer question "what class" directly
  - ⊕ relatively easy – take direct route, no detour
  - ⊖ often hard to interpret, <u>how</u> model makes decisions "black box" behavior of neural networks

- generative model: learns $p(Y=k)$ and $\underbrace{p(X \mid Y=k)}$ (RHS of Bayes)
  - first learn, how "the world" works: understand mechanism, how observations ("phenotypes") arise from hidden properties ("genes")
    then use this to answer question
  - ⊖ more difficult: need powerful models & a lot of data
  - ⊕ more interpretable
- history:
  - traditional science seeks generative models: can create synthetic data that are indistinguishable from real data
  - ≈ 1990: ML researchers realized that their models were too weak
    ⇒ field focused on discriminative models
  - ≈ 2012: neural networks solved many hard discriminative tasks
    ⇒ field is again interested in generative models (ChatGPT, Midjourney
    subfield "explainable / interpretable ML"

How does Bayes help to evaluate accuracy?

- how good can a learned classifier be?

Def: Bayes classifier uses Bayes rule (LHS or RHS) with true probs. p^*

Thm: No learned classifier using $\hat{p}$ can be better than Bayes with p^*

ex: $p^*(Y=1|X)=0.6$ then $\underline{\text{error rate}}$ less $1-60\% = 40\%$ $\underline{\text{impossible}}$

$1-p^*(Y=y^*|X)$

- how bad can a classifier be?

$\underline{\text{case 1}}$: all classes are equally probable $p(Y=k)=\frac{1}{C}$ for all $k=1,\dots,C$

two classes $p(Y=k)=0.5$

worst classifier: pure guessing $\Rightarrow$ correct 50% of time

$\underline{\text{case 2}}$: unbalanced classes, e.g. $p(Y=1)=0.01$ $p(Y=-1)=0.99$

↑ prevalence of disease

↑ does not have this disease

worst classifier: always return majority label $\Rightarrow$ 99% correct

ex: breast cancer screening $p(Y=1)=0.01$ $p(Y=-1)=0.99$

test, e.g. mammography $p(X=\text{test positive}\mid Y=1)=0.99$

$p(X=\text{test positive}\mid Y=-1)=0.01$

"false alarm"

Q: If a test is positive, should you panic?

$$p(Y=1\mid X=\text{test positive}) = \frac{p(X=1\mid Y=1)\,p(Y=1)}{p(X=1\mid Y=1)\,p(Y=1)+p(X=1\mid Y=-1)\,p(Y=-1)}$$

$$= \frac{0.99\cdot 0.01}{0.99\cdot 0.01 + 0.01\cdot 0.99} = 0.5$$