CSE/ISyE 6740
Computational Data Analysis

Naive Bayes, KNN and Logistic Regression

09/15/2025

Kai Wang, Assistant Professor in Computational Science and Engineering
kwang692@gatech.edu

Georgia Tech

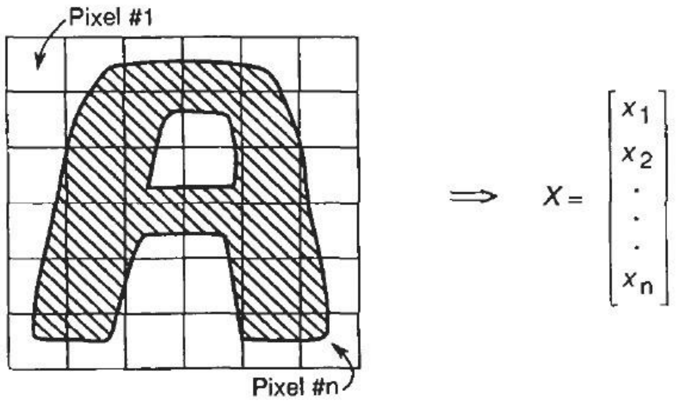
Outline

- Supervised Learning
  - Bayes and Naïve Bayes
  - K-nearest neighbors
  - Logistic regression
  - Support Vector Machine (Wednesday!!)

Naive Bayes, KNN and Logistic Regression

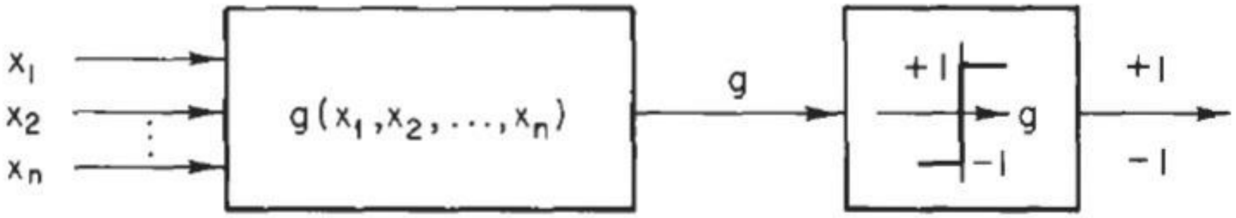
Classification

- Given a dataset $D = \{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}$, can we learn a classifier $f: X \rightarrow Y$?



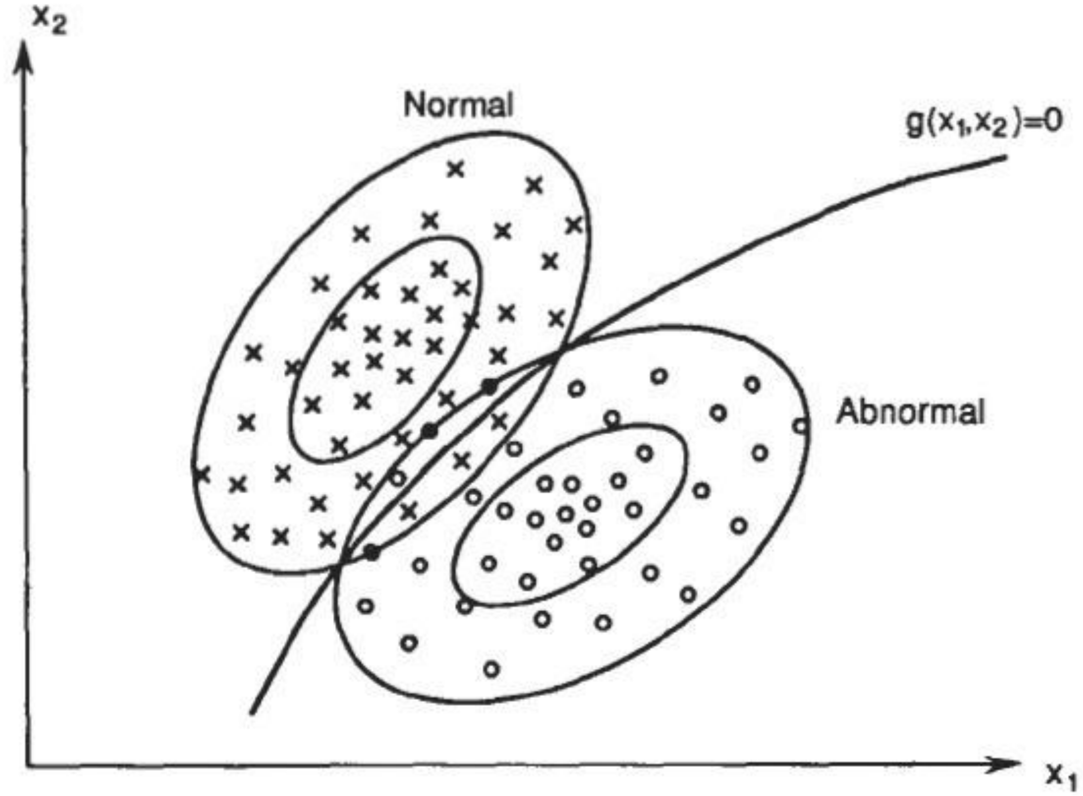
$Y = \text{“}A\text{”}$

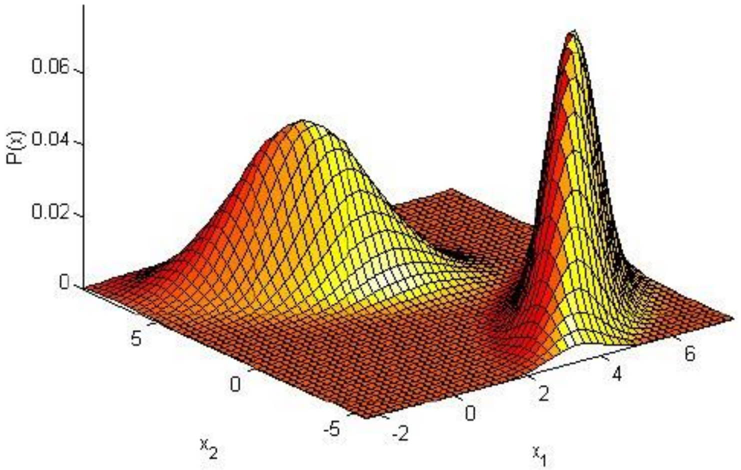
- Classifier:



Decision-making: Divide High-dimensional Space

- Distributions of sample from normal (positive class) and abnormal (negative class) tissues
  - How to construct the classifier $f: X \to Y$?





What Do People Do in Practice?

- **Bayes classifier**: use the trick $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$
  - Make some assumption on $P(x|y)$ (e.g., Gaussian distribution)

- **K-nearest neighbor**
  - Geometric intuitions: closer data points must have similar labels

- **Logistic regression**
  - Directly go for the decision boundary $h(x) = \log\frac{q_1(x)}{q_0(x)}$
  - Neural networks

What Do People Do in Practice?

- **Bayes classifier**: use the trick $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$
  - Make some assumption on $P(x|y)$ (e.g., Gaussian distribution)

- **K-nearest neighbor**
  - Geometric intuitions: closer data points must have similar labels

- **Logistic regression**
  - Directly go for the decision boundary $h(x) = \log \frac{q_1(x)}{q_0(x)}$
  - Neural networks

Use Bayes Rule

likelihood prior

$$\underbrace{P(y|x)}_{\text{posterior}} = \frac{P(x|y)P(y)}{\underbrace{P(x)}_{\text{Normalization constant}}} = \frac{P(x,y)}{\sum_{y'} P(x,y')}$$

- Prior: $P(y)$
- Likelihood (class conditional distribution): $P(x|y) = N(x; \mu_y, \Sigma_y)$ (e.g., Gaussian)
- Posterior: $P(y|x) = \frac{P(y)N(x;\mu_y,\Sigma_y)}{\sum_{y'} P(y')N(x;\mu_{y'},\Sigma_{y'})}$

Bayes Decision Rule

- **Learning**:
  - prior: $P(y)$
  - Compute class conditional distribution: $P(x|y)$

- The posterior probability of a test point:

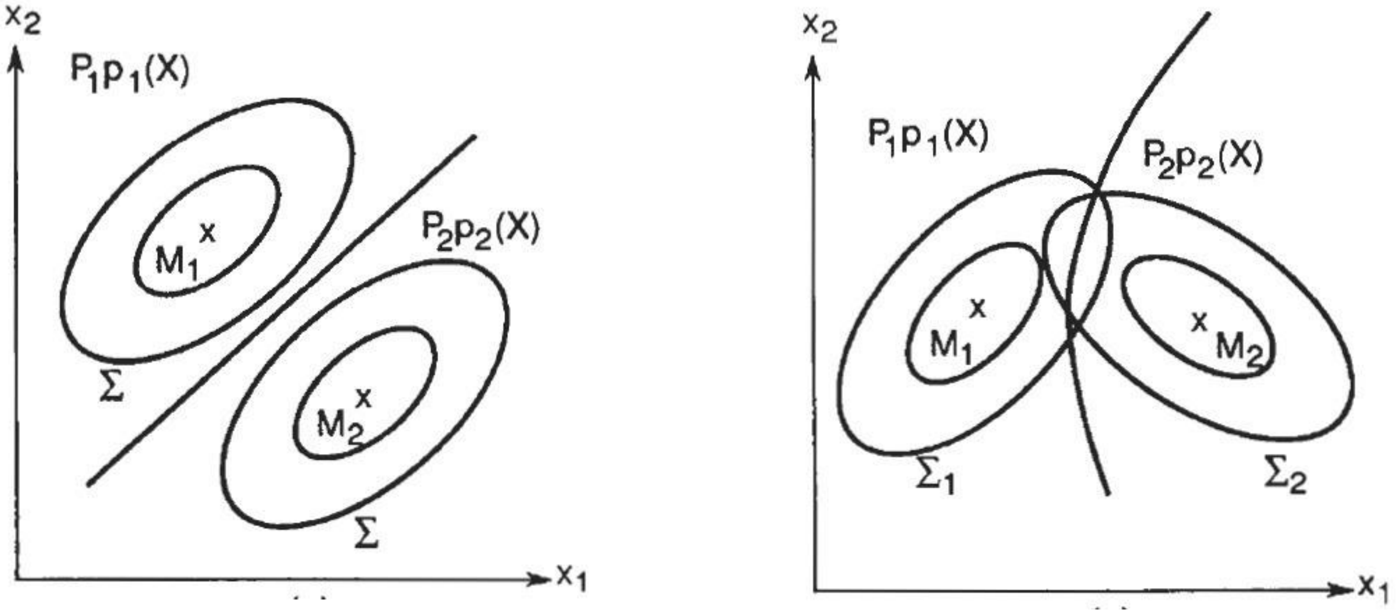
$$q_i(y) := P(y = i|x) = \frac{P(x|y = i)P(y = i)}{P(x)}$$

- Two classes ($y \in \{0,1\}$)
  - If ratio $g(x) = \frac{\mathrm{q}_1(x)}{\mathrm{q}_0(\mathrm{x})} = \frac{P(x|y = 1)P(y=1)}{P(x|y = 0)P(y=0)} > 1$, then $y = 1$, otherwise $y = 0$
  - Or look at the log-likelihood ratio: $\mathrm{h}(\mathrm{x}) = \log \frac{\mathrm{q}_1(x)}{\mathrm{q}_0(\mathrm{x})}$

- Multiple classes
  - Find the i with the largest $q_i(y)$

Example: Gaussian Class Conditional Distribution

- Depending on the Gaussian distributions, the decision boundary can be very different



- Decision boundary: $h(x) = \log \frac{q_1(x)}{q_0(x)} = 0$

Classification Using Density Estimation

- Simple binary classifier for input $x^i \in \mathbb{R}^d$ and label $y^i \in \{0,1\}$
  - **Step 1**: use label to estimate $p(y=1)$ and $p(y=0)$
  - **Step 2**: divide your data according to the value of y, and estimate

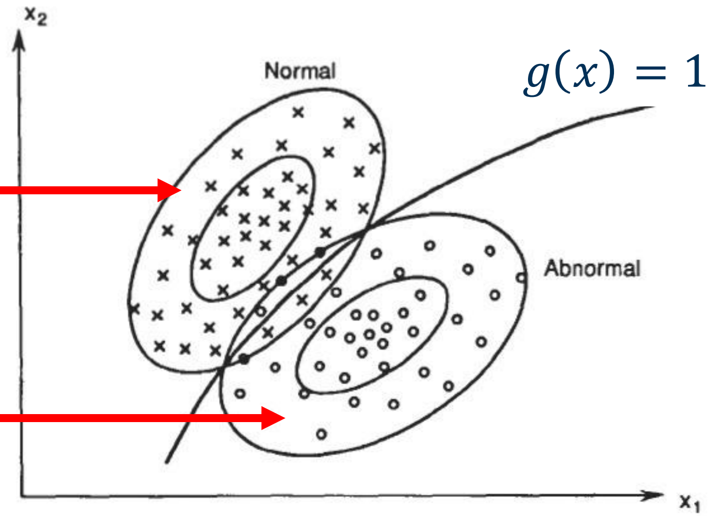
$$p(x|y=1) \text{ and } p(x|y=0)$$

  - **Step 3**: classify a new test point x as:

$$\begin{cases} 1, & if\ g(x) := \dfrac{p(y=1|x)}{p(y=0|x)} = \dfrac{p(x|y=1)p(y=1)}{p(x|y=0)p(y=0)} > 1 \\ 0, & \text{otherwise} \end{cases}$$

$p(x|y=0)$

$p(x|y=1)$



Naïve Bayes Classifier

- Use Bayes decision rule for classification

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

- In general, density estimation of $P(x|y)$ is difficult for high-dimensional x…

- Assume all features $x = (x_1, x_2, \dots, x_d)$ are fully independent, i.e.,

$$P(x|y = 1) = \prod_{i=1}^{d} P(x_i|y = 1)$$

Naïve Bayes Classifier

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

What is $P(stolen \mid red, SUV, domestic)$?

$$P(stolen \mid red, SUV, domestic)$$
$$\propto P(red, SUV, domestic \mid stolen) * P(stolen)$$
$$= P(red \mid stolen) * P(SUV \mid stolen) * P(domestic \mid stolen) * P(stolen)$$
$$= \frac{3}{5} * \frac{1}{5} * \frac{2}{5} * \frac{5}{10}$$
$$= \frac{3}{125}$$

- Features are assumed to be **independent**
  - None of the features are irrelevant and assumed to be contributing **equally** to the outcome
- Decision: $y = \arg\max_y p(y) \prod_{i=1}^{d} p(x_i \mid y)$

Georgia Tech

What Do People Do in Practice?

- **Bayes classifier**: use the trick $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$
  - Make some assumption on $P(x|y)$ (e.g., Gaussian distribution)

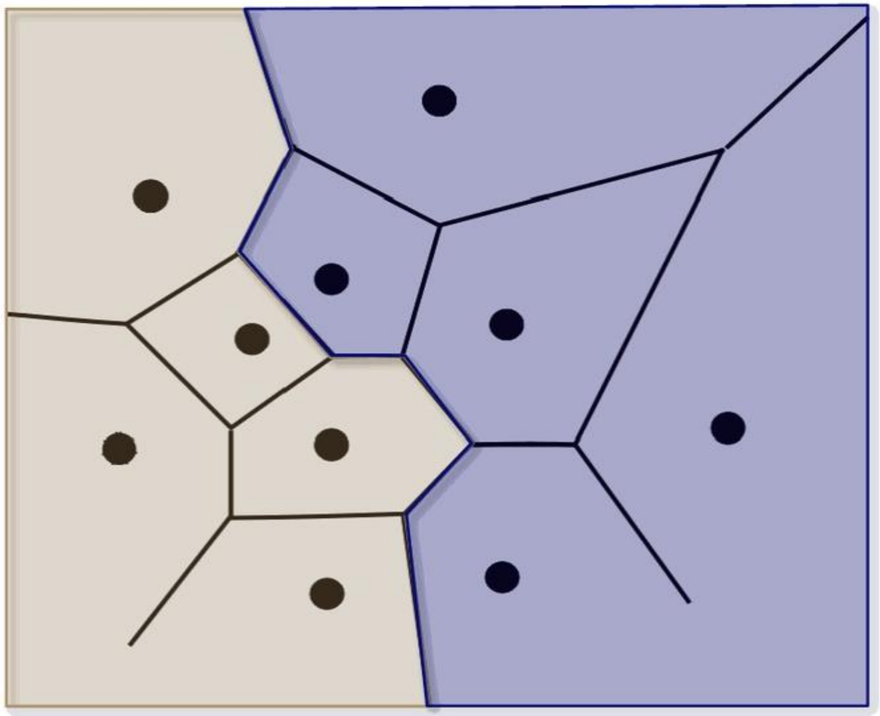
- **K-nearest neighbor**
  - Geometric intuitions: closer data points must have similar labels

- **Logistic regression**
  - Directly go for the decision boundary $h(x) = \log \frac{q_1(x)}{q_0(x)}$
  - Neural networks

Nearest Neighbor Classifier

- The **nearest neighbor classifier**: assign x the same label as the closest training point x_i

- The nearest neighbor rule defines a **Voronoi partition** of the input space



- Parametric or non-parametric?
- Linear or nonlinear?
- Easy to kernelize

k-nearest Neighbor Classifier

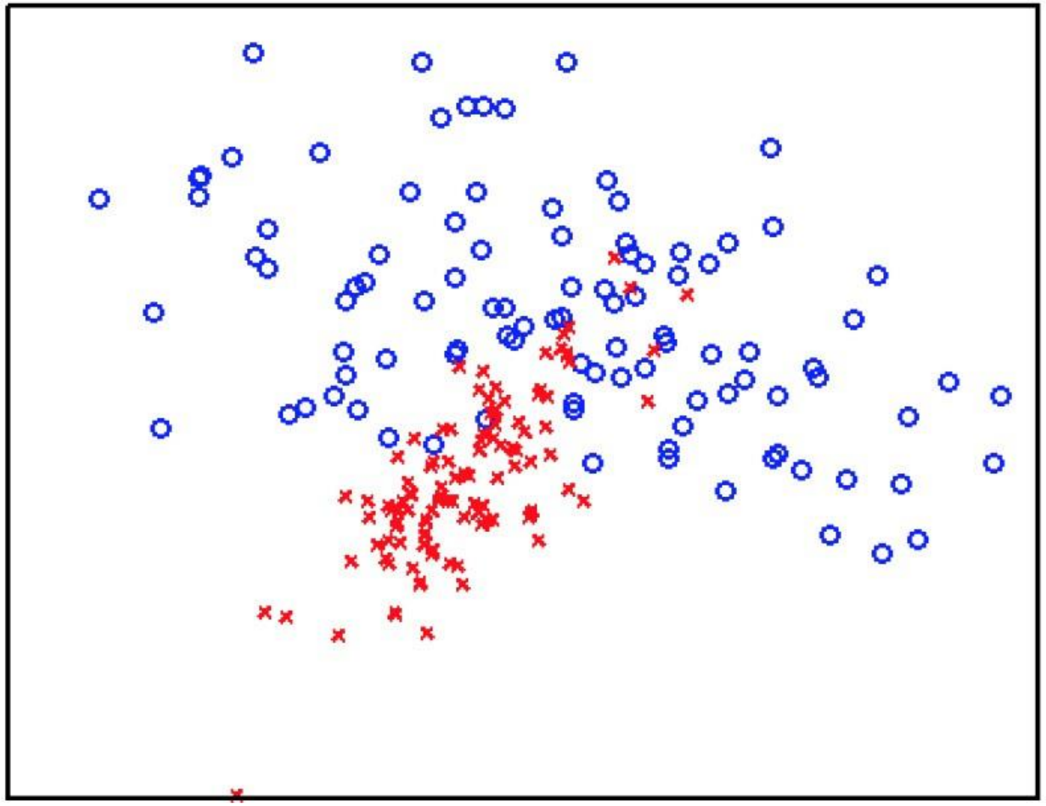
- The **nearest neighbor classifier**: assign x a label by taking a majority vote over **the k training point x_i closest to x**

- For $k \geq 1$, the k-nearest neighbor rule generalizes the nearest neighbor rule

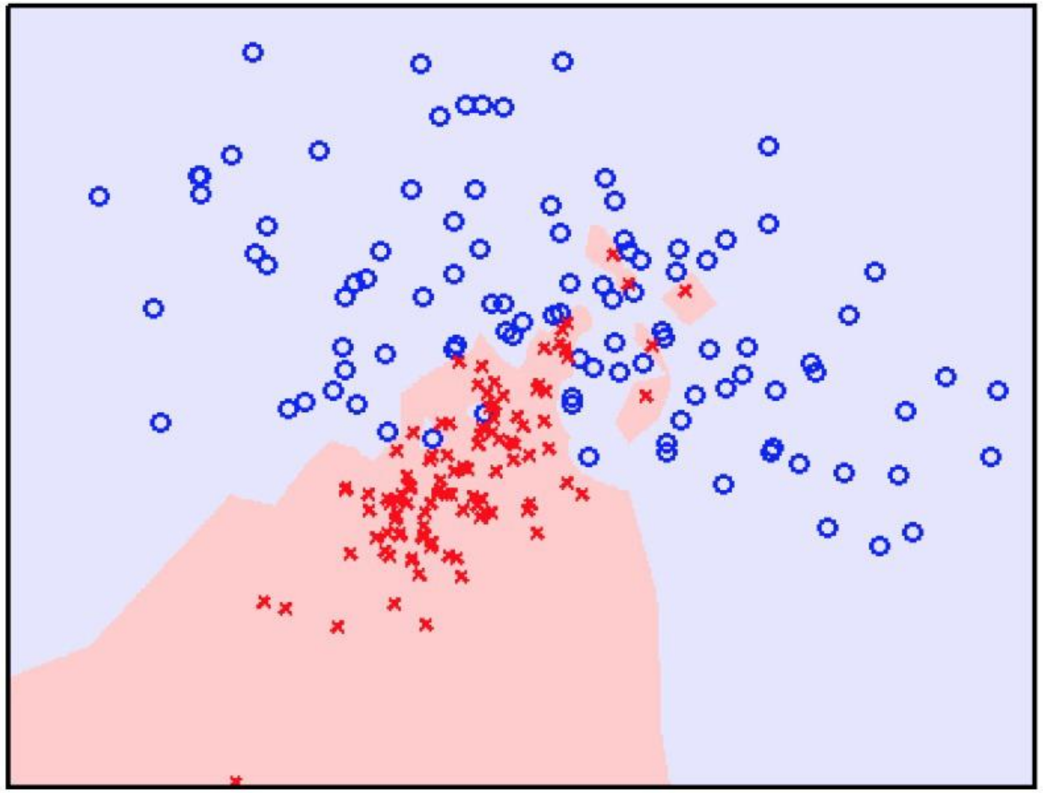
- To define this more mathematically:
  - $I_k(x) :=$ indices of the k nearest training points closest to x
  - If $y_i = \pm 1$, then we can write the k-nearest neighbor classifier as:

$$f_k(x) := sign\left(\sum_{i \in I_k(x)} y_i\right)$$

Example

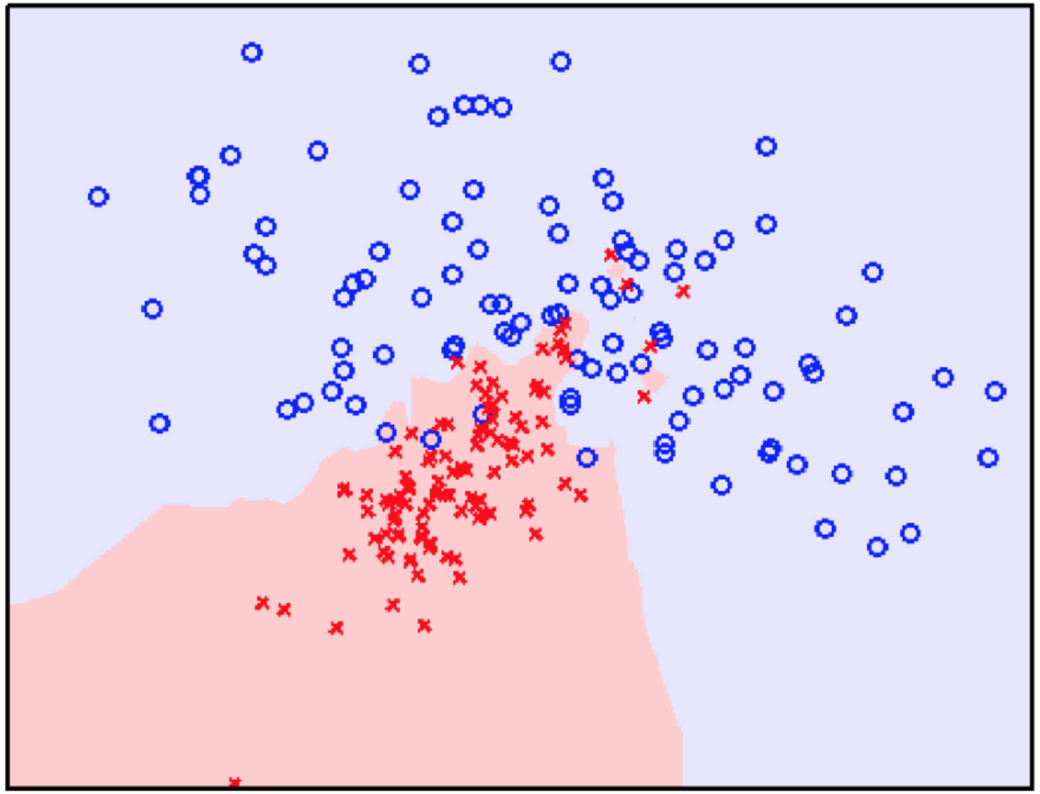


Example



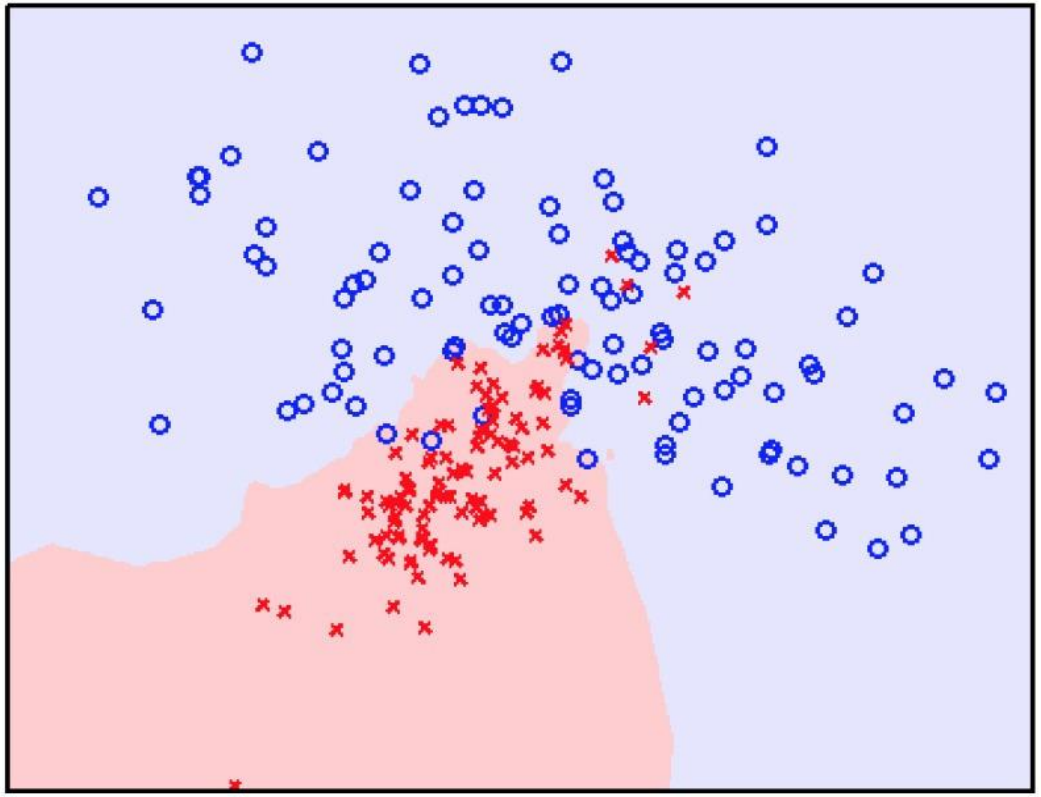
$k = 1$

Example



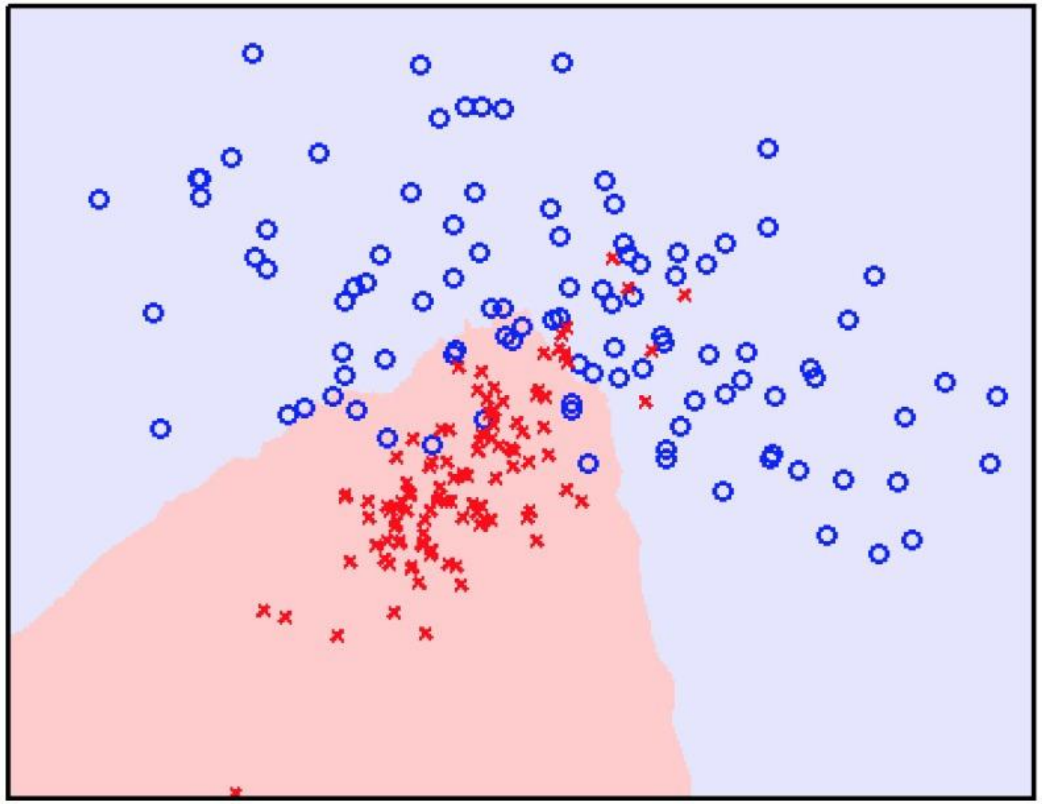
$k = 3$

Example



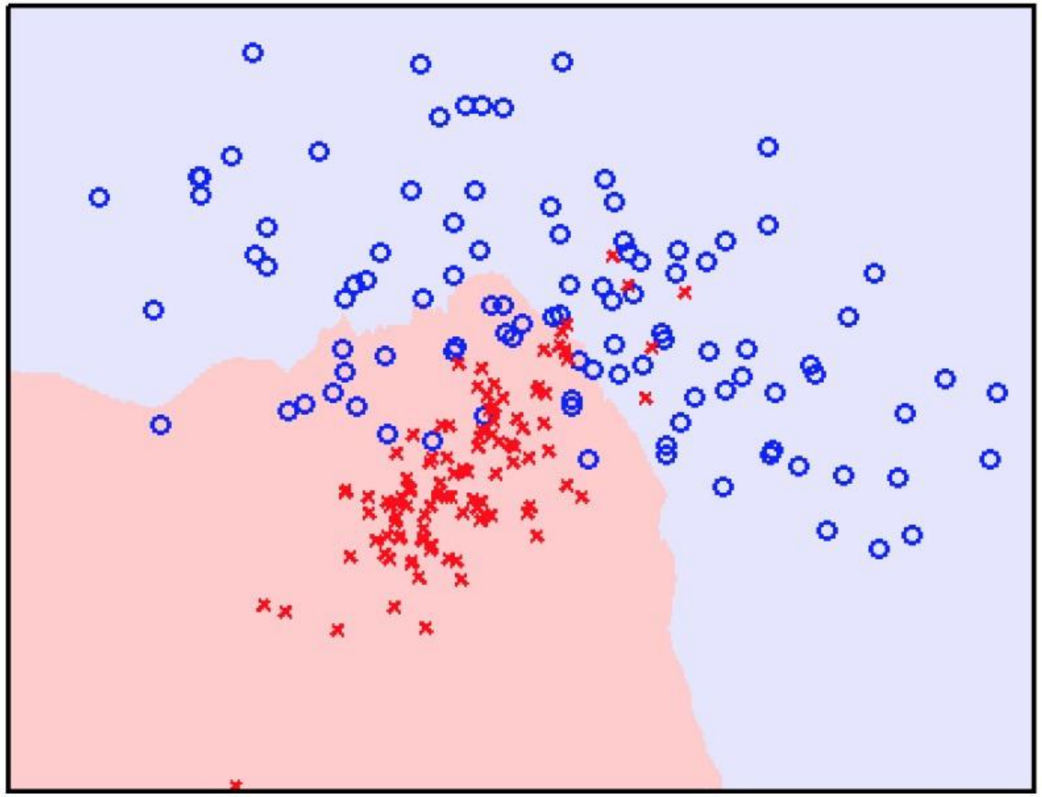
$k = 5$

Example



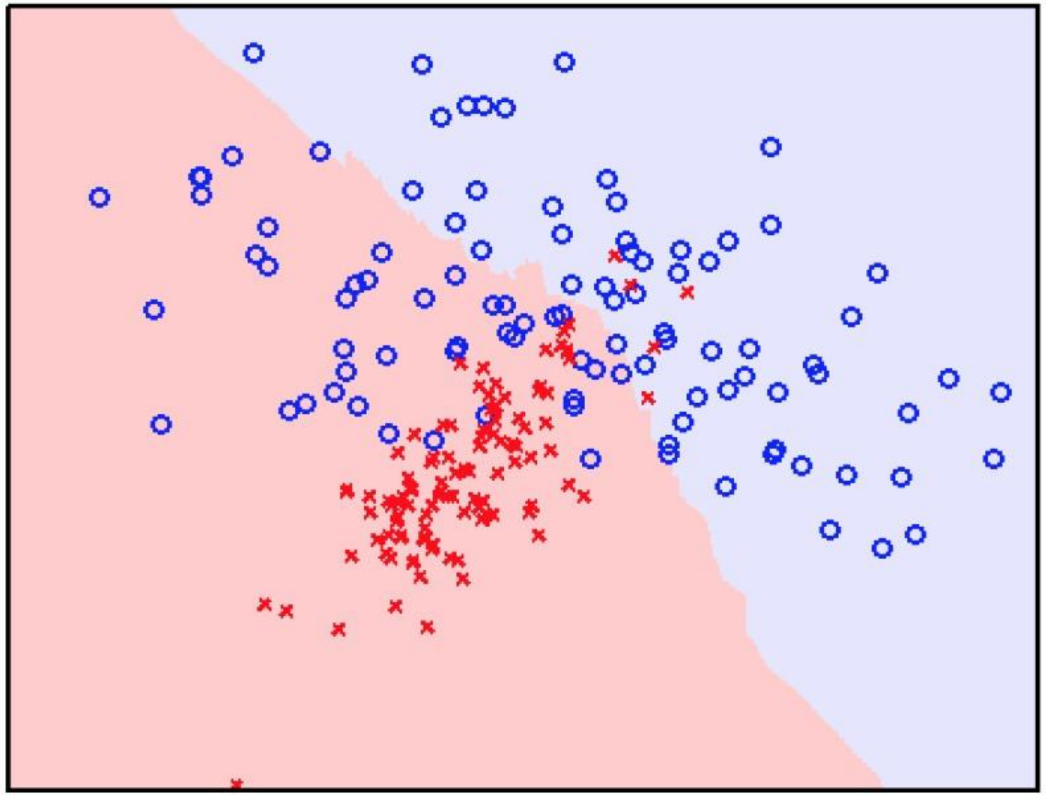
$k = 25$

Example



$k = 51$

Example



$k = 101$

Choosing the Size of the Neighborhood

- Setting the parameter k is a problem of **model selection**

- Choosing k with the smallest training error is **NOT** a good idea:

$$\hat{R}_n(f_k) = \frac{1}{n}\sum_{i=1}^{n} 1_{\{f_k(x_i) \neq y_i\}}$$

- By choosing $k = 1$, we always have training error $\hat{R}_n(f_1) = 0$

- Not much practical guidance from the theory, so we typically must rely on estimates based on holdout sets or more sophisticated model selection techniques.

Computations in KNN

- Similar to KDE, essentially no “training” or ”learning” phase. Computation is needed when applying the classifier
  - Memory: $O(nd)$

- Finding the nearest neighbors out of a set of millions of examples is still pretty hard
  - Test computation: $O(nd)$

- Use smart data structures and algorithms to index training data
  - Memory: $O(nd)$
  - Training computation (preprocessing): $O(n \log n)$
  - Test computation: $O(\log n)$
  - K-D tree, ball tree

What Do People Do in Practice?

- **Bayes classifier**: use the trick $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$
  - Make some assumption on $P(x|y)$ (e.g., Gaussian distribution)

- **K-nearest neighbor**
  - Geometric intuitions: closer data points must have similar labels

- **Logistic regression**
  - Directly go for the decision boundary $h(x) = \log \frac{q_1(x)}{q_0(x)}$
  - Neural networks

Discriminative Classifier

- Directly estimate decision boundary $h(x) = -\log\frac{q_1(x)}{q_0(x)}$ or posterior $p(y|x)$
  - Logistic regression, neural networks
  - Do NOT estimate $p(x|y)$ and $p(y)$

- Model $h(x)$ or $f(x) \coloneqq P(y=1|x)$as a function of x, and
  - Do not have probabilistic meaning
  - Hence can not be used to sample data points

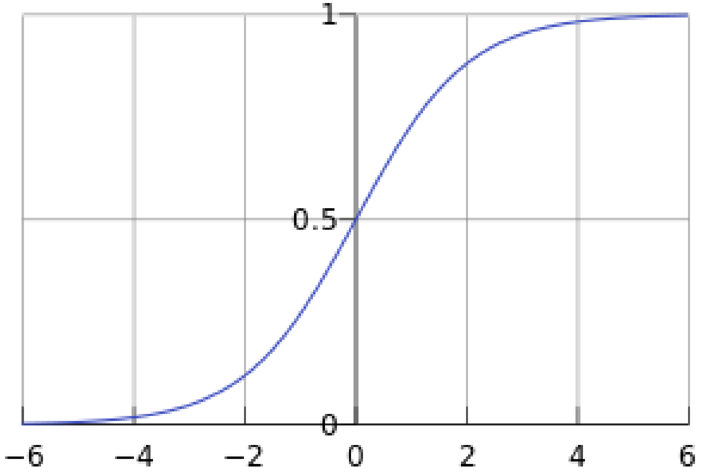
- Why discriminative classifier?
  - Avoid difficult density estimation problem
  - Empirically achieve better classification results

What is Logistic Regression Model

- Assume that the posterior distribution $P(y = 1|x)$ take a particular form:

$$P(y = 1|x, \theta) = \frac{1}{1 + \exp(-\theta^\top x)}$$

- Logistic function $f(u) = \frac{1}{1+\exp(-u)}$



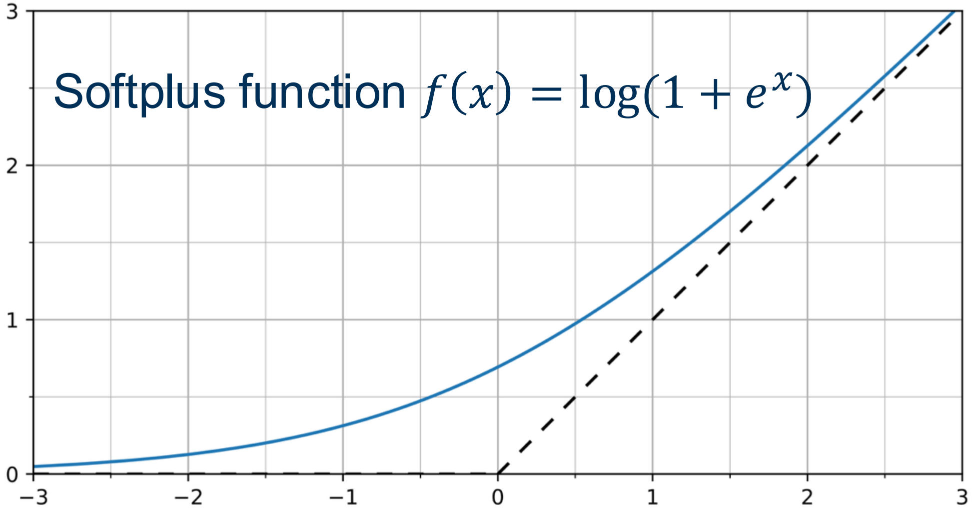
The larger $\theta^\top x$, the higher the chance it is $y = 1$

Learning Parameters in Logistic Regression

- Find θ, such that the conditional likelihood of the labels is maximized

$$\max_{\theta} l(\theta) := \log \prod_{i=1}^{n} P\left(y^i \middle| x^i, \theta\right)$$

- **Good news**: $l(\theta)$ is a concave function of θ, and there is a single global optimum (if it exists).



- **Bad news**: no closed form solution (we need to use numerical method)

The Objective Function $l(\theta)$

- Logistic regression model

$$p(y=1|x,\theta)=\frac{1}{1+\exp(-\theta^\top x)}$$

- Note that

$$p(y=0|x,\theta)=1-\frac{1}{1+\exp(-\theta^\top x)}=\frac{\exp(-\theta^\top x)}{1+\exp(-\theta^\top x)}$$

- Plug in

$$l(\theta):=\log\prod_{i=1}^{n}p\left(y^i\middle|x^i,\theta\right).$$

$$=\sum_i (y^i-1)\theta^\top x^i-\log\left(1+\exp\left(-\theta^\top x^i\right)\right)$$

The Gradient of $l(\theta)$

$$l(\theta) := \log \prod_{i=1}^{n} p(y^i | x^i, \theta)$$

$$= \sum_i (y^i - 1)\theta^\top x^i - \log(1 + \exp(-\theta^\top x^i))$$

- Gradient

$$\frac{\partial l(\theta)}{\partial \theta} = \sum_i (y^i - 1)x^i + \frac{\exp(-\theta^\top x^i)\, x^i}{1 + \exp(-\theta^\top x^i)}$$

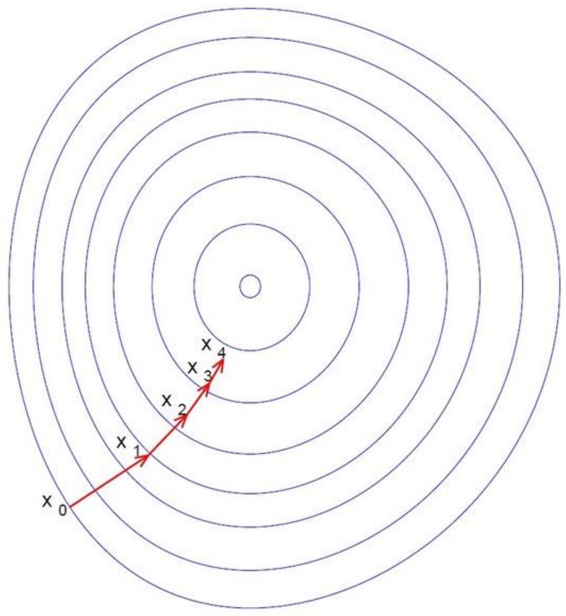
- But setting it to 0 does not lead to closed form solution

Gradient Descent

- Gradient descent is a popular and general method to solve optimization problems

- Given an initial guess, we iteratively refine the guess by taking the direction of the negative gradient

- Think about going down a hill by taking the steepest direction at each step



- Update rule

$$x_{k+1} = x_k - \gamma_k \nabla_x f(x_k)$$

γ_k is called the step size or learning rate

Gradient Ascent/Descent Algorithm

- Initialize parameter θ^0

- Do

$$\theta^{t+1} \leftarrow \theta^t + \eta \sum_i (y^i - 1)x^i + \frac{\exp(-\theta^\top x^i)x^i}{1 + \exp(-\theta^\top x^i)}$$

- While $\|\theta^{t+1} - \theta^t\| > \epsilon$

Outline

- Supervised Learning
  - Bayes and Naïve Bayes
  - K-nearest neighbors
  - Logistic regression
  - Support Vector Machine (Wednesday!!)