

Computational Data Analysis

Machine Learning

Yao Xie, Ph.D.

Associate Professor

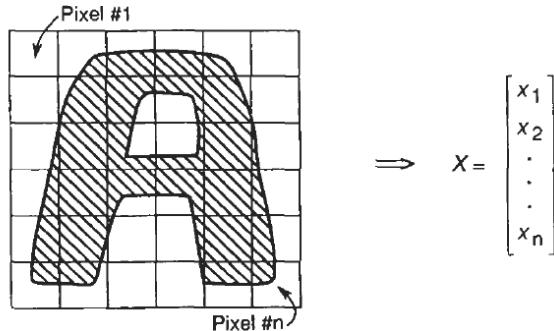
Harold R. and Mary Anne Nash Early Career Professor
H. Milton Stewart School of Industrial and Systems
Engineering

Classification (I)

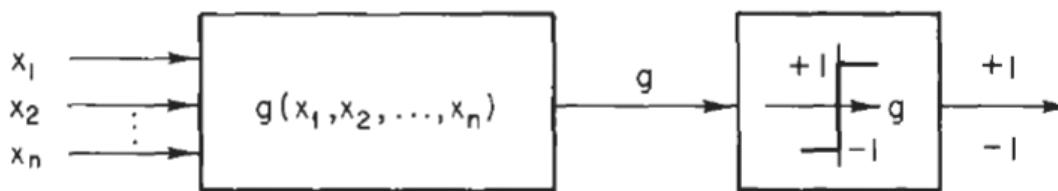


Classification

- Represent the data as a vector



- A label is provided for each data point, eg., $y \in \{-1, +1\}$
- Classifier

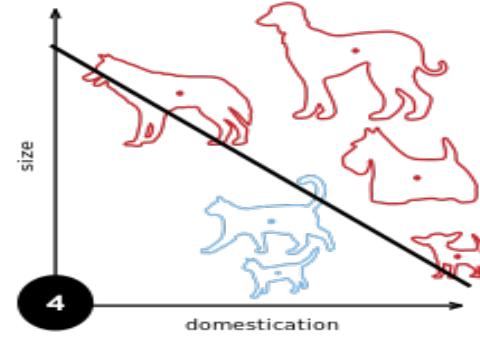
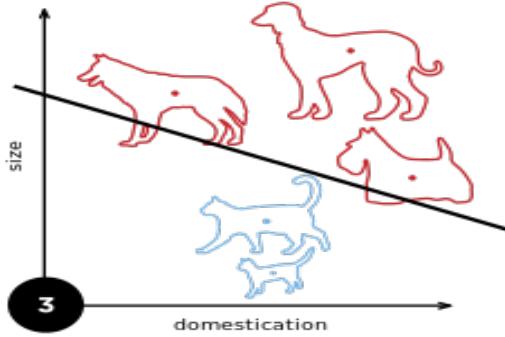
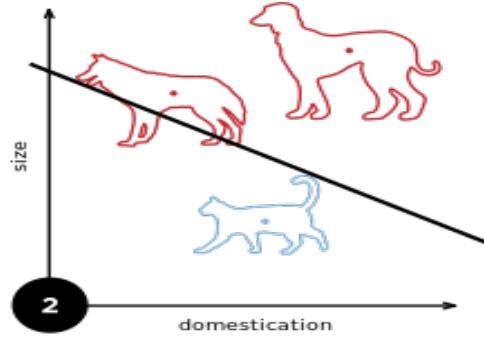
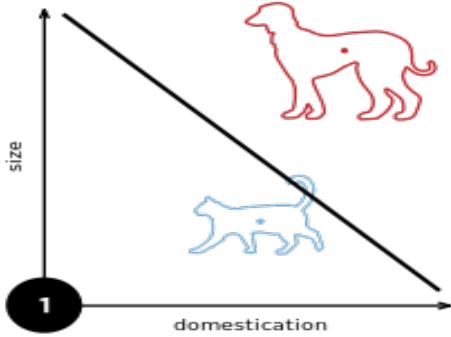


Classification algorithms

- Bayes classifier
- K-nearest neighbors
- Logistic regression

(more to come)

Classify cats from dogs



Classification

- Classification is a predictive task in which the response takes the values across several categories (in the fundamental case, two categories)
- Supervised classification: know the labels for the training data
- Examples:
 - Predicting whether a new patient will develop breast cancer or remain healthy, given genetic information
 - Predicting whether or not a user will like a new product, based on user covariates and a history of his/her previous ratings
 - Predicting the voting preference based on voter's social, political, and economical status



'vote': <http://www.livenewsmalta.com/index.php/2017/05/04/early-voting-in-2017-general-election/>



'Doctor in clinic':
<https://www.leafly.com/news/health/how-to-find-a-doctor-or-clinic-that-specializes-in-medical-marijuana>

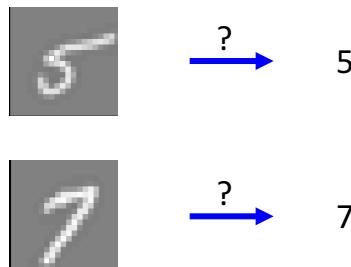
A screenshot of an Amazon recommendation page. At the top left is the Amazon logo and at the top right is the text "Recommended for You". Below this, there is a heading "Amazon.com has new recommendations for you based on items you purchased or told us you own." Three book covers are displayed with "LOOK INSIDE!" buttons: "Google Apps Deciphered: Compute in the Cloud to Streamline Your Desktop" by Jason Rich, "Google Apps Administrator Guide" by Matt Asay, and "Googlepedia: The Ultimate Google Resource (3rd Edition)" by Matt Asay.

'amazon recommendation':
<https://www.mageplaza.com/blog/product-recommendation-how-amazon-succeeds-with-it.html>

Classification versus clustering

Classification (training with label)

1	4	9	9	5	3	2	8	6	1
6	7	1	1	9	7	2	3	6	5
2	4	9	5	6	1	9	2	1	0
1	0	7	0	3	1	1	2	0	8
3	1	0	4	0	5	2	9	3	9
3	7	8	4	4	7	4	2	5	6
5	7	1	4	9	8	4	1	8	3
2	3	6	6	2	9	7	2	2	4
5	1	5	4	7	3	4	7	4	4
9	3	4	9	6	9	2	0	5	0



Supervised learning

Clustering (no label)



Unsupervised learning

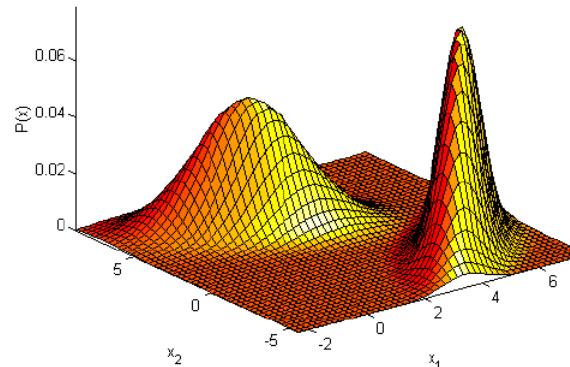
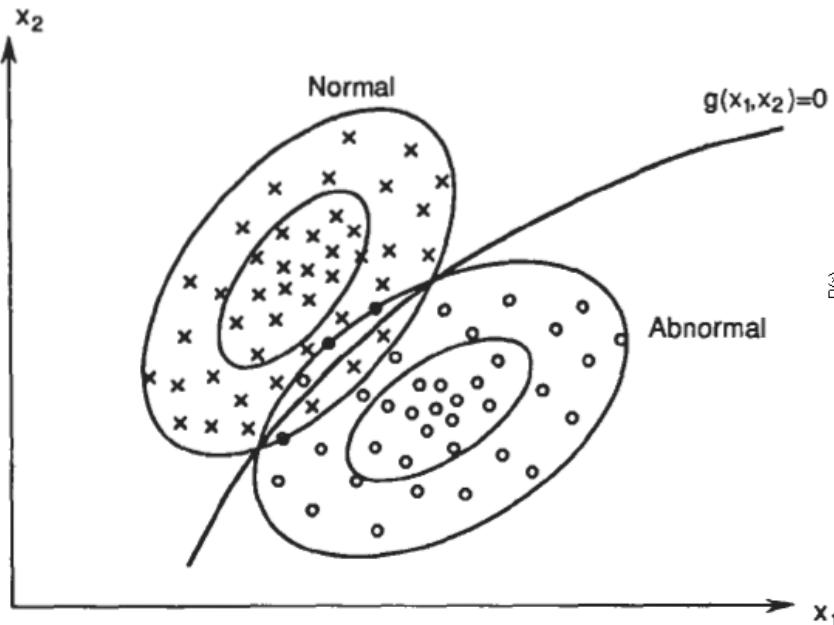
Classification algorithms

- Bayes classifier
- K-nearest neighbors
- Logistic regression

(more to come)

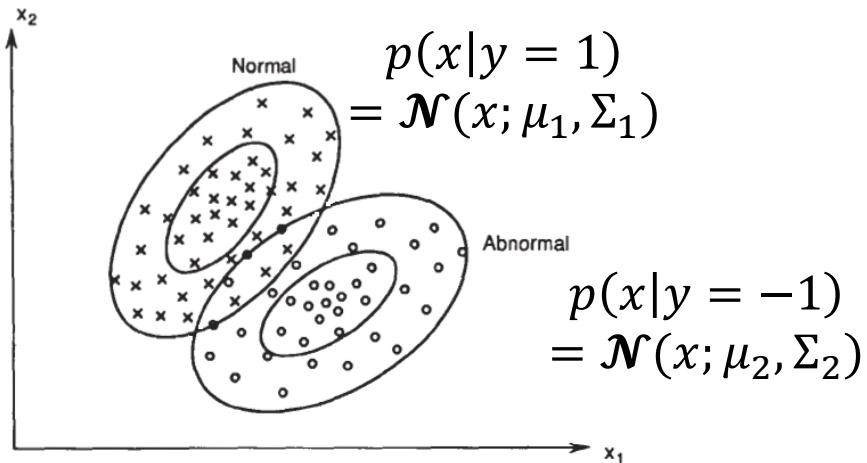
Decision-making: divide high-dimensional space

- Distributions of sample from normal (positive class) and abnormal (negative class) tissues



Class conditional distribution

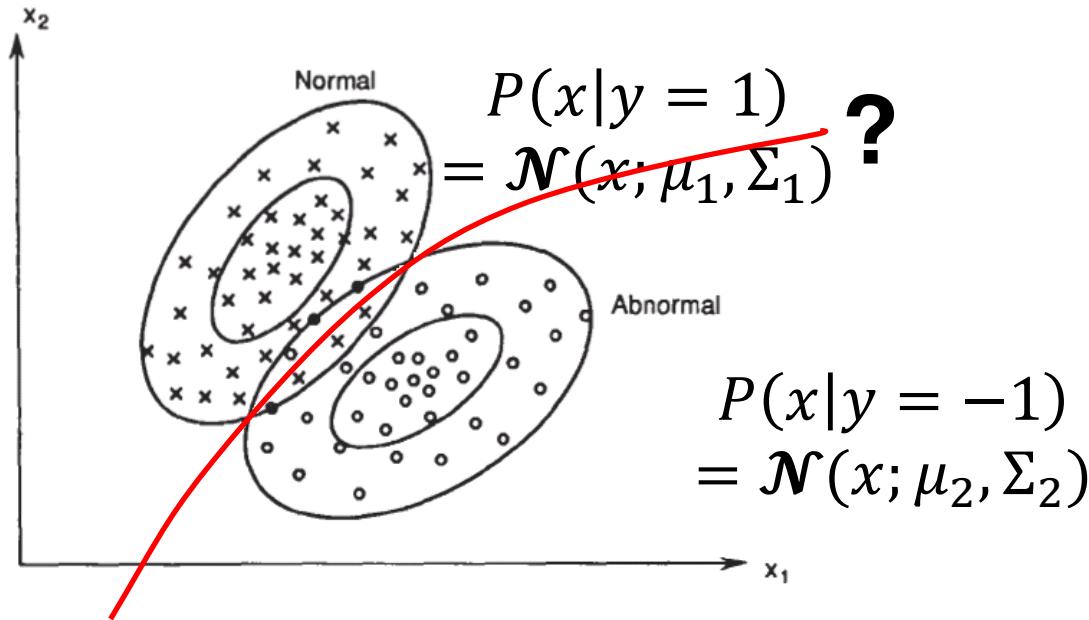
- In classification case, we are given the label for each data point
- Class conditional distribution: $P(x|y = 1), P(x|y = -1)$ (how to compute?)



- Class prior: $P(y = 1), P(y = -1)$

How to come up with decision boundary

- Given class conditional distribution: $P(x|y = 1), P(x|y = -1)$, and class prior: $P(y = 1), P(y = -1)$



Use Bayes rule

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

likelihood Prior
 ↓
 posterior normalization constant

Prior: $P(y)$

Likelihood (class conditional distribution) : $p(x|y) = \mathcal{N}(x|\mu_y, \Sigma_y)$

Posterior: $P(y|x) = \frac{P(y)\mathcal{N}(x|\mu_y, \Sigma_y)}{\sum_y P(y)\mathcal{N}(x|\mu_y, \Sigma_y)}$

Bayes Decision Rule

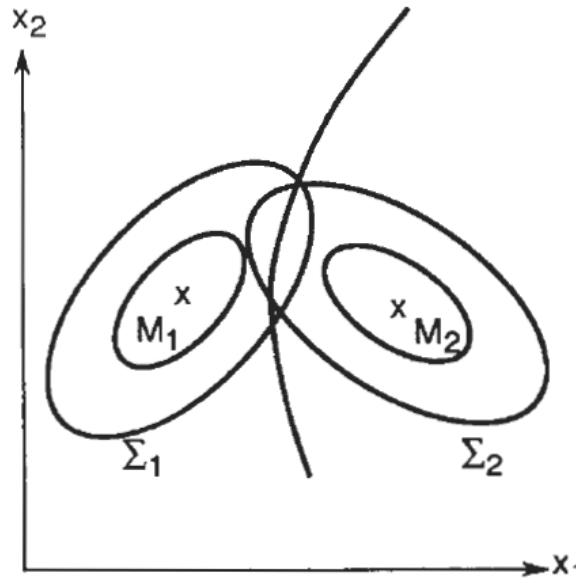
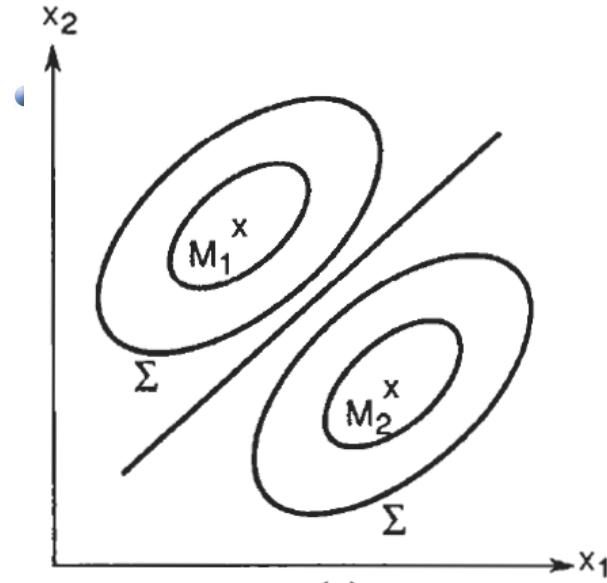
- Learning: prior: $p(y)$, class conditional distribution : $p(x|y)$
- The poster probability of a test point

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

- Bayes decision rule:
 - If $q_i(x) > q_j(x)$, then $y = i$, otherwise $y = j$
- Alternatively:
 - If ratio $l(x) = \frac{P(x|y=i)}{P(x|y=j)} > \frac{P(y=j)}{P(y=i)}$, then $y = i$, otherwise $y = j$
 - Or look at the log-likelihood ratio $h(x) = -\ln \frac{q_i(x)}{q_j(x)}$

Example: Gaussian class conditional distribution

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



- Decision boundary: $h(x) = -\ln \frac{q_i(x)}{q_j(x)} = 0$

Naïve Bayes Classifier

- Use Bayes decision rule for classification

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

- But assume $p(x|y = 1)$ is fully factorized

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

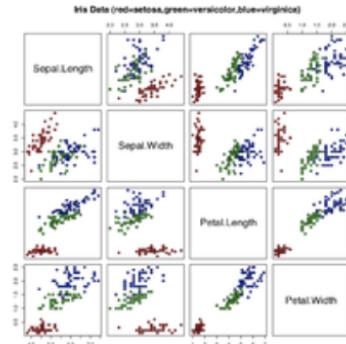
- Or the variables corresponding to each dimension of the data are independent given the label

Fisher's Iris Example

- The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by Sir Ronald Fisher (1936) as an example of discriminant analysis.
- The data set consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimetres.
- Based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other



- Demo:



Flower:

<https://wnellie.tumblr.com/post/143155937907/analysis-of-the-famous-iris-flower-dataset-part>

Fisher:

<https://www.umass.edu/wsp/resources/tales/fisher.html>

Data set image :

https://en.wikipedia.org/wiki/Iris_flower_data_set

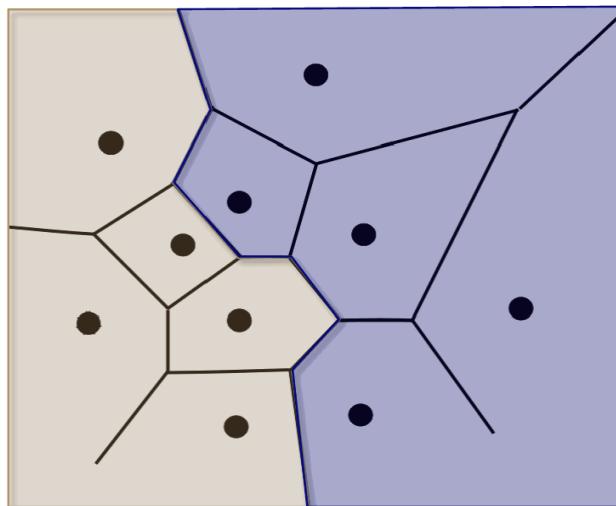
Classification algorithms

- Bayes classifier
- K-nearest neighbors
- Logistic regression

(more to come)

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



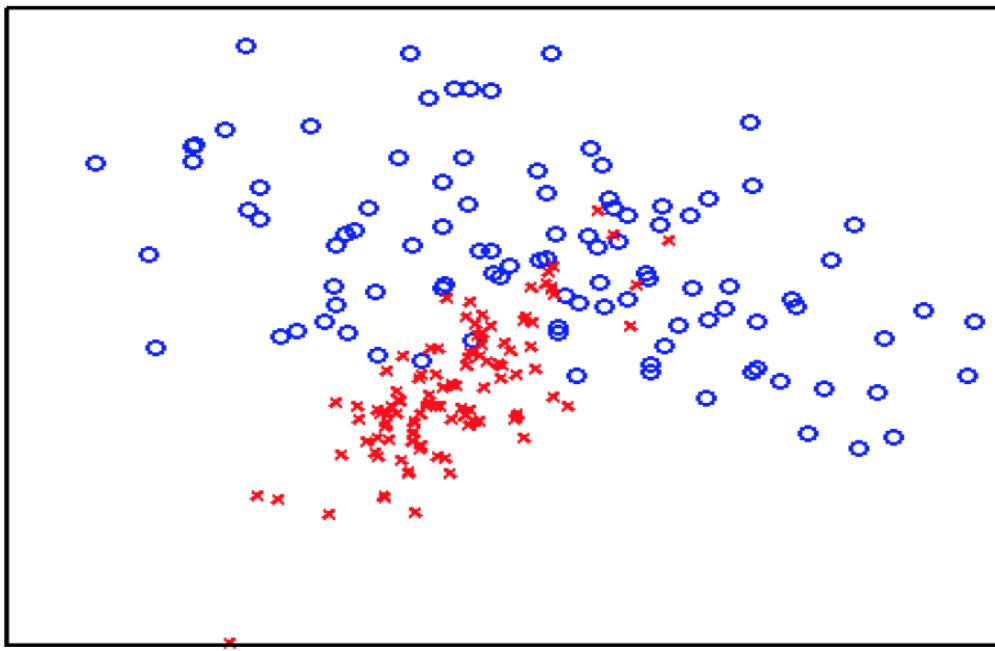
- Nonparametric
- Nonlinear
- Easy to kernelize

K -nearest neighbors

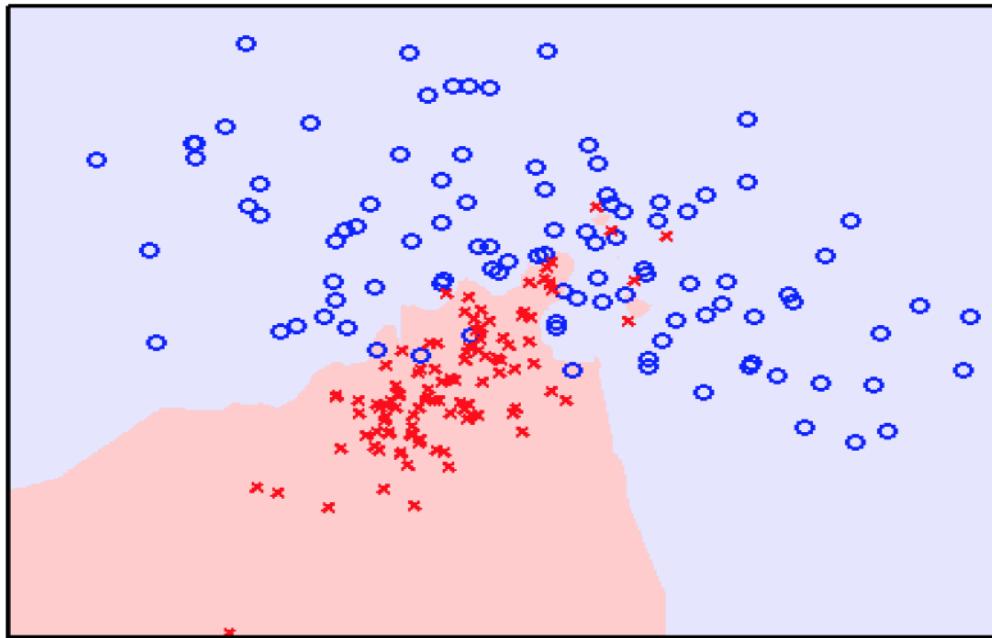
- **k -nearest neighbor classifier:** assign x a label by taking a majority vote over the k training points x_i closest to x
- For $K \geq 1$, the k -nearest neighbor rule generalizes the nearest neighbor rule
- To define this more mathematically:
 - $I_k(x) :=$ indices of the k training points closest to x .
 - If $y_i = \pm 1$, then we can write the k -nearest neighbor classifier as:

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

Example

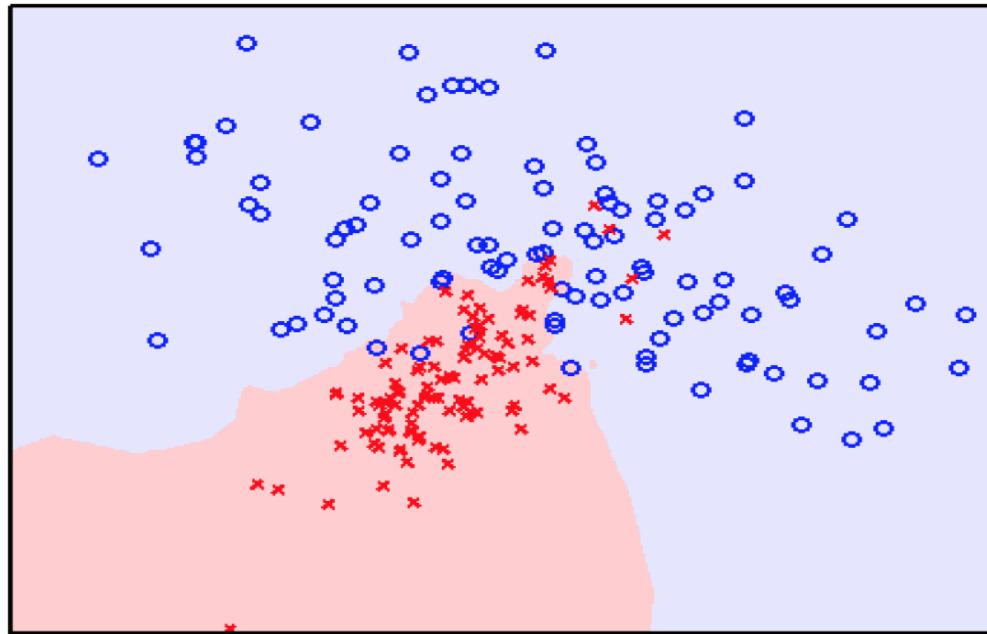


Example



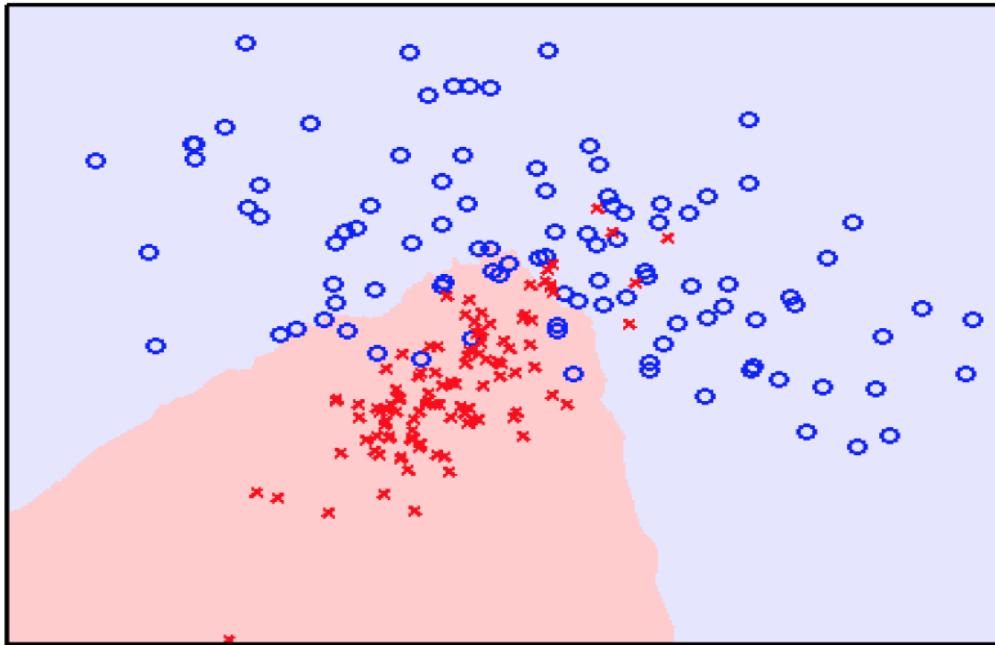
$$K = 3$$

Example



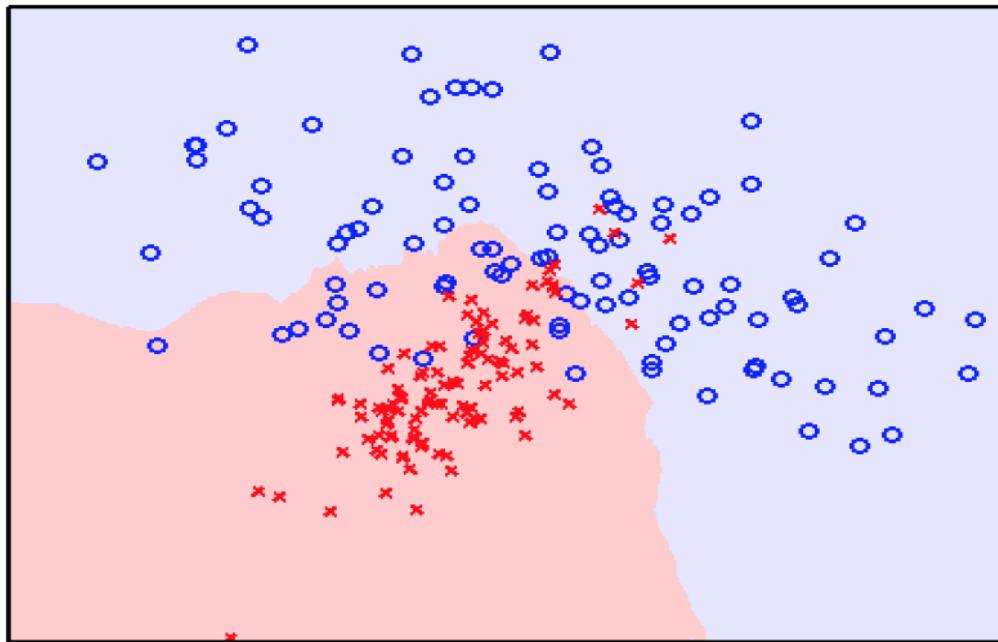
$$K = 5$$

Example



$$K = 25$$

Example



$$K = 51$$

Classification algorithms

- Bayes classifier
- K-nearest neighbors
- Logistic regression

(more to come)

Classification from a generative model

- Assume data are from a generative model determined by
 - For each data point
 - Sample a label y^i according to a prior $p(y), y = 0, 1$
 - Sample the value of x^i from the conditional probability $p(x|y^i, \theta)$
- Logistic regression: specify $p(x|y^i, \theta)$ using logistic function
 - logistic regression model

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

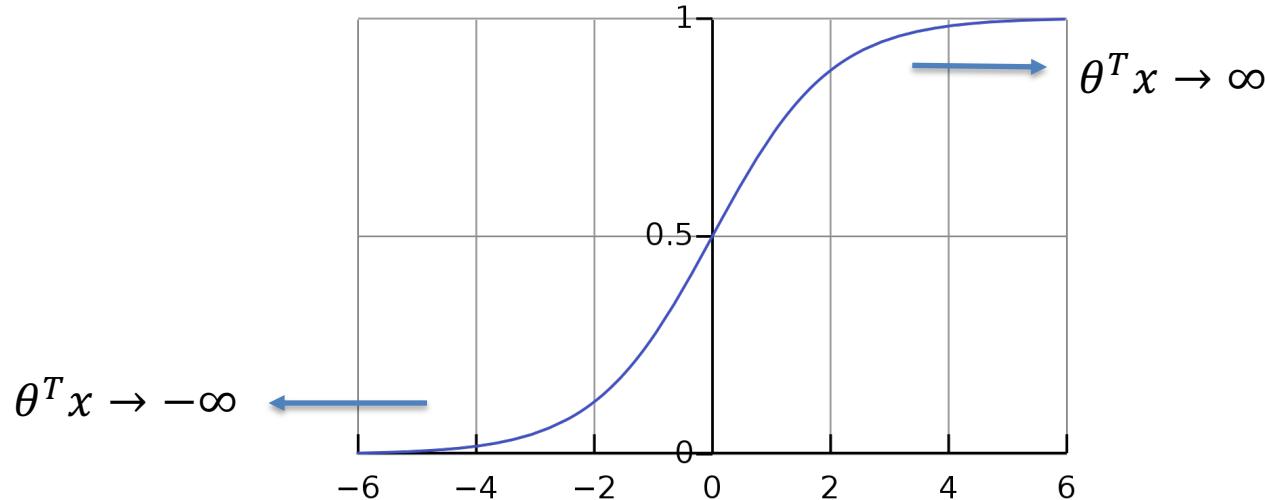
- Note that

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

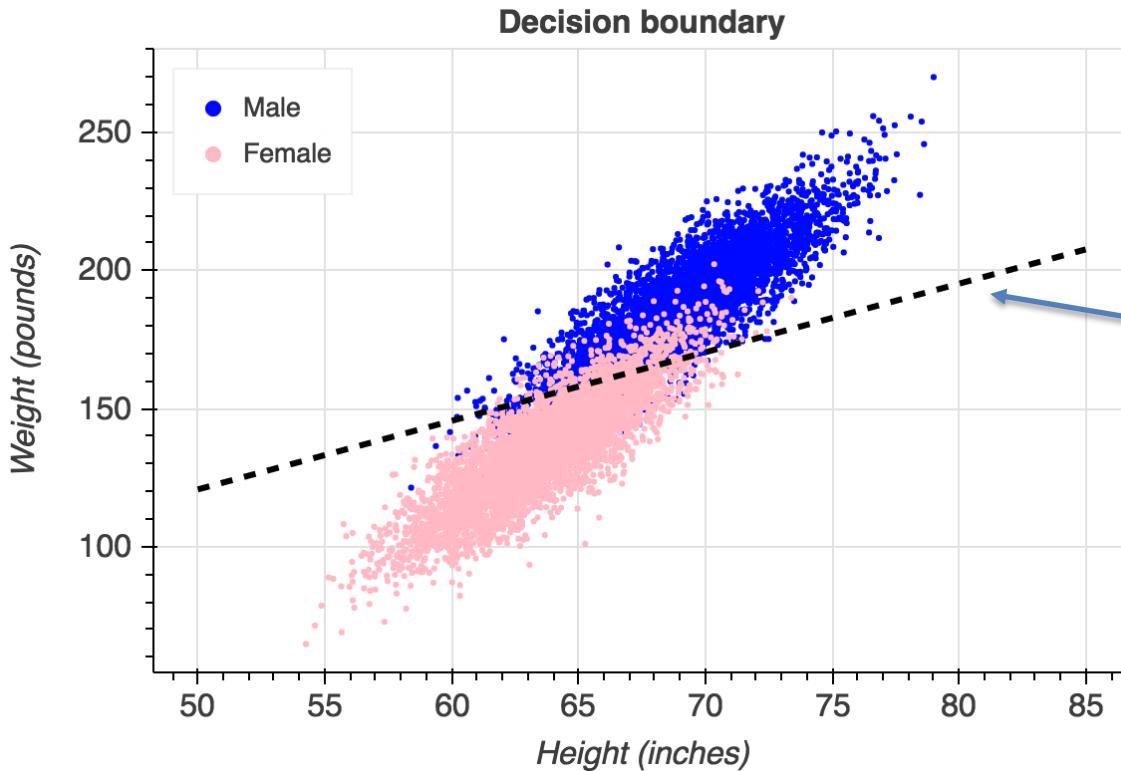
A closer look at logistic regression model

- logistic regression model

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



Decision boundary of logistic regression



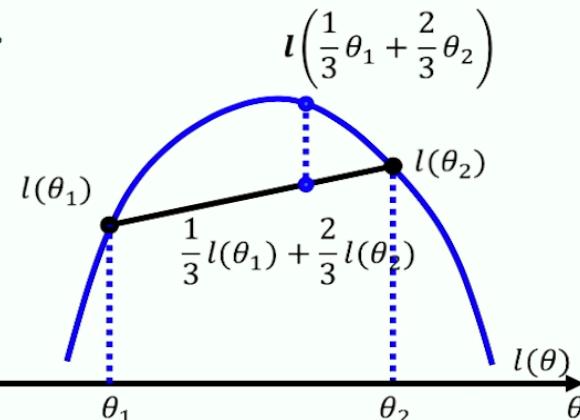
Learning in logistic regression is to find θ to optimally separate the classes in training data

Learning parameters in logistic regression

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

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

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



- Bad news: no closed form solution (resort to numerical method)

Learning parameters in logistic regression

- 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

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

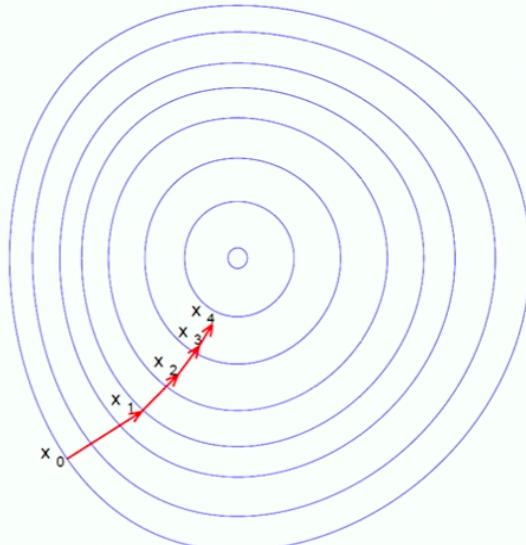
$$= \sum_i (y^i - 1) \theta^\top x^i - \log(1 + \exp(-\theta^\top x^i))$$

Gradient descent to minimize $f(x)$

- One way to solve an *unconstrained* optimization problem is gradient descent
- 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

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

γ_k is called the step size or learning rate



Gradient Ascent/Descent Algorithm

$$\begin{aligned} \max_{\theta} l(\theta) &:= \log \prod_{i=1}^m P(y^i | x^i, \theta) \\ &= \sum_i (y^i - 1) \theta^\top x^i - \log(1 + \exp(-\theta^\top x^i)) \end{aligned}$$

- Initialize parameter θ^0
Negative gradient
 $\nabla l(\theta)$ since we are
solving maximization
- Do

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

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

Batch gradient vs stochastic gradient

Note that gradient **decouples**: consists of sum of evaluations over individual samples

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

Batch gradient descent

- To compute the gradient at *each* iteration, we need to sum over *all* data points in the dataset
- What if we have a huge dataset? For example, 1 Million data points?

Stochastic gradient descent

- At each iteration, we randomly sample a small subset S_k data point (x_i, y_i) , $i \in S_k$ (in the extreme case, just one sample in S_k), and update using gradient estimated using a small subset of data

$$\nabla \hat{l}(\theta) = \sum_{i \in S_k} (y^i - 1) \theta^\top x^i - \log(1 + \exp(-\theta^\top x^i))$$

- Using a different subset each time, so eventually loop through entire training data

Stochastic gradient descent (SGD) vs. Batch Gradient Descent (GD)

- Time complexity: $O(1)$ (SGD) vs $O(n)$ (GD)
- The key thing is that the gradient is unbiased, thus the expectation equals the true gradient

Step sizes and stopping criterion

- For SGD, due to noise introduced by random sampling data, need to use decreasing step size, such as $O\left(\frac{1}{k}\right)$
 - Step size too small: not making too much progress
 - Step size too large: overshoot
- The optimality condition is $\nabla l(\theta) = 0$

Stop when the gradient becomes small $\|\nabla l(\theta)\|$, e.g., l -2 norm

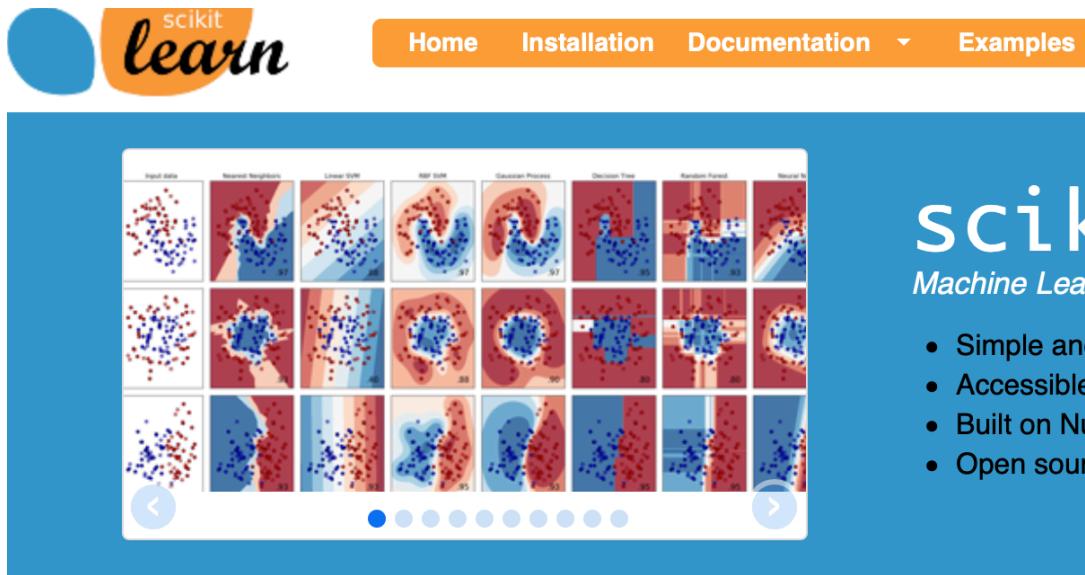
Comparing Bayes classifier and logistic regression

- Consider $y \in \{1, -1\}$, $x \in R^n$
- Number of parameters
 - Naïve Bayes : $2n + 1$
 - When all random variables are binary
 - $4n+1$ for Gaussians: $2n$ mean, $2n$ variance, and 1 for prior
 - logistic regression: $n + 1$
 - $\theta_0, \theta_1, \theta_2, \dots, \theta_n$

- Asymptotic comparison (# training examples → infinity)
- When model assumptions correct
 - Naïve Bayes, logistic regression produce identical classifiers
- When model assumptions incorrect
 - logistic regression is less biased – does not assume conditional independence
 - logistic regression has fewer parameters
 - therefore expected to outperform Naïve Bayes

- Estimation method:
 - Naïve Bayes parameter estimates are decoupled (super easy)
 - Logistic regression parameter estimates are coupled (less easy)
- How to estimate the parameters in logistic regression?
 - Maximum likelihood estimation
 - More specifically, maximize the conditional likelihood the label

Implementation



The screenshot shows the official scikit-learn website. At the top, there is a navigation bar with links for Home, Installation, Documentation, Examples, and a Google Custom Search bar with a magnifying glass icon. Below the navigation bar is a large blue banner featuring the text "scikit-learn" in white, followed by "Machine Learning in Python". To the left of the banner is a grid of nine small plots, each showing a different machine learning model's performance on a 2D dataset. The plots include "Input data", "Nearest Neighbors", "Linear SVM", "RBF SVM", "Gaussian Process", "Decision Tree", "Random Forest", and "Naive Bayes". Below the banner is a bulleted list of features:

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Look at logistic regression:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

Nearest neighbors:

<https://scikit-learn.org/stable/modules/neighbors.html>

Naïve Bayes:

https://scikit-learn.org/stable/modules/naive_bayes.html

Demo: Statlog (Heart) Data Set

- 13 attributes (see heart.docx for details)
 - 2 demographic (age, gender)
 - 11 clinical measures of cardiovascular status and performance
- 2 classes: absence (1) or presence (2) of heart disease
- 270 samples
- Dataset taken from UC Irvine Machine Learning Repository
- Demo code



