MAT8034: Machine Learning

Generative Learning Algorithms

Fang Kong

https://fangkongx.github.io/Teaching/MAT8034/Spring2025/index.html

Course Outline

Week	Content	Week	Content
1	Introduction; Linear regression	9	PCA, ICA (Hw2)
2	Logistic regression; Exponential family, GLM	10	MDPs I
3	Generative learning	11	MDPs II; Bandits
4	Kernel methods, SVM (Project)	12	RL I
5	Deep learning (Hw1)	13	RL II (Hw3)
6	Generalization	14	Ethics of AI
7	Regularization and model selection	15	Final review
8	K-means, EM	16	Presentation

Coding practice

- 动手学习机器学习
  - https://hml.boyuai.com/books
- 动手学习强化学习
  - https://hrl.boyuai.com/

Outline

- Generative learning algorithms
  - Motivation/Intuition
  - Gaussian discriminant analysis
  - Naïve Bayes

Motivation

Motivation

- In previous methods: Linear/logistic regression
  - Model $p(y|x;\theta)$

- Example
  - Consider the image classification with elephants (y = 1) and dogs (y = 0)
  - Observe some features of an animal
  - Previous methods
    - Find a decision boundary—that separates the elephants and dogs
    - When new animal comes, check on which side of the decision boundary it falls

Motivation

- Example
  - Consider the image classification with elephants ($y = 1$) and dogs ($y = 0$)
  - Observe some features of an animal
  - Previous methods
    - Find a decision boundary—that separates the elephants and dogs
    - When new animal comes, check on which side of the decision boundary it falls
  - New methods
    - Build a model of what elephants/dogs look like
    - To classify a new animal, we can match the new animal against the elephant model, and match it against the dog model, to see whether the new animal looks more like the elephants or more like the dogs

Discriminative and generative learning algorithms

- Discriminative learning algorithms
  - Try to learn $p(y|x)$
- Generative learning algorithms
  - Try to learn $p(x|y)$ and also $p(y)$

  - Example
  - $p(x|y = 1)$ models the distribution of elephants' features
  - $p(x|y = 0)$ models the distribution of dogs' features

Generative learning algorithms

- Use Bayes rule
  - $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$
- Prediction
  - $\mathrm{argmax}_{\mathrm{y}} p(y|x) = \mathrm{argmax}_{\mathrm{y}} p(x|y)p(y)$

Gaussian discriminant analysis

Gaussian discriminant analysis

- Assume that $p(x|y)$ is distributed according to a multivariate Gaussian distribution

Multivariate Gaussian distribution

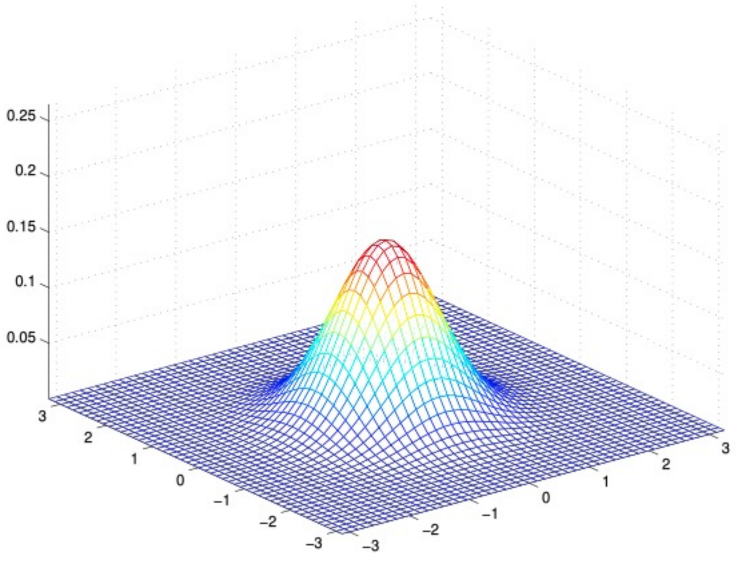
- d-dimension
- Mean vector $\mu \in \mathbb{R}^d$
- Covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$ (symmetric, positive semi-definite)

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

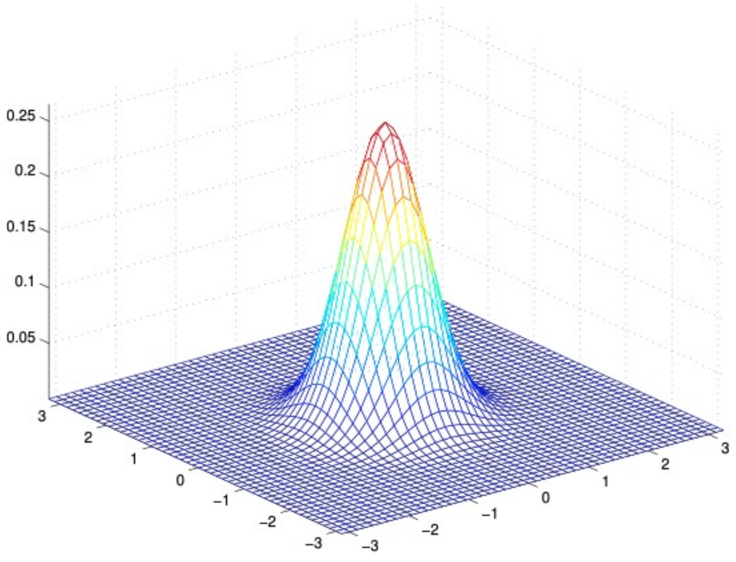
  - $|\Sigma|$ denotes the determinant of the matrix Σ
- Expectation and covariance

$$\mathrm{E}[X] = \int_x x\, p(x; \mu, \Sigma) dx = \mu \qquad \mathrm{E}[(Z - \mathrm{E}[Z])(Z - \mathrm{E}[Z])^T]$$

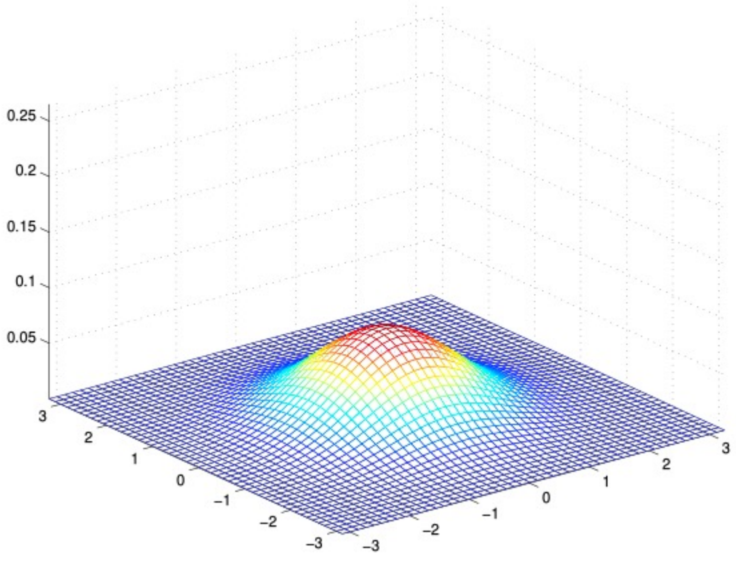
Illustration – 2d



$\mu = (0,0), \Sigma = I$

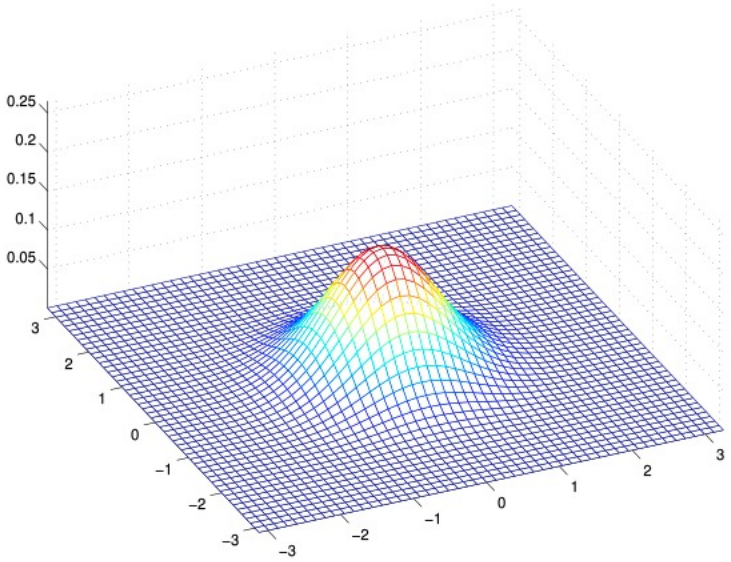


$\mu = (0,0), \Sigma = 0.6I$



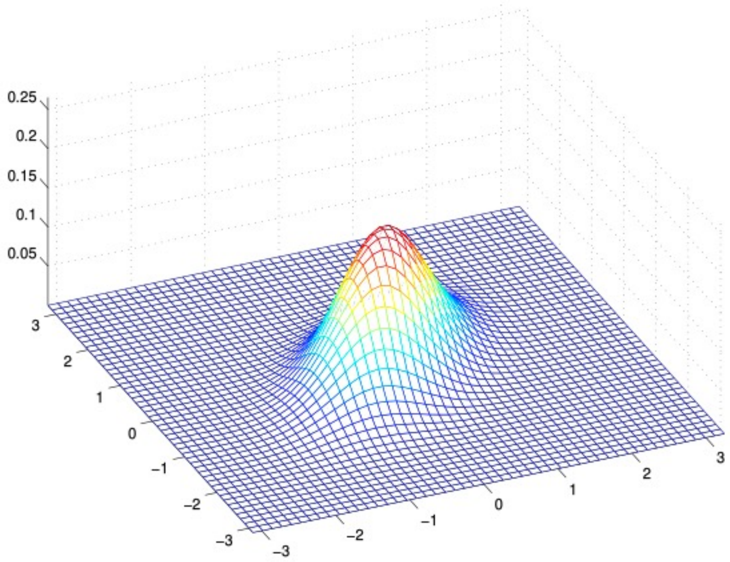
$\mu = (0,0), \Sigma = 2I$

Illustration – 2d



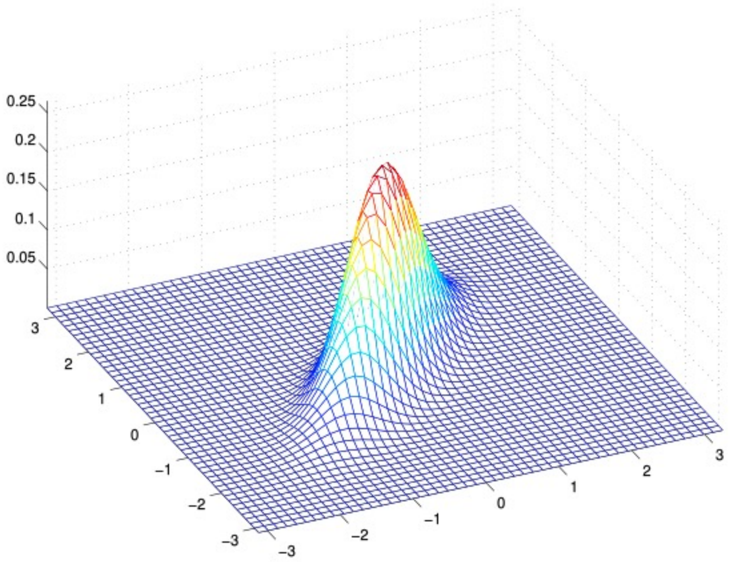
$\mu = (0,0)$

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$\mu = (0,0)$

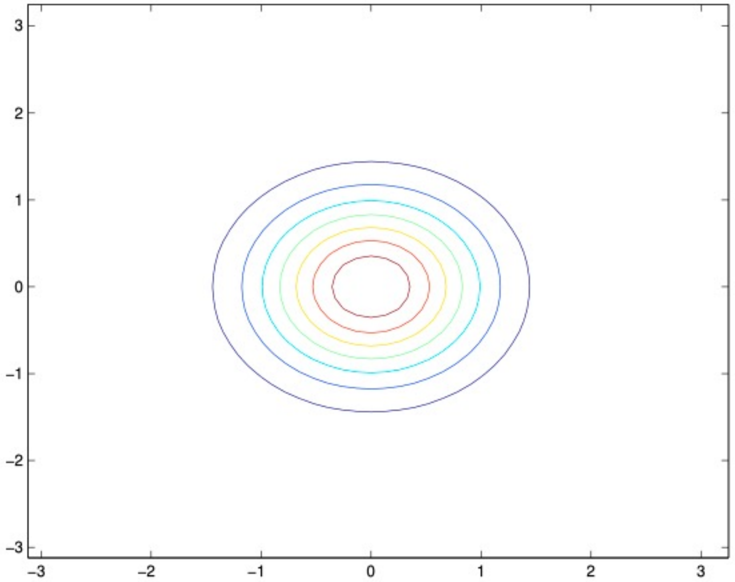
$$\Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

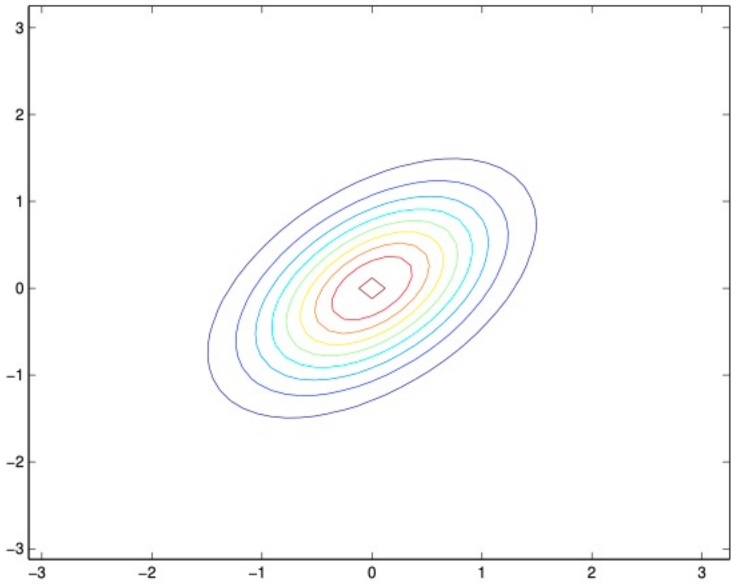


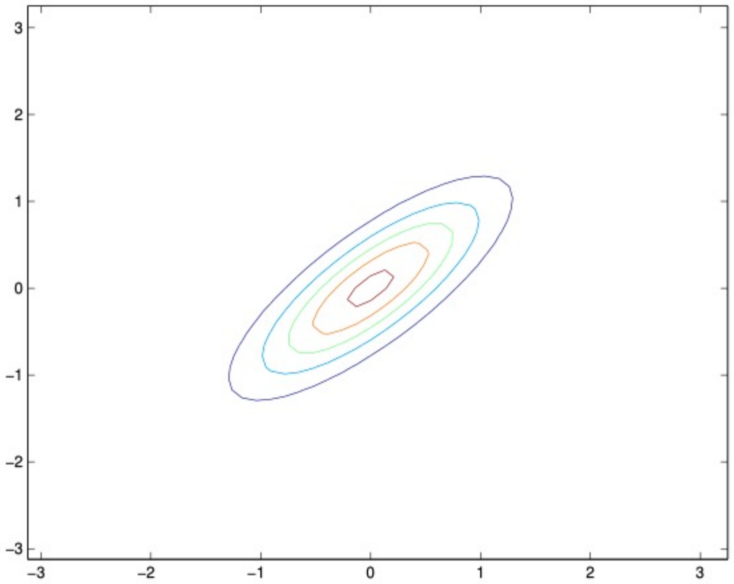
$\mu = (0,0)$

$$\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

Illustration – 2d







$\mu = (0,0)$

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

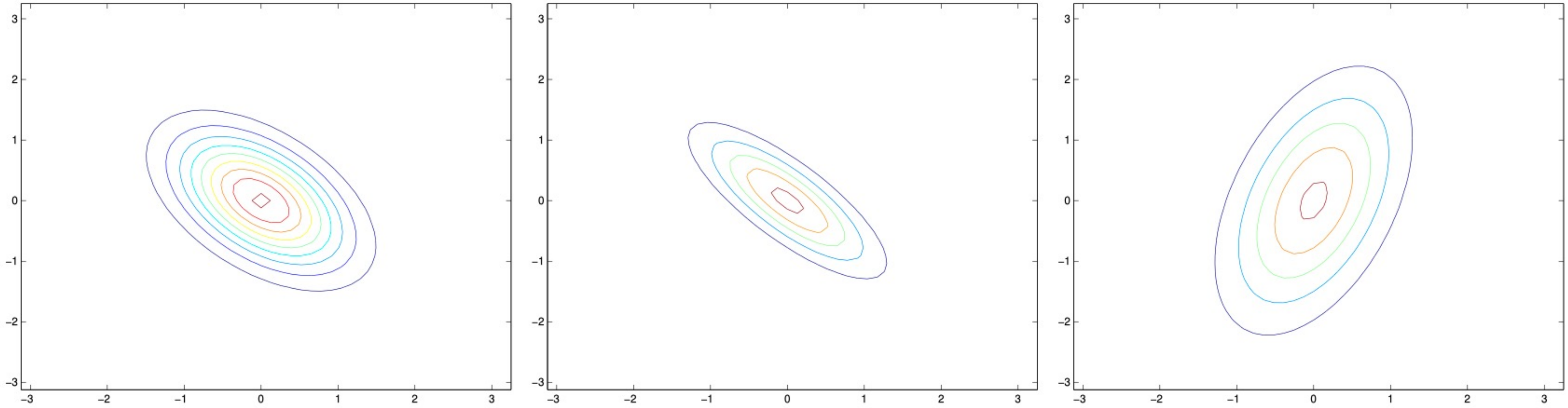
$\mu = (0,0)$

$$\Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$\mu = (0,0)$

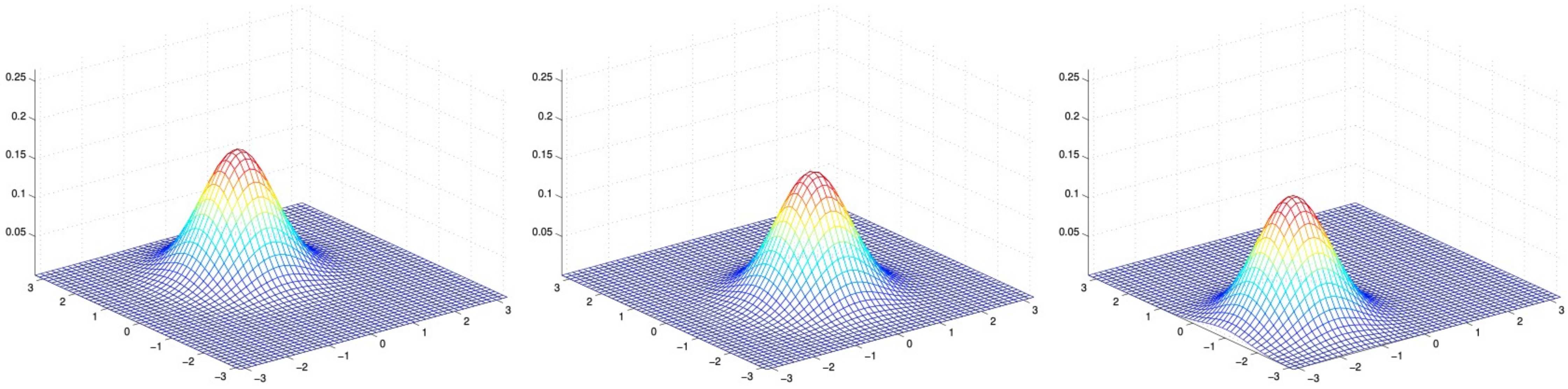
$$\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

Illustration – 2d



$$\Sigma = \begin{bmatrix} 1 & \text{-}0.5 \\ \text{-}0.5 & 1 \end{bmatrix}; \;\; \Sigma = \begin{bmatrix} 1 & \text{-}0.8 \\ \text{-}0.8 & 1 \end{bmatrix}; \;\; \Sigma = \begin{bmatrix} 3 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

Illustration – 2d



$$\mu = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \;\; \mu = \begin{bmatrix} \text{-}0.5 \\ 0 \end{bmatrix}; \;\; \mu = \begin{bmatrix} \text{-}1 \\ \text{-}1.5 \end{bmatrix}$$

The GDA model

- Model $p(x|y)$ using a multivariate normal distribution

$$
\begin{aligned}
y &\sim \text{Bernoulli}(\phi) \\
x|y=0 &\sim \mathcal{N}(\mu_0, \Sigma) \\
x|y=1 &\sim \mathcal{N}(\mu_1, \Sigma)
\end{aligned}
$$

- Probabilities with

$$
\begin{aligned}
p(y) &= \phi^y (1-\phi)^{1-y} \\
p(x|y=0) &= \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (x-\mu_0)^T \Sigma^{-1} (x-\mu_0) \right) \\
p(x|y=1) &= \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (x-\mu_1)^T \Sigma^{-1} (x-\mu_1) \right)
\end{aligned}
$$

How to estimate the parameters?

- The parameters are φ, Σ, μ_0 and μ_1 (Usually assume common Σ)
- The log-likelihood function

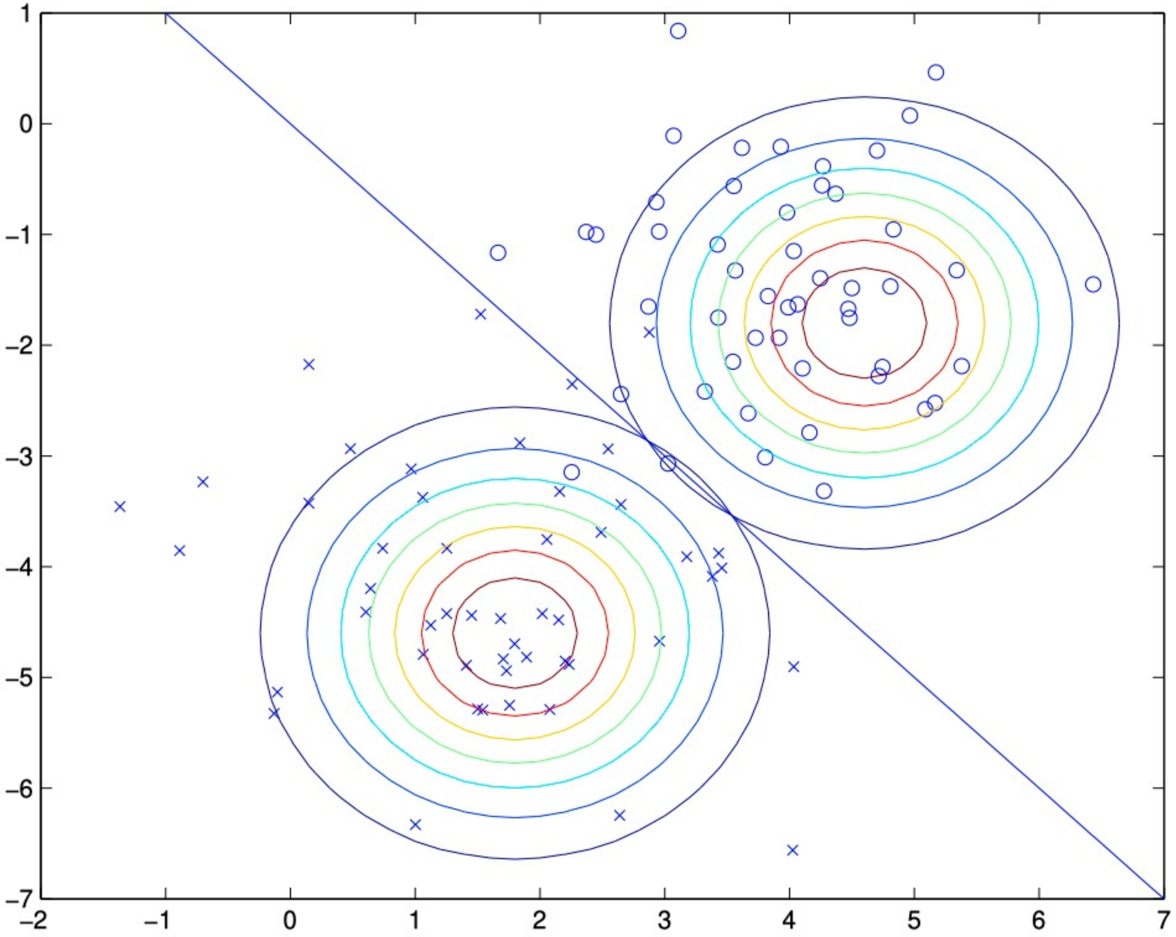
$$
\begin{aligned}
\ell(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^{n} p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\
&= \log \prod_{i=1}^{n} p(x^{(i)} | y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi).
\end{aligned}
$$

Maximum likelihood

- Maximum likelihood yields the result (see the offline derivation)

$$
\begin{aligned}
\phi &= \frac{1}{n}\sum_{i=1}^{n} 1\{y^{(i)} = 1\} \\
\mu_0 &= \frac{\sum_{i=1}^{n} 1\{y^{(i)} = 0\} x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)} = 0\}} \\
\mu_1 &= \frac{\sum_{i=1}^{n} 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}} \\
\Sigma &= \frac{1}{n}\sum_{i=1}^{n} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T.
\end{aligned}
$$

Illustration of GDA



Decision boundary: $p(y = 1|x) = 0.5$

GDA V.S. logistic regression

- GDA has the same form as logistic regression

$$p(y=1|x;\phi,\Sigma,\mu_0,\mu_1)=\frac{1}{1+\exp(-\theta^T x)}$$

- How?
  - (see the offline derivation)

Which one is better?

- GDA assume multivariate normal distribution
- p(y|x) being a logistic function does not imply p(x|y) follows multivariate normal distribution
  - Other distributions can also yield this form
- GDA makes stronger modeling assumptions
  - When the modeling assumptions are (approximate) correct
  - Is more data efficient
- Logistic regression makes weaker assumptions
  - More robust to deviations from modeling assumptions
  - When the data is indeed non-Gaussian, logistic regression is better
- In practice logistic regression is used more often

Naïve Bayes

Motivation

- In GDA, the feature vectors x were continuous, real-valued vectors.
- Now consider a different learning algorithm in which the x is discrete-valued.

Example: Spam Filter

- Input: an email
- Output: spam/non-spam

- Setup:
  - Get a large collection of example emails, each labeled "spam" or "ham"
  - Want to learn to predict labels of new, future emails

- Features:

$$x = \begin{array}{cl} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} & \begin{array}{l} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \text{buy} \\ \vdots \\ \text{zygmurgy} \end{array} \end{array}$$

  - Dimension: the number of words in the vocabulary
  - $x_j = 1$ if the email contains the j-th word

Dear Sir.

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. …

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES
FOR ONLY $99

Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Build the model

- Suppose there are 5000 words
- $x \in \{0,1\}^{5000}$
- 2^{5000} possible outcomes
- Hard to compute $p(x_1, x_1, \ldots, x_{5000} \mid y)$

- To make it easier
  - Assume conditionally independence

Naive Bayes (NB) assumption

- Intuition
  - If $y = 1$ means spam email
  - I tell you $y = 1$, then knowledge of x_{2087} (whether “buy” appears in the message) will have no effect on your beliefs about the value of x_{39831} (whether “price” appears)
  - i.e., $\mathrm{p}(x_{2087}|y) = \mathrm{p}(x_{2087}|y, x_{39831})$
- Compute the joint distribution

$$
\begin{aligned}
& p(x_1, \ldots, x_{50000}|y) \\
& \quad = \quad p(x_1|y)p(x_2|y, x_1)p(x_3|y, x_1, x_2) \cdots p(x_{50000}|y, x_1, \ldots, x_{49999}) \\
& \quad = \quad p(x_1|y)p(x_2|y)p(x_3|y) \cdots p(x_{50000}|y) \\
& \quad = \quad \prod_{j=1}^{d} p(x_j|y)
\end{aligned}
$$

Estimate the parameters

- Parameters
  - $\phi_{j|y=1} = p(x_j = 1 \mid y = 1), \phi_{j|y=0} = p(x_j = 1 \mid y = 0)$
  - $\phi_y = p(y = 1)$
- Given n training data, the joint likelihood is

$$\mathcal{L}(\phi_y, \phi_{j|y=0}, \phi_{j|y=1}) = \prod_{i=1}^{n} p(x^{(i)}, y^{(i)}).$$

- Maximum likelihood yields the result

Estimate the parameters

- Maximum likelihood yields the result

$$\phi_{j|y=1} = \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}}$$

$$\phi_{j|y=0} = \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 0\}}$$

$$\phi_y = \frac{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}}{n}$$

Make predictions

- Given a new example with feature x
- Compute the probability

$$
\begin{aligned}
p(y=1|x) &= \frac{p(x|y=1)p(y=1)}{p(x)} \\
&= \frac{\left(\prod_{j=1}^{d} p(x_j|y=1)\right) p(y=1)}{\left(\prod_{j=1}^{d} p(x_j|y=1)\right) p(y=1) + \left(\prod_{j=1}^{d} p(x_j|y=0)\right) p(y=0)}
\end{aligned}
$$

Extension

- x_j can take values in $\{1, 2, \ldots, k_j\}$
  - Model $p(x_j|y)$ as multinomial rather than as Bernoulli
- x_j is continuous valued
  - Discretize it
  - E.g., for house price prediction

Living area (sq. feet)	< 400	400-800	800-1200	1200-1600	>1600
x_i	1	2	3	4	5

  - When the original, continuous-valued attributes are not well-modeled by a multivariate normal distribution, discretizing the features and using Naive Bayes (instead of GDA) will often result in a better classifier

Laplace smoothing

Problem of NB

- Suppose "machine learning" is the 3500th word
- But this is no email containing this word in the training set
- What happens?

$$p(y=1|x) \quad = \quad \frac{\prod_{j=1}^{d} p(x_j|y=1)p(y=1)}{\prod_{j=1}^{d} p(x_j|y=1)p(y=1) + \prod_{j=1}^{d} p(x_j|y=0)p(y=0)}$$

- But…

$$\phi_{35000|y=1} \quad = \quad \frac{\sum_{i=1}^{n} 1\{x_{35000}^{(i)}=1 \wedge y^{(i)}=1\}}{\sum_{i=1}^{n} 1\{y^{(i)}=1\}} = 0$$

$$\phi_{35000|y=0} \quad = \quad \frac{\sum_{i=1}^{n} 1\{x_{35000}^{(i)}=1 \wedge y^{(i)}=0\}}{\sum_{i=1}^{n} 1\{y^{(i)}=0\}} = 0$$

Problem of NB

- Then $p(\, y = 1 \mid x\,) = \frac{0}{0}$
- How to make a prediction?
- Probability 0?
  - It is a bad idea to estimate the probability of some event to be zero just because you haven't seen it before in your finite training set

How to solve it?

- Suppose we are estimating the mean of a multinomial random variable z taking values in {1, . . . , k} with $\phi_j = p(z = j)$
- Previous maximum likelihood methods show

$$\phi_j = \frac{\sum_{i=1}^{n} 1\{z^{(i)} = j\}}{n}$$

- Some ϕ_j may be zero since it never appears
- To avoid this, give small probability for those non-appeared words

$$\phi_j = \frac{1 + \sum_{i=1}^{n} 1\{z^{(i)} = j\}}{k + n}$$

Laplace smoothing

$$\phi_j = \frac{1 + \sum_{i=1}^{n} 1\{z^{(i)} = j\}}{k + n}$$

- It still guarantees $\sum_j \phi_j = 1$
- Also guarantees $\phi_j \neq 0$ for all j
- Estimation for binary cases

$$\phi_{j|y=1} = \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 1\}}$$

$$\phi_{j|y=0} = \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 0\}}$$

Summary

- Generative learning algorithms
  - Motivation/Intuition
  - Gaussian discriminant analysis
    - Multivariate normal distribution
    - Model
    - Discussion: GDA and logistic regression
  - Naïve Bayes
    - Model
    - Laplace smoothing