NYC DATA SCIENCE ACADEMY

Python Machine Learning Class 2: Classification Part I

NYC Data Science Academy

Outline

- Limitation of Linear Regression
  - Logistic Regression
  - Discriminant Analysis: Motivation
  - Discriminant Analysis: Models
    - One Dimensional Cases
    - Higher Dimensional Cases
  - Naive Bayes

Classification Problems

- *Categorical* (qualitative) variables: takes values in a finite set (usually unordered).
  - email: {spam, non-spam}
  - blood type: {A, B, AB, O}
  - tumor: {malignant, benign}
- *Classification*: given a feature (or a set of features), we want to predict categorical output.
- Sometimes people are also interested in estimating the probabilities that *an observation* belongs to each category.

A Classification Example

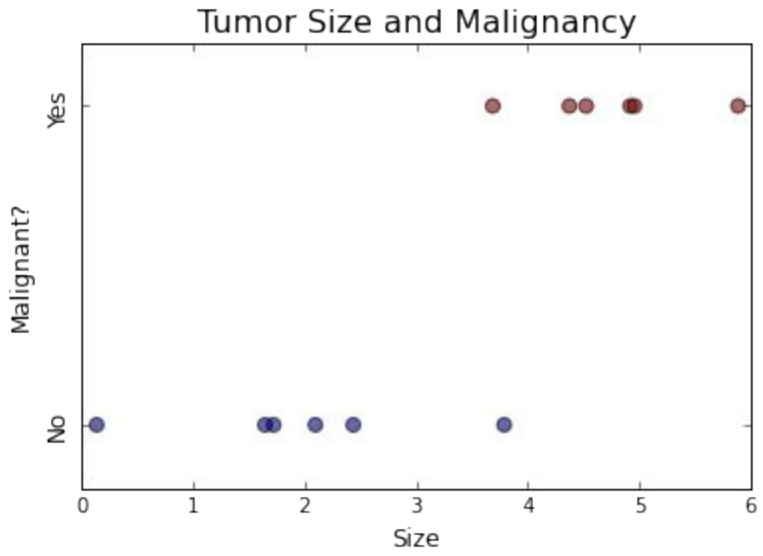
- Predict whether a tumor is malignant or benign based on the tumor size
- The output is binary:
  - 0: benign
  - 1: malignant
- Here is a simulated data set:

	Size	Malignant
0	3.788628	0
1	2.436510	0
2	2.096497	0
3	0.136507	0
4	1.722612	0
5	1.645241	0

	Size	Malignant
6	4.917259	1
7	4.372999	1
8	4.956182	1
9	4.522782	1
10	3.686135	1
11	5.884622	1

A Classification Example

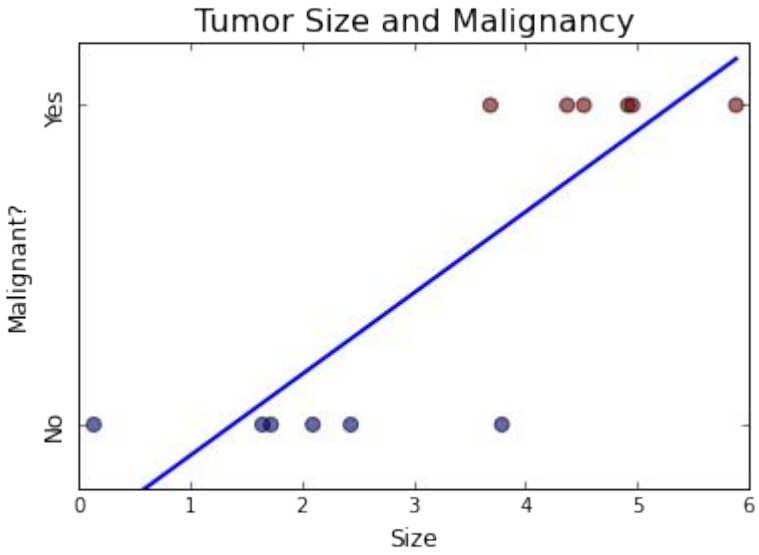
- By selecting the Size as feature and Malignant as output, we can visualize the data as:



- Question: Is linear regression suitable for the classification task?

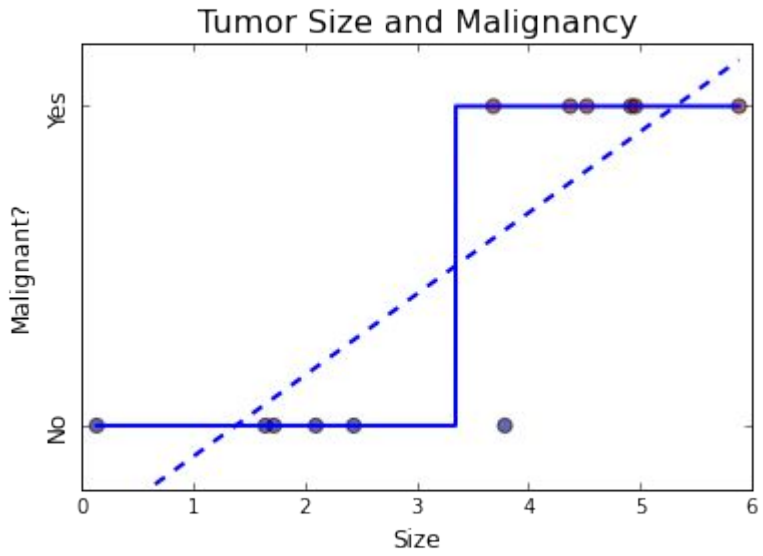
Can We Use Linear Regression?

- Let's fit a linear regression model with the simulated tumor data:



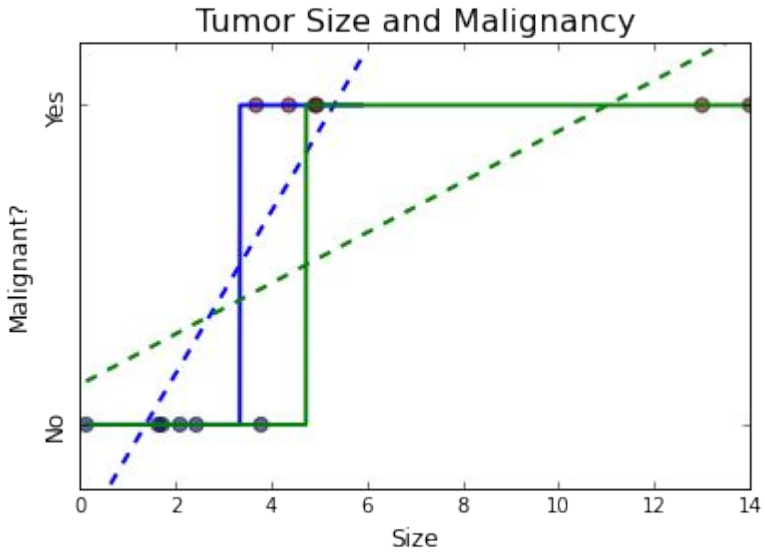
Is Linear Regression suitable for classification?

- We may then set a threshold
  - Predict 1 if $\hat{y} \geq 0.5$
  - Predict 0 if $\hat{y} < 0.5$
- The predicted values become binary:



Issues with Linear Regression

- It looks like the binary prediction with linear regression is not a bad idea. However, we do have the following two issues:
  - the continuous output often exceeds the unit interval [0, 1]. Therefore we cannot interpret it as a probability.
  - the prediction can be affected by outliers easily.



Outline

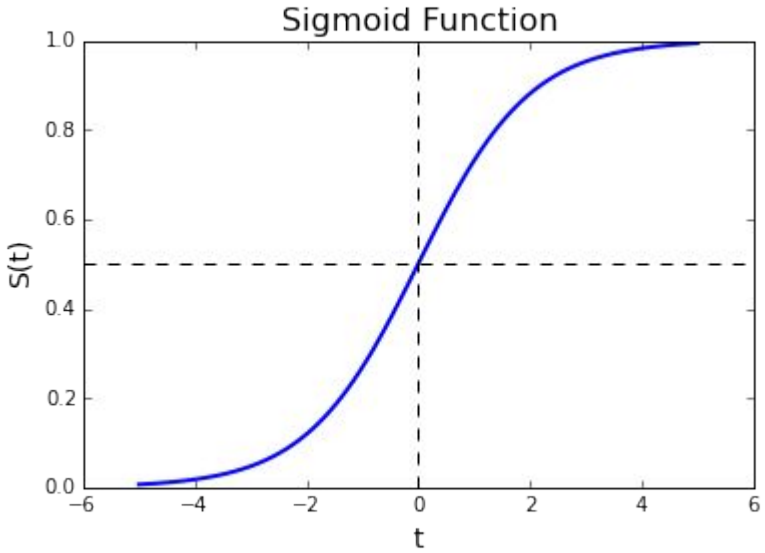
- Limitation of Linear Regression
- **Logistic Regression**
- Discriminant Analysis: Motivation
- Discriminant Analysis: Models
  - One Dimensional Cases
  - Higher Dimensional Cases
- Naive Bayes

Sigmoid Function

- Sigmoid Function: a monotonically increasing smooth function which maps a real value to a positive value bounded between 0 and 1.

$$S(t) = \frac{e^t}{1 + e^t}$$

- ***e*** = 2.718 is a mathematical constant (Euler's number).



Logistic Regression

Logistic regression, despite its name, is a linear model for classification rather than regression.

- Idea: if we transform the linear function $\beta_0+\beta_1 X$ using the sigmoid function $S(t)$, then no matter what values β_0, β_1 or X take, y will always have values between 0 and 1.
- *Logistic Regression models* use this form to estimate the probability that $y = 1$ given its size X:

$$Pr(Y = 1|X = x) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

- So different β_0 and β_1 will give different estimations Pr.

Maximum Likelihood

- Let's write the likelihood function

$$p(x_i, \beta_0, \beta_1) = Pr(Y = 1|X = x_i)$$

to describe the probability of observed outcomes to be of class 1 given $X = x_i$.

- Then, given an input X with n observations, the likelihood gives the probability of having the observations with the prescribed labels:

$$L(\beta_0, \beta_1) = \prod_{i,y_i=1} p(x_i, \beta_0, \beta_1) \prod_{i,y_i=0} (1-p(x_i, \beta_0, \beta_1))$$

where the first product gives the probability of successfully predicting the "1"s and the second product is the probability of successfully predicting the "0"s in the given data.

Maximum Likelihood

- The likelihood function $L(\beta_0, \beta_1)$ gives the probability of making the same prediction as the observed data.
- Among all the linear models, the pair with the higher L has a higher probability to produce the prescribed class labels.
- We want to pick β_0 and β_1 to maximize the likelihood $L(\beta_0, \beta_1)$, i.e., to maximize the "agreement" of the selected model with the observed data.

Log-Likelihood

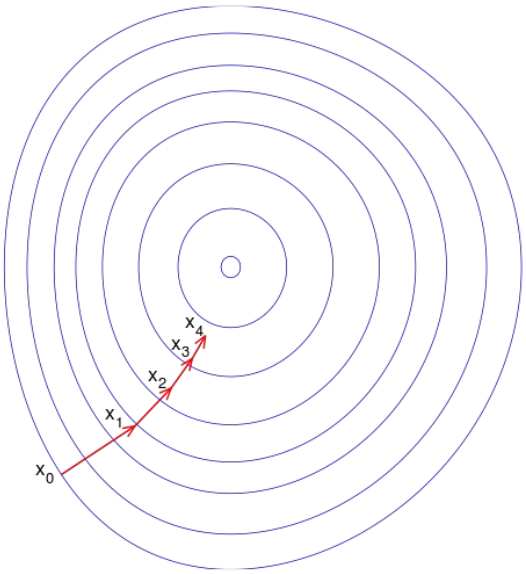
- In practice it is often more convenient to work with the logarithm of the likelihood function, called the **log-likelihood**:

$$\log L(\beta_0, \beta_1) = \sum_{i=1}^{n} \{y_i \log p(x_i, \beta_0, \beta_1) + (1 - y_i) \log(1 - p(x_i, \beta_0, \beta_1))\}$$
$$= \sum_{i=1}^{n} \{y_i(\beta_0 + \beta_1 X) - \log(1 + e^{\beta_0 + \beta_1 X})\}$$

- Logistic regression models are usually fitted by maximum likelihood, i.e., to find β_0 and β_1 that maximize the log likelihood function above.

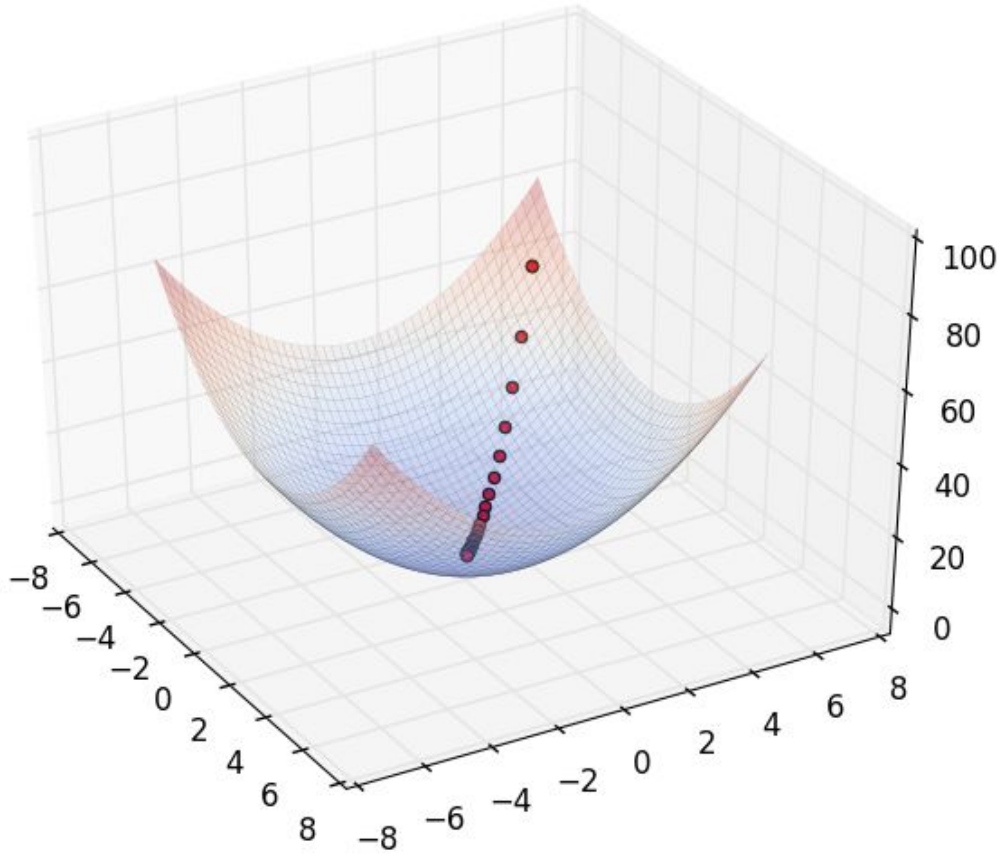
Gradient Descent

- To maximize the log-likelihood, most packages, including scikit-learn, use a numerical method called *gradient descent*, i.e., to find the maximum or minimum by search along a steepest path on the log-likelihood function.



source: https://en.wikipedia.org/wiki/Gradient_descent

Gradient Descent



Making Predictions

- After estimating the parameters, the likelihood function $p(x, \beta_0, \beta_1)$ predicts the probability of output Y to be 1 given x. If we set a threshold, then we can predict binary outputs.
- Let's consider using tumor size to predict malignancy:
  - Hypothetically if the maximum likelihood gives $Pr(Y=1|X=x) = 0.2$, then our prediction is 20% chance of tumor being malignant, or equivalently, 80% of change it's benign.
  - If we set the threshold, for example, to be 0.5, then we can predict the tumor to be benign.

Hands-on Session

- Please go to the **"Logistic Regression in Scikit-Learn"** in the lecture code.

Outline

- Limitation of Linear Regression
- Logistic Regression
- **Discriminant Analysis: Motivation**
- Discriminant Analysis: Models
  - One Dimensional Cases
  - Higher Dimensional Cases
- Naive Bayes

Conditional Probability

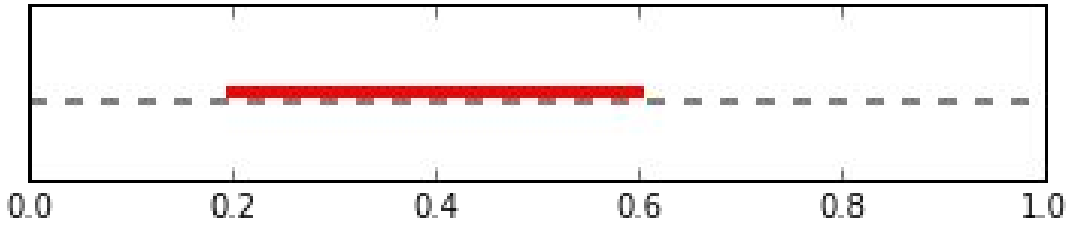
- Let *Y* be an event with probability *P*(*Y*) > 0, the *conditional probability* of observing *X* given that *Y* has occurred is defined as:

$$P(X|Y) = \frac{P(X,Y)}{P(Y)}$$

  - *P*(*X*, *Y*) refers to the joint probability that *X* and *Y* occur at the same time.
  - *P*(*X*|*Y*) is the probability of *X* after insuring *Y*'s occurrence.
  - P(X) may be different from P(X|Y)

Conditional Probability

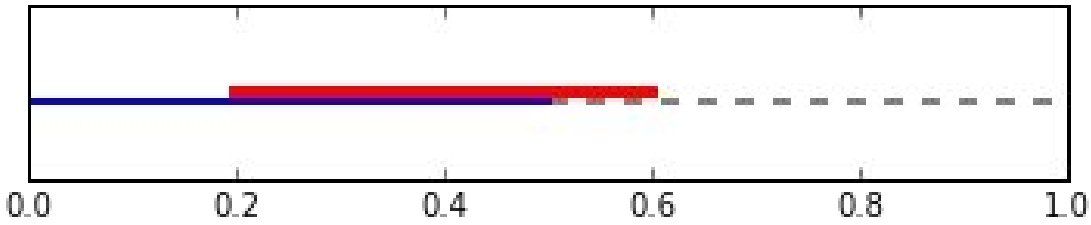
- Suppose that we draw a random number uniformly distributed in the unit interval [0,1]
- How do we compute the probability of obtaining a number in the red region?



$$0.4 \div 1 = 0.4$$

Conditional Probability

- How about the probability of obtaining a number in the red region when restricted in the blue region?



$$\frac{P(\text{red, blue})}{P(\text{blue})} = \frac{0.3}{0.5} = 0.6$$

Independent Events

- If X and Y are independent, $P(X, Y) = P(X)P(Y)$:
  - Then the conditional probability is

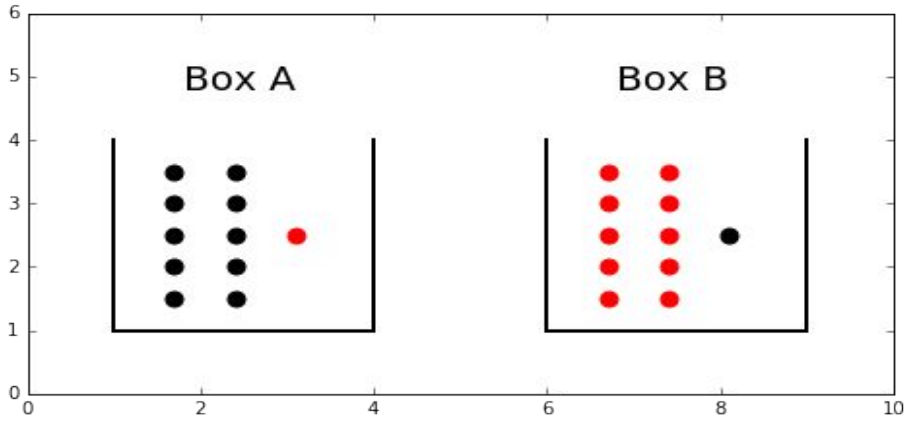
$$P(X|Y) = \frac{P(X,Y)}{P(Y)} = \frac{P(X)P(Y)}{P(Y)} = P(X)$$

  - This implies that the occurrence of Y does not have any impact on the occurrence of X.

Conditional Probability Example

Consider an experiment of picking balls of two colors, red and black, from two boxes labeled A and B.

1. There are 10 black balls and 1 red ball in box A, and 1 black ball and 10 red balls in box B.
2. We randomly choose a box (with equal chance) and then pick a ball from it.
3. What is the probability that we draw a red ball finally?



Conditional Probability Example

When choosing a box to pick, we have:

1. $P(A) = P(B) = 0.5$.
2. If we choose A, $P(red|A) = 1/11$.
3. If we choose B, $P(red|B) = 10/11$.

So the probability to get one red ball from either box A or box B is:

$$\begin{aligned} P(red) &= P(red|A) \cdot P(A) + P(red|B) \cdot P(B) \\ &= \frac{1}{11} \times 0.5 + \frac{10}{11} \times 0.5 \\ &= \frac{1}{2} \end{aligned}$$

Bayes Theorem

- Bayes theorem is named after Thomas Bayes.
- It describes the probability of an event, based on conditions that might be related to the event.
- Bayes theorem states (assuming Y is discrete):

$$Pr(Y|X) = \frac{Pr(X|Y) \cdot Pr(Y)}{Pr(X)}$$
$$= \frac{Pr(X|Y) \cdot Pr(Y)}{\sum_l Pr(X|Y=l) \cdot Pr(Y=l)}$$

Bayes Theorem Example

- Consider the same experiment of picking balls from two boxes.
- If the ball we picked is red, then what is the probability that the ball was from box *A*?

According to Bayes' theorem, we have:

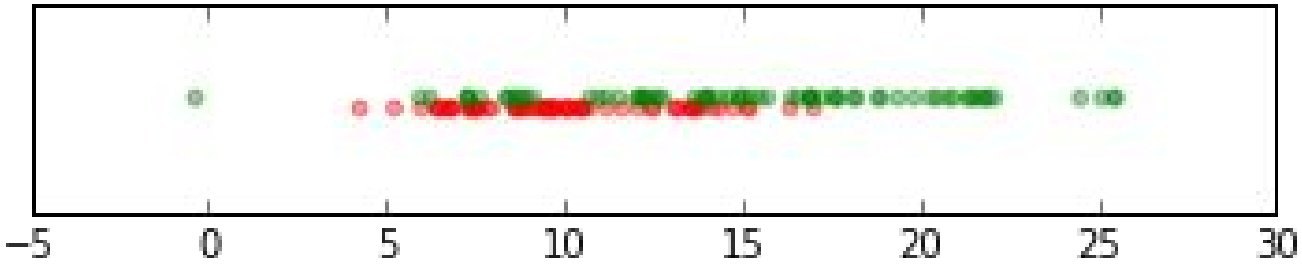
$$
\begin{aligned}
P(A|red) &= \frac{P(red|A) \cdot P(A)}{P(red)} \\
&= \frac{P(red|A) \cdot P(A)}{P(red|A)P(A) + P(red|B)P(B)} \\
&= \frac{\frac{1}{11} \times 0.5}{(\frac{1}{11} \times 0.5 + \frac{10}{11} \times 0.5)} \\
&= \frac{1}{11}
\end{aligned}
$$

Bayes Theorem Example

- How does this relate to our classification problem? Consider from the train set we realize that for a red ball:
  - the probability that the red ball was from box A is 1/11
  - and the probability that the red ball was from box B is 10/11
- Next time if we get a red ball, shouldn't we be more confident that the ball was from box B?

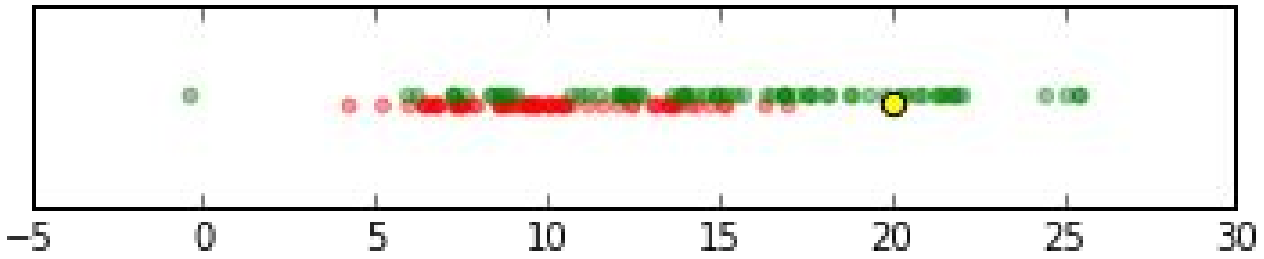
Discriminant Analysis

- *Discriminant analysis* is a statistical analysis technique which classifies based on hypothesizing the per class probability distribution to be normal and pinning down them by data fitting.
- **Motivation**: To be more precise, let's consider binary classification based on a numerical feature with a simulated data.



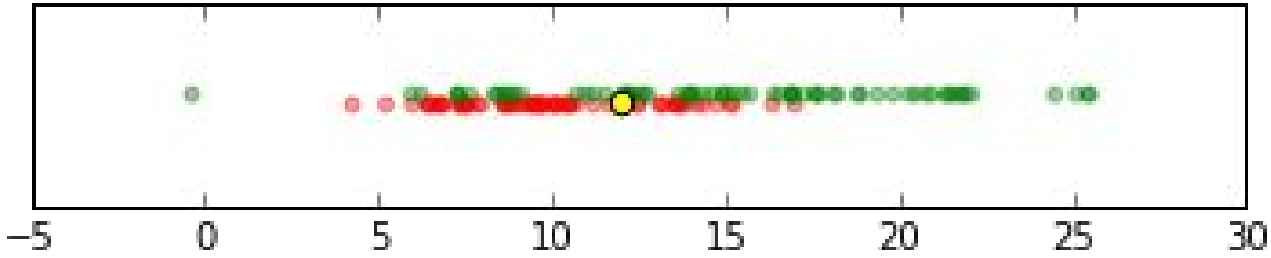
Discriminant Analysis

- If we add a new observation, which class do you think it belongs to?



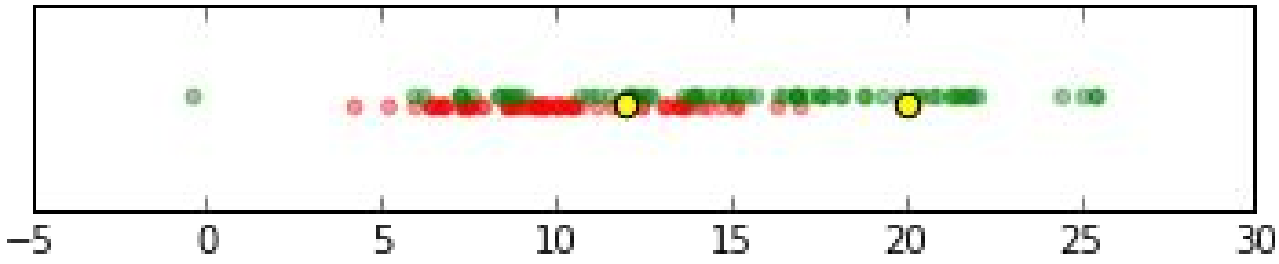
Discriminant Analysis

- What about this one?



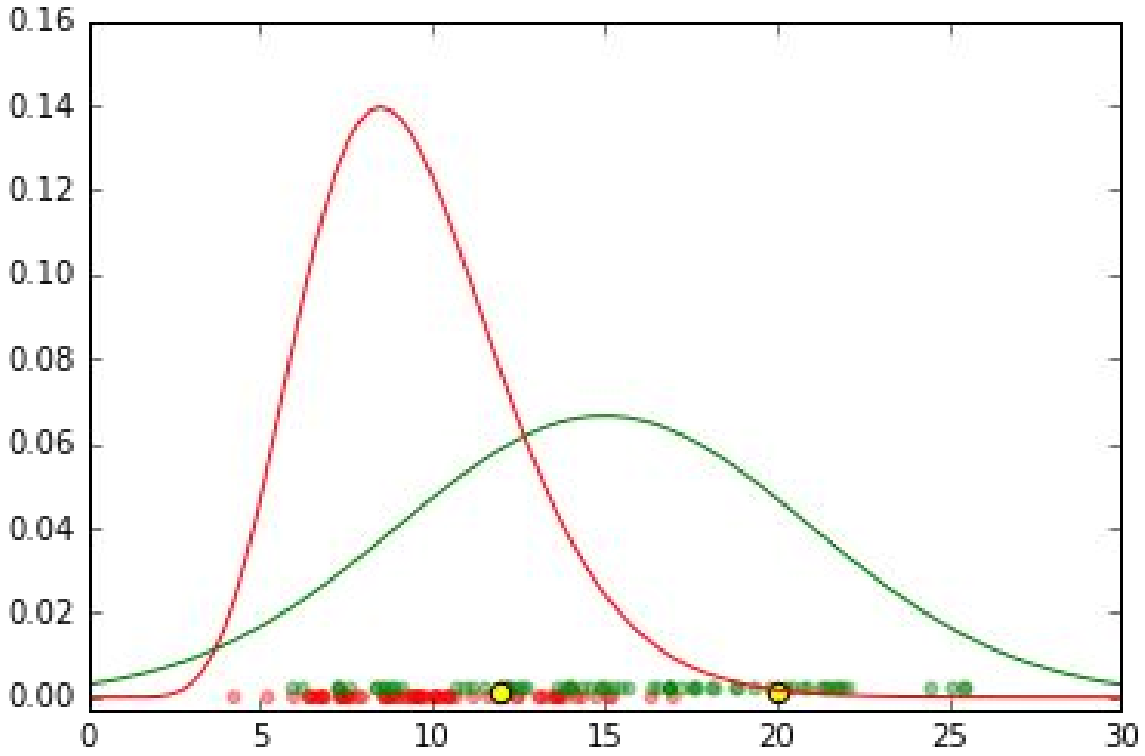
Discriminant Analysis

- What makes us feel differently?
  - If there is some other information tacitly guided us to the conclusion, can we somehow name it? or visualize it?



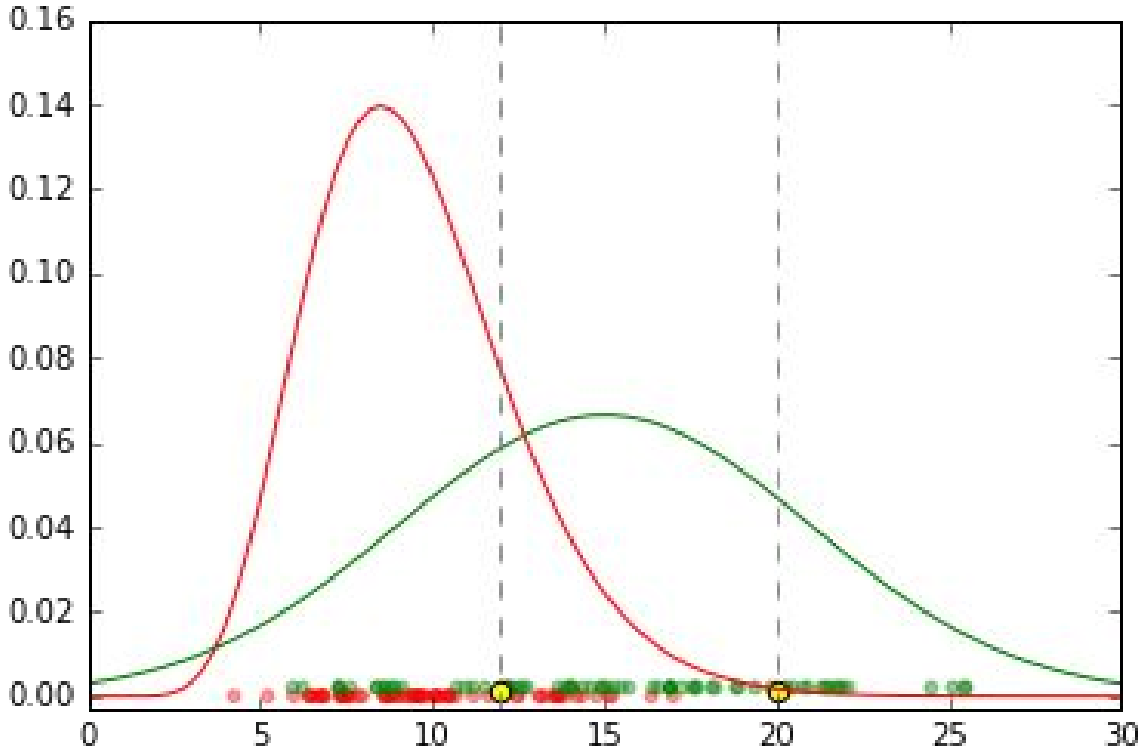
Discriminant Analysis

- How about density plot for each class?



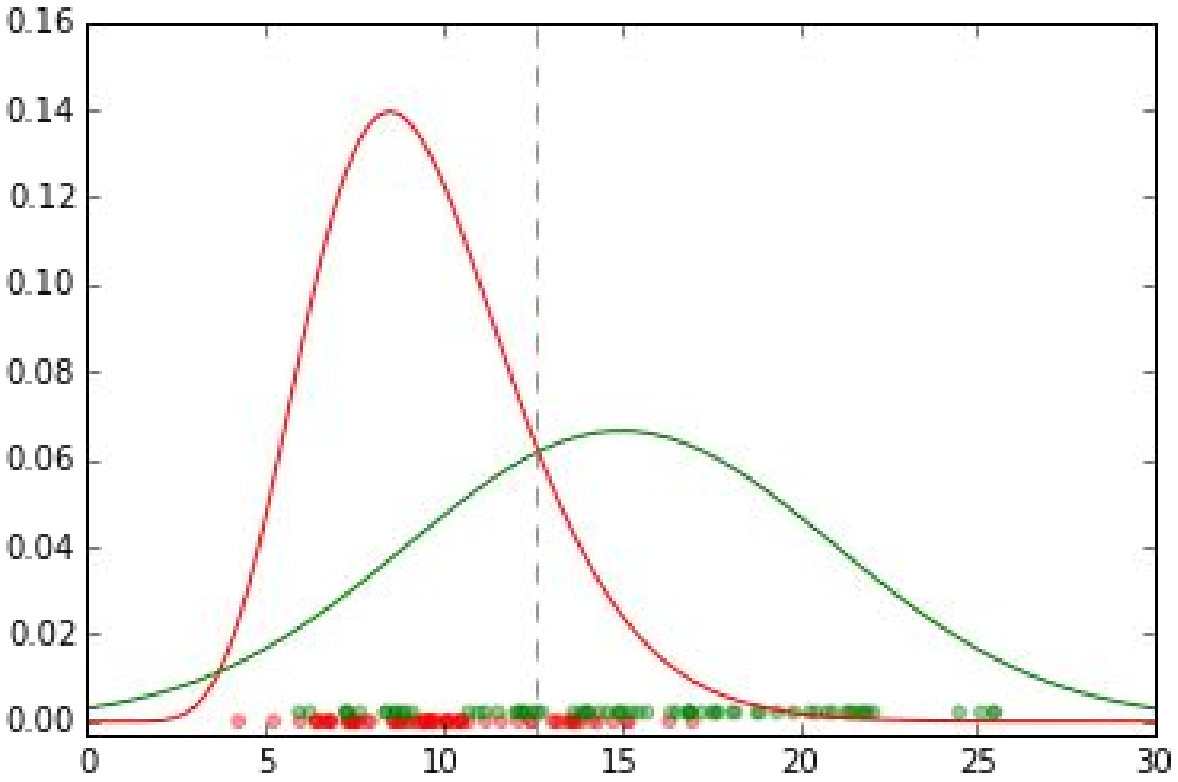
Discriminant Analysis

- What happens to the density plots at the two yellow observations?



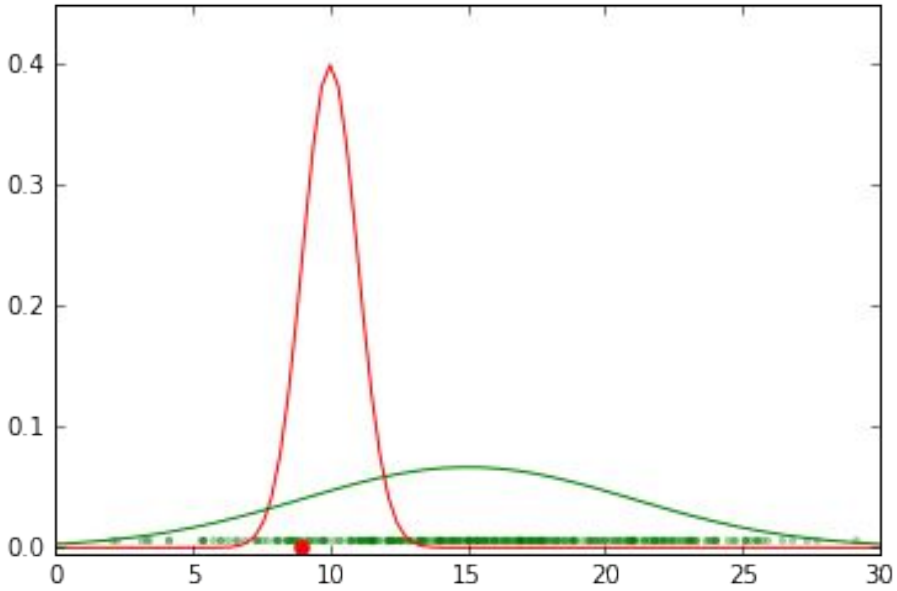
Bayes Classifier

- So is this how we classify?



Bayes Classifier

- **Caution:**
  - To emphasize the effect of the density within each class, we intentionally created two classes with the same size. When the sizes are different, **missing the prior would cause a big trouble**.
  - Below is an extreme case:



Discriminant Analysis

- Note the goal of classification is to compute

$$P(Y = k \mid X = x) \text{ for each class k}$$

- But we just found that

$$p(X = x \mid Y = k) \text{ for each class k}$$

Is helpful! How do we relate the two kinds of conditional probabilities?

Discriminant Analysis and Bayes Theorem

- Bayes theorem comes into play because we want to relate the two conditional probabilities above.

$$P(Y = k \mid X = x) = \frac{p(X = x \mid Y = k)P(Y = k)}{\sum_l p(X = x \mid Y = l)P(Y = l)}$$

- **Questions:**
  - How do we model $P(Y = k)$ (this is called the prior probability for class k)?
  - How do we model $p(X = x \mid Y = k)$?

Discriminant Analysis and Bayes Theorem

- **Answers:**

  - $P(Y = k)$ can be estimated by $\frac{n_k}{n}$

  Where:

  - n_k = the number of observations in class k.
  - n = the total count of observations.

  - Modeling $p(X = x \mid Y = k)$ is nontrivial. Different models result in different classifiers as we will see.

Bayes Classifier

- Now that we can predict the probability of belonging to a particular class, we can then label the observation to the class with the highest probability.
  - This is known as Bayes classifier. It minimises the probability of misclassification.
  - The boundary of classification is simply where the probabilities of different classes happen to be the same.

Outline

- Limitation of Linear Regression
- Logistic Regression
- Discriminant Analysis: Motivation
- **Discriminant Analysis: Models**
  - One Dimensional Cases
  - Higher Dimensional Cases
- Naive Bayes

Discriminant Analysis: Models

- To build a Bayes classifier, the only thing we miss is

$$p(X = x \mid Y = k)$$

- Since this is a continuous distribution, the Gaussian distribution is widely used to model it. Different kinds of Gaussian distribution result in different kind of classifiers. The following three are most common:
  - **Linear Discriminant Analysis** (LDA)
  - **Quadratic Discriminant Analysis** (QDA)
  - **Gaussian Naive Bayes** (This is the same as QDA in a one dimensional case)

Outline

- Limitation of Linear Regression
- Logistic Regression
- Discriminant Analysis: Motivation
- Discriminant Analysis: Models
  - **One Dimensional Cases**
    - Higher Dimensional Cases
- Naive Bayes

One Dimensional Cases

- When we have only one feature, we use one dimensional Gaussian distribution.

$$N(\mu, \sigma)(x) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

  - Note that it is sufficient to specify the mean and the standard deviation to specify a Gaussian distribution.

One Dimensional Cases

- We always allow **different means** among different classes, but....

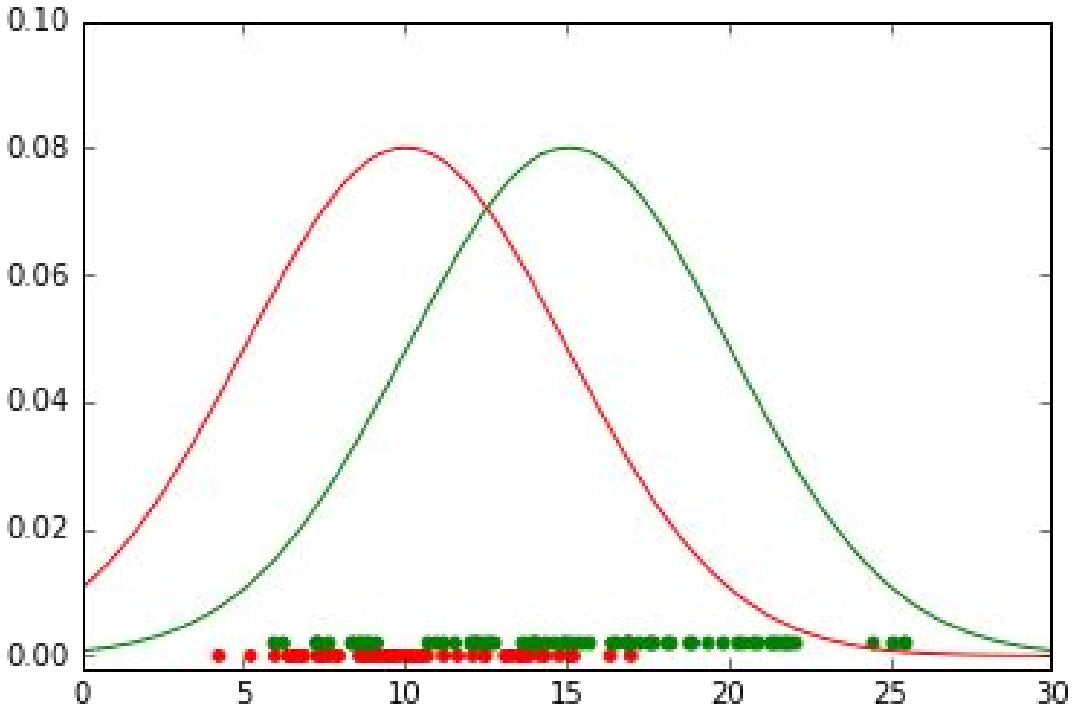
Linear Discriminant Analysis

- For LDA, we assume that the standard deviation is the same for every class. In one dimensional case, this means that the distribution density function for each class k is:

$$p(X = x \mid Y = k) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2}\left(\frac{x - \mu_k}{\sigma}\right)^2\right]$$

Linear Discriminant Analysis

- With visualization, this means the width of the distribution for every class is unchanged.



Linear Discriminant Analysis

- **Question:** Now we know that with LDA the distribution for each class k is:

$$p(X = x \mid Y = k) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{1}{2}\left(\frac{x - \mu_k}{\sigma}\right)^2\right]$$

  - How do we decide μ_k and σ?

Linear Discriminant Analysis

- **Answer:**

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i;y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i;y_i=k} (x_i - \hat{\mu}_k)^2$$

$$= \sum_{k=1}^{K} \frac{n_k - 1}{n-K} \cdot \hat{\sigma}_k^2$$

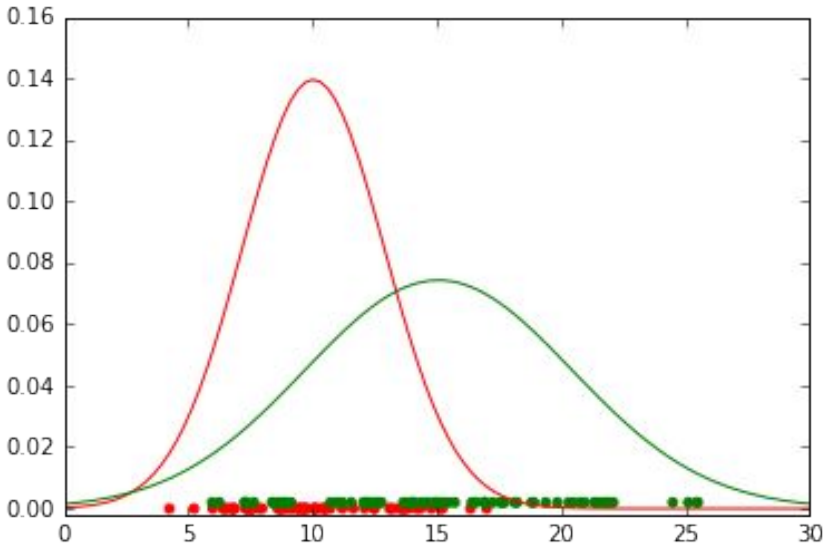
where

- K is the total number of classes.
- $\hat{\sigma}_k^2 = \frac{1}{n_k - 1} \sum_{i;y_i=k} (x_i - \hat{\mu}_k)^2$ is the sample variance of class k.

Quadratic Discriminant Analysis

- For QDA, the standard deviation can vary among the classes. In one dimensional case, this means the width of the distribution for every class can be different. Therefore:

$$p(X = x \mid Y = k) = \frac{1}{\sqrt{2\pi}\sigma_k} exp\Big[-\frac{1}{2}\Big(\frac{x - \mu_k}{\sigma_k}\Big)^2\Big]$$



Quadratic Discriminant Analysis

- **Question:** Now we know that with QDA the distribution for each class k is:

$$p(X = x \mid Y = k) = \frac{1}{\sqrt{2\pi}\sigma_k} exp\left[-\frac{1}{2}\left(\frac{x - \mu_k}{\sigma_k}\right)^2\right]$$

  - How do we estimate $\hat{\mu}_k$ and $\hat{\sigma}_k$?

Outline

- Limitation of Linear Regression
- Logistic Regression
- Discriminant Analysis: Motivation
- Discriminant Analysis: Models
  - One Dimensional Cases
  - **Higher Dimensional Cases**
- Naive Bayes

Higher Dimensional Cases

- We start with the discussion on higher dimensional Gaussian distribution. This is essentially the only difference in higher dimensional discriminant analysis.

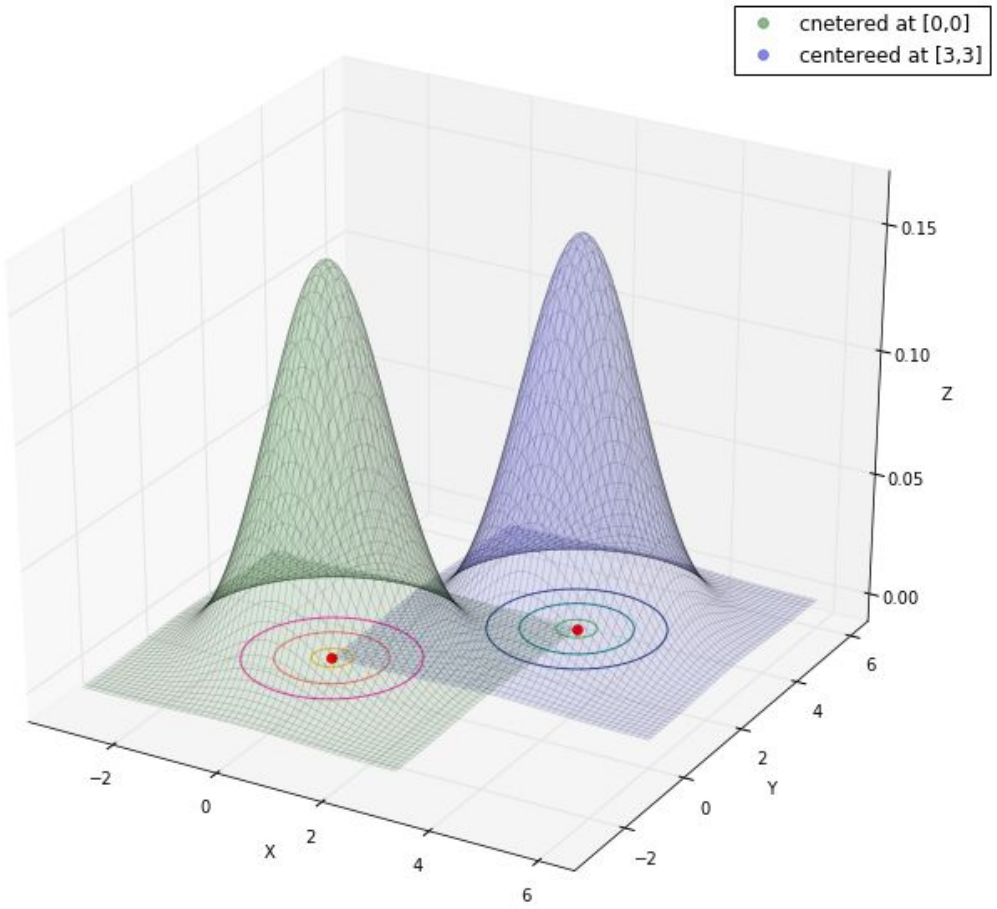
Higher Dimensional Gaussian Distribution

- We still need only "two" parameters to specify higher dimensional Gaussian distribution: the mean and the covariance. However, for a p dimensional case (with p features):
  - the mean is a p-dimensional vector.
  - the covariance is a $p \times p$ symmetric matrix.
- The distribution becomes:

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

Higher Dimensional Gaussian Distribution

- **mean**: The mean still decides the center where the "bell" is centered at.

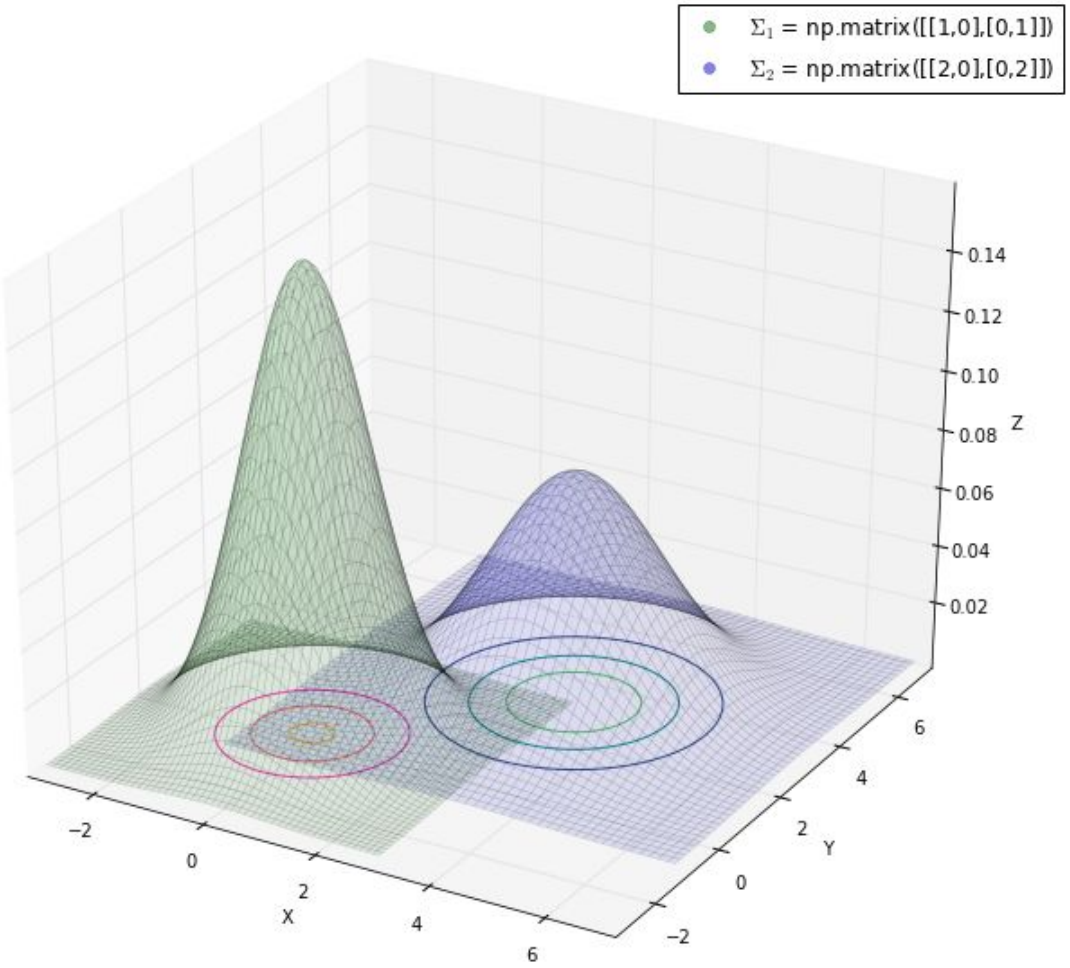


Higher Dimensional Gaussian Distribution

- **Covariance Matrix**: The covariance matrix is a $p \times p$ matrix. The covariance matrix, one of whose special cases is the square of standard deviation in one dimensional space, decides the shape of the "bell". However, the shape means more than just the width in a higher dimensional space.
- **Width**: Let's compare two Gaussian distributions with different covariance matrices in a two dimensional space.

$$\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \Sigma_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Higher Dimensional Gaussian Distribution

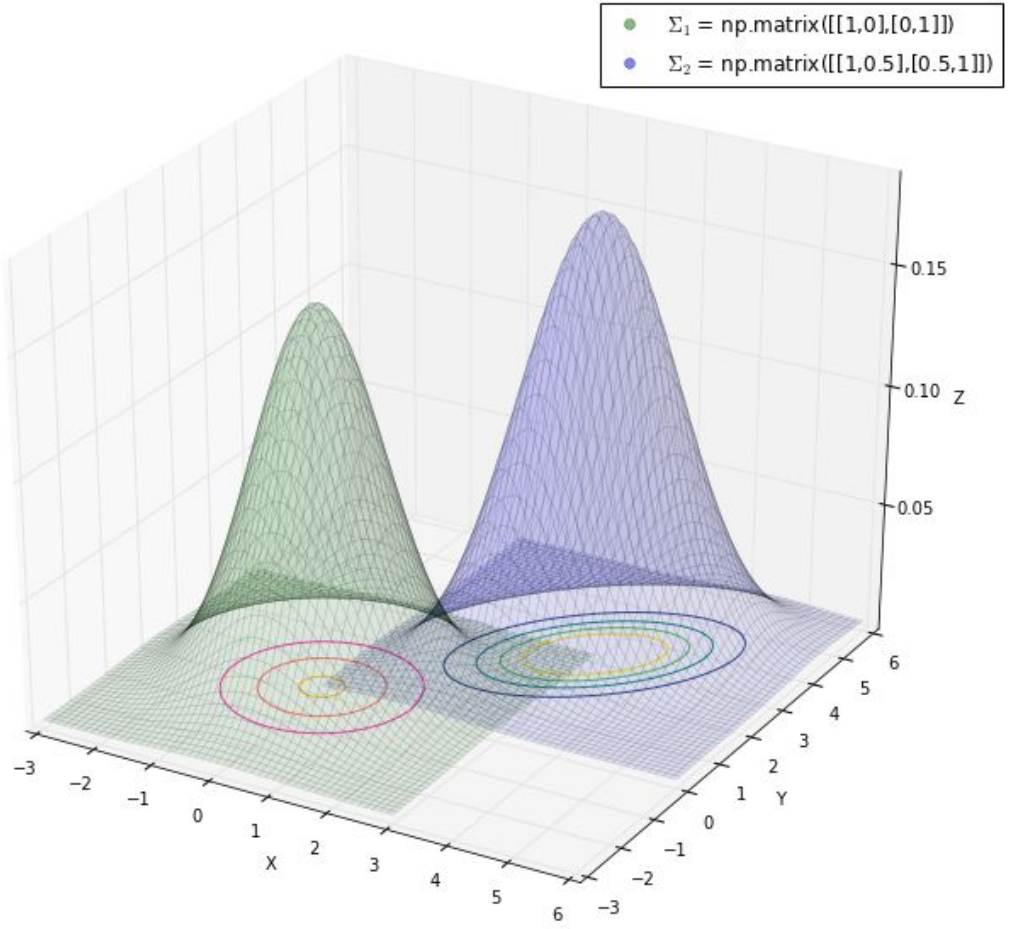


Higher Dimensional Gaussian Distribution

- **Correlation**: Let's compare two Gaussian distributions with different covariance matrices in a two dimensional space.

$$\Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \Sigma_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

Higher Dimensional Gaussian Distribution



Models in Higher Dimension

- So we only need to decide:
  - prior probability.
  - distribution of the features in each class.
- Since the prior probabilities are always estimated in the same way, the difference among LDA, QDA and GNB are stemmed from the assumptions on the Gaussian distribution.

Models in Higher Dimension: LDA

- LDA assumes the identical covariance matrix across all the classes. In the formula, we see that the mean depends on k, but the covariance matrix does not.

$$p(X = x \mid Y = k) = \frac{1}{(2\pi)|\Sigma|^{1/2}} exp\left[-\frac{1}{2}(x - \mu_k)^T \Sigma^{-1}(x - \mu_k)\right]$$

Models in Higher Dimension: QDA

- QDA allows different covariance matrices for different classes. In the formula, we see that the covariance matrix now depends on k as well.

$$p(X = x \mid Y = k) = \frac{1}{(2\pi)|\Sigma_k|^{1/2}} exp\left[-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)\right]$$

Models in Higher Dimension: GNB

- GNB also allows different covariance matrices for different classes.

$$p(X = x \mid Y = k) = \frac{1}{(2\pi)|\Sigma_k|^{1/2}} exp\left[-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)\right]$$

- The difference from QDA is that GNB assumes no correlation among the features, so

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_p^2 \end{bmatrix}$$

Models in Higher Dimension: GNB

- The assumption of zero correlation actually simplifies the conditional distribution. Within each class, the multivariate normal distribution can be written as the product of univariate normal distributions.

$$\prod_{j=1}^{p} \frac{1}{\sqrt{2\pi}\sigma_j} exp\left[-\frac{1}{2}\left(\frac{x_j - \mu_j}{\sigma_j}\right)^2\right]$$

  - Here each j indicates a feature subscript.

Hands-on Session

- Please go to the **"Discriminant Analysis in Scikit-Learn"** in the lecture code.

Outline

- Limitation of Linear Regression
- Logistic Regression
- Discriminant Analysis: Motivation
- Discriminant Analysis: Models
  - One Dimensional Cases
  - Higher Dimensional Cases
- **Naive Bayes Models**

Naive Bayes

- Recall that Bayes theorem assumes the probability that the output is in class *k,* given $X = x$, can be estimated by:

- LDA and QDA use multivariate Gaussian densities (but with different assumptions on the covariance matrices). These do not work well when the number of features is large.
- *Naive Bayes* models make a simplifying assumption that the features are conditionally independent in each class so it works with dataset of a large number of features.

Naive Bayes

- The naive Bayes classifier is based on Bayes theorem with independence assumptions between predictors.
- The assumption of conditional independence requires:

$$f_k(x) = \prod_{j=1}^{p} f_{jk}(x)$$

where $f_{jk}(x)$ is the probability density for the j^{th} feature X_j in class k.

- We will introduce three kinds of Naive Bayesian models:
  - Gaussian Naive Bayes
  - Multinomial Naive Bayes
  - Bernoulli Naive Bayes

Gaussian Naive Bayes

- Gaussian Naive Bayes assumes each feature follows a gaussian distribution ($\boldsymbol{\Sigma}_k$ is diagonal):

$$f_{jk}(x) = \frac{1}{\sqrt{2\pi}\sigma_{jk}} exp\Big[-\frac{(x_j - \mu_{jk})^2}{2\sigma_{jk}^2}\Big]$$

  where:

  - μ_{jk}: the mean of the j^{th} feature X_j in class k;
  - σ_{jk}^2: the variance of the j^{th} feature X_j in class k.

- Since we assume Gaussian densities, Gaussian Naive Bayes is best suited for continuous features.

- `In scikit-learn GaussianNB` implements the Gaussian Naive Bayes algorithm for classification.

Hands-on Session

- Please go to the **"Gaussian Naive Bayes in Scikit-Learn"** in the lecture code.

Coin Flipping

Rolling Dices

Some terminologies on Bernoulli and Multinomial Distributions

- **Bernoulli distribution** models an unfair coin flip, head & tail, once of probabilities p and 1-p.
- **Binomial distribution** models an unfair coin-flips N times independently
- **Multinomial distribution** models a M-sided unfair dice rolling N times independently.
- If we take N=1, a binomial distribution reduces to a **Bernoulli distribution**
- If we take N=1, a multinomial distribution reduces to a **categorical distribution**
- Suppose that we flip an unfair coin N times, the result is a long sequence

  H, T, T, T, H, H, ……..H, T, …..T.

- This is a text (long sentence) with two words 'H' and 'T'.
- **binomial distribution** models how many times do the head and the tail occur in a sample sequence.

Continued

- Suppose that the faces of an unfair dice is coded by M distinct symbols, S1, S2, …., SM, then the result of N independent flips of the dice is nothing but a long sequence like: S1, S1, S1, S2, S1, S3, …..SM, S3, ….S2.
- This is nothing but a long sentence of formed by the vocabulary

S1, S2, ….., SM.

- We model on the times S1 occurring in this 'sentence', S2 occurring in the sentence, …., SM occurring in the sentence. This is what multinomial distribution tries to capture.
- In general a multinomial distribution is determined by the probabilities of rolling the dice with symbols S1, S2, S3, ….SM, summing to 1.
- Alternatively, we may picture multinomial distribution as modeling the repeated drawings of a bag of symbols.

Multinomial Naive Bayes

- If all the columns of the raw data are categorical within the same value range, then we can parameterize the multinomial distribution by vectors $\theta_k = (\theta_{k1}, \theta_{k2}, \ldots, \theta_{kn})$ for each class k,
  where:
  - n: the number of features (the different values of the raw columns).
  - θ_{ki}: probability $P(x_i|k)$ of feature i appearing in a sample labelled to class k.
- `In scikit-learn MultinomialNB` implements the naive Bayes algorithm for multinomial distributed data, and is widely used in text classification/categorization.

Multinomial Naive Bayes Example

- In our spam email data, we choose three words to build the model: "sale", "money", "work", denoted by x_1, x_2, x_3.
  - Among all the spams, "sale" appears 48 times,"money" appears 50 times, "work" 2 times, 100 in total.

    Thus we estimate:

    - θ_1 = {0.48, 0.50, 0.02}
  - Among the non-spam emails, the frequency count of x_1, x_2, x_3 are 5, 10, 85, respectively.

    Thus we estimate:

    - θ_0 = {0.05, 0.10, 0.85}

Hands-on Session

- Please go to the **"Multinomial Naive Bayes in Scikit-Learn"** in the lecture code.

Bernoulli Naive Bayes

- Bernoulli Naive Bayes is used for data that:
  - is distributed according to a multivariate Bernoulli distribution;
  - each feature is assumed to be a binary-valued variable.
- `BernoulliNB` implements the naive Bayes training and classification algorithms.

Bernoulli Naive Bayes

- Consider the spam filter problem. In Bernoulli naive Bayes we do not care about the frequency count of a feature. We are just interested in whether it appears or not.
- Given a feature x_k which denotes a word, does it appear in an email or not? What is the probability of its appearance?

Bernoulli Naive Bayes Example

- Suppose we have 80 non-spams, and the word "sale" (denoted by x_k) appears in 10 of them; we also have 20 spams, and x_k appears in 16 of them. We use $y = 1$ to label a spam email. Then:

$$p(x_k = 1|y = 1) = \frac{16}{20} = \frac{4}{5}, \quad p(x_k = 0|y = 1) = \frac{1}{5}$$
$$p(x_k = 1|y = 0) = \frac{10}{80} = \frac{1}{8}, \quad p(x_k = 0|y = 0) = \frac{7}{8}$$

- Given a new email which contains the word “sale”, we have class = 1. If we use this single feature to predict:

$$p(y = 1|x_k = 1) = \frac{p(y = 1)p(x_k = 1|y = 1)}{p(x_k = 1)} = \frac{\frac{20}{100} \times \frac{4}{5}}{p(x_k = 1)} = \frac{0.16}{p(x_k = 1)}$$
$$p(y = 0|x_k = 1) = \frac{p(y = 0)p(x_k = 1|y = 0)}{p(x_k = 1)} = \frac{\frac{80}{100} \times \frac{1}{8}}{p(x_k = 1)} = \frac{0.1}{p(x_k = 1)}$$

then we will label this email to be spam.

Hands-on Session

- Please go to the **"Bernoulli Naive Bayes in Scikit-Learn"** in the lecture code.