Indian Institute of Information Technology, Allahabad

Introduction to Machine Learning

Bayesian Classification and Gaussian Discriminant Analysis

By

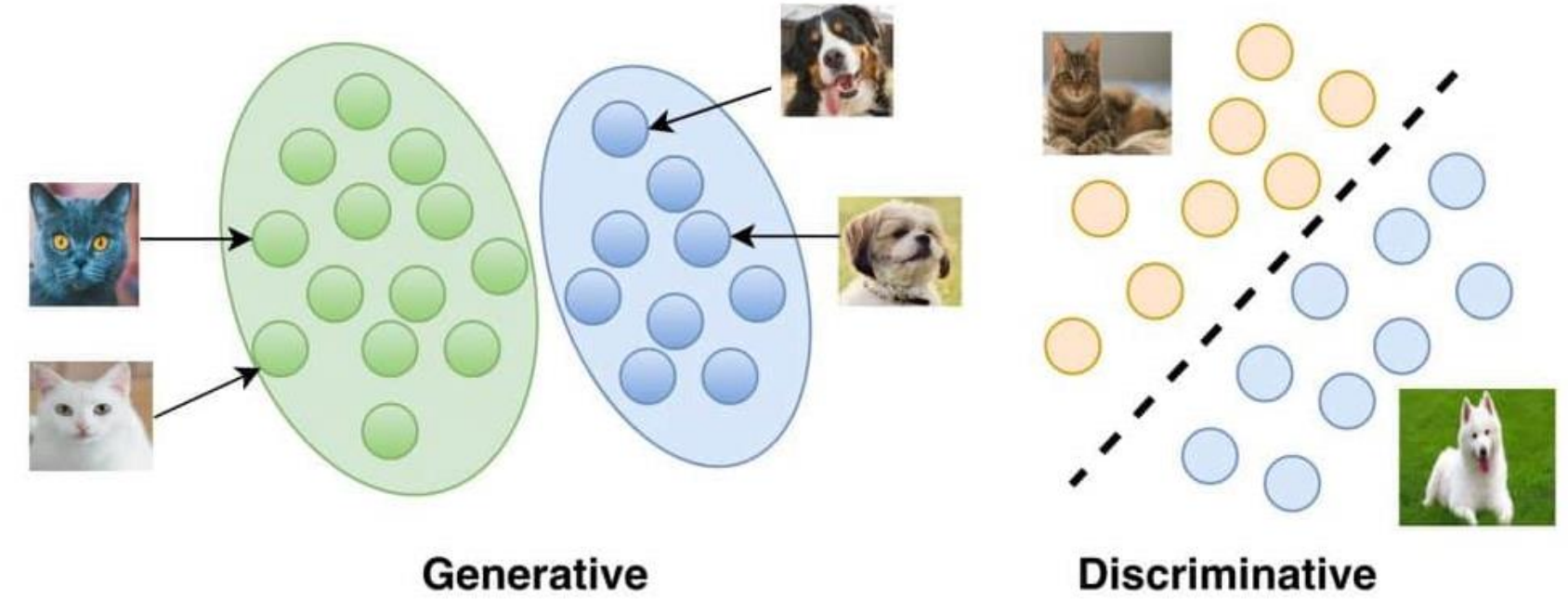
Dr. Shiv Ram Dubey
Associate Professor
Computer Vision and Biometrics Lab
Department of Information Technology
Indian Institute of Information Technology, Allahabad
Web: https://profile.iiita.ac.in/srdubey/

DISCLAIMER

The content (text, image, and graphics) used in this slide are adopted from many sources for academic purposes. Broadly, the sources have been given due credit appropriately. However, there is a chance of missing out some original primary sources. The authors of this material do not claim any copyright of such material.

Generative and Discriminative Models

- Classifying an image as a dog or a cat falls under Discriminative Modelling
- Producing a realistic dog or a cat image is a Generative Modelling problem



Generative and Discriminative Models

- Both Generative (G) and Discriminative (D) can be used for classification.
- G models work with both supervised and unsupervised learning. But D models are only used for supervised learning problems.
- The goal of the D model is to estimate the conditional probability P(Y|X). In contrast, the G model learns to approximate P(X) and P(X|Y) in an unsupervised setting, then deduces P(Y|X) in a supervised setting.

Generative and Discriminative Models

- Both Generative (G) and Discriminative (D) can be used for classification.
- G models work with both supervised and unsupervised learning. But D models are only used for supervised learning problems.
- The goal of the D model is to estimate the conditional probability P(Y|X). In contrast, the G model learns to approximate P(X) and P(X|Y) in an unsupervised setting, then deduces P(Y|X) in a supervised setting.
- D models learn a linear and non-linear decision boundary, using the data points and their respective labels, without knowing how the data was generated. In learning to model the probability distribution of the data, G models get to understand the data's underlying characteristics.
- In a supervised setting, D models are known to outperform G models. Especially, when G models do not fit the data well.
- G models can provide rich insights about the data, when you do not have labels.

Supervised Learning

Data $Z = \{(x_i, y_i)\}_{i=1}^{n}$

$\widehat{w}(Z) = \arg\min_{\mathrm{w}} L(\mathrm{w}; Z)$

L encodes $y_i \approx f_{\mathrm{w}}(x_i)$

Model $f_{\widehat{w}(\mathrm{Z})}$

Regression

Data $Z = \{(x_i, y_i)\}_{i=1}^{n}$

$\widehat{w}(Z) = \arg\min_{\mathrm{w}} L(\mathrm{w}; Z)$

L encodes $y_i \approx f_{\mathrm{w}}(x_i)$

Model $f_{\widehat{w}(\mathrm{Z})}$

Label is a **continuous value** y_i

Classification

Data $Z = \{(x_i, y_i)\}_{i=1}^{n}$

$\widehat{w}(Z) = \arg\min_{\mathrm{w}} L(\mathrm{w}; Z)$

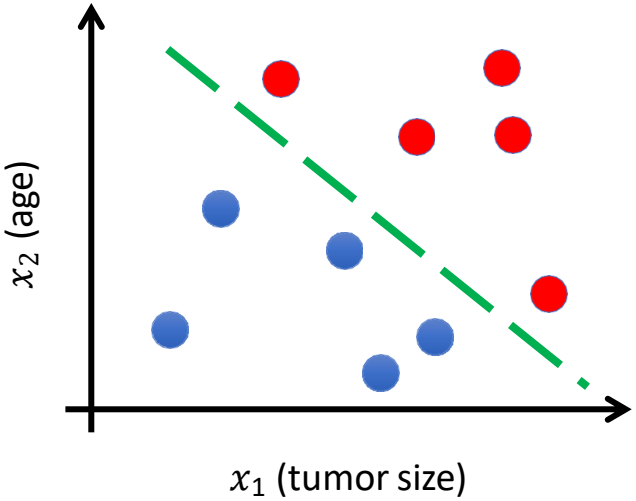
L encodes $y_i \approx f_{\mathrm{w}}(x_i)$

Model $f_{\widehat{w}(\mathrm{Z})}$

Label is a **discrete value** $y_i \in \mathcal{Y} = \{1, \dots, k\}$

(Binary) Classification

- **Input:** Dataset $Z = \{ (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \}$
- **Output:** Model $y_i \approx f_w(x_i)$



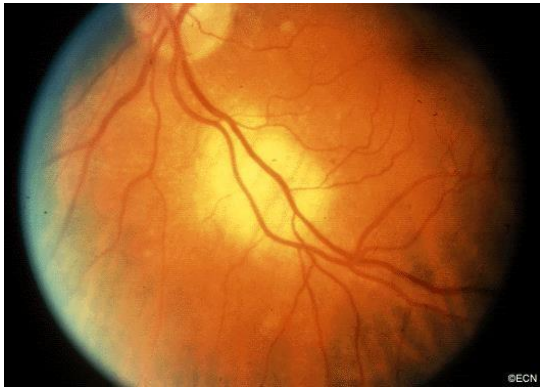
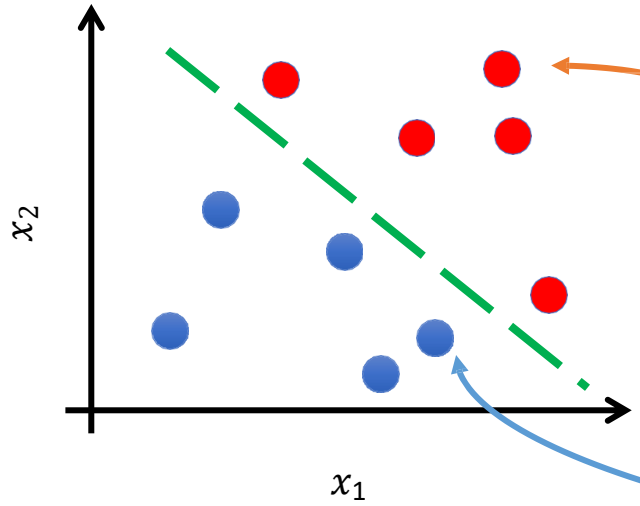


Image: https://eyecancer.com/uncategorized/choroidal-metastasis-test/

Example: Malignant vs. Benign Ocular Tumor

(Binary) Classification

- **Input:** Dataset $Z = \{ (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \}$
- **Output:** Model $y_i \approx f_w(x_i)$



Example: Cat vs. Dog

Maximum Likelihood Estimation (MLE) vs Maximum A Posteriori Estimation (MAP)

- Suppose we are given a dataset $D = \{x_1, x_2, \ldots, x_n\}$, which we assume is generated from a probabilistic model parameterized by θ.
- Our goal is to estimate the unknown parameter θ using the observed data.
- Let $p(x \mid \theta)$ denote the likelihood function, i.e., the probability of observing data x given parameter θ.

MLE vs MAP

Maximum Likelihood Estimation (MLE)

- MLE aims to find the parameter θ that maximizes the likelihood of the observed data:

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta} p(D \mid \theta)$$

- Assuming that the data are independent and identically distributed (i.i.d.), we can write the likelihood function as:

$$p(D \mid \theta) = \prod_{i=1}^{n} p(x_i \mid \theta)$$

- To make the optimization tractable, we often take the logarithm of the likelihood (log-likelihood):

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta} \log p(D \mid \theta) = \arg\max_{\theta} \sum_{i=1}^{n} \log p(x_i \mid \theta)$$

- MLE selects the parameter that makes the observed data most probable.

MLE vs MAP

Maximum A Posteriori Estimation (MAP)

- MAP estimation incorporates prior knowledge or belief about the parameter θ via a prior distribution $p(\theta)$. It aims to maximize the posterior probability: $\hat{\theta}_{\text{MAP}} = \arg\max_\theta p(\theta \mid D)$
- Using Bayes' theorem: $p(\theta \mid D) = \frac{p(D|\theta)p(\theta)}{p(D)}$
- Since $p(D)$ is constant with respect to θ, we can write: $\hat{\theta}_{\text{MAP}} = \arg\max_\theta p(D \mid \theta)p(\theta)$
- Or, equivalently: $\hat{\theta}_{\text{MAP}} = \arg\max_\theta [\log p(D \mid \theta) + \log p(\theta)]$
- MAP estimation combines observed data with prior beliefs. The prior can help regularize the estimate, especially when data is sparse or noisy.

MLE vs MAP

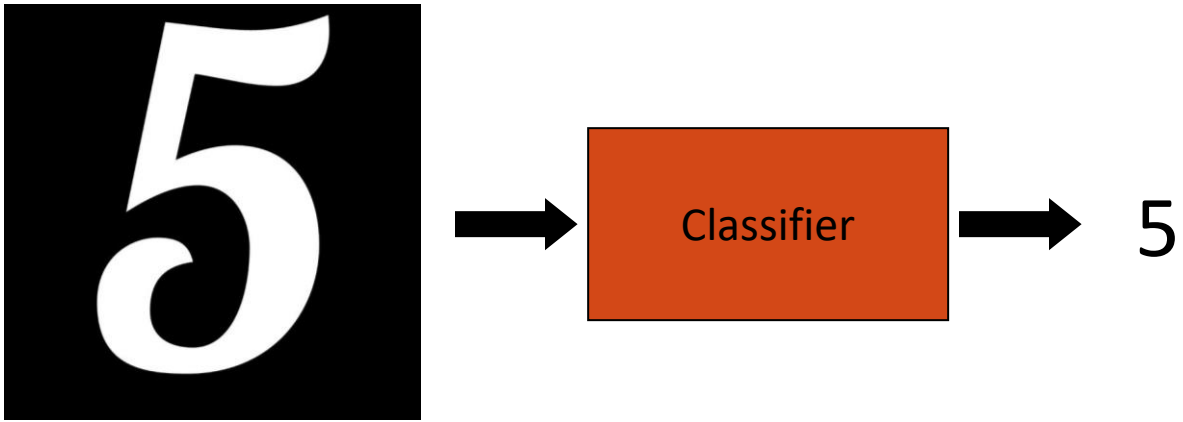
Aspect	MLE	MAP
Prior Used?	No	Yes
Objective	Maximize likelihood	Maximize posterior
Regularization	No implicit regularization	Implicit regularization via prior
Interpretation	Best fit to observed data	Best estimate given data and prior
Sensitivity to data	High (may overfit)	Lower (can reduce overfitting)

Bayesian Classification

- Problem statement:
  - Given features $X_1, X_2, \ldots, X_n$
  - Predict a label Y

An Application: Digit Recognition

- $X_1, \ldots, X_n \in \{0,1\}$ (Black vs. White pixels)
- $Y \in \{5,6\}$ (predict whether a digit is a 5 or a 6)



The Bayes Classifier

- A good strategy is to predict:

$$\arg\max_{Y} P(Y|X_1, \ldots, X_n)$$

  - (for example: what is the probability that the image represents a 5 given its pixels?)

- So … How do we compute that?

The Bayes Classifier

- Use Bayes Rule!

Likelihood, probability of features X given class Y

Prior probability of class Y

Posterior probability, probability of class Y given features X

$$P(Y|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y)P(Y)}{P(X_1,\ldots,X_n)}$$

Marginal likelihood or Evidence (Normalization Constant)

- Why did this help? Well, we think that we might be able to specify how features are "generated" by the class label

The Bayes Classifier

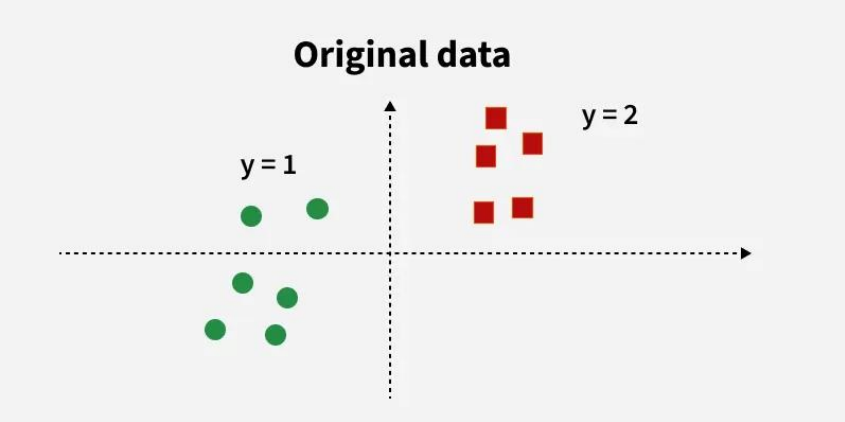
- Let's expand this for digit recognition task:

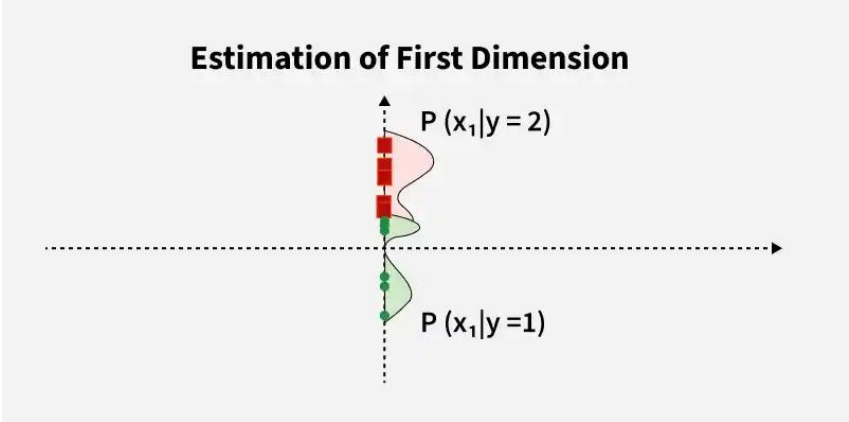
$$P(Y=5|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y=5)P(Y=5)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)}$$

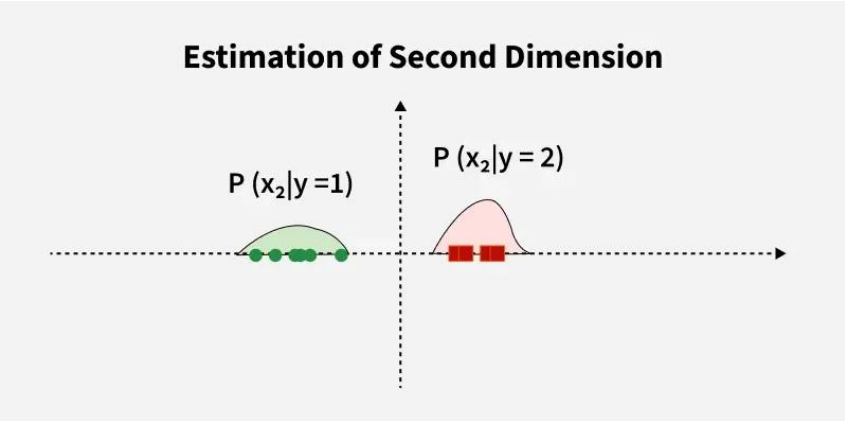
$$P(Y=6|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y=6)P(Y=6)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)}$$

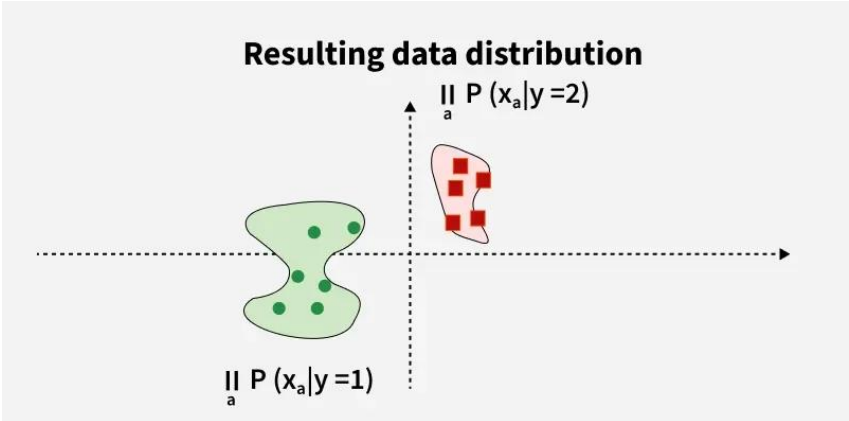
- To classify, we'll simply compute these two probabilities and predict based on which one is greater

The Bayes Classifier









Source: https://www.geeksforgeeks.org/

The Bayes Classifier

- For the Bayes classifier, we need to “learn” two functions,
  - the likelihood and
  - the prior

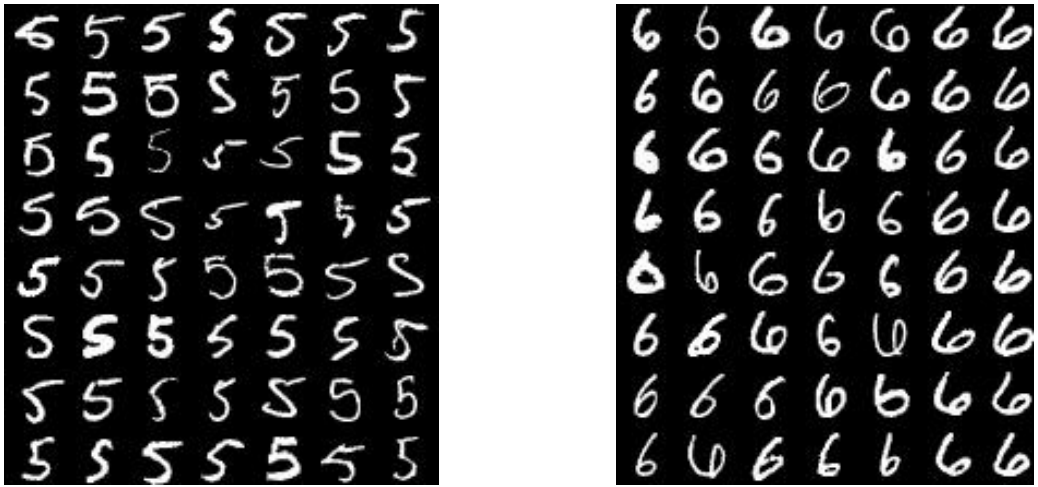
The Naïve Bayes Model

- The *Naïve Bayes Assumption*: Assume that all features are independent **given the class label Y**
- Equationally speaking:

$$P(X_1, \ldots, X_n|Y) = \prod_{i=1}^{n} P(X_i|Y)$$

Naïve Bayes Training

- Now that we've decided to use a Naïve Bayes classifier, we need to train it with some data:



MNIST Training Data

Naïve Bayes Training

- Training in Naïve Bayes is **easy**:
  - Estimate $P(Y = v)$ as the fraction of records with $Y = v$

$$P(Y = v) = \frac{Count(Y = v)}{\#\ records}$$

  - Estimate $P(X_i = u|Y = v)$ as the fraction of records with $Y = v$ for which $X_i = u$

$$P(X_i = u|Y = v) = \frac{Count(X_i = u \wedge Y = v)}{Count(Y = v)}$$

This corresponds to Maximum Likelihood estimation of model

Naïve Bayes Training

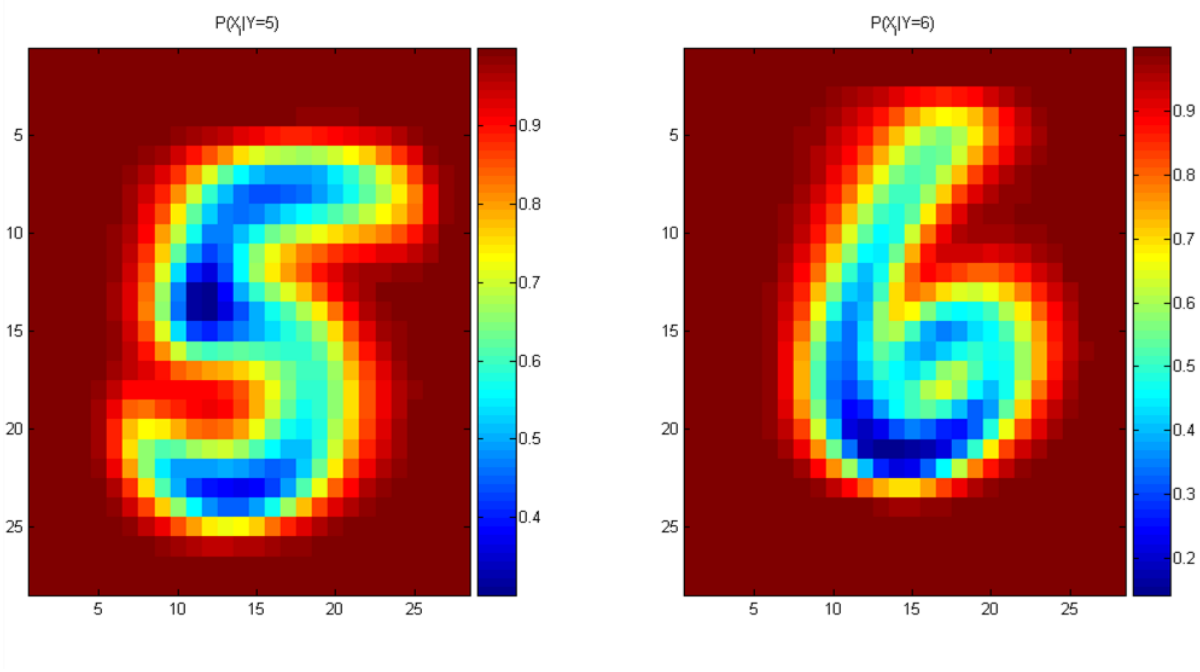
- In practice, some of these counts can be zero
- Fix this by adding "virtual" counts:

$$P(X_i = u | Y = v) = \frac{Count(X_i = u \wedge Y = v) + 1}{Count(Y = v) + 2}$$

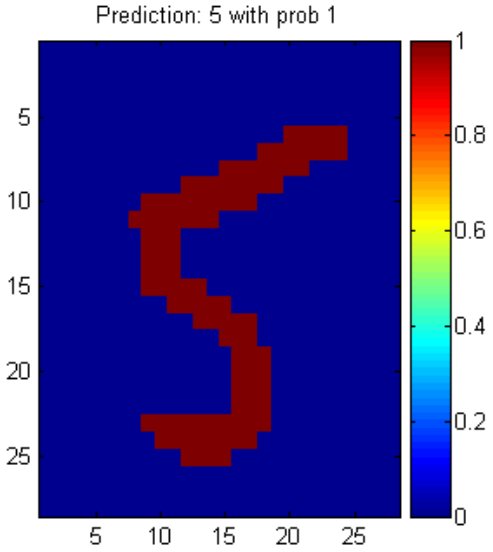
- This is like putting a prior on parameters and doing MAP (Maximum A Posteriori) estimation instead of MLE
- This is called *Smoothing*

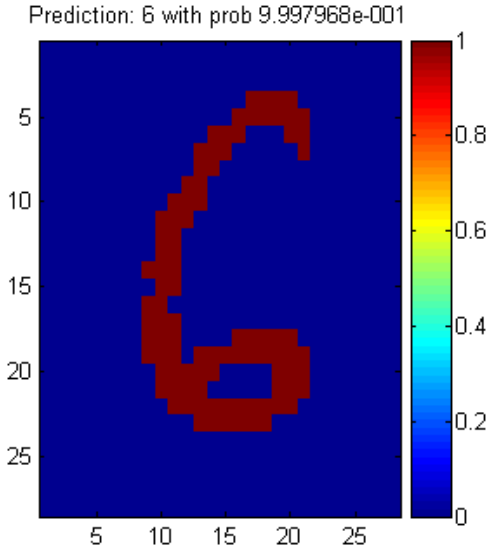
Naïve Bayes Training

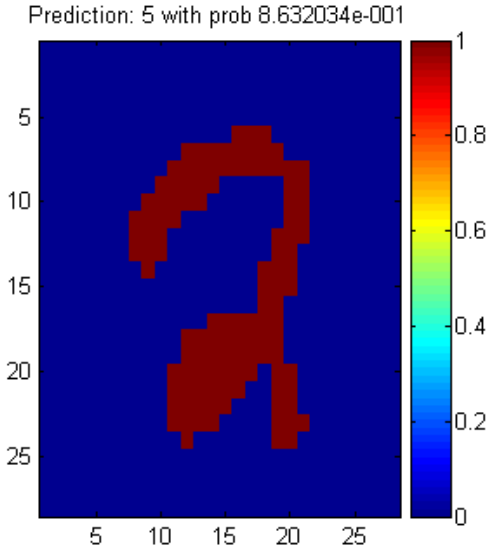
- For binary digits, training amounts to averaging all of the training fives together and all of the training sixes together.



Naïve Bayes Classification







Another Example of the Naïve Bayes Classifier

The weather data, with counts and probabilities

outlook			temperature			humidity			windy		play	
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2	3	hot	2	2	high	3	4	false	6	2	9	5
overcast	4	0	mild	4	2	normal	6	1	true	3	3		
rainy	3	2	cool	3	1								
sunny	2/9	3/5	hot	2/9	2/5	high	3/9	4/5	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9	3/5		
rainy	3/9	2/5	cool	3/9	1/5								

A new day

outlook	temperature	humidity	windy	play
sunny	cool	high	true	?

Another Example of the Naïve Bayes Classifier

- Likelihood of yes $= \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14} = 0.0053$

- Likelihood of no $= \frac{3}{5} \times \frac{1}{5} \times \frac{4}{5} \times \frac{3}{5} \times \frac{5}{14} = 0.0206$

- Therefore, the prediction is No

Naive Bayes Classifier: Numerical Attribute Values

- One common practice to handle numerical attribute values is to assume normal distributions for numerical attributes.

Naive Bayes Classifier: Numerical Attribute Values

The numeric weather data with summary statistics

outlook			temperature		humidity			windy		play	
	yes	no	yes	no	yes	no		yes	no	yes	no
sunny	2	3	83	85	86	85	false	6	2	9	5
overcast	4	0	70	80	96	90	true	3	3		
rainy	3	2	68	65	80	70					
			64	72	65	95					
			69	71	70	91					
			75		80						
			75		70						
			72		90						
			81		75						
sunny	2/9	3/5	mean	73	74.6	mean	79.1	86.2	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	std dev	6.2	7.9	std dev	10.2	9.7	true	3/9	3/5		
rainy	3/9	2/5											

Naive Bayes Classifier: Numerical Attribute Values

- Let $x_1, x_2, \ldots, x_n$ be the values of a numerical attribute in the training data set.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \mu)^2$$

$$f(w) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(w-\mu)^2}{\sigma^2}}$$

Naive Bayes Classifier: Numerical Attribute Values

- For examples,

$$f(\text{temperature} = 66 \mid \text{Yes}) = \frac{1}{\sqrt{2\pi}(6.2)} e^{-\frac{(66-73)^2}{2(6.2)^2}} = 0.0340$$

- Likelihood of Yes = $\frac{2}{9} \times 0.0340 \times 0.0221 \times \frac{3}{9} \times \frac{9}{14} = 0.000036$

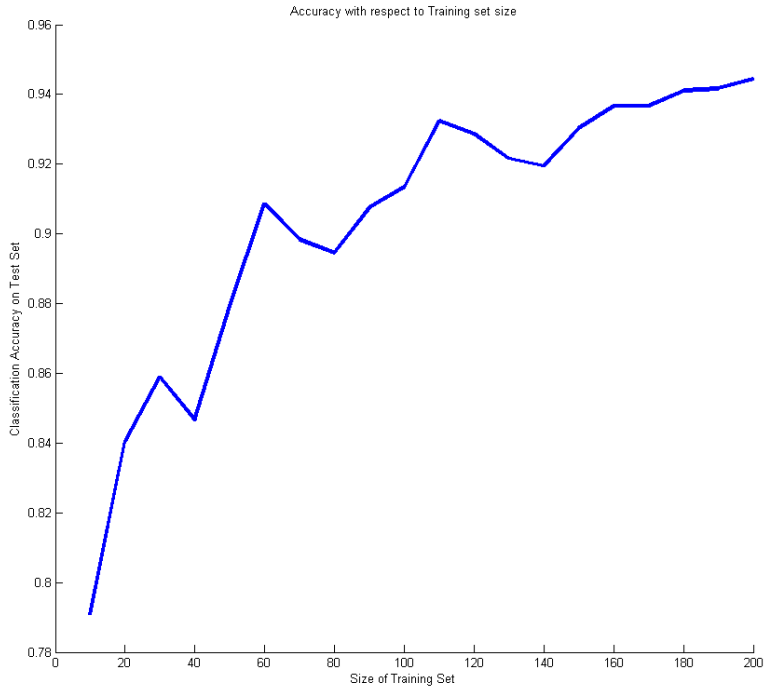
- Likelihood of No = $\frac{3}{5} \times 0.0291 \times 0.038 \times \frac{3}{5} \times \frac{5}{14} = 0.000136$

Outputting Probabilities

- What's nice about Naïve Bayes (and generative models in general) is that it returns probabilities
  - These probabilities can tell us how confident the algorithm is
  - So… don't throw away those probabilities!

Outputting Probabilities

- Naïve Bayes is often a good choice if you don't have much training data!



Naïve Bayes Assumption

- Recall the Naïve Bayes assumption:
  - that all features are independent **given the class label Y**
- Does this hold for the digit recognition problem?

Exclusive-OR Example

- For an example where conditional independence fails:
  - $Y = XOR(X_1, X_2)$

X_1	X_2	$P(Y=0\|X_1,X_2)$	$P(Y=1\|X_1,X_2)$
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

Naïve Bayes Assumption

- Actually, the Naïve Bayes assumption is almost never true

- Still… Naïve Bayes often performs surprisingly well even when its assumptions do not hold

Numerical Stability

- It is often the case that machine learning algorithms need to work with very small numbers
  - Imagine computing the probability of 2000 independent coin flips
  - Programming Platform thinks that $(0.5)^{2000}=0$

Underflow Prevention

- Multiplying lots of probabilities

  ➔ floating-point underflow.

- Recall: $\log(xy) = \log(x) + \log(y)$,

  ➔ better to sum logs of probabilities rather than multiplying probabilities.

  ➔ Class with highest final un-normalized log probability score is still the most probable.

Numerical Stability

- Instead of comparing

$$P(Y = 5|X_1, \ldots, X_n) \text{ with } P(Y = 6|X_1, \ldots, X_n),$$

  - Compare their logarithms

$$\begin{aligned}
\log\left(P(Y|X_1, \ldots, X_n)\right) &= \log\left(\frac{P(X_1, \ldots, X_n|Y) \cdot P(Y)}{P(X_1, \ldots, X_n)}\right) \\
&= \text{constant} + \log\left(\prod_{i=1}^{n} P(X_i|Y)\right) + \log P(Y) \\
&= \text{constant} + \sum_{i=1}^{n} \log P(X_i|Y) + \log P(Y)
\end{aligned}$$

Naïve Bayes Classifier - Summary

- We defined a *Bayes classifier* but saw that it's intractable to compute $P(X_1, \ldots, X_n|Y)$
- We then used the *Naïve Bayes assumption* – that everything is independent given the class label Y

- Naïve Bayes is:
  - Really easy to implement and often works well
  - Often a good first thing to try

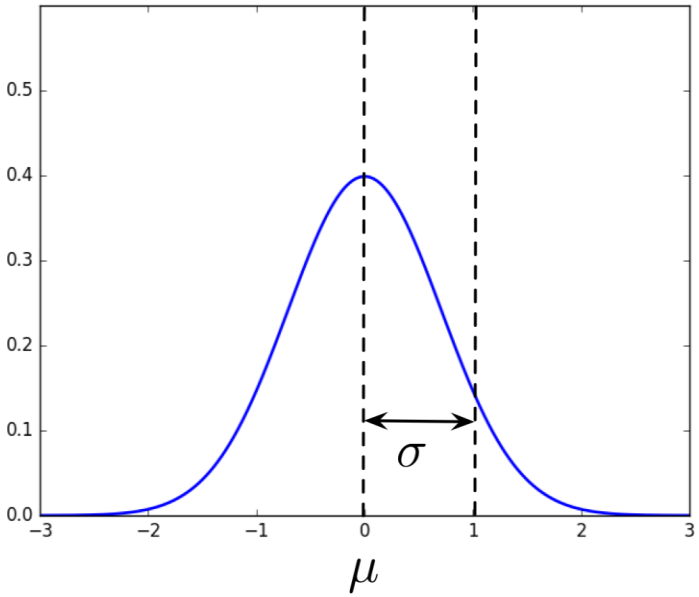
Recall: Univariate Gaussian Distribution

- Recall the Gaussian, or normal, distribution:

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

- The Central Limit Theorem says that sums of lots of independent random variables are approximately Gaussian.

- In machine learning, we use Gaussians a lot because they make the calculations easy.



Multivariate Data

- Multiple measurements (sensors)
- D inputs/features/attributes
- N instances/observations/examples

$$\mathbf{X} = \begin{bmatrix} [\mathbf{x}^{(1)}]^\top \\ [\mathbf{x}^{(2)}]^\top \\ \vdots \\ [\mathbf{x}^{(N)}]^\top \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix}$$

Multivariate Mean and Covariance

- Mean

$$\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}] = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_d \end{pmatrix}$$

- Covariance

$$\boldsymbol{\Sigma} = \mathrm{Cov}(\mathbf{x}) = \mathbb{E}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^\top] = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1D} \\ \sigma_{12} & \sigma_2^2 & \cdots & \sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{D1} & \sigma_{D2} & \cdots & \sigma_D^2 \end{pmatrix}$$

The statistics ($\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$) uniquely define a multivariate Gaussian (or multivariate Normal) distribution, denoted $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ or $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Gaussian Discriminant Analysis

- When we have a classification problem in which the input features x are continuous-valued random variables, we can use the Gaussian Discriminant Analysis (GDA) model, which models $P(x \mid y)$ using a multivariate normal distribution.
- Let's consider the tumor-identification task and model it as a Gaussian distribution. This can be expressed using two equations $P(x \mid y = 0)$ for benign and $P(x \mid y = 1)$ for malignant. On the other hand, $y \in \{0,1\}$ is a Bernoulli random variable. Formally, the model is given by,

$$x \mid y = 0 \sim \mathcal{N}(\mu_0, \Sigma)$$
$$x \mid y = 1 \sim \mathcal{N}(\mu_1, \Sigma)$$
$$y \sim \text{Bernoulli}(\phi)$$

- Parameters of our GDA model are μ_0, μ_1, Σ and ϕ.
- Note that we used the same covariance matrix Σ for both classes, but different mean vectors μ_0 and μ_1.
- You can also use different covariance matrices Σ_0 and Σ_1, but that is not commonly seen.

ϕ is the estimate of probability of y being equal to 1

Gaussian Discriminant Analysis

- Assume X is distributed as a multivariate Gaussian, i.e, $X \in \mathbb{R}^n$, parameterized by a mean vector $\mu \in \mathbb{R}^n$ and a covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$, where $\Sigma \geq 0$ is symmetric and positive semi-definite. Formally, this can be written as,

$$X \sim \mathcal{N}(\mu, \Sigma)$$

- The probability density function (PDF) of X is given by:

$$P(X; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)\right)$$

  - where "$|\Sigma|$" denotes the determinant of the matrix Σ.
- The expected value of X is given by μ:

$$\mathrm{E}[X] = \int_x x P(x; \mu, \Sigma) dx = \mu$$

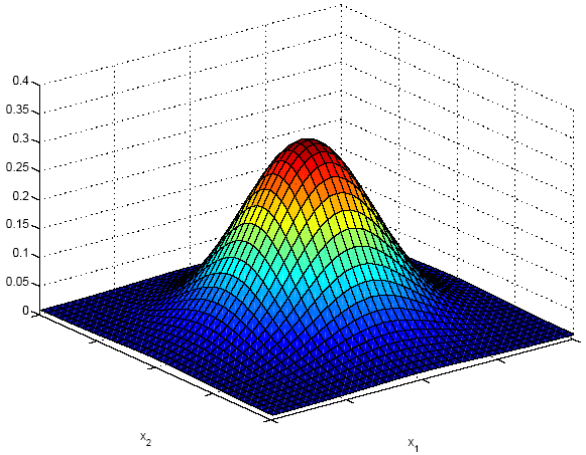
Gaussian Discriminant Analysis

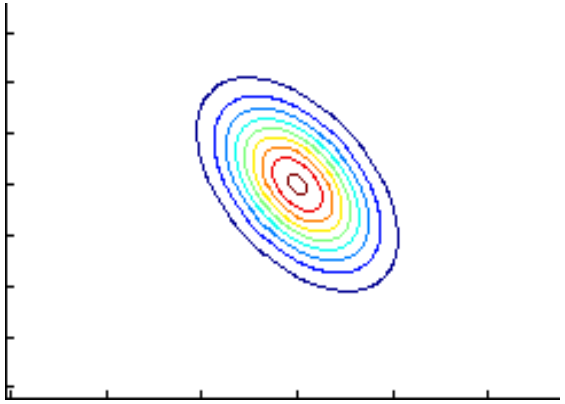
- Note that $\phi \in \mathbb{R}$, $\mu_0 \in \mathbb{R}^n$, $\mu_1 \in \mathbb{R}^n$, and $\Sigma \in \mathbb{R}^{n \times n}$

$$P(y) = \phi^y (1-\phi)^{1-y}$$

$$P(x \mid y=0) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (x-\mu_0)^T \Sigma^{-1} (x-\mu_0) \right)$$

$$P(x \mid y=1) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (x-\mu_1)^T \Sigma^{-1} (x-\mu_1) \right)$$





Gaussian Discriminant Analysis

- Suppose you have a training set, $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$.
- To fit the aforementioned parameters to our training set, maximize the joint likelihood L(·),

$$L(\phi, \mu_0, \mu_1, \Sigma) = \prod_{i=1}^{m} P(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma)$$
$$= \prod_{i=1}^{m} P(x^{(i)} \mid y^{(i)}; \mu_0, \mu_1, \Sigma) P(y^{(i)}; \phi)$$

- The log-likelihood of the data $\ell(\cdot)$ is simply the log of the likelihood $L(\cdot)$. Formally,

$$\ell(\phi, \mu_0, \mu_1, \Sigma) = \log \prod_{i=1}^{m} P(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma)$$
$$= \log \prod_{i=1}^{m} P(x^{(i)} \mid y^{(i)}; \mu_0, \mu_1, \Sigma) P(y^{(i)}; \phi)$$

To maximize $\ell(\cdot)$ with respect to the parameters, take the derivative of $\ell(\cdot)$, set it equal to 0 and then solve for the values of the parameters that maximize the expression for $\ell(\cdot)$.

Gaussian Discriminant Analysis

- This yields the maximum likelihood estimate of ϕ to be,

$$\phi = \frac{\sum_{i=1}^{m} y^{(i)}}{m} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\}}{m}$$

The maximum likelihood estimate for ϕ is just the fraction of training examples with label $y = 1$.

Gaussian Discriminant Analysis

- This yields the maximum likelihood estimate of ϕ to be,

$$\phi = \frac{\sum_{i=1}^{m} y^{(i)}}{m} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\}}{m}$$

The maximum likelihood estimate for ϕ is just the fraction of training examples with label $y = 1$.

- The maximum likelihood estimate of the mean vectors μ_0, μ_1 is,

$$\mu_0 = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 0\right\} x^{(i)}}{\sum_{i=1}^{m} 1\left\{y^{(i)} = 0\right\}}$$

$$\mu_1 = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\} x^{(i)}}{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\}}$$

The maximum likelihood estimate of the mean of all of the features for a particular class

Gaussian Discriminant Analysis

- This yields the maximum likelihood estimate of ϕ to be,

$$\phi = \frac{\sum_{i=1}^{m} y^{(i)}}{m} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\}}{m}$$

The maximum likelihood estimate for ϕ is just the fraction of training examples with label $y = 1$.

- The maximum likelihood estimate of the mean vectors μ_0, μ_1 is,

$$\mu_0 = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 0\right\} x^{(i)}}{\sum_{i=1}^{m} 1\left\{y^{(i)} = 0\right\}}$$

$$\mu_1 = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\} x^{(i)}}{\sum_{i=1}^{m} 1\left\{y^{(i)} = 1\right\}}$$

The maximum likelihood estimate of the mean of all of the features for a particular class

- The maximum likelihood estimate of the covariance matrix Σ is,

$$\Sigma = \frac{1}{m} \sum_{i=1}^{m} \left(x^{(i)} - \mu_{y^{(i)}}\right)\left(x^{(i)} - \mu_{y^{(i)}}\right)^T$$

Making Predictions Using GDA

- So given a new tumor, how do you make a prediction for whether it is malignant or benign?
- The problem boils down to predicting the most likely class label y given an input sample x, which can be formally written as:

$$\underset{y}{\operatorname{argmax}}\, P(y \mid x)$$

We're essentially looking to choose a value of $y = \{0,1\}$ that maximizes this expression.

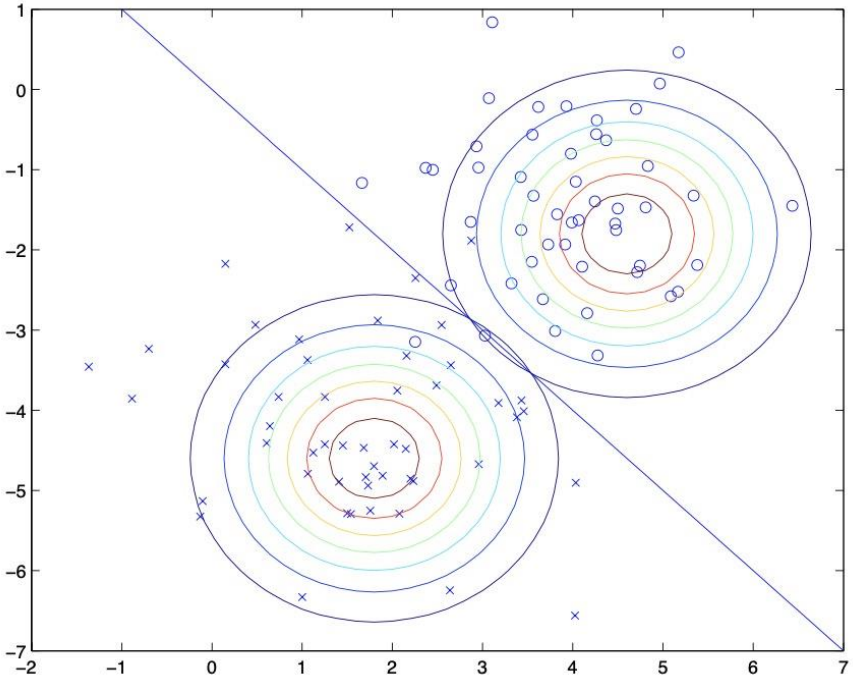
- Using Bayes' rule,

$$\underset{y}{\operatorname{argmax}}\, P(y \mid x) = \underset{y}{\operatorname{argmax}}\, \frac{P(x \mid y)P(y)}{P(x)}$$

$$\approx \underset{y}{\operatorname{argmax}}\, P(x \mid y)P(y)$$

Making Predictions Using GDA

- Pictorially, GDA's operation can be seen as follows:



The contours of the two Gaussian distributions that have been fit to the data in positive and negative examples

- Note that the two Gaussians have contours that are the same shape and orientation, since they share a covariance matrix Σ, but they have different means μ_0 and μ_1.
- For each point, the class label is determined using Bayes' rule.
- They imply the decision boundary (in blue), at which $P(y = 1 \mid x) = 0.5$.
- On one side of the boundary, we predict $y = 1$ to be the most likely outcome, and on the other side, we predict $y = 0$.
- Points to the upper right of the decision boundary are classified as negative examples and points to the lower left are classified as positive examples.

Acknowledgement

Most of the slides are based on the ML course of:

- John R. Rose, University of South Carolina
- Matt Barnes and Matt Golub, University of Washington
- Mingmin Zhao and Jiatao Gu, University of Pennsylvania

THANK YOU