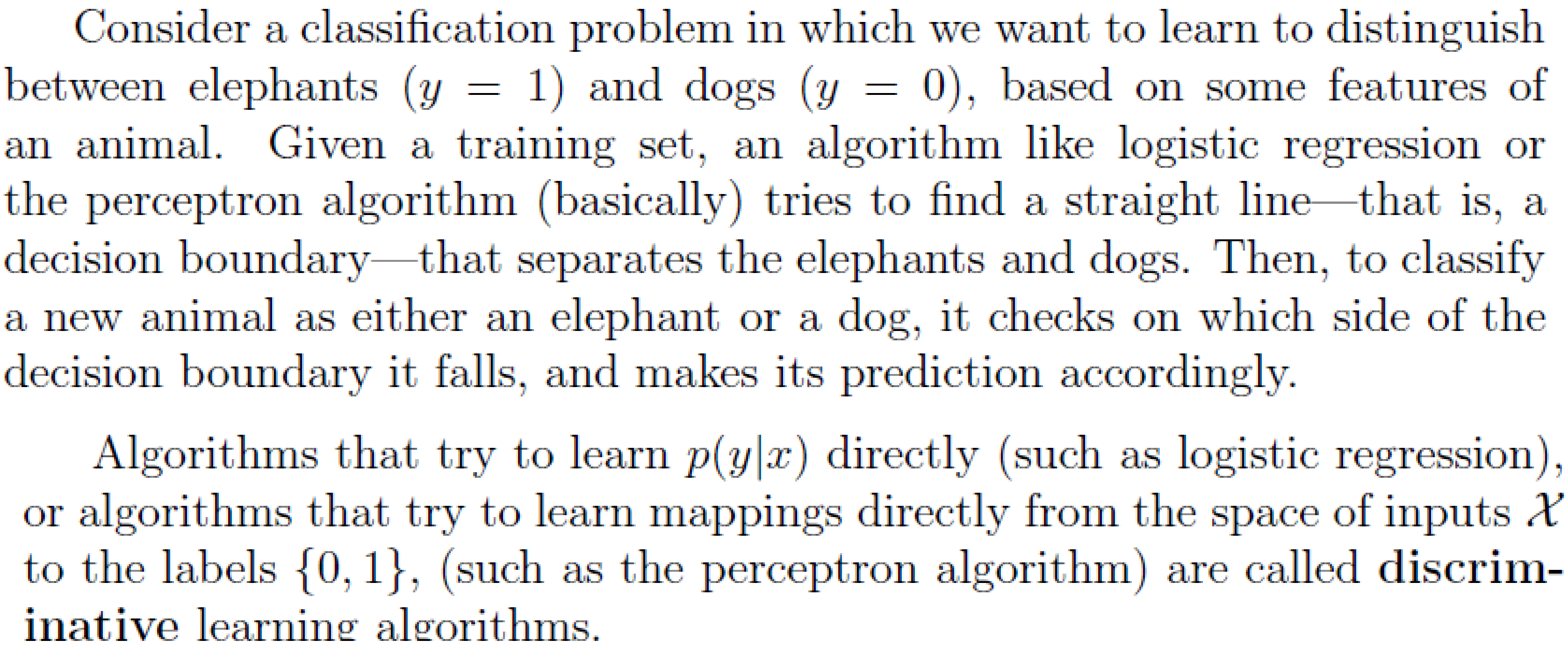
Generative Learning Algorithms

Mohsen Afsharchi

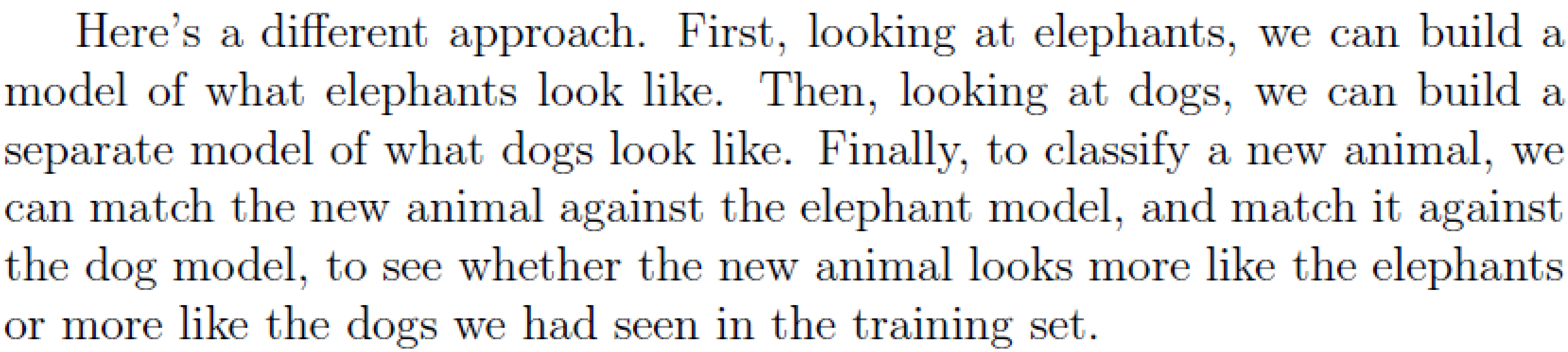
Discriminative Approach



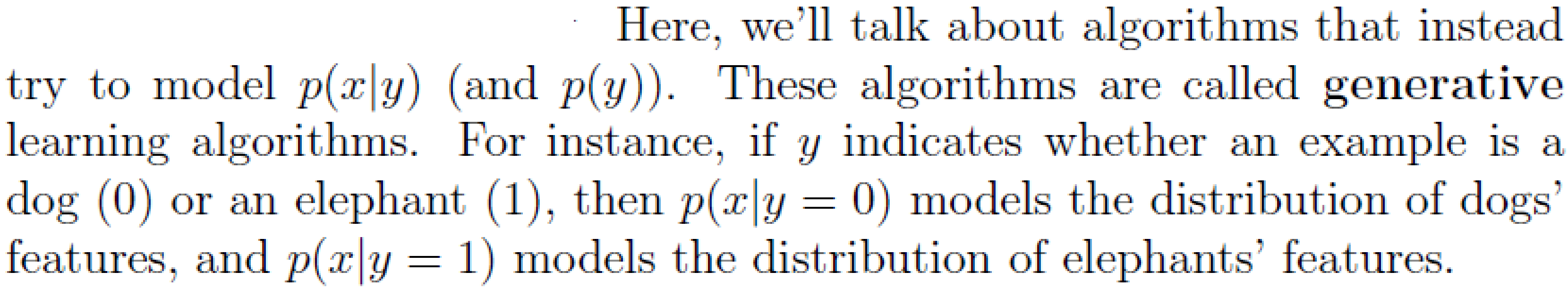
Consider a classification problem in which we want to learn to distinguish between elephants ($y = 1$) and dogs ($y = 0$), based on some features of an animal. Given a training set, an algorithm like logistic regression or the perceptron algorithm (basically) tries to find a straight line—that is, a decision boundary—that separates the elephants and dogs. Then, to classify a new animal as either an elephant or a dog, it checks on which side of the decision boundary it falls, and makes its prediction accordingly.

Algorithms that try to learn $p(y|x)$ directly (such as logistic regression), or algorithms that try to learn mappings directly from the space of inputs $\mathcal{X}$ to the labels $\{0, 1\}$, (such as the perceptron algorithm) are called **discriminative** learning algorithms.

Generative Approach



Here's a different approach. First, looking at elephants, we can build a model of what elephants look like. Then, looking at dogs, we can build a separate model of what dogs look like. Finally, 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 we had seen in the training set.



Here, we'll talk about algorithms that instead try to model $p(x|y)$ (and $p(y)$). These algorithms are called **generative learning algorithms**. For instance, if y indicates whether an example is a dog (0) or an elephant (1), then $p(x|y = 0)$ models the distribution of dogs' features, and $p(x|y = 1)$ models the distribution of elephants' features.

Generative Approach

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}.$$

$$p(x) = p(x|y = 1)p(y = 1) + p(x|y = 0)p(y = 0)$$

$$\begin{aligned}\arg\max_y p(y|x) &= \arg\max_y \frac{p(x|y)p(y)}{p(x)} \\ &= \arg\max_y p(x|y)p(y).\end{aligned}$$

Gaussian Discriminant Analysis

Multivariate Normal Distribution ▪

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

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

$$\mathrm{Cov}(Z) = \mathrm{E}[(Z - \mathrm{E}[Z])(Z - \mathrm{E}[Z])^T]$$

Example: bivariate normal density

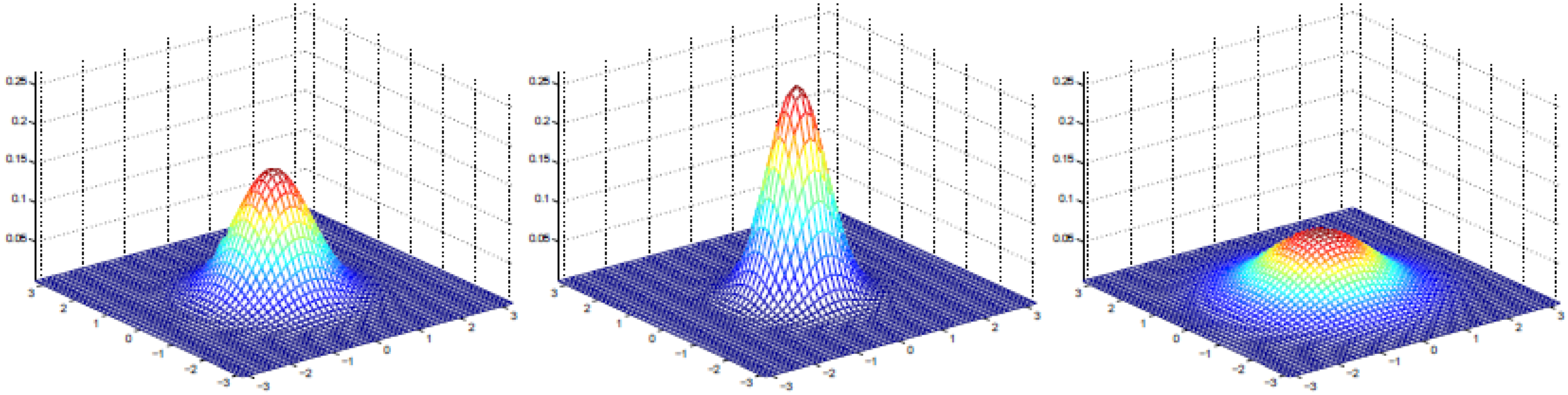
$$\mu_1 = E(X_1), \quad \mu_2 = E(X_2),$$

$$\sigma_{11} = \mathrm{Var}(X_1), \sigma_{22} = \mathrm{Var}(X_2), \text{and } \sigma_{12} = \sigma_{12}/(\sqrt{\sigma_{11}}\sqrt{\sigma_{22}}) = \mathrm{Corr}(X_1, X_2).$$

▪ $$\mathbf{\Sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & \sigma_{22} \end{bmatrix} \qquad \mathbf{\Sigma}^{-1} = \frac{1}{\sigma_{11}\sigma_{22} - \sigma_{12}^2} \begin{bmatrix} \sigma_{22} & -\sigma_{12} \\ -\sigma_{12} & \sigma_{11} \end{bmatrix}$$

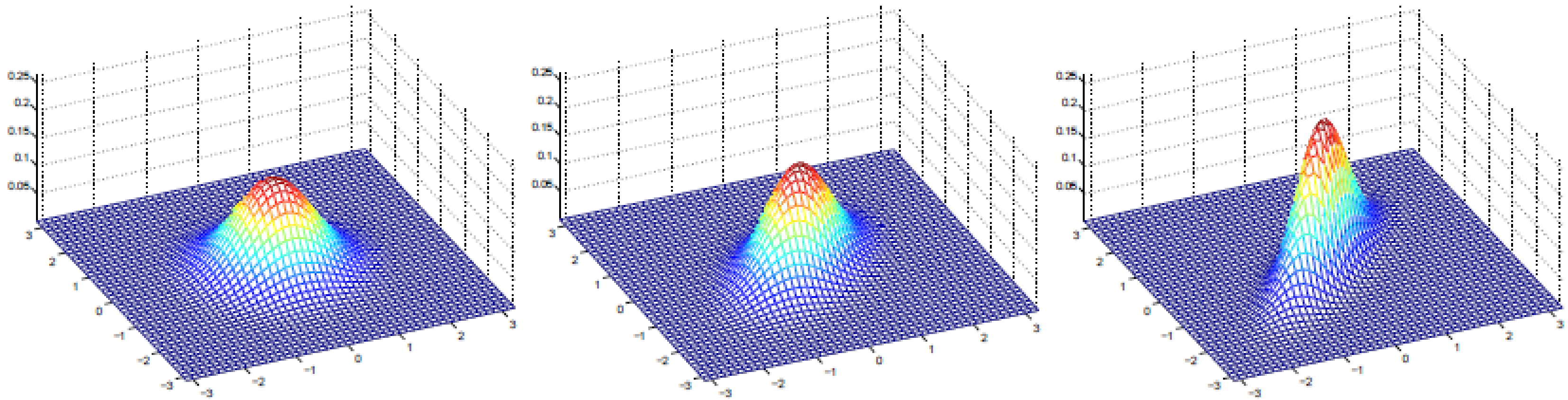
Some Examples

Here're some examples of what the density of a Gaussian distribution looks like:



The left-most figure shows a Gaussian with mean zero (that is, the 2x1 zero-vector) and covariance matrix $\Sigma = I$ (the 2x2 identity matrix). A Gaussian with zero mean and identity covariance is also called the **standard normal distribution**. The middle figure shows the density of a Gaussian with zero mean and $\Sigma = 0.6I$; and in the rightmost figure shows one with , $\Sigma = 2I$. We see that as Σ becomes larger, the Gaussian becomes more "spread-out," and as it becomes smaller, the distribution becomes more "compressed."

More Examples

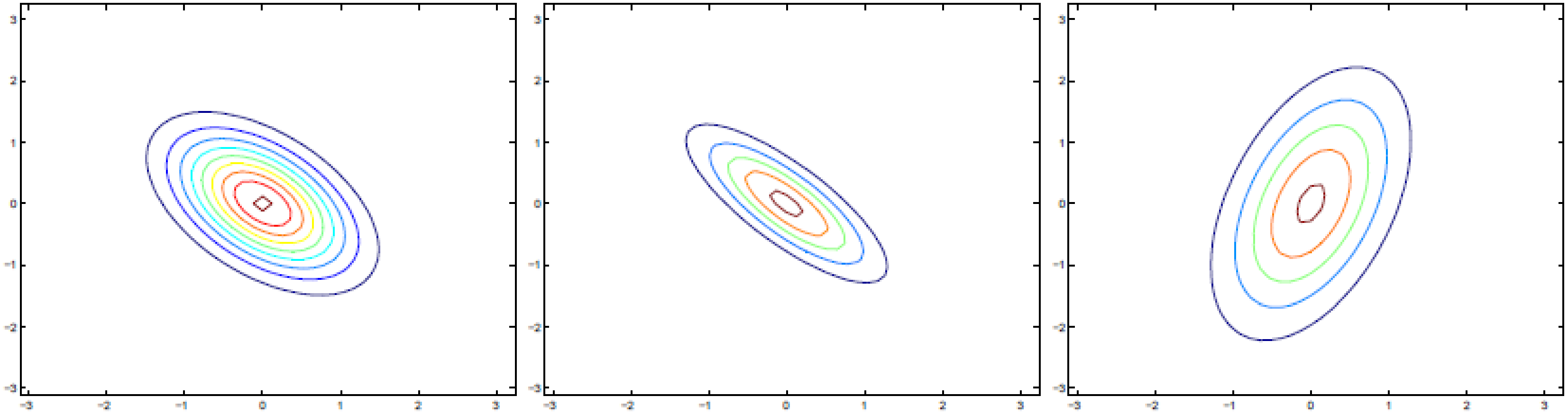


The figures above show Gaussians with mean 0, and with covariance matrices respectively

$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix};\ \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix};\ .\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}.$$

Varying Covariance Matrix

Here's one last set of examples generated by varying Σ:



The plots above used, respectively,

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

Gaussian Discriminant Analysis Model

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

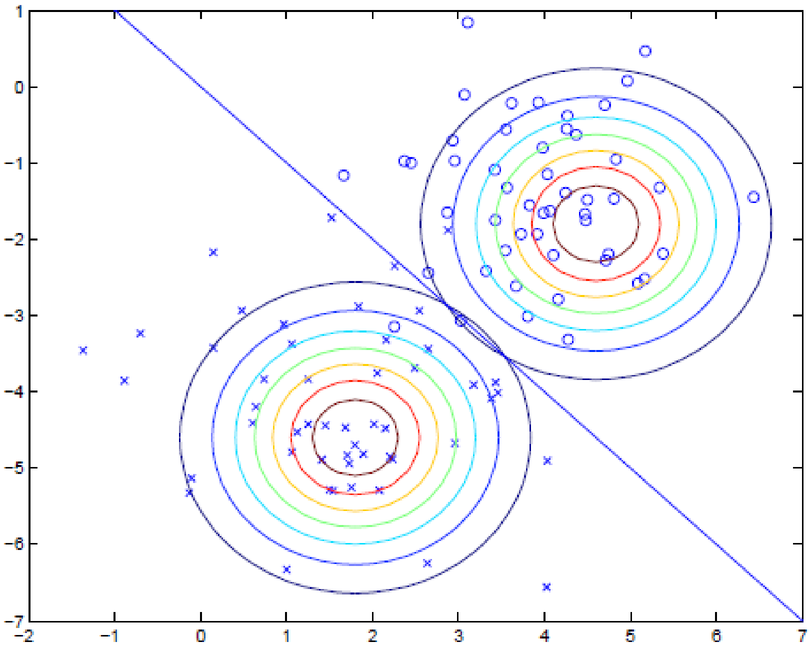
$$\begin{aligned} p(y) &= \phi^y (1-\phi)^{1-y} \\ p(x|y=0) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_0)^T \Sigma^{-1}(x-\mu_0)\right) \\ p(x|y=1) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right) \end{aligned}$$

Log Likelihood

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

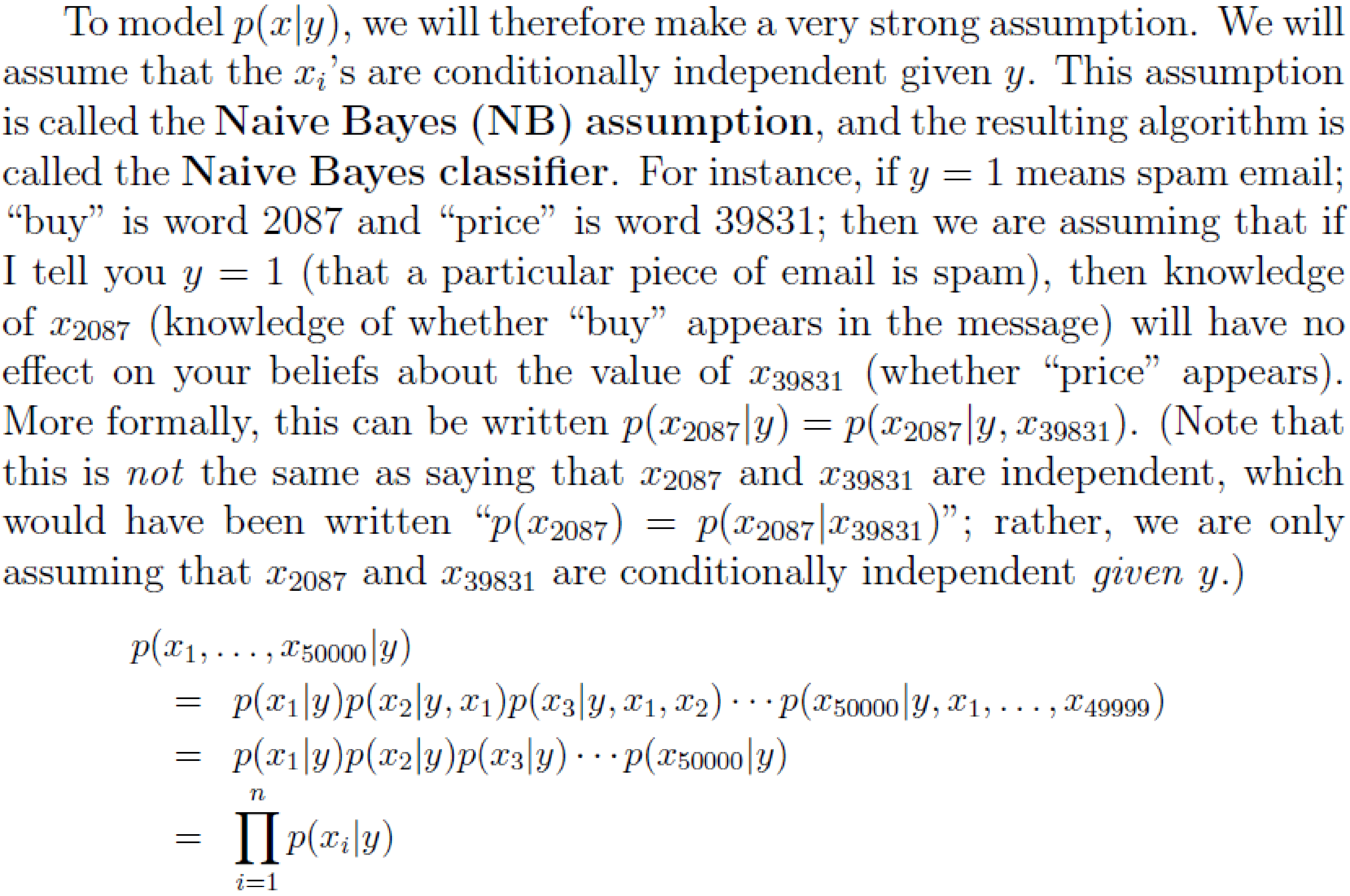
$$
\begin{aligned}
\phi &= \frac{1}{m} \sum_{i=1}^{m} 1\{y^{(i)} = 1\} \\
\mu_0 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\}} \\
\mu_1 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}} \\
\Sigma &= \frac{1}{m} \sum_{i=1}^{m} (x^{(i)}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

Pictorial Interpretation



Shown in the figure are the training set, as well as the contours of the two Gaussian distributions that have been fit to the data in each of the two classes. 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. Also shown in the figure is the straight line giving the decision boundary at which $p(y = 1|x) = 0.5$. On one side of the boundary, we'll predict $y = 1$ to be the most likely outcome, and on the other side, we'll predict $y = 0$.

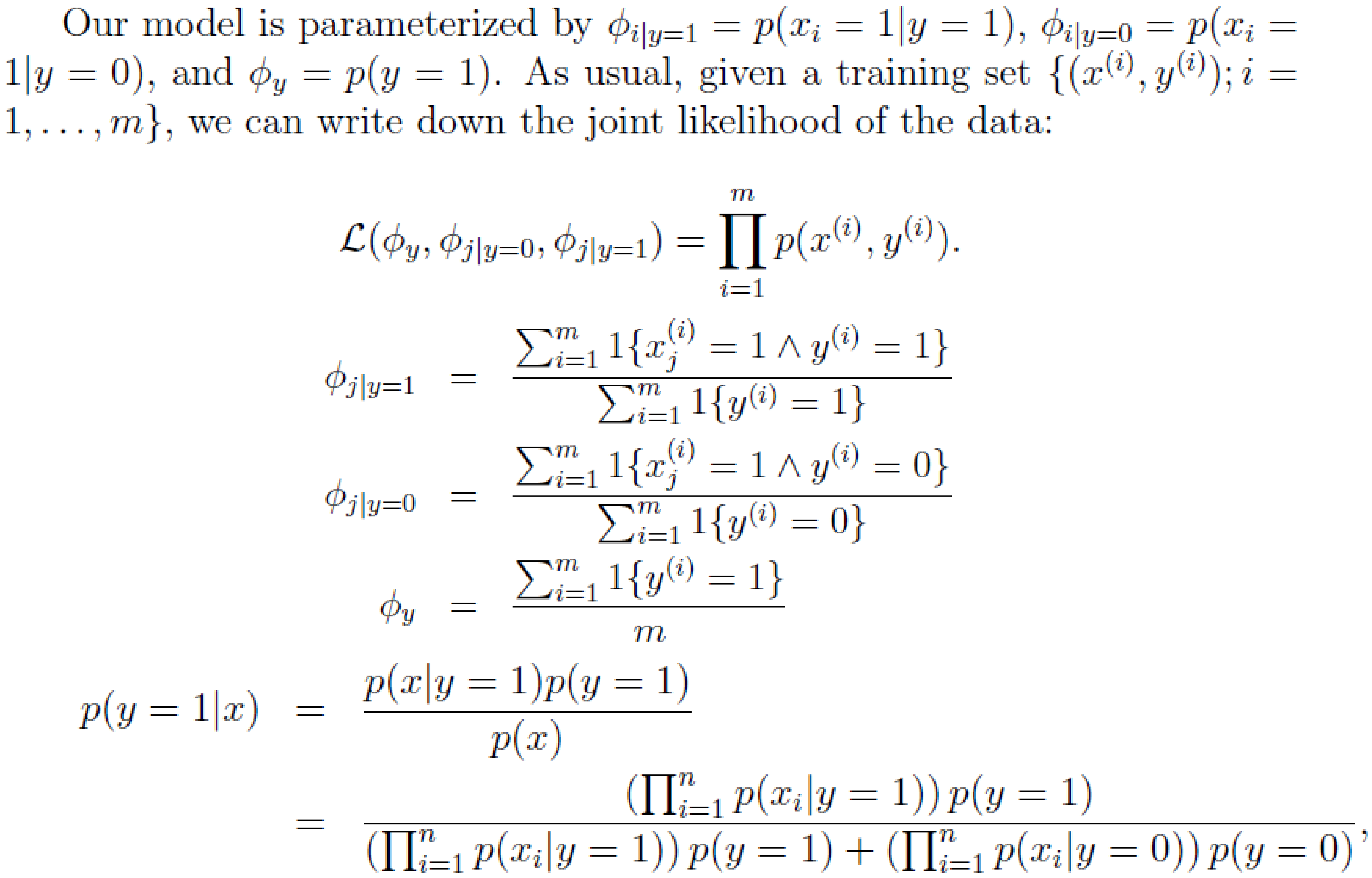
Naïve Bayes



To model $p(x|y)$, we will therefore make a very strong assumption. We will assume that the x_i's are conditionally independent given y. This assumption is called the **Naive Bayes (NB) assumption**, and the resulting algorithm is called the **Naive Bayes classifier**. For instance, if $y = 1$ means spam email; "buy" is word 2087 and "price" is word 39831; then we are assuming that if I tell you $y = 1$ (that a particular piece of email is spam), then knowledge of x_{2087} (knowledge of whether "buy" appears in the message) will have no effect on your beliefs about the value of x_{39831} (whether "price" appears). More formally, this can be written $p(x_{2087}|y) = p(x_{2087}|y, x_{39831})$. (Note that this is *not* the same as saying that x_{2087} and x_{39831} are independent, which would have been written "$p(x_{2087}) = p(x_{2087}|x_{39831})$"; rather, we are only assuming that x_{2087} and x_{39831} are conditionally independent *given* y.)

$$
\begin{aligned}
& p(x_1, \ldots, x_{50000}|y) \\
& \quad = \; p(x_1|y)p(x_2|y, x_1)p(x_3|y, x_1, x_2) \cdots p(x_{50000}|y, x_1, \ldots, x_{49999}) \\
& \quad = \; p(x_1|y)p(x_2|y)p(x_3|y) \cdots p(x_{50000}|y) \\
& \quad = \; \prod_{i=1}^{n} p(x_i|y)
\end{aligned}
$$

Naïve Bayes



Our model is parameterized by $\phi_{i|y=1} = p(x_i = 1|y = 1)$, $\phi_{i|y=0} = p(x_i = 1|y = 0)$, and $\phi_y = p(y = 1)$. As usual, given a training set $\{(x^{(i)}, y^{(i)}); i = 1, \ldots, m\}$, we can write down the joint likelihood of the data:

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

$$\begin{aligned}
\phi_{j|y=1} &= \frac{\sum_{i=1}^m 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}} \\
\phi_{j|y=0} &= \frac{\sum_{i=1}^m 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^m 1\{y^{(i)} = 0\}} \\
\phi_y &= \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\}}{m}
\end{aligned}$$

$$\begin{aligned}
p(y = 1|x) &= \frac{p(x|y = 1)p(y = 1)}{p(x)} \\
&= \frac{\left(\prod_{i=1}^n p(x_i|y = 1)\right) p(y = 1)}{\left(\prod_{i=1}^n p(x_i|y = 1)\right) p(y = 1) + \left(\prod_{i=1}^n p(x_i|y = 0)\right) p(y = 0)},
\end{aligned}$$

Laplace Smoothing

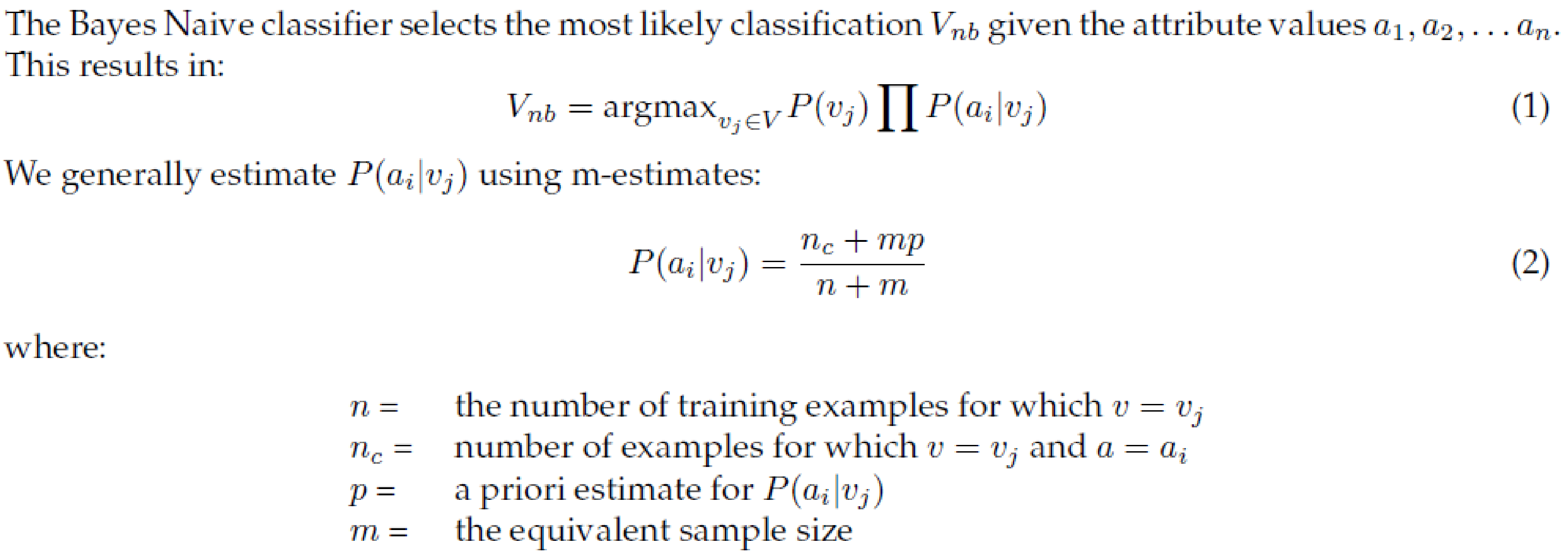
$$\begin{aligned} p(y=1|x) &= \frac{\prod_{i=1}^{n} p(x_i|y=1)p(y=1)}{\prod_{i=1}^{n} p(x_i|y=1)p(y=1) + \prod_{i=1}^{n} p(x_i|y=0)p(y=0)} \\ &= \frac{0}{0}. \end{aligned}$$

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

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

$$\begin{aligned} \phi_{j|y=1} &= \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} + 2} \\ \phi_{j|y=0} &= \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\} + 1}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} + 2} \end{aligned}$$

Example



The Bayes Naive classifier selects the most likely classification V_{nb} given the attribute values $a_1, a_2, \ldots a_n$. This results in:

$$V_{nb} = \text{argmax}_{v_j \in V} P(v_j) \prod P(a_i|v_j) \tag{1}$$

We generally estimate $P(a_i|v_j)$ using m-estimates:

$$P(a_i|v_j) = \frac{n_c + mp}{n + m} \tag{2}$$

where:

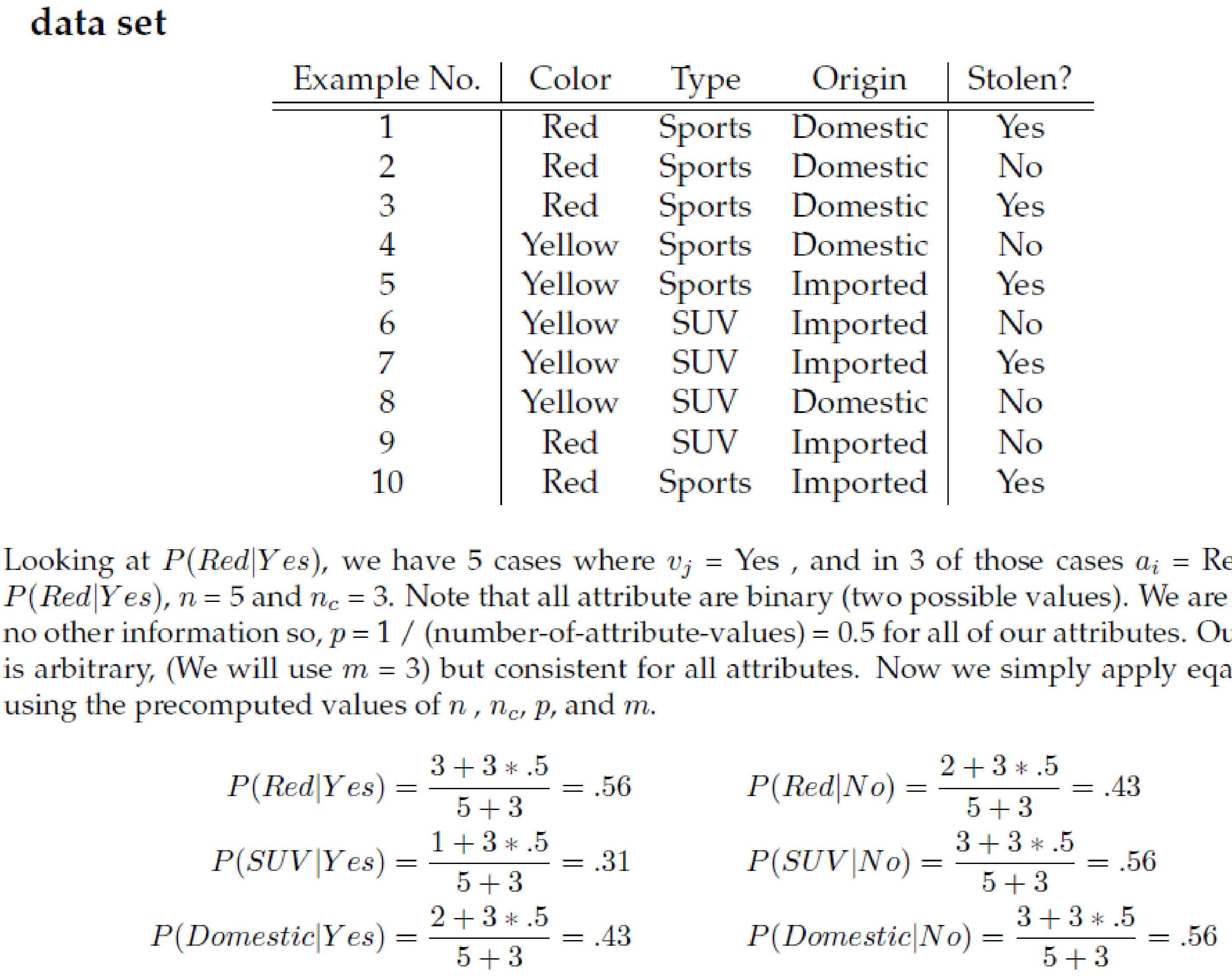
$n =$ the number of training examples for which $v = v_j$

$n_c =$ number of examples for which $v = v_j$ and $a = a_i$

$p =$ a priori estimate for $P(a_i|v_j)$

$m =$ the equivalent sample size

Example



data set

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

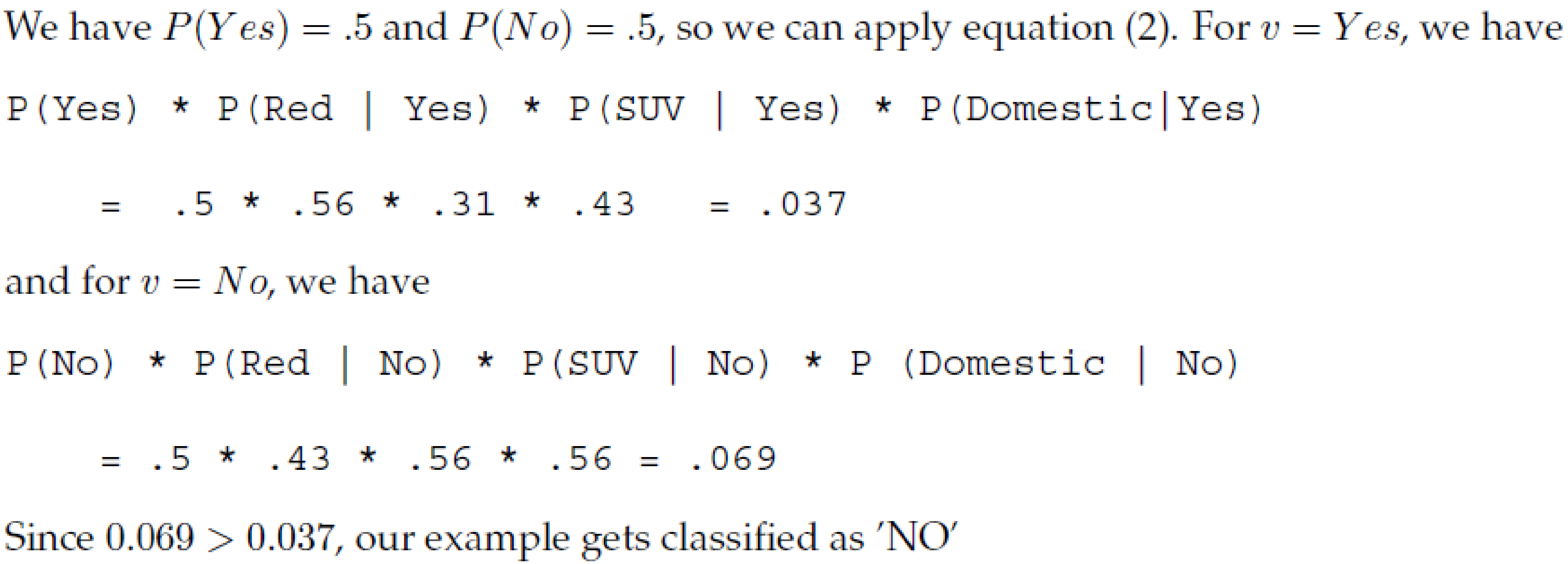
Looking at $P(Red|Yes)$, we have 5 cases where v_j = Yes , and in 3 of those cases a_i = Red. So for $P(Red|Yes)$, n = 5 and n_c = 3. Note that all attribute are binary (two possible values). We are assuming no other information so, p = 1 / (number-of-attribute-values) = 0.5 for all of our attributes. Our m value is arbitrary, (We will use m = 3) but consistent for all attributes. Now we simply apply eqauation (3) using the precomputed values of n , n_c, p, and m.

$$P(Red|Yes) = \frac{3 + 3 * .5}{5 + 3} = .56 \qquad P(Red|No) = \frac{2 + 3 * .5}{5 + 3} = .43$$

$$P(SUV|Yes) = \frac{1 + 3 * .5}{5 + 3} = .31 \qquad P(SUV|No) = \frac{3 + 3 * .5}{5 + 3} = .56$$

$$P(Domestic|Yes) = \frac{2 + 3 * .5}{5 + 3} = .43 \qquad P(Domestic|No) = \frac{3 + 3 * .5}{5 + 3} = .56$$

Example



We have $P(Yes) = .5$ and $P(No) = .5$, so we can apply equation (2). For $v = Yes$, we have

```
P(Yes) * P(Red | Yes) * P(SUV | Yes) * P(Domestic|Yes)

    =  .5 * .56 * .31 * .43   = .037
```

and for $v = No$, we have

```
P(No) * P(Red | No) * P(SUV | No) * P (Domestic | No)

    = .5 * .43 * .56 * .56 = .069
```

Since $0.069 > 0.037$, our example gets classified as 'NO'