9. Generative Classifiers

STA3142 Statistical Machine Learning

Kibok Lee

Assistant Professor of

Applied Statistics / Statistics and Data Science

Mar {26, 28}, 2024

연세대학교
YONSEI UNIVERSITY

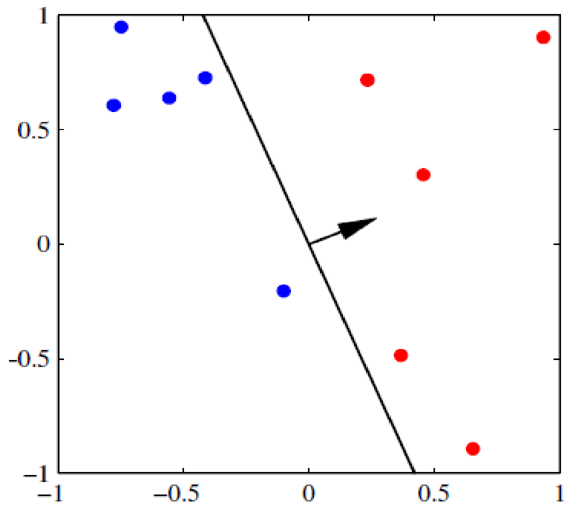
Assignment 1

- Due **Friday 3/29, 11:59pm**
- Topics
  - (Programming) NumPy basics
  - (Programming) Linear regression on a polynomial
  - (Math) Derivation and proof for linear regression

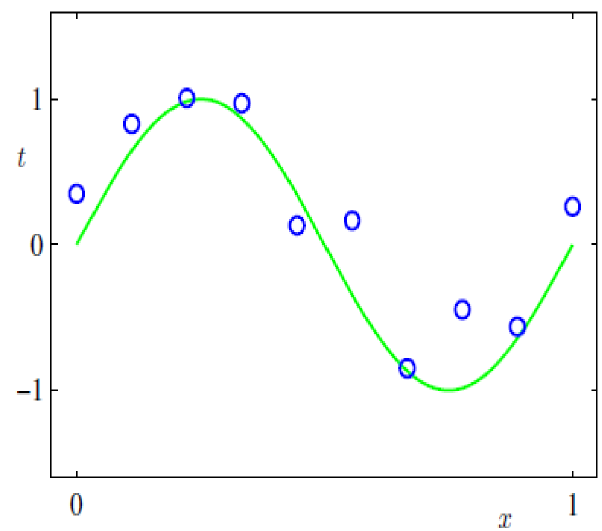
- Please read the instruction carefully!
  - Submit one pdf and one zip file separately
  - Write your code only in the designated spaces
  - Do not import additional libraries
  - …
- If you feel difficult, consider to take **option 2**.

Recap: Supervised Learning

- Learning a function $h: \mathcal{X} \to \mathcal{Y}$
- Labels could be discrete or continuous
  - Discrete labels: **classification**
  - Continuous labels: **regression**



classification



regression

Classification Strategies

- Learning the distributions $p(C_k|x)$
  - Discriminative models: Directly model $p(C_k|x)$ and learn parameters from the training set.
  - Generative models: Learn class densities $p(x|C_k)$ and priors $p(C_k)$ to obtain $p(x, C_k) = p(x|C_k)p(C_k)$
- Nearest neighbor classification
  - Given query data x, find the closest training points and do majority vote.
- Discriminant functions
  - Learn a function $h(x)$ that maps x onto some C_k.

Outline

- Generative models: Learn class densities $p(x|C_k)$ and priors $p(C_k)$ s.t. $p(x, C_k) = p(x|C_k)p(C_k)$
    - Gaussian Discriminant Analysis
    - Naïve Bayes Classifier

Probabilistic Generative Models

- Bayes' theorem reduces the classification problem $p(C_k|x)$ to estimating the distribution of the data.
- Density estimation problems are easy to learn from labeled training data.
  - Priors: $p(C_k)$
  - Class densities: $p(x|C_k)$
- Learning: Maximum likelihood estimation (MLE)
- Classification: Maximum a posteriori (MAP) estimation

$$\operatorname*{argmax}_k p(C_k|x) = \operatorname*{argmax}_k p(C_k, x)$$

Probabilistic Generative Models

- For two-class classification, Bayes' theorem says:

$$p(C_1|\mathbf{x}) = \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_1)p(C_1) + p(\mathbf{x}|C_2)p(C_2)}$$

- The posterior is then expressed as the sigmoid of log odds:

$$p(C_1|\mathbf{x}) = \frac{1}{1+\exp(-a)} = \sigma(a)$$

where $a = \ln \frac{p(C_1|\mathbf{x})}{p(C_2|\mathbf{x})} = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}$

Generative vs. Discriminative

- The **generative** approach is typically model-based and makes it possible to generate synthetic data from $p(x|C_k)$.
  - By comparing the synthetic data and real data, we get a sense of how good the generative model is.
- The **discriminative** approach typically has fewer parameters to estimate and have less assumptions about the data distribution (i.e., no $p(x, \ldots)$).
  - Linear (e.g., logistic regression) or quadratic (e.g., Gaussian discriminant analysis) in the input.
  - Less generative assumptions about the data (i.e., constructing the features may need prior knowledge)

Gaussian Discriminant Analysis

Gaussian Discriminant Analysis

- Prior distribution $p(C_k)$: Constant (e.g., Bernoulli)
- Likelihood $p(\mathbf{x}|C_k)$: Gaussian distribution

$$p(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\mathbf{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_k)^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu_k)\right\}$$

- Classification using Bayes' rule:

$$p(C_k|\mathbf{x}) = p(\mathbf{x}|C_k)p(C_k)/p(\mathbf{x})$$

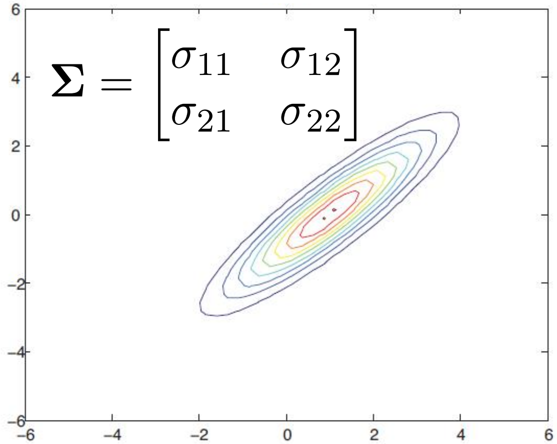
  - For two-class classification, $p(C_1|\mathbf{x}) = \sigma(a)$

where $$a = \ln \frac{p(C_1|\mathbf{x})}{p(C_2|\mathbf{x})} = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}$$

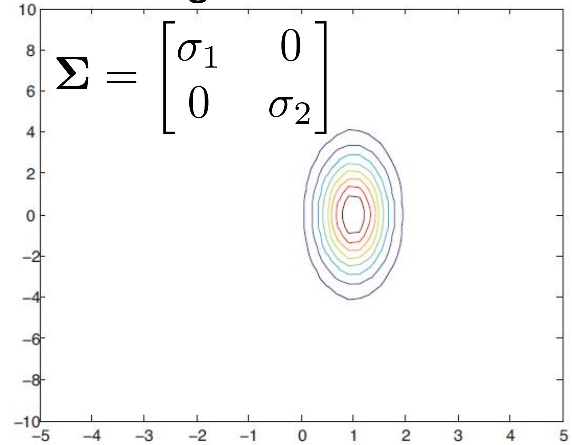
Examples of Gaussian Distributions

- Probability density $p(\mathbf{x})$ for 2-dim case

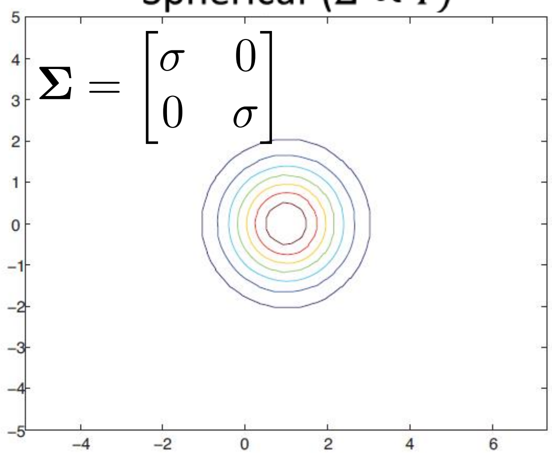
Full covariance



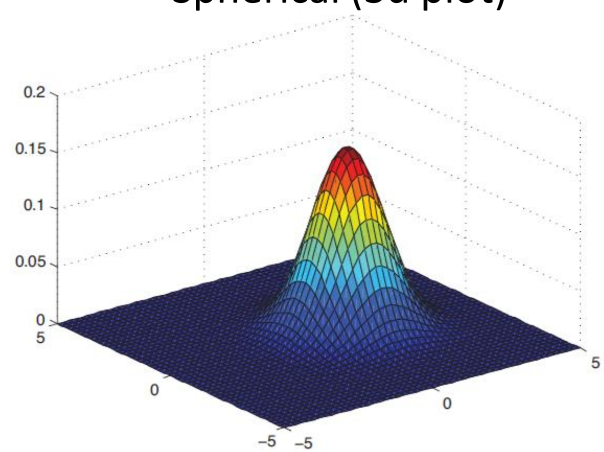
Diagonal covariance



Spherical ($\Sigma \propto I$)

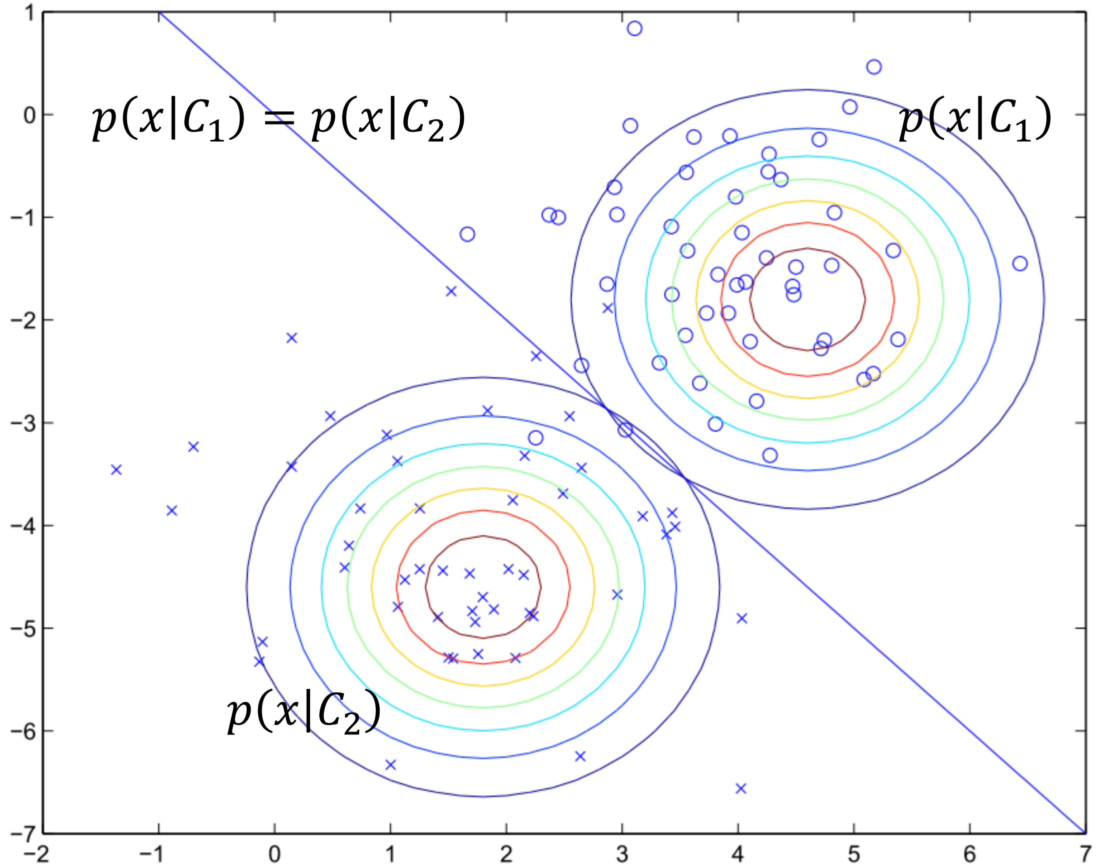


Spherical (3d plot)



Example: Two-Class GDA

- GDA assumes the **same covariance** for all classes.
  - The decision boundary is linear.
  - e.g., two-class classification



Two-Class GDA Formulation

- We model $p(\mathbf{x}|C_k)$ as Gaussian distributions with the **same covariance** matrix.

$$p(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\mathbf{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x}-\mu_k)^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu_k) \right\}$$

- Then, the posterior is derived as

$$p(C_1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + w_0)$$

where $\mathbf{w} = \mathbf{\Sigma}^{-1}(\mu_1 - \mu_2)$

$$w_0 = -\frac{1}{2}\mu_1^T \mathbf{\Sigma}^{-1}\mu_1 + \frac{1}{2}\mu_2^T \mathbf{\Sigma}^{-1}\mu_2 + \ln \frac{p(C_1)}{p(C_2)}$$

Two-Class GDA Derivation

$$
\begin{aligned}
P(x, C_1) &= P(x|C_1)P(C_1) \\
&= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right\} P(C_1) \\
P(x, C_2) &= P(x|C_2)P(C_2) \\
&= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_2)^T \Sigma^{-1}(x-\mu_2)\right\} P(C_2)
\end{aligned}
$$

$$
\log \frac{P(C_1|x)}{P(C_2|x)} = \log \frac{P(C_1|x)}{1 - P(C_1|x)}
$$

"Log-odds"

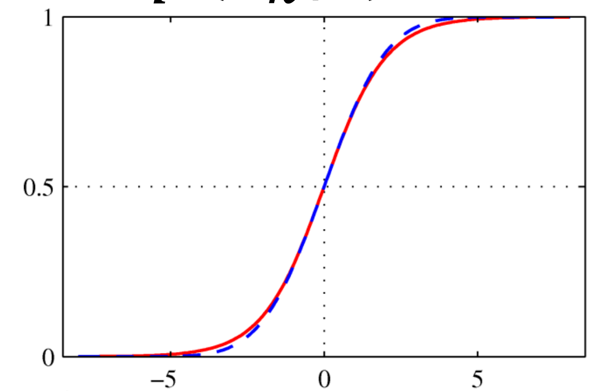
$$
\begin{aligned}
&= \log \frac{\exp\left\{-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right\}}{\exp\left\{-\frac{1}{2}(x-\mu_2)^T \Sigma^{-1}(x-\mu_2)\right\}} + \log \frac{P(C_1)}{P(C_2)} \\
&= \left\{-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right\} - \left\{-\frac{1}{2}(x-\mu_2)^T \Sigma^{-1}(x-\mu_2)\right\} + \log \frac{P(C_1)}{P(C_2)} \\
&= (\mu_1 - \mu_2)^T \Sigma^{-1} x - \frac{1}{2}\mu_1^T \Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2^T \Sigma^{-1}\mu_2 + \log \frac{P(C_1)}{P(C_2)} \\
&= \left(\Sigma^{-1}(\mu_1 - \mu_2)\right)^T x + w_0
\end{aligned}
$$

Quadratic term canceled out because of the shared covariance

$$
\text{where } w_0 = -\tfrac{1}{2}\mu_1^T \Sigma^{-1}\mu_1 + \tfrac{1}{2}\mu_2^T \Sigma^{-1}\mu_2 + \log \tfrac{P(C_1)}{P(C_2)}
$$

Extension to Multi-Class GDA

- For two-class classification, the posterior $p(C_k|\mathbf{x})$ is the sigmoid of the log odds



$$\sigma(a) = \frac{1}{1+\exp(-a)}$$

$$a = \log\left(\frac{\sigma}{1-\sigma}\right) = \left(\Sigma^{-1}(\mu_1 - \mu_2)\right)^T x + w_0$$

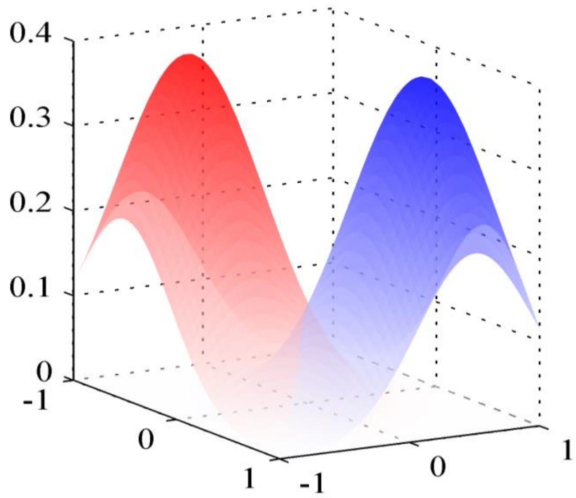
$$\text{where } w_0 = -\frac{1}{2}\mu_1\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2\Sigma^{-1}\mu_2 + \log\frac{P(C_1)}{P(C_2)}$$

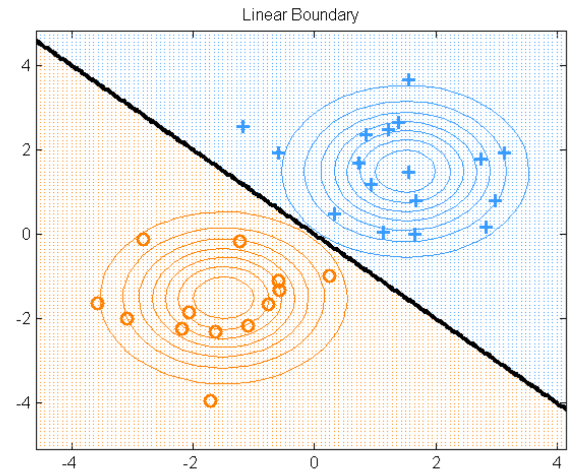
- For multi-class classification, the posterior $p(C_k|\mathbf{x})$ is softmax.

$$p_i = \frac{\exp(q_i)}{\sum_j \exp(q_j)}$$

GDA Decision Boundaries

- At decision boundary, we have $p(C_1|\mathbf{x}) = p(C_2|\mathbf{x})$
- Under the same covariance assumption, the boundary is linear.
  - Different priors $p(C_1), p(C_2)$ does not change the linearity but shift it around.





Learning GDA via MLE

- Given training data $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$ and a generative model with the shared covariance

- Priors: $p(y) = \phi^y (1-\phi)^{1-y}$ where $\phi = p(y=1)$

- Class densities:

$$p(\mathbf{x}|y=0) = \frac{1}{\sqrt{2\pi}\,|\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\mathbf{x}-\mu_0)^T \Sigma^{-1} (\mathbf{x}-\mu_0))$$

$$p(\mathbf{x}|y=1) = \frac{1}{\sqrt{2\pi}\,|\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\mathbf{x}-\mu_1)^T \Sigma^{-1} (\mathbf{x}-\mu_1))$$

Learning GDA via MLE

- Maximum likelihood estimation (MLE):

$$\phi = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{y^{(i)} = 1\}$$

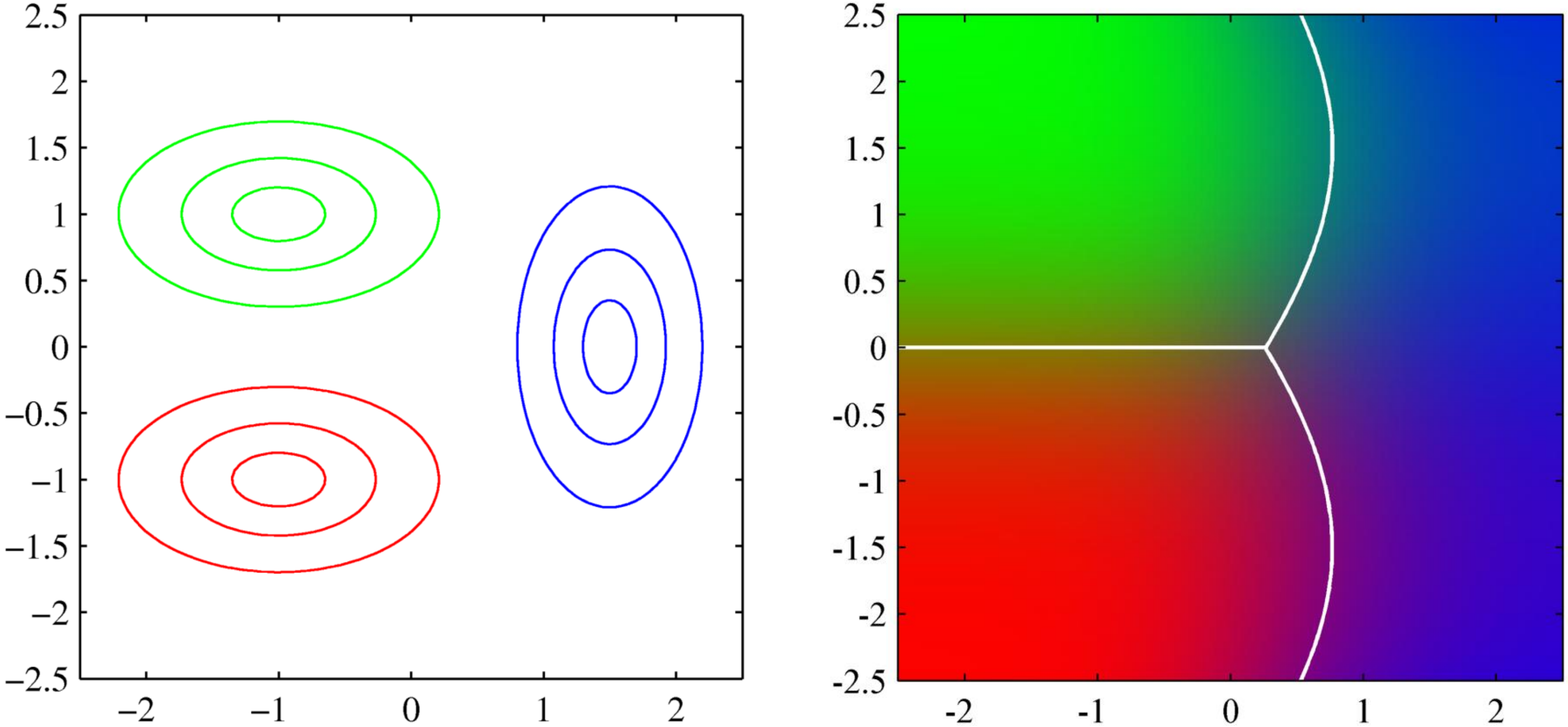
$$\mu_0 = \frac{\sum_{i=1}^{N} \mathbf{1}\{y^{(i)} = 0\}\mathbf{x}^{(i)}}{\sum_{i=1}^{N} \mathbf{1}\{y^{(i)} = 0\}}$$

$$\mu_1 = \frac{\sum_{i=1}^{N} \mathbf{1}\{y^{(i)} = 1\}\mathbf{x}^{(\mathrm{i})}}{\sum_{i=1}^{N} \mathbf{1}\{y^{(i)} = 1\}}$$

$$\sum = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}^{(i)} - \mu_{y^{(i)}})(\mathbf{x}^{(i)} - \mu_{y^{(i)}})^T$$

GDA with Different Covariance

- Decision boundaries are quadratic when each class has different covariance.



GDA vs. Logistic Regression

- GDA requires $O(M^2)$ learnable parameters
  - $2M$ parameters for the means of $p(\mathbf{x}|C_1)$ and $p(\mathbf{x}|C_2)$
  - $M(M+1)/2$ parameters for shared covariance matrix
  - Cf. Logistic regression requires only M parameters

- GDA has a strong modeling assumption and works well when the distribution follows the assumption.
  - Cf. Logistic regression has less parameters and is more flexible about data distribution.

Naïve Bayes Classifier

Naïve Bayes Classifier

- Prior distribution $p(C_k)$: Constant (e.g., Bernoulli)
- Likelihood $p(\mathbf{x}|C_k)$:

$$P(x_1, ..., x_M|C_k) = P(x_1|C_k) \cdots P(x_M|C_k) = \prod_{j=1}^{M} P(x_j|C_k)$$

  - Naïve Bayes assumption: Each coordinate of x is conditionally independent of other coordinates given the class label.

- Classification using Bayes' rule:

$$p(C_k|\mathbf{x}) = p(\mathbf{x}|C_k)p(C_k)/p(\mathbf{x})$$

  - For two-class classification,

$$P(C_1|\mathbf{x}) = \frac{P(C_1, \mathbf{x})}{P(\mathbf{x})} = \frac{P(C_1, \mathbf{x})}{P(C_1, \mathbf{x}) + P(C_2, \mathbf{x})}$$

Naïve Bayes Classifier

- Classification using Bayes' rule:

$$p(C_k|\mathbf{x}) = p(\mathbf{x}|C_k)p(C_k)/p(\mathbf{x})$$

- Classification is done by the MAP estimation:

$$\arg\max_k P(C_k|\mathbf{x}) = \arg\max_k P(C_k, \mathbf{x})$$

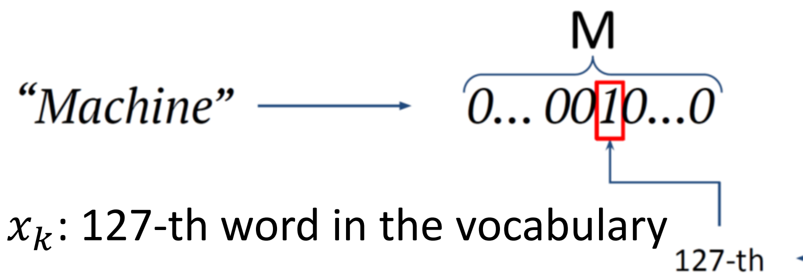
$$= \arg\max_k P(C_k)P(\mathbf{x}|C_k)$$

Naïve Bayes Assumption

$$= \arg\max_k P(C_k) \prod_{j=1}^{M} P(x_j|C_k)$$

Example: Spam Mail Classification

- Label: $y = 1$ (spam), $y = 0$ (ham or non-spam)
- Features $\mathbf{x} = [x_1, x_2, \ldots]$
  - x_k: k-th word in a mail, where M is the vocabulary size
  - Each word is represented as **one-hot encoding**.



	Vocabulary			
Index	…	127	128	…
Word	…	*Machine*	*Mail*	

- Naïve Bayes assumption: Given a class label y, each word in a mail is an independent **multinomial** variable.

Naïve Bayes Classifier Formulation

- Prior: $P(\text{spam}) = Bernoulli(\phi)$
- Likelihood: $P(\text{word}|\text{spam}) = Multinomial(\mu_1^s, \ldots, \mu_M^s)$

  $P(\text{word}|\text{nonspam}) = Multinomial(\mu_1^{ns}, \ldots, \mu_M^{ns})$
- Learning to find ϕ, μ^s, μ^{ns} that best fits the training data $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$

$$\prod_{i=1}^{N} P(\mathbf{x}^{(i)}, y^{(i)})$$

$$= \prod_{i=1}^{N} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)})$$

$$= \boxed{\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)})\right)} \boxed{\left(\prod_{i:y^{(i)}=0} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)})\right)}$$

Spam　　　　Ham (Non-spam)

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right)$$

$$x_k^{(i)}$$

i-th mail

k-th word

- Naïve Bayes assumption:
  - Prior: $P(\text{spam}) = Bernoulli(\phi)$

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

  - Likelihood: $P(\text{word}|\text{spam}) = Multinomial(\mu_1^s, \ldots, \mu_M^s)$

$$\begin{aligned} P(x^{(i)}|y^{(i)} = 1) \quad &= \quad \prod_{k=1}^{len(x^{(i)})} P(x_k^{(i)}|y^{(i)} = 1) \\ &= \quad \prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)} = j\text{-th word}\right)} \end{aligned}$$

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N} \left(\prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} \left(\mu_j^s\right)^{I\left(x_k^{(i)}=j\text{-th word}\right)} \right) \phi \right)$$

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)})P(y^{(i)}) \right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N} \left(\prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)} \right) \phi \right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N} \prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)} \right) \left(\prod_{i:y^{(i)}=1}^{N} \phi \right)$$

$$\prod_i a_i b = \prod_i a_i \prod_i b$$

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)})P(y^{(i)})\right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N}\left(\prod_{k=1}^{len(x^{(i)})}\prod_{j=1}^{M}(\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)}\right)\phi\right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N}\prod_{k=1}^{len(x^{(i)})}\prod_{j=1}^{M}(\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)}\right)\left(\prod_{i:y^{(i)}=1}^{N}\phi\right)$$

$$= \left(\prod_{j=1}^{M}(\mu_j^s)^{\sum_{i:y^{(i)}=1}^{N}\sum_{k=1}^{len(x^{(i)})} I\left(x_k^{(i)}=j\text{-th word}\right)}\right)\left(\prod_{i:y^{(i)}=1}^{N}\phi\right)$$

$$\prod_i a_j^b = a_j^{\sum_i b}$$

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N} \left(\prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)} \right) \phi \right)$$

$$= \left(\prod_{i:y^{(i)}=1}^{N} \prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)}=j\text{-th word}\right)} \right) \left(\prod_{i:y^{(i)}=1}^{N} \phi \right)$$

$$= \left(\prod_{j=1}^{M} (\mu_j^s)^{\sum_{i:y^{(i)}=1}^{N} \boxed{\sum_{k=1}^{len(x^{(i)})} I\left(x_k^{(i)}=j\text{-th word}\right)}} \right) \left(\prod_{i:y^{(i)}=1}^{N} \phi \right)$$

For i-th email,
count # of j-th word in the vocabulary

$$= \left(\prod_{j=1}^{M} (\mu_j^s)^{N_j^{spam}} \right) \phi^{N^{spam}}$$

N_j^{spam}: Total # of j-th word in spam emails
N^{spam}: Total # of spam emails

Naïve Bayes Classifier Derivation

- Likelihood for spam:

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right)$$

$$x_k^{(i)}$$

i-th mail

k-th word

- Naïve Bayes assumption:
  - Prior: $P(\text{spam}) = Bernoulli(\phi)$

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

  - Likelihood: $P(\text{word}|\text{spam}) = Multinomial(\mu_1^s, \ldots, \mu_M^s)$

$$P(x^{(i)}|y^{(i)} = 1) = \prod_{k=1}^{len(x^{(i)})} P(x_k^{(i)}|y^{(i)} = 1)$$

$$= \prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^s)^{I\left(x_k^{(i)} = j\text{-th word}\right)}$$

Naïve Bayes Classifier Derivation

- Likelihood for ham (non-spam):

$$\left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)})P(y^{(i)})\right)\left(\prod_{i:y^{(i)}=0} P(\mathbf{x}^{(i)}|y^{(i)})P(y^{(i)})\right)$$

- Naïve Bayes assumption:
  - Prior: $P(\text{spam}) = Bernoulli(\phi)$

$$P(y^{(i)}=0) = 1-\phi$$

  - Likelihood: $P(\text{word}|\text{nonspam}) = Multinomial(\mu_1^{ns},\ldots,\mu_M^{ns})$

$$\begin{aligned} P(x^{(i)}|y^{(i)}=0) &= \prod_{k=1}^{len(x^{(i)})} P(x_k^{(i)}|y^{(i)}=0) \\ &= \prod_{k=1}^{len(x^{(i)})} \prod_{j=1}^{M} (\mu_j^{ns})^{I\left(x_k^{(i)}=j\text{-th word}\right)} \end{aligned}$$

Naïve Bayes Classifier Derivation

- Putting together:

$$\prod_{i=1}^{N} P(\mathbf{x}^{(i)}, y^{(i)})$$

$$= \left(\prod_{i:y^{(i)}=1} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right) \left(\prod_{i:y^{(i)}=0} P(\mathbf{x}^{(i)}|y^{(i)}) P(y^{(i)}) \right)$$

$$= \left(\phi^{N^{spam}} \prod_{word\, j} (\mu_j^s)^{N_j^{spam}} \right) \left((1-\phi)^{N^{nonspam}} \prod_{word\, j} (\mu_j^{ns})^{N_j^{nonspam}} \right)$$

- Log-likelihood

$$\begin{aligned} & \log P(\mathcal{D}) \\ = \quad & \log \prod_{i=1}^{N} P(x^{(i)}, y^{(i)}) \\ = \quad & N^{spam} \log \phi + \sum_{word\, j} N_j^{spam} \log \mu_j^s + N^{nonspam} \log(1-\phi) + \sum_{word\, j} N_j^{nonspam} \log \mu_j^{ns} \end{aligned}$$

Learning NB Classifier via MLE

- Log-likelihood

$$
\begin{aligned}
& \log P(\mathcal{D}) \\
= \quad & \log \prod_{i=1}^{N} P(x^{(i)}, y^{(i)}) \\
= \quad & N^{spam} \log \phi + \sum_{word\, j} N_j^{spam} \log \mu_j^s + N^{nonspam} \log(1-\phi) + \sum_{word\, j} N_j^{nonspam} \log \mu_j^{ns}
\end{aligned}
$$

- Maximum (log-)likelihood estimation
  - Take the derivative of log-likelihood with respect to the parameters $\{\phi, \mu^s, \mu^{ns}\}$, and set it to zero.

Learning NB Classifier via MLE

- Find ϕ

$$\log P(\mathcal{D})$$

$$= \quad N^{spam} \log \phi + \sum_{word\, j} N_j^{spam} \log \mu_j^s + N^{nonspam} \log(1-\phi) + \sum_{word\, j} N_j^{nonspam} \log \mu_j^{ns}$$

$$\Rightarrow \qquad \frac{\partial l}{\partial \phi} = \frac{1}{\phi} N^{spam} - \frac{1}{1-\phi} N^{nonspam} = 0$$

$$\phi = \frac{N^{spam}}{N^{spam} + N^{nonspam}}$$

Learning NB Classifier via MLE

- Find μ^s (or similarly, μ^{ns})
  - $\{\mu^s\}$'s are NOT independent to each other; $\sum_{j=1}^{M} \mu_j^s = 1$
  - To only deal with variables independent to each other, we can express μ_M^s as $1 - \sum_{j=1}^{M-1} \mu_j^s$

$$\Downarrow$$

$$\sum_{word\, j=1}^{M} N_j^{spam} \log \mu_j^s = \sum_{word\, j=1}^{M-1} N_j^{spam} \log \mu_j^s + N_M^{spam} \log(1 - \sum_{j=1}^{M-1} \mu_j^s)$$

Learning NB Classifier via MLE

- Find μ^s (or similarly, μ^{ns})
  - $\{\mu^s\}$'s are NOT independent to each other; $\sum_{j=1}^{M} \mu_j^s = 1$
  - To only deal with variables independent to each other, we can express μ_M^s as $1 - \sum_{j=1}^{M-1} \mu_j^s$

$$\sum_{word\,j=1}^{M} N_j^{spam} \log \mu_j^s = \sum_{word\,j=1}^{M-1} N_j^{spam} \log \mu_j^s + N_M^{spam} \log(1 - \sum_{j=1}^{M-1} \mu_j^s)$$

Take derivative

$$\frac{\partial}{\partial \mu_j^s}\left(\sum_{word\,j=1}^{M} N_j^{spam} \log \mu_j^s\right) = \frac{N_j^{spam}}{\mu_j^s} - \boxed{\frac{N_M^{spam}}{1 - \sum_{j=1}^{M-1} \mu_j^s}} = 0$$

Constant w.r.t. j

Learning NB Classifier via MLE

- Find μ^s (or similarly, μ^{ns})
  - $\{\mu^s\}$'s are NOT independent to each other; $\sum_{j=1}^{M} \mu_j^s = 1$
  - To only deal with variables independent to each other, we can express μ_M^s as $1 - \sum_{j=1}^{M-1} \mu_j^s$

$$\sum_{word\, j=1}^{M} N_j^{spam} \log \mu_j^s = \sum_{word\, j=1}^{M-1} N_j^{spam} \log \mu_j^s + N_M^{spam} \log(1 - \sum_{j=1}^{M-1} \mu_j^s)$$

$$\frac{\partial}{\partial \mu_j^s}\left(\sum_{word\, j=1}^{M} N_j^{spam} \log \mu_j^s\right) = \frac{N_j^{spam}}{\mu_j^s} - \boxed{\frac{N_M^{spam}}{1 - \sum_{j=1}^{M-1} \mu_j^s}} = 0$$

Constant w.r.t. j

$$\frac{a}{b} = \frac{c}{d} = \frac{a+c}{b+d}$$

$$\frac{N_j^{spam}}{\mu_j^s} = \text{constant} = \frac{\sum_{j=1}^{M} N_j^{spam}}{\sum_{j=1}^{M} \mu_j^s} = \sum_{j=1}^{M} N_j^{spam} = N^{spam}$$

Learning NB Classifier via MLE

- Find μ^s (or similarly, μ^{ns})
  - $\{\mu^s\}$'s are NOT independent to each other; $\sum_{j=1}^{M} \mu_j^s = 1$
  - To only deal with variables independent to each other, we can express μ_M^s as $1 - \sum_{j=1}^{M-1} \mu_j^s$

$$\sum_{word\, j=1}^{M} N_j^{spam} \log \mu_j^s = \sum_{word\, j=1}^{M-1} N_j^{spam} \log \mu_j^s + N_M^{spam} \log(1 - \sum_{j=1}^{M-1} \mu_j^s)$$

$$\frac{\partial}{\partial \mu_j^s} \left(\sum_{word\, j=1}^{M} N_j^{spam} \log \mu_j^s \right) = \frac{N_j^{spam}}{\mu_j^s} - \frac{N_M^{spam}}{1 - \sum_{j=1}^{M-1} \mu_j^s} = 0$$

$$\therefore \quad \mu_j^s = \frac{N_j^{spam}}{\sum_j N_j^{spam}}$$

Learning NB Classifier via MLE

- Summary

$$P(spam) = \phi = \frac{N^{spam}}{N^{spam} + N^{nonspam}}$$

$$P(word = j|spam) = \mu_j^s = \frac{N_j^{spam}}{\sum_j N_j^{spam}}$$

$$P(word = j|non - spam) = \mu_j^{ns} = \frac{N_j^{nonspam}}{\sum_j N_j^{nonspam}}$$

  - N^{spam}: Total # of spam emails
  - $N^{nonspam}$: Total # of ham (non-spam) emails
  - N_j^{spam}: Total # of j-th word in spam emails
  - $N_j^{nonspam}$: Total # j-th word in ham (non-spam) emails

Laplace Smoothing

- Maximum likelihood is problematic when a specific word count is 0.
  - Leads to probability of the specific word 0
- Solution: Put **imaginary** counts for each word
  - Prevent zero probability estimates (overfitting)
  - Add "1" as imaginary count for each word

$$P(spam) = \phi = \frac{N^{spam}}{N^{spam} + N^{nonspam}}$$

$$P(word = j|spam) = \mu_j^s = \frac{N_j^{spam} \boxed{+1}}{\sum_j N_j^{spam} \boxed{+M}}$$

$$P(word = j|non-spam) = \mu_j^{ns} = \frac{N_j^{nonspam} \boxed{+1}}{\sum_j N_j^{nonspam} \boxed{+M}}$$

Laplace Smoothing

- Maximum likelihood is problematic when a specific word count is 0.
  - Leads to probability of the specific word 0
- Solution: Put **imaginary** counts for each word
  - Prevent zero probability estimates (overfitting)
  - Add "1" as imaginary count for each word

$$P(spam) = \phi = \frac{N^{spam}}{N^{spam} + N^{nonspam}}$$

  - If we smooth prior as well:

$$P(spam) = \phi = \frac{N^{spam} \boxed{+1}}{N^{spam} + N^{nonspam} \boxed{+2}}$$

    - (Don't have to do this)

Next: Other Classifiers