# Supervised Learning

Hong Chang

Institute of Computing Technology,
Chinese Academy of Sciences

Pattern Recognition and Machine Learning (Fall 2022)

# Outline I

1. Linear Models for Regression
   - Linear Regression
   - Probabilistic Interpretation
   - Generalized Linear Regression

2. Discriminative Classification
   - Logistic Regression

3. Generative Classification
   - Gaussian Discriminative Analysis
   - Naive Bayes

# A Regression Example

- Consider the house price problem as follows:

| living area ($m^2$) | location | price (RMB10,000) |
|---|---|---|
| 65 | Zhongguancun | 350 |
| 100 | Shangdi | 320 |
| 105 | Shangdi | 380 |
| 110 | Changping | 168 |
| $\vdots$ | $\vdots$ | $\vdots$ |

- Input samples: $\mathcal{X} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$
  - Features include $x_1^{(i)}$: living area; $x_2^{(i)}$: location
- Output target: $\mathcal{Y} = \{y^{(1)}, \dots, y^{(N)}\}$
- Linear regression model:

$$f_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1 x_1 + \ldots + w_D x_D = \sum_{j=0}^{D} w_j x_j = \mathbf{w}^T \mathbf{x}$$

  - $\mathbf{w} = (w_0, w_1, \dots, w_D)^T$, $\mathbf{x} = (1, x_1, \dots, x_D)^T$

# Least Mean Square Algorithm

- Cost function:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)})^2$$

- Gradient descent:

$$w_j := w_j - \alpha \frac{\partial}{\partial w_j} J(\mathbf{w})$$

- For a single training example:

$$w_j := w_j + \alpha(y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)}))x_j^{(i)}$$

# Least Mean Square Algorithm (2)

- For a training set of more than one example:
  - Batch gradient descent:

```
Repeat until convergence {
        $w_j := w_j + \alpha \sum_{i=1}^{N} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)})) x_j^{(i)}$
}
```

  - Stochastic gradient descent:

```
Loop {
        For $i = 1$ to $N$ {
                $w_j := w_j + \alpha (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)})) x_j^{(i)}$
        }
}
```

# Probabilistic Assumptions

- Relate the inputs and target via:

$$y^{(i)} = \mathbf{w}^T \mathbf{x}^{(i)} + \epsilon^{(i)}$$

with $\epsilon^{(i)}$ IID Gaussian distributed, i.e.,

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(\epsilon^{(i)})^2}{2\sigma^2})$$

- Thus,

$$p(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y^{(i)} - \mathbf{w}^T \mathbf{x}^{(i)})^2}{2\sigma^2})$$

- Likelihood function:

$$L(\mathbf{w}) = L(\mathbf{w}; \mathbf{X}, \mathbf{y}) = p(\mathbf{y}|\mathbf{X}; \mathbf{w}) = \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w})$$

# Maximum Likelihood

- Log likelihood

$$\begin{aligned}\mathcal{L}(\mathbf{w}) = \log L(\mathbf{w}) &= \log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2}{2\sigma^2}\right) \\ &= N \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \cdot \frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2\end{aligned}$$

- Maximizing $\mathcal{L}(\mathbf{w})$ gives the same answer as minimizing

$$\frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2$$

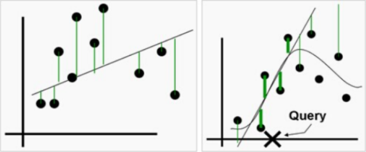
- Under the previous probabilistic assumptions on the data, least-square regression corresponds to finding the maximum likelihood estimate of $\mathbf{w}$.

# Locally Weighted Linear Regression (LWR)

- The LWR algorithm:
  1. Fit $\mathbf{w}$ to minimize $\sum_i \theta^{(i)}(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2$
  2. Output $\mathbf{w}^T\mathbf{x}$
- A standard choice for the weights is:

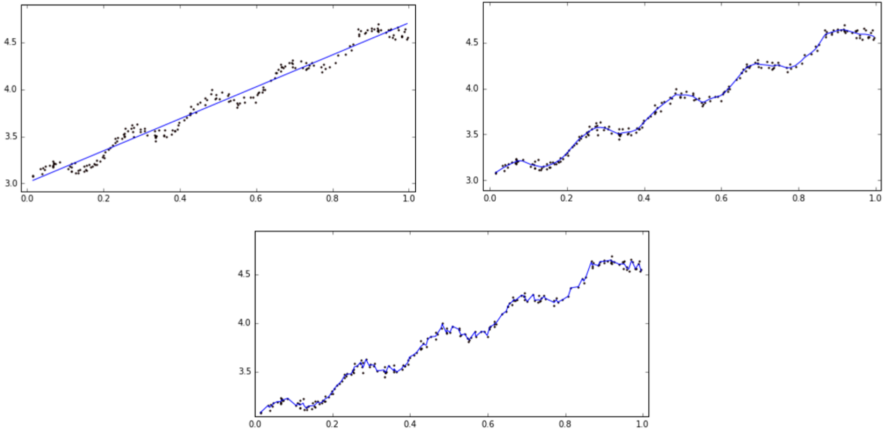
$$\theta^{(i)} = \exp(-\frac{\|\mathbf{x}^{(i)} - \mathbf{x}\|^2}{2\tau^2})$$

  - More general form is possible
  - The weights depend on the particular point $\mathbf{x}$ at which we are trying to evaluate.
- LWR is an example of non-parametric algorithm.



# Locally Weighted Linear Regression (LWR)

- The effect of $\tau$ in the weight function:



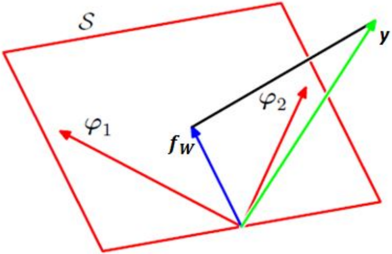
# Linear Regression with Nonlinear Basis

- Consider linear combinations of nonlinear basis functions of the input variables:

$$f_{\mathbf{w}}(\mathbf{x}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

  - $\phi = (\phi_0, \phi_1, \ldots, \phi_{M-1}), \phi_0(\mathbf{x}) = 1$
- Examples of basis functions:
  - $\phi(x) = (1, x, x^2, \ldots, x^{M-1})$
  - $\phi_j(x) = \exp(-\frac{(x-\mu_j)^2}{2s^2})$
- Can still use least squares method to estimate
- Linear method can model nonlinear functions through using nonlinear basis functions.

# Geometry of Least Squares



- $\varphi_{j+1} = [\phi_j(\mathbf{x}^{(1)}), \ldots, \phi_j(\mathbf{x}^{(N)})] \in \mathbb{R}^N$, $j = 0, \ldots, M-1$
- If $M < N$, vectors $\varphi_j$ span a linear subspace $\mathcal{S}$ of dimensionality $M$
- $f_\mathbf{w}(\mathbf{X}) \in \mathbb{R}^N$ is linear combination of vectors $\varphi_j$
- The least-squares regression function is the orthogonal projection of the target vector **y** onto the subspace $\mathcal{S}$

# Two-Class Example

- Let $P(y=1|\mathbf{x}) = t, P(y=0|\mathbf{x}) = 1-t$
- Classification rule:

$$\text{Choose} = \begin{cases} y=1 & \text{if } t > 0.5 \\ y=0 & \text{otherwise} \end{cases}$$

- Equivalent test for classification rule: choose $y=1$ if

$$\frac{t}{1-t} > 1 \text{ or } \log\frac{t}{1-t} > 0$$

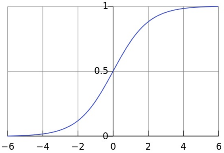
where $\frac{t}{1-t}$ is called the odds of $t$ and $\log\frac{t}{1-t}$ is called the log odds or logit function of $t$.

# Logistic Regression

- The posterior probability (or the hypothesis) is written as:

$$P(y=1|\mathbf{x}) = f_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^T\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$

where $g(\cdot)$ is called logistic or sigmoid function.



- The derivative of the sigmoid function:

$$g'(z) = g(z)(1-g(z))$$

- Note that this is a model for classification rather than regression.

# Parameter Estimation

- Since

$$\begin{aligned} P(y=1|\mathbf{x};\mathbf{w}) &= f_{\mathbf{w}}(\mathbf{x}) \\ P(y=0|\mathbf{x};\mathbf{w}) &= 1-f_{\mathbf{w}}(\mathbf{x}), \end{aligned}$$

  thus

$$P(y|\mathbf{x};\mathbf{w}) = (f_{\mathbf{w}}(\mathbf{x}))^y(1-f_{\mathbf{w}}(\mathbf{x}))^{1-y}.$$

- The likelihood of the parameters (given $N$ training examples):

$$L(\mathbf{w}) = \prod_{i=1}^{N} P(y^{(i)}|\mathbf{x}^{(i)};\mathbf{w}) = \prod_{i=1}^{N}(f_{\mathbf{w}}(\mathbf{x}^{(i)}))^{y^{(i)}}(1-f_{\mathbf{w}}(\mathbf{x}^{(i)}))^{1-y^{(i)}}$$

- The log likelihood function:

$$\mathcal{L}(\mathbf{w}) = \log L(\mathbf{w}) = \sum_{i=1}^{N}\{y^{(i)}\log f_{\mathbf{w}}(\mathbf{x}^{(i)}) + (1-y^{(i)})\log(1-f_{\mathbf{w}}(\mathbf{x}^{(i)}))\}$$

# Parameter Estimation (2)

- As for one training example $(x, y)$, maximizing the log likelihood gives

$$\frac{\partial}{\partial w_j} \mathcal{L}(\mathbf{w}) = (y - f_{\mathbf{w}}(x)) x_j$$

  - Use the fact that $f_{\mathbf{w}}(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x})$ and $g'(z) = g(z)(1 - g(z))$
- This therefore gives the stochastic gradient ascent rule:

$$w_j := w_j + \alpha(y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)})) x_j^{(i)}$$

- The solution is identical to least mean squares update rule for linear regression!

# Logistic Discrimination

- Logistic discrimination does not model the class-conditional densities $p(\mathbf{x}|C_k)$ but their ratio.
- Parametric classification approach (studied later) learns the classifier by estimating the parameters of $p(\mathbf{x}|C_k)$; logistic discrimination estimates the parameters of the discriminant function directly.

# Discriminative vs. Generative classification (Revisited)

- Discriminative classification: model the posterior probabilities or learn discriminant function directly.
  - Compute $P(C_k|\mathbf{x})$: such as logistic regression.
  - Compute discriminant function: such as perceptron, SVM.
  - Makes an assumption on the form of the discriminants, but not the densities.
- Generative classification: explicitly or implicitly model the distribution of inputs as well as outputs.
  - Assume a model for the class-conditional probability densities $p(\mathbf{x}|C_k)$.
  - Estimate $p(\mathbf{x}|C_k)$ and $P(C_k)$ from data, and apply Bayes' rule to compute the posterior probabilities $P(C_k|\mathbf{x})$.
  - Perform optimal classification based on $P(C_k|\mathbf{x})$.

# Bernoulli and Multinomial

- Bernoulli: the distribution for a single binary variable $x \in \{0, 1\}$, governed by a single continuous parameter $\beta \in [0, 1]$ which represents the probability of $x = 1$.

$$Bern(x|\beta) \quad = \quad \beta^x (1-\beta)^{1-x}$$

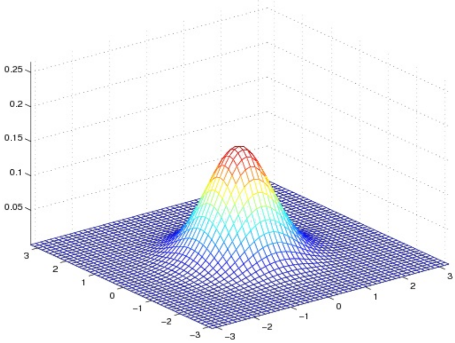
- Binomial: gives the probability of observing $m$ occurrences of $x = 1$ in a set of $N$ samples from a Bernoulli distribution.

$$Bin(m|N, \beta) = \binom{N}{m} \beta^m (1-\beta)^{N-m}$$

- Multinomial: multivariate generalization of binomial, gives the distribution over counts $m_k$ for $K-$state discrete variable to be in state $k$:

$$Mult(m_1, \ldots, m_K|N, \boldsymbol{\beta}) = \binom{N}{m_1 m_2 \ldots m_K} \prod_{k=1}^{K} \beta_k^{m_k}$$

# Multivariate Normal Distribution



$$
\begin{aligned}
\mathbf{x} &\sim \mathcal{N}_D(\boldsymbol{\mu}, |\boldsymbol{\Sigma}) \\
p(\mathbf{x}) &= \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))
\end{aligned}
$$

# Multivariate Normal Distribution (2)

- Mahalanobis distance measures the distance from $\mathbf{x}$ to $\boldsymbol{\mu}$ in terms of $\boldsymbol{\Sigma}$:

$$(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$$

- $(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) = c^2$ is the D-dimensional hyperellipsoid centered at $\boldsymbol{\mu}$. Its shape and orientation are defined by $\boldsymbol{\Sigma}$.
- Euclidean distance is a special case of Mahalanobis distance when $\boldsymbol{\Sigma} = s^2\mathbf{I}$; the hyperellipsoid degenerates into a hypersphere.

# Gaussian Discriminant Analysis (GDA)

- GDA models $p(\mathbf{x}|y)$ using a multivariate normal distribution:

$$\begin{aligned} y &\sim \text{Bernoulli}(\beta) \\ \mathbf{x}|y=0 &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}) \\ \mathbf{x}|y=1 &\sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \end{aligned}$$

- The parameters of the model are $\{\beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}\}$. Note that common covariance matrix is usually used.

# Gaussian Discriminant Analysis (2)

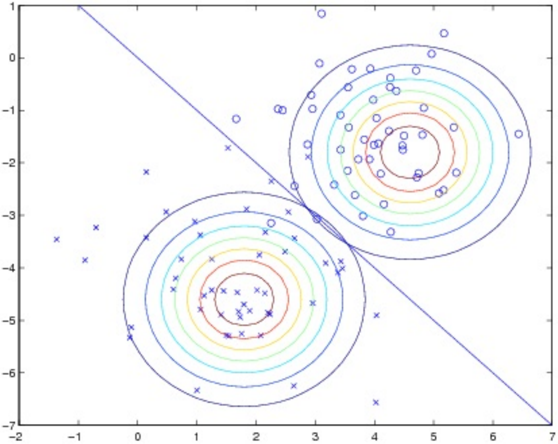
- The log likelihood of the data is:

$$\begin{aligned}\mathcal{L}(\beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) &= \log \prod_{i=1}^{N} p(\mathbf{x}^{(i)}, y^{(i)}; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) \\ &= \log \prod_{i=1}^{N} p(\mathbf{x}^{(i)} | y^{(i)}; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) p(y^{(i)}; \beta)\end{aligned}$$

- The maximum likelihood estimates are:

$$\begin{aligned}\hat{\beta} &= \frac{1}{N} \sum_{i=1}^{N} 1\{y^{(i)} = 1\} \\ \hat{\boldsymbol{\mu}}_k &= \frac{\sum_{i=1}^{N} 1\{y^{(i)} = k\} \mathbf{x}^{(i)}}{\sum_{i=1}^{N} 1\{y^{(i)} = k\}}, \quad k = \{0, 1\} \\ \hat{\boldsymbol{\Sigma}} &= \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{y^{(i)}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}_{y^{(i)}})^T\end{aligned}$$

# Illustration of GDA



# Classification with Discriminant Functions

- Assume class-conditional densities $p(\mathbf{x}|C_i)$ are Gaussian with common covariance $\mathbf{\Sigma}$:

$$p(\mathbf{x}|C_i) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i))$$

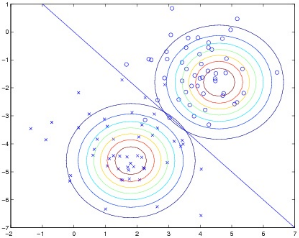
- Discriminant functions:

$$\begin{aligned} g_i(\mathbf{x}) &= \log P(\mathbf{x}|C_i) + \log P(C_i) \\ &= -\frac{D}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{\Sigma}| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) \\ &\quad + \log P(C_i) \\ &= \mathbf{w}_i^T\mathbf{x} + w_{i0} \end{aligned}$$

where

$$\begin{aligned} \mathbf{w}_i &= \mathbf{\Sigma}^{-1}\boldsymbol{\mu}_i \\ w_{i0} &= -\frac{1}{2}\boldsymbol{\mu}_i^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_i + \log P(C_i) \end{aligned}$$

# Linear Discriminant Function



- If class-conditional densities are Gaussian with a common covariance, the discriminant functions are linear.
- Given samples $\mathcal{X}$, we can find ML estimates for $\mu_i, \mathbf{\Sigma}$ and plug them into the discriminant functions.

# GDA and Logistic Regression

- If we view $P(y = 1|\mathbf{x}; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma})$ as a function of $\mathbf{x}$, it can be expressed in the form:

  $$P(y = 1|\mathbf{x}; \beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x})},$$

  where $\mathbf{w}$ is some appropriate function of $\beta, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}$. This is exactly the form that logistic regression (a discriminative algorithm) used to model $P(y = 1|\mathbf{x})$.
- GDA makes stronger modeling assumptions (i.e., $p(\mathbf{x}|y)$ is multivariate Gaussian) about the data, and is more data efficient.
- Logistic regression makes weaker assumptions, and is more robust to deviations from modeling assumptions.

# Email Spam Filter

- Classifying emails to spam ($y = 1$) or non-spam ($y = 0$)
- One example of broader set of problems called text classification
- Feature vector:

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

  - $\mathbf{x} \in \{0, 1\}^D$, D is the size of the set of words (vocabulary).
  - Word selection: include words that appear in the emails but not in a dictionary; exclude very high frequency words; etc.
  - $x_j' s$ can take more general values in $\{1, 2, \ldots, k_j\}$, then the model $p(x_j|y)$ is multinomial rather than binormial.

# Email Spam Filter (2)

- To model $p(\mathbf{x}|y)$, we assume that $x_j's$ are conditionally independent given $y$. This is Naive Bayes assumption.
- In Naive Bayes classifier:

$$p(x_1, \ldots, x_D|y) = p(x_1|y)p(x_2|y) \ldots p(x_D|y) = \prod_{j=1}^{D} p(x_j|y)$$

- Given a training set $\{\mathbf{x}^{(i)}, y^{(i)}\}, i = 1, \ldots, N$, the joint log likelihood of the data:

$$\mathcal{L}(p(x_j|y), P(y)) = \log \prod_{i=1}^{N} p(\mathbf{x}^{(i)}, y^{(i)})$$

# Email Spam Filter (3)

- Maximizing $\mathcal{L}$ w.r.t. $p(x_j|y), P(y)$ gives the following estimates:

$$\begin{aligned}
p(x_j = 1|y = 1) &= \frac{\sum_{i=1}^{N} 1\{x_j^{(i)} = 1 \bigwedge y^{(i)} = 1\}}{\sum_{i=1}^{N} 1\{y^{(i)} = 1\}} \\
p(x_j = 1|y = 0) &= \frac{\sum_{i=1}^{N} 1\{x_j^{(i)} = 1 \bigwedge y^{(i)} = 0\}}{\sum_{i=1}^{N} 1\{y^{(i)} = 0\}} \\
p(y = 1) &= \frac{\sum_{i=1}^{N} 1\{y^{(i)} = 1\}}{N}
\end{aligned}$$

# Email Spam Filter (4)

- To make a prediction on a new example $\mathbf{x}$, we simply calculate:

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

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

# Laplace Smoothing

- If you haven't seen some word before in your finite training set, the Naive Bayes classifier does not know how to make a decision.
- To estimate the mean of multinomial random variable $x$ taking values $\{1, \ldots, k\}$, given $N$ observations $\{x^{(1)}, \ldots, x^{(N)}\}$, the maximum likelihood estimates are:

$$p(x = j) = \frac{\sum_{i=1}^{N} 1\{x^{(i)} = j\}}{N}, \; j = 1, \ldots, k$$

- Laplace smoothing:

$$p(x = j) = \frac{\sum_{i=1}^{N} 1\{x^{(i)} = j\} + 1}{N + k}, \; j = 1, \ldots, k$$

  - $\sum_{j=1}^{k} p(x = j) = 1$ still holds
  - $p(x = j) \neq 0$ for all $j$

# Laplace Smoothing (2)

- Returning to Naive Bayes classifier, with Laplace smoothing we obtain the following estimates:

$$
\begin{aligned}
p(x_j = 1 | y = 1) &= \frac{\sum_{i=1}^{N} 1\{x_j^{(i)} = 1 \bigwedge y^{(i)} = 1\} + 1}{\sum_{i=1}^{N} 1\{y^{(i)} = 1\} + 2} \\
p(x_j = 1 | y = 0) &= \frac{\sum_{i=1}^{N} 1\{x_j^{(i)} = 1 \bigwedge y^{(i)} = 0\} + 1}{\sum_{i=1}^{N} 1\{y^{(i)} = 0\} + 2}
\end{aligned}
$$

- In practice, it usually doesn’t matter whether we apply Laplace smoothing to $p(y = 1)$ or not, since we typically have a fair fraction between spam and non-spam emails.

# Event Models for Text Classification

- Naive Bayes uses multivariate Bernoulli event model: the probability of an email was given by $P(y)\prod_{j=1}^{D} p(x_j|y)$.
- A different way to represent emails: $\mathbf{x} = (x_1, \ldots, x_M)$, $x_j$ denotes the $j^{th}$ word in the email, taking values in $\{1, \ldots, |V|\}$; $V$ is the vocabulary; $M$ is the length of the email.
- Multinomial event model: $p(x_j|y)$ is now multinomial. The overall probability of an email is $P(y)\prod_{j=1}^{M} p(x_j|y)$.

# Event Models for Text Classification (2)

- Given a training set $\{\mathbf{x}^{(i)}, y^{(i)}\}, i = 1, \ldots, N$, where $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_{M_i}^{(i)})^T$ ($M_i$ is the number of words in $i^{th}$ training example), the maximum likelihood estimates of the model are:

$$\begin{aligned}
p(x_j = k|y = 1) &= \frac{\sum_{i=1}^{N} \sum_{j=1}^{M_i} 1\{x_j^{(i)} = k \bigwedge y^{(i)} = 1\}}{\sum_{i=1}^{N} 1\{y^{(i)} = 1\} M_i}, \quad \text{for any } j \\
p(x_j = k|y = 0) &= \frac{\sum_{i=1}^{N} \sum_{j=1}^{M_i} 1\{x_j^{(i)} = k \bigwedge y^{(i)} = 0\}}{\sum_{i=1}^{N} 1\{y^{(i)} = 0\} M_i}, \quad \text{for any } j \\
P(y = 1) &= \frac{\sum_{i=1}^{N} 1\{y^{(i)} = 1\}}{N}
\end{aligned}$$

# Naive Bayes vs. Logistic Regression

- Asymptotic comparison ($\sharp$ training examples $\rightarrow$ infinite)
- When model assumption is correct:
  - Naive Bayes (NB), Logistic regression (LR) produce identical classifier
- When model assumption is incorrect:
  - LR is less biased - does not assume conditional independence
  - therefore expected to outperform NB.

# Naive Bayes vs. Logistic Regression (2)

- Non-asymptotic comparison (see [Ng and Jordan, 2002])
- Convergence rate of parameter estimation - how many training examples needed to achieve good estimates?
  - NB order $\log D$ (where $D = \sharp$ of attributes in $\mathbf{X}$)
  - LR order $D$
- NB converges more quickly to its asymptotic estimates.

# K-fold Cross Validation

- The data set $\mathcal{X}$ is randomly partitioned into $K$ equal-sized subsets $\mathcal{X}_i, i = 1, \ldots, K$, called folds.
- Stratification: the class distributions in different subsets are kept roughly the same (stratified cross validation).
- $K$ training/validation set pairs $\{\mathcal{T}_i, \mathcal{V}_i\}_{i=1}^{K}$:

$$\begin{aligned}
\mathcal{T}_1 &= \mathcal{X}_2 \cup \mathcal{X}_3 \cup \ldots \cup \mathcal{X}_K, \quad \mathcal{V}_1 = \mathcal{X}_1 \\
\mathcal{T}_2 &= \mathcal{X}_1 \cup \mathcal{X}_3 \cup \ldots \cup \mathcal{X}_K, \quad \mathcal{V}_2 = \mathcal{X}_2 \\
&\vdots \\
\mathcal{T}_K &= \mathcal{X}_1 \cup \mathcal{X}_2 \cup \ldots \cup \mathcal{X}_{K-1}, \quad \mathcal{V}_K = \mathcal{X}_K
\end{aligned}$$

# Leave-One-Out

- If $N$ is small, $K$ should be large to allow large enough training sets.
- One extreme case of $K$-fold cross validation is leave-one-out where only one instance is left out as the validation set and the remaining $N - 1$ for training, giving $N$ separate training/validation set pairs.

# Bootstrapping

- When the sample size $N$ is very small, a better alternative to cross validation is bootstrapping.
- Bootstrapping generates new samples, each of size $N$, by drawing instances randomly from the original sample with replacement.
- The bootstrap samples usually overlap more than the cross validation samples and hence their estimates are more dependent.
- Probability that an instance is not chosen after $N$ random draws:

$$(1-\frac{1}{N})^N \approx e^{-1} = 0.368$$

  So each bootstrap sample contains only approximately 63.2% of the instances.
- Multiple bootstrap samples are used to maximize the chance that the system is trained on all the instances.

# Error Measure

- If $0-1$ loss is used, error calculations will be based on the confusion matrix:

|  | Predicted Class |  |
|---|---|---|
| True Class | Yes | No |
| Yes<br>No | TP: true positive<br>FP: false postivie | FN: false negative<br>TN: true negative |

- Error rate:

$$\text{Error rate} = \frac{|FN|+|FP|}{|TP|+|TP|+|TN|+|FN|} = \frac{|FN|+|FP|}{N}$$

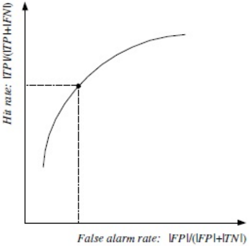
  where $N$ is the total number of instances in the validation set.

- For more general loss functions, the risks should be measured instead.
- For $K>2$ classes, the class confusion matrix is a $K \times K$ matrix such that its $(i,j)$-th entry contains the number of instances that belong to $C_i$ but are assigned to $C_j$.

# ROC Curve

- Receiver operating characteristics (ROC) curve:

$$\text{Hit rate } \frac{|TP|}{|TP|+|FN|} \text{ vs. False alarm rate } \frac{|FP|}{|FP|+|TN|}$$



- The curve is obtained by varying a certain parameter (e.g., threshold for making decision) of the classification algorithm.
- The area under curve (AUC) is often used as a performance measure.

# Point Estimation vs. Interval Estimation

- A point estimator (e.g., MLE) specifies a value for a parameter $\theta$.
- An interval estimator specifies an interval within which $\theta$ lies with a certain degree of confidence.
- The probability distribution of the point estimator is used to obtain an interval estimator.

# Example: Estimation of Mean of Normal Distribution

- Given an i.i.d. sample $\mathcal{X} = \{x^{(i)}\}_{i=1}^{N}$, where

$$x^{(i)} \sim \mathcal{N}(\mu, \sigma^2)$$

- Point estimation of the mean $\mu$:

$$m = \frac{\sum_i x^{(i)}}{N} \sim \mathcal{N}(\mu, \sigma^2/N)$$

  since $x^{(i)}$ are i.i.d.

- Define a statistic $\mathcal{Z}$ with a unit normal distribution:

$$\mathcal{Z} = \frac{\sqrt{N}(m - \mu)}{\sigma} \sim \mathcal{N}(0, 1)$$

# Two-Sided Confidence Interval

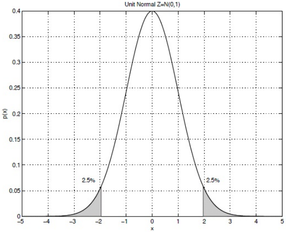
- For a unit normal distribution, about 95% of $\mathcal{Z}$ lies in $(-1.96, 1.96)$:

$$P(-1.96 < \frac{\sqrt{N}(m-\mu)}{\sigma} < 1.96) = 0.95$$

or

$$P(m - 1.96\frac{\sigma}{\sqrt{N}} < \mu < m + 1.96\frac{\sigma}{\sqrt{N}}) = 0.95$$

meaning that with 95% confidence, $\mu$ lies within $1.96\sigma/\sqrt{N}$ units of the sample mean $m$.



# Two-Sided Confidence Interval (2)

- Let us denote $z_\alpha$ such that

$$P(\mathcal{Z} > z_\alpha) = \alpha, \quad 0 < \alpha < 1$$

- Because $\mathcal{Z}$ is symmetric around the mean, we have

$$P(\mathcal{Z} < -z_{\alpha/2}) = P(\mathcal{Z} > z_{\alpha/2}) = \alpha/2$$

- Hence, for any specified level of confidence $1 - \alpha$, we have

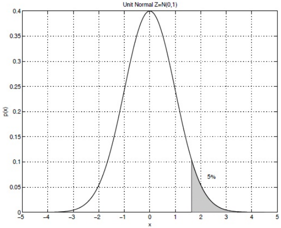
$$P(-z_{\alpha/2} < \frac{\sqrt{N}(m - \mu)}{\sigma} < z_{\alpha/2}) = 1 - \alpha$$

or

$$P(m - z_{\alpha/2}\frac{\sigma}{\sqrt{N}} < \mu < m + z_{\alpha/2}\frac{\sigma}{\sqrt{N}}) = 1 - \alpha$$

# One-Sided Upper Confidence Interval

- $P(\mathcal{Z} < 1.64) = 0.95$



- 95% one-sided upper confidence interval for $\mu$:

$$P(\frac{\sqrt{N}(m-\mu)}{\sigma} < 1.64) = 0.95$$

or

$$P(m - 1.64\frac{\sigma}{\sqrt{N}} < \mu) = 0.95$$

# One-Sided Upper Confidence Interval (2)

- The one-sided upper confidence interval defines a lower bound for $\mu$.
- In general, a $100(1-\alpha)$ percent one-sided upper confidence interval for $\mu$ can be computed from

$$P(m - z_\alpha \frac{\sigma}{\sqrt{N}} < \mu) = 1 - \alpha$$

- The one-sided lower confidence interval that defines an upper bound can be calculated similarly.

# $t$ Distribution

- When the variance $\sigma^2$ is not known, it can be replaced by the sample variance

$$S^2 = \frac{\sum_i (x^{(i)} - m)^2}{N-1}$$

- The statistic $\sqrt{N}(m-\mu)/S$ follows a $t$ distribution with $N-1$ degrees of freedom:

$$\sqrt{N}(m-\mu)/S \sim t_{N-1}$$

- For any $\alpha \in (0, 1/2)$,

$$P(-t_{\alpha/2,N-1} < \frac{\sqrt{N}(m-\mu)}{S} < t_{\alpha/2,N-1}) = 1-\alpha$$

or

$$P(m - t_{\alpha/2,N-1}\frac{S}{\sqrt{N}} < \mu < m + t_{\alpha/2,N-1}\frac{S}{\sqrt{N}}) = 1-\alpha$$

# $t$ Distribution (2)

- One-sided confidence intervals can also be defined.
- The $t$ distribution has larger spread (longer tails) than the unit normal distribution, and generally the interval given by the $t$ distribution is larger due to additional uncertainty about the unknown variance.
- As $N$ increases, the $t$ distribution will get closer to the normal distribution.

# Hypothesis Testing

- Instead of explicitly estimating some parameters, hypothesis testing uses the sample to test some hypothesis concerning the parameters, e.g., whether the mean is $< 0.02$.
- Hypothesis testing:
  - We define a statistic that obeys a certain distribution if the hypothesis is correct.
  - If the random sample is consistent with the hypothesis under consideration (i.e., if the statistic calculated from the sample has a high enough probability of being drawn from the probability), the hypothesis is accepted.
  - Otherwise the hypothesis is rejected.
- There are parametric and nonparametric tests. Only parametric tests are considered here.

# Null Hypothesis

- Given a sample from a normal distribution $\mathcal{N}(\mu, \sigma^2)$ where $\sigma$ is known but $\mu$ is unknown.
- We want to test a specific hypothesis about $\mu$, called the null hypothesis, e.g.,

$$H_0 : \mu = \mu_0$$

against the alternative hypothesis

$$H_1 : \mu \neq \mu_0$$

for some specified constant $\mu_0$

# Two-Sided Test

- We accept $H_0$ if the sample mean $m$, a point estimate of $\mu$, is not too far from $\mu_0$.
- We accept $H_0$ with level of significance $\alpha$ if $\mu_0$ lies in the $100(1-\alpha)$ percent confidence interval, i.e.,

$$\frac{\sqrt{N}(m-\mu_0)}{\sigma} \in (-z_{\alpha/2}, z_{\alpha/2})$$

# One-Sided Test

- Null and alternative hypotheses:

$$\begin{aligned} H_0 &: \mu \leq \mu_0 \\ H_1 &: \mu > \mu_0 \end{aligned}$$

- We accept $H_0$ with level of significance $\alpha$ if

$$\frac{\sqrt{N}(m - \mu_0)}{\sigma} \in (-\infty, z_\alpha)$$

# $t$ Tests

- If the variance $\sigma^2$ is not known, the sample variance $S^2$ will be used instead and so

$$\frac{\sqrt{N}(m-\mu_0)}{S} \sim t_{N-1}$$

- Two-sided $t$ test:

$$\begin{aligned} H_0 &: \quad \mu = \mu_0 \\ H_1 &: \quad \mu \neq \mu_0 \end{aligned}$$

  We accept at significance level $\alpha$ if

$$\frac{\sqrt{N}(m-\mu_0)}{S} \in (-t_{\alpha/2,N-1}, t_{\alpha/2,N-1})$$

- One-side $t$ test can be defined similarly.

# Binomial Test

- Single training/validation set pair: a classifier is trained on a training set $\mathcal{T}$ and tested on a validation set $\mathcal{V}$.
- Let $p$ be the (unknown) probability that the classifier makes a misclassification error.
- For the $i$th instance in $\mathcal{V}$, we define $x^{(i)}$ as a Bernoulli variable to denote the correctness of the classifier's decision:

$$x^{(i)} = \begin{cases} 1 & \text{with probability } p \\ 0 & \text{with probability } 1 - p \end{cases}$$

- Point estimate of $p$:

$$\hat{p} = \frac{\sum_i x^{(i)}}{|\mathcal{V}|} = \frac{\sum_i x^{(i)}}{N}$$

# Binomial Test (2)

- Hypothesis test:

$$H_0 : p \leq p_0 \text{ vs. } H_1 : p > p_0$$

Given that the classifier makes $e$ errors on $\mathcal{V}$ with $|\mathcal{V}| = N$, can we say that the classifier has error probability $p_0$ or less?

- Let $X = \sum_{i=1}^{N} x^{(i)}$ denote the number of errors on $\mathcal{V}$.
- Because $x^{(i)}$ are independent Bernoulli variables, $X$ is binomial:

$$P(X = j) = \binom{N}{j} p^j (1-p)^{N-j}$$

- Under the null hypothesis $p \leq p_0$, so the probability that there are $e$ errors or less is

$$P(X \leq e) = \sum_{j=1}^{e} \binom{N}{j} p_0^j (1-p_0)^{N-j}$$

- Binomial test: accept $H_0$ if $P(X \leq e) < 1 - \alpha$; reject otherwise.

# Paired *t* Test

- Multiple training/validation set pairs: if we run the algorithm $K$ times on $K$ training/validation set pairs, we get $K$ error probabilities, $p_k, k = 1, \ldots, K$, on the $K$ validation sets.
- We can use the paired *t* test to determine whether to accept the null hypothesis $H_0$ that the classifier has error probability $p_0$ or less at significance level $\alpha$.
- Bernoulli variables:

$$x_k^{(i)} = \begin{cases} 1 & \text{if classifier trained on } \mathcal{T}_k \text{ makes an error on instance } i \text{ of } \mathcal{V}_k \\ 0 & \text{otherwise} \end{cases}$$

- Test statistic:

$$\frac{\sqrt{K}(m - p_0)}{S} \sim t_{K-1}$$

where

$$p_i = \frac{\sum_{i=1}^{N} x^{(i)}}{N}, m = \frac{\sum_{k=1}^{K} p_k}{K}, S^2 = \frac{\sum_{k=1}^{K} (p_k - m)^2}{K - 1}$$

# $K$-Fold Cross-Validated Paired $t$ Test

- Given two classification algorithms and a data set, we want to compare and test whether the two algorithms construct classifiers that have the same expected error rate on a new instance.
- $K$-fold cross validation is used to obtain $K$ training/validation set pairs, $\{(\mathcal{T}_k, \mathcal{V}_k)\}_{k=1}^K$.
- For each training/validation set pair $(\mathcal{T}_k, \mathcal{V}_k)$, the two classification algorithms are trained on $\mathcal{T}_k$ and tested on $\mathcal{V}_k$, with error probabilities $p_k^1$ and $p_k^2$.
- We define
  $$p_k = p_k^1 - p_k^2, k = 1, \ldots, K$$
  The distribution of $p_k$ is used for hypothesis testing.
- if $p_k^1$ and $p_k^2$ are both normally distributed, then $p_k$ should also be normal (with mean $\mu$).

# *K*-Fold Cross-Validated Paired *t* Test (2)

- Null and alternative hypotheses:

$$\begin{aligned} H_0 &: \mu = 0 \\ H_1 &: \mu \neq 0 \end{aligned}$$

- Test statistic:

$$\frac{\sqrt{K}(m-0)}{S} = \frac{\sqrt{K}m}{S} \sim t_{K-1}$$

where

$$m = \frac{\sum_{k=1}^{K} p_k}{K}, S^2 = \frac{\sum_{k=1}^{K}(p_k - m)^2}{K-1}$$

- *K*-fold CV paired *t* test: accept $H_0$ at significance level $\alpha$ if $\sqrt{K}m/S \in (-t_{\alpha/2,K-1}, t_{\alpha/2,K-1})$; reject otherwise.