Machine Learning
CS690

Lecture 06

Razvan C. Bunescu
School of Electrical Engineering and Computer Science
bunescu@ohio.edu
Three Parametric Approaches to Classification

1) Discriminant Functions: construct $f: X \rightarrow T$ that directly assigns a vector $x$ to a specific class $C_k$.
   - Inference and decision combined into a single learning problem.
   - *Linear Discriminant*: the decision surface is a hyperplane in $X$:
     - Fisher’s Linear Discriminant
     - Perceptron
     - Support Vector Machines
Three Parametric Approaches to Classification

2) **Probabilistic Discriminative Models**: directly model the posterior class probabilities $p(C_k | x)$.
   - Inference and decision are separate.
   - Less data needed to estimate $p(C_k | x)$ than $p(x | C_k)$.
   - Can accommodate many overlapping features.
     - Logistic Regression
     - Conditional Random Fields
Three Parametric Approaches to Classification

3) **Probabilistic Generative Models:**
   - Model class-conditional $p(x \mid C_k)$ as well as the priors $p(C_k)$, then use Bayes’s theorem to find $p(C_k \mid x)$.
     - or model $p(x, C_k)$ directly, then marginalize to obtain the posterior probabilities $p(C_k \mid x)$.
   - Inference and decision are separate.
   - Can use $p(x)$ for **outlier** or **novelty detection**.
   - Need to model dependencies between features.
     - Naïve Bayes.
     - Hidden Markov Models.
Probabilistic Generative Models: Binary Classification ($K = 2$)

- Model class-conditional $p(x \mid C_1)$, $p(x \mid C_2)$ as well as the priors $p(C_1)$, $p(C_2)$, then use Bayes’s theorem to find $p(C_1 \mid x)$, $p(C_2 \mid x)$:

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

$$= \sigma(a(x))$$

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

$$a(x) = \ln \frac{p(x \mid C_1)p(C_1)}{p(x \mid C_2)p(C_2)} = \ln \frac{p(C_1 \mid x)}{p(C_2 \mid x)}$$
Probabilistic Generative Models: Binary Classification ($K = 2$)

- If $a(x)$ is a linear function of $x \Rightarrow p(C_1 \mid x)$ is a generalized linear model:

$$p(C_1 \mid x) = \frac{1}{1 + \exp(-a(x))} = \sigma(a(x)) = \sigma(\lambda^T x)$$

$\sigma(a)$ is a squashing function
Probabilistic Generative Models: Multiple Classes ($K \geq 2$)

- Model class-conditional $p(x \mid C_k)$ as well as the priors $p(C_k)$, then use Bayes’s theorem to find $p(C_k \mid x)$:

$$
p(C_k \mid x) = \frac{p(x \mid C_k)p(C_k)}{\sum_j p(x \mid C_j)p(C_j)}
= \frac{\exp(a_k(x))}{\sum_j \exp(a_j(x))}
$$

where $a_k(x) = \ln p(x \mid C_k)p(C_k)$

- If $a_k(x) = \lambda_k^T x \Rightarrow p(C_k \mid x)$ is a generalized linear model.
Unbiased Learning of Generative Models

• Let \( \mathbf{x} = [x_1, x_2, \ldots, x_M]^T \) be a feature vector with \( M \) features.

• Assume Boolean features:
  \[ \Rightarrow \text{distribution } p(\mathbf{x} | C_k) \text{ is completely specified by a table of } 2^M \text{ probabilities, of which } 2^M - 1 \text{ are independent.} \]

• Assume binary classification:
  \[ \Rightarrow \text{need to estimate } 2^M - 1 \text{ parameters for each class} \]
  \[ \Rightarrow \text{total of } 2(2^M - 1) \text{ independent parameters to estimate.} \]
  – 30 features \( \Rightarrow \) more than 3 billion parameters to estimate!
The Naïve Bayes Model

• Assume features are conditionally independent given the target output:

\[ p(x | C_k) = \prod_{i=1}^{M} p(x_i | C_k) \]

• Assume binary classification & features:
  ⇒ need to estimate only \( 2M \) parameters, a lot less than \( 2(2^M - 1) \).
The Naïve Bayes Model

- Assume binary features $x_i \in \{0, 1\}$:
  \[
  p(x \mid C_k) = \prod_{i=1}^{M} p(x_i \mid C_k)
  \]
  \[
  = \prod_{i=1}^{M} \mu_{ki}^{x_i} (1 - \mu_{ki})^{1-x_i}, \text{ where } \mu_{ki} = p(x_i = 1 \mid C_k)
  \]
  \[
  \Rightarrow p(C_k \mid x) = \frac{\exp(a_k(x))}{\sum_j \exp(a_j(x))}
  \]
  , where $a_k(x) = \sum_{i=1}^{M} \{x_i \ln \mu_{ki} + (1-x_i) \ln(1-\mu_{ki})\} + p(C_k)
  \]
  \[
  = \lambda_k^T x \quad \Rightarrow \text{NB is a generalized linear model.}
  \]
The Naïve Bayes Model: Inference

• Posterior distribution:
  \[
  p(C_k \mid x) = \frac{p(x \mid C_k)p(C_k)}{p(x)}, \quad \text{where } p(x) = \sum_j p(x \mid C_j)p(C_j)
  \]
  \[
  = \frac{p(C_k)\prod_j p(x_j \mid C_k)}{p(x)}
  \]

• Inference \(\equiv\) find \(C_*\) to minimize misclassification rate:
  \[
  C_* = \arg \max_{C_k} p(C_k \mid x)
  \]
  \[
  = \arg \max_{C_k} p(C_k)\prod_j p(x_j \mid C_k)
  \]
The Naïve Bayes Model: Training

• Training ≡ estimate parameters \( p(x_i|C_k) \) and \( p(C_k) \).

• Maximum Likelihood (ML) estimation:

\[
\hat{p}(x_i = v \mid t = C_k) = \frac{\sum \delta_{v}(x_i) \delta_{C_k}(t)}{\sum \delta_{C_k}(t)}
\]

\[
\hat{p}(t = C_k) = \frac{\sum \delta_{C_k}(t)}{|D|}
\]

# training examples in which \( x_i = v \) and \( t = C_k \)

# training examples in which \( t = C_k \)
The Naïve Bayes Model: Training

- **Maximum A-Posteriori (MAP) estimation:**
  - assume a Dirichlet prior over the NB parameters, with equal-valued parameters.
  - assume $x_i$ can take $V$ values, label $t$ can take $K$ values.

$$\hat{p}(x_i = v \mid t = C_k) = \frac{\sum_{(x,t) \in D} \delta_v(x_i) \delta_{C_k}(t) + l}{\sum_{(x,t) \in D} \delta_{C_k}(t) + lV}$$

$$\hat{p}(t = C_k) = \frac{\sum_{(x,t) \in D} \delta_{C_k}(t) + l}{|D| + lK}$$

$\Leftrightarrow lV$ “hallucinated” examples spread evenly over all $V$ values of $x_i$.

$l = 1 \Rightarrow Laplace$ smoothing
Text Categorization with Naïve Bayes

• Text categorization problems:
  – Spam filtering.
  – Targeted advertisement in Gmail.
  – Classification in Yahoo! Directory or ODP.

• Representation as one feature per word:
  ⇒ each document is a very high dimensional feature vector.

• Most words are rare:
  – Zipf’s law and heavy tail distribution.
  ⇒ feature vectors are sparse.
Text Categorization with Naïve Bayes

• Generative model of documents:
  1) Generate document category by sampling from $p(C_k)$.
  2) Generate a document as a bag of words by repeatedly sampling with replacement from a vocabulary $V = \{w_1, w_2, \ldots, w_{|V|}\}$ based on $p(w_i \mid C_k)$.

• Inference with Naïve Bayes:
  – Input:
    • Document $x$ with $n$ words $v_1, v_2, \ldots, v_n$.
  – Output:
    • Category $C_* = \arg \max_{C_k} p(C_k) \prod_{j=1}^{n} p(v_j \mid C_k)$
Text Categorization with Naïve Bayes

- **Training** with Naïve Bayes:
  - Input:
    - Dataset of training documents $D$ with vocabulary $V$.
  - Output:
    - Parameters $p(C_k)$ and $p(w_i | C_k)$.

1. **for each** category $C_k$:
2. let $D_k$ be the subset of documents in category $C_k$
3. set $p(C_k) = |D_k| / |D|$
4. let $n_k$ be the total number of words in $D_k$
5. **for** each word $w_i \in V$:
6. let $n_{ki}$ be the number of occurrences of $w_i$ in $D_k$
7. set $p(w_i | C_k) = (n_{ki} + 1) / (n_k + |V|)$
Medical Diagnosis with Naïve Bayes

- Diagnose a disease $T=\{Yes, No\}$, using information from various medical tests.

\[ p(\mathbf{x} \mid C_k) = \prod_{i=1}^{M} p(x_i \mid C_k) \]

Medical tests may result in continuous values
⇒ need Naïve Bayes to work with continuous features.
Naïve Bayes with Continuous Features

• Assume $p(x_i \mid C_k)$ are Gaussian distributions $N(\mu_{ik}, \sigma_{ik})$.

• Training: use ML or MAP criteria to estimate $\mu_{ik}, \sigma_{ik}$:

$$\hat{\mu}_{ik} = \frac{\sum_{(x, t) \in D} x_i \delta_{C_k}(t)}{\sum_{(x, t) \in D} \delta_{C_k}(t)}$$

$$\hat{\sigma}_{ik}^2 = \frac{\sum_{(x, t) \in D} (x_i - \hat{\mu}_{ik})^2 \delta_{C_k}(t)}{\sum_{(x, t) \in D} \delta_{C_k}(t)}$$

• Inference:

$$C_* = \arg \max_{C_k} p(C_k \mid x) = \arg \max_{C_k} p(C_k) \prod_i p(x_i \mid C_k)$$
Numerical Issues

• Multiplying lots of probabilities may result in underflow:
  – especially when many attributes (e.g. text categorization).

• Compute everything in log space:

\[
p(x \mid C_k) = \prod_{i=1}^{M} p(x_i \mid C_k) \Leftrightarrow \ln p(x \mid C_k) = \sum_{i=1}^{M} \ln p(x_i \mid C_k)
\]

\[
C_* = \arg \max_{C_k} p(C_k \mid x) \Leftrightarrow C_* = \arg \max_{C_k} \ln p(C_k \mid x)
\]

\[
= \arg \max_{C_k} \left\{ \ln p(C_k) + \ln p(x \mid C_k) \right\}
\]
Naïve Bayes

• Often has good performance, despite strong independence assumptions:
  – quite competitive with other classification methods on UCI datasets.

• It does not produce accurate probability estimates when independence assumptions are violated:
  – the estimates are still useful for finding max-probability class.

• Does not focus on completely fitting the data ⇒ resilient to noise.