Intelligent Systems: Reasoning and Recognition

James L. Crowley

ENSIMAG 2 / MoSIG M1 Second Semester 2015/2016

Lesson 8 4 March 2016

Non-Parameteric Bayesian Recognition

Bibliographical sources:
"Pattern Recognition and Machine Learning", C. M. Bishop, Springer Verlag, 2006.
"Pattern Recognition and Scene Analysis", R. E. Duda and P. E. Hart, Wiley, 1973.

Notation

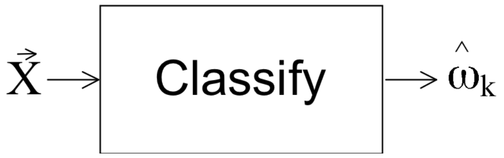
x	a variable
X	a random variable (unpredictable value)
M	The number of possible values for X (Can be infinite).
$\vec{x}$	A vector of D variables.
$\vec{X}$	A vector of D random variables.
D	The number of dimensions for the vector $\vec{x}$ or $\vec{X}$
E	An observation. An event.
C_k	The class k
k	Class index
K	Total number of classes
ω_k	The statement (assertion) that E $\in C_k$
$P(\omega_k) = P(E \in C_k)$	Probability that the observation E is a member of the class k.
M_k	Number of examples for the class k.
M	Total number of examples. $M = \sum_{k=1}^{K} M_k$
$\{\vec{X}_m\}$	A set of training samples
$\{\vec{y}_m\}$	A set of indicator vectors for the training samples in $\{\vec{X}_m\}$ $\vec{y}_n$ is a vector of binary values, with 1 for the kth component and 0 elsewhere.
$p(X)$	Probability density function for X
$p(\vec{X})$	Probability density function for $\vec{X}$
$p(\vec{X} / \omega_k)$	Probability density for $\vec{X}$ the class k. ω_k = E $\in C_k$.
Q	Number of cells in h(x). $Q = N^D$
P	A sum of V adjacent histogram cells: $P = \sum_{\vec{X} \in V} h(\vec{X})$

Note that in this lecture N is the number of values for an integer feature X.
V will be used for "Volume".

Bayesian Classification

In the next few lectures we will look at techniques that learn to recognize classes using Bayes rule. These methods are based on an explicit probabilistic model of the training data. We will start with some simple, non-parametric models. We will then look at models using the Gaussian or Normal density functions.

In either case, our problem is to build a box that maps a set of features $\vec{X}$ from an Observation, E into a class C_k from a set of K possible classes.



Let ω_k be the proposition that the event belongs to class k: $\omega_k = E \in C_k$

ω_k Proposition that event E $\in$ the class k

In order to minimize the number of mistakes, we will maximize the probability that $\omega_k \equiv E \in C_k$

$$\hat{\omega}_k = \arg-\max_{\omega_k}\left\{P(\omega_k \mid \vec{X})\right\}$$

Our primary tool for this is Baye's Rule : $P(\omega_k \mid \vec{X}) = \frac{P(\vec{X} \mid \omega_k)P(\omega_k)}{P(\vec{X})}$

To apply Baye's rule, we require a representation for the probabilities $P(\vec{X} \mid \omega_k)$, $P(\vec{X})$, and $p(\omega_k)$. The term $p(\omega_k)$ is a number that represents the a-priori probability of encountering an event of class K. For a training set of M samples of which M_k are from class k, this is simply the frequency of occurrence of class k.

$$P(\omega_k) = \frac{M_k}{M}$$

The terms $P(\vec{X} \mid \omega_k)$, $P(\vec{X})$ are more subtle.
We have already seen how to use histograms to represent $P(\vec{X} \mid \omega_k)$ and $P(\vec{X})$
Today will look at three non-parametric representations for $P(\vec{X} \mid \omega_k)$ and $P(\vec{X})$

1) Ratio of Histograms
2) Kernel Density Estimators
3) K-Nearest Neighbors

Baye's Rule as a Ratio of Histograms

We can use a histogram representation to illustrate a basic principle. This also provides a useful but easy to implement form of learning.

Consider an example of K classes of objects where objects are described by a feature, *X*, with N possible values.
Assume that for each of the K classes, we have a "training set" of M_k samples $\{X_m^k\}$.

For each class k, we allocate a histogram, *$h_k()$*, with *N* cells and count the values in the training set.

$$\forall_{k=1}^{K} \forall_{m=1}^{M_k} : h_k(X_m^k) \leftarrow h_k(X_m^k) + 1$$

Then the probability of observing a value X in the training set is

$$p(X = x \mid E \in C_k) = p(X \mid \omega_k) = \frac{1}{M_k} h_k(x)$$

The combined probability for all classes is just the sum of the histograms.

$$h(X) = \sum_{k=1}^{K} h_k(X) \text{ and } M = \sum_{k=1}^{K} M_k$$

Thus

$$p(X = x) = \frac{1}{M} h(x)$$

$P(\omega_k)$ can be estimated from the relative size of the training set.

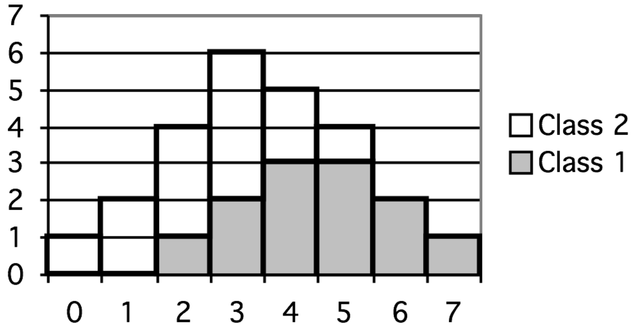
$$p(E \in C_k) = p(\omega_k) = \frac{M_k}{M}$$

the probability that an observation E, with feature X belongs to class K is

$$p(\omega_k \mid X) = \frac{p(X \mid \omega_k) p(\omega_k)}{p(X)} = \frac{\frac{1}{M_k} h_k(X) \frac{M_k}{M}}{\frac{1}{M} h(X)} = \frac{h_k(X)}{h(X)} = \frac{h_k(X)}{\sum_{k=1}^{K} h_k(X)}$$

Example: K=2 classes, N=8 Values

To illustrate, consider an example with 2 classes (K=2) and where X can take on 8 values (N=8, D=1).



If we observe an unknown object E with property X=2, then $p(\omega_1| X=2) = ¼$

Note: Using Histograms requires two assumptions.

1) that the training set is large enough (M > 8 Q, where $Q=M^D$), and
2) That the observing conditions do not change with time (stationary),

We also assumed that the feature values were natural numbers in the range [1, N]. this can be easily obtained from any features.

Variable Sized Histogram Cells

Suppose that we have a D-dimensional feature vector $\vec{X}$ with each feature quantized to N possible values, and suppose that we represent $p(\vec{X})$ as a D-dimensional histogram $h(\vec{X})$. Let us fill the histogram with M training samples $\{\vec{X}_m\}$.

Let us define the volume of each cell as 1.
The volume for any block of V cells is V.
Then the volume of the entire space is $Q=N^D$.

If the quantity of training data is too small, ie if M < 8Q, then we can combine adjacent cells so as to amass enough data for a reasonable estimate.

Suppose we merge V adjacent cells such that we obtain a combined sum of S.

$$S = \sum_{\vec{X}\in V} h(\vec{X})$$

The volume of the combined cells would be V.
To compute the probability we replace $h(\vec{X})$ with $\frac{S}{V}$.
The probability $p(\vec{X})$ for $\vec{X} \in V$ is:

$$p(\vec{X} \in V) = \frac{1}{M} \cdot \frac{S}{V}$$

This is typically written as: $p(\vec{X}) = \frac{S}{MV}$

We can use this equation to develop two alternative non-parametric methods.

Fix V and determine S => Kernel density estimator.
Fix S and determine V => K nearest neighbors.

(note that the symbol “K” is often used for the sum the cells.
This conflicts with the use of K for the number of classes.
Thus we will use the symbol S for the sum of adjacent cells).

Kernel Density Estimators

For a Kernel density estimator, we represent each training sample with a kernel function $k(\vec{X})$.

Popular Kernel functions include

- a hypercube centered of side w
- a triangular function with base of w
- a sphere of radius w
- a Gaussian of standard deviation w.

We can define the function for the hypercube as

$$k(\vec{u}) = \begin{cases} 1 & if \ |\mathrm{u_d}| \leq 1/2 \ \text{ for all d} = 1,...,\mathrm{D} \\ 0 & \qquad otherwise \end{cases}$$

This is called a Parzen window.
Subtracting a point, $\vec{z}$, centers the parzen window at that point.
Dividing by w scales the parzen window to a hyper-cube of side w.

$k\left(\frac{\vec{X}-\vec{z}}{w}\right)$ is a cube of size $\mathrm{w^D}$ centered at $\vec{z}$.

The M training samples define M overlapping Parzen windows.
For an feature value, $\vec{X}$, the probability $p(\vec{X})$ is the sum of Parzen windows at $\vec{X}$

$$S = \sum_{m=1}^{M} k\left(\frac{\vec{X}-\vec{X}_m}{w}\right)$$

The volume of the parzen window is $V = \frac{1}{w^D}$.

Thus the probability $p(\vec{X}) = \frac{S}{MV} = \frac{1}{Mw^D}\sum_{m=1}^{M} k\left(\frac{\vec{X}-\vec{X}_m}{w}\right)$

The Parzen window are discontinuous at the boundaries, creating boundary effects.
We can soften this using a triangular function evaluated within the window.

$$k(\vec{u}) = \begin{cases} 1 - 2\|\vec{u}\| & if \ \|\vec{u}\| \le 1/2 \\ 0 & otherwise \end{cases}$$

Even better is to use a Gaussian kernel with standard deviation $\sigma = w$.

$$k(\vec{u}) = e^{-\frac{1}{2}\frac{\|\vec{u}\|^2}{w^2}}$$

We can note that the volume is $V = (2\pi)^{D/2} w^D$

In this case $p(\vec{X}) = \frac{S}{MV} = \frac{1}{M}\frac{1}{(2\pi)^{D/2} w^D}\sum_{m=1}^{M} k\left(\vec{X} - \vec{X}_m\right)$

This corresponds to placing a Gaussian at each training sample.
The probability for a value $\vec{X}$ is the sum of the Gaussians.

In fact, we can choose any function $k(\vec{u})$ as kernel, provided that

$$k(\vec{u}) \ge 0 \quad \text{and} \quad \int k(\vec{u}) d\vec{u} = 1$$

The Gaussian Kernel tends to be popular for Machine Learning.
The "Radial Basis Function" is a simple form spherical Gaussian function with fixed sigma.

K Nearest Neighbors

For K nearest neighbors, we hold S constant and vary V. (We have used the symbol S for the number of neighbors, rather than K to avoid confusion with the number of classes).

For each training sample, $\vec{X}_m$, we construct a tree structure (such as a KD Tree) that allows us to easily find the S nearest neighbors for any point.

To compute $p(\vec{X})$ we need the volume of a sphere of radius $||\vec{X}-\vec{X}_S||$ in D dimensions. This is:

$$V = C_D \left\|\vec{X} - \vec{X}_S\right\|^D \qquad \text{where} \qquad C_D = +\frac{\pi^{\frac{D}{2}}}{\Gamma\left(\frac{D}{2}+1\right)}$$

Where $\Gamma(D) = (D-1)!$

For even D this is easy to evaluate

For odd D, use a table to determine $\Gamma\left(\frac{D}{2}+1\right)$

Then as before: $p(\vec{X}) = \frac{S}{MV}$