Pattern Recognition and Machine Learning

James L. Crowley

ENSIMAG 3 - MMIS
Lesson 1

Fall Semester 2017
4 October 2017

Learning and Evaluation for Pattern Recognition

Outline

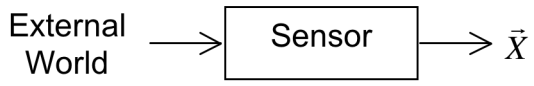
Notation

x_d	A feature. An observed or measured value.
$\vec{X}$	A vector of D features.
D	The number of dimensions for the vector $\vec{X}$
K	Number of classes
C_k	The kth class
$\vec{X} \in C_k$	Statement that an observation $\vec{X}$ is a member of class C_k
$\hat{C}_k$	The estimated class
$R(\vec{X})$	A Recognition function
$\hat{C}_k = R(\vec{X})$	A recognition function that predicts $\hat{C}_k$ from $\vec{X}$
	For a detection function (K=2), $C_k \in \{P, N\}$
y	The true class for an observation $\vec{X}$
$\{\vec{X}_m\}$ $\{y_m\}$	Training samples of $\vec{X}$ for learning, along with ground truth $\vec{y}$
y_m	An annotation (or ground truth) for sample *m*
M	The number of training samples.

1. The Pattern Recognition Problem

Pattern Recognition is the process of assigning observations to categories.
An observation is sometimes called an "entity" or "data" depending on the context and domain.

Observations are produced by some form of sensor.
A sensor is a transducer that transforms physical phenomena into digital measurements. These measurements are classically called "Features".



Features may be Boolean, natural numbers, integers, real numbers or symbolic labels.

In most interesting problems, the sensor provides a feature vector, $\vec{X}$, composed of *D* properties or features $\vec{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix}$

Our problem is to build a function, called a recognizer or classifier, $\mathrm{R}(\vec{X})$, that maps the observation, $\vec{X}$ into a statement that the observation belongs to a class $\hat{C}_k$ from a set of K possible classes. $R(\vec{X}) \rightarrow \hat{C}_k$

$$\vec{X} \rightarrow \boxed{R(\vec{X})} \rightarrow \hat{C}_k$$

In most classic techniques, the class $\hat{C}_k$ is from a set of K known classes $\{C_k\}$.
For a Detection function, there are K=2 classes : *k=1* is a positive detection P, *k=2* is a negative detection, N. Thus $C_k \in \{P,N\}$

Almost all current classification techniques require the number of classes, K, to be fixed. ($\{C_k\}$ is a closed set). An interesting research problem is how to design classification algorithms that allow $\{C_k\}$ to be an open set that grows with experience.

Discriminant and Decision Functions

The classification function $R(\vec{X})$ can typically be decomposed into two parts:

$$\hat{C}_k \leftarrow R(\vec{X}) = d\left(\vec{g}\left(\vec{X}\right)\right)$$

where $\vec{g}\left(\vec{X}\right)$ is a discriminant function and $d\left(\vec{g}\left(\vec{X}\right)\right)$ is a decision function.

$\vec{g}\left(\vec{X}\right)$: A discriminant function that transforms: $\vec{X} \rightarrow \mathbb{R}^K$
(A vector of real numbers)

$d\left(\vec{g}\left(\vec{X}\right)\right)$: A decision function $\mathbb{R}^K \rightarrow \hat{C}_k \in \{C_k\}$

The discriminant is typically a vector of functions, with one for each class.

$$\vec{g}(\vec{X}) = \begin{pmatrix} g_1(\vec{X}) \\ g_2(\vec{X}) \\ \vdots \\ g_K(\vec{X}) \end{pmatrix}$$

The decision function, $d\left(\vec{g}\left(\vec{X}\right)\right)$, can be an arg-max{}, a logistics sigmoid function, or any other function that selects C_k from $\vec{X}$.
A common choice is arg-max{}.

$$\hat{C}_k = d\left(\vec{g}\left(\vec{X}\right)\right) = \arg\max_{C_k}\left\{g_k\left(\vec{X}\right)\right\}$$

In some problems, there is a notion of "cost" for errors that the cost is different for different decisions. In this case we will seek to minimize the cost of an error rather than the number of errors by biasing the classification with a notion of risk.

Machine Learning

Machine learning explores the study and construction of algorithms that can learn from and make predictions about data. Learning for Pattern Recognition is only one of many forms of machine learning. Over the last 50 years, many forms of machine learning have been developed for many different applications.

For pattern recognition, the common approach is to a set of "training data" to estimate the discriminant function $\vec{g}(\vec{X})$.

The danger with supervised learning is that the model may not generalize to data outside the training set. The quality of the recognizer depends on the degree to which the training data $\{\vec{X}_m\}$ represents the range of variations of real data.

A variety of algorithms have been developed, each with its own advantages and disadvantages.

Supervised Learning

Most classical methods for learning a recognition function, learn from a set of labeled training data, composed of *M* independent examples, $\{\vec{X}_m\}$ for which we know the true class $\{y_m\}$.

The set $\{\vec{X}_m\}$, $\{y_m\}$ is called the training data.

Having the true class $\{y_m\}$ makes it much easier to estimate the functions $g_k(\vec{X})$ of $\vec{g}(\vec{X}) = \begin{pmatrix} g_1(\vec{X}) \\ g_2(\vec{X}) \\ \vdots \\ g_K(\vec{X}) \end{pmatrix}$

Most of the technique that we will see will use supervised learning.

Unsupervised Learning

Unsupervised Learning techniques learn the recognition function without a labeled training set. Such methods typically require a much larger sample of data for learning.

Semi-Supervised Learning.

A number of hybrid algorithms exist that initiate learning from a labeled training set and then extend the learning with unlabeled data.

Training and Validation

In most learning algorithms, we use the training data both to estimate the recognizer and to evaluate the results. However, there is a FUNDAMENTAL RULE in machine learning:

NEVER TEST WITH THE SAME DATA !

A typical approach is to use cross validation (also known as rotation estimation) in learning.

Cross validation partitions the training data into N folds (or complementary subsets). A subset of the folds are used to train the classifier, and the result is tested on the other folds. A taxonomy of common techniques include:

- Exhaustive cross-validation
  - Leave p-out cross-validation
  - Leave one-out cross-validation

- Non-exhaustive cross-validation
  - k-fold cross-validation
  - 2-fold cross-validation
  - Repeated sub-sampling validation

2. Two-Class Pattern Detectors

A pattern detector is a classifier with K=2.

Class k=1: The target pattern, also known as P or positive

Class k=2: Everything else, also known as N or negative.

Pattern detectors are used in computer vision, for example to detect faces, road signs, publicity logos, or other patterns of interest. They are also used in signal communications, data mining and many other domains.

The pattern detector is learned as a detection function $g(\vec{X})$ followed by a decision rule, *d()*. For K=2 this can be reduced to a single function, as

$$g(\vec{X}) = \frac{g_1(\vec{X})}{g_2(\vec{X})} \geq 0.5 \text{ is equivalent to } g_1(\vec{X}) \geq g_2(\vec{X})$$

(assuming $g_2(\vec{X}) \neq 0$)

Note that a "threshold" value other than 0.5 can be used. This is equivalent to "biasing" the detector.

The detection function is learned from a set of training data composed of *M* sample observations $\{\vec{X}_m\}$ where each sample observation is labeled with an indicator variable $\{y_m\}$

$y_m =$ P or Positive for examples of the target pattern (class k=1)

$y_m =$ N or Negative for all other examples (class k=2)

Observations for which $g(\vec{X}) > 0.5$ are estimated to be members of the target class. This will be called POSITIVE or P.

Observations for which $g(\vec{X}) \leq 0.5$ are estimated to be members of the background. This will be called NEGATIVE or N.

We can encode this as a decision function to define our detection function $R(\vec{X}_m)$

$$R(\vec{X}) = d(g(\vec{X})) = \begin{cases} P & \text{if } g(\vec{X}) \geq 0.5 \\ N & \text{if } g(\vec{X}) < 0.5 \end{cases}$$

For training we need ground truth (annotation). For each training sample the annotation or ground truth tells us the real class y_m

$$y_m = \begin{cases} P & \vec{X}_m \in \text{Target - Class} \\ \text{N} & \text{otherwise} \end{cases}$$

The Classification can be TRUE or FALSE.

if $R(\vec{X}_m) = y_m$ then T else F

This gives

$R(\vec{X}_m) = y_m$ AND $R(\vec{X}_m) = P$ is a TRUE POSITIVE or TP
$R(\vec{X}_m) \neq y_m$ AND $R(\vec{X}_m) = P$ is a FALSE POSITIVE or FP
$R(\vec{X}_m) \neq y_m$ AND $R(\vec{X}_m) = N$ is a FALSE NEGATIVE or FN
$R(\vec{X}_m) = y_m$ AND $R(\vec{X}_m) = N$ is a TRUE NEGATIVE or TN

To better understand the detector we need a tool to explore the trade-off between making false detections (false positives) and missed detections (false negatives). The Receiver Operating Characteristic (ROC) provides such a tool

3. Performance Metrics for 2 Class Detectors

ROC Curves

Two-class classifiers have long been used for signal detection problems in communications and have been used to demonstrate optimality for signal detection methods. The quality metric that is used is the Receiver Operating Characteristic (ROC) curve. This curve can be used to describe or compare any method for signal or pattern detection.

The ROC curve is generated by adding a variable Bias term to a discriminant function.

$$R(\vec{X}) = d(g(\vec{X}) + B)$$

and plotting the rate of true positive detection vs false positive detection where $R(\vec{X}_m)$ is the classifier as in lesson 1. As the bias term, B, is swept through a range of values, it changes the ratio of true positive detection to false positives.

For a ratio of histograms, $g(\vec{X}_m)$ is a probability ranging from 0 to 1.
B can range from less than -0.5 to more than $+0.5$.
When $B \le -0.5$ all detections will be Negative.
When $B > +0.5$ all detections will be Positive.
Between -0.5 and $+0.5$ $R(\vec{X})$ will give a mix of TP, TN, FP and FN.

The bias term, B, can act as an adjustable gain that sets the sensitivity of the detector. The bias term allows us to trade False Positives for False Negatives.

The resulting curve is called a Receiver Operating Characteristics (ROC) curve.
The ROC plots True Positive Rate (TPR) against False Positive Rate (FNR) as a function of B for the training data $\{\vec{X}_m\}$, $\{y_m\}$.

True Positives and False Positives

For each training sample, the detection as either Positive (P) or Negative (N)

IF $g(\vec{X}_m) + B > 0.5$ THEN P else N

The detection can be TRUE (T) or FALSE (F) depending on the indicator variable y_m

IF $y_m = R(\vec{X}_m)$ THEN T else F

Combining these two values, any detection can be a True Positive (TP), False Positive (FP), True Negative (TN) or False Negative (FN).

For the M samples of the training data $\{\vec{X}_m\}$, $\{y_m\}$ we can define:

#P as the number of Positives,
#N as the number of Negatives,
#T as the number of True and
#F as the number of False,

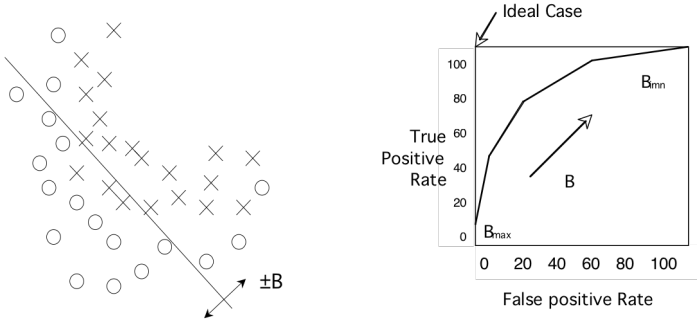
From this we can define:

#TP as the number of True Positives,
#FP as the number of False Positives,
#TN as the number of True Negative,
#FN as the number of False Negatives.

Note that #P = #TP + #FN
And #N = #FP+ #TN
The True Positive Rate (TPR) is $TPR = \frac{\#TP}{\#P} = \frac{\#TP}{\#TP+\#FN}$

The False Positive Rate (FPR) is $FPR = \frac{\#FP}{\#N} = \frac{\#FP}{\#FP+\#TN}$

The ROC plots the TPR against the FPR as a bias B is swept through a range of values.



When B is less than –0.5, all the samples are detected as N, and both the TPR and FPR are 0. As B increases both the TPR and FPR increase. Normally TPR should rise monotonically with FPR. If TPR and FPR are equal, then the detector is no better than chance.

The closer the curve approaches the upper left corner, the better the detector.

		$y_m = R(\vec{X}_m)$	
$d(g(\vec{X}_m)+B > 0.5)$		T	F
	P	True Positive (TP)	False Positive (FP)

	N	True Negative (TN)	False Negative (FN)

Precision and Recall

Precision, also called Positive Predictive Value (PPV), is the fraction of retrieved instances that are relevant to the problem.

$$PP = \frac{TP}{TP + FP}$$

A perfect precision score (PPV=1.0) means that every result retrieved by a search was relevant, but says nothing about whether all relevant documents were retrieved.

Recall, also known as sensitivity (S), hit rate, and True Positive Rate (TPR) is the fraction of relevant instances that are retrieved.

$$S = TPR = \frac{TP}{T} = \frac{TP}{TP + FN}$$

A perfect recall score (TPR=1.0) means that all relevant documents were retrieved by the search, but says nothing about how many irrelevant documents were also retrieved.

Both precision and recall are therefore based on an understanding and measure of relevance. In our case, “relevance” corresponds to “True”.
Precision answers the question “How many of the Positive Elements are True ?”
Recall answers the question “How many of the True elements are Positive”?

In many domains, there is an inverse relationship between precision and recall. It is possible to increase one at the cost of reducing the other.

F-Measure
The F-measures combine precision and recall into a single value. The F measures measure the effectiveness of retrieval with respect to a user who attaches 2 times as much importance to recall as precision.

The F_1 score weights recall higher than precision.

F_1 Score:

$$F_1 = \frac{2TP}{2TP + FP + FN}$$

The F1 score is the harmonic mean of precision and sensitivity.
This is the geometric mean divided by the arithmetic mean.

Accuracy
Accuracy is the fraction of test cases that are correctly classified (T).

$$ACC = \frac{T}{M} = \frac{TP + TN}{M}$$

where M is the quantity of test data.

Note that the terms Accuracy and Precision have a very different meaning in Measurement theory. In measurement theory, accuracy is the average distance from a true value, while precision is a measure of the reproducibility for the measurement.

Matthews Correlation Coefficient
The Matthews correlation coefficient is a measure of the quality of binary (two-class) classifications. This measure was proposed by the biochemist Brian W. Matthews in 1975.

MCC takes into account true and false positives and negatives and is generally regarded as a balanced measure that can be used even if the classes are of very different sizes.

The MCC is in essence a correlation coefficient between the observed and predicted binary classifications

MCC results a value between +1 and -1, where +1 represents a perfect prediction, 0 no better than random prediction and −1 indicates total disagreement between prediction and observation.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

The original formula given by matthews was:

M = Total quanitity of test data:

$$M = TN + TP + FN + FP$$

$$S = \frac{TP + FN}{M}$$

$$P = \frac{TP + FP}{M}$$

$$MCC = \frac{TP/M - S \cdot P}{\sqrt{PS(1-S)(1-P)}}$$