COMP 9517 Computer Vision

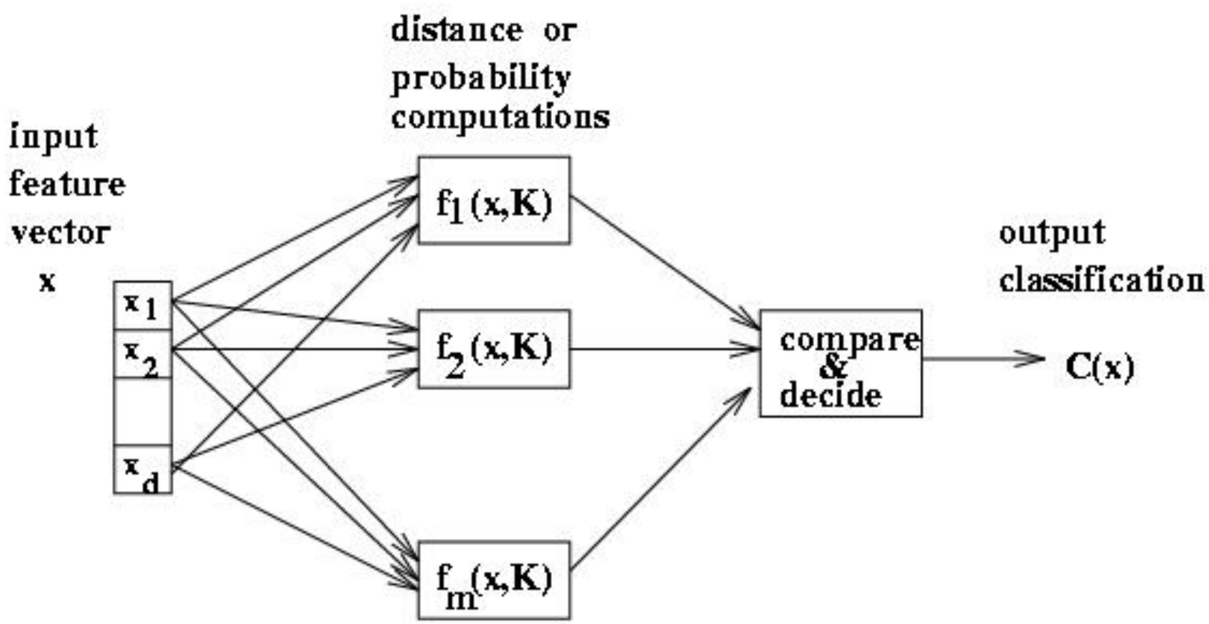
Pattern Recognition (3)

Classification Principles

- A statistical classifier has *n* inputs and *1* output.
  - Each input describes information about one of the *n* features x_1, x_2,...x_n
  - An *R*-class classifier will generate one of *R* symbols Ω_1, Ω_2,... Ω_R as an output
- The Ω are called the ***class identifiers***
- $d(\mathbf{x}) = \Omega_R$ is the ***decision rule***
  - It divides the feature space into R disjoint subsets K_R, $r = 1, 2, \ldots R$, each of which includes all the feature vectors $\mathbf{x}$ for which $d(\mathbf{x}) = \Omega_R$
- ***Discrimination hyper-surfaces***
  - The borders between the subsets K_r
- ***Discriminant functions***
  - *R* scalar functions $g_1(\mathbf{x})$, $g_2(\mathbf{x})$,... $g_R(\mathbf{x})$ define the hyperspaces

Discriminant functions

- Functions f(x, K) perform some computation on feature vector x
- Knowledge K about the class is used
- Final stage determines class



Separability

- ***Separable classes***
  - if a discrimination hyperspace exists that separates the feature space such that only objects from one class are in each region, then the recognition task has separable classes
- ***Linearly separable***
  - if the discrimination hyperspaces are hyperplanes, it is linearly separable

Linear Classifier

- For all $\boldsymbol{x} \in K_r$ and for any $s \in \{1,..., R\}, s \neq r: g_r(\boldsymbol{x}) \geq g_s(\boldsymbol{x})$
- Therefore, the discrimination hyperspace between classes K_r and K_s is defined by $g_r(\boldsymbol{x}) - g_s(\boldsymbol{x}) = 0$
- From this definition, we obtain the following decision rule:
  – Classify the object pattern ***x*** into that class whose discrimination function gives a maximum of all the discriminant functions:

    $$d(\boldsymbol{x}) = \Omega_R \Leftrightarrow g_r(\boldsymbol{x}) = max\, g_s(\boldsymbol{x})$$

  – If the discriminant functions are linear, their form is:

    $g_r(\boldsymbol{x}) = q_{r0} + q_{r1}x_1 + ... + q_{rn}x_n$, for all $r = 1,..., R$.

    The corresponding classifier is called a ***linear classifier***

Minimum Distance Principle

- Special case of classifiers based on discriminant functions, but computationally simpler
  - Nearest Class Mean Classifier
  - Nearest Neighbours
- Assume R points are defined in feature space $v_1, v_2, \ldots, v_R$ that represent exemplars of the $\Omega_1, \Omega_2, \ldots \Omega_R$.
- A minimum distance classifier classifies pattern ***x*** into the class to whose exemplar it is closest.
  $d(\mathbf{x}) = \Omega_R \Leftrightarrow |v_r - x| = min(|v_s - x|)$
- In this case, each discriminant hyper-plane is perpendicular to the line segment $v_r v_s$ and bisects it.
- If each class is represented by just one exemplar, we get a linear classifier.
- If more than one exemplar per class is used, we get piece-wise linear discrimination hyper-planes.

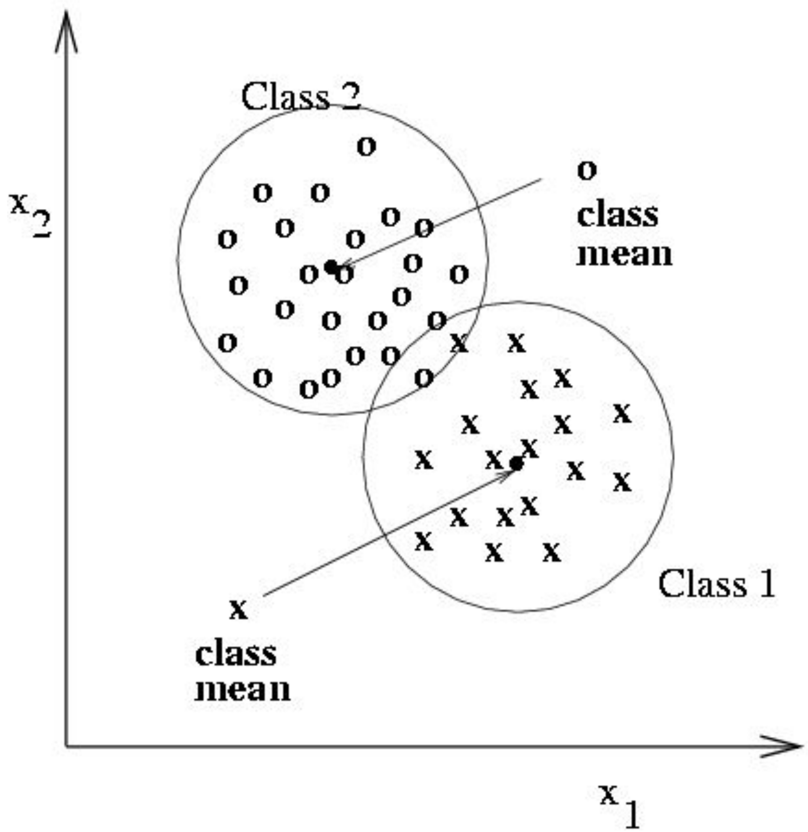
Nearest Class Mean Classifier

- This is a classifier based on minimum distance principle, where the class exemplars are just the centroids (or means)
- Training
  - summarises sample data from each class using the class mean vector or *centroid*:

$$x_i = \frac{1}{n_i} \sum_{j=1 \ldots n_i} x_{i,j}$$

    where $x_{i,j}$ is the j_{th} sample feature vector from class *I*
- Test
  - A new unknown object with feature vector ***x*** is classified as class *i* if it is much closer to the mean vector of class *i* than to any other class mean vector

Nearest Class Mean Classifier

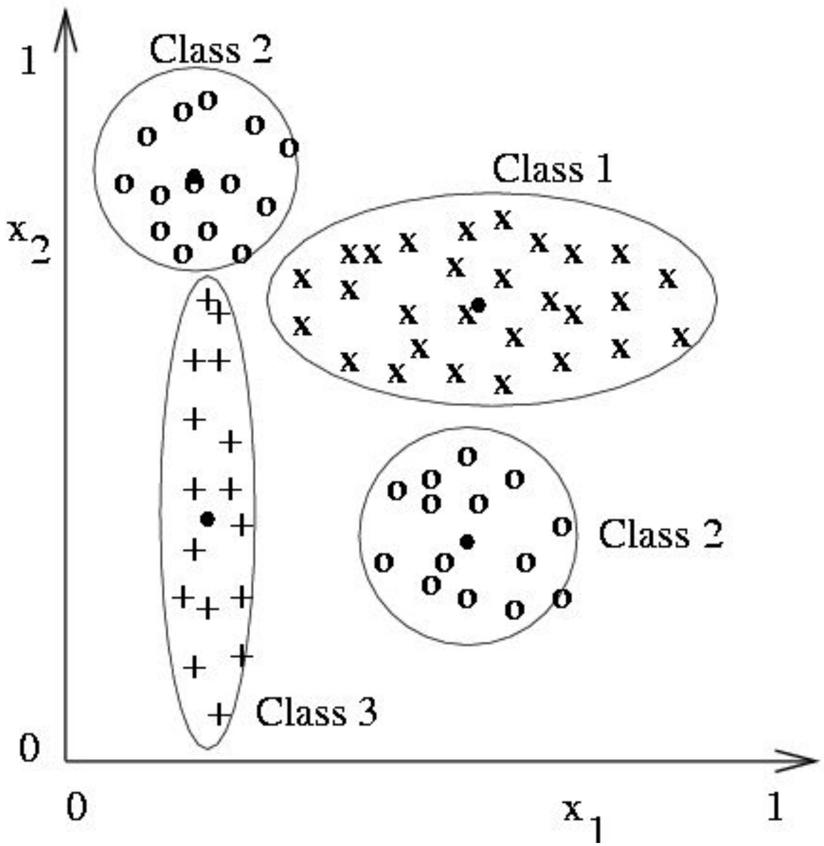


- Compute the Euclidean distance between feature vector X and the mean of each class
- Choose closest class, if close enough (reject otherwise)

Nearest Class Mean Classifier

- Simple, fast, works when classes are compact and far from each other.
- However, if classes are complex (eg. multimodal, non-spherical) nearest mean classification may give poor results
- One solution in such cases is to scale the distance by the spread, or **standard deviation** σ_i of class *c* along each dimension *i*.
- Co-ordinate transforms may be required if class axes are not aligned with co-ordinate axes

Nearest Class Mean Classifier



- Class 2 has two modes; where is its mean?
- But if modes are detected, two subclass mean vectors can be used

Nearest Neighbours

- Training
  - simply store the training examples
- Test
  - classify unknown sample vector $\boldsymbol{x}$ into the class of the individual sample closest to it
- More flexible but also more expensive
- Works well when classes have complex structure or overlap
- No assumptions on models, uses only existing training samples

Nearest Neighbour

- Brute force approach computes distance from ***x*** to all samples, and remembers minimum distance
- Works in incremental setting
- Trees or grids may be used as data structures to eliminate unnecessary distance computations
- A better version examines the nearest k feature vectors, $k > 1$
- As number of samples grows, the error rate for even $k = 1$ is no worse than twice the optimal error rate
- Transferring the original features space into another may improve the performance
  - Using metric learning

Structural Techniques

- Simple numeric or symbolic features may not be sufficient for object recognition
- ***Relationships*** among features can be used as higher-level, more powerful features for recognition
- In this approach, called ***structural pattern recognition***, an object is represented by its primitive parts, their attributes and relationships, as well as its global features
- When the relationships between primitive features are binary, a structural description is a ***graph structure***
- Recognition is then by graph-matching techniques

Other Classifiers

- Neural Networks, including Deep Learning
- Support Vector Machines
- Graphical Models, including Bayesian Networks

Evaluation of Error

- **Error rate**
  - error rate of classification system measures how well the system solves the problem it was designed for
- **Reject class**
  - generic class for objects that cannot be placed in any of the known classes
- **Performance**
  - Performance determined by both error and rejections made
  - Classifying all inputs into reject class means system makes no errors, but is useless!
- **Classification error**
  - The classifier makes classification error whenever it classifies input object as class C_i when true class is C_j, $i \neq j$, and $C_i \neq C_r$, the reject class
- **Empirical error rate**
  - Empirical error rate is the number of errors made on independent test data divided by number of classifications attempted

Evaluation of Error

- **Empirical reject rate**
  – is the number of rejects on independent test data divided by number of classifications attempted
- **Independent test data**
  – are sample objects with true class (labels) known, including objects from the reject class, and that were not used in designing the feature extraction and classification algorithms
- Samples used for training and testing should be representative

False Alarms and False Dismissals

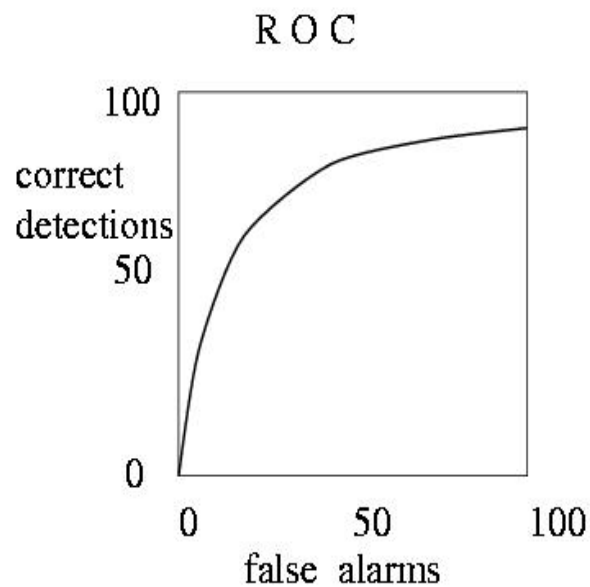
- For two-class problems, the errors have a special meaning and are not symmetric
- For example, in medical diagnosis, when a person has disease versus not have disease:
  - If the person does NOT have the disease, but the system incorrectly says she does, then the error is a ***false alarm/false positive***
  - On the other hand, if the person DOES have the disease, but the system incorrectly says he does NOT, then the error is a ***false dismissal or false negative***
- Consequences and costs of the two errors are very different

False Alarms and False Dismissals

- There are bad consequences to both, but false negative is generally more catastrophic
- So, we generally try to bias the system to minimize false negatives, possibly at the cost of increasing the false positives
- The ***Receiver Operator Curve (ROC)*** relates the false alarm rate to correct detection rate
- In order to increase correct detections, we may have to pay the cost of higher number of false alarms.

Receiver Operating Curve ROC

- Plots correct detection rate versus false alarm rate
- Generally, false alarms go up with attempts to detect higher percentages of known objects
- AUC



actual input object	decision	error type?
frack	frack	correct alarm (no error)
not a frack	frack	false alarm (error)
frack	not a frack	false dismissal (error)
not a frack	not a frack	correct dismissal (no error)

Confusion Matrix

- Confusion Matrix
  - Matrix whose entry *(i, j)* records the number of times that an object truly of class *i* was classified as class *j (True positive)*
- Used to report results of classification experiments
- The diagonal entries indicate the successes
- High off-diagonal numbers indicate confusion between classes

class j output by the pattern recognition system

true object class i	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	'R'
'0'	97	0	0	0	0	0	1	0	0	1	1
'1'	0	98	0	0	1	0	0	1	0	0	0
'2'	0	0	96	1	0	1	0	1	0	0	1
'3'	0	0	2	95	0	1	0	0	1	0	1
'4'	0	0	0	0	98	0	0	0	0	2	0
'5'	0	0	0	1	0	97	0	0	0	0	2
'6'	1	0	0	0	0	1	98	0	0	0	0
'7'	0	0	1	0	0	0	0	98	0	0	1
'8'	0	0	0	1	0	0	1	0	96	1	1
'9'	1	0	0	0	3	0	0	0	1	95	0

confusion may be unavoidable between some classes for example, between 9's and 4's, or between u's and j's for handprinted characters

Confusion Matrix

- Table of Confusion
  - For binary classification

		Prediction Outcome	
		P	N
Actual Vale	P’	**True Positive(TP)**	False Negative (FN)
	N’	False Positive(FP)	**True Negative(TN)**

- Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision versus Recall

- ***Precision/correctness***
  - is the number of relevant objects retrieved / classified divided by the total number of objects retrieved/classified

$$\text{Precision} = \frac{TP}{TP + FP}$$

- ***Recall/sensitivity/completeness***
  - is the number of relevant objects retrieved / classified divided by total number of relevant/correct objects

$$\mathrm{Re}\,call = \frac{TP}{TP + FN}$$

More Terminology

true positive (TP)

eqv. with hit

true negative (TN)

eqv. with correct rejection

false positive (FP)

eqv. with false alarm, Type I error

false negative (FN)

eqv. with miss, Type II error

sensitivity or true positive rate (TPR)

eqv. with hit rate, recall

$TPR = TP / P = TP / (TP + FN)$

false positive rate (FPR)

eqv. with fall-out

$FPR = FP / N = FP / (FP + TN)$

accuracy (ACC)

$ACC = (TP + TN) / (P + N)$

specificity (SPC) or True Negative Rate

$SPC = TN / N = TN / (FP + TN) = 1 - FPR$

positive predictive value (PPV)

eqv. with precision

$PPV = TP / (TP + FP)$

negative predictive value (NPV)

$NPV = TN / (TN + FN)$

false discovery rate (FDR)

$FDR = FP / (FP + TP)$

Matthews correlation coefficient (MCC)

$$MCC = (TP * TN - FP * FN)/\sqrt{PNP'N'}$$

F1 score

$F1 = 2TP^2 / (P + P')$

References and Acknowledgements

- Shapiro and Stockman, Chapter 4
- Duda, Hart and Stork, Chapter 1
- Richard Szeliski， Chapter 14
- More references
  - Sergios Theodoridis, Konstantinos Koutroumbas, *Pattern Recognition*, 2009
  - Ian H. Witten, Eibe Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2005
- Some diagrams are extracted from the above resources