Lecture 12: Classification

- Discriminant functions
- The optimal Bayes classifier
- Quadratic classifiers
- Euclidean and Mahalanobis metrics
- K Nearest Neighbor Classifiers
Discriminant functions

- A convenient way to represent a pattern classifier is in terms of a family of discriminant functions $g_i(x)$ with a simple MAX gate as the classification rule.

- How do we choose the discriminant functions $g_i(x)$?
  - Depends on the objective function to minimize:
    - Probability of error
    - Bayes Risk

Assign $x$ to class $\omega_i$ if $g_i(x) > g_j(x)$ $\forall j \neq i$
Minimizing probability of error

- Probability of error $P[\text{error}|x]$ is “the probability of assigning $x$ to the wrong class”
  - For a two-class problem, $P[\text{error}|x]$ is simply
    $$P(\text{error} | x) = \begin{cases} P(\omega_1 | x) & \text{if we decide } \omega_2 \\ P(\omega_2 | x) & \text{if we decide } \omega_1 \end{cases}$$
  - It makes sense that the classification rule be designed to minimize the average probability of error $P[\text{error}]$ across all possible values of $x$
    $$P(\text{error}) = \int_{-\infty}^{+\infty} P(\text{error}, x)dx = \int_{-\infty}^{+\infty} P(\text{error} | x)P(x)dx$$
  - To ensure $P(\text{error})$ is minimum we minimize $P(\text{error}|x)$ by choosing the class with maximum posterior $P(\omega_i | x)$ at each $x$
    - This is called the MAXIMUM A POSTERIORI (MAP) RULE
    - And the associated discriminant functions become
      $$g_i^{\text{MAP}}(x) = P(\omega_i | x)$$
Minimizing probability of error

- We “prove” the optimality of the MAP rule graphically
  - The right plot shows the posterior for each of the two classes
  - The bottom plots shows the $P(\text{error})$ for the MAP rule and another rule
  - Which one has lower $P(\text{error})$ (color-filled area)?

THE MAP RULE

THE “OTHER” RULE


**Quadratic classifiers**

- Let us assume that the likelihood densities are Gaussian

\[
P(x \mid \omega_i) = \frac{1}{(2 \pi)^{n/2} |\Sigma_i|^{1/2}} \exp \left[ -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right]
\]

- Using Bayes rule, the MAP discriminant functions become

\[
g_i(x) = P(\omega_i \mid x) = \frac{P(x \mid \omega_i) P(\omega_i)}{P(x)} = \frac{1}{(2 \pi)^{n/2} |\Sigma_i|^{1/2}} \exp \left[ -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right] P(\omega_i) \frac{1}{P(x)}
\]

  - Eliminating constant terms

\[
g_i(x) = |\Sigma_i|^{1/2} \exp \left[ -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right] P(\omega_i)
\]

  - We take natural logs (the logarithm is monotonically increasing)

\[
g_i(x) = -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) - \frac{1}{2} \log(|\Sigma_i|) + \log(P(\omega_i))
\]

  - This is known as a **Quadratic Discriminant Function**
  - The quadratic term is known as the **Mahalanobis distance**
Mahalanobis distance

- The Mahalanobis distance can be thought of vector distance that uses a $\Sigma_i^{-1}$ norm

$$\|x - y\|_{\Sigma_i^{-1}}^2 = (x - y)^T \Sigma_i^{-1} (x - y)$$

- $\Sigma^{-1}$ can be thought of as a stretching factor on the space
- Note that for an identity covariance matrix ($\Sigma_i = I$), the Mahalanobis distance becomes the familiar Euclidean distance

- In the following slides we look at special cases of the Quadratic classifier
  - For convenience we will assume equiprobable priors so we can drop the term $\log(P(\omega_i))$
Special case I: $\Sigma_i=\sigma^2 I$

- In this case, the discriminant becomes

$$g_i(x) = -(x - \mu_i)^T (x - \mu_i)$$

- This is known as a **MINIMUM DISTANCE CLASSIFIER**
- Notice the linear decision boundaries
Special case 2: \( \Sigma_i = \Sigma (\Sigma \text{ diagonal}) \)

- In this case, the discriminant becomes

\[
g_i(x) = -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)
\]

- This is known as a MAHALANOBIS DISTANCE CLASSIFIER
- Still linear decision boundaries
**Special case 3: $\Sigma_i = \Sigma$ ($\Sigma$ non-diagonal)**

- In this case, the discriminant becomes
  \[
g_i(x) = -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)
  \]

  - This is also known as a **MAHALANOBIS DISTANCE CLASSIFIER**
  - Still linear decision boundaries
Case 4: $\Sigma_i = \sigma_i^2 I$, example

- In this case the quadratic expression cannot be simplified any further.
- Notice that the decision boundaries are no longer linear but quadratic.
Case 5: $\Sigma_i \neq \Sigma_j$ general case, example

- In this case there are no constraints so the quadratic expression cannot be simplified any further
- Notice that the decision boundaries are also quadratic
Limitations of quadratic classifiers

- The fundamental limitation is the unimodal Gaussian assumption
  - For non-Gaussian or multimodal Gaussian, the results may be significantly sub-optimal

- A practical limitation is associated with the minimum required size for the dataset
  - If the number of examples per class is less than the number of dimensions, the covariance matrix becomes singular and, therefore, its inverse cannot be computed
    - In this case it is common to assume the same covariance structure for all classes and compute the covariance matrix using all the examples, regardless of class
Conclusions

- We can extract the following conclusions
  - The Bayes classifier for normally distributed classes is quadratic
  - The Bayes classifier for normally distributed classes with equal covariance matrices is a linear classifier
  - The minimum Mahalanobis distance classifier is optimum for
    - normally distributed classes and equal covariance matrices and equal priors
  - The minimum Euclidean distance classifier is optimum for
    - normally distributed classes and equal covariance matrices proportional to the identity matrix and equal priors
  - Both Euclidean and Mahalanobis distance classifiers are linear

- The goal of this discussion was to show that some of the most popular classifiers can be derived from decision-theoretic principles and some simplifying assumptions
  - It is important to realize that using a specific (Euclidean or Mahalanobis) minimum distance classifier implicitly corresponds to certain statistical assumptions
  - The question whether these assumptions hold or don’t can rarely be answered in practice; in most cases we are limited to posting and answering the question “does this classifier solve our problem or not?”
**K Nearest Neighbor classifier**

- The kNN classifier is based on non-parametric density estimation techniques
  - Let us assume we seek to estimate the density function $P(x)$ from a dataset of examples
  - $P(x)$ can be approximated by the expression
    \[
    P(x) \approx \frac{k}{NV} \quad \text{where} \quad \begin{cases} 
    V & \text{is the volume surrounding } x \\
    N & \text{is the total number of examples} \\
    k & \text{is the number of examples inside } V
    \end{cases}
    \]
  - The volume $V$ is determined by the $D$-dim distance $R_k^D(x)$ between $x$ and its $k$ nearest neighbor
    \[
    P(x) \approx \frac{k}{NV} = \frac{k}{N \cdot c_D \cdot R_k^D(x)}
    \]
  - Where $c_D$ is the volume of the unit sphere in $D$ dimensions
  
  \[
  V = \pi R^2 \\
  P(x) = \frac{k}{N \pi R^2}
  \]
**K Nearest Neighbor classifier**

- We use the previous result to estimate the posterior probability
  - The unconditional density is, again, estimated with
    \[ P(x | \omega_i) = \frac{k_i}{N_i} \]
  - And the priors can be estimated by
    \[ P(\omega_i) = \frac{N_i}{N} \]
  - The posterior probability then becomes
    \[ P(\omega_i | x) = \frac{P(x | \omega_i)P(\omega_i)}{P(x)} = \frac{k_i \cdot N_i}{N \cdot k} = \frac{k_i}{k} \]
  - Yielding discriminant functions
    \[ g_i(x) = \frac{k_i}{k} \]
- This is known as the k Nearest Neighbor classifier
K Nearest Neighbor classifier

- The kNN classifier is a very intuitive method
  - Examples are classified based on their similarity with training data
  - For a given unlabeled example $x_u \in \mathbb{R}^D$, find the k “closest” labeled examples in the training data set and assign $x_u$ to the class that appears most frequently within the k-subset

- The kNN only requires
  - An integer k
  - A set of labeled examples
  - A measure of “closeness”

![Graph showing data points with different labels and a question mark indicating an unlabeled example.](image-url)
kNN in action: example 1

- We generate data for a 2-dimensional 3-class problem, where the class-conditional densities are multi-modal, and non-linearly separable
- We used kNN with
  - $k = $ five
  - Metric = Euclidean distance
**kNN in action: example 2**

- We generate data for a 2-dim 3-class problem, where the likelihoods are unimodal, and are distributed in rings around a common mean
  - These classes are also non-linearly separable
- We used kNN with
  - k = five
  - Metric = Euclidean distance
kNN versus 1NN

1-NN  
5-NN  
20-NN
Characteristics of the kNN classifier

- **Advantages**
  - Analytically tractable, simple implementation
  - Nearly optimal in the large sample limit \((N \to \infty)\)
    - \(P[\text{error}]_{\text{Bayes}} > P[\text{error}]_{1-\text{NNR}} < 2P[\text{error}]_{\text{Bayes}}\)
  - Uses local information, which can yield highly adaptive behavior
  - Lends itself very easily to parallel implementations

- **Disadvantages**
  - Large storage requirements
  - Computationally intensive recall
  - Highly susceptible to the curse of dimensionality

- **1NN versus kNN**
  - The use of large values of \(k\) has two main advantages
    - Yields smoother decision regions
    - Provides probabilistic information: The ratio of examples for each class gives information about the ambiguity of the decision
  - However, too large values of \(k\) are detrimental
    - It destroys the locality of the estimation
    - In addition, it increases the computational burden