Pattern Classification

EET3053

Lecture 04: Estimation Techniques

Dr. Kundan Kumar
Associate Professor
Department of ECE

Faculty of Engineering (ITER)
S'O'A Deemed to be University, Bhubaneswar, India-751030

Parametric Estimation Techniques

Introduction

- Data availability in a Bayesian framework
  - We could design an optimal classifier if we know
    - $P(w_j)$ (priors)
    - $p(\mathrm{x}|w_j)$ (class-conditional densities)

    Unfortunately, we rarely have this complete information.
- Design a classifier from training samples
  - No problem with prior estimation
  - Samples are often too small for class-conditional estimation (large dimension of feature space)
- Some priori information about the problem should be known.
  - Normality of $p(\mathrm{x}|w_j)$

$$p(\mathrm{x}|w_j) \sim N(\mu_j, \Sigma_j)$$

Introduction

- Parametric estimation techniques
  - *Maximum-Likelihood Estimation (MLE)* and
  - *Bayesian Estimations*
- Some other approaches for parameter estimation
  - Histogram based technique
  - Parzen-Rosenblatt window technique (Kernel/Window based technique)
- Results are nearly identical, but approaches are different
- Parameters in MLE are fixed but unknown.
- Best parameters are obtained by maximizing the probability of obtaining the samples observed.
- Bayesian methods view the parameters as random variables having some known distribution
- In either approach, we use $P(w_i|\mathrm{x})$ for our classification rule.

Difference between ML and Bayesian estimation

- *Maximum-Likelihood Estimation (MLE)*
  - Views the parameters as quantities whose values are fixed but unknown.
  - We Estimate these values by maximizing the probability of obtaining the samples observed.
- *Bayesian Estimations*
  - Views the parameters as random variables having some known prior distribution.
  - We observe new samples and converts the prior to a posterior density.

Maximum Likelihood Estimation

Maximum Likelihood Estimation

- $C \rightarrow$ no. of classes
- $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \ldots, \mathcal{D}_C$ (set of features for different classes)
- $p(\mathrm{x}|w_j) \rightarrow$ known parametric form

$$p(\mathrm{x}|w_j) \sim N(\mu_j, \Sigma_j)$$

where μ_j is the mean vector, and Σ_j is the co-variance matrix.

- For parameter vector $\theta_j = [\mu_j, \Sigma_j]^T$, the parametric probability distribution function as

$$p(\mathrm{x}|w_j) \equiv p(\mathrm{x}|w_j, \theta_j) = p(\mathrm{x}|\theta_j)$$

- Here our objective is to use the information from the training samples in set $\mathcal{D}_j$ to obtain good estimates for the unknown parameter vector θ_j.
- We can apply MLE on individual set to estimate the parameters.

Maximum Likelihood Estimation

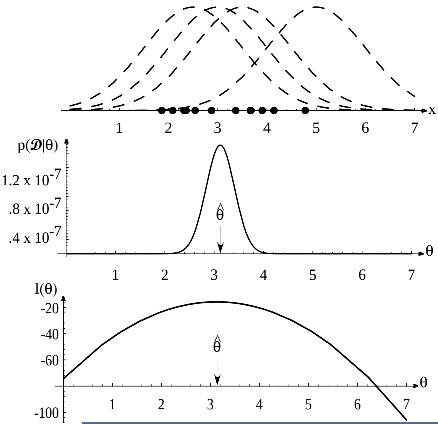
- Let us assume the set $\mathcal{D}_j = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n\}$ of independent and identically distributed (i.i.d.) samples drawn from the density $p(\mathrm{x}|\theta_j)$
- That means $\mathcal{D}_i$ does not provide any information about the parameter vector θ_j for $i \neq j$, *i.e.*, samples from one class do not provide any information of the parameter vector of the *probability density function* of another class.
- Thus, we can work with each class separately and omit the class labels (j), so that we write the probability density as $p(\mathrm{x}|\theta)$.
- Thus, the probability of observing $\mathcal{D} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n\}$ is

$$p(\mathcal{D}|\theta) = p(\mathrm{x}_1|\theta) * p(\mathrm{x}_2|\theta) * \cdots * p(\mathrm{x}_n|\theta) = \prod_{k=1}^{n} p(\mathrm{x}_k|\theta)$$

  where n is the number of data samples in set $\mathcal{D}$.
- $p(\mathcal{D}|\theta)$ is also called the likelihood of θ with respect to the set of samples $\mathcal{D}$.

Maximum Likelihood Estimation

- The *maximum-likelihood estimation* of θ is, by definition, the value $\hat{\theta}$ that maximizes $p(\mathcal{D}|\theta)$.



$$p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathrm{x}_k|\theta)$$

$$l(\theta) = \ln p(\mathcal{D}|\theta) = \sum_{k=1}^{n} \ln p(\mathrm{x}_k|\theta)$$

Solution

$$\hat{\theta} = \arg\max_{\theta} l(\theta)$$

Maximum Likelihood Estimation: Optimal estimation

- Let $\theta = (\theta_1, \theta_2, \ldots, \theta_p)^T$, and ∇_θ be the gradient operator

$$\nabla_\theta = \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_p} \end{bmatrix}$$

- $\nabla_\theta l(\theta) = 0$
- Example of a specific case: Gaussian distribution
- Multivariate normal population with (μ, Σ)

Gaussian case: Unknown μ

- σ^2 is known, only μ is unknown.

$$\ln p(\mathbf{x}_k|\boldsymbol{\mu}) = -\frac{1}{2}\ln\left[(2\pi)^d|\boldsymbol{\Sigma}|\right] - \frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu})$$

$$\nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\mu}) = \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}).$$

- The maximum likelihood estimate for μ must satisfy

$$\sum_{k=1}^{n}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}) = \mathbf{0}$$

- Each of the d component of $\hat{\mu}$ must vanish.

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

Gaussian case: Unknown μ and Σ

- $\theta_1 = \mu$ and $\theta_2 = \sigma^2$ are unknown.
- The log-likelihood of a single point is

$$\ln p(x_k|\boldsymbol{\theta}) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2$$

- Derivative is

$$\nabla_{\boldsymbol{\theta}} l = \nabla_{\boldsymbol{\theta}} \ln p(x_k|\boldsymbol{\theta}) = \begin{bmatrix} \frac{1}{\theta_2}(x_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(x_k-\theta_1)^2}{2\theta_2^2} \end{bmatrix}$$

- After simplification

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k \qquad \hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \hat{\mu})^2$$

Example to be solved

- Estimate optimal parameter $\hat{\theta}$

$$p(x|\theta) = \begin{cases} \theta e^{-\theta x} & x \geq 0 \\ 0 & otherwise \end{cases}$$

using log-maximum likelihood estimation approach.

Solution: $\hat{\theta} = \frac{1}{\frac{1}{n}\sum_{k=1}^{n} x_k} = \frac{1}{\mu}$

Example to be solved

The random variable x follows the following pdf

$$p(x|\theta) = \begin{cases} \theta^2 x e^{-\theta x} & x \geqslant 0 \\ 0 & otherwise \end{cases}$$

Derive the maximum likelihood estimate of $\hat{\theta}$ given N measurements $x_1, x_2, \ldots, x_N$.

Example to be solved

Lex x be a d-dimensional binary (0 or 1) vector with a multivariate Bernoulli distribution

$$P(\mathbf{x}/\boldsymbol{\theta}) = \prod_{i=1}^{d} \theta_i^{x_i}(1-\theta_i)^{1-x_i}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)^t$ is an unknown parameter vector, θ_i being the probability that $x_i = 1$. Show that the maximum-likelihood estimate for $\boldsymbol{\theta}$ is

$$\hat{\boldsymbol{\theta}} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

Non-parametric parameter estimation

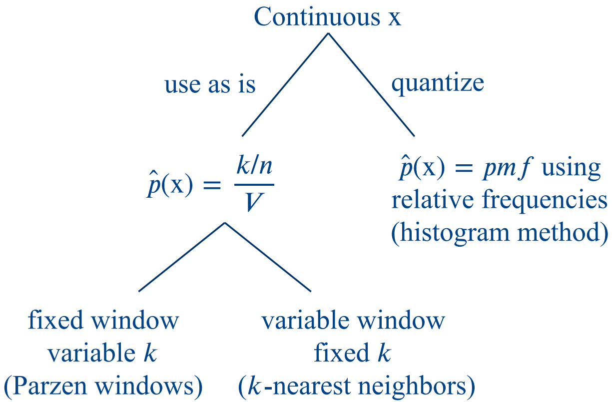
Introduction

- We have already seen that for statistical pattern classification, density function are to be known for each class.
- The type of density function, such as the Normal or Poisson, are to be known to estimate the parameters of the densities called *parametric estimation*.
- In most real problems, even the types of the density functions of interest are unknown.

Introduction

- Looking at histograms, scatter plots or tables of the data may suggest that a particular type of class density may be used or some arbitrary density can be used.
- Arbitrary density function can be estimated from the data samples using *nonparametric methods*.
- In addition, most of the classical parametric densities are unimodal, whereas many practical problems involve multimodal densities.
- Non-parametric methods can be used with arbitrary distributions and without the assumption that the forms of the underlying densities are known.

Non-parametric Methods



Histogram Method

Histogram Method

- A very simple method is to partition the space into a number of equally-sized cells (bins) and compute a histogram.

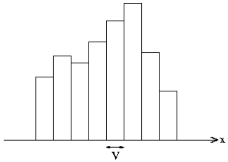


Figure: Histogram in one dimension

- The estimate of the density at a point x becomes

$$p(\mathrm{x}) = \frac{k}{nV}$$

where n is the total number of samples, k is the number of samples in the bin that includes x, and V is the volume of that cell.

- For 1-D feature, V is width of bin. Similarly for 2-D feature, V is the area of the bin.
- Thumb rule to choose the number of intervals (bins) to be equal to the square root of the number of samples.

Histogram Method

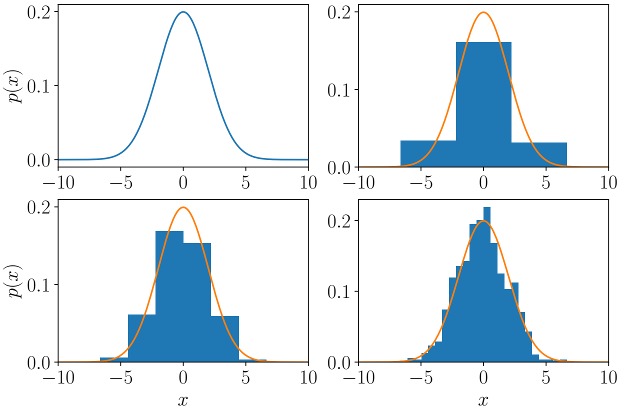


Figure: (a) The true normal density from which 50 random numbers were chosen. (b) A histogram of 50 normally distributed random numbers with three intervals. (c) A histogram of 50 normally distributed random numbers with six intervals. (d) A histogram of 50 normally distributed random numbers with 24 intervals.

Example to be solved

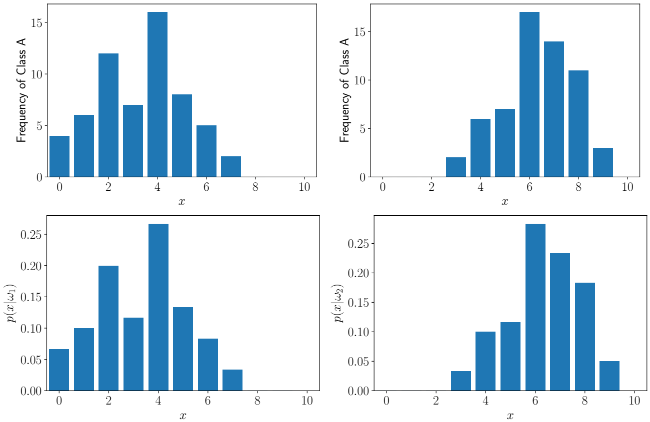
Question: Classification of samples using histograms and Bayesian decision rule
Use the following data to classify a sample with $x = 7.5$, given that $P(A) = P(B) = 0.5$. The following data are the values of feature x for 60 randomly chosen samples from
Class A:

0.80	0.91	0.93	0.95	1.32	1.53	1.57	1.63	1.67	1.74
2.01	2.18	2.27	2.31	2.40	2.61	2.64	2.64	2.67	2.85
2.96	2.97	3.17	3.17	3.38	3.67	3.73	3.83	3.99	4.06
4.10	4.12	4.18	4.20	4.23	4.27	4.27	4.39	4.40	4.46
4.47	4.61	4.64	4.89	4.96	5.12	5.15	5.33	5.33	5.47
5.64	5.85	5.99	6.29	6.42	6.53	6.70	6.78	7.18	7.22

Class B:

3.54	3.88	4.24	4.30	4.30	4.70	4.75	4.97	5.21	5.42
5.60	5.77	5.87	5.94	5.95	6.04	6.05	6.15	6.19	6.21
6.33	6.41	6.43	6.49	6.52	6.58	6.60	6.63	6.65	6.75
6.90	6.92	7.03	7.08	7.18	7.29	7.33	7.41	7.41	7.46
7.61	7.67	7.68	7.68	7.78	7.96	8.03	8.12	8.20	8.22
8.33	8.36	8.44	8.45	8.49	8.75	8.76	9.14	9.20	9.86

Example to be solved



$P(A|7.5) = 0.033$ and $P(B|7.5) = 0.233$, so the sample should be classified into class B.

2-D Histogram Method

- Histograms are not restricted to one-dimensional densities, but can be used in any number of dimensions.
- $p(x, y)$ can be approximated by dividing both x and y into intervals, and determining the number of samples that fall within each rectangular histogram bin with dimensions Δx and Δy.
- The volume under the surface of this two-dimensional histogram is to be normalized to equal one, to yield an estimate of the density function $p(x, y)$.
- The histogram technique becomes impractical for spaces of high dimension.

Kernel and Window Methods

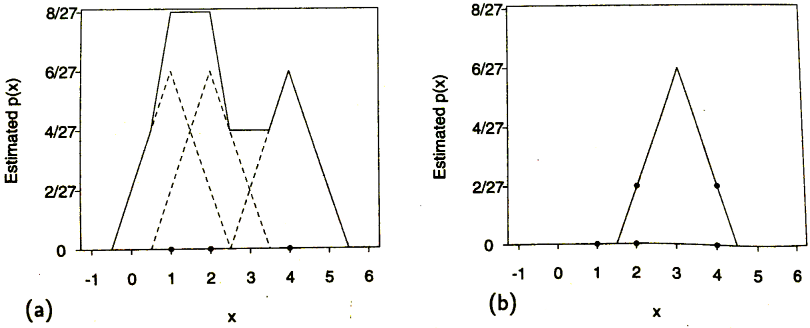
Kernel and Window Estimators

- The samples gives a very rough approximation to the true density function, namely a set of spikes or delta functions, one at each sample value, each with a very small width and a very large height.
- The combined area of all the spikes is one.
- Histogram based density approximation to a continuous density function is not useful in decision making.
- If the delta functions at each sample point are replaced by other function called *Kernels* – such as *rectangles*, *triangles*, or *normal density functions*, which have been scaled so that their combined area equals one-their sum produces a smoother, more satisfactory estimate.

Example to be solved

Question: Using a triangle kernel.

Consider the data set with one feature x and three samples at $x = 1, 2$, and 4. We have decided to use a triangular kernel with a base of three units. Plot the estimated density function $p(x)$.



Example to be solved

Question: Following sets of 2-D feature vectors from classes A and B are given

$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 3 \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2.5 \\ 2 \end{pmatrix}, \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 2 \end{pmatrix} \right\} \in A$$

$$\left\{ \begin{pmatrix} 3 \\ 2 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \end{pmatrix}, \begin{pmatrix} 4.5 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \begin{pmatrix} 6 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 6 \end{pmatrix}, \begin{pmatrix} 7 \\ 3 \end{pmatrix} \right\} \in B$$

Using rectangular window of size 3×3, compute $p((3.5, 3)^t | A)$ and $p((3.5, 3)^t | B)$. Classify $(3.5, 3)^t$ if $P(A) = 1/3$ and $P(B) = 2/3$.

Non-parametric Density Estimation

- Suppose that n samples $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n$ are drawn i.i.d. according to the distribution $p(\mathrm{x})$.
- The probability P that a vector x will fall in a region $\mathcal{R}$ is given by

$$P = \int_{\mathcal{R}} p(\mathrm{x}')d\mathrm{x}'$$

- The probability that k of the n will fall in $\mathcal{R}$ is given by the binomial law

$$P_k = \binom{n}{k} P^k (1-P)^{n-k}.$$

- The expected value of k is $E[k] = nP$ and the MLE for P is $\hat{P} = \frac{k}{n}$.

Non-parametric Density Estimation

- If we assume that $p(\mathrm{x})$ is continuous and $\mathcal{R}$ is small enough so that $p(\mathrm{x})$ does not vary significantly in it, we can get the approximation

$$\int_{\mathcal{R}} p(\mathrm{x}')d\mathrm{x}' \simeq p(\mathrm{x})V$$

  where x is a point in $\mathcal{R}$ and V is the volume of $\mathcal{R}$.
- Then, the density estimate becomes

$$p(\mathrm{x}) \simeq \frac{k/n}{V}.$$

Non-parametric Density Estimation

- Let n be the number of samples used, $\mathcal{R}_n$ be the region with n samples, V_n be the volume of $\mathcal{R}_n$, k_n be the number of samples falling in $\mathcal{R}_n$, and $p_n(\mathrm{x}) = \frac{k_n/n}{V_n}$ be the estimate for $p(\mathrm{x})$.
- If $p_n(\mathrm{x})$ is to converge to $p(\mathrm{x})$, three conditions are required:

$$\lim_{n\to\infty} V_n = 0$$

$$\lim_{n\to\infty} k_n = \infty$$

$$\lim_{n\to\infty} \frac{k_n}{n} = 0$$

Nearest Neighbor Classification

The Nearest Neighbor Classifier

- We have been using Bayesian classifiers that make decisions according to the posterior probabilities.
- We have discussed parametric and non-parametric methods for learning classifiers by estimating the probabilities using training data.
- We will study new techniques that use training data to learn the classifiers directly without estimating any probabilistic structure.
- In particular, we will study the k-nearest neighbour classifier, linear discriminant functions, and support vector machines.

The Nearest Neighbor Classifier

- Given the training data $\mathcal{D} = \{\mathrm{x}_1, \cdots, \mathrm{x}_n\}$ as a set of n labeled examples, the nearest neighbor classifier assigns a test point x the label associated with its closest neighbor in $\mathcal{D}$.
- Closeness is defined using a distance function.
- Given the distance function, the nearest neighbor classifier partitions the feature space into cells consisting of all points closer to a given training point than to any other training points.

The Nearest Neighbor Classifier

- All points in such a cell are labeled by the class of the training point, forming a Voronoi tesselation of the feature space.

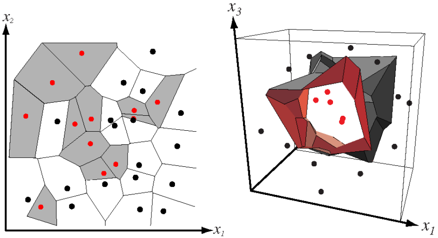


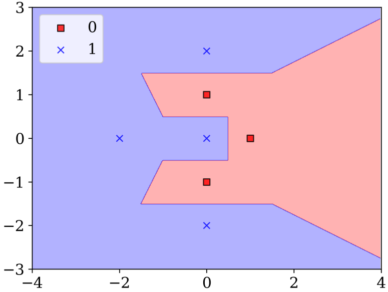
Figure: In two dimensions, the nearest neighbor algorithm leads to a partitioning of the input space into Voronoi cells, each labeled by the class of the training point it contains. In three dimensions, the cells are three-dimensional, and the decision boundary resembles the surface of a crystal.

Example to be solved

Question: Consider the following set of seven 2-dimensional feature vectors:

$$X_1 = (1,0)^t,\ X_2 = (0,1)^t,\ X_3 = (0,-1)^t,$$
$$X_4 = (0,0)^t,\ X_5 = (0,2)^t,\ X_6 = (0,-2)^t,\ X_7 = (-2,0)^t$$

If $X_1, X_2, X_3 \in \omega_1$ and $X_4, X_5, X_6, X_7 \in \omega_2$, sketch the decision boundary resulting from the nearest neighbor rule.



Nearest Neighbor Algorithm

Learning Algorithm:

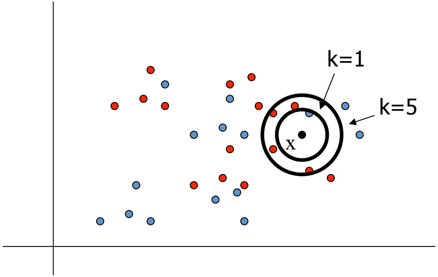
- Store training examples

Prediction Algorithm:

- To classify a new example x by finding the training example (x_i, y_i) that is nearest to x
- Guess the class $y = y_i$

k-Nearest Neighbor Classifier

- To classify a new input vector x, examine the k-closest training data points to x and assign the object to the most frequently occurring class.
- In other words, a decision is made by examining the labels on the k-nearest neighbors and taking a vote.



- common values for k: 3, 5

k-Nearest Neighbor Classifier

- The computational complexity of the nearest neighbor algorithm – both in space (storage) and time (search) – has received a great deal of analysis.
- In the most straightforward approach, we inspect each stored training point one by one, calculate its distance to x, and keep a list of the k closest ones.
- There are some parallel implementations and algorithmic techniques for reducing the computational load in nearest neighbor searches.

Distance Functions

Distance Functions

- The nearest neighbor classifier relies on a metric or a distance function between points.
- For all points x, y, and z, a metric $D(\cdot,\cdot)$ must satisfy the following properties:
  - Non-negativity: $D(\mathrm{x},\mathrm{y}) \geq 0$.
  - Reflexivity: $D(\mathrm{x},\mathrm{y}) = 0$ if and only if $\mathrm{x} = \mathrm{y}$.
  - Symmetry: $D(\mathrm{x},\mathrm{y}) = D(\mathrm{y},\mathrm{x})$.
  - Triangle inequality: $D(\mathrm{x},\mathrm{y}) + D(\mathrm{y},\mathrm{z}) \geq D(\mathrm{x},\mathrm{z})$.
- If the second property is not satisfied, $D(\cdot,\cdot)$ is called a pseudometric.

Distance Functions

- A general class of metrics for d-dimensional patterns is the Minkowski metric

$$L_p(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|^p \right)^{1/p}$$

  also referred to as the L_p norm.
- The Euclidean distance is the L_2 norm

$$L_2(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|^2 \right)^{1/2}.$$

- The Manhattan or city block distance is the L_1 norm

$$L_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|.$$

Distance Functions

- The L_∞ norm is the maximum of the distances along individual coordinate axes

$$L_\infty(\mathbf{x}, \mathbf{y}) = \max_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|.$$

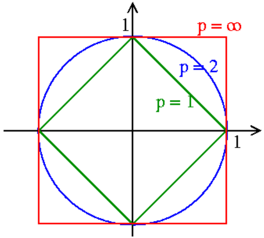


Figure: Each colored shape consists of points at a distance 1.0 from the origin, measured using different values of p in the Minkowski L_p metric.

Feature Normalization

- We should be careful about scaling of the coordinate axes when we compute these metrics.
- When there is great difference in the range of the data along different axes in a multidimensional space, these metrics implicitly assign more weighting to features with large ranges than those with small ranges.
- Feature normalization can be used to approximately equalize ranges of the features and make them have approximately the same effect in the distance computation.
- The following methods can be used to independently normalize each feature.

Feature Normalization

- Min-max normalization or Linear scaling to unit range:

$$\tilde{x} = \frac{x - \min}{\max - \min}$$

results in $\tilde{x}$ being in the $[0, 1]$ range, where $x \in \mathbb{R}$

- Standardization or Linear scaling to unit variance:
A feature $x \in \mathbb{R}$ can be transformed to a random variable with zero mean and unit variance as

$$\tilde{x} = \frac{x - \mu}{\sigma}$$

where μ and σ are the sample mean and the sample standard deviation of that feature, respectively.

References

[1] Hart, P. E., Stork, D. G., & Duda, R. O. (2000). Pattern classification. Hoboken: Wiley.

[2] Gose, E. (1997). Pattern recognition and image analysis.

Thank you!