SysEng 5212 /EE 5370
Introduction to Neural Networks and Applications

Lecture 9: Data Processing

Cihan H Dagli, PhD
Professor of Engineering Management and Systems Engineering
Professor of Electrical and Computer Engineering
Founder and Director of Systems Engineering Graduate Program

dagli@mst..edu

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY
Rolla, Missouri, U.S.A.

Lecture outline

- Project Status Presentations Schedule
- Generalized RBFN Example
- RBF Learning Approaches
- Data Preprocessing
  - Data Characterization
  - Data Cleaning
  - Normalization
  - Data Reduction

Project Status Review Presentations

April 24, 2018

Murat Aslan, Tatiana Cardona Sepulveda, Prince Codjoe, Xiongming Dai
Jeffrey Dierker, Venkata Sai Abhishek Dwivadula, Brian Guenther, Anthony Guertin
Timothy Guertin, Seth Kitchen

May 1, 2018

Gregory Leach, Yu Li, John Nganga, Igor Povarich, Jack Savage, William Symolon
Wayne Viers III, Tao Wang, Kari Ward, Julia White, Jun Xu

Project Status Presentation Template

- Project Title
- Description of engineering problem to be solved by Neural Network (one slide)
- What methods are currently being used? (one slide)
- What will be impact of problem solution success? (one slide)

Project Status Presentation Template

- Data structure to be used in solving the engineering problem (one slide)
- Input data representation and possible pre-processing (one slide)
- Out put data representation and possible post processing (one slide)
- Possible neural network architectures to be used and why they are selected? (one slide)

Project Status Presentation Template

- Schematic representation of the solution architecture proposed (one slide) (It is possible for you to use more than one neural network or collection of neural networks for solving your problem)
- Input and output data statistics (one slide)
  - Number of data points
  - Training data size
  - Testing data size
  - Validation data size
  - Type of data normalization

Project Status Presentation Template

- Progress you made so far (multiple slides)
- Problems that you are facing and how are you mitigate them? (one or two slides)

Regularization Theory and RBF

- When the stabilizer D is *translationally* and *rotationally* invariant, the Green's function becomes a radial-basis function.
- One example of a Green's function that corresponds to a *translationally* and *rotationally* invariant D is the multivariate Gaussian function.

$$G(\mathbf{x}, \mathbf{x}_i) = G(\|\mathbf{x} - \mathbf{x}_i\|)$$

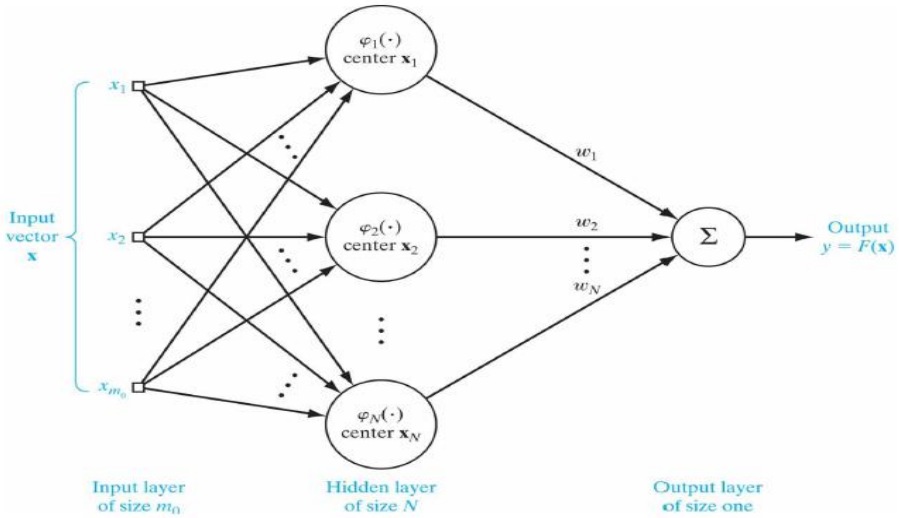
$$G(\mathbf{x}, \mathbf{x}_i) = exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x} - \mathbf{x}_i\|^2 \right)$$

- The regularized solution becomes

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x} - \mathbf{x}_i\|^2 \right)$$

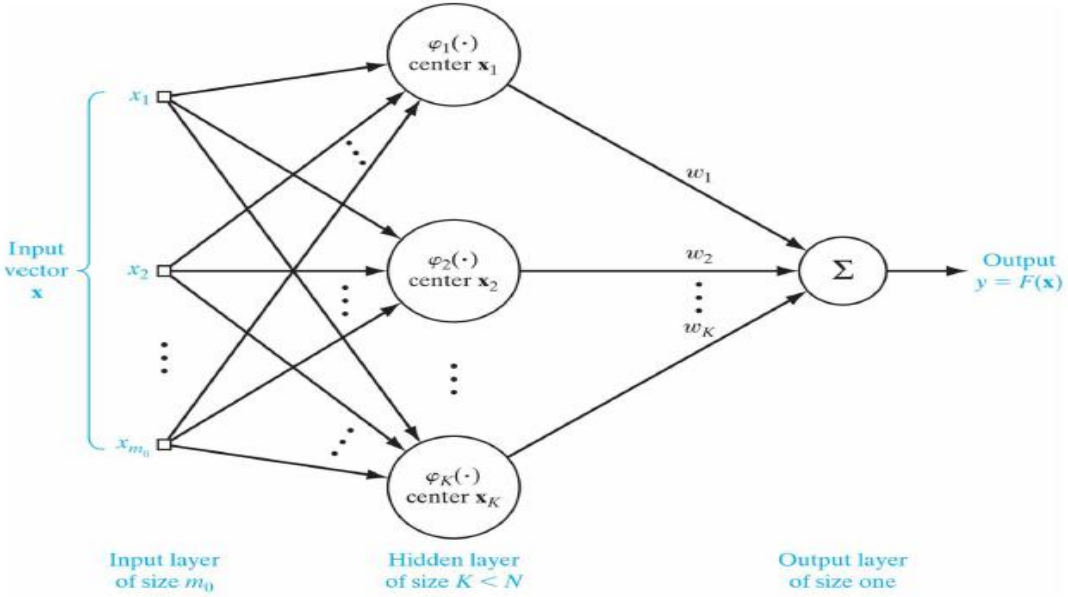
- This is a RBF network! It is known to be a *universal approximator*.

Regularized Solution as a Network



- Each hidden unit computes $G(x, x_i)$
- There is one hidden unit for each input pattern. N units for N inputs!
- Output is the linear weighted sum of the hidden unit activation potentials.

Generalized Radial-Basis Function Networks



$$F(x) = \sum_{i=1}^{m_1} w_i \phi_i(x), \text{ where } m_1 < N, \text{ and } \phi_i(x) = G(\|\mathbf{x} - \mathbf{t}_i\|)$$

$$F(x) = \sum_{i=1}^{m_1} w_i G(\|\mathbf{x} - \mathbf{t}_i\|)$$

Regularization Networks vs. Generalized RBF Networks

- Generalized RBFNs are computationally less expensive. The hidden layer is much smaller, $m_1 < N$.
- In regularization networks all inputs are used as centers. In GRBFNs weights and centers are both unknown.
- Matlab: newrbe is the function for regularization networks and newrb is the function for generalized RBFN.

Solving the XOR Problem using Generalized RBFN

- Construct an RBF pattern classier that produces,

  0 for inputs (1, 1) and (0, 0)

  1 for inputs (0, 1) and (1, 0)

- Define a pair of Gaussian hidden functions

$$\phi_1(x) = e^{\|x-t_1\|^2}, \; t_1 = [1,1]^T$$

$$\phi_2(x) = e^{\|x-t_2\|^2}, \; t_2 = [0,0]^T$$

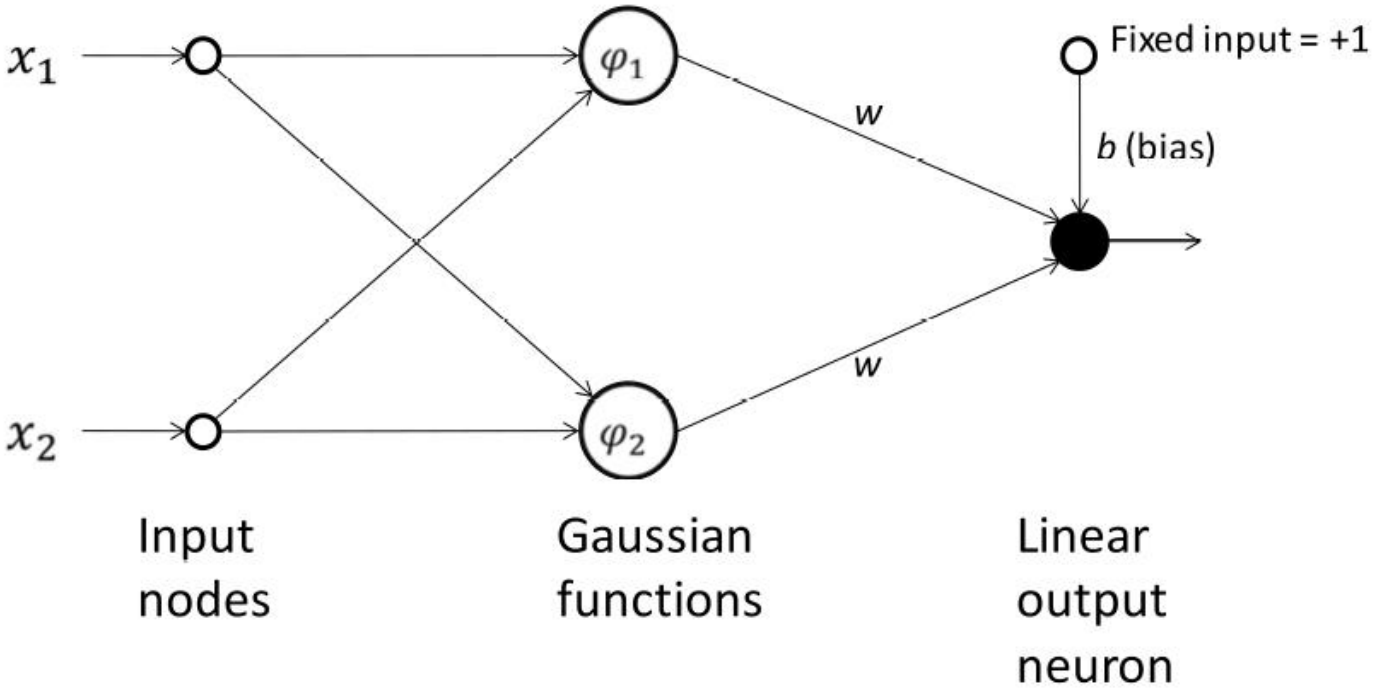
Finding the Hidden Functions for the Inputs

TABLE 5.1 Specification of the Hidden Functions for the XOR Problem of Example 1

Input Pattern $\mathbf{x}$	First Hidden Function $\varphi_1(\mathbf{x})$	Second Hidden Function $\varphi_2(\mathbf{x})$
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(0,0)	0.1353	1
(1,0)	0.3678	0.3678

XOR Problem (Contd.)

Using the following network to solve the XOR problem,



XOR Problem (Contd.)

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1353 & 1 \\ 0.3678 & 0.3678 & 1 \\ 0.1353 & 1 & 1 \\ 0.3678 & 0.3678 & 1 \end{bmatrix}$$

$$\mathbf{d} = \begin{bmatrix} 0 & 1 & 0 & 1 \end{bmatrix}^T$$

$$\mathbf{w} = \begin{bmatrix} w & w & b \end{bmatrix}^T$$

The interpolation matrix is rectangular, so we use the pseudo inverse to find the weights,

$$\begin{aligned} \mathbf{w} &= \mathbf{G}^{+}\mathbf{d} \\ &= (\mathbf{G}^{\mathbf{T}}\mathbf{G})^{-1}\mathbf{G}^{\mathbf{T}}\mathbf{d} \end{aligned}$$

XOR Problem (Contd.)

Solving for $\mathbf{G}^+$ and $\mathbf{w}$

$$\mathbf{G}^+ = \begin{bmatrix} 1.8292 & -1.2509 & 0.6727 & -1.2509 \\ 0.6727 & -1.2509 & 1.8292 & -1.2509 \\ -0.9202 & 1.4202 & -0.9202 & 1.4202 \end{bmatrix}$$

$$\mathbf{w} = \begin{bmatrix} -2.5018 \\ -2.5018 \\ +2.8404 \end{bmatrix}$$

The weights completely specify the RBF network.

Hybrid Learning Algorithm for RBFN

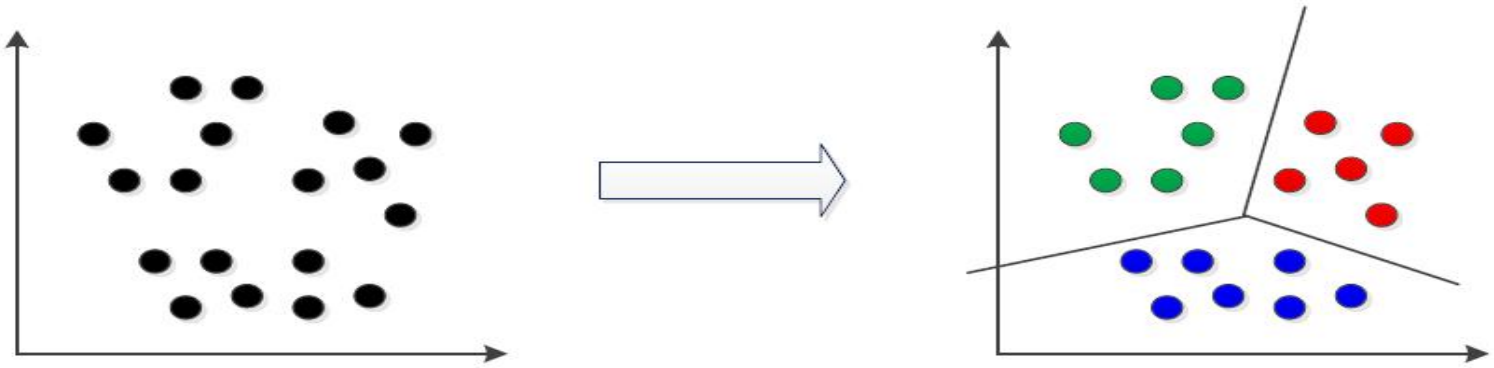
- Unsupervised learning stage for computing the centers
  - Use cluster analysis to group unlabeled data samples based on similarity
  - Clustering algorithm: K-means clustering
- Supervised learning stage for finding the weights of the RBFN
- Least-Mean Squares (LMS) and Recursive Least Squares algorithm (RLS)

RBN Example

Half-moon classification problem using HalfmoonData.mat file
Inputs: Rows 1 and 2
Targets: Row 3
Number of data samples: 2000
Use Matlab NN toolbox to implement RBF neural network architecture

K-Means Clustering

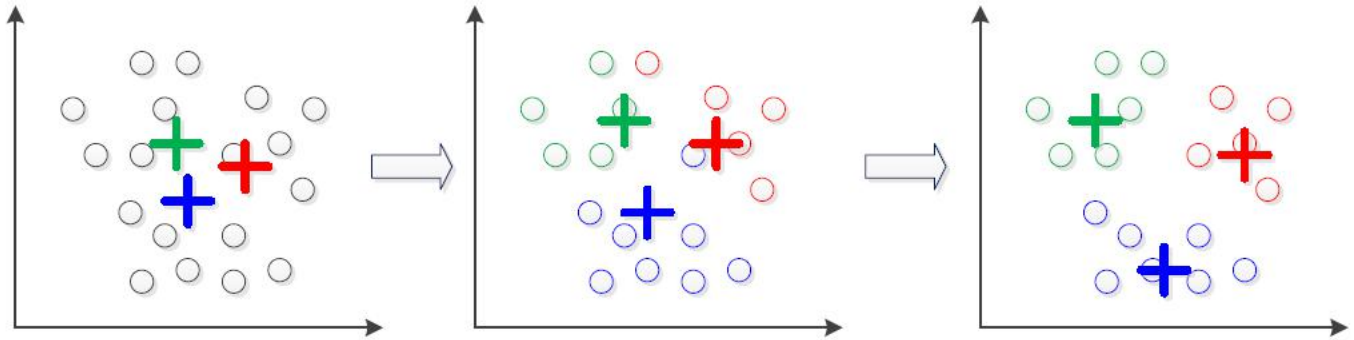
- Partitional clustering technique: attempts to separate data into its `natural' groupings
- Data is separated into a user-specified number of clusters (K)
- Similarity is determined based on some statistical measure
- Clusters are represented by their centroids



K-Means Clustering

- Clusters are defined based on a centroid
- Centroid is the mean of a group of points
- Basic K-means algorithm
  - Step 1: Select K points as initial centroids
  - Step 2: Assign each input to its closest centroid
  - Step 3: Recalculate the centroids
  - Step 4: Repeat step 2 and 3 until centroids no longer change

Illustration of K-means Clustering



- Centroid: Mean
- Proximity function: Euclidean distance
- Instead of strict convergence sometimes a weaker condition is used
- Algorithm is terminated when only 1% of points change clusters

Computing Proximity

- Sum of the squared error is used as the cost function for determining the quality of clustering

$$SSE = \sum_{K}^{i=1} \sum_{x \in C_i} \|x - c_i\|^2$$

where x is a data sample, C_i is the i-th cluster, c_i is the centroid of cluster C_i and K is the number of clusters

- The centroid (mean) of a cluster with m_i elements is given by,

$$c_i = \frac{1}{m_i} \sum_{x \in C_i} x$$

- The initial centers are selected randomly from the data.
- The spreads of the RBFs are set by calculating the variance of each selected center or by using,

$$\sigma = \frac{d_{max}}{\sqrt{2K}}$$

Least Mean Squares Algorithm

For a data sample x and centers c,

Step 1: Calculate the output of the RBF

$$y = F(x) = \sum_{i=1}^{m_1} w_i \phi(\|\mathbf{x} - c_i\|)$$

Step 2: Find the error in the network response

$$e = d - y$$

Step 3: Update the weights

$$w(n+1) = w(n) + \eta e \phi$$

Recursive Least Squares Algorithm

Step 1: To initialize the algorithm set the weights $w(0) = 0$ and the inverse of the correlation matrix $P(n) = \lambda^{-1} I$. λ is small constant and $I_{j \times j}$ is an identity matrix of size j, the number of hidden neurons.

Step 2: Find $g(n)$

$$g(n) = P(n)\phi(n)$$

Step 3: Find $\alpha(n) = d(n) - w_T(n-1)\phi(n)$

Step 4: Update the weights,

$$w(n) = w(n-1) + g(n)\alpha(n)$$

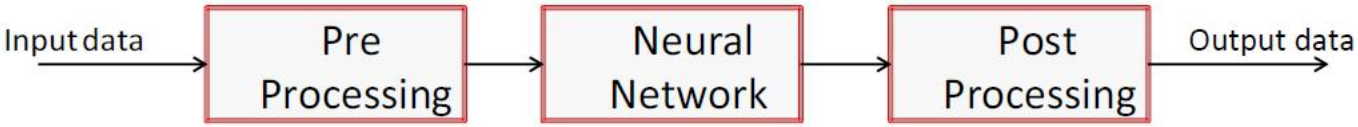
Hybrid Learning Procedure

- Initialize network parameters
- Find centers using an unsupervised learning approach
- Calculate the spreads using the distance measure or the variance of the clusters
- Train the weights using a supervised learning approach (LMS or RLS)
- Test the network

The Need for Preprocessing Data

- Real world data is incomplete, noisy and inconsistent.
- Data quality significantly impacts performance of a classier.
- NN's can perform any non-linear mapping between sets of unprocessed inputs and targets
  - leads to poor results in practice
- For most practical problems preprocessing and post-processing of the output is very essential
- Preprocessing may involve simple linear transformations like scaling or normalization
- More complex preprocessing may include dimensionality reduction methods.
  - Principal Component Analysis, Wavelet transforms
- Additional preprocessing includes incorporation of prior knowledge such as via regularization.
  - Information that can be used to develop a solution in addition to that provided by the training data

Preprocessing and Post-processing Stages



- Majority of time is devoted to preprocessing and relatively less to
- classier development.
- Preprocessing steps are applied to the entire dataset at once.
- With applications involving online learning, each new data point must be preprocessed before being passed on to the network.
- Post-processing may be required to convert network outputs into the target data format.

Data Preprocessing Techniques

- Data Cleaning: fill in missing values, smoothing noisy data, identifying and removing outliers, resolving inconsistencies
- Data Transformation: normalization, aggregation
- Data Reduction
  - attribute subset selection
  - dimensionality reduction
  - numerosity reduction

Data Characterization

- Successful preprocessing depends on a good overall understanding of data characteristics.
- It is important to identify properties of data and highlight any outliers.
- Measures of the data's central tendencies
  - Mean, median, mode, midrange
- Measures of data dispersion
  - Quartiles, Inter-Quartile Range (IQR), variance

Measuring Central Tendencies

Most common measure of the center is the mean,

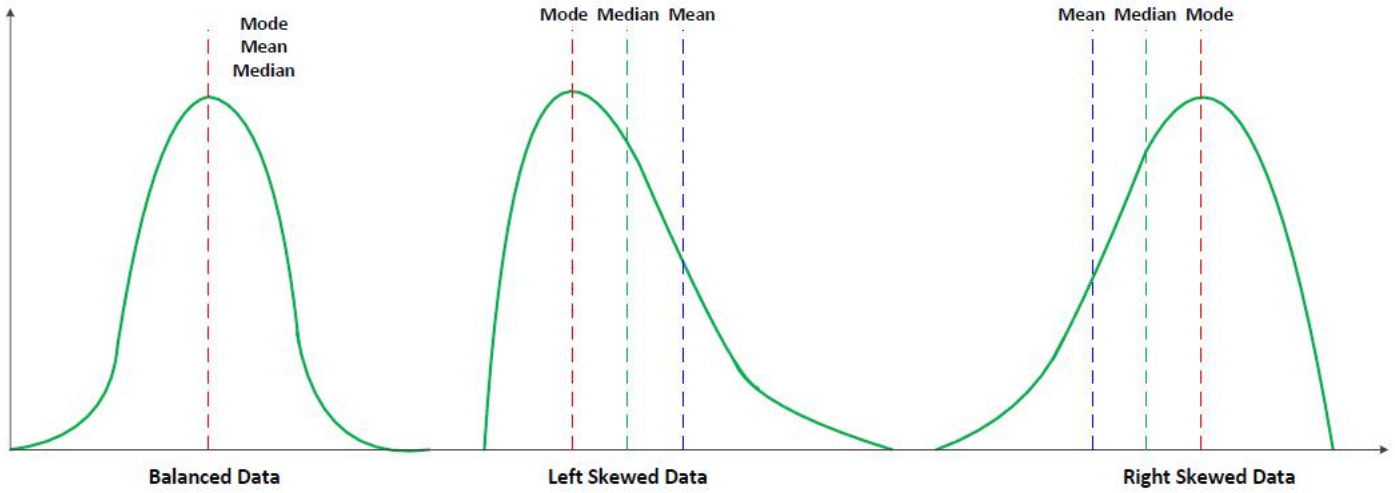
$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \ldots + x_N}{N}$$

or the weighted mean,

$$\bar{x} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i} = \frac{w_1 x_1 + w_2 x_2 + \ldots + w_N x_N}{w_1 + w_2 + \ldots + w_N}$$

Measuring Central Tendencies

- For skewed data, the median is a better measure of the center
- For an odd numbered set of data, the median is the middle value.
- For an even set, median is the average of the 2 middle values



Measures of Dispersion

- Range: difference between the largest (max) and smallest (min) values.
- kth percentile is that value of x_i (data point) such that k percent of the data is at or below x_i.
  - data are sorted in increasing order
  - e.g. the median is the 50th percentile.
- Quartiles are the most commonly used percentiles
  - Q1: 25th percentile is the first quartile
  - Q3: 75th percentile, third quartile
  - Interquartile range: IQR = Q3 - Q1 is used to identify suspected outliers
- Values that lie (1.5 X IQR) above Q3 or (1.5 X IQR) below Q1 are suspected outliers.
- Complete information about a distribution is given by the five number summary: Minimum;Q1;Median;Q3;Maximum

Data Variance

Variance of N observations:

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2 = \frac{1}{N}\left[\sum x_i^2 - \frac{1}{N}\left(\sum x_i\right)^2\right]$$

σ is the standard deviation, $\bar{x}$ is the mean

- $\boldsymbol{\sigma}$ measures spread about the mean and should be used only when mean is chosen as a measure of center.

Data Cleaning

- Handling missing values
  - Ignore the data sample
    - Not very useful unless we have a large amount of data
  - Fill in missing values manually
    - Time consuming, may not be possible to find data values
  - Use the mean to fill in the missing values
  - Use predictive methods such as regression to fill in the values

Noisy Data

- Binning: Divide data into equal sized bins or buckets and replace all values by the mean

$$\underbrace{4, 8, 15}, \underbrace{21, 21, 24}, \underbrace{25, 28, 39}$$

$$\Downarrow$$

$$\underbrace{9, 9, 9}, \underbrace{22, 22, 22}, \underbrace{29, 29, 29}$$

- Regression: Fit the data to a function; use one attribute to predict the other
- Clustering: Organize data into groups based on some similarity measure. Values that don't t into any clusters can be treated as outliers.

Removing Redundancies In Data

Attributes are considered redundant if they can be derived from other attributes.

- Use correlation analysis to detect redundancies
- Measure how strongly one attribute implies the other

For numerical attributes, we can compute the correlation coefficient:

$$r_{A,B} = \frac{\sum_{i=1}^{N}\left(a_i - \bar{A}\right)\left(b_i - \bar{B}\right)}{N\sigma_A\sigma_B}, \quad -1 \leq r_{A,B} \leq 1$$

$r_{A,B} > 0$ denotes positive correlation

$r_{A,B} < 0$ denotes negative correlation

$r_{A,B} = 0$ independent, no correlation

Chi-Square Test

- For categorized data (discrete data) correlation can be determined using the χ^2 test.
- Consider two categorical inputs A and B,
- Input variable A $\{a_1 a_2 \ldots\ldots a_c\}$
- Input variable B $\{b_1 b_2 \ldots\ldots b_r\}$
- The χ^2 statistic tests for independence of variables A and B.

Chi-Square Test: Contingency Table

Create a contingency table where the χ^2 value for each pair of (A_i, B_j) values is given by:

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{\left(o_{ij} - e_{ij}\right)^2}{e_{ij}}$$

o_{ij} = Observed frequency of the joint event (A_i, B_j)

e_{ij} = Expected frequency of (A_i, B_j)

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{N}$$

N is the number of data samples.

$\text{count}(A = a_i)$ is the number of data samples having the value a_i for A.

$\text{count}(B = b_j)$ is the number of data samples having the value b_j for B.

Chi-Square Test Example

Problem: Are gender and reading preferences correlated?

	male	female	total
fiction	250(90)	200(360)	450
non-fiction	50(210)	1000(840)	1050
total	300	1200	1500

The expected frequencies (shown in the table in parentheses) are computed by,

$$e_{11} = \frac{count(male) \times count(fiction)}{N} = \frac{300 \times 450}{1500} = 90$$

$$e_{12} = \frac{count(female) \times count(fiction)}{N} = \frac{1200 \times 450}{1500} = 360$$

$$e_{21} = \frac{count(male) \times count(nonfiction)}{N} = \frac{300 \times 1050}{1500} = 210$$

$$e_{22} = \frac{count(female) \times count(nonfiction)}{N} = \frac{1200 \times 1050}{1500} = 840$$

Chi-Square Test Example

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-350)^2}{360} + \frac{(1000-840)^2}{840}$$
$$= 507.93$$

$$\begin{aligned}\text{degrees of freedom} &= (r-1) \times (c-1) \\ &= (2-1) \times (2-1) \\ &= 1\end{aligned}$$

For a degree of freedom of 1, the χ^2 value needed to reject the hypothesis that gender and reading preference are independent is 10.828. We can conclude that gender and reading preference are strongly correlated.

Example from reference 1

Data Normalization

Normalization is the process of scaling data to t into a small range such as -1 to 1 or 0 to 1.

- Useful when inputs have widely varying typical values
  - e.g. Temperature versus pressure in a plant
  - 100s of deg F, 1000s of psi
- By normalizing the inputs and outputs, we ensure that all weights are also of the order of unity. This allows us to initialize weights to small values and improve convergence speed.
- For radial-basis function networks scaling is even more essential since the activation depends on the Euclidean distance between the inputs and the centers.

$$distance^2 = \| x - x_j \|^2 = \sum_{i=1}^{M_0} \left(x_i - x_{ji} \right)^2$$

- If one of the variables is much smaller than another, it will not contribute much to the Euclidean distance

Normalization Methods

Types of normalization include,

- min-max normalization
- Z-score normalization
- Normalization by decimal scale

Min-Max Normalization

- min_A - minimum value of an attribute
- max_A - maximum value of an attribute

To map the value of x to a new value x_n in the range $[newmin_A, newmax_A]$

$$x_n = \frac{x - min_A}{max_A - min_A}\left(newmax_A - newmin_A\right) + newmin_A$$

Example: Consider income in the range \$15,000 to \$95,000.

$$x = 45000, \text{ map to } [0, 1.0]$$

$$x_n = \frac{45000 - 15000}{95000 - 15000}\left(1.0 - 0\right)$$
$$= \frac{30000}{80000} = 3.7$$

The normalized value of an income of \$45,000 is 3.7.

Z-Score Normalization

- Also called zero-mean normalization.
- Values are normalized based on mean and standard deviations.
- A value x of a set X is normalized by:

$$x_n = \frac{x - \bar{x}}{\sigma_x}$$

- Example: Suppose the mean and standard deviation for the income range are $50000 and $12000. The normalized value of an income of $45,000 is,

$$x_n = \frac{45000 - 50\bar{0}00}{12000} = -0.4167$$

Normalization by Decimal Scale

- Normalize by moving the decimal points in the data sample.
- $x_n = \frac{x}{10^j}$
- where *j* is the smallest integer such that $\max(|x_n|) < 1$.
- Example: Range of X is -98 to 95. The largest absolute value is 98.

To normalize by decimal scaling, we must select *j* such that,

$$\frac{98}{j} < 1$$

Thus, j = 100. The entire range of A is normalized to -0.98 to 0.95.

Normalizing discrete data

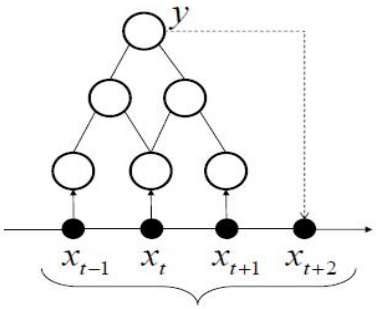
- Discrete data is of two main types:

1. ordinal Numerical variables have a natural ordering (e.g. age)
2. categorical No natural ordering (shapes, colors)

- Encoding these variables using numbers can impose an artificial ordering on them (0, 1, 2, 3, …) etc.
- One possible way around this is to use 1 - of - c encoding similar to the encoding scheme used for neural network outputs.
  - red - 1 0 0
  - blue - 0 1 0
  - green - 0 0 1

Working with Time Dependent Data

- Sometimes the input data is time- dependent $x = x(t), i.e. x$ varies as a function of time t.
- For such problems the objective is to predict the value of x sometime into the future.
- The input to the network should consist of a series of discrete values x_{t-1}, x_t, x_{t+1}, sampled at different times.
- A moving window of input values can be chosen by the user. For every p numbers of inputs, the $(p + 1)th$ number is the target.
- If the data exhibits other trends besides time-dependence, it should be detrended to avoid poor performance.



Dimensionality Reduction

- Encode or transform data to obtain a reduced or compressed representation of the original data.
  - Lossless reduction: The original data can be reconstructed from the compressed data with loss of information
  - Lossy reduction: Information loss occurs during transformation and original data cannot be recovered

Attribute Subset Selection

Attribute subset selection methods:

- The goal is to find a minimum set of attributes such that the resulting probability distribution of data classes is as close as possible to the original distribution of data classes obtained using all attributes
- For n attributes, there are 2^n possible subsets
- Stepwise forward selection: Select the number of attributes to include in the reduced subset. Start with an empty subset. At the first step add the best of the original attributes. At each subsequent step the best of the remaining attributes is added until the required number of attributes have been selected.
- Stepwise backward elimination: Start with a full set of attributes and eat step eliminate the worst attribute. Continue until the required number of attributes remain in the subset.

Best and worst attributes are determined using statistical significance tests.

Exhaustive Search Procedure

Consider a dataset x_i with d features, $i = 1 \ldots d$.

Let the eliminated features be denoted x_{zi} where $zi = 1 \ldots M$ for a total of M discarded features.

We can create a search tree:

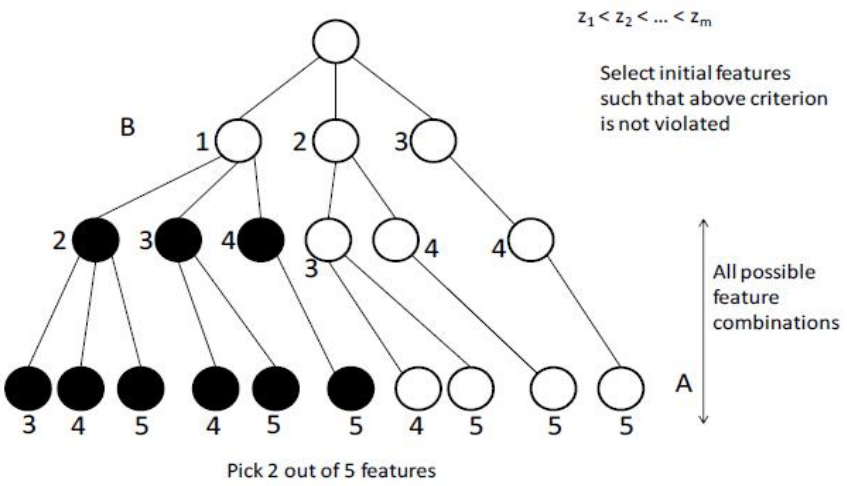


Image adapted from reference 4

Feature Reduction

- Feature reduction is the process of reducing the size of the original dataset by transforming it into an alternative representation.
- The reduced representation retains the essential features of the original data.
- Feature reduction requires two main components:
  - A criterion to judge whether one subset of inputs is superior to another
    - Sum of squared error or classification error
- A procedure to search for the optimal subset
- Two common methods for feature reduction are *discrete wavelet transforms* and *principal component analysis*.

Feature Reduction Criteria

- Ideally, we would train a network with all possible feature subsets and evaluate its performance on an independent dataset
  - Computationally very expensive
- In practice, simple linear mappings are used to select the features
- For function approximation problems, we can use a simpler linear model consisting of a single layer network with linear outputs. Decision is made based on the sum of squares error.
- For pattern classification problems, the selection criterion is the misclassification rate. Features are retained as long as misclassification rate reduces.

Wavelet Transforms

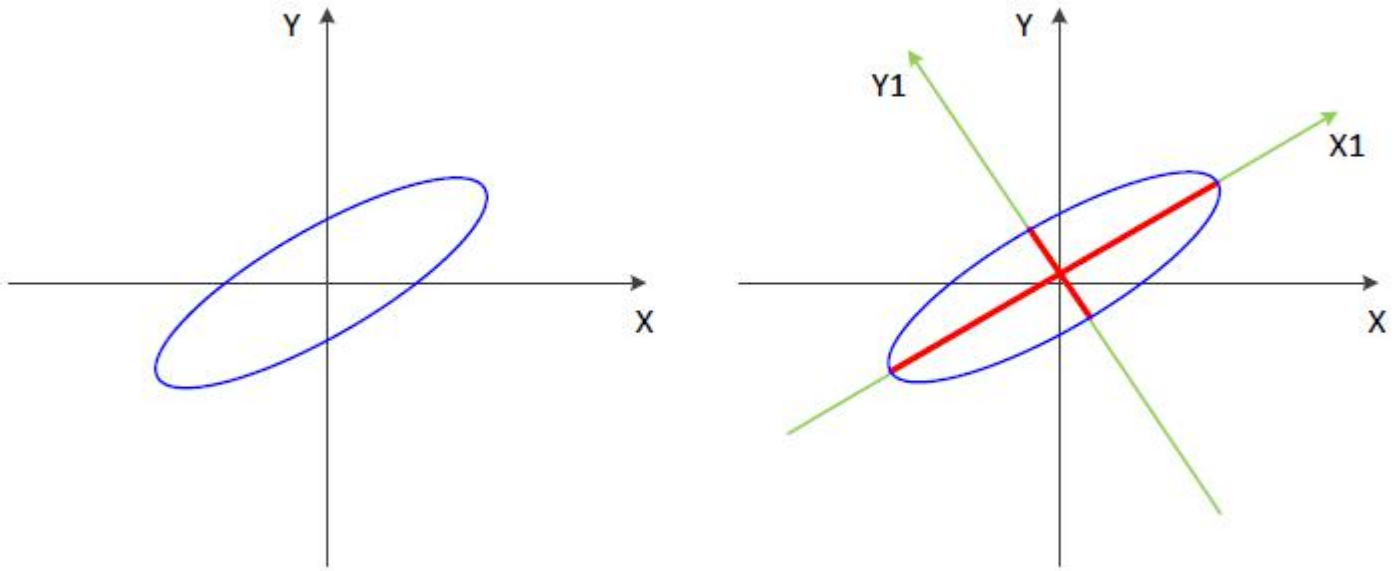
- Discrete wavelet transforms (DWT) are used to transform data into wavelet coefficients
- Transformed data vector is the same length as original data vector
- Only the strongest coefficients are retained; others are set to 0
  - Resulting data is sparse
- Can also help in smoothing out noise without loss of data features
- Original data can be reconstructed by applying the inverse of the DWT
- Types of DWT
  - Haar-2 transform
  - Daubechies-4 transform
  - Daubechies-6 transform

Procedure for Applying DWT

General procedure for applying DWT:

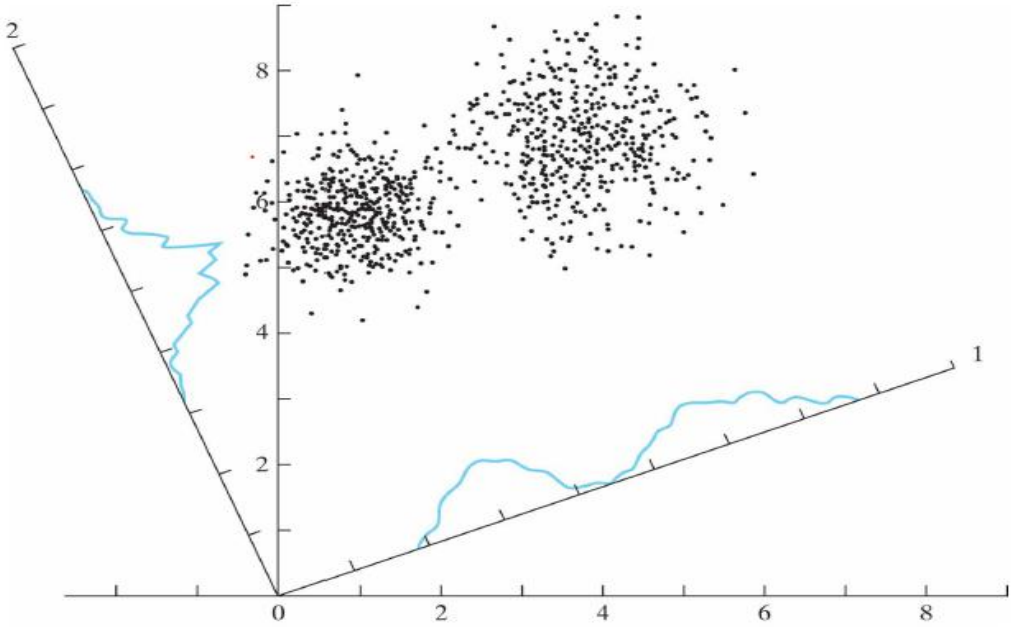
- Length of input vector L must be an integer power of 2.
  - Pad the data vector with zeroes as necessary
- Find the sum or weighted average of the data vector to smooth out the data
- Perform a weighted difference to bring out the detailed features of the data
- The above transformations are applied to pairs of data points in X. Two sets of data of length L/2 are generated - one being the smoothed or low frequency version, and the other is the high frequency version of it.
- The two functions are recursively applied to each set of reduced data until the final datasets have length 2
- Selected values from the datasets obtained in the above iterations are designated the wavelet coefficients of the transformed data

Principal Components Illustration



The X1-Y1 axis are orthogonal basis vectors that capture the variation in the ellipse.

Principal Components Illustration Showing Data Variation



The variation in the data can be seen in the projection along the new basis vectors.

Principal Component Analysis

- Consider an input vector $X\ with\ m$ attributes.
- The objective of PCA is to find $l, m\ dimensional$ orthogonal vectors that can represent the variation in the data $where\ l \leq m.$
- The original data is projected onto a smaller space resulting in reduction of input dimensionality.
- Using PCA, we can find an orthogonal transformation matrix, Q which can transform the original input vector into a reduced dimensionality input vector.

$$X_{reduced} = QX$$

Finding the Orthogonal Transformation Matrix

- Let X denote a set of m-dimensional input vectors $\{x_1, x_2, \ldots, x_m\}$.
- X should have zero-mean. If X has non-zero mean, subtract the mean of X from X.
- Let a be a set of projections $\{a_j | j = 1, 2, \ldots, m\}$
  $a = [a_1, a_2, \ldots, a_m]^T$
- The projection of the input data A is given by the unit vector q,

$$A = X^T q = q^T X$$

Finding the Orthogonal Transformation Matrix (Contd.)

The variance of A,

$$\begin{aligned}\sigma^2 &= E[A^2] \\ &= E[(q^T X)(X^T q)] \\ &= q^T E[X X^T] q \\ &= q^T R q\end{aligned}$$

R is an $m \times m$ correlation matrix of X.

R is a symmetric matrix, thus $R^T = R$,

$$a^T R b = b^T R a$$

here a and b are $m \times 1$ vectors.

Finding the Orthogonal Transformation Matrix (Contd.)

The variance σ^2 of the projection A is a function of q,

$$\psi(q) = \sigma^2 = q^T R q$$

$\psi(q)$ can be considered a variance probe which we want to maximize,

$$(\delta q)^T (Rq - \lambda q) = 0$$

or,

$$Rq = \lambda q \tag{1}$$

Equation (1) is the eigenvalue problem and λ are the eigenvalues of the correlation matrix R. The values of q associated with λ are the eigenvectors.

Finding the Orthogonal Transformation Matrix (Contd.)

- Using $(R - \lambda I)q = 0$ we can compute the eigenvalues λ of R.
- For each λ we also compute the corresponding eigenvector q.
- The eigenvectors $Q = [q_1, q_2, \ldots, q_m]$ represent the set of orthogonal vectors or principal directions of the projection.
- The associated eigenvalues define the contribution of each eigenvector to the overall variation in the data.
- Total variance of the projected data is $\sum_{j=l+1}^{m} \lambda_j$.
- The percentage of variation unaccounted by the principal components is $\sum_{j=1}^{l} \lambda_j \Big/ \sum_{j=1}^{l} \lambda_j$

MATLAB EXAMPLE: PCA Implementation

PCA Summary

- Input data are normalized to have zero mean
- Compute the principal component vectors (eigenvectors) using the eigenvalue equation.
- Sort the principal components in order or decreasing significance (variance).
- The first component captures the most variance, the second component the next highest and so on.
- The size of the dataset is reduced by eliminating the weaker component vectors that show the least variance.
- The final reduced dataset consists of the strongest principal components.

Final Project Report and Exam

- Final project report is due on May 4, 2018 at 4:00 PM US CDT
- Final Exam will be posted on blackboard on May 4, 2018 at 4:00 PM US CDT and will be due on May 7, 2018 at 11:59 PM your local time.