A study of Statistical Learning on DNA datasets

By (Group 1)
Jing Gao, Prasenjit Kapat, Dept. of Stat.
Manli Zhu, Dept. of ECE.
Ruoming Jin , Dept. of CSE.

Advisor: Prof. Goel and Prof. Verducci

The Ohio State University

The Splice-Recognition Problem

- A little Biology
  - Exons
    - One of the parts of a gene whose sequence is present in the mature mRNA. (The parts of the DNA sequence retained after splicing)
  - Introns
    - A part of the gene which is transcribed but which is removed (by the process of splicing) from the primary transcript in the formation of mRNA. Introns therefore remain in the nucleus.
  - Splice junctions
    - Points on a DNA sequences at which “superfluous” DNA is removed during the process of protein creation in higher organisms.
    - EI sites (donors) /IE sites (acceptors)
- Splice-junction problem
  - To recognize, given a sequence of DNA, the boundaries between exons and Introns

The Splice Dataset

- DNA Sequences
  - Nucleotide: A,C,T,G.
- What's in the dataset?
  - A,T,C,A,A,T,A,A,G,C,T,C,C,T,A,G,T,C,C,A,G,A,C,G,C,C,A,T,G,G,G, T,C,A,T,T,T,C,A,C,A,G,A,G,G,A,G,G,A,C,A,A,G,G,C,T,A,C,T,A, EI
  - 3190 examples, length 60.
  - One class attribute:
    - IE (25%) , EI (25%), Neither (50%)
  - No missing data
  - The boundary sit (splice junction) is between position 30 and 31.
  - UCI machine learning website
- The problem:
  - How to build a classifier to accurately predicate which group (IE, EI, or Neither) a new DNA sequence should belong to?

Previous Study on this Dataset

- AI group in University of Wisconsin
- The goal
  - Applying domain-theory to help the inductive process
  - KBANN (Knowledge based Artificial Neural Networks)
- Results
  - When the number of training examples is small (less than 200), KBANN performs well
  - When the number of training examples is beyond 500 or 1000, even a randomly-weighted network performs as well as KBANN

The Challenges

- Multi-dimension continuous data/ sequence Data
  - How to transform to multi-dimensional space?
- How to perform feature selection?
  - What kind of feature can help the classification?
- What kind of classifier can perform well?

Our approach

- Feature Transformation and Selection
  - Method 1: each position maps to one dimension (60 * 4)
  - Method 2: three consecutive position combines and maps to one dimension (58 * 64)
- Three Classifiers
  - K Nearest Neighbor (K-NN): Manli
  - Artificial Neural Networks (ANN): Prasenjit
  - Decision Tree (C4.5): Jing

K-Nearest Neighbor

- Why K-NN?
  - Easy to understand and implement.
  - Naïve classifier may get as good performance as a much complicated one.

K-Nearest Neighbor (cont.)

- Distance measurement

  Numerical Vector:

1.3	3	2.5	0	….	….	8.7	7	1

DNA sequence:

A	C	T	G	….	….	A	T	G

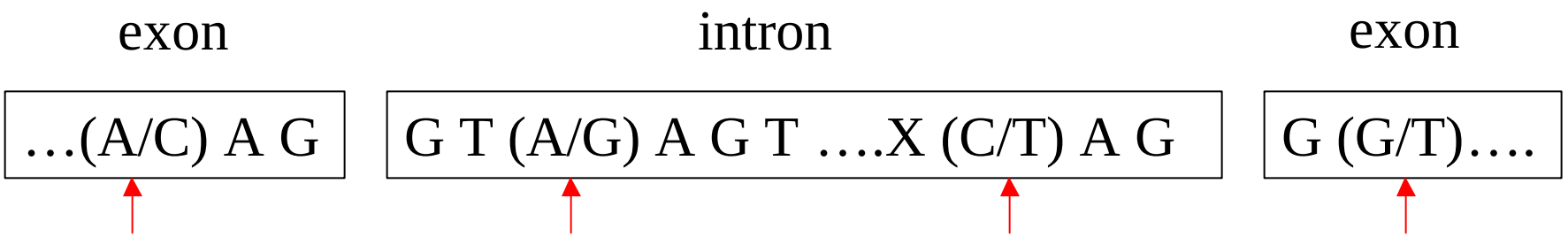
K-Nearest Neighbor (cont.)

- Defined Distance Measurement:

Sample 1	A	C	T	G	A	C	T	G	…	A	C	T	G	A	C	T
Sample 2	A	G	T	A	A	T	T	C	…	T	C	A	G	C	C	A
D1	0	1	0	1	0	1	0	0	…	1	0	1	0	1	0	1

K-Nearest Neighbor (cont.)

- Domain knowledge:



So, exact match?

K-Nearest Neighbor (cont.)

- New Distance Measurement:

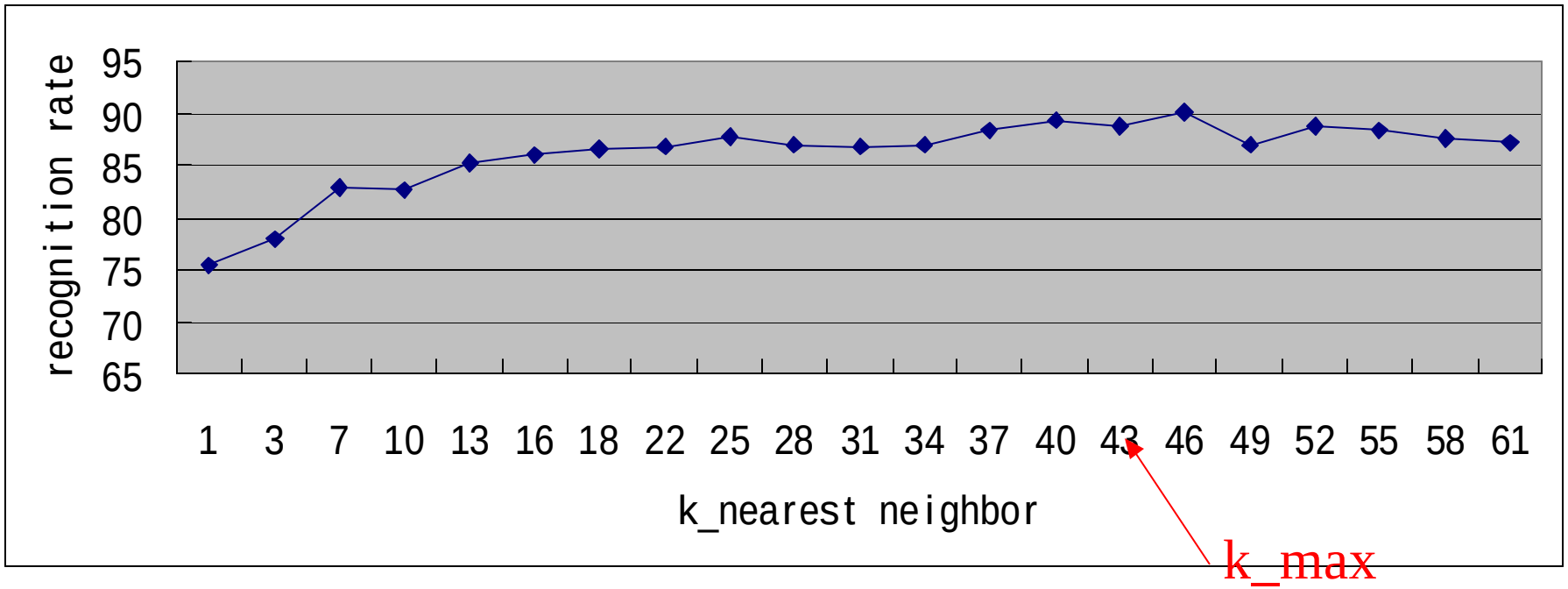
K-Nearest Neighbor (cont.)

- How to determine k ?

k ? , bias ? , variance ? , over-fitting

k ? , bias ? , variance ? , away from true boundary

K-Nearest Neighbor (cont.)

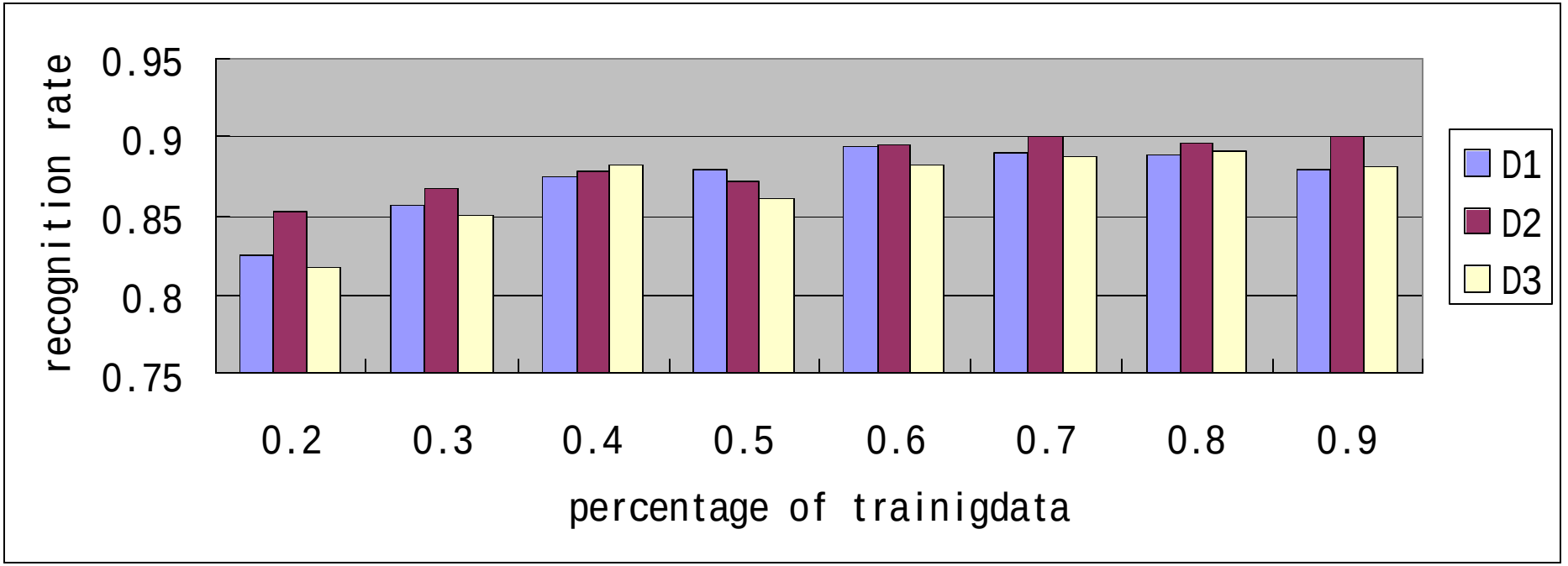


$$R_{optimal} = \frac{k_{\max}}{0.5n}$$

$$k_{optimal} = n_{training} R_{optimal}$$

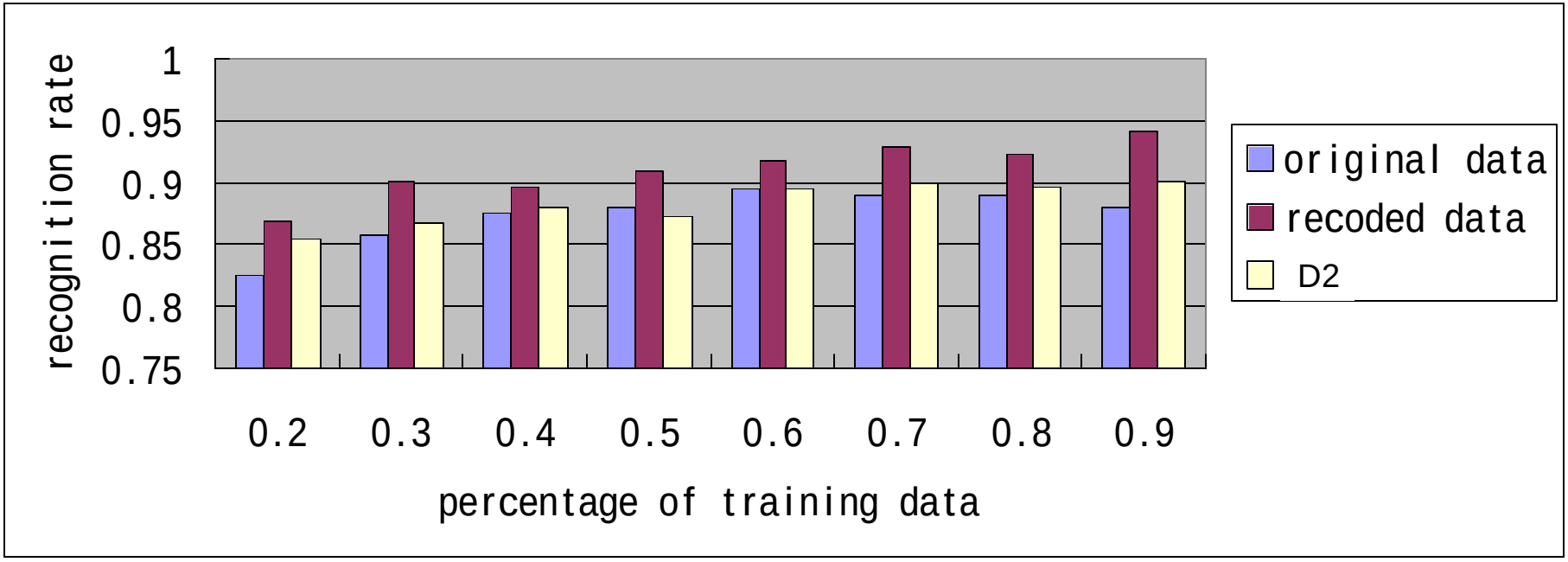
K-Nearest Neighbor (cont.)

- Experimental Result

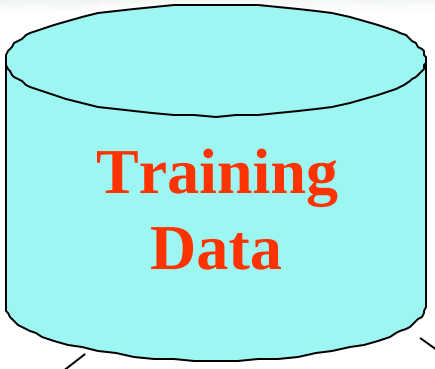


K-Nearest Neighbor (cont.)

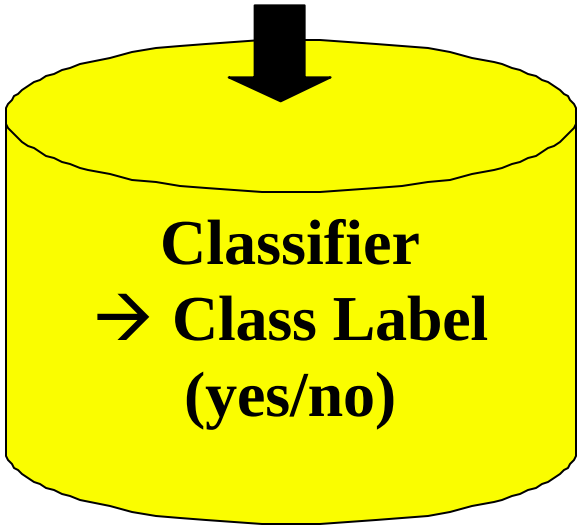
- Experimental Result



Classification Model

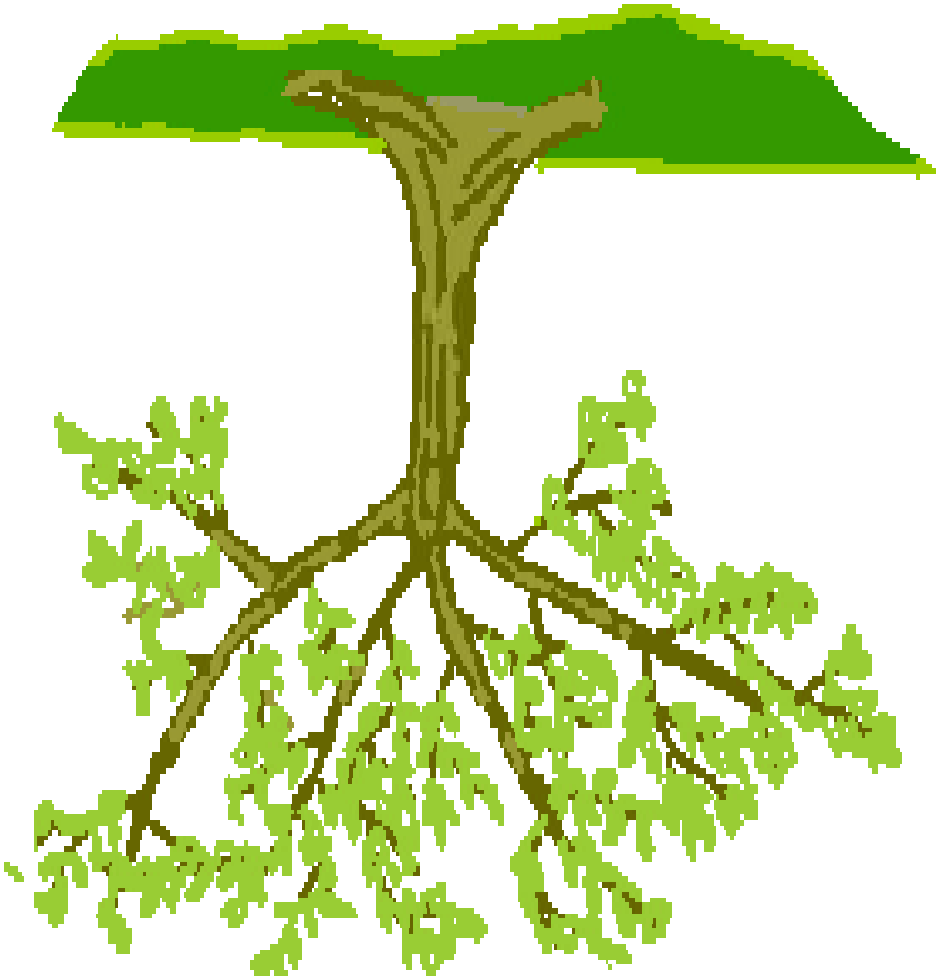


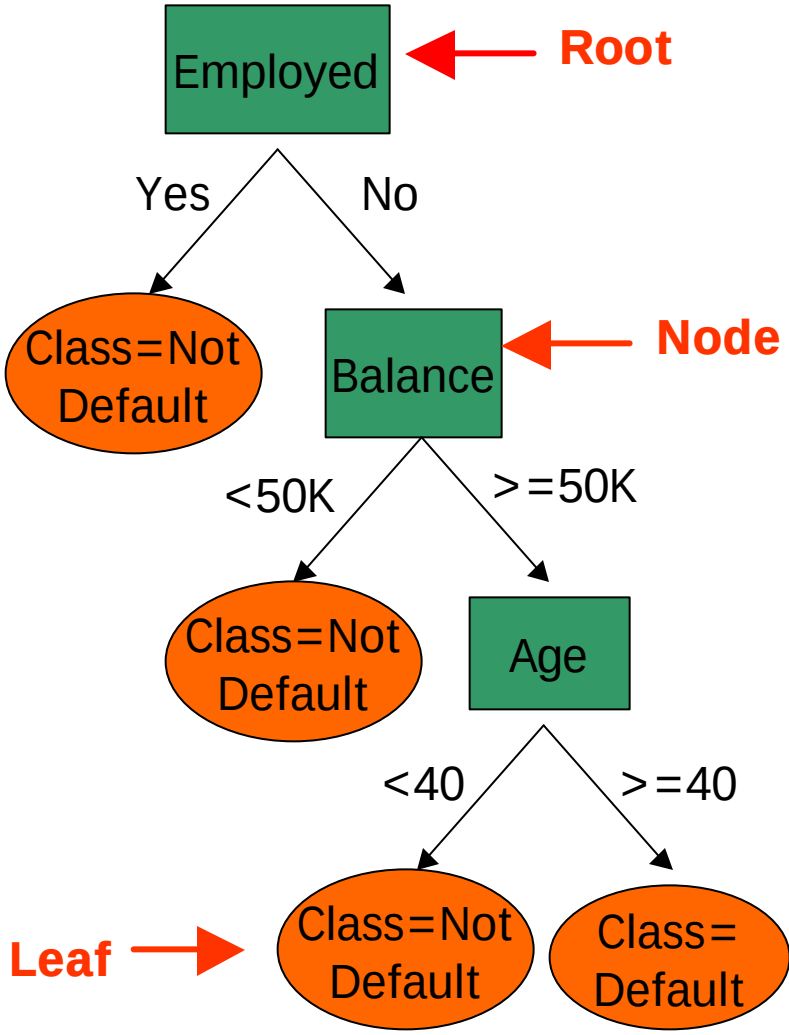
Classification Algorithms (ID3 or C4.5)



NAME	Balance	Age	Default
Brain	32,900	32	yes
Lisa	51,980	39	yes
Bill	73,000	57	no
Kiddo	68,000	50	no
David	32,000	41	yes
Amy	110,000	49	no

Classification Decision Tree (Representation: A Binary Tree (upside down)

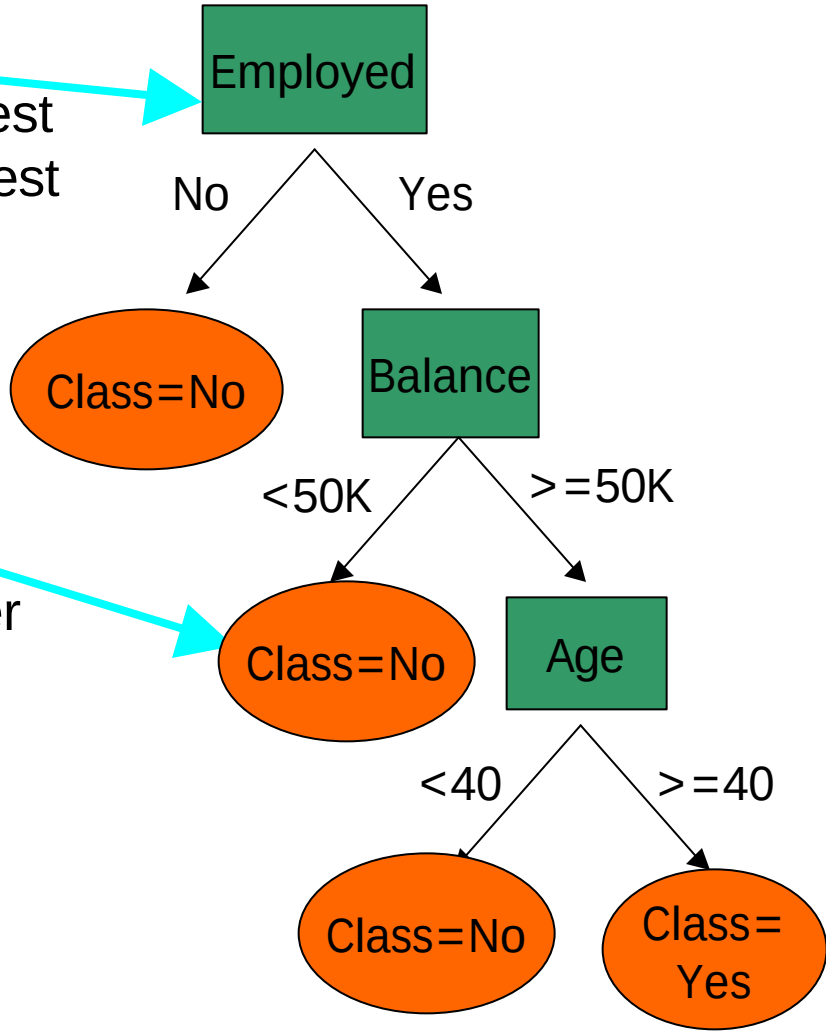




Classification Decision Tree Representation (C4.5): Divide and Conquer

A series of nested tests:

- Each **node** represents a test on only one attribute:
  - Tests on nominal attribute: Subset test
  - Numeric attributes are discretized: Best Split point
- **Leaves**
  - A class assignment (EI/IE/Neither)
  - Also provide a distribution over all possible classes
- Over-fitting
  - Each leaf should have a fairly number of instances
  - Pruning the nodes which have little instances



C4.5 Decision Tree Classifier

- Developed by Quinlan
- The current version is R8, and running in UNIX environment
- Download from the author's website
  - http://www.cse.unsw.edu.au/quinlan
- What's new features in C4.5
  - Avoid over-fitting the data by determining how deeply to grow a decision tree
  - Handling both continuous and discrete attributes
  - Handling missing data
  - Improving computational efficiency

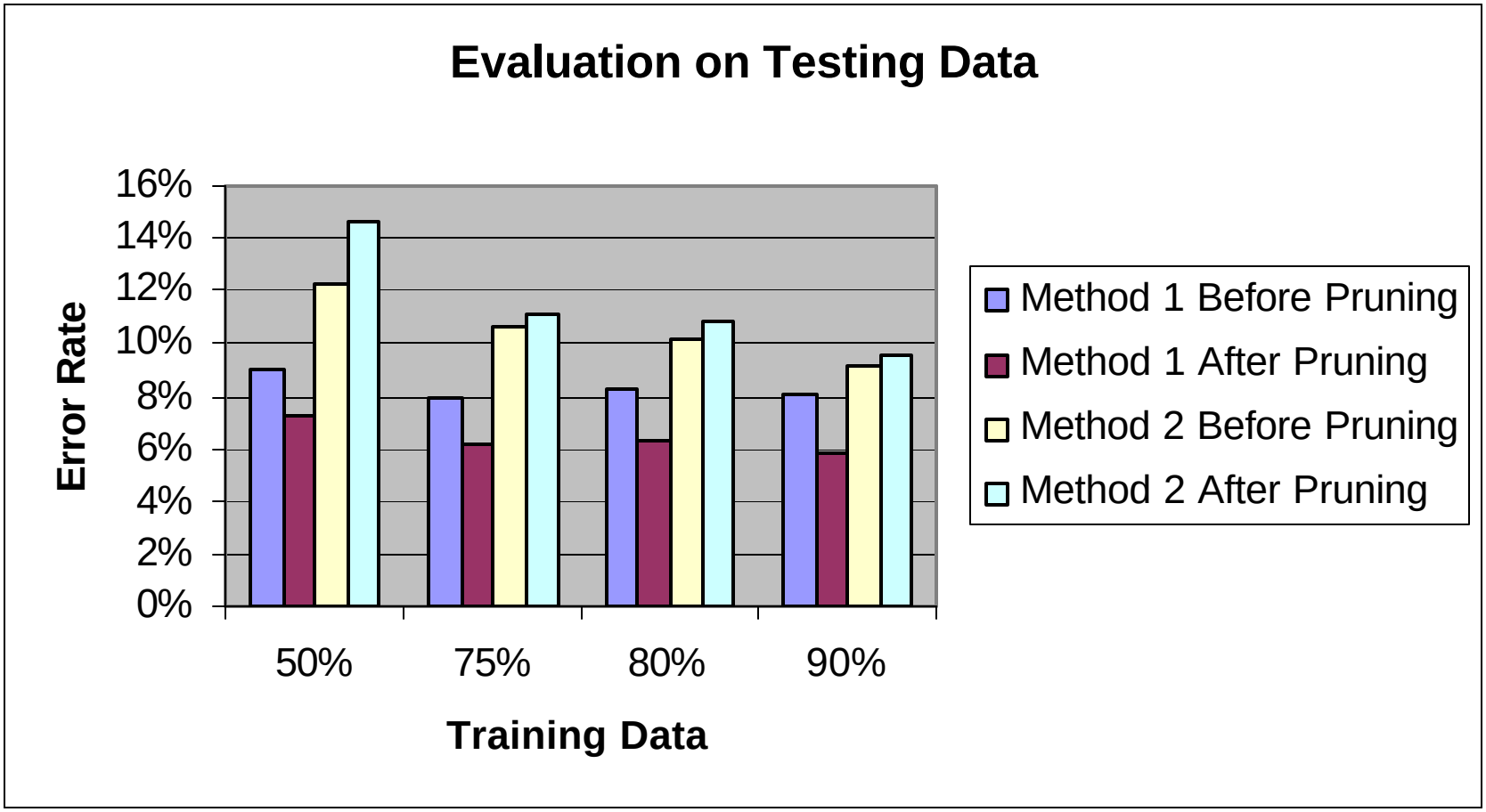
Experiments on input data with method 1

Training Data	Evaluation on training data		Evaluation on test data	
	Before Pruning	After Pruning	Before Pruning	After Pruning
50%	1%	3.55%	9%	7.20%
75%	1.10%	3.85%	8%	6.20%
80%	1.24%	3.82%	8.24%	6.30%
90%	1.15%	3.82%	8.08%	5.82%

Experiment on input data with method 2

Training Data	Evaluation on training data		Evaluation on test data	
	Before Pruning	After Pruning	Before Pruning	After Pruning
50%	1.60%	9.70%	12.30%	14.55%
75%	1.25%	6.80%	10.68%	11.13%
80%	1.18%	6.31%	10.16%	10.86%
90%	1.15%	5.23%	9.12%	9.63%

Comparing with different feature selection methods



Detailed Analysis –Method 1

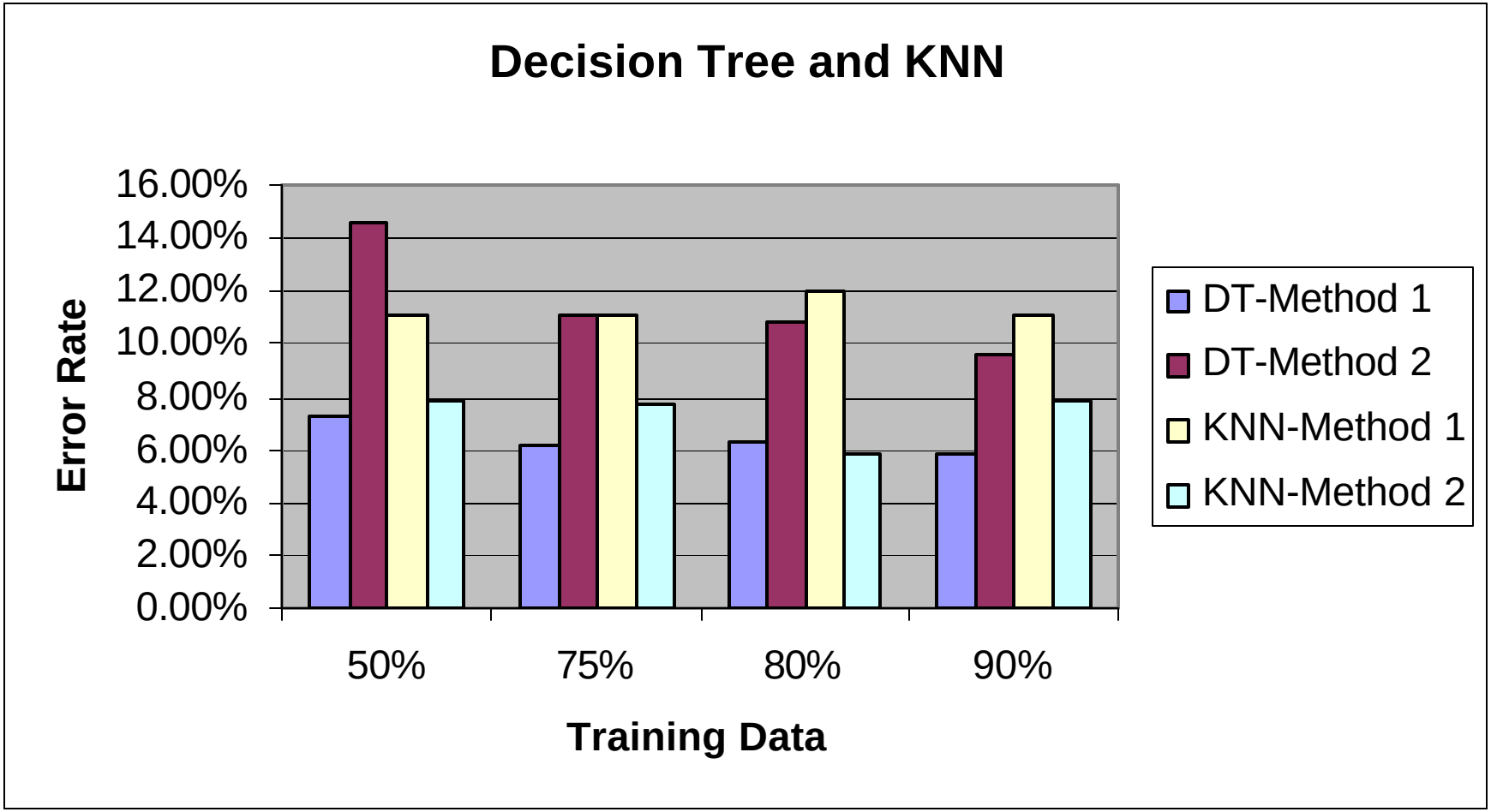
50%	class 1	class 2	class 3
class 1	93.70%	3.81%	2.36%
class 2	5.23%	91.11%	3.53%
class 3	2.79%	4.13%	93.08%

75%	class 1	class 2	class 3
class 1	95.14%	2.23%	2.62%
class 2	4.58%	92.29%	3.14%
class 3	2.25%	3.88%	93.87%

80%	class 1	class 2	class 3
class 1	95.28%	1.97%	2.76%
class 2	5.36%	91.37%	3.27%
class 3	2.12%	3.82%	94.05%

90%	class 1	class 2	class 3
class 1	96.46%	1.97%	1.57%
class 2	4.84%	92.03%	3.14%
class 3	2.06%	3.82%	94.11%

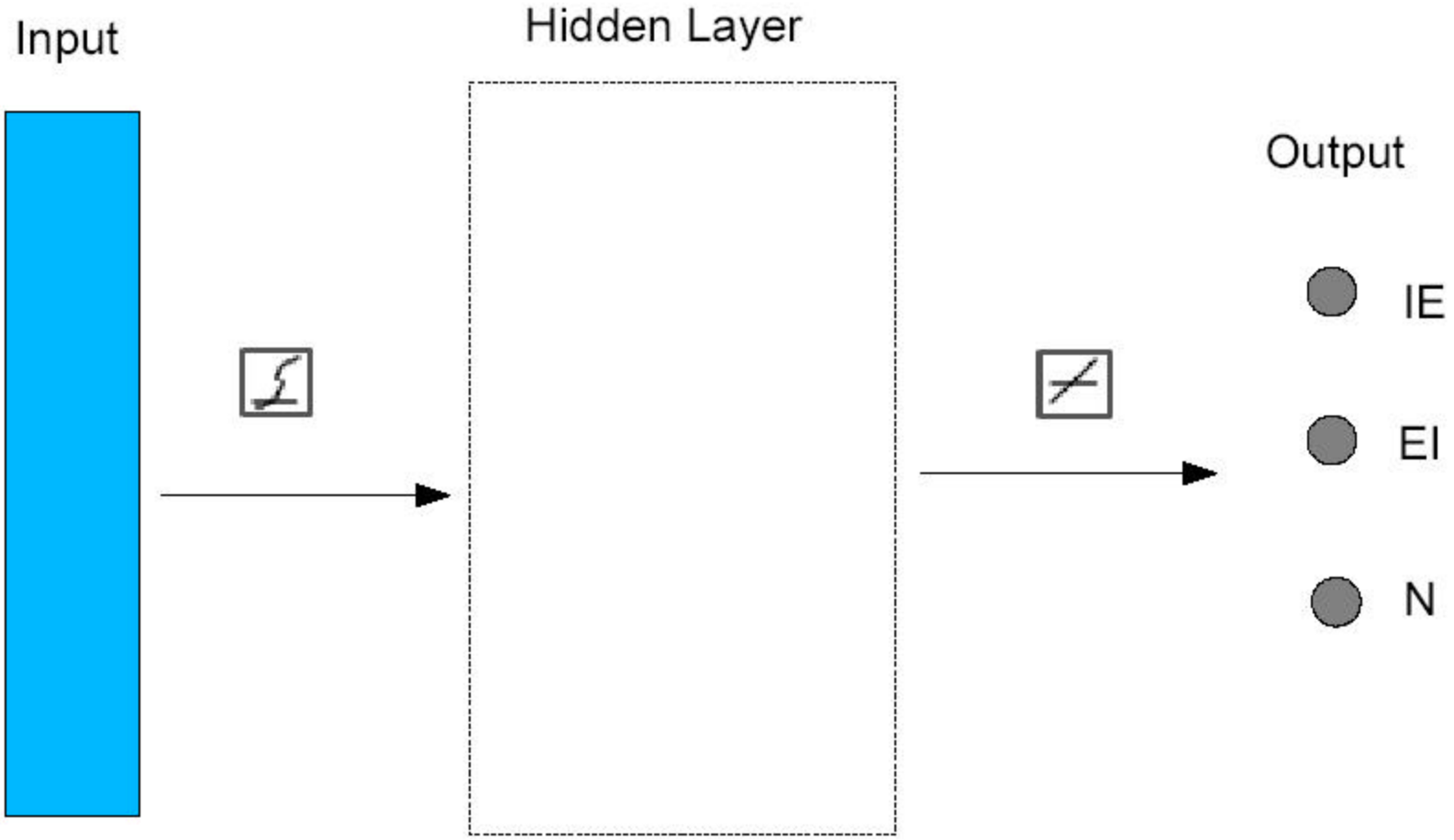
Error Rate Comparison



Artificial Neural Network

An approach for analyzing the
splice dataset

Backpropagation Network



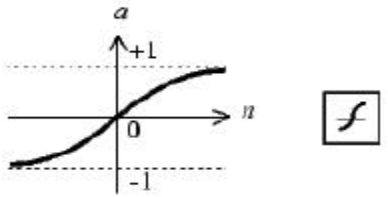
Backpropagation Network

- Input transfer function :

  solution to $n = \frac{2}{1 + \exp(-2*n)} - 1$

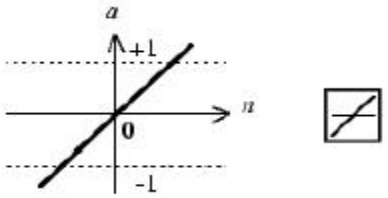
- Output transfer function :

- Error = $\frac{1}{N}\sum_p \sum_i (t_{pi} - a_{pi})^2$



$a = tansig(n)$

Tan-Sigmoid Transfer Function



$a = purelin(n)$

Linear Transfer Function

where N = the size of the training data

t_{pi} = target value(0 or 1)

a_{pi} = estimated value

p = 1,2,…,N and i = 1,2,3.

Backpropagation Network

- One Hidden Layer with 30 neurons.
- Output Layer with 3 units.
- Input – explained later.
- 80% of the data was used for training the network.
- Preliminary recoding of the data:
  - A : 00 C : 01 T : 10 G : 11

The Naïve Approach

- A : 1 C: 2 T: 3 G: 4
- • 60 dimensional input å $\{1,2,3,4\}^{60}$
- • Results :

	IE	EI	N
Training Err (%)	13.8	4.6	8.21
Testing Err (%)	18.76	15.78	13.89

A Better Approach

- Using sliding windows of width 3 to
- capture all the contiguous (overlapped) codons.

ACTTGC…

- • Input was 58 dimensional ε {1,2,…,64}^{58}
- • Intrinsic ‘distance’ between codons
  - – vice & virtue.

Results

Naïve Method	IE	EI	N
Training Err (%)	13.8	4.6	8.21
Testing Err (%)	18.76	15.78	13.89

Codon Method	IE	EI	N
Training Err (%)	10.97	8.09	7.38
Testing Err (%)	16.49	12.73	11.45

An Unfruitful Attempt

- Each codon was represented by 64 indicator variables.
- Instead of taking all the 58 contiguous codons, took only the 20 disjoint ones.
- Even that meant 20*64 = 1280 indicators.
- Pre-constructed network could not be used because of MATLAB memory shortage.
- • Constructing a network from scratch was unwieldy.

Concluding Remarks

- Lack of scientific choice for the ANN parameters.
- Not incorporating any ***domain theory.***
- Training a Backpropagation Network was very time consuming. Each training-run took nearly 20 to 30 mins.
- Results of our BP network were comparable with a similar study on the classification of Eukaryotic and Prokaryotic cells based on DNA sequences (mentioned in the report).

Conclusions

- The Decision Tree approach with method 1 performs the best
- Interestingly, different feature selection methods work quite differently for different classifiers
  - For Decision Tree, the method 1 works better than the method 2
  - For K-NN, method 2 works better than method 1
- Artificial Neural Networks (ANN) works not as good as expected.
  - The problem might be we just have one single hidden layer and for that layer the number of elements can not be too large, otherwise, the computation takes too long time.