Data Mining
Classification: Alternative Techniques

Topics: Model Overfitting, Nearest-Neighbor classifiers, and Bayesian Classifiers

Introduction to Data Mining , by

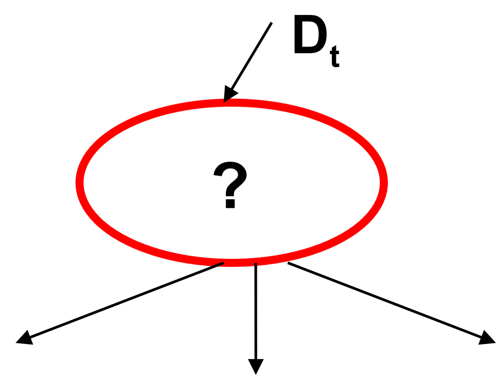
Tan, Steinbach, Karpatne, Kumar

Review of decision tree

General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

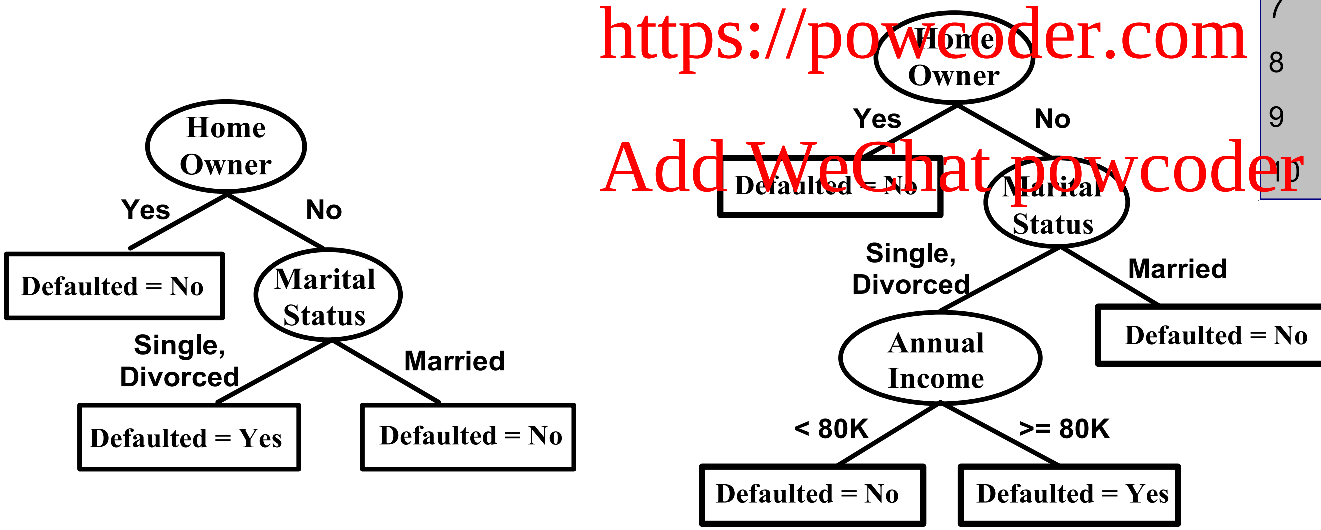
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Hunt's Algorithm

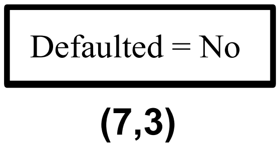
(a) (b)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

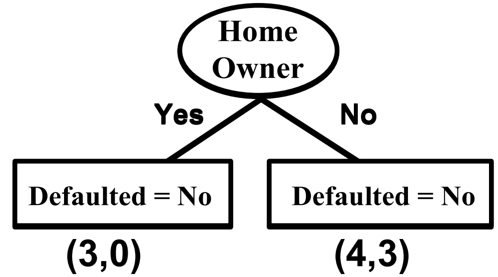


(c) (d)

Hunt's Algorithm

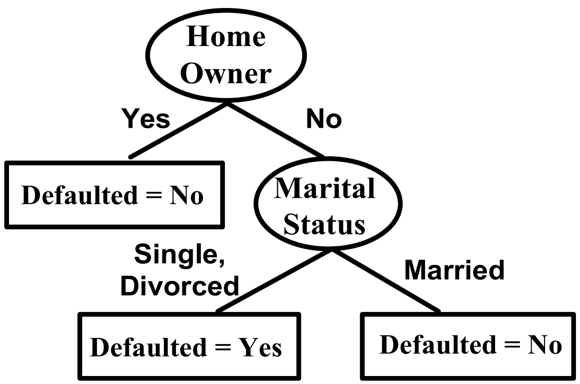


(a)

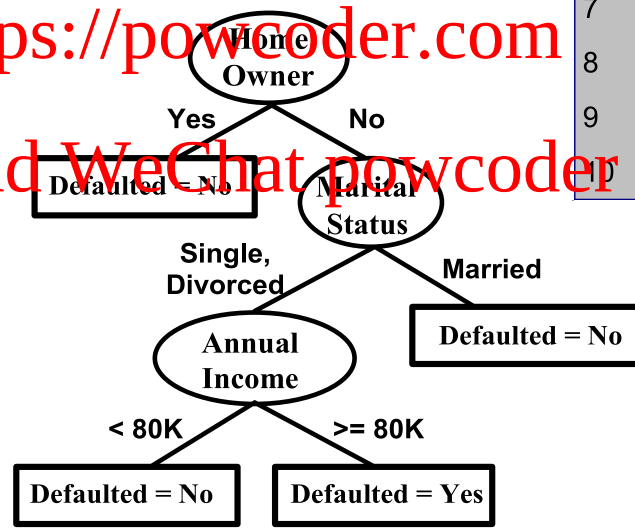


(b)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

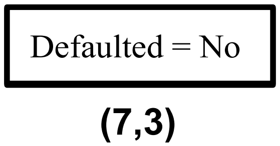


(c)

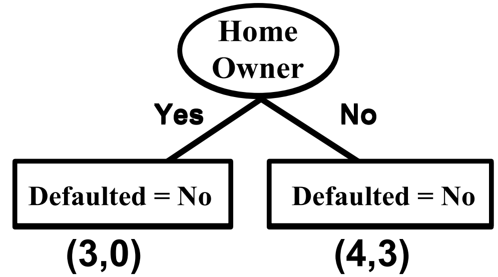


(d)

Hunt’s Algorithm

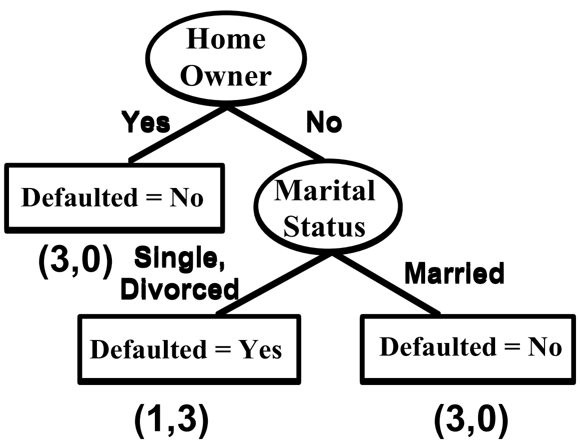


(a)

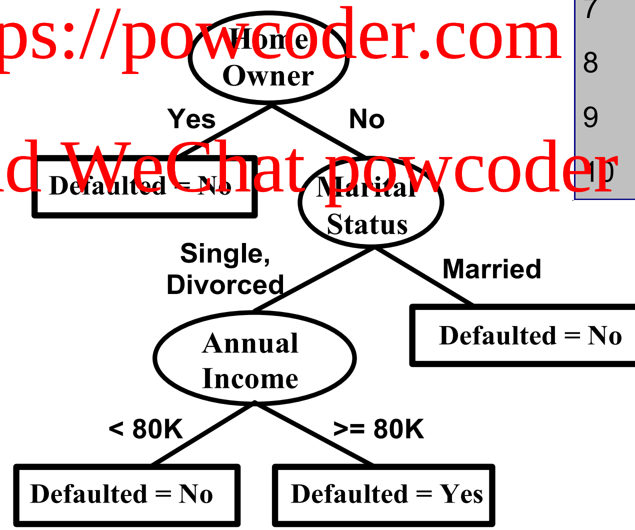


(b)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

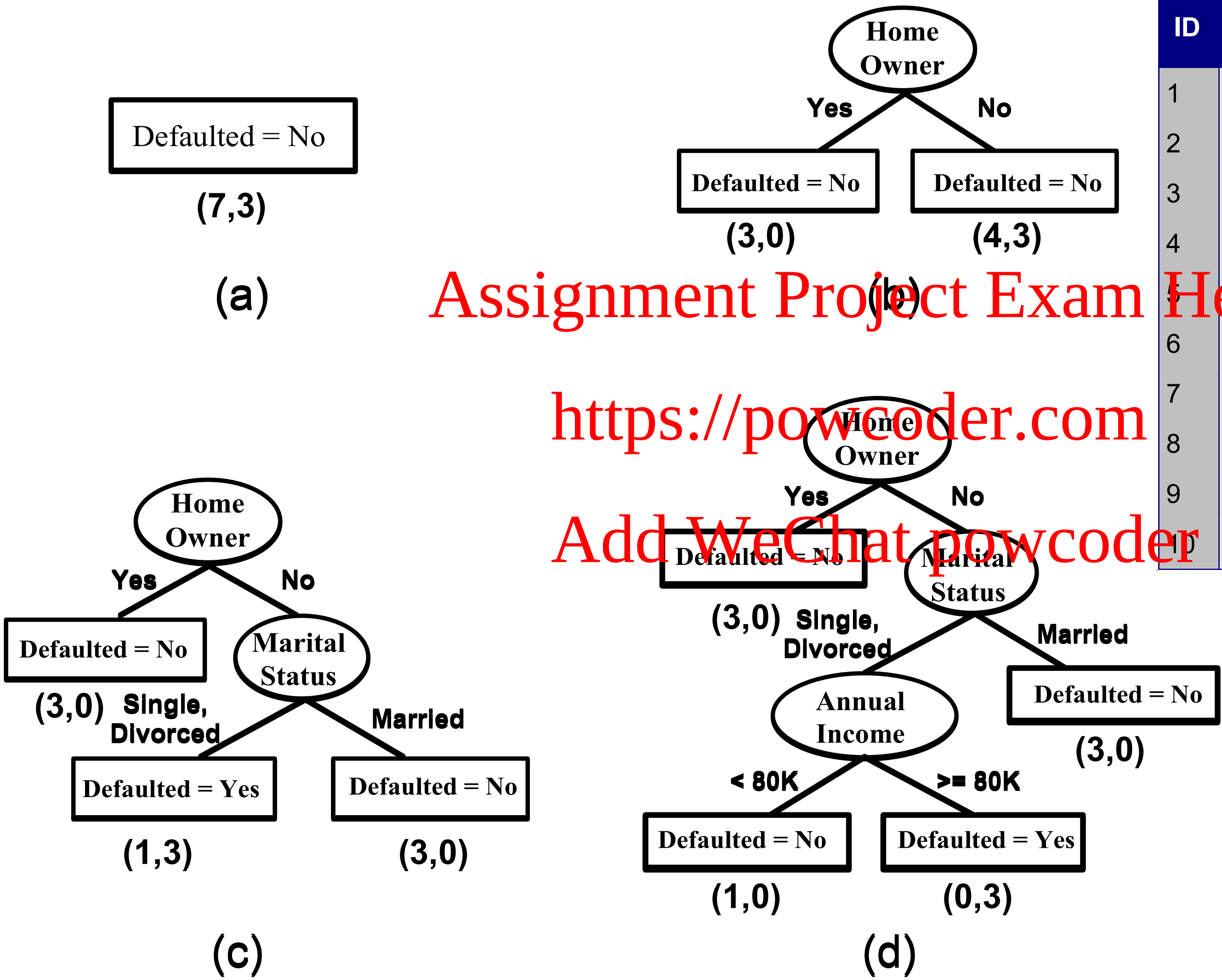


(c)



(d)

Hunt's Algorithm



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Measures of Node Impurity

- Gini Index

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

t is a node

- Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

Entropy quantifies uncertainty

- Misclassification error

$$Error(t) = 1 - \max_i P(i \mid t)$$

Finding the Best Split

1. Compute impurity measure (P) before splitting
2. Compute impurity measure (M) after splitting
   - Compute impurity measure of each child node
   - M is the weighted impurity of children
3. Choose the attribute test condition that produces the highest gain

Gain = P – M

or equivalently, lowest impurity measure after splitting (M)

Summary of decision tree

- Finding an optimal decision tree is NP-complete
- Existing algorithms for tree building are efficient. Classification is efficient O(w), w is the tree depth.
- Small trees are easy to interpret
- Robust to the presence of noise
- When using a single attribute for a test condition, the decision boundary (border between different classes) are rectilinear (e.g. parallel to the coordinate axes)

Data Mining

Model Overfitting

(section Model Overfitting)

Introduction to Data Mining by

Tan, Steinbach, Karpatne, Kumar

Classification Errors

- Training errors (apparent errors)
  – Errors committed on the training set

- Test errors

  – Errors committed on the test set

- Generalization errors
  – Expected error of a model over random selection of records from same distribution

Evaluate the classification performance

- Training data

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower	Predicted class
1	Yes	Single	125K	**No**	No
2	No	Married	100K	**No**	No
3	No	Single	70K	**No**	Yes
4	Yes	Married	120K	**No**	No
5	No	Divorced	95K	**Yes**	No
6	No	Married	60K	**No**	Yes
7	Yes	Divorced	220K	**No**	No
8	No	Single	85K	**Yes**	Yes
9	No	Married	75K	**No**	No
10	No	Single	90K	**Yes**	Yes

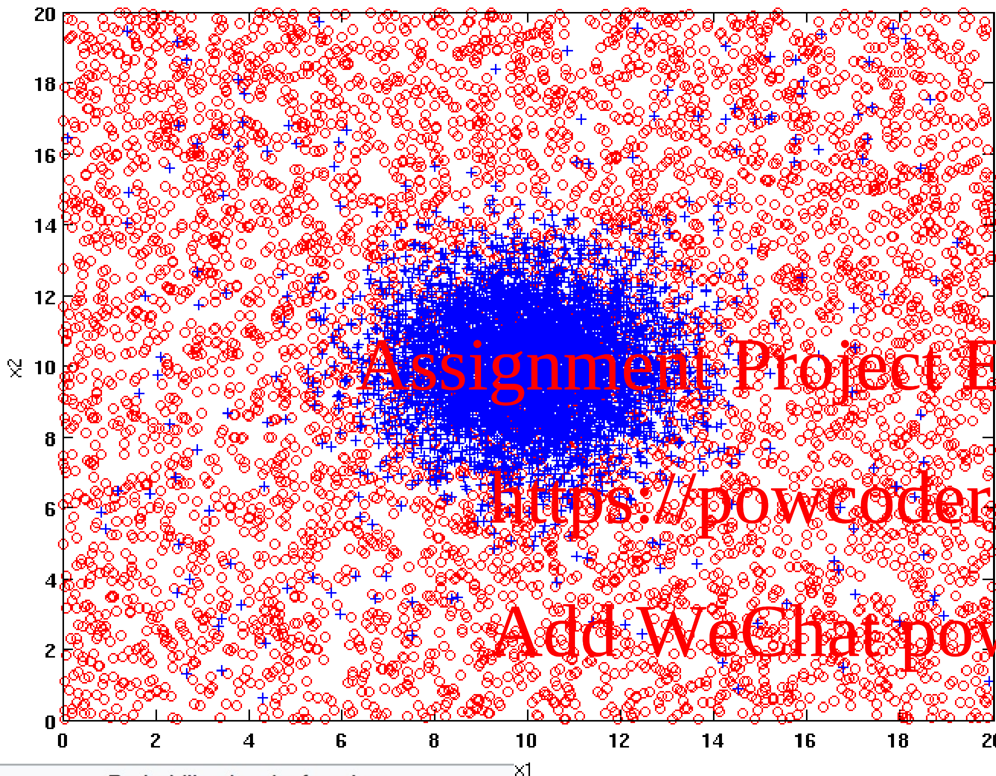
Accuracy = the number of correct predictions / total records
= 7/10 = 0.7

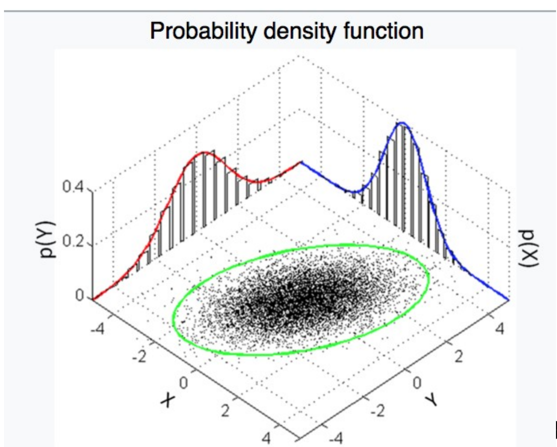
Error rate = the number of wrong predictions / total records
= 3/10 = 0.3

Confusion matrix

		Predicted Class	
		C=Yes	C=No
Actual Class	C=Yes	2	1
	C=No	2	5

Example Data Set





Two class problem:

+ : 5200 instances

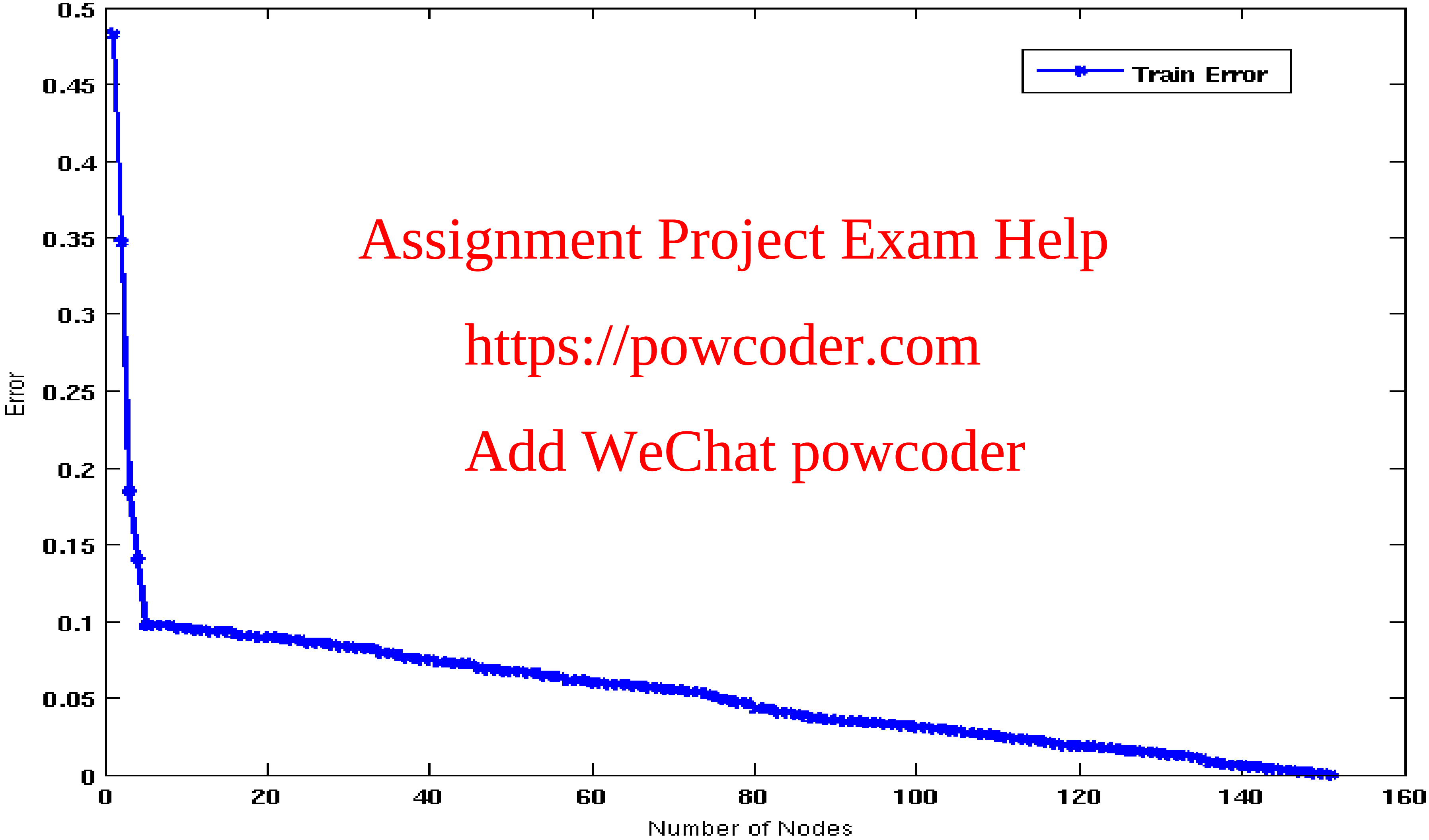
- **5000 instances generated from a Gaussian centered at (10,10)**
- **200 noisy instances added**

o : 5200 instances

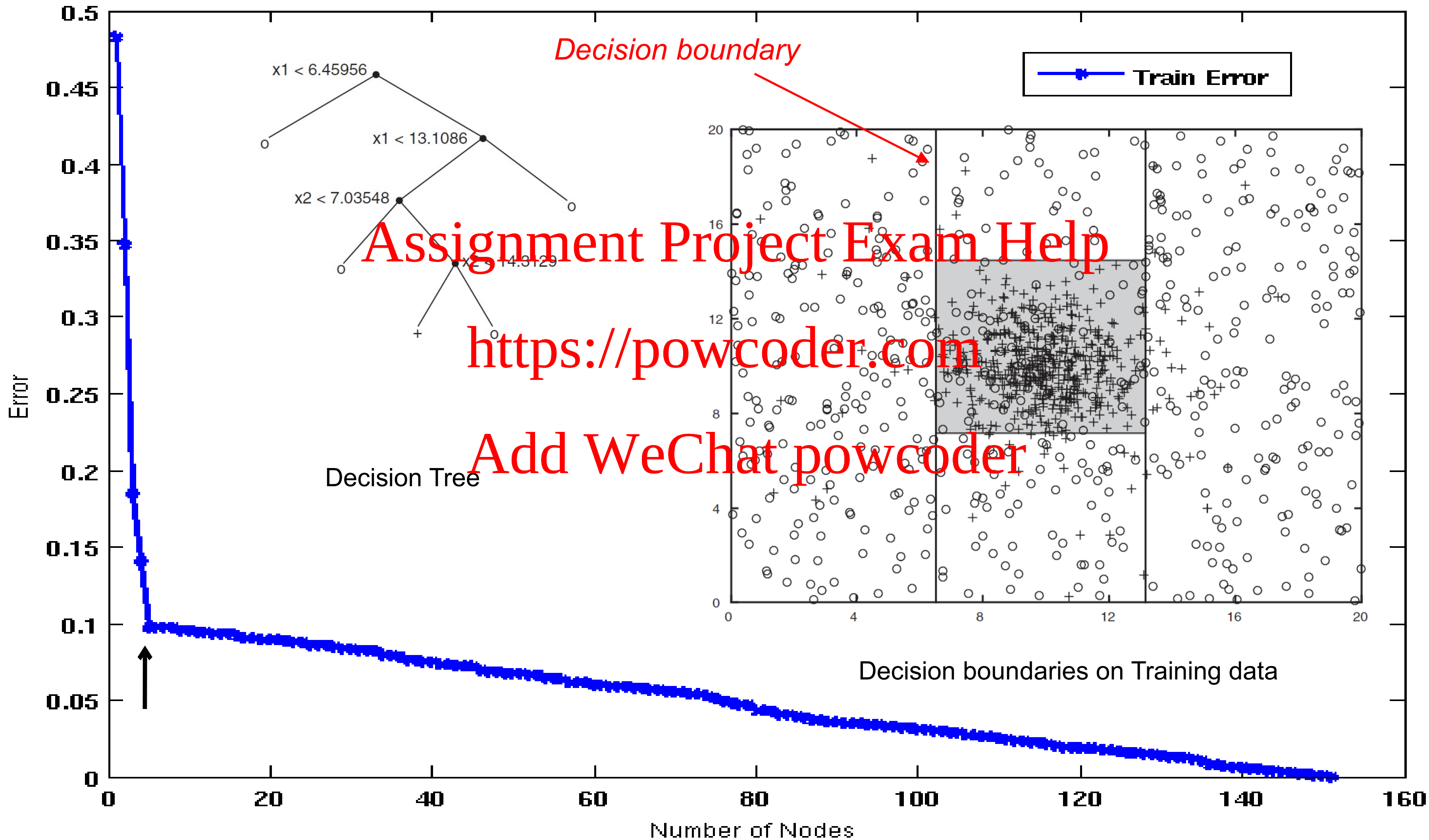
- **Generated from a uniform distribution**

10 % of the data used for training and 90% of the data used for testing

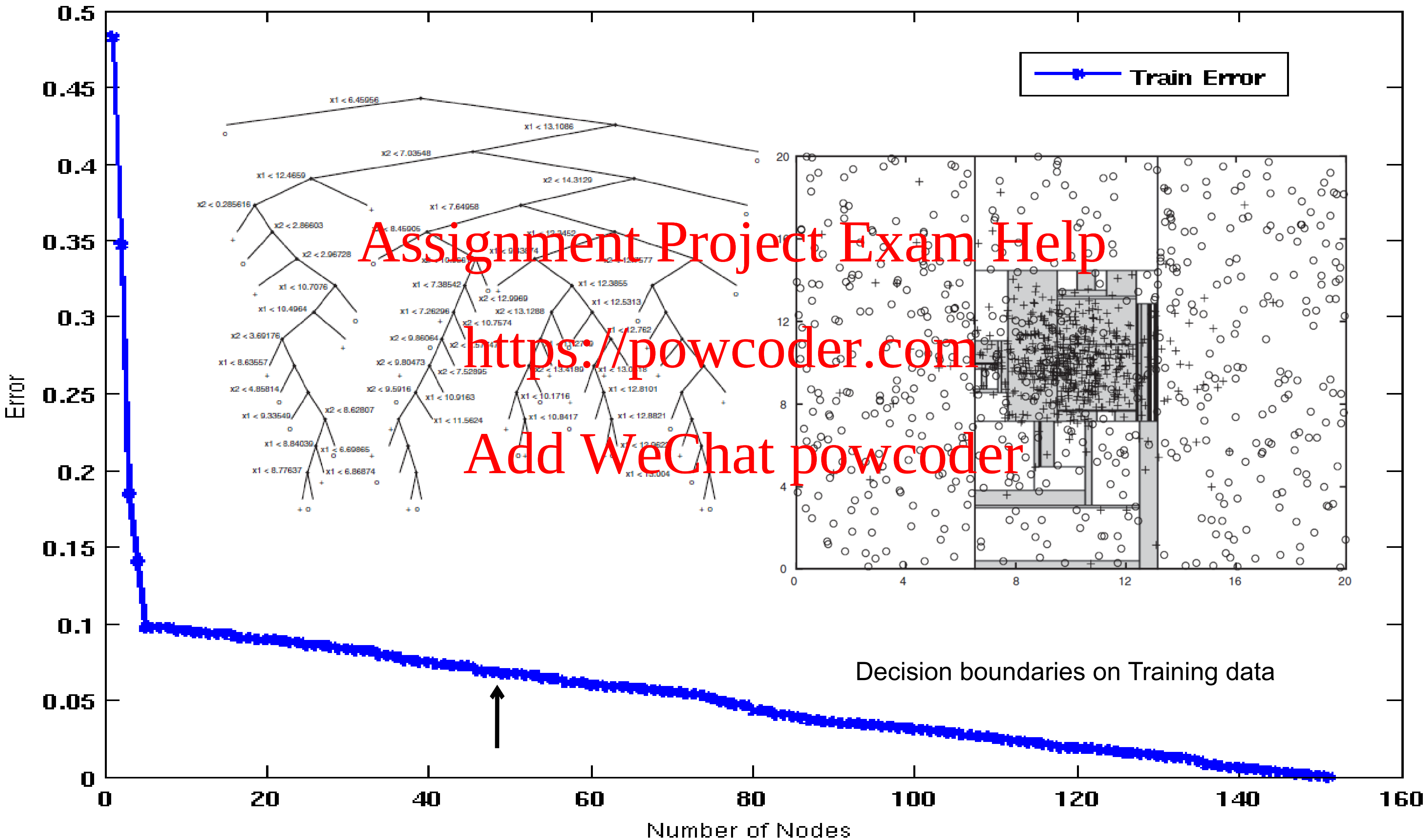
Increasing number of nodes in Decision Trees



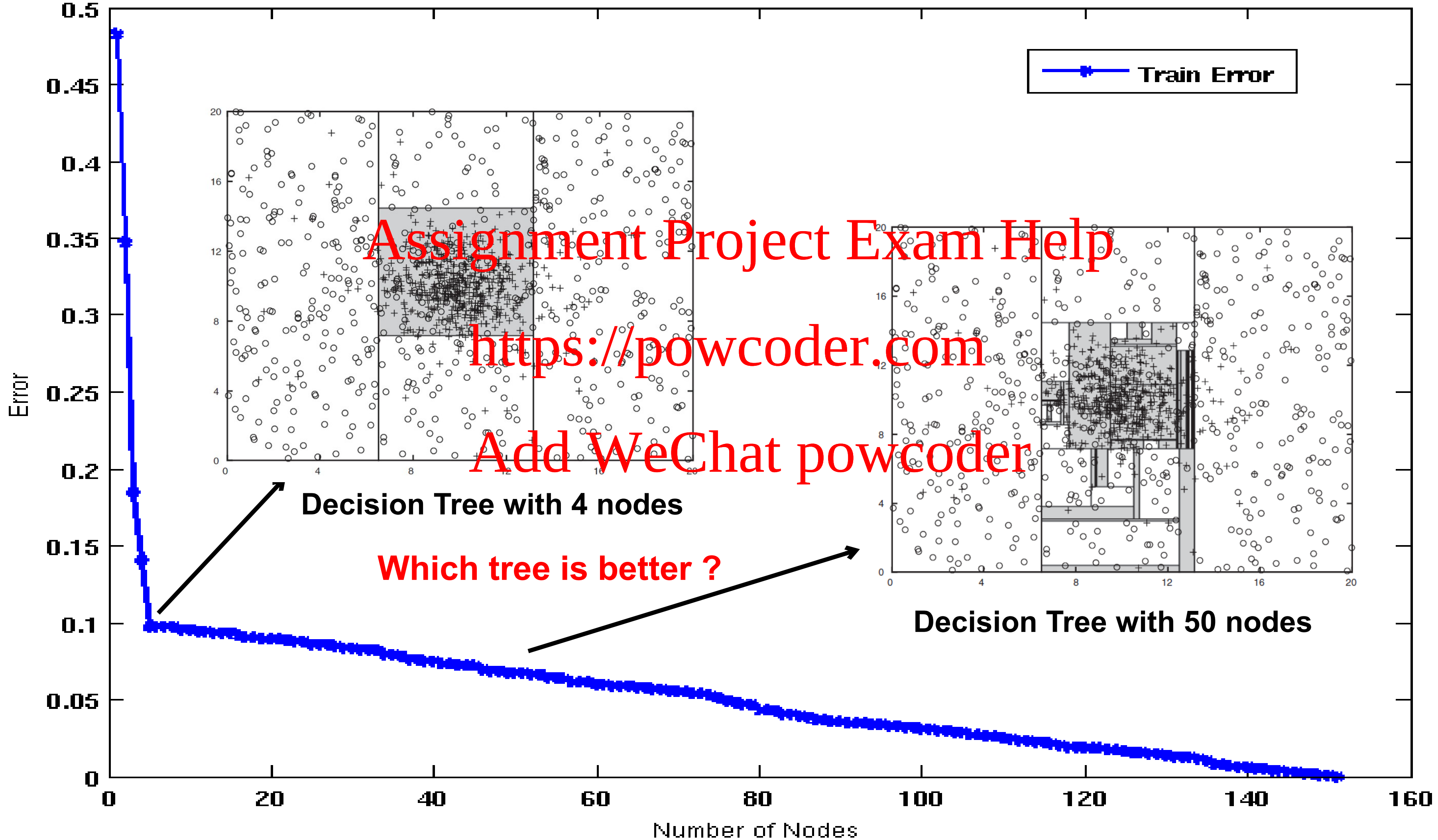
Decision Tree with 4 nodes



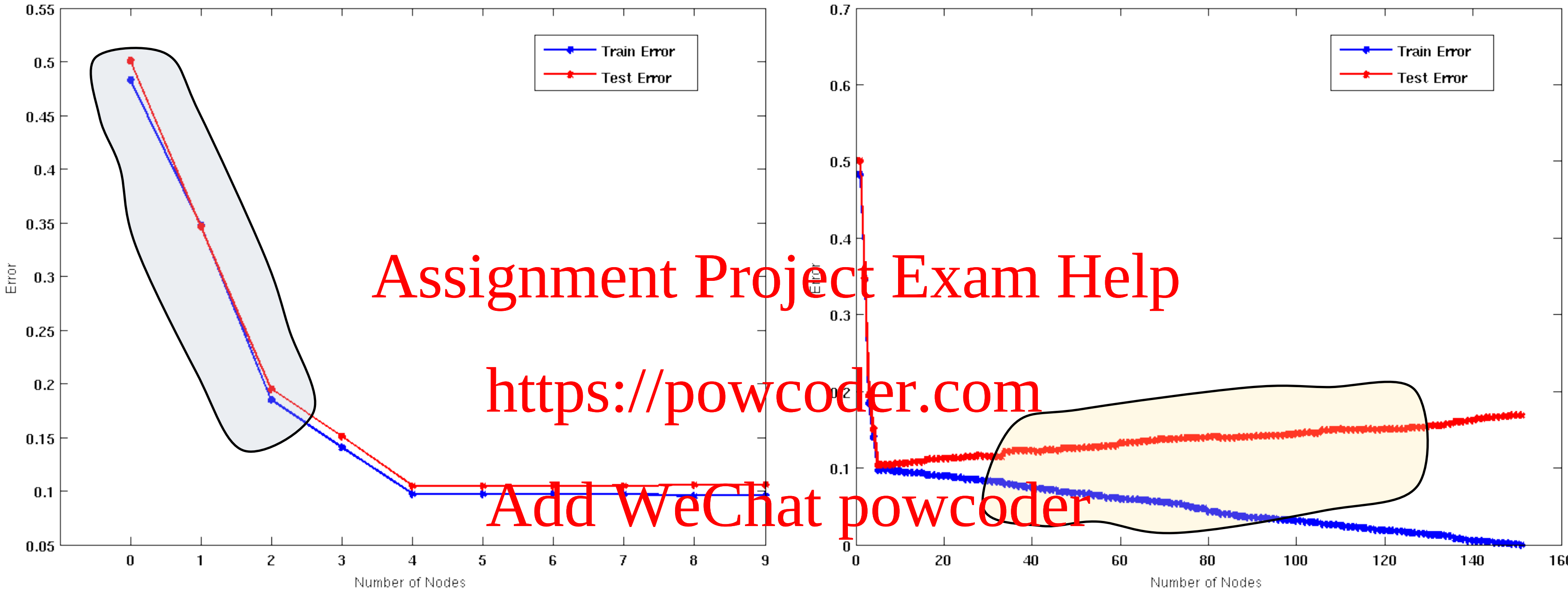
Decision Tree with 50 nodes



Which tree is better?



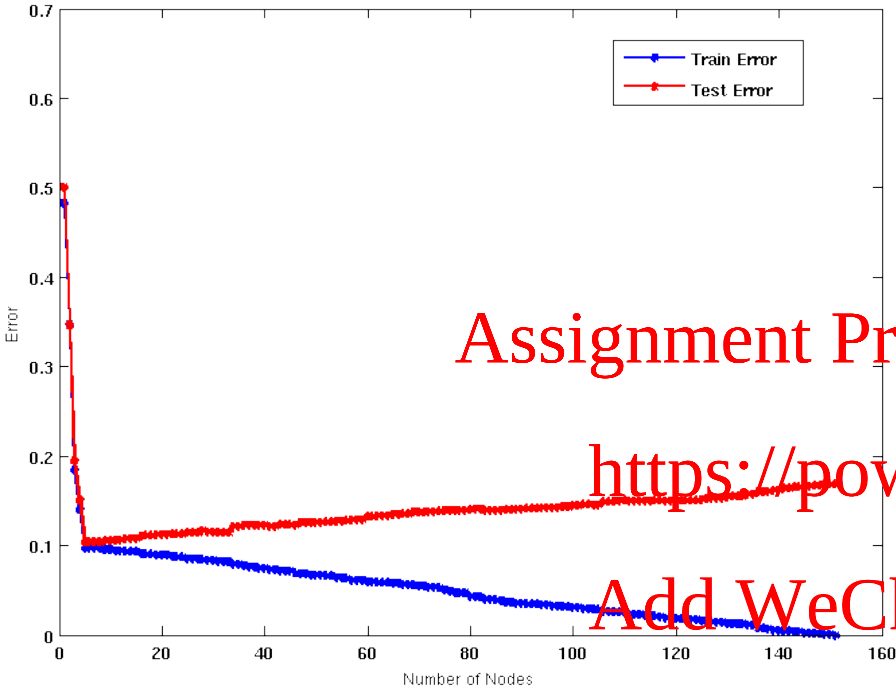
Model Overfitting

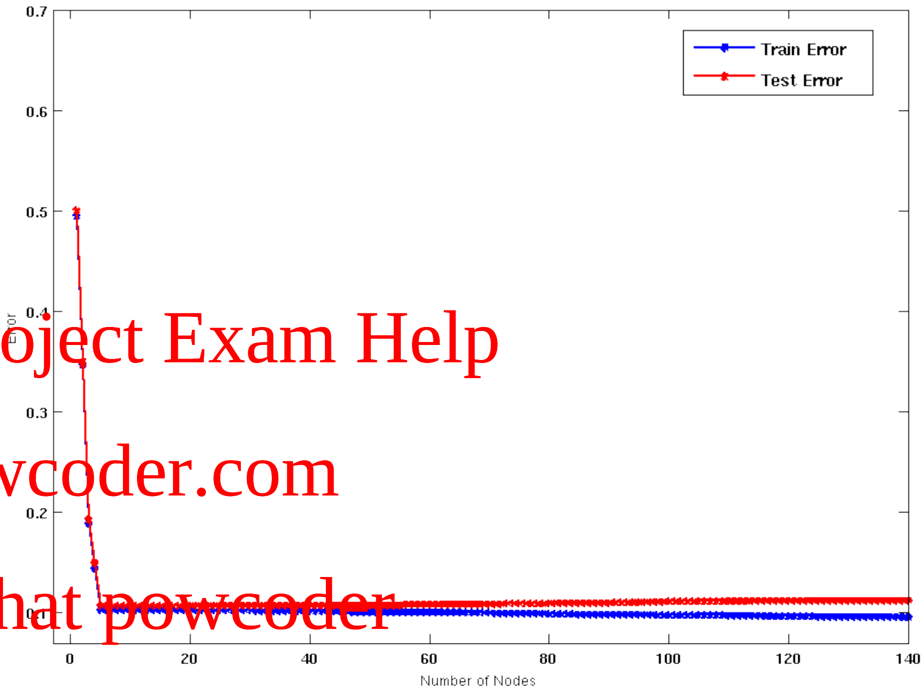


Underfitting: when model is too simple, both training and test errors are large

Overfitting: when model is too complex, training error is small but test error is large

Model Overfitting

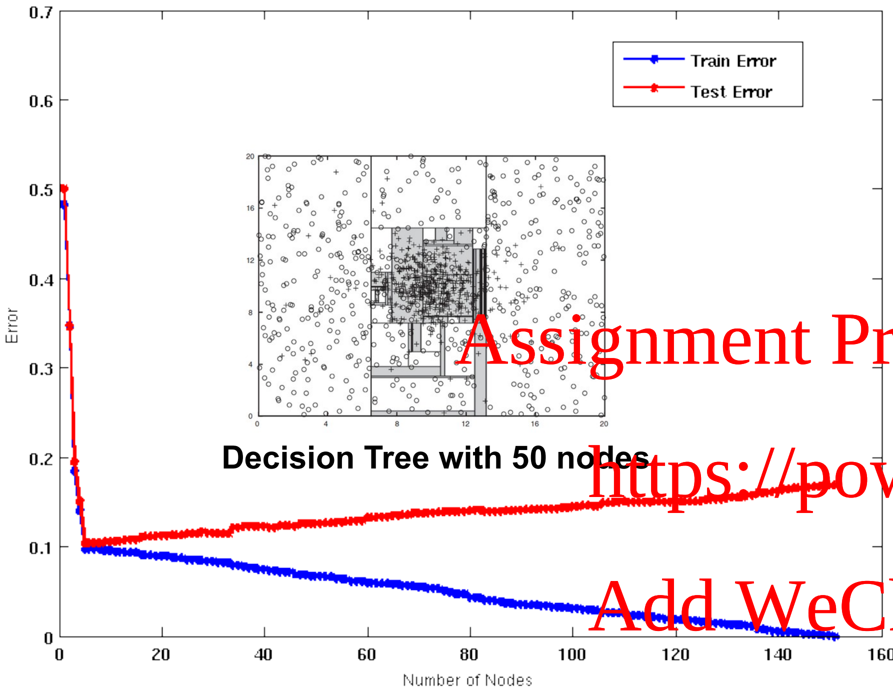


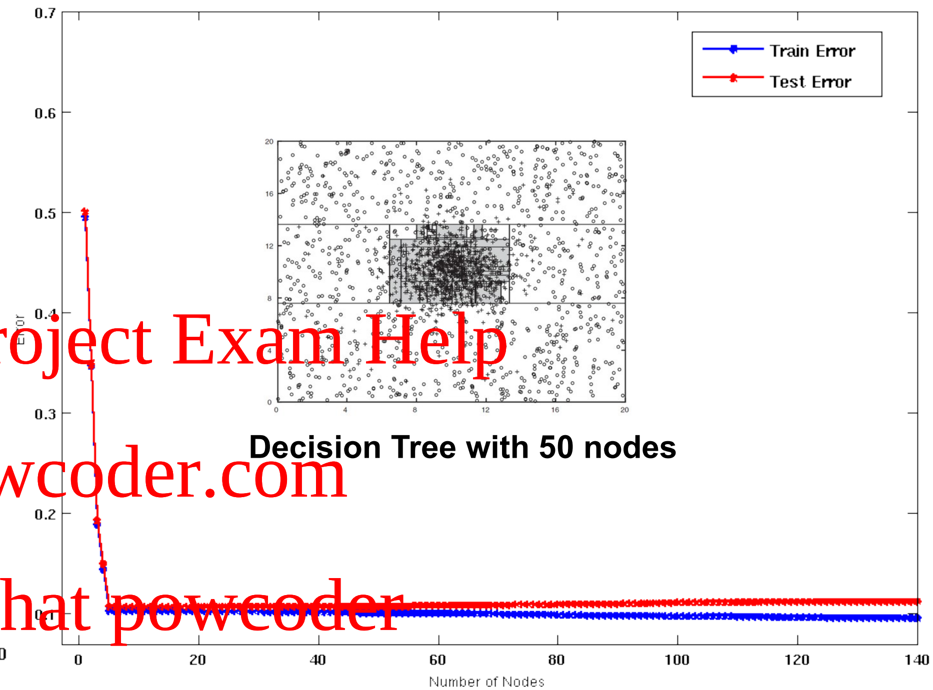


Using twice the number of data instances

- If **training data is under-representative**, testing errors increase and training errors decrease on increasing number of nodes
- Increasing the size of training data reduces the difference between training and testing errors at a given number of nodes

Model Overfitting





Using twice the number of data instances

- If training data is under-representative, testing errors increase and training errors decrease on increasing number of nodes
- Increasing the size of training data reduces the difference between training and testing errors at a given number of nodes

Reasons for Model Overfitting

- Limited Training Size

- High Model Complexity
  - Multiple Comparison Procedure

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary

- Training error does not provide a good estimate of how well the tree will perform on previously unseen records

- Need ways for estimating generalization errors

Model Selection

- Performed during model building
- Purpose is to ensure that model is not overly complex (to avoid overfitting)

- Need to estimate generalization error

  – Using Validation Set

  – Incorporating Model Complexity
  – Estimating Statistical Bounds

Model Selection: Using Validation Set

- Divide training data into two parts:
  - Training set:
    - use for model building
  - Validation set:
    - use for estimating generalization error
    - Note: validation set is not the same as test set
      - You know the labels for samples in the validation set

- Drawback:
  - Less data available for training

Model Selection:
Incorporating Model Complexity

- Rationale:
  - Given two models of similar generalization errors, one should prefer the simpler model over the more complex model

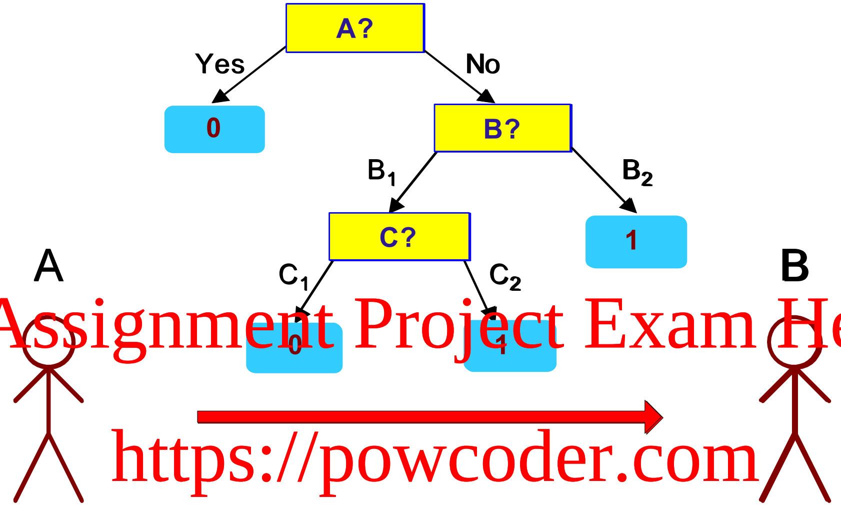
  - A complex model has a greater chance of being fitted accidentally by errors in data

  - Therefore, one should include model complexity when evaluating a model

Gen. Error(Model) = Train. Error(Model, Train. Data) + α x Complexity(Model)

Minimum Description Length (MDL)

X	y
X_1	1
X_2	0
X_3	0
X_4	1
…	…
X_n	1



X	y
X_1	?
X_2	?
X_3	?
X_4	?
…	…
X_n	?

- Cost(Model,Data) = Cost(Data|Model) + α x Cost(Model)
  - Cost is the number of bits needed for encoding.
  - Search for the least costly model.
- Cost(Data|Model) encodes the misclassification errors.
- Cost(Model) uses node encoding (number of children) plus splitting condition encoding.
- The ideal version of MDL is given by the Kolmogorov Complexity, which is defined as the length of the shortest computer program that prints the sequence of observed data and halts.

Model Selection for Decision Trees

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).
    - Stop if estimated generalization error falls below certain threshold

Model Selection for Decision Trees

- Post-pruning
  - Grow decision tree to its entirety
  - Subtree replacement
    - Trim the nodes of the decision tree in a bottom-up fashion
    - If generalization error improves after trimming, replace sub-tree by a leaf node
    - Class label of leaf node is determined from majority class of instances in the sub-tree
  - Subtree raising
    - Replace subtree with most frequently used branch

Model Evaluation

- Purpose:
  - To estimate performance of classifier on previously unseen data (test set)
- Holdout

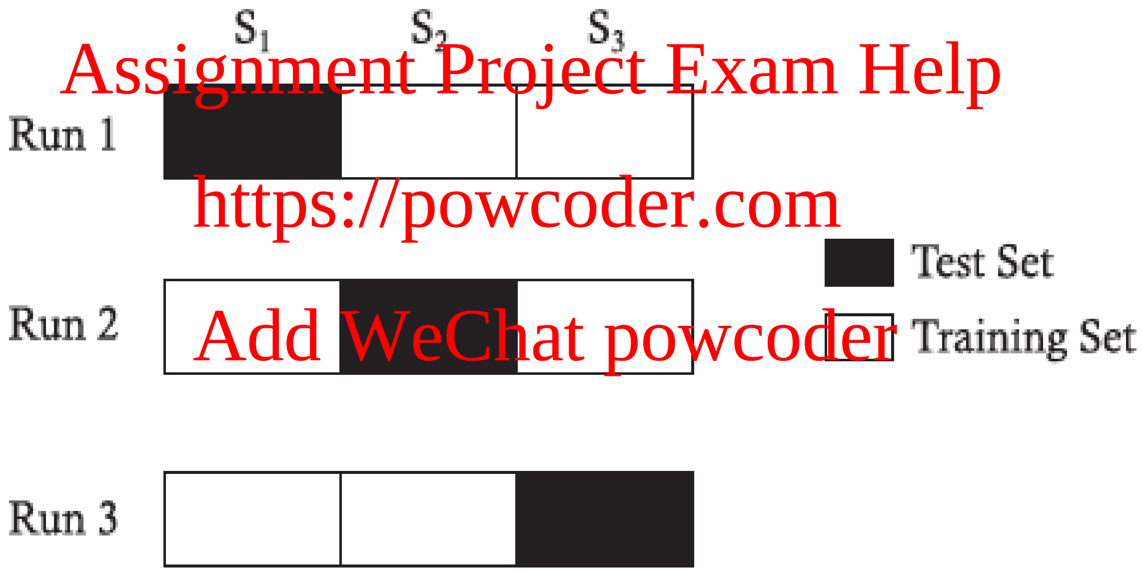
  - Reserve k% for training and (100-k)% for testing

  - Random subsampling: repeated holdout
- Cross validation

  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n

Cross-validation Example

- 3-fold cross-validation

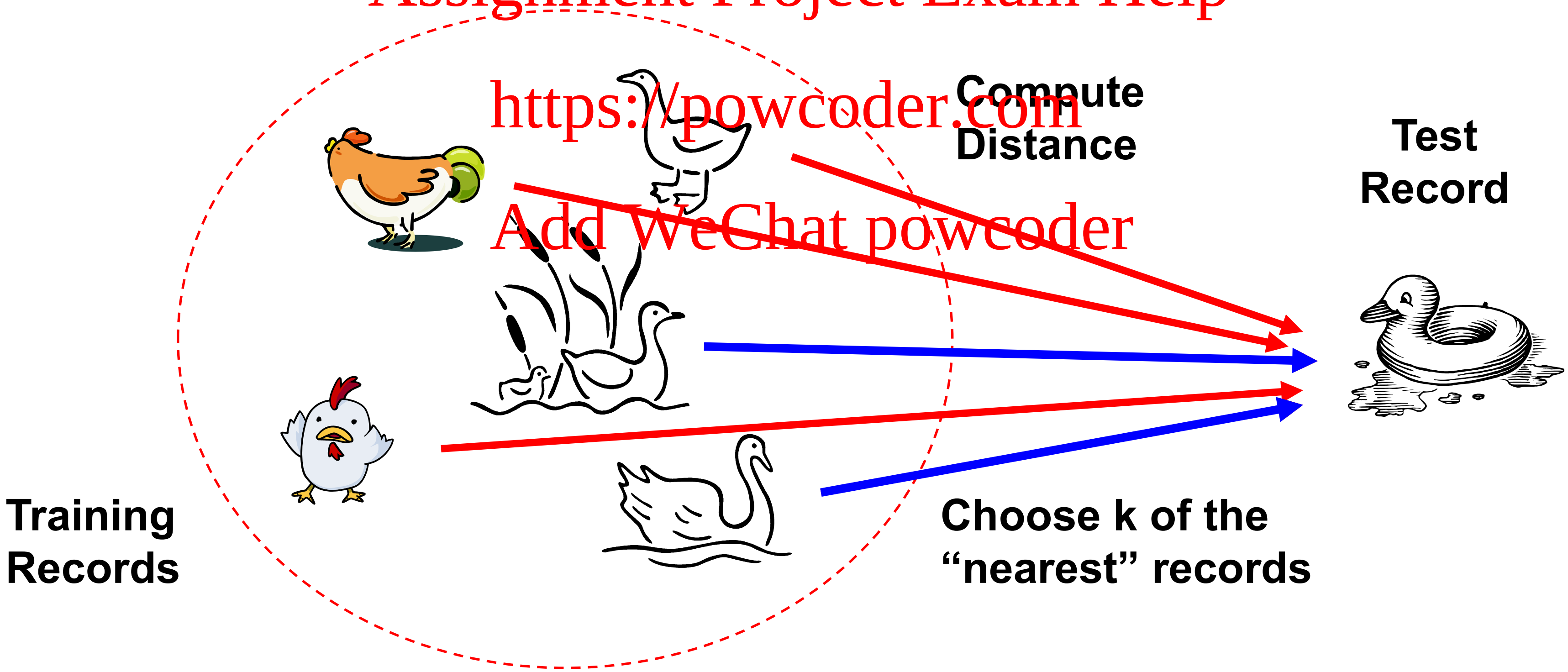


Alternative classifiers: instance Based Classifiers

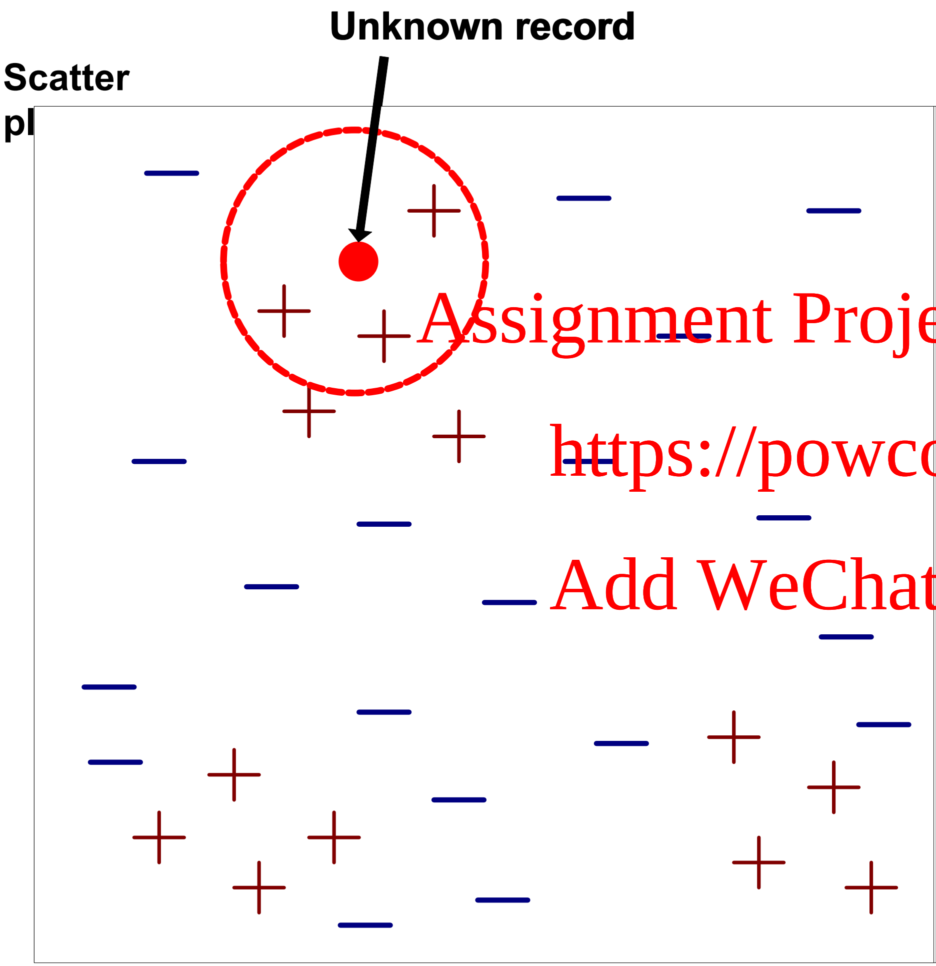
- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor
    - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



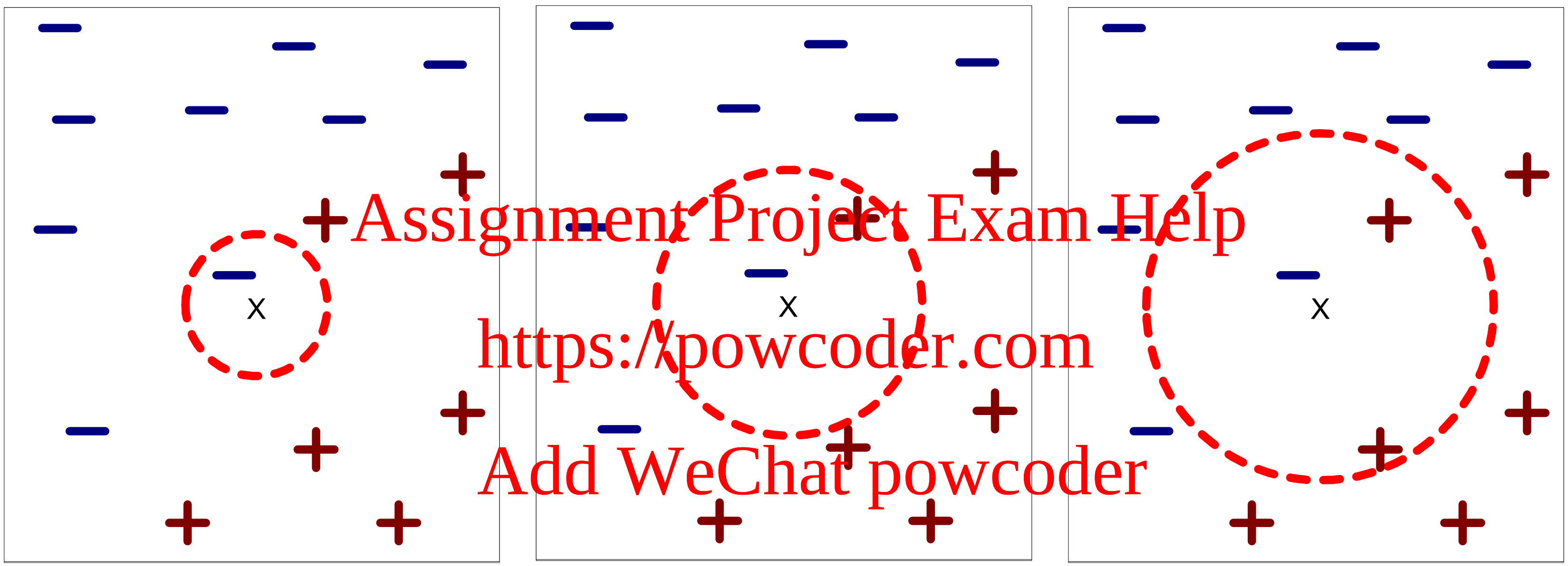
Nearest-Neighbor Classifiers



Think about the new coronavirus example, the training set contains the genomes of known viruses

- Requires three things
  - The set of labeled records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve
- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

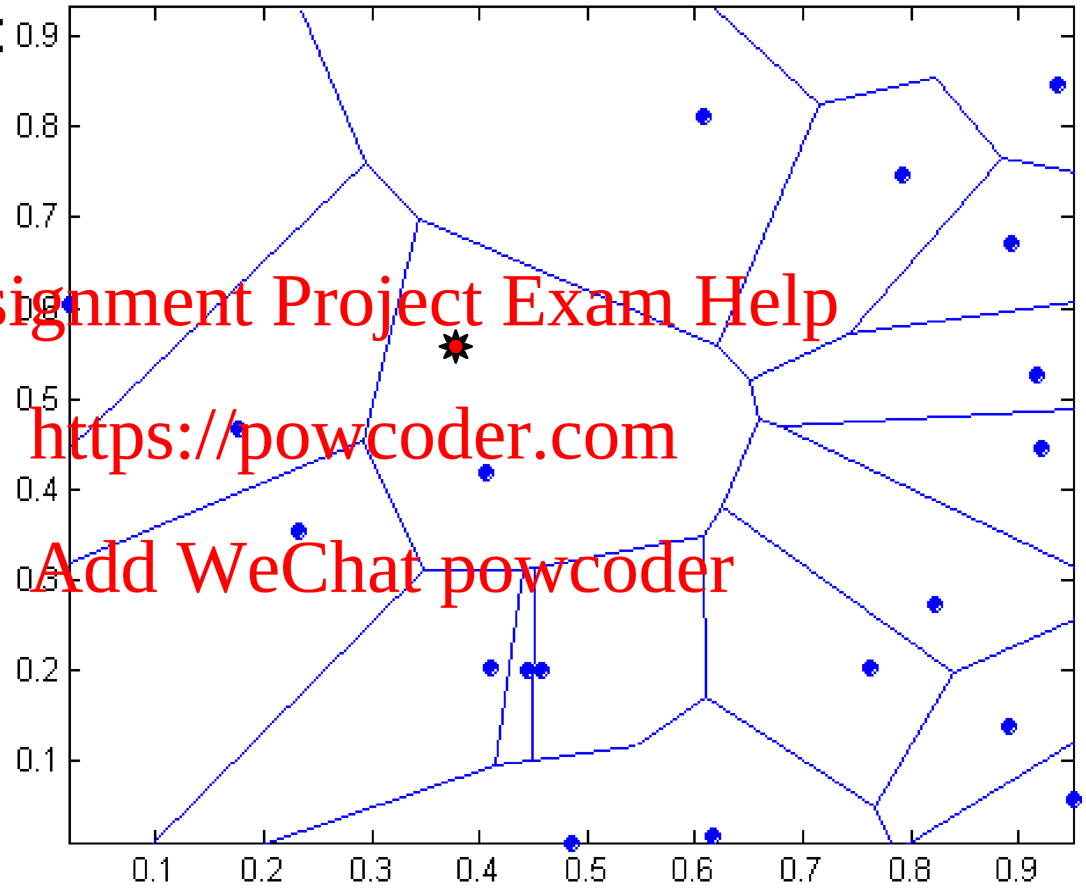
K-nearest neighbors of a record x are data points that have the k smallest distances to x

1 nearest-neighbor (fast version)

Voronoi Diagram:

Blue dots: seeds.

Around each seed is a cell. Inside the cell for a seed, any point in the Cell (e.g. the red) is closer to its seed than to other points.



https://en.wikipedia.org/wiki/Voronoi_diagram#/media/File:Voronoi_growth_euclidean.gif

Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

    - Example: p=(2, 3), p=(2, 0), d=?

  - Hamming distance
    - Example: the Hamming distance between "cat" and "bat" is 1. Same length, Hamming distance = the positions of mismatches

Nearest Neighbor Classification

- Determine the class from nearest neighbor list
  - Take the majority vote of class labels among the k-nearest neighbors

Algorithm 1 The k-nearest neighbor classification algorithm

1: Let k be the number of nearest neighbors and D be the set of training examples

2: **for** each test example $z = (\mathbf{x}', y')$ **do** y' is unknown

3: Compute $d(\mathbf{x}', \mathbf{x})$, the distance between z and every example, $(\mathbf{x}, y) \in D$
4: Select $D_z \subseteq D$, the set of k closest training examples to z
5: $y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} \mathbf{I}(v = y_i)$

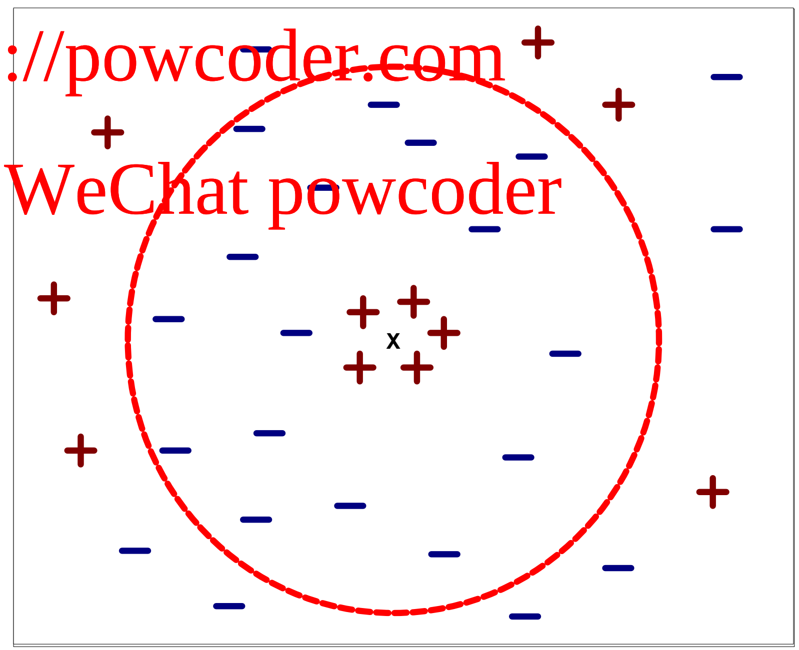
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

In-class exercise, problem 1:
Describe a method to decide the value of k.

Please answer it on Canvas.

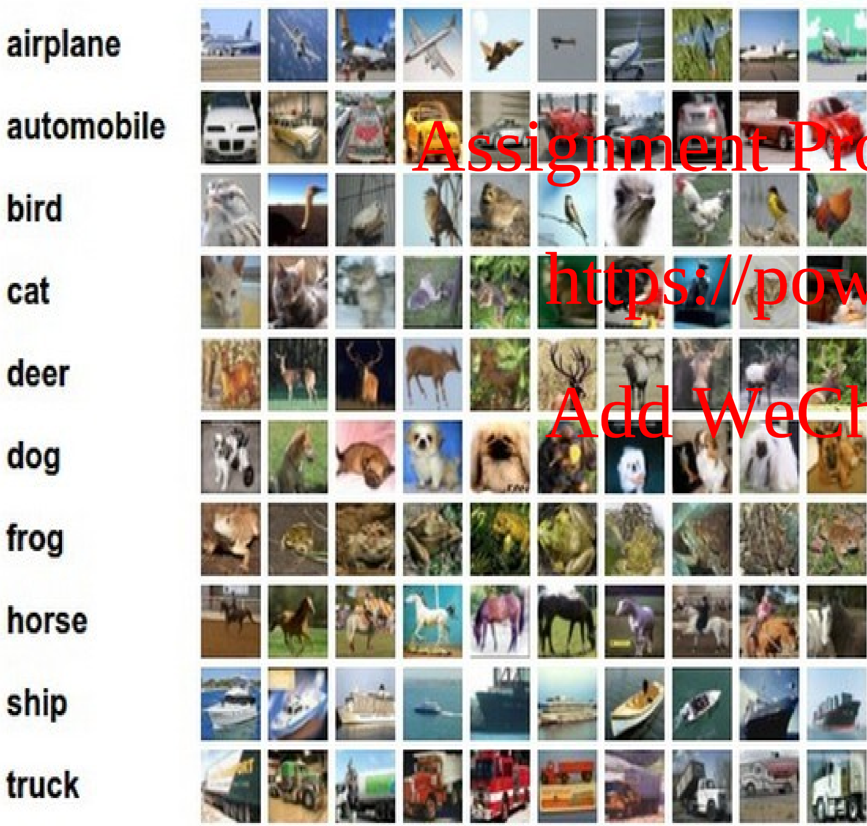


Nearest Neighbor Classification…

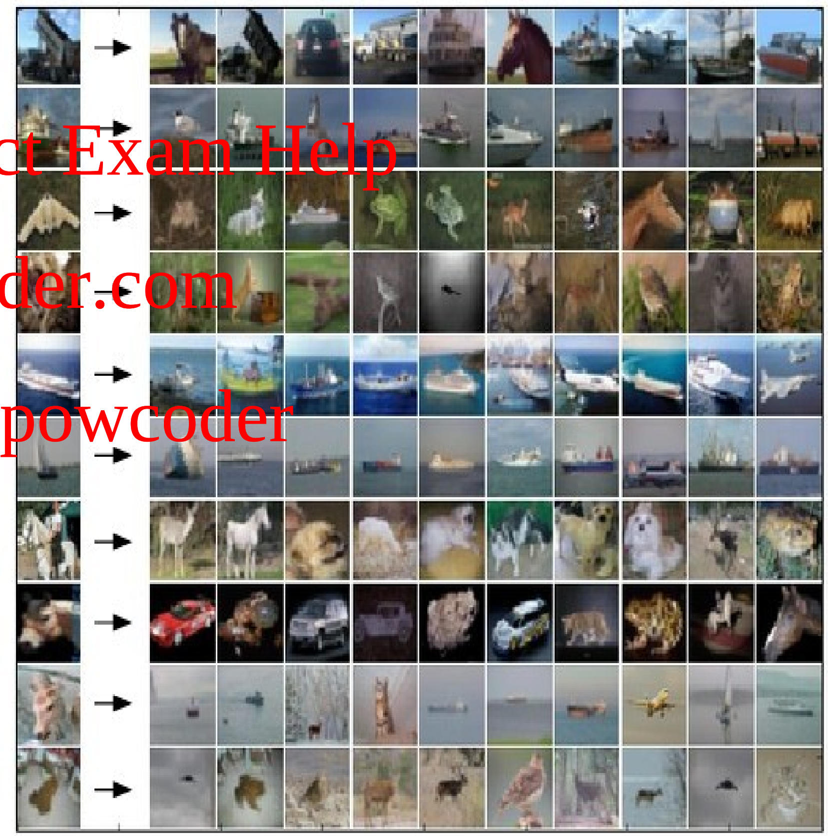
- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Example: image classification

10 labels
50,000 training images
10,000 test images.



For every test image (first column), examples of nearest neighbors in rows



L1 distance: $$d_1(I_1, I_2) = \sum_p \left| I_1^p - I_2^p \right|$$

Sum up the difference in all p dimensions

test image

56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

-

training image

10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences

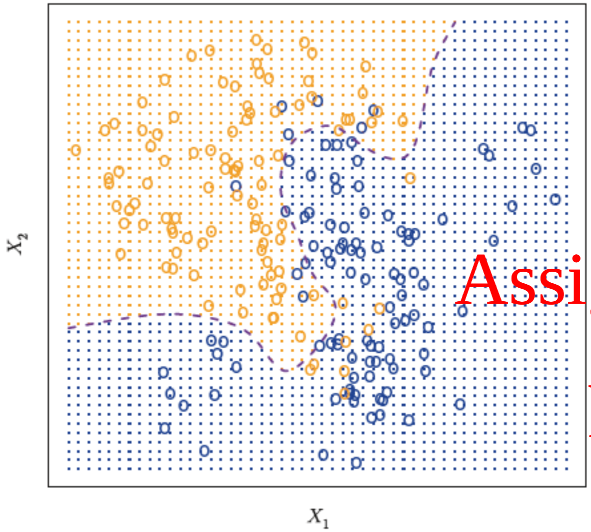
46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

add → 456

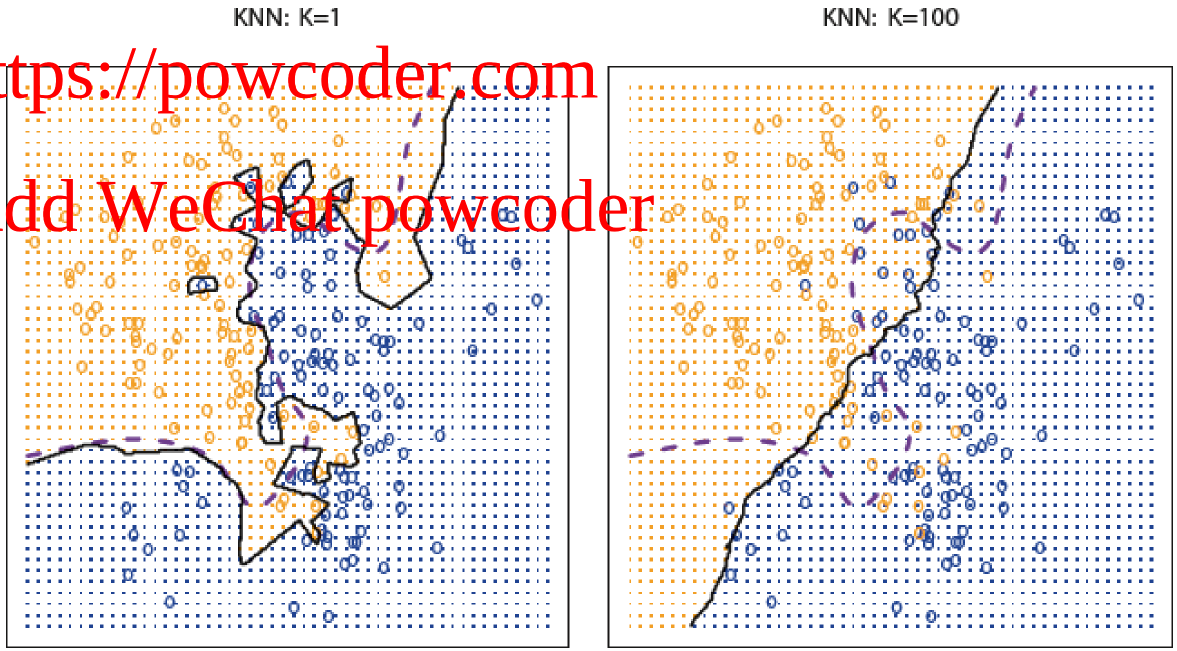
Nearest neighbor Classification…

- k-NN classifiers are lazy learners since they do not build models explicitly
- Classifying unknown records are relatively expensive
- Can produce arbitrarily shaped decision boundaries
- Easy to handle variable interactions since the decisions are based on local information
- Selection of right proximity measure is essential
- Superfluous or redundant attributes can create problems
- Missing attributes are hard to handle

Decision boundary for different k values



Small K → more flexible, possibly high test error (over fitting) more sensitive to noise



Improving KNN Efficiency

- Avoid having to compute distance to all objects in the training set
  - Fast approximate similarity search
  - Locality Sensitive Hashing (LSH)
- Condensing
  - Determine a smaller set of objects that give the same performance
- Editing
  - Remove objects to improve efficiency

Alternative classifiers: Bayes Classifier

- A probabilistic framework for solving classification problems

Bayes Classifier

- A probabilistic framework for solving classification problems
- Given:
  - A doctor knows that meningitis causes stiff neck 50% of the time
  - Prior probability of any patient having meningitis is 1/50,000
  - Prior probability of any patient having stiff neck is 1/20
- In-class exercise problem 2: If a patient has stiff neck, what's the probability he/she has meningitis? (see Canvas)

Meningitis is an acute inflammation of the protective membranes covering the brain and spinal cord, known collectively as the meninges. The most common symptoms are fever, headache, and neck stiffness.

Using Bayes Theorem for Classification

- Consider each attribute and class label as random variables

- Given a record with attributes (X_1, X_2,…, X_d)
  - Goal is to predict class Y
  - Specifically, we want to find the value of Y that maximizes P(Y| X_1, X_2,…, X_d)

- Can we estimate P(Y| X_1, X_2,…, X_d) directly from data?

Probability review

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder