CEFET/RJ

Classification

Eduardo Ogasawara
eduardo.ogasawara@cefet-rj.br
https://eic.cefet-rj.br/~eogasawara

Supervised vs. Unsupervised Learning

- Supervised learning (prediction)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Prediction Problems: Classification vs. Regression

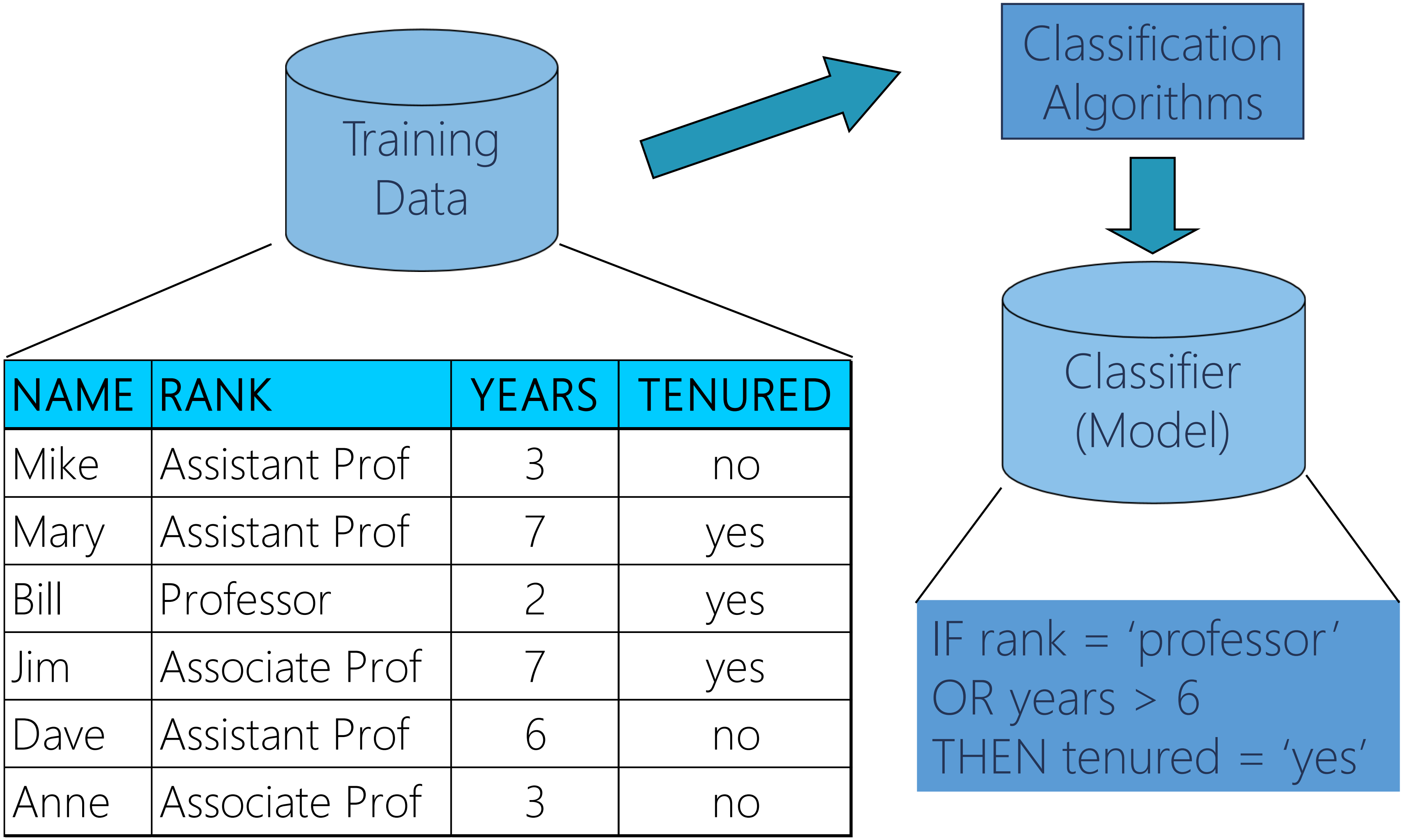
- Classification
  - predicts categorical class labels (discrete or nominal)
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Regression
  - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical applications
  - Credit/loan approval
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classification - A Two-Step Process

- Model construction: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formula
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data
- Note: During training, if necessary to select models, a slice of the training set is separated for validation (validation set)

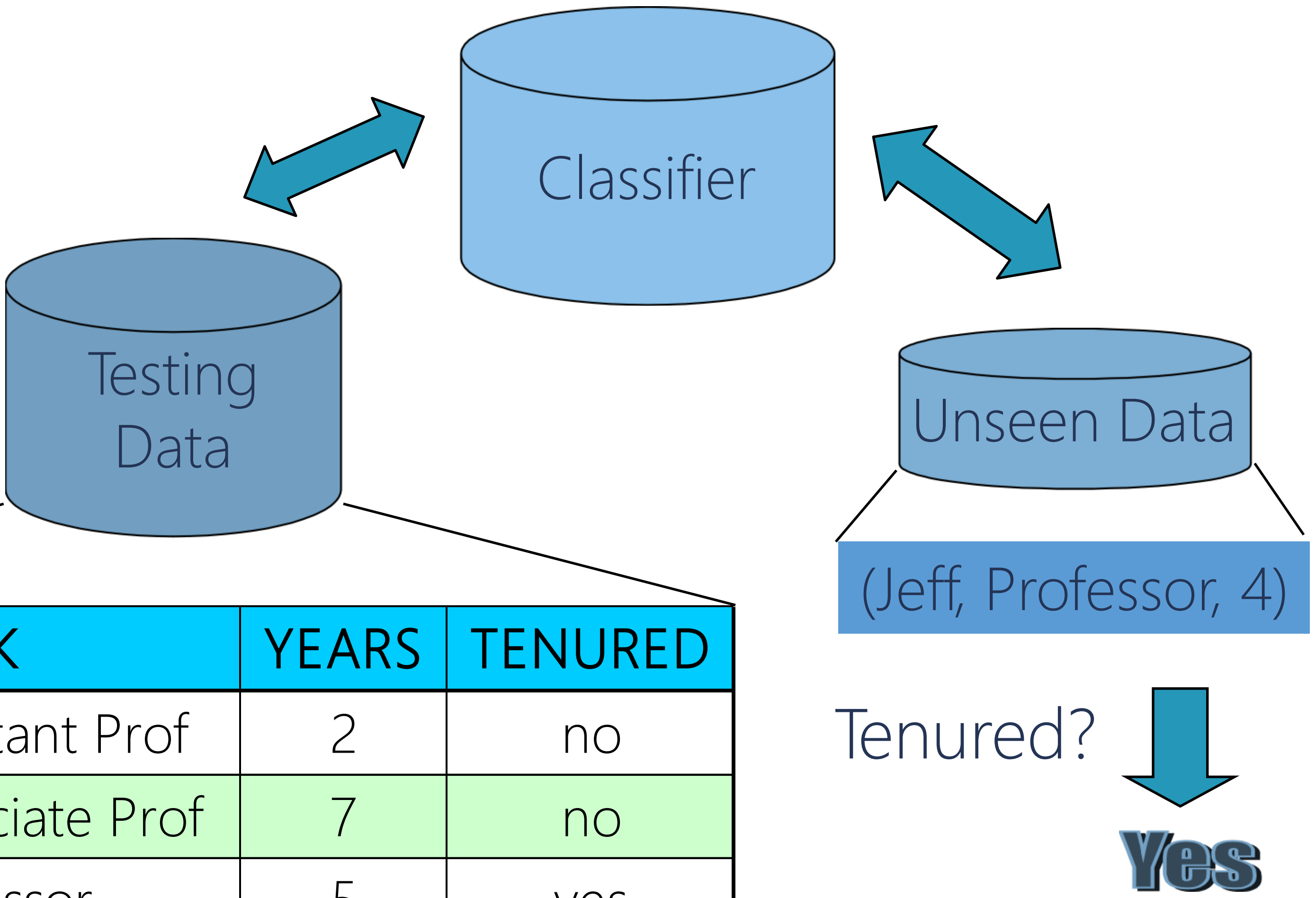
Model Training


Training Data
Classification Algorithms
Classifier (Model)
IF rank = 'professor' OR years > 6 THEN tenured = 'yes'

NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

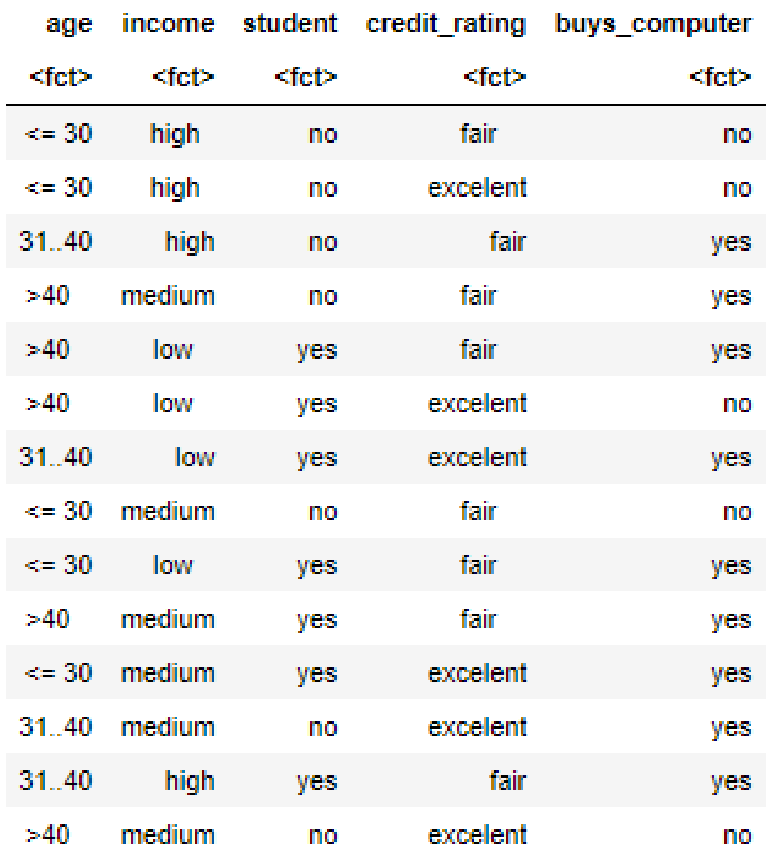
Model Usage for Prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Zero Rule – Baseline Method

- Majority class
  - Buys_computer
    - No: 5
    - Yes: 9

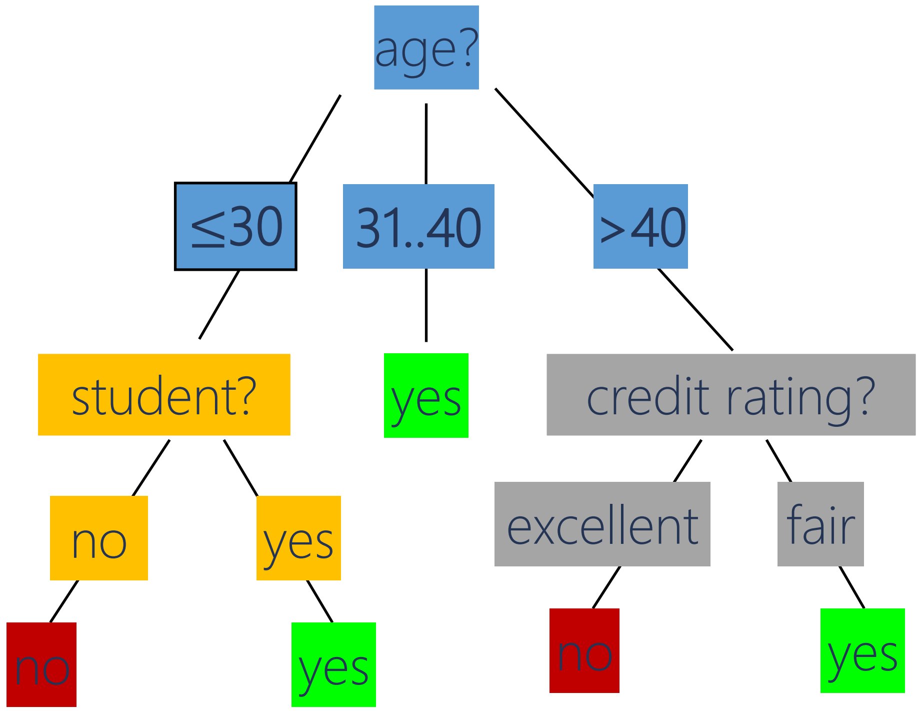


age	income	student	credit_rating	buys_computer
<fct>	<fct>	<fct>	<fct>	<fct>
<= 30	high	no	fair	no
<= 30	high	no	excelent	no
31..40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excelent	no
31..40	low	yes	excelent	yes
<= 30	medium	no	fair	no
<= 30	low	yes	fair	yes
>40	medium	yes	fair	yes
<= 30	medium	yes	excelent	yes
31..40	medium	no	excelent	yes
31..40	high	yes	fair	yes
>40	medium	no	excelent	no

[1] I.H. Witten, E. Frank, M.A. Hall, and C.J. Pal, 2016, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

Decision Tree Induction

- A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails)
  - Each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes)
- It is one way to display an algorithm that only contains conditional control statements



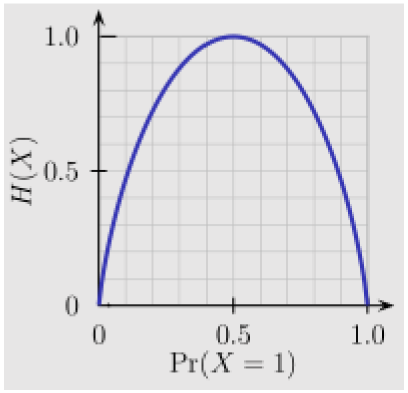
Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected based on a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Brief Review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random variable
  - Calculation: for a discrete random variable Y taking m distinct values $\{y1, \dots, ym\}$
  - $H(Y) = \sum_{i=1}^{m} pi \log(pi)$, where $pi = P(Y = yi)$
  - Interpretation
    - Higher entropy → higher uncertainty
    - Lower entropy → lower uncertainty
- Conditional Entropy
  - $H(Y|X) = \sum_x p(x)\, H(Y|X = x)$

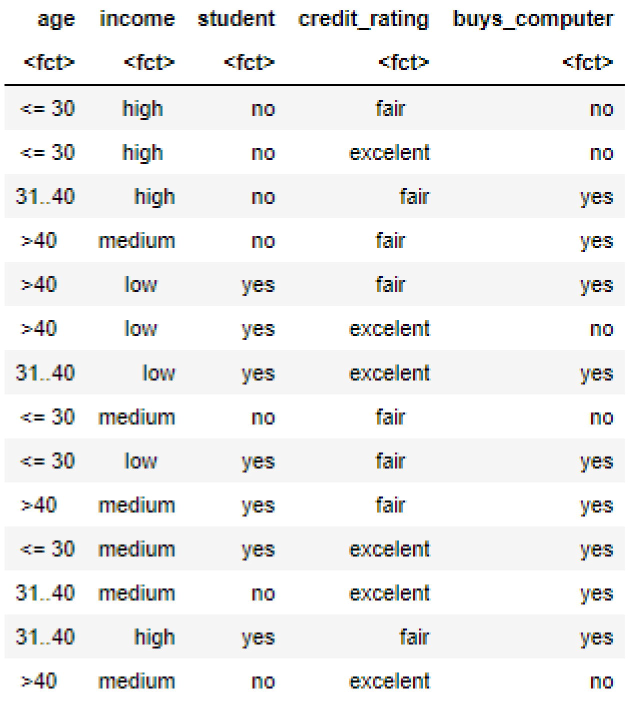


Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let pi be the probability that an arbitrary tuple in D belongs to class Ci, estimated by $\frac{|Ci,D|}{|D|}$
- Expected information (entropy) needed to classify a tuple in D:
  - $Info(D) = -\sum_{i=1}^{m} pi \log_2(p_i)$
- Information needed (after using A to split D into v partitions) to classify D:
  - $Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \cdot Info(D_j)$
- Information gained by branching on attribute A
  - $Gain(A) = Info(D) - Info_A(D)$

Attribute Selection: Information Gain

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971



age	income	student	credit_rating	buys_computer
<fct>	<fct>	<fct>	<fct>	<fct>
<= 30	high	no	fair	no
<= 30	high	no	excelent	no
31..40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excelent	no
31..40	low	yes	excelent	yes
<= 30	medium	no	fair	no
<= 30	low	yes	fair	yes
>40	medium	yes	fair	yes
<= 30	medium	yes	excelent	yes
31..40	medium	no	excelent	yes
31..40	high	yes	fair	yes
>40	medium	no	excelent	no

Class P: buys_computer = "yes"
Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.940$$

Evaluating age attribute

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$
$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Evaluating other attributes

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Pre-pruning: Halt tree construction early - do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Post-pruning: Remove branches from a "fully grown" tree - get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

Other measures

- Gain ratio
  - $SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \cdot log_2(\frac{|D_j|}{|D|})$
  - $GainRatio_A(D) = \frac{Gain_A(D)}{SplitInfo_A(D)}$
  - The attribute with the maximum gain ratio is selected as the splitting attribute
- Comparison
  - Information gain:
    - biased towards multivalued attributes
  - Gain ratio:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others

Enhancements to Basic Decision Tree Induction

- Enable continuous-valued attributes
  - Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals
- Handle missing attribute values
  - Assign the most common value of the attribute
  - Assign probability to each of the possible values
- Attribute construction
  - Create new attributes based on existing ones that are sparsely represented
  - This reduces fragmentation, repetition, and replication

Classification in Large Databases

- Classification: a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why is decision tree induction popular?
  - relatively faster learning speed (than other classification methods)
  - convertible to simple and easy to understand classification rules
  - can use SQL queries for accessing databases
  - comparable classification accuracy with other methods

Bayesian Classification

- A statistical classifier: performs probabilistic prediction, i.e., predicts class membership probabilities
- Foundation: Based on Bayes' Theorem
- Performance: A simple Bayesian classifier, naïve Bayesian classifier, has comparable performance with decision tree and selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Bayes' Theorem: Basics

- Total probability Theorem:
  - $P(B) = \sum_{i=1}^{m} P(B|A_i)P(A_i)$
- Bayes' Theorem:
  - $P(H|X) = \frac{P(X|H)P(H)}{P(X)}$
  - Let X be a data sample ("evidence"): class label is unknown
  - Let H be a hypothesis that X belongs to class C
  - Classification is to determine $P(H|X)$, (i.e., posteriori probability): the probability that the hypothesis holds given the observed data sample X
  - $P(H)$ (prior probability): the initial probability
    - E.g.: X will buy computer, regardless of age, income, …
  - $P(X)$: probability that sample data is observed
  - $P(X|H)$ (likelihood): the probability of observing the sample X, given that the hypothesis holds
    - E.g.: Given that X will buy computer, the prob. that X is 31..40, medium income

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Prediction Based on Bayes' Theorem

- Given training data X, posteriori probability of a hypothesis H, $P(H|X)$, follows the Bayes' theorem
- $P(H|X) = \frac{P(X|H)P(H)}{P(X)}$
- Informally, this can be viewed as
  - $posteriori = \frac{likehood \cdot prior}{evidence}$
- Predicts X belongs to C_i iff the probability $P(C_i|X)$ is the highest among all the $P(C_k|X)$ for all the k classes
- Practical difficulty: It requires initial knowledge of many probabilities, involving significant computational cost

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classification Is to Derive the Maximum Posteriori

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector $X = (x_1, x_2, \dots, x_n)$
- Suppose there are m classes $C_1, C_2, \dots, C_m$
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|X)$
- This can be derived from Bayes' theorem
  - $P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Naïve Bayes Classifier

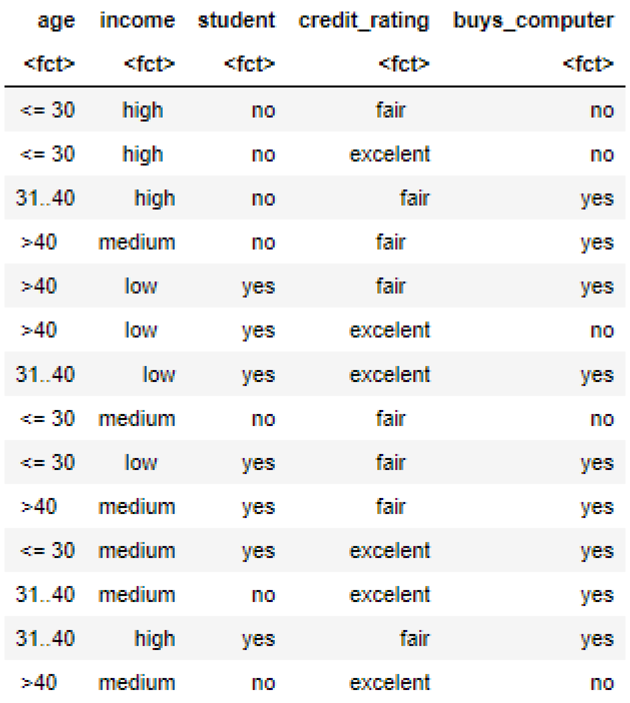
- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):
  - $P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \cdot P(x_2|C_i) \cdot \cdots \cdot P(x_n|C_i)$
- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k|C_i)$ is the number of tuples in C_i having value$x_k$for$A_k$divided by $|C_i, D|$ (number of tuples of Ci in D)
- If$A_k$is continuous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ
  - $g(x,\mu,\sigma)= \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
  - and $P(x_k|C_i)$ is $P(X|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Naïve Bayes Classifier: Training Dataset

- Class:
  - C_1:buys_computer = 'yes'
  - C_2:buys_computer = 'no'

- Data to be classified:
  - X = (age <=30,
  - Income = medium,
  - Student = yes
  - Credit_rating = Fair)

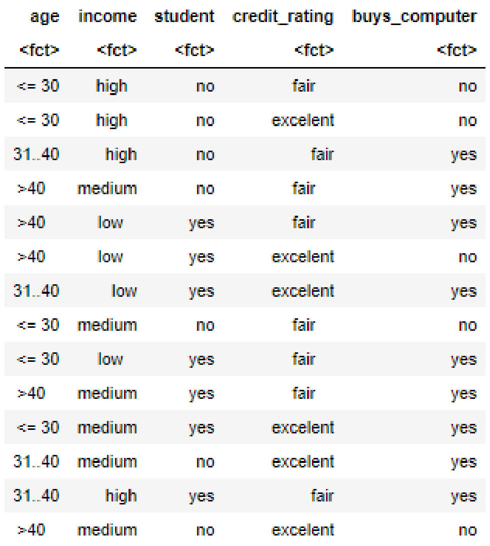


age	income	student	credit_rating	buys_computer
<fct>	<fct>	<fct>	<fct>	<fct>
<= 30	high	no	fair	no
<= 30	high	no	excelent	no
31..40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excelent	no
31..40	low	yes	excelent	yes
<= 30	medium	no	fair	no
<= 30	low	yes	fair	yes
>40	medium	yes	fair	yes
<= 30	medium	yes	excelent	yes
31..40	medium	no	excelent	yes
31..40	high	yes	fair	yes
>40	medium	no	excelent	no

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Naïve Bayes Classifier: An Example

- $P(C_i)$:
  - P(buys_computer = “yes”) = 9/14 = 0.64
  - P(buys_computer = “no”) = 5/14= 0.36
- Compute $P(X|C_i)$ for each class
  - P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.22
  - P(age = “<= 30” | buys_computer = “no”) = 3/5 = 0.60
  - P(income = “medium” | buys_computer = “yes”) = 4/9 = 0.44
  - P(income = “medium” | buys_computer = “no”) = 2/5 = 0.40
  - P(student = “yes” | buys_computer = “yes) = 6/9 = 0.67
  - P(student = “yes” | buys_computer = “no”) = 1/5 = 0.20
  - P(credit_rating = “fair” | buys_computer = “yes”) = 6/9 = 0.67
  - P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.40
- X = (age <= 30 , income = medium, student = yes, credit_rating = fair)
- P($X|C_i$) :
  - P(X|buys_computer = “yes”) = 0.22 x 0.44 x 0.67 x 0.67 = 0.044
  - P(X|buys_computer = “no”) = 0.60 x 0.40 x 0.20 x 0.40 = 0.019
- P($X|C_i$)*P(C_i):
  - P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028
  - P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007
- Therefore, X belongs to class (“buys_computer = yes”)



age	income	student	credit_rating	buys_computer
<fct>	<fct>	<fct>	<fct>	<fct>
<= 30	high	no	fair	no
<= 30	high	no	excelent	no
31..40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excelent	no
31..40	low	yes	excelent	yes
<= 30	medium	no	fair	no
<= 30	low	yes	fair	yes
>40	medium	yes	fair	yes
<= 30	medium	yes	excelent	yes
31..40	medium	no	excelent	yes
31..40	high	yes	fair	yes
>40	medium	no	excelent	no

Naïve Bayes Classifier: Comments

- Advantages
  - Easy to implement
  - Good results are obtained in most of the cases
- Disadvantages
  - Assumption: class conditional independence, therefore loss of accuracy
  - Practically, dependencies exist among variables

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Model Evaluation and Selection

- Evaluation metrics:
  - How can we measure accuracy?
  - Other metrics to consider?
- Use validation test set of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier's accuracy:
  - Holdout method, random subsampling
  - Cross-validation
  - Bootstrap
- Comparing classifiers:
  - Confidence intervals
  - Cost-benefit analysis and ROC Curves

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classifier Evaluation Metrics: Confusion Matrix

Example of Confusion Matrix (CM):

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	6954	46	7000
buy_computer = no	412	2588	3000
Total	7366	2634	10000

Confusion Matrix (CM):

Actual class\Predicted class	C_1	$\neg C_1$
C_1	True Positives (TP)	False Negatives (FN)
$\neg C_1$	False Positives (FP)	True Negatives (TN)

Given m classes, an entry, $CM_{i,j}$ in a confusion matrix indicates number of tuples in class i that were labeled by the classifier as class j
May have extra rows/columns to provide totals

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

- Classifier Accuracy: percentage of test set tuples that are correctly classified
  - $Accurary\ = \frac{TP+TN}{All}$
- Error rate:
  - $Error\ rate\ = \frac{FP+FN}{All}$
  - $= 1 - Accurary$

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P′	N′	All

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- Precision: exactness
  - % of tuples that the classifier labeled as positive are actually positive
  - $precision = \frac{TP}{TP+FP}$
- Recall: completeness
  - % of positive tuples did the classifier label as positive?
  - $recall = \frac{TP}{TP+FN}$

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P′	N′	All

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- F measure (F_1): harmonic mean of precision and recall
  - $F = \frac{2 \cdot precision \cdot recall}{precision + recall}$
- F_β: weighted measure of precision and recall
  - $F_\beta = \frac{(1+\beta^2) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall}$
  - assigns β times as much weight to recall as to precision

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P'	N'	All

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

- Class Imbalance Problem:
  - One class may be rare, e.g. fraud, or HIV-positive
  - Significant majority of the negative class and minority of the positive class
  - Sensitivity: True Positive recognition rate
    - $Sensitivity = \frac{TP}{P}$
    - $recall = sensitivity$
  - Specificity: True Negative recognition rate
    - $Specificity = \frac{TN}{N}$

A\P	C	¬C	
C	TP	FN	P
¬C	FP	TN	N
	P′	N′	All

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Classifier Evaluation Metrics: Example

- Precision = 90/230 = 39.13%
- Recall = 90/300 = 30.00%

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	90	210	300	30.00 (*sensitivity*)
cancer = no	140	9560	9700	98.56 (*specificity*)
Total	230	9770	10000	96.40 (*accuracy*)

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- Holdout method
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- Cross-validation (k-fold, where k = 10 is most popular)
  - Randomly partition the data into k mutually exclusive subsets, each approximately equal size
  - At i-th iteration, use Di as test set and others as training set
  - Leave-one-out: k folds where k = # of tuples, for small sized data
  - *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Evaluating Classifier Accuracy: Bootstrap

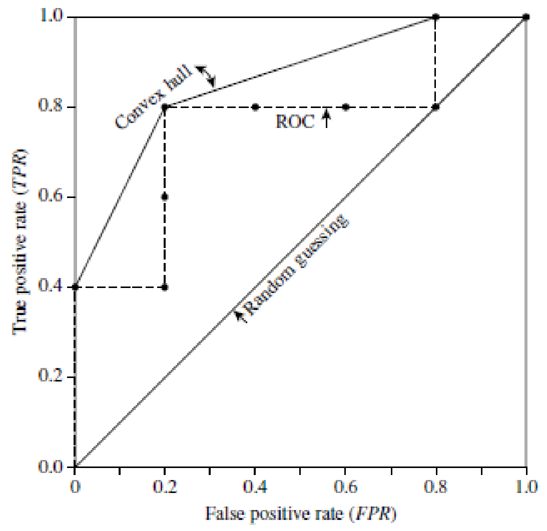
- Bootstrap
  - Works well with small data sets
  - Samples the given training tuples uniformly with replacement
    - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is 0.632 boostrap
  - A data set with d tuples is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $\left(1-\frac{1}{d}\right)d \approx e^{-1} = 0.368$)
  - Repeat the sampling procedure k times, overall accuracy of the model:
- $Acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \cdot Acc(M_i)_{test} + 0.368 \cdot Acc(M_i)_{train})$

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Model Selection: ROC Curves

- ROC (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list rate
- The area under the ROC curve is a measure of the accuracy of the model
    - Vertical axis represents the true positive rate
    - Horizontal axis represents the false positive
- The plot also shows a diagonal line
    - The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model
- A model with perfect accuracy will have an area of 1.0

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	P	0.90	1	0	5	4	0.2	0
2	P	0.80	2	0	5	3	0.4	0
3	N	0.70	2	1	4	3	0.4	0.2
4	P	0.60	3	1	4	2	0.6	0.2
5	P	0.55	4	1	4	1	0.8	0.2
6	N	0.54	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.51	4	4	1	1	0.8	0.8
9	P	0.50	5	4	0	1	1.0	0.8
10	N	0.40	5	5	0	0	1.0	1.0



Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- Loss function: measures the error between y_i and the predicted value y_i'
  - Absolute error: $|y_i - y_i'|$
  - Squared error:$(y_i - y_i')^2$
- Test error (generalization error): the average loss over the test set
  - Mean absolute error: $\frac{\sum_{i=1}^{d} |y_i - y_i'|}{d}$
  - Mean squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$
  - Relative absolute error: $\frac{\sum_{i=1}^{d} |y_i - y_i'|}{\sum_{i=1}^{d} |y_i - \bar{y}|}$
  - Relative squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$
  - The mean squared-error exaggerates the presence of outliers
    - Popularly use (square) root mean-square error, similarly, root relative squared error

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

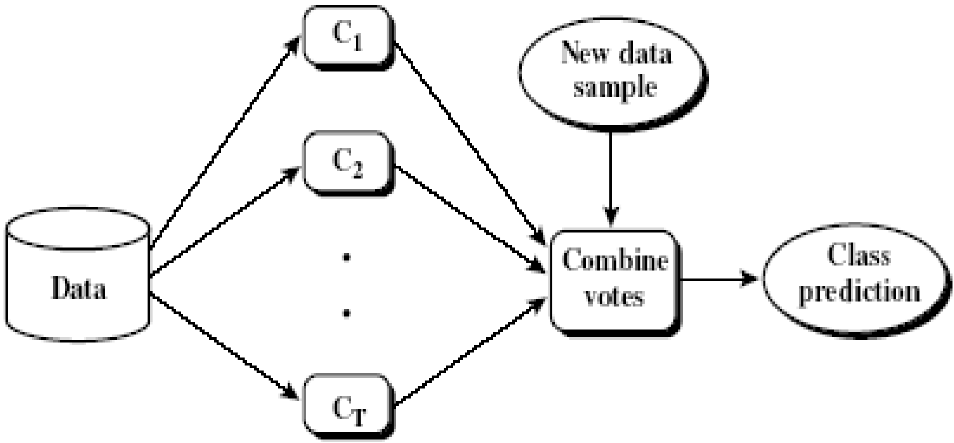
Issues Affecting Model Selection

- Accuracy
  - classifier accuracy: predicting class label
- Speed
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- Robustness: handling noise and missing values
- Scalability: efficiency in disk-resident databases
- Interpretability
  - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Ensemble Methods: Increasing the Accuracy

- Ensemble methods
  - Use a combination of models to increase accuracy
  - Combine a series of T learned models, $C_1, C_2, \ldots, C_T$, with the aim of creating an improved model C*
- Popular ensemble methods
  - Bagging: averaging the prediction over a collection of classifiers
  - Boosting: weighted vote with a collection of classifiers
  - Ensemble: combining a set of heterogeneous classifiers



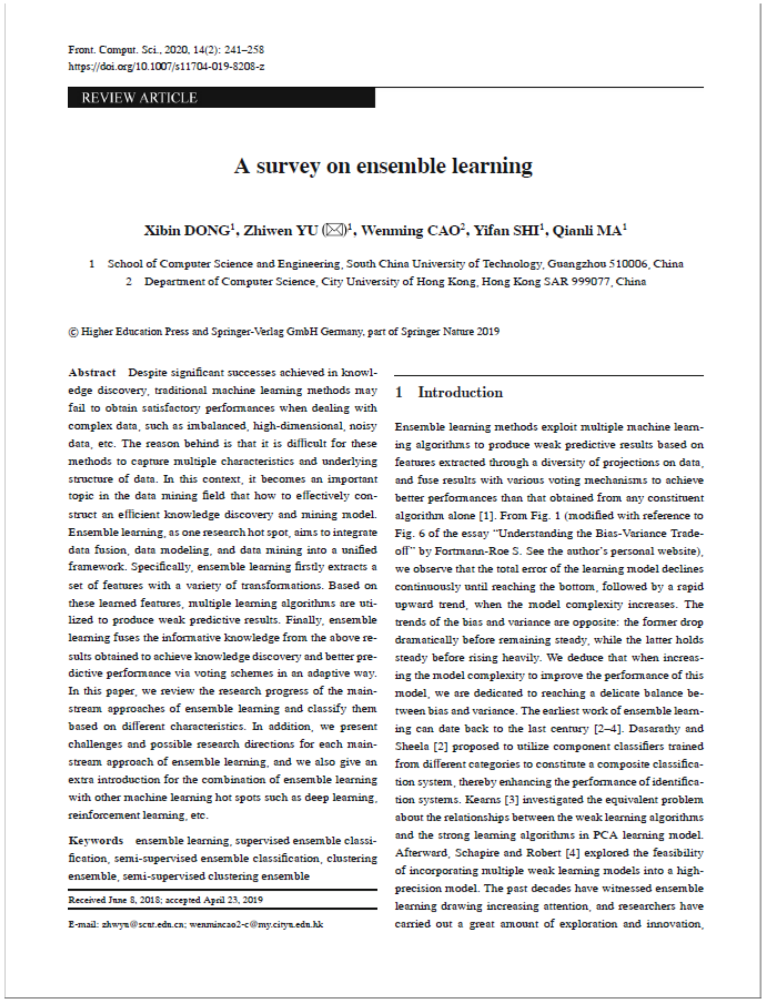
[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Random Forest

- Random Forest
  - Each classifier in the ensemble is a decision tree classifier and is generated using a random selection of attributes at each node to determine the split
  - During classification, each tree votes and the most popular class is returned
- Two Methods to construct Random Forest:
  - Forest-RI (random input selection): Randomly select, at each node, F attributes as candidates for the split at the node. The CART methodology is used to grow the trees to maximum size
  - Forest-RC (random linear combinations): Creates new attributes (or features) that are a linear combination of the existing attributes (reduces the correlation between individual classifiers)
- Insensitive to the number of attributes selected for consideration at each split, and faster than bagging or boosting

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Reading papers



Front. Comput. Sci., 2020, 14(2): 241–258
https://doi.org/10.1007/s11704-019-8208-z

REVIEW ARTICLE

A survey on ensemble learning

Xibin DONG[1], Zhiwen YU (✉)[1], Wenming CAO[2], Yifan SHI[1], Qianli MA[1]

1 School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
2 Department of Computer Science, City University of Hong Kong, Hong Kong SAR 999077, China

Abstract Despite significant successes achieved in knowledge discovery, traditional machine learning methods may fail to obtain satisfactory performances when dealing with complex data, such as imbalanced, high-dimensional, noisy data, etc. The reason behind is that it is difficult for these methods to capture multiple characteristics and underlying structure of data. In this context, it becomes an important topic in the data mining field that how to effectively construct an efficient knowledge discovery and mining model. Ensemble learning, as one research hot spot, aims to integrate data fusion, data modeling, and data mining into a unified framework. Specifically, ensemble learning firstly extracts a set of features with a variety of transformations. Based on these learned features, multiple learning algorithms are utilized to produce weak predictive results. Finally, ensemble learning fuses the informative knowledge from the above results obtained to achieve knowledge discovery and better predictive performance via voting schemes in an adaptive way. In this paper, we review the research progress of the mainstream approaches of ensemble learning and classify them based on different characteristics. In addition, we present challenges and possible research directions for each mainstream approach of ensemble learning, and we also give an extra introduction for the combination of ensemble learning with other machine learning hot spots such as deep learning, reinforcement learning, etc.

Keywords ensemble learning, supervised ensemble classification, semi-supervised ensemble classification, clustering ensemble, semi-supervised clustering ensemble

Received June 8, 2018; accepted April 23, 2019

E-mail: zhwyu@scut.edu.cn; wenmincao2-c@my.cityu.edu.hk

1 Introduction

Ensemble learning methods exploit multiple machine learning algorithms to produce weak predictive results based on features extracted through a diversity of projections on data, and fuse results with various voting mechanisms to achieve better performances than that obtained from any constituent algorithm alone [1]. From Fig. 1 (modified with reference to Fig. 6 of the essay "Understanding the Bias-Variance Tradeoff" by Fortmann-Roe S. See the author's personal website), we observe that the total error of the learning model declines continuously until reaching the bottom, followed by a rapid upward trend, when the model complexity increases. The trends of the bias and variance are opposite: the former drop dramatically before remaining steady, while the latter holds steady before rising heavily. We deduce that when increasing the model complexity to improve the performance of this model, we are dedicated to reaching a delicate balance between bias and variance. The earliest work of ensemble learning can date back to the last century [2–4]. Dasarathy and Sheela [2] proposed to utilize component classifiers trained from different categories to constitute a composite classification system, thereby enhancing the performance of identification systems. Kearns [3] investigated the equivalent problem about the relationships between the weak learning algorithms and the strong learning algorithms in PCA learning model. Afterward, Schapire and Robert [4] explored the feasibility of incorporating multiple weak learning models into a high-precision model. The past decades have witnessed ensemble learning drawing increasing attention, and researchers have carried out a great amount of exploration and innovation,

Dong, X., Yu, Z., Cao, W., Shi, Y., Ma, Q., (2020), "A survey on ensemble learning", Frontiers of Computer Science, v. 14, n. 2, p. 241–258.
Breiman, L., (2001), "Random forests", Machine Learning, v. 45, n. 1, p. 5–32.

Bagging: Boostrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
  - Given a set D of d tuples, at each iteration i, a training set $D_i$of d tuples is sampled with replacement from D (i.e., bootstrap)
  - A classifier model M_i is learned for each training set D_i
- Classification: classify an unknown sample X
  - Each classifier$M_i$returns its class prediction
  - The bagged classifier M^* counts the votes and assigns the class with the most votes to X
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
  - Often significantly better than a single classifier derived from D
  - For noise data: not considerably worse, more robust
  - Proved improved accuracy in prediction

Boosting

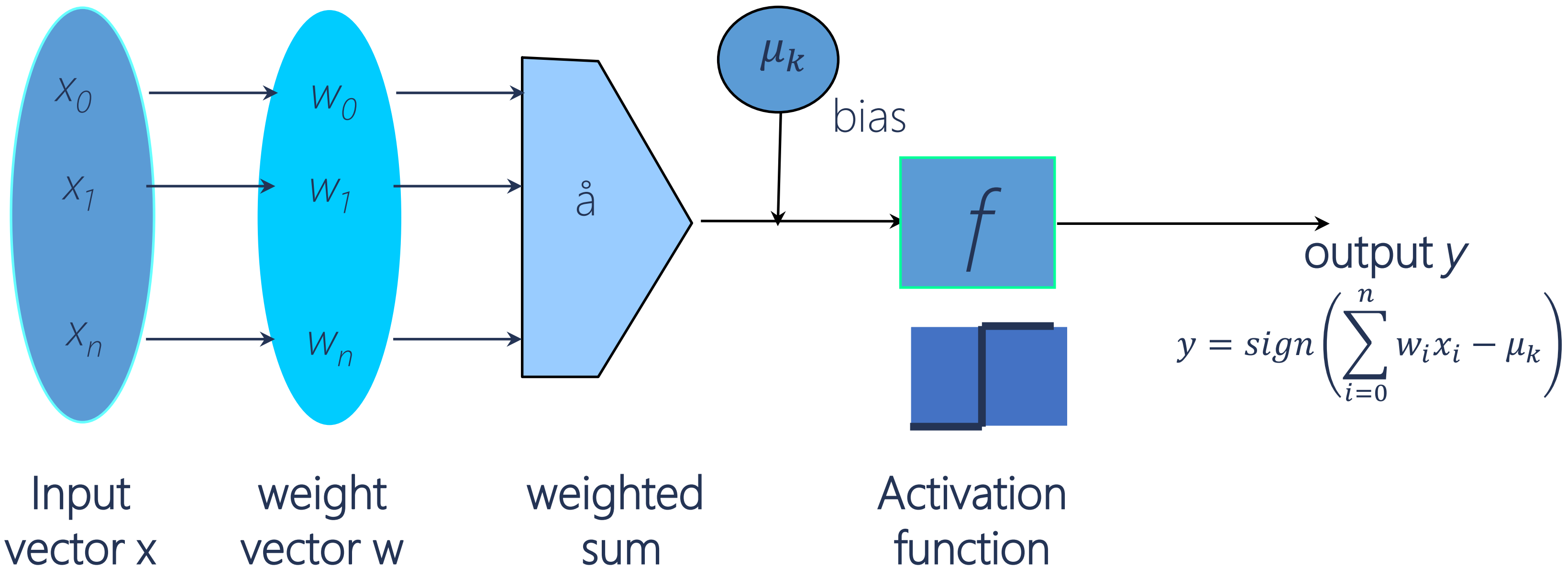
- Analogy: Consult several doctors, based on a combination of weighted diagnoses—weight assigned based on the previous diagnosis accuracy
- How boosting works?
  - Weights are assigned to each training tuple
  - A series of k classifiers is iteratively learned
  - After a classifier M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1}, to pay more attention to the training tuples that were misclassified byM_i
  - The final M^* combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy
- Boosting algorithm can be extended for numeric prediction
- Comparing with bagging: Boosting tends to have greater accuracy, but it also risks overfitting the model to misclassified data

Classification by Backpropagation

- Backpropagation: A neural network learning algorithm
- Started by psychologists and neurobiologists to develop and test computational analogues of neurons
- A neural network: A set of connected input/output units where each connection has a weight associated with it
- During the learning phase, the network learns by adjusting the weights to be able to predict the correct class label of the input tuples
- Also referred to as connectionist learning due to the connections between units

Neuron: A Hidden/Output Layer Unit

- An n-dimensional input vector x is mapped into variable y by means of the scalar product and a nonlinear function mapping
- The inputs to unit are outputs from the previous layer. They are multiplied by their corresponding weights to form a weighted sum, which is added to the bias associated with unit. Then a nonlinear activation function is applied to it.



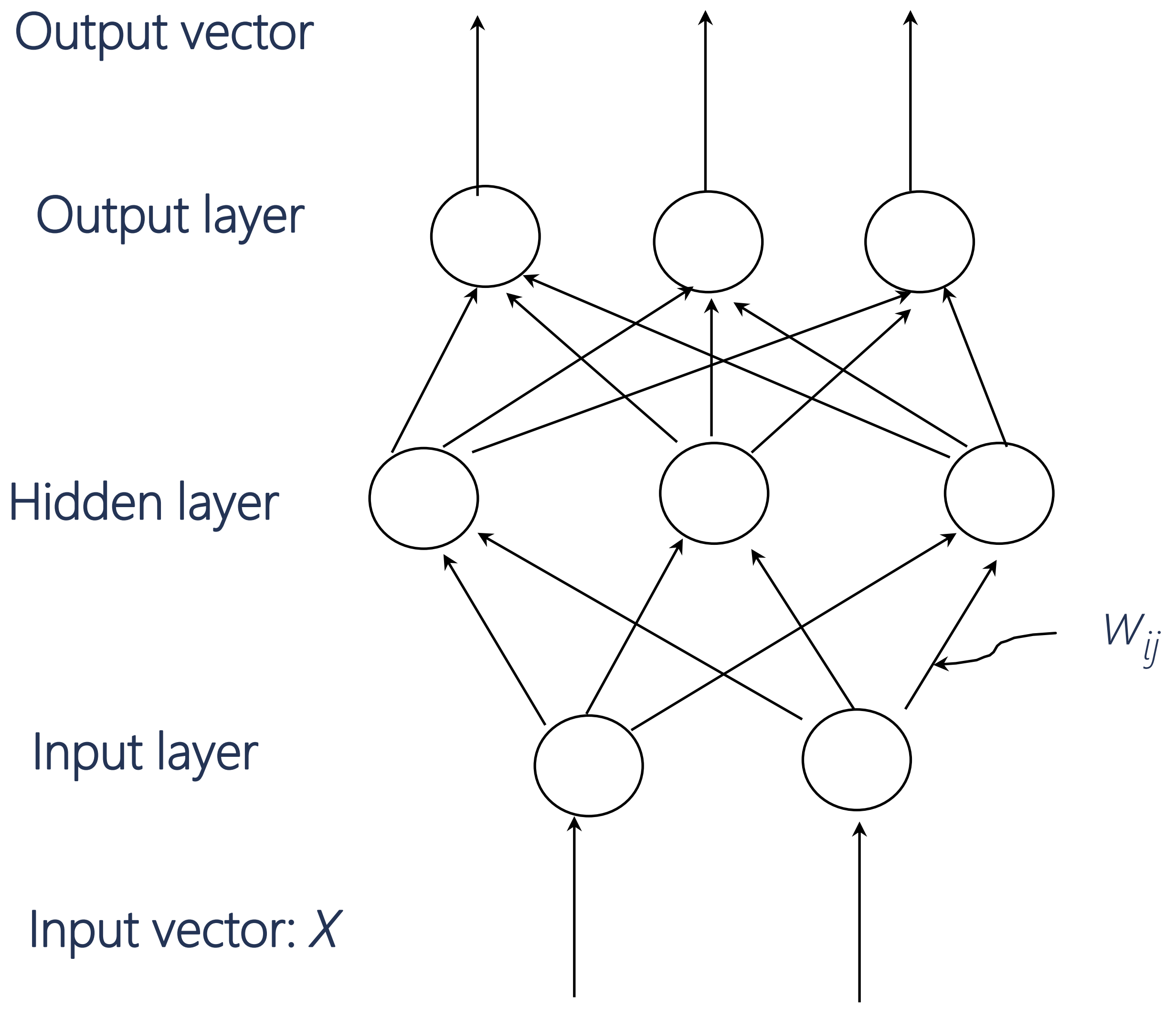
$$y = sign\left(\sum_{i=0}^{n} w_i x_i - \mu_k\right)$$

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

How A Multi-Layer Neural Network Works

- The inputs to the network correspond to the attributes measured for each training tuple
- Inputs are fed simultaneously into the units making up the input layer
- They are then weighted and fed simultaneously to a hidden layer
- The number of hidden layers is arbitrary, although usually only one
- The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction
- The network is feed-forward: None of the weights cycles back to an input unit or to an output unit of a previous layer
- From a statistical point of view, networks perform nonlinear regression: Given enough hidden units and enough training samples, they can closely approximate any function

A Multi-Layer Feed-Forward Neural Network



[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Defining a Network Topology

- Decide the network topology:
  - Specify number of units in the input layer,
  - number of hidden layers (if > 1),
  - number of units in each hidden layer
  - number of units in the output layer (regression versus classification)
- Normalize the input values for each attribute measured in the training tuples to [0.0—1.0]
- One input unit per domain value, each initialized to 0
- Output, if for classification and more than two classes, one output unit per class is used
- Once a network has been trained and its accuracy is unacceptable, repeat the training process with a different network topology or a different set of initial weights

Neural Network as a Classifier

- Weakness
  - Long training time
  - Require parameters typically best determined empirically, e.g., the network topology or “structure”
  - Poor interpretability: Difficult to interpret the symbolic meaning behind the learned weights and of “hidden units” in the network
- Strength
  - High tolerance to noisy data
  - Ability to classify untrained patterns
  - Well-suited for continuous-valued inputs and outputs
  - Successful on an array of real-world data, e.g., hand-written letters
  - Algorithms are inherently parallel
  - Techniques have recently been developed for the extraction of rules from trained neural networks

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

SVM—Support Vector Machines

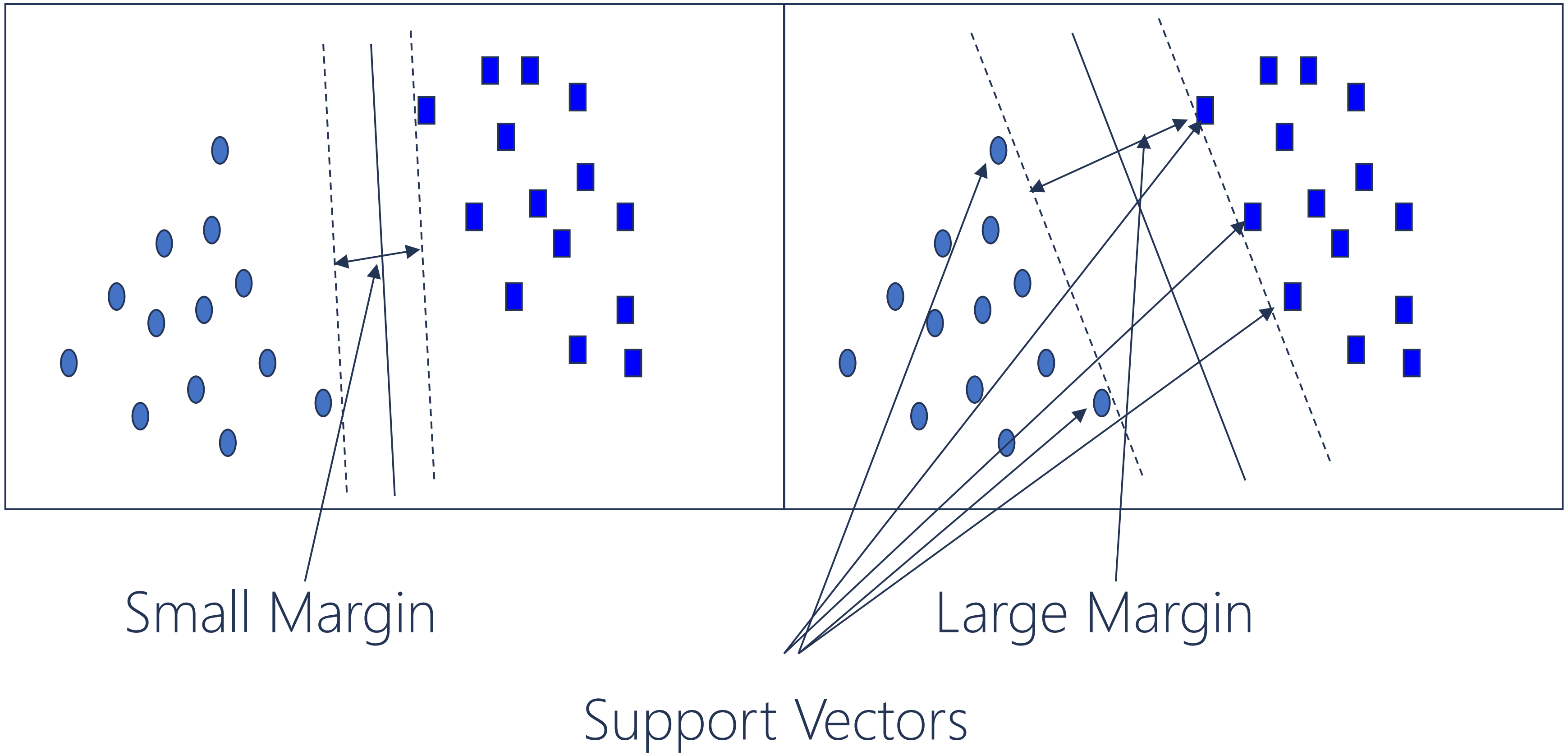
- A relatively new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using support vectors (“essential” training tuples) and margins (defined by the support vectors)

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

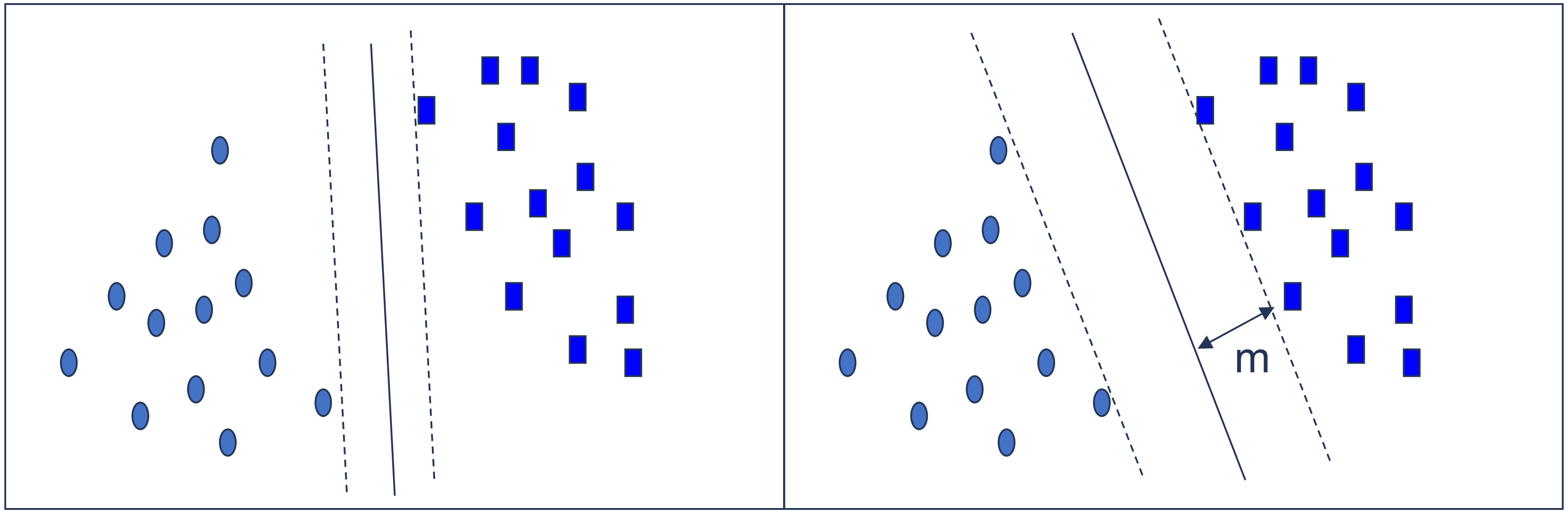
SVM—History and Applications

- Vapnik and colleagues (1992)
  - groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s
- Features: training can be slow, but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Used for: classification and numeric prediction

SVM—General Philosophy



SVM—When Data Is Linearly Separable



Let data D be (X_1,y_1), …, ($X_{|D|}$,$y_{|D|}$), where X_i are training tuples associated with the class labels y_i

There are infinite lines (hyperplanes) separating the two classes, but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane** (MMH)

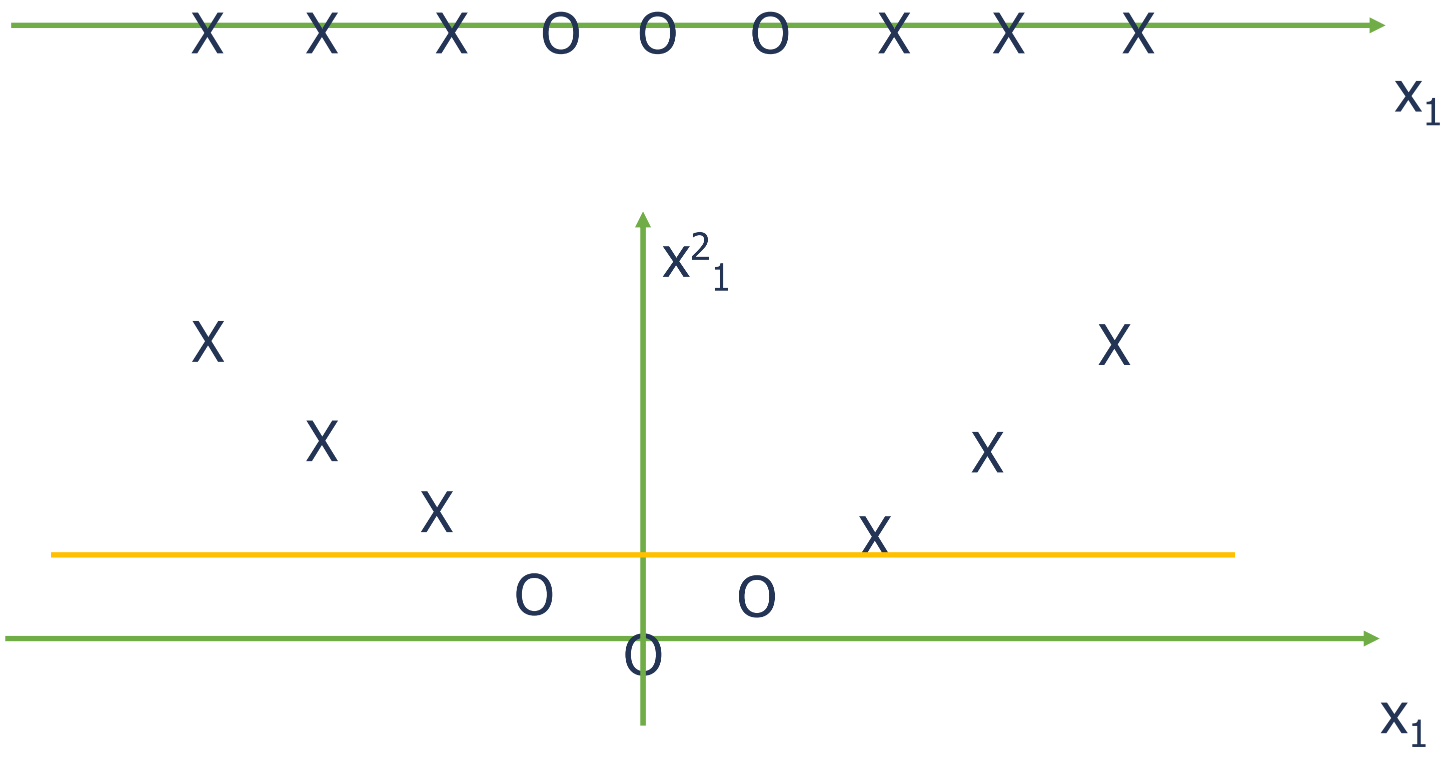
SVM—Linearly Separable

- A separating hyperplane can be written as
  - $W \cdot X + b = 0$
  - where $W = \{w_1, w_2, \ldots, w_n\}$ is a weight vector and b a scalar (bias)
- For 2-D it can be written as
  - $w_0 + w_1 x_1 + w_2 x_2 = 0$
- The hyperplane defining the sides of the margin:
  - $H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1$ for $y_i = +1$, and
  - $H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1$ for $y_i = -1$
- Any training tuples that fall on hyperplanesH_1orH_2(i.e., the sides defining the margin) are support vectors
- This becomes a constrained (convex) quadratic optimization problem:
  - Quadratic objective function and linear constraints →
  - Quadratic Programming (QP) →
  - Lagrangian multipliers

Why Is SVM Effective on High Dimensional Data?

- The complexity of trained classifier is characterized by the number of support vectors rather than the dimensionality of the data
- The support vectors are the essential or critical training examples - they lie closest to the decision boundary (MMH)
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

Use a line to separate classes



SVM - Linearly Inseparable

- Transform the original input data into a higher dimensional space
- Ex.: Consider a $3D$ input vector X (x_1, x_2, x_3) mapped into a $6D$ space using Z
  - $\phi_1(\mathrm{X}) = x_1$, $\phi_2(\mathrm{X}) = x_2$, $\phi_3(\mathrm{X}) = x_3$,
  - $\phi_4(\mathrm{X}) = {x_1}^2$, $\phi_5(\mathrm{X}) = x_1 x_2$, $\phi_6(\mathrm{X}) = x_1 x_3$
  - $d(\mathrm{Z}) = WZ + b$, where W and Z are vectors
    - This is linear
    - We solve for W and b and the substitute back so that we see that the linear decision hyperplane in the new(Z) space corresponds to a nonlinear second order polynomial in the original 3-D input space
  - $d(\mathrm{Z}) = \mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \mathrm{w}_3 x_3 + \mathrm{w}_4 {x_1}^2 + \mathrm{w}_5 x_1 x_2 + \mathrm{w}_6 x_1 x_3 + b$
  - $= \mathrm{w}_1 \mathrm{z}_1 + \mathrm{w}_2 \mathrm{z}_2 + \mathrm{w}_3 \mathrm{z}_3 + \mathrm{w}_4 \mathrm{z}_4 + \mathrm{w}_5 \mathrm{z}_5 + \mathrm{w}_6 \mathrm{z}_6 + b$
- Search for a linear separating hyperplane in the new space

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Kernel functions for Nonlinear Classification

- Instead of computing the dot product on the transformed data, it is math. equivalent to applying a kernel function $K(X_i, X_j)$ to the original data, i.e., $K(X_i, X_j) = \phi(X_i)\phi(X_j)$
- Typical Kernel Functions
  - Polynomial kernel of degree h: $K(X_i, X_j) = (X_i \cdot X_j + 1)^h$
  - Gaussian radial basis function kernel: $K(X_i, X_j) = e^{\frac{-|X_i - X_j|^2}{2\sigma^2}}$
  - Sigmoid kernel: $K(X_i, X_j) = tanh\ (kX_i \cdot X_j - \delta)$
- SVM can also be used for classifying multiple (> 2) classes and for regression analysis (with additional parameters)

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

SVM vs. Neural Network

- SVM
  - Deterministic algorithm
  - Nice generalization properties
  - Hard to learn – learned in batch mode using quadratic programming techniques
  - Using kernels can learn very complex functions

- Neural Network
  - Nondeterministic algorithm
  - Generalizes well but does not have strong mathematical foundation
  - Can easily be learned in incremental fashion
  - To learn complex functions - use multilayer perceptron (nontrivial)

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

Lazy vs. Eager Learning

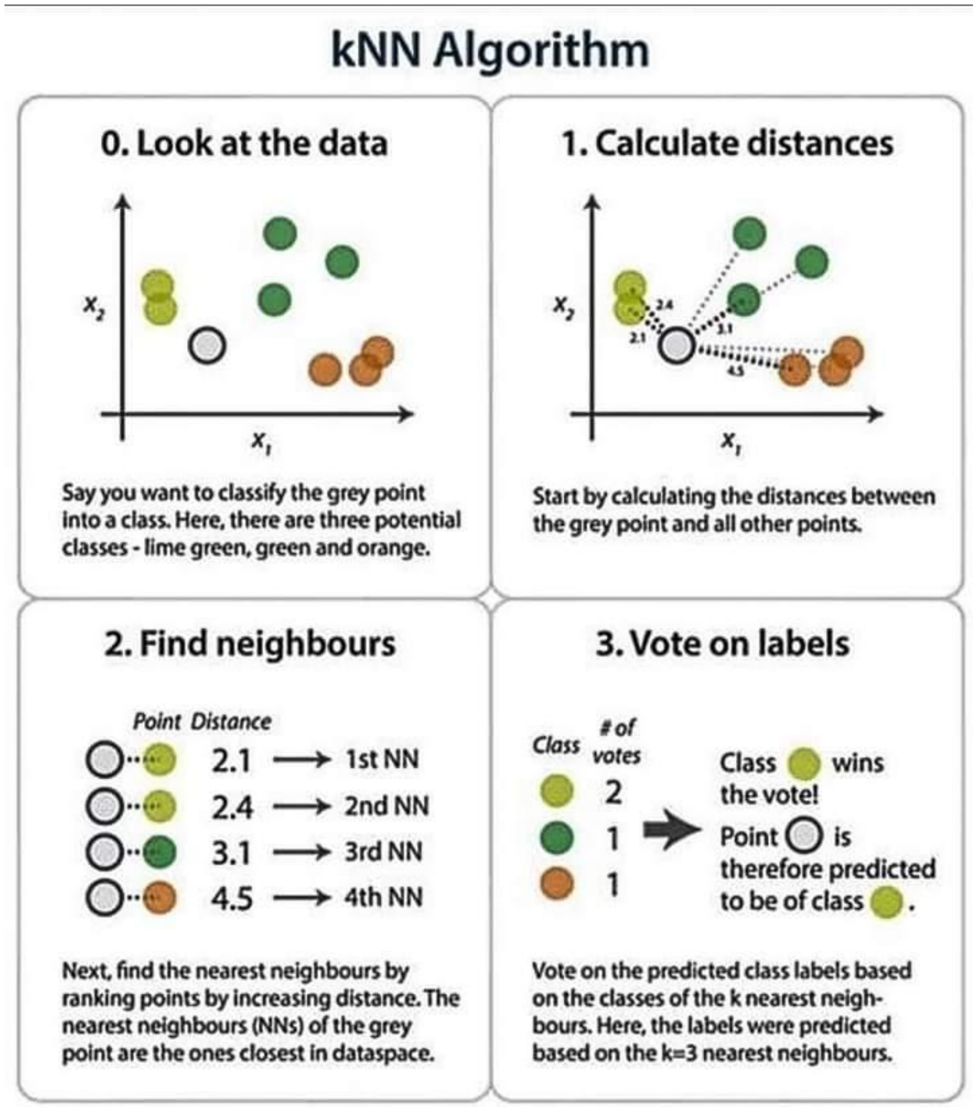
- Lazy vs. eager learning
  - Eager learning (the above discussed methods): Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify
  - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
- Lazy
  - less time in training but more time in predicting
- Accuracy
  - Lazy method: effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
  - Eager method: commits to a single hypothesis that covers the entire instance space

Lazy Learner: Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing (“lazy evaluation”) until a new instance must be classified
- Typical approaches
  - k-nearest neighbor approach
    - Instances represented as points in a Euclidean space
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022..

The k-Nearest Neighbor Algorithm



kNN Algorithm

0. Look at the data

Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances

Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point	Distance	
○··●	2.1	⟶ 1st NN
○··●	2.4	⟶ 2nd NN
○··●	3.1	⟶ 3rd NN
○··●	4.5	⟶ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

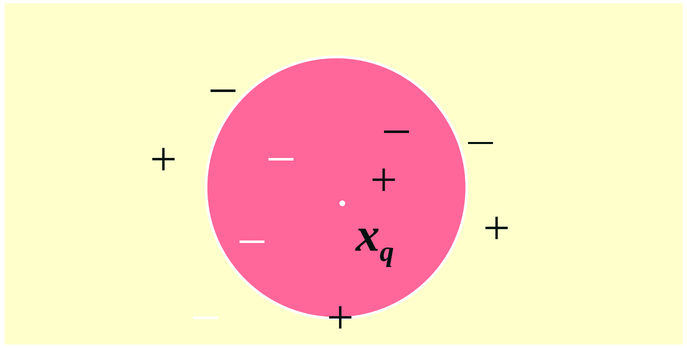
Class	# of votes
●	2
●	1
●	1

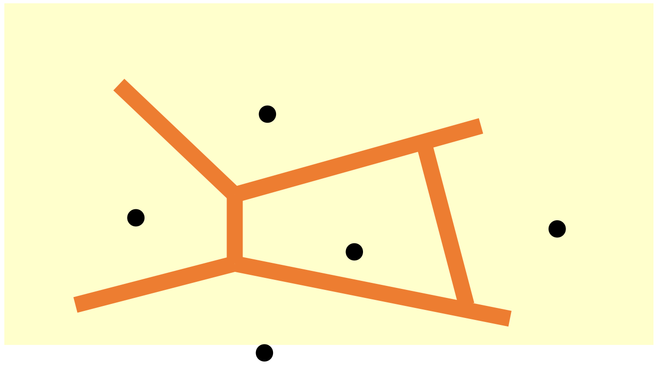
➔ Class ● wins the vote! Point ○ is therefore predicted to be of class ●.

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

The k-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist(x_i, x_j)
- Target function could be discrete-valued or real-valued
- For discrete-valued, k-NN returns the most common value among the k training examples nearest to x_q
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples





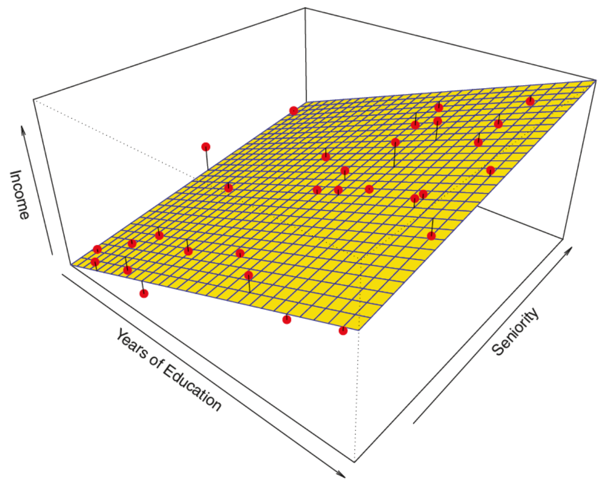
Discussion on the k-NN Algorithm

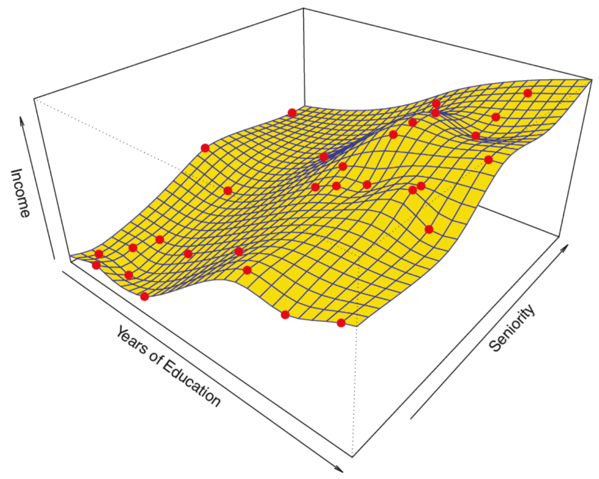
- k-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the k nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query x_q
    - Give greater weight to closer neighbors $\mathrm{w} = \frac{1}{d(x_\mathrm{q}, x_\mathrm{i})^2}$
- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Feature selection

Curse of dimensionality

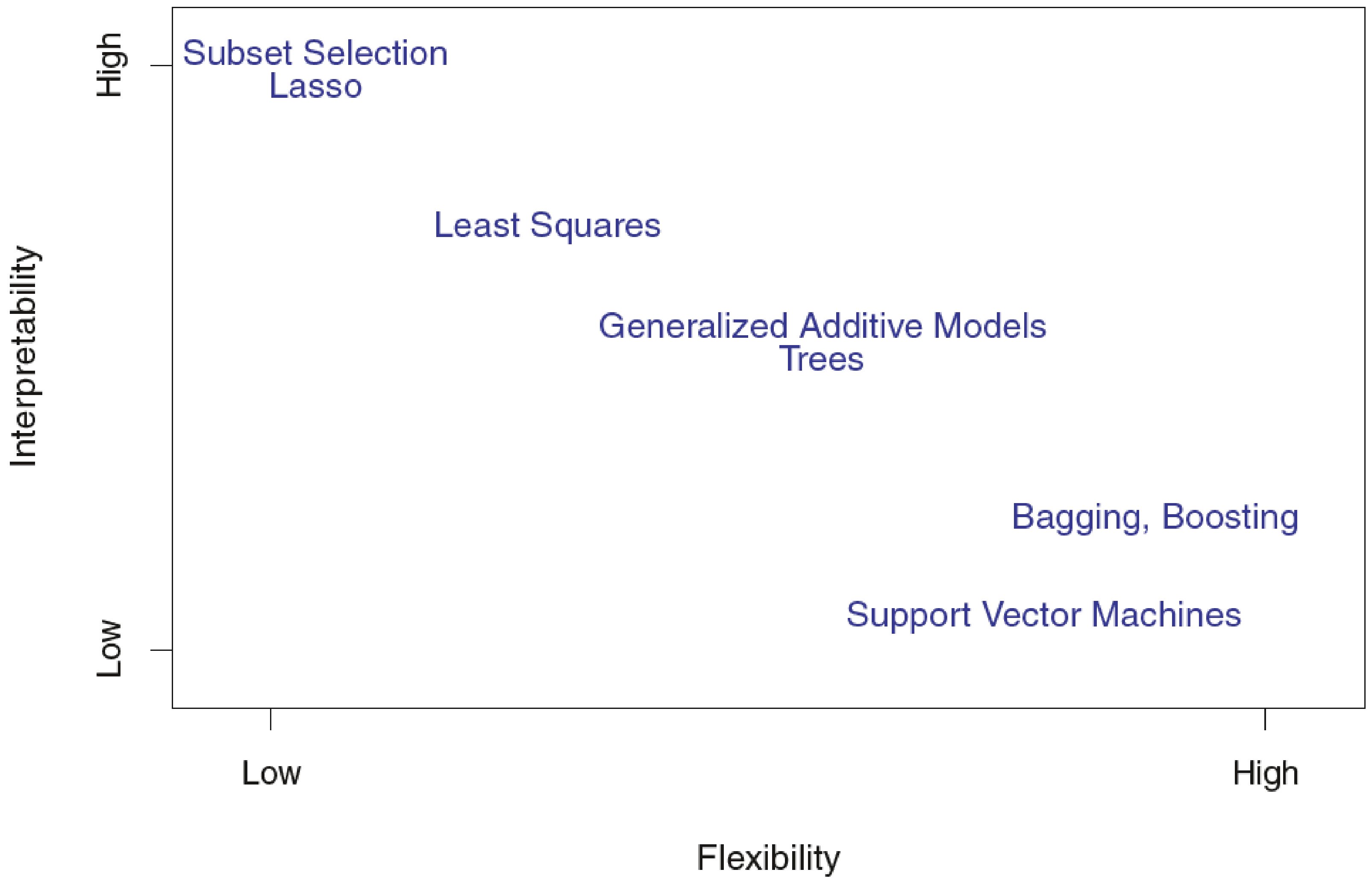
- When dimensionality increases, data becomes increasingly sparse
- Possible combinations of subspaces grow exponentially
- It interferes the density and distance between points
  - Affects grouping, prediction, and outlier analysis





James, G., Witten, D., Hastie, T., Tibshirani, R., (2013), *An Introduction to Statistical Learning: with Applications in R*. 1 ed. Springer.

Interpretability of models

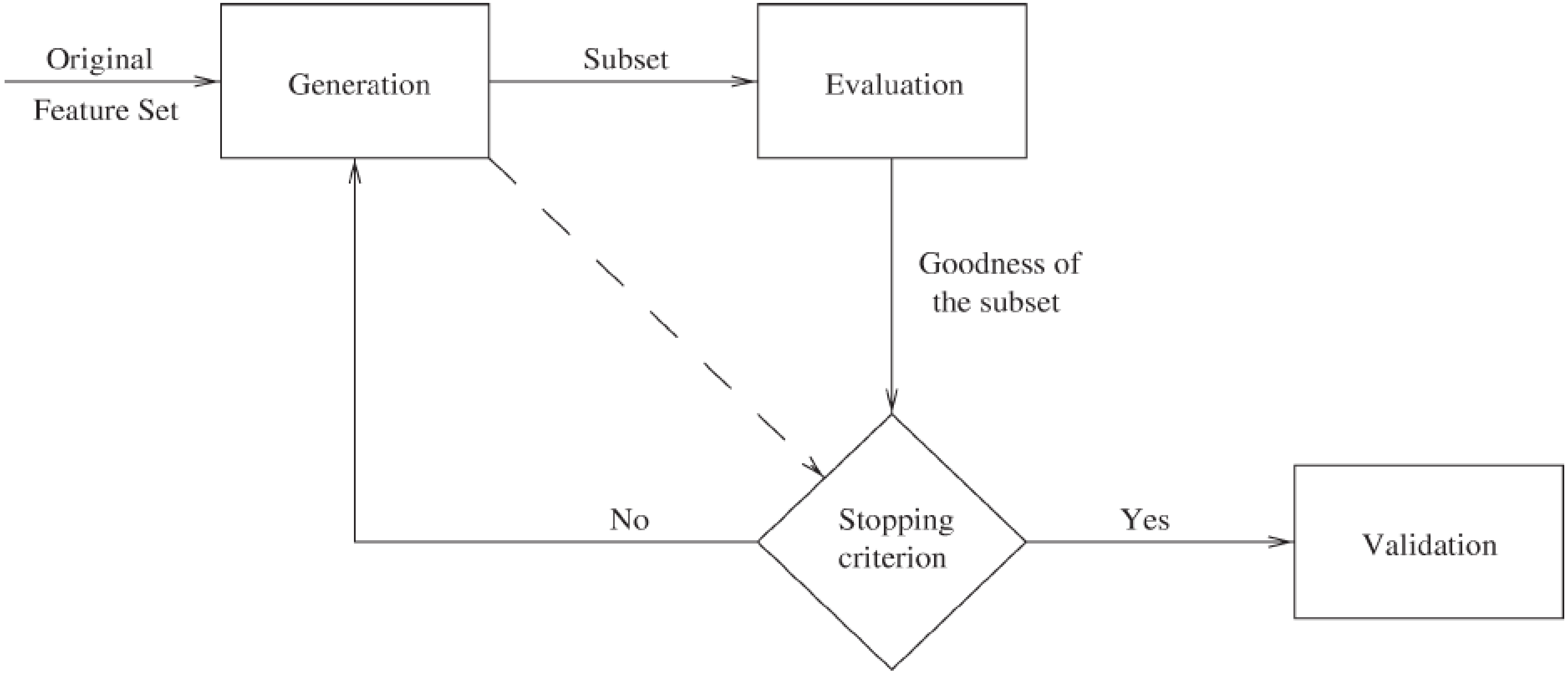


James, G., Witten, D., Hastie, T., Tibshirani, R., (2013), *An Introduction to Statistical Learning: with Applications in R*. 1 ed. Springer.

Attribute subset selection

- Redundant attributes
  - Duplicates the information contained in one or more other attributes
  - For example, the purchase price of a product and the amount of sales tax paid
- Irrelevant attributes
    - Do not contain information that is useful for the data mining task at hand
    - For example, student IDs are irrelevant to the task of predicting GPA students

Feature selection principles



Attribute selection methods

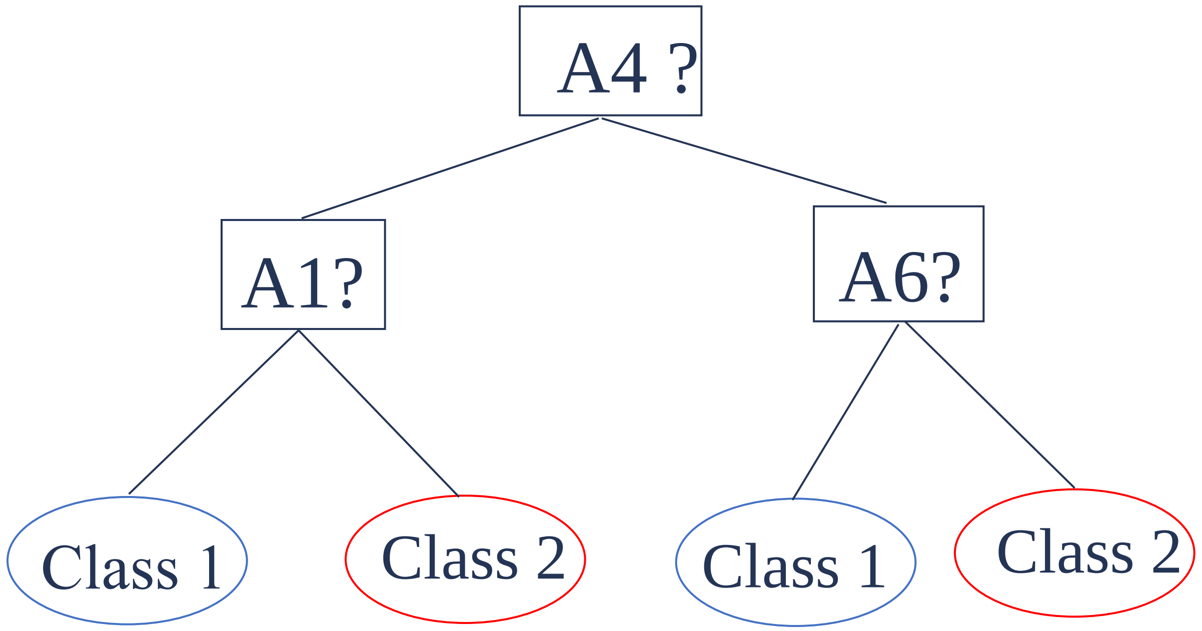
- Information Gain (INFOGAIN)
- Forward Stepwise selection (FSS)
- Least Absolute Shrinkage and Selection Operator (LASSO)
- Correlation-based Feature Selection (CFS)
- Recursive Elimination of Features (RELIEF)

Information Gain

Initial attribute set:

{A1, A2, A3, A4, A5, A6}

Approach based on information gain



------> Reduced attribute set: {A1, A4, A6}

Forward Stepwise selection

- Consider a dataset of n attributes
- Start with a null model
- Adjust n simple linear regression models
  - Set $i = 1$
  - choose p_i the one that produces the best fit
- Repeat
  - Increment the variable i
  - Adjust p_i regression models that combine the p_{i-1} attributes with an additional variable that produces the best fit
  - Measure improvement using anova-test(p_i, p_{i-1})
- Until there is no improvement in adjustment

Least Absolute Shrinkage and Selection Operator (LASSO)

- Derived from linear regression
  - Penalizes the absolute size of the regression coefficients (especially smaller ones), until the coefficient is set to zero

Traditional linear regression: $y_i \approx \beta_0 + \beta_1 x_i$ and e_i is the residual

$$\mathrm{RSS} = e_1^2 + e_2^2 + \cdots + e_n^2$$

$$\mathrm{RSS} = (y_1 - \hat\beta_0 - \hat\beta_1 x_1)^2 + (y_2 - \hat\beta_0 - \hat\beta_1 x_2)^2 + \ldots + (y_n - \hat\beta_0 - \hat\beta_1 x_n)^2$$

$$\mathrm{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

Multiple regressions expands from previous equations

Lasso (introduce λ as a constraint for RSS):

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

Not relevant β_i are set to zero

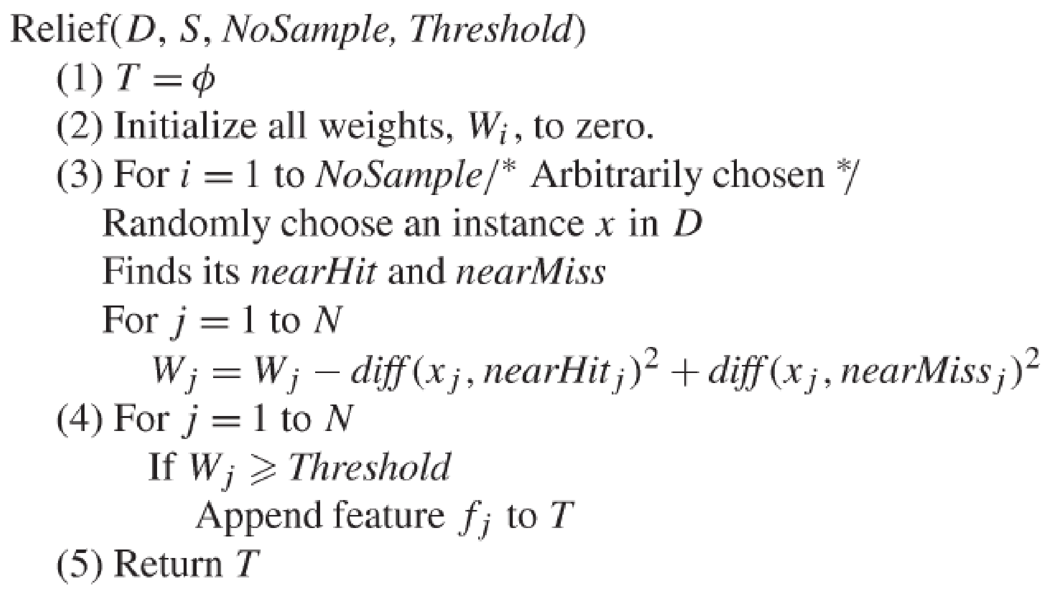
Lasso works for regression problems and attributes should be numeric
James, G., Witten, D., Hastie, T., Tibshirani, R., (2013), *An Introduction to Statistical Learning: with Applications in R*. 1 ed. Springer.
https://nbviewer.org/github/cefet-rj-dal/daltoolbox-examples/blob/main/classification/cla_fs_lasso.ipynb

Correlation-based Feature Selection (CFS)

- Evaluates the predictive capacity of each attribute individually and the degree of redundancy between them, using a correlation coefficient r_{zc}
- Selects sets of attributes that are correlated with the class ($\overline{r_{zi}}$), but with low inter-correlation between them ($\overline{r_{ii}}$)
  - $r_{zc} = \frac{k\overline{r_{zi}}}{\sqrt{k+k(k-1)\overline{r_{ii}}}}$
    - r_{zc} is the correlation between subsets features and class label
    - k is the number of subsets features
    - r_{zi} is the average of the correlations between subset features and the class label
    - $\overline{r_{ii}}$ is the average inter-correlation between subset features
- CFS is computed using a search strategy, such as Genetic Algorithms (GA)

Recursive Elimination of Features (RELIEF)

- Random samples of instances are taken
- For each instance, there is a neighboring instance closest to the same class (nearHit) and closest to different classes (nearMiss)
- Attribute score (average behavior) is calculated
- Users choose attributes by threshold of score



```
Relief($D$, $S$, $NoSample$, $Threshold$)
  (1) $T = \phi$
  (2) Initialize all weights, $W_i$, to zero.
  (3) For $i = 1$ to $NoSample$/* Arbitrarily chosen */
      Randomly choose an instance $x$ in $D$
      Finds its $nearHit$ and $nearMiss$
      For $j = 1$ to $N$
          $W_j = W_j - diff(x_j, nearHit_j)^2 + diff(x_j, nearMiss_j)^2$
  (4) For $j = 1$ to $N$
          If $W_j \geqslant Threshold$
              Append feature $f_j$ to $T$
  (5) Return $T$
```

Advanced topics on classification

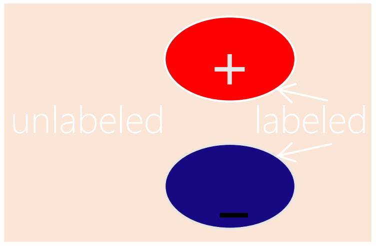
Classification of Class-Imbalanced Data Sets

- Class-imbalance problem: Rare positive example but numerous negative ones, e.g., medical diagnosis, fraud, oil-spill, fault, etc.
- Traditional methods assume a balanced distribution of classes and equal error costs: not suitable for class-imbalanced data
- Typical methods for imbalance data in 2-class classification:
  - Oversampling: re-sampling of data from positive class
  - Under-sampling: randomly eliminate tuples from negative class
  - Threshold-moving: moves the decision threshold, t, so that the rare class tuples are easier to classify, and hence, less chance of costly false negative errors
  - Ensemble techniques: Ensemble multiple classifiers introduced above
- Still difficult for class imbalance problem on multiclass tasks

Multiclass Classification

- Classification involving more than two classes (i.e., > 2 Classes)
- Method 1. One-vs.-all (OVA): Learn a classifier one at a time
  - Given m classes, train m classifiers: one for each class
  - Classifier j: treat tuples in class j as positive & all others as negative
  - To classify a tuple X, the set of classifiers vote as an ensemble
- Method 2. All-vs.-all (AVA): Learn a classifier for each pair of classes
  - Given m classes, construct $\frac{m(m-1)}{2}$ binary classifiers
  - A classifier is trained using tuples of the two classes
  - To classify a tuple X, each classifier votes
    - X is assigned to the class with maximal vote
- Comparison
  - All-vs-all tends to be superior to one-vs.-all
  - Problem: Binary classifier is sensitive to errors, and errors affect vote count

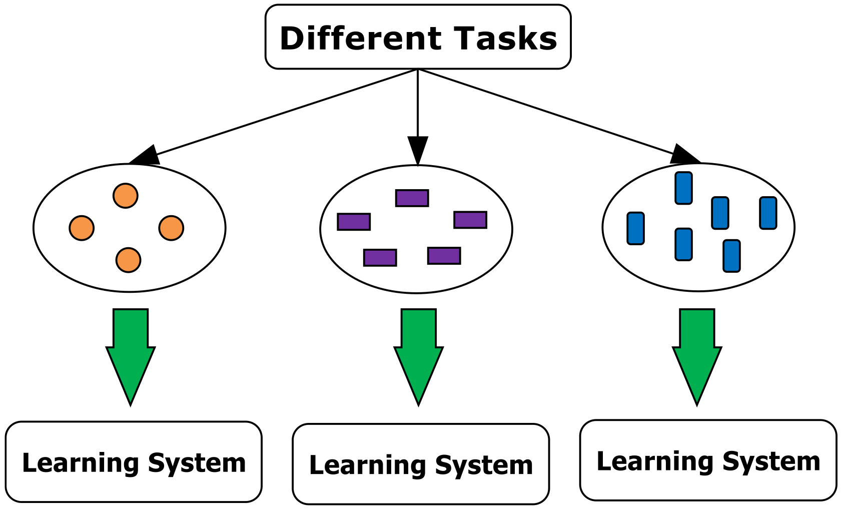
Semi-Supervised Classification

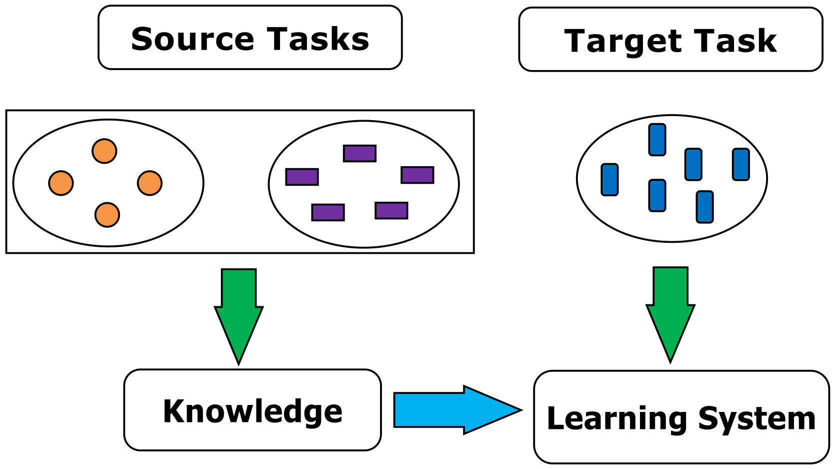


- Semi-supervised:
  - Uses labeled and unlabeled data to build a classifier
- Self-training:
  - Build a classifier using the labeled data
  - Use it to label the unlabeled data, and those with the most confident label prediction are added to the set of labeled data
  - Repeat the above process
  - Adv: easy to understand
  - Disadv: may reinforce errors
- Co-training:
  - Use two or more classifiers to teach each other
  - Each learner uses a mutually independent set of features of each tuple to train a good classifier, say f_1 and f_2
  - Then f_1 and f_2 are used to predict the class label for unlabeled data X
  - Teach each other: The tuple having the most confident prediction from f_1 is added to the set of labeled data for f_2 & vice versa
- Other methods, e.g., joint probability distribution of features and labels

Transfer Learning: Conceptual Framework

- Transfer learning: Extract knowledge from one or more source tasks and apply the knowledge to a target task
- Traditional learning: Build a new classifier for each new task
- Transfer learning: Build new classifier by applying existing knowledge learned from source tasks





Transfer Learning: Methods and Applications

- Instance-based transfer learning
  - Reweight some of the data from source tasks and use it to learn the target task
- Transfer AdaBoost
  - Assume source and target data each described by the same set of attributes (features) & class labels, but rather diff. distributions
  - Require only labeling a small amount of target data
  - Use source data in training:
    - When a source tuple is misclassified, reduce the weight of such tuples so that they will have less effect on the subsequent classifier
- Research issues
  - Negative transfer: When it performs worse than no transfer at all
  - Heterogeneous transfer learning: Transfer knowledge from different feature space or multiple source domains
  - Large-scale transfer learning

Reading papers

Weiss *et al. J Big Data (2016) 3:9*
DOI 10.1186/s40537-016-0043-6

Journal of Big Data

SURVEY PAPER Open Access

A survey of transfer learning

Karl Weiss*, Taghi M. Khoshgoftaar and DingDing Wang

*Correspondence:
kweiss6@fau.edu
Florida Atlantic University,
777 Glades Road, Boca Raton,
FL 33431, USA

Abstract
Machine learning and data mining techniques have been used in numerous real-world applications. An assumption of traditional machine learning methodologies is the training data and testing data are taken from the same domain, such that the input feature space and data distribution characteristics are the same. However, in some real-world machine learning scenarios, this assumption does not hold. There are cases where training data is expensive or difficult to collect. Therefore, there is a need to create high-performance learners trained with more easily obtained data from different domains. This methodology is referred to as transfer learning. This survey paper formally defines transfer learning, presents information on current solutions, and reviews applications applied to transfer learning. Lastly, there is information listed on software downloads for various transfer learning solutions and a discussion of possible future research work. The transfer learning solutions surveyed are independent of data size and can be applied to big data environments.

Keywords: Transfer learning, Survey, Domain adaptation, Machine learning, Data mining

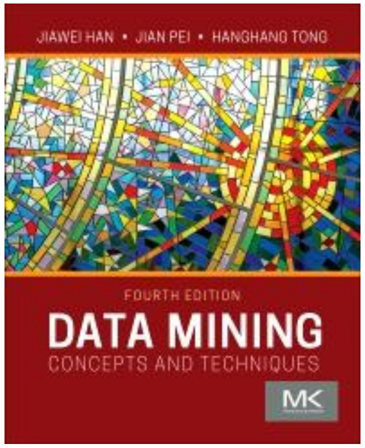
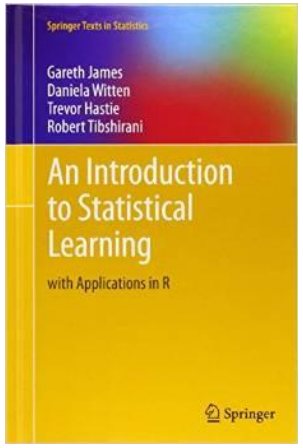
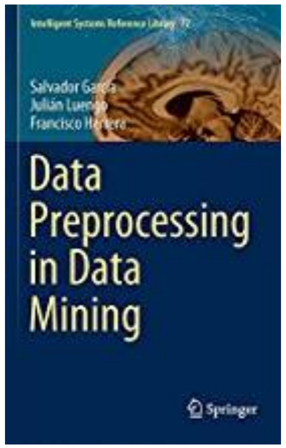
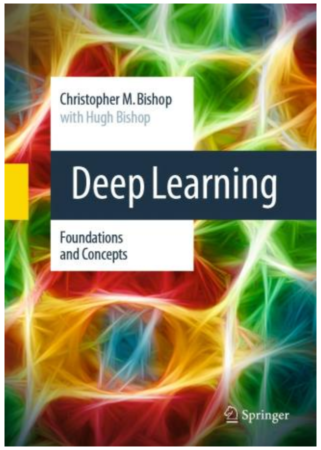
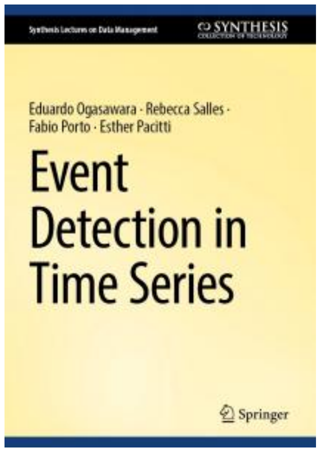
Background
The field of data mining and machine learning has been widely and successfully used in many applications where patterns from past information (training data) can be extracted in order to predict future outcomes [129]. Traditional machine learning is characterized by training data and testing data having the same input feature space and the same data distribution. When there is a difference in data distribution between the training data and test data, the results of a predictive learner can be degraded [107]. In certain scenarios, obtaining training data that matches the feature space and predicted data distribution characteristics of the test data can be difficult and expensive. Therefore, there is a need to create a high-performance learner for a target domain trained from a related source domain. This is the motivation for transfer learning.

Transfer learning is used to improve a learner from one domain by transferring information from a related domain. We can draw from real-world non-technical experiences to understand why transfer learning is possible. Consider an example of two people who want to learn to play the piano. One person has no previous experience playing music, and the other person has extensive music knowledge through playing the guitar. The person with an extensive music background will be able to learn the piano in a more efficient manner by transferring previously learned music knowledge to the task of learning

SpringerOpen

He, H., Garcia, E. A., (2009), "Learning from imbalanced data", IEEE Transactions on Knowledge and Data Engineering, v. 21, n. 9, p. 1263–1284.
Weiss, K., Khoshgoftaar, T. M., Wang, D. D., (2016), "A survey of transfer learning", Journal of Big Data, v. 3, n. 1

Main References

[1] J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022.

[2] G. M. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning: With Applications in R. Springer Nature, 2021.

[3] S. Garcia, J. Luengo, and F. Herrera, Data Preprocessing in Data Mining. Springer, 2014.

[4] C. M. Bishop and H. Bishop, Deep Learning: Foundations and Concepts. Springer Nature, 2023.

[5] E. Ogasawara, R. Salles, F. Porto, and E. Pacitti, Event Detection in Time Series, 1st ed. in Synthesis Lectures on Data Management. Cham: Springer Nature Switzerland, 2025. doi: 10.1007/978-3-031-75941-3.

Slides and videos at: https://eic.cefet-rj.br/~eogasawara/data-mining/