

Master of Technology in Knowledge Engineering

Unit 1

Intelligent Systems & Techniques for Business Analytics

Machine Learning: Models, Tasks, and Statistical Concepts

Sam GU Zhan 顾瞻

zhan.gu@nus.edu.sg

© 2017 NUS. The contents contained in this document may not be reproduced in any form or by any means, without the written permission of ISS, NUS other than for the purpose for which it has been supplied.

Objective

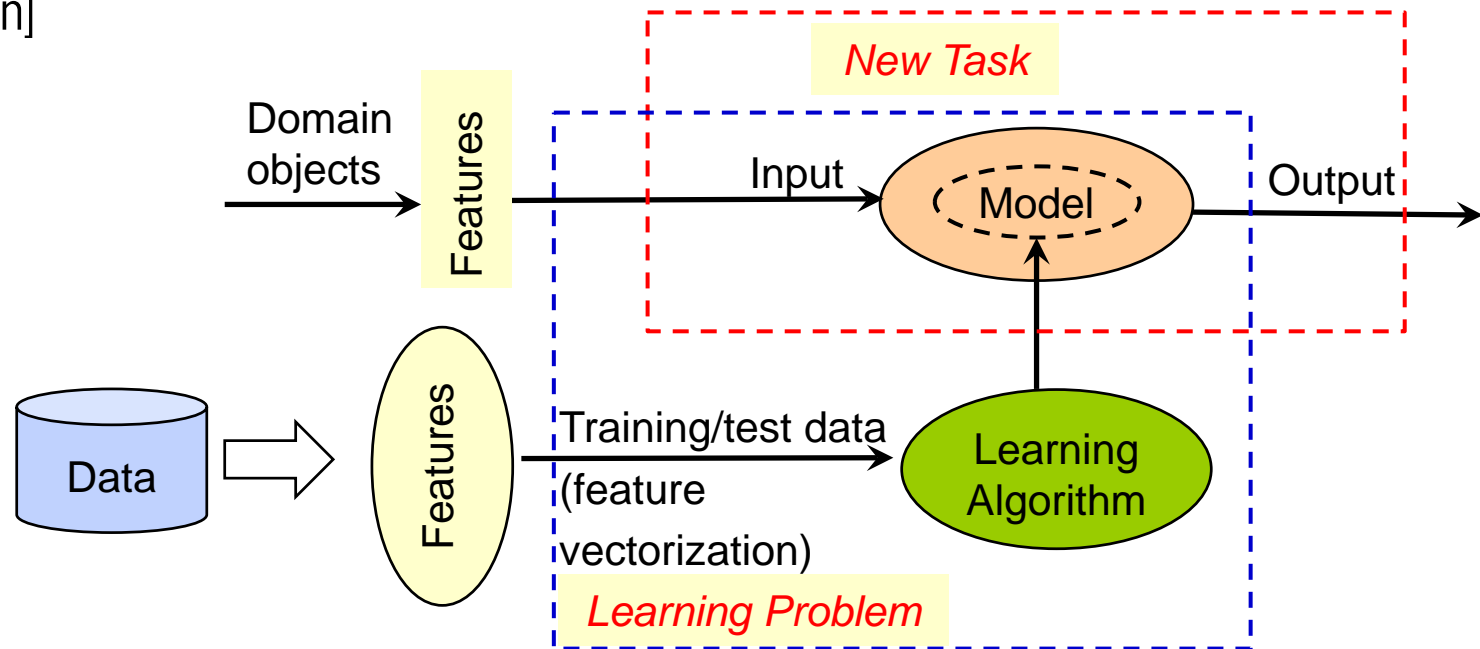
- To introduce machine learning's main paradigms, important methods and algorithms
- To discuss predictive models.

Outline

- ML to address a given task
- ML paradigms and methods
- ML models and features

ML to Address a Given Task

- An overview of how machine learning is used to address a given task [P. Flach, "Machine Learning", Cambridge, 2012, with modification]



👉 *A model is a selected hypothesis in a particular representation (computer understandable form). Task, Experience & Performance*

ML: Tasks, Models, Algorithms, Features

- Learning the knowledge are solved by *learning algorithms* that produce models
- *Task* knowledge is stored in trained *models: parameters, weights, structure.*
- Learning algorithms build models using featuralized, vectorized data
 - » The process of *feature extraction* decides which variables to use for learning and then making prediction
- ML is concerned with using the right *features* to build the right *models* that achieve the right *tasks*

ML Models & Tasks

- *Models lend the ML field diversity, but tasks and features give it unity*
 - » a given task may be solved by different kinds of models
 - » a kind of model may be used to solve problem of different tasks with different features
 - » Mathematicians, engineers, psychologists, computer scientists and many others have discovered ways to solve generic tasks and brought their specific background into ML
 - » Consequently the principles underlying these models/algorithms are very diverse
 - ♦ a considerable range of models to choose from

Main Paradigms of ML Algorithms

- **Supervised learning**

- » Mainly for **predictive** type of applications, e.g.:

- ♦ classification (predict discrete categorical value), regression (predict continuous numeric value), forecast, function approximation, curve fitting

- » Training data:

- ♦ labeled data with target output provided for specific input:
 - many goods in shopping cart → gender (classification)
 - house location, size, rooms → house price (regression)

- **Unsupervised learning**

- » Mainly for **descriptive** type of application

- ♦ clustering, grouping, granulation, association, graph mining

- » Training data:

- ♦ un-labeled data with only input patterns but no output provided

Main Paradigms of ML Algorithms (cont.)

- **Reinforcement learning**

- » Concerned with how software agents learn to act, given an observation of the world.
 - ♦ Agent takes action, which has some impact in the environment to change it to new state, and the environment also provides feedback (rewards) that guides an underlying learning algorithm.
- » The problem is studied in many disciplines, such as game theory, control theory, operations research, simulation-based optimization, genetic algorithms, ...
- » Applied to robot learning: Markov process, Q-learning, Dynamic programming, Google Deepmind's deep Q-networks, StarCraft2, etc
 - 👉 Reinforcement learning differs from supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected

ML Applications in Robotics

- *Robot learning*

- » Is a research field at the intersection of ML and *robotics*
- » Allows a robot generate its own sequences (also called curriculum) of learning situations to cumulatively acquire repertoires of novel skills through
 - ◆ autonomous self-exploration, and
 - ◆ social interaction with *human teachers*
- » Using guidance mechanisms such as
 - ◆ active learning, maturation, motor synergies, and imitation.



Asimo (28 April, 2011)
Year of creation: 2000
Website:
asimo.honda.com

- Examples of modern ML application in robotics
(<http://techemergence.com/machine-learning-in-robotics/>)

Important Learning Methods

- **Decision tree** learning
 - » uses a decision tree to capture and represent a *predictive model* which maps observations about an item to conclusions about the item's target value.
- **Association rule** learning (market basket analysis)
 - » Association rule^(*) learning is a method for discovering interesting relations between variables in large databases.
 - (*) the key point is “association/correlated” relation, rules here should not be confused with decision rules and *implication*

Important Learning Methods (cont.)

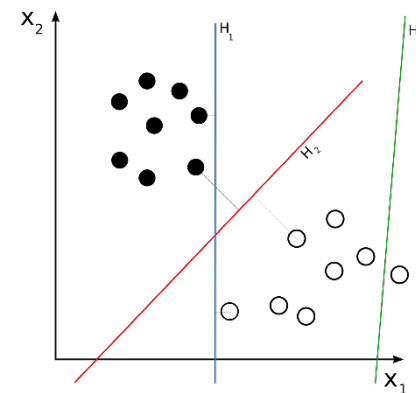
- **Artificial neural networks (ANN) learning**
 - » Computations are structured in terms of an interconnected group of artificial neurons
 - » Processing information using a connectionist approach
 - **NNs are non-linear statistical data modeling tools**
 - » They are usually used
 - ◆ To model complex non-linear relationships between inputs and outputs,
 - ◆ To find patterns in data, feature dimension reduction, or
 - ◆ To capture the statistical structure in an unknown joint probability distribution between observed variables.
- (Neural networks will be further discussed in *Computational Intelligence*)

Important Learning Methods (cont.)

- *Support vector machines* (SVMs)

- » are a set of related supervised learning methods used for classification and regression (called *support vector regression*, SVR).
- » Given a set of training examples, each marked as belonging to one of two/more categories, an SVM training algorithm builds a model that predicts whether a new example falls into which category.

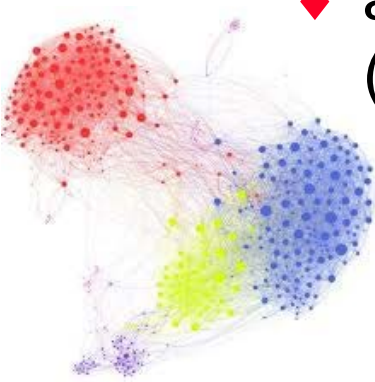
(SVM will be will be further discussed in
Computational Intelligence)



Important Learning Methods (cont.)

- **Clustering**, Cluster analysis

- » is a method of unsupervised learning, and a common technique for statistical data analysis, data exploration, customer segmentation.
- » is the assignment of a set of observations into subsets (called *clusters*) so that
 - ♦ according to some pre-designated criterion or criteria (*similarity metric*)
 - observations within the same cluster are *more similar*
 - while observations drawn from different clusters are *less similar*.



Models & Features

Categories of ML Models

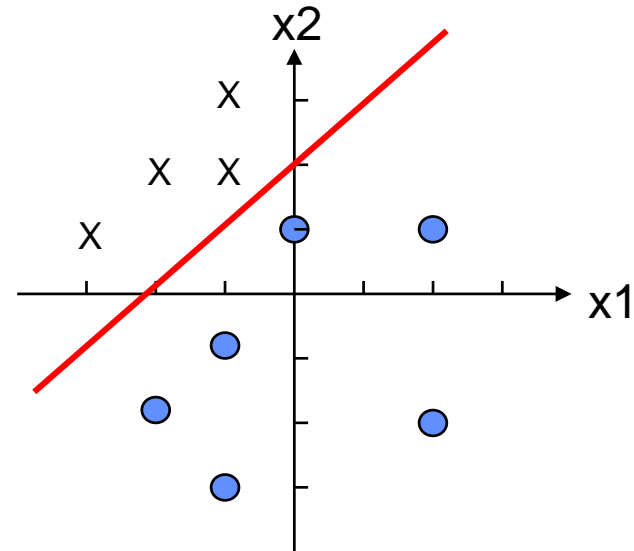
- Models are what is being learned from data, in order to solve a given task
- Two different dimensions, perspectives used for categorizing ML models
 - » Geometrical vs. Probabilistic vs. Logical models
 - » Grouping vs. Grading models

Geometric Models

- The *instance space* is the set of all possible or describable instances (whether they are presented in our data set or not)
 - » Usually this set has some geometric structure
 - ♦ E.g.: when all features are numerical, then we can use each feature as a coordination in a Cartesian coordinate system
- A *geometric model* is constructed directly in instance space, using geometric concepts such as lines, planes and distances

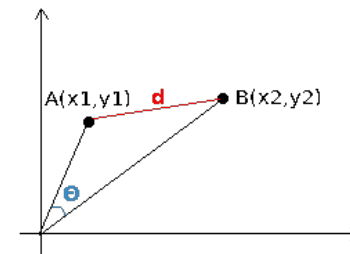
Geometric Models: geometric classifier

- Example 7.1:
 - » the *linear classifier* is a geometric classifier
 - » One main advantage: They are easy to visualize, as long as we keep to two or three dimensions
- If there exists a linear *decision boundary* separating the two classes, we say that the data is *linearly separable*



Geometric Models: distance

- A very useful geometric concept in ML is the notion of **distance** (therefore to measure **similarity**)
 - » If the distance between two instances is small then the instances are considered similar in terms of their feature values, so nearby instances would be expected to
 - ♦ receive the same class label (for classification), or
 - ♦ belong to the same cluster (for clustering)
 - » Is usually measured by the **Euclidian distance**, which is the square root of the sum of the squared distances along each coordinate **d** = $\sqrt{\sum_{i=1}^d (x_i - y_i)^2}$



Geometric Models: distance-based classifier

- **Nearest-neighbor classifier**

近朱者赤，近墨者黑。

One who stays near vermilion gets stained red,
and one who stays near ink gets stained black.

- » To classify a new instance, we

- ♦ retrieve from memory the most similar training instance (i.e.: the training instance with the smallest Euclidean distance from the instance to be classified), and
- ♦ simply assign that known training instance's class (Example: Identify Sam's dressing style from class)

- **Variations: *k-nearest neighbor (k-NN)***

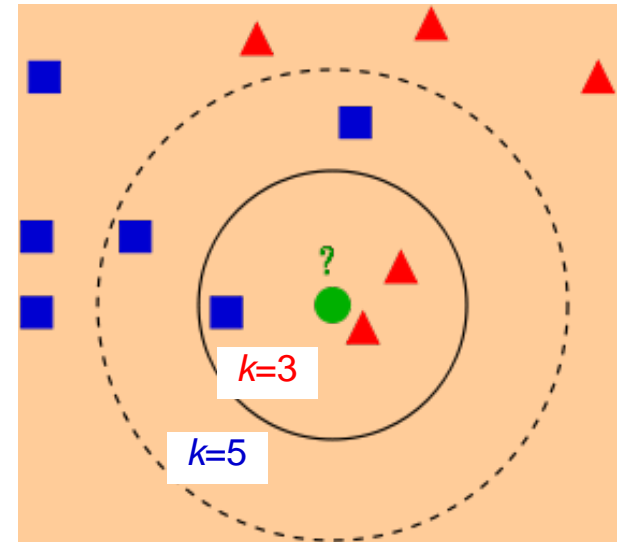
- » a type of lazy learning, among the simplest of all ML algorithms, can be for both classification and regression
- ♦ where the function is only approximated locally and all computation is **deferred** until classification / regression

Geometric Models: **k-NN**

- In *k-NN classification*

- » The output is a class membership.
- » An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small)

- ♦ If $k = 1$, then the object is simply assigned to the class of that single nearest neighbour



Geometric Models: **k-NN** (cont.)

- In *k-NN regression*
 - » The output is the property value for the object
 - » This value is the average of the values of its *k*-nearest neighbours
- Weighting scheme
 - » it can be useful to weight the contributions of the neighbours, so that the nearer neighbours contribute more to the average. E.g.: $w = 1/d$
 - ♦ where d is the distance of the instance under evaluation to the neighbour

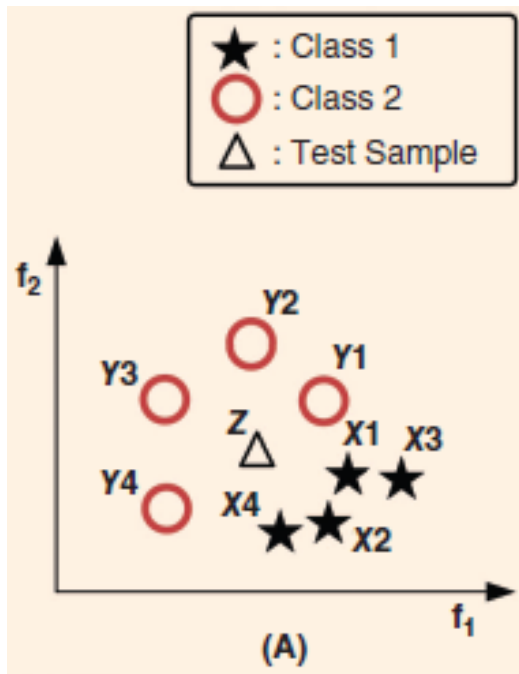
*Extended Nearest Neighbor (ENN)

- Issues with k-NN
 - » The error rates are increased when sample sizes are small or data are imbalanced. Only local majority is taken into account but not the global statistics, so misclassification may occur when nearest neighbors are dominated by samples from other classes
- *Extended Nearest Neighbor (ENN)* [Tang & He, 2015]
 - » The prediction is made based on which decision resulting the largest intra-class coherence when iteratively assume the test sample to each possible class
 - » By exploiting the information from all available data to maximize the intra-class coherence, ENN is *able to learn from the global distribution*

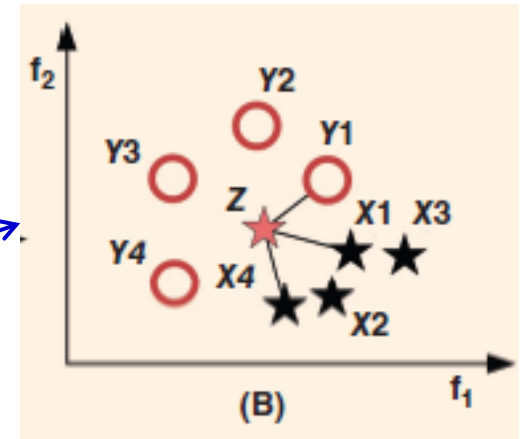
*Extended Nearest Neighbor (ENN) (cont.)

• ENN: 3-nearest neighbors

consider not only 'who' are the nearest neighbors of the test sample, but also 'who' consider the test sample as their nearest neighbors



Assume Z to be Class 1



Assume Z to be Class 2



Figures from: B. Tang & H. He, "ENN Extended Nearest Neighbor Method for Pattern Recognition", IEEE Computational Intelligence, Vol. 10, No. 3, 2015

Probabilistic Models: basic idea

- Let
 - » X denote the variables we know about
 - ♦ e.g.: our instance's feature values; and
 - » Y denote the *target variables* we are interested in
 - ♦ e.g., the instance's class
- The statistician's approach to model the relationship between X and Y is to
 - » assume that there is some underlying random process that generates the values for these variables, according to a well-defined but *unknown* probability *distribution*, and
 - » use the data to find out more about this distribution

Probabilistic Models: simple example

- X is known for a particular instance, but Y may not be,
 - » we are interested in the conditional probability $P(Y|X)$
- Example 7.2: a simple *Bayesian classifier*
 - » Y — indicates whether the e-mail is spam
 - ♦ Y is the class variable, with values ‘yes’ and ‘no’
 - » X — indicates whether the e-mail contains the words ‘Viagra’ and ‘lottery’.
 - ♦ Viagra and lottery are two Boolean variables which together constitute the feature vector X
 - » The probability of interest is then $P(Y|\text{Viagra, lottery})$

Probabilistic Models: simple example (cont.)

Example 7.2 (cont.)

- » For a particular e-mail that contains the word 'Viagra', but no 'lottery' $P(Y | \text{Viagra}=1, \text{lottery}=0)$ is called **posterior probability** after the features X are observed
- » An example of posterior distribution (*estimate from data*)

Viagra	lottery	$P(Y=\text{yes} \text{Viagra}, \text{lottery})$	$P(Y=\text{no} \text{Viagra}, \text{lottery})$
0	0	0.30	0.70
0	1	0.60	0.40
1	0	0.80	0.20
1	1	0.40	0.60

- ♦ If an e-mail does not contain 'Viagra', the observing 'lottery' increases the spam probability from 0.30 to 0.60
- ♦ But if the e-mail contains 'Viagra', then the observing 'lottery' decreases the spam probability from 0.80 to 0.40

Probabilistic Models: prediction

Example 7.2 (cont.)

» *Decision rule*

- ♦ e.g., with the observation of words ‘Viagra’, and ‘lottery’, we **predict** spam if the probability exceeds 0.5 and no-spam otherwise.

» *Bayes’ rule*, a simple property of conditional probabilities

$$P(Y | X) = \frac{P(X | Y)P(Y)}{P(X)}$$

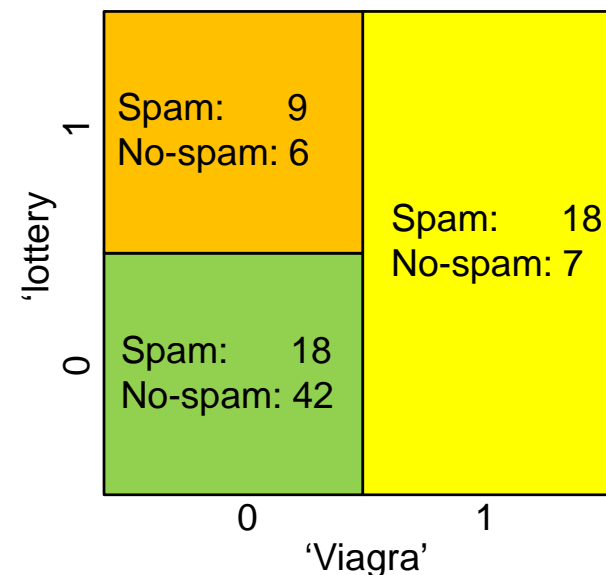
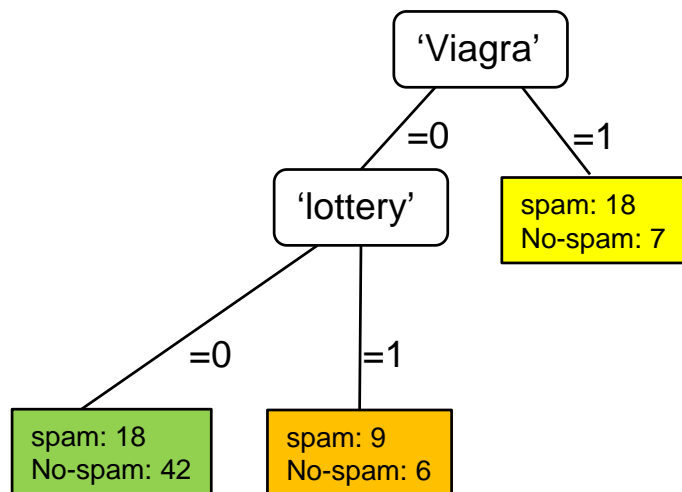
- ♦ $P(Y)$ is the **prior probability**, **before** observation of X
- ♦ $P(X)$ is the probability of the data, which is independent of Y

Logical Models

- The type of logical models is more algorithmic in nature.
 - » It is called '*logical*' because models of this type can be easily translated into rules, such as
 - ♦ If Viagra = 1 then Class = spam
- The rules representing logical models are easily organized in a tree structure, called *feature tree*
 - » Features are used to iteratively partition the instance space
 - » The leaves of the tree correspond to *instance space segments*, *segments* for short (i.e. traditional customer segmentation tree)

Logical Models: feature tree

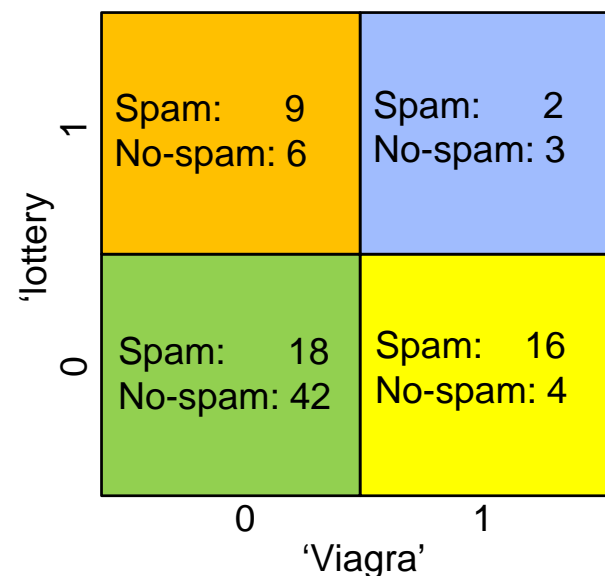
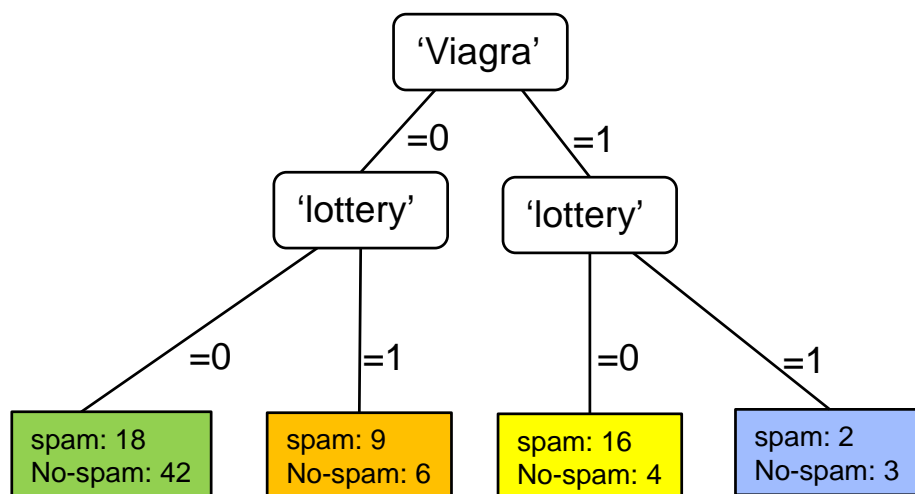
- Example 7.3:
 - » Consider the same 'spam e-mail' case from Example 7.2
 - » (left) A feature tree combining two Boolean features
 - » (right) The corresponding instance space partition



Logical Models: **feature tree** (cont.)

Example 7.3 (cont.)

- » (left) A complete feature tree
- » (right) The corresponding instance space partition is the finest partition that can be achieved with those two features



Feature trees whose leaves are labelled with classes are commonly called **decision trees**

Logical Models: decision list

- Consider further the feature tree in Example 7.3:
 - » If we were to label the leaves by majority class we obtain the following *decision list*
 - if Viagra = 1 **then** Class = 'Y = spam'
 - else** if lottery = 1 **then** Class = 'Y = spam'
 - else** Class = 'Y = no-spam'
 - » Describe as un-nested rules
 - if Viagra = 1 **then** Class = 'Y = spam'
 - if Viagra = 0 \wedge lottery = 1 **then** Class = 'Y = spam'
 - if Viagra = 0 \wedge lottery = 0 **then** Class = 'Y = no-spam'
 - 👉 Here, every path from root to a leaf is translated in to a rule

Grouping or Grading

- We have looked models at the dimension with three general types, for an ease of understanding:
 - » geometric models, probabilistic models, logical models
- Another important dimension, perspective:
 - » Grouping models, grading models
 - ♦ They handle the instance space in different ways
 - 👉 It is somewhat more abstract dimension that is in some sense orthogonal to the geometric-probabilistic-logical dimension

Models for Grouping

- *Grouping models* handle the instance space by breaking up it into groups, or segments,
 - » the number of which is determined at training time
- A good example: tree-based models
 - » Trees are usually of limited depth and don't contain all the available features
 - ◆ The subsets at the leaves of the tree partition the instance space with finite resolution
 - » Instances filtered into the same leaves of the tree are treated the same
 - ◆ regardless of any features not in the tree might be able to distinguish them

Models for Grading

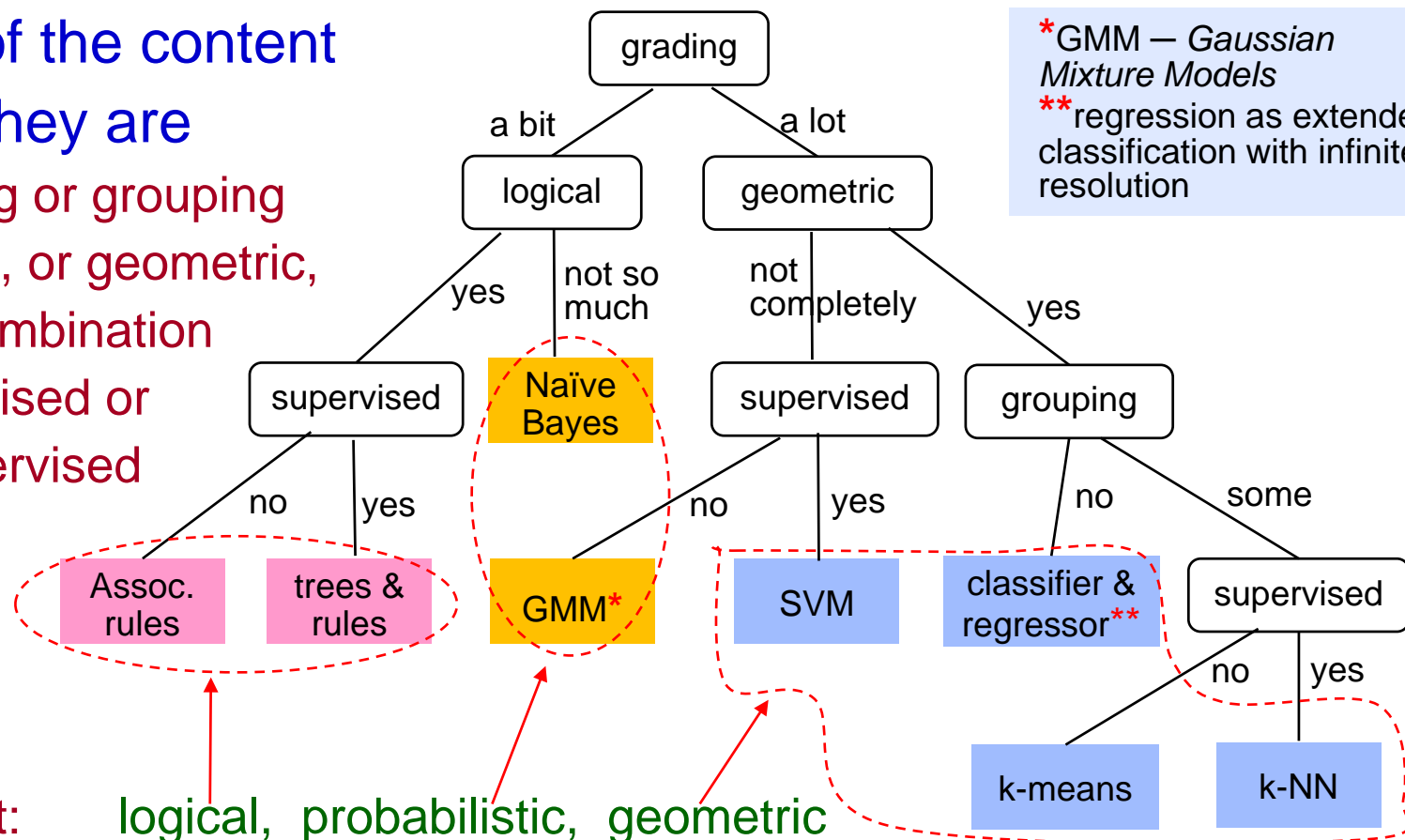
- Grading models usually are able to provide infinite resolution
 - » Regression is of *grading* model
 - » Support vector machines (including SVR) and other geometric classifiers are examples of *grading models*
 - ♦ They work in Cartesian instance space, and are able to represent and exploit the minutest difference between instances
 - ♦ It is always possible to come up with a new test instance that receives a *score* that has not been given to any previous test instance
 - 👉 the distinction between grouping and grading models is relative rather than absolute

ML Methods: a taxonomy

- In terms of the content to which they are

- » Grading or grouping
- » Logical, or geometric, or a combination
- » Supervised or unsupervised

👉 Not intend to include all the ML methods



*GMM — Gaussian Mixture Models
 ** regression as extended classification with infinite resolution

[Ref: Fig.1.8 from “Machine Learning”, by P. Flach, 2012, with modification]

Features: workhorses of ML

- Determine much of the success of a ML application
 - » *A model is only as good as its features*
- Mathematically, features are *functions* that
 - » Map from the instance space to some set of feature values called the *domain* of the feature.
 - » Some of the common feature domains:
 - ♦ the set of real numbers:
 - e.g., measures of objects for classification
 - ♦ the set of integers
 - e.g., counts of occurrence of a particular words
 - ♦ the Booleans (a statement that can be true or false)
 - e.g., 'Sam's e-mails are all spams'

Two Uses of Features: **splits**

- Features as *splits*
 - » Particularly in logical models, is to zoom in on a particular area of the instance space
- Example 7.4: *binary splits*
 - » Let f be a feature counting the number of occurrence of the word 'Viagra' in an e-mail, and let x stand for an arbitrary e-mail. Some conditions as binary splits:
 - ♦ $f(x) = 0$ selects e-mails that don't contain 'Viagra';
 - ♦ $f(x) > 0$ selects e-mails that do;
 - ♦ $f(x) > 2$ selects e-mails that contain the word more than twice, and so on

Two Uses of Features: splits (cont.)

- Example 7.5: non-binary splits

- » Let g be a feature that has the value

- ♦ 'tweet' for e-mails with up to 20 words
- ♦ 'short' for e-mails with 21 to 50 words
- ♦ 'medium' for e-mails with 51 to 200 words
- ♦ 'long' for e-mails with more than 200 words

then the expression $g(x)$ represents a *four-way split* of the instance space

- ♦ such splits can be combined in a feature tree,
 - from which respective models can be built

Two Uses of Features: predictors

- Features as *predictors*
 - » to make precise and measurable contribution to final prediction
 - » Individual feature are not split by 'threshold', but their full range of value 'resolution' is used in computing and instance's score
- These two uses of features as 'splits' and 'predictors' are sometimes combined in a single model

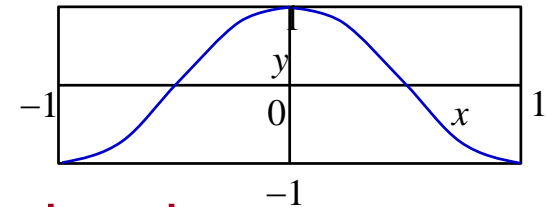
Two Uses of Features: combined

- Example 7.6:

- » Suppose we want to approximate $y = \cos \pi x$ on the interval $-1 \leq x \leq 1$, and a linear approximation is not much use here.

- » We split the x-axis in two intervals:

- ♦ $-1 \leq x < 0$ and $0 \leq x \leq 1$

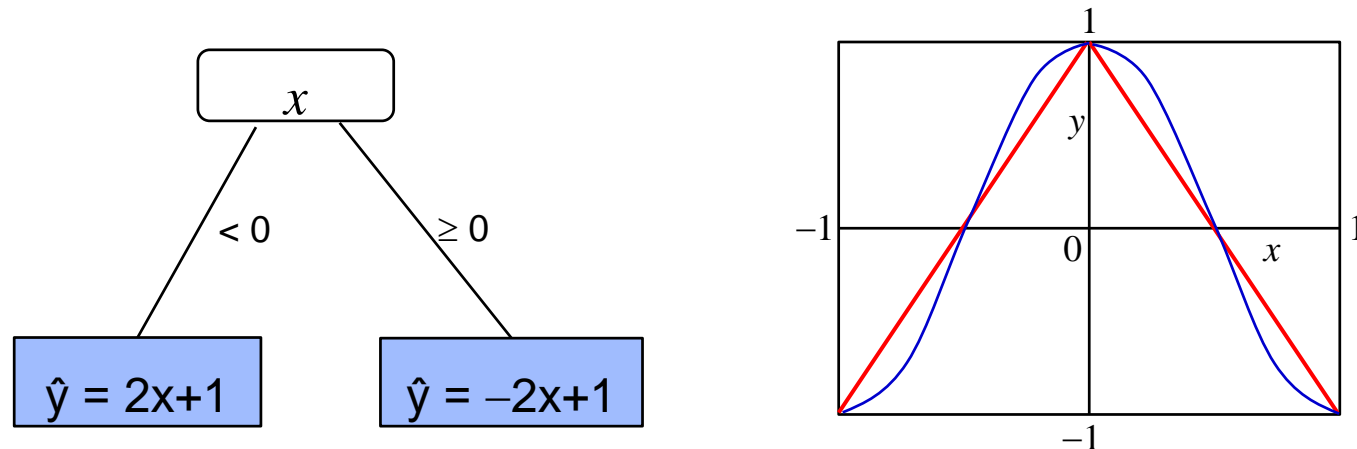


and could find reasonable linear approximations on each interval

- ♦ We achieve this by using x both as a splitting feature and as a regression variable (predictor)

Two Uses of Features: **combined** (cont.)

Example 7.6 (cont.)



- » **Left:** a regression tree combining a one-split feature tree with linear regression models in the leaves. Here x is used as both a splitting feature and a regression variable
- » **Right:** the function y on the interval $-1 \leq x \leq 1$, and the piecewise linear approximation achieved by the regression tree

Feature Extraction

- *Feature extraction* is the process of taking data from a free-form arrangement, and arranging them into rows and columns of numbers
 - » E.g.: in spam filter, or text classification more generally
 - ◆ The messages or documents don't come with built-in (ready-to-use) features, rather, they need to be constructed by the developer of the ML application,
 - Index an e-mail by the words that occur in it, called *Bag of words*
 - Mark up a word in a text (corpus) as corresponding to a particular part of speech in grammar *POS tagging*

Feature Extraction (cont.)

- From the bag of words, one may further extract things such as
 - » the number of capital letters in the document,
 - » the number of words in all caps,
 - » the number of times the word “buy” appears in the document, and other numeric features selected
- The extracted features can be used to highlight more subtle differences between spam and non-spam emails

Feature Extraction (cont.)

- E.g.: building a system for trading securities
 - » Feature extraction would be deciding what things will be used to predict prices, such as
 - ♦ past prices, prices of related securities, interest rates, and features extracted from news releases, ...
 - » Further more, *feature engineering* is the process of manipulating and combining features to arrive at more informative ones: *engineered features* that are basically functions of past prices believed to be useful:
 - ♦ such as MACD (moving average convergence divergence), RSI (relative strength index), ...

Feature Transformation

- It is often natural to build a model in terms of the given features
 - » However, we are also free to change the features as we see fit, remove some of them, or introduce new features
- For complex classification tasks, we may transform the entire instance space by mapping it into a new 'feature space', so the problem is more solvable
 - » E.g., the XOR problem is not linearly separable in original instance space, but we can solve it using linear model through *feature transformation*

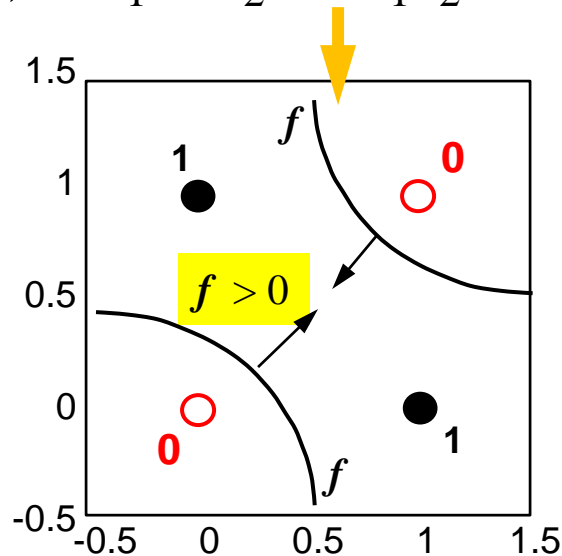
Feature Transformation (cont.)

- Mapping into feature space

$$z_1 = x_1, \quad z_2 = x_2, \quad z_3 = x_1 x_2$$

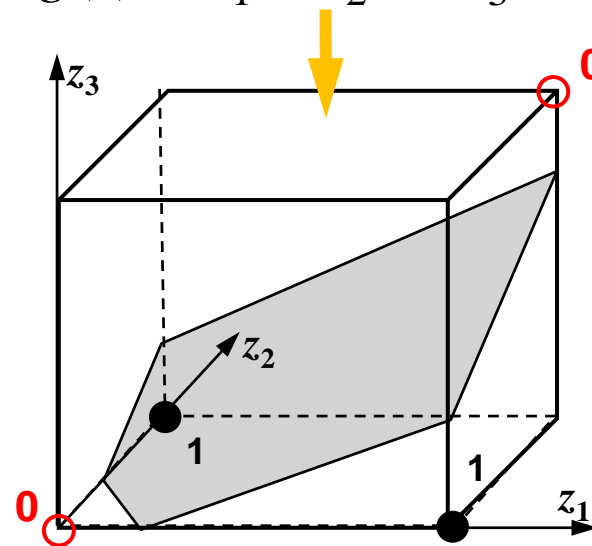
» Nonlinear function

$$f(x) = x_1 + x_2 - 2x_1x_2 - 1/3$$



Linear function in feature space

$$g(z) = z_1 + z_2 - 2z_3 - 1/3$$



[Adopted from Vojislav Kecman, “Learning and Soft Computing”, MIT Press, 2001]

Discussion



- Credit Card & Risk Management
 - » What is the task?
 - » What are the (possible) features?
 - » What kinds of model can be used to solve this problem?

Tasks of Learning

- We have discussed learning approaches from the perspectives of
 - » Representation of hypothesis (models)
 - » Paradigms of learning mechanism (algorithms)
- Considering the tasks of learning from data for problem solving, there are two major groups
 - » **Predictive** through supervised learning
 - ♦ *Classification, Prediction, Regression*
 - » **Descriptive** through unsupervised learning
 - ♦ *Clustering, Association, Dimension reduction, Segmentation*

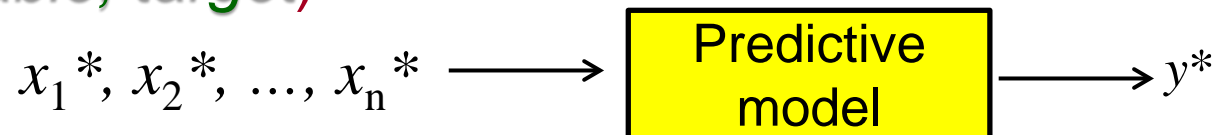
Predictive Models

Predictive Model

- In classification / prediction, the hypothesis obtained through learning is a ***predictive model***

$$y = f(x_1, \dots, x_n)$$

- » x_1, x_2, \dots, x_n are input variables, called ***predictors*** (also ***attributes, features***)
- » y is the output variable, called ***response*** (also ***dependent variable, target***)



- » For continuous data, $y \in V$, where V is a range of values
- » For categorical data, $y \in C$, where $C = \{C_1, C_2, \dots, C_m\}$ is the set of class labels

Prediction: broad or narrow sense

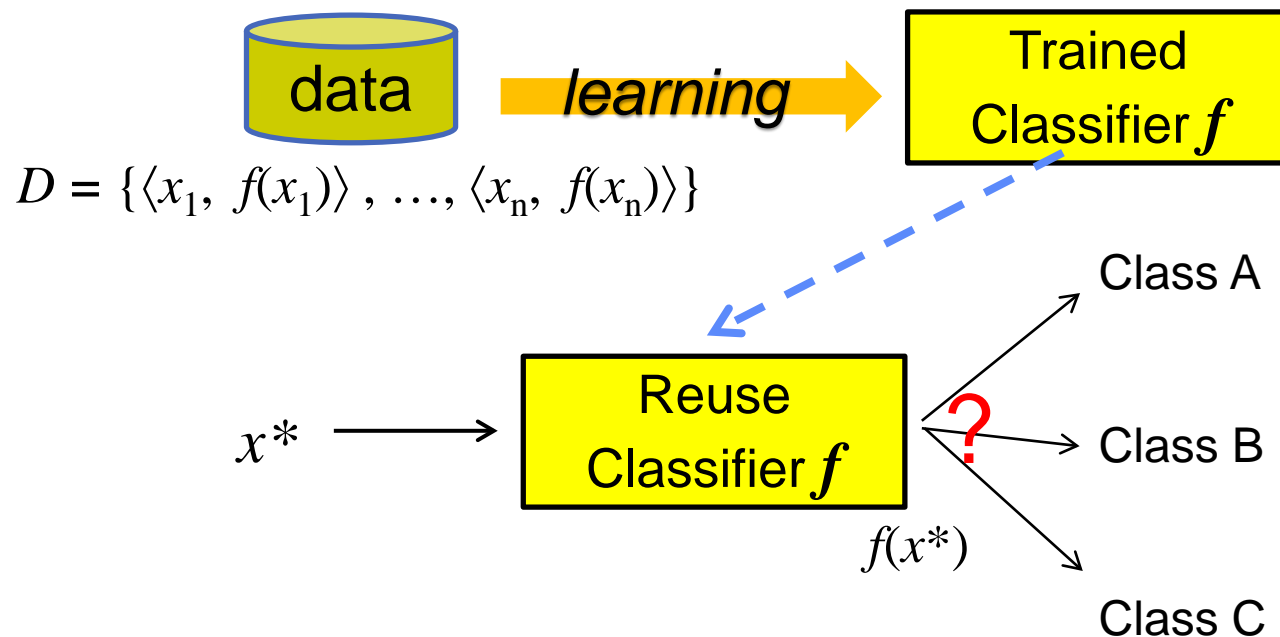
- In Classification
 - » we are trying to **predict** a class (usually not in certainty)
- **Prediction**, in its narrow sense, is similar to classification, but broader
 - » we can also **predict** value of a numerical variable (e.g., amount of purchase) rather than a class (e.g., purchaser or non-purchaser)
 - » **Estimation** is also used to refer to prediction of the value of a continuous variable
- In its broad sense, “**prediction**” may be used for both continuous and categorical data

Classification

- **Classification** is perhaps the most basic form of data analysis, and the most basic task of learning
- E.g.:
 - » An applicant for a loan can repay
 - ♦ on time, late, or declare bankruptcy
 - » A credit card transaction can be
 - ♦ normal, or fraudulent
 - » The victim of an illness (e.g.: SARS) can be
 - ♦ recovered, recovering, or deceased

Classification: learning & reasoning

- Data set (featuralized)
 - » $D = \{\langle x_1, f(x_1) \rangle, \dots, \langle x_n, f(x_n) \rangle\}$
- Learning and reasoning (generalization):

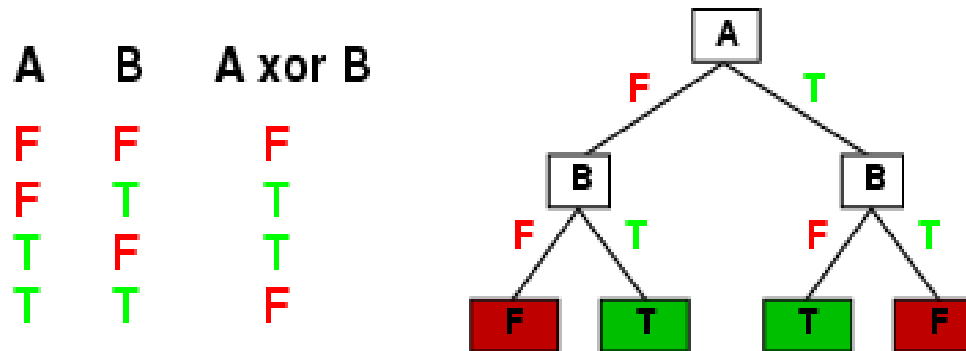


Classifiers

- **Classifiers** (models) are constructed through learning based on the training data
- Different kinds of model can be formed, such as
 - » Decision trees
 - » Neural networks (supervised learning)
 - ◆ Different learning algorithms available for different NN architectures
 - » Support vector machine
 - » K-Nearest Neighbours

* Decision Tree

- Decision trees can express any function of the input attributes.
 - » for Boolean functions: truth table row is path to leaf



* Example: Wait for Table

- Example 7.7:

- » Decide whether to wait for a table at a restaurant, based on the following attributes:

1. Alternate: is there an alternative restaurant nearby?
2. Bar: is there a comfortable bar area to wait in?
3. Fri/Sat: is today Friday or Saturday?
4. Hungry: are we hungry?
5. Patrons: number of people in the restaurant (None, Some, Full)
6. Price: price range (\$, \$\$, \$\$\$)
7. Raining: is it raining outside?
8. Reservation: have we made a reservation?
9. Type: kind of restaurant (French, Italian, Thai, Burger)
10. WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60 mins)

* Wait for table: Attribute-based Representations

Example 7.7 (cont.)

- » Situations where I will/won't wait for a table, described by attribute values (Boolean, discrete, continuous):

Example	Attributes										Target Wait
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	\$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	\$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	\$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	\$	F	F	Burger	30–60	T

- » Classification of examples is positive (T) or negative (F)

* Wait for table: Decision Tree Learned

Example 7.7 (cont.)

- Decision tree learned from the 12 sample cases:

- » Substantially simpler than “true” tree that has one leaf for each case



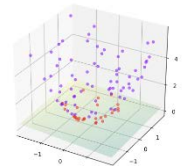
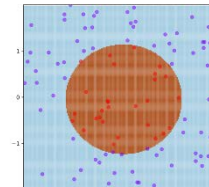
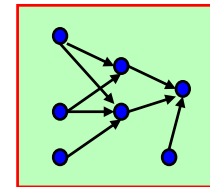
- » Note:

♦ 4 attributes (among 10) used here

👉 **feature selection** and **dimension reduction**
(not to further discuss)

* SVM and NN as Non-linear Classifier

- SVM / NN are both referred to as **black box** methods
 - » the **process** of how SVM and NN transform from input to output is less clear and can be hard to interpret (unlike tree-based or probabilistic-based methods)
- SVM and NN are similar in structure after learning, but differ by the **learning method** used
 - » **NN**: Creating a group of connected simple linear calculation unit + non-linear activation/transfer function
 - » **SVM**: Mapping (kernel trick) data into a richer feature space including non-linear features, then use a linear classifier (decision hyperplane) to separate feature space



* Support Vector Machines

- Support vector machines are a method of obtaining the optimal boundary of two sets in a vector space *independently on the probabilistic distributions* of training vectors in the sets.
 - » Optimal hyperplane for linearly separable patterns
 - ◆ Support vectors are the data points that lie closest to the decision surface, and so the most difficult to classify
 - » Extend to patterns that are not linearly separable by transformations of original data to map into new space — *kernel function*, i.e. radio basis, polynomial

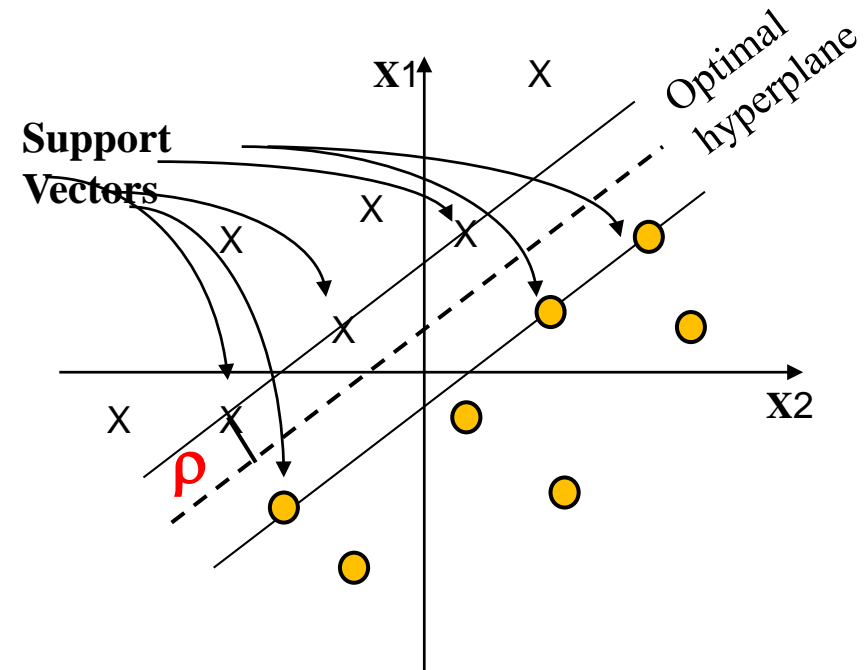
* Support Vector Machines: Optimal Hyperplane

- *Margin of separation* (denoted as ρ)

- » The separation between the decision surface hyperplane and the closest data point

- *Optimal hyperplane*

- » Is the particular hyperplane with the margin of separation ρ maximized
- » This hyperplane is uniquely determined by the vectors on the margin \longrightarrow the *support vectors*!



Regression

- The most popular model for making predictions is the *multiple linear regression* model.
 - » This model is used to **fit** a linear relation between quantitative response variable Y and a set of predictors X_1, X_2, \dots, X_n
- The assumption is that in the population of interest, the following relation holds

$$y = f(x_1, x_2, \dots, x_n) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \varepsilon$$

where β_0, \dots, β_n are **coefficients** and ε is the **noise** or unexplained part.

Regression: model fitting

- The two popular objectives behind fitting a model
 - » Understanding the relationship between these factors
 - ♦ Explanatory model
 - Classical statistical approach focuses on fitting the best model to the data in an attempt to learn about the underlying relationship in the population
 - » Predicting the outcomes of new cases
 - ♦ Predictive model
- Learning process
 - » To estimate the coefficients and the variability of the noise, using sample data from the population.

← Our primary interest

Regression: different target

- Both explanatory and predictive model learning involve
 - » using a dataset to train a model,
 - » checking model validity, and
 - » assessing performance and compare to other baseline
- Different targets
 - » Explanatory model = good **fitting**
 - ♦ fits the data closely for good explanation, but may have low prediction accuracy (because of overfitting)
 - » Predictive model = good **prediction** (generalization)
 - ♦ predicts new cases accurately, but may have a looser fit to the data on which it is based/trained

Regression: application

- Examples of multiple linear regression applied in business / industry
 - » Deterring credit limit for customer credit cards from their demographics and historical activity patterns
 - » Predicting the time to failure of equipment based on utilization and environment conditions (survival analysis)
 - » Predicting expenditures on vacation travel based on historical frequent flyer data, web browsing history, shopping carts

*Ensemble Classification & Regression

- Ensemble methods

- » Basic idea: use multiple models to get better performance
- » Main theory: *Bias-variance-covariance decomposition*
 - ♦ Improved performance of an ensemble over its constituent base predictors with diversities in
 - data, parameter, structure
- » Conventional ensemble methods include:
 - ♦ *Bagging*, *boosting*, and *stacking* based methods

- Key point:

- » *If we are able to design low-correlated individual learners, we can expect an increase in performance*



[Ref: Ren, Zhang & Suganthan, “Ensemble Classification and Regression – Recent Development, Applications and Future Direction”, IEEE Computational Intelligence, Feb 2016]

Measuring Quality of Fit

- Suppose we have a regression problem.
 - » One common measure of accuracy is the *mean squared error* (MSE) i.e.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- ♦ Where \hat{y}_i is the prediction our method gives for the observation in our training data.

Performance of Learning Methods

- Our method has generally been designed to make MSE small on the training data we are looking at
 - » i.e. we choose the line such that MSE is minimized.
- However, what we really care about is how well the method works on new data. We call this new data *Test Data*.
- There is no guarantee that the method with the smallest training MSE will have the smallest test MSE.

Our Concerns: Training vs. Test MSE

- In general the more flexible a method is the lower its training MSE will be
 - » i.e. it will “fit” or explain the training data very well.
- However, the test MSE may in fact be higher for a more flexible method than for a simple approach like linear regression.

Our Concerns: Model Flexibility & Interpretability

- More flexible methods (such as splines) can generate a wider range of possible shapes to estimate f as compared to less flexible and more restrictive methods (such as linear regression).
- The less flexible the method, the easier to interpret the model.

Model interpretability is important for analytics:

Interpretation, decision, and action in business decision

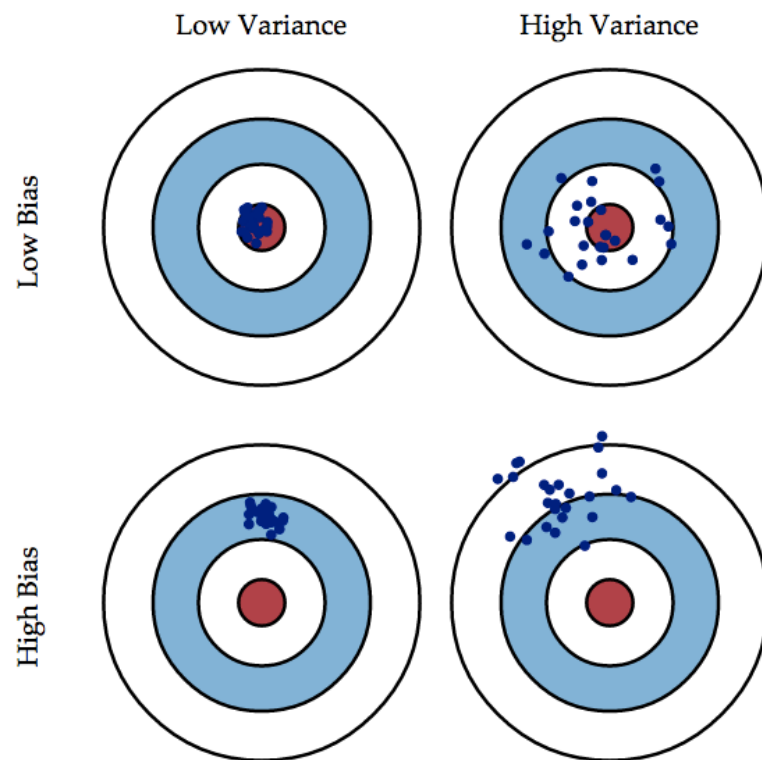
- Thus, there is a trade-off between flexibility and model interpretability.

Our Concerns: Bias and Variance

- There are always two competing forces that govern the choice of learning method i.e. bias and variance.
- **Bias** of learning methods refers to the error that is introduced by modeling a real life problem (that is usually extremely complicated) by a much simpler model.
 - » For example, linear regression assumes that there is a linear relationship between Y and X . It is unlikely that, in real life, the relationship is exactly linear so some bias will be present. Suits (Bias towards) specific 'linear' problems

Our Concerns: Bias and Variance (cont.)

- **Variance** of learning methods refers to how much your estimate for f would change by if you had a different training data set.
 - » The more flexible a method is the more variance it has.
 - » The more variance/complex a method is the less bias it will generally have.



Our Concerns: Bias and Variance (cont.)

- Dimensionality reduction and feature selection can decrease variance by simplifying models. Similarly, a larger training set tends to decrease variance. Adding features (predictors) tends to decrease bias, at the expense of introducing additional variance. Learning algorithms typically have some tunable parameters that control bias and variance, e.g.:
 - » (Generalized) linear models can be regularized to decrease their variance at the cost of increasing their bias.
 - » In artificial neural networks, the variance increases and the bias decreases with the number of hidden units. Like in GLMs, regularization is typically applied.
 - » In k-nearest neighbor models, a high value of k leads to high bias and low variance (see below).
 - » In decision trees, the depth of the tree determines the variance. Decision trees are commonly pruned to control variance.
- One way of resolving the trade-off is to use mixture models and ensemble learning. For example, boosting combines many "weak" (high bias) models in an ensemble that has lower bias than the individual models.

https://en.wikipedia.org/wiki/Bias%E2%80%93variance_tradeoff

The Classification Setting

- For a regression problem, we used the MSE to assess the accuracy of the statistical learning method
- For a classification problem we can use the error rate i.e.

$$\text{Error Rate} = \sum_{i=1}^n I(y_i \neq \hat{y}_i) / n$$

- » $I(y_i \neq \hat{y}_i)$ is an indicator function, which will give 1 if the condition $(y_i \neq \hat{y}_i)$ is correct, otherwise it gives a 0.
- » Thus the error rate represents the fraction of incorrect classifications, or misclassifications

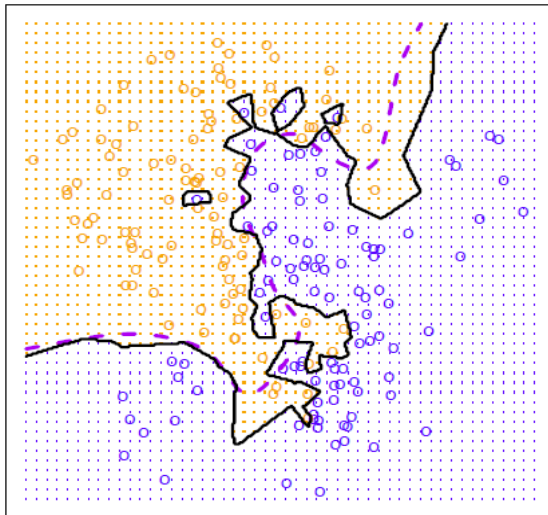
***k*-NN Classifier**

- *k* Nearest Neighbors is a flexible classifier.
 - » For any given X we find the k closest neighbors to X in the training data, and examine their corresponding Y . The prediction will be determined by the majority of the Y 's.
- Model flexibility is determined by k
 - » The smaller that k is, the more flexible & complex (more variance, less bias) the model will be.
 - » Larger k , less complex model, more generalization (smoother prediction)

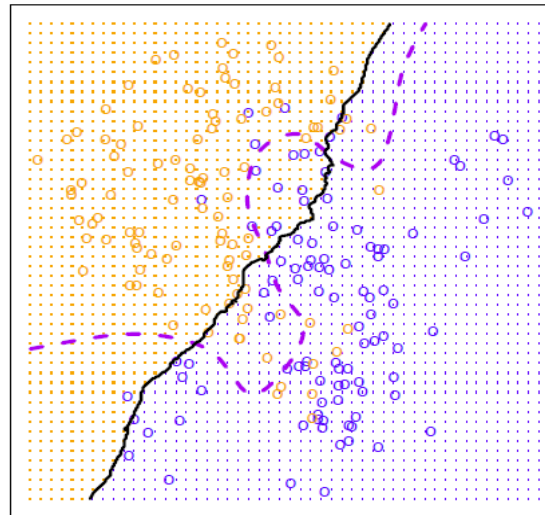
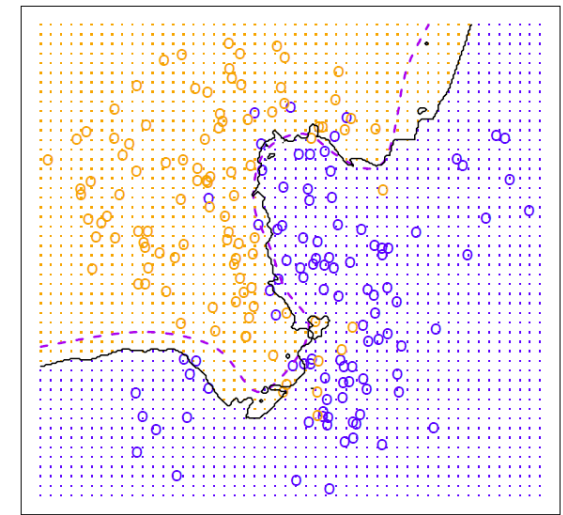
$K = 10$, $K = 1$ and $K = 100$

- A larger k makes the boundary closer to linear
 - » Which of them can potentially have a better generalization?

- Highest model flexibility and complexity
- Lowest computation complexity

KNN: $K=1$ 

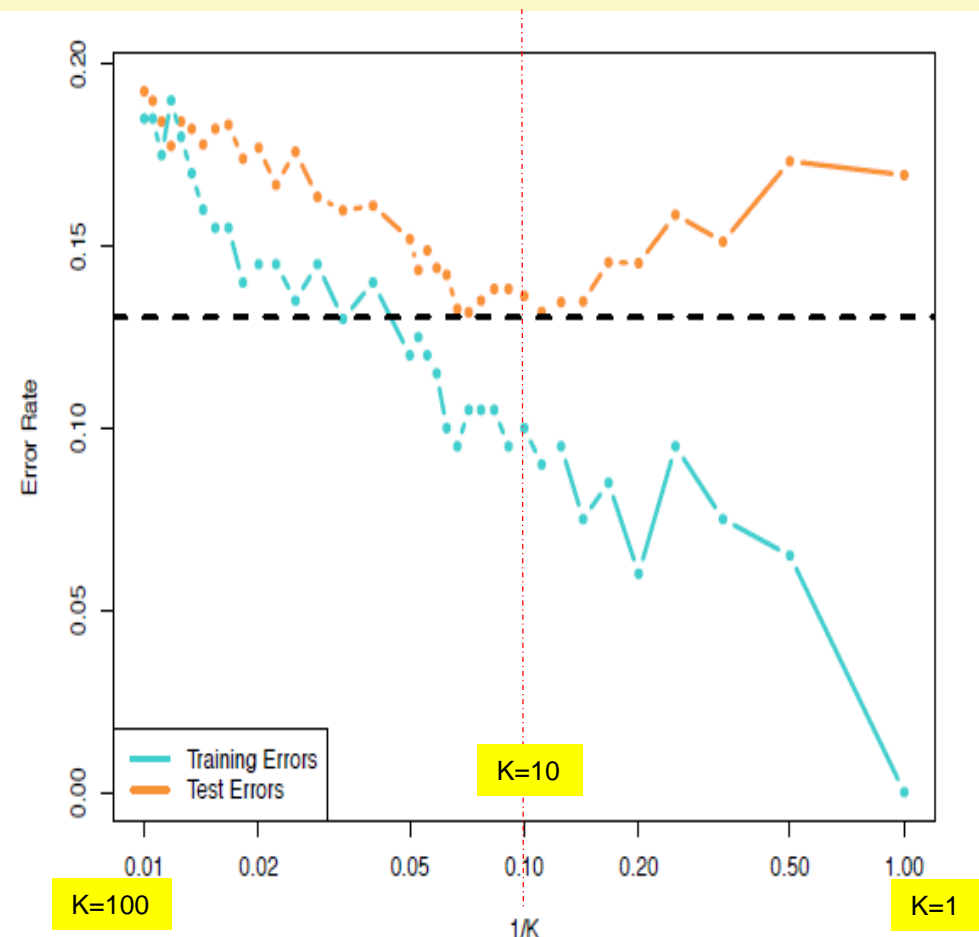
- Lowest model flexibility and complexity
- Highest computation complexity

KNN: $K=100$ KNN: $K=10$ 

[Source: G. James, D. Witten, T. Hastie, R. Tibshirani, "An Introduction to Statistical Learning"]

Training vs. Test Error Rates on the Simulated Data

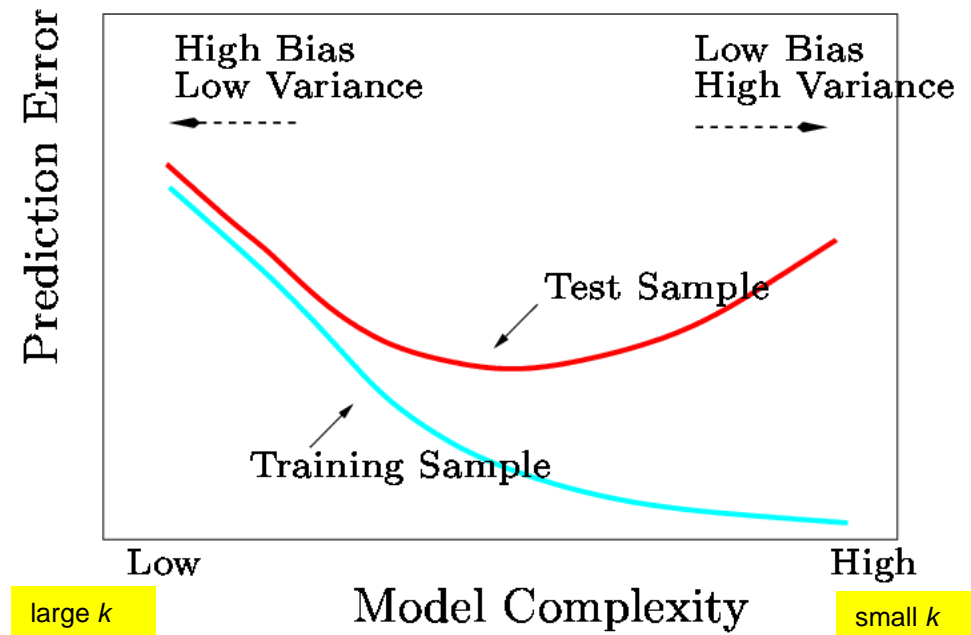
- Notice that training error rates keep going down as k decreases
 - » i.e. the flexibility increases and the bias decreases
- However, the test error rate at first decreases but then starts to increase again.



[Source: G. James, D. Witten, T. Hastie, R. Tibshirani, "An Introduction to Statistical Learning"]

A Fundamental Picture

- In general training errors will always decline.
- Test errors will decline at first (in bias dominate) but will then start to increase again (in variance dominate).



[Source: G. James, D. Witten, T. Hastie, R. Tibshirani, "An Introduction to Statistical Learning"]

Either too high or too low complexity of model will not lead to good generalization (prediction performance for test data).

Linear Regression

The Linear Regression Model

$$Y_i = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \square + \beta_p X_p + \varepsilon$$

- The parameters in the linear regression model are very easy to interpret.
 - » β_0 is the average value for Y if all the X 's are zero
 - » β_j is the average increase in Y when X_j is increased by one and all other X 's are held constant.
- 👉 When all the β 's are known, it is called **true regression line**
 - ♦ However, with the error term ε , which was generated from a normal distribution with mean zero, even if we knew the true regression line, we would not be able to perfectly predict Y from X 's.

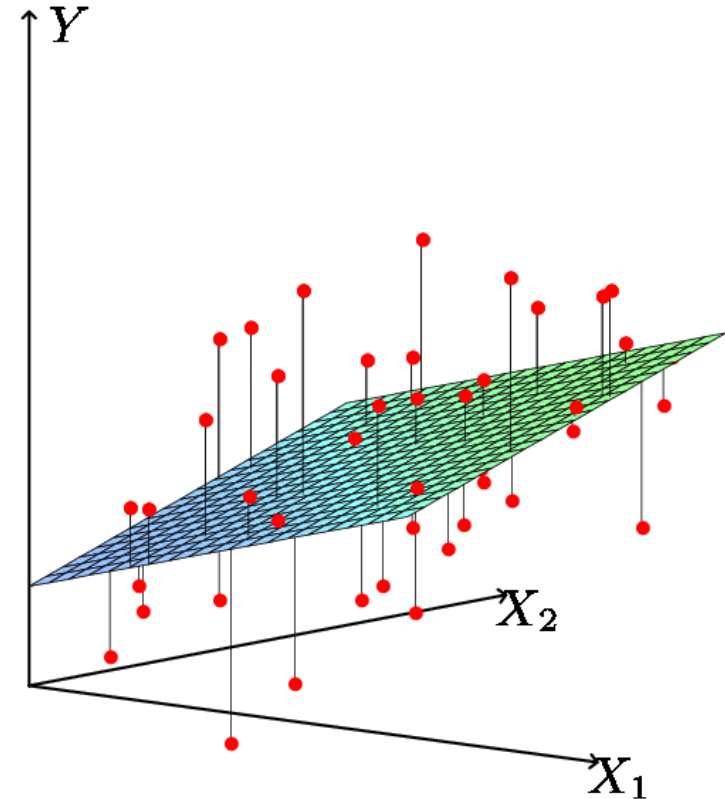
Linear Regression in Fitting

- The learning method estimates the parameters using least squares i.e. minimize MSE

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$= \frac{1}{n} \sum_{i=1}^n (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_1 - \dots - \hat{\beta}_p x_p)^2$$

Where \hat{y}_i is the prediction our method gives for the observation in our training data.



[Source: G. James, D. Witten, T. Hastie, R. Tibshirani, "An Introduction to Statistical Learning"]


Measures for Model

- *Mean squared error, MSE*

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_1 - \dots - \hat{\beta}_p x_p)^2$$

- *Residual sum of squares, RSS*

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$


Prediction

» The amount of variability that is left unexplained after performing the regression

Measures for Model (cont.)

- Residual standard error, RSE

$$RSE = \sqrt{\frac{1}{n-2} RSS} = \sqrt{\frac{1}{n-2} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

- » Roughly speaking, it is the average amount that the response will deviate from the true regression line
- » Provides an absolute measure of lack of fit of the model

- (*) Total sum of squares, TSS

- ♦ The **total variance** in the response Y, can be thought of as the amount of variability inherent in the response before the regression is performed

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

Mean(y)

Measures for Model (cont.)

- R^2 Statistic

» The proportion of variance explained, always between 0 ~ 1

$$R^2 = 1 - \frac{RSS}{TSS} \approx 1 - \frac{\text{Ending - variance}}{\text{Starting - variance}}$$

- » a number close to 1 indicates that a large portion of the variability in the response has been explained by the regression
- » a number near 0 indicates the regression did not explain much of the variability in the response, i.e. does not tell much more than an average of response

(*) *squared correlation, r^2* in simple linear regression setting, $R^2 = r^2$

Selecting & Comparing Models: *Adjusted R^2*

- *Adjusted R^2 Statistic*

- » Another popular approach for selecting among a set of models that contain different number of variables.

$$\text{adjusted } R^2 = 1 - \frac{RSS / (n - p - 1)}{TSS / (n - 1)}$$

n — sample size
 p — number of predictors

- » In theory, the model with the largest *Adjusted R^2* will have only correct variables and no noise variables
 - ♦ *Intuition behind:* once all of the correct variables have been included in the model, adding additional noise variables will lead to only a very small decrease in RSS . Adding more variables leads to an increase in p and increase in $RSS/(n-p-1)$, so a decrease in the *adjusted R^2* can be more significant.

Measures for Model (cont.)

- *F-statistic*

- » For hypothesis test to answer whether there is a relationship between the response y and predictors x_i under the linear regression setting

$$F = \frac{(TSS - RSS) / p}{RSS / (n - p - 1)}$$

- ♦ Where p is the number of predictors, n the sample size
- » The large F-statistic (we expect $F > 1$) suggests that *at least* one of the predictors must be related to the response

Measures for Coefficient

- There are measures applied to individual coefficients of a regression model

$$t_{\hat{\beta}} = \frac{\hat{\beta} - \beta_0}{\text{s. e.}(\hat{\beta})}$$

» *t-statistic*

- ♦ t = estimated coefficient - hypothesized value / $\text{SE}(\text{estimated-coefficient})$
 - SE: standard error of the estimate

» *p-value*

- ♦ gives the **probability** of observing any value $\geq |t|$, assuming the *null hypothesis* H_0 is true
 - H_0 : $\beta_i = 0$, there is NO relationship between X_i and Y
- ♦ A small p-value indicates that there is an association between the predictor and the response: $|t| = 0.05$

Lab 1: Linear Regression

- Objectives

- » Get familiar with the R / Python commands for performing and analyzing
 - ◆ simple linear regression
 - ◆ multiple linear regression
- » Understand the performance summary
- » R user: [here](#) & [here](#)
- » Python user: [here](#) & [here](#)



Workshop: Advertising

- Data set
 - » Advertising.csv (contained in library “ISLR”)
- Objective
 - » Further understand the concepts and applications of linear regression
 - » R user: [here](#) [Amit]
 - » Python user: [here](#) [Jordi Warmenhoven]

Optional Advanced Workshop

- Analysis of Credit Approval Data

[Ryan Kuhn]



Reducible and Irreducible Error

- The accuracy of \hat{Y} as a prediction for Y depends on two quantities
 - » *reducible error* caused by the inaccuracy of \hat{f} as an imperfect estimate of f
 - ♦ It is reducible, as we can potentially improve the accuracy of \hat{f} by using the most appropriate learning technique
 - » *irreducible error* caused by the error term ε
 - ♦ This quantity may contain unmeasured variables that are useful in predicting Y , or unmeasurable variance
- 👉 The focus of our discussion is on techniques of learning with the aim of minimizing the reducible error. The irreducible error will always provide an upper bound on the accuracy of our prediction for Y . The bound is almost always unknown in practice.

Prediction with Uncertainty

- When we fit the multiple regression model, and apply the model to predict the response Y on the basis of a set of the values of predictor variables
 - » Three sorts of uncertainty associated with this prediction
 - ◆ *Reducible error* from the coefficients estimate
 - *confidence interval* to determine how close \hat{Y} will be to $f(X)$
 - ◆ *Model bias*
 - ◆ *Irreducible error*
 - *prediction interval* to answer how much will Y vary from \hat{Y}

Prediction with Uncertainty (cont.)

- Assume using the linear regression model with two predictors: *TV* and *Radio*.
 - » Given that \$100,000 is spent on *TV* and \$20,000 is spent on *Radio* advertising in *each city*, we get
 - ♦ the 95% *confidence interval* for *average Sales*
- (*) if we collect a large number of data sets like the Advertising data set, and we construct a confidence interval for the average Sales on the basis of each data set (given \$100, 000 in TV and \$20,000 in Radio advertising), then 95% of these confidence intervals will contain the true value of average Sales

Prediction with Uncertainty (cont.)

(cont.)

- » Given that \$100,000 is spent on *TV* and \$20,000 is spent on *Radio* advertising in a *particular city*, we get
 - ♦ the 95% *prediction interval* for *Sales for this city*
- » both confidence and prediction intervals should be centered around the same value, but the prediction interval will be wider than the confidence interval, reflecting the increased uncertainty about Sales for a particular city in comparison to the average Sales over many locations

References

- Flach, P., “Machine Learning, the Arts and Science of Algorithms that Make Sense of Data”, Cambridge University Press, 2012
- G. James, D. Witten, T. Hastie & R. Tibshirani, “An Introduction to Statistical Learning with Applications in R”, Springer, 2013
- Russell, S. & P. Norvig, “Artificial Intelligence: A Modern Approach”, Prentice Hall, 2009
- <http://techemergence.com/machine-learning-in-robotics/>