# Course: Analytics, Machine Learning, and the Digital Economy

## Unsupervised learning

Lecturer Radjabova Dilnora

# Tree Induction

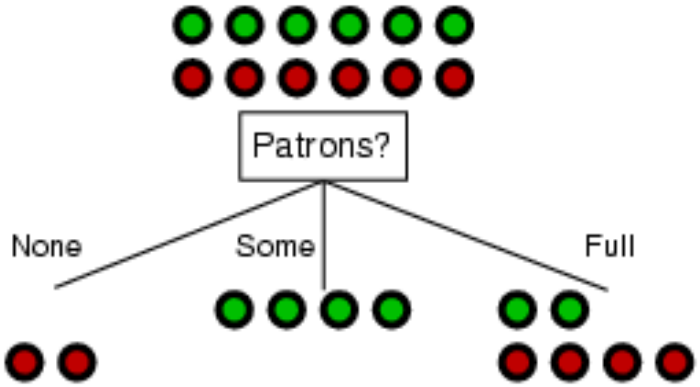
- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

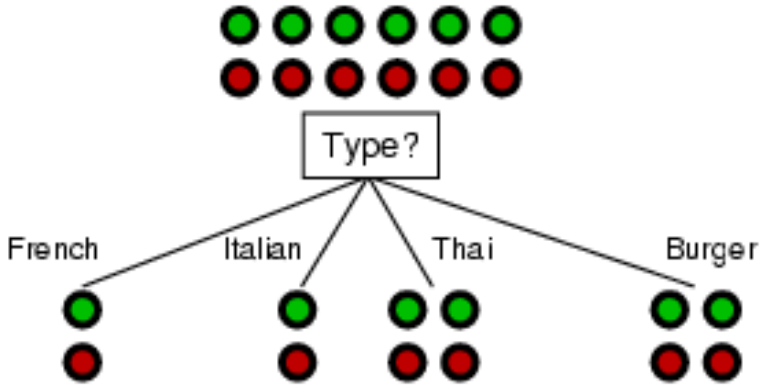
# Tree Induction

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

# Choosing an attribute

- Idea: a good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative"




- *Patrons?* is a better choice

# How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

| C0: 5<br>C1: 5 | C0: 9<br>C1: 1 |
|---|---|
| **Non-homogeneous,**<br>**High degree of impurity** | **Homogeneous,**<br>**Low degree of impurity** |

# Measures of Node Impurity

- Information Gain
- Gini Index
- Misclassification error

**Choose attributes to split to achieve minimum impurity**

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

- **Select the attribute with the highest information gain**
- Let $p_i$ be the probability that an arbitrary tuple in D belongs to class $C_i$, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$I(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

# Information gain

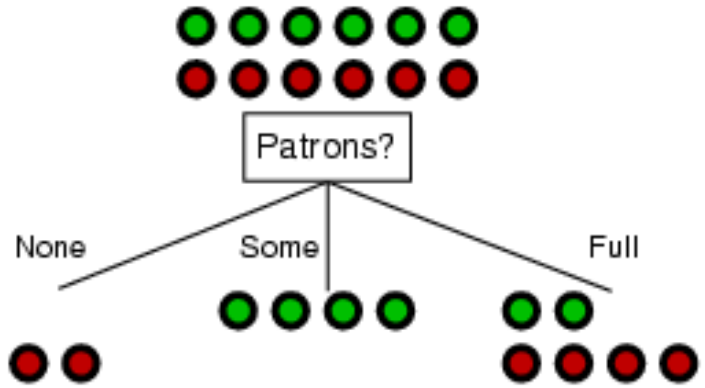
For the training set, $p = n = 6$, *I(6/12, 6/12) = 1* bit

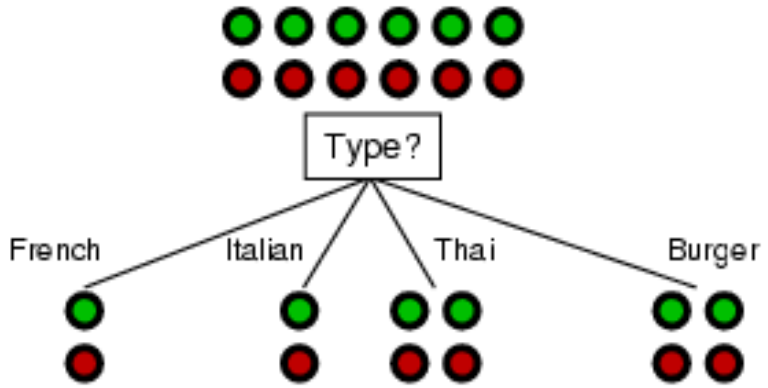
Consider the attributes *Patrons* and *Type* (and others too):

$$IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .0541 \text{ bits}$$

$$IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

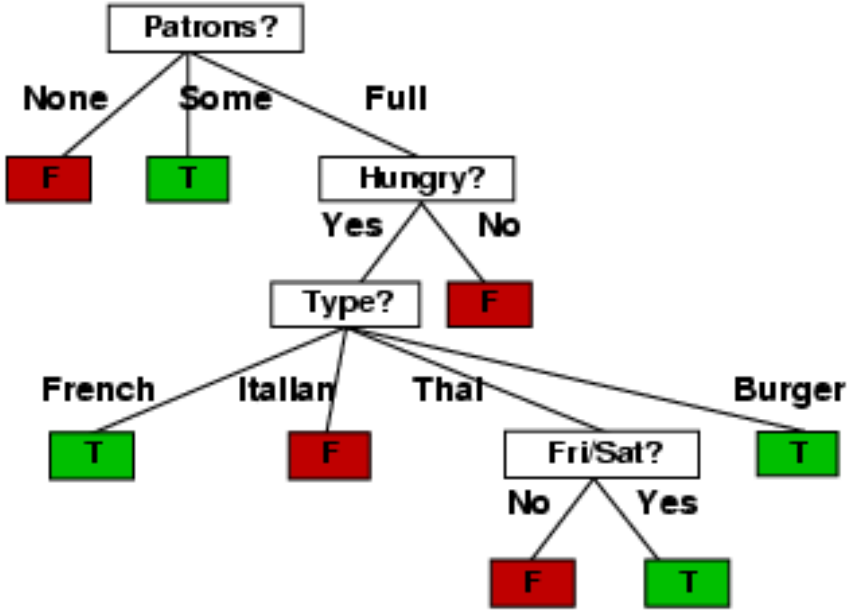
*Patrons* has the highest IG of all attributes and so is chosen by the DTL algorithm as the root



# Example contd.

- Decision tree learned from the 12 examples:




- Substantially simpler than “true” tree---a more complex hypothesis isn’t justified by small amount of data

# Measure of Impurity: GINI
## (CART, IBM IntelligentMiner)

- Gini Index for a given node t :

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

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

  - Maximum (1 - $1/n_c$) when records are equally distributed among all classes, implying least interesting information
  - **Minimum (0.0)** when all records belong to one class, **implying most interesting information**

| C1 | **0** | C1 | **1** | C1 | **2** | C1 | **3** |
|---|---|---|---|---|---|---|---|
| C2 | **6** | C2 | **5** | C2 | **4** | C2 | **3** |
| **Gini=0.000** | | **Gini=0.278** | | **Gini=0.444** | | **Gini=0.500** | |

# Splitting Based on GINI

- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, $n_i$ = number of records at child i,

n = number of records at node p.

# Comparison of Attribute Selection Methods

- The three measures return good results but
  - Information gain:
    - biased towards multivalued attributes

# Comparison of Attribute Selection Methods

- The three measures return good results but
  - Gain ratio:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others

# Comparison of Attribute Selection Methods

- The three measures return good results but
  - Gini index:
    - biased to multivalued attributes
    - has difficulty when # of classes is large
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

# Example Algorithm: C4.5

- Simple depth-first construction.
- Uses Information Gain
- Sorts Continuous Attributes at each node.
- Needs entire data to fit in memory.
- Unsuitable for Large Datasets.

- You can download the software from Internet

# Decision Tree Based Classification

- Advantages:
  - **Easy** to construct/implement
  - Extremely **fast** at classifying unknown records
  - Models are **easy to interpret for small-sized trees**
  - **Accuracy is comparable** to other classification techniques for many simple data sets
  - Tree models make no assumptions about the distribution of the underlying data : **nonparametric**
  - Have a **built-in feature selection** method that makes them immune to the presence of useless variables

# Decision Tree Based Classification

- Disadvantages
  - Computationally **expensive to train**
  - **Some decision trees can be overly complex** that do not generalise the data well.
  - **Less expressivity**: There may be concepts that are hard to learn with limited decision trees

# 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

# Overfitting and Tree Pruning

- Two approaches to avoid overfitting
  - **Prepruning:** 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
  - **Postpruning:** 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"

# Rule-Based Classifier

- Classify records by using a collection of "if…then…" rules
- Rule: (*Condition*) $\rightarrow$ *y*
  - where
    - *Condition* is a conjunctions of attributes
    - *y* is the class label
  - *LHS*: rule antecedent or condition
  - *RHS*: rule consequent
  - Examples of classification rules:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) $\rightarrow$ Evade=No

# Rule-based Classifier (Example)

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---|---|---|---|---|---|
| human | warm | yes | no | no | mammals |
| python | cold | no | no | no | reptiles |
| salmon | cold | no | no | yes | fishes |
| whale | warm | yes | no | yes | mammals |
| frog | cold | no | no | sometimes | amphibians |
| komodo | cold | no | no | no | reptiles |
| bat | warm | yes | yes | no | mammals |
| pigeon | warm | no | yes | no | birds |
| cat | warm | yes | no | no | mammals |
| leopard shark | cold | yes | no | yes | fishes |
| turtle | cold | no | no | sometimes | reptiles |
| penguin | warm | no | no | sometimes | birds |
| porcupine | warm | yes | no | no | mammals |
| eel | cold | no | no | yes | fishes |
| salamander | cold | no | no | sometimes | amphibians |
| gila monster | cold | no | no | no | reptiles |
| platypus | warm | no | no | no | mammals |
| owl | warm | no | yes | no | birds |
| dolphin | warm | yes | no | yes | mammals |
| eagle | warm | no | yes | no | birds |

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

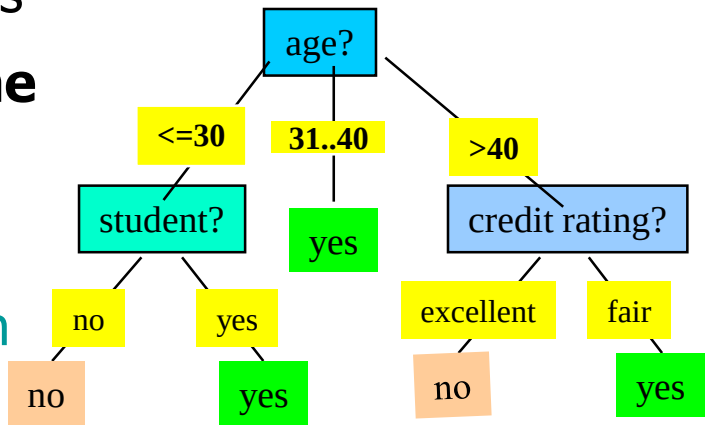
R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

# Rule Extraction from a Decision Tree

- Rules are easier to understand than large trees
- **One rule is created for each path from the root to a leaf**
- Each attribute-value pair along a path forms a conjunction: the leaf holds the class prediction




- Example: Rule extraction from our *buys_computer* decision-tree

IF *age* = young AND *student* = *no* THEN *buys_computer* = *no*

IF *age* = young AND *student* = *yes* THEN *buys_computer* = *yes*

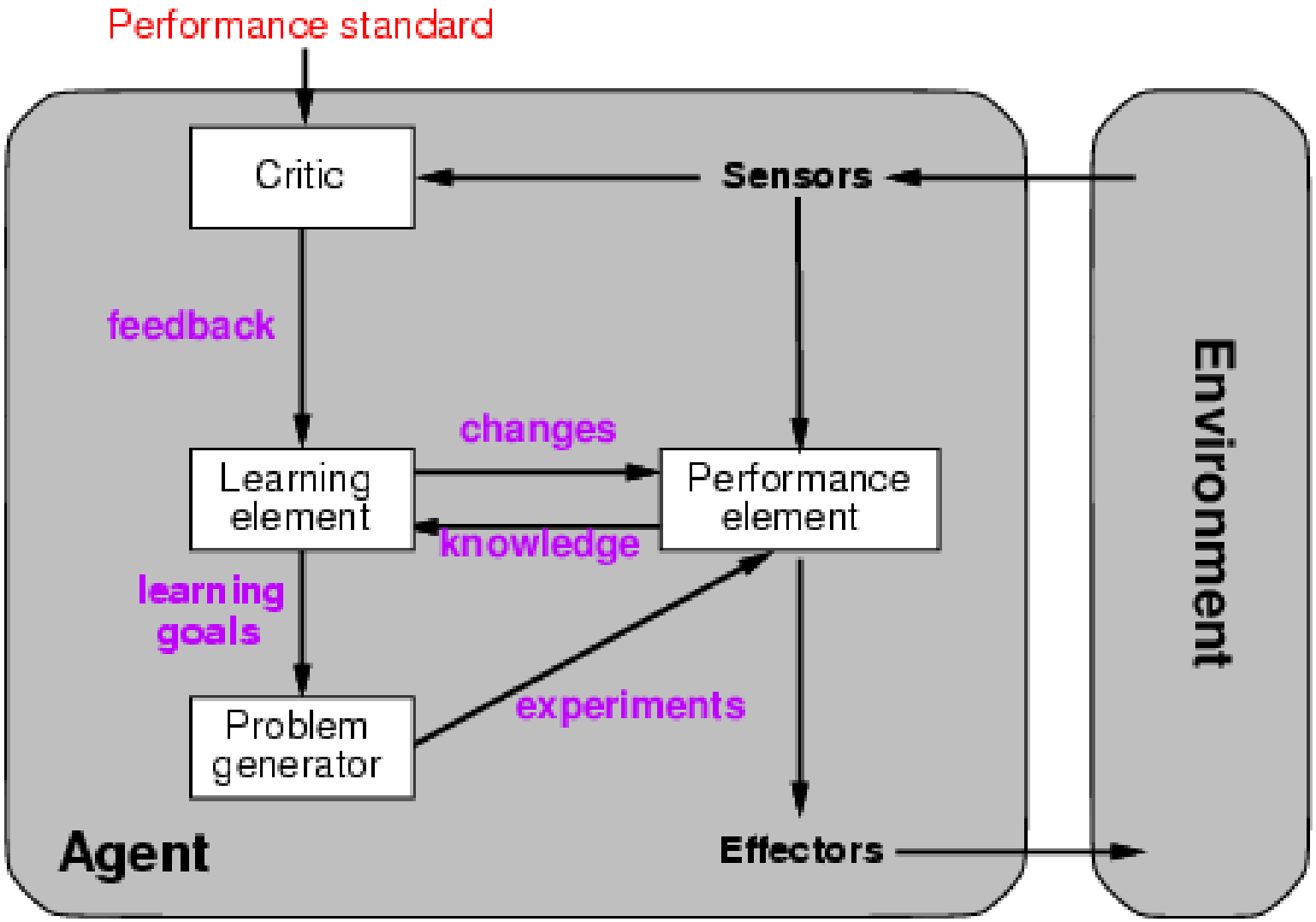
IF *age* = mid-age THEN *buys_computer* = *yes*

IF *age* = old AND *credit_rating* = *excellent* THEN *buys_computer* = *yes*

IF *age* = young AND *credit_rating* = *fair* THEN *buys_computer* = *no*

# Extra Slides

# Learning agents



# Classification(Sınıflandırma)

- **IDEA: Build a model based on past data to predict the class of the new data**
- Given a collection of records (*training set* )
  - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.

- **Goal:** previously unseen records should be assigned a class as accurately as possible.

# Hypothesis spaces

How many distinct decision trees with *n* Boolean attributes?

= number of Boolean functions

= number of distinct truth tables with $2^n$ rows = $2^{2^n}$

- E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees

How many purely conjunctive hypotheses (e.g., *Hungry* $\wedge$ $\neg$*Rain*)?

- Each attribute can be in (positive), in (negative), or out
  $\Rightarrow$ $3^n$ distinct conjunctive hypotheses
- More expressive hypothesis space
  - increases chance that target function can be expressed
  - increases number of hypotheses consistent with training set
    $\Rightarrow$ may get worse predictions

# Using information theory

- To implement `Choose-Attribute` in the DTL algorithm
- Information Content (Entropy):

$$I(P(v_1), \ldots, P(v_n)) = \Sigma_{i=1} -P(v_i) \log_2 P(v_i)$$

- For a training set containing *p* positive examples and *n* negative examples:

$$I(\frac{p}{p+n}, \frac{n}{p+n}) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

# Information gain

- A chosen attribute *A* divides the training set *E* into subsets $E_1, \ldots, E_v$ according to their values for *A*, where *A* has $v$ distinct values.

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

- Information Gain (IG) or reduction in entropy from the attribute test:

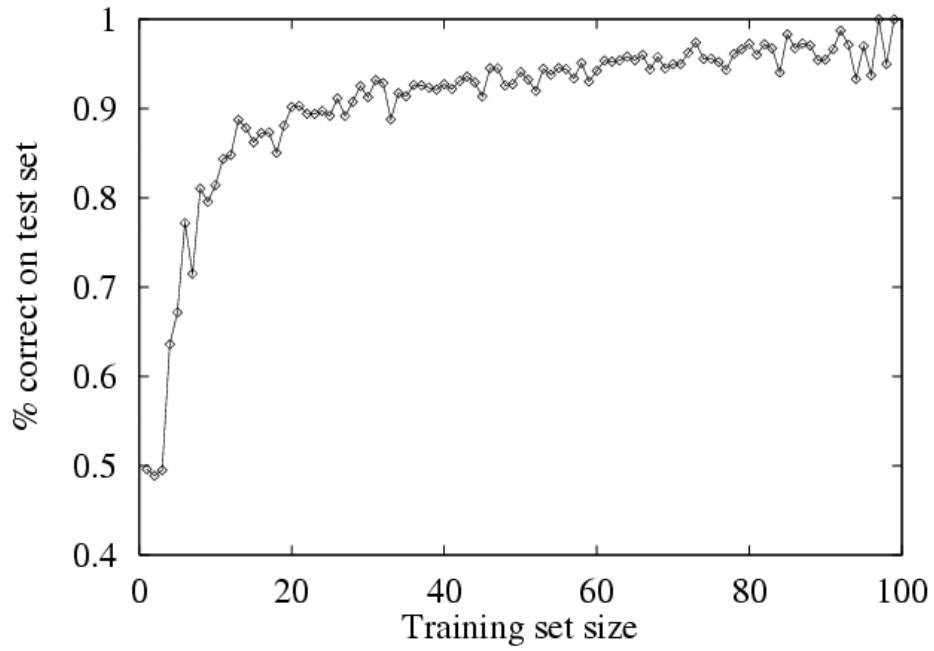
$$IG(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - remainder(A)$$

- Choose the attribute with the largest IG

# Performance measurement

- How do we know that $h \approx f$ ?
  1. Use theorems of computational/statistical learning theory
  2. Try $h$ on a new test set of examples
     (use same distribution over example space as training set)

Learning curve = % correct on test set as a function of training set size



# Reference


- 1. Shalev-Shwartz and Ben-David. Understanding Machine Learning: From Theory to Algorithms (Cambridge University Press, 2014)
- 2. Daum´e. A Course in Machine Learning.
- 3. The Art of Statistics: How to Learn from Data by David Shpigelter
- 4. Learning From Data – January 1, 2012 by Yaser S. Abu-Mostafa (Author), Malik Magdon-Ismail (Author), Hsuan-Tien Lin (Author)
- 5. Statistics: The Art and Science of Learning from Data by Alan Agresti
- 6. Learning From Data: An Introduction To Statistical Reasoning by M.Glenber.
- 7. Statistics: Learning from Data (with JMP Printed Access Card) by Rocky Pek
- 8. The Elements of Statistical Learning by Gerim Garold
- 9. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 2nd Edition
- by Aurélien Géron (Author)