Université Mohamed Boudiaf – M'sila
Département Informatique
Master Intelligence Artificielle

2020-2021

Chapitre 6
Arbres de Décision

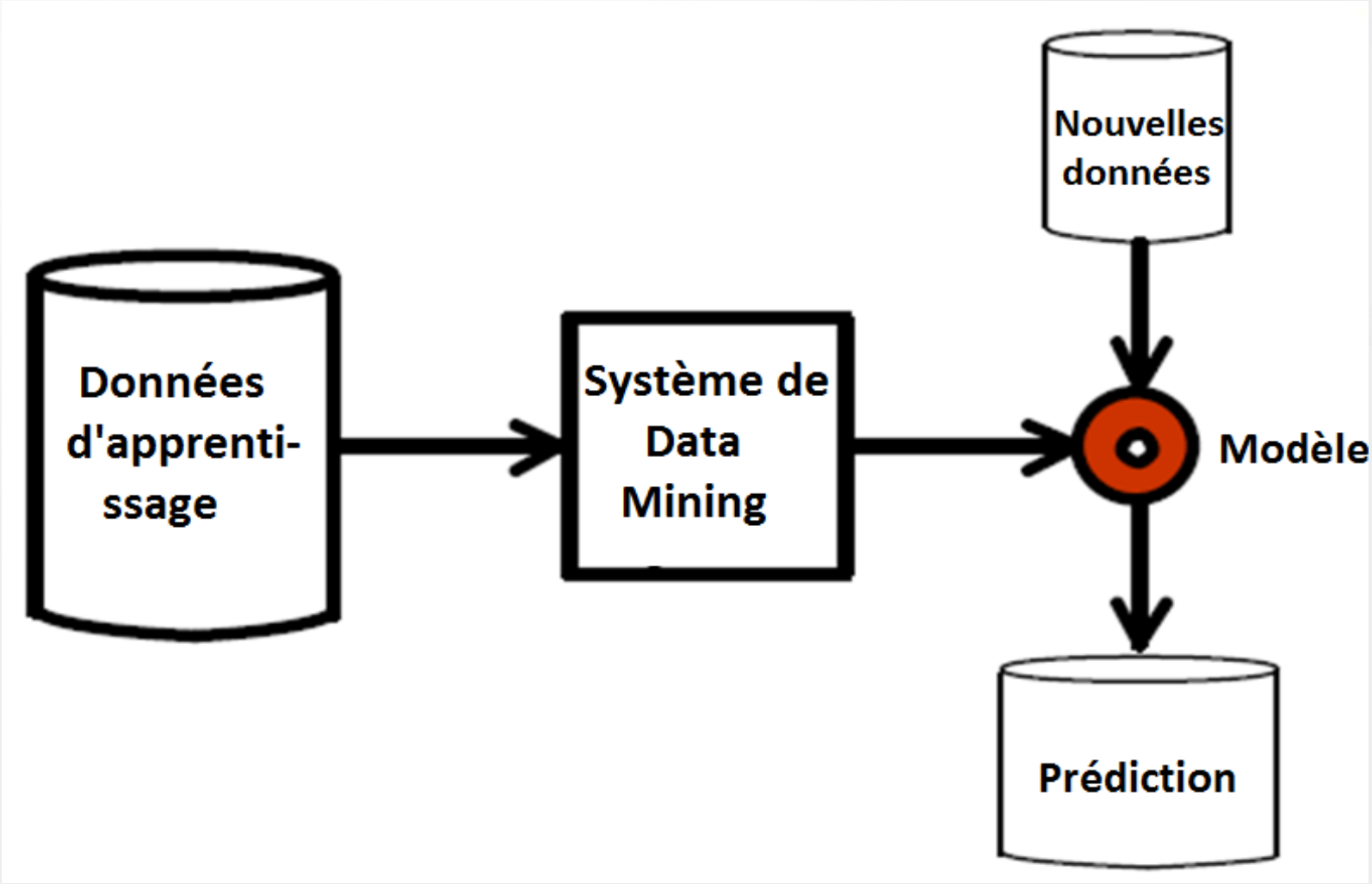
Dr. Mehenni Tahar

Définition du problème de Classification

Etant donnée une base de données $D=\{t_1,t_2,\ldots,t_n\}$ et un ensemble de classes $C=\{C_1,\ldots,C_m\}$, *le problème de Classification* est de définir une application

$f: D \rightarrow C$ où chaque t_i est affecté à une seule classe.

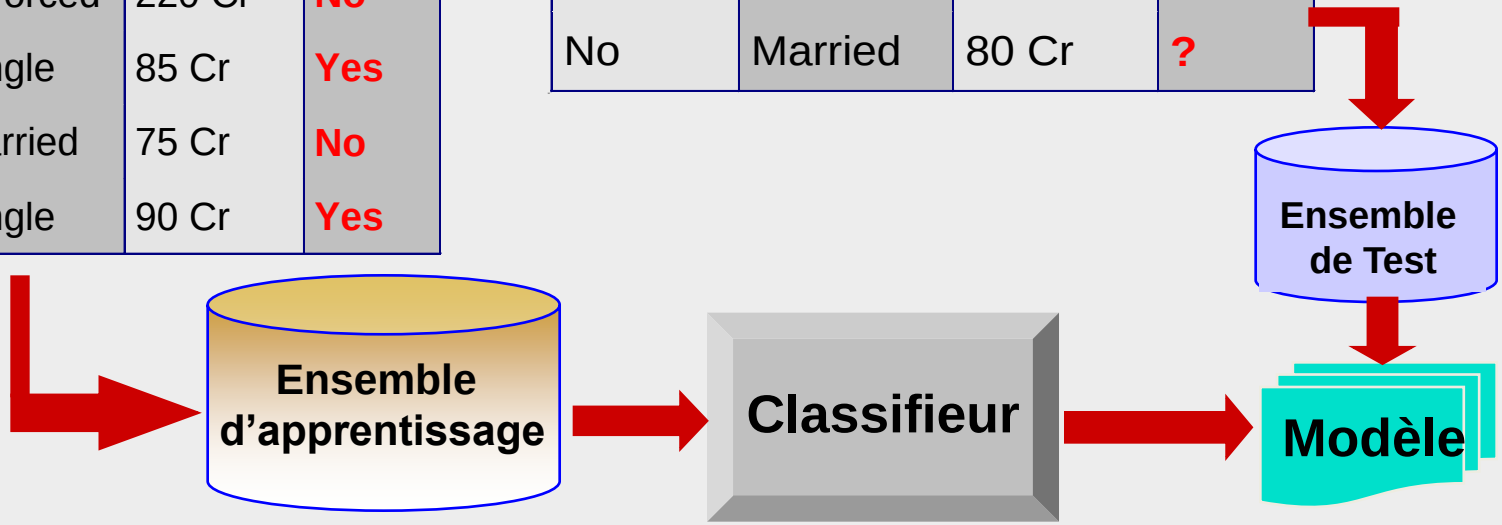
Modèle de Classification



Exemple: Carte de Crédit

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125 Cr	No
2	No	Married	100 Cr	No
3	No	Single	70 Cr	No
4	Yes	Married	120 Cr	No
5	No	Divorced	95 Cr	Yes
6	No	Married	60 Cr	No
7	Yes	Divorced	220 Cr	No
8	No	Single	85 Cr	Yes
9	No	Married	75 Cr	No
10	No	Single	90 Cr	Yes

Refund	Marital Status	Taxable Income	Cheat
No	Single	75 Cr	?
Yes	Married	50 Cr	?
No	Married	150 Cr	?
Yes	Divorced	90 Cr	?
No	Single	40 Cr	?
No	Married	80 Cr	?



Classification vs. Prédiction

- **Classification**
  - Prédicte (prévoit) les classes
  - Classifie les données (construit un modèle) basé sur l'ensemble d'apprentissage et les classes et l'utilise pour déterminer les classes de nouvelles données.

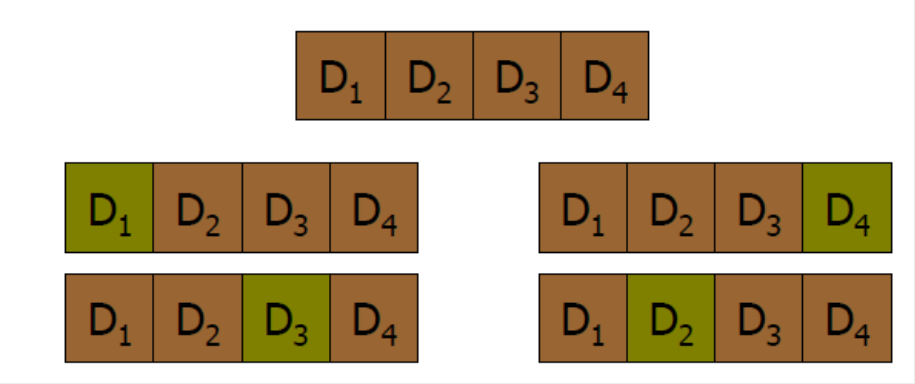
- **Prédiction (ou prévision ou régression)**
  - Des modèles de fonctions à valeurs continues, i.e., prédiction des valeurs absentes dans le procéssus de nettoyage.

Classification—Un Procéssus à trois étapes

- **Construction du modèle:** description d'un ensemble de classes prédeterminées
  - Chaque tuple/exemple doit être affecté à une classe parmi les classes prédeterminées
  - L'ensemble de tuples/exemples est appelé ensemble d'apprentissage
  - Le modèle est représenté sous une forme bien définie: règles de classification, arbre de décision, ou formules mathématiques
- **Test de la précision (accuracy) du modèle**
  - La classification connue (définie auparavant) d'un exemple de test est comparée avec la classification déterminée par le modèle
  - Le taux d'erreur (Accuracy rate) est le pourcentage des exemples de test qui sont correctement classés par le modèle
  - L'ensemble de test doit être indépendent de l'ensemble d'apprentissage
  - Si le taux d'erreur est acceptable, le modèle est utilisé pour la classification des objets dont les classes sont inconnues.
- **Utilisation (ou exploitation) du modèle:** dans l'ojectif de classer (ou classifier) les futurs ou nouvels objets

Estimation des taux d'erreurs (accuracy)

- **Partitionnement** : apprentissage et test (ensemble de données important)
  - Utiliser 2 ensembles indépendents, e.g., ensemble d'apprentissage(2/3), ensemble test (1/3)**:**
- **Validation croisée** (ensemble de données modéré)
  - Diviser les données en k sous-ensembles
  - Utiliser k-1 sous-ensembles comme données d'apprentissage et un sous-ensemble comme données test



Evaluation des méthodes de classification

- Taux d'erreur (Accuracy)
- Temps d'exécution
  - Temps de construction du modèle (temps d'apprentissage)
  - Temps d'utilisation du modèle (temps de classification/prédiction)
- Robustesse: Prise en charge des données manquantes et du bruit
- Extensibilité: efficacité du modèle pour des données issues des bases de données et stockées dans le disque dur
- Interprétabilité: Compréhension et capacité d'explication des résultats donnés par le modèle
- Simplicité

Méthodes de Classification

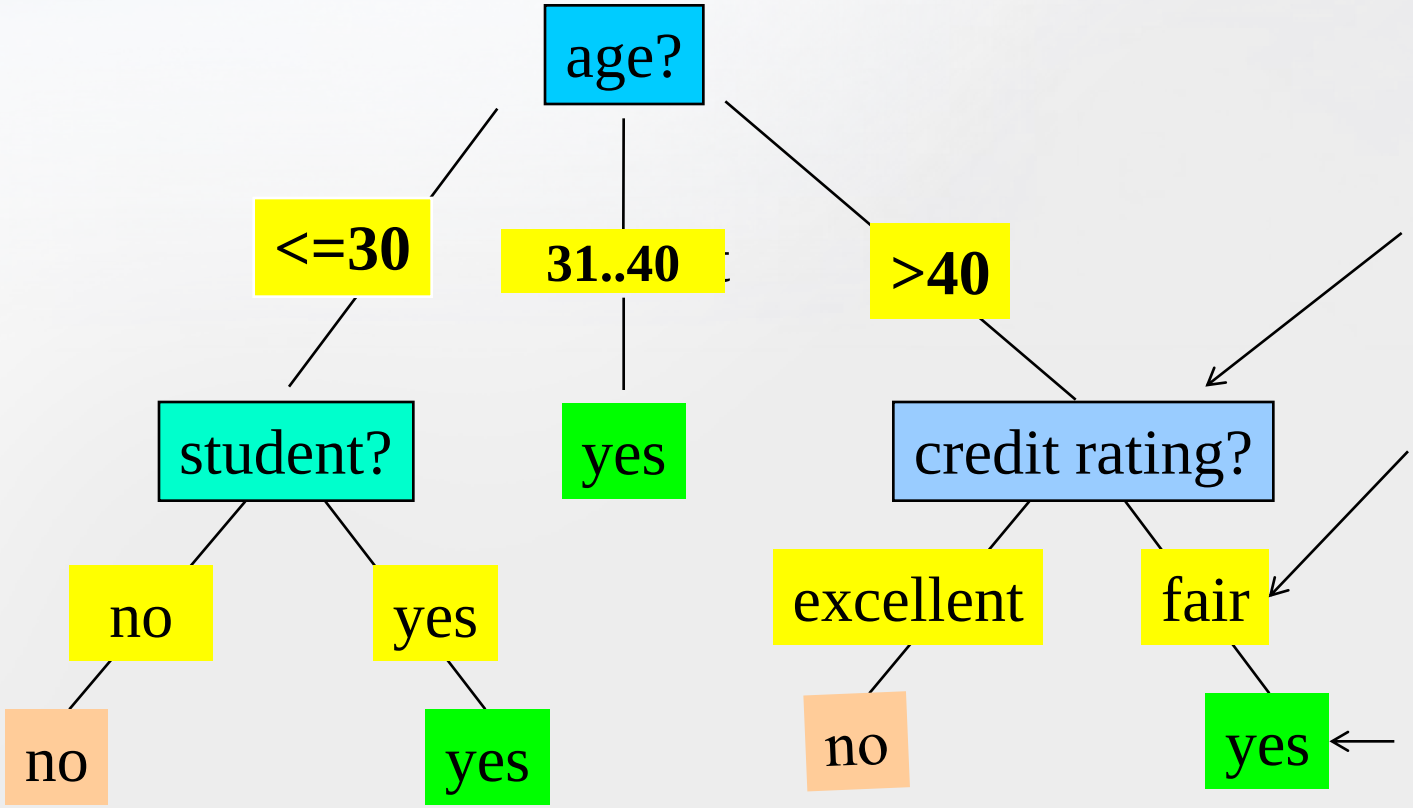
- **Arbres de décision**
- Méthode K-NN (plus proche voisin)
- Réseaux de neurones
- Classification bayésienne

Arbre de Décision: Un ensemble d'apprentissage

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

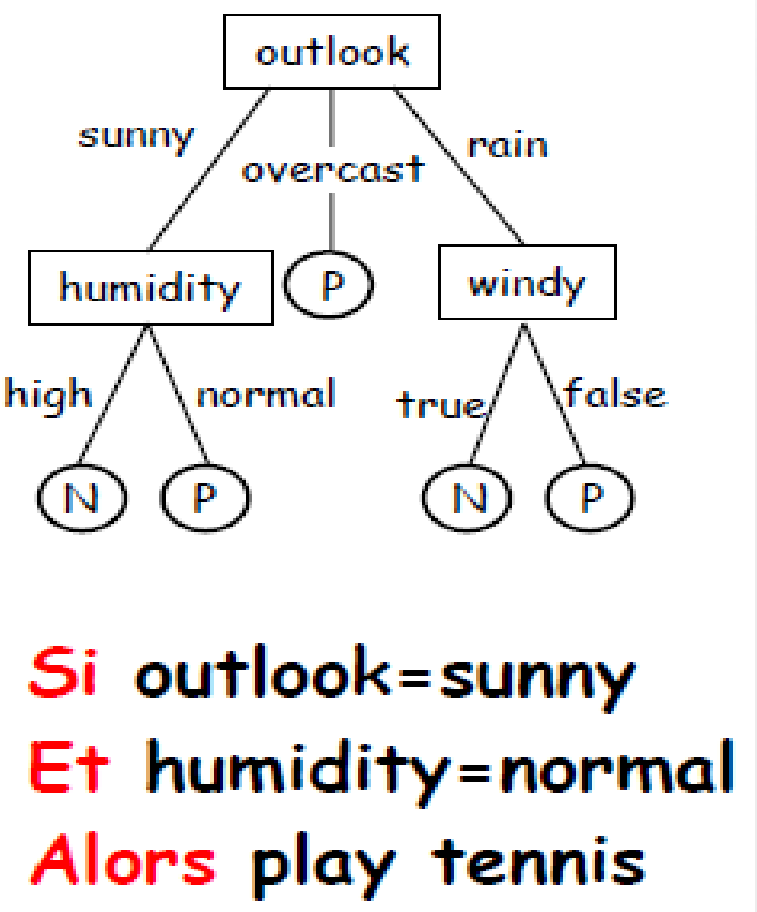
Données des personnes pouvant acquérir un ordinateur ou non

Modèle : Un arbre de Décision pour l'achat d'orinateur



- Chaque nœud interne teste un attribut
- Chaque branche correspond à une valeur de l'attribut
- Chaque feuille représente une classe

De l'arbre de décision aux règles de classification



- une **règle** est générée pour chaque **chemin** de l'arbre (de la racine à une feuille)
- Les paires attribut-valeur d'un chemin forment une conjonction
- Le nœud terminal représente la classe prédite
- Les règles sont généralement plus faciles à comprendre que les arbres

Génération de l'arbre de décision

Deux phases dans la génération de l'arbre :

- **Construction de l'arbre**
  - Arbre peut atteindre une taille élevée
- **Elaguer l'arbre (Pruning)**
  - Identifier et supprimer les branches qui représentent du "bruit"
  - Améliorer le taux d'erreur

Algorithme de classification (1)

Construction de l'arbre

- Au départ, toutes les instances d'apprentissage sont à la racine de l'arbre
- Sélectionner un attribut et choisir un test de séparation(split)sur l'attribut, qui sépare le "mieux" les instances.
- La sélection des attributs est basée sur une heuristique ou une mesure statistique.
- Partitionner les instances entre les noeuds fils suivant la satisfaction des tests logiques
- Traiter chaque noeud fils de façon récursive
- Répéter jusqu'à ce que tous les noeuds soient des terminaux. Un nœud courant est terminal si:
  - Il n'y a plus d'attributs disponibles
  - Le noeud est"pur",i.e. toutes les instances appartiennent à une seule classe,
  - Le noeud est "presque pur",i.e. la majorité des instances appartiennent à une seule classe(Ex : 95%)
  - Nombre minimun d'instances par branche (Ex : algorithme C5 évite la croissance de l'arbre, k=2 par défaut)
- Etiqueter le nœud terminal par la classe majoritaire

Algorithme de classification (2)

Elaguer l'arbre obtenu(pruning)

- Supprimer les sous-arbres qui n'améliorent pas l'erreur de la classification (accuracy) ⇨ arbre ayant un meilleur pouvoir de généralisation, même si on augmente l'erreur sur l'ensemble d'apprentissage
- Eviter le problème de sur-spécialisation(over-fitting),i.e., on a appris "par coeur" l'ensemble d'apprentissage, mais on n'est pas capable de généraliser

Sur-spécialisation (overfitting) de l'arbre de décision

- **L'arbre généré peut sur-spécialiser l'ensemble d'apprentissage**
  - Plusieurs branches
  - Taux d'erreur important pour les instances inconnues
- **Raisons de la sur-spécialisation**
  - Bruits et exceptions
  - Peu de données d'apprentissage
- **Comment éviter l'overfitting : Deux approches :**
  - **Pré-élagage :**
    - Arrêter de façon prématurée la construction de l'arbre
  - **Post-élagage :**
    - Supprimer des branches de l'arbre complet ("fully grown")
    - Convertir l'arbre en règles; élaguer les règles de façon indépendante C4.5)

Algorithme de classification: Synthèse et variantes

Construction de l'arbre - Synthèse

- Evaluation des différents branchements pour tous les attributs
- Sélection du "meilleur" branchement" et de l'attribut "gagnant"
- Partitionner les données entre les fils
- Construction en largeur(C4.5) ou en profondeur(SPLIT)

Questions critiques :

- Formulation des tests de branchement
- Mesure de sélection des attributs

Algorithme de base

- Construction récursive d'un arbre de manière "diviser-pour-régner" descendante
- Attributs considérés énumératifs

Plusieurs variantes : ID3, C4.5, CART, CHAID, …
Différence principale:

- Mesure de sélection d'un attribut
- Critère de branchement (split)

Mesures de sélection d'attributs

- **Gain d'Information (ID_3, $C_{4.5}$)**
- **Indice Gini (CART)**
- Table de contingence statistique $\chi 2$ (CHAID)
- G-statistic

Mesure de Sélection de l'Attribut: Gain en Information (ID3/C4.5)

- Sélectionner l'attribut ayant le plus grand gain en information
- Soit p_i la probabilité qu'un tuple quelconque de D appartient à la classe C_i, estimée par $|C_{i,D}|/|D|$
- L'information attendue (entropie) nécessaire pour classer un tuple de D:

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

- L'information nécessaire (après utilisation de A pour diviser D en v partitions) pour classer D:

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

- L'Information gagnée (Gain en information) en branchant sur l'atttribut A

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

Sélection de l'attribut: Gain en Information

- Classe P: buys_computer = "yes"
- Classe N: buys_computer = "no"

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

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
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

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

$\frac{5}{14}I(2,3)$ signifie que "age <=30" possède 5 de 14 exemples, avec 2 yes et 3 no. D'où

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Calcul du Gain en Information pour les attributs à valeur continue

- Soit un attribut A à valeur continue
- Il faut déterminer le meilleur point de séparation (*split point*) pour A
  - Trier les valeurs de A par ordre croissant
  - Généralement, le point médiane entre chaque deux valeurs adjacentes est considéré comme un point de séparation possible.
    - $(a_i+a_{i+1})/2$ est le point médiane entre les valeurs a_i et a_{i+1}
  - Le point présentant l'entropie minimale est sélectionné comme le point de séparation (split-pointt) de A
- Séparation:
  - D1 est l'ensemble de tuples de D satisfaisant A ≤ split-point, et D2 l'ensemble de tuples de D satisfaisant A > split-point

Ratio de Gain pour un attribut (C4.5)

- La mesure du gain en Information est influencée par les attributs à valeurs larges.
- C4.5 (un successeur de ID3) utilise le ratio de Gain pour résoudre ce problème (normalisation du Gain en information)
  - GainRatio(A) = Gain(A)/SplitInfo(A)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- Exemple.

$$SplitInfo_A(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 0.926$$

  - gain_ratio(income) = 0.029/0.926 = 0.031
- L'attribut ayant le maximum ratio de gain est sélectionné comme l'attribut de séparation.

Index de Gini (CART, IBM IntelligentMiner)

- Si un ensemble de données *D contient des exemples de n* classes, l'index de Gini, *gini*(*D*) est défini par

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

où p_j est la fréquence relative de la casse *j* dans *D*

- Si un ensemble de données *D* est séparé selon A en deux sous-ensembles D_1 et D_2, l'index de G*ini*, *gini*(*D*) est défini par

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Réduction d'impurité:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- L'attribut qui donne le plus petit index de Ginin, $gini_{split}(D)$ (ou la plus grande réduction d'impurité) est choisi comme un noeud de séparation (*Il faut énumérer tout les points de séparation possibles pour chaque attribut*)

Index de Gini (CART, IBM IntelligentMiner)

- **Exemple.** D a 9 tuples avec buys_computer = "yes" et 5 avec "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Supposons que l'attribut "income" sépare D en 10 tuples dans D_1: {low, medium} et 4 tuples dans D_2 : {high}

$$gini_{income \in \{low,medium\}}(D) = \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_1)$$

$$= \frac{10}{14}(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2) + \frac{4}{14}(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2)$$
$$= 0.450$$
$$= Gini_{income \in \{high\}}(D)$$

Mais $gini_{\{medium,high\}}$ = 0.30, d'où il est le meilleur car le plus petit

- Tous les attributs sont supposés à valeur continue
- Peut utiliser d'autres outils, e.g., clustering, pour avoir les valeurs possibles de séparation
- Peut être modifié pour prendre en charge les attributs catégoriques