DOI – EXAMEN

Les exercices sont indépendants. Tout document de cours et de TD est autorisé. Le barème est donné à titre indicatif.

1 Optimisation : modélisation et décision (4 points)

La fabrique RadioIn crée deux types de radios A et B. Chaque radio produite est le fruit des efforts conjoints de trois spécialistes Pierre, Paul et Jean. Pierre travaille au plus 24 heures par semaine. Paul travaille au moins 10 heures et au plus 45 heures par semaine. Jean travaille au plus 30 heures par semaine. Les ressources nécessaires pour construire chaque type de radio ainsi que leurs prix de vente sont donnés dans la table 1 ci-dessous.

	Radio A	Radio B
Pierre	1h	2h
Paul	2h	1h
Jean	1h	3h
Prix de vente	15 euros	10 euros

TABLE 1 – Données du problème de RadioIn

On suppose que l'entreprise n'a aucun problème à vendre sa production, quelle qu'elle soit.

Q1.1 Modéliser le problème de la recherche d'un plan de production hebdomadaire maximisant le chiffre d'affaires de RadioIn sous forme d'un programme linéaire (P). Préciser clairement les variables de décision, la fonction objectif et les contraintes.

Q1.2 Donner le programme dual du programme (P).

Q1.3 La résolution par CPLEX de (P) a conduit à la solution optimale $X^* = (22, 1)$, calculer la solution du programme dual.

2 Programmation en nombres entiers (2 points)

Soit le programme en nombres entiers (P1) suivant :

$$\begin{aligned} \text{Max } & Z = 5x_1 + 4x_2 \\ & x_1 + x_2 \leq 5 \\ & 10x_1 + 6x_2 \leq 45 \\ & x_1, x_2 \geq 0, \text{ entiers} \end{aligned}$$

La relaxation de (P1) permet d'obtenir la solution optimale réelle $X^* = (3.75, 1.25)$.

Q2.1 Vérifier si $X = (3, 2)$ est une solution réalisable pour (P1).

Q2.2 Résoudre (P1).

[L'examen continue sur la page suivante]

3 Partitionnement de données (5 points)

Considérons un schéma d'un dépôt de données composé de trois tables de dimension : `Client`, `Produit` et `Temps`, et d'une table des faits : `Ventes`. Chaque tuple est associé à un `RID` (Row Identifier). La population de ce dépôt est donnée dans la figure 1. Ce schéma est exploité pour des requêtes analytiques.

Considérons la requête suivante, appelée Q, définie sur ce schéma.

Listing 1 – Requête Q

```
SELECT Count(*)
FROM Client C, Produit P, Temps T, Ventes V
WHERE C.City ='Poitiers' AND P.Type= 'Beauté'
  AND T.Mois = 'Juin'    AND P.PID = V.PID
  AND C.CID = V.CID      AND T.TID = V.TID
```

Q3.1 Proposer un schéma de fragmentation horizontale, dérivée du dépôt de données de la figure 1, qui pourrait optimiser la requête Q.

Q3.2 Quel est le résultat retourné par cette requête?

Q3.3 Montrer comment la fragmentation optimise la requête Q.

CLIENT

C_RID	CID	Nom	Ville
6	616	Gilles	Poitiers
5	515	Yves	Paris
4	414	Brice	Nantes
3	313	Ladjel	Nantes
2	212	Eric	Poitiers
1	111	Pascal	Poitiers

PRODUIT

P_RID	PID	Nom	Type
6	106	Sonoflore	Beauté
5	105	Clarins	Beauté
4	104	WebCam	Multimédia
3	103	Barbie	Jouet
2	102	Engrais	Jardinage
1	101	SlimForm	Fitness

TEMPS

T_RID	TID	Jour	Année
1	11	janvier	2003
2	22	février	2003
3	33	mars	2003
4	44	avril	2003
5	55	mai	2003
6	66	juin	2003

VENTES

V_RID	CID	PID	TID	Ventes
1	616	106	11	25
2	616	106	66	28
3	616	104	33	50
4	515	104	11	10
5	414	105	66	14
6	212	106	55	14
7	111	101	44	20
8	111	101	33	27
9	212	101	11	100
10	313	102	11	200
11	414	102	11	102
12	414	102	55	203
13	515	102	66	100
14	515	103	55	17
15	212	103	44	45
16	111	105	66	44
17	212	104	66	40
18	515	104	22	20
19	616	104	22	20
20	616	104	55	20
21	212	105	11	10
22	212	105	44	10
23	212	105	55	18
24	212	106	11	18
25	313	105	66	19
26	313	105	22	17
27	313	106	11	15

FIGURE 1 – Population du dépôt de données

4 Analyse des co-auteurs du LIAS (4 points)

La table 2 liste les publications de l'équipe IDD du LIAS depuis 2020. Pour condenser la notation, les publications avec les mêmes auteurs sont agrégées. Par exemple, la première ligne indique qu'il existe sept publications dont l'unique auteur [1] est AH.

AH	AM	BC	LB	MB	SB	SJ	count
1	0	0	0	0	0	0	7
0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	15
0	0	0	0	0	1	0	1
1	1	0	0	0	0	0	1
0	1	0	1	0	0	0	4
1	0	1	0	0	0	1	1
1	0	0	0	1	0	1	2
1	0	1	0	1	0	1	1

TABLE 2 – Vision condensée des auteurs de l'équipe

Pour cet exercice, la colonne **count** est à prendre en compte comme un multiplicateur des occurrences de chaque ligne. Le support de l'itemset {MB, SJ} est donc de 3 (en support absolu), ou de 3/33 (en support relatif).

Q4.1 Lister les itemsets fréquents pour un support absolu minimum de 3, avec leur valeur de support.

Q4.2 En vous basant sur votre réponse à la question précédente, lister les itemsets fréquents clos pour un support absolu minimum de 3.

Q4.3 Calculer la confiance, la mesure de Kulczynski et l'*imbalance ratio* (IR) des règles d'association suivantes :

— MB → SJ
— AM → LB
— AH, BC → MB

Nous cherchons à constituer des groupes thématiques des chercheurs de l'équipe sur la base des données de la table 2. Pour cela, plusieurs approches valides semblent envisageables (dont celle ayant abouti au résultat de la figure 2 ci-dessous).

Q4.4 En détaillant votre démarche, donner une proposition de constitution d'un tel regroupement.

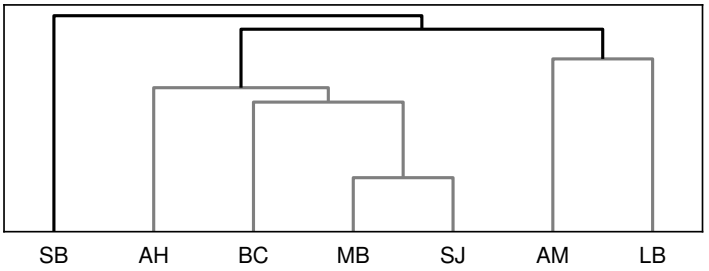

FIGURE 2 – Exemple de classification

1. Parmi les enseignants-chercheurs de l'équipe, c'est-à-dire que d'autres auteurs peuvent être présents mais ne sont pas référencés dans cette table.

5 Pipeline d'apprentissage automatique (5 points)

Nous considérons une liste, retranscrite dans la table 3, de villes françaises avec leur région d'appartenance, leur population (nombre d'habitants, et densité en hab./m^2) et le revenu médian par unité de consommation. Une ville est qualifiée comme *étudiante* si elle compte au moins 10 000 étudiants.

	ville	région	population	densité	revenu médian	étudiante
entraînement	Poitiers	NAQ	88 665	2 106	19 300	True
	La Rochelle	NAQ	76 114	2 677	–	True
	Cannes	PAC	73 965	–	20 340	False
	Nice	PAC	341 032	4 742	20 530	True
test	Toulon	PAC	176 198	4 113	–	True
	Annecy	ARA	128 199	–	24 870	False

TABLE 3 – Liste des villes labellisées

Nous cherchons à entraîner un modèle de classification automatique (binaire) pour déterminer la qualification des villes comme *étudiantes* ou non. Pour cela, le pipeline de traitement suivant a été élaboré.

```
Pipeline(steps=[
  ('column_preprocessing', ColumnTransformer(
    transformers=[
      ('categories', OneHotEncoder(handle_unknown='ignore'), ['région']),
      ('missing', SimpleImputer(strategy='median'), ['densité', 'revenu médian']),
      ('others' , 'passthrough', ['population']),
    ])),
  ('classifier', RandomForestClassifier())
])
```

Q5.1 Donner les valeurs des données transformées arrivant en entrée du `RandomForestClassifier` pour les six villes de la table 3 (jeu d'entraînement et jeu de test).

Après la phase entraînement et d'évaluation du modèle, nous cherchons à prédire la valeur cible pour l'entrée suivante, correspondant à la ville de Nîmes.

ville	région	population	densité	revenu médian
Nîmes	OCC	–	924	18 020

L'exécution de la fonction `predict` sur le pipeline donne malheureusement l'erreur suivante :

```
ValueError: Input contains NaN, infinity or a value too large for dtype('float32').
```

Q5.2 D'où cette erreur provient-elle et que suggérez-vous pour la corriger ?

Après correction, votre modèle ne donne toujours pas le résultat attendu (False au lieu de True, Nîmes étant bien une ville étudiante).

Q5.3 Quelle démarche mettriez-vous en œuvre pour expliquer l'origine de ce résultat erroné ?

Afin d'améliorer les performances du modèle, nous cherchons à déterminer les options de création qui en améliorerait l'exactitude parmi :

— le nombre d'arbres dans la forêt,
— la profondeur maximale des arbres,
— le nombre d'attributs en entrée de chaque arbre.

Q5.4 En détaillant votre approche, que proposez-vous pour réaliser ce type d'optimisation ?