
# Classifieur de Bayes et approximation softmax

MAM4 - Projet S7

2020/2021

Polytech Nice-Sophia

Référent : Mr Lionel Fillatre

Andrieu Grégoire

Gille Cyprien

Negre Philippe

Youssef Alan

# I. Introduction

Le but de ce projet est d'étudier la différence entre les fonctions **argmax** et **softargmax** dans le cadre d'un problème de **classification**. Explicitons ces trois termes.

La fonction **argmax** associe à un vecteur de réels un autre vecteur, de même dimension et ne comportant que des zéros, sauf à la position du maximum du vecteur d'entrée :

$$argmax : \begin{matrix} \mathbb{R}^n \to \{0,1\}^n \\ V = [v_i] \mapsto W = [w_i] \; avec \end{matrix} \begin{cases} w_i = & 1 \; si \; v_i = \max\limits_i(V) \\ & 0 \; sinon \end{cases}$$

La fonction softargmax, ou **softmax**, effectue une approximation différentiable d'argmax, et produit un vecteur de réels positifs compris entre 0 et 1 et de somme 1 (assimilables à un vecteur de probabilités) selon la formule ci-dessous:

$$softmax : \begin{matrix} \mathbb{R}^n \to [0,1]^n \\ V = [v_i] \mapsto W = [w_i] \; avec \; w_i = \dfrac{\exp(v_i)}{\sum_{k=1}^{n} \exp(v_k)} \end{matrix}$$

Un problème de classification consiste à ranger une donnée observée dans une classe parmi un certain nombre de classes choisies a priori.

Si l'on note $\mathcal{D}$ un classifieur, $X$ une donnée (aussi appelée observation) et $Y_k, k \in [1, K]$ les $K$ différentes classes, une classification se représente donc ainsi:

$$\mathcal{D}(X) = Y_i$$

Avec ces notations, on cherche donc le meilleur classifieur D*, celui qui minimise le risque d'erreur global R :

$$D^* = \underset{\mathcal{D} \in \mathbf{D}}{argmin} \; R(\mathcal{D})$$

Il se trouve que ce classifieur optimal est le classifieur de Bayes, et que c'est donc ce dernier que nous utiliserons pour étudier les différences entre les fonctions argmax et softmax.

### Le classifieur de Bayes

Le classifieur de Bayes est un classifieur très simple, qui repose sur la formule de Bayes et sur l'hypothèse d'indépendance des différentes features des données. La classification par un classifieur de Bayes s'effectue comme suit:

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{argmax}\ p(Y = k) \prod_{i=1}^{n} p(X_i \mid Y = k)$$

Il apparaît que le label prédit, y hat, dépend des probabilités conditionnelles et des probabilités a priori (i.e. des répartitions des données dans les différentes classes). C'est sur ce second paramètre que nous allons porter notre étude du risque global, sur lequel nous reviendrons dans la partie III.

Pour se familiariser avec le sujet, nous commencerons par l'étude d'un cas particulier, défini et implémenté par nos soins.

## II. Étude d'un cas particulier

### Mise en situation

Afin de saisir tous les aspects d'un classifieur de Bayes, nous l'avons implémenté dans le cas idéal où notre connaissance des distributions probabilistes des données était totale. Nous avons choisi d'ancrer nos données dans le réel en travaillant sur des pizzas, aux différentes qualités et caractéristiques.

Nous avons donc:

- Choisi comme classes/labels la qualité d'une pizza donnée, pouvant être 'Excellente', 'Ok' ou 'Honteuse'.
- Choisi quatre caractéristiques/features pour une pizza: son degré de cuisson, le nombre de pepperonis, la quantité de fromage ainsi que son rayon.
    - Le degré de cuisson est un entier compris entre 1 et 5.
    - Le nombre de pepperonis va de 0 à 25 avec un pas de 5.
    - La quantité de fromage peut être: 'Aucun', 'Un peu', 'Beaucoup', 'Une montagne', codés par des entiers allant de 1 à 4.

- Le rayon de la pizza , quant à lui, peut être de 3 dimensions: 10, 15 et 20 cm.

Une pizza est donc de dimension 4:

$$X \in \mathbb{R}^4 \text{ et } Y \in \{\text{Excellente}, \text{Ok}, \text{Honteuse}\}$$

Travaillant dans un cas idéal, nous nous sommes également donnés arbitrairement la répartition des pizzas dans les trois classes, et toutes les probabilités conditionnelles, dont nous donnons un exemple ci-dessous (le reste est en annexe).

## Données

- Proportions des classes

| P(Y) | 0 : Excellente | 1 : Ok | 2 : Honteuse |
|---|---|---|---|
| | 0.30 | 0.50 | 0.20 |

- Degré de cuisson

Disons que le degré de cuisson va de 1 : Pas cuite à 5 : En cendres. On peut donc supposer que les distributions de probabilité ressemblent à quelque chose comme ça:

| DdC selon acceptabilité | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Excellente | 0.0 | 0.1 | 0.8 | 0.1 | 0.0 |
| Ok | 0.0 | 0.3 | 0.4 | 0.3 | 0.0 |
| Honteuse | 0.4 | 0.1 | 0.0 | 0.1 | 0.4 |

Une fois ces probabilités conditionnelles entrées dans des arrays numpy, on implémente également la formule de Bayes, et on utilise les fonctions argmax et softmax de numpy et scipy respectivement.

Étant donné une pizza aléatoire, notre classifieur idéal donne des résultats cohérents. On obtient notamment qu'une pizza parfaite a une chance non négligeable d'être classée comme pizza Ok dans le cas softmax, et réciproquement.

### Exemple

Prenons une pizza non cuite, avec aucun pepperoni, pas de fromage et un rayon de 10 cm. Une telle pizza est donc représentée par le vecteur `[1, 0, 1, 10]` dans notre code.

Avec `pizza_bc` le classifieur de bayes basé sur les probabilités (conditionnelles et à priori) susmentionnées:

```
print("\nClassification d'une pizza la plus mauvaise possible")
mauvaise_pizza = [1, 0, 1, 10]
pizza_bc.classify(mauvaise_pizza)
pizza_bc.bayes_softmax(mauvaise_pizza)
```

Donne:

```
Classification d'une pizza la plus mauvaise possible
Classe: 2
Classe 0. Confiance: 0.2119
Classe 1. Confiance: 0.2119
Classe 2. Confiance: 0.5761
[Finished in 0.7s]
```

## III. Le risque d'erreur global

### Expression

Afin de pouvoir étudier le risque mathématiquement, nous en avons établi une expression. Les calculs et notations sont inspirés du cours de D. Richard Braun.

Le risque dépend du classifieur utilisé (argmax ou softmax) et de la répartition des données dans les classes. Le classifieur bayésien (avec argmax) reste noté $\mathcal{D}^*$, et on note le classifieur softmax $\widetilde{\mathcal{D}}$.

On note $\pi$ le vecteur contenant les probabilités a priori: $\pi_i = P(Y = i)$

L'expression du risque est alors:

$$R(\mathcal{D}, \pi) = \sum_{i \in \{1,\dots,K\}} \pi_i R_i(\mathcal{D})$$

avec $R_i$ les risques d'erreur conditionnels.

Il nous faut donc une expression des risques conditionnels. Pour cela, nous allons exprimer les prédictions faites par notre classifieur sous la forme d'une matrice, pour pouvoir manipuler ses coefficients dans la suite.

Notons $N$ le nombre d'observations (i.e. de données), $K$ restant le nombre de classes. Notre matrice de probabilités $A'$ est alors de dimension $(K, N)$:

$$A' = \begin{pmatrix} P(Y = 1 \mid X_1) & \dots & P(Y = 1 \mid X_N) \\ \vdots & \ddots & \vdots \\ P(Y = K \mid X_1) & \dots & P(Y = K \mid X_N) \end{pmatrix}$$

On applique le théorème de Bayes, et on obtient la matrice $A$ des probabilités postérieures, où apparaît la première dépendance en $\pi$.

$$A = \begin{pmatrix} P(X_1 \mid Y = 1)\pi_1 & \dots & P(X_N \mid Y = 1)\pi_1 \\ \vdots & \ddots & \vdots \\ P(X_1 \mid Y = K)\pi_k & \dots & P(X_N \mid Y = K)\pi_k \end{pmatrix}$$

Ainsi:

$$a_{ij} = \pi_i P(X_j \mid Y = i)$$
$$i \in \{1 \dots K\}, j \in \{1 \dots N\}$$

C'est à la prochaine étape du calcul qu'apparaît la différence entre softmax et argmax, nous commencerons donc par la matrice de décision obtenue après l'application d'argmax à la matrice des probabilités.

L'application d'argmax se fait selon les colonnes:

$$D_j^* = \underset{i}{argmax}[A_j]$$

Ce qui donne donc:

$$d_{ij}^* = \left( \underset{k}{argmax} P(X_j \mid Y = k)\pi_k \right)_i$$

Le même processus, en utilisant softmax, donne:

$$\tilde{D}_j = softmax[A_j]$$

Ce qui donne donc:

$$\tilde{d}_{ij} = \frac{\exp(\pi_i P(X_j \mid Y = i))}{\sum_{k=1}^{K} \exp(\pi_k P(X_j \mid Y = k))}$$

Maintenant que nous avons obtenu nos matrices de décision pour argmax et softmax, nous pouvons revenir à la définition d'un risque conditionnel. Le risque conditionnel étant le risque de classer une donnée incorrectement sachant sa classe donnée, il suffit de faire la somme des probabilités pour toutes les classes qui ne sont pas celle considérée actuellement, puis la somme de cette quantité pour toutes les données.

$$R_i(D) = \sum_j \sum_{k \neq i} P(Y = k | X_j)$$
$$= \sum_j C^i D$$

Avec $C^i$ un vecteur $(1, K)$ tel que :

$$C^i_j = \begin{cases} 1 \; si \; j \neq i \\ 0 \; sinon \end{cases}$$

Notes: la somme selon l'indice *j* indique une somme des colonnes (important pour le calcul du gradient plus loin). Chaque terme du vecteur $C^i$ est l'opposé de $\delta_{ij}$ (symbole de Kronecker).

Nous pouvons enfin injecter les expressions de D dans cette expression du risque conditionnel, et l'expression alors obtenue dans l'expression du risque global énoncée en début de cette partie.

Pour le risque argmax:

$$R(D^*, \pi) = \sum_{i \in \{1,\dots,K\}} \pi_i R_i(D^*)$$
$$= \sum_i \sum_j \pi_i C^i D^*$$

Note: il n'est pas très utile d'aller plus loin, car il n'est pas possible d'écrire une expression succincte d'un terme seul de la matrice de décision à cause de la fonction argmax, qui se repose toujours sur toute une colonne.

Pour le risque softmax:

$$
\begin{aligned}
R(\tilde{D}, \pi) &= \sum_{i \in \{1,\dots,K\}} \pi_i R_i(\tilde{D}) \\
&= \sum_i \sum_j \pi_i C^i \tilde{D} \\
&= \sum_i \sum_j \pi_i C^i_j \tilde{d}_{ij} \\
&= \sum_{i=1}^{K} \sum_{j=1}^{N} \pi_i (1 - \delta_{ij}) \frac{\exp(\pi_i P(X_j \mid Y = i))}{\sum_{k=1}^{K} \exp(\pi_k P(X_j \mid Y = k))}
\end{aligned}
$$

## Visualisations graphiques

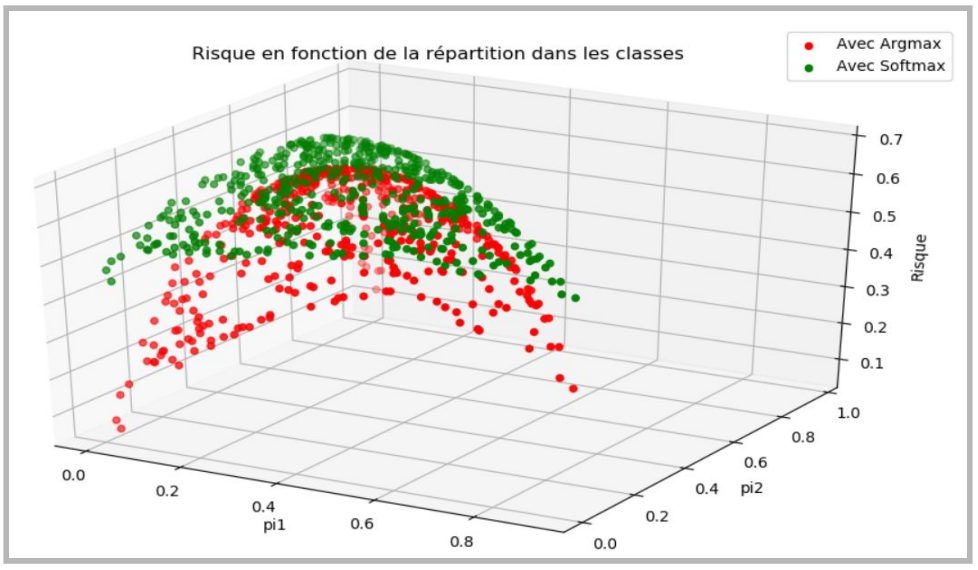
Nous avons généralisé le code de notre exemple (partie I.) pour pouvoir rapidement créer des classifieurs de bayes idéaux (i.e. avec une connaissance parfaite des probabilités conditionnelles) en attribuant de manière aléatoire toutes les probabilités (il reste bien sûr possible d'attribuer les probabilités manuellement). Cette nouvelle version du code permet également d'expérimenter avec un grand nombre de classes ou des données vivant en grande dimension.

L'intérêt principal d'une telle généralisation est la possibilité de représenter sur un graphe les deux risques en fonction des probabilités à priori.

Notons que si l'on a K classes, il suffit de déterminer P(Y=k) $\forall k \varepsilon [|1,\ K-1|]$ pour pouvoir calculer le risque (puisque la dernière probabilité est juste $1 - \sum_{k=1}^{K-1} P(Y = k)$). Ainsi, nous pouvons représenter les risques pour k=2 et k=3.

**Pour k = 3:**

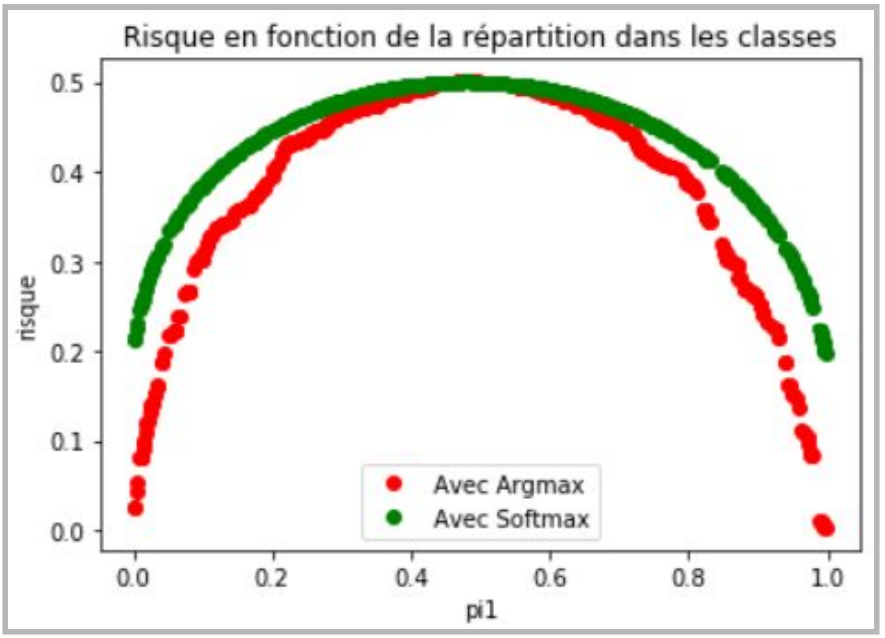
On représente en 3 dimensions le risque (en utilisant argmax (vert) et softmax (rouge)) sur l'axe z, en fonction des proportions de répartition dans 2 classes, respectivement sur les axes x et y.




On peut remarquer que les deux risques semblent concaves, et que leur maximums semblent être au niveau de l'égalité des probabilités a priori.

**Pour k = 2 :**

On représente en 2 dimensions le risque en utilisant le même jeu de couleurs que pour k=3. On peut alors tirer les mêmes observations que pour k=3.



## Propriété de concavité :

Une des hypothèses que l'on peut émettre après les représentations graphiques est la concavité du risque. Si l'on arrive à la démontrer, on pourra l'exploiter pour assurer la convergence d'une méthode de recherche de maximum à base de gradient, comme celle que l'on utilisera plus loin.

- Cas softmax :

  D'après *Reverdy et al.*, la fonction softmax est concave sur le domaine qui nous intéresse (celui des réels positifs entre 0 et 1). Ainsi, d'après l'expression du risque d'erreur global, ce dernier est également concave en tant que combinaison linéaire à coefficients positifs de fonctions concaves.

- Cas argmax :

Posons $V(\pi) = R(D^*, \pi)$ le risque d'erreur global pour des probabilités a priori $\pi$.

Soit $\alpha \in [0, 1]$, soient $\pi$ et $\pi'$ deux probabilités à priori, et $\pi''$ une troisième telle que $\pi'' = \alpha\pi + (1 - \alpha)\pi'$

On a alors, en vertu de la minimisation du risque par le classifieur de Bayes argmax:

$$\begin{aligned} V(\pi'') &= \alpha\pi^T R(D^*, \pi'') + (1 - \alpha)\pi'^T R(D^*, \pi'') \\ &\geq \alpha V(\pi) + (1 - \alpha)V(\pi') \end{aligned}$$

Par la définition d'une fonction concave, le risque avec argmax est bien concave. Notons qu'il n'est cependant pas différentiable.

# IV. Maximisation du risque par la méthode du gradient projeté

Pour calculer le gradient du risque d'erreur global, il va falloir (comme on peut le constater dans la formule trouvée à la partie précédente) notamment dériver la fonction softmax. Nous nous appuierons pour cela sur les travaux déjà effectués par Eli Bendersky, qui nous fournissent les dérivées partielles de la fonction softmax.

Si l'on note :

$$S_i = \frac{\exp(a_i)}{\sum_{k=1}^{K} \exp(a_k)}$$

Alors on a:

$$\frac{\partial S_i}{\partial a_j} = \begin{cases} S_i(1 - S_j) \; si \; i = j \\ -S_j S_i \; si \; i \neq j \end{cases}$$

Notons que l'on peut bien se servir de cette notation, car l'on veut dériver par rapport à $\pi_j$ et ce dernier n'a pour facteur que la probabilité conditionnelle, qui est constante. Notons également que notre $S_i$ est un vecteur ligne de longueur le nombre de données, et que les produits entre S se font terme à terme, puisque l'on somme leurs termes avec la somme d'indice *j* (comme on l'a fait précédemment).

On a donc:

$$\begin{aligned}
\frac{\partial R(\tilde{D}, \pi)}{\partial \pi_i} &= \frac{\partial}{\partial \pi_i} \sum_{i=1}^{K} \pi_i \sum_{j=1}^{N} (1 - \delta_{ij}) \frac{\exp(\pi_i P(X_j \mid Y = i))}{\sum_{k=1}^{K} \exp(\pi_k P(X_j \mid Y = k))} \\
&= \frac{\partial}{\partial \pi_i} \sum_{i=1}^{K} \pi_i \sum_{j=1}^{N} (1 - \delta_{ij}) S_i \\
&= \sum_{j=1}^{N} (1 - \delta_{ij}) S_i + \pi_i \sum_{j=1}^{N} (1 - \delta_{ij}) \frac{\partial S_i}{\partial \pi_i} + \sum_{k \neq i}^{K} \sum_{j=1}^{N} (1 - \delta_{kj}) \frac{\partial S_k}{\partial \pi_i} \\
&= \sum_{j=1}^{N} (1 - \delta_{ij}) S_i + \pi_i \sum_{j=1}^{N} (1 - \delta_{ij}) S_i (1 - S_i) + \sum_{k \neq i}^{K} \sum_{j=1}^{N} (1 - \delta_{kj})(-S_k S_i) \\
&= \sum_{j=1}^{N} \left[ (1 - \delta_{ij}) S_i + \pi_i (1 - \delta_{ij}) S_i (1 - S_i) + \sum_{k \neq i}^{K} (1 - \delta_{kj})(-S_k S_i) \right] \\
\frac{\partial R(\tilde{D}, \pi)}{\partial \pi_i} &= \sum_{j=1}^{N} \left[ (1 - \delta_{ij}) \left( S_i + \pi_i S_i (1 - S_i) \right) + \sum_{k \neq i}^{K} (1 - \delta_{kj})(-S_k S_i) \right]
\end{aligned}$$

Notre but à présent est de trouver le vecteur des probabilités à priori qui maximise le risque d'erreur global softmax.

Très souvent en machine learning, on utilise la méthode du gradient pour minimiser une fonction convexe. On peut faire de même pour maximiser le risque, que l'on a démontré concave.

**Principe de la descente du gradient** :

Nous nous donnons au départ un nombre total d'itérations N à faire, un vecteur de départ $\pi_0$ arbitraire (qui satisfait bien les propriétés d'un vecteur de probabilités) et un coefficient $\eta$ lui aussi arbitraire, qui correspond à la 'learning rate', autrement dit le coefficient multiplicatif placé avant le gradient. Notons que cette learning rate doit être ajustée selon le problème, et que la convergence (et la vitesse de convergence) dépend de sa valeur.

Ensuite pour chaque itération, on doit effectuer ces étapes :

- calculer le gradient appliqué au vecteur $\pi_{k-1}$
- faire une première mise à jour de ce vecteur en lui faisant une transformation telle que : $\pi_k=\pi_{k-1}+\eta * \nabla R(\check{D}, \pi)(\pi_{k-1})$

Ce calcul va nous donner un nouveau vecteur $\pi_k$, mais rien ne garantit que ce dernier respecte les propriétés probabilistes :

- $\sum_i \pi_i=1$
- $\forall$ i, $\pi_i \geq 0$

Pour remédier à cela, on va donc utiliser une méthode de projection sur le simplex, qui nous assurera le respect des 2 propriétés précédentes, et qui donne son nom à l'algorithme du gradient 'projeté'.

La projection sur le simplex se fait ainsi:

1) stocker dans une première variable ( disons z ) un vecteur étant la somme cumulée de ses éléments de manière décroissante, auquel on aura pour chaque terme soustrait la valeur 1 ( car on veut projeter sur [0,1] )
2) stocker dans une seconde variable ( disons a ) un vecteur contenant les nombres de 1 à N+1 , N étant le nombre d'élément du vecteur d'entrée
3) stocker dans une troisième variable ( disons maxi ) le maximum du vecteur de la division terme à terme de z par a
4) enfin, stocker dans le vecteur final ( celui qui sera retourner par la fonction ) le maximum pour chaque élément entre le vecteur d'entrée moins maxi et 0

Et enfin pour finir l'algorithme du gradient projeté, on réalise cette dernière étape :

- utiliser la fonction de la projection sur un simplex sur $\pi_k$

On réitère ainsi ces étapes N fois.

Après implémentation dans python de tout cela, (gradient du risque, algorithme du gradient projeté) l'algorithme converge bien vers le π qui maximise le risque:

```
Etape: 0 | [0.1, 0.2, 0.7]
Etape: 20 | [0.24258121 0.27990927 0.47750952]
Etape: 40 | [0.29918982 0.31194929 0.38886089]
Etape: 60 | [0.32166766 0.32480849 0.35352385]
Etape: 80 | [0.33059316 0.32997157 0.33943526]
Etape: 100 | [0.33413718 0.33204496 0.33381786]
[Finished in 0.7s]
```

On voit que c'est l'équirépartition des données dans les différentes classes qui maximise le risque d'erreur global.

# V. Bibliographie

Classification bayésienne et risque, expressions et calculs:
ECE531 Lecture 2a: A Mathematical Model for Hypothesis Testing (wpi.edu)
ECE531 Lecture 2b: Bayesian Hypothesis Testing (wpi.edu)
ECE531 Lecture 3: Minimax Hypothesis Testing (wpi.edu)
Projection sur le simplex:
http://www.gipsa-lab.fr/~laurent.condat/download/proj_simplex_l1ball.m
Concavité de la fonction softmax:
https://arxiv.org/pdf/1502.04635.pdf

# VI. Annexe

Données de l'exemple des pizzas (suite):

- Nombre de pepperonis

| NbPp selon acceptabilité | 0 | 5 | 10 | 15 | 20 | 25 |
|---|---|---|---|---|---|---|
| Excellente | 0.0 | 0.6 | 0.4 | 0.0 | 0.0 | 0.0 |
| Ok | 0.2 | 0.2 | 0.2 | 0.2 | 0.1 | 0.1 |
| Honteuse | 0.3 | 0.0 | 0.0 | 0.0 | 0.1 | 0.6 |

- Quantité de fromage

| QdFr selon acceptabilité | Aucun | Un peu | Beaucoup | Une montagne |
|---|---|---|---|---|
| Excellente | 0.0 | 0.0 | 0.5 | 0.5 |
| Ok | 0.0 | 0.45 | 0.45 | 0.1 |
| Honteuse | 0.6 | 0.4 | 0.0 | 0.0 |

On codera ces 4 niveaux par 1 2 3 et 4.

- Rayon de la pizza

| Rayon selon acceptabilité | 10 | 15 | 20 |
|---|---|---|---|
| Excellente | 0.1 | 0.4 | 0.5 |
| Ok | 0.3 | 0.4 | 0.3 |
| Honteuse | 0.89 | 0.1 | 0.01 |