

Mini-projet Choixpeau magique



I. Le choixpeau magique dans le roman Harry Potter



Le Choixpeau magique (Sorting Hat) est un artefact doué d'une âme qui détermine de façon magique dans laquelle des quatre maisons de Poudlard chaque nouvel élève doit être envoyé. Ces quatre maisons ont pour nom : Gryffondor (Gryffindor), Poufsouffle (Hufflepuff), Serdaigle (Ravenclaw) et Serpentard (Slytherin).

Le Choixpeau est investi de l'intelligence des fondateurs et a le don de Legilimancie, lui permettant de lire dans les pensées de celui qui le met sur sa tête et de ressentir ses aptitudes ou ses humeurs et peut également répondre aux pensées de celui qui le porte.

Il est utilisé à chaque rentrée le 1^{er} septembre lors de la cérémonie de répartition des élèves de première année, et ce depuis plusieurs siècles. Le Choixpeau choisit la maison adéquate pour chaque élève en lisant dans ses pensées, et déclare à haute voix sa décision finale prise en fonction des vertus, des projets et des centres d'intérêts de celui-ci.

https://harrypotter.fandom.com/fr/wiki/Choixpeau_magique

II. Le choixpeau magique en NSI

L'objectif de ce mini-projet est de **modéliser un choixpeau magique numérique**. Votre création numérique devra donc être en mesure de déterminer la maison la mieux adaptée à un profil d'« élève cible ».

Vous devrez donc :

- **Construire un modèle, basé sur l'algorithme des k plus proches voisins**, capable d'attribuer une maison lorsqu'on lui fournit un profil d'élève cible et des profils d'élèves en référence.
- **Construire une procédure permettant de créer ce profil d'élève cible**, en élaborant un QCM pertinent (2^{nde} partie).
- **Construire une interface graphique ergonomique**, permettant la saisie du QCM (2^{nde} partie), puis l'affichage du verdict du choixpeau magique.

III. Le choixpeau magique : 1^{ère} partie

1. Objectif

La première partie consiste à concevoir le cœur du programme, celle qui utilisera l'algorithme des kPPV en s'appuyant sur les élèves références (Caracteristiques_des_persos.csv) et sur un profil d'élève cible prédéfini.

Le programme devra renvoyer les k plus proches voisins de plusieurs profils d'élèves donnés ci-dessous, puis leurs maisons attitrées.

2. Cahier des charges

Votre programme doit être conçu pour faciliter son test et son évaluation. Il doit donc être complet, exécutable immédiatement avec un interpréteur Python.

L'utilisation du tri built-in de Python `sort()` est autorisée.

Il est à nouveau conseillé de construire une fonction par objectif et de structurer au mieux votre programme (Encodage, commentaire introductif, variables globales, définitions des fonctions, appel des fonctions dans un corps de programme structuré).

Vos fonctions doivent être parfaitement documentées dans leur docstring.

Le choixpeau magique devra se prononcer sur les profils suivants :

- Courage : 9 Ambition : 2 Intelligence : 8 Good : 9
- Courage : 9 Ambition : 4 Intelligence : 8 Good : 9
- Courage : 3 Ambition : 8 Intelligence : 6 Good : 3
- Courage : 2 Ambition : 3 Intelligence : 7 Good : 8

Les données doivent résulter d'une fusion entre les fichiers Characters.csv et Caracteristiques_des_persos.csv.

Il n'est pas autorisé de modifier les caractéristiques des personnages.

Les données étant en anglais, il est préférable d'utiliser des noms de fonctions et de variables en langue anglaise.

La distance par défaut sera la distance euclidienne.

La valeur de k sera fixée à 5 par défaut.

Votre IHM doit nécessairement automatiser les passages de chaque profil devant le choixpeau magique, avec une visualisation claire des résultats obtenus (la maison choisie + les plus proches voisins et leurs maisons respectives).

L'IHM ne doit pas nécessairement passer par une interface graphique de type Tkinter, Pygame ou page web, même si cette option est envisageable (bonus). Considérant que ce sera l'objet du cahier des charges obligatoire de la seconde partie, il paraît inutile d'y dépenser trop d'énergie dans un premier temps.

Un algorithme doit être rédigé en pseudo-code et ne doit décrire que le cœur de la problématique (l'algorithme kPPV), pas les créations de structures de données ni l'IHM.

Au besoin, votre algorithme pourra utiliser des appels de fonctions...

- dont il est inutile de détailler le pseudo-code.
- avec un nom suffisamment explicite pour comprendre leur utilité.
- avec un descriptif sous le mot clé **FONCTIONS** au dessus du détail des **VARIABLES**.
- avec un équivalent de leur docstring dans ce descriptif

Enregistrer le programme sous le format : kPPV_1_NOM1_NOM2_NOM3.py

Si vous avez un document PDF (ex : algorithme, lien vers vos outils collaboratifs, aide à l'utilisation de votre programme,...) : kPPV_1_NOM1_NOM2_NOM3.pdf

Créer un fichier compressé (kPPV_1_NOM1_NOM2_NOM3.zip) pour organiser vos fichiers. Après décompression, la structure de dossiers / fichiers doit permettre une exécution facile du programme (pas de changement de nom de fichier ou de dossier à effectuer).

Votre projet doit être déposé sur l'espace de travail NSI d'e-lyco (« Vue d'ensemble », « tâches »), en temps voulu. Vous devrez y créer votre groupe avant de déposer vos fichiers.

Remarque : les notebooks 4_11, 4_12, 4_13, P1, P2 et P4 pourront vous servir.

3. Bonus possibles

- La saisie manuelle d'un profil d'élève à tester par le choixpeau peut être ajoutée.
- La valeur de k doit pouvoir être modifiée dans l'IHM.
- Une validation croisée est effectuée, avec affichage clair des résultats en pourcentage.
- Une optimisation des résultats est recherchée (à l'aide de la validation croisée, avec un travail sur les données, sur la valeur de k, ...).

4. Évaluation

Critères d'évaluation (note sur 10 points) :

- Respect du cahier des charges (5 points)
- Algorithme (2 points)
- Bonnes pratiques PEP-8 (dont commentaires et docstrings) (1 point)
- Travail d'équipe / Utilisation d'outils collaboratifs numériques (1 point)
- Point « subjectif » (1 point)
- Bonus (+ 2 points envisageables en fonction de l'ampleur du bonus)