PROJET 1 PYTHON
AMAZON PRIME VIDEO, DISNEY +, NETFLIX ?

MASTER 1 ECONOMETRIE STATISTIQUES
PARIS 1 PANTHEON-SORBONNE

23 NOVEMBRE 2021

Anisoara ABABII, Armand L'HUILLIER, Yanis REHOUNE, Camil ZAHI
Sous la direction de Imane LOUKAH

Sommaire

Introduction

Le projet ci-présent a pour objectif d'aider les utilisateurs à choisir quelle serait la plateforme de streaming la plus adéquate en fonction de leurs gouts.

Pour cela, nous avons pu accéder à trois bases de données synthétisant le contenu des catalogues de Amazon Prime video, Disney + et Netflix, grâce aux liens suivants :

Amazon Prime Movies and TV Shows | Kaggle

Disney+ Movies and TV Shows | Kaggle

Netflix Movies and TV Shows | Kaggle

Nous nous sommes également procurés une quatrième base de données qui permet d'avoir pour les films présents sur l'Internet Movie Data, la note donnée en moyenne par les utilisateurs du site aux films, que vous retrouverez sur le lien suivant :

IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows

Pour ce faire, notre projet s'est découpé en 3 grandes parties qui font l'objet de la structure de ce rapport. A travers chaque partie, vous aurez l'occasion de suivre notre démarche qui vous conduiront aux résultats conclus et interprétés. Cela étant certaines difficultés ont eu lieu à travers ce mémoire mais nous avons pris soin de mentionner ces dernières de manière à tenir compte de ces problèmes d'arbitrages.

Nous espérons que ce projet vous plaira ! Bonne lecture !

Instructions pour exécuter votre programme

Le dossier « sujet_1_ABABII_L_HUILLIER_REHOUNE_ZAHI » est composé de :

- Un dossier pour la PARTIE 1 : « 1. ANALYSE » ;
- Un dossier pour les PARTIES 2 ET 3 : « 2. MOTEUR DE RECHERCHE ET INTERFACE » ;

  Dans chacun de ces dossiers, vous retrouverez :

  - Les données (cela permettra d'éviter de changer de chemin) ;
    - « amazon_prime_titles »,
    - « disney_plus_titles »,
    - « netflix_titles »,
    - « movies_ratings_IMDB ».
  - Le/Les notebook (là où se trouveront les résultats) ;
  - Les modules appelés dans les notebooks correspondants.
- Le fichier PDF ci-présent : « RAPPORT »,
- Le fichier PDF : « SUJET ».

Il vous faudra télécharger l'ensemble du dossier de base : « sujet_1_ABABII_L_HUILLIER_REHOUNE_ZAHI ».

Une fois cela fait, nous vous conseillons d'ouvrir votre interface jupyter pour passer à l'analyse du sujet.

1. Avant de commencer, il est important d'aller voir le type de données auquel nous avons à faire. Vous pouvez ainsi ouvrir les documents csv sous excel par exemple. Vous les retrouver dans le dossier « 1. ANALYSE ».

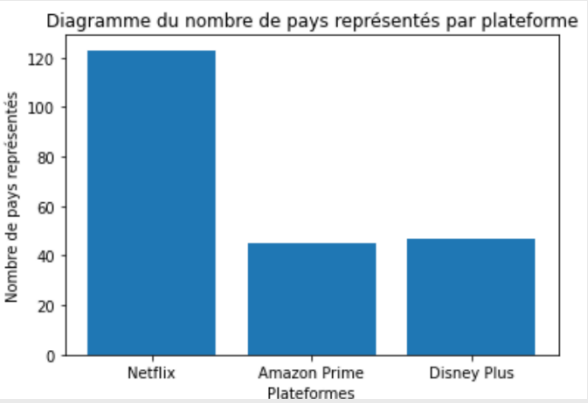
2. Nous pouvons désormais commencer. Accédez au dossier « 1. ANALYSE » et ouvrez le notebook « 1. ANALYSE DESCRIPTIVE DES BASES ET VISUALISATION ». Ce notebook ne comprend que très peu de codes et se contente en grande partie d'appeler un certain nombre de modules. Ces modules contiennent déjà les inputs des fonctions, il ne vous suffira donc d'exécuter les cellules du notebook. Vous trouverez la partie programmation en complément grâce aux modules :
   « DIVERSITE_GEOGRAPHIQUE », « EVOLUTION_OFFRE_DANS_LE_TEMPS »,
   « GENRES_LES_PLUS_REPRESENTES », « NOTE_MOYENNE_DES_FILMS ».

3. Une fois l'analyse réalisée, il vous faudra accéder au dossier « 2. MOTEUR DE RECHERCHE ET INTERFACE » où vous trouverez le notebook « 2. MOTEUR DE RECHERCHE » qui vous donnera la topologie du travail. Ce notebook est simplement là pour reprendre les questions du sujet tout en permettant une certaine organisation au sein du dossier. Une fois cela fait, vous pourrez accéder au notebook « 3. INTERFACE » qui affichera en fonction de vos préférences, du contenu associé. Ce notebook fait notamment appel aux modules ci-dessous réalisés à travers la partie 2 :
   « RECHERCHE_A_PARTIR_DE_TITRE », « RECHERCHE_A_PARTIR_DU_NOM_ACTEUR »,
   « RECHERCHE_A_PARTIR_DU_TYPE_GENRE », « SUGGESTION_SELON_PREFERENCES ».

Analyse descriptive des bases et visualisation

Diversité géographique

Concernant la diversité géographique, on a pu observer que le catalogue de Netflix était de loin celui qui contient le plus de titre issu de pays différents. Les catalogues de Disney+ et Amazon Prime video ont un niveau de diversité géographique plutôt similaire (voir diagramme ci-dessous).



On pourrait être amené à se dire que ces résultats traduisent la place de leader que Netflix possède aujourd'hui dans le secteur du streaming. Les deux autres plateformes étant plus récentes, on pourrait se dire qu'elles n'ont probablement pas encore pu développer leur offre aussi largement que Netflix.

Cette analyse a tout de même ses limites car si on observe le nombre de donnée manquante dans chaque dataframe (à l'aide de la fonction df.info()), on s'aperçoit qu'il y a seulement 672 données non-nulles dans la colonne Country dans le dataframe de Amazon et 1193 dans celui de Disney. C'est d'autant plus préoccupant que pour la même colonne, le dataframe de Netflix a 7976 données non-nulles. Ainsi, l'analyse est biaisée car la diversité géographique des plateformes Amazon et Disney est très fortement sous-estimée.

Dans cette partie, notre démarche a été de créé un dictionnaire contenant chaque pays et le nombre d'occurrences associé pour ensuite compter le nombre de clés (pays). Le principal problème a été de trouver comment isoler chaque pays individuellement de tel sorte à ce que chaque élément de la liste de la colonne country (du dataframe concaténé) soit un pays. On a pu résoudre ce problème en transformant la liste de la colonne country en chaine de caractère à l'aide de la fonction "join". Ensuite, on a remis la chaine de caratère en liste avec la fonction strip() en précisant le séparateur entre chaque élément (" , "). Ainsi, la fonction affecte bien à chaque pays son nombre d'occurences puis compte le nombre de clés de dictionnaire (équivalent aux pays).

Evolution de l'offre

Graphique 1 : Pour commencer, nous gardons les dataframes séparés des plateformes et nous nous intéressons en particulier à la colonne date_added. En effet, cette dernière nous renseigne sur la date à laquelle le film a été ajouté sur la plateforme.
Cependant, le format de la colonne n'était pas approprié pour apporter une étude de l'évolution, nous devions séparer en deux cette colonne, à l'aide de la commande datarfame['colonne'].str.split. De cette manière, nous nous retrouvions avec deux colonnes, une qui garde les années, et une autre avec le jour et le mois de la date. Nous utilisons seulement la première et donc nous regroupons tous les

films qui ont été ajoutés la même année sur la plateforme de streaming. Ainsi, on perd de l'information avec cette méthode car pour représenter l'évolution du nombre de films, on ne retient que l'année, alors qu'une date précise nous a été donnée (exemple : 21 september, 2019). Finalement, ces premiers graphiques montrent qu'Amazon Prime n'a presque aucune donnée dans la colonne d'intérêt, donc nous ne pouvons pas retracer l'évolution des films/séries avec ce dataframe. En revanche, pour Netflix et Disney+, nous pouvons continuer l'analyse à travers les prochains graphiques.

Graphique 2 : Nous cherchions à donner l'évolution en 'flux', c'est-à-dire l'évolution du nombre brut de films ajoutés chaque année. Pour ce faire, nous ajoutons le code dataframe.sort_values(by=['colonne']) pour classer le dataframe selon les années. On obtient un graphique qui donne une évolution chronologique (2019 à 2021 pour Disney+ et 2011 à 2021 pour Netflix). De plus, nous supprimons les quelques données manquantes des dataframes, ce qui permet de retirer une barre 'nan' dans les graphiques. Finalement, nous retirons les années pendant lesquelles moins de 5 films/séries ont été ajoutés pour donner plus de clarté au graphique. Nous apprenons donc avec le graphique 2 que Disney+ met de moins en moins de films/séries sur sa plateforme (évolution marginale décroissante de 2019 à 2021). D'autre part, Netflix a ajouté du contenu sur sa plateforme de manière exponentielle de 2008 à 2019. Mais depuis 2019, Netflix met en ligne de moins en moins de films/séries aussi.

Graphique 3 : Nous voulons passer d'une évolution marginale à une évolution brute dans ces graphiques. Pour donner l'évolution en 'stock', dans un premier temps nous voulions créer un graphique ECDF avec une fonction seaborn en utilisant le dataframe créé. Cependant les résultats étaient visiblement faux. N'ayant pas réussi à utiliser correctement cette fonction, nous avons opté pour une autre méthode : Pour chaque année (i) nous rajoutons le nombre de films déjà présent sur la plateforme en année (i-1) en plus du nombre de film stricto sinsu de l'année (i). Ce qui peut nous donner un nouveau graphique. On obtient que malgré l'évolution marginale décroissante depuis 2 ans, le contenu proposé par les deux plateformes ne cesse d'augmenter. Cet effet est davantage marqué sur pour la plateforme Netflix.

Graphique 4 - Analyse supplémentaire : Amoureux de la culture de la seconde moitié du 20ème siècle, nous avons voulu nous intéresser aux films et aux séries des plateformes en fonction de leur date de sortie, afin de voir s'il y a beaucoup de contenu qui appartient à nos décennies préférées. Grâce à la colonne 'release_year', nous avons pu créer une fonction qui donne le nombre de films sortis pendant une certaine période. Les périodes sont données dans une liste au paramètre 'bins' de la fonction Dataframe['colonne'].value_counts(bins=Ascending).plot. Les périodes dans la liste Ascending dépendent de la date de lancement de la plateforme (donnée en input dans la fonction) et sont calculées dans une boucle. Nous obtenons qu'une écrasante majorité des films/séries pour chaque plateforme a été produite dans les années 2000 et 2010, peu importe la date de lancement de la plateforme, pourtant prise en compte.

Graphique 5 : Pour terminer, nous avons dû analyser l'évolution du contenu en séparant pour les plateformes Netflix et Disney+, les films et les séries. Nous avons choisi de séparer chaque dataframe en deux selon cet unique critère. Nous voulions mettre sur un même graphique les deux résultats (ce qui aurait donné un seul graphique par plateforme). Cependant, la longueur des dataframes créés n'étant pas les mêmes pour les années (aucune série a été ajouté entre 2008 et 2013 sur Netflix alors

que des films oui), il était difficile de créer un unique graphique pour les films et les séries. Donc dans le code, nous disons que si le dataframe 'Série' où l'on compte les occurrences des années est de même longueur que le dataframe 'Film', alors on peut mettre les deux évolutions sur le même graphique. Sinon nous collons les deux graphiques côte à côte avec « fig, axes = plt.subplots() »

Genres de films/séries les plus représentés

Pour cette question, nous avons converti la colonne « listed_in » dans une liste, car il y a des films qui ont plusieurs genres dans une meme cellule. Pour pouvoir séparer les éléments de la liste et pour compter les occurrences de chaque genre différent, nous avons converti la liste dans un string est supprimé tous les espaces entre les mots en ayant comme séparateur la virgule. La fonction « split » nous a permis de diviser la chaîne de caractères dans une liste où chaque mot représente un élément de la liste. Ensuite, pour faire apparaitre l'incrémentation pour chaque genre une seule fois dans la liste vide (déclarée plus haut) on s'est servi de la fonction « append ». Grace au stockage de toutes les occurrences dans une liste appelée list_occurence, nous avons réussi à faire un dictionnaire à partir de deux listes : list_occurence = les genres des films, et, list_0= nrb de répétition. Finalement, pour avoir en sortie les genres les plus représentés de chaque plateforme nous avons trié le dictionnaire par ordre décroissant et stocké ensuite dans un nouveau dataframe.

D'après l'output de notre script nous avons obtenu les 5 genres suivants :

```
('Dans le catalogue netflix_titles.csv, les genres de films/séries les plus représentés sont:  ',
0
InternationalMovies   2752
Dramas                2427
Comedies              1674
InternationalTVShows  1351
Documentaries          869)
('Dans le catalogue disney_plus_titles.csv, les genres de films/séries les plus représentés sont:
0
Family            602
Animation         516
Comedy            497
Action-Adventure  438
ComingofAge       199)
('Dans le catalogue amazon_prime_titles.csv, les genres de films/séries les plus représentés sont:
0
Drama     3687
Comedy    2099
Action    1657
Suspense  1501
Kids      1085)
```

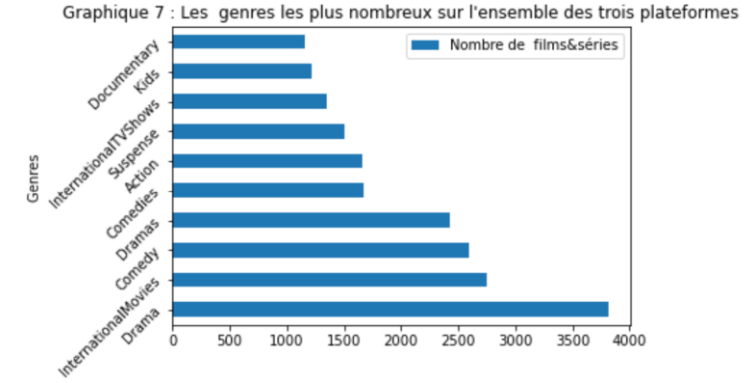
Les graphiques 2, 3, 4, permettent de visualiser les genres de films les plus représentés sur chaque plateforme de manière individuelle. Par conséquent, la bibliothèque d'Amazon Prime video contient plus de « Drama » que Netflix (davantage concentré sur les International Movies) ou Disney + (plus centré sur films/séries Family).

Les 6 derniers graphiques nous permettent d'analyser en détail chaque plateforme. Ainsi, nous pouvons voir les genres de films et de séries les plus représentés mais de manière séparée.

Par exemple, pour la question 1.3 (Partie 1) le plus grand problème c'était que nous avons eu à parcourir une colonne de DataFrame, mais qui ne contenait pas qu'un seul mot, mais plusieurs (c'était la colonne listed_in : Genres). Pour surmonter cette problématique nous avons vu que dans la df tous les genres ont eu comme séparateur la « , » - indicateur qui nous a servi de diviser le string dans une liste. En conclusion, la meilleure solution c'était de coller tous les genres qui contenaient 2 ou plusieurs mots dans un seul.

Une autre barrière c'était le manque de connaissance de toutes les fonctions que nous pouvons utiliser en travaillant sur Python (ex : zip, sorted, ect).

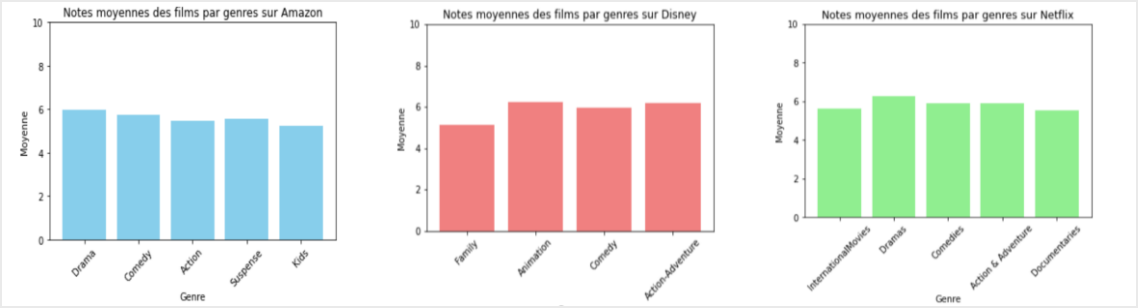
Les graphiques nous ont aussi possé un peu de difficulté, parce qu'au début nous avons fait des graphiques en mettant directement à la main les données obtenues à l'exécution du script. Donc, nous avons pu combattre cela de manière collective. Mais encore ce n'est pas encore optimal, c'est à améliorer le graphique global, parce que comme dans une df il avait Comédies comme genres et dans une autre Comedy, donc en sortie nous obtenions un graphique dans lequel les noms de gères se répètent. Le graphique global posé un doute car sur les trois plateforme, Netflix, Amazon, Disney, il n'y avait pas les mêmes genres de films les plus représentés.



Notes moyennes des films

Ici, on cherche à associer quand c'est possible le film à sa note correspondante entre 2 dataframes. Nous avons donc pris soin d'associer les films à leurs notes en fonction du titre en faisant l'intersection des titres entre les deux dataframes. Nous avons ensuite fait en sorte de diviser les lignes à chaque fois qu'un film avait plusieurs genres pour que l'on puisse connaitre avec précision les moyennes par genre. Enfin, on réalise un graphique en reprenant les moyennes trouvées (inclus dans le module) pour les cinq genres trouvés en question 1.3..

Voici les graphiques qui ressortent :



Pour les 5 genres les plus représentés sur chacune des plateformes de streaming, la moyenne des notes données aux films et quasiment toujours la même puisqu'elle tourne entre 5 et 6. Pas de différences grandement significatives. On peut néanmoins en dire que les films de genre « Drama » plaisent davantage au grand public chez Amazon Prime video et Netflix (et non chez Disney qui est davantage tourné vers du contenu familial et donc non associé au genre drama ».

Moteur de recherche

Recherche à partir d'un titre

Concernant la recherche selon le titre de film, notre démarche était de créé une liste contenant les titres du dataframe concaténé et une liste avec les films de la base IMDb afin de pouvoir afficher la note du titre si l'utilisateur entre un film présent dans le dataframe concaténé ainsi que dans IMDb. Nous avons ensuite fait en sorte de traiter tous les cas possibles selon ce que l'utilisateur entre dans l'interface. A ce stade, notre fonction était bonne mais il fallait entrer le titre avec les majuscules placés au bon endroit.

Nous voulions que le moteur de recherche fonctionne peu importe que l'utilisateur entre le titre avec des minuscules ou pas. Il a donc fallu que l'on créé une autre fonction qui permettrai de transformer l'input en minuscule pour ensuite aller chercher dans la liste des titres du dataframe concaténé que l'on a au préalable mis en minuscule pour faciliter la recherche. Ensuite nous avons utilisé l'output de cette fonction (le titre en minuscule) dans la fonction initiale afin que notre moteur de recherche fonctionne même si l'utilisateur entre « spider-man » dans l'interface.

Recherche à partir du nom d'un acteur

Pour cette question il a fallu construire un moteur de recherche, de telle façon que si l'utilisateur rentre un nom d'acteur, le programme est sensé lui retourner du contenu pour chaque plateforme dans lequel l'acteur est présent. Ensuite nous proposons à l'utilisateur la plateforme qui est la plus pertinente selon la quantité de séries/films associée à l'acteur.

Par ailleurs, nous avons rencontré d'autres problèmes. Nous avons d'abord pensé qu'il fallait faire un « Explode » du dataframe concaténée comme pour la question 1.3. Mais cette méthode ne gardait que le premier acteur pour chaque ligne, ce qui ne correspondait pas à l'objectif recherché. De manière collective et sans utiliser la méthode « Explode », nous avons réussi à résoudre ce problème en retirant toutes les lignes du dataframe qui ne contiennaient aucun acteur (caracterisées par des 'NaN'). Puis, nous avons créé un nouveau dataframe df1 qui supprimait tous les films/séries sans l'acteur entré en input avec la fonction « str.contains ». Et avec la même ligne de code, nous avons aussi pu rendre les lettres des noms d'acteurs 'insensitive' avec le paramètre 'case=False' : par conséquence l'utilisateur aura les mêmes résultats avec les inputs 'emma watson', 'EMMA WATSON' ou bien 'EmMa WaTSoN'. Avec ces étapes, nous réussissons à exposer tous les résultats qui respectent l'input de l'utilisateur. Pour terminer, nous calculons le nombre de films/séries obtenus pour chaque plateforme avec la fonction Nombre. Grâce à celle-ci, nous suggérons ensuite à l'utilisateur la plateforme qui propose le plus de résultats.

Recherche à partir d'un type et d'un genre de contenu

Dans ce moteur de recherche, il s'agissait de demander à l'utilisateur d'entrer en input le genre du film voulu ainsi que le type de contenu désiré. A partir de ces données, nous devions montrer les trois plus récents films/séries qui satisfont les demandes de l'utilisateur.

Nous sommes partis des dataframes des plateformes en ajoutant une colonne avec le nom de la plateforme. Avant d'appliquer les choix de l'utilisateur, nous avons aussi séparé les dataframes en

deux, selon le type de contenu proposé à l'aide de la fonction dataframe.getgroup(). Nous pouvions par la suite sélectionner les dataframes correspondant au type choisi en input.

Une fois les bons dataframes sélectionnés, nous devions prendre en compte le genre. Or, les genres étaient répertoriés différemment selon les plateformes (comédies v/s comedy). Malgré le fait que nous nous sommes aperçus de ce problème, nous n'avons pas réussi à réunir tous les genres synonymes ensemble. Ainsi, nous avons continué en gardant seulement le genre dans sa forme écrit en input. Nous obtenons à la fin un seul dataframe pour chaque plateforme avec, à chaque fois, le genre et le type demandés. En triant ces dataframes, il a suffi de prendre les 3 premières lignes pour afficher les résultats les plus récents qui respectent les inputs.

Suggestion selon préférences

Ici, il s'agissait de demander à l'utilisateur des informations sur ses préférences pour que la fonction renvoie les suggestions associées. On commence par créer un dataframe qui récupère les notes quand elle existe pour chacun des films/séries des plateformes.

Ensuite, nous avons créé une fonction préférences avec des sous fonctions pour chacun des critères retenus qui va chercher du contenu en fonction des critères retenus. Le problème ici est que nous n'avons pas su aire fonctionner la fonction sans faire appel aux fonctions à l'intérieur du module. Ainsi, il est nécessaire de noter les critères retenus à l'intérieur du module. Autre problème, la partie [3.4.] ne retourne pas l'intersection des résultats en fonction des critères choisis. Ainsi, si l'utilisateur souhaite regarder un film du genre Comédie et tourné en France, la fonction va retourner un dataframe par plateforme et par critère mentionné, soit 9 au total.

Interface

Dans cette partie, deux méthodes se sont offertes à nous. La première demande à l'utilisateur d'écrire ses préférences avec 1 critère ou 2 à la fois. Quant à la seconde, nous avons essayé de créer une interface Tkinter, mais cette méthode n'a pas été concluante puisqu'elle ne retourne pas les recommandations en fonction de ce que l'utilisateur entre dans la barre de recherche. Nous n'avons pas su y inclure les fonctions de recherche présentes dans les modules, mais nous avons tout de même tenu à vous partager notre essai pour que vous puissiez voir le visuel de notre interface « Amix ».

Pour revenir sur la première méthode qui a en revanche fonctionné, en lançant le programme, l'utilisateur a simplement à écrire le nom du film qu'il recherche [3.1.] et notre module s'occupe de trouver si le film est bien présent dans notre base de données concaténée regroupant la base de données de Amazon Prime Video, de Disney + et de Netflix.

De même, si l'utilisateur recherche les films d'un acteur en particulier, comme « Elon Musk », le programme affichera les résultats suivants.

3.2. Recherche par nom d'acteur/actrice

```
[5]: import RECHERCHE_A_PARTIR_DU_NOM_ACTEUR as mr #on importe le module
cast=input("Entrez le nom d'un acteur svp: \n ") #on demande à l'utilisateur son critère de recherche
sortie2= mr.Paracteur(cast)
print(sortie2) #on affiche les résultats
```

```
Entrez le nom d'un acteur svp:
  elon musk
```

	type	title	cast	release_year	listed_in	description	plateforme
14	Movie	Elon Musk: The Real Life Iron Man	Elon Musk, Per Wimmer, Julie Anderson-Ankenbra...	2018	Documentary	Discover the meteoric rise of Elon Musk, the m...	Amazon prime
6555	Movie	Lo and Behold, Reveries of the Connected World	Elon Musk, Dr. Robert Kahn, Kevin Mitnick	2016	Documentary	Oscar®-nominated documentarian Werner Herzog (...	Amazon prime

```
La plateforme suggérée pour obtenir le plus de choix de films/série avec votre actrice/acteur demandé est Amazon prime
```

Il en est de même pour ce qui est des autres critères de préférences.

En revanche, la partie [3.4.] présente un problème puisqu'elle ne retourne pas l'intersection des résultats en fonction des critères choisis. Ainsi, si l'utilisateur souhaite regarder un film du genre Comédie et tourné en France, la fonction va retourner un dataframe par plateforme et par critère mentionné, soit 9 au total. Bien que cela présente un problème, cela peut néanmoins permettre à un utilisateur d'avoir une vision plus large en contenu puisqu'il pourrait finalement être tenté par un film/série qui ne lui aurait pas été suggéré en indiquant trop de critères de préférences. C'est également de cette façon que l'algorithme YouTube fonctionne en suggérant du contenu potentiellement intéressant pour l'utilisateur.

Ce que nous avons appris ?

Réponse de Anisoara ABABII

« Personnellement, j'ai appris à mieux utiliser le langage de programmation Python et à utiliser des fonctions différentes de celles vues en cours. Je me suis persuadée que la syntaxe de Python est très permissive en donnant la possibilité de résoudre le même problème par des méthodes différentes et de construire des codes bien lisibles. Par ailleurs, la partie 3, interface, m'a permis de mieux comprendre comment utiliser les modules (lacune que j'avais avant de commencer le projet). Une chose vraiment importante pour laquelle maintenant je fais attention de plus en plus après ce projet, c'est l'organisation du code, c'est-à-dire que, j'ai constaté qu'une bonne organisation du script peut nous donner le bon résultat tout de suite en écrivant que quelques lignes en Python. En conclusion je voudrais mentionner qu'en travaillant ce projet de manière collective, j'ai enrichi mon horizon de connaissances en ce qui concerne la manipulation des dataframes et du langage Python. Cette expérience m'a donné le désir de continuer à travailler par la suite dans ce domaine. »

Réponse de Armand L'HUILLIER

« Plus que tout, j'ai appris à résoudre des problèmes au niveau de mon code en déchiffrant les messages d'erreur que Jupiter affichait. Ceci m'a mené à parfois changer de stratégie pour atteindre un même objectif. Aussi, j'ai dû utiliser des techniques et des fonctions nouvelles grâce à des forums et aux diverses ressources sur internet. En finalité, j'ai pu utiliser des ressources diverses que je réussissais à adapter à mes objectifs. Une difficulté que j'ai rencontrée plusieurs fois venait de l'utilisation d'une fonction de fonction. Mais au fur et à mesure que j'avançais dans le projet, j'ai réussi à utiliser l'output d'une fonction, dans une autre fonction. »

Réponse de Yanis REHOUNE

« Tout d'abord, ce projet ma fait progresser en terme d'algorithmie pure. Il m'a permis de réfléchir à comment aborder les problèmes efficacement, à schématiser les étapes avant même de coder. J'ai notamment progressé dans ma manière de réaliser mes fonctions, ce qui m'a permis d'avoir des codes plus efficient. De plus, j'ai appris à chercher efficacement les solutions à mes erreurs et me familiariser avec les sites Internet et forums références comme Stack Overflow. En effet, lorsque l'on est bloqué, j'ai compris qu'il est important de savoir formuler efficacement son problème pour avancer. Ce projet m'a également appris à structurer un projet grâce aux modules qui permettent de clarifier le code. Enfin, j'ai appris à travailler en groupe sur un projet de programmation ce qui est une première pour moi et qui sera très utiles pour les prochains projets et stages. Nous avons apprécié travailler sur ce projet »

Réponse de Camil ZAHI

« Ce projet m'a permis d'adopter certains réflexes lorsque l'on travaille sur des projets en data-science. Il est important d'avoir un esprit organisé. Toujours savoir ce qu'on a en input et ce qu'on veut avoir en output. Poser le problème et prendre le temps de voir comment on le résout. Apprendre c'est une chose, comprendre en est une autre, mais savoir-faire, c'est se poser des questions qu'on ne se poserait pas dans les autres cas. Alors il est vrai que parfois les erreurs retournées par Python ne sont pas les bienvenues, mais le sentiment de récompense une fois l'erreur résolu, en vaut réellement l'intérêt de s'être poser sur le problème. Par ailleurs, ce projet m'a aussi apporté une composante comportementale sur la manière de travailler et collaborer en équipe. Un grand merci à mes trois collaborateurs, avec qui j'ai eu grand plaisir de réaliser ce projet. »