CORNATON Maxime
NGUYEN Hien

CPE LYON
4ETI - Conception Logicielle et Big Data

RAPPORT DE PROJET - DATA MINING

Introduction

L'objectif principal de ce projet est de mettre en place un système de recommandation d'images pour un utilisateur à partir des choix de ce-dernier. Il est de plus dans notre mission d'apporter une représentation graphique des préférences des différents utilisateurs ainsi que la répartition des tags des images constituants notre base de données. Il est important de noter que cette base de données est constituée exclusivement d'images libres de droit issues de la plateforme Wikidata.

I. Collecte des données

Le processus de récupération des images est entièrement automatisé.

1) Mise en place de la structure d'accueil

Dans un premier temps, nous vérifions qu'un répertoire 'images' est déjà créé, si ce n'est pas le cas, alors il l'est.

De la, nous regardons s'il contient des images, si c'est le cas, le processus de téléchargement suivant n'est pas nécessaire.

2) Requêtes a Wikidata

Nous récupérons les images des différentes catégories Wikidata (Q…) dont nous définissons un nombre d'images à télécharger respectivement (limit).

Le choix des catégories ainsi que le nombre de photos à récupérer est bien évidemment modifiable.

C'est via des requêtes sur Wikidata que les images peuvent être retrouvées. En effet, une requête permet de récupérer les liens de chaque image listée dans une catégorie.

Seulement quelques problèmes surviennent car wikidata impose ces conditions:

- *One client (user agent + IP) is allowed 60 seconds of processing time each 60 seconds*
- *One client is allowed 30 error queries per minute*

C'est pour cela que nous pouvons avoir un téléchargement un peu long avec quelques périodes d'attente dans notre code (time.sleep()) ainsi que des échecs lors de certains téléchargements.

3) Téléchargement

Le téléchargement se fait simplement grâce à la librairie urllib qui enregistre les images une à une à partir de leur lien.

Nous enregistrons chaque image sous le format suivant:
'identifiant_wikidata_image'-'identiffiant_wikidata_categorie'.jpg

Cela nous permettra de récupérer ces données à n'importe quel moment, en particulier lorsque les images seront déjà téléchargées.

4) Sauvegarde

Parallèlement à cela, nous initialisons toutes les clés d'un dictionnaire en lui ajoutant pour chaque image le nom donné à l'image téléchargée.

II. Etiquetage et annotation

L'analyse des images par la suite reposera sur ce que nous aurons étiqueter, annoter et enregistrer sur chaque image.

1) Données Exifs

Dans les images téléchargées, il nous était possible de retrouver leur taille, leur format, et pour certaines des dates (clé exif 36867) et des modèles de caméra (clé exif 272).

Ce sont des données qu'il a été facile de récupérer et qui nous ont permis d'en déduire d'autres.

Effectivement, à l'aide des données Exif et des multiples informations qui sont extraites des métadonnées de l'image. Nous pouvons créer des tags supplémentaires afin de proposer de meilleures recommandations. Nous proposons ainsi pour chaque élément de la base de données des informations sur :

- l'orientation de l'image (landscape, portrait, square)
- la taille de l'image (small, medium, large)

2) Couleurs dominantes

Nous pouvons analyser les images en utilisant des algorithmes de regroupement pour trouver les couleurs prédominantes.

En plus des données citées précédemment, nous analysons les couleurs présentes dans le fichier dans le but d'en tirer la couleur dominante présente dans la photo. Pour cela, nous avons utilisé *MiniBatchKMeans* de sklearn.

3) Tags

Pour l'ajout des tags dans les données de chaque images, il nous était soufflé l'idée de les rentrer un à un à la main, au moins pour une partie des photos comme pourrait le faire un utilisateur.

Nous avons choisi d'automatiser ce processus afin d'obtenir plus simplement et pour toutes les images des tags.

Pour cela, à partir des noms des images, nous effectuons des requêtes API à wikidata pour récupérer les informations de chaque catégorie dont les images appartiennent et en particulier le libellé de la catégorie en anglais ('en') par défaut dans notre code.

Nous avons commenté une seconde partie du code, jugeant que la récupération d'un tag était satisfaisante pour ce projet, qui permettait à partir du nom des images, d'effectuer une requête API sur l'image elle-même dans wikidata. Nous récupérons plusieurs propriétés wikidiata dont certaines pouvaient faire office de tags.

4) Exportation des données

Enfin nous exportons toutes nos informations du dictionnaire vers un fichier JSON.

III. Analyses

1) Utilisateur

Nous avons développé une classe User pour gérer toutes les informations requises à l'ajout d'un utilisateur à la base de données ainsi que ses préférences.

Par la suite, nous créons alors un profil utilisateur qui aime aléatoirement entre 10 et 20 images de notre base de données pour effectuer l'analyse de son profil.

A noter que les valeurs apparaissant par la suite sont les résultats d'une analyse effectuée sur ***150 images****.*

2) Préparation des données

Le traitement des données est fait via la bibliothèque Python pandas. Nous lisons toutes les données présentes dans notre fichier JSON. (Annexe 1)

Nous ne traitons pas les dates et les modèles d'appareils.

Nous décidons de séparer en 2 colonnes les valeurs rgb des 2 couleurs dominantes par images que nous traduisons en hexadécimal. De la, nous traduisons toutes nos données en valeurs numériques grâce aux LabelEncoders.

Enfin, nous créons un second dataFrame avec une colonne correspondant aux likes de l'utilisateur.

3) Apprentissage et Test

Nous avons utilisé Scikit-learn.

Dans un premier temps nous séparons nos données en données d'entraînement et en données de test avant de normaliser ces données.

Afin d'offrir à l'utilisateur des images susceptibles de lui plaire, nous avons opté pour des approches basées sur la classification multi-classe (Decision Tree et Random Forest Classifiers). Nous avons opté pour ces choix en raison de la multitude de tags différents que nous voulions prendre en compte. De plus, nos données n'étant pas linéaires, le choix d'une classification binaire tel qu'un perceptron simple n'est pas adapté.

Decision Tree Classifier

Nous entraînons ce modèle et évaluons notre ensemble de test. Il est intéressant d'afficher l'arbre de décision (Annexe 3), l'importance des variables (Annexe 4) et l'évaluation du modèle (Annexe 5).

Le modèle a été évalué sur un ensemble de données contenant 30 images. Le modèle a prédit deux classes, notées 0 et 1 - liké et pas liké.

La précision est une mesure de la capacité du modèle à prédire correctement les exemples positifs. Dans ce cas, la précision pour la classe 0 est faible, à 0,25, ce qui indique que le modèle a eu du mal à identifier les exemples positifs pour cette classe. Pour la classe 1, la précision est de 1,00, ce qui indique que le modèle a identifié tous les exemples positifs de manière précise.

Le rappel est une mesure de la capacité du modèle à identifier tous les exemples positifs. Ici, le rappel pour la classe 0 est de 1,00, ce qui signifie que le modèle a correctement identifié tous les exemples positifs de cette classe. Pour la classe 1, le rappel est de 0,90, ce qui indique que le modèle a manqué quelques exemples positifs.

Le score F1 est une moyenne harmonique de la précision et du rappel, qui est une mesure globale de la performance du modèle. Le score F1 pour la classe 0 est faible, à 0,40, en raison de la faible précision de cette classe. Le score F1 pour la classe 1 est de 0,95, ce qui est élevé en raison de la précision élevée de cette classe.

L'exactitude est une mesure de la capacité du modèle à prédire correctement toutes les classes. Ici, l'exactitude est de 0,90, ce qui est élevé, mais ne donne pas une image complète de la performance du modèle car les classes ne sont pas équilibrées.

Enfin, les mesures macro-avg et weighted-avg sont des moyennes des mesures de précision, de rappel et de score F1 pour toutes les classes. La moyenne macro-avg donne une importance égale à toutes les classes, tandis que la moyenne weighted-avg tient compte de la taille de chaque classe. Dans ce cas, la moyenne macro-avg est faible pour la précision, à 0,62, en raison de la faible précision de la classe 0. La moyenne weighted-avg est plus élevée, à 0,97 pour la précision, car la classe 1 est beaucoup plus grande que la classe 0.

Random Forest Classifier

Et si nous avions la possibilité d'avoir plusieurs arbres de décision ? C'est ce que nous faisons en utilisant Random Forest Classifier (peut être vu comme une collection de plusieurs arbres de décision).

De la même manière que précédemment, on l'entraîne et on extrait les annexes 6 et 7..

Dans ce cas, la précision pour la classe 0 est élevée, à 1,00, ce qui signifie que le modèle a correctement identifié tous les exemples positifs de cette classe. Cependant, pour la classe 1, la précision est de 0,83, ce qui indique que le modèle a identifié la plupart des exemples positifs de manière précise, mais a commis des erreurs dans quelques cas.

Ici, le rappel pour la classe 0 est très faible, à 0,00, ce qui signifie que le modèle n'a pas identifié d'exemples positifs pour cette classe. Pour la classe 1, le rappel est élevé, à 1,00, ce qui indique que le modèle a correctement identifié tous les exemples positifs de cette classe.

Le score F1 pour la classe 0 est très faible, à 0,00, en raison du rappel nul pour cette classe. Le score F1 pour la classe 1 est de 0,91, ce qui est élevé malgré la précision légèrement inférieure à 1.

Ici, l'exactitude est de 0,83, ce qui est élevé, mais ne donne pas une image complète de la performance du modèle car les classes ne sont pas équilibrées.

Dans ce cas, la moyenne macro-avg est faible pour la précision, à 0,92, en raison de la précision parfaite pour la classe 0 et de la précision légèrement inférieure pour la classe 1. Le rappel macro-avg est très faible, à 0,50, en raison du rappel nul pour la classe 0. La moyenne weighted-avg est plus élevée, à 0,86 pour la précision, car la classe 1 est beaucoup plus grande

que la classe 0. Le score F1 weighted-avg est de 0,76, ce qui est relativement faible en raison du rappel nul pour la classe 0.

IV. Visualisation des données

La visualisation est faite avec la bibliothèque Python matplotlib. Nous avons opté pour un affichage sous forme d'un histogramme le nombre d'images selon les informations. Cela permet de visualiser rapidement les caractéristiques de notre base de données.

Nous avons ainsi obtenu les différents graphes en annexe 8:

- nombre d'images par année
- nombre d'images par taille
- nombre d'images par orientation
- nombre d'image par modèle de caméra puis par marque
- caractéristiques 2D et 3D des couleurs

Nous aurions souhaité affiché de la même manière les préférences de l'utilisateur, mais ce n'est pas implémenté au moment de ce rendu.

V. Système de recommandation

Pour automatiser les tests, nous avons créé un user dont les préférences (like/dislike) ont été choisies aléatoirement parmi les images récupérées. Cependant, cette méthode montre des limites car elle ne permet pas d'obtenir une tendance de goût réelle pour un utilisateur car les choix "like/dislike" sont fait arbitrairement.
On le voit de par la longueur et la complexité de notre arbre, ainsi que les résultats d'analyse de nos modèles de classification.

C'est pour cela que nous avons décidé de créer une section d'Expérimentation par la suite, à laquelle nous n'avons malheureusement pas eu le temps d'implémenter tous les différents tests d'évaluations du modèle.

VI. Remarques

Pour aller plus loin, voila les améliorations possibles:

- téléchargements des images sans bugs sans problèmes avec wikidata pour augmenter le nombre d'images
- implémentation de la visualisation des données utilisateur
- implémentation des tests d'évaluations du modèle expérimental.

VII. Conclusion

La réalisation de ce projet a permis d'explorer en profondeur les concepts et les algorithmes abordés pendant les cours et les travaux pratiques.

Grâce à une collecte automatisée d'images à partir d'un site web et à un processus d'étiquetage autonome, nous avons pu accéder à de nouvelles informations nous permettant de créer des graphiques et des diagrammes aidant à la bonne compréhension de la base de données.

Cette problématique est très pertinente dans le contexte actuel des réseaux sociaux. De plus, la manipulation de graphiques et la visualisation des données ont été des compétences clés pour rendre les données plus accessibles et compréhensibles. En somme, ce projet a été une occasion unique de mettre en pratique les connaissances acquises tout au long du cours.

ANNEXES

Annexe 1

	size	orientation	format	colors	tags	date	model
Q194215-Q23442	medium	landscape	JPEG	[[39, 41, 30], [202, 207, 160]]	[island]	NaN	NaN
Q53216650-Q146	large	portrait	JPEG	[[216, 215, 219], [67, 61, 64]]	[house cat]	2012:01:05 11:35:19	DMC-FZ20
Q193253-Q23442	large	landscape	JPEG	[[95, 86, 78], [30, 37, 89]]	[island]	NaN	NaN
Q613928-Q3305213	large	portrait	JPEG	[[65, 55, 48], [137, 135, 119]]	[painting]	NaN	NaN
Q4485643-Q14660	small	landscape	PNG	[[125, 126, 124], [93, 34, 97]]	[flag]	NaN	NaN
...	...	...	...	...	...	...	...
Q611583-Q3305213	large	landscape	JPEG	[[59, 40, 24], [151, 119, 84]]	[painting]	2008:04:24 08:46:09	Hasselblad CF528-39
Q7704028-Q144	medium	landscape	JPEG	[[173, 173, 173], [36, 36, 36]]	[dog]	NaN	NaN
Q384348-Q144	large	landscape	JPEG	[[177, 166, 151], [82, 75, 62]]	[dog]	2005:07:03 19:53:46	DMC-FZ20
Q1320354-Q144	medium	portrait	JPEG	[[25, 20, 23], [215, 222, 209]]	[dog]	NaN	NaN
Q196042-Q23442	medium	landscape	JPEG	[[190, 194, 188], [35, 48, 59]]	[island]	NaN	NaN

Données des images

Annexe 2

	size	orientation	format	couleur_1	couleur_2	tag
Q194215-Q23442	1	0	0	19	120	3
Q53216650-Q146	0	1	0	134	31	2
Q193253-Q23442	0	0	0	65	11	3
Q613928-Q3305213	0	1	0	34	69	4
Q4485643-Q14660	2	0	2	82	44	1
...	...	...	...	...	...	...
Q611583-Q3305213	0	0	0	30	74	4
Q7704028-Q144	1	0	0	105	14	0
Q384348-Q144	0	0	0	110	38	0
Q1320354-Q144	1	1	0	7	131	0
Q196042-Q23442	1	0	0	114	13	3

Données traitées des images

Annexe 3

Arbre de décision - dtc

Annexe 4

Importance des variables :
1. couleur_2 (0.576450)
2. size (0.195052)
3. orientation (0.120120)
4. tag (0.060095)
5. couleur_1 (0.048283)
6. format (0.000000)

Importance des variables - dtc

Annexe 5

	precision	recall	f1-score	support
0	0.25	1.00	0.40	1
1	1.00	0.90	0.95	29
accuracy			0.90	30
macro avg	0.62	0.95	0.67	30
weighted avg	0.97	0.90	0.93	30

Evaluation du modèle - dtc

Annexe 6

Importance des variables :
1. couleur_2 (0.395777)
2. couleur_1 (0.312987)
3. tag (0.138456)
4. size (0.073640)
5. orientation (0.040545)
6. format (0.038595)

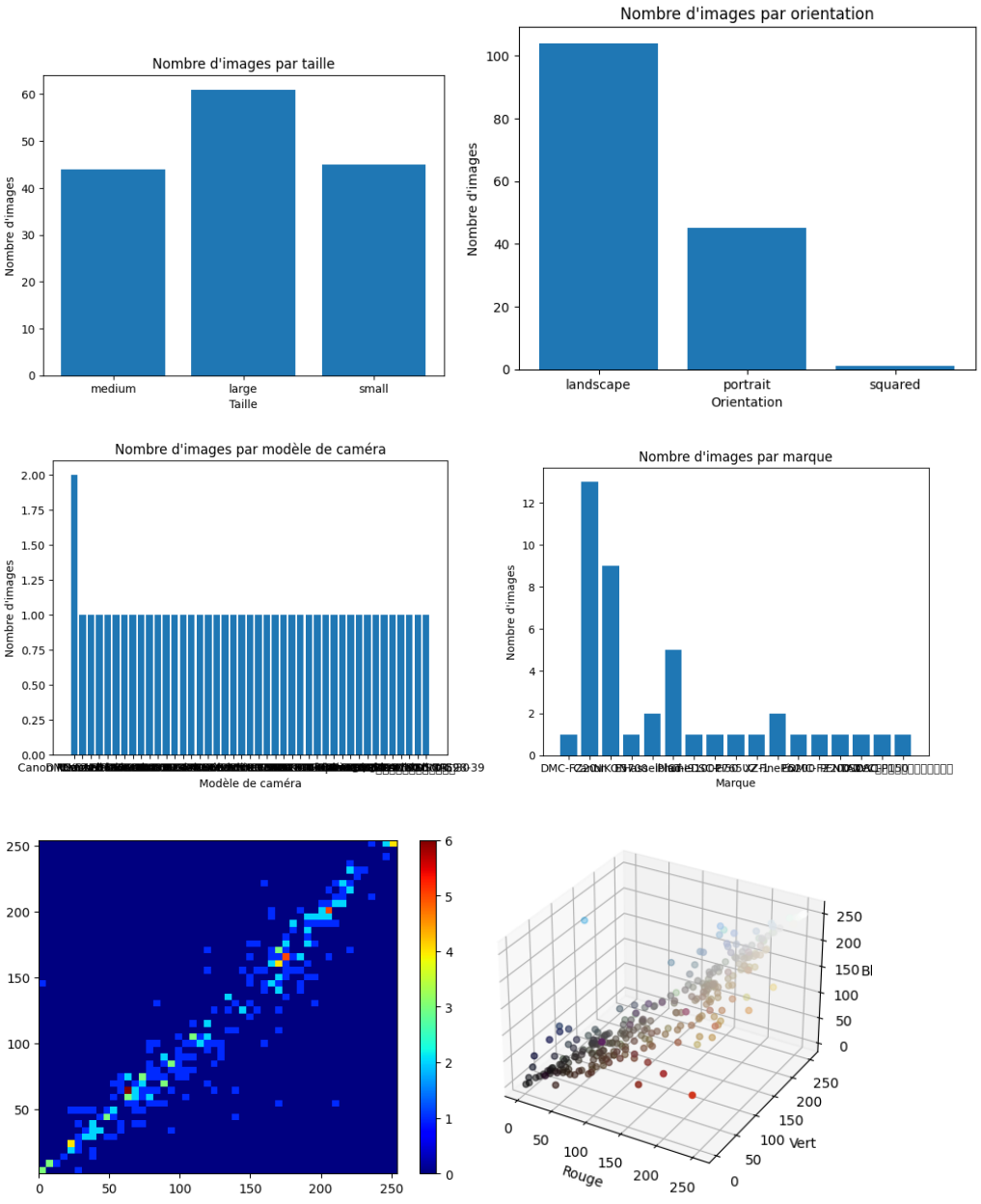
Importance des variables - rfc

Annexe 7

	precision	recall	f1-score	support
0	1.00	0.00	0.00	5
1	0.83	1.00	0.91	25
accuracy		0.83	30	
macro avg	0.92	0.50	0.45	30
weighted avg	0.86	0.83	0.76	30

Evaluation du modèle - rfc

Annexe 8



Visualisation de nos données

Bibliographie

https://www.mediawiki.org/wiki/Wikidata_Query_Service/User_Manual#:~:text=One%20client%20is%20allowed%2030%20error%20queries%20per%20minute

https://link.infini.fr/johnsamueldatam