Rapport de projet DATA MINING

Joseph Pouradier-duteil & Pierre-Louis TELEP

Sommaire

- Rapport de projet DATA MINING
  - Sommaire
  - 1. Introduction
  - 2. Présentation du projet
  - 3. Présentation des données
  - 4. Analyse des données `JO`
  - 5. Prédiction
  - 6. Auto-évaluation
  - 7. Remarques
  - 8. Conclusion

1. Introduction

Ce rapport a pour but de présenter le projet de Data Mining. Le projet a été réalisé par **Joseph POURADIER-DUTEIL** et **Pierre-Louis TELEP**.

Le but de ce projet est de réaliser un système de recommandation. Un système de recommandation est un algorithme qui va analyser les données des images proposées à l'utilisateur. En fonction de si l'utilisateur aime ou pas les images l'algorithme va pouvoir proposer de nouvelles images et prédire si l'utilisateur va aimer ou non cette image. Pour cela nous avons utilisés des images de `Motos`,`Voitures`,`Pokemons` et `Exoplanet`.

2. Présentation du projet

Ce projet a pour but de faire un système de recommandation. Pour cela nous avons utilisés des images de `motos`,`voitures`,`Pokémons` et `planètes`. Nous avons utilisés des images trouvées en ligne sur WikiData que nous avons ensuite analyser pour récuperer leurs métadonnées.

Pour la partie de recommandation nous avons utilisé `SKLearn` et les `DecisionTree` pour analyser les choix de l'utilisateur en fonction des images proposées et prédire des images qu'il aime ou pas. Nous avons fait un deuxieme modéle de recommandation mais basé sur des préférences utilisateur aléatoire.

3. Présentation des données

Nous sommes aller sur wikidata pour recupérer les images de `motos`,`voitures`,`Pokémons` et `planètes`. Pour télécharger les images nous avons utilisés un csv contenant les Urls des images.

Pour ensuite télécharger les images nous avons utilisé un script python qui va parcourir le csv et générer un dictionnaire contenant le nom de l'image, le dossier où enregistrer l'image, le lien pour téléchager l'image ainsi que certaines métadonnées. Ce même script va ensuite télécharger chaque image en utilisant `requests` avec des `get` et les enregistrer dans un dossier.

Une fois que toutes les images étaient téléchargées nous les avons utilisé un script python pour les mettre les photo au même format. Pour cela nous avons utilisé `PIL` et `numpy` pour les redimensionner au format 16:9, les normaliser et les convertir en `RGB`.

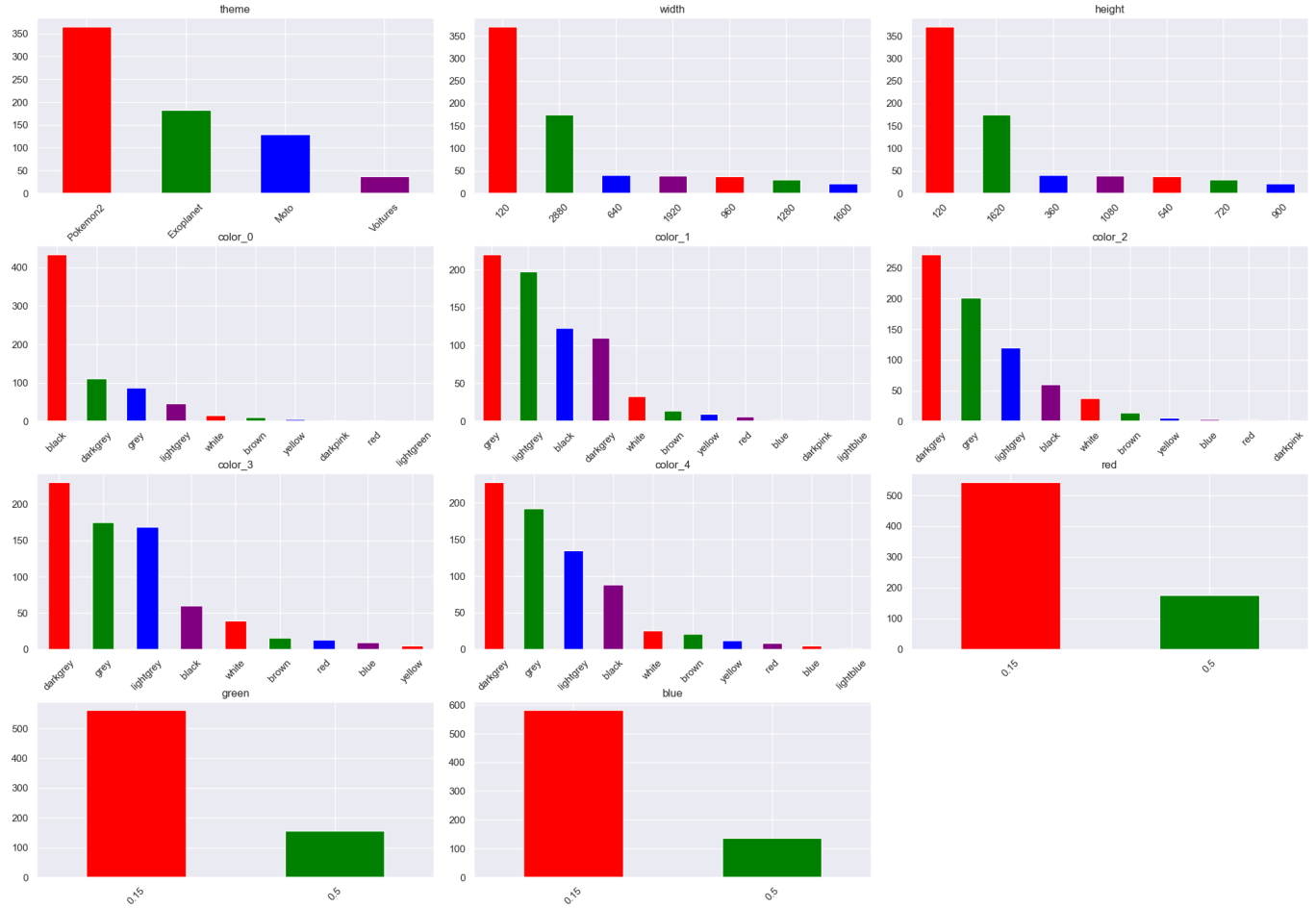
4. Analyse des données `JO`

Après avoir récupéré toutes les images, nous avons récupéré leurs métadonnées.
Nous avont utilisé les données exif des images tel que son theme, sa taille, son orientation quand elle était disponible et le format. Ensuite nous avons relevé les 5 coleurs prédominantes dans les images grace a l'algorithme de K-means. Par soucis de performance nous avons utilisé les K-means MiniBatch qui est une version plus rapide des K-means. Toutefois le resultat des K-means était en hexadecimal et nous donnais une trop grande diffusion des couleurs (tres peu probable d'avoir deux fois la meme couleur meme parmis nos 720 images). Nous avons donc utilisé la fonction color_hex_to_rgb pour convertir les couleurs en hexadecimal. Puis nous avon pris la couleur le plus proche dans un lot de 17 couleurs (closest_color). Nous avons ensuite utilisé l'histogram des pixels rgb de chaque image pour en prendre une valeur normalisé entre 0 et 1: on fait une moyenne sur l'histogramme puis on prend la valeur normalisé de cette moyenne (getDiagram).

Enfin nous avons écris toutes ces données dans un fichier json : ExifDatatest2.json

On peut maintenant en visualiser la répartition des données sur un graphique:

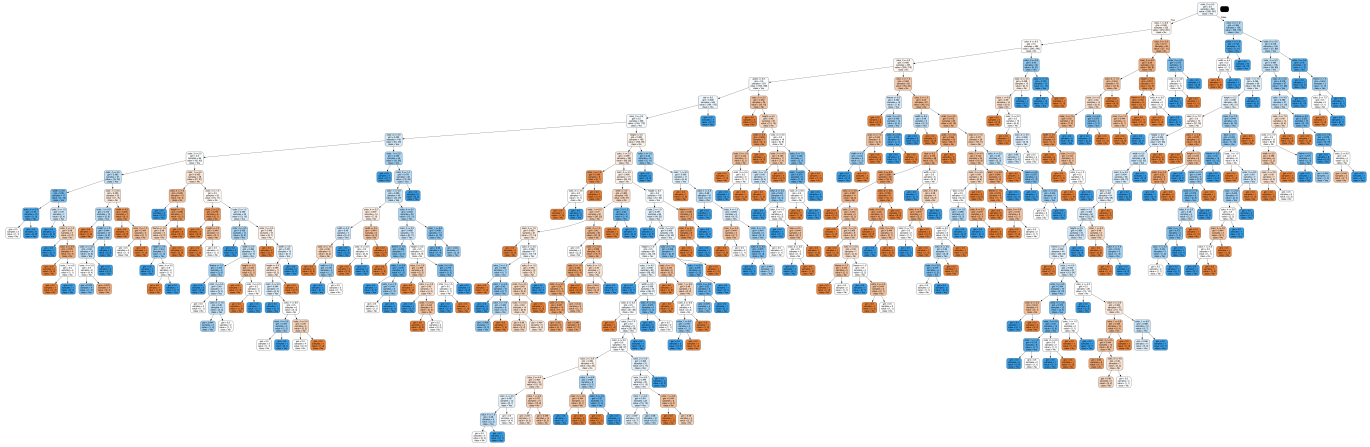
Number of element per value of each columns



5. Prédiction

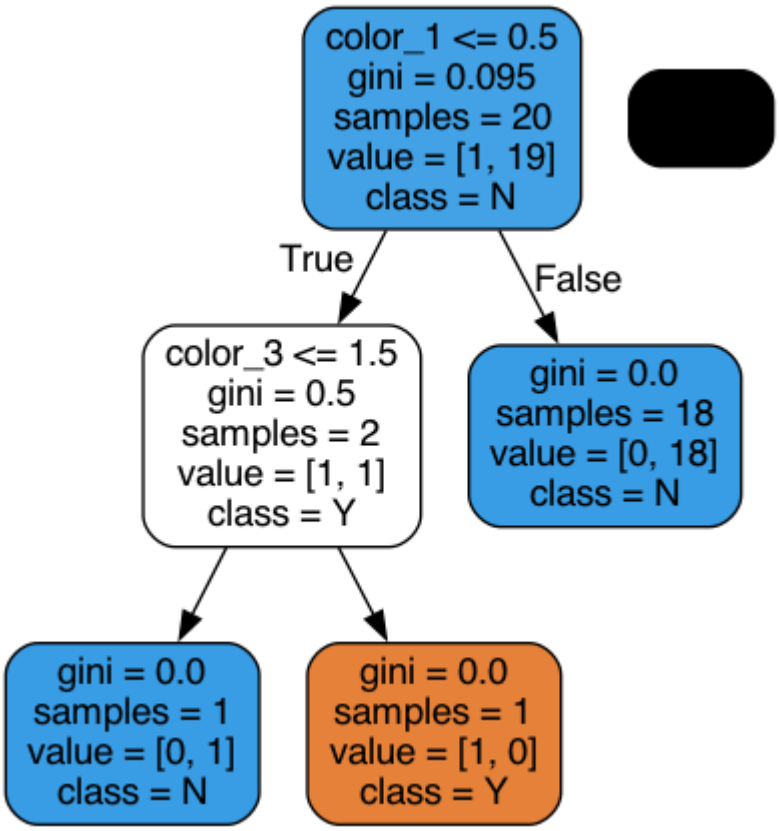
Pour la partie prédiction nous avons utilisé 2 manière de faire différentes. La première est basé sur des préférences utilisateur aléatoire. La deuxième est basé sur les préférences utilisateur.

Pour la première méthode nous avons rempli la colonne préférence pour chaque image de façon aléatoire. Pour ensuite générer une prédiction en fonction de ces préférences avons utilisé `SKLearn` et les `DecisionTree` pour analyser les choix de l'utilisateur en fonction des images proposées et prédire des images qu'il aime ou pas. Evidemment cette méthode n'est pas très fiable car les préférences sont aléatoires. On voit dans le modèle de décision qu'il y a beaucoup de noeuds qui ne font pas forcément sens.



.

Pour la deuxième façon de faire nous avons proposés à l'utilisateur 20 images et il devait choisir celles qu'il aimait. Nous avons alors analysé les préférences de l'utilisateur. Avec ces analyses nous avons pu proposer 3 images à l'utilisateur dont le modèle avait prédit si la personne allait aimé ou non. Dans l'abre de décision on voit que les noeuds sont plus cohérents et que le modèle est plus fiable.



6. Auto-évaluation

ff

7. Remarques

ff

8. Conclusion

ff