Notation automatique d'album photo

Table of Contents

Problématique

Contexte

Un site de stockage de photos propose à ses utilisateur de donner leur avis sur les photos exposées. Pour mettre en valeur les plus belles photos, le site aimerait pouvoir noter automatiquement les albums. La notation se baserait sur les notes mises par les utilisateur, pour améliorer de façon continue l'affichage de “belles” photos.

Objectifs

Notre but sera donc d'analyser les photos et de proposer une méthode pour les noter.

Particularités

À cause de la taille mémoire de photos il est impossible de faire un traitement sur les photos en elle même. Nous allons donc faire notre analyse à partir des méta données des différents albums.

Le jeux de données

Pour chaque album nous avons les informations suivantes :

- Latitude : latitude de l'album photo (sous forme entière)
- Longitude : longitude de l'album photo (sous forme entière)
- Width : largeur moyenne des images de l'album (pixels)
- Height : hauteur moyenne des images de l'album (pixels)
- Size : nombre d'images dans l'album
- Name : identifiants du nom de l'album
- Description : identifiants dans la description de l'album
- Caption : identifiants de toutes les photos de l'album
- Good : les utilisateurs aiment l'album (1) ou n'aiment pas (0)

Pour les champs, Name, Description et Caption nous n'avons pas des mots mais des identifiants. Les données ont été pré-traités pour ne laisser que les mots apparaissant au total plus de 20 fois

Nous avons en tout 40 262 albums à analyser.

Nos résultats

Nous avons commencé le problème par analyser quelles pourraient être les pistes nous permettant de noter les photos. Ensuite nous avons créer une fonction de notation de nouveau albums.

Résultats selon trois critères

Nous avons finalement fait ressortir Trois critères pouvant nous aider à noter les albums

Localisation

Nous avons pu déterminer des zones avec beaucoup d'album bien notés ou au contraire des albums mal notés.

Taille des albums

Nous nous somme rendu compte que nous pouvions avoir une relation direct entre l’appréciation d'un album et le nombre de photos dans un album.

Étude des mots

Nous avons fait une étude des mots utilisé par les albums bien noté et les albums mal noté pour déterminer quels étaient les mots pouvant pousser l'utilisateur à mal noter les photos.

Notation

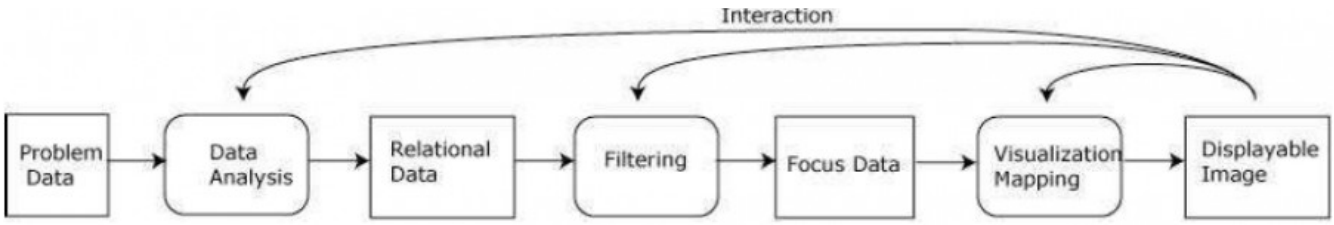
Finalement nous avons écrit un algorithme de notation nous permettant de déterminer si des albums

vont être apréciées ou non juste avec leurs méta-données.

En l'état actuel des choses, nous n'avons pas une estimation parfaite, il faudrait faire des tests pour trouver les paramètres optimaux pour obtenir de meilleurs résultats.

Visualisation

Approche du problème



Pour cette analyse nous ne pouvons pas obtenir de données supplémentaires, cependant nous n'avons pas de traitement à faire sur les données, tout est bien formaté.

Dans un premier temps nous nous sommes concentré sur données métriques que nous avons (longitute, latitude, height, weight et size + good).

Nous avons ressorti trois pistes pour l'analyse :

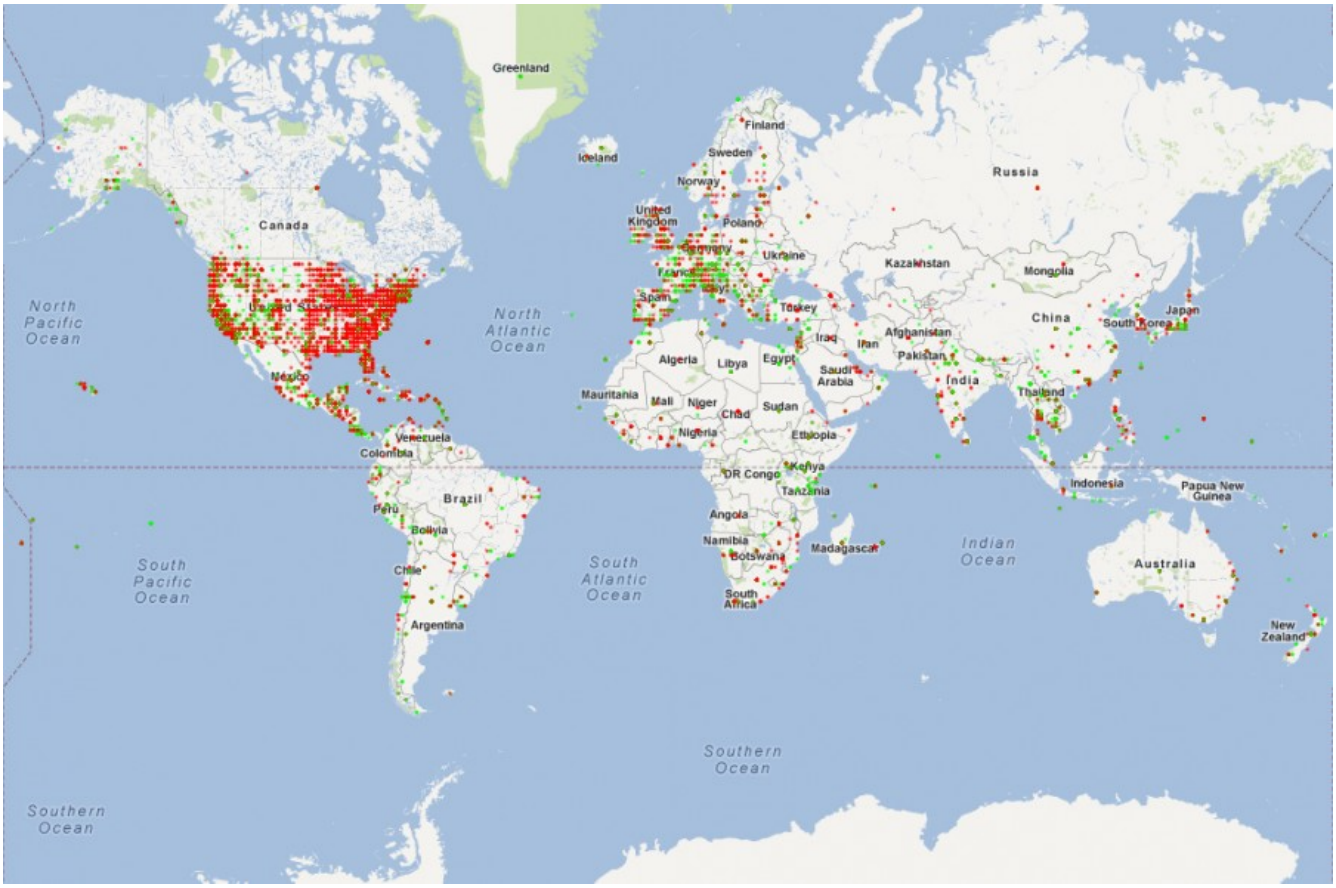
- La localisation des photos.
- la taille moyenne des images dans l'album
- Le nombre de photos dans l'album

Après avoir analysés les métriques des données nous avons concentré notre analyse sur les identifiants (mots).

Localisation

Travail effectué

Nous avons commencé par disposer nos différents albums sur une carte (via le plug-in "Google maps" de tulip) pour avoir un apperçu global de la situation



Nous avons coloré les points (représentant les albums) en fonction de leur appréciation 1→ vert; 0→ rouge.

Grâce à cette visualisation on peut facilement voir d'où ont été prises les photos. Cependant on remarque que la répartition des points sur la carte est étrange, on regardant les données on s'aperçois que les longitudes et latitudes ont été arrondis. Malheureusement, on ne peut rien faire pour arranger ces données.

À cause de la superposition des points on ne peut voir que la couleur d'un point par coordonnées.

Nous avons donc mis en place un algorithme de “quadrillage” permettant d'avoir un aperçu moyen de l’appréciation en fonction de zones de tailles prédéfinies.

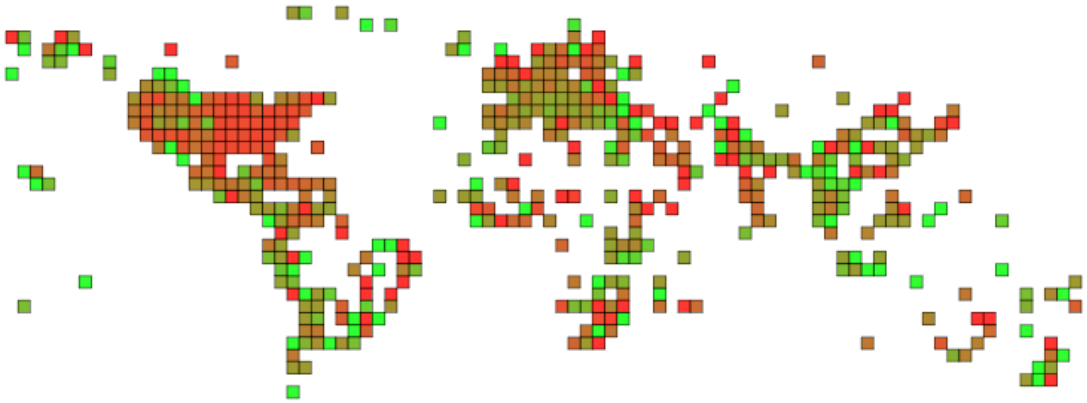
Algorithme

- Trouver la taille de la boite englobante de nos points.
    - Trouver xmin, xmax, ymin, ymax /!\ il est possible que pour que ça soit plus facile, il faille repasser toutes les coordonnées en positives
- déterminer le nombre de carrés souhaité
    - (appréciation personnelle)
- Créer un sous graphe “quadrillage” qui contiendra comme node ces “carrés”

    - tlp.newSubGraph(root[, name = “unnamed”])
    - tlp.newCloneSubGraph(root[, name = “unnamed”])
- Pour chaque carré garder tous les individus associés
    - graph.addNode()
- le noeud aura la taille du nombre d'individu (/!\ aux tailles limites) Pas implémenté
    - ajouter un variable “nbInd” au noeud : self.nbInd = graph.getDoubleProperty(“nbInd”)
    - changer sa taille : siz = tlp.Size(x, y, z); self.viewSize.setNodeValue(n, siz)
- faire un moyenne sur la variables “good” par carré.
- faire un dégrader vert-rouge avec cette moyenne
    - Utiliser l'algorithme de base de python
        - à la main avec l'API
        - en python avec : http://tulip.labri.fr/Documentation/3_6/pythonDoc/graphalgorithms.html?highlight=call
    - *Problèmes pour choisir le dégradé que l'on souhaite, on le fait manuellement.*

Fichier Code/quadrillage.py

On obtient la représentation suivante.



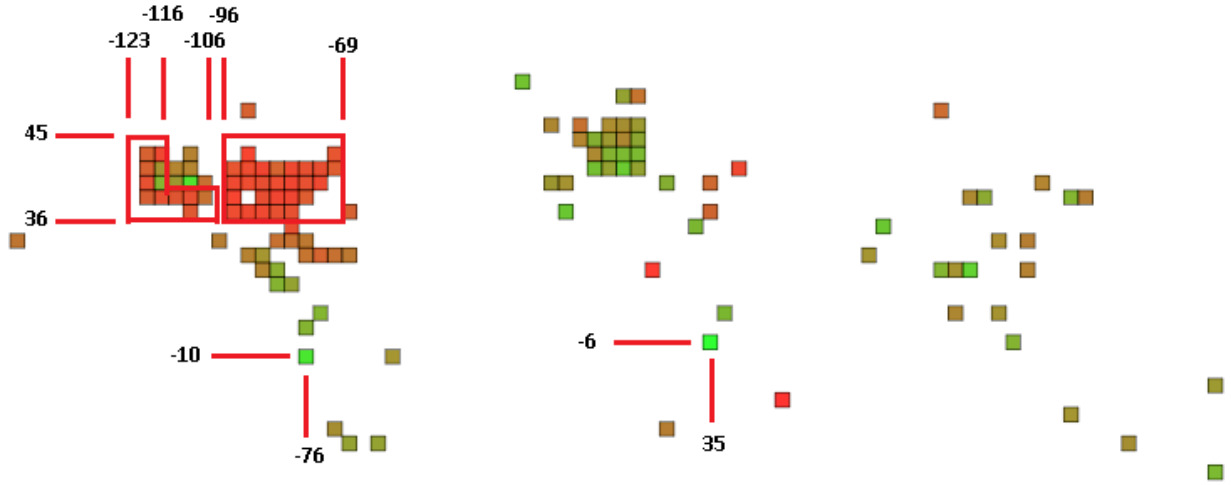
Avec cette représentation on a un problème, certain carré ne continent qu'un ou très peu d'album, s'il ont la même note il ont une moyenne très forte (ou très faible) et ressortent donc, cepandant ils ne sont pas forcement représentatifs de leur région.

On a donc fait un seuil sur le nombre d'album par carré, cela nous permet de faire ressortir uniquement les zones ayant un certain “poids”. En mettant un certain seuil nous obtenons :

Voir la différence entre les deux zones. Colorisation uniforme pour faire ressortir les bonnes et mauvaises zones.

Analyse

Avec nos visualisation nous avons pu faire ressortir certaines zones du mon avec une forte (ou faible) moyenne d’appréciation.

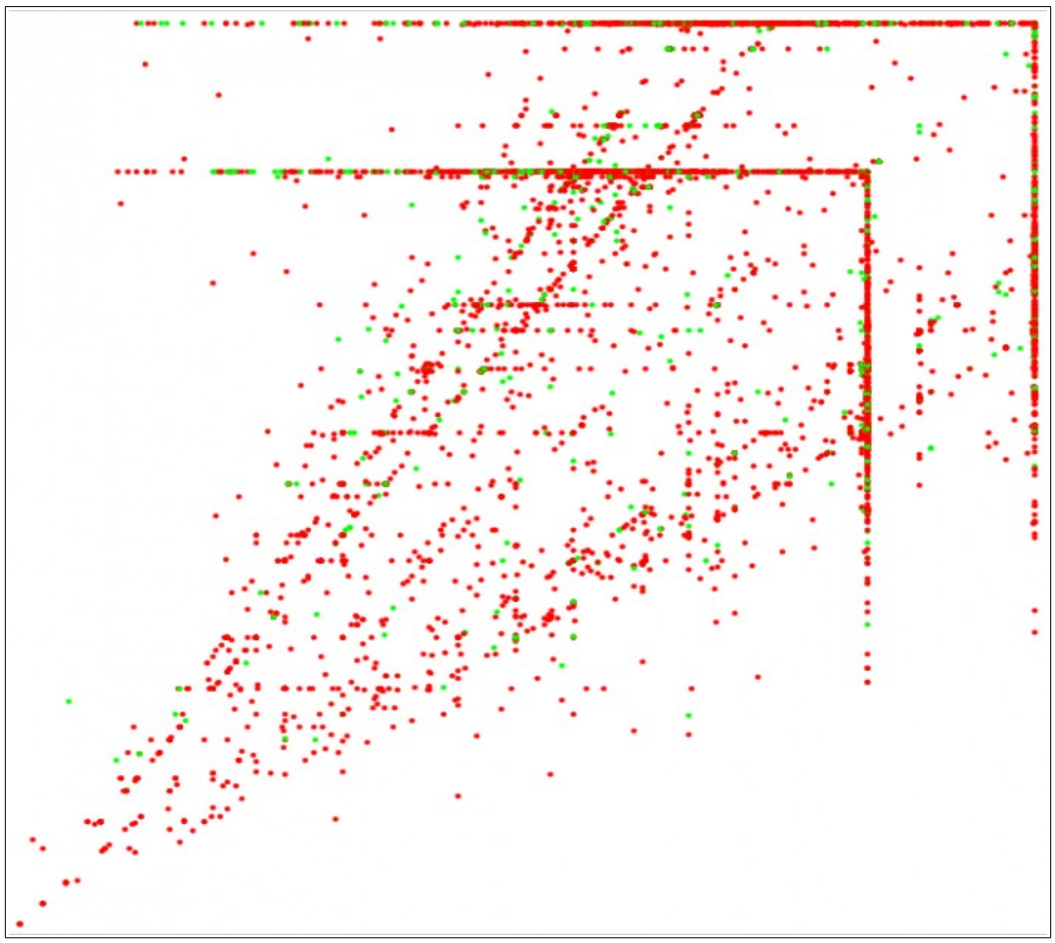


Cependant nous avons fait ressortir trop peu de zones, pour pouvoir noter les futures photos il faudra surement être moins strictes, nous réglerons les différents seuils pour la notation des photos.

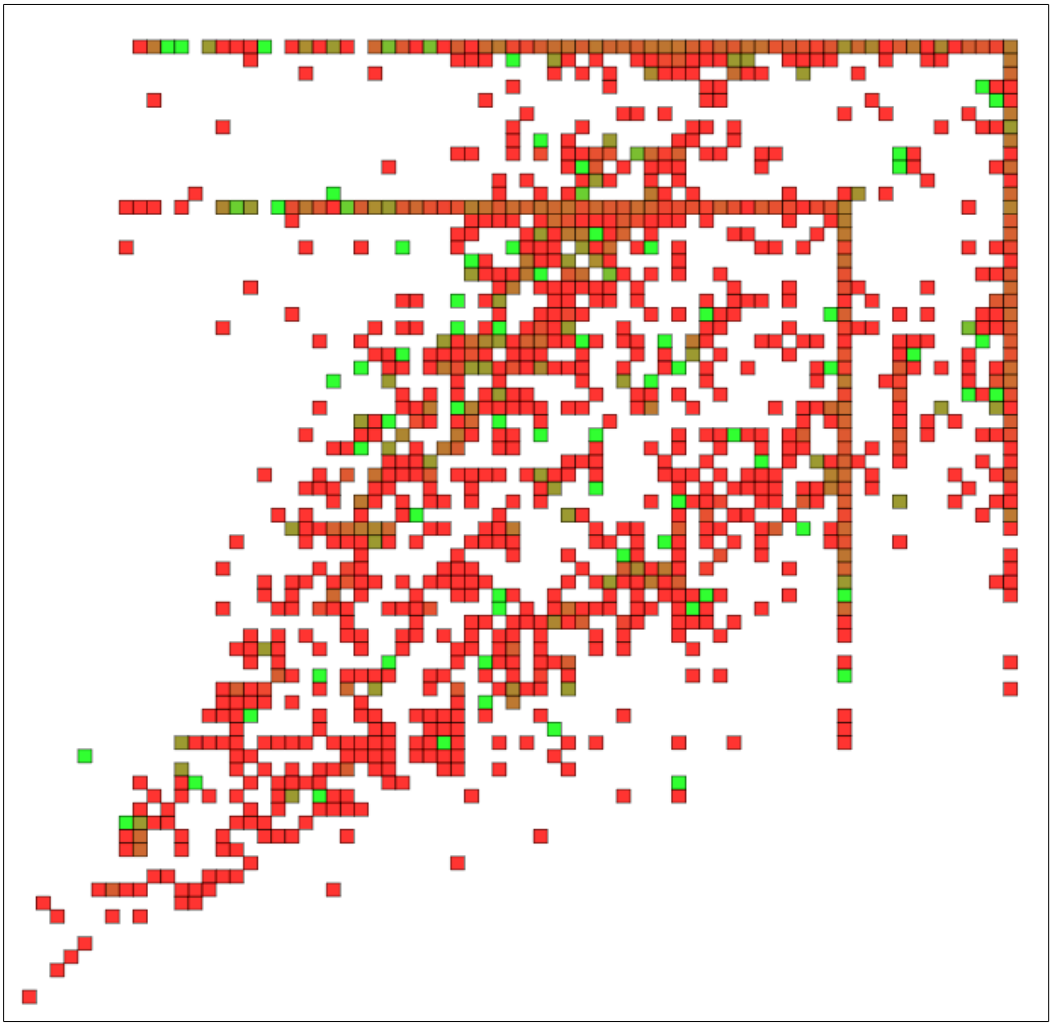
Taille des photos

Travail effectué

Pour avoir un aperçu de la répartition des photos en fonction de la taille nous avons changer le Layout des albums en fonction de “height” et “width”.



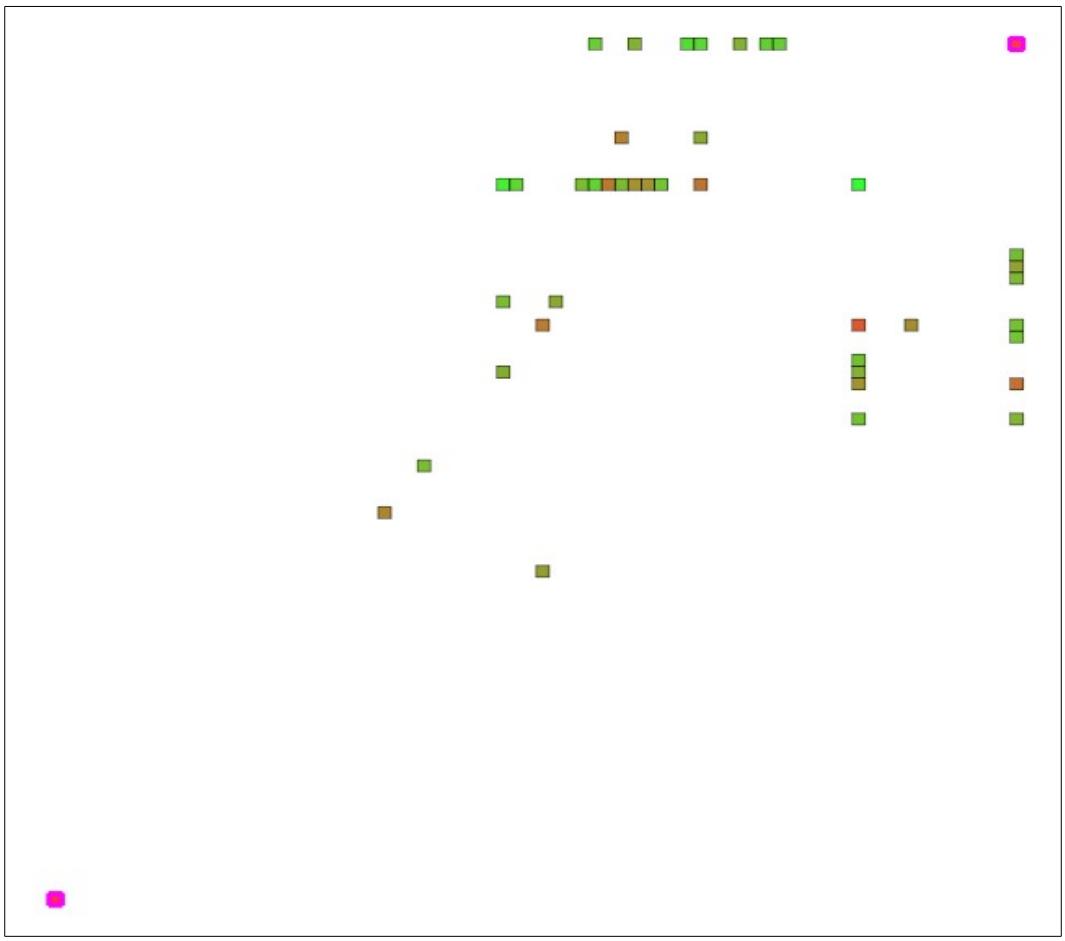
Nous pouvons voir efficacement comment se repartissent les albums. Cependant, comme pour la localisation, nous avons un problème de superposition des points. Nous avons donc appliqué l'algorithme de quadrillage.



analyse

Ce qui est étrange c'est que nous ne voyons pas de format panoramiques, format qui pourrait être souvent utilisé dans certaines régions (paysages étendus). On peut penser que ce phénomène vient du fait que nous n'avons là qu'une taille moyenne des photos de l'album.

Pour cette étude nous faisons aussi ressortir les zones du quadrillage avec une population assez forte.



Nous voyons là qu'aucune zone ne ressort particulièrement. On peut tout de même remarquer que les photo de taille 0*0 ne sont pas appréciées (ce qui parait logique).

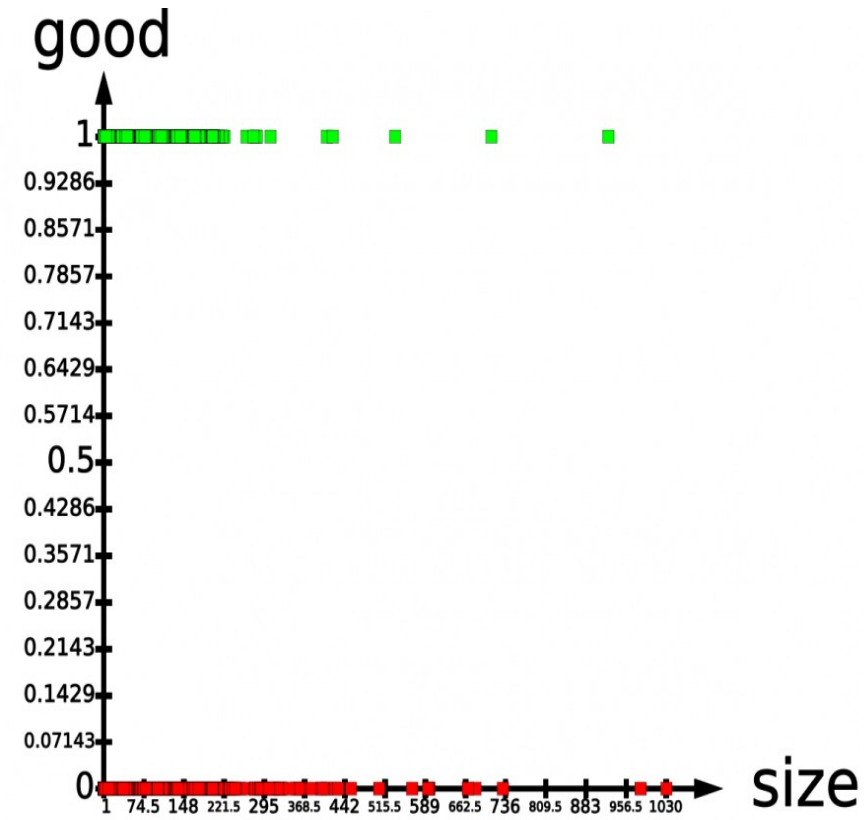
Nous ne pouvons déduire des informations utiles pour cette piste.

Cependant on se rends bien compte que quel que soit les autres méta données qu'il peut y avoir sur les photos de taille nulles, elles seront toujours mal notée, on a donc supprimé toutes les albums avec des photos de tailles nulles (ou quasi nulle) de notre graphe.

Taille des albums

Travail effectué

On a tous d'abord répartie les point en fonction de leur taille et de leur apréciation.



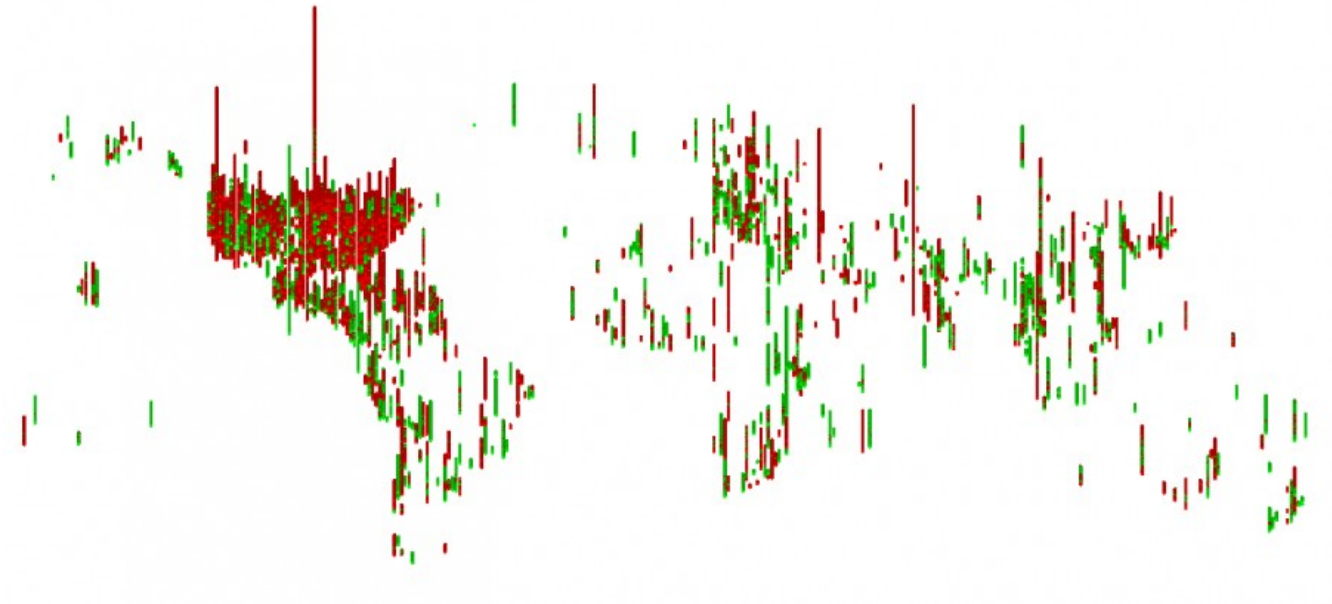
Cependant on ne voit pas grand chose, tout est trop sérré, on à donc chercher un moyen de chnager la visualisation pour avoir un meilleur aperçu de l'ensemble des points.

Hauteur de cylindre

Nous avons arrangé les points selon leurs coordonnées et nous leur avons donner une hauteur (sur la 3ième dimension) en fonctionne de la taille des albums.

fonction hauteur_cylindres() de Code/outils.py

Nous avons en rendu la visualisation suivante.



Même en essayant de joué avec la vue on ne voit rien, de plus, comme avec la localisation, les points se chevauchent.

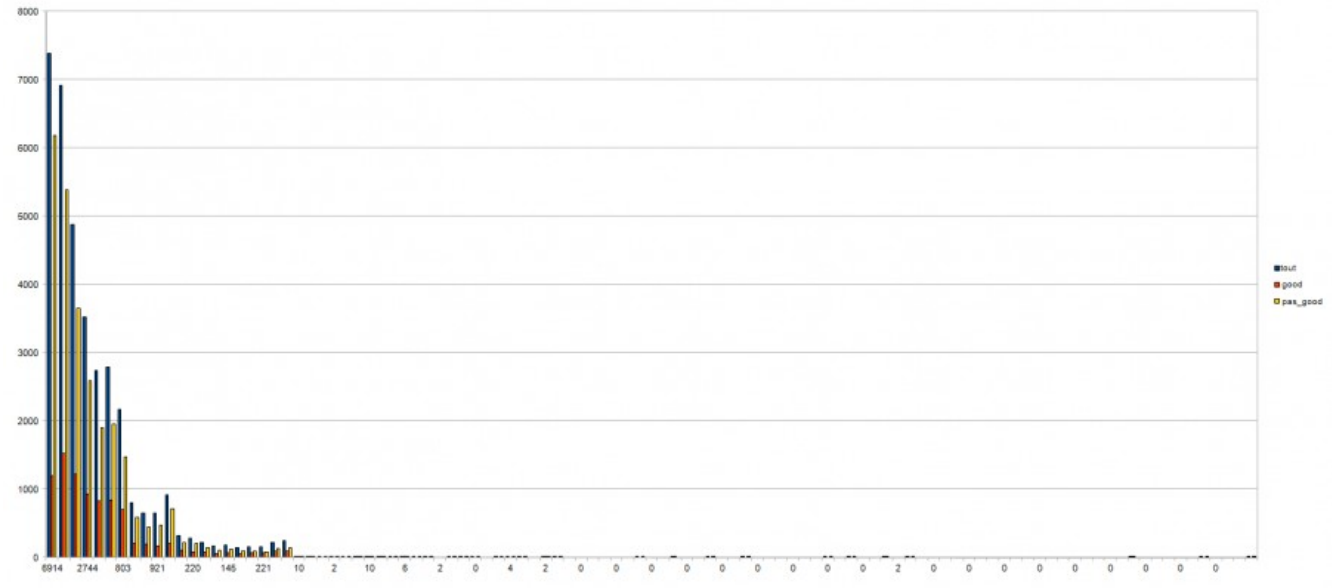
Histogramme

Nous nous sommes tourné vers une étude statistique de nos données.

Nous avons essayer de voir s'il y avait une répartition différentes des albums en fonction de leur taille si nous séparions les albums appréciés et les albums non appréciés.

fonction proportion_good_size() de Code/outils.py

Nous avons ensuite afficher les courbes sur un tableur.

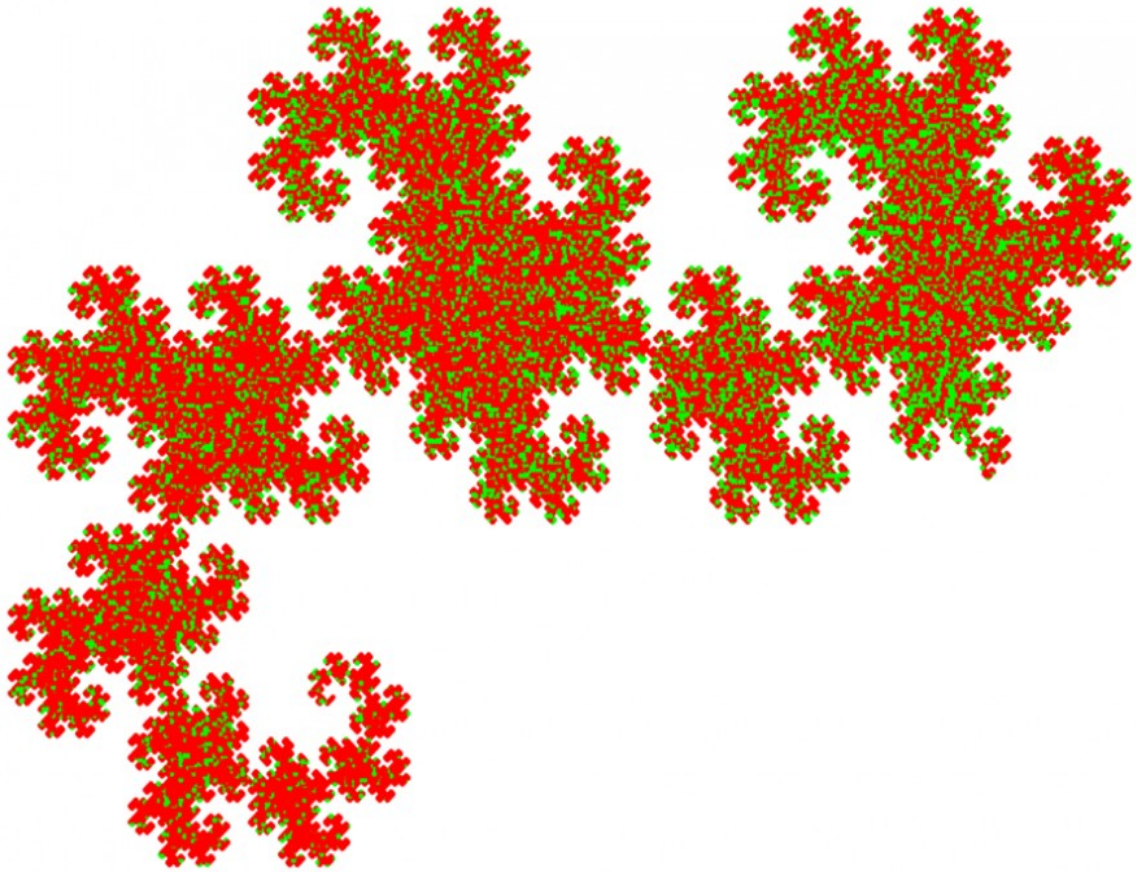


Les deux distributions nous ont l'air trop ressemblantes, on ne peut toujours pas tirer de conclusions.

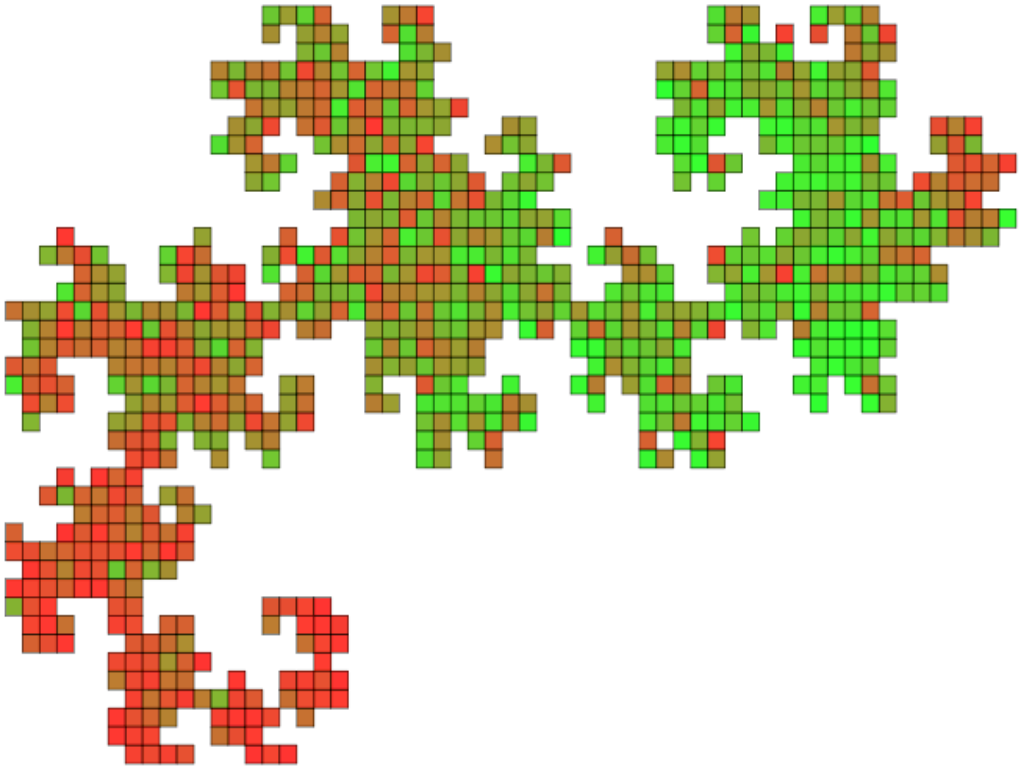
Arrangement orienté pixel

Dragon

On a essayé d'afficher les points avec l'algorithme de Dragon en fonction de leur taille. Et … nous avons obtenu quelque chose !



On a moyenné avec un un petit quadrillage pour voir avoir moins de zones à étudier et fait une coloration uniforme sur la moyenne d'appréciation de chaque carré.



Analyse

On voit qu'il y a une corrélation entre la taille et l'appréciation des album. L'appréciation moyenne des album suit la taille des album et chute brusquement quand les albums sont trop grands.

On peut noter que, même s'il y aune différence entre les petits et les grands albums, celle-ci reste faible.

On a calculé la proportion de good en fonction de la taille des albums. On a ensuite déterminé une fonction de puissance pour représenter ces données

ratio good = 0,140 * tailleDeLAlbum^0,180

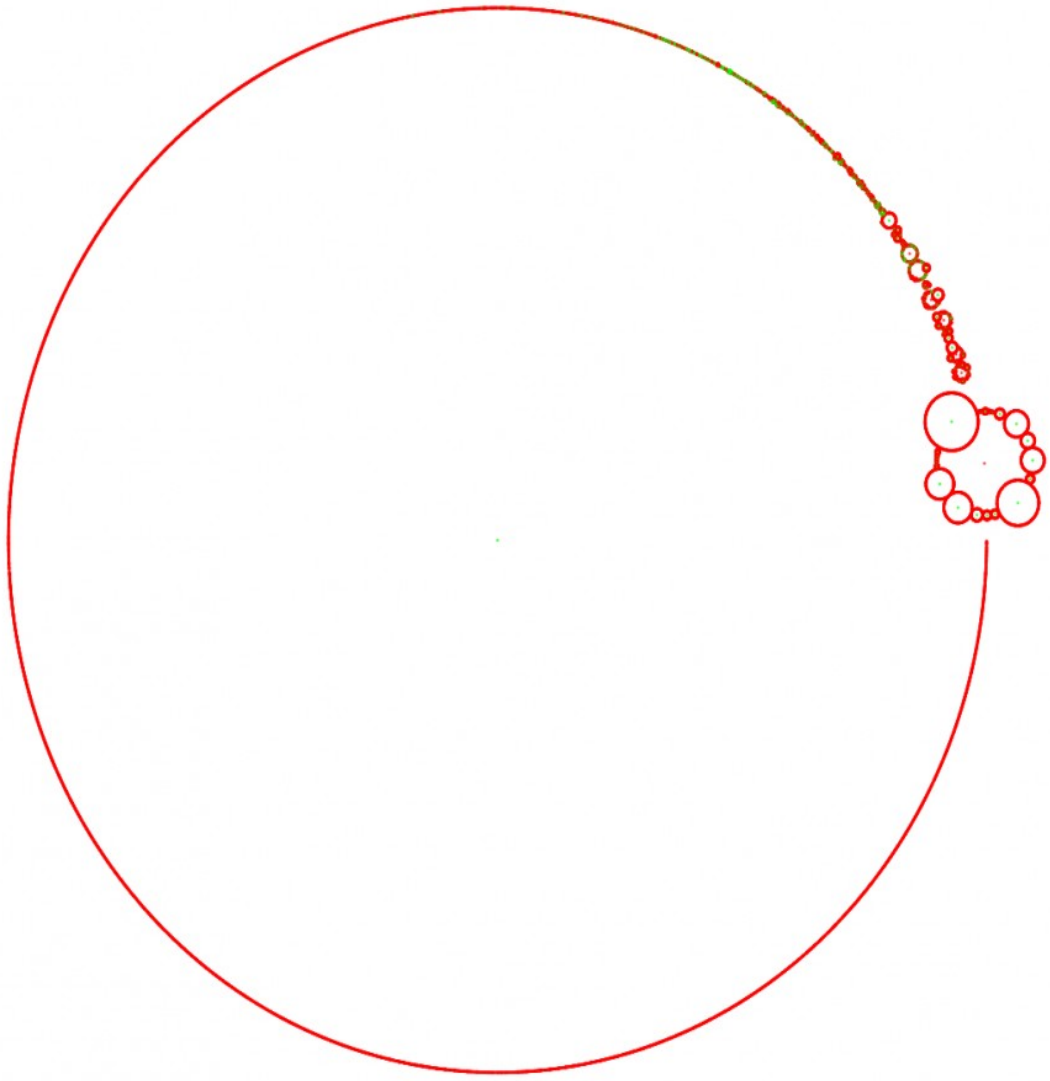
Atention Les résultats que nous avons ne sont pas flagrant, La taille n'a pas un fort impact sur la notation. De plus lors de l'étape de la notation il faudra faire attention car les ratio sont pour l'instant tous inférieur à 0,5.

Étude des textes

Pour l'étude des identifiants on a décidé de créer un graphe "mot/album", cela nous permet de voir les relation qu'il peut y avoir entre les différents mots utilisé et les albums photos.

Travail effectue

Pour le créer graphe mot/album, on a créer un noeud par mot et par album, puis on a créer une arrête pour chaque mot utilisé par un album. On obtient une graphique de 41 422 noeuds et 983 972 arrêtes.



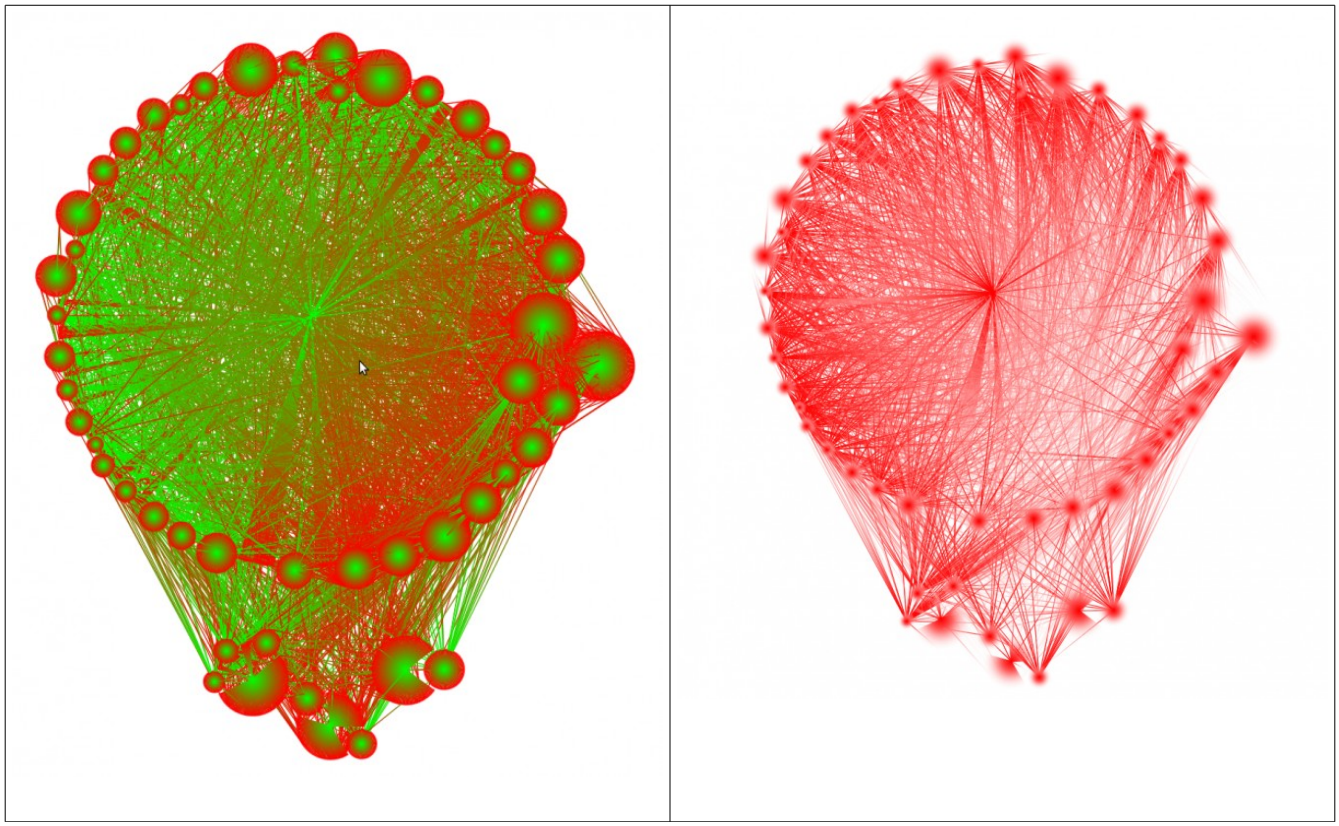
Ici nous avons utilisé un layout de bubble tree sinon tous les nœuds sont les uns sur les autres. Nous avons utilisé la liste de tous les mots. Les mots sont en vert et les albums en rouge.

On remarque sur cette image qu'il y a de nombreux albums avec peu de mots, et quelques mots ou albums avec de nombreuses relation (sur la droite de l'image).

On va essayer de se concentrer sur les mots avec de nombreux albums en relation, donc aux mots ayant un certains poids.

On va donc faire un seuillage sur le degré des mots, il est aussi important de voir la moyenne qu'on chaque mot en terme d’appréciation (moyenne sur les albums voisin).

Fichier Code/mots.py



Affichage avec un seuil de 100 pour le degré et {[0; 0,2] [0,8 : 1]} comme moyenne.

À gauche nous avons les mots en vert et les albums en rouge. À droite nous avons en rouge les mots avec une faible moyenne et en vert les mots avec une forte moyenne (ici il n'y en a pas) les album sont transparents.

On récupère ici 50 nœuds intéressants. On remarquera cependant que tous ces nœuds ont une moyenne négative.

Comme avec les autres donnée on remarque qu'on a qu'une très faible manœuvre de notation. on pu récupérer uniqument 50 noeuds sur les 2150 au total et ils sont tous négatif.

Il faudra donc jouer sur les différents seuils pour toucher un plus grand nombre de photos pour la notation des photos.

Analyse

Notation des photos

Maintenant que nous avons nos différentes pistes il va falloir essayer de les combiner pour pouvoir noter les albums.

Même si la taille des photos ne peux pas nous guider pour la notation, l'étude des tailles nous a fait remarqué qu'il y a des photos de taille nulles sont mal notés (cela parait évident mais on y aurait pas

forcement pensé).

La taille des albums nous donnes des résultats continus, cella va nous permettre de donner facilement une note pour la plus part des albums en fonction de leur taille. Cependant comme on a vu l'exploitation de cette piste n'es pas flagrant, on a une variation très faible de plus toutes les variation sont négatives.

On aura ensuite la location et l'étude des mots. ces deux pistes nous donnent des résultats discret. Comme on a vu si on s'occupe des critère ne donnant que des bon résultats on aura que très peu de photos à noté, en baissant les critères de notation on risque de mal noter les albums.

Fichier Code/notation.py

Test 1

Notation :

- Mettre toutes les notes en fonction de la taille de l'album (0,5 si hors interval connu). Les notes iront de 0,12 à 0,35
- Changer la note en fonction de sa localisation.
- Changer la note en fonction de ses mots
- Mettre à '0' tous album avec des photos de taille nulles

Pour la localisation et les mots on ajoute à la note courante : (moyenne d'appréciation pour le mot/ la localisation) - 0,5;

Résultats :

- Sans limite : On a que 8 albums marqué positif, on ne fait pas le test.
- Avec limites : On a que 108 albums marqué positif, on ne fait pas le test.

A cause de la taille de l'abum, on a un note trop faible dès le départ, ce qui pénalise tous les albums

Test 2

Notation :

- Mettre toutes les notes à 0,5.
- Changer la note en fonction de sa localisation.
- Changer la note en fonction de ses mots
- Mettre à '0' tous album avec des photos de taille nulles

Résultats :

- Sans limite : 66 bonnes photos; note : 0,49234 ; classement : 180

- Avec limites : 4684 bonnes photos; note : 0,26376 ; classement : 168

On a pour le moment un mauvais classement, on ne tiens pas compte de la taille de l'album, il faut voir si on n peux pas centrer les notes qu'il donnent autour de 0,5.

Test 3

Notation :

- Mettre toutes les notes en fonction de la size, en centrant ces note autour de 0,5.
- Changer la note en fonction de sa localisation.
- Changer la note en fonction de ses mots
- Mettre à '0' tous album avec des photos de taille nulles

Résultats :

- Avec limites : 3356 bonnes photos; note : 0,23187 ; classement : 134

Sur 213 participants.

Analyse

On a maintenant une bonne base pour travailler sur la notation des photos.

Pour améliorer notre score il faudra aller bidouiller différents paramètres essayer de les pondérer et voir ce que l'on obtient.

Traitement des données

Nous n'avons pas eu trop de difficultés pour traiter les données du problèmes. La plupart des données ont pu être traiter directement après voir été importé depuis tulip.

Cependant pour le traitement des mots nous avons fait quelques travaux préliminaires. Premièrement nous avons enlevé les données numérique ne nous intéressant pas. Ensuite nous avons du séparer les mots qui se trouvait dans des chaîne de caractère, nous les ajoutions ensuite dans des liste en fonction de s'ils viennent du nom de l'album, de la description de l'album ou des photos.

Fonction extraction_donnees_fichier() de Code/mots.py

Ensuite il nous fallait juste créer un graphe avec la liste choisie. et créer des arrêtes entre les mots et les albums (les arrêtes sont orientées album → mot).

Conclusion

Pistes suivies

Récapitulatif des pistes suivies :

Localisation

Résultats concluant mais sur un ensemble discret réduit.

Tailles des photos

Aucune piste utile.

Taille des albums

résultats continus sur la tailles des albums mais résultats avec peu d'impacts

Étude des mots

Résultats concluant mais sur un ensemble discret réduit.

Notation des photos

On a réussi à établir un algorithme de base pour noter les photos. Cependant il ne donne pas encore des résultats satisfaisant, pour améliorer la notation il faudra jouer sur différents facteurs.

Améliorations possibles

Données

- Tailles individuelles des photos
- Dates (lié a des évènements)
- Méta donné techniques (pour regarder si mauvaise balance des blanc, flou, …)
- Photographe
- Type de photos (portrait, paysage, etc.)

Techniques

- Travailler sur les différents paramètre pour peaufiner la notation.

Sources

- Site où nous avons récupéré la problématique et les données : http://www.kaggle.com/c/PhotoQualityPrediction