Documentation

Capocasale Romain

Freiburghaus Jonas

Moulin Vincent

Cours de visualisation

Semstesre d'automne

Le projet

Ce projet s'intéresse au vocabulaire des artistes de hip-hop français et vise à fournir des réponses aux questions suivantes :

- Est-ce que le vocabulaire de tel artiste est plus riche que celui d’un autre ?
- Est-ce que les mots les plus fréquemment utilisés sont vulgaires ?
- Est-ce que certains mots deviennent plus fréquents au fil des années ?

Les publics ciblés sont :

- **Prioritaire :** Personnes écoutant de la musique hip-hop française
- **Secondaire :** Personnes curieuses et intéressées par des statistiques diverses

Nous essayons de répondre à ces questions à l'aide des outils de visualisation suivants :

- **Graphique beeswarm :** Utile pour comparer des distributions tout en affichant des détails sur des éléments ayant des fréquences similaires sous form continue. Il est alors facile pour l'utilisateur de comparer la différence entre le vocabulaire de deux artistes à l'aide la distance qui les sépare.
- **Histogramme empilé :** Utile pour visualiser des distributions détaillées par des axes d'analyse sous forme discrète. Ce graphique présente la richesse de vocabulaire entre les artistes d'une manière différente de celle exploité ci-dessus. Cette option permet de regrouper les artistes dans des catégories en fonction de leur vocabulaire.

- **Nuage de mots :** Utile pour visualiser des mots-clés en faisant ressortir les éléments les plus importants
- **Graphique en ligne :** Utile pour visualiser une évolution au cours du temps
- **Histogramme :** Utile pour visualiser une évolution au cours du temps

Un utilisateur peut obtenir plus d'informations ou faire ressortir des éléments efficacement à l'aide d'outils d'interaction mis à disposition :

- Filtres
- Barres de recherches

Un soin particulier est apporté pour que ces informations soient compréhensibles et manipulables par un large public. Pour ce faire nous avons réalisé des :

- Tests d'utilisabilité : interfaces intuitives et agréables à utiliser
- Tests d'accessibilité : pour ce projet, s'assurer que les graphiques soient lisibles par des personnes atteintes de Daltonisme

Ainsi, nous espérons satisfaire la curiosité des utilisateurs à l'aide des ces différents graphiques et outils mis à disposition.

Implémentation

Les prochaines sections s'adressent à toutes les personnes qui s'intéressent aux détails de l'implémentation de cette application.

Jeu de données

Toutes les données de ce projet proviennent du site web Genius et ont été récupérées par le service API de ce même site.

Artistes

Le nom des artistes a été récupéré depuis la page Wikipédia répertoriant une liste de rappeurs français. Les noms des artistes sont récupérés via un script Python. Les chanteurs récupérés sont ensuite triés à la main pour assurer la cohérence des données par rapport aux besoins de l'API Genius. D'autres artistes non-présents sur la page wikipédia sont rajoutés manuellement à la liste.

Récupération des données

Les paroles des musiques sont récupérées et prétraitées par un script Python. Le téléchargement des données est effectué via la libraire LyricsGenius disponible sur GitHub.

Cette bibliothèque permet de récupérer les informations suivantes :

- Le nom de l'artiste
- Une photo de profile
- Une liste des titres de l'artiste, qui contient pour chaque chanson :
  - Le titre

    - L'année de sortie
    - Les paroles

Il est à noter que le genre et le type d'un artiste (indiduel ou groupe) sont spécifiés manuellement comme ces informations ne sont pas disponibles dans l'API Genius.

Toutes ces informations à propos des oeuvres des artistes sont récupérées par ordre de popularité avec une limite maximum de **50** oeuvres.

Prétraitement des données

Le nom de l'artiste et l'image de profile ne nécessitent pas de prétraitement.

Les paroles de chaque musique sont traitées suivant les étapes ci-dessous :

- Conversion du texte en minuscule
- Suppression des accents
- Suppression des caractères de ponctuation
- Suppression des caractères de mise en page tel que les retours à la ligne
- Suppression des en-têtes de paragraphes et des couplets
- Création de "token"

Pour chaque artiste, les mots utilisés sont groupés et comptés par année. Résultant en une structure de données similaire à l'exemple suivant :

```
      [{"year" : 2018,
       "words" : [{
                  "word" : "Artiste",
                  "count" : 10,
                  },
                  {
                    "word" : "Musique",
                    "count" : 20,
                  } ...]
      },
      {
        "year" : 2020,
        "words" : [...]
      }]
```

Cette représentation permet d'avoir une relation entre l'utilisation des mots par un artiste d'une année à l'autre. A noter qu'un même mot peut se trouver dans deux années différentes. Ce doublon permet de conserver une information temporelle au niveau de l'utilisation des mots.

Pour chaque artiste, les paroles de toutes les musiques sont mises bout à bout. Les **20'000** premiers mots sont sélectionnés. Sur ces **20'000**, le nombre de mot unique est compté. Cette information permet de comparer la richesse de vocabulaire des artistes. La limitation du nombre de mots a pour but de ne pas favoriser les artistes ayant plus d'oeuvres à leur actif.

Cependant, tous les artistes n'atteignent pas ce seuil de **20'000** mots. La richesse de vocabulaire de ces artistes est alors calculée avec tous les mots à disposition. Un champs est reservé pour indiquer si un artiste a atteint le seuil ou non.

En résumé, les informations suivantes sont disponibles pour chaque artiste :

- Le nom
- L'image de profile
- Le genre
- Le type de l'artiste (individuel ou groupe)
- Le nombre de mot pour chaque artiste (20'000 si le seuil atteint, une autre valeur dans le cas contraires)
- Le nombre de mot unique
- Un "flag" qui indique si l'artiste a atteint le seuil des 20'000 mots
- Le nombre de musique anylsées (maximum 50)
- Le titre des musiques anylsées (maximum 50)
- Les années durant lesquelles l'artiste a publié des musiques
- L'occurence de chaque mot utilisé, groupés par année

Base de données

Les données de chaque artistes sont enregistrées sous forme de document dans une collection. Une base de données orientée documents Mongo DB est utilisée à cet effet.

Serveur API

Le serveur API a été réalisé avec Node.js. Il permet de servir les informations de la base de données dans le format JSON. L'ORM Mongoose a été utilisé pour faire le lien entre la base de données et les objets Node JS. De plus, cette librairie permet de simplifier la création de requête MangoDB.

Les routes API ont pour racine l'URL suivante https://www.zumbacafew.ch/api et permettent d'obtenir les informations ci-dessous :

- `api/artists/` : Liste de tous les artistes
- `api/artists/artistcount` : Nombre d'artiste
- `api/artists/maxyear` : Année de sortie de l'oeuvre analysée la plus ancienne
- `api/artists/minyear` : Année de sortie de l'oeuvre analysée la plus récente
- `api/artists/soundcount` : Nombre de musique analysées
- `api/artists/stats` : Retourne un ensemble de statistiques sur chaque artiste
- `api/artists/termfrequency` : Dictionnaire de la fréquence d'utilisation des mots
- `api/artists/termfrequency/:artistName` : Dictionnaire de la fréquence d'utilisation des mots pour un artiste donné
- `api/artists/termfrequencyByYear/:word` : Utilisation par année du mot passé en paramètre
- `api/artists/terms` : ensemble des mots utilisés
- `api/artists/wordcount` : Nombre de mot analysés

Pour les statistiques utilisant la fréquence d'appartion des mots, un filtre est utilisé pour retirer les mots vides. Un mot vide est un mot très commun dans une langue qui généralement n'apporte pas d'information pour la tâche réalisée. En effet, il ne serait pas très intéressant de visualiser que le mot le plus utilisé par les artistes soit par exemple un déterminant ou un pronom.

Frontend

Le framework frontend utilisé pour réaliser cette application est Vue.js. Les données sont récupérées par le client via des requêtes à l'API du serveur détaillée au précédent paragraphe. Le package Axios est utilisé pour réaliser ces requêtes.

Les graphiques du nuage d'artiste et du nuage de mot sont réalisés à l'aide de la bibliothèque D3.js. L'histogramme d'artiste est réalisé en Javascript Vanilla sans aucune librairie. Le linechart et l'histogramme ont eux été réalisés avec la bibliothèque Javascript Chart.js. Pour plus de détail sur la réalisation de ces graphiques, veuillez consulter la section appropriée sous chacune des visualisations en question.

Déploiement

Les diverses applications formant le projet sont déployables sous forme de "container" Docker. Les 3 "containers" sont dédiés à :

- L'application frontend
- L'application backend
- La base de données

Ces "containers" sont déployés sur une infrastructure IaaS d'un hébergeur cloud.

Conclusion

Pour conclure, on peut dire que les différentes visualisations mises en place pour ce projet permettent de répondre le plus efficacement possible aux questions posées tout en s'adressant au maximum au publique cible.

Le graphique beeswarm et l'histogramme d'artiste permettent de répondre chacun à leur manière à la question de la richesse du vocabulaire des artistes. Ces deux solutions permettent de comparer efficacement le vocabulaire entre deux artistes ou une catégorie d'artiste (genre, année, etc). La première représentation permet une comparaison du vocabulaire des artistes à partir de la distance qui les sépare sur l'axe x. La seconde permet de les comparer en fonction de la catégorie de l'histogramme à laquel ils appartiennent.

Le nuage de mots permet de répondre à la question de la fréquence des mots en général dans la musique francaise ou pour un artiste en particulier. La représentation permet de comparer efficacement l'utilisation d'un mot par rapport à un autre en fonction de la taille qu'il occupe sur le graphique. Ainsi nous parvenons à faire ressortir les sujets principales traitaient par les artistes.

Les deux derniers graphiques, le linechart et l'histogramme, permettent eux de répondre à la question de l'évolution de l'utilisation d'un mot en fonction du temps. Ils permettent de visualiser l'évolution de l'utilisation d'un mot au cours des années avec des graphiques adaptés aux séries temporelles.

Pour finir, on peut dire que les tests d'utilisabilités et d'accessibilités mis en place nous permettent d'optimiser l'ergonomie de l'application pour les utilisateurs.