Groupe 9 de dataviz : APB 2016-2017

Jalons

1	- Création d'un projet gitlab de groupe et d'un README - Import sur Visual Studio Code du fichier .csv d'APB provenant de data.gouv
2	- Concertation sur la répartition des fichiers, les fonctions à écrire pour manipuler le jeu de données et répartition de ces fonctions entre nous
3	- Ecriture des fonctions dans les fichiers data_utils.py, data_statistics.py, data_analysis.py, et visualisation sur matplotlib - Création du fichier dataviz_main_command_line.py qui exécute le programme
4	- Visualisation des fonctions sur Dash pour un meilleur rendu : création du fichier data_analysis_dash.py - Mise en place d'une interface graphique Dash : création du fichier app.py qui exécute le programme
5	- Création de deux cartes de France colorées en fonction des statistiques choisies
6	- Mise en place d'une interface dash user-friendly avec menus déroulants,, zones textuelles et descriptions

…

Idées de manipulation de données :

- mesure d'attractivité des filières/établissements grâce au vœu n°1 et au rang du dernier admis (et au nombre de mentions bien et très bien ?), indice = nb_vœu_n°1/nb_vœux_tot - (rang_dernier_admis – nb_admis)/nb_voeux_tot
- carte des départements colorées selon un critère (la filière la plus suivie, le nombre d'étudiants…) : FAIRE A LA FIN
- selon des critères (sexe, région, notes), indiquer où tu as le plus de chance de finir, en comparaison avec les données (peut être compliqué parce qu'on a pas les infos de chaque candidat) : besoin de ML : DIFFICILE
- proportion de filières en France et/ou par région
- filières qui recrutent le plus
- filières où le taux de boursiers est le plus élevé
- filières où le taux de filles est le + élevé
- évolution d'une année à l'autre
- histogramme des filières avec le plus de places disponibles
- camembert du pourcentage des différentes mentions au bac des élèves par établissement (tel lycée a x% de mention TB, y% de mention B, z% de mention AB,…)
- filière la plus attractive pour chaque lycée
- Mettre garde-fou qui demande une info en plus si on demande des infos sur un lycée et que plusieurs ont le même nom

Les différents fichiers python du projet ?

- 1er fichier : data_utils.py, qui load et qui traite le dataframe et fait qq modifications de base dessus (par exemple le nom des colonnes, ou qq modifications de données comme changer le “72.0” en “72” pour les départements, et séparer en 2016/2017)
- 2eme fichier : data_statistics.py, qui fait qq manipulations intéressantes sur les données, pour faire qq analyses plus poussées sur les données chargées par data_utils.py (exemple de manipulation intéressante : renvoyer pour une région en particulier, la liste des établissements)
- 3eme fichier : data_viz.py (ou data_analysis.py), qui utilise les analyses de data_statistics.py pour afficher les données sous forme de graphiques
- 4eme fichier : main.py (qui exécute tout)

Que mettre dans les fichiers ?

1) data_utils.py

Manipulations à effectuer sur la database :

- Mettre un entier pour le département
- Renommer certaines colonnes :
- Séparer le dataframe en deux (2016 et 2017)

Faudrait écrire une fonction **load_dataframe(annee)** pour pouvoir charger le dataframe dans les autres fichiers

Y a des données inconnues parfois (par exemple capacité d’un établissement inconnue), faudrait régler ça aussi - > pour l’instant tout les établissements ayant une donnée quelconque inconnue sont virés

2) data_statistics.py

Idées de fonctions :

- Une fonction simple **liste_filieres(dataframe),** qui liste toutes les filières du dataframe, et pareil pour **liste_regions**, **liste_departements**, et les sous-filières par filière l**iste_sous_filiere(dataframe,filiere)**

- Une fonction **separation_dataframe_par_region(dataframe, region)**, qui ne garde les lignes du dataframe (sans modifier les colonnes) que pour une région particulière, et pareil par département

  (Idée derrière ça : pour les statistiques à l'échelle nationale, on peut travailler sur le dataframe entier, et quand on voudra des statistiques par région par exemple, il suffira de séparer le dataframe par région grâce à cette fonction)

Fonctions de réarrangement du dataframe sous forme de dictionnaire :

- Une fonction **classification_par_filiere(dataframe)**, qui classe les écoles par filière

  { filiere1 : { lycée1 : {capacité : , rang_dernier_admis: , nombre_mentions_tb: , ... },

  lycée2 : { … },

  },

  filiere2 : { lycée1 : { … } ,

  lycée2 : { … },

  },

  }

  Cette fonction pourrait ensuite servir à calculer la capacité totale pour une filière donnée

- Une fonction **capacite_par_filiere(dataframe)**, qui pour chaque filière donne la capacité totale, grâce à la fonction précédente classification_par_filiere. Elle retourne alors qqch de la forme :

  capacite = { filiere1 : {capacité: 20 000, proportion: 0.4},

  filiere2 : {capacité: 13 000, proportion: 0.25 },

  filiere3 : …

  }

- Une fonction **demande_par_filiere(dataframe)**, qui pour chaque filière donne la demande totale (demande du 1er vœu), grâce à la fonction classification_par_filiere. Elle retourne alors :

  demande_totale = { filiere1 : {demande: 30 000, proportion_demande: 0.6, nb_filles: 1200},

  filiere2 : {demande: 10 000, proportion_demande: 0.2, nb_filles: 4500},

  filiere3 : …

  }

Les mêmes fonctions, qui marchent exactement de la même manière, mais pour les sous-filières et les filieres fines:

- **classification_par_sous_filiere (dataframe, liste_filiere)**
- **classification_par_filiere_fine(dataframe, liste_filiere)**

- **capacite_par_filiere_fines(dataframe, liste_filiere)**
- **capacite_par_sous_filiere (dataframe, liste_filiere)**

- **demande_voeu1_par_sous_filiere(dataframe, liste_filiere)**

3) data_analysis.py

On voudrait avoir :

- ~~Un histogramme (ou camembert) des filières avec le plus de places disponibles~~ (Aurélien)
- ~~Un histogramme (ou camembert) des filières les plus demandées (en vœu 1)~~ (Greg)
- ~~Un histogramme représentant pour chaque filière, le nombre d'élèves qui n'ont pas eu cette filière alors qu'ils la voulaient (grâce au nb de demandes, nb d'eleves qui ont reçu une proposition et nb qui l'ont accepté) (Vish) et pareil en filières fines~~
- ~~Un camembert pour la proportion des différentes filières~~ (Aurélien)
- ~~Un histogramme pour les filières où le taux de filles (ou boursiers par ex) est le plus élevé (Aurélien) --> la même chose que l'histogramme suivant en fait~~
- ~~Un histogramme montrant l'évolution du taux de filles dans les filières entre 2016 et 2017 (Yanis)~~
- ~~Pour un lycée en particulier, filière la plus attractive (Evan)~~

- ~~Un histogramme des filières les plus attractives (mais cela nécessite de déterminer un indicateur d'attractivité) (NB : ça semble à peu près équivalent au 2eme point, l'histogramme des filières les plus demandées) (Evan)~~
- ~~Histogramme des filières qui atteignent leur capacité max: acceptations/capacité (Aurélien) -> fait dans refus_filiere~~
- ~~Histogramme filières qui engendrent le plus de "frustration" ie où il y a beaucoup d'élèves qui n'ont pas leur premier vœu dans cette filière mais qui ont la filière quand même (Vishnou)~~
- ~~camembert du pourcentage des différentes mentions au bac des élèves par établissement (tel lycée a x% de mention TB, y% de mention B, z% de mention AB,…) (Aurélien)~~

Problèmes avec matplot:

- ☹ Pb en général avec les histo: il faudrait ajouter les valeurs en haut des barres
- ☹ Pb avec les sous filieres (MPSI etc): trop de données sur l'axe des abscisses au sein d'une filiere => illisible (solution: partitionner ss_filiere et filiere_fine?)

On peut travailler ces fonctions du point de vue national (où pour une filière, on considère toutes les écoles qui la proposent en France), et on pourra aussi les proposer du point de vue régional (ce qui sera très simple car il suffit de restreindre le dataframe à une région, grâce à separation_dataframe_par_region de data_statistics.py

On peut comparer aussi les résultats pour 2016 et 2017 (par exemple évolution du nombre de filles)

Pour comparer les régions entre elles, on pourra proposer :

- Carte des départements colorées selon un critère (la filière la plus suivie par exemple)
- Carte des départements où pour une filière donnée, plus un département serait coloré en rouge, plus la proportion de demande (ou la proportion de capacité) pour cette filière dans ce département est important

4) data_analysis_dash.py

On va adapter les fonctions précédentes pour dash: on renvoie à chaque fois la figure qu'on aura juste à tracer dans le main

Fonctions à transformer pour dash : (on reprend le meme nom de fonction et on rajoute dash_ devant)

NB : ces fonctions renvoient fig

- ~~camembert_filieres_places_disponibles(df) (Aurélien)~~
- ~~camembert_proportion_etablissement_filieres(df) (Aurélien)~~
- ~~histogramme_taux_critere_par_filiere(df2016, df2017, critere='fille') **(Yanis)**~~
- ~~histogramme_taux_critere_par_sous_filiere(df2016, df2017, liste_filieres, critere='fille') **(Yanis)**~~
- ~~histo_plus_demande(df)~~
- ~~histogramme_refus_filiere(df) (vish)~~
- ~~histogramme_refus_sous_filiere(df, l_filieres) (vish)~~

- ~~histogramme_refus_filiere_fine(df, l_filieres) (vish)~~
- ~~camembert_mention_par_etablissement_par_filiere(df, etablissement, filiere) (Aurélien)~~
- ~~histo_plus_demande_annees(df2016, df2017)~~ (Greg)
- ~~histogramme_frustration_filieres(df) (vish)~~
- comparaison_filles_admises_parfiliere(df2016, df2017)(Evan)
- ~~viz_attractivite(filiere, df, n) (Evan)~~
- ~~attractivite_etablissement(etablissement, df, n, type_cat)(Evan)~~
- ~~Histogramme_excellence_academique(df, filiere, n) (Aurélien)~~

5) data_analysis_dash_desc.py

Les descriptions de chaque graph sont dans ce programme qu'on utilise ensuite pour les afficher.

Recap des fonctions dont on dispose pour dash

Pour le taux d'un critère (par exemple le taux de filles ou de boursiers)

- Pour avoir les taux sur une seule année :

  dash_histogramme_taux_critere_par_filiere(df, critere='fille')

  dash_histogramme_taux_critere_par_filiere_fine(df, liste_filieres, critere='fille')

  dash_histogramme_taux_critere_par_sous_filiere(df, liste_filieres, critere='fille')

- Pour avoir l'évolution des taux sur 2016 et 2017 :

  dash_evolution_histogramme_taux_critere_par_filiere(df2016, df2017, critere='fille')

  dash_evolution_histogramme_taux_critere_par_filiere_fine(df2016, df2017, liste_filieres, critere='fille')

  dash_evolution_histogramme_taux_critere_par_sous_filiere(df2016, df2017, liste_filieres, critere='fille')

Pour le nombre de candidatures, de propositions et d'affectations pour chaque filière :

- dash_histogramme_refus_filiere(df)

  dash_histogramme_refus_filiere_fine(df,l_filieres)

  dash_histogramme_refus_sous_filiere(df,l_filieres)

Pour la répartition des vœux de rang 1 selon les filières

- Dash_histo_plus_demande(df)
- Dash_histo_plus_demande_annee(df2016,df2017)

Pour le nombre de places disponibles pour chaque filière :

dash_camembert_filieres_places_disponibles(df)

Pour le nombre d'établissement proposant chaque filière :

dash_camembert_proportion_etablissement_filieres(df)

Pour la répartition des mentions des élèves admis pour une filière donnée et un établissement donné :

dash_camembert_mention_par_etablissement_par_filiere2(df, etablissement, filiere)

Pour le nuage du point classant les établissements les plus attractifs d'une filière, sous-filière ou filière fine ou les filieres/sous-filieres/filieres fines les plus attractives d'un établissement:

dash_viz_attractivite (filiere, df, n)

dash_attractivite_etablissement(etablissement, df, n, type_cat)

Pour la proportion d'étudiants allant dans une filière mais dans un vœu qui n'est pas leur premier choix :

dash_histogramme_frustration_filieres(df)

Pour le classement des établissements selon la proportion d'admis avec la mention très bien, pour une filière donnée :

dash_histogramme_excellence_academique(df, filiere, n)