LIFPROJET

RAPPORT DU PROJET

Sujet RC3 :

Kaggle Challenge

AUTEURS

Lina ISMAIL 11802713

Simon KLOPFENSTEIN 11803372

Judith MILLET 11811933

ENCADRANT

Remy CAZABET

2020-2021

Lyon 1

kaggle

SOMMAIRE :

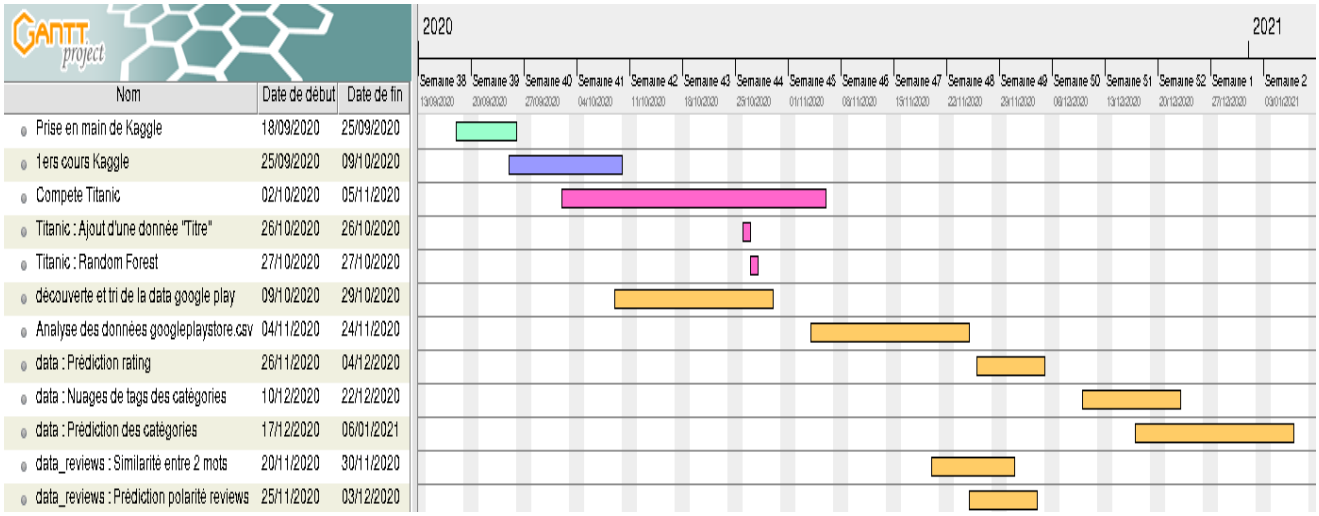
Dans le cadre de l'UE **LIFPROJET**, nous avons décidé de nous initier au **Machine Learning**. Pour ce faire nous avons **suivi les cours**, ainsi qu'un **tutoriel** fourni par le site de challenges en science des données **Kaggle**, afin **d'acquérir les bases** nécessaires pour être plus autonome. Par la suite, nous avons décidé **d'étudier un des jeux de données** fourni également par Kaggle.

A : DESCRIPTION DE L'ORGANISATION

DESCRIPTION

Notre travail a été réalisé sur **Google Colab**, un service gratuit de Google qui nous a permis de créer des modèles de Machine Learning directement dans le cloud. Nous avons donc effectué nos différents projets sur ces **notebooks**, et partagé nos données avec **Google Drive**. Ainsi, chacun a pu travailler sur une partie dans une copie d'un notebook commun (analyse, prédiction, etc) et nous faisions une mise en commun sur ce notebook commun. Comme moyens de communication, nous utilisions **discord** (partage d'écran, etc) entre nous, et WebEx avec notre encadrant.

DIAGRAMME DE GANTT



Nom	Date de début	Date de fin
Prise en main de Kaggle	18/09/2020	25/09/2020
1ers cours Kaggle	25/09/2020	09/10/2020
Compete Titanic	02/10/2020	05/11/2020
Titanic : Ajout d'une donnée "Titre"	26/10/2020	26/10/2020
Titanic : Random Forest	27/10/2020	27/10/2020
découverte et tri de la data google play	09/10/2020	29/10/2020
Analyse des données googleplaystore.csv	04/11/2020	24/11/2020
data : Prédiction rating	26/11/2020	04/12/2020
data : Nuages de tags des catégories	10/12/2020	22/12/2020
data : Prédiction des catégories	17/12/2020	06/01/2021
data_reviews : Similarité entre 2 mots	20/11/2020	30/11/2020
data_reviews : Prédiction polarité reviews	25/11/2020	03/12/2020

Figure 1 : Diagramme de Gantt du projet JUSINA.

B : DEROULEMENT DU PROJET

NOS PREMIER PAS SUR KAGGLE

Tout d'abord, nous avons découvert en profondeur le site de **Kaggle**[1] et les **notebooks** qui étaient pour nous un nouvel environnement de développement. Après s'être familiarisé avec ceux-ci, nous nous sommes entraînés, grâce aux cours de la catégorie "Courses", à appréhender et comprendre le **Machine Learning** et la **Data Visualisation**. Nous avons suivi principalement les cours "Python", "Intro to Machine Learning" et "Data Visualization". Nous avons également étudié les **documentations des libraires** utilisées (Matplotlib[2], Seaborn[3], Pandas[4] …)

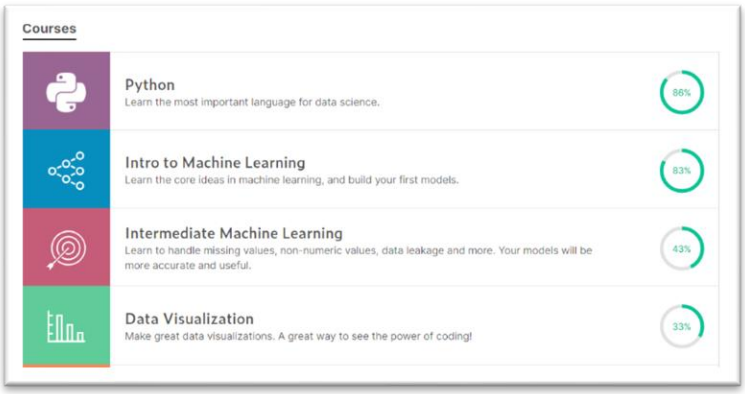


Figure 2 : Menu des cours Kaggle et leurs avancements.

LE TITANIC

Par la suite, nous avons travaillé sur la base de données de la **compétition** "**Titanic** : Machine Learning from Disaster". Le but de cette compétition d'apprentissage était de **prédire quels passagers avaient survécu au naufrage du Titanic**. Pour répondre à cette problématique nous avons commencé par réfléchir aux différentes **caractéristiques** qui pourraient **influencer la survie** ou non des passagers présents sur le bateau, nous les avons ensuite **analysées**. Ces données sont les suivantes :

- **SibSp** (frères, sœurs ou conjoints présents sur le bateau)
- **Parch** (parents ou enfants présents sur le bateau)
- **Sex** (le genre de la personne)
- **Cabin** (dans quelle cabine, entre A et G, se trouvait la personne)
- **Age** (son âge)
- **Title** (son titre comme par exemple : Mrs, Mr, etc) que nous avons extrait des noms pour en faire une catégorie
- **Embarked** (à quel port la personne a embarqué).

Afin de tester notre prédiction, nous avons utilisé une **Random Forest**[5].

Les caractéristiques influentes que nous avons pris en compte dans celle-ci sont les titres principaux (‘Master’, ‘Mr’, ‘Mrs’, ‘Miss’), la classe, le port d'embarquement (C, Q, S) ainsi que le sexe.

	Master	Mr	Mrs	Miss	Pclass	NoCabin	Embarked_C	Embarked_Q	Embarked_S	Sex_female	Sex_male
0	0.023584	0.263432	0.039272	0.028633	0.157889	0.08239	0.020643	0.015687	0.021114	0.200677	0.14668

Figure 3 : Features de la Random Forest et leurs taux d’importance.

Cette compétition nous a appris à **coder** en python, à **analyser** les données d’une data et à faire des **Random Forests**. Elle nous a aussi appris à gérer les problèmes très communs de **manques de données** dans une dataset.

LE GOOGLE PLAYSTORE

Pour continuer, nous avons choisi de travailler sur la data “Google Play Store Apps” où nous avons pu **développer nos connaissances en termes de Machine Learning**. Elle représente les applications du Google Play Store en 2018, ainsi que différentes caractéristiques pour chacune d’elles. Cette data comporte **deux jeux de données**.

1\ La première table a pour chaque application, les caractéristiques suivantes :

- Son nom,
- Sa catégorie,
- Sa note sur 5,
- Son nombre de commentaires.
- Son nombre d’installations,
- Sa taille,
- Son prix,
- Son genre,
- Sa version.

Nous avons eu comme premier objectif avec ces données de comprendre ce qui pourrait **influer** sur **la popularité d’une application**. Grâce à nos conclusions nous avons essayé de **prédire la note d’une** application.

Dans un second temps, nous avons essayé de **prédire une catégorie d’une application en fonction de son nom**. Pour cette dernière prédiction, nous nous sommes appuyés sur deux nouveaux jeux de données que nous avons créés : **data_féquence** (importance caractéristique des mots de chaque catégorie) et **data_category** (mots avec leur catégorie assimilée et leurs vecteurs).

2\ La deuxième table a pour chaque application, les caractéristiques suivantes :

- Les commentaires d'utilisateurs de cette application traduits en anglais,
- Le nom de l'application,
- Sa positivité ou non,
- Son score de polarité,
- Son score de subjectivité.

L'objectif ici était de retrouver la **polarité d'un commentaire** de la table grâce à une prédiction en s'appuyant sur les mots.

Nous avons alors utilisé le **word embedding**[6], une liste de mots positifs et une liste de mots négatifs. Après avoir compris le fonctionnement de l'embedding grâce à l'implémentation des différentes fonctions, on a calculé **la positivité** d'un mot de deux manières : avec une fonction et une Random Forest. Puis nous avons implémenté plusieurs façons de calculer la positivité d'un commentaire en s'appuyant sur les mots qui le composent.

C : LES OBSTACLES

L'IDE

Tout d'abord, nous avons essayé d'utiliser le **notebook fourni par Kaggle**. Ce dernier nous semblait le plus approprié car il évitait les problèmes de surchauffe de nos PC. Malheureusement, lorsque nous avons commencé à travailler tous sur le même notebook, nous avons alors rencontré quelques problèmes de version qui nous ont contraints à changer d'IDE.

Comme dit précédemment, la solution fut d'utiliser **Google Colab**[7], une alternative qui, en plus de nous donner une meilleure gestion des versions, nous permettait de mieux partager les données via le google drive, mais qui malheureusement ne nous permettait pas de travailler sur un notebook commun. Pour pallier ce problème, nous avons effectué une copie du notebook sur nos drives respectifs afin de travailler séparément sur différentes parties, puis de les transcrire sur un notebook commun.

LE CHOIX DES LIBRAIRIES

Les premières librairies qui nous sont présentées par Kaggle sont **limitées**, nous avons donc dû **trouver** et **comprendre** de nouvelles librairies pour choisir les **librairies adéquates**[8] (plotly, numpy, Ipython.display, collection.counter, WordCloud, scikit-learn) à un problème.

Exemple : nuages de tags => WordCloud, etc.

TEMPS D'EXECUTION

Lorsqu'une base de données est très grande, certaines manipulations/fonctions prennent **plusieurs minutes, heures à s'exécuter**.

GOOGLE PLAYSTORE

Pour la data du Google Play Store, les applications du premier jeu de données étaient **différentes** de celles du deuxième jeu de données. Donc il était impossible de lier ces deux jeux. Outre cela, nous avons eu des difficultés à **choisir la bonne méthode d'analyse** des commentaires, nous n'avons pas suivi de cours pour ce type de données (texte).

LES ECHECS

La prédiction des ratings était impossible, en effet ces dernières n'étaient pas suffisamment réparties. Les données tournaient trop autour de la note 4.5. Il était donc impossible de prédire ce rating avec les modèles que nous avions choisis (linear regression[9], Random Forest, Decision Tree[10])

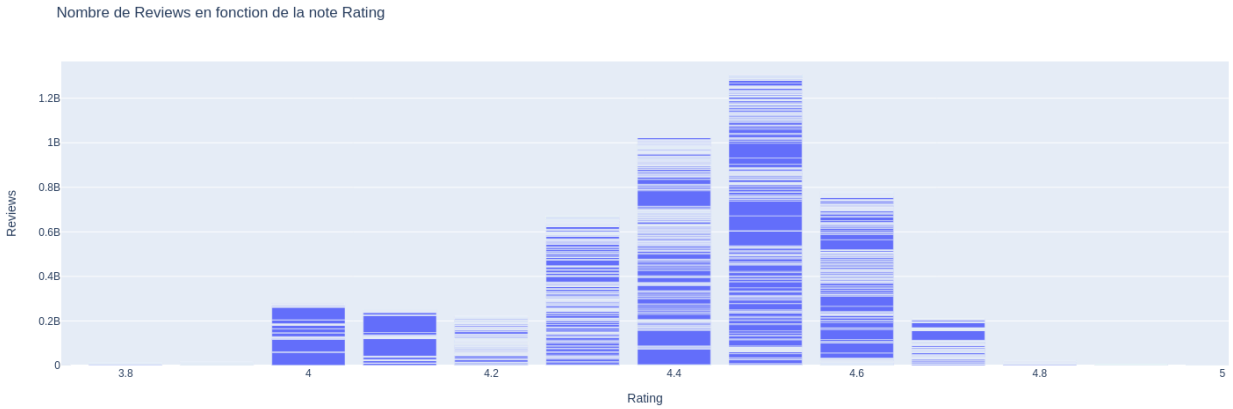


Figure 4 : Histogramme du nombre de « Reviews » en fonction de la note « Rating ».

DIVERS

- Compréhension des paramètres des decisions tree
- Gestion des sorties du notebook vers le drive
- La partie nettoyage des données a pris un certain temps pour le google PlayStore

(Exemple : La colonne taille d'une application n'avait pas une unité de mesure principale donc on a dû convertir dans la même unité)

D : CONCLUSION

Pour conclure, ce projet fut très constructif, il nous a permis de **découvrir** le Machine Learning par le biais de Kaggle qui fournit un nombre important de données à **étudier**, mais aussi grâce à un bon encadrement. De ce fait, notre intérêt pour le projet ne fut qu'amplifier, fortifiant notre **cohésion d'équipe** malgré les conditions difficiles actuelles. Nous aurions aimé **explorer** un peu plus en profondeur les réseaux de neurones qui permettent l'utilisation 'plus appropriée' des vecteurs des mots.

E : RESSOURCES

Kaggle: https://www.kaggle.com/

Matplotlib: https://matplotlib.org/

Seaborn: http://seaborn.pydata.org/

Pandas: https://pandas.pydata.org/

Random Forest: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Word Embedding: https://dataanalyticspost.com/Lexique/word-embedding/

Google Colab: https://colab.research.google.com/notebooks/intro.ipynb

Librairies adéquates:

- https://plotly.com/python/
- https://numpy.org/doc/stable/
- https://ipython.readthedocs.io/en/stable/api/generated/IPython.display.html
- https://docs.python.org/fr/3/library/collections.html#collections.Counter
- https://ichi.pro/fr/creer-des-nuages-de-mots-personnalises-en-python-145227399773811
- https://scikit-learn.org/stable/

Linear Regression: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

Decision Tree: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html