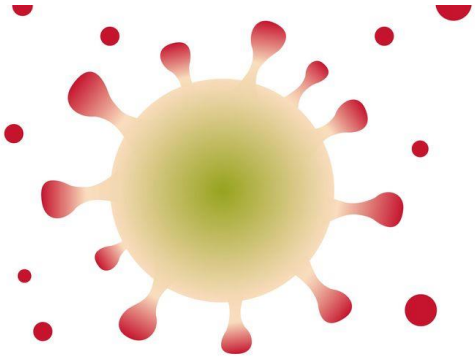
MINI-PROJET PYTHON SUR CORONA



Réalisé par :

Abderahmane HAMDOUCHI

Master : SDAD

Table des matières

I. Analyse des données

Il faut d'abord préparer notre Data ; Alors on doit enlever missing Data et analyser chaque feature avec un petit descriptif et son type (catégorique, numérique), afin de bien mener notre analyse

Dans cette étape on doit utiliser le jupyter (l'analyse de nos données)

Importer les biblio tiques :

Importation des données

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import plotly.offline as pyo
# ignore warning message
import warnings
warnings.filterwarnings("ignore")
```

Figure 1:Importation des librairies

Ensuite, afficher les premier raw comme suit :

```
Entrée [3]: df =pd.read_csv('patients-maroc.csv')

Entrée [4]: df
```

Out[4]:

	n	sex	age	country	province	disease	group	exposure_start	exposure_end	infection_reason	infection_order	infected_by	contact_number
0	0	male	39.0	Maroc	Casablanca - Settat	NaN	NaN	NaN	NaN	Imported	NaN	NaN	NaN
1	1	female	89.0	Maroc	Casablanca - Settat	1.0	NaN	NaN	NaN	Imported	NaN	NaN	NaN
2	2	male	59.0	France	Marrakech - Safi	NaN	NaN	3/7/20	NaN	Imported	NaN	NaN	NaN
3	3	female	NaN	France	Marrakech - Safi	NaN	NaN	3/7/20	NaN	Imported	NaN	NaN	NaN

Figure 2: le fichier Csv importé

On remarque qu'il y'a plusieurs valeurs qui sont nulle (Missing data) on doit les repérer tout d'abord pour les enlever ou bien les remplacer (raw or feature), on doit utiliser le traitement ci-après:

```
# Voir  s'il y a des valeurs qui sont nuls
#
missing_data = df.isnull()
# count missing value by columns
for column in missing_data.columns.values.tolist():
    print(column)
    print (missing_data[column].value_counts())
    print("")
```

Figure 3:count les valeurs qui sont nulles dans chaque feature

On obtient les données ci-après :

```
False    1184
Name: n, dtype: int64

sex
True     1120
False      64
Name: sex, dtype: int64

age
True     1145
False      39
Name: age, dtype: int64

country
True     1174
False      10
Name: country, dtype: int64

province
False    1184
Name: province, dtype: int64

disease
True     1174
False      10
Name: disease, dtype: int64

group
True    1184
Name: group, dtype: int64

exposure_start
True     1172
False      12
Name: exposure_start, dtype: int64

exposure_end
True     1169
False      15
Name: exposure_end, dtype: int64

infection_reason
False    1025
True      159
Name: infection_reason, dtype: int64

infection_order
True     1183
False       1
Name: infection_order, dtype: int64

infected_by
True    1184
Name: infected_by, dtype: int64

contact_number
True    1184
Name: contact_number, dtype: int64

confirmed_date
False    1184
Name: confirmed_date, dtype: int64

released_date
True     1178
False       6
Name: released_date, dtype: int64

deceased_date
True     1173
False      11
Name: deceased_date, dtype: int64

state
False    1184
Name: state, dtype: int64
```

Figure 4: Le nombre de valeurs nulle (True) de chaque colonne

```
df_new = df.drop(['n','sex','age','country','disease','group','exposure_start','exposure_end',
                  'infection_order','infected_by','contact_number'],axis=1)
df_new
```

Figure 5:: méthode de la suppression des features

On remarque que les features encadrés dans la figure précédente contiennent beaucoup de valeurs nulles plus de 98% de valeurs alors ils ne seront pas trop utiles à notre étude on doit les supprimer

Ainsi la première colonne qui ne contient que les indices, alors il ne sert à rien.

Mais on garde la colonne « released_date » et la colonne « deceased_date », malgré qu'elles contiennent plus de 96% de valeurs qui sont nulles, car ces gens-là sont en cours de traitement.

Pour se faire on effectue la méthode ci-après pour obtenir la data ci-après :

Axis=1 : désigne qu'on désire supprimer les colonnes

Et on obtient la nouvelle dataFrame ci-après :

	province	infection_reason	confirmed_date	released_date	deceased_date	state
0	Casablanca - Settat	Imported	3/2/20	3/17/20	NaN	Exit
1	Casablanca - Settat	Imported	3/5/20	NaN	3/10/20	Deceased
2	Marrakech - Safi	Imported	3/10/20	3/29/20	NaN	Exit
3	Marrakech - Safi	Imported	3/11/20	3/29/20	NaN	Exit
4	Marrakech - Safi	Imported	3/11/20	3/29/20	NaN	Exit
...	...	...	...	...	...	...
1179	Daraa - Tafilalet	Local	4/7/20	NaN	NaN	isolated
1180	Daraa - Tafilalet	Local	4/7/20	NaN	NaN	isolated
1181	Daraa - Tafilalet	Local	4/7/20	NaN	NaN	isolated
1182	Daraa - Tafilalet	Local	4/7/20	NaN	NaN	isolated
1183	Daraa - Tafilalet	Local	4/7/20	NaN	NaN	isolated

1184 rows × 6 columns

Figure 6: le nouveau DataFrame

Après je détermine le nombre des gens qui sont isolés, rétablis e décédés.

Par ce que la colonne State représente le Target ou bien la classe qu'on doit prédire :

```
df_new['state'].value_counts()

isolated    1164
Deceased      12
Exit           8
Name: state, dtype: int64
```

Figure 7: count le nombre des trois classe de la colonne « state » dans notre data

On n'as pas donc assez de données pour construire un bon modèle qui vas nous déterminer si la personne isolé va se rétablir ou bien il va être suicidé à cause de coronas (Deceased=12 et Exit=8), ainsi on as pas assez de features pour le faire et notre base de données contient beaucoup de missing data, si non, on vas passer au **« feature-Selection »** pour garder les features qui influences sur le Target, après on procède méthode de **« Imbalanced data »** , et ensuite on vas entrainer notre modèle (KNN, Arbre de décision, réseaux de neurones…….) et enfin on évalue nos modèle pour trouver le plus optimale à notre dataBase.

II. PLOTLY

Alors on va se focaliser sur la visualisation par le plotly :

Dans notre new_df, on n'a pas d'attributs numérique excepté les dates bien sûr, on a uniquement des attributs catégoriques, alors on doit chercher des graphes correspond à notre dataFrame :

On utilise la méthode ci-après pour voir la distribution de l'attribut statut :

```
import plotly.graph_objects as go
fig = go.Figure(
    data=[go.Bar(y=df_new['state'].value_counts() , x=df_new['state'].value_counts().index)],
    layout_title_text="La distribution de statut des gens infectés par Corona"
)
fig.show()
```

Figure 8: code pour afficher la première plot

Et on obtient comme résultat :

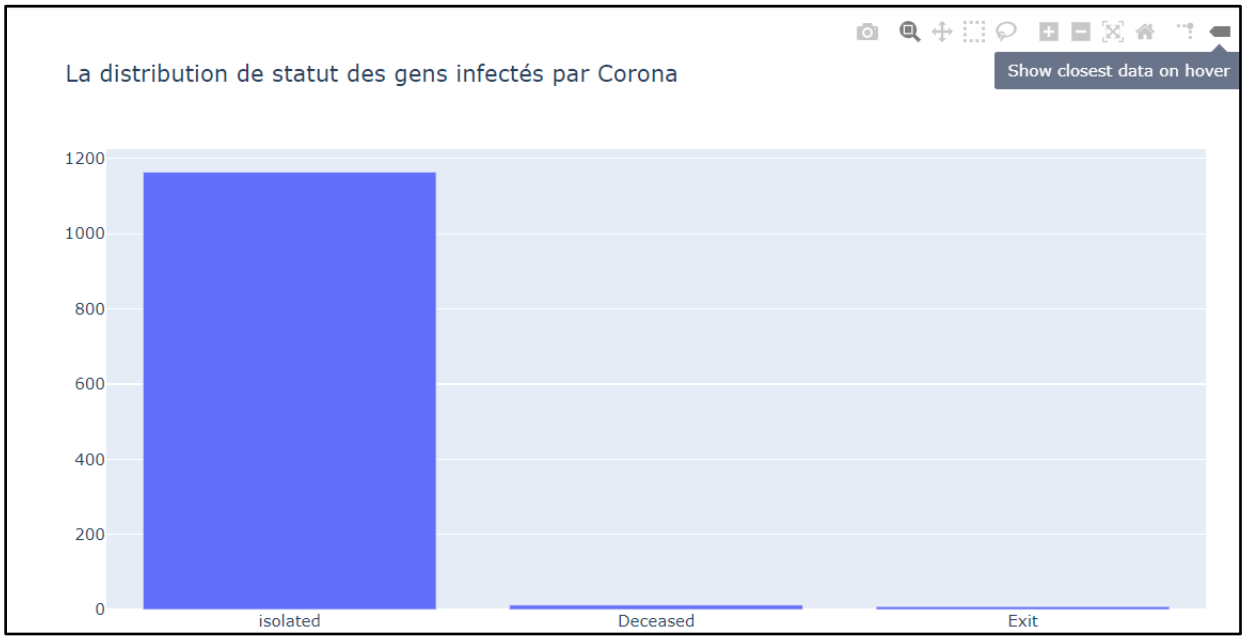


Figure 9: les barre représentant le nombre des 3 classe de state

Ensuite on doit convertir l'attribut confirmed_date (type String) to date pour avoir l'évaluation des cas infecté en fonction de la date et on le remplace à notre table

```
time= pd.to_datetime(df_new['confirmed_date'])
time
```

```
0       2020-03-02
1       2020-03-05
2       2020-03-10
3       2020-03-11
4       2020-03-11
           ...
1179    2020-04-07
1180    2020-04-07
1181    2020-04-07
1182    2020-04-07
1183    2020-04-07
Name: confirmed_date, Length: 1184, dtype: datetime64[ns]
```

Figure 10: convertir confirmed_date en type date

Et le même les 2 autres dates et mon obtient la table ci-après

```
df_new= df_new.drop(['confirmed_date','deceased_date','released_date'],axis=1)
df_new= pd.concat([time,time2,time3,df_new],axis=1)
df_new
```

	confirmed_date	released_date	deceased_date	province	infection_reason	state
0	2020-03-02	2020-03-17	NaT	Casablanca - Settat	Imported	Exit
1	2020-03-05	NaT	2020-03-10	Casablanca - Settat	Imported	Deceased
2	2020-03-10	2020-03-29	NaT	Marrakech - Safi	Imported	Exit
3	2020-03-11	2020-03-29	NaT	Marrakech - Safi	Imported	Exit
4	2020-03-11	2020-03-29	NaT	Marrakech - Safi	Imported	Exit
...	...	...	...	...	...	...
1179	2020-04-07	NaT	NaT	Daraa - Tafilalet	Local	isolated
1180	2020-04-07	NaT	NaT	Daraa - Tafilalet	Local	isolated
1181	2020-04-07	NaT	NaT	Daraa - Tafilalet	Local	isolated
1182	2020-04-07	NaT	NaT	Daraa - Tafilalet	Local	isolated
1183	2020-04-07	NaT	NaT	Daraa - Tafilalet	Local	isolated

1184 rows × 6 columns

Figure 11: Remplacer les anciens date de type String par ces correspondants parser en Date

Plot express figure en fonction des régions en utilisant le code ci-après et on obtient :

```
import plotly.express as px
fig = px.scatter(df_new, x="province", y="confirmed_date", color="state", title="A Plotly Express Figure")

fig.show()
```

Figure 12: méthode pour afficher la distribution en gens en fonction des régions et des dates par express

Et on le figure ci-après :

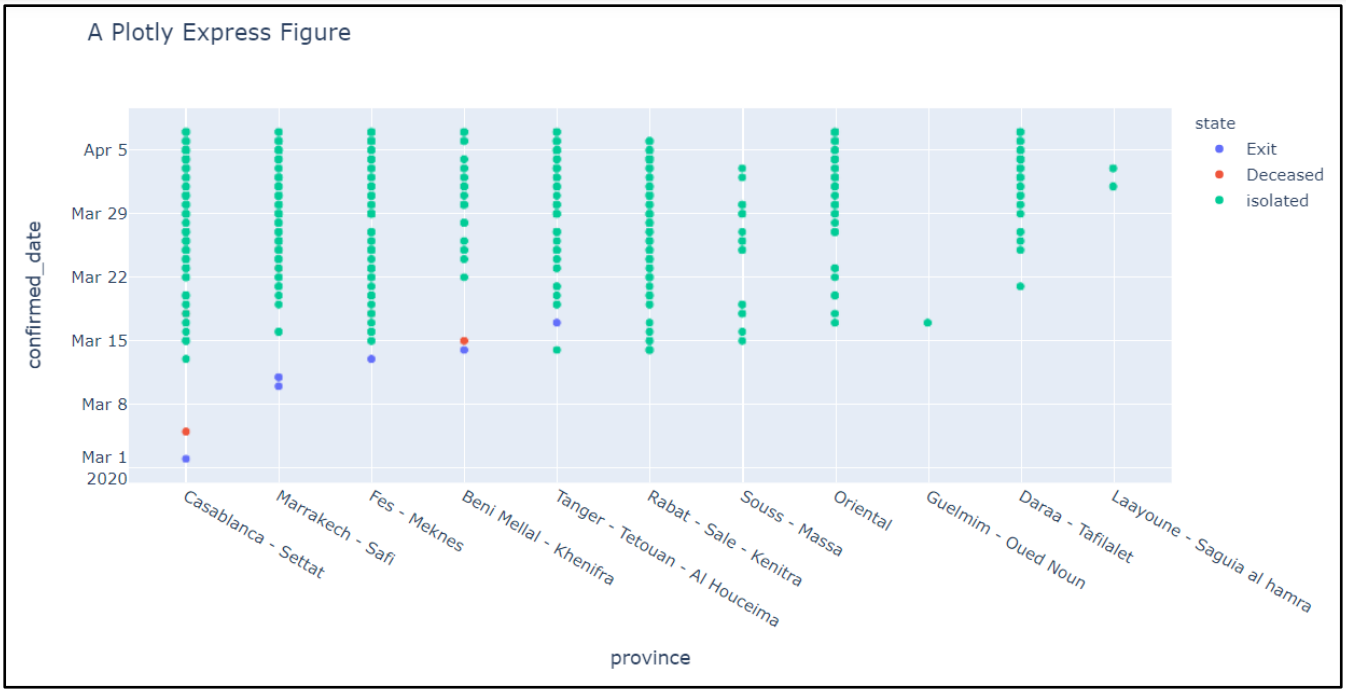


Figure 13:la distribution en gens en fonction des régions et des dates

La distribution en fonction des régions, en utilisant le code ci-après

```
from plotly.subplots import make_subplots

fig = make_subplots(rows=1, cols=2)

fig.add_trace(go.Scatter(y=df_new['province'].value_counts() , x=df_new['province'].value_counts().index, mode="lines"), row=1,
fig.add_trace(go.Bar(y=df_new['province'].value_counts() , x=df_new['province'].value_counts().index), row=1, col=2)

fig.show()
```

Figure 14:Méthode pour afficher la 3éme plot

On obtient :

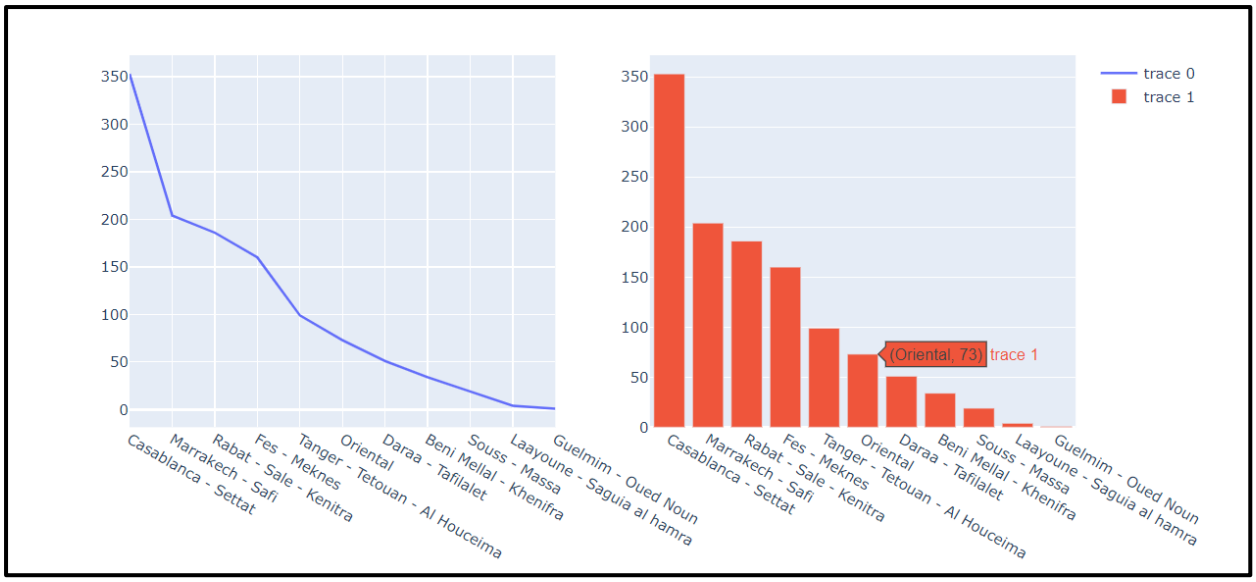


Figure 15: la distribution des cas en fonction des régions

La bibliothèque plotly est riche en graphes, presque une infinité de figures.

Maintenant on passe à l'implémentation de l'application Web sur Pycharm ;

Mais on doit tout d'abord exporter notre nouvelle base de donnée par la commande ci-après, pour en travailler avec :

```
df_new.to_csv("corona_maroc.csv")
```

III. Projet WEB du python

Ensuite créer le projet python par Pycharm :

Il faut télécharger les fichiers CSS contenant le style et suivre le même processus que celui de la 2eme playlist du YouTube .

Mais dans mon application le callback je l'ai lié avec une dropDown

```
@app.callback(Output('graph_close', 'figure'),
              [Input("myDropDown", "value")],
              )
```

Figure 16: Controller de call-back

Et j'ai mis une API de Journal du MAROC en général, car je n'ai pas trouvé une API du journal de la santé marocaine gratuit et je l'ai intégré via la méthode :

```
news_requests = requests.get(
    "http://newsapi.org/v2/top-headlines?country=ma&apiKey=9954e86486a5468eba3adc8943a53389"
)

# API Call to update news
def update_news():
```

Figure 17: méthode pour récupérer les news de l'API newapi

Dont son corps est disponible sur mon code source ainsi l'interface obtenue est la suivante :

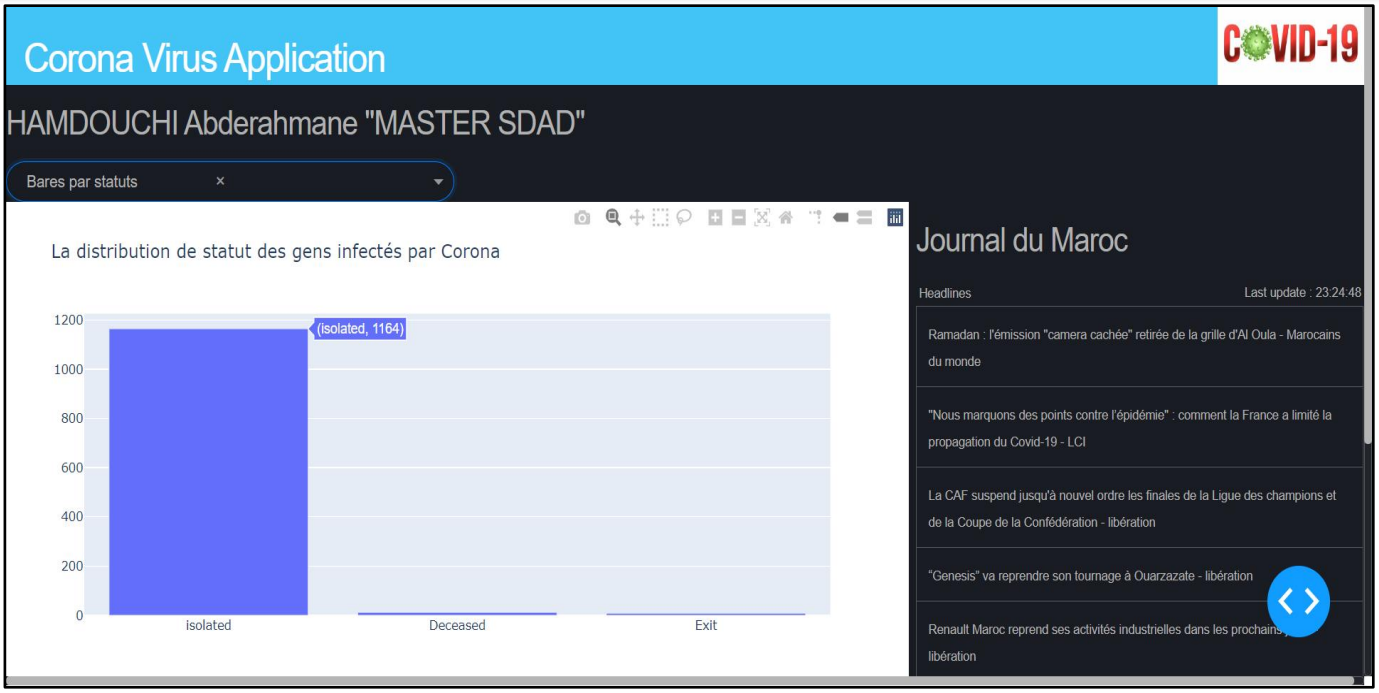


Figure 18: Mon application

IV. La mise de projet sur GitHub

Ensuite Commit and push le projet au GitHub :

1) Vous devez avoir le Git installé sur votre ordinateur
2) Ensuite Créer le projet « corona » sur ton compte GitHub

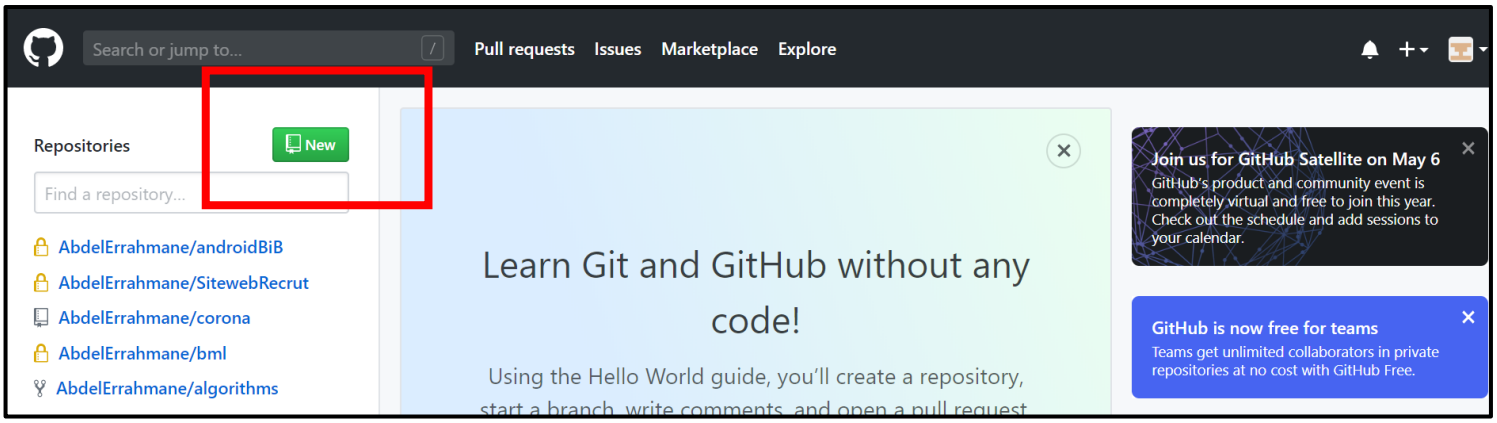


Figure 19:Créer nouvelle repository sur GitHub

VCS-> Checkout from Version Control -> Git

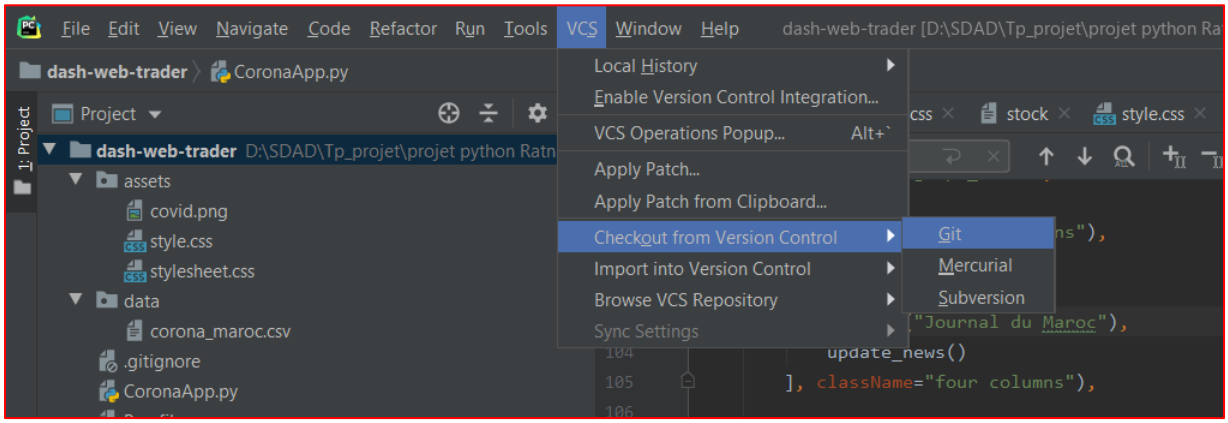


Figure 20: connecter notre projet au Git

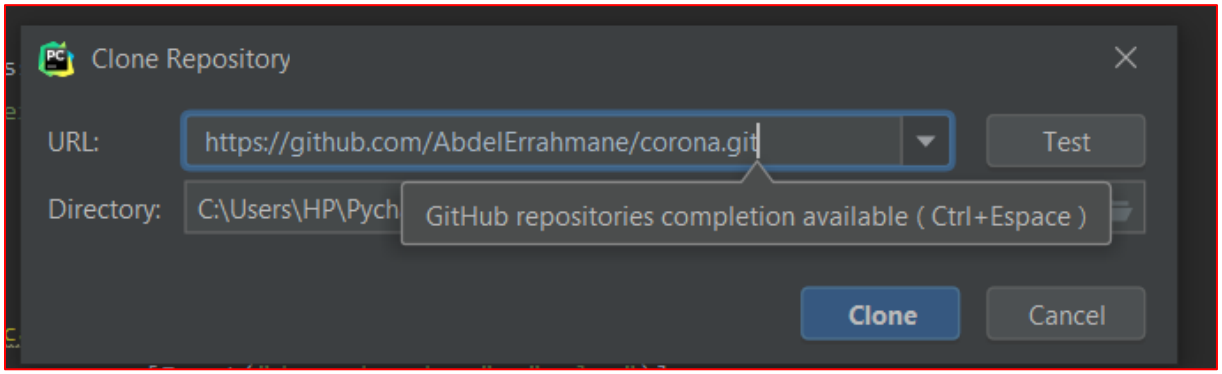


Figure 21: coller le lien de mon repository sur URL

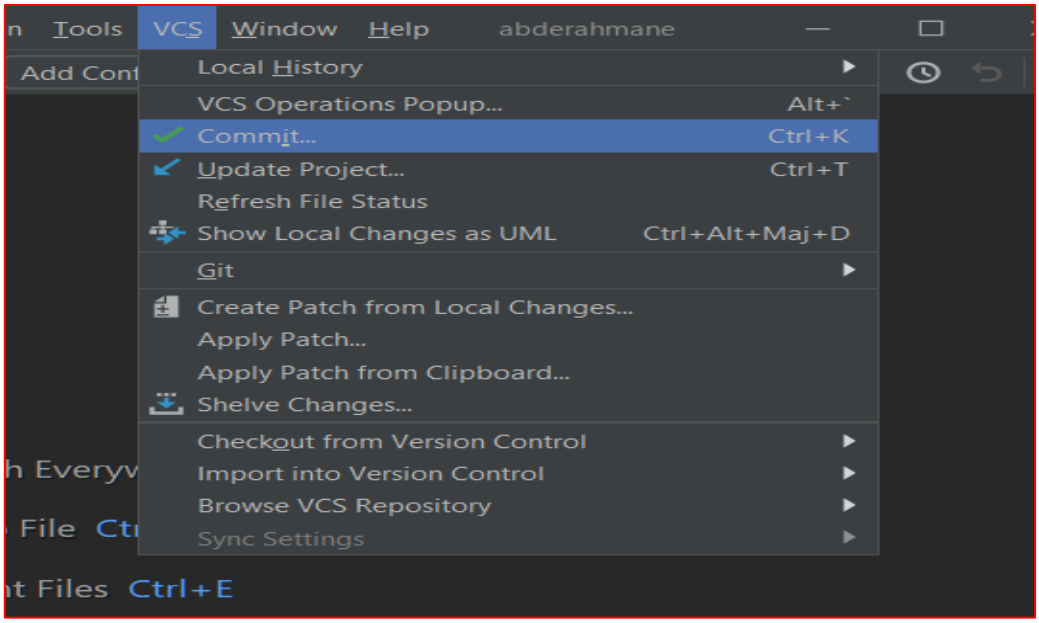


Figure 22: Ensuit "commit" mon projet

Ensuite on fait commit avec un commentaire

Et enfin on push

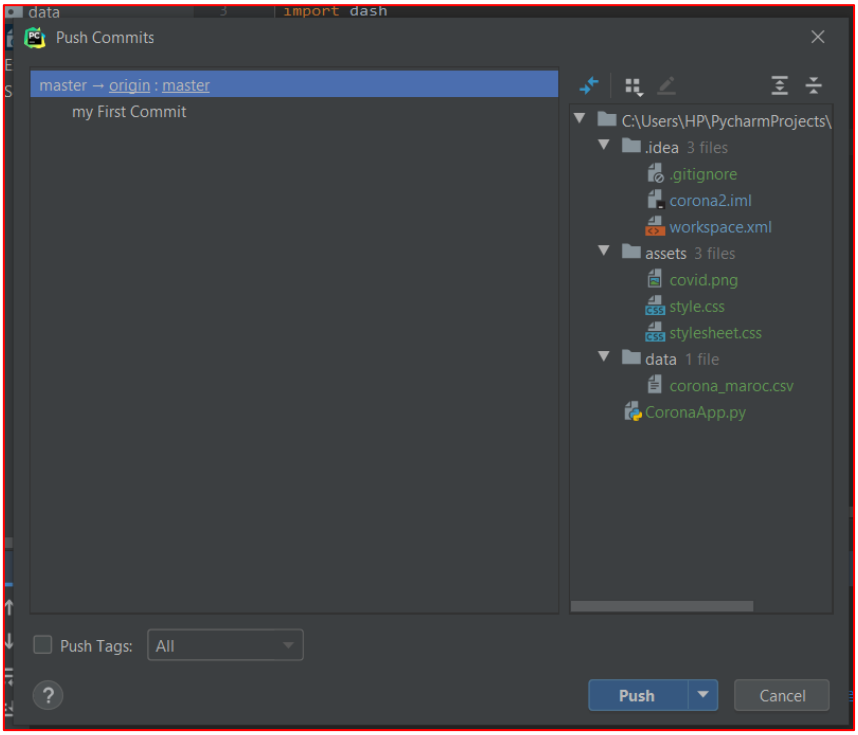


Figure 23: pull mon projet sur GitHub

Ainsi mon projet est disponible sur mon compte GitHub sur le lien ci-après

Voici le lien de mon projet

https://github.com/AbdelErrahmane/corona

Table des figures