PROJET INTEGRE | 2021

Compte rendu

Chawali Rania
Aouini Sabrine
Zalleg Mohamed Adem
Akrout Yosr

Contents

List of Figures

Chapter 2

introduction générale

Nous sommes des étudiants en 2 éme année licence en Mathématiques appliquées à l'analyse des données et aide à la décision à Esprit School of Business.
Nous allons vous présenter notre projet intégré pour cette année.

Zindi User Behaviour Birthday Challenge

2.1 Problématique

- On ne sait pas si les utilisateurs du zindi vont être actives l'année suivante ou non !
- Quelle méthode du Machine learning on va utiliser ?
- Quelle est la meilleure méthode ?

2.2 Solution

- Nettoyer les datas
- Analyser les datas et produire des graphiques pour mieux visualiser la relation entre eux
- Appliquer des différentes méthodes Machine Learning
- Faire les soumissions sur Zindi pour avoir un score

Chapter 3

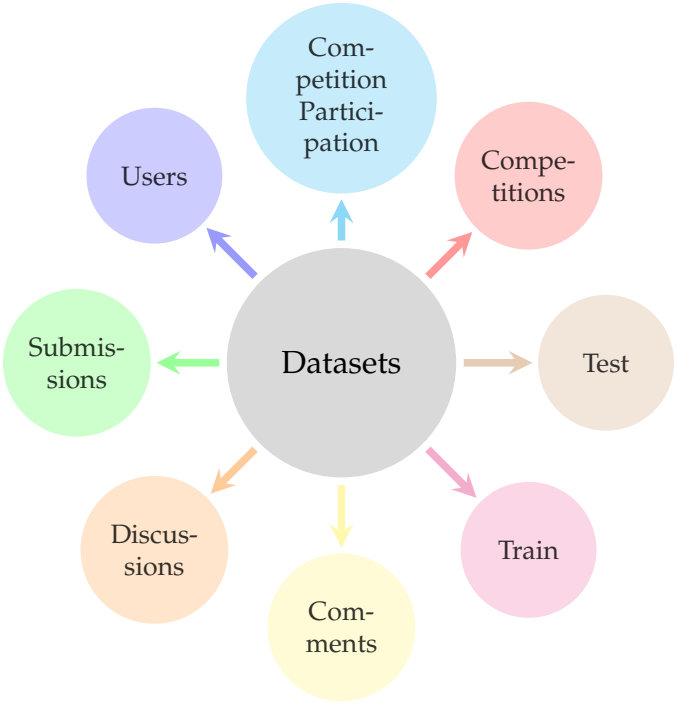
Contexe du travail

3.1 Description des données

3.1.1 Description du compétition

- **Compétition** : Zindi
- **Lien** : https://zindi.africa/competitions/zindi-user-behaviour-birthday-challenge
- **Nom d'équipe** : MAD
- **classement final** : 91 °/° 111

3.1.2 Les Datas



3.2 Description des outils

- **Jupyter Notebook** : Python en utilisant les librairies :

- **Pandas** : Analyse et Manipulation des données
- **Matlotlib et Seaborn** : Création des visualisations et des graphes
- **Scikit-Learn** :est une bibliothèque libre Python destinée à l'apprentissage automatique

3.3 Plan d'action

- Importer les fichiers CSV avec pandas
- Traitement et analyse des données avec python
- Visualisation avec pandas et matplotlib
- Prédiction avec sklearn
- Soumission sur zindi

Chapter 4

Analyse des données

4.1 importation des données

on a commencé par les datas originales et on les met dans un dossier

[1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

[118]:
```
Comments=pd.read_csv('Data_Zindi/Comments.csv')
CompetitionPartipation=pd.read_csv('Data_Zindi/CompetitionPartipation.csv')
Competitions=pd.read_csv('Data_Zindi/Competitions.csv')
Discussions=pd.read_csv('Data_Zindi/Discussions.csv')
Submissions=pd.read_csv('Data_Zindi/Submissions.csv')
users=pd.read_csv('Data_Zindi/users.csv')
train=pd.read_csv('Data_Zindi/train.csv')
test=pd.read_csv('Data_Zindi/test.csv')
```

[ ]:
```
#ici on va faire le traitement et l'analyse des données de nos datas
```

4.2 Comments

[119]:
```
Comments
```

[119]:
```
            UserID  CommentDate Year  CommentDate Month  \
0       ID_MVIB05DL                 3                  1
1       ID_MVIB05DL                 3                  1
2       ID_KBRFRAR9                 3                  1
3       ID_UDS6FRR8                 3                  1
4       ID_UDS6FRR8                 3                  1
...             ...               ...                ...
11746   ID_9UP7X8IA                 3                  5
11747   ID_9MJ12JJR                 3                  5
11748   ID_0B9SK73T                 3                  6
11749   ID_0B9SK73T                 3                  6
11750   ID_2IKJFHYK                 3                  6

       CommentDate Day_of_week
```

```
0                          7
1                          7
2                          7
3                          7
4                          7
...                      ...
11746                      7
11747                      4
11748                      5
11749                      4
11750                      1

[11751 rows x 4 columns]
```

[120]:
```
Comments.UserID.value_counts()
```

[120]:
```
ID_XGHA96QN    338
ID_29FH6HS1    275
ID_1OOXSUUG    209
ID_KBRFRAR9    186
ID_CQ6Q59UO    180
               ...
ID_AZP7WZDW      1
ID_O7SNGCRA      1
ID_GFTA2ZFG      1
ID_PBTIHYO2      1
ID_QFONRAYP      1
Name: UserID, Length: 2055, dtype: int64
```

[121]:
```
Comments.isna().sum()
```

[121]:
```
UserID                    0
CommentDate Year          0
CommentDate Month         0
CommentDate Day_of_week   0
dtype: int64
```

[ ]:
```
#on a supprimé la colonne day of week car on a besoin juste des month et year
```

[122]:
```
Comments.drop('CommentDate Day_of_week', axis = 1, inplace = True)
```

[123]:
```
Comments['comment_counter'] = Comments.groupby(['UserID', 'CommentDate Year']).
 ↪transform('count')['CommentDate Month']
```

[124]:
```
Comments =Comments.sort_values(by = ['CommentDate Year', 'CommentDate Month']).
 ↪reset_index(drop = True)
```

comment$_c$*ounter*: une nouvelle colonne qui contien la somme de comment date year et comment date month

[125]:
```
Comments['comment_counter'] = Comments.groupby(['UserID', 'CommentDate Year',␣
 ↪'CommentDate Month']).transform('sum')['comment_counter']
```

[126]:
```
Comments.drop_duplicates(inplace = True)
```

[127]:
```
Comments
```

[127]:
```
            UserID  CommentDate Year  CommentDate Month  comment_counter
0      ID_XGHA96QN                 1                  4               86
1      ID_PBBBZV83                 1                  4                1
3      ID_B2YLACIW                 1                  4                6
4      ID_S08HX5EF                 1                  4                6
5      ID_FG94IRYT                 1                  4                1
...            ...               ...                ...              ...
11744  ID_0W83JIX0                 3                 12                3
11745  ID_1I0QBZW8                 3                 12               26
11747  ID_4TIJ09OK                 3                 12               21
11748  ID_ZXKRXPJL                 3                 12                1
11750  ID_ZVE0W61X                 3                 12                1

[3899 rows x 4 columns]
```

[128]:
```
Comments['comment_counter'].value_counts()
```

[128]:
```
1       1099
4        385
2        248
6        150
3        131
        ...
1428       1
1440       1
3509       1
1540       1
1008       1
Name: comment_counter, Length: 371, dtype: int64
```

[129]:
```
co_counter = np.array(Comments['comment_counter'])
co_counter
```

[129]:
```
array([86,  1,  6, ..., 21,  1,  1], dtype=int64)
```

[130]:
```
for i in range(len(co_counter)):
    if co_counter[i] >= 50:
        co_counter[i] = 1
    else:
        co_counter[i] = 0
```

[134]:
```
Comments['comment_counter']=pd.DataFrame(co_counter)
Comments
```

[134]:
```
            UserID  CommentDate Year  CommentDate Month  comment_counter
0      ID_XGHA96QN                 1                  4              1.0
1      ID_PBBBZV83                 1                  4              0.0
3      ID_B2YLACIW                 1                  4              0.0
4      ID_S08HX5EF                 1                  4              0.0
```

```
5      ID_FG94IRYT                 1                   4              0.0
...            ...               ...                 ...              ...
11744  ID_0W83JIX0                 3                  12              NaN
11745  ID_1I0QBZW8                 3                  12              NaN
11747  ID_4TIJ09OK                 3                  12              NaN
11748  ID_ZXKRXPJL                 3                  12              NaN
11750  ID_ZVE0W61X                 3                  12              NaN

[3899 rows x 4 columns]
```

[135]:
```
Comments = Comments.fillna(0)
Comments
```

[135]:
```
            UserID  CommentDate Year  CommentDate Month  comment_counter
0      ID_XGHA96QN                 1                   4              1.0
1      ID_PBBBZV83                 1                   4              0.0
3      ID_B2YLACIW                 1                   4              0.0
4      ID_SO8HX5EF                 1                   4              0.0
5      ID_FG94IRYT                 1                   4              0.0
...            ...               ...                 ...              ...
11744  ID_0W83JIX0                 3                  12              0.0
11745  ID_1I0QBZW8                 3                  12              0.0
11747  ID_4TIJ09OK                 3                  12              0.0
11748  ID_ZXKRXPJL                 3                  12              0.0
11750  ID_ZVE0W61X                 3                  12              0.0

[3899 rows x 4 columns]
```

[136]:
```
comment_counter=pd.DataFrame(Comments[['UserID','comment_counter']])
comment_counter.columns = ['User_ID','comment_counter']
comment_counter
```

[136]:
```
           User_ID  comment_counter
0      ID_XGHA96QN              1.0
1      ID_PBBBZV83              0.0
3      ID_B2YLACIW              0.0
4      ID_SO8HX5EF              0.0
5      ID_FG94IRYT              0.0
...            ...              ...
11744  ID_0W83JIX0              0.0
11745  ID_1I0QBZW8              0.0
11747  ID_4TIJ09OK              0.0
11748  ID_ZXKRXPJL              0.0
11750  ID_ZVE0W61X              0.0

[3899 rows x 2 columns]
```

[143]:
```
comment_counter.to_csv("Data_Clean/comment_counter.csv", index = False)
```

4.3 Submissions

[49]:
```
Submissions
```

[49]:
```
             UserID  FeatureG   CompID  SubDate Year  SubDate Month  \
0       ID_8JP75F20         1  ID_GFDE             3              3
1       ID_8JP75F20         1  ID_GFDE             3              3
2       ID_8JP75F20         1  ID_GFDE             3              3
3       ID_8JP75F20         1  ID_GFDE             3              3
4       ID_8JP75F20         1  ID_GFDE             3              3
...             ...       ...     ...           ...            ...
375758  ID_CX5N3Q88         1  ID_EZD0             3             11
375759  ID_CX5N3Q88         1  ID_EZD0             3             11
375760  ID_CX5N3Q88         1  ID_EZD0             3             11
375761  ID_J6MM98N2         1  ID_92AG             3             12
375762  ID_J6MM98N2         1  ID_92AG             3             12

        SubDate Day_of_week
0                         2
1                         2
2                         4
3                         2
4                         4
...                     ...
375758                    3
375759                    3
375760                    3
375761                    4
375762                    4

[375763 rows x 6 columns]
```

[50]:
```
Submissions.UserID.value_counts()
```

[50]:
```
ID_6L3A64FZ    4762
ID_JQJGZA7V    4452
ID_2UG5YJF0    3659
ID_8X0LVG1S    3229
ID_29FH6HS1    3134
               ...
ID_VTMAZ00K       1
ID_6RLHTEM5       1
ID_MDLHTKS6       1
ID_1LE1Z77G       1
ID_70Q40ISS       1
Name: UserID, Length: 7265, dtype: int64
```

[51]:
```
Submissions.isna().sum()
```

[51]:
```
UserID                0
FeatureG              0
CompID                0
SubDate Year          0
```

```
SubDate Month          0
SubDate Day_of_week    0
dtype: int64
```

[52]:
```
Submissions['submission_counter'] = Submissions.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate Month']).transform('count')['FeatureG']
```

[53]:
```
Submissions['FeatureG_0'] = 0
Submissions['FeatureG_1'] = 0
Submissions['FeatureG_3'] = 0
```

[54]:
```
Sub0 = Submissions[Submissions['FeatureG'] == 0].reset_index(drop = True)
Sub1 = Submissions[Submissions['FeatureG'] == 1].reset_index(drop = True)
Sub3 = Submissions[Submissions['FeatureG'] == 3].reset_index(drop = True)
```

[55]:
```
Sub0['FeatureG_1'] = Sub0.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate Month']).transform('count')['FeatureG']
Sub1['FeatureG_0'] = Sub1.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate Month']).transform('count')['FeatureG']
Sub3['FeatureG_3'] = Sub3.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate Month']).transform('count')['FeatureG']
```

[56]:
```
Submit = pd.concat([Sub0,Sub1,Sub3]).reset_index(drop = True)
Submit
```

[56]:

	UserID	FeatureG	CompID	SubDate Year	SubDate Month	SubDate Day_of_week	submission_counter	FeatureG_0	FeatureG_1	FeatureG_3
0	ID_8JP75F20	0	ID_GFDE	3	3	1	34	0	2	0
1	ID_8JP75F20	0	ID_GFDE	3	3	1	34	0	2	0
2	ID_8JP75F20	0	ID_GCU7	3	1	2	37	0	1	0
3	ID_CB7ZH4MJ	0	ID_GFDE	3	3	5	31	0	1	0
4	ID_QUU4R5LF	0	ID_GFDE	3	3	7	23	0	6	0
...	...	...	...	...	...	...	...	...	...	...
375758	ID_ALY59XCD	3	ID_1INW	1	12	1	35	0	0	0
375759	ID_PAQ0CL27	3	ID_1INW	1	12	3	15	0	0	0
375760	ID_PAQ0CL27	3	ID_1INW	1	12	5	15	0	0	0
375761	ID_C5FVVZMY	3	ID_2KEY	2	8	5	3	0	0	0
375762	ID_6DNCWYE3	3	ID_R4NQ	3	11	5	39	0	0	0

```
1                0
2                0
3                0
4                0
...            ...
375758           3
375759           2
375760           2
375761           1
375762           1

[375763 rows x 10 columns]
```

[57]:
```
Submit.drop('FeatureG', inplace = True, axis = 1)
```

[58]:
```
Submit = Submit.sort_values(by = ['SubDate Year', 'SubDate Month']).reset_index(drop =␣
↪True)
```

[59]:
```
Submit.drop_duplicates(keep = 'first', inplace = True)
```

[60]:
```
Submit['FeatureG_0'] = Submit.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate␣
↪Month']).transform('sum')['FeatureG_0']
Submit['FeatureG_1'] = Submit.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate␣
↪Month']).transform('sum')['FeatureG_1']
Submit['FeatureG_3'] = Submit.groupby(['UserID', 'CompID', 'SubDate Year', 'SubDate␣
↪Month']).transform('sum')['FeatureG_3']
```

[61]:
```
Submit.drop_duplicates(['UserID', 'CompID', 'SubDate Year', 'SubDate Month'], inplace␣
↪= True)
```

[62]:
```
Submit['competition_counter'] = Submit.groupby(['UserID', 'SubDate Year', 'SubDate␣
↪Month']).transform('count')['CompID']
```

[63]:
```
Submit['submission_counter'] = Submit.groupby(['UserID', 'SubDate Year','SubDate␣
↪Month']).transform('sum')['submission_counter']
```

[64]:
```
Submit['FeatureG_0'] = Submit.groupby(['UserID', 'SubDate Year', 'SubDate Month']).
↪transform('sum')['FeatureG_0']
Submit['FeatureG_1'] =Submit.groupby(['UserID', 'SubDate Year', 'SubDate Month']).
↪transform('sum')['FeatureG_1']
Submit['FeatureG_3'] = Submit.groupby(['UserID', 'SubDate Year', 'SubDate Month']).
↪transform('sum')['FeatureG_3']
```

[65]:
```
Submit.drop_duplicates(['UserID', 'SubDate Year', 'SubDate Month'], inplace = True)
Submit =Submit.reset_index(drop = True)
Submit
```

[65]:
```
            UserID   CompID  SubDate Year  SubDate Month  SubDate Day_of_week  \
0      ID_0LFEI0ID  ID_BT9Z             1              4                    2
1      ID_CQ6Q59U0  ID_BT9Z             1              4                    6
2      ID_Z9RTXR5U  ID_BT9Z             1              4                    5
3      ID_71X8WTAB  ID_BT9Z             1              4                    5
4      ID_B2YLACIW  ID_BT9Z             1              4                    5
```

```
...            ...      ...            ...             ...                    ...
16838  ID_DCHE16A6  ID_WIA6              3              12                      1
16839  ID_Q6VU06QH  ID_WIA6              3              12                      7
16840  ID_ZVE0W61X  ID_BQO2              3              12                      2
16841  ID_SKAVKKTY  ID_BQO2              3              12                      3
16842  ID_J6MM98N2  ID_92AG              3              12                      4

      submission_counter FeatureG_0 FeatureG_1 FeatureG_3  competition_counter
0                     13         21         18          0                     2
1                     10         27          1          0                     1
2                     28        103          2          0                     3
3                     20        114          1          0                     1
4                     15         39          2          0                     1
...                  ...        ...        ...        ...                   ...
16838                  1          1          0          0                     1
16839                  1          1          0          0                     1
16840                  8         24          0          0                     1
16841                 14         28          0          0                     1
16842                  2          2          0          0                     1

[16843 rows x 10 columns]
```

[132]:
```
Submissions.drop('FeatureG', axis = 1, inplace = True)
```

[138]:
```
submission_counter=pd.DataFrame(Submit[['UserID','submission_counter']])
submission_counter.columns = ['User_ID','submission_counter']
submission_counter
```

[138]:
```
           User_ID submission_counter
0      ID_0LFEI0ID                 13
1      ID_CQ6Q59U0                 10
2      ID_Z9RTXR5U                 28
3      ID_71X8WTAB                 20
4      ID_B2YLACIW                 15
...            ...                ...
16838  ID_DCHE16A6                  1
16839  ID_Q6VU06QH                  1
16840  ID_ZVE0W61X                  8
16841  ID_SKAVKKTY                 14
16842  ID_J6MM98N2                  2

[16843 rows x 2 columns]
```

[144]:
```
submission_counter.to_csv("Data_Clean/submission_counter.csv", index = False)
```

4.4 CompetitionPartipation

[146]:
```
CompetitionPartipation
```

[146]:
```
         CompID       UserID PublicRank Successful Submission Count  \
0      ID_WMUF  ID_UWBBZ9OF    rank 11                    count 10
1      ID_MPSN  ID_UWBBZ9OF        NaN                         NaN
```

```
2      ID_WMUF  ID_1N5J2PGO        NaN                    count 10
3      ID_7MLO  ID_1N5J2PGO        NaN                         NaN
4      ID_2KEY  ID_1N5J2PGO        NaN                         NaN
...        ...          ...        ...                         ...
48560  ID_XYJZ  ID_5C4D0VO2    rank 11                    count 10
48561  ID_XYJZ  ID_JRJZQB8S    rank 11                    count 10
48562  ID_XYJZ  ID_H4FM7RBV        NaN                         NaN
48563  ID_XYJZ  ID_C3QOEMU4    rank 11                    count 10
48564  ID_XYJZ  ID_WF3E1TND        NaN                         NaN

       CompPartCreated Year  CompPartCreated Month  \
0                         3                      1
1                         3                      1
2                         3                      1
3                         2                     10
4                         2                      5
...                     ...                    ...
48560                     3                      7
48561                     3                      7
48562                     3                      7
48563                     3                      7
48564                     3                      7

       CompPartCreated Day_of_week
0                                4
1                                7
2                                5
3                                5
4                                4
...                            ...
48560                            6
48561                            5
48562                            5
48563                            5
48564                            7

[48565 rows x 7 columns]
```

[147]:
```
CompetitionPartipation.UserID.value_counts()
```

[147]:
```
ID_XGHA96QN    117
ID_KBRFRAR9     70
ID_HD4C3MI8     68
ID_RUNWIVIX     67
ID_4S9JJCZA     65
               ...
ID_RZ4PIUYI      1
ID_V6LQ7QAF      1
ID_WJ7HTGJT      1
ID_17SUY9EW      1
ID_IKVNGORR      1
Name: UserID, Length: 14854, dtype: int64
```

[148]:
```
CompetitionPartipation.isna().sum()
```

[148]:
```
CompID                           0
UserID                           0
PublicRank                   34598
Successful Submission Count  32426
CompPartCreated Year             0
CompPartCreated Month            0
CompPartCreated Day_of_week      0
dtype: int64
```

[149]:
```
CompetitionPartipation.drop('CompPartCreated Day_of_week', axis = 1, inplace = True)
```

[150]:
```
# on a crée une nouvelle colonne issubmit qui contient 0 si successful submission␣
 ↪count is nan and 1 si # 0
```

[151]:
```
CompetitionPartipation['isSubmitted'] = 0
CompetitionPartipation.loc[~CompetitionPartipation['Successful Submission Count'].
 ↪isna(), 'isSubmitted'] = 1
```

[ ]:
```
#on a supprimer rank et count dans les deux colonnes pour avoir des scores numeriques
```

[152]:
```
CompetitionPartipation['PublicRank'] = CompetitionPartipation['PublicRank'].str.
 ↪replace('rank','')
CompetitionPartipation['Successful Submission Count'] =␣
 ↪CompetitionPartipation['Successful Submission Count'].str.replace('count','')
CompetitionPartipation = CompetitionPartipation.fillna(0)
```

[153]:
```
CompetitionPartipation
```

[153]:

	CompID	UserID	PublicRank	Successful Submission Count	CompPartCreated Year	CompPartCreated Month	isSubmitted
0	ID_WMUF	ID_UWBBZ9OF	11	10	3	1	1
1	ID_MPSN	ID_UWBBZ9OF	0	0	3	1	0
2	ID_WMUF	ID_1N5J2PGO	0	10	3	1	1
3	ID_7MLO	ID_1N5J2PGO	0	0	2	10	0
4	ID_2KEY	ID_1N5J2PGO	0	0	2	5	0
...	...	...	...	...	...	...	...
48560	ID_XYJZ	ID_5C4DOVO2	11	10	3	7	1
48561	ID_XYJZ	ID_JRJZQB8S	11	10	3	7	1
48562	ID_XYJZ	ID_H4FM7RBV	0	0	3	7	0
48563	ID_XYJZ	ID_C3QOEMU4	11	10			
48564	ID_XYJZ	ID_WF3E1TND	0	0			

```
48563                    3                    7            1
48564                    3                    7            0

[48565 rows x 7 columns]
```

[154]:
```
Succeful_sub=pd.DataFrame(CompetitionPartipation[['UserID','Successful Submission␣
↪Count']])
Succeful_sub.columns = ['User_ID','Successful Submission Count']
Succeful_sub
```

[154]:
```
           User_ID Successful Submission Count
0      ID_UWBBZ90F                          10
1      ID_UWBBZ90F                           0
2      ID_1N5J2PG0                          10
3      ID_1N5J2PG0                           0
4      ID_1N5J2PG0                           0
...            ...                         ...
48560  ID_5C4D0VO2                          10
48561  ID_JRJZQB8S                          10
48562  ID_H4FM7RBV                           0
48563  ID_C3QOEMU4                          10
48564  ID_WF3E1TND                           0

[48565 rows x 2 columns]
```

[155]:
```
Succeful_sub.to_csv("Data_Clean/Succeful_sub.csv", index = False)
```

4.5 Competitions

[156]:
```
Competitions
```

[156]:
```
       CompID  Country FeatureA FeatureB  FeatureC  Kind  Points Reward  \
0    ID_WGZ2  ID_HWRH      [1]     [14]       1.0     1              27
1    ID_G370      NaN      [1]       []       2.0     1              52
2    ID_R5HL      NaN      [1]       []       3.0     1             126
3    ID_Y6XI  ID_ARVG      [1]       []       4.0     1              52
4    ID_8PEN  ID_I1L9      [1]       []       5.0     0               2
..       ...      ...      ...      ...       ...   ...             ...
149  ID_F7X4      NaN      [1]       []       2.0     1               2
150  ID_E1LI  ID_I1L9   [3, 2]     [14]       7.0     1               2
151  ID_MPSN      NaN      [1]      [9]       2.0     1               2
152  ID_Z5QP  ID_5OWN      [1]       []      31.0     0               2
153  ID_D42Y  ID_ARVG      [1]       []       2.0     1             151

     FeatureD  SecretCode  SubmissionLimitPerDay FeatureE CompEndTime Year  \
0           1           0                  100.0      [1]               1
1           2           0                   10.0      [1]               4
2           2           0                   10.0      [1]               4
3           2           0                   10.0      [1]               4
4           2           0                  200.0      [1]               2
..        ...         ...                    ...      ...             ...
149         3           0                   30.0      [1]       not mapped
```

```
150         3           0                   50.0      [2]       not mapped
151         3           0                   10.0   [1, 5]       not mapped
152         2           1                    NaN      [1]                3
153         1           0                   30.0      [1]                2

     CompEndTime Month  CompEndTime Day_of_week  CompStartTime Year  \
0                  8.0                      7.0                  1
1                  1.0                      6.0                  3
2                  1.0                      6.0                  3
3                  1.0                      6.0                  3
4                 11.0                      6.0                  2
..                 ...                      ...                ...
149                NaN                      NaN                  1
150                NaN                      NaN                  3
151                NaN                      NaN                  2
152                6.0                      5.0                  3
153                7.0                      7.0                  2

     CompStartTime Month  CompStartTime Day_of_week
0                      6                          5
1                     12                          7
2                     10                          4
3                     11                          4
4                     11                          4
..                   ...                        ...
149                    5                          3
150                    2                          3
151                    2                          7
152                    6                          5
153                    3                          4

[154 rows x 17 columns]
```

[157]:
```
Competitions.isna().sum()
```

[157]:
```
CompID                       0
Country                     23
FeatureA                     0
FeatureB                     0
FeatureC                    17
Kind                         0
Points Reward                0
FeatureD                     0
SecretCode                   0
SubmissionLimitPerDay        4
FeatureE                     0
CompEndTime Year             0
CompEndTime Month           20
CompEndTime Day_of_week     20
CompStartTime Year           0
CompStartTime Month          0
CompStartTime Day_of_week    0
dtype: int64
```

[158]:
```
Competitions=Competitions.drop('FeatureA',axis=1)
Competitions=Competitions.drop('FeatureB',axis=1)
Competitions=Competitions.drop('FeatureC',axis=1)
Competitions=Competitions.drop('Kind',axis=1)
Competitions=Competitions.drop('FeatureD',axis=1)
Competitions=Competitions.drop('SecretCode',axis=1)
Competitions=Competitions.drop('FeatureE',axis=1)
Competitions=Competitions.drop('CompEndTime Day_of_week',axis=1)
Competitions=Competitions.drop('CompStartTime Day_of_week',axis=1)
```

[159]:
```
Competitions['CompEndTime Year'].replace(['not mapped'],[5],inplace=True)
```

[160]:
```
Competitions['CompEndTime Year'].astype(int)
Competitions['CompEndTime Month'].astype(float)
Competitions['CompStartTime Year'].astype(float)
Competitions['CompStartTime Month'].astype(float)
Competitions['SubmissionLimitPerDay'].astype(float)
```

[160]:
```
0      100.0
1       10.0
2       10.0
3       10.0
4      200.0
       ...
149     30.0
150     50.0
151     10.0
152      NaN
153     30.0
Name: SubmissionLimitPerDay, Length: 154, dtype: float64
```

[ ]:
```
#on a calculer ici la durée de cette competition par mois
```

[161]:
```
Competitions['comp_duration'] = (Competitions['CompEndTime Year'].astype(int) -␣
 ↪Competitions['CompStartTime Year']) *12
```

[162]:
```
Competitions['month_diff_temp'] = np.nan
Competitions.loc[~Competitions['CompEndTime Month'].isna(), 'month_diff_temp'] =␣
 ↪Competitions[~Competitions['CompEndTime Month'].isna()]['CompEndTime Month'].
 ↪astype(int) - Competitions[~Competitions['CompStartTime Month'].
 ↪isna()]['CompStartTime Month'].astype(int)
Competitions['month_diff_temp'].fillna(0, inplace=True)
```

[163]:
```
Competitions['comp_duration'] = Competitions['comp_duration'] +␣
 ↪Competitions['month_diff_temp']
Competitions.drop('month_diff_temp', axis = 1, inplace = True)
```

[164]:
```
Competitions.drop('Country', axis = 1, inplace = True)
Competitions.drop('CompStartTime Year', axis = 1, inplace = True)
Competitions.drop('CompStartTime Month', axis = 1, inplace = True)
Competitions.drop('CompEndTime Year', axis = 1, inplace = True)
Competitions.drop('CompEndTime Month', axis = 1, inplace = True)
```

[165]:
```
Competitions
```

[165]:
```
       CompID  Points Reward  SubmissionLimitPerDay  comp_duration
0    ID_WGZ2             27                  100.0            2.0
1    ID_G370             52                   10.0            1.0
2    ID_R5HL            126                   10.0            3.0
3    ID_Y6XI             52                   10.0            2.0
4    ID_8PEN              2                  200.0            0.0
..       ...            ...                    ...            ...
149  ID_F7X4              2                   30.0           48.0
150  ID_E1LI              2                   50.0           24.0
151  ID_MPSN              2                   10.0           36.0
152  ID_Z5QP              2                    NaN            0.0
153  ID_D42Y            151                   30.0            4.0

[154 rows x 4 columns]
```

[166]:
```
Competitions.to_csv("Data_Clean/Competitions_Clean.csv", index = False)
```

4.6 Discussions

[167]:
```
Discussions
```

[167]:
```
      FeatureF  DiscDate Year  DiscDate Month  DiscDate Day_of_week  \
0            1              3              10                     1
1            0              3              12                     3
2            1              3               1                     3
3            0              3               9                     7
4            1              3               5                     7
...        ...            ...             ...                   ...
6206         0              3               7                     1
6207         1              3               7                     2
6208         0              3               7                     5
6209         1              3              11                     2
6210         0              3              11                     3

         DiscID      UserID
0     ID_Z77ETQ  ID_F2757IAI
1     ID_E47JKY  ID_F2757IAI
2     ID_CB4Y0N  ID_F2757IAI
3     ID_BNIHCF  ID_F2757IAI
4     ID_MLPYC0  ID_F2757IAI
...         ...         ...
6206  ID_07HHT5  ID_E2Q1K4TQ
6207  ID_9TID7A  ID_8I5VPQIF
6208  ID_IMGAT1  ID_UC2B2DBT
6209  ID_W3CY00  ID_VVUWHX7W
6210  ID_GT26RF  ID_A9FRILEL

[6211 rows x 6 columns]
```

[168]:
```
Discussions.UserID.value_counts()
```

[168]:
```
ID_XGHA96QN    228
ID_OZTCVTQP     81
ID_YGSLWHG3     80
ID_29FH6HS1     75
ID_GD2YO4JH     42
               ...
ID_8IQ4KUIQ      1
ID_7M46EVTI      1
ID_BS9K1C4Y      1
ID_1OP5P4GG      1
ID_8JP75F20      1
Name: UserID, Length: 2653, dtype: int64
```

[169]:
```
Discussions.isna().sum()
```

[169]:
```
FeatureF               0
DiscDate Year          0
DiscDate Month         0
DiscDate Day_of_week   0
DiscID                 0
UserID                 0
dtype: int64
```

[170]:
```
Discussions.drop('DiscDate Day_of_week', axis = 1, inplace = True)
Discussions.drop('FeatureF', axis = 1, inplace = True)
```

[ ]:
```
#on a crée une nouvelle colonne disc_count qui contien la somme des disc
```

[171]:
```
Discussions['discussions_counter'] = Discussions.groupby(['UserID','DiscDate Year',␣
↪'DiscDate Month']).transform('count')
```

[172]:
```
Discussions.drop('DiscID', axis = 1, inplace = True)
Discussions
```

[172]:

	DiscDate Year	DiscDate Month	UserID	discussions_counter
0	3	10	ID_F2757IAI	1
1	3	12	ID_F2757IAI	1
2	3	1	ID_F2757IAI	2
3	3	9	ID_F2757IAI	1
4	3	5	ID_F2757IAI	1
...	...	...	...	...
6206	3	7	ID_E2Q1K4TQ	1
6207	3	7	ID_8I5VPQIF	1
6208	3	7	ID_UC2B2DBT	1
6209	3	11	ID_VVUWHX7W	1
6210	3	11	ID_A9FRILEL	1

[6211 rows x 4 columns]

[173]:
```
discussions_counter=pd.DataFrame(Discussions[['UserID','discussions_counter']])
discussions_counter.columns = ['User_ID','discussions_counter']
discussions_counter
```

[173]:
```
          User_ID  discussions_counter
0     ID_F2757IAI                    1
1     ID_F2757IAI                    1
2     ID_F2757IAI                    2
3     ID_F2757IAI                    1
4     ID_F2757IAI                    1
...           ...                  ...
6206  ID_E2Q1K4TQ                    1
6207  ID_8I5VPQIF                    1
6208  ID_UC2B2DBT                    1
6209  ID_VVUWHX7W                    1
6210  ID_A9FRILEL                    1

[6211 rows x 2 columns]
```

[174]:
```
discussions_counter.to_csv("Data_Clean/discussions_counter.csv", index = False)
```

4.7 Users

[175]:
```
users
```

[175]:
```
            UserID  FeatureX  Country  FeatureY   Points  UserDate Year  \
0      ID_N5LTBAPU         0  ID_DMRM         1  group 3              2
1      ID_CLSFQBOS         0   ID_QO2         3  group 3              1
2      ID_RE6T58Y4         0   ID_QO2         0  group 3              2
3      ID_XJQQRJV3         0  ID_Z8BI         0  group 3              2
4      ID_1JHU6A8S         0   ID_QO2         3  group 3              2
...            ...       ...      ...       ...      ...            ...
22402  ID_D4SARSC7         0  ID_5OWN         1  group 3              1
22403  ID_B8VJJMWK         0   ID_QO2         3  group 3              2
22404  ID_XAQGPGAZ         0   ID_QO2         3  group 3              2
22405  ID_1AO7PVP2         0   ID_QO2         3  group 3              2
22406  ID_3ZXJIREU         0   ID_QO2         3  group 3              1

       UserDate Month  UserDate Day_of_week
0                   4                     4
1                   5                     4
2                  12                     3
3                   9                     2
4                  10                     1
...               ...                   ...
22402               5                     3
22403               3                     4
22404               3                     1
22405               5                     5
22406               7                     3

[22407 rows x 8 columns]
```

[176]:
```
users.UserID.value_counts()
```

[176]:
```
ID_FCSQOBX8    1
ID_WHU3O6GQ    1
ID_AFQYWGZ3    1
ID_KCUZHVEV    1
ID_2PZKAKR9    1
               ..
ID_ALQHSTVD    1
ID_JNCFWCM6    1
ID_SKQO8CEF    1
ID_9BEJ2OA6    1
ID_LNXE8Z4A    1
Name: UserID, Length: 22407, dtype: int64
```

[177]:
```
users.isna().sum()
```

[177]:
```
UserID                  0
FeatureX                0
Country                 0
FeatureY                0
Points                  0
UserDate Year           0
UserDate Month          0
UserDate Day_of_week    0
dtype: int64
```

[178]:
```
users.drop('UserDate Day_of_week', axis = 1, inplace = True)
users.drop('FeatureX', axis = 1, inplace = True)
users.drop('FeatureY', axis = 1, inplace = True)
```

[179]:
```
users['Points'] = users['Points'].str.replace('group','')
```

[180]:
```
users
```

[180]:

	UserID	Country	Points	UserDate Year	UserDate Month
0	ID_N5LTBAPU	ID_DMRM	3	2	4
1	ID_CLSFQBOS	ID_QO2	3	1	5
2	ID_RE6T58Y4	ID_QO2	3	2	12
3	ID_XJQQRJV3	ID_Z8BI	3	2	9
4	ID_1JHU6A8S	ID_QO2	3	2	10
...	...	...	...	...	...
22402	ID_D4SARSC7	ID_5OWN	3	1	5
22403	ID_B8VJJMWK	ID_QO2	3	2	3
22404	ID_XAQGPGAZ	ID_QO2	3	2	3
22405	ID_1AO7PVP2	ID_QO2	3	2	5
22406	ID_3ZXJIREU	ID_QO2	3	1	7

[22407 rows x 5 columns]

[181]:
```
users.to_csv("Data_Clean/users_Clean.csv", index = False)
```

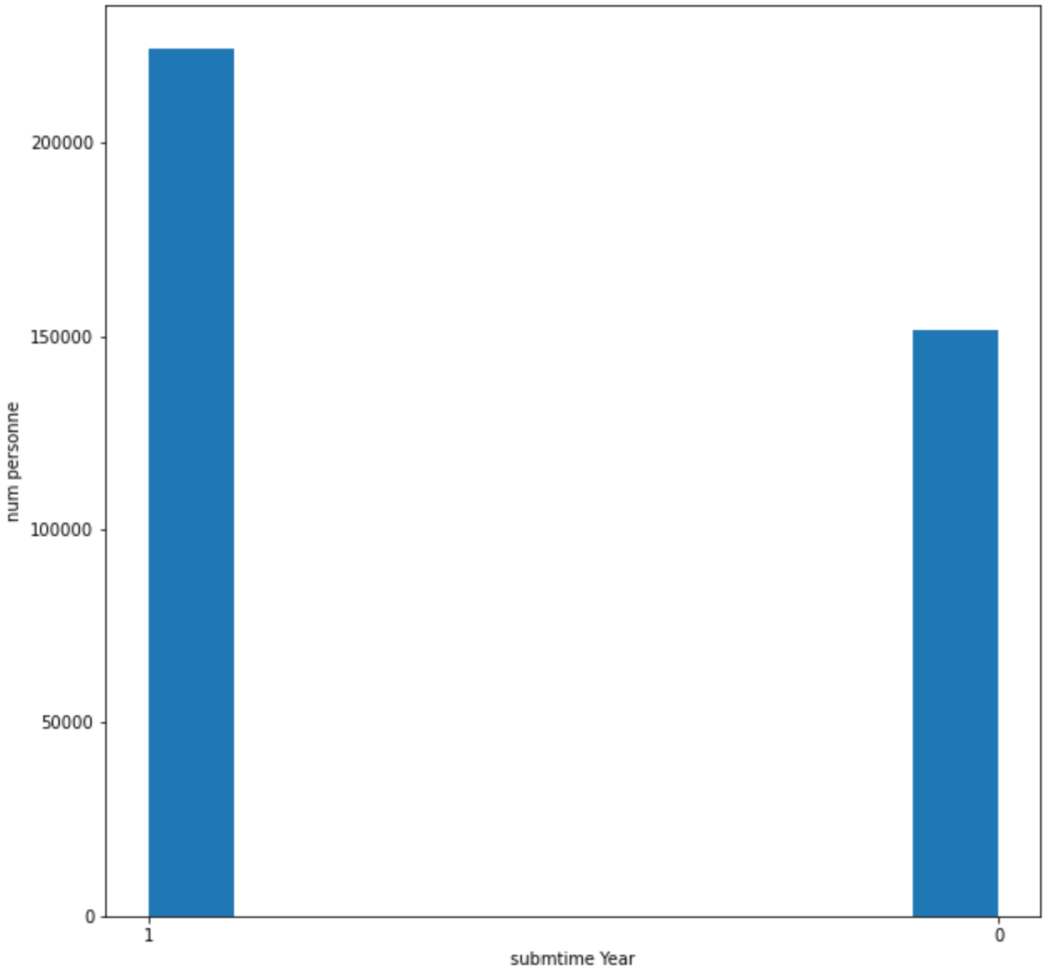
[ ]:

[ ]:

Chapter 5

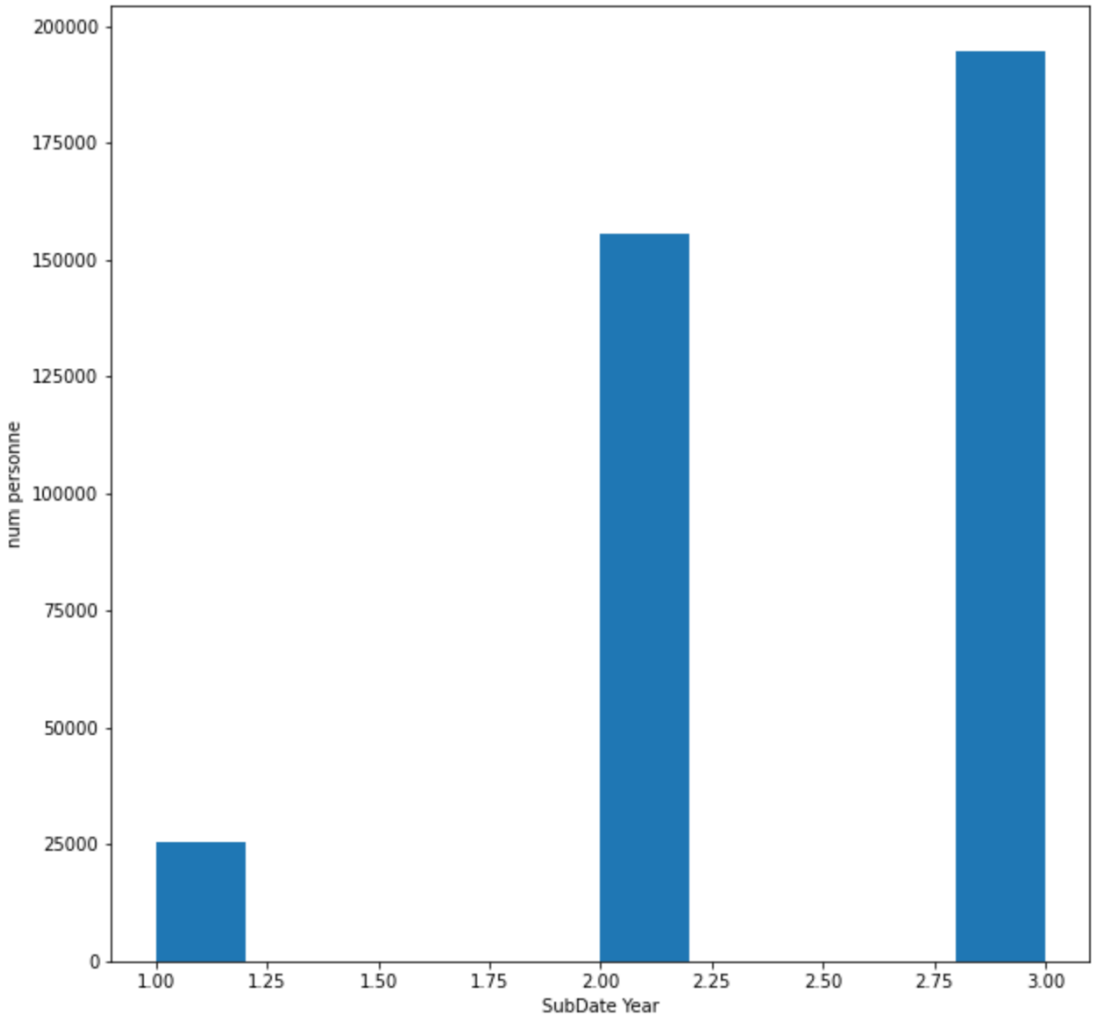
Visualisation

5.1 Interpretation

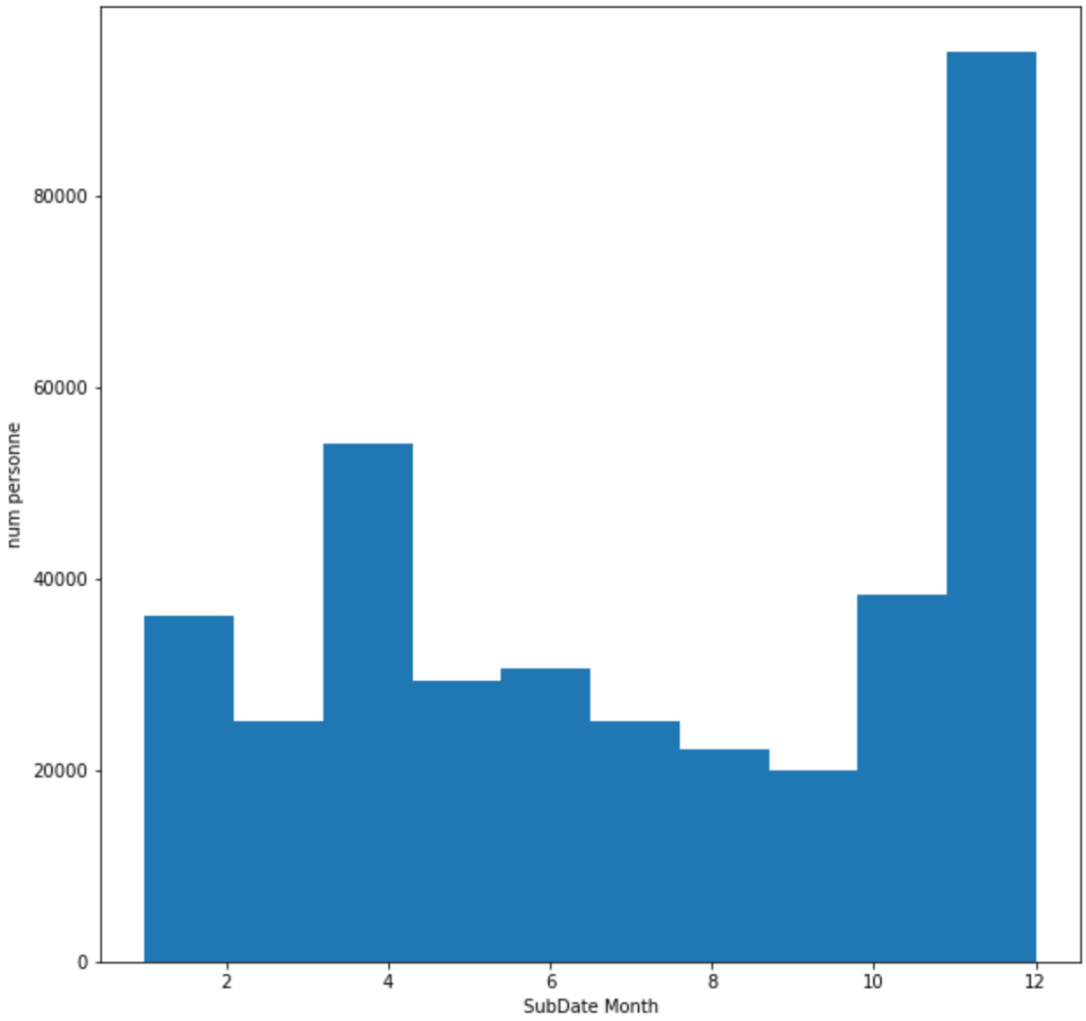


1 : personnes font la submission dans la même année de création des comptes sur zindi
0 : personnes font la submission après la première année de création des comptes sur zindi
Les personnes font la submission dans la même année de création des comptes sur zindi sont plus que les

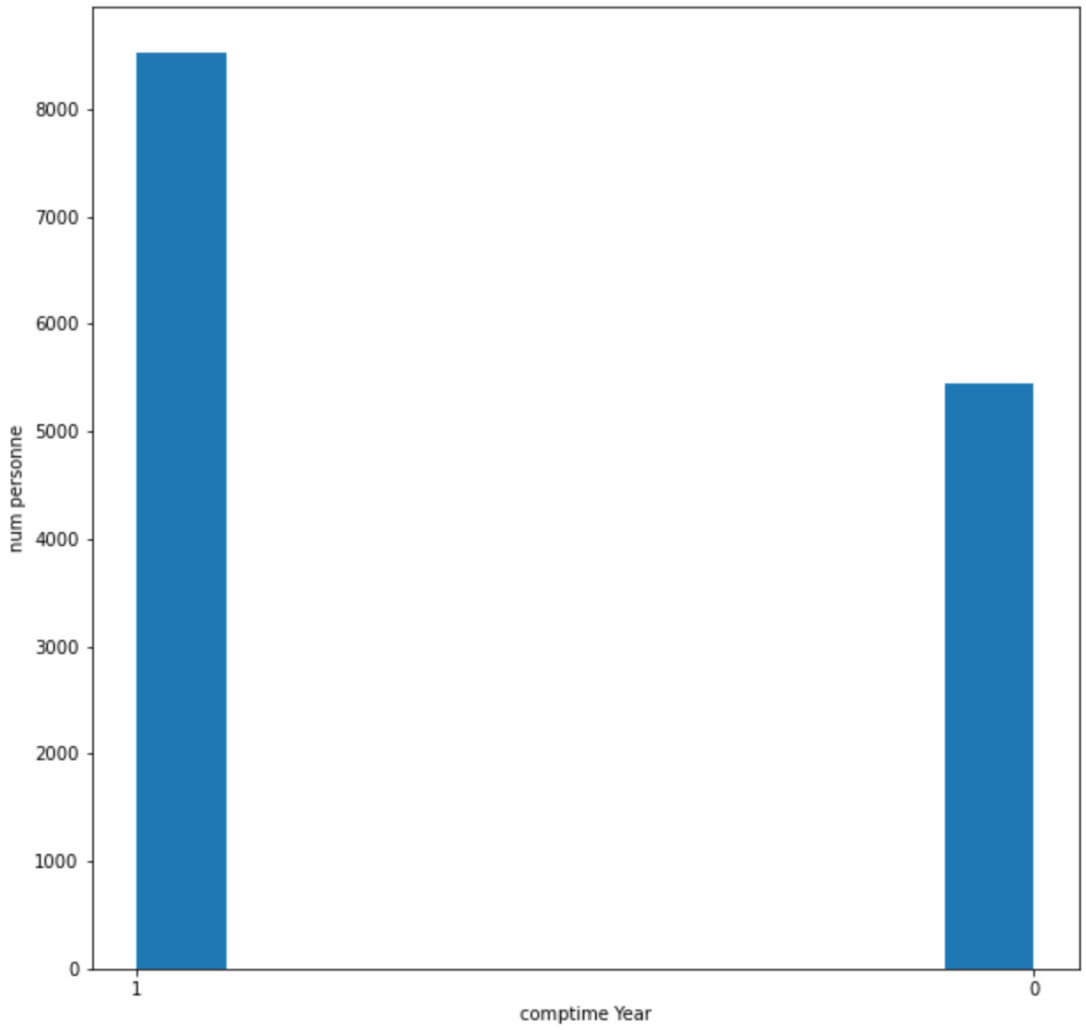
personnes qui font la submission après la première année de création.



La plus part des personnes font la submission pendant la 3 -ème année d'utilisation zindi



D’apres le plot de submission date month on observe que le nombre de submission en mois Numéro 12 est plus élevé.



On remarque qu'il y a beaucoup de personnes participent à un défi zindi dans le Même année d'utilisation

5.2 Notebook Visualisation

ici dans cette notebook , on va comprendre un peu les datas et leurs relations

[1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
```

[2]:
```
comments = pd.read_csv("Comments.csv",sep=',',header = 0)
competition_paricipation = pd.read_csv("CompetitionPartipation.csv",sep=',',header = 0)
competitions = pd.read_csv("Competitions.csv",sep=',',header = 0)
discussions = pd.read_csv("Discussions.csv",sep=',',header = 0)
sample_submission = pd.read_csv("SampleSubmission.csv",sep=',',header = 0)
submissions = pd.read_csv("Submissions.csv",sep=',',header = 0)
test = pd.read_csv("Test.csv",sep=',',header = 0)
```

```
train = pd.read_csv("Train.csv",sep=',',header = 0)
users = pd.read_csv("Users.csv",sep=',',header = 0)
variable_def=pd.read_csv("VariableDefinitions.csv",sep=',',header = 0)
```

5.3 Users et Submissions

[3]:
```
#concaténation des deux datas users et submissions group by userID*
data1=pd.merge(users,submissions)
data1
```

[3]:

	UserID	FeatureX	Country	FeatureY	Points	UserDate Year	UserDate Month	UserDate Day_of_week	FeatureG	CompID	SubDate Year	SubDate Month	SubDate Day_of_week
0	ID_N5LTBAPU	0	ID_DMRM	1	group 3	2	4	4	1	ID_HJ9S	2	4	5
1	ID_CLSFQBOS	0	ID_Q02	3	group 3	1	5	4	1	ID_LMFN	1	5	4
2	ID_RE6T58Y4	0	ID_Q02	0	group 3	2	12	3	1	ID_H4L8	2	12	5
3	ID_RE6T58Y4	0	ID_Q02	0	group 3	2	12	3	1	ID_H4L8	2	12	5
4	ID_RE6T58Y4	0	ID_Q02	0	group 3	2	12	3	1	ID_H4L8	2	12	6
...	...	...	...	...	...	...	...	...	...	...	...	...	...
375758	ID_1AO7PVP2	0	ID_Q02	3	group 3	2	5	5	1	ID_AWEI	2	6	5
375759	ID_3ZXJIREU	0	ID_Q02	3	group 3	1	7	3	1	ID_BT9Z	1	7	3
375760	ID_3ZXJIREU	0	ID_Q02	3	group 3	1	7	3	1	ID_BT9Z	1	7	3
375761	ID_3ZXJIREU	0	ID_Q02	3	group 3	1	7	3	1	ID_BT9Z	1	7	3
375762	ID_3ZXJIREU	0	ID_Q02	3	group 3	1	7	3	1	ID_BT9Z	1	7	3

[375763 rows x 13 columns]

[4]:
```
del data1['FeatureX']
```

[5]:
```
del data1['FeatureY']
```

[6]:
```
del data1['Points']
```

[7]:
```
del data1['FeatureG']
```

[8]:
```
data1
```

[8]:

	UserID	Country	UserDate Year	UserDate Month	UserDate Day_of_week	CompID	SubDate Year	SubDate Month	SubDate Day_of_week
0	ID_N5LTBAPU	ID_DMRM	2	4	4	ID_HJ9S	2	4	5
1	ID_CLSFQBOS	ID_QO2	1	5	4	ID_LMFN	1	5	4
2	ID_RE6T58Y4	ID_QO2	2	12	3	ID_H4L8	2	12	5
3	ID_RE6T58Y4	ID_QO2	2	12	3	ID_H4L8	2	12	5
4	ID_RE6T58Y4	ID_QO2	2	12	3	ID_H4L8	2	12	6
...	...	...	...	...	...	...	...	...	...
375758	ID_1AO7PVP2	ID_QO2	2	5	5	ID_AWEI	2	6	5
375759	ID_3ZXJIREU	ID_QO2	1	7	3	ID_BT9Z	1	7	3
375760	ID_3ZXJIREU	ID_QO2	1	7	3	ID_BT9Z	1	7	3
375761	ID_3ZXJIREU	ID_QO2	1	7	3	ID_BT9Z	1	7	3
375762	ID_3ZXJIREU	ID_QO2	1	7	3	ID_BT9Z	1	7	3

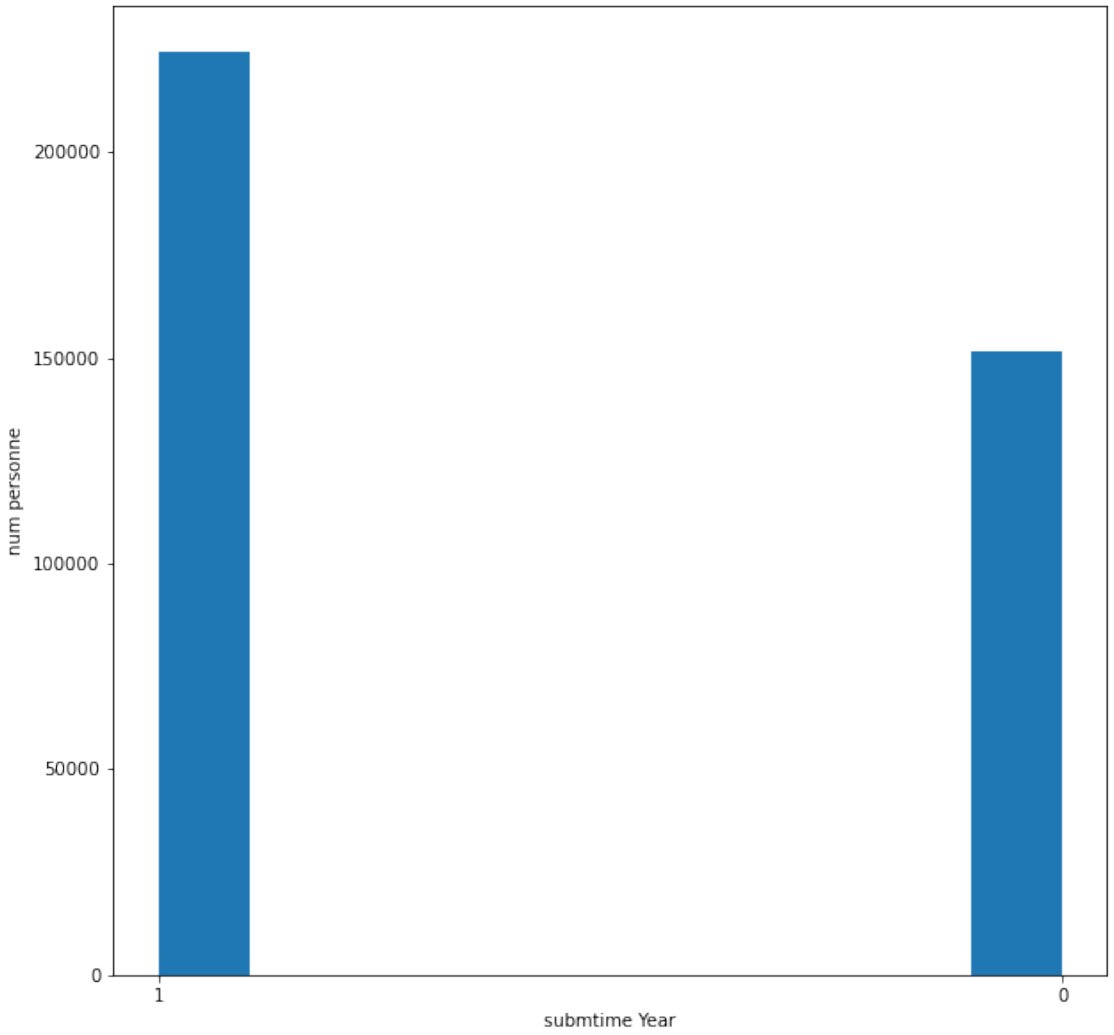
[375763 rows x 9 columns]

[9]:
```
data1['submtime Year']=np.where(data1['UserDate Year']==data1['SubDate Year'],'1','0')
data1
```

[9]:
```
             UserID  Country  UserDate Year  UserDate Month  \
0       ID_N5LTBAPU  ID_DMRM              2               4
1       ID_CLSFQB0S   ID_Q02              1               5
2       ID_RE6T58Y4   ID_Q02              2              12
3       ID_RE6T58Y4   ID_Q02              2              12
4       ID_RE6T58Y4   ID_Q02              2              12
...             ...      ...            ...             ...
375758  ID_1AO7PVP2   ID_Q02              2               5
375759  ID_3ZXJIREU   ID_Q02              1               7
375760  ID_3ZXJIREU   ID_Q02              1               7
375761  ID_3ZXJIREU   ID_Q02              1               7
375762  ID_3ZXJIREU   ID_Q02              1               7

        UserDate Day_of_week   CompID  SubDate Year  SubDate Month  \
0                          4  ID_HJ9S             2              4
1                          4  ID_LMFN             1              5
2                          3  ID_H4L8             2             12
3                          3  ID_H4L8             2             12
4                          3  ID_H4L8             2             12
...                      ...      ...           ...            ...
375758                     5  ID_AWEI             2              6
375759                     3  ID_BT9Z             1              7
375760                     3  ID_BT9Z             1              7
375761                     3  ID_BT9Z             1              7
375762                     3  ID_BT9Z             1              7

        SubDate Day_of_week submtime Year
0                         5             1
1                         4             1
2                         5             1
3                         5             1
4                         6             1
...                     ...           ...
375758                    5             1
375759                    3             1
375760                    3             1
375761                    3             1
375762                    3             1

[375763 rows x 10 columns]
```

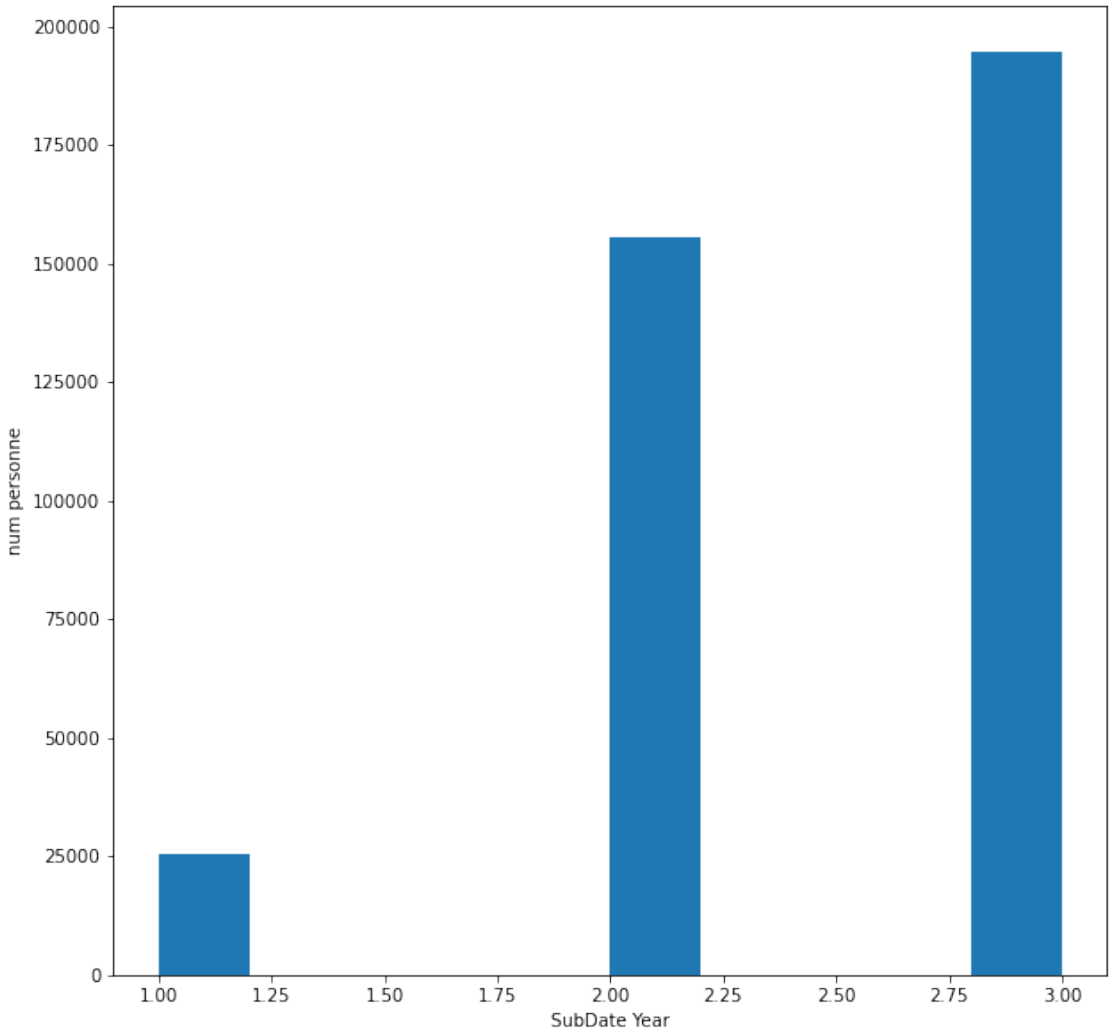
[10]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data1['submtime Year'])
plt.xlabel('submtime Year')
plt.ylabel('num personne')
#plus de personnes font la submission dans la meme année de creation des comptes sur␣
 ↪zindi
```

[10]:
```
Text(0, 0.5, 'num personne')
```



[11]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data1['SubDate Year'])
plt.xlabel('SubDate Year')
plt.ylabel('num personne')
#la plus part des personnes font la submissions pendant la 3 eme année d'utilisation␣
 ↪zindi
```

[11]:
```
Text(0, 0.5, 'num personne')
```



[12]:
```
data1['submtime Month']=np.where(data1['UserDate Month']==data1['SubDate␣
 ↪Month'],'1','0')
data1
```

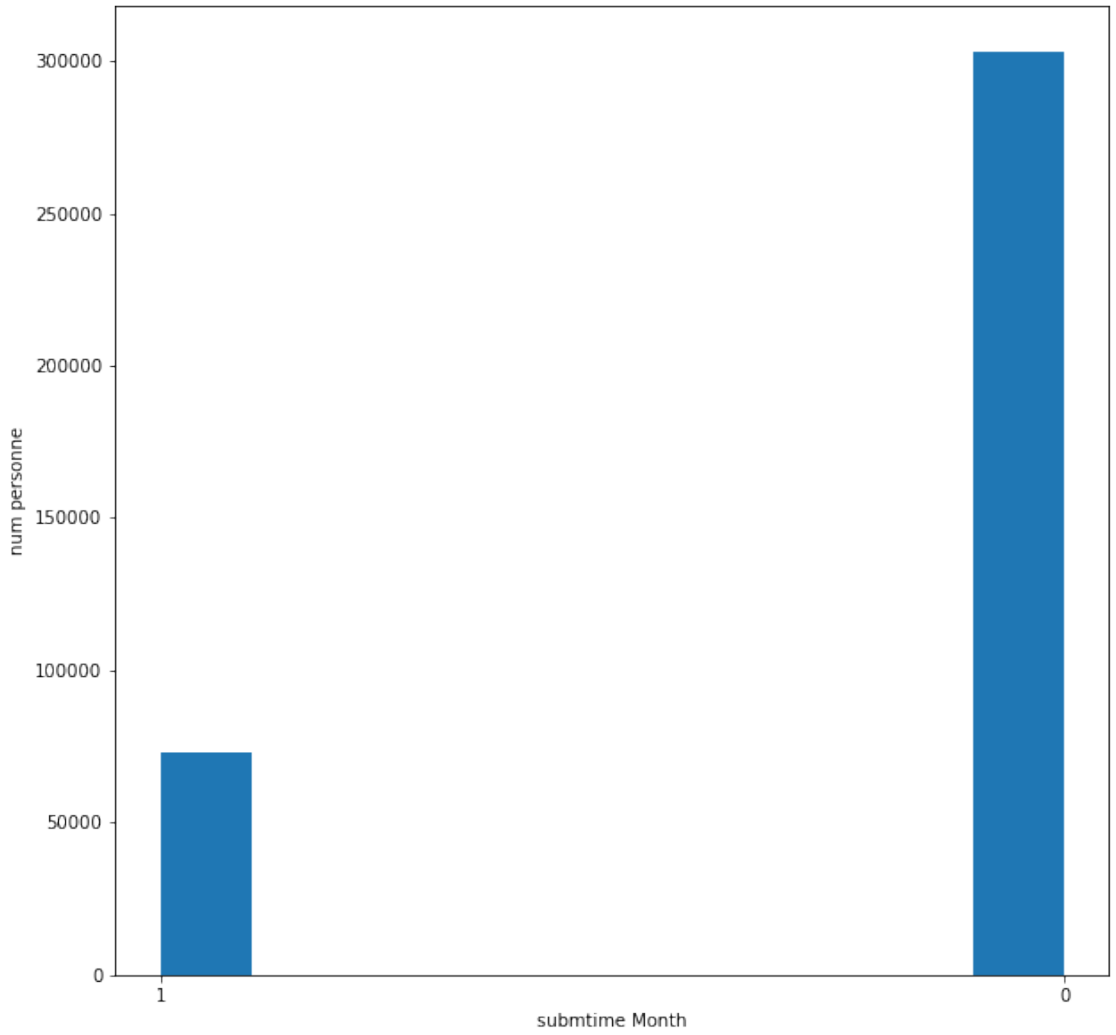
[12]:

	UserID	Country	UserDate Year	UserDate Month	\
0	ID_N5LTBAPU	ID_DMRM	2	4	
1	ID_CLSFQBOS	ID_QO2	1	5	
2	ID_RE6T58Y4	ID_QO2	2	12	
3	ID_RE6T58Y4	ID_QO2	2	12	
4	ID_RE6T58Y4	ID_QO2	2	12	
...	...	...	...	...	
375758	ID_1AO7PVP2	ID_QO2	2	5	
375759	ID_3ZXJIREU	ID_QO2	1	7	
375760	ID_3ZXJIREU	ID_QO2	1	7	
375761	ID_3ZXJIREU	ID_QO2	1	7	
375762	ID_3ZXJIREU	ID_QO2	1	7	

```
        UserDate Day_of_week   CompID  SubDate Year  SubDate Month  \
0                          4  ID_HJ9S             2              4
1                          4  ID_LMFN             1              5
2                          3  ID_H4L8             2             12
3                          3  ID_H4L8             2             12
4                          3  ID_H4L8             2             12
...                      ...      ...           ...            ...
375758                     5  ID_AWEI             2              6
375759                     3  ID_BT9Z             1              7
375760                     3  ID_BT9Z             1              7
375761                     3  ID_BT9Z             1              7
375762                     3  ID_BT9Z             1              7

        SubDate Day_of_week submtime Year submtime Month
0                          5             1              1
1                          4             1              1
2                          5             1              1
3                          5             1              1
4                          6             1              1
...                      ...           ...            ...
375758                     5             1              0
375759                     3             1              1
375760                     3             1              1
375761                     3             1              1
375762                     3             1              1

[375763 rows x 11 columns]
```

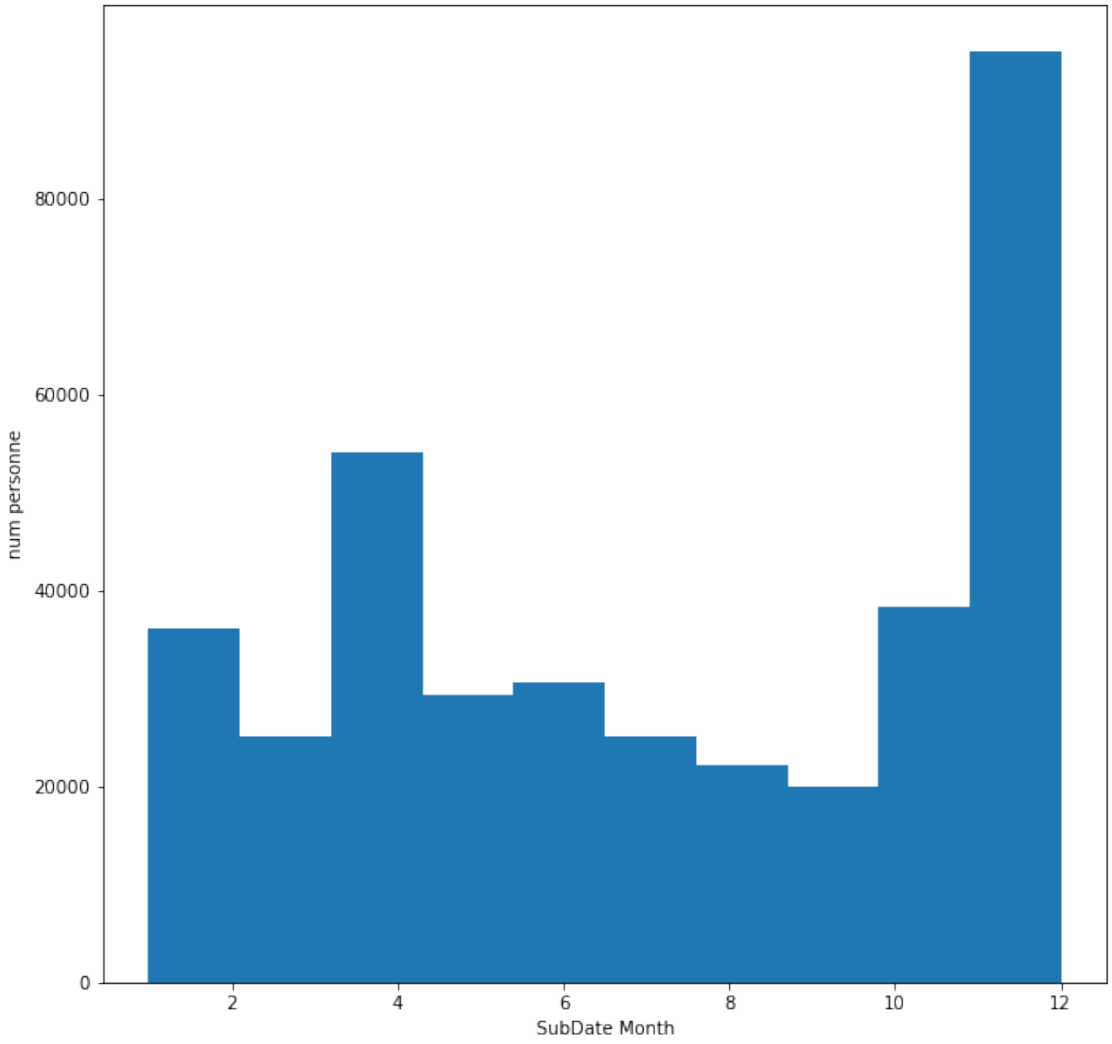
[13]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data1['submtime Month'])
plt.xlabel('submtime Month')
plt.ylabel('num personne')
```

[13]:
```
Text(0, 0.5, 'num personne')
```



[14]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data1['SubDate Month'])
plt.xlabel('SubDate Month')
plt.ylabel('num personne')
```

[14]:
```
Text(0, 0.5, 'num personne')
```



5.4 Users et competition participation

[15]:
```
comppart=competition_paricipation.dropna()
data2=pd.merge(users,comppart)
data2
```

[15]:

	UserID	FeatureX	Country	FeatureY	Points	UserDate Year	\
0	ID_N5LTBAPU	0	ID_DMRM	1	group 3	2	
1	ID_CLSFQBOS	0	ID_QO2	3	group 3	1	
2	ID_CLSFQBOS	0	ID_QO2	3	group 3	1	
3	ID_RE6T58Y4	0	ID_QO2	0	group 3	2	
4	ID_XJQQRJV3	0	ID_Z8BI	0	group 3	2	
...	...	...	...	...	...	...	
13962	ID_D4SARSC7	0	ID_5OWN	1	group 3	1	
13963	ID_B8VJJMWK	0	ID_QO2	3	group 3	2	

```
13964  ID_XAQGPGAZ         0   ID_Q02         3  group 3              2
13965  ID_1AO7PVP2         0   ID_Q02         3  group 3              2
13966  ID_3ZXJIREU         0   ID_Q02         3  group 3              1

       UserDate Month  UserDate Day_of_week   CompID PublicRank  \
0                   4                     4  ID_HJ9S    rank 10
1                   5                     4  ID_7MLO     rank 1
2                   5                     4  ID_LMFN     rank 1
3                  12                     3  ID_H4L8    rank 11
4                   9                     2  ID_H4L8     rank 7
...               ...                   ...      ...        ...
13962               5                     3  ID_LMFN    rank 11
13963               3                     4  ID_MPSN     rank 4
13964               3                     1  ID_MPSN     rank 4
13965               5                     5  ID_AWEI     rank 7
13966               7                     3  ID_BT9Z    rank 10

      Successful Submission Count  CompPartCreated Year  \
0                        count 10                     2
1                         count 9                     2
2                        count 10                     1
3                         count 8                     2
4                        count 10                     2
...                           ...                   ...
13962                    count 10                     1
13963                    count 10                     2
13964                    count 10                     2
13965                     count 6                     2
13966                    count 10                     1

       CompPartCreated Month  CompPartCreated Day_of_week
0                          4                           4
1                         10                           4
2                          5                           4
3                         12                           3
4                          9                           2
...                      ...                         ...
13962                      5                           5
13963                      3                           4
13964                      4                           4
13965                      6                           3
13966                      7                           3

[13967 rows x 14 columns]
```

[16]:
```
del data2['FeatureX']
```

[17]:
```
del data2['FeatureY']
```

[18]:
```
data2['comptime Year']=np.where(data2['UserDate Year']==data2['CompPartCreated␣
 ↪Year'],'1','0')
data2
```

[18]:

	UserID	Country	Points	UserDate Year	UserDate Month
0	ID_N5LTBAPU	ID_DMRM	group 3	2	4
1	ID_CLSFQB0S	ID_Q02	group 3	1	5
2	ID_CLSFQB0S	ID_Q02	group 3	1	5
3	ID_RE6T58Y4	ID_Q02	group 3	2	12
4	ID_XJQQRJV3	ID_Z8BI	group 3	2	9
...	...	...	...	...	...
13962	ID_D4SARSC7	ID_5OWN	group 3	1	5
13963	ID_B8VJJMWK	ID_Q02	group 3	2	3
13964	ID_XAQGPGAZ	ID_Q02	group 3	2	3
13965	ID_1AO7PVP2	ID_Q02	group 3	2	5
13966	ID_3ZXJIREU	ID_Q02	group 3	1	7

	UserDate Day_of_week	CompID	PublicRank	Successful Submission Count
0	4	ID_HJ9S	rank 10	count 10
1	4	ID_7MLO	rank 1	count 9
2	4	ID_LMFN	rank 1	count 10
3	3	ID_H4L8	rank 11	count 8
4	2	ID_H4L8	rank 7	count 10
...	...	...	...	...
13962	3	ID_LMFN	rank 11	count 10
13963	4	ID_MPSN	rank 4	count 10
13964	1	ID_MPSN	rank 4	count 10
13965	5	ID_AWEI	rank 7	count 6
13966	3	ID_BT9Z	rank 10	count 10

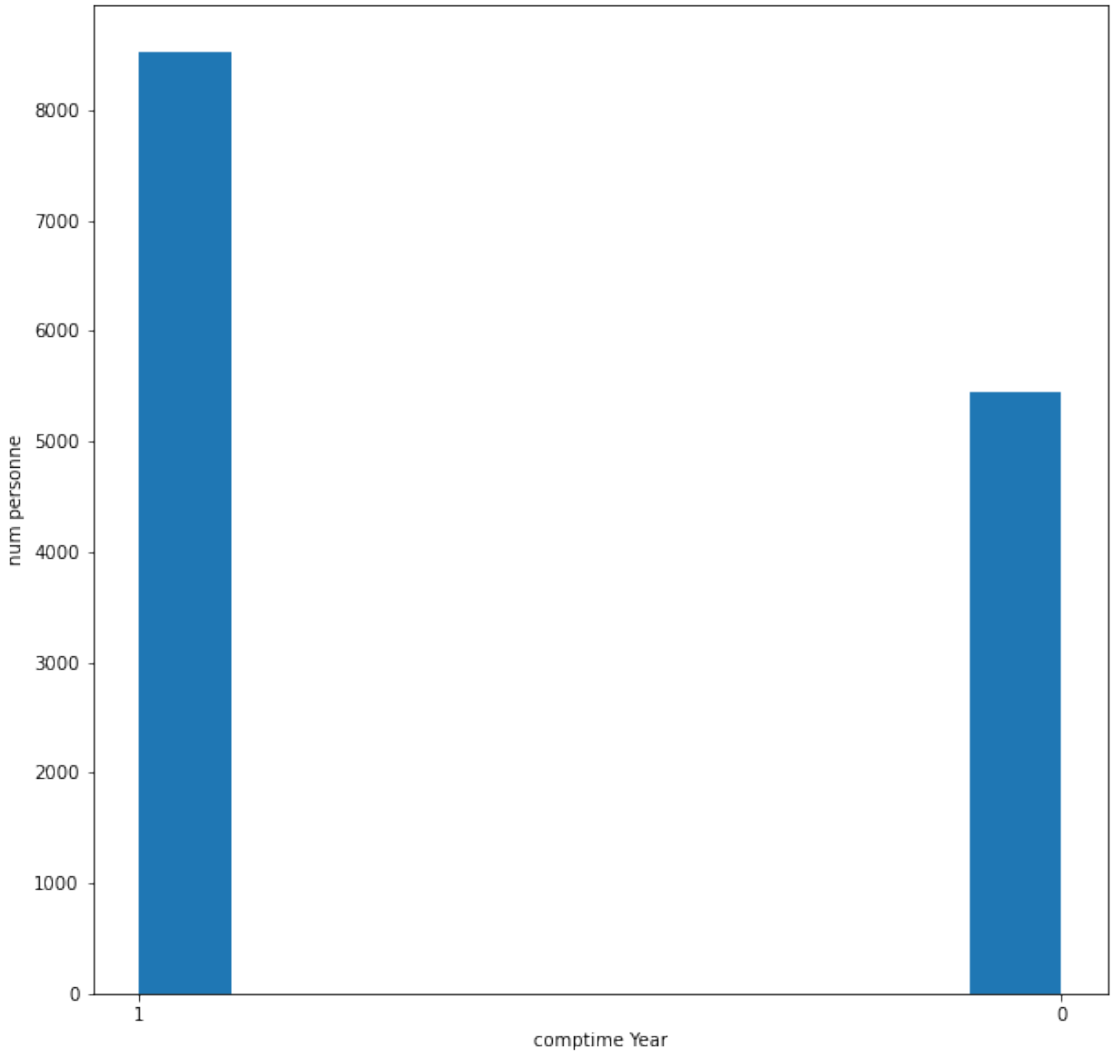
	CompPartCreated Year	CompPartCreated Month
0	2	4
1	2	10
2	1	5
3	2	12
4	2	9
...	...	...
13962	1	5
13963	2	3
13964	2	4
13965	2	6
13966	1	7

	CompPartCreated Day_of_week	comptime Year
0	4	1
1	4	0
2	4	1
3	3	1
4	2	1
...	...	...
13962	5	1
13963	4	1
13964	4	1
13965	3	1
13966	3	1

[13967 rows x 13 columns]

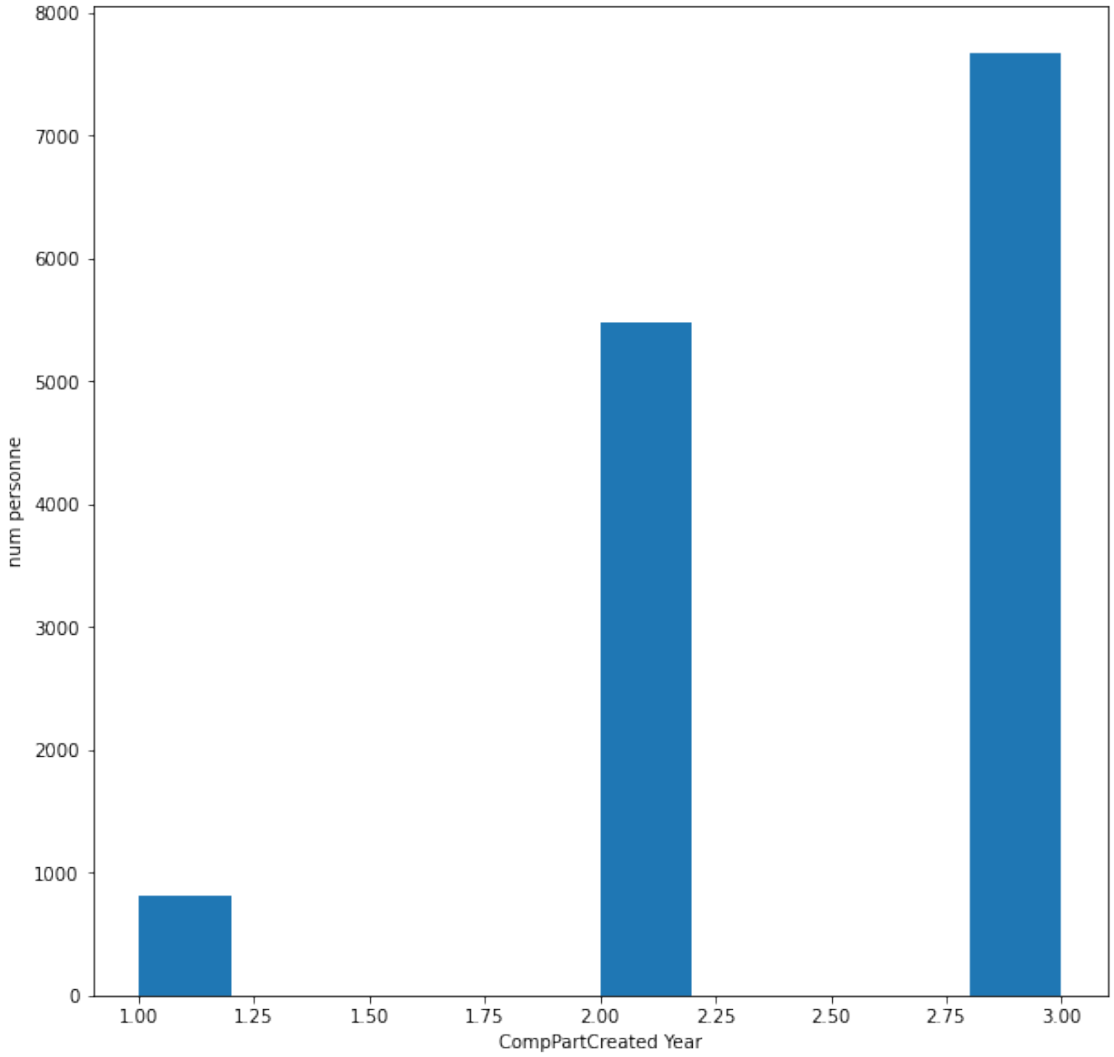
[19]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['comptime Year'])
plt.xlabel('comptime Year')
plt.ylabel('num personne')
#on remarque que il y a beaucoup de personnes participent a un deffis zindi dans le␣
 ↪meme année d'utilisation
```

[19]:
```
Text(0, 0.5, 'num personne')
```



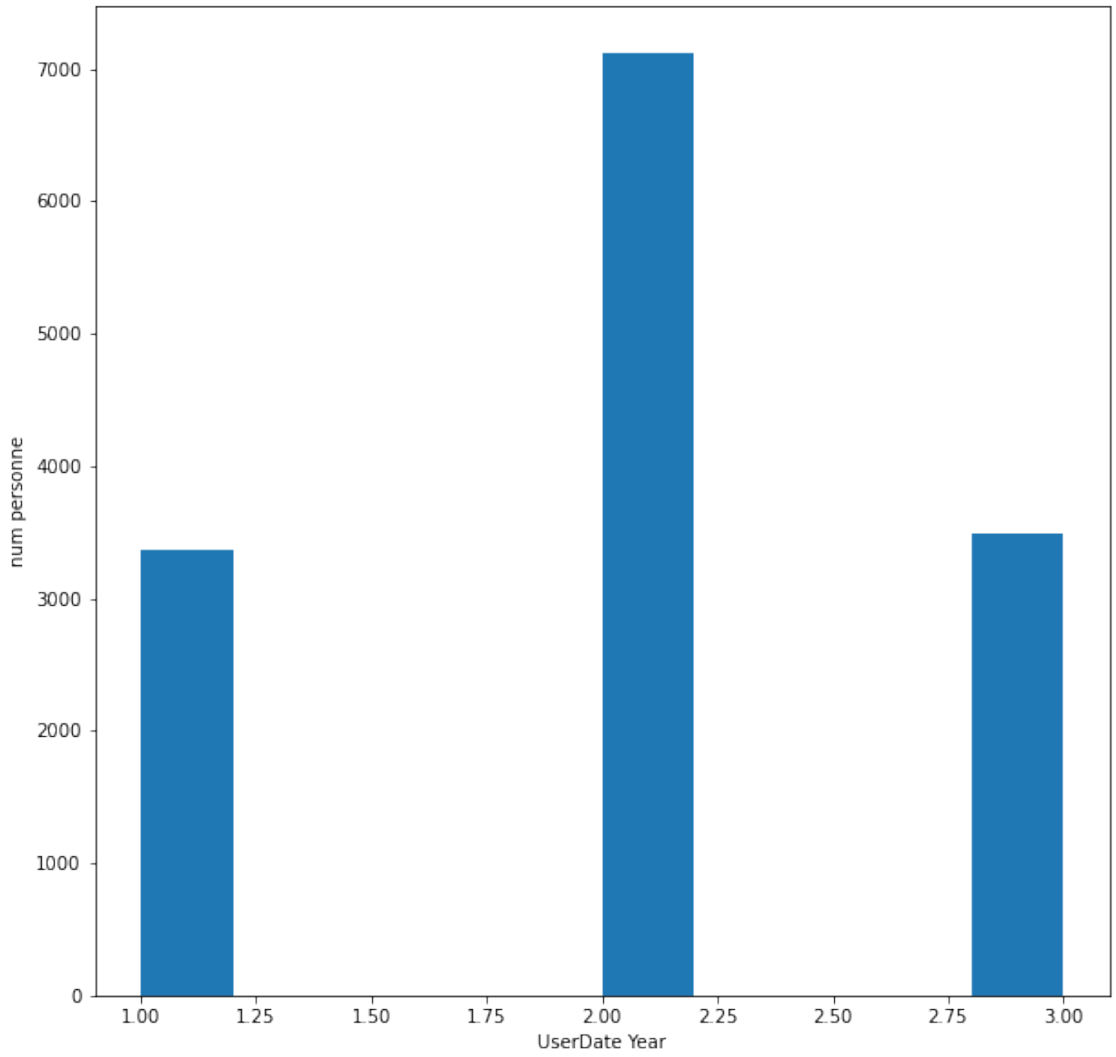
[20]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['CompPartCreated Year'])
plt.xlabel('CompPartCreated Year')
plt.ylabel('num personne')
```

[20]: Text(0, 0.5, 'num personne')



[21]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['UserDate Year'])
plt.xlabel('UserDate Year')
plt.ylabel('num personne')
```

[21]: Text(0, 0.5, 'num personne')



meme si le nombre du personnes qu'ils font la creation des comptes sur zindi est plus dans la 2 eme année mais ils participent plus dans la 3 eme années d'utilisation sur des competitions

[22]:
```
data2['comptime Month']=np.where(data2['UserDate Month']==data2['CompPartCreated␣
↪Month'],'1','0')
data2
```

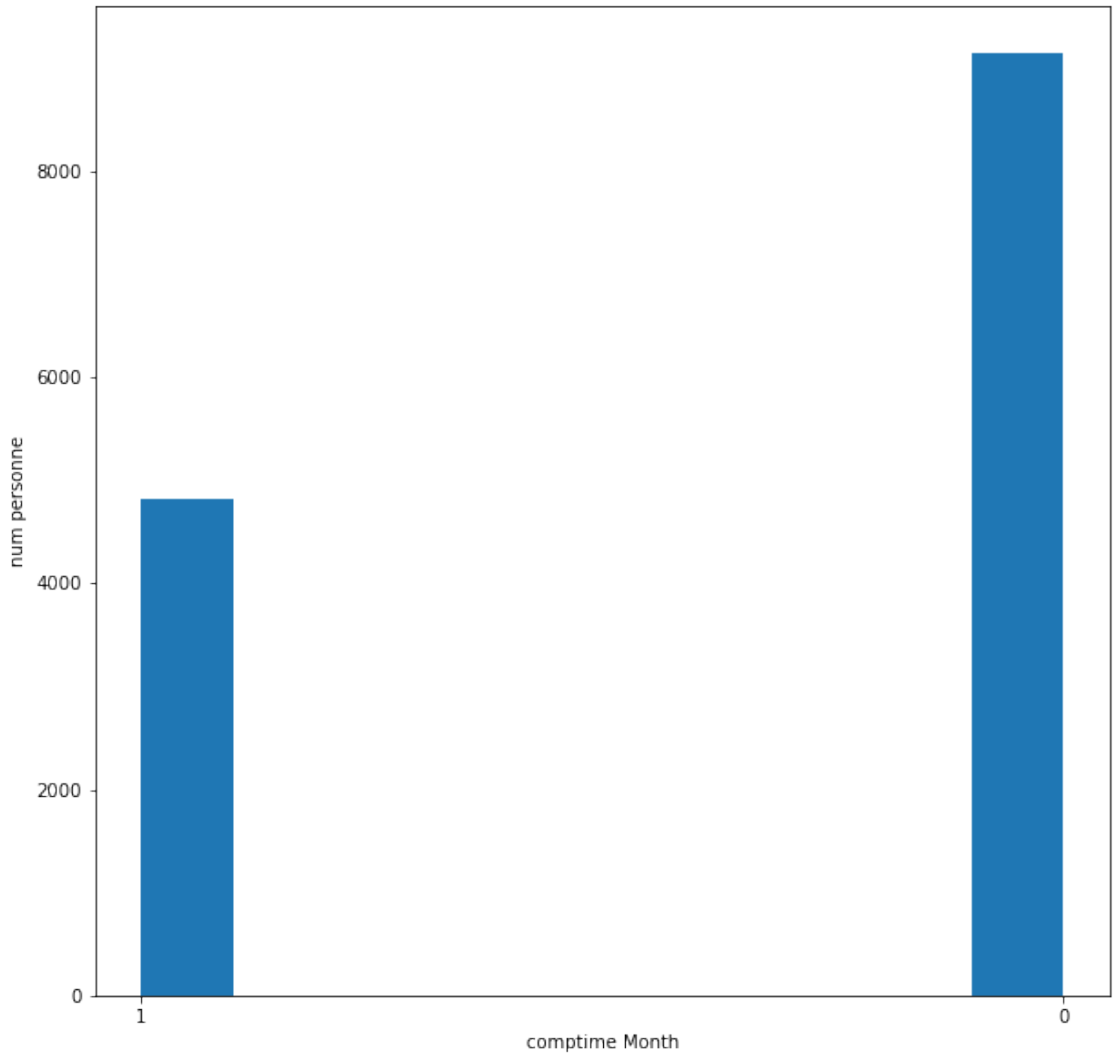
[22]:

	UserID	Country	Points	UserDate Year	UserDate Month	\
0	ID_N5LTBAPU	ID_DMRM	group 3	2	4	
1	ID_CLSFQBOS	ID_QO2	group 3	1	5	
2	ID_CLSFQBOS	ID_QO2	group 3	1	5	
3	ID_RE6T58Y4	ID_QO2	group 3	2	12	
4	ID_XJQQRJV3	ID_Z8BI	group 3	2	9	
...	...	...	...	...	...	
13962	ID_D4SARSC7	ID_5OWN	group 3	1	5	
13963	ID_B8VJJMWK	ID_QO2	group 3	2	3	
13964	ID_XAQGPGAZ	ID_QO2	group 3	2	3	
13965	ID_1AO7PVP2	ID_QO2	group 3	2	5	

```
13966  ID_3ZXJIREU   ID_Q02  group 3              1               7

       UserDate Day_of_week   CompID PublicRank Successful Submission Count  \
0                         4  ID_HJ9S    rank 10                    count 10
1                         4  ID_7MLO     rank 1                     count 9
2                         4  ID_LMFN     rank 1                    count 10
3                         3  ID_H4L8    rank 11                     count 8
4                         2  ID_H4L8     rank 7                    count 10
...                     ...      ...        ...                         ...
13962                     3  ID_LMFN    rank 11                    count 10
13963                     4  ID_MPSN     rank 4                    count 10
13964                     1  ID_MPSN     rank 4                    count 10
13965                     5  ID_AWEI     rank 7                     count 6
13966                     3  ID_BT9Z    rank 10                    count 10

       CompPartCreated Year  CompPartCreated Month  \
0                         2                      4
1                         2                     10
2                         1                      5
3                         2                     12
4                         2                      9
...                     ...                    ...
13962                     1                      5
13963                     2                      3
13964                     2                      4
13965                     2                      6
13966                     1                      7

       CompPartCreated Day_of_week comptime Year comptime Month
0                                4             1              1
1                                4             0              0
2                                4             1              1
3                                3             1              1
4                                2             1              1
...                            ...           ...            ...
13962                            5             1              1
13963                            4             1              1
13964                            4             1              0
13965                            3             1              0
13966                            3             1              1

[13967 rows x 14 columns]
```

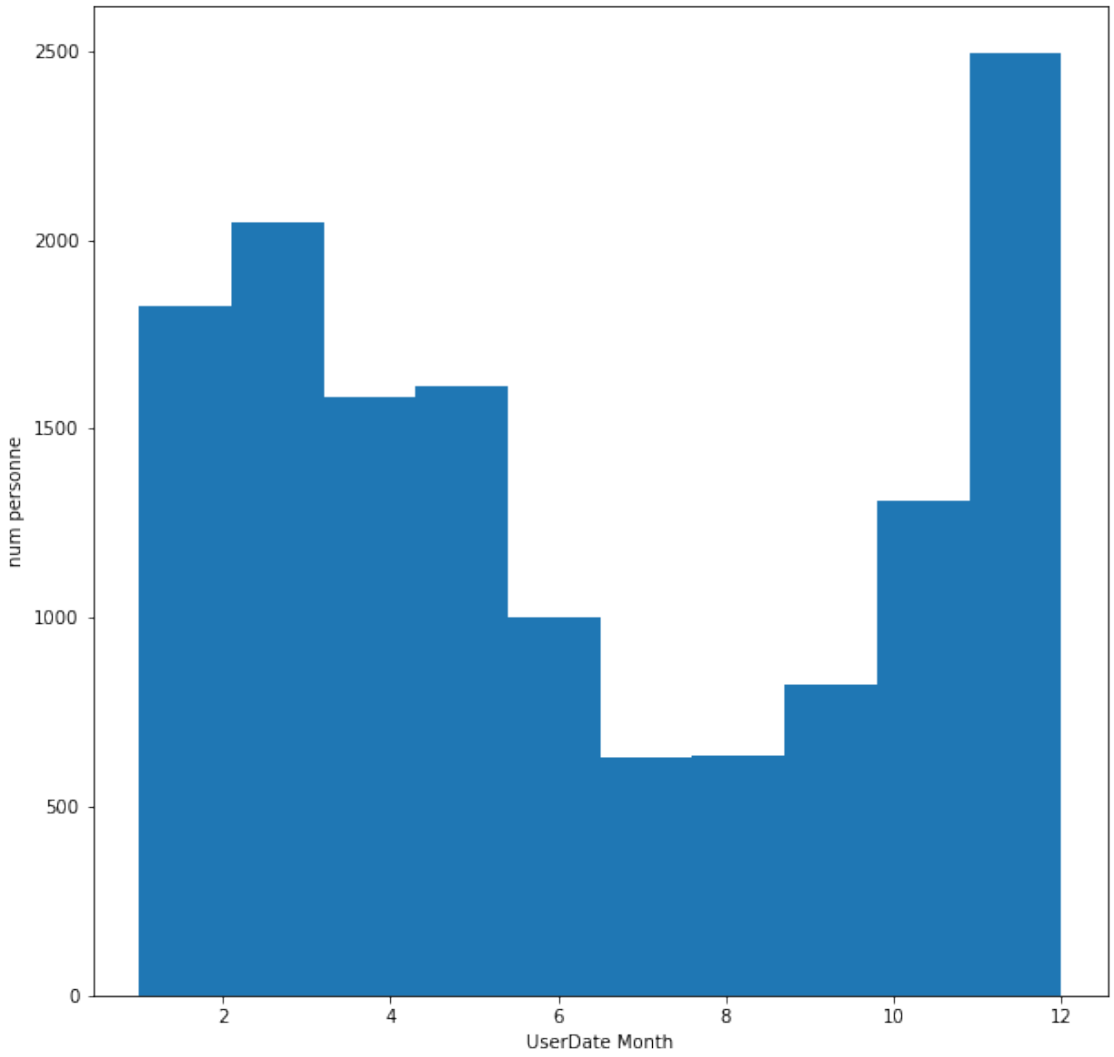
[23]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['comptime Month'])
plt.xlabel('comptime Month')
plt.ylabel('num personne')
# on remarque que le mois de visite zindi n'est pas le meme de comencer un defi
```

[23]:
```
Text(0, 0.5, 'num personne')
```



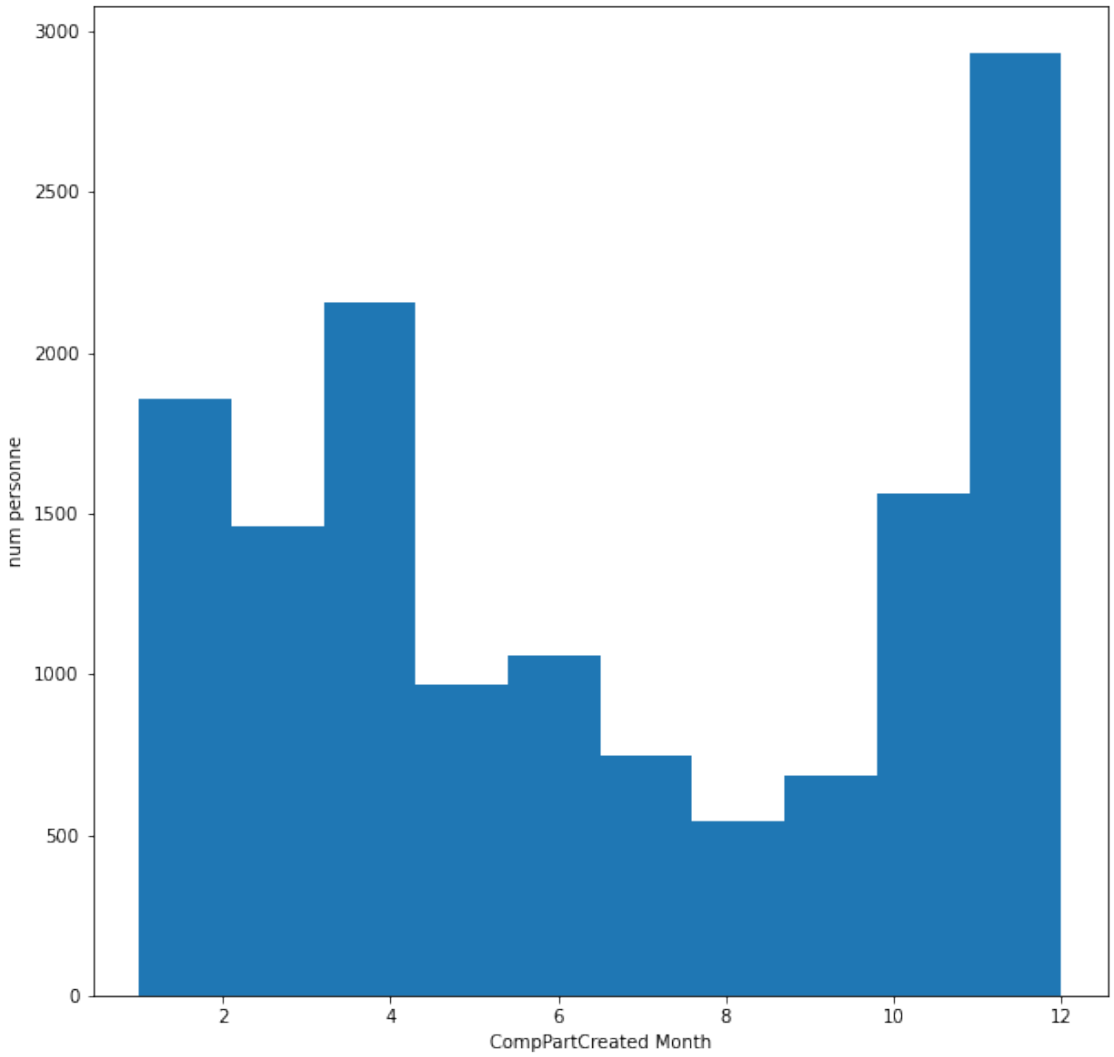
[24]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['UserDate Month'])
plt.xlabel('UserDate Month')
plt.ylabel('num personne')
```

[24]:
```
Text(0, 0.5, 'num personne')
```



[25]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 10))
plt.hist(data2['CompPartCreated Month'])
plt.xlabel('CompPartCreated Month')
plt.ylabel('num personne')
```

[25]:
```
Text(0, 0.5, 'num personne')
```


3000
2500
2000
1500
1000
500
0
num personne
2
4
6
8
10
12
CompPartCreated Month

Chapter 6

Modeling : Machine learning

dans cette notebook , on va faire la prediction et le calcul de score entre les features et label qu'on a, en utilisant plusieurs modeles comme knn et tree decision , logical regression , linear regression ..... 1- on distingue les features en utilisants tous les datas 2- on concate les features avec test 3- on applique les modeles

[143]:
```
import pandas as pd
import numpy as np
```

[144]:
```
comment_counter=pd.read_csv('comment_counter.csv')
Succeful_sub=pd.read_csv("Succeful_sub.csv")
submission_counter=pd.read_csv("submission_counter.csv")
discussions_counter=pd.read_csv("discussions_counter.csv")
test=pd.read_csv('test.csv')
train=pd.read_csv('train.csv')
```

[145]:
```
comment_counter.drop('UserMonthYear', axis = 1, inplace = True)
Succeful_sub.drop('UserMonthYear', axis = 1, inplace = True)
submission_counter.drop('UserMonthYear', axis = 1, inplace = True)
discussions_counter.drop('UserMonthYear', axis = 1, inplace = True)
```

[146]:
```
test
```

[146]:

	User_ID	month	year
0	ID_H1ELY25E	1	4
1	ID_H1ELY25E	2	4
2	ID_H1ELY25E	3	4
3	ID_463Q2BCO	1	4
4	ID_463Q2BCO	2	4
...	...	...	...
65218	ID_4XKWR8UN	3	4
65219	ID_L54061S5	3	4
65220	ID_I3C1N5RO	3	4
65221	ID_WWNR6I15	3	4
65222	ID_V69HAZHO	3	4

[65223 rows x 3 columns]

on a fait la concaténation des 3 colonnes User, month et year

[147]:
```
test['UserMonthYear'] = test['User_ID'].astype(str)+'_'+test['month'].
↪astype(str)+'_'+test['year'].astype(str)
test
```

[147]:

	User_ID	month	year	UserMonthYear
0	ID_H1ELY25E	1	4	ID_H1ELY25E_1_4
1	ID_H1ELY25E	2	4	ID_H1ELY25E_2_4
2	ID_H1ELY25E	3	4	ID_H1ELY25E_3_4
3	ID_463Q2BCO	1	4	ID_463Q2BCO_1_4
4	ID_463Q2BCO	2	4	ID_463Q2BCO_2_4
...	...	...	...	...
65218	ID_4XKWR8UN	3	4	ID_4XKWR8UN_3_4
65219	ID_L54061S5	3	4	ID_L54061S5_3_4
65220	ID_I3C1N5RO	3	4	ID_I3C1N5RO_3_4
65221	ID_WWNR6I15	3	4	ID_WWNR6I15_3_4
65222	ID_V69HAZHO	3	4	ID_V69HAZHO_3_4

[65223 rows x 4 columns]

6.1 Merge

on a ajouté à test les colonnes des autres datas dans le but de calculer le score et la prédiction

[148]:
```
test = test.merge(discussions_counter, on='User_ID')
```

[149]:
```
test = test.merge(submission_counter, on='User_ID')
test = test.merge(Succeful_sub, on='User_ID')
test = test.merge(comment_counter, on='User_ID')
```

[150]:
```
test.drop_duplicates(inplace = True)
```

[151]:
```
test
```

[151]:

	User_ID	month	year	UserMonthYear	discussions_counter	submission_counter	PublicRank	comment_counter
0	ID_IYC1NOTL	1	4	ID_IYC1NOTL_1_4	1	0	3	4
1	ID_IYC1NOTL	2	4	ID_IYC1NOTL_2_4	1	0	3	4
2	ID_IYC1NOTL	3	4	ID_IYC1NOTL_3_4	1	0	3	4
3	ID_F9RXMPBT	1	4	ID_F9RXMPBT_1_4	1	0	11	4
4	ID_F9RXMPBT	1	4	ID_F9RXMPBT_1_4	1	0	0	4
...	...	...	...	...	...			
35388972	ID_KK440U4N	3	4	ID_KK440U4N_3_4	1			
35388976	ID_KK440U4N	3	4	ID_KK440U4N_3_4	1			
35388978	ID_KK440U4N	3	4	ID_KK440U4N_3_4	1			
35388986	ID_KK440U4N	3	4	ID_KK440U4N_3_4	1			
35389004	ID_KK440U4N	3	4	ID_KK440U4N_3_4	1			

```
...                      ...         ...              ...
35388972                   0           2                9
35388976                   1          11                9
35388978                   1           0                9
35388986                   1           1                9
35389004                   1           2                9

[89427 rows x 8 columns]
```

6.2 Train

[152]:
```
train
```

[152]:
```
            User_ID  month  year  CompPart  Comment  Sub  Disc  Target
0       ID_XI7BAR4Y      8     3         0        0    0     0       0
1       ID_XI7BAR4Y      8     2         0        0    0     0       0
2       ID_XI7BAR4Y      9     2         0        0    0     0       0
3       ID_XI7BAR4Y      9     3         0        0    0     0       0
4       ID_XI7BAR4Y     10     3         0        0    0     0       0
...             ...    ...   ...       ...      ...  ...   ...     ...
259827  ID_MAP5X6D4     12     3         0        0    0     0       0
259828  ID_QHUAHU76     12     3         0        0    0     0       0
259829  ID_8IKU22O5     12     3         1        0    0     0       1
259830  ID_NHWCR1IY     12     3         0        0    0     0       0
259831  ID_XMD7EIYV     12     3         1        0    0     1       1

[259832 rows x 8 columns]
```

[153]:
```
y=train['Target']
```

[154]:
```
x=train[['month','year']]
```

[155]:
```
x_CompPart=train[['month','year','CompPart']]
```

[156]:
```
x_Comment=train[['month','year','CompPart','Comment']]
```

[157]:
```
x_Sub=train[['month','year','CompPart','Comment','Sub']]
```

[158]:
```
x_Disc=train[['month','year','CompPart','Comment','Sub','Disc']]
```

[ ]:
```

```

6.3 Train Score en utilisant knn

K Nearest Neighbor (KNN) est un algorithme très simple, facile à comprendre, polyvalent et l'un des meilleurs algorithmes d'apprentissage automatique. Algorithme KNN utilisé pour les problèmes de classification et de régression. Algorithme KNN basé sur une approche de similarité des caractéristiques. KNN est un algorithme d'apprentissage non paramétrique et paresseux

[159]:
```
from sklearn.neighbors import KNeighborsClassifier
```

[160]:
```
model = KNeighborsClassifier()
```

[161]:
```
x1=x_Disc.copy()
```

[162]:
```
model.fit(x1,y)
model.score(x1,y)
```

[162]:
```
0.9999769081560392
```

[ ]:
```

```

6.4 Test Model

[163]:
```
del test['User_ID']
```

[164]:
```
test=test.drop('UserMonthYear',axis=1)
```

[165]:
```
test
```

[165]:
```
          month  year  discussions_counter  submission_counter  PublicRank  \
0             1     4                    1                   0           3
1             2     4                    1                   0           3
2             3     4                    1                   0           3
3             1     4                    1                   0          11
4             1     4                    1                   0           0
...         ...   ...                  ...                 ...         ...
35388972      3     4                    1                   0           2
35388976      3     4                    1                   1          11
35388978      3     4                    1                   1           0
35388986      3     4                    1                   1           1
35389004      3     4                    1                   1           2

          comment_counter
0                       4
1                       4
2                       4
3                       4
4                       4
...                   ...
35388972                9
35388976                9
35388978                9
35388986                9
35389004                9

[89427 rows x 6 columns]
```

[166]:
```
y_pred=model.predict(test)
```

[167]:
```
predict=pd.DataFrame(model.predict(test))
predict
```

[167]:
```
        0
0       1
1       1
2       1
3       1
4       1
...    ..
89422   1
89423   1
89424   1
89425   1
89426   1

[89427 rows x 1 columns]
```

[168]:
```
test1=pd.read_csv('test.csv')
test1['UserMonthYear'] = test1['User_ID'].astype(str)+'_'+test1['month'].
 ↪astype(str)+'_'+test1['year'].astype(str)
test1
```

[168]:
```
            User_ID  month  year    UserMonthYear
0      ID_H1ELY25E      1     4  ID_H1ELY25E_1_4
1      ID_H1ELY25E      2     4  ID_H1ELY25E_2_4
2      ID_H1ELY25E      3     4  ID_H1ELY25E_3_4
3      ID_463Q2BCO      1     4  ID_463Q2BCO_1_4
4      ID_463Q2BCO      2     4  ID_463Q2BCO_2_4
...            ...    ...   ...              ...
65218  ID_4XKWR8UN      3     4  ID_4XKWR8UN_3_4
65219  ID_L54061S5      3     4  ID_L54061S5_3_4
65220  ID_I3C1N5RO      3     4  ID_I3C1N5RO_3_4
65221  ID_WWNR6I15      3     4  ID_WWNR6I15_3_4
65222  ID_V69HAZHO      3     4  ID_V69HAZHO_3_4

[65223 rows x 4 columns]
```

[169]:
```
idd=pd.DataFrame(test1['UserMonthYear'])
```

[170]:
```
predict['UserMonthYear']=idd
```

[171]:
```
predict.columns=['Target','UserMonthYear']
```

[172]:
```
predict = np.fliplr(predict)
```

[173]:
```
predict=pd.DataFrame(predict)
```

[174]:
```
predict.columns=['UserMonthYear','Target']
```

[175]:
```
predict
```

[175]:
```
          UserMonthYear Target
0      ID_H1ELY25E_1_4      1
1      ID_H1ELY25E_2_4      1
2      ID_H1ELY25E_3_4      1
```

```
3      ID_463Q2BCO_1_4      1
4      ID_463Q2BCO_2_4      1
...                ...    ...
89422              NaN      1
89423              NaN      1
89424              NaN      1
89425              NaN      1
89426              NaN      1

[89427 rows x 2 columns]
```

[176]:
```
predict.dropna(subset = ["UserMonthYear"], inplace=True)
predict
```

[176]:
```
          UserMonthYear Target
0      ID_H1ELY25E_1_4      1
1      ID_H1ELY25E_2_4      1
2      ID_H1ELY25E_3_4      1
3      ID_463Q2BCO_1_4      1
4      ID_463Q2BCO_2_4      1
...                ...    ...
65218  ID_4XKWR8UN_3_4      1
65219  ID_L54061S5_3_4      1
65220  ID_I3C1N5RO_3_4      1
65221  ID_WWNR6I15_3_4      1
65222  ID_V69HAZHO_3_4      1

[65223 rows x 2 columns]
```

[177]:
```
predict.to_csv("PredictV2.csv", index = False)
```

6.5 test de decision tree

on utilise les arbres de classification pour expliquer et/ou prédire l'appartenance d'objets (observations, individus) à une classe (ou modalité ou catégorie) d'une variable qualitative, sur la base de variables explicatives quantitatives et/ou qualitatives. nous utilisons la methode decision tree de SK-learn pour mieux visualiser la prediction et la correlation entre les colonnes

train_test_split : cette fonction divise notre data train en 2 datas train et test : 70% train et 30% test. ("juste pour le tester")

[178]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(x_Disc, y, test_size=0.344153,␣
 ↪random_state=0)
```

[179]:
```
print(X_train)
```

```
        month  year  CompPart  Comment  Sub  Disc
167936     11     3         0        0    0     0
76889       6     3         0        0    0     0
92052       7     3         0        0    0     0
248961      6     2         0        0    0     0
256421     10     3         1        0    1     0
```

```
...       ...   ...        ...       ...  ...   ...
176963     12     1          0         0    0     0
117952      2     3          0         0    0     0
173685      7     3          0         0    0     0
43567       9     2          0         0    0     0
199340      3     2          1         0    1     0

[170410 rows x 6 columns]
```

[180]:
```
from sklearn.tree import DecisionTreeClassifier
```

[181]:
```
tree = DecisionTreeClassifier()
```

[182]:
```
fit=tree.fit(X_train, y_train)
```

[183]:
```
y_pred=tree.predict(X_test)
```

[184]:
```
tree_predict=pd.DataFrame(tree.predict(X_test))
tree_predict
```

[184]:
```
       0
0      0
1      0
2      0
3      0
4      0
...   ..
89417  0
89418  0
89419  1
89420  0
89421  0

[89422 rows x 1 columns]
```

[185]:
```
idd
```

[185]:
```
          UserMonthYear
0      ID_H1ELY25E_1_4
1      ID_H1ELY25E_2_4
2      ID_H1ELY25E_3_4
3      ID_463Q2BCO_1_4
4      ID_463Q2BCO_2_4
...                ...
65218  ID_4XKWR8UN_3_4
65219  ID_L54061S5_3_4
65220  ID_I3C1N5RO_3_4
65221  ID_WWNR6I15_3_4
65222  ID_V69HAZHO_3_4

[65223 rows x 1 columns]
```

[186]:
```
tree_predict['UserMonthYear']=idd

tree_predict.columns=['Target','UserMonthYear']
tree_predict
```

[186]:
```
       Target    UserMonthYear
0           0  ID_H1ELY25E_1_4
1           0  ID_H1ELY25E_2_4
2           0  ID_H1ELY25E_3_4
3           0  ID_463Q2BCO_1_4
4           0  ID_463Q2BCO_2_4
...       ...              ...
89417       0              NaN
89418       0              NaN
89419       1              NaN
89420       0              NaN
89421       0              NaN

[89422 rows x 2 columns]
```

[187]:
```
tree_predict.columns=['Target','UserMonthYear']

tree_predict = np.fliplr(tree_predict)

tree_predict=pd.DataFrame(tree_predict)

tree_predict.columns=['UserMonthYear','Target']

tree_predict
```

[187]:
```
         UserMonthYear Target
0      ID_H1ELY25E_1_4      0
1      ID_H1ELY25E_2_4      0
2      ID_H1ELY25E_3_4      0
3      ID_463Q2BCO_1_4      0
4      ID_463Q2BCO_2_4      0
...                ...    ...
89417              NaN      0
89418              NaN      0
89419              NaN      1
89420              NaN      0
89421              NaN      0

[89422 rows x 2 columns]
```

[188]:
```
tree_predict.dropna(subset = ["UserMonthYear"], inplace=True)
tree_predict

tree_predict.to_csv("TreePredict.csv", index = False)
```

6.6 test tree decision

par contre ici on travaille sur test et train donné et notre propres colonnes

[189]:
```
fite=tree.fit(x_Disc, y)
fite
```

[189]:
```
DecisionTreeClassifier()
```

[190]:
```
test_pred=tree.predict(test)
test_pred
```

[190]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

[191]:
```
test_pred=pd.DataFrame(test_pred)
test_pred
```

[191]:

	0
0	1
1	1
2	1
3	1
4	1
...	..
89422	1
89423	1
89424	1
89425	1
89426	1

[89427 rows x 1 columns]

[192]:
```
test_pred.value_counts()
```

[192]:
```
1    89427
dtype: int64
```

[193]:
```
test_pred['UserMonthYear']=idd

test_pred.columns=['Target','UserMonthYear']
test_pred

test_pred.columns=['Target','UserMonthYear']

test_pred = np.fliplr(test_pred)

test_pred=pd.DataFrame(test_pred)

test_pred.columns=['UserMonthYear','Target']



test_pred.dropna(subset = ["UserMonthYear"], inplace=True)


test_pred.to_csv("TreetestPred.csv", index = False)
```

ici on va a chaque fois ajouter une seule colonne de train

6.6.1 month year

[194]:
```
x=train[['month','year']]
```

[195]:
```
test_my=test[['month','year']]
```

[196]:
```
m_y=tree.fit(x, y)
m_y
```

[196]:
```
DecisionTreeClassifier()
```

[197]:
```
test_pred_my=tree.predict(test_my)
test_pred_my
```

[197]:
```
array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

[198]:
```
test_pred_my=pd.DataFrame(test_pred_my)
test_pred_my
```

[198]:
```
       0
0      0
1      0
2      0
3      0
4      0
...   ..
89422  0
89423  0
89424  0
89425  0
89426  0

[89427 rows x 1 columns]
```

[199]:
```
test_pred_my.value_counts()
```

[199]:
```
0    89427
dtype: int64
```

[200]:
```
test_pred_my['UserMonthYear']=idd

test_pred_my.columns=['Target','UserMonthYear']

test_pred_my.columns=['Target','UserMonthYear']

test_pred_my = np.fliplr(test_pred_my)

test_pred_my=pd.DataFrame(test_pred_my)

test_pred_my.columns=['UserMonthYear','Target']
```

```
test_pred_my.dropna(subset = ["UserMonthYear"], inplace=True)


test_pred_my.to_csv("TreetestPredmonthyear.csv", index = False)
```

6.6.2 month year compart

[201]:
```
x_CompPart=train[['month','year','CompPart']]
```

[202]:
```
y
```

[202]:
```
0         0
1         0
2         0
3         0
4         0
         ..
259827    0
259828    0
259829    1
259830    0
259831    1
Name: Target, Length: 259832, dtype: int64
```

[203]:
```
test_comp=test[['month','year','PublicRank']]
```

[204]:
```
m_y_comp=tree.fit(x_CompPart, y)
m_y_comp
```

[204]:
```
DecisionTreeClassifier()
```

[205]:
```
test_pred_my_comp=tree.predict(test_comp)
test_pred_my_comp
```

[205]:
```
array([1, 1, 1, ..., 0, 1, 1], dtype=int64)
```

[206]:
```
test_pred_my_comp=pd.DataFrame(test_pred_my_comp)
test_pred_my_comp
```

[206]:
```
       0
0      1
1      1
2      1
3      1
4      0
...   ..
89422  1
89423  1
89424  0
89425  1
89426  1

[89427 rows x 1 columns]
```

[207]:
```
test_pred_my_comp.value_counts()
```

[207]:
```
1    69759
0    19668
dtype: int64
```

[208]:
```
test_pred_my_comp['UserMonthYear']=idd

test_pred_my_comp.columns=['Target','UserMonthYear']

test_pred_my_comp.columns=['Target','UserMonthYear']

test_pred_my_comp = np.fliplr(test_pred_my_comp)

test_pred_my_comp=pd.DataFrame(test_pred_my_comp)

test_pred_my_comp.columns=['UserMonthYear','Target']



test_pred_my_comp.dropna(subset = ["UserMonthYear"], inplace=True)


test_pred_my_comp.to_csv("TreetestPredMonthYearComp.csv", index = False)
```

6.6.3 month year comp comment

[209]:
```
x_Comment=train[['month','year','CompPart','Comment']]
```

[210]:
```
test_comment=test[['month','year','PublicRank','comment_counter']]
```

[211]:
```
m_y_comment=tree.fit(x_Comment, y)
m_y_comment
```

[211]:
```
DecisionTreeClassifier()
```

[212]:
```
test_pred_my_comment=tree.predict(test_comment)
test_pred_my_comment
```

[212]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

[213]:
```
test_pred_my_comment=pd.DataFrame(test_pred_my_comment)
test_pred_my_comment
```

[213]:

	0
0	1
1	1
2	1
3	1
4	1
...	..
89422	1
89423	1

```
89424  1
89425  1
89426  1

[89427 rows x 1 columns]
```

[214]:
```
test_pred_my_comment.value_counts()
```

[214]:
```
1    89427
dtype: int64
```

[215]:
```
test_pred_my_comment['UserMonthYear']=idd

test_pred_my_comment.columns=['Target','UserMonthYear']

test_pred_my_comment.columns=['Target','UserMonthYear']

test_pred_my_comment = np.fliplr(test_pred_my_comment)

test_pred_my_comment=pd.DataFrame(test_pred_my_comment)

test_pred_my_comment.columns=['UserMonthYear','Target']



test_pred_my_comment.dropna(subset = ["UserMonthYear"], inplace=True)


test_pred_my_comment.to_csv("TreetestPredMonthYearCompComment.csv", index = False)
```

6.6.4 month year comp comment submission

[216]:
```
x_Sub=train[['month','year','CompPart','Comment','Sub']]
```

[217]:
```
test_sub=test[['month','year','PublicRank','comment_counter','submission_counter']]
```

[218]:
```
m_y_sub=tree.fit(x_Sub, y)
m_y_sub
```

[218]:
```
DecisionTreeClassifier()
```

[219]:
```
test_pred_sub=tree.predict(test_sub)
test_pred_sub
```

[219]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

[220]:
```
test_pred_sub=pd.DataFrame(test_pred_sub)
test_pred_sub
```

[220]:
```
       0
0      1
1      1
```

```
2      1
3      1
4      1
...   ..
89422  1
89423  1
89424  1
89425  1
89426  1

[89427 rows x 1 columns]
```

[221]:
```
test_pred_sub.value_counts()
```

[221]:
```
1    89427
dtype: int64
```

[222]:
```
test_pred_sub['UserMonthYear']=idd

test_pred_sub.columns=['Target','UserMonthYear']

test_pred_sub.columns=['Target','UserMonthYear']

test_pred_sub = np.fliplr(test_pred_sub)

test_pred_sub=pd.DataFrame(test_pred_sub)

test_pred_sub.columns=['UserMonthYear','Target']

test_pred_sub.dropna(subset = ["UserMonthYear"], inplace=True)


test_pred_sub.to_csv("TreetestPredMonthYearCompCommentSub.csv", index = False)
```

6.7 LogisticRegression

la régression logistique est courante et constitue une méthode de régression utile pour résoudre le problème de classification binaire.

La régression logistique est l'un des algorithmes d'apprentissage automatique les plus simples et les plus couramment utilisés pour la classification à deux classes. Il est facile à mettre en œuvre et peut être utilisé comme référence pour tout problème de classification binaire. La régression logistique décrit et estime la relation entre une variable binaire dépendante et des variables indépendantes.

[223]:
```
from sklearn.linear_model import LogisticRegression
```

[224]:
```
log = LogisticRegression()
```

[225]:
```
log.fit(x_Disc, y)
log.score(x_Disc,y)
```

[225]:
```
1.0
```

[226]:
```
log_pred=log.predict(test)
log_pred
```

[226]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

[227]:
```
log_pred=pd.DataFrame(log_pred)
log_pred
```

[227]:

	0
0	1
1	1
2	1
3	1
4	1
...	..
89422	1
89423	1
89424	1
89425	1
89426	1

[89427 rows x 1 columns]

[228]:
```
log_pred.value_counts()
```

[228]:
```
1    89427
dtype: int64
```

[229]:
```
log_pred['UserMonthYear']=idd

log_pred.columns=['Target','UserMonthYear']

log_pred.columns=['Target','UserMonthYear']

log_pred = np.fliplr(log_pred)

log_pred=pd.DataFrame(log_pred)

log_pred.columns=['UserMonthYear','Target']

log_pred.dropna(subset = ["UserMonthYear"], inplace=True)


log_pred.to_csv("logPred.csv", index = False)
```

[ ]:
```

```

6.8 month year log

[230]:
```
x=train[['month','year']]
```

[231]:
```
test_my=test[['month','year']]
```

[232]:
```
log_my=log.fit(x,y)
log_my
log.score(x,y)
```

[232]:
```
0.8602712521937251
```

[233]:
```
log_pred_my=log.predict(test_my)
log_pred_my
```

[233]:
```
array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

[234]:
```
log_pred_my=pd.DataFrame(log_pred_my)
log_pred_my
```

[234]:

	0
0	0
1	0
2	0
3	0
4	0
...	..
89422	0
89423	0
89424	0
89425	0
89426	0

[89427 rows x 1 columns]

[235]:
```
log_pred_my.value_counts()
```

[235]:
```
0    89427
dtype: int64
```

[236]:
```
log_pred_my['UserMonthYear']=idd

log_pred_my.columns=['Target','UserMonthYear']

log_pred_my.columns=['Target','UserMonthYear']

log_pred_my = np.fliplr(log_pred_my)

log_pred_my=pd.DataFrame(log_pred_my)

log_pred_my.columns=['UserMonthYear','Target']

log_pred_my.dropna(subset = ["UserMonthYear"], inplace=True)

log_pred_my.to_csv("logPredMonthYear.csv", index = False)
```

6.9 month year comppart

[237]:
```
x_CompPart=train[['month','year','CompPart']]
```

[238]:
```
test_comp=test[['month','year','PublicRank']]
```

[239]:
```
log_comp=log.fit(x_CompPart,y)
log_comp
log.score(x_CompPart,y)
```

[239]:
```
0.9811378121247575
```

[240]:
```
log_pred_comp=pd.DataFrame(log.predict(test_comp))
log_pred_comp
```

[240]:
```
       0
0      1
1      1
2      1
3      1
4      0
...   ..
89422  1
89423  1
89424  0
89425  1
89426  1

[89427 rows x 1 columns]
```

[241]:
```
log_pred_comp.value_counts()
```

[241]:
```
1    69759
0    19668
dtype: int64
```

[242]:
```
log_pred_comp['UserMonthYear']=idd

log_pred_comp.columns=['Target','UserMonthYear']

log_pred_comp.columns=['Target','UserMonthYear']

log_pred_comp = np.fliplr(log_pred_comp)

log_pred_comp=pd.DataFrame(log_pred_comp)

log_pred_comp.columns=['UserMonthYear','Target']

log_pred_comp.dropna(subset = ["UserMonthYear"], inplace=True)


log_pred_comp.to_csv("logPredMonthYearComp.csv", index = False)
```

6.10 month year comp comment

[243]:
```
x_Comment=train[['month','year','CompPart','Comment']]
```

[244]:
```
test_comment=test[['month','year','PublicRank','comment_counter']]
```

[245]:
```
log_comment=log.fit(x_Comment,y)
log_comment
log.score(x_Comment,y)
```

[245]:
```
0.9852789494750455
```

[246]:
```
log_pred_comment=pd.DataFrame(log.predict(test_comment))
log_pred_comment
```

[246]:
```
       0
0      1
1      1
2      1
3      1
4      1
...   ..
89422  1
89423  1
89424  1
89425  1
89426  1

[89427 rows x 1 columns]
```

[247]:
```
log_pred_comment.value_counts()
```

[247]:
```
1    89427
dtype: int64
```

[248]:
```
log_pred_comment['UserMonthYear']=idd

log_pred_comment.columns=['Target','UserMonthYear']

log_pred_comment.columns=['Target','UserMonthYear']

log_pred_comment = np.fliplr(log_pred_comment)

log_pred_comment=pd.DataFrame(log_pred_comment)

log_pred_comment.columns=['UserMonthYear','Target']

log_pred_comment.dropna(subset = ["UserMonthYear"], inplace=True)

log_pred_comment.to_csv("logPredMonthYearCompComment.csv", index = False)
```

6.11 month year comp comment submission

[249]:
```
x_Sub=train[['month','year','CompPart','Comment','Sub']]
```

[250]:
```
test_sub=test[['month','year','PublicRank','comment_counter','submission_counter']]
```

[251]:
```
log_sub=log.fit(x_Sub,y)
log_sub
log.score(x_Sub,y)
```

[251]:
```
0.9981834416084239
```

[252]:
```
log_pred_sub=pd.DataFrame(log.predict(test_sub))
log_pred_sub
```

[252]:
```
       0
0      1
1      1
2      1
3      1
4      1
...   ..
89422  1
89423  1
89424  1
89425  1
89426  1

[89427 rows x 1 columns]
```

[253]:
```
log_pred_sub.value_counts()
```

[253]:
```
1    89427
dtype: int64
```

[254]:
```
log_pred_sub['UserMonthYear']=idd

log_pred_sub.columns=['Target','UserMonthYear']

log_pred_sub.columns=['Target','UserMonthYear']

log_pred_sub = np.fliplr(log_pred_sub)

log_pred_sub=pd.DataFrame(log_pred_sub)

log_pred_sub.columns=['UserMonthYear','Target']

log_pred_sub.dropna(subset = ["UserMonthYear"], inplace=True)


log_pred_sub.to_csv("logPredMonthYearCompCommentSub.csv", index = False)
```

[ ]:
```

```

6.12 GaussianNB

Naive Bayes est l'algorithme de classification le plus simple et le plus rapide, qui convient à une grande quantité de données. Le classificateur Naive Bayes est utilisé avec succès dans diverses applications telles que le filtrage du courrier indésirable, la classification de texte, l'analyse des sentiments et les systèmes de recommandation. Il utilise le théorème de probabilité de Bayes pour la prédiction de classe inconnue.

[255]:
```
from sklearn.naive_bayes import GaussianNB
```

[256]:
```
guass = GaussianNB()
guass.fit(x_Disc, y)
guass.score(x_Disc,y)
```

[256]:
```
1.0
```

[257]:
```
guass_pred = pd.DataFrame(guass.predict(test))
guass_pred
```

[257]:
```
        0
0       1
1       1
2       1
3       1
4       1
...    ..
89422   1
89423   1
89424   1
89425   1
89426   1

[89427 rows x 1 columns]
```

[258]:
```
guass_pred.value_counts()
```

[258]:
```
1    89427
dtype: int64
```

[259]:
```
guass_pred['UserMonthYear']=idd

guass_pred.columns=['Target','UserMonthYear']

guass_pred.columns=['Target','UserMonthYear']

guass_pred = np.fliplr(guass_pred)

guass_pred=pd.DataFrame(guass_pred)

guass_pred.columns=['UserMonthYear','Target']

guass_pred.dropna(subset = ["UserMonthYear"], inplace=True)

guass_pred.to_csv("guassPred.csv", index = False)
```

[ ]:

6.13 linear regression

pour cette fois nous avons essayé de travailler avec la régression linéaire
pour la partie train, nous avons importé $LinearRegressiondesklearn.linear_model$
puis nous avons applique fit et score sur nos x et y

pour la partie test on a supprimé la colonne user id afin de n'avoir que des colonnes avec des nombres puis nous avons commencé notre prédiction avec model.predict(test) pour aboutir à un vecteur de 0 et 1 nous l'avons donc transformé en ensemble de données avec (dataset) pd.DataFrame(model.predict(test)) puis nous avons créé une fonction qui renvoie 1 si la prédiction >1 pour le premier subbmition , >2 pour la deuxième et >5 pour le troisième sinon elle renvoie 0 à la fin, nous avons transformé notre ensemble de données final en csv afin de le soumettre

[260]:
```
from sklearn.linear_model import LinearRegression
```

[261]:
```
reg = LinearRegression()
```

[262]:
```
model.fit(x_Disc,y)
model.score(x_Disc,y)
```

[262]:
```
0.9999769081560392
```

[263]:
```
reg_pred=model.predict(test)
reg_pred
```

[263]:
```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

[264]:
```
reg_predd=pd.DataFrame(model.predict(test))
```

[265]:
```
reg_predd.to_csv("regPred.csv", index = False)
```

[266]:
```
for i in range(len(reg_pred)):
    if reg_pred[i] >= 5:
        reg_pred[i] = 1
    else:
        reg_pred[i] = 0
```

[267]:
```
reg_pred1=pd.DataFrame(reg_pred)
reg_pred1.astype(int)
```

[267]:

	0
0	0
1	0
2	0
3	0
4	0
...	..
89422	0
89423	0
89424	0

```
89425  0
89426  0

[89427 rows x 1 columns]
```

[268]:
```
reg_predd.value_counts()
```

[268]:
```
1    89427
dtype: int64
```

[269]:
```
reg_pred1['UserMonthYear']=idd

reg_pred1.columns=['Target','UserMonthYear']

reg_pred1.columns=['Target','UserMonthYear']

reg_pred1 = np.fliplr(reg_pred1)

reg_pred1=pd.DataFrame(reg_pred1)

reg_pred1.columns=['UserMonthYear','Target']

reg_pred1.dropna(subset = ["UserMonthYear"], inplace=True)

reg_pred1.to_csv("regPred1.csv", index = False)
```

[ ]:

Chapter 7

Conclusion

Afin de connaître les utilisateurs qui vont rester actives Le mois suivant on a commencé par l'analyse exploratoire de nos donnes par plusieurs plots pour mieu comprendre nos données qui sont composées par 8 datasets qu'on aissaye par la suite de les nettoyer et d'en extraire ce que nous interesse et nous aiderons le plus. Puis on a essayée de creer notre propre nouvelle dataset en les fusionnant tous ensemble.

Par la suite on a entamé la phase de modeling (train, fit, test, predict) par l'exploit plusieurs models afin de sortir avec celui qui nous rend le score des predictions le plus pertinent (KNN avec 0.509958266790094) lors de nos soumissions sur la plate-forme zindi

Chapter 8

Sites visités

https://scikit-learn.org/stable/
https://datascientest.com/machine-learning-tout-savoir
https://www.youtube.com/watch?v=EUD07IiviJg&list=PLO_fdPEVlfKqUF5BPKjGSh7aV9aBshrpY

Merci beaucoup

Manai Elyes