Base Model

August 22, 2018

1 Firts attempts & Base Model

```
In [1]: from pathlib import Path
        import pandas as pd
        import numpy as np
        from datetime import datetime
        import time
        import matplotlib.pyplot as plt
        %matplotlib inline
        #%pylab inline
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
        from sklearn.linear_model import LogisticRegression
        from sklearn.model_selection import cross_val_score, GridSearchCV
        from sklearn.decomposition import PCA
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.metrics import confusion_matrix
        import itertools
        from sklearn.preprocessing import LabelEncoder
        from sklearn.preprocessing import LabelBinarizer
        from sklearn.preprocessing import OneHotEncoder
```

```
In [2]: path_project = Path.home() / Path('Google Drive/Felix')
        path_data = path_project / Path("data")
```

```
In [3]: # loading cdv data
        file = path_data / Path("felix.csv")
        with Path.open(file, 'rb') as fp:
            cdv = pd.read_csv(fp,  encoding='cp1252',low_memory=False)
```

1.1 1) Feature engineering & selection

```
In [4]: df = cdv.loc[:,["JUSTICE","RELEG",
                        "CONFPUB","CONFENTR","CONFASSO","CONFPOLI","CONFBANK",
                        "CONFPRES","CONFECOL","CONFKEUF",
                        "TRANSFST","PROGRAD","RADIQUOI","ORDLIB",
                        "CONFMEFI","PREOCCU1","PREOCCU2",
                        "RADI1","RADI2","RADI3",
```

```
                "OPICULT",
                "COMMU1","COMMU2","COMMU3","COMMU4",
                "COMMU5","COMMU6","COMMU7","COMMU8",
                "MONDIAL",
                "INQMALAD","INQAGRES","INQROUTE","INQCHOMA",
                "INQGUERR","INQNUCLE","INQALIM",
                "CLASSESO","ISEGO",
                "NOT_FAMI", "NOT_PROF", "NOT_AMIS",
                "NOT_COHE", "NOT_POLI", "NOT_LIBR",
                "NOT_LOG", "NOT_CAD",
                "HEUREUX",
                "CONFGOUV","ECHPOL",
                "REVENQ","AUTREREV","REV_TR7",]]
```

1.1.1 Selection of variable with low frequency of missing values

```
In [5]: missing_values = np.sum(df.isnull()).sort_values(ascending = False)
        missing_values
```

```
Out[5]: RADIQUOI    10146
        COMMU2       8115
        COMMU8       8115
        COMMU7       8115
        COMMU5       8115
        COMMU4       8115
        COMMU3       8115
        COMMU6       8115
        RADI3        8115
        RADI2        8115
        RADI1        8115
        RELEG        8115
        COMMU1       8115
        NOT_CAD      5206
        NOT_LOG      5199
        ISEGO        5095
        MONDIAL      3050
        OPICULT      2045
        CONFKEUF     2045
        CONFECOL     2045
        CONFMEFI     2045
        CONFBANK     2045
        CONFPRES     2045
        PROGRAD      1843
        NOT_PROF      185
        NOT_POLI      184
        NOT_COHE      177
        NOT_LIBR      171
        NOT_AMIS      157
```

```
NOT_FAMI      133
CONFPOLI        0
ORDLIB          0
PREOCCU1        0
CONFASSO        0
CONFENTR        0
CONFPUB         0
TRANSFST        0
REV_TR7         0
PREOCCU2        0
AUTREREV        0
INQMALAD        0
INQAGRES        0
INQROUTE        0
INQCHOMA        0
INQGUERR        0
INQNUCLE        0
INQALIM         0
CLASSESO        0
HEUREUX         0
CONFGOUV        0
ECHPOL          0
REVENQ          0
JUSTICE         0
dtype: int64
```

```
In [6]: selected_variable = missing_values[missing_values < 1500].index
```

```
In [7]: print("\n".join(selected_variable))
```

```
NOT_PROF
NOT_POLI
NOT_COHE
NOT_LIBR
NOT_AMIS
NOT_FAMI
CONFPOLI
ORDLIB
PREOCCU1
CONFASSO
CONFENTR
CONFPUB
TRANSFST
REV_TR7
PREOCCU2
AUTREREV
INQMALAD
INQAGRES
```

```
INQROUTE
INQCHOMA
INQGUERR
INQNUCLE
INQALIM
CLASSESO
HEUREUX
CONFGOUV
ECHPOL
REVENQ
JUSTICE
```

```
In [8]: df = cdv.loc[:,selected_variable]
```

```
In [9]: np.sum(df.isnull())
```

```
Out[9]: NOT_PROF    185
        NOT_POLI    184
        NOT_COHE    177
        NOT_LIBR    171
        NOT_AMIS    157
        NOT_FAMI    133
        CONFPOLI      0
        ORDLIB        0
        PREOCCU1      0
        CONFASSO      0
        CONFENTR      0
        CONFPUB       0
        TRANSFST      0
        REV_TR7       0
        PREOCCU2      0
        AUTREREV      0
        INQMALAD      0
        INQAGRES      0
        INQROUTE      0
        INQCHOMA      0
        INQGUERR      0
        INQNUCLE      0
        INQALIM       0
        CLASSESO      0
        HEUREUX       0
        CONFGOUV      0
        ECHPOL        0
        REVENQ        0
        JUSTICE       0
        dtype: int64
```

Dropping remaining missing values

```
In [10]: print(f"Number of exemple before dropping missing value {df.shape[0]}")
         df.dropna(inplace=True)
         print(f"Number of exemple after dropping missing value {df.shape[0]}")
```

```
Number of exemple before dropping missing value 11131
Number of exemple after dropping missing value 10701
```

```
In [11]: df.dtypes
```

```
Out[11]: NOT_PROF     float64
         NOT_POLI     float64
         NOT_COHE     float64
         NOT_LIBR     float64
         NOT_AMIS     float64
         NOT_FAMI     float64
         CONFPOLI      object
         ORDLIB        object
         PREOCCU1      object
         CONFASSO      object
         CONFENTR      object
         CONFPUB       object
         TRANSFST      object
         REV_TR7       object
         PREOCCU2      object
         AUTREREV       int64
         INQMALAD      object
         INQAGRES      object
         INQROUTE      object
         INQCHOMA      object
         INQGUERR      object
         INQNUCLE      object
         INQALIM       object
         CLASSESO      object
         HEUREUX       object
         CONFGOUV      object
         ECHPOL        object
         REVENQ         int64
         JUSTICE       object
         dtype: object
```

```
In [12]: obj_df = df.select_dtypes(include=['object']).copy()
```

```
In [13]: # Variable to be encoded
         obj_df
```

```
Out[13]:                     CONFPOLI                    ORDLIB  \
         0       Plutôt pas confiance       Un peu plus d'ordre
         1       Plutôt pas confiance       Un peu plus d'ordre
```

3	Plutôt pas confiance	Un peu plus d'ordre
4	Plutôt confiance	Beaucoup plus d'ordre
5	Pas du tout confiance	Beaucoup plus d'ordre
6	Pas du tout confiance	Beaucoup plus d'ordre
7	Plutôt confiance	Un peu plus de liberté
8	Plutôt pas confiance	Beaucoup plus d'ordre
9	Plutôt pas confiance	Beaucoup plus d'ordre
10	Pas du tout confiance	Beaucoup plus d'ordre
11	Plutôt confiance	Un peu plus d'ordre
13	Pas du tout confiance	Beaucoup plus d'ordre
14	Plutôt pas confiance	Un peu plus d'ordre
15	Pas du tout confiance	Un peu plus d'ordre
16	Pas du tout confiance	Un peu plus d'ordre
17	Plutôt pas confiance	Beaucoup plus d'ordre
18	Plutôt pas confiance	Beaucoup plus d'ordre
19	Pas du tout confiance	Un peu plus d'ordre
20	Pas du tout confiance	Beaucoup plus d'ordre
21	Plutôt pas confiance	Un peu plus d'ordre
22	Pas du tout confiance	Beaucoup plus d'ordre
23	[Nsp]	Beaucoup plus d'ordre
24	Pas du tout confiance	Beaucoup plus d'ordre
25	Plutôt pas confiance	Un peu plus de liberté
26	Plutôt pas confiance	Un peu plus d'ordre
27	Pas du tout confiance	Beaucoup plus d'ordre
28	Pas du tout confiance	Un peu plus d'ordre
29	Plutôt pas confiance	Beaucoup plus d'ordre
30	Pas du tout confiance	Beaucoup plus d'ordre
31	Plutôt pas confiance	Un peu plus d'ordre
...	...	...
11100	Plutôt pas confiance	Un peu plus d'ordre
11101	Pas du tout confiance	Beaucoup plus d'ordre
11102	[Nsp]	[Nsp]
11103	Pas du tout confiance	Beaucoup plus d'ordre
11104	Pas du tout confiance	Un peu plus d'ordre
11105	Plutôt pas confiance	Un peu plus d'ordre
11106	Pas du tout confiance	Beaucoup plus de liberté
11107	Pas du tout confiance	Un peu plus d'ordre
11108	Pas du tout confiance	Beaucoup plus d'ordre
11109	Plutôt pas confiance	Un peu plus de liberté
11110	Plutôt pas confiance	Un peu plus d'ordre
11111	Pas du tout confiance	Un peu plus d'ordre
11112	Plutôt pas confiance	Beaucoup plus d'ordre
11113	Pas du tout confiance	Beaucoup plus d'ordre
11114	Plutôt confiance	Un peu plus d'ordre
11115	Pas du tout confiance	Beaucoup plus d'ordre
11116	Plutôt pas confiance	Beaucoup plus d'ordre
11117	Plutôt confiance	Un peu plus d'ordre
11118	Plutôt confiance	Un peu plus d'ordre

```
11119   Plutôt pas confiance     Beaucoup plus d'ordre
11120       Plutôt confiance    Un peu plus de liberté
11121  Pas du tout confiance    Un peu plus de liberté
11122  Pas du tout confiance     Beaucoup plus d'ordre
11123  Pas du tout confiance       Un peu plus d'ordre
11124  Pas du tout confiance     Beaucoup plus d'ordre
11125       Plutôt confiance       Un peu plus d'ordre
11127         Très confiance       Un peu plus d'ordre
11128  Pas du tout confiance       Un peu plus d'ordre
11129   Plutôt pas confiance     Beaucoup plus d'ordre
11130       Plutôt confiance       Un peu plus d'ordre

                                PREOCCU1               CONFASSO  \
0      La dégradation de l'environnement   Plutôt pas confiance
1            La violence et l'insécurité       Plutôt confiance
3           Les tensions internationales       Plutôt confiance
4                  La pauvreté en France       Plutôt confiance
5                          L'immigration       Plutôt confiance
6                             Le chômage       Plutôt confiance
7                             Le chômage       Plutôt confiance
8                             Le chômage   Plutôt pas confiance
9                          L'immigration   Plutôt pas confiance
10                         L'immigration       Plutôt confiance
11          Les tensions internationales       Plutôt confiance
13                            Le chômage       Plutôt confiance
14                  Les maladies graves       Plutôt confiance
15                 La pauvreté en France       Plutôt confiance
16           La violence et l'insécurité   Plutôt pas confiance
17                            Le chômage       Plutôt confiance
18                  Les conflits sociaux       Plutôt confiance
19                 La pauvreté en France       Plutôt confiance
20     La dégradation de l'environnement         Très confiance
21                  Les maladies graves       Plutôt confiance
22                 La pauvreté en France   Plutôt pas confiance
23                  Les maladies graves                  [Nsp]
24                         L'immigration  Pas du tout confiance
25     La dégradation de l'environnement       Plutôt confiance
26                            Le chômage       Plutôt confiance
27           La violence et l'insécurité  Pas du tout confiance
28                              L'Europe         Très confiance
29           La violence et l'insécurité       Plutôt confiance
30                            Le chômage   Plutôt pas confiance
31          Les tensions internationales   Plutôt pas confiance
...                                  ...                    ...
11100                           L'Europe   Plutôt pas confiance
11101        La violence et l'insécurité  Pas du tout confiance
11102                          [Nsp, NR]                  [Nsp]
11103        La violence et l'insécurité  Pas du tout confiance
```

```
11104  La dégradation de l'environnement       Plutôt confiance
11105                          Le chômage       Plutôt confiance
11106                          Le chômage                   [Nsp]
11107                       L'immigration         Très confiance
11108                           La drogue  Pas du tout confiance
11109        Les tensions internationales       Plutôt confiance
11110                 Les maladies graves       Plutôt confiance
11111           La pauvreté dans le monde       Plutôt confiance
11112         La violence et l'insécurité   Plutôt pas confiance
11113                       L'immigration   Plutôt pas confiance
11114  La dégradation de l'environnement         Très confiance
11115                       L'immigration       Plutôt confiance
11116                          Le chômage         Très confiance
11117                 Les maladies graves       Plutôt confiance
11118                          Le chômage       Plutôt confiance
11119                       L'immigration       Plutôt confiance
11120           La pauvreté dans le monde   Plutôt pas confiance
11121                 Les maladies graves   Plutôt pas confiance
11122               La pauvreté en France       Plutôt confiance
11123         La violence et l'insécurité       Plutôt confiance
11124                       L'immigration  Pas du tout confiance
11125               La pauvreté en France   Plutôt pas confiance
11127                       L'immigration   Plutôt pas confiance
11128                       L'immigration  Pas du tout confiance
11129                       L'immigration       Plutôt confiance
11130               La pauvreté en France       Plutôt confiance

                   CONFENTR                CONFPUB TRANSFST  \
0             Très confiance         Très confiance      Non
1          Plutôt confiance       Plutôt confiance      Oui
3          Plutôt confiance       Plutôt confiance      Non
4       Plutôt pas confiance       Plutôt confiance      Oui
5       Plutôt pas confiance   Plutôt pas confiance      Oui
6       Plutôt pas confiance       Plutôt confiance      Oui
7          Plutôt confiance       Plutôt confiance      Oui
8       Plutôt pas confiance   Plutôt pas confiance      Oui
9             Très confiance       Plutôt confiance      Non
10         Plutôt confiance       Plutôt confiance      Oui
11         Plutôt confiance       Plutôt confiance      Oui
13      Plutôt pas confiance   Plutôt pas confiance      Oui
14         Plutôt confiance       Plutôt confiance      Oui
15         Plutôt confiance   Plutôt pas confiance      Oui
16         Plutôt confiance       Plutôt confiance      Oui
17         Plutôt confiance   Plutôt pas confiance      Oui
18         Plutôt confiance       Plutôt confiance      Oui
19      Plutôt pas confiance       Plutôt confiance      Oui
20     Pas du tout confiance   Plutôt pas confiance      Oui
21         Plutôt confiance   Plutôt pas confiance      Oui
```

22	Plutôt confiance	Plutôt confiance	Oui
23	[Nsp]	Très confiance	Oui
24	Très confiance	Pas du tout confiance	Oui
25	Plutôt pas confiance	Plutôt confiance	Oui
26	Plutôt confiance	Plutôt confiance	Oui
27	Pas du tout confiance	Plutôt pas confiance	Oui
28	Plutôt pas confiance	Plutôt confiance	Oui
29	Plutôt confiance	Plutôt confiance	Oui
30	Plutôt pas confiance	Plutôt pas confiance	Oui
31	Plutôt confiance	Plutôt pas confiance	Oui
...	...	...	...
11100	Plutôt pas confiance	Plutôt pas confiance	Oui
11101	Plutôt pas confiance	Plutôt pas confiance	Oui
11102	[Nsp]	[Nsp]	[Nsp]
11103	Très confiance	Pas du tout confiance	Oui
11104	Plutôt pas confiance	Pas du tout confiance	Oui
11105	[Nsp]	Plutôt confiance	Oui
11106	Pas du tout confiance	Plutôt pas confiance	Non
11107	Plutôt confiance	Plutôt pas confiance	Oui
11108	Plutôt pas confiance	Plutôt pas confiance	Oui
11109	Plutôt confiance	Plutôt pas confiance	Oui
11110	Plutôt confiance	Plutôt confiance	Oui
11111	Plutôt confiance	Plutôt pas confiance	Oui
11112	Plutôt confiance	Plutôt confiance	Oui
11113	Plutôt confiance	Plutôt confiance	Oui
11114	Très confiance	Plutôt confiance	Oui
11115	Plutôt pas confiance	Plutôt pas confiance	Oui
11116	Plutôt pas confiance	Plutôt confiance	Oui
11117	Plutôt confiance	Plutôt confiance	Oui
11118	[Nsp]	Plutôt confiance	Oui
11119	Plutôt pas confiance	Pas du tout confiance	Oui
11120	Plutôt pas confiance	Plutôt pas confiance	Non
11121	Plutôt confiance	Très confiance	Oui
11122	Pas du tout confiance	Pas du tout confiance	Oui
11123	Plutôt confiance	Plutôt confiance	Oui
11124	Plutôt pas confiance	Pas du tout confiance	Oui
11125	Très confiance	Plutôt confiance	Non
11127	Très confiance	Plutôt confiance	Oui
11128	Plutôt confiance	Plutôt pas confiance	Oui
11129	Très confiance	Pas du tout confiance	Oui
11130	Plutôt confiance	Plutôt confiance	Non

	REV_TR7	PREOCCU2 \
0	De 900 à 1500	La pauvreté en France
1	De 2300 à 3100	Les tensions internationales
3	Moins de 900	La pauvreté en France
4	De 1500 à 2300	Le chômage
5	Moins de 900	La violence et l'insécurité

```
6                        De 900 à 1500   La dégradation de l'environnement
7                        De 900 à 1500                        L'immigration
8                       De 1500 à 2300                Les maladies graves
9                         4000  et plus                           La drogue
10                      De 1500 à 2300         La violence et l'insécurité
11                      De 1500 à 2300                             Le chômage
13                        4000  et plus                       L'immigration
14                       3100  à 4000                          Le chômage
15                       De 900 à 1500         La violence et l'insécurité
16                       De 900 à 1500                        L'immigration
17                      De 1500 à 2300               La pauvreté en France
18                        4000  et plus                       L'immigration
19                       De 900 à 1500                             Le chômage
20                        4000  et plus               La pauvreté en France
21                      De 2300 à 3100                             Le chômage
22                       De 900 à 1500                        L'immigration
23                      De 1500 à 2300                        L'immigration
24                      De 2300 à 3100         La violence et l'insécurité
25                      De 2300 à 3100                             Le chômage
26                        4000  et plus               La pauvreté en France
27     Non déclaré (ne sait pas, refus)                         L'immigration
28                       De 900 à 1500                             Le chômage
29                      De 2300 à 3100                        L'immigration
30                       3100  à 4000                       L'immigration
31                      De 2300 à 3100         La violence et l'insécurité
...                                ...                                  ...
11100                    3100  à 4000         La violence et l'insécurité
11101                   De 2300 à 3100                             Le chômage
11102                   De 1500 à 2300                              [Nsp, NR]
11103                    3100  à 4000                 Les maladies graves
11104                   De 2300 à 3100            La pauvreté dans le monde
11105  Non déclaré (ne sait pas, refus)                  Les maladies graves
11106                   De 1500 à 2300            La pauvreté dans le monde
11107                    De 900 à 1500                La pauvreté en France
11108                    3100  à 4000        Les tensions internationales
11109                   De 2300 à 3100                 Les maladies graves
11110                    3100  à 4000   La dégradation de l'environnement
11111                   De 2300 à 3100   La dégradation de l'environnement
11112                   De 1500 à 2300        Les tensions internationales
11113                     4000  et plus       Les tensions internationales
11114                   De 2300 à 3100        Les tensions internationales
11115                   De 2300 à 3100           La pauvreté dans le monde
11116                    De 900 à 1500           La pauvreté dans le monde
11117                   De 2300 à 3100         La violence et l'insécurité
11118                    3100  à 4000   La dégradation de l'environnement
11119                   De 2300 à 3100               La pauvreté en France
11120                     Moins de 900                 Les maladies graves
11121                    3100  à 4000        Les tensions internationales
```

11122	De 900 à 1500	La violence et l'insécurité
11123	De 900 à 1500	Les maladies graves
11124	De 1500 à 2300	La pauvreté en France
11125	Moins de 900	Le chômage
11127	De 900 à 1500	La drogue
11128	4000 et plus	La violence et l'insécurité
11129	De 1500 à 2300	La pauvreté en France
11130	De 900 à 1500	La violence et l'insécurité

	INQMALAD	...	INQROUTE	INQCHOMA	INQGUERR	\
0	Beaucoup	...	Assez	Assez	Beaucoup	
1	Un peu	...	Un peu	Assez	Pas du tout	
3	Un peu	...	Un peu	Un peu	Un peu	
4	Beaucoup	...	Un peu	Pas du tout	Pas du tout	
5	Pas du tout	...	Assez	Beaucoup	Pas du tout	
6	Beaucoup	...	Beaucoup	Beaucoup	Un peu	
7	Un peu	...	Un peu	Assez	Un peu	
8	Beaucoup	...	Beaucoup	Assez	Un peu	
9	Assez	...	Un peu	Un peu	Un peu	
10	Beaucoup	...	Assez	Un peu	Assez	
11	Beaucoup	...	[Nsp]	[Nsp]	[Nsp]	
13	Un peu	...	Un peu	Un peu	Un peu	
14	Assez	...	Assez	Beaucoup	Beaucoup	
15	Beaucoup	...	Beaucoup	Pas du tout	Un peu	
16	Beaucoup	...	Assez	Assez	Assez	
17	Beaucoup	...	Beaucoup	Beaucoup	Beaucoup	
18	Beaucoup	...	Assez	Pas du tout	Pas du tout	
19	Assez	...	Assez	Assez	Assez	
20	Beaucoup	...	Pas du tout	Pas du tout	Pas du tout	
21	Beaucoup	...	Beaucoup	Beaucoup	Beaucoup	
22	Beaucoup	...	Beaucoup	Beaucoup	Beaucoup	
23	Beaucoup	...	Assez	Beaucoup	Assez	
24	Beaucoup	...	Beaucoup	Beaucoup	Beaucoup	
25	Un peu	...	Un peu	Beaucoup	Pas du tout	
26	Un peu	...	Un peu	Pas du tout	Un peu	
27	Assez	...	Assez	Un peu	Pas du tout	
28	Un peu	...	Un peu	Un peu	Un peu	
29	Assez	...	Assez	Pas du tout	Pas du tout	
30	Beaucoup	...	Assez	Assez	Un peu	
31	Beaucoup	...	Assez	Un peu	Assez	
...	...	...	...	...	...	
11100	Beaucoup	...	Beaucoup	Un peu	Assez	
11101	Beaucoup	...	Beaucoup	Beaucoup	Beaucoup	
11102	[Nsp]	...	[Nsp]	[Nsp]	[Nsp]	
11103	Beaucoup	...	Beaucoup	Beaucoup	Un peu	
11104	Pas du tout	...	Un peu	Un peu	Pas du tout	
11105	Beaucoup	...	Assez	Assez	Beaucoup	
11106	Un peu	...	Pas du tout	Un peu	Pas du tout	

```
11107     Beaucoup     ...          Beaucoup        Assez     Beaucoup
11108     Beaucoup     ...          Beaucoup        Assez       Un peu
11109        Assez     ...             Assez        Assez     Beaucoup
11110        Assez     ...           Un peu       Un peu       Un peu
11111     Beaucoup     ...          Beaucoup        Assez     Beaucoup
11112        Assez     ...             Assez        Assez        Assez
11113      Un peu      ...           Un peu  Pas du tout        Assez
11114     Beaucoup     ...             Assez        Assez     Beaucoup
11115      Un peu      ...           Un peu       Un peu       Un peu
11116     Beaucoup     ...             Assez     Beaucoup     Beaucoup
11117     Beaucoup     ...             Assez        Assez        Assez
11118        Assez     ...             Assez        Assez        Assez
11119     Beaucoup     ...             Assez        Assez        Assez
11120  Pas du tout     ...       Pas du tout  Pas du tout  Pas du tout
11121        Assez     ...           Un peu       Un peu        [Nsp]
11122  Pas du tout     ...          Beaucoup       Un peu       Un peu
11123     Beaucoup     ...          Beaucoup     Beaucoup     Beaucoup
11124     Beaucoup     ...           Un peu      Beaucoup     Beaucoup
11125     Beaucoup     ...          Beaucoup     Beaucoup     Beaucoup
11127        Assez     ...          Beaucoup     Beaucoup        Assez
11128        Assez     ...          Beaucoup       Un peu        Assez
11129     Beaucoup     ...          Beaucoup     Beaucoup     Beaucoup
11130      Un peu      ...             Assez        Assez       Un peu

          INQNUCLE      INQALIM                      CLASSESO  \
0         Beaucoup        Assez  La classe moyenne supérieure
1      Pas du tout  Pas du tout  La classe moyenne inférieure
3           Un peu       Un peu  La classe moyenne supérieure
4      Pas du tout     Beaucoup  La classe moyenne inférieure
5      Pas du tout     Beaucoup               Les défavorisés
6            Assez     Beaucoup               Les défavorisés
7           Un peu       Un peu  La classe moyenne supérieure
8           Un peu        Assez           La classe populaire
9           Un peu       Un peu                Les gens aisés
10          Un peu     Beaucoup           La classe populaire
11           [Nsp]        [Nsp]  La classe moyenne inférieure
13     Pas du tout  Pas du tout  La classe moyenne supérieure
14          Un peu       Un peu  La classe moyenne inférieure
15     Pas du tout       Un peu               Les défavorisés
16        Beaucoup        Assez  La classe moyenne inférieure
17          Un peu        Assez               Les défavorisés
18     Pas du tout  Pas du tout  La classe moyenne inférieure
19          Un peu        Assez  La classe moyenne inférieure
20           Assez       Un peu                Les gens aisés
21        Beaucoup     Beaucoup           La classe populaire
22        Beaucoup        Assez           La classe populaire
23          Un peu        Assez           La classe populaire
24           Assez       Un peu  La classe moyenne inférieure
```

25	Un peu	Un peu	La classe moyenne supérieure
26	Un peu	Un peu	La classe moyenne inférieure
27	Un peu	Assez	La classe populaire
28	Assez	Beaucoup	La classe moyenne supérieure
29	Pas du tout	Assez	La classe moyenne inférieure
30	Pas du tout	Assez	La classe moyenne supérieure
31	Pas du tout	Pas du tout	La classe moyenne inférieure
...	...	...	...
11100	Un peu	Assez	La classe moyenne supérieure
11101	Assez	Beaucoup	Les défavorisés
11102	[Nsp]	[Nsp]	[Nsp]
11103	Un peu	Un peu	La classe populaire
11104	Pas du tout	Un peu	La classe populaire
11105	Un peu	Un peu	La classe moyenne inférieure
11106	Assez	Pas du tout	Les privilégiés
11107	Beaucoup	Assez	Les privilégiés
11108	Beaucoup	Beaucoup	La classe moyenne inférieure
11109	Assez	Beaucoup	La classe populaire
11110	Pas du tout	Un peu	La classe moyenne inférieure
11111	Assez	Beaucoup	La classe populaire
11112	Assez	Assez	La classe moyenne inférieure
11113	Pas du tout	Un peu	La classe moyenne supérieure
11114	Beaucoup	Assez	La classe moyenne inférieure
11115	Pas du tout	Pas du tout	La classe populaire
11116	Beaucoup	Assez	La classe moyenne inférieure
11117	Assez	Un peu	La classe moyenne inférieure
11118	Un peu	Un peu	La classe moyenne inférieure
11119	Assez	Beaucoup	La classe moyenne inférieure
11120	Pas du tout	Pas du tout	La classe populaire
11121	Assez	Un peu	La classe moyenne supérieure
11122	Pas du tout	Un peu	Les défavorisés
11123	Beaucoup	Assez	La classe populaire
11124	Assez	Un peu	La classe moyenne inférieure
11125	Beaucoup	Beaucoup	Les gens aisés
11127	Assez	Un peu	Les gens aisés
11128	Un peu	Assez	Les défavorisés
11129	Beaucoup	Beaucoup	Les défavorisés
11130	Un peu	Un peu	La classe moyenne inférieure

	HEUREUX	CONFGOUV	ECHPOL	JUSTICE
0	Très souvent	Tout à fait confiance	Plutôt à gauche	Assez bien
1	Très souvent	Plutôt pas confiance	Plutôt à droite	Assez mal
3	Très souvent	Plutôt confiance	A gauche	Assez mal
4	Très souvent	Pas du tout confiance	Très à droite	Très mal
5	Occasionnellement	Pas du tout confiance	Au centre	Assez mal
6	Occasionnellement	Pas du tout confiance	Très à gauche	Assez mal
7	Assez souvent	Plutôt confiance	Plutôt à droite	Assez bien
8	Assez souvent	Plutôt pas confiance	Au centre	Assez mal

9	Très souvent	Pas du tout confiance	À droite	Très mal
10	Assez souvent	Plutôt pas confiance	Au centre	Assez mal
11	Occasionnellement	Plutôt pas confiance	A gauche	Assez bien
13	Assez souvent	Pas du tout confiance	Plutôt à droite	Assez mal
14	Assez souvent	Plutôt confiance	Très à gauche	Assez bien
15	Occasionnellement	Tout à fait confiance	Plutôt à gauche	Assez mal
16	Occasionnellement	Plutôt pas confiance	Très à droite	Très mal
17	Occasionnellement	Plutôt confiance	Au centre	Assez mal
18	Assez souvent	Plutôt pas confiance	Plutôt à droite	Assez bien
19	Occasionnellement	Plutôt pas confiance	A gauche	Assez mal
20	Occasionnellement	Plutôt confiance	Très à droite	Très mal
21	Occasionnellement	Pas du tout confiance	Au centre	Assez mal
22	Occasionnellement	Pas du tout confiance	Très à droite	Très mal
23	Assez souvent	Plutôt pas confiance	Plutôt à gauche	Assez bien
24	Occasionnellement	Pas du tout confiance	Très à droite	Assez mal
25	Occasionnellement	Pas du tout confiance	Très à gauche	Assez mal
26	Occasionnellement	Plutôt pas confiance	Plutôt à gauche	Assez bien
27	Assez souvent	Pas du tout confiance	Très à droite	Très mal
28	Occasionnellement	Pas du tout confiance	Très à gauche	Assez mal
29	Occasionnellement	Pas du tout confiance	Au centre	Assez mal
30	Occasionnellement	Pas du tout confiance	À droite	Assez mal
31	Assez souvent	Plutôt pas confiance	À droite	Assez mal
...	...	...	...	...
11100	[Nsp]	Plutôt confiance	Au centre	Très bien
11101	Occasionnellement	Pas du tout confiance	Au centre	Très mal
11102	Très souvent	Pas du tout confiance	[Nsp, NR]	[Nsp]
11103	Très souvent	Pas du tout confiance	Très à droite	Très mal
11104	Occasionnellement	Pas du tout confiance	Au centre	Assez mal
11105	Assez souvent	Plutôt pas confiance	[Nsp, NR]	Assez mal
11106	Assez souvent	Pas du tout confiance	À droite	Assez mal
11107	Très souvent	Pas du tout confiance	Au centre	Assez bien
11108	Occasionnellement	Plutôt pas confiance	Plutôt à droite	Assez bien
11109	Assez souvent	Plutôt confiance	À droite	Assez bien
11110	Très souvent	Plutôt confiance	Plutôt à droite	Assez bien
11111	Assez souvent	Pas du tout confiance	Plutôt à droite	Très mal
11112	Assez souvent	Plutôt confiance	Au centre	Assez mal
11113	Occasionnellement	Pas du tout confiance	Plutôt à droite	Très mal
11114	Très souvent	Plutôt confiance	À droite	Assez mal
11115	Très souvent	Plutôt pas confiance	Très à droite	Assez mal
11116	Occasionnellement	Plutôt confiance	Plutôt à droite	Assez bien
11117	Occasionnellement	Plutôt confiance	Au centre	Assez mal
11118	Très souvent	Plutôt confiance	Au centre	Assez bien
11119	Occasionnellement	Plutôt pas confiance	À droite	Très mal
11120	Assez souvent	Pas du tout confiance	Au centre	Assez mal
11121	Occasionnellement	Plutôt confiance	Plutôt à gauche	Assez bien
11122	Occasionnellement	Pas du tout confiance	À droite	Très mal
11123	Assez souvent	Pas du tout confiance	Au centre	Assez bien
11124	Occasionnellement	Pas du tout confiance	À droite	Très mal

```
11125      Assez souvent       Plutôt confiance  Plutôt à gauche  Assez bien
11127             Jamais  Tout à fait confiance         À droite  Assez bien
11128             Jamais  Pas du tout confiance  Plutôt à droite    Très mal
11129  Occasionnellement  Pas du tout confiance    Très à droite    Très mal
11130      Assez souvent       Plutôt confiance        Au centre  Assez bien

[10701 rows x 21 columns]
```

1.1.2 Variable encoding - HEUREUX

```
In [14]: df["HEUREUX"].value_counts()

Out[14]: Assez souvent        5240
         Occasionnellement    3537
         Très souvent         1702
         Jamais                195
         [Nsp]                  27
         Name: HEUREUX, dtype: int64

In [15]: # reducing problem to a 2 class classification problem
         df["HEUREUX_CLF"] = 0
         df.loc[df["HEUREUX"]=="Très souvent", "HEUREUX_CLF"] = 1
         df.loc[df["HEUREUX"]=="Assez souvent", "HEUREUX_CLF"] = 1

In [16]: # Modelisation as a 5 multi class classification problem
         df["HEUREUX_MCLF"] = 0
         df.loc[df["HEUREUX"]=="Très souvent", "HEUREUX_MCLF"] = 4
         df.loc[df["HEUREUX"]=="Assez souvent", "HEUREUX_MCLF"] = 3
         df.loc[df["HEUREUX"]=="Occasionnellement", "HEUREUX_MCLF"] = 2
         df.loc[df["HEUREUX"]=="Jamais", "HEUREUX_MCLF"] = 1
         df.loc[df["HEUREUX"]=="[Nsp]", "HEUREUX_MCLF"] = 0

In [17]: # Modelisation as a regression problem
         df["HEUREUX_REG"] = None
         df.loc[df["HEUREUX"]=="Très souvent", "HEUREUX_REG"] = 3
         df.loc[df["HEUREUX"]=="Assez souvent", "HEUREUX_REG"] = 2
         df.loc[df["HEUREUX"]=="Occasionnellement", "HEUREUX_REG"] = 1
         df.loc[df["HEUREUX"]=="Jamais", "HEUREUX_REG"] = 0
         df.loc[df["HEUREUX"]=="[Nsp]", "HEUREUX_REG"] = None
         # NB '[Nsp]' lines should be removed, ...
```

1.1.3 Variable encoding - others

```
In [18]: # Encoding using "Find & replace"

In [19]: confiance = {
             "Pas du tout confiance" : -2,
             "Plutôt pas confiance" : -1,
             "[Nsp]" :0,
```

```
    "Plutôt confiance" : 1,
    "Très confiance" : 2
}

confiance_2 = {
    "Pas du tout confiance" : -2,
    "Plutôt pas confiance" : -1,
    "[Nsp]" :0,
    "Plutôt confiance" : 1,
    "Tout à fait confiance" : 2
}

bien  = {
    "Très bien" : 2,
    "Assez bien": 1,
    "[Nsp]" : 0,
    "Assez mal" : -1,
    "Très mal" : -2
}

beaucoup  = {
    "Beaucoup" : 3,
    "Assez": 2,
    "Un peu" : 1,
    "Pas du tout" : 0,
    "[Nsp]" : None
}

frequence = {
    "Très souven" : 3,
    "Assez souvent": 2,
    "Rarement" : 1,
    "Jamais" : 0,
    "[Nsp]" : None
}

cleanup_nums = {
    "JUSTICE" : bien,
    "CONFPUB" : confiance,
    "CONFENTR" : confiance,
    "CONFASSO" : confiance,
    "CONFPOLI" : confiance,
    "CONFGOUV" : confiance_2,
    "INQMALAD" : beaucoup,
    "INQAGRES" : beaucoup,
    "INQROUTE" : beaucoup,
    "INQCHOMA" : beaucoup,
    "INQGUERR" : beaucoup,
```

```
    "INQNUCLE" : beaucoup,
    "INQALIM" : beaucoup,
    "ISEGO" : frequence
}
```

In [20]:
```
df.replace(cleanup_nums, inplace=True)
df.head()
```

Out[20]:

	NOT_PROF	NOT_POLI	NOT_COHE	NOT_LIBR	NOT_AMIS	NOT_FAMI	CONFPOLI	\
0	6.0	3.0	6.0	6.0	6.0	5.0	-1	
1	6.0	5.0	7.0	6.0	6.0	7.0	-1	
3	5.0	5.0	6.0	6.0	6.0	7.0	-1	
4	7.0	6.0	6.0	7.0	6.0	7.0	1	
5	5.0	2.0	6.0	6.0	6.0	6.0	-2	

	ORDLIB	PREOCCU1	CONFASSO	\
0	Un peu plus d'ordre	La dégradation de l'environnement	-1	
1	Un peu plus d'ordre	La violence et l'insécurité	1	
3	Un peu plus d'ordre	Les tensions internationales	1	
4	Beaucoup plus d'ordre	La pauvreté en France	1	
5	Beaucoup plus d'ordre	L'immigration	1	

	...	INQALIM	CLASSESO	HEUREUX	\
0	...	2.0	La classe moyenne supérieure	Très souvent	
1	...	0.0	La classe moyenne inférieure	Très souvent	
3	...	1.0	La classe moyenne supérieure	Très souvent	
4	...	3.0	La classe moyenne inférieure	Très souvent	
5	...	3.0	Les défavorisés	Occasionnellement	

	CONFGOUV	ECHPOL	REVENQ	JUSTICE	HEUREUX_CLF	HEUREUX_MCLF	\
0	2	Plutôt à gauche	1100	1	1	4	
1	-1	Plutôt à droite	2000	-1	1	4	
3	1	A gauche	200	-1	1	4	
4	-2	Très à droite	998	-2	1	4	
5	-2	Au centre	750	-1	0	2	

	HEUREUX_REG
0	3
1	3
3	3
4	3
5	1

[5 rows x 32 columns]

In [21]:
```
# Encoding categorical variable using Hot Encoder
```

In [22]:
```
df = pd.get_dummies(df, columns=["PREOCCU1","PREOCCU2",
                                 "CLASSESO","ECHPOL",
```

```
                                        "TRANSFST","ORDLIB"],
                            drop_first=1)
```

In [23]:
```
revenues = {
    "4000  et plus" : 4000,
    "3100  à 4000 " : 3100,
    "De 2300 à 3100 " : 2300,
    "De 1500 à 2300 " : 1500,
    "De 900 à 1500 " : 900,
    "Moins de 900 " : 0,
    "Non déclaré (ne sait pas, refus)": None
}

cleanup = {"REV_TR7" : revenues}

df.replace(cleanup, inplace=True)
```

In [25]:
```
df.dtypes
```

Out[25]:
```
NOT_PROF                                    float64
NOT_POLI                                    float64
NOT_COHE                                    float64
NOT_LIBR                                    float64
NOT_AMIS                                    float64
NOT_FAMI                                    float64
CONFPOLI                                      int64
CONFASSO                                      int64
CONFENTR                                      int64
CONFPUB                                       int64
REV_TR7                                     float64
AUTREREV                                      int64
INQMALAD                                    float64
INQAGRES                                    float64
INQROUTE                                    float64
INQCHOMA                                    float64
INQGUERR                                    float64
INQNUCLE                                    float64
INQALIM                                     float64
HEUREUX                                      object
CONFGOUV                                      int64
REVENQ                                        int64
JUSTICE                                       int64
HEUREUX_CLF                                   int64
HEUREUX_MCLF                                  int64
HEUREUX_REG                                  object
PREOCCU1_L'immigration                        uint8
PREOCCU1_La drogue                            uint8
PREOCCU1_La dégradation de l'environnement    uint8
```

```
PREOCCU1_La pauvreté dans le monde              uint8
                                               ...
PREOCCU2_L'immigration                         uint8
PREOCCU2_La drogue                             uint8
PREOCCU2_La dégradation de l'environnement     uint8
PREOCCU2_La pauvreté dans le monde             uint8
PREOCCU2_La pauvreté en France                 uint8
PREOCCU2_La violence et l'insécurité           uint8
PREOCCU2_Le chômage                            uint8
PREOCCU2_Les conflits sociaux                  uint8
PREOCCU2_Les maladies graves                   uint8
PREOCCU2_Les tensions internationales          uint8
PREOCCU2_[Nsp, NR]                             uint8
CLASSESO_La classe moyenne supérieure          uint8
CLASSESO_La classe populaire                   uint8
CLASSESO_Les défavorisés                       uint8
CLASSESO_Les gens aisés                        uint8
CLASSESO_Les privilégiés                       uint8
CLASSESO_[Nsp]                                 uint8
ECHPOL_Au centre                               uint8
ECHPOL_Plutôt à droite                         uint8
ECHPOL_Plutôt à gauche                         uint8
ECHPOL_Très à droite                           uint8
ECHPOL_Très à gauche                           uint8
ECHPOL_[Nsp, NR]                               uint8
ECHPOL_À droite                                uint8
TRANSFST_Oui                                   uint8
TRANSFST_[Nsp]                                 uint8
ORDLIB_Beaucoup plus de liberté                uint8
ORDLIB_Un peu plus d'ordre                     uint8
ORDLIB_Un peu plus de liberté                  uint8
ORDLIB_[Nsp]                                   uint8
Length: 67, dtype: object
```

In [26]:
```
obj_df = df.select_dtypes(include=['object']).copy()
```

In [28]:
```
obj_df
```

Out[28]:

	HEUREUX	HEUREUX_REG
0	Très souvent	3
1	Très souvent	3
3	Très souvent	3
4	Très souvent	3
5	Occasionnellement	1
6	Occasionnellement	1
7	Assez souvent	2
8	Assez souvent	2
9	Très souvent	3

```
10          Assez souvent          2
11      Occasionnellement          1
13          Assez souvent          2
14          Assez souvent          2
15      Occasionnellement          1
16      Occasionnellement          1
17      Occasionnellement          1
18          Assez souvent          2
19      Occasionnellement          1
20      Occasionnellement          1
21      Occasionnellement          1
22      Occasionnellement          1
23          Assez souvent          2
24      Occasionnellement          1
25      Occasionnellement          1
26      Occasionnellement          1
27          Assez souvent          2
28      Occasionnellement          1
29      Occasionnellement          1
30      Occasionnellement          1
31          Assez souvent          2
...                    ...        ...
11100               [Nsp]       None
11101   Occasionnellement          1
11102        Très souvent          3
11103        Très souvent          3
11104   Occasionnellement          1
11105       Assez souvent          2
11106       Assez souvent          2
11107        Très souvent          3
11108   Occasionnellement          1
11109       Assez souvent          2
11110        Très souvent          3
11111       Assez souvent          2
11112       Assez souvent          2
11113   Occasionnellement          1
11114        Très souvent          3
11115        Très souvent          3
11116   Occasionnellement          1
11117   Occasionnellement          1
11118        Très souvent          3
11119   Occasionnellement          1
11120       Assez souvent          2
11121   Occasionnellement          1
11122   Occasionnellement          1
11123       Assez souvent          2
11124   Occasionnellement          1
11125       Assez souvent          2
```

```
11127             Jamais           0
11128             Jamais           0
11129  Occasionnellement           1
11130      Assez souvent           2

[10701 rows x 2 columns]
```

In [29]:
```
np.sum(df.isnull())
```

Out[29]:
```
NOT_PROF                                      0
NOT_POLI                                      0
NOT_COHE                                      0
NOT_LIBR                                      0
NOT_AMIS                                      0
NOT_FAMI                                      0
CONFPOLI                                      0
CONFASSO                                      0
CONFENTR                                      0
CONFPUB                                       0
REV_TR7                                     291
AUTREREV                                      0
INQMALAD                                     49
INQAGRES                                     98
INQROUTE                                    104
INQCHOMA                                    105
INQGUERR                                    118
INQNUCLE                                    117
INQALIM                                     109
HEUREUX                                       0
CONFGOUV                                      0
REVENQ                                        0
JUSTICE                                       0
HEUREUX_CLF                                   0
HEUREUX_MCLF                                  0
HEUREUX_REG                                  27
PREOCCU1_L'immigration                        0
PREOCCU1_La drogue                            0
PREOCCU1_La dégradation de l'environnement    0
PREOCCU1_La pauvreté dans le monde            0
                                           ...
PREOCCU2_L'immigration                        0
PREOCCU2_La drogue                            0
PREOCCU2_La dégradation de l'environnement    0
PREOCCU2_La pauvreté dans le monde            0
PREOCCU2_La pauvreté en France                0
PREOCCU2_La violence et l'insécurité          0
PREOCCU2_Le chômage                           0
PREOCCU2_Les conflits sociaux                 0
```

```
PREOCCU2_Les maladies graves                    0
PREOCCU2_Les tensions internationales           0
PREOCCU2_[Nsp, NR]                              0
CLASSESO_La classe moyenne supérieure           0
CLASSESO_La classe populaire                    0
CLASSESO_Les défavorisés                        0
CLASSESO_Les gens aisés                         0
CLASSESO_Les privilégiés                        0
CLASSESO_[Nsp]                                  0
ECHPOL_Au centre                                0
ECHPOL_Plutôt à droite                          0
ECHPOL_Plutôt à gauche                          0
ECHPOL_Très à droite                            0
ECHPOL_Très à gauche                            0
ECHPOL_[Nsp, NR]                                0
ECHPOL_À droite                                 0
TRANSFST_Oui                                    0
TRANSFST_[Nsp]                                  0
ORDLIB_Beaucoup plus de liberté                 0
ORDLIB_Un peu plus d'ordre                      0
ORDLIB_Un peu plus de liberté                   0
ORDLIB_[Nsp]                                    0
Length: 67, dtype: int64
```

```
In [30]: print(f"Number of exemple before dropping missing value {df.shape[0]}")
         df.dropna(inplace=True)
         print(f"Number of exemple after dropping missing value {df.shape[0]}")
```

```
Number of exemple before dropping missing value 10701
Number of exemple after dropping missing value 10096
```

1.2 2) Base Classification model

1.2.1 a) Builing training set and test set

```
In [39]: a = set(df.columns)
         b = set(df.columns.drop(['HEUREUX',"HEUREUX_CLF",
                                  "HEUREUX_MCLF", "HEUREUX_REG"]))
         a-b
```

```
Out[39]: {'HEUREUX', 'HEUREUX_CLF', 'HEUREUX_MCLF', 'HEUREUX_REG'}
```

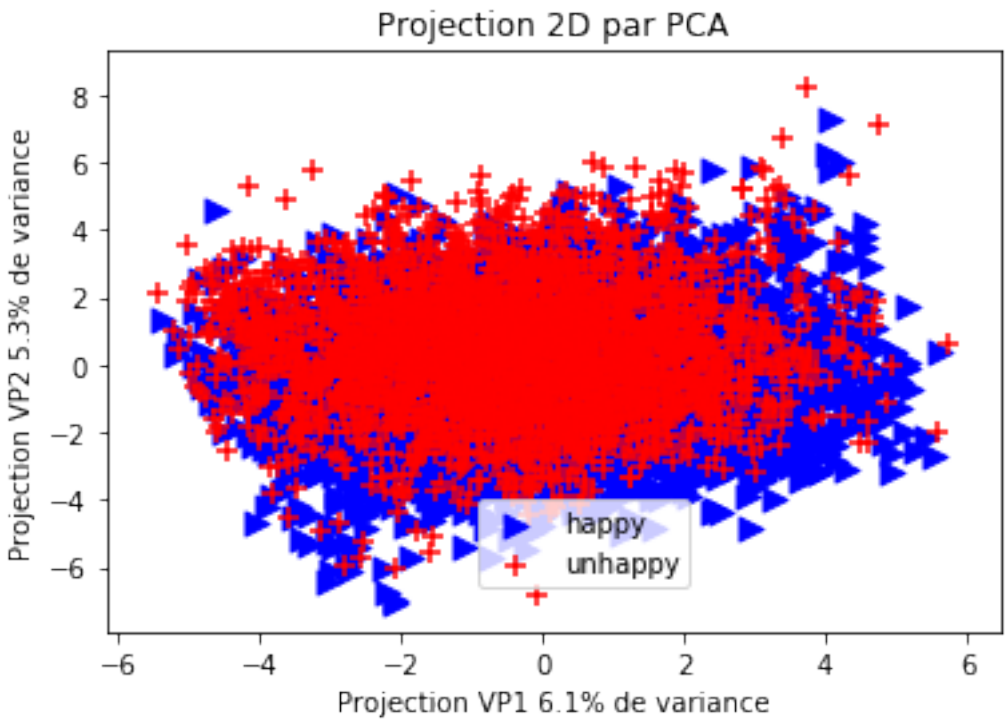
```
In [40]: #features = ["NOT_FAMI", "NOT_PROF", "NOT_AMIS","NOT_COHE",
         #             "NOT_POLI", "NOT_LIBR", "CONFPUB_NUM",
         #             "REVENQ","AUTREREV","REV_TR7_NUM",
         #             "CONFENTR_NUM","CONFASSO_NUM","CONFPOLI_NUM"]

         features = df.columns.drop(['HEUREUX',"HEUREUX_CLF",
```

```
                                    "HEUREUX_MCLF", "HEUREUX_REG"])

        X = df.loc[:,features]
        y = df["HEUREUX_CLF"]

        X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                            test_size=0.2,
                                                            random_state=42)

        scaler = StandardScaler().fit(X_train)
        X_train = scaler.transform(X_train)
        X_test = scaler.transform(X_test)

        print(f"Number exemple: \n- training set: \
        {y_train.shape[0]}\n- test set: {y_test.shape[0]}")
        print(f"Number of features: p={X_train.shape[1]}")
```

```
Number exemple:
- training set: 8076
- test set: 2020
Number of features: p=63
```

1.2.2 b) Visualisation of the data

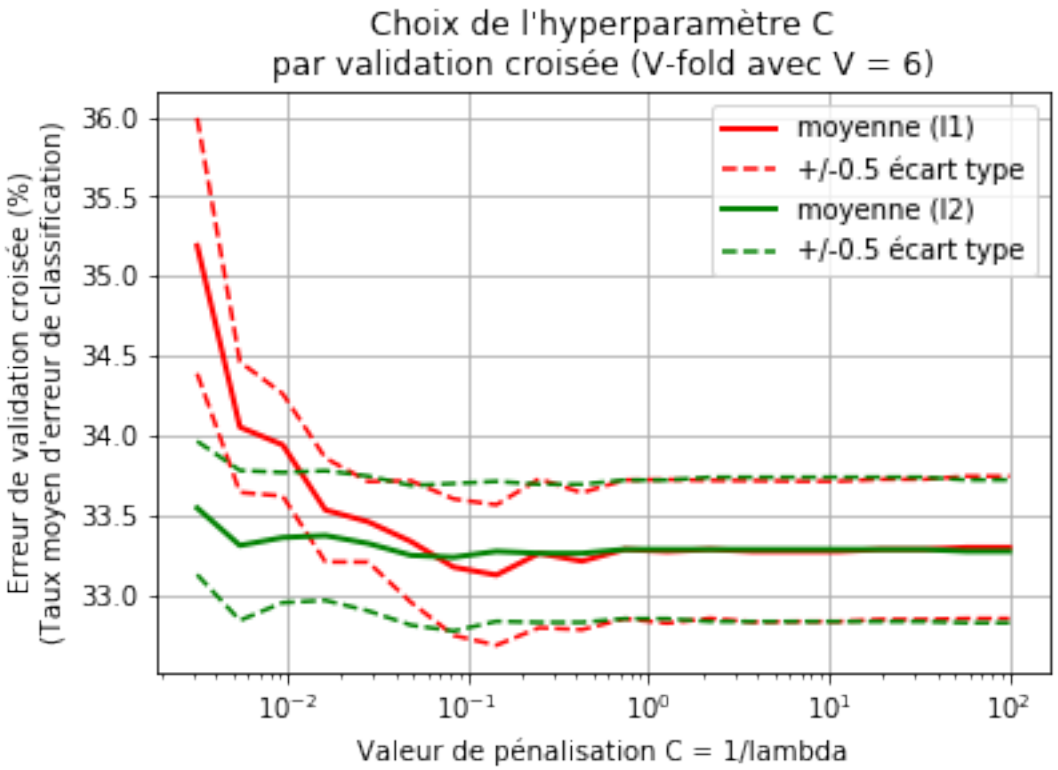
```
In [41]: # Reduction dim PCA
         pca = PCA(n_components=2)
         pca.fit(X_train)
         X_r = pca.transform(X_train)
         happy = (y_train==1)
         unhappy = (y_train==0)
         plt.scatter(X_r[happy,0], X_r[happy,1],
                     s=80, c='blue',marker=">", label="happy")
         plt.scatter(X_r[unhappy,0], X_r[unhappy,1],
                     s=80, c='red',marker='+', label="unhappy")
         plt.ylabel(f'Projection VP2 \
         {pca.explained_variance_ratio_[1]*100:0.1f}% de variance')
         plt.xlabel(f'Projection VP1 \
         {pca.explained_variance_ratio_[0]*100:0.1f}% de variance')
         plt.title("Projection 2D par PCA")
         plt.legend(bbox_to_anchor=(0.4, 0.25))
         plt.show()
```



1.2.3 c) Logistic Regression - 2 class

```
In [42]: nb_value = 20 # Nombre de valeurs testées pour l'hyperparamètre
         mean_score_l1 = np.zeros(nb_value)
         mean_score_l2 = np.zeros(nb_value)
         C_log = np.logspace(-2.5,2,nb_value)
         cv = 6 # V-fold, nombre de fold

         mean_score_l1 = np.empty(nb_value)
         std_scores_l1 = np.empty(nb_value)

         mean_score_l2 = np.empty(nb_value)
         std_scores_l2 = np.empty(nb_value)

         np.random.seed(seed=42)

         startTime = time.time()

         for i, C in enumerate(C_log):
             clf = LogisticRegression(C=C, penalty='l1',
                                      tol=0.01, random_state=42,
                                      class_weight='balanced')
             mean_score_l1[i] = 100*np.mean(1-cross_val_score(clf,
```

```
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='accuracy'))
    std_scores_l1[i] = 100*np.std(1-cross_val_score(clf,
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='accuracy'))


for i, C in enumerate(C_log):
    clf = LogisticRegression(C=C, penalty='l2', tol=0.01, random_state=42, class_weight
    mean_score_l2[i] = 100*np.mean(1-cross_val_score(clf,
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='accuracy'))
    std_scores_l2[i] = 100*np.std(1-cross_val_score(clf,
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='accuracy'))

plt.figure()
plt.semilogx(C_log,mean_score_l1[:],'r',linewidth=2,label='moyenne (l1)')
plt.semilogx(C_log,mean_score_l1[:]-0.5*std_scores_l1[:],
             'r--', label=u'+/-0.5 écart type')
plt.semilogx(C_log,mean_score_l1[:]+0.5*std_scores_l1[:],'r--')

plt.semilogx(C_log,mean_score_l2[:],'g',linewidth=2,label='moyenne (l2)')
plt.semilogx(C_log,mean_score_l2[:]-0.5*std_scores_l2[:], 'g--', label=u'+/-0.5 écart t
plt.semilogx(C_log,mean_score_l2[:]+0.5*std_scores_l2[:],'g--')

plt.xlabel("Valeur de pénalisation C = 1/lambda")
plt.ylabel(u"Erreur de validation croisée (%)\n(Taux moyen d'erreur de classification)"
plt.title(u"Choix de l'hyperparamètre C\npar validation croisée \
(V-fold avec V = %s)" % (cv))
plt.legend(bbox_to_anchor=(1, 1))
plt.grid()
plt.show()
print("Détermination des paramètres optimaux en %0.1f s" % (time.time() - startTime))
print("Pénalisation l1, valeur optimale : C = %0.2f" % (C_log[np.argmin(mean_score_l1)]
print("Pénalisation l2, valeur optimale : C = %0.2f" % (C_log[np.argmin(mean_score_l2)]
```




```
Détermination des paramètres optimaux en 45.1 s
Pénalisation l1, valeur optimale : C = 0.14
Pénalisation l2, valeur optimale : C = 0.08
```

```
In [43]: # Learning on full training set with optimals hyperparameters
         # and score evaluation on test set
         clf = LogisticRegression(C=C_log[np.argmin(mean_score_l1)],
                                  penalty='l1',
                                  tol=0.01,
                                  random_state=42,
                                  class_weight='balanced')
         clf.fit(X_train, y_train)
         #y_pred = clf.predict(X_test)
         accuracy = clf.score(X_test, y_test)
         print(accuracy)
```

```
0.65
```

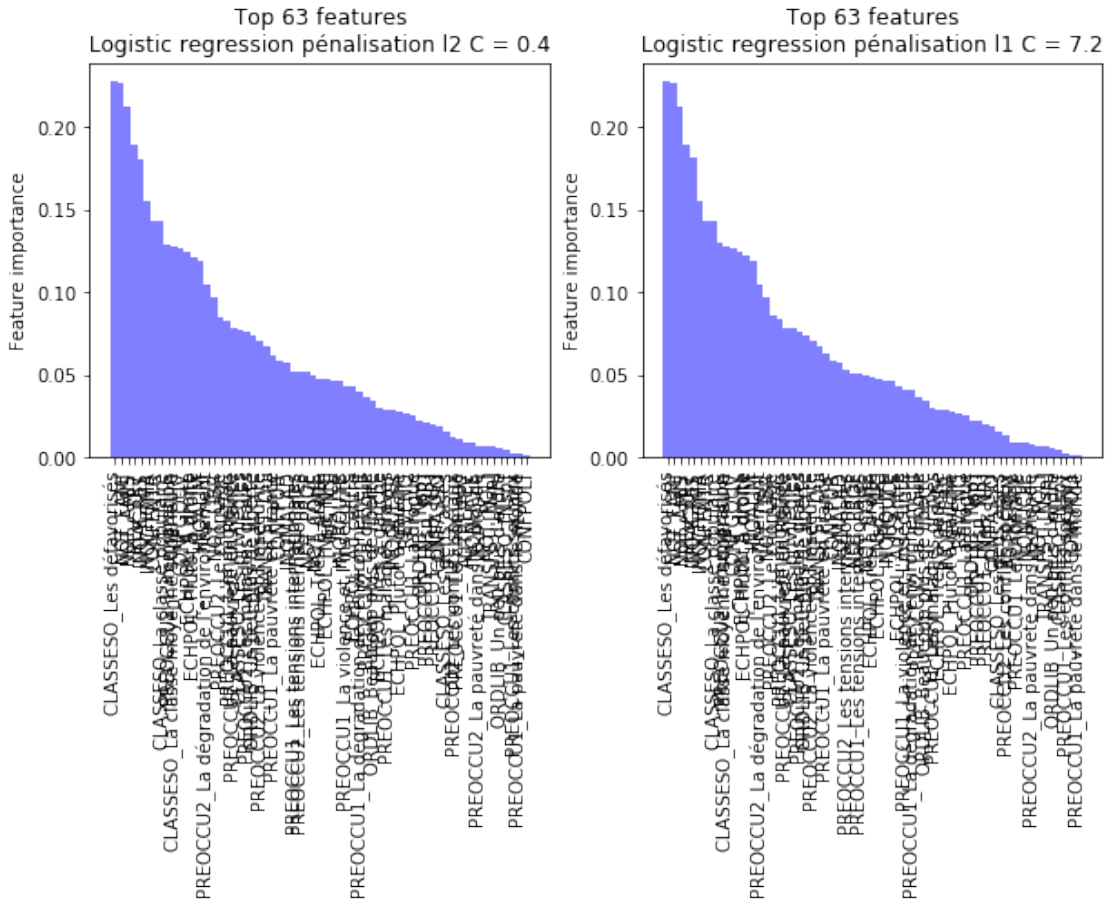
```
In [44]: y_pred = clf.predict(X_test)
         for i in range(20):
             print(y_pred[i], y_test.iloc[i])
```

```
0 0
0 1
0 1
1 1
1 1
0 0
1 1
1 1
1 1
0 1
1 1
1 1
1 1
0 1
1 1
0 0
0 0
1 1
0 1
1 1
```

In [49]:
```
# Use regression coefficients to rank features
clf = LogisticRegression(penalty='l2',C=0.4)

clf.fit(X_train,y_train)
coef_l2 = abs(clf.coef_)
coef_sorted_l2 = -np.sort(-coef_l2).reshape(-1)
print(coef_sorted_l2)
features_sorded_l2 = np.argsort(-coef_l2).reshape(-1)
print(features_sorded_l2)
features_name = np.array(features)
features_name_sorted_l2 = features_name[features_sorded_l2]

clf = LogisticRegression(penalty='l1',C=7.2)

clf.fit(X_train,y_train)
coef_l1 = abs(clf.coef_)
coef_sorted_l1 = -np.sort(-coef_l1).reshape(-1)
features_sorded_l1 = np.argsort(-coef_l1).reshape(-1)
features_name_sorted_l1 = features_name[features_sorded_l1]

nf = len(features)

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))
ind = np.arange(nf)    # the x locations for the groups

plt.subplot(1, 2, 1)
```

```
        p1 = plt.bar(ind, coef_sorted_l2[0:nf], 1, color='b',alpha=0.5)
        plt.ylabel('Feature importance')
        plt.title(u'Top %i features\nLogistic regression pénalisation l2 C = 0.4' % nf)
        plt.xticks(ind + 0.35/2.0, features_name_sorted_l2[0:nf], rotation = 90)

        plt.subplot(1, 2, 2)
        p1 = plt.bar(ind, coef_sorted_l1[0:nf], 1, color='b',alpha=0.5)
        plt.ylabel('Feature importance')
        plt.title(u'Top %i features\nLogistic regression pénalisation l1 C = 7.2' % nf)
        plt.xticks(ind + 0.35/2.0, features_name_sorted_l1[0:nf], rotation = 90)

        plt.show()
```

```
[ 0.22772182  0.22669885  0.21282355  0.18931107  0.18103767  0.15477363
  0.14318045  0.14273368  0.12872802  0.12809305  0.12655601  0.12409137
  0.12165396  0.11949709  0.10429326  0.09661988  0.0852505   0.08276178
  0.07773891  0.07769943  0.07591614  0.07395955  0.07002458  0.06730892
  0.06219479  0.05887143  0.0570756   0.05225386  0.05223556  0.05142397
  0.04977487  0.04785303  0.04712959  0.04628608  0.04616966  0.04276429
  0.04274725  0.03937455  0.03605774  0.034043    0.02939674  0.02873882
  0.02847372  0.02812717  0.02709788  0.02594569  0.02219453  0.02157691
  0.01946564  0.01866204  0.01569384  0.01274503  0.01074473  0.00924259
  0.00886303  0.00711545  0.00707863  0.00652487  0.0053105   0.00493716
  0.0026723   0.00188325  0.00141475]
[46  5  4 10 13 15  8 45 33 44 19 56 51 18 35  3 39 22 37 48 41 61 38 57 26
  9 12 31 42  0  2 53 55 21 14 27 50 24 34 59 54 30 16 52 20 23 62 43 32  7
 47 40 28 11 17 36  1 58 60 49 29 25  6]
```




```
In [51]: startTime = time.time()
         n_estimators_range = [16,32,64,128]
         max_depth_range = [2,4,8,16,32,64,128,256]
         param_grid = dict(n_estimators=n_estimators_range, max_depth = max_depth_range)


         params = {'max_features' :'sqrt', 'random_state' : 32, 'min_samples_split' : 2, 'class_
         clf = RandomForestClassifier(**params)

         grid = GridSearchCV(clf, scoring='accuracy', param_grid=param_grid)
         grid.fit(X_train, y_train)
         print(f"Determination of optimal hyperparameters in {time.time() - startTime:0.1f} s")
         print(f"Optimal values are {grid.best_params_} \nAccuracy Score of cross valdation {100

         # Learning on full training set with optimals hyperparameters and score on test set
         params = {'max_features' :'sqrt', 'random_state' : 32,
                   'min_samples_split' : 2, 'class_weight' : 'balanced',
                   'n_estimators' : grid.best_params_['n_estimators'],
                   'max_depth' : grid.best_params_['max_depth']}
```

```
        clf = RandomForestClassifier(**params).fit(X_train, y_train)
        y = clf.predict(X_test)
        accruacy = clf.score(X_test, y_test)
        print(accruacy)
        #rmse = np.sqrt(mean_squared_error(y_test, y))
        #print "RMSE: {:0.4f}".format(rmse)
        #r2 = r2_score(y_test, y)
        #print "R2: {:0.2f}%".format(r2*100)
        print(f"... done in {time.time() - startTime:0.1f}")
```

```
Determination of optimal hyperparameters in 70.6 s
Optimal values are {'max_depth': 16, 'n_estimators': 128}
Accuracy Score of cross valdation 69.68%
0.694554455446
... done in 72.5
```

```
In [52]: clf.score(X_test, y_test, sample_weight=None )
```

```
Out[52]: 0.69455445544554451
```

1.2.4 d) Multi class regression

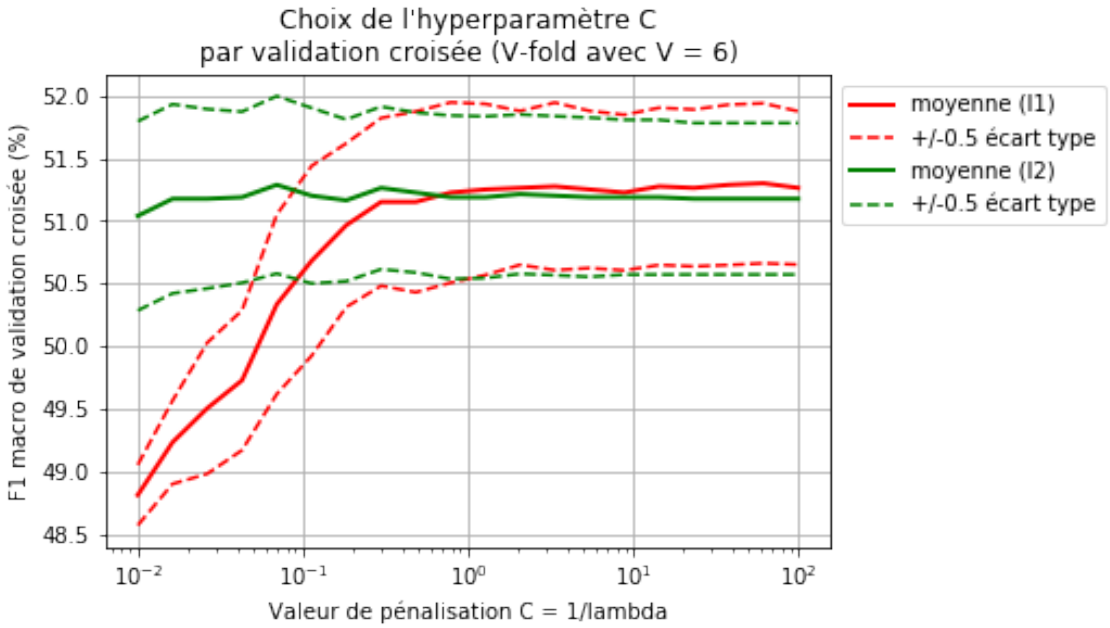
```
In [53]: y = df["HEUREUX_MCLF"]

        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4

        scaler = StandardScaler().fit(X_train)
        X_train = scaler.transform(X_train)
        X_test = scaler.transform(X_test)

        print(f"Number exemple: \n- training set: {y_train.shape[0]}\n- test set: {y_test.shape
        print(f"Number of features: p={X_train.shape[1]}")
```

```
Number exemple:
- training set: 8076
- test set: 2020
Number of features: p=63
```

```
In [54]: nb_value = 20 # Nombre de valeurs testées pour l'hyperparamètre
        mean_score_l1 = np.zeros(nb_value)
        mean_score_l2 = np.zeros(nb_value)
        C_log = np.logspace(-2,2,nb_value)
        cv = 6 # V-fold, nombre de fold

        mean_score_l1 = np.empty(nb_value)
        std_scores_l1 = np.empty(nb_value)
```

```
mean_score_l2 = np.empty(nb_value)
std_scores_l2 = np.empty(nb_value)

np.random.seed(seed=42)

startTime = time.time()

for i, C in enumerate(C_log):
    clf = LogisticRegression(C=C, penalty='l1', tol=0.01, random_state=42, class_weight
    mean_score_l1[i] = 100*np.mean(1-cross_val_score(clf,
                                                     X_train,
                                                     y_train,
                                                     cv=cv,
                                                     scoring='f1_micro'))
    std_scores_l1[i] = 100*np.std(1-cross_val_score(clf,
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='f1_micro'))


for i, C in enumerate(C_log):
    clf = LogisticRegression(C=C, penalty='l2', tol=0.01, random_state=42, class_weight
    mean_score_l2[i] = 100*np.mean(1-cross_val_score(clf,
                                                     X_train,
                                                     y_train,
                                                     cv=cv,
                                                     scoring='f1_micro'))
    std_scores_l2[i] = 100*np.std(1-cross_val_score(clf,
                                                    X_train,
                                                    y_train,
                                                    cv=cv,
                                                    scoring='f1_micro'))

plt.figure()
plt.semilogx(C_log,mean_score_l1[:],'r',linewidth=2,label='moyenne (l1)')
plt.semilogx(C_log,mean_score_l1[:]-0.5*std_scores_l1[:],
             'r--', label=u'+/-0.5 écart type')
plt.semilogx(C_log,mean_score_l1[:]+0.5*std_scores_l1[:],'r--')

plt.semilogx(C_log,mean_score_l2[:],'g',linewidth=2,label='moyenne (l2)')
plt.semilogx(C_log,mean_score_l2[:]-0.5*std_scores_l2[:], 'g--',
             label=u'+/-0.5 écart type')
plt.semilogx(C_log,mean_score_l2[:]+0.5*std_scores_l2[:],'g--')

plt.xlabel("Valeur de pénalisation C = 1/lambda")
plt.ylabel("F1 macro de validation croisée (%)")
plt.title(u"Choix de l'hyperparamètre C\npar validation croisée \
```

```
(V-fold avec V = %s)" % (cv))
plt.legend(bbox_to_anchor=(1, 1))
plt.grid()
plt.show()
print("Détermination des paramètres optimaux en \
{time.time() - startTime:0.1f} s")
print(f"Pénalisation l1, valeur optimale : \
C = {C_log[np.argmax(mean_score_l1)]:0.4f}")
print(f"Pénalisation l2, valeur optimale : \
C = {C_log[np.argmax(mean_score_l2)]:0.4f}")
```



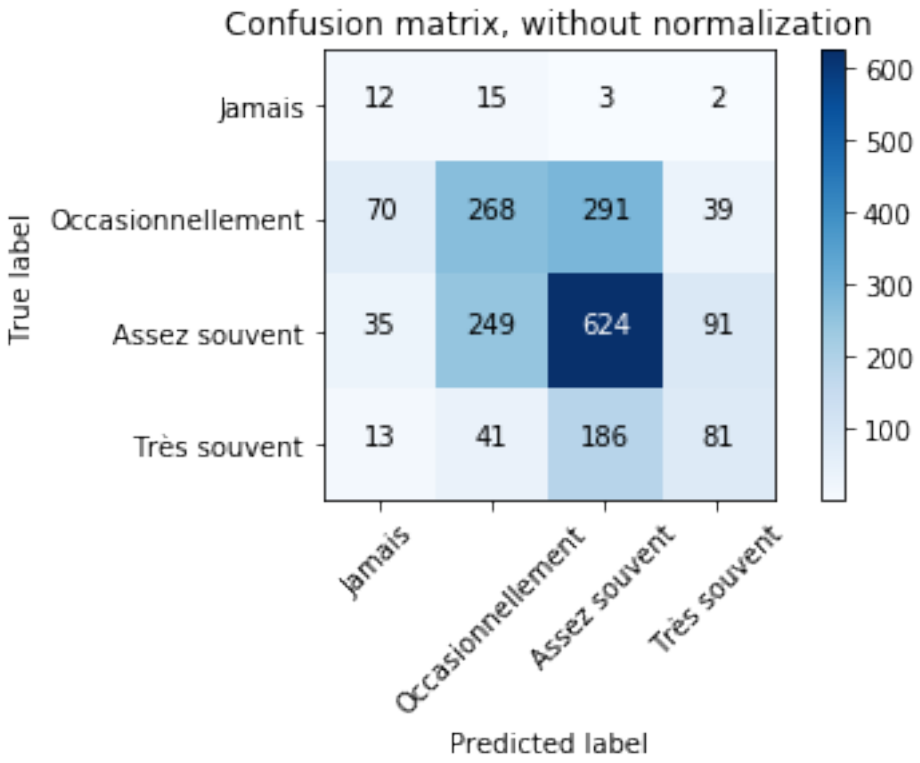

```
Détermination des paramètres optimaux en {time.time() - startTime:0.1f} s
Pénalisation l1, valeur optimale : C = 61.5848
Pénalisation l2, valeur optimale : C = 0.0695
```

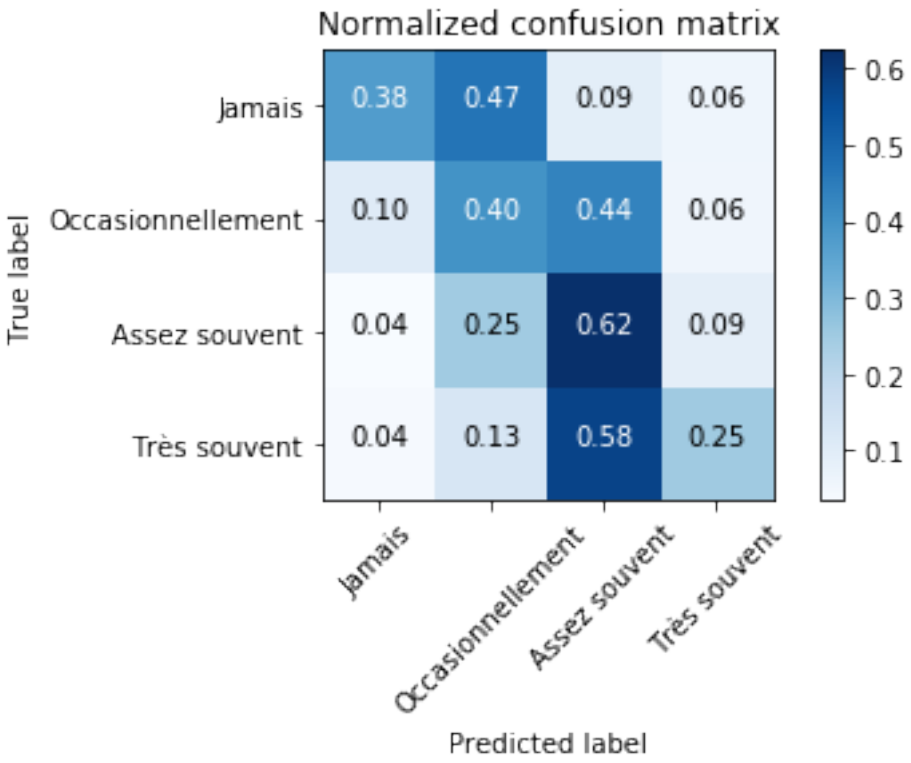
In [55]:
```
# Learning on full training set with optimals hyperparameters and score on test set
clf = LogisticRegression(C=C_log[np.argmax(mean_score_l2)],
                         penalty='l2',
                         tol=0.01,
                         random_state=42,
                         class_weight='balanced')
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

```
        # Modelisation as a 5 multi class classification problem

        class_names = ["Jamais",
                       "Occasionnellement",
                       "Assez souvent",
                       "Très souvent" ]

In [56]: def plot_confusion_matrix(cm, classes,
                                   normalize=False,
                                   title='Confusion matrix',
                                   cmap=plt.cm.Blues):
            """
            This function prints and plots the confusion matrix.
            Normalization can be applied by setting `normalize=True`.
            """
            if normalize:
                cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
                print("Normalized confusion matrix")
            else:
                print('Confusion matrix, without normalization')

            print(cm)

            plt.imshow(cm, interpolation='nearest', cmap=cmap)
            plt.title(title)
            plt.colorbar()
            tick_marks = np.arange(len(classes))
            plt.xticks(tick_marks, classes, rotation=45)
            plt.yticks(tick_marks, classes)

            fmt = '.2f' if normalize else 'd'
            thresh = cm.max() / 2.
            for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
                plt.text(j, i, format(cm[i, j], fmt),
                         horizontalalignment="center",
                         color="white" if cm[i, j] > thresh else "black")

            plt.tight_layout()
            plt.ylabel('True label')
            plt.xlabel('Predicted label')

        # Compute confusion matrix
        cnf_matrix = confusion_matrix(y_test, y_pred)
        np.set_printoptions(precision=2)

        # Plot non-normalized confusion matrix
        plt.figure()
        plot_confusion_matrix(cnf_matrix, classes=class_names,
```

```
                          title='Confusion matrix, without normalization')

    # Plot normalized confusion matrix
    plt.figure()
    plot_confusion_matrix(cnf_matrix, classes=class_names, normalize=True,
                          title='Normalized confusion matrix')

    plt.show()
```

```
Confusion matrix, without normalization
[[ 12  15   3   2]
 [ 70 268 291  39]
 [ 35 249 624  91]
 [ 13  41 186  81]]
Normalized confusion matrix
[[ 0.38  0.47  0.09  0.06]
 [ 0.1   0.4   0.44  0.06]
 [ 0.04  0.25  0.62  0.09]
 [ 0.04  0.13  0.58  0.25]]
```






```
In [57]: y_test.value_counts()

Out[57]: 3    999
         2    668
         4    321
         1     32
         Name: HEUREUX_MCLF, dtype: int64
```